SAMRH71

Rad-Hard 32-bit Arm® Cortex®-M7 Microcontroller for Aerospace Applications

Introduction

The SAMRH71 is a high-performance microcontroller based on the 32-bit Arm Cortex-M7 RISC (5.04 CoreMark/MHz) processor with Floating Point Unit (FPU). The device operates at a maximum speed of 100 MHz and features 128 Kbytes of Flash, 16 Kbytes of dual-cache memory, more than 1 Mbyte of embedded SRAM (384 Kbytes of TCM RAM and 768 Kbytes of multiport SRAM), external memory interfaces, such as EEPROM, Flash, SRAM, SDRAM (with embedded Error Correction Code) and QSPI. It is available in a 256-pin package.

The SAMRH71 is ideal for many aerospace applications by integrating deterministic features (timers), analog controls (PWM), data analysis features, data storage features (Embedded SRAM memory and external memories) and various aerospace reception/transmission links to vehicle data, such as SpaceWire and 1553.

The device peripheral set includes Ethernet 10/100, two CAN-FD, up to 10 FLEXCOMs (USART/UART/SPI/I^2C), SpaceWire and Mil Std 1553 links, as well as high-performance processors SHA and TRNG.

The SAMRH71 devices are high-performance Space microcontrollers that target critical missions for on-board computing, sensor management application and connectivity applications. They embed a concurrent AMBA system bus with advanced DMA features to perform several tasks simultaneously, without excessive processor workload. The SAMRH71 devices are implemented using the ATMX150RHA process that has been specially designed for aerospace applications by implementing on-chip transient and permanent error detection and correction.

Features

- Core Arm Cortex-M7 running at up to 100 MHz
  - 16 Kbytes of I-Cache and 16 Kbytes of D-Cache with Error Code Correction (ECC)
  - Simple-precision and double-precision HW Floating Point Unit (FPU)
  - Memory Protection Unit (MPU) with 16 zones
  - DSP Instructions, Thumb®-2 Instruction Set
  - Embedded Trace Module (ETM) with instruction trace stream, including Trace Port Interface Unit (TPIU)
- Memories
  - 128 Kbytes embedded Flash with built-in ECC (up to 2 errors correction)
  - 384 Kbytes embedded SRAM for Tightly Coupled Memory (TCM) interface, running at the same frequency as the Cortex-M7 with built-in ECC (up to 1 error correction)
  - 768 Kbytes embedded Multiport SRAM with built-in ECC (up to 1 error correction) connected to the AHB system, running at the same frequency as system clock
  - Hardened External Memory Controller (HEMC) to address PROM, SRAM and SDRAM with variable data size (from 8 bits to 48 bits)
    - Six independent Chip Selects
    - Up to 2 Gbytes of external memory accessible
    - Built-in ECC, allowing the correction of up to 2 bits per 32 bits
- System Peripherals
  - Built-In Power Fail Detection (PFD), Programmable Supply Monitors and Independent Watchdog for safe operation
  - Non-Maskable Interrupt Controller (NMIC)

  - Crystal or ceramic resonator oscillators: 3 MHz to 20 MHz main oscillator with failure detection
  - RTC with Gregorian calendar and UTC mode, waveform generation in low-power modes
  - RTC counter calibration circuitry compensates for 32.768 kHz crystal frequency variations
  - 32-bit, low-power Real-time Timer (RTT)
  - 4/8/10/12 MHz internal RC oscillator with 4 MHz default frequency for device start-up. In-application trimming access for frequency adjustment
  - 32.768 kHz crystal oscillator or embedded 32 kHz RC oscillator as source of Low-Power mode device clock (SCLK)
  - One PLL for system clock, and one PLL for peripherals
  - One dual-port 32-channel central DMA Controller (XDMAC)
  - Four 3-channel 32-bit Timer Counters (TC) with Capture, Waveform, Compare and PWM modes. Quadrature decoder logic and 2-bit Gray Up/Down Counter for stepper motor
  - Two 4-channel 16-bit PWMs with complementary outputs, Dead Time Generator and several fault inputs per PWM for motor control, two external triggers to manage Power Factor Correction (PFC), DC-DC and lighting control
- Communication Peripherals
  - One 10/100 Ethernet MAC (GMAC)
    - Energy efficiency support (IEEE® 802.3az standard)
    - Ethernet AVB support with IEEE 802.1AS time stamping and IEEE 802.1Qav credit-based traffic-shaping hardware support
    - IEEE 1588 Precision Time Protocol (PTP)
  - Ten FLEXCOMs, each supporting USART/UART, SPI and TWI/I^2C
  - Single data rate transfer Quad I/O Serial Peripheral Interface (QSPI)
  - CAN Controller compliant with CAN Protocol Version 2.0 Part A, B and ISO 11898-1:2015 and with Bosch CANFD Specification
  - SpaceWire interface with two SpaceWire ports and an Embedded SpaceWire router. Each link has a 200 Mbits/sec bit-rate capability. Integrated RMAP support.
  - One 1553 interface with redundant links compliant to MIL-STD-1553B standard. Bus Controller and Remote Terminal configurations available.
- Safety
  - Integrity Check Monitoring (ICM) on memories using Secure Hash Algorithm (SHA)
  - FlexProtect Matrix enabling user/privilege operations on peripherals
  - Watchdog running on independent uninterruptible monolithic clock
  - Clock and power failure detection events routed to normal interrupt or to the Non-Maskable Interrupt (NMI) controller:
    - CPU Frequency Monitor
    - Main Crystal Oscillator Failure Detection
    - 32.768 kHz Crystal Oscillator Frequency Monitor
    - Core Voltage Supply Monitor with internal Power Fail Detection
  - Write protection registers on selected peripherals
- Security
  - True Random Number Generator (TRNG)
  - Secure Hash Algorithm supporting SHA1, SHA224, SHA256, SHA384 and SHA512
- I/O
  - Up to 194 I/O lines with external interrupt capability (edge-sensitivity or level-sensitivity)
  - Up to seven parallel Input/Output Controllers (PIO)
- Operating Conditions
  - Supply voltage: 3.0V to 3.6V for VDDIO, 1.65V to 1.95V for VDDCORE, VDDPLLA and VDDPLLB, LVDS references: 1.25V
  - Ambient temperature: -55°C to +125°C
- Fault Tolerance by Design

  - SEU detection through ECC protection (up to 2 corrections)
  - Radiation Performance - TBC
    - For detailed radiation results, contact your local Microchip Technology sales representative
    - Total Ionizing Dose under evaluation
    - No Single Event Latch-Up below a LET threshold of 62 MeV.cm²/mg @ 125°C (TBC)
- Packages
  - CQFP256, 256-lead, 18g

Table of Contents

1. Configuration Summary

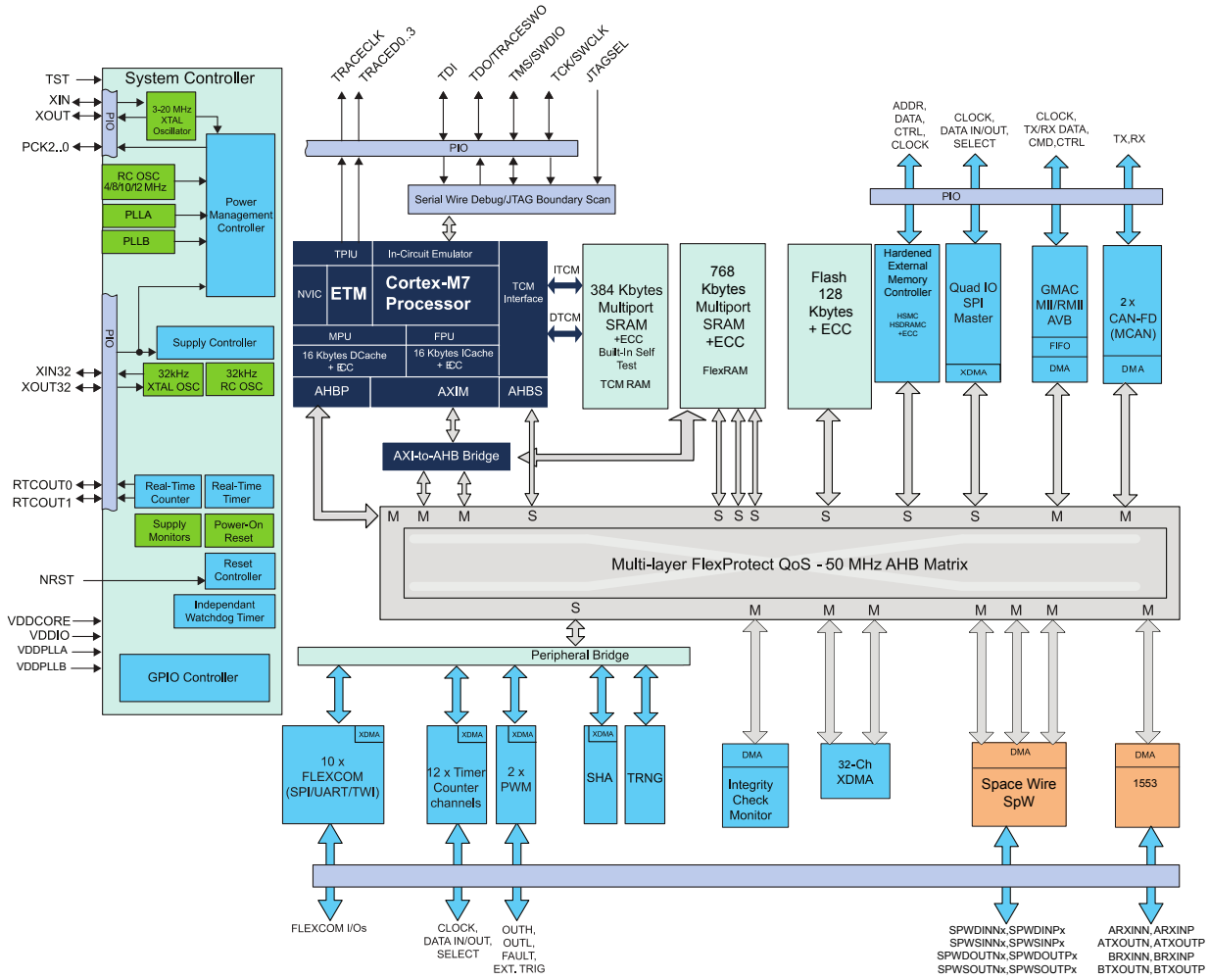
Table 1-1. Configuration Summary

Feature	SAMRH71
Flash	128 Kbytes
Flash Page Size	256 bytes
Flash Pages	512
Flash Lock Region Size	256 bytes
Flash Lock Bits	32
TCM RAM	384 Kbytes
FlexRAM	768 Kbytes
I-Cache/D-Cache	16 Kbytes/16 Kbytes
Package	QFP256
Number of PIOs	194
Hardened External Memory Controller (HEMC)	8-bit, 16-bit, 32-bit, 40-bit, 48-bit data Up to 6 Chip Selects
HSDRAM Interface	Yes
Central DMA Channels	32
32-bit Timer Counter Channels	12 (Four blocks of 3 channels each)
Timer Counter Channels I/O	12 x 3
16-bit Pulse Width Modulator Channels with Complementary Outputs	2 x 4
Serial Communication Interface (FLEXCOM) Instances	10
Quad I/O Serial Peripheral	1
CAN with Flexible Data-Rate	2
Ethernet MAC	1 x MII, RMII
Embedded Trace Macrocell (ETM)	Yes
IP1553	BC/RT
SpaceWire	2 channels

2. Block Diagram

The figure below illustrates the block diagram of the SAMRH71.

Figure 2-1. SAMRH71 Block Diagram



3. Signal Description

Table 3-1. Signal Description List

Signal Name	Function	Type	Active Level	Voltage Reference	Comments
Clocks, Oscillators and PLLs					
XIN	Main Oscillator Input	Input	–	VDDCORE & VDDIO	–
XOUT	Main Oscillator Output	Output	–		
XIN32	Slow Clock Oscillator Input	Input	–	VDDPLLA & VDDIO	–
XOUT32	Slow Clock Oscillator Output	Output	–		
PCK0 – PCK2	Programmable Clock Output	Output	–	VDDIO	–
Real Time Clock					
RTCOUT0	Programmable RTC waveform output	Output	–	VDDIO	–
RTCOUT1	Programmable RTC waveform output	Output	–		
Watchdog					
NWDT0	Watchdog Output 0	Output	Low	VDDIO	–
NWDT1	Watchdog Output 1	Output	Low		–
NMI					
NMIC–NMI	Non maskable input	Input	High	VDDIO	–
Serial Wire/JTAG Debug Port–SWJ-DP					
TCK_SWCLK	Test Clock (boundary scan mode only) / Serial Wire Clock	Input	–	VDDIO	–
TMS_SWDIO	Test Mode Select (boundary scan mode only) / Serial Wire Input/Output	Input / I/O	–		–
TDO_TRACESWO	Test Data Out (boundary scan mode only)/Trace Asynchronous Data Out	Output	–		–
TDI	Test Data In (boundary scan mode only)	Input	–	VDDIO	–
Trace Debug Port–SWJ-DP					
TRACECLK	Trace Clock	Output	–	VDDIO	–
TRACED0–3	Trace Data	Output	–		–
Reset/Test					
NRST	Synchronous Microcontroller Reset	Input	Low	VDDIO	–
TST	Test Select	Input	–		–
JTAGSEL	JTAG Selection	Input	High		–
PIO Controller					

.....continued

Signal Name	Function	Type	Active Level	Voltage Reference	Comments
PA0–PA13, PA16–PA17, PA19–28	Parallel I/O Controller A	I/O	–	VDDIO	–
PB0–PB29	Parallel I/O Controller B	I/O	–		–
PC0–PC31	Parallel I/O Controller C	I/O	–		–
PD0–PD31	Parallel I/O Controller D	I/O	–		–
PE0–PE12	Parallel I/O Controller E	I/O	–		–
PF0–PF30	Parallel I/O Controller F	I/O	–		–
PG0–PG27, PG30–PG31	Parallel I/O Controller G	I/O	–		–
Configuration					
CFG0–3	Configuration Input pins select HEMC PROM bus size activate or not HECC (8 bits, 12 bits) on HECC PROM	Input	–	VDDIO	- CFG[3] selects the HECC code corrector applied for all NCSx: '0': Hamming(32,7), '1 :'BCH(32,12) - CFG[2] selects the HECC activation/deactivation for all NCSx: '0': HECC OFF, '1 :HECC ON - CFG[1:0] selects the bus width for NCS0: 0b'00: 8 bits, 0b'01: 16 bits, 0b'10: 32 bits, 0b'11: reserved
BOOT_MODE	Boot mode selection	Input	–		0b0: Internal Flash 0b1: External Memory Controller CS0
Hardened External Memory Controller–HEMC					
D0–D31	Data Bus	I/O	–	VDDIO	–
CB0–CB15	Checkbit bus	I/O	–		–
A0–A27	Address Bus	Output	–		–
NWAIT	External Wait Signal	Input	Low		–
NCS0–NCS5	Common Chip Select Lines	Output	Low		–
Hardened Static Memory Controller–HSMC					
NRD	Read Signal	Output	Low	VDDIO	–
NWE	Write Enable	Output	Low		–
NWR0–NWR4	Write Byte Enable Signal	Output	Low		–
EXT_CTRL_BUFFER	Extern control buffer	Output	Low		–
Hardened SDR-SDRAM Controller–HSDRAMC					

.....continued

Signal Name	Function	Type	Active Level	Voltage Reference	Comments
SDCK	SDRAM Clock	Output	–	VDDIO	–
SDCKE	SDRAM Clock Enable	Output	–		–
BA0–BA1	Bank Select	Output	–		–
SDWE	SDRAM Write Enable	Output	–		–
RAS–CAS	Row and Column Signal	Output	–		–
SDA10	SDRAM Address 10 Line	Output	–		–
SDNBS0–SDNBS4	Byte Mask Signal	Output	Low		–
Flexcom–FLEXCOMx (x=[0:9])					
FLEXCOMx_IO0	FLEXCOMx Transmit Data	I/O	–	VDDIO	FLEXCOM0–9
FLEXCOMx_IO1	FLEXCOMx Receive Data	I/O	–		FLEXCOM0–9
FLEXCOMx_IO2	FLEXCOMx Serial Clock	I/O	–		FLEXCOM0–5
FLEXCOMx_IO3	FLEXCOMx Clear To Send / Peripheral Chip select	I/O	–		
FLEXCOMx_IO4	FLEXCOMx Request To Send / Peripheral Chip Select	Output	–		
FLEXCOMx_IO5	FLEXCOMx Peripheral Chip Select	Output	–		FLEXCOM0 FLEXCOM3
FLEXCOMx_IO6	FLEXCOMx Peripheral Chip Select	Output	–		FLEXCOM4
FLEXCOMx_IO7	LON Collision Detection	Input	–		Only on FLEXCOM 0
Timer/Counter–TCx (x=[0:11])					
TCLK0..11	TC Channel x External Clock Input	Input	–	VDDIO	–
TIOA0..11	TC Channel x I/O Line A	I/O	–		
TIOB0..11	TC Channel x I/O Line B	I/O	–		
Pulse Width Modulation Controller–PWMx (x=[0:1])					
PWMCx_PWMH0–PWMCx_PWMH3	Waveform Output High	Output	–	VDDIO	–
PWMCx_PWML0–PWMCx_PWML3	Waveform Output Low	Output	–		Only output in complementary mode when dead time insertion is enabled.
PWMCx_PWMFI0–PWMCx_PWMFI2	Fault Input	Input	–		–
PWMCx_PWMEXTRG0–PWMCx_PWMEXTRG1	External Trigger Input	Input	–		–
Quad IO SPI (Master)–QSPI					
QSCK	Quad I/O SPI Serial Clock	Output	–	VDDIO	–
QCS	Quad I/O SPI Chip Select	Output	–		–
QIO0..3	Quad I/O SPI I/O 0 to 3	I/O	–		–
Ethernet MAC 10/100–GMAC					

.....continued

Signal Name	Function	Type	Active Level	Voltage Reference	Comments
GREFCK	Reference Clock	Input	–	VDDDIO	RMII only
GTXCK	Transmit Clock	Input	–		MII only
GRXCK	Receive Clock	Input	–		MII only
GTXEN	Transmit Enable	Output	–		–
GTX0–GTX3	Transmit Data	Output	–		GTX0–GTX1 only in RMII
GTXER	Transmit Coding Error	Output	–		MII only
GRXDV	Receive Data Valid	Input	–		MII only
GRX0–GRX3	Receive Data	Input	–		GRX0–GRX1 only in RMII
GRXER	Receive Error	Input	–		–
GCRS	Carrier Sense	Input	–		MII only
GCOL	Collision Detected	Input	–		MII only
GMDC	Management Data Clock	Output	–		–
GMDIO	Management Data Input/Output	I/O	–		–
GTSUCOMP	TSU Timer Comparison Valid	Output	–		–
Controller Area Network–MCANx (x=[0:1])					
CANRXx	CAN Receive	Input	–	VDDIO	–
CANTXx	CAN Transmit	Output	–		–
IP1553 Interface					
ATXOUTP	1553 link A–transmitter (positive)	Output	–	VDDIO	–
ATXOUTN	1553 link A–transmitter (negative)	Output	–		–
ARXINP	1553 link A–receiver (positive)	Input	–		–
ARXINN	1553 link A–receiver (negative)	Input	–		–
BTXOUTP	1553 link B–transmitter (positive)	Output	–		–
BTXOUTN	1553 link B–transmitter (negative)	Output	–		–
BRXINP	1553 link B–receiver (positive)	Input	–		–
BRXINN	1553 link B–receiver (negative)	Input	–		–
SpaceWire Interface–SPWx (x=[0:1])					
SPWDINPx	SpaceWire data input (positive)	Input	–	–	Differential signal based on level referenced SPWxREF=1.25V x=[0:1]
SPWDINNx	SpaceWire data input (negative)	Input	–	–	
SPWSINPx	SpaceWire strobe input (positive)	Input	–	–	Differential signal based on level referenced SPWxREF=1.25V x=[0:1]
SPWSINNx	SpaceWire strobe input (negative)	Input	–	–	
SPWDOUTPx	SpaceWire data output (positive)	Output	–	–	Differential signal based on level referenced SPWxREF=1.25V x=[0:1]
SPWDOUTNx	SpaceWire data output (negative)	Output	–	–	
SPWSOUTPx	SpaceWire strobe output (positive)	Output	–	–	Differential signal based on level referenced SPWxREF=1.25V x=[0:1]
SPWSOUTNx	SpaceWire strobe output (negative)	Output	–	–	

.....continued

Signal Name	Function	Type	Active Level	Voltage Reference	Comments
SPWREFx	SpaceWire reference voltage	Power	–	–	x=[0:1]

4. Space Quality Grade

The hermetic SAMRH71F20C-7GB is developed and manufactured in compliance with MIL-PRF-38535 and ESCC requirements for aerospace applications.

Screening and qualification flows are described in Aerospace & Defense AEQA0242 specification, which is available for download from the Microchip web site, http://www.microchip.com/.

Table 4-1. Temperature Grade for Space Products

Temperature (°C)	Comments
-55°C to +125°C	TBD

5. Package and Pinout

In the tables that follow, the column “Reset State” indicates the reset state of the line with mnemonics.

- “PIO” “/” signal

Indicates whether the PIO line resets in I/O mode or in peripheral mode. If “PIO” is mentioned, the PIO line is maintained in a static state as soon as the reset is released. As a result, the bit corresponding to the PIO line in the register PIO_PSR (Peripheral Status Register) resets low.

If a signal name is mentioned in the “Reset State” column, the PIO line is assigned to this function and the corresponding bit in the PIO_PSR resets high. This is the case of pins controlling memories, in particular the address lines, which require the pin to be driven as soon as the reset is released.

- “I” / ”O”

Indicates whether the signal is in the input or output state.

- “PU” / “PD”

Indicates whether pull up, pull down, or nothing is enabled.

- “ST”

Indicates if Schmitt Trigger is enabled.

5.1 256-pin CQFP Package

For the 256-pin CQFP package drawing, refer to the section Mechanical Characteristics.

5.2 256-pin Pinout

Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary
1	GNDCORE	34	PA22	67	PB4	100	PB29	133	PC28	166	PD3	199	PG0	232	PG31
2	VDDCORE	35	PA23	68	PB5	101	PC0	134	PC29	167	PD4	200	PG1	233	PF11
3	PF29	36	PA24	69	PB6	102	PC1	135	PC30	168	PD5	201	PG2	234	PF12
4	PF30	37	PA25	70	PB7	103	PC2	136	PC31	169	PD6	202	PG3	235	PF13
5	ATXOUTN	38	PA26	71	PB8	104	PC3	137	PE12	170	PD7	203	PG4	236	PF14
6	ATXOUTP	39	PA27	72	PB9	105	PC4	138	PE11	171	PD8	204	PG5	237	PF15
7	ARXINN	40	PA28	73	PB10	106	PC5	139	PE10	172	PD9	205	PG6	238	PF16
8	ARXINP	41	SPWDINP1	74	PB11	107	PC6	140	PE9	173	PD10	206	PG7	239	PF17
9	PA0	42	SPWDINN1	75	PB12	108	PC7	141	PE8	174	PD11	207	PG8	240	PF18
10	PA1	43	SPWDOUTP1	76	PB13	109	PC8	142	PE7	175	PD12	208	PG9	241	PF19
11	PA2	44	SPWDOUTN1	77	PB14	110	PC9	143	PE6	176	PD13	209	PG10	242	PF20
12	PA3	45	SPWREF1	78	PB15	111	PC10	144	PE5	177	PD14	210	PG11	243	PF21
13	PA4	46	SPWSOUTP1	79	PB16	112	PC11	145	PE4	178	PD15	211	PG12	244	PF22
14	PA5	47	SPWSOUTN1	80	XOUT	113	PC12	146	PE3	179	PD16	212	PG13	245	PF23
15	PA6	48	SPWSINP1	81	XIN	114	PC13	147	PE2	180	PD17	213	PG14	246	PF24
16	PA7	49	SPWSINN1	82	VDD18_PLLA	115	PC14	148	PE1	181	PD18	214	PG15	247	PF25
17	PA8	50	SPWDINP0	83	GND_PLLA	116	PC15	149	PE0	182	PD19	215	PG16	248	BTXOUTN
18	PA9	51	SPWDINN0	84	GND_PLLB	117	PC16	150	PF0	183	PD20	216	PG17	249	BTXOUTP
19	PA10	52	SPWDOUTP0	85	VDD18_PLLB	118	PC17	151	PF1	184	PD21	217	PG18	250	BRXINN
20	PA11	53	SPWDOUTN0	86	PB17	119	PC18	152	PF2	185	PD22	218	PG19	251	BRXINP
21	PA12	54	SPWREF0	87	PB18	120	PC19	153	PF3	186	PD23	219	PG20	252	PF26
22	PA13	55	SPWSOUTP0	88	PB19	121	PC20	154	PF4	187	PD24	220	PG21	253	PF27
23	TST	56	SPWSOUTN0	89	PB20	122	PC21	155	PF5	188	PD25	221	PG22	254	PF28

.....continued															
Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary	Pin	Primary
24	JTAGSEL	57	SPWSINP0	90	PB21	123	PC22	156	PF6	189	PD26	222	PG23	255	VDDCORE
25	NMIC_NMI	58	SPWSINN0	91	PB22	124	PC23	157	PF7	190	PD27	223	PG24	256	GNDCORE
26	PA16	59	PB0	92	PB23	125	PC24	158	PF8	191	VDDCORE	224	VDDIO		
27	PA17	60	PB1	93	PB24	126	PC25	159	PF9	192	GNDCORE	225	GNDIO		
28	NRST	61	PB2	94	PB25	127	VDDCORE	160	VDDIO	193	GNDCORE	226	PG25		
29	PA19	62	PB3	95	PB26	128	GNDCORE	161	GNDIO	194	VDDCORE	227	PG26		
30	PA20	63	VDDCORE	96	VDDIO	129	GNDCORE	162	PF10	195	PD28	228	PG27		
31	PA21	64	GNDCORE	97	GNDIO	130	VDDCORE	163	PD0	196	PD29	229	XOUT32		
32	VDDIO	65	GNDCORE	98	PB27	131	PC26	164	PD1	197	PD30	230	XIN32		
33	GNDIO	66	VDDCORE	99	PB28	132	PC27	165	PD2	198	PD31	231	PG30		

5.3 Signal Multiplexing

Table 5-1. Pins Description

QFP256 Pins	Primary		Alternate		PIO Peripheral				Reset State
	Signal	Dir	Signal	Dir	Func	Signal	Dir	I/O Set	Signal, Dir, PU, PD, HiZ, ST,SEC,FILTER
3	PF29	I/O	–	–	A	FLEXCOM1_IO1	I/O	1	PIO, I, PU, ST
4	PF30	I/O	–	–	A	FLEXCOM1_IO0	I/O	1	PIO, I, PU, ST
9	PA0	I/O	–	–	A	–	–	–	PIO, I, PU, ST
					B	TCLK0	I	1	
					C	PWMC0_PWMH0	O	1	
10	PA1	I/O	–	–	A	–	–	–	PIO, I, PU, ST
					B	TIOA0	I/O	1	
					C	PWMC0_PWMH1	O	1	
11	PA2	I/O	–	–	A	FLEXCOM2_IO0	I/O	1	PIO, I, PU, ST
					B	TIOB0	I/O	1	
					C	PWMC0_PWMH2	O	1	
12	PA3	I/O	–	–	A	FLEXCOM2_IO4	O	1	PIO, I, PU, ST
					B	TCLK1	I	1	
					C	PWMC0_PWMH3	O	1	
13	PA4	I/O	–	–	A	–	–	–	PIO, I, PU, ST
					B	TIOA1	I/O	1	
					C	PWMC0_PWML0	O	1	
14	PA5	I/O	–	–	A	–	–	–	PIO, I, PU, ST
					B	TIOB1	I/O	1	
					C	PWMC0_PWML1	O	1	
15	PA6	I/O	–	–	A	FLEXCOM2_IO1	I/O	1	PIO, I, PU, ST
					B	TCLK2	I	1	
					C	PWMC0_PWML2	O	1	
16	PA7	I/O	–	–	A	FLEXCOM2_IO3	O	1	PIO, I, PU, ST
					B	TIOA2	I/O	1	
					C	PWMC0_PWML3	O	1	
17	PA8	I/O	–	–	A	FLEXCOM2_IO2	I/O	1	PIO, I, PU, ST
					B	TIOB2	I/O	1	
					C	PWMC0_PWMFI0	I	1	
18	PA9	I/O	–	–	A	FLEXCOM5_IO2	I/O	1	PIO, I, PU, ST
					C	PWMC0_PWMFI1	I	1	

.....continued

QFP256 Pins	Primary		Alternate		PIO Peripheral				Reset State
	Signal	Dir	Signal	Dir	Func	Signal	Dir	I/O Set	Signal, Dir, PU, PD, HiZ, ST,SEC,FILTER
19	PA10	I/O	–	–	A	FLEXCOM5_IO3	O	1	PIO, I, PU, ST
					B	PCK2	O	1	
					C	PWMC0_PWMEXTRG0	I	1	
20	PA11	I/O	–	–	A	FLEXCOM5_IO4	O	1	PIO, I, PU, ST
					B	PCK0	O	1	
					C	PWMC0_PWMEXTRG1	I	1	
21	PA12	I/O	–	–	A	FLEXCOM3_IO6		1	PIO, I, PU, ST
					C	PWMC0_PWMFI2	I	1	
22	PA13	I/O	–	–	A	NWDT0	–	1	NWDT0
26	PA16	I/O	–	–	A	–	–	–	TCK_SWCLK, ST
					C	TCK_SWCLK	–	1	
27	PA17	I/O	–	–	A	–	–	–	TMS_SWDIO, ST
					C	TMS_SWDIO	–	1	
29	PA19	I/O	–	–	A	FLEXCOM3_IO1	I/O	1	PIO, I, PU, ST
					C	QIO3	I/O	1	
30	PA20	I/O	–	–	A	FLEXCOM3_IO0	I/O	1	PIO, I, PU, ST
					C	QIO2	I/O	1	
31	PA21	I/O	–	–	A	FLEXCOM3_IO4	O	1	PIO, I, PU, ST
					C	QSCK	O	1	
34	PA22	I/O	–	–	A	FLEXCOM3_IO5	I	1	PIO, I, PU, ST
					C	QCS	O	1	
35	PA23	I/O	–	–	A	FLEXCOM3_IO2	I/O	1	PIO, I, PU, ST
					C	QIO1	I/O	1	
36	PA24	I/O	–	–	A	FLEXCOM3_IO3	O	1	PIO, I, PU, ST
					C	QIO0	I/O	1	
37	PA25	I/O	–	–	A	FLEXCOM9_IO0	I/O	1	PIO, I, PU, ST
					C	TRACED0	O	1	
38	PA26	I/O	–	–	A	FLEXCOM9_IO1	I/O	1	PIO, I, PU, ST
					C	TRACED1	O	1	
39	PA27	I/O	–	–	A	FLEXCOM8_IO0	I/O	1	PIO, I, PU, ST
					C	TRACED2	O	1	
40	PA28	I/O	–	–	A	FLEXCOM8_IO1	I/O	1	PIO, I, PU, ST
					C	TRACED3	O	1	
59	PB0	I/O	–	–	A	–	–	–	PIO, I, PU, ST
					C	TRACECTL	O	1	
60	PB1	I/O	–	–	A	–	–	–	TDI, PU, ST
					C	TDI	I	1	
61	PB2	I/O	–	–	A	–	–	–	PIO, I, PU, ST
					C	TRACECLK	O	1	
62	PB3	I/O	–	–	A	–	–	–	TDO_TRACESWO, PU
					C	TDO_TRACESWO	–	1	
67	PB4	I/O	–	–	A	FLEXCOM7_IO0	I/O	1	PIO, I, PU, ST
					C	CANTX1	O	2	
68	PB5	I/O	–	–	A	FLEXCOM7_IO1	I/O	1	PIO, I, PU, ST
					C	CANRX1	I	2	

.....continued

QFP256 Pins	Primary		Alternate		PIO Peripheral				Reset State
	Signal	Dir	Signal	Dir	Func	Signal	Dir	I/O Set	Signal, Dir, PU, PD, HiZ, ST,SEC,FILTER
69	PB6	I/O	–	–	A	FLEXCOM6_IO0	I/O	1	PIO, I, PU, ST
					C	CANTX0	O	2	
70	PB7	I/O	–	–	A	FLEXCOM6_IO1	I/O	1	PIO, I, PU, ST
					C	CANRX0	I	2	
71	PB8	I/O	–	–	A	TCLK3	I	1	PIO, I, PU, ST
72	PB9	I/O	–	–	A	TIOA3	I/O	1	PIO, I, PU, ST
73	PB10	I/O	–	–	A	TIOB3	I/O	1	PIO, I, PU, ST
74	PB11	I/O	–	–	A	TCLK4	I	1	PIO, I, PU, ST
75	PB12	I/O	–	–	A	TIOA4	I/O	1	PIO, I, PU, ST
76	PB13	I/O	–	–	A	TIOB4	I/O	1	PIO, I, PU, ST
77	PB14	I/O	–	–	A	TCLK5	I	1	PIO, I, PU, ST
78	PB15	I/O	–	–	A	TIOA5	I/O	1	PIO, I, PU, ST
79	PB16	I/O	–	–	A	TIOB5	I/O	1	NWDT1
					B	NWDT1	–	1	
86	PB17	I/O	–	–	A	PWMC1_PWMEXTRG0	I	1	PIO, I, PU, ST
87	PB18	I/O	–	–	A	PWMC1_PWMEXTRG1	I	1	PIO, I, PU, ST
88	PB19	I/O	–	–	A	PWMC1_PWMH0	O	1	PIO, I, PU, ST
89	PB20	I/O	–	–	A	PWMC1_PWMH1	O	1	PIO, I, PU, ST
90	PB21	I/O	–	–	A	PWMC1_PWMH2	O	1	PIO, I, PU, ST
91	PB22	I/O	–	–	A	PWMC1_PWMH3	O	1	PIO, I, PU, ST
92	PB23	I/O	–	–	A	PWMC1_PWML0	O	1	PIO, I, PU, ST
93	PB24	I/O	–	–	A	PWMC1_PWML1	O	1	PIO, I, PU, ST
94	PB25	I/O	–	–	A	PWMC1_PWML2	O	1	PIO, I, PU, ST
95	PB26	I/O	–	–	A	PWMC1_PWML3	O	1	PIO, I, PU, ST
98	PB27	I/O	–	–	A	PWMC1_PWMFI0	I	1	PIO, I, PU, ST
99	PB28	I/O	–	–	A	PWMC1_PWMFI1	I	1	PIO, I, PU, ST
100	PB29	I/O	–	–	A	PWMC1_PWMFI2	I	1	PIO, I, PU, ST
101	PC0	I/O	–	–	A	FLEXCOM4_IO0	I/O	1	PIO, I, PU, ST
102	PC1	I/O	–	–	A	FLEXCOM4_IO1	I/O	1	PIO, I, PU, ST
103	PC2	I/O	–	–	A	FLEXCOM4_IO2	I/O	1	PIO, I, PU, ST
104	PC3	I/O	–	–	A	FLEXCOM4_IO3	O	1	PIO, I, PU, ST
105	PC4	I/O	–	–	A	FLEXCOM4_IO4	O	1	PIO, I, PU, ST
106	PC5	I/O	–	–	A	FLEXCOM4_IO5	I	1	PIO, I, PU, ST
107	PC6	I/O	–	–	A	FLEXCOM4_IO6	–	1	PIO, I, PU, ST
108	PC7	I/O	–	–	A	PCK0	O	2	PIO, I, PU, ST
109	PC8	I/O	–	–	A	PCK1	O	1	PIO, I, PU, ST
110	PC9	I/O	–	–	A	FLEXCOM5_IO0	I/O	1	PIO, I, PU, ST
111	PC10	I/O	–	–	A	FLEXCOM5_IO1	I/O	1	PIO, I, PU, ST
112	PC11	I/O	–	–	A	TCLK6	I	1	PIO, I, PU, ST
					B	GRX1	I	1	
113	PC12	I/O	–	–	A	TIOA6	I/O	1	PIO, I, PU, ST
					B	GRX0	I	1	
114	PC13	I/O	–	–	A	TIOB6	I/O	1	PIO, I, PU, ST
					B	GTX1	O	1	
115	PC14	I/O	–	–	A	TCLK7	I	1	PIO, I, PU, ST
					B	GTX0	O	1	

.....continued

QFP256 Pins	Primary		Alternate		PIO Peripheral				Reset State
	Signal	Dir	Signal	Dir	Func	Signal	Dir	I/O Set	Signal, Dir, PU, PD, HiZ, ST,SEC,FILTER
116	PC15	I/O	–	–	A	TIOA7	I/O	1	PIO, I, PU, ST
					B	GRXER	I	1	
117	PC16	I/O	–	–	A	TIOB7	I/O	1	PIO, I, PU, ST
					B	GRXDV	I	1	
118	PC17	I/O	–	–	A	TCLK8	I	1	PIO, I, PU, ST
					B	GTXEN	O	1	
119	PC18	I/O	–	–	A	TIOA8	I/O	1	PIO, I, PU, ST
					B	GTXCK	I/O	1	
120	PC19	I/O	–	–	A	TIOB8	I/O	1	PIO, I, PU, ST
					B	GMDIO	I/O	1	
121	PC20	I/O	–	–	A	–	–	–	PIO, I, PU, ST
					B	GMDC	O	1	
122	PC21	I/O	–	–	A	FLEXCOM0_IO0	I/O	1	PIO, I, PU, ST
					D	TCLK9	I	1	
123	PC22	I/O	–	–	A	FLEXCOM0_IO1	I/O	1	PIO, I, PU, ST
					D	TIOA9	I/O	1	
124	PC23	I/O	–	–	A	FLEXCOM0_IO2	I/O	1	PIO, I, PU, ST
					B	GTXER	O	1	
125	PC24	I/O	–	–	A	FLEXCOM0_IO3	O	1	PIO, I, PU, ST
					B	GTX2	O	1	
					D	TIOB9	I/O	1	
126	PC25	I/O	–	–	A	FLEXCOM0_IO4	O	1	PIO, I, PU, ST
					B	GTX3	O	1	
					D	TCLK11	I	1	
131	PC26	I/O	–	–	A	FLEXCOM0_IO5	I	1	PIO, I, PU, ST
					B	GCOL	I	1	
					D	TIOA11	I/O	1	
132	PC27	I/O	–	–	A	FLEXCOM0_IO6	–	1	PIO, I, PU, ST
					B	GCRS	I	1	
					D	TIOB11	I/O	1	
133	PC28	I/O	–	–	A	FLEXCOM0_IO7	–	1	PIO, I, PU, ST
					B	GRXCK	I	1	
					D	TCLK10	I	1	
134	PC29	I/O	–	–	A	CB15	–	1	CB15
					B	GRX3	I	1	
					D	TIOA10	I/O	1	
135	PC30	I/O	–	–	A	CB14	–	1	CB14
					B	GRX2	I	1	
					D	TIOB10	I/O	1	
136	PC31	I/O	–	–	A	CB13	–	1	CB13
					B	GTSUCOMP	O	1	
137	PE12	I/O	–	–	A	CB12	–	1	CB12
					B	PCK1	O	2	
138	PE11	I/O	–	–	A	CB11	–	1	CB11
					C	CANTX1	O	1	

.....continued

QFP256 Pins	Primary		Alternate		PIO Peripheral				Reset State
	Signal	Dir	Signal	Dir	Func	Signal	Dir	I/O Set	Signal, Dir, PU, PD, HiZ, ST,SEC,FILTER
139	PE10	I/O	–	–	A	CB10	–	1	CB10
					C	CANRX1	I	1	
140	PE9	I/O	–	–	A	CB9	–	1	CB9
					C	CANTX0	O	1	
141	PE8	I/O	–	–	A	CB8	–	1	CB8
					C	CANRX0	I	1	
142	PE7	I/O	–	–	A	CB7	–	1	CB7
143	PE6	I/O	–	–	A	CB6	–	1	CB6
144	PE5	I/O	–	–	A	CB5	–	1	CB5
145	PE4	I/O	–	–	A	CB4	–	1	CB4
146	PE3	I/O	–	–	A	CB3	–	1	CB3
147	PE2	I/O	–	–	A	CB2	–	1	CB2
148	PE1	I/O	–	–	A	CB1	–	1	CB1
149	PE0	I/O	–	–	A	CB0	–	1	CB0
150	PF0	I/O	–	–	A	RAS	O	1	PIO, I, PU, ST
151	PF1	I/O	–	–	A	SDA10	O	1	PIO, I, PU, ST
152	PF2	I/O	–	–	A	SDCK	I/O	1	PIO, I, PU, ST
153	PF3	I/O	–	–	A	SDCKE	O	1	PIO, I, PU, ST
154	PF4	I/O	–	–	A	SDNBS0	–	1	PIO, I, PU, ST
155	PF5	I/O	–	–	A	SDNBS1	–	1	PIO, I, PU, ST
156	PF6	I/O	–	–	A	SDNBS2	–	1	PIO, I, PU, ST
157	PF7	I/O	–	–	A	SDNBS3	–	1	PIO, I, PU, ST
158	PF8	I/O	–	–	A	SDNBS4	–	1	PIO, I, PU, ST
159	PF9	I/O	–	–	A	SDWE	O	1	PIO, I, PU, ST
162	PF10	I/O	–	–	A	CAS	O	1	PIO, I, PU, ST
163	PD0	I/O	–	–	A	D0	I/O	1	D0
164	PD1	I/O	–	–	A	D1	I/O	1	D1
165	PD2	I/O	–	–	A	D2	I/O	1	D2
166	PD3	I/O	–	–	A	D3	I/O	1	D3
167	PD4	I/O	–	–	A	D4	I/O	1	D4
168	PD5	I/O	–	–	A	D5	I/O	1	D5
169	PD6	I/O	–	–	A	D6	I/O	1	D6
170	PD7	I/O	–	–	A	D7	I/O	1	D7
171	PD8	I/O	–	–	A	D8	I/O	1	D8
172	PD9	I/O	–	–	A	D9	I/O	1	D9
173	PD10	I/O	–	–	A	D10	I/O	1	D10
174	PD11	I/O	–	–	A	D11	I/O	1	D11
175	PD12	I/O	–	–	A	D12	I/O	1	D12
176	PD13	I/O	–	–	A	D13	I/O	1	D13
177	PD14	I/O	–	–	A	D14	I/O	1	D14
178	PD15	I/O	–	–	A	D15	I/O	1	D15
179	PD16	I/O	–	–	A	D16	I/O	1	D16
180	PD17	I/O	–	–	A	D17	I/O	1	D17
181	PD18	I/O	–	–	A	D18	I/O	1	D18
182	PD19	I/O	–	–	A	D19	I/O	1	D19
183	PD20	I/O	–	–	A	D20	I/O	1	D20

.....continued

QFP256 Pins	Primary		Alternate		PIO Peripheral				Reset State
	Signal	Dir	Signal	Dir	Func	Signal	Dir	I/O Set	Signal, Dir, PU, PD, HiZ, ST,SEC,FILTER
184	PD21	I/O	–	–	A	D21	I/O	1	D21
185	PD22	I/O	–	–	A	D22	I/O	1	D22
186	PD23	I/O	–	–	A	D23	I/O	1	D23
187	PD24	I/O	–	–	A	D24	I/O	1	D24
188	PD25	I/O	–	–	A	D25	I/O	1	D25
189	PD26	I/O	–	–	A	D26	I/O	1	D26
190	PD27	I/O	–	–	A	D27	I/O	1	D27
195	PD28	I/O	–	–	A	D28	I/O	1	D28
196	PD29	I/O	–	–	A	D29	I/O	1	D29
197	PD30	I/O	–	–	A	D30	I/O	1	D30
198	PD31	I/O	–	–	A	D31	I/O	1	D31
199	PG0	I/O	–	–	A	A0	O	1	A0
200	PG1	I/O	–	–	A	A1	O	1	A1
201	PG2	I/O	–	–	A	A2	O	1	A2
202	PG3	I/O	–	–	A	A3	O	1	A3
203	PG4	I/O	–	–	A	A4	O	1	A4
204	PG5	I/O	–	–	A	A5	O	1	A5
205	PG6	I/O	–	–	A	A6	O	1	A6
206	PG7	I/O	–	–	A	A7	O	1	A7
207	PG8	I/O	–	–	A	A8	O	1	A8
208	PG9	I/O	–	–	A	A9	O	1	A9
209	PG10	I/O	–	–	A	A10	O	1	A10
210	PG11	I/O	–	–	A	A11	O	1	A11
211	PG12	I/O	–	–	A	A12	O	1	A12
212	PG13	I/O	–	–	A	A13	O	1	A13
213	PG14	I/O	–	–	A	A14	O	1	A14
214	PG15	I/O	–	–	A	A15/BA0	O	1	A15/BA0
215	PG16	I/O	–	–	A	A16/BA1	O	1	A16/BA1
216	PG17	I/O	–	–	A	A17	O	1	A17
217	PG18	I/O	–	–	A	A18	O	1	A18
218	PG19	I/O	–	–	A	A19	O	1	A19
219	PG20	I/O	–	–	A	A20	O	1	A20
220	PG21	I/O	–	–	A	A21	O	1	A21
221	PG22	I/O	–	–	A	A22	O	1	A22
222	PG23	I/O	–	–	A	A23	O	1	A23
223	PG24	I/O	CFG0	–	A	A24	O	1	PIO,I,ST
226	PG25	I/O	CFG1	–	A	A25	O	1	PIO,I,ST
227	PG26	I/O	CFG2	–	A	A26	O	1	PIO,I,ST
228	PG27	I/O	CFG3	–	A	A27	O	1	PIO,I,ST
231	PG30	I/O	–	–	A	RTCOUT0	O	1	PIO, I, PU, ST
232	PG31	I/O	–	–	A	RTCOUT1	O	1	PIO, I, PU, ST
233	PF11	I/O	–	–	A	NCS0	O	1	NCS0
234	PF12	I/O	–	–	A	NCS1	O	1	PIO, I, PU, ST
235	PF13	I/O	–	–	A	NCS2	O	1	PIO, I, PU, ST
236	PF14	I/O	–	–	A	NCS3	O	1	PIO, I, PU, ST
237	PF15	I/O	–	–	A	NCS4	O	1	PIO, I, PU, ST

.....continued

QFP256 Pins	Primary		Alternate		PIO Peripheral				Reset State
	Signal	Dir	Signal	Dir	Func	Signal	Dir	I/O Set	Signal, Dir, PU, PD, HiZ, ST,SEC,FILTER
238	PF16	I/O	–	–	A	NCS5	O	1	PIO, I, PU, ST
239	PF17	I/O	–	–	A	NRD	O	1	NRD
240	PF18	I/O	–	–	A	NWE	O	1	NWE
241	PF19	I/O	–	–	A	NWR0	O	1	NWR0
242	PF20	I/O	–	–	A	NWR1	O	1	NWR1
243	PF21	I/O	–	–	A	NWR2	O	1	NWR2
244	PF22	I/O	–	–	A	NWR3	O	1	NWR3
245	PF23	I/O	–	–	A	NWR4	O	1	NWR4
246	PF24	I/O	BOOT_MODE	–	A	NWAIT	I	1	NWAIT, PU, ST
247	PF25	I/O	–	–	A	EXT_CTRL_BUFFER	–	1	EXT_CTRL_BUFFER
252	PF26	I/O	–	–	A	FLEXCOM1_IO4	O	1	PIO, I, PU, ST
253	PF27	I/O	–	–	A	FLEXCOM1_IO3	O	1	PIO, I, PU, ST
254	PF28	I/O	–	–	A	FLEXCOM1_IO2	I/O	1	PIO, I, PU, ST

6. Power Supply and Power Control

6.1 Power Supply Inputs

The table below lists the power supply pins available on the SAMRH71.

Table 6-1. Power Supply Inputs

Name	Voltage Range, Nominal	Associated Ground	Cells Powered	Estimated Max Consumption
VDDCORE	1.65 V–1.95 V, 1.8V	GNDCORE	Core Logic, Embedded memories (RAM and Flash), Embedded power switches, peripherals, RC oscillators (32K RC and mainRC), crystal oscillators (Main crystal (logic part)) and POR	TBD
VDDPLLA		GNDPLLA	PLLA Cell and crystal oscillators (32 kHz crystal (logic part))	TBD
VDDPLLB		GNDPLLB	PLLB Cell	TBD
VDDIO	3.00V–3.60V, 3.3V	GNDIO	Peripheral I/O lines, Embedded memories (Flash), oscillator pads and crystal oscillators (32 kHz crystal and Main crystal (peripheral part))	TBD
SPWREF	1.25V	GNDIO	SpaceWire reference level for LVDS signals (SpaceWire0–1)	TBD

6.2 Power-up and Power-down Sequencing

6.2.1 Power-up

Power supplies must be established in the following order:

1. VDDIO
2. VDDCORE (as VDDPLLA/B)

6.2.2 Power-down

Power supplies must be removed in the following order:

1. VDDCORE (as VDDPLLA/B)
2. VDDIO

6.3 Active Mode

Active mode is the nominal mode with the core clock running from the fast RC oscillator, the main crystal oscillator or the PLLA/B. The Power Management Controller can be used to adapt the frequency and to disable the peripheral clocks.

7. Input/Output Lines

7.1 General-Purpose I/O (GPIO) Lines

General-purpose I/O lines are managed by PIO Controllers. All I/Os have several input or output modes, such as pull-up or pull-down, input Schmitt triggers, multi drive (open-drain) or input change interrupt. Programming of these modes is performed independently for each I/O line through the PIO controller user interface. For additional information, refer to “Parallel Input/Output Controller (PIO)”.

The typical pull-up value is 15 kOhms and the typical pull-down value is 10 kOhms.

7.2 System I/O Lines

7.2.1 Serial Wire Debug Port (SW-DP) Pins

The SW-DP pins SWCLK and SWDIO are commonly provided on a standard 20-pin JTAG connector defined by Arm. For more details about voltage reference and reset state, refer to the table Signal Description.

At start-up, debug pins are configured in JTAG mode to allow connection with debugging probe. For more details, refer to section Debug and Test Features.

Debug pins can be used as standard I/Os to provide users more general input/output pins when the debug port is not needed in the end application. Mode selection between JTAG mode (System IO mode) and general IO mode is performed through the PIO Controller.

The JTAGSEL pin is used to select the JTAG boundary scan when asserted at a high level. It integrates a permanent pull-down resistor of about 10 kOhms to GND, so that it can be left unconnected for normal operations.

7.2.2 Embedded Trace Module (ETM) Pins

The Embedded Trace Module (ETM) depends on the Trace Port Interface Unit (TPIU) to export data out of the system.

The TPUI features the following pins:

- TRACECLK is always exported to enable synchronization with the data.
- TRACED0–TRACED3 is the instruction trace stream.

7.3 NRST Pin

The NRST pin is a pure input, with a permanent pull-up resistor of 15 kOhms.

8. Architecture Interconnect

The system architecture is based on the Arm Cortex-M7 processor connected to the main AHB Bus Matrix, the embedded Flash and the multiport SRAM.

8.1 Core, Bus Matrices and Memories Interconnect

The AHBP interface is a single 32-bit wide interface enabling access to the peripherals connected on the main bus matrix. It is used for data access only. Instruction fetches are never performed on the interface. The bus, AHBP or AXIM, accessing the peripheral memory area [0x40000000 to 0x5FFFFFFF] is selected in the AHBP Control register.

The memory blocks used for ITCM and DTCMs are fully accessible for all the DMAs through the AHBS interface.

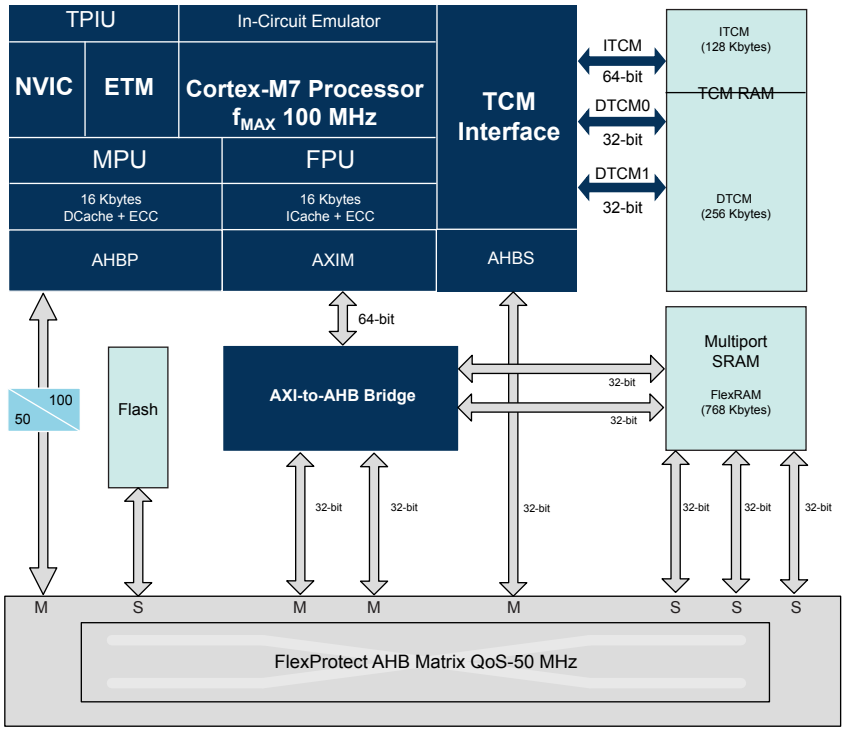
The AXIM interface is a single 64-bit wide interface that connects to the system through the AXI bridge and the AHB bus matrix and allows the following:

- Instruction fetches
- Data-cache linefills and evictions
- Non-cacheable normal-type memory data accesses
- Device and strongly-ordered type data accesses, normally to peripherals

The interconnect of the other masters and slaves is described in Bus Matrix (MATRIX).

The figure below shows the connections of the different Cortex-M7 ports.

Figure 8-1. General Interconnect Block Diagram



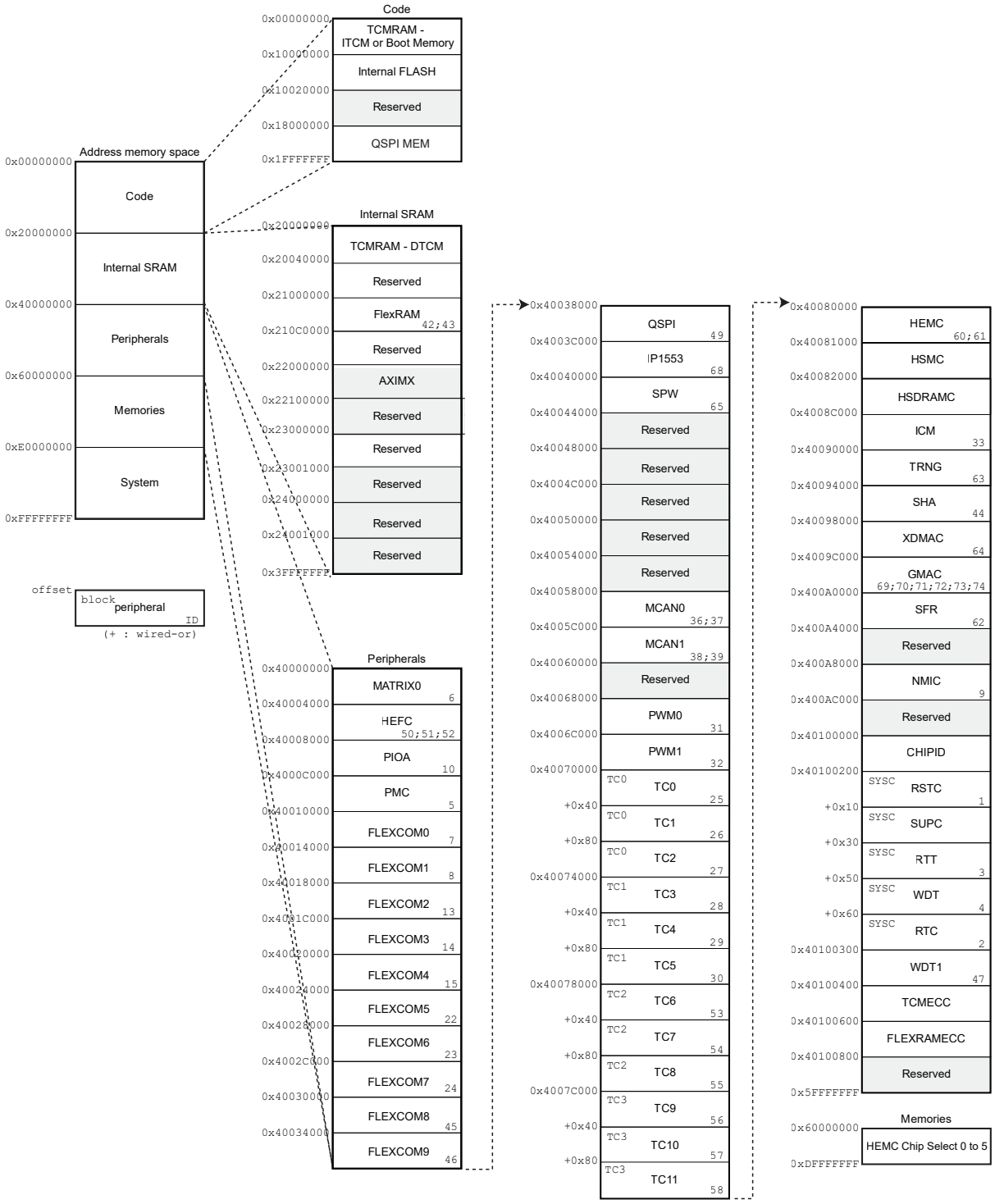
8.2 Memory Protection with FlexProtect

FlexProtect is a protection unit on the AHB matrix that manages the access rights for privilege and user accesses. It prevents any access to the memory or peripheral areas that might result in a configuration change or data corruption/ modification by a DMA.

This feature, in conjunction with the Cortex-M7 MPU, provides full protection of each slave for any master access.

9. Product Mapping

Figure 9-1. SAMRH71 Product Mapping



10. Memories

10.1 External Memories

The SAMRH71 features one Hardened External Memory Controller (HEMC) to provide an interface to a wide range of external memories and to any parallel peripheral. The PROM, SRAM and SDRAM are supported. For additional information, refer to these sections Hardened External Memory Controller (HEMC), Hardened SDRAM Controller (HSDRAMC) and Hardened Static Memory Controller (HSMC).

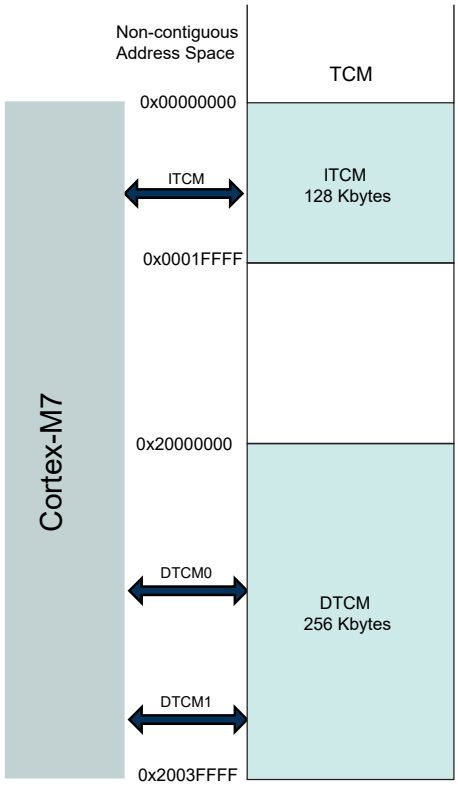
10.2 Embedded Memories

10.2.1 Tightly Coupled Memory (TCM) RAM

The TCM RAM, protected by ECC, optimizes the efficiency of the Cortex-M7 and AHB masters accesses. The TCM RAM is composed of these two blocks:

- One 128-Kbyte block used for ITCM
- One 256-Kbyte block used for DTCM0/DTCM1

Figure 10-1. TCM RAM Mapping



The 384 Kbytes of TCM RAM are accessible in a non-contiguous address space for the Cortex-M7 TCM interfaces:

- 128 Kbytes at address 0 through ITCM interface
- 256 Kbytes at address 0x20000000 through DTCM0 and DTCM1

After reset :

- ITCM is disabled to be make the internal Flash (HEFC) or HEMC accessible at address 0. The Flash or HEMC must embed code to enable the TCM link and to perform a copy from Flash or HEMC to ITCM memory through the AHBS.

- DTCM0/1 are enabled for boot and variables.

To execute the code from ITCM, follow these steps:

1. Start executing the code from Flash or HEMC.
2. Enable the ITCM interface.
3. Perform a copy (through AHBS) of the code to execute from Flash or HEMC to TCM RAM (ITCM).
4. The stack and variables are allocated in the DTCM address space.
5. Jump in the ITCM code.

10.2.2 Internal Multiport SRAM (FlexRAM)

The 768-Kbyte Multiport SRAM, protected by ECC, has several ports connected either to the AHB matrix or directly to the Cortex-M7 to enable simultaneous accesses:

- Three 32-bit ports are connected to the AHB matrix
- Two 32-bit ports are connected to the Cortex-M7 processor

10.2.3 Flash Memories

10.2.3.1 Embedded Flash Overview

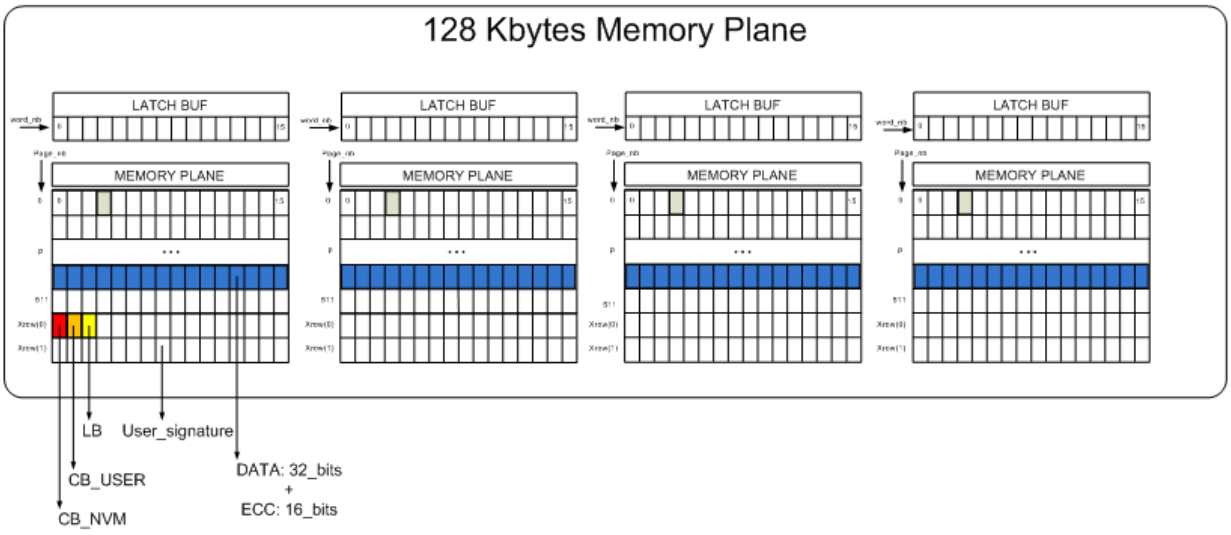
The Flash memory plane is made of 4 x 32 Kbytes planes, placed in parallel, hence the embedded Flash memory has a global size of 128 Kbytes.

The embedded Flash memory is made of 512 pages of 256 bytes 4 x (16 x 4 bytes) of data. Additionally, the Flash memory embeds more space to store HECC data (128 bytes total). The embedded HECC can detect and correct up to 2 bits in each 32-bit data.

Additionally, the embedded Flash memory has two additional rows (xrow0, xrow1), which contains the following:

- Xrow0: 128 user fuse bits, 32 NVM calibration bits and 32 lock bits (the first 32 pages are lockable)
- Xrow1: End user signature: 256 bytes available to the user. The user signature can be used to store information, such as trimming, keys and so on, that the user does not want to be erased by the software ERASE command. Read, write and erase of this area is allowed.

Figure 10-2. Memory Plane Overview



10.2.3.2 Hardened Embedded Flash Controller (HEFC)

The Hardened Embedded Flash Controller (HEFC) manages accesses performed by the system masters. It enables reading the Flash and writing the latch buffer. It also contains a User Interface mapped on the APB. The HEFC ensures the interface of the Flash block. It manages the programming, erasing, locking and unlocking sequences of the Flash using a full set of commands. Refer to the section "Hardened Embedded Flash Controller (HEFC)" for details.

10.2.3.3 General-purpose NVM (GPNVM) Bits

SAMRH71 devices feature four words of general-purpose NVM (GPNVM). The bits can be cleared or set, through the “Clear GPNVM Bit” and “Set GPNVM Bit” commands of the HEFC User Interface.

The GPNVM0[1] bit selects the debug interface. Other bits throughout the GPNVM words are used to store the factory-programmed trim values for the on-board clock sources.

Table 10-1. General-purpose Non volatile Memory Bits

GPNVM Word	Bit	Function
0	1	Debug interface selection 0: JTAG interface 1: SWD interface
	10:8	Factory value for PFD trim
1	1:0	Factory value for Main RC temperature trim.
	3:2	Factory value for Second RC temperature trim.
	6:4	Factory value for Main RC 4 MHz frequency trim.
	9:7	Factory value for Main RC 8 MHz frequency trim.
	12:10	Factory value for Main RC 10 MHz frequency trim.
	15:13	Factory value for Main RC 12 MHz frequency trim.
	18:16	Factory value for Second RC 4 MHz frequency trim.
	21:19	Factory value for Second RC 8 MHz frequency trim.
	24:22	Factory value for Second RC 10 MHz frequency trim.
	27:25	Factory value for Second RC 12 MHz frequency trim.
2 and 3	27:0	Duplicates of GPNVM Word 1

10.2.4 Boot Strategies

The SAMRH71 can boot from several memories:

- Internal Flash memory through HEFC
- External PROM memory through HEMC

Note: The ITCM memory can also be used for boot (by default, the DTCM is enabled and ITCM is disabled at reset), if code has been previously copied into TCM RAM memories.

The selection is performed by reading the value BOOT_MODE PIO (PF24) during reset:

- 0b0: Internal Flash memory (HEFC)
- 0b1: External PROM memory (HEMC)

The selection made by the BOOT_MODE PIO(PF24) mirrors the address of the selected memory to 0x00000000, allowing the processor to boot on this memory. By default, the ITCM is located at address 0x00000000 when enabled.

11. Event System

11.1 Real-time Management

The signals generated by the peripherals are designed to be directly routed to peripherals managing/using these events without processor intervention. Peripherals receiving events contain logic by which to select the one required.

Table 11-1. Real-time Mapping List

Function	Application	Description	Source	Destination
Safety	General-purpose	Automatic switch to reliable main RC oscillator in case of main crystal clock failure	Power Management Controller (PMC)	PMC
	General-purpose	Automatic switch to reliable main RC oscillator in case of main crystal clock failure	Power Management Controller (PMC)	NMI
	General-purpose, motor control, power factor correction (PFC)	Puts the PWM outputs in Safe mode in case of main crystal clock failure	PMC	Pulse Width Modulation 0 and 1 (PWM0 and PWM1)
	Motor control	Puts the PWM outputs in Safe mode (overspeed detection through timer quadrature decoder)	Timer Counter Block 0	PWM0
			Timer Counter Block 1	PWM1
	General-purpose, motor control, power factor correction (PFC)	Puts the PWM outputs in Safe mode (general-purpose fault inputs)	PWMC0_PWMFI0: PIO PA8 PWMC0_PWMFI1: PIO PA9 PWMC0_PWMFI2: PA12	PWM0
		Puts the PWM outputs in Safe mode (general-purpose fault inputs)	PWMC1_PWMFI0: PIO PB27 PWMC1_PWMFI1: PIO PB28 PWMC1_PWMFI2: PIO PB29	PWM1
Measurement trigger	Power factor correction (DC-DC, lighting, etc.) 0	Duty cycle output waveform correction Trigger source selection in PWM	PWMC0_PWMEXTRG0: PIO PA10 PWMC0_PWMEXTRG1: PIO PA11	PWM0
			PWMC1_PWMEXTRG0: PIO PB17 PWMC1_PWMEXTRG1: PIO PB18	PWM1

.....continued

Function	Application	Description	Source	Destination
Delay measurement 2	Motor control 1	Propagation delay of external components (IOs, power transistor bridge driver, and so on.)	PWM0 Comparator Output OC0	TC TIOA0 and TIOB0
			PWM0 Comparator Output OC1	TC TIOA1 and TIOB1
			PWM0 Comparator Output OC2	TC TIOA2 and TIOB2
			PWM1 Comparator Output OC0	TC TIOA3 and TIOB3
			PWM1 Comparator Output OC1	TC TIOA4 and TIOB4
			PWM1 Comparator Output OC2	TC TIOA5 and TIOB5
			PWM0 Comparator Output OC0	TC TIOA6 and TIOB6
			PWM0 Comparator Output OC1	TC TIOA7 and TIOB7
			PWM0 Comparator Output OC2	TC TIOA8 and TIOB8
			PWM1 Comparator Output OC0	TC TIOA9 and TIOB9
			PWM1 Comparator Output OC1	TC TIOA10 and TIOB10
			PWM1 Comparator Output OC	TC TIOA11
Direct Memory Access	General-purpose	Peripheral trigger generation to transfer data to/from system memory	FLEXCOMs, QSPI, TC (Capture), PWM, PIO	XDMAC

12. Peripherals

12.1 Peripheral Identifiers

The table below defines the Peripheral Identifiers of the SAMRH71. A peripheral identifier is required for the control of the peripheral interrupt with the Nested Vectored Interrupt Controller (NVIC) and control of the peripheral clock with the Power Management Controller (PMC).

Table 12-1. Peripheral Identifiers

Instance ID	Instance Name	NVIC Interrupt	PMC Clock Control	Generic Clock	Instance Description
0	SUPC	X	–	–	Supply Controller
1	RSTC	X	–	–	Reset Controller
2	RTC	X	–	–	Real Time Clock
3	RTT	X	–	–	Real Time Timer
4	WDT	X	–	–	Watchdog Timer
5	PMC	X	–	–	Power Management Controller
6	MATRIX0	X	–	–	Matrix 0
7	FLEXCOM0	X	X	X	FLEXCOM 0
8	FLEXCOM1	X	X	X	FLEXCOM 1
9	NMIC	X	–	X	NMI Controller
10	PIOA	X	X	–	Parallel I/O Controller A
11	PIOB	X	–	–	Parallel I/O Controller B
12	PIOC	X	–	–	Parallel I/O Controller C
13	FLEXCOM2	X	X	X	FLEXCOM 2
14	FLEXCOM3	X	X	X	FLEXCOM 3
15	FLEXCOM4	X	X	X	FLEXCOM 4
16	PIOD	X	–	–	Parallel I/O Controller D
17	PIOE	X	–	–	Parallel I/O Controller E
18	ARM	CCW	–	–	Arm Cache ECC Warning
19	ARM	CCF	–	–	Arm Cache ECC Fault
20	ARM	FPU	–	–	Floating Point Unit except IXC
21	ARM	IXC	–	–	Floating Point Unit Interrupt IXC associated with FPU cumulative exception bit
22	FLEXCOM5	X	X	X	FLEXCOM 5
23	FLEXCOM6	X	X	X	FLEXCOM 6
24	FLEXCOM7	X	X	X	FLEXCOM 7
25	TC0	X	X	X	Timer Counter Channel 0
26	TC1	X	X	–	Timer Counter Channel 1

.....continued

Instance ID	Instance Name	NVIC Interrupt	PMC Clock Control	Generic Clock	Instance Description
27	TC2	X	X	–	Timer Counter Channel 2
28	TC3	X	X	X	Timer Counter Channel 3
29	TC4	X	X	–	Timer Counter Channel 4
30	TC5	X	X	–	Timer Counter Channel 5
31	PWM0	X	X	–	Pulse Width Modulation 0
32	PWM1	X	X	–	Pulse Width Modulation 1
33	ICM	X	X	–	Integrity Check Monitor
34	PIOF	X	–	–	Parallel I/O Controller F
35	PIOG	X	–	–	Parallel I/O Controller G
36	MCAN0	INT0	X	X	MCAN Controller 0
37	MCAN0	INT1	–	–	MCAN 0 IRQ 1
38	MCAN1	INT0	X	X	MCAN Controller 1
39	MCAN1	INT1	–	–	MCAN 1 IRQ 1
40	TCMRAM	INTFIX	–	–	TCM RAM–HECC Fixable error detected
41	TCMRAM	INTNOFIX	–	–	TCM RAM–HECC Unfixable error detected
42	FlexRAM	INTFIX	–	–	FlexRAM–HECC Fixable error detected
43	FlexRAM	INTNOFIX	–	–	FlexRAM–HECC Unfixable error detected
44	SHA	X	X	–	Secure Hash Algorithm
45	FLEXCOM8	X	X	X	FLEXCOM 8
46	FLEXCOM9	X	X	X	FLEXCOM 9
47	WDT1	INTWDT1	–	–	Watchdog Timer 1
48	–	–	–	–	Reserved
49	QSPI	X	X	–	Quad I/O Serial Peripheral Interface
50	HEFC	INT0	–	–	HEFC–Clock for writing in Flash can be disabled
51	HEFC	INTFIX	–	–	HEFC–HECC Fixable error detected
52	HEFC	INTNOFIX	–	–	HEFC–HECC Unfixable error detected
53	TC6	X	X	X	Timer Counter Channel 6
54	TC7	X	X	–	Timer Counter Channel 7
55	TC8	X	X	–	Timer Counter Channel 8
56	TC9	X	X	X	Timer Counter Channel 9
57	TC10	X	X	–	Timer Counter Channel 10
58	TC11	X	X	–	Timer Counter Channel 11
59	HEMC	INT	X	–	HEMC
60	HEMC	INTFIX	X	X	HEMC–HECC Fixable error detected/HSDRAMC

.....continued

Instance ID	Instance Name	NVIC Interrupt	PMC Clock Control	Generic Clock	Instance Description
61	HEMC	INTNOFIX	–	–	HEMC–HECC Unfixable error detected
62	SFR	–	–	–	Special Function Register
63	TRNG	X	X	–	True Random Generator
64	XDMAC	X	X	–	Extended DMA Controller
65	SPW	INT0	X	X	SpaceWire
66	SPW	INT1	X	X	SpaceWire
67	–	–	–	–	Reserved
68	IP1553	INT0	X	X	1553
69	GMAC	X	X	–	Ethernet MAC
70	GMAC	Q1	–	–	GMAC Priority Queue 1
71	GMAC	Q2	–	–	GMAC Priority Queue 2
72	GMAC	Q3	–	–	GMAC Priority Queue 3
73	GMAC	Q4	–	–	GMAC Priority Queue 4
74	GMAC	Q5	–	–	GMAC Priority Queue 5
75	–	–	–	–	Reserved
76	–	–	–	–	Reserved
77	–	–	–	–	Reserved
78	–	–	–	–	Reserved
79	–	–	–	–	Reserved
80	–	–	–	–	Reserved
81	–	–	–	–	Reserved
82	–	–	–	–	Reserved
83	–	–	–	–	Reserved
84	–	–	–	–	Reserved
85	–	–	–	–	Reserved
86	–	–	–	–	Reserved
87	–	–	–	–	Reserved
88	–	–	–	–	Reserved
89	–	–	–	–	Reserved
90	–	–	–	–	Reserved

12.2 XDMAC Peripheral Connections

Table 12-2. Peripheral Hardware Requests

Instance Name	Channel T/R	DMA Channel HW Interface Number
FLEXCOM0	Transmit	0
FLEXCOM0	Receive	1
FLEXCOM1	Transmit	2
FLEXCOM1	Receive	3
FLEXCOM2	Transmit	4
FLEXCOM2	Receive	5
FLEXCOM3	Transmit	6
FLEXCOM3	Receive	7
FLEXCOM4	Transmit	8
FLEXCOM4	Receive	9
FLEXCOM5	Transmit	10
FLEXCOM5	Receive	11
FLEXCOM6	Transmit	12
FLEXCOM6	Receive	13
FLEXCOM7	Transmit	14
FLEXCOM7	Receive	15
FLEXCOM8	Transmit	16
FLEXCOM8	Receive	17
FLEXCOM9	Transmit	18
FLEXCOM9	Receive	19
QSPI	Transmit	20
QSPI	Receive	21
PWM0	Transmit	22
PWM1	Transmit	23
TC0	Receive	24
TC1	Receive	25
TC2	Receive	26
TC3	Receive	27
SHA	Transmit	28
TC1_CPA	Compare Counter A, Timer Channel 1	29
TC4_CPA	Compare Counter A, Timer Channel 4	30
TC7_CPA	Compare Counter A, Timer Channel 7	31
TC10_CPA	Compare Counter A, Timer Channel 10	32

.....continued

Instance Name	Channel T/R	DMA Channel HW Interface Number
TC1_CPB	Compare Counter B, Timer Channel 1	33
TC4_CPB	Compare Counter B, Timer Channel 4	34
TC7_CPB	Compare Counter B, Timer Channel 7	35
TC10_CPB	Compare Counter B, Timer Channel 10	36
TC1_CPC	Compare Counter C, Timer Channel 1	37
TC4_CPC	Compare Counter C, Timer Channel 4	38
TC7_CPC	Compare Counter C, Timer Channel 7	39
TC10_CPC	Compare Counter C, Timer Channel 10	40
TC1_ETRG	External Event trigger, Timer Channel 1 for TC1_ETRG	41
TC4_ETRG	External Event trigger, Timer Channel 4 for TC1_ETRG	42
TC7_ETRG	External Event trigger, Timer Channel 7 for TC1_ETRG	43
TC10_ETRG	External Event trigger, Timer Channel 10 for TC1_ETRG	44

12.3 FLEXCOM Features

The FLEXCOM embeds all communication interfaces.

Important: HW flow control is not functional with the DMA.

Table 12-3. FLEXCOM Configuration

FUNC	Sub Function	FLEXCOM									
		0	1	2	3	4	5	6	7	8	9
TWI	Normal/Fast/FM+	X	X	X	X	X	X	X	X	X	X
	Alternate Cmd	X	X	X	X	X	X	X	X	X	X
	3 Slave ADDR	X	X	X	X	X	X	X	X	X	X
	High Speed	X	X	X	X	X	X	X	X	X	X
	SMBUS	X	X	X	X	X	X	X	X	X	X
	Sniffer	X	X	X	X	X	X	X	X	X	X
	FiFO Size	8-byte	8-byte	8-byte	8-byte	8-byte	8-byte	8-byte	8-byte	8-byte	8-byte

.....continued

FUNC	Sub Function	FLEXCOM									
		0	1	2	3	4	5	6	7	8	9
USART	Basic	X	X	X	X	X	X	X	X	X	X
	Hardware Handshaking/RS485	X	X	X	X	X	X	–	–	–	–
	ISO7816	X	X	X	X	X	X	–	–	–	–
	IrDA	X	X	X	X	X	X	X	X	X	X
	LIN	X	X	X	X	X	X	X	X	X	X
	LON	X	–	–	–	–	–	–	–	–	–
	BASIC SPI	–	–	–	–	–	–	–	–	–	–
	MANCHESTER	X	X	X	X	X	X	–	–	–	–
	FiFO size	16-byte	16-byte	16-byte	16-byte	16-byte	16-byte	16-byte	16-byte	16-byte	16-byte
SPI	2CS		X	X			X	–	–	–	–
	4CS	X	–	–	X	X	–	–	–	–	–
	FIFO Size	8-byte	8-byte	8-byte	8-byte	8-byte	8-byte	–	–	–	–

Table 12-4. FLEXCOM Default Functionality (IP Mode after Reset)

FUNC	FLEXCOM									
	0	1	2	3	4	5	6	7	8	9
TWI	–	–	–	–	–	–	–	–	–	–
USART	X	X	X	X	X	X	X	X	X	X
SPI	–	–	–	–	–	–	–	–	–	–

13. ARM Cortex-M7 (ARM)

Refer to the following Arm reference documents, which are available for download at http://www.arm.com.

- *Cortex-M7 Processor User Guide* (ARM DUI 0646B)
- *Cortex-M7 Technical Reference Manual* (ARM DDI 0489D

13.1 ARM Cortex-M7 Configuration

The table below gives the configuration for the Arm Cortex-M7 processor in SAMRH71.

Table 13-1. Arm Cortex-M7 Configuration

Features	Configuration
Debug	
Comparator set	Full comparator set: 4 DWT and 8 FPB comparators
ETM support	Instruction ETM interface
Internal Trace support (ITM)	ITM and DWT trace functionality implemented
CTI and WIC	Not embedded
TCM	
ITCM maximum size	128 KB
DTCM maximum size	256 KB
Cache	
Cache size	16 KB for instruction cache, 16 KB for data cache
Number of sets	256 for instruction cache, 128 for data cache
Number of ways	2 for instruction cache, 4 for data cache
Number of words per cache line	8 words (32 bytes)
ECC on Cache	Embedded
NVIC	
IRQ number	91
IRQ priority levels	3
MPU	
Number of regions	16
FPU	
FPU precision	Single and double precision
AHB Port	
AHBP addressing size	512 MB

14. TCM Hardened Error Correction Code (TCMHECC)

14.1 Description

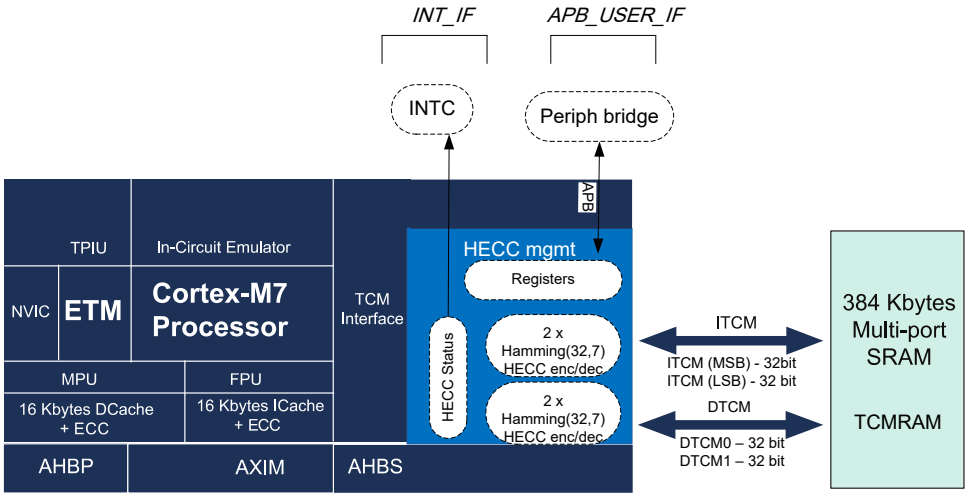
The TCM Hardened Error Correction Code (TCMHECC) is a module embedded in the TCM interface to protect data against radiation effects.

14.2 Embedded Characteristics

- Embedded Hamming (32,7) Code to Correct One Error and Detect up to Two Errors per 32-bit Word
- Interrupt Capabilities
  - One interrupt dedicated to fixable errors
  - One interrupt dedicated to unfixable errors
- Error Counters Available for Statistical Reporting
  - One counter dedicated to fixable errors
  - One counter dedicated to unfixable errors
- Embedded TCMHECC Testing Mode

14.3 Block Diagram

Figure 14-1. TCM Memories – HECC Protection – Block Overview



14.4 Functional Description

14.4.1 Overview

The TCMHECC comprises the following:

- Four Hamming (32,7) encoders/decoders:
  - One Hamming (32,7) for ITCM MSB – 32 bits
  - One Hamming (32,7) for ITCM LSB – 32 bits
  - One Hamming (32,7) for DTCM0 – 32 bits
  - One Hamming (32,7) for DTCM1 – 32 bits
- An APB end user interface:

  - To enable/disable the TCMHECC protection
  - To activate/deactivate the TCMHECC test mode for all Hamming (32,7) blocks (independently in read or write mode)
  - To reset the TCMHECC fixable and unfixable counters
- An APB end user interface for errors status and reporting:
  - Indicates whether fixable errors have been detected and corrected on-the-fly, and quantifies them
  - Indicates whether unfixable errors have been detected and quantifies them
  - The indication can be done either by interruption or by polling
  - Indicates the address and the origin of the last TCMHECC error detected

14.4.2 Operation

When enabled, the TCMHECC operates at each access to the memories.

However, the TCMHECC detection and correction is performed only on read/fetch accesses, by comparing the TCMHECC checksum stored in memory to the one locally computed from data read.

Note: When TCMHECC is enabled on the TCM (ITCM/DTCM) memories, the read-modify-write feature on ITCM/DTCM memories must be enabled to ensure the integrity of the ECC checkbits on sub-word writes.

Note: At power up, the DATA and CHECKSUM of the (I/D)TCM memories are not coherent.
Before enabling the TCCHECC feature, a full initialization of the (I/D)TCM memories must align DATA and CHECKSUM information. For that, the following procedure must be applied:

1. Deactivate RMW and RETRY feature and activate the ENABLE feature on (I/D)TCM.
2. Read and write back all content of I/D(TCM).
   2.1. Deactivate the TCMHECC feature and read the DATA of the (I/D)TCM memories.
   2.2. Activate the TCMHECC feature and write the DATA +CHECKSUM on the (I/D)TCM memories.

Important: At this step, only Write 64-bit instruction must be used.

3. Reactivate the RMW feature and RETRY and the ENABLE feature on (I/D)TCM.

At this step, the (I/D)TCM is fully initialized and DATA and CHECKSUM are aligned.

14.4.3 Interrupt Sources

Using the TCMHECC interrupt requires the interrupt controller to be programmed first.

14.5 Hamming (32, 7) Code

For each word, a 7-bit checksum is generated. With this correction code, errors of one bit can be corrected (fixable error) and errors of up to two bits (unfixable errors) can be detected.

14.5.1 Write Access

When the processor performs a write access to a TCMHECC protected memory, the TCMHECC also writes the 7-bit TCMHECC checksum on this memory.

14.5.2 Read Access

When the processor performs a read access to a TCMHECC protected memory, it reads data and the associated checksum. The checksum read from the TCMHECC protected memory is compared against the checksum generated by the TCMHECC from the same read data. Any discrepancy yields an error and a syndrome is computed to further qualify the error as a fixable error or unfixable error.

14.5.3 Fixable Error

If a fixable error occurs, a retry signal is raised, then the processor may re-access the same data. For this second access, no timing penalty is incurred.

The fixable error event is reported in the Status register (TCMHECC_SR) and the address of the corresponding error is reported in the Fail Address register (TCMHECC_FAILAR).

If the fixable error occurs in the Instruction TCM (ITCM) or in Data TCM (DTCM) memory, the flag TCMHECC_SR.FIX_I or TCMHECC_SR.FIX_D is raised, respectively.

If previously enabled, fixable interrupts are also generated.

When a fixable error is detected, a memory number indicates which memory is involved (TCMHECC_SR.MEM_ID_I or TCMHECC_SR.MEM_ID_D) .

The fixable error remains in memory until a software-initiated rewrite is performed at the faulty memory location.

⚠ CAUTION The Write-back sequence shall be done only in 64-bit for ITCM.

To track the number of fixable errors, a fixable error counter is available. It is incremented each time a fixable error, whether in the DTCM or the ITCM, is detected and until a counter overload is detected.

The fixable counter is reset using RST_NOFIX_CPT in TCMHECC_CR.

14.5.4 Unfixable Error

The unfixable error event is reported in the Status register (TCMHECC_SR) and the address of the corresponding error is reported in the Fail Address register (TCMHECC_FAILAR).

If the unfixable error occurs in the Instruction TCM (ITCM) or in Data TCM (DTCM) memory, the flag TCMHECC_SR.NOFIX_I or TCMHECC_SR.NOFIX_D is raised, respectively.

If previously enabled, unfixable interrupts are also generated.

When an unfixable error is detected, a memory number indicates which memory is involved (TCMHECC_SR.MEM_ID_I or TCMHECC_SR.MEM_ID_D) .

The unfixable error remains in memory until a software-initiated rewrite is performed at the faulty memory location.

⚠ CAUTION The Write sequence shall be done only in 64-bit for ITCM.

To track the number of unfixable errors, an unfixable error counter is available. It is incremented each time an unfixable error, whether in the DTCM or the ITCM, is detected and until a counter overload is detected.

The unfixable counter is reset using RST_NOFIX_CPT in TCMHECC_CR.

14.6 TCMHECC Testing

Operation of the TCMHECC can be bypassed for testing purposes and is controlled in the Control register (TCMHECC_CR).

When testing mode is activated, it is applied on the four Hamming(32,7) encoder/decoder blocks.

14.6.1 Write Test

When TCMHECC write bypass is enabled (TCMHECC_CR.TEST_MODE_WR=1), the test checksum (TCMHECC_TESTCB1.TCB1 and TCMHECC_TESTCB1.TCB2) replaces the TCMHECC-generated checksum during a data store.

- TCMHECC_TESTCB1.TCB1 is relative to DTCM0, DTCM1, ITCM (LSB)
- TCMHECC_TESTCB1.TCB2 is relative to to ITCM (MSB)

Note: When a write 64-bit data is used in Write Data Test mode, the same data must be written twice (on the 32-bit MSB and the 32-bit LSB) because only ECC_TESTCB1.TCB1 is used to inject checksum on DTCM0 and DTCM1.

14.6.2 Read Test

When the TCMHECC read diagnostic is enabled (TCMHECC_CR.TEST_MODE_RD=1), the test checksum (TCMHECC_TESTCB1.TCB1 and TCMHECC_TESTCB1.TCB2) is updated with the read checksum during a data load or an instruction fetch.

Note: In read test mode, only read of 8-, 16- or 32-bit data can be performed in order to have the corresponding checksum read stored in ECC_TESTCB1.TCB1.
If 64-bit read data instruction is used, only checksum corresponding to the 32-bit LSB is stored in ECC_TESTCB1.TCB1. Reading the checksum corresponding to the 32-bit MSB is not possible.

14.7 Register Summary

Offset	Name	Bit Pos.								
0x00	TCMHECC_CR	7:0			RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ENABLE
		15:8								
		23:16								
		31:24								
0x04	TCMHECC_TESTCB1	7:0	TCB1[7:0]							
		15:8	TCB2[7:0]							
		23:16								
		31:24								
0x08 ... 0x0B	Reserved									
0x0C	TCMHECC_SR	7:0	OVER_FIX	CPT_FIX[4:0]					MEM_FIX_D	MEM_FIX_I
		15:8	OVER_NOFIX	CPT_NOFIX[4:0]					MEM_NOFIX_D	MEM_NOFIX_I
		23:16								
		31:24			MEM_ID_D	MEM_ID_I	ONE	HES[2:0]		
0x10	TCMHECC_IER	7:0					MEM_NOFIX_D	MEM_FIX_D	MEM_NOFIX_I	MEM_FIX_I
		15:8								
		23:16								
		31:24								
0x14	TCMHECC_IDR	7:0					MEM_NOFIX_D	MEM_FIX_D	MEM_NOFIX_I	MEM_FIX_I
		15:8								
		23:16								
		31:24								
0x18	TCMHECC_IMR	7:0					MEM_NOFIX_D	MEM_FIX_D	MEM_NOFIX_I	MEM_FIX_I
		15:8								
		23:16								
		31:24								
0x1C	TCMHECC_FAILAR	7:0	ADDRESS[7:0]							
		15:8	ADDRESS[15:8]							
		23:16	ADDRESS[23:16]							
		31:24	ADDRESS[31:24]							
0x20	TCMHECC_FAILARD	7:0	ADDRESS[7:0]							
		15:8	ADDRESS[15:8]							
		23:16	ADDRESS[23:16]							
		31:24	ADDRESS[31:24]							

14.7.1 TCMHECC Control Register

Name: TCMHECC_CR
Offset: 0x00
Reset: 0x00000001
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ENABLE
Access			R/W	R/W		R/W	R/W	R/W
Reset			0	0		0	0	1

Bit 5 – RST_NOFIX_CPT Reset the Unfixable Error Counter
This bit automatically returns to ’0’ once the operation is done.

Value	Description
0	No effect.
1	Resets the TCMHECC unfixable error counter.

Bit 4 – RST_FIX_CPT Reset the Fixable Error Counter
This bit automatically returns to ’0’ once the operation is done.

Value	Description
0	No effect.
1	Resets the TCMHECC fixable error counter.

Bit 2 – TEST_MODE_WR Test Mode of TCMHECC Protection - Write Mode
The ENABLE bit must be enabled to use TEST_MODE_WR.

Value	Description
0	Write test mode is deactivated.
1	Write test mode is activated.

Bit 1 – TEST_MODE_RD Test Mode of TCMHECC Protection - Read Mode
The ENABLE bit must be enabled to use TEST_MODE_RD.

Value	Description
0	Read test mode is deactivated.
1	Read test mode is activated.

Bit 0 – ENABLE TCMHECC Protection Enable

Value	Description
0	Protection is deactivated.
1	Protection is activated.

14.7.2 TCMHECC Test Mode Register 1

Name: TCMHECC_TESTCB1
Offset: 0x4
Reset: 0x00000000
Property: Read/Write

TCB1[7] and TCB2[7] are fixed to 0.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TCB2[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TCB1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:8 – TCB2[7:0] Test Check Bit (8-bit)
Check bit for test purposes (ITCM_MSB is stored here).
TCB2[7] is fixed to 0.

Bits 7:0 – TCB1[7:0] Test Check Bit (8-bit)
Check bit for test purposes (D0TCM, D1TCM or ITCM LSB is stored here).
TCB1[7] is fixed to 0.

14.7.3 TCMHECC Status Register

Name: TCMHECC_SR
Offset: 0xC
Reset: 0x0A000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
			MEM_ID_D	MEM_ID_I	ONE	HES[2:0]		
Access			R	R	R	R	R	R
Reset			0	0	1	0	1	0

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	OVER_NOFIX	CPT_NOFIX[4:0]					MEM_NOFIX_D	MEM_NOFIX_I
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	OVER_FIX	CPT_FIX[4:0]					MEM_FIX_D	MEM_FIX_I
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 29 – MEM_ID_D Memory Identification Number
Defines the memory to which the error corresponds.

Value	Description
0	DTCM0 is selected.
1	DTCM1 is selected.

Bit 28 – MEM_ID_I Memory Identification Number
Defines the memory to which the error corresponds.

Value	Description
0	ITCM LSB is selected.
1	ITCM MSB is selected.

Bit 27 – ONE One
Always fixed to 1.

Bits 26:24 – HES[2:0] Hardware Error Size
Identifies the size of the failed access.
Always fixed to 010=word.

Bit 15 – OVER_NOFIX Counter Overflow

Value	Description
0	No overflow has occurred since the last read on the unfixable counter.
1	An overflow has occurred since the last read on the unfixable counter.

Bits 14:10 – CPT_NOFIX[4:0] 5-bit Counter
Counts the number of unfixable errors detected for one type of memory during the mission.

Bit 9 – MEM_NOFIX_D Unfixable Error Status in Data Memory

Value	Description
0	No unfixable error detected on data memory
1	An unfixable error has been detected in data memory.

Bit 8 – MEM_NOFIX_I Unfixable Error Status in Instruction Memory

Value	Description
0	No unfixable error detected on instruction memory
1	An unfixable error has been detected in instruction memory.

Bit 7 – OVER_FIX Counter Overflow

Value	Description
0	No overflow has occurred since the last read on the fixable counter.
1	An overflow has occurred since the last read on the fixable counter.

Bits 6:2 – CPT_FIX[4:0] 5-bit Counter
Counts the number of fixable errors detected for one type of memory during the mission.

Bit 1 – MEM_FIX_D Fixable Error Status in Data Memory

Value	Description
0	No fixable error detected on data memory
1	A fixable error has been detected on data memory.

Bit 0 – MEM_FIX_I Fixable Error Status in Instruction Memory

Value	Description
0	No fixable error detected on instruction memory
1	A fixable error has been detected on instruction memory.

14.7.4 TCMHECC Interrupt Enable Register

Name: TCMHECC_IER
Offset: 0x10
Reset: 0x00000000
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					MEM_NOFIX_D	MEM_FIX_D	MEM_NOFIX_I	MEM_FIX_I
Access					W	W	W	W
Reset					0	0	0	0

Bit 3 – MEM_NOFIX_D Unfixable Error On Data

Bit 2 – MEM_FIX_D Fixable Error On Data

Bit 1 – MEM_NOFIX_I Unfixable Error On Instruction

Bit 0 – MEM_FIX_I Fixable Error On Instruction

14.7.5 TCMHECC Interrupt Disable Register

Name: TCMHECC_IDR
Offset: 0x14
Reset: 0x00000000
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					MEM_NOFIX_D	MEM_FIX_D	MEM_NOFIX_I	MEM_FIX_I
Access					W	W	W	W
Reset					0	0	0	0

Bit 3 – MEM_NOFIX_D Unfixable Error On Data

Bit 2 – MEM_FIX_D Fixable Error On Data

Bit 1 – MEM_NOFIX_I Unfixable Error On Instruction

Bit 0 – MEM_FIX_I Fixable Error On Instruction

14.7.6 TCMHECC Interrupt Mask Register

Name: TCMHECC_IMR
Offset: 0x18
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					MEM_NOFIX_D	MEM_FIX_D	MEM_NOFIX_I	MEM_FIX_I
Access					R	R	R	R
Reset					0	0	0	0

Bit 3 – MEM_NOFIX_D Unfixable Error On Data

Bit 2 – MEM_FIX_D Fixable Error On Data

Bit 1 – MEM_NOFIX_I Unfixable Error On Instruction

Bit 0 – MEM_FIX_I Fixable Error On Instruction

14.7.7 TCMHECC Fail Address Register

Name: TCMHECC_FAILAR
Offset: 0x1C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ADDRESS[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDRESS[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDRESS[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDRESS[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDRESS[31:0] Address of the Error Detected in ITCM
The address of the faulty data detected when a fixable or an unfixable error occurs in ITCM memory.

14.7.8 TCMHECC Fail Address Register Data

Name: TCMHECC_FAILARD
Offset: 0x20
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ADDRESS[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDRESS[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDRESS[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDRESS[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDRESS[31:0] Address of the Error Detected on DCTM
The address of the faulty data detected when a fixable or an unfixable error occurs in DTCM memory.

15. FlexRAM Memory and Embedded Hardened ECC (FLEXRAMECC) Controller

15.1 Description

The FlexRAM comprises 768 Kbytes of internal protected memory.

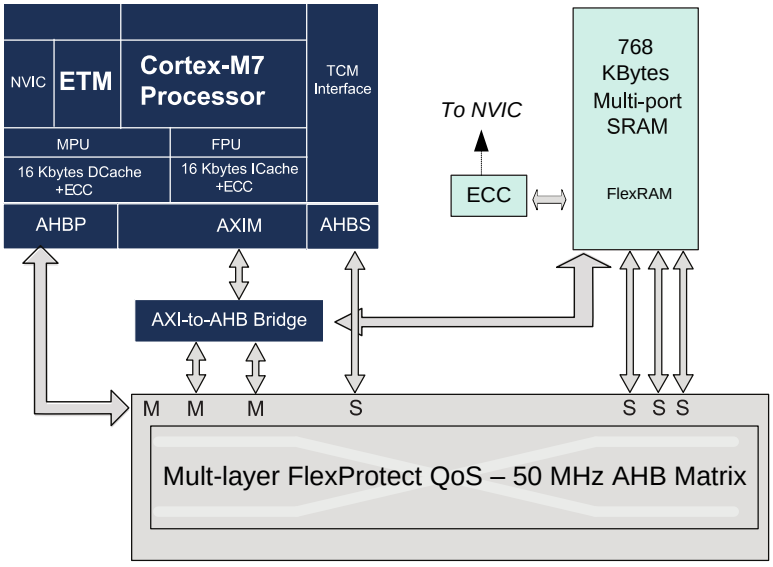
The embedded Hardened Error Correction Code (HECC) Controller for FlexRAM protects this data against the effects of radiation.

15.2 Embedded Characteristics

- 768 Kbytes of Internal Memory
- Three Slaves Access the HMATRIX
- Three Channels Depending on the Memory Type and the Access Used
- Embedded Hamming (32,7) Code to Correct One Error and Detect up to Two Errors
- Interrupt Capabilities
  - One interrupt dedicated to fixable errors
  - One interrupt dedicated to unfixable errors
- Error Counter Available for Statistical Reporting
  - One counter dedicated to fixable errors
  - One counter dedicated to unfixable errors
- Embedded HECC for FlexRAM Testing Mode

15.3 Block Diagram

Figure 15-1. Block Diagram



15.4 Functional Description

15.4.1 Overview

The HECC for FlexRAM comprises the following:

- One encoder/decoder (Hamming (32,7)
- An APB end user interface used to configure:
  - The FLEXRAMECC test mode (independently in read or write mode)
  - The reset of the FLEXRAMECC fixable and unfixable counters
- An APB end user interface for errors status and reporting:
  - Indicates whether fixable errors have been detected and corrected on-the-fly, and quantifies them
  - Indicates whether unfixable errors have been detected and quantifies them
  - The indication can be done either by interruption or by polling
  - Indicates the address and the status of the last FLEXRAMECC error detected

15.4.2 Operation

When enabled, the FLEXRAMECC operates at each access to the memories.

However, the FLEXRAMECC detection and correction is performed only on read/fetch accesses, by comparing the FLEXRAMECC checksum stored in memory to the one locally computed from data read.

15.4.3 Interrupt Sources

Using the FLEXRAMECC interrupt requires the interrupt controller to be programmed first.

15.5 Hamming (32, 7) Code

For each word, a 7-bit checksum is generated. With this correction code, errors of one bit can be corrected (fixable error) and errors of up to two bits (unfixable errors) can be detected.

15.5.1 Write Access

When the processor performs a write access to an FLEXRAMECC protected memory, it also writes the 7-bit FLEXRAMECC checksum on the FLEXRAMECC protected memory.

15.5.2 Read Access

When the processor performs a read access to a FLEXRAMECC protected memory, it reads data and the associated checksum. The checksum read from the FLEXRAMECC protected memory is compared against the checksum generated by the FLEXRAMECC from the same read data. Any discrepancy yields an error and a syndrome is computed to further qualify the error as a fixable error or unfixable error.

15.5.3 Fixable Error

The correction of a fixable error is performed on-the-fly inside the processor during the current access and no timing penalty is incurred. It is also automatically corrected inside the memory.

The fixable error event is reported in the Status register (FLEXRAMECC_SR) and the address of the corresponding error is reported in the Fail Address register (FLEXRAMECC_FAILAR). If previously enabled, a fixable interrupt is generated.

To track the number of fixable errors, a fixable error counter is available. It is incremented each time a fixable error is detected and until a counter overload is detected.

It is possible for the end user to reset the fixable counter using RST_FIX_CPT in the Control register (FLEXRAMECC_CR).

15.5.4 Unfixable Error

The unfixable error event is reported in FLEXRAMECC_SR and the address of the corresponding error is reported in FLEXRAMECC_FAILAR. If previously enabled, an unfixable interrupt is generated.

To track the number of unfixable errors, an unfixable error counter is available. It is incremented each time an unfixable error is detected and until a counter overload is detected.

It is possible for the end user to reset the unfixable counter using RST_NOFIX_CPT in FLEXRAMECC_CR.

15.6 FLEXRAMECC Testing

Operation of the FLEXRAMECC can be bypassed for testing purposes and is controlled in FLEXRAMECC_CR.

15.6.1 Write Test

When FLEXRAMECC write bypass is enabled (FLEXRAMECC_CR.TEST_MODE_WR=1), the test checksum (FLEXRAMECC_TESTCB1.TCB1) replaces the FLEXRAMECC generated checksum during a data store.

15.6.2 Read Test

When the FLEXRAMECC read diagnostic is enabled (FLEXRAMECC_CR.TEST_MODE_RD=1), the test checksum (FLEXRAMECC_TESTCB1.TCB1) is updated with the read checksum during a data load or an instruction fetch.

Note: The FLEXRAMECC test routine must be executed entirely from the instruction cache (when activated) or from an area without the FLEXRAMECC activated and different from the one being accessed. Otherwise, the checksum read during instruction fetch will overwrite those from the area to be tested.

15.7 Register Summary

Offset	Name	Bit Pos.								
0x00	FLEXRAMECC_CR	7:0			RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ONE
		15:8								
		23:16								
		31:24								
0x04	FLEXRAMECC_TESTCB1	7:0	TCB1[7:0]							
		15:8								
		23:16								
		31:24								
0x08 ... 0x0B	Reserved									
0x0C	FLEXRAMECC_SR	7:0	OVER_FIX	CPT_FIX[4:0]						MEM_FIX
		15:8	OVER_NOFIX	CPT_NOFIX[4:0]						MEM_NOFIX
		23:16								
		31:24					TYPE	HES[2:0]		
0x10	FLEXRAMECC_IER	7:0							MEM_NOFIX	MEM_FIX
		15:8								
		23:16								
		31:24								
0x14	FLEXRAMECC_IDR	7:0							MEM_NOFIX	MEM_FIX
		15:8								
		23:16								
		31:24								
0x18	FLEXRAMECC_IMR	7:0							MEM_NOFIX	MEM_FIX
		15:8								
		23:16								
		31:24								
0x1C	FLEXRAMECC_FAILAR	7:0	ADDRESS[7:0]							
		15:8	ADDRESS[15:8]							
		23:16	ADDRESS[23:16]							
		31:24	ADDRESS[31:24]							

15.7.1 FLEXRAMECC Control Register

Name: FLEXRAMECC_CR
Offset: 0x0
Reset: 0x00000001
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ONE
Access			R/W	R/W		R/W	R/W	R/W
Reset			0	0		0	0	1

Bit 5 – RST_NOFIX_CPT Reset the Unfixable Error Counter
This bit automatically returns to ’0’ once the operation is done.

Value	Description
0	No effect.
1	Resets the ECC unfixable error counter.

Bit 4 – RST_FIX_CPT Reset the Fixable Error Counter
This bit automatically returns to ’0’ once the operation is done.

Value	Description
0	No effect.
1	Resets the ECC fixable error counter.

Bit 2 – TEST_MODE_WR Test Mode of ECC Protection - Write Mode

Value	Description
0	Write test mode is deactivated.
1	Write test mode is activated.

Bit 1 – TEST_MODE_RD Test Mode of ECC Protection - Read Mode

Value	Description
0	Read test mode is deactivated.
1	Read test mode is activated.

Bit 0 – ONE Always ‘1’.

15.7.2 FLEXRAMECC Test Mode Register 1

Name: FLEXRAMECC_TESTCB1
Offset: 0x4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	TCB1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – TCB1[7:0] Test Check Bit
Check bit for test purposes.

15.7.3 FLEXRAMECC Status Register

Name: FLEXRAMECC_SR
Offset: 0xC
Reset: 0x0A000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
					TYPE	HES[2:0]		
Access					R/W	R/W	R/W	R/W
Reset					1	0	1	0

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	OVER_NOFIX	CPT_NOFIX[4:0]						MEM_NOFIX
Access	R/W	R/W	R/W	R/W	R/W	R/W		R/W
Reset	0	0	0	0	0	0		0

Bit	7	6	5	4	3	2	1	0
	OVER_FIX	CPT_FIX[4:0]						MEM_FIX
Access	R/W	R/W	R/W	R/W	R/W	R/W		R/W
Reset	0	0	0	0	0	0		0

Bit 27 – TYPE Write or Read Access. Always ‘1’.

Value	Description
0	Write access.
1	Read access.

Bits 26:24 – HES[2:0] Hardware Error Size
Identifies the size of the failed access.

Bit 15 – OVER_NOFIX Counter Overflow

Value	Description
0	No overflow has occurred since the last read on unfixable counter.
1	An overflow has occurred since the last read on unfixable counter.

Bits 14:10 – CPT_NOFIX[4:0] 5-bit Counter
Counts the number of unfixable errors detected for one type of memory during the mission.

Bit 8 – MEM_NOFIX Unfixable Error Status

Value	Description
0	No unfixable error detected.
1	An unfixable error has been detected.

Bit 7 – OVER_FIX Counter Overflow

Value	Description
0	No overflow has occurred since the last read.
1	An overflow has occurred since the last read.

Bits 6:2 – CPT_FIX[4:0] 5-bit Counter
Counts the number of fixable errors detected for one type of memory during the mission.

Bit 0 – MEM_FIX Fixable Error Status
A fixable error has been detected.

Value	Description
0	No fixable error detected.
1	A fixable error has been detected.

15.7.4 FLEXRAMECC Interrupt Enable Register

Name: FLEXRAMECC_IER
Offset: 0x10
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							MEM_NOFIX	MEM_FIX
Access							W	W
Reset							–	–

Bit 1 – MEM_NOFIX Unfixable Error

Bit 0 – MEM_FIX Fixable Error

15.7.5 FLEXRAMECC Interrupt Disable Register

Name: FLEXRAMECC_IDR
Offset: 0x14
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							MEM_NOFIX	MEM_FIX
Access							W	W
Reset							–	–

Bit 1 – MEM_NOFIX Unfixable Error

Bit 0 – MEM_FIX Fixable Error

15.7.6 FLEXRAMECC Interrupt Mask Register

Name: FLEXRAMECC_IMR
Offset: 0x18
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							MEM_NOFIX	MEM_FIX
Access							R	R
Reset							0	0

Bit 1 – MEM_NOFIX Unfixable Error

Bit 0 – MEM_FIX Fixable Error

15.7.7 FLEXRAMECC Fail Address Register

Name: FLEXRAMECC_FAILAR
Offset: 0x1C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ADDRESS[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDRESS[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDRESS[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDRESS[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDRESS[31:0] Address of the Error Detected
The address of the faulty data detected when a fixable or unfixable error occurs.

16. Debug and Test Features

16.1 Description

The device features a number of complementary debug and test capabilities. The Serial Wire Debug Port (SW-DP) is used for standard debugging functions, such as downloading code and single-stepping through programs. It also embeds a serial wire trace.

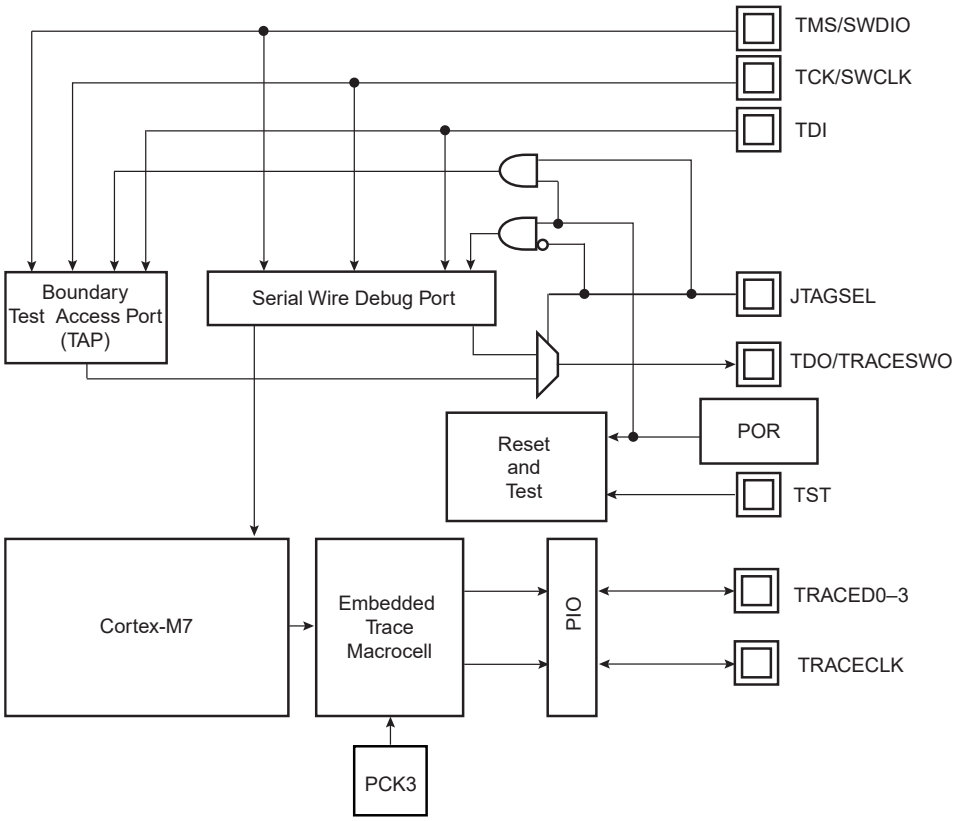
16.2 Associated Documents

The device implements the standard Arm CoreSight macrocell. For detailed information on CoreSight, refer to the following documents which are available at http://www.arm.com/:

- Cortex-M7 User Guide Reference Manual (ARM DUI 0646B)
- Cortex-M7 Technical Reference Manual (ARM DDI 0489D)
- CoreSight Technology System Design Guide (ARM DGI 0012)
- CoreSight Components Technical Reference Manual (ARM DDI 0314)
- Arm Debug Interface v5 Architecture Specification (Doc. ARM IHI 0031)
- ARMv7-M Architecture Reference Manual (ARM DDI 0403E)

16.3 Debug and Test Block Diagram

Figure 16-1. Debug and Test Block Diagram



16.4 Debug and Test Pin Description

Table 16-1. Debug and Test Signal List

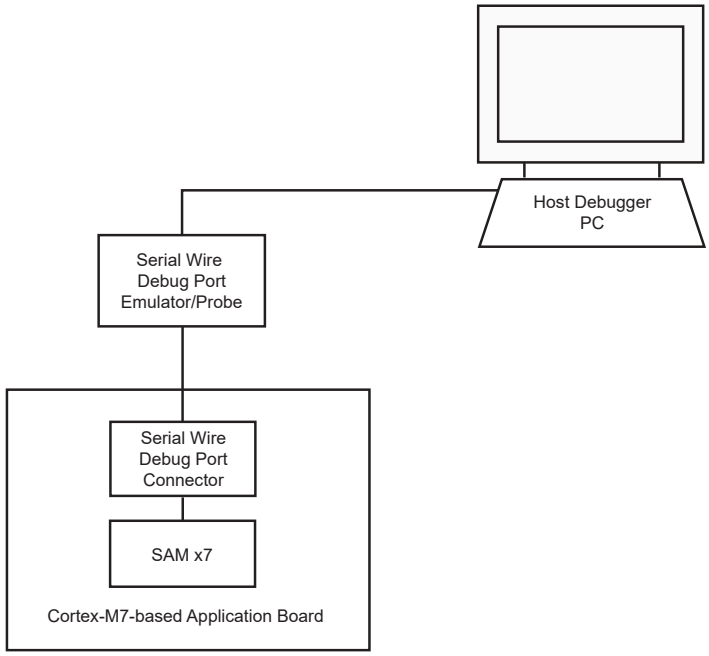
Signal Name	Function	Type	Active Level
Reset/Test			
NRST	Microcontroller Reset	Input	Low
TST	Test Select	Input	–
Serial Wire Debug Port/JTAG Boundary Scan			
TCK/SWCLK	Test Clock/Serial Wire Clock	Input	–
TDI	Test Data In	Input	–
TDO/TRACESWO	Test Data Out/Trace Asynchronous Data Out	Output	–
TMS/SWDIO	Test Mode Select/Serial Wire Input/Output	Input	–
JTAGSEL	JTAG Selection	Input	High
Trace Debug Port			
TRACECLK	Trace Clock	Output	–
TRACED0–3	Trace Data	Output	–

16.5 Application Examples

16.5.1 Debug Environment

The figure below shows a complete debug environment example. The SW-DP interface is used for standard debugging functions, such as downloading code and single-stepping through the program and viewing core and peripheral registers.

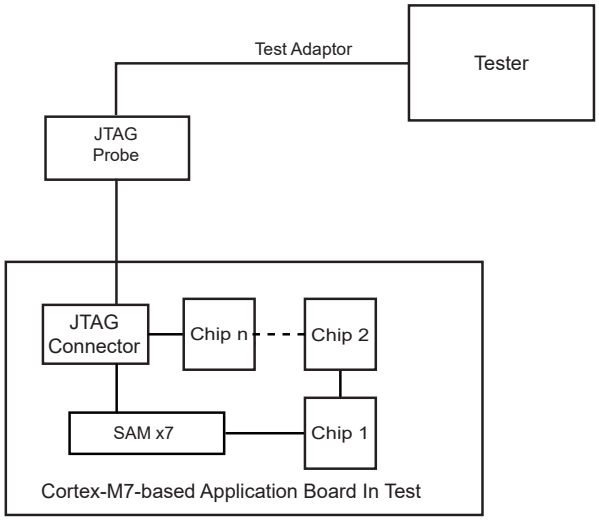
Figure 16-2. Application Debug Environment Example



16.5.2 Test Environment

The figure below shows a test environment example (JTAG Boundary scan). Test vectors are sent and interpreted by the tester. In this example, the “board in test” is designed using a number of JTAG-compliant devices. These devices can be connected to form a single scan chain.

Figure 16-3. Application Test Environment Example



16.6 Functional Description

16.6.1 Test Pin

The TST pin is used for JTAG Boundary Scan Manufacturing Test or Fast Flash Programming mode. The TST pin integrates a permanent pull-down resistor of about 10 kΩ to GND, so that it can be left unconnected for normal operations.

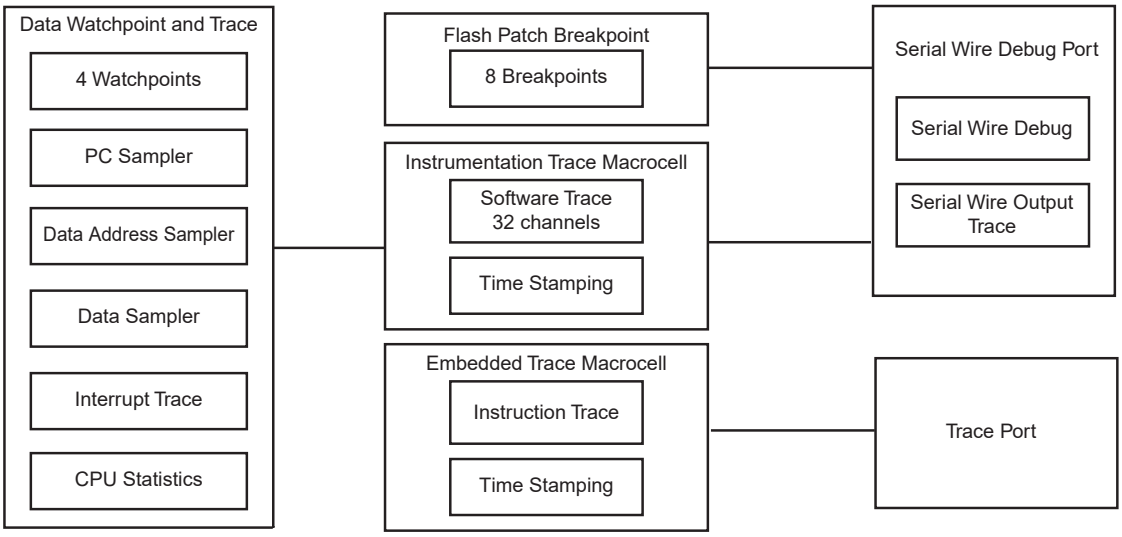
16.6.2 Debug Architecture

The figure below shows the debug architecture used. The Cortex-M7 embeds six functional units for debug:

- Serial Wire Debug Port (SW-DP) debug access
- FPB (Flash Patch Breakpoint), (no FP_REMAP register supported in Cortex-M7)
- DWT (Data Watchpoint and Trace)
- ITM (Instrumentation Trace Macrocell)
- 6-pin Embedded Trace Macrocell (ETM) for instruction trace stream, including CoreSight Trace Port Interface Unit (TPIU)
- IEEE1149.1 JTAG Boundary scan on all digital pins

The debug architecture information that follows is mainly dedicated to developers of SW-DP Emulators/Probes and debugging tool vendors for Cortex-M7-based microcontrollers. For further details on SW-DP, refer to the Cortex-M7 Technical Reference Manual.

Figure 16-4. Debug Architecture



16.6.3 Serial Wire Debug Port (SW-DP) Pins

The SW-DP pins SWCLK and SWDIO are commonly provided on a standard 20-pin JTAG connector defined by Arm. For more details on voltage reference and reset state, refer to "Signal Description List".

At start-up, SW-DP pins are configured in SW-DP mode to allow connection with debugging probe.

SW-DP pins can be used as standard I/Os to provide users more general input/output pins when the debug port is not needed in the end application. Mode selection between SW-DP mode (System I/O mode) and general I/O mode is performed through the PIO Controller. Configuration of the pad for pull-up, triggers, debouncing and glitch filters is possible regardless of the mode.

The JTAGSEL pin is used to select the JTAG boundary scan when asserted at a high level. It integrates a permanent pull-down resistor of about 10 kΩ to GND, so that it can be left unconnected for normal operations.

The JTAG debug ports TDI, TDO, TMS and TCK are inactive. They are provided for Boundary Scan Manufacturing Test purposes only. By default the SW-DP is active; TDO/TRACESWO can be used for trace.

Table 16-2. SW-DP Pin List

Pin Name	JTAG Boundary Scan	Serial Wire Debug Port
TMS/SWDIO	TMS	SWDIO
TCK/SWCLK	TCK	SWCLK
TDI	TDI	–
TDO/TRACESWO	TDO	TRACESWO (optional: ITM trace)

SW-DP is selected when JTAGSEL is low. It is not possible to switch directly between SW-DP and JTAG boundary scan operations. A chip reset must be performed after JTAGSEL is changed.

16.6.4 Embedded Trace Module (ETM) Pins

The Embedded Trace Module (ETM) uses the Trace Port Interface Unit (TPIU) to export data out of the system.

The TPIU features these pins:

- TRACECLK – Always exported to enable synchronization back with the data. PCK3 is used internally.
- TRACED0–3 – The instruction trace stream.

16.6.5 Flash Patch Breakpoint (FPB)

The FPB implements hardware breakpoints. No Flash patching is available in the Cortex-M7.

16.6.6 Data Watchpoint and Trace (DWT)

The DWT contains four comparators which can be configured to generate:

- PC sampling packets at set intervals
- PC or Data watchpoint packets
- Watchpoint event to halt core

The DWT contains counters for:

- Clock cycle (CYCCNT)
- Folded instructions
- Load Store Unit (LSU) operations
- CPI (all instruction cycles except for the first cycle)
- Interrupt overhead

16.6.7 Instrumentation Trace Macrocell (ITM)

The ITM is an application driven trace source that supports printf style debugging to trace Operating System (OS) and application events, and emits diagnostic system information. The ITM emits trace information as packets which can be generated by three different sources with several priority levels:

- Software trace: Software can write directly to ITM stimulus registers. This can be done using the “printf” function. For more information, see Flash Patch Breakpoint (FPB).
- Hardware trace: The ITM emits packets generated by the DWT.
- Timestamping: Timestamps are emitted relative to packets. The ITM contains a 21-bit counter to generate the timestamp.

16.6.7.1 How to Configure the ITM

The following example describes how to output trace data in asynchronous trace mode.

1. Configure the TPIU for asynchronous trace mode. See How to Configure the TPIU.
2. Enable the write accesses into the ITM registers by writing “0xC5ACCE55” into the Lock Access Register (Address: 0xE0000FB0)
3. Write 0x00010015 into the Trace Control register:
   - Enable ITM.
   - Enable Synchronization packets.
   - Enable SWO behavior.
   - Fix the ATB ID to 1.
4. Write 0x1 into the Trace Enable register:
   - Enable the Stimulus port 0.
5. Write 0x1 into the Trace Privilege register:
   - Stimulus port 0 only accessed in privileged mode (Clearing a bit in this register will result in the corresponding stimulus port being accessible in user mode.)
6. Write into the Stimulus port 0 register: TPIU (Trace Port Interface Unit)
   The TPIU acts as a bridge between the on-chip trace data and the Instruction Trace Macrocell (ITM).

   The TPIU formats and transmits trace data off-chip at frequencies asynchronous to the core.

16.6.7.2 Asynchronous Mode

The TPIU is configured in Asynchronous mode, trace data are output using the single TRACESWO pin. The TRACESWO signal is multiplexed with the TDO signal. As a consequence, asynchronous trace mode is only available when the Serial Wire Debug mode is selected.

Two encoding formats are available for the single pin output:

- Manchester encoded stream. This is the reset value.
- NRZ_based UART byte structure

16.6.7.3 How to Configure the TPIU

This example only concerns the asynchronous trace mode.

1. Set the TRCENA bit to 1 into the Debug Exception and Monitor Register (0xE000EDFC) to enable the use of trace and debug blocks.
2. Write 0x2 into the Selected Pin Protocol Register.
   – Select the Serial Wire output – NRZ
3. Write 0x100 into the Formatter and Flush Control Register.
4. Set the suitable clock prescaler value into the Async Clock Prescaler Register to scale the baud rate of the asynchronous output (this can be done automatically by the debugging tool).

16.6.8 IEEE1149.1 JTAG Boundary Scan

IEEE1149.1 JTAG Boundary Scan allows pin-level access independent of the device packaging technology.

IEEE1149.1 JTAG Boundary Scan is enabled when TST is tied to low and JTAGSEL tied to high during power-up. These pins must be maintained in their respective states for the duration of the boundary scan operation. The SAMPLE, EXTEST and BYPASS functions are implemented. In Serial Wire Debug mode, the Arm processor responds with a non-JTAG chip ID that identifies the processor. This is not IEEE1149.1 JTAG-compliant.

It is not possible to switch directly between JTAG Boundary Scan and SWJ Debug Port operations. A chip reset must be performed after JTAGSEL is changed.

A Boundary Scan Descriptor Language (BSDL) file to set up the test is provided on http://www.microchip.com.

16.6.8.1 JTAG Boundary Scan Register

The Boundary Scan Register (BSR) contains a number of bits which correspond to active pins and associated control signals.

Each input/output pin corresponds to a 3-bit register in the BSR. The OUTPUT bit contains data that can be forced on the pad. The INPUT bit facilitates the observability of data applied to the pad. The CONTROL bit selects the direction of the pad.

For more information, refer to BDSL files available on http://www.microchip.com.

16.6.9 ID Code Register

Name: ID Code Register
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	VERSION[3:0]				PART NUMBER[15:12]			
Access	R	R	R	R	R	R	R	R
Reset								

Bit	23	22	21	20	19	18	17	16
	PART NUMBER[11:4]							
Access	R	R	R	R	R	R	R	R
Reset								

Bit	15	14	13	12	11	10	9	8
	PART NUMBER[3:0]				MANUFACTURER IDENTITY[10:7]			
Access	R	R	R	R	R	R	R	R
Reset								

Bit	7	6	5	4	3	2	1	0
	MANUFACTURER IDENTITY[6:0]							Bit [0]=1
Access	R	R	R	R	R	R	R	
Reset								

Bits 31:28 – VERSION[3:0] Product Version Number
Set to 0x0.

Bits 27:12 – PART NUMBER[15:0] Product Part Number

PART NUMBER
0x05D0

Bits 11:1 – MANUFACTURER IDENTITY[10:0]
Set to 0x01F.

Bit 0 – Bit [0]=1 Required by IEEE Std. 1149.1
Set to 0x1.

JTAG ID Code
0x05D0_503F

17. Bus Matrix (MATRIX)

17.1 Description

The Bus Matrix (MATRIX) implements a multi-layer AHB , based on the AHB-Lite protocol, that enables parallel access paths between multiple masters and slaves in a system, thus increasing the overall bandwidth.

The MATRIX interconnects up to 16 masters to up to 16 slaves. The normal latency to connect a master to a slave is one cycle, except for the default master of the accessed slave which is connected directly (zero cycle latency).

The MATRIX user interface is compliant with the ARM Advanced Peripheral Bus and provides 16 Special Function registers (MATRIX_SFR) that enable the MATRIX to support application-specific features.

17.1.1 Matrix Masters

Table 17-1. List of Bus Matrix Masters

Master 0	Cortex-M7 Port 0
Master 1	Cortex-M7 Port 1
Master 2	Cortex-M7 Peripheral Port (AHBP)
Master 3	XDMAC Port 0
Master 4	CAN0 DMA
Master 5	CAN1 DMA
Master 6	GMAC DMA
Master 7	SPW Master 0–TX
Master 8	SPW Master 1–RX
Master 9	SPW Master 2–RMAP
Master 10	IP1553
Master 11	ICM
Master 12	XDMAC Port 1

17.1.2 Matrix Slaves

Table 17-2. List of Bus Matrix Slaves

Slave 0	FlexRAM
Slave 1	FlexRAM
Slave 2	Internal Flash (HEFC)
Slave 3	HEMC
Slave 4	QSPI
Slave 5	Peripheral Bridge
Slave 6	AHB Slave
Slave 7	FlexRAM

17.1.3 Master to Slave Access

All the Masters can normally access all the Slaves. However, some paths do not make sense. Thus, these paths are forbidden or simply not wired, and shown as “–” in the following table.

Table 17-3. Master to Slave Access

Slave	Master	0	1	2	3	4	5	6	7	8	9	10	11	12
		CM7		CM7 AHBP	XDMAC Port 0	CAN0 DMA	CAN1 DMA	GMAC DMA	SpW Tx	SpW Rx	SpW RMAP	IP1553	ICM	XDMAC Port 1
S0	FlexRAM	–	–	–	X	X	–	–	X	–	–	X	–	–
S1	FlexRAM	–	–	–	–	–	X	–	–	–	X	–	–	X
S2	Internal Flash (HEFC)	X	–	–	X	–	–	–	X	X	X	X	X	–
S3	External Bus Interface (HEMC)	X	–	–	–	X	X	X	X	X	X	X	X	X
S4	QSPI	–	X	–	–	–	–	–	–	–	–	–	X	X
S5	Peripheral Bridge	–	X	X	–	–	–	–	–	–	–	–	–	X
S6	AHB Slave	–	–	–	X	X	X	X	X	X	X	X	X	X
S7	FlexRAM	–	–	–	–	–	–	X	–	X	–	–	X	–
S8	Reserved	–	–	–	–	–	–	–	–	–	–	–	–	–
S9	Reserved	–	–	–	–	–	–	–	–	–	–	–	–	–

17.2 Embedded Characteristics

- AMBA Advanced High-performance Bus (AHB Lite) Compliant Interfaces
- 32-bit Data Bus
- APB-Compliant User Interface
- Thirteen Masters
- Eight Slaves
- One Decoder for Each Master
- Several Possible Boot Memories for Each Master before Remap
- One Remap Function for Each Master
- Support for Long Bursts of 32, 64, 128 and Up to the 256-beat Word Burst AHB Limit
- Enhanced Programmable Mixed Arbitration for Each Slave
  - Round-Robin
  - Fixed Priority
  - Latency Quality of Service
- Programmable Default Master for Each Slave
  - No Default Master
  - Last Accessed Default Master
  - Fixed Default Master
- Deterministic Maximum Access Latency for Masters
- Zero or One Cycle Arbitration Latency for the First Access of a Burst
- Bus Lock Forwarding to Slaves
- Master Number Forwarding to Slaves
- One Special Function Register for Each Slave (Not dedicated)
- Register Write Protection Write Protection of User Interface Registers
- User/Privilege Protection

17.3 Memory Mapping

The Bus Matrix provides one decoder for every AHB master interface. The decoder offers each AHB master several memory mappings. Each memory area may be assigned to several slaves. Booting at the same address while using different AHB slaves (i.e., external RAM, internal Flash, etc.) becomes possible.

The Bus Matrix user interface provides the Master Remap Control Register (MATRIX_MRCR) that performs the remap action for every master independently.

17.4 Special Bus Granting Techniques

The MATRIX provides some speculative bus granting techniques in order to anticipate access requests from masters. Hence, latency is reduced at first access of a burst or, for a single transfer, as long as the slave is free from any other master access. It does not provide any benefit if the slave is continuously accessed by more than one master, since arbitration is pipelined and has no negative effect on the slave bandwidth or access latency.

This bus granting technique sets a different default master for every slave.

At the end of the current access, if no other request is pending, the slave remains connected to its associated default master. A slave can be associated with three kinds of default masters:

- no default master
- last access master
- fixed default master

To change from one type of default master to another, the user interface provides Slave Configuration registers (MATRIX_SCFGx), one for every slave, which set a default master for each slave. MATRIX_SCFGx contain two fields to manage master selection: DEFMSTR_TYPE and FIXED_DEFMSTR. The 2-bit DEFMSTR_TYPE field selects the default master type (no default, last access master, fixed default master), whereas the 4-bit FIXED_DEFMSTR field selects a fixed default master provided that DEFMSTR_TYPE is set to fixed default master. Refer to section 17.11.3 MATRIX_SCFG0.

17.5 No Default Master

After the end of the current access, if no other request is pending, the slave is disconnected from all masters.

This configuration incurs one latency clock cycle for the first access of a burst after bus Idle. Arbitration without default master may be used for masters that perform significant bursts or several transfers with no Idle cycle in-between, or if the slave bus bandwidth is widely used by one or more masters.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput whatever the number of requesting masters.

17.6 Last Access Master

After the end of the current access, if no other request is pending, the slave remains connected to the last master that performed an access request.

This enables the Bus Matrix to remove the one latency cycle for the last master that accessed the slave. Other non-privileged masters still get one latency clock cycle if they need to access the same slave. This technique is useful for masters that mainly perform single accesses or short bursts with some Idle cycles in-between.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput whatever is the number of requesting masters.

17.7 Fixed Default Master

After the end of the current access, if no other request is pending, the slave connects to its fixed default master. Unlike the last access master, the fixed default master does not change unless the user modifies it by software (FIXED_DEFMSTR field of the related MATRIX_SCFG).

This enables the Bus Matrix arbiters to remove the one latency clock cycle for the fixed default master of the slave. All requests attempted by the fixed default master do not cause any arbitration latency, whereas other non-privileged masters get one latency cycle. This technique is useful for a master that mainly performs single accesses or short bursts with Idle cycles in-between.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput, regardless of the number of requesting masters.

17.8 Arbitration

The Bus Matrix provides an arbitration technique that reduces latency when conflict cases occur, i.e., when two or more masters try to access the same slave at the same time. One arbiter per AHB slave is provided, thus arbitrating each slave specifically.

The user can either choose one of the following arbitration types, or mix them for each slave:

1. Round-robin Arbitration (default)
2. Fixed Priority Arbitration

The resulting algorithm may be complemented by selecting a default master configuration for each slave.

When re-arbitration must be done, specific conditions apply. See section 17.8.1 Arbitration Scheduling .

17.8.1 Arbitration Scheduling

Each arbiter has the ability to arbitrate between two or more different master requests. In order to avoid burst breaking as well as to provide the maximum throughput for slave interfaces, arbitration may only take place during the following cycles:

1. Idle Cycles: When a slave is not connected to any master or is connected to a master which is not currently accessing it.
2. Single Cycles: When a slave is currently doing a single access.
3. End of Burst Cycles: When the current cycle is the last cycle of a burst transfer. For defined length burst, predicted end of burst matches the size of the transfer but is managed differently for undefined length burst. See section 17.8.1.1 Undefined Length Burst Arbitration.
4. Slot Cycle Limit: When the slot cycle counter has reached the limit value indicating that the current master access is too long and must be broken. See section 17.8.1.2 Slot Cycle Limit Arbitration.

17.8.1.1 Undefined Length Burst Arbitration

In order to prevent long AHB burst lengths that can lock the access to the slave for an excessive period of time, the user can trigger the re-arbitration before the end of the incremental bursts. The re-arbitration period can be selected from the following Undefined Length Burst Type (ULBT) possibilities:

- Unlimited: no predetermined end of burst is generated. This value enables 1 Kbyte burst lengths.
- 1-beat bursts: predetermined end of burst is generated at each single transfer during the INCR transfer.
- 4-beat bursts: predetermined end of burst is generated at the end of each 4-beat boundary during INCR transfer.
- 8-beat bursts: predetermined end of burst is generated at the end of each 8-beat boundary during INCR transfer.
- 16-beat bursts: predetermined end of burst is generated at the end of each 16-beat boundary during INCR transfer.
- 32-beat bursts: predetermined end of burst is generated at the end of each 32-beat boundary during INCR transfer.
- 64-beat bursts: predetermined end of burst is generated at the end of each 64-beat boundary during INCR transfer.
- 128-beat bursts: predetermined end of burst is generated at the end of each 128-beat boundary during INCR transfer.

The use of undefined length 8-beat bursts, or less, is discouraged since this may decrease the overall bus bandwidth due to arbitration and slave latencies at each first access of a burst.

However, if the usual length of undefined length bursts is known for a master it is recommended to configure the ULBT accordingly.

This selection can be done through the ULBT field of the Master Configuration Registers (MATRIX_MCFG).

17.8.1.2 Slot Cycle Limit Arbitration

The Bus Matrix contains specific logic to break long accesses, such as very long bursts on a very slow slave (e.g., an external low speed memory). At each arbitration time, a counter is loaded with the value previously written in the SLOT_CYCLE field of the related Slave Configuration Register (MATRIX_SCFG) and decreased at each clock cycle. When the counter elapses, the arbiter has the ability to re-arbitrate at the end of the current AHB bus access cycle.

Unless a master has a very tight access latency constraint, which could lead to data overflow or underflow due to a badly undersized internal FIFO with respect to its throughput, the Slot Cycle Limit should be disabled (SLOT_CYCLE = 0) or set to its default maximum value in order not to inefficiently break long bursts performed by some Atmel masters.

In most cases, this feature is not needed and should be disabled for power saving.

⚠WARNING This feature cannot prevent any slave from locking its access indefinitely.

17.8.2 Arbitration Priority Scheme

The MATRIX arbitration scheme is organized in priority pools, each corresponding to an access criticality class as shown in the “Latency Quality of Service” column in the table below. When the Latency Quality of Service is enabled for a master-slave pair through the MATRIX, the priority pool number to use for arbitration at the slave port is determined from the master. When the Latency Quality of Service is disabled, it is determined through the MATRIX user interface. See 17.11.12 MATRIX_PRAS0.

After reset, the Latency Quality of Service is enabled by default on all of the master ports that are connected to a master driving the Latency Quality of Service signals, as shown in the bit LQOSEN of 17.11.12 MATRIX_PRAS0 and 17.11.13 MATRIX_PRBS0.

Table 17-4. Arbitration Priority Pools

Priority Pool	Latency Quality of Service
3	Latency Critical
2	Latency Sensitive
1	Bandwidth Sensitive
0	Background Transfers

Round-robin priority is used in the highest and lowest priority pools 3 and 0, whereas fixed level priority is used between priority pools and in the intermediate priority pools 2 and 1.

For each slave, each master is assigned to one of the slave priority pools based on the Latency Quality of Service inputs or to the priority registers for slaves (MxPR fields of MATRIX_PRAS and MATRIX_PRBS). When evaluating master requests, this priority pool level always takes precedence.

After reset, most of the masters belong to the lowest priority pool (MxPR = 0, Background Transfer) and are therefore granted bus access in a true round-robin order.

The highest priority pool must be specifically reserved for masters requiring very low access latency. If more than one master belongs to this pool, those masters are granted bus access in a biased round-robin manner which enables tight and deterministic maximum access latency from system bus requests. In the worst case, any currently occurring high-priority master request is granted after the current bus master access has ended and any other high priority pool master requests have been granted once each.

The lowest priority pool shares the remaining bus bandwidth between masters.

Intermediate priority pools enable fine priority tuning. Typically, a latency-sensitive master or a bandwidth-sensitive master use such a priority level. The higher the priority level (MxPR value), the higher the master priority.

For good CPU performance, it is recommended to let the CPU priority configured with the default reset value 2 (Latency Sensitive).

All combinations of MxPR values are allowed for all masters and slaves. For example, some masters might be assigned the highest priority pool (round-robin), and remaining masters the lowest priority pool (round-robin), with no master for intermediate fixed priority levels.

17.9 Register Write Protection

To prevent any single software error from corrupting MATRIX behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the Write Protection Mode register (MATRIX_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the Write Protection Status register (MATRIX_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS flag is reset by writing the MATRIX_WPMR with the appropriate access key WPKEY.

The registers listed below can be write-protected.

17.10 Privilege Protection of AHB and APB

User/Privilege protection is supported through the filtering of each slave access with the master User/Privilege protection bit.

The Protection Unit adds the ability to manage the access rights for Privilege and User accesses. The access rights are defined through the hardware and software configuration of the device. The operating mode is the following:

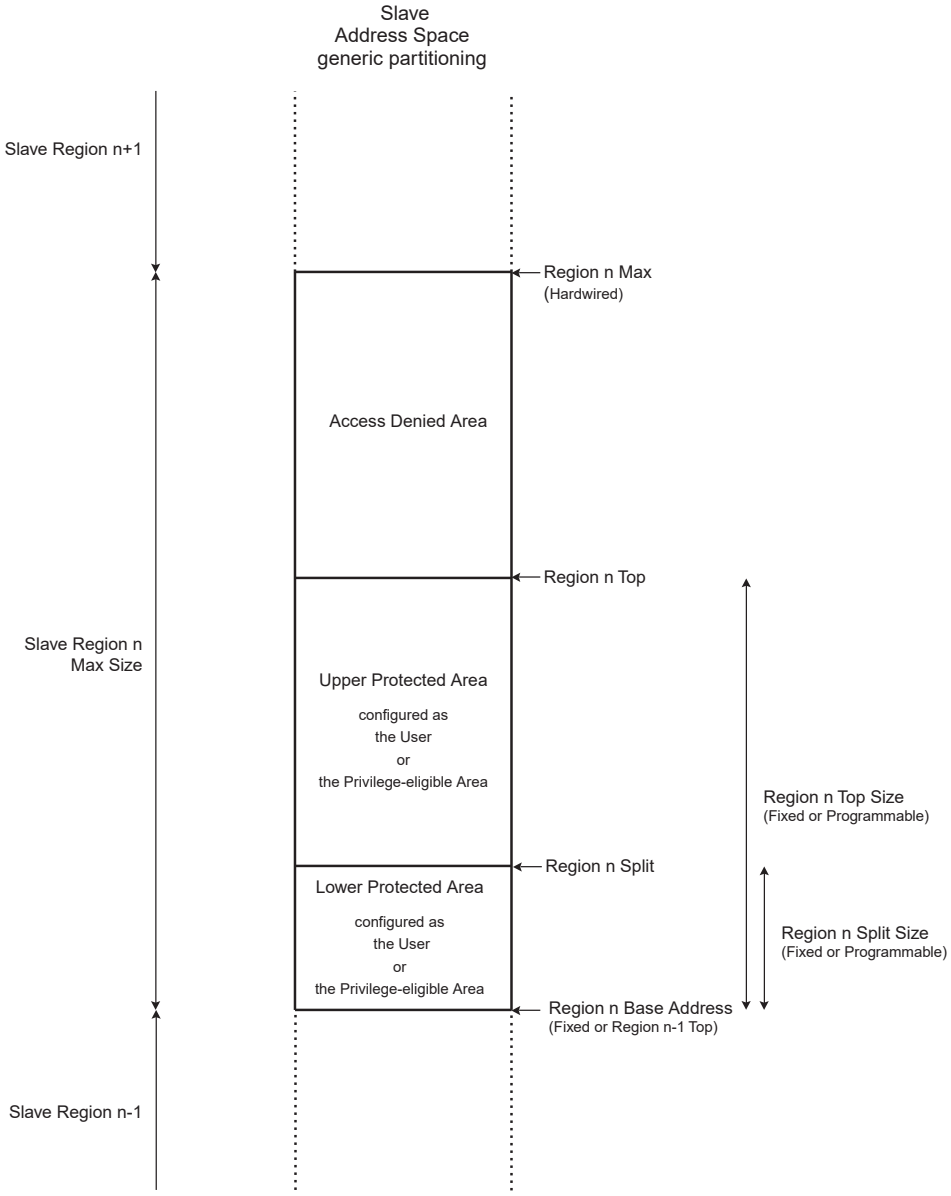
- The Bus Masters transmit requests with the Privilege or User access right.
- The MATRIX, according to its configuration and the request, grants or denies the access.

The slave address space is divided into one or more slave regions. The slave regions are generally contiguous parts of the slave address space. The slave region is potentially split into an access denied area (upper part) and a protected region which can be split (lower part), unless the slave-protected region occupies the whole slave region. The protected region itself may or may not be split into one Privilege-eligible area and one User area. The Privilege-eligible area may be independently Privileged for read access and for write access.

For one slave region, the following characteristics are configured by hardware or software:

- Base Address of the slave region
- The Max Size of the slave region—a maximum size for the region’s physical content
- The Top Size of the slave-protected region—the actually programmed or fixed size for the region’s physical content
- The Split Size of the slave-protected region—the size of one of the two protected areas of the region
- Whether Split Size is defined downward from the top of the protected region, or upward from its base address The following figure shows how the terms defined here are implemented in a Slave address space.

Figure 17-1. Generic Partitioning of the Slave Address Space



A set of MATRIX protection registers are used to specify, for each slave, slave-protected region or slave-protected area, the protection mode required for accessing this slave, slave-protected region or slave-protected area. See 17.11.37 MATRIX_PSR0, 17.11.44 MATRIX_PASSR0 and 17.11.53 MATRIX_PRTSRx.

Additional MATRIX protection registers are used to specify, for each peripheral bus slave, the protection mode required for accessing this slave (see 17.11.54 MATRIX_PPSELR0).

The MATRIX registers can only be accessed in Privileged mode.

The MATRIX propagates the protection bit down to the slaves to let them perform additional protection checks, and the MATRIX itself allows or not the access to the slaves via its embedded protection unit.

Access violations may be reported either by a slave through the bus error response (example from the system-to-peripheral bus (AHB/APB Bridge)), or by the MATRIX embedded protection unit. In both cases, a bus error response is sent to the offending master and the error is flagged in the Master Error Status Register. An interrupt can be sent if it has been enabled for that master by writing into the Master Error Interrupt Enable Register. Thus, the offending master is identified. The offending address is registered in the Master Error Address Registers, so that the slave and the targeted protected region are also known.

Depending on the hardware parameters and software configuration, the address space of each slave-protected region may or may not be split into two parts: one User and one Privileged.

Five different protection types of slaves are supported. The number of protected regions is set by design for each slave, independently, from 1 to 8, totaling from 1 up to 16 protected areas for each protection-configurable slave.

17.10.1 Protection Types of AHB Slaves

17.10.2 Principles

The MATRIX supports five different protection types of slaves: two fixed types and three configurable types. The protection type of an AHB slave is set at hardware design among the following:

- Always User
- Always Privileged
- Internal Protected
- External Protected
- Scalable Protected

The protection type is set at hardware design on a per-master and a per-slave basis. Always User and Always Privilege protection types are not software-configurable.

The different protection types have the following characteristics:

- Always User slaves have no access right restriction. Their address space is precisely set by design. Any out-of-address range access is denied and reported.
- Always Privilege slaves can only be accessed by a Privileged master request. Their address space is precisely set by design. Any User access or out-of-address range access is denied and reported.
- Internal Protected type is intended for internal memories such as RAM, ROM or embedded Flash. The Internal Protected slave has one or several slave regions, and each has a hardware-fixed base address and Protected Region Top. Each slave region may be split through software configuration into one User area plus one Privilege-eligible area. Inside each slave-protected region, the split boundary is programmable in powers of 2 from 4 Kbytes up to the full slave-protected region address space. The protected area located below the split boundary may be configured as the User or the Privilege-eligible one. The Privilege-eligible area may be independently configured as Read User and/or Write Privileged. Any access with access right or address range violation is denied and reported.
- External Protected type is intended for external memories on the EBI, such as DDR, SDRAM or external ROM. The External Protected slave has identical features as the Internal Protected slave, plus the ability to configure each of its slave-protected region address space sizes according to the external memory parts used. This avoids mirroring Privileged areas into User areas, and further restricts the overall accessible address range. Any access with access right violation or configured address range violation is denied and reported.
- Scalable Protected type is intended for external memories with a dedicated slave, such as DDR. The Scalable Protected slave is divided into a fixed number of scalable, equally sized, and contiguous protected regions. Each of them can be split in the same way as for Internal or External Protected slaves. The protected region size must be configured by software, so that the equally-sized regions fill the actual available memory. This avoids mirroring Privileged areas into User areas, and further restricts the overall accessible address range. Any access with access right violation or configured address range violation is denied and reported.

As the protection type is set at hardware design on a per-master and per-slave basis, it is possible to set some slave access protection as configurable from one or some particular masters, and to set the access as Always Privileged from all the other masters.

As the protection type is set by design at the slave region level, different protected region types can be mixed inside a single slave.

Likewise, the mapping base address and the accessible address range of each slave or slave region may have been hardware-restricted on a per-master basis from no access to full slave address space.

Table 17-5. Slave Protection Types

Slave	IP	#hsel	Mode	Max Size
0	FlexRAM	0 @ 0x21000000	Internal Protected	768 KB
1	FlexRAM	0 @ 0x21000000	Internal Protected	768 KB
7	FlexRAM	0 @ 0x21000000	Internal Protected	768 KB
2	Internal Flash (HEFC) from CPU	0 @ 0x10000000	Always User	128 KB
2	Internal Flash (HEFC) from others	0 @ 0x10000000	External Protected	128 KB
3	HEMC (HEMC) (1)	0 @ 0x60000000 1 @ 0x70000000 2 @ 0x80000000 3 @ 0x90000000 4 @ 0xA0000000 5 @ 0xB0000000 6 @ 0xC0000000 7 @ 0xD0000000	Always User Always User Always User Always User Always User Always User Always User Always User	256 MB each hsel with total 2048 MB
4	QSPI	0 @ 0x18000000	Always User	128 MB
5	APB Bridge	0 @ 0x40000000	Always User	2 MB (no mirroring)
6	AHB Slave	0 @ 0x00000000 1 @ 0x20000000 2 @ 0x20020000	Internal Protected Internal Protected Internal Protected	128 KB 128 KB 128 KB

Note:

1. The HEMC embeds its own protection unit. Refer to the section "Hardened External Memory Controller (HEMC)" for details.

17.10.3 Examples

Note: The following examples are generic and and not product-specific. They illustrate in a general manner the operation of the protection functionality.

The table below shows an example of Protection Type settings. This example is constructed with the following characteristics:

- Slave0 is an Internal Memory containing 1 region:
  - The Access from Master0 to Slave0 is Always User
  - The Access from Master1 and Master2 to Slave0 is Internal Protected with 1 region and with the same Software Configuration (Choice of SPLIT0 and the Protection Configuration bits LAUSERH, RDUSERH, WRUSERH).
- Slave1 is an EBI containing 2 regions:
  - The Access from Master1 to Slave1 is Always User
  - The Access from Master0 and Master2 to Slave1 is External Protected with 2 regions and with the same Software Configuration (Choice of TOP0, TOP1, SPLIT0, SPLIT1 and the Protection Configuration bits LAUSERH, RDUSERH, WRUSERH).

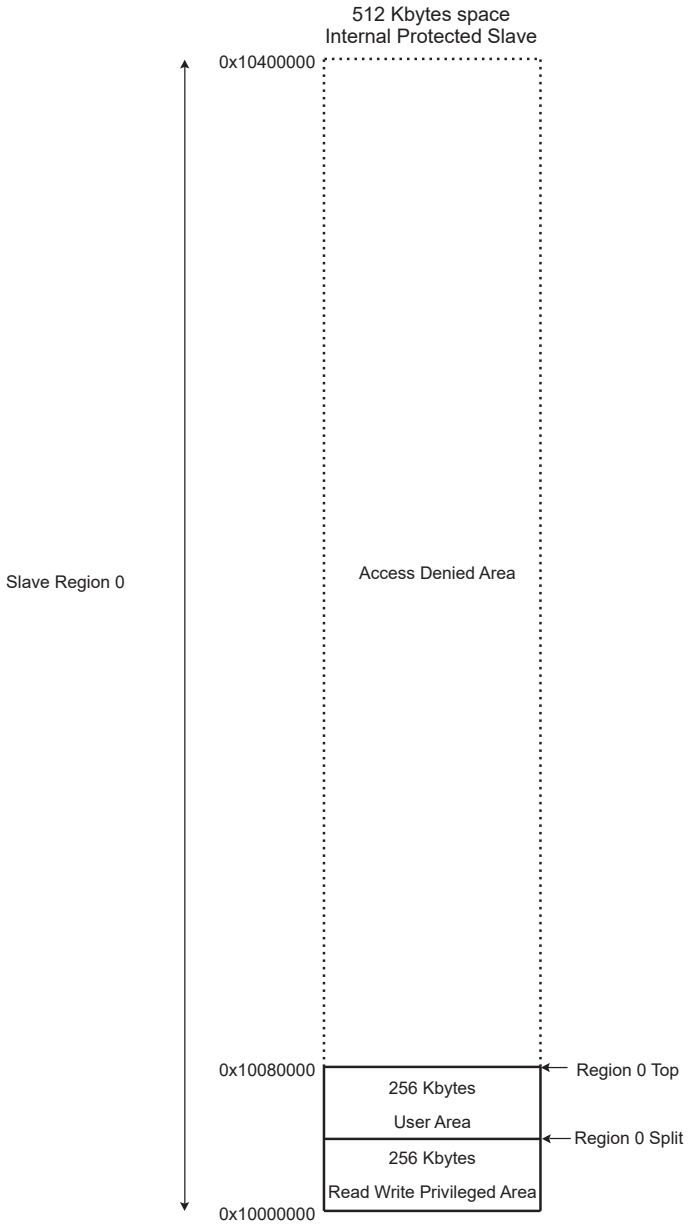
Table 17-6. Protection Type Setting Example

Slave	Master0	Master1	Master2
Slave0 Internal Memory	Always User	Internal Protected 1 Region	Internal Protected 1 Region
Slave1 EBI	External Protected 2 Regions	Always Privileged	External Protected 2 Regions

The following figure shows an Internal Protected slave example. This example is based on the following hypothesis:

- The slave is an Internal Memory containing 1 region. The Slave region Max Size is 4 Mbytes.
- The slave region 0 base address equals 0x10 000 000. Its Top Size is 512 Kbytes (hardware configuration).
- The slave software configuration is as follows:
  - SPLIT0 is set to 256 Kbytes.
  - LAUSERH0 is set to 0, the low area of region 0 is the Privilege-eligible one.
  - RDUSERH0 is set to 0, region 0 Privilege-eligible area is Privileged for reads.
  - WRUSERH0 is set to 0, region 0 Privilege-eligible area is Privileged for writes.

Figure 17-2. Partitioning Example of an Internal Protected Slave Featuring 1 Protected Region of 512 KB Split into 1 or 2 Protected Areas of 4 KB to 512 KB



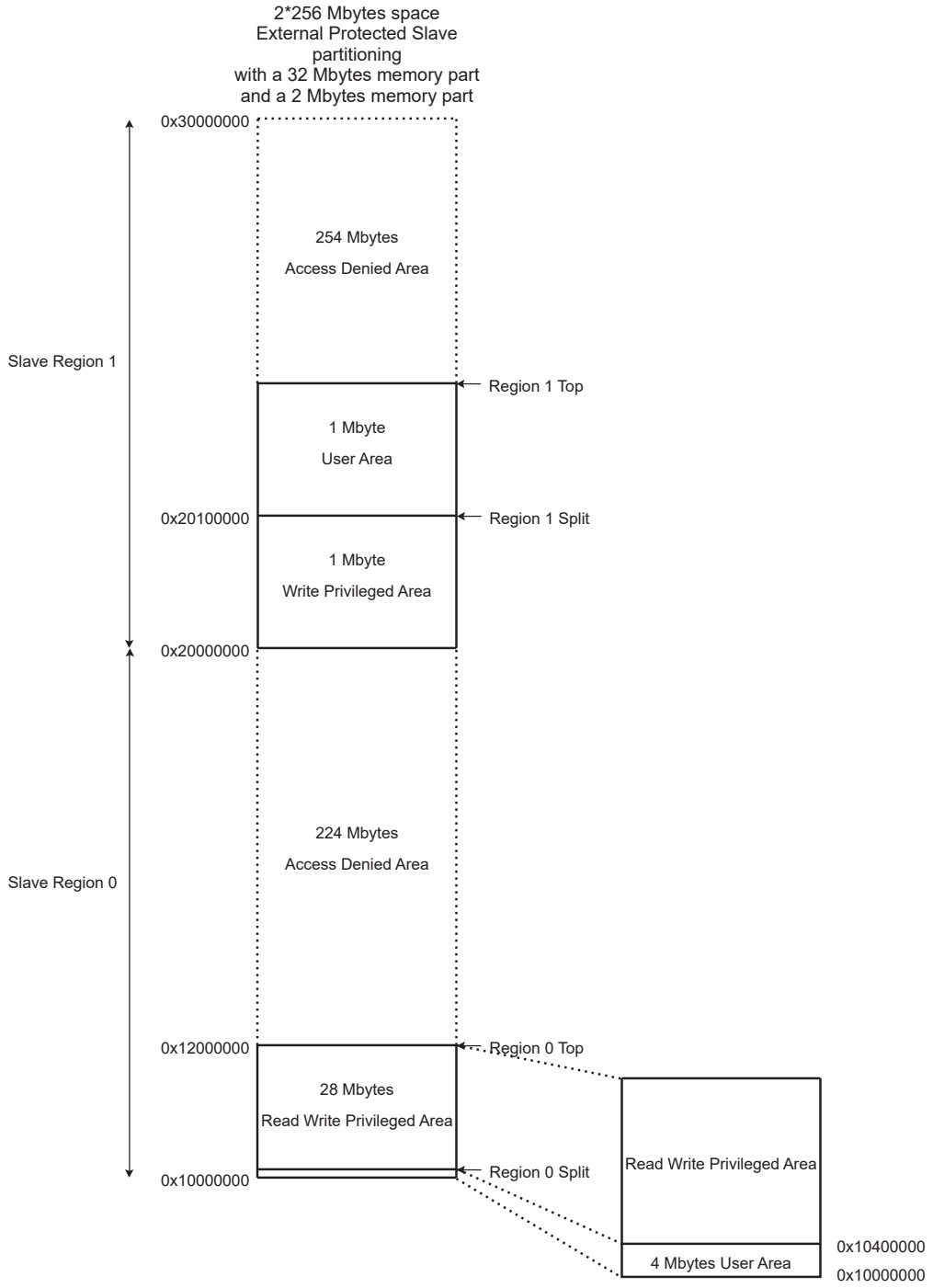
Note: The slave-protected areas split inside the protected region are configured by writing into the Protected Areas Split Slave Registers.

The following figure shows an External Protected slave example. This example is based on the following hypothesis:

- The slave is an EBI containing 2 regions. The slave size is 2*256 Mbytes. Each slave region Max Size is 256 Mbytes.
- The slave region 0 base address equals 0x10 000 000. It is connected to a 32 Mbytes memory, for example an external DDR. The slave region 0 Top Size must be set to 32 Mbytes.
- The slave region 1 base address equals 0x20 000 000. It is connected to a 2 Mbytes memory, for example an external Flash. The slave region 1 Top Size must be set to 2 Mbytes.
- The slave software configuration is as follows:
  - TOP0 is set to 32 Mbytes.
  - TOP1 is set to 2 Mbytes.

- SPLIT0 is set to 4 Mbytes.
- SPLIT1 is set to 1 Mbyte.
- LAUSERH0 is set to 1, the low area of region 0 is the User one.
- RDUSERH0 is set to 0, region 0 Privilege-eligible area is Privileged for reads.
- WRUSERH0 is set to 0, region 0 Privilege-eligible area is Privileged for writes.
- LAUSERH1 is set to 0, the low area of region 1 is the Privilege-eligible one.
- RDUSERH1 is set to 1, region 1 Privilege-eligible area is User for reads.
- WRUSERH1 is set to 0, region 1 Privilege-eligible area is Privileged for writes.

Figure 17-3. Partitioning Example: External Protected Slave Featuring 2 Protected Regions of 4 KB to 128 MB Each and Up to 4 Protected Areas of 4 KB to 128 MB



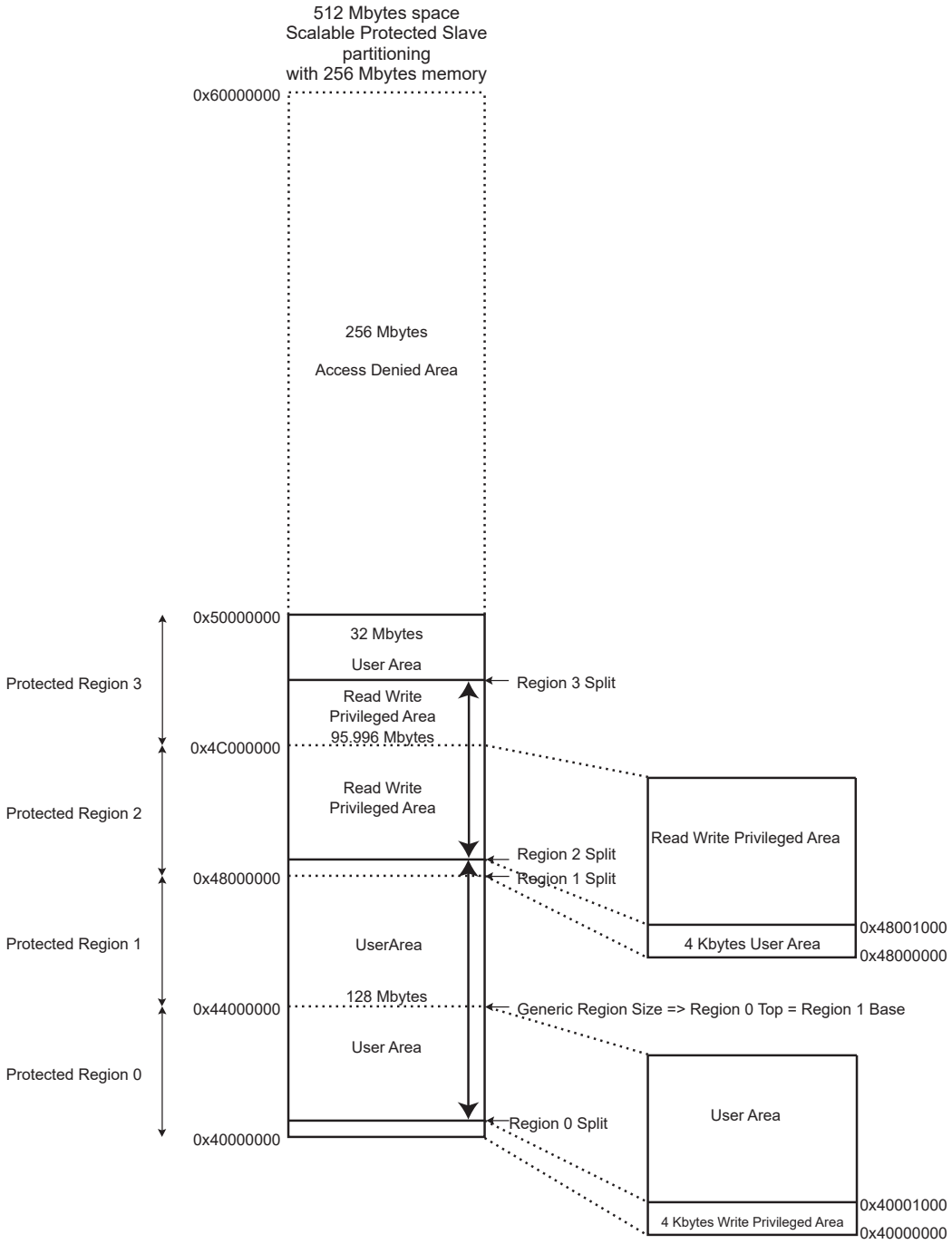
Note: The slave region sizes are configured by writing into the Protected Region Top Slave Registers. The slave-protected area split inside each region is configured by writing into the Protected Areas Split Slave Registers..

The following figure shows a Scalable Protected slave example. This example is based on the following hypothesis:

- The slave is an external memory with dedicated slave containing 4 regions, for example an external DDR.
- The slave size is 512 Mbytes.
- The slave base address equals 0x40 000 000. It is connected to a 256 Mbytes external memory.

- As the connected memory size is 256 Mbytes and there are 4 regions, the size of each region is 64 Mbytes. This gives the value of the slave region Max Size and Top Size. The slave region 0 Top Size must be configured to 64 Mbytes.
- The slave software configuration is as follows:
  - TOP0 is set to 64 Mbytes.
  - SPLIT0 is set to 4 Kbytes.
  - SPLIT1 is set to 64 Mbytes, so its low area occupies the whole region 1.
  - SPLIT2 is set to 4 Kbytes.
  - SPLIT3 is set to 32 Mbytes.
  - LAUSERH0 is set to 0, the low area of region 0 is the Privilege-eligible one.
  - RDUSERH0 is set to 1, region 0 Privilege-eligible area is User for reads.
  - WRUSERH0 is set to 0, region 0 Privilege-eligible area is Privileged for writes.
  - LAUSERH1 is set to 1, the low area of region 1 is the User one.
  - RDUSERH1 is ‘don’t care’ since the low area occupies the whole region 1.
  - WRUSERH1 is ‘don’t care’ since the low area occupies the whole region 1.
  - LAUSERH2 is set to 1, the low area of region 2 is the User one.
  - RDUSERH2 is set to 0, region 2 Privilege-eligible area is Privileged for reads.
  - WRUSERH2 is set to 0, region 2 Privilege-eligible area is Privileged for writes.
  - LAUSERH3 is set to 0, the low area of region 3 is the Privilege-eligible one.
  - RDUSERH3 is set to 0, region 3 Privilege-eligible area is Privileged for reads.
  - WRUSERH3 is set to 0, region 3 Privilege-eligible area is Privileged for writes.

Figure 17-4. Partitioning Example: Scalable Protected Slave Featuring 4 Equally-sized Protected Regions of 1 MB to 128 MB Each and Up to 8 Protected Areas of 4 KB to 128 MB



Note: The slaves' generic protected regions sizes are configured by writing into field PRTOP0 of the Protected Region Top Slave Registers and the custom slave-protected areas splits inside each region are configured by writing into the Protected Areas Split Slave Registers.

17.11 Register Summary

Offset	Name	Bit Pos.								
0x00	MATRIX_MCFG0	7:0						ULBT[2:0]		
		15:8								
		23:16								
		31:24								
...										
0x30	MATRIX_MCFG12	7:0						ULBT[2:0]		
		15:8								
		23:16								
		31:24								
0x34	MATRIX_MCFG13	7:0						ULBT[2:0]		
		15:8								
		23:16								
		31:24								
0x38	MATRIX_MCFG14	7:0						ULBT[2:0]		
		15:8								
		23:16								
		31:24								
0x3C	MATRIX_MCFG15	7:0						ULBT[2:0]		
		15:8								
		23:16								
		31:24								
0x40	MATRIX_SCFG0	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x44	MATRIX_SCFG1	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x48	MATRIX_SCFG2	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x4C	MATRIX_SCFG3	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x50	MATRIX_SCFG4	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x54	MATRIX_SCFG5	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x58	MATRIX_SCFG6	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x5C	MATRIX_SCFG7	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x60	MATRIX_SCFG8	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x64	MATRIX_SCFG9	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x68	MATRIX_SCFG10	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x6C	MATRIX_SCFG11	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x70	MATRIX_SCFG12	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x74	MATRIX_SCFG13	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x78	MATRIX_SCFG14	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x7C	MATRIX_SCFG15	7:0	SLOT_CYCLE[7:0]							
		15:8								SLOT_CYCLE[8]
		23:16			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
		31:24								
0x80	MATRIX_PRAS0	7:0		LQOSE1	M1PR[1:0]		LQOSE0		M0PR[1:0]	
		15:8		LQOSE3	M3PR[1:0]		LQOSE2		M2PR[1:0]	
		23:16		LQOSE5	M5PR[1:0]		LQOSE4		M4PR[1:0]	
		31:24		LQOSE7	M7PR[1:0]		LQOSE6		M6PR[1:0]	
0x84	MATRIX_PRBS0	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8		M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10		M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12		M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14		M14PR[1:0]	
0x88	MATRIX_PRAS1	7:0		LQOSE1	M1PR[1:0]		LQOSE0		M0PR[1:0]	
		15:8		LQOSE3	M3PR[1:0]		LQOSE2		M2PR[1:0]	
		23:16		LQOSE5	M5PR[1:0]		LQOSE4		M4PR[1:0]	
		31:24		LQOSE7	M7PR[1:0]		LQOSE6		M6PR[1:0]	

.....continued

Offset	Name	Bit Pos.							
0x8C	MATRIX_PRBS1	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0x90	MATRIX_PRAS2	7:0		LQOSE1	M1PR[1:0]		LQOSE0	M0PR[1:0]	
		15:8		LQOSE3	M3PR[1:0]		LQOSE2	M2PR[1:0]	
		23:16		LQOSE5	M5PR[1:0]		LQOSE4	M4PR[1:0]	
		31:24		LQOSE7	M7PR[1:0]		LQOSE6	M6PR[1:0]	
0x94	MATRIX_PRBS2	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0x98	MATRIX_PRAS3	7:0		LQOSE1	M1PR[1:0]		LQOSE0	M0PR[1:0]	
		15:8		LQOSE3	M3PR[1:0]		LQOSE2	M2PR[1:0]	
		23:16		LQOSE5	M5PR[1:0]		LQOSE4	M4PR[1:0]	
		31:24		LQOSE7	M7PR[1:0]		LQOSE6	M6PR[1:0]	
0x9C	MATRIX_PRBS3	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xA0	MATRIX_PRAS4	7:0		LQOSE1	M1PR[1:0]		LQOSE0	M0PR[1:0]	
		15:8		LQOSE3	M3PR[1:0]		LQOSE2	M2PR[1:0]	
		23:16		LQOSE5	M5PR[1:0]		LQOSE4	M4PR[1:0]	
		31:24		LQOSE7	M7PR[1:0]		LQOSE6	M6PR[1:0]	
0xA4	MATRIX_PRBS4	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xA8	MATRIX_PRAS5	7:0		LQOSE1	M1PR[1:0]		LQOSE0	M0PR[1:0]	
		15:8		LQOSE3	M3PR[1:0]		LQOSE2	M2PR[1:0]	
		23:16		LQOSE5	M5PR[1:0]		LQOSE4	M4PR[1:0]	
		31:24		LQOSE7	M7PR[1:0]		LQOSE6	M6PR[1:0]	
0xAC	MATRIX_PRBS5	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xB0	MATRIX_PRAS6	7:0		LQOSE1	M1PR[1:0]		LQOSE0	M0PR[1:0]	
		15:8		LQOSE3	M3PR[1:0]		LQOSE2	M2PR[1:0]	
		23:16		LQOSE5	M5PR[1:0]		LQOSE4	M4PR[1:0]	
		31:24		LQOSE7	M7PR[1:0]		LQOSE6	M6PR[1:0]	
0xB4	MATRIX_PRBS6	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xB8	MATRIX_PRAS7	7:0		LQOSEN1	M1PR[1:0]		LQOSEN0	M0PR[1:0]	
		15:8		LQOSEN3	M3PR[1:0]		LQOSEN2	M2PR[1:0]	
		23:16		LQOSEN5	M5PR[1:0]		LQOSEN4	M4PR[1:0]	
		31:24		LQOSEN7	M7PR[1:0]		LQOSEN6	M6PR[1:0]	
0xBC	MATRIX_PRBS7	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xC0	MATRIX_PRAS8	7:0		LQOSEN1	M1PR[1:0]		LQOSEN0	M0PR[1:0]	
		15:8		LQOSEN3	M3PR[1:0]		LQOSEN2	M2PR[1:0]	
		23:16		LQOSEN5	M5PR[1:0]		LQOSEN4	M4PR[1:0]	
		31:24		LQOSEN7	M7PR[1:0]		LQOSEN6	M6PR[1:0]	

.....continued

Offset	Name	Bit Pos.							
0xC4	MATRIX_PRBS8	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xC8	MATRIX_PRAS9	7:0		LQOSEN1	M1PR[1:0]		LQOSEN0	M0PR[1:0]	
		15:8		LQOSEN3	M3PR[1:0]		LQOSEN2	M2PR[1:0]	
		23:16		LQOSEN5	M5PR[1:0]		LQOSEN4	M4PR[1:0]	
		31:24		LQOSEN7	M7PR[1:0]		LQOSEN6	M6PR[1:0]	
0xCC	MATRIX_PRBS9	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xD0	MATRIX_PRAS10	7:0		LQOSEN1	M1PR[1:0]		LQOSEN0	M0PR[1:0]	
		15:8		LQOSEN3	M3PR[1:0]		LQOSEN2	M2PR[1:0]	
		23:16		LQOSEN5	M5PR[1:0]		LQOSEN4	M4PR[1:0]	
		31:24		LQOSEN7	M7PR[1:0]		LQOSEN6	M6PR[1:0]	
0xD4	MATRIX_PRBS10	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xD8	MATRIX_PRAS11	7:0		LQOSEN1	M1PR[1:0]		LQOSEN0	M0PR[1:0]	
		15:8		LQOSEN3	M3PR[1:0]		LQOSEN2	M2PR[1:0]	
		23:16		LQOSEN5	M5PR[1:0]		LQOSEN4	M4PR[1:0]	
		31:24		LQOSEN7	M7PR[1:0]		LQOSEN6	M6PR[1:0]	
0xDC	MATRIX_PRBS11	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xE0	MATRIX_PRAS12	7:0		LQOSEN1	M1PR[1:0]		LQOSEN0	M0PR[1:0]	
		15:8		LQOSEN3	M3PR[1:0]		LQOSEN2	M2PR[1:0]	
		23:16		LQOSEN5	M5PR[1:0]		LQOSEN4	M4PR[1:0]	
		31:24		LQOSEN7	M7PR[1:0]		LQOSEN6	M6PR[1:0]	
0xE4	MATRIX_PRBS12	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xE8	MATRIX_PRAS13	7:0		LQOSEN1	M1PR[1:0]		LQOSEN0	M0PR[1:0]	
		15:8		LQOSEN3	M3PR[1:0]		LQOSEN2	M2PR[1:0]	
		23:16		LQOSEN5	M5PR[1:0]		LQOSEN4	M4PR[1:0]	
		31:24		LQOSEN7	M7PR[1:0]		LQOSEN6	M6PR[1:0]	
0xEC	MATRIX_PRBS13	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xF0	MATRIX_PRAS14	7:0		LQOSEN1	M1PR[1:0]		LQOSEN0	M0PR[1:0]	
		15:8		LQOSEN3	M3PR[1:0]		LQOSEN2	M2PR[1:0]	
		23:16		LQOSEN5	M5PR[1:0]		LQOSEN4	M4PR[1:0]	
		31:24		LQOSEN7	M7PR[1:0]		LQOSEN6	M6PR[1:0]	
0xF4	MATRIX_PRBS14	7:0		LQOSEN9	M9PR[1:0]		LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]		LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]		LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]		LQOSEN14	M14PR[1:0]	
0xF8	MATRIX_PRAS15	7:0		LQOSEN1	M1PR[1:0]		LQOSEN0	M0PR[1:0]	
		15:8		LQOSEN3	M3PR[1:0]		LQOSEN2	M2PR[1:0]	
		23:16		LQOSEN5	M5PR[1:0]		LQOSEN4	M4PR[1:0]	
		31:24		LQOSEN7	M7PR[1:0]		LQOSEN6	M6PR[1:0]	

.....continued

Offset	Name	Bit Pos.								
0xFC	MATRIX_PRBS15	7:0		LQOSEN9	M9PR[1:0]			LQOSEN8	M8PR[1:0]	
		15:8		LQOSEN11	M11PR[1:0]			LQOSEN10	M10PR[1:0]	
		23:16		LQOSEN13	M13PR[1:0]			LQOSEN12	M12PR[1:0]	
		31:24		LQOSEN15	M15PR[1:0]			LQOSEN14	M14PR[1:0]	
0x0100	MATRIX_MRCR	7:0	RCB7	RCB6	RCB5	RCB4	RCB3	RCB2	RCB1	RCB0
		15:8	RCB15	RCB14	RCB13	RCB12	RCB11	RCB10	RCB9	RCB8
		23:16								
		31:24								
0x0104 ... 0x010F	Reserved									
0x0110	MATRIX_SFR0	7:0	SFR[7:0]							
		15:8	SFR[15:8]							
		23:16	SFR[23:16]							
		31:24	SFR[31:24]							
...										
0x014C	MATRIX_SFR15	7:0	SFR[7:0]							
		15:8	SFR[15:8]							
		23:16	SFR[23:16]							
		31:24	SFR[31:24]							
0x0150	MATRIX_MEIER	7:0	MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0
		15:8	MERR15	MERR14	MERR13	MERR12	MERR11	MERR10	MERR9	MERR8
		23:16								
		31:24								
0x0154	MATRIX_MEIDR	7:0	MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0
		15:8	MERR15	MERR14	MERR13	MERR12	MERR11	MERR10	MERR9	MERR8
		23:16								
		31:24								
0x0158	MATRIX_MEIMR	7:0	MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0
		15:8	MERR15	MERR14	MERR13	MERR12	MERR11	MERR10	MERR9	MERR8
		23:16								
		31:24								
0x015C	MATRIX_MESR	7:0	MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0
		15:8	MERR15	MERR14	MERR13	MERR12	MERR11	MERR10	MERR9	MERR8
		23:16								
		31:24								
0x0160	MATRIX_MEAR0	7:0	ERRADD[7:0]							
		15:8	ERRADD[15:8]							
		23:16	ERRADD[23:16]							
		31:24	ERRADD[31:24]							
...										
0x019C	MATRIX_MEAR15	7:0	ERRADD[7:0]							
		15:8	ERRADD[15:8]							
		23:16	ERRADD[23:16]							
		31:24	ERRADD[31:24]							
0x01A0 ... 0x01E3	Reserved									
0x01E4	MATRIX_WPMR	7:0	CFGFRZ							WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0x01E8	MATRIX_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16	WPVSRC[15:8]							
		31:24								
0x01EC ... 0x01FF	Reserved									

.....continued

Offset	Name	Bit Pos.								
0x0200	MATRIX_PSR0	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0204	MATRIX_PSR1	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0208	MATRIX_PSR2	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x020C	MATRIX_PSR3	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0210	MATRIX_PSR4	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0214	MATRIX_PSR5	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0218	MATRIX_PSR6	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x021C	MATRIX_PSR7	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0220	MATRIX_PSR8	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0224	MATRIX_PSR9	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0228	MATRIX_PSR10	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x022C	MATRIX_PSR11	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0230	MATRIX_PSR12	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0234	MATRIX_PSR13	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0

.....continued

Offset	Name	Bit Pos.								
0x0238	MATRIX_PSR14	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x023C	MATRIX_PSR15	7:0	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
		15:8	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
		23:16	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
		31:24	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
0x0240	MATRIX_PASSR0	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0244	MATRIX_PASSR1	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0248	MATRIX_PASSR2	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x024C	MATRIX_PASSR3	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0250	MATRIX_PASSR4	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0254	MATRIX_PASSR5	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0258	MATRIX_PASSR6	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x025C	MATRIX_PASSR7	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0260	MATRIX_PASSR8	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0264	MATRIX_PASSR9	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0268	MATRIX_PASSR10	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x026C	MATRIX_PASSR11	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			

.....continued

Offset	Name	Bit Pos.								
0x0270	MATRIX_PASSR12	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0274	MATRIX_PASSR13	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0278	MATRIX_PASSR14	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x027C	MATRIX_PASSR15	7:0	PASPLIT1[3:0]				PASPLIT0[3:0]			
		15:8	PASPLIT3[3:0]				PASPLIT2[3:0]			
		23:16	PASPLIT5[3:0]				PASPLIT4[3:0]			
		31:24	PASPLIT7[3:0]				PASPLIT6[3:0]			
0x0280	MATRIX_PRTSR0	7:0	PRTOP1[3:0]				PRTOP0[3:0]			
		15:8	PRTOP3[3:0]				PRTOP2[3:0]			
		23:16	PRTOP5[3:0]				PRTOP4[3:0]			
		31:24	PRTOP7[3:0]				PRTOP6[3:0]			
...										
0x02BC	MATRIX_PRTSR15	7:0	PRTOP1[3:0]				PRTOP0[3:0]			
		15:8	PRTOP3[3:0]				PRTOP2[3:0]			
		23:16	PRTOP5[3:0]				PRTOP4[3:0]			
		31:24	PRTOP7[3:0]				PRTOP6[3:0]			
0x02C0	MATRIX_PPSELR0	7:0	USERP7	USERP6	USERP5	USERP4	USERP3	USERP2	USERP1	USERP0
		15:8	USERP15	USERP14	USERP13	USERP12	USERP11	USERP10	USERP9	USERP8
		23:16	USERP23	USERP22	USERP21	USERP20	USERP19	USERP18	USERP17	USERP16
		31:24	USERP31	USERP30	USERP29	USERP28	USERP27	USERP26	USERP25	USERP24
0x02C4	MATRIX_PPSELR1	7:0	USERP7	USERP6	USERP5	USERP4	USERP3	USERP2	USERP1	USERP0
		15:8	USERP15	USERP14	USERP13	USERP12	USERP11	USERP10	USERP9	USERP8
		23:16	USERP23	USERP22	USERP21	USERP20	USERP19	USERP18	USERP17	USERP16
		31:24	USERP31	USERP30	USERP29	USERP28	USERP27	USERP26	USERP25	USERP24
0x02C8	MATRIX_PPSELR2	7:0	USERP7	USERP6	USERP5	USERP4	USERP3	USERP2	USERP1	USERP0
		15:8	USERP15	USERP14	USERP13	USERP12	USERP11	USERP10	USERP9	USERP8
		23:16	USERP23	USERP22	USERP21	USERP20	USERP19	USERP18	USERP17	USERP16
		31:24	USERP31	USERP30	USERP29	USERP28	USERP27	USERP26	USERP25	USERP24

17.11.1 Bus Matrix Master Configuration Register x [x=0...12]

Name: MATRIX_MCFGx
Offset: 0x00 + x*0x04 [x=0..12]
Reset: 0x00000004
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						ULBT[2:0]		
Access						R/W	R/W	R/W
Reset						1	0	0

Bits 2:0 – ULBT[2:0] Undefined Length Burst Type

Value	Name	Description
0	UNLIMITED	Unlimited Length Burst—No predicted end of burst is generated, therefore INCR bursts coming from this master can only be broken if the Slave Slot Cycle Limit is reached. If the Slot Cycle Limit is not reached, the burst is normally completed by the master, at the latest, on the next AHB 1 Kbyte address boundary, allowing up to 256-beat word bursts or 128-beat double-word bursts. This value should not be used in the very particular case of a master capable of performing back-to-back undefined length bursts on a single slave, since this could indefinitely freeze the slave arbitration and thus prevent another master from accessing this slave.
1	SINGLE	Single Access—The undefined length burst is treated as a succession of single accesses, allowing re-arbitration at each beat of the INCR burst or bursts sequence.
2	4_BEAT	4-beat Burst—The undefined length burst or bursts sequence is split into 4-beat bursts or less, allowing re-arbitration every 4 beats.
3	8_BEAT	8-beat Burst—The undefined length burst or bursts sequence is split into 8-beat bursts or less, allowing re-arbitration every 8 beats.
4	16_BEAT	16-beat Burst—The undefined length burst or bursts sequence is split into 16-beat bursts or less, allowing re-arbitration every 16 beats.
5	32_BEAT	32-beat Burst—The undefined length burst or bursts sequence is split into 32-beat bursts or less, allowing re-arbitration every 32 beats.
6	64_BEAT	64-beat Burst—The undefined length burst or bursts sequence is split into 64-beat bursts or less, allowing re-arbitration every 64 beats.
7	128_BEAT	128-beat Burst—The undefined length burst or bursts sequence is split into 128-beat bursts or less, allowing re-arbitration every 128 beats. Unless duly needed, the ULBT should be left at its default 0 value for power saving.

17.11.2 Bus Matrix Master Configuration Register x [x=13..15]

Name: MATRIX_MCFGx
Offset: 0x34 + (x-13)*0x04 [x=13..15]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						ULBT[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 2:0 – ULBT[2:0] Undefined Length Burst Type

Value	Name	Description
0	UNLIMITED	Unlimited Length Burst—No predicted end of burst is generated, therefore INCR bursts coming from this master can only be broken if the Slave Slot Cycle Limit is reached. If the Slot Cycle Limit is not reached, the burst is normally completed by the master, at the latest, on the next AHB 1 Kbyte address boundary, allowing up to 256-beat word bursts or 128-beat double-word bursts. This value should not be used in the very particular case of a master capable of performing back-to-back undefined length bursts on a single slave, since this could indefinitely freeze the slave arbitration and thus prevent another master from accessing this slave.
1	SINGLE	Single Access—The undefined length burst is treated as a succession of single accesses, allowing re-arbitration at each beat of the INCR burst or bursts sequence.
2	4_BEAT	4-beat Burst—The undefined length burst or bursts sequence is split into 4-beat bursts or less, allowing re-arbitration every 4 beats.
3	8_BEAT	8-beat Burst—The undefined length burst or bursts sequence is split into 8-beat bursts or less, allowing re-arbitration every 8 beats.
4	16_BEAT	16-beat Burst—The undefined length burst or bursts sequence is split into 16-beat bursts or less, allowing re-arbitration every 16 beats.
5	32_BEAT	32-beat Burst—The undefined length burst or bursts sequence is split into 32-beat bursts or less, allowing re-arbitration every 32 beats.
6	64_BEAT	64-beat Burst—The undefined length burst or bursts sequence is split into 64-beat bursts or less, allowing re-arbitration every 64 beats.
7	128_BEAT	128-beat Burst—The undefined length burst or bursts sequence is split into 128-beat bursts or less, allowing re-arbitration every 128 beats. Unless duly needed, the ULBT should be left at its default 0 value for power saving.

17.11.3 Bus Matrix Slave Configuration Register 0

Name: MATRIX_SCFG0
Offset: 0x40
Reset: 0x000101FF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
								SLOT_CYCLE[8]
Access								R/W
Reset								1

Bit	7	6	5	4	3	2	1	0
	SLOT_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 21:18 – FIXED_DEFMSTR[3:0] Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Bits 17:16 – DEFMSTR_TYPE[1:0] Default Master Type

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

Bits 8:0 – SLOT_CYCLE[8:0] Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See section Slot Cycle Limit Arbitration for details.

17.11.4 Bus Matrix Slave Configuration Register 1

Name: MATRIX_SCFG1
Offset: 0x44
Reset: 0x000101FF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
								SLOT_CYCLE[8]
Access								R/W
Reset								1

Bit	7	6	5	4	3	2	1	0
	SLOT_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 21:18 – FIXED_DEFMSTR[3:0] Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Bits 17:16 – DEFMSTR_TYPE[1:0] Default Master Type

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

Bits 8:0 – SLOT_CYCLE[8:0] Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See section Slot Cycle Limit Arbitration for details.

17.11.5 Bus Matrix Slave Configuration Register 2

Name: MATRIX_SCFG2
Offset: 0x48
Reset: 0x000201FF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	1	0

Bit	15	14	13	12	11	10	9	8
								SLOT_CYCLE[8]
Access								R/W
Reset								1

Bit	7	6	5	4	3	2	1	0
	SLOT_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 21:18 – FIXED_DEFMSTR[3:0] Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Bits 17:16 – DEFMSTR_TYPE[1:0] Default Master Type

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

Bits 8:0 – SLOT_CYCLE[8:0] Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See section Slot Cycle Limit Arbitration for details.

17.11.6 Bus Matrix Slave Configuration Register 3

Name: MATRIX_SCFG3
Offset: 0x4C
Reset: 0x000101FF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
								SLOT_CYCLE[8]
Access								R/W
Reset								1

Bit	7	6	5	4	3	2	1	0
	SLOT_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 21:18 – FIXED_DEFMSTR[3:0] Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Bits 17:16 – DEFMSTR_TYPE[1:0] Default Master Type

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

Bits 8:0 – SLOT_CYCLE[8:0] Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See section Slot Cycle Limit Arbitration for details.

17.11.7 Bus Matrix Slave Configuration Register 4

Name: MATRIX_SCFG4
Offset: 0x50
Reset: 0x000101FF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
								SLOT_CYCLE[8]
Access								R/W
Reset								1

Bit	7	6	5	4	3	2	1	0
	SLOT_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 21:18 – FIXED_DEFMSTR[3:0] Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Bits 17:16 – DEFMSTR_TYPE[1:0] Default Master Type

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

Bits 8:0 – SLOT_CYCLE[8:0] Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See section Slot Cycle Limit Arbitration for details.

17.11.8 Bus Matrix Slave Configuration Register 5

Name: MATRIX_SCFG5
Offset: 0x54
Reset: 0x000201FF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	1	0

Bit	15	14	13	12	11	10	9	8
								SLOT_CYCLE[8]
Access								R/W
Reset								1

Bit	7	6	5	4	3	2	1	0
	SLOT_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 21:18 – FIXED_DEFMSTR[3:0] Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Bits 17:16 – DEFMSTR_TYPE[1:0] Default Master Type

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

Bits 8:0 – SLOT_CYCLE[8:0] Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See section Slot Cycle Limit Arbitration for details.

17.11.9 Bus Matrix Slave Configuration Register 6

Name: MATRIX_SCFG6
Offset: 0x58
Reset: 0x000001FF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
								SLOT_CYCLE[8]
Access								R/W
Reset								1

Bit	7	6	5	4	3	2	1	0
	SLOT_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 21:18 – FIXED_DEFMSTR[3:0] Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Bits 17:16 – DEFMSTR_TYPE[1:0] Default Master Type

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

Bits 8:0 – SLOT_CYCLE[8:0] Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See section Slot Cycle Limit Arbitration for details.

17.11.10 Bus Matrix Slave Configuration Register 7

Name: MATRIX_SCFG7
Offset: 0x5C
Reset: 0x000101FF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
								SLOT_CYCLE[8]
Access								R/W
Reset								1

Bit	7	6	5	4	3	2	1	0
	SLOT_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 21:18 – FIXED_DEFMSTR[3:0] Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Bits 17:16 – DEFMSTR_TYPE[1:0] Default Master Type

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

Bits 8:0 – SLOT_CYCLE[8:0] Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See section Slot Cycle Limit Arbitration for details.

17.11.11 Bus Matrix Slave Configuration Register x [x=8..15]

Name: MATRIX_SCFGx
Offset: 0x60 + (x-8)*0x04 [x=8..15]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			FIXED_DEFMSTR[3:0]				DEFMSTR_TYPE[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
								SLOT_CYCLE[8]
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
	SLOT_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 21:18 – FIXED_DEFMSTR[3:0] Fixed Default Master
This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

Bits 17:16 – DEFMSTR_TYPE[1:0] Default Master Type

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

Bits 8:0 – SLOT_CYCLE[8:0] Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See section Slot Cycle Limit Arbitration for details.

17.11.12 Bus Matrix Priority Register A For Slave 0

Name: MATRIX_PRAS0
Offset: 0x80
Reset: 0x00007000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN7	M7PR[1:0]			LQOSEN6	M6PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN5	M5PR[1:0]			LQOSEN4	M4PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN3	M3PR[1:0]			LQOSEN2	M2PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		1	1	1		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN1	M1PR[1:0]			LQOSEN0	M0PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.13 Bus Matrix Priority Register B For Slave 0

Name: MATRIX_PRBS0
Offset: 0x84
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN15	M15PR[1:0]			LQOSEN14	M14PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN13	M13PR[1:0]			LQOSEN12	M12PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN11	M11PR[1:0]			LQOSEN10	M10PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN9	M9PR[1:0]			LQOSEN8	M8PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.14 Bus Matrix Priority Register A For Slave 1

Name: MATRIX_PRAS1
Offset: 0x88
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN7	M7PR[1:0]			LQOSEN6	M6PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN5	M5PR[1:0]			LQOSEN4	M4PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN3	M3PR[1:0]			LQOSEN2	M2PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN1	M1PR[1:0]			LQOSEN0	M0PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.15 Bus Matrix Priority Register B For Slave 1

Name: MATRIX_PRBS1
Offset: 0x8C
Reset: 0x00070000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN15	M15PR[1:0]			LQOSEN14	M14PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN13	M13PR[1:0]			LQOSEN12	M12PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		1	1	1

Bit	15	14	13	12	11	10	9	8
		LQOSEN11	M11PR[1:0]			LQOSEN10	M10PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN9	M9PR[1:0]			LQOSEN8	M8PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.16 Bus Matrix Priority Register A For Slave 2

Name: MATRIX_PRAS2
Offset: 0x90
Reset: 0x00007002
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN7	M7PR[1:0]			LQOSEN6	M6PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN5	M5PR[1:0]			LQOSEN4	M4PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN3	M3PR[1:0]			LQOSEN2	M2PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		1	1	1		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN1	M1PR[1:0]			LQOSEN0	M0PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	1	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.17 Bus Matrix Priority Register B for Slave 2

Name: MATRIX_PRBS2
Offset: 0x94
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN15	M15PR[1:0]			LQOSEN14	M14PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN13	M13PR[1:0]			LQOSEN12	M12PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN11	M11PR[1:0]			LQOSEN10	M10PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN9	M9PR[1:0]			LQOSEN8	M8PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.18 Bus Matrix Priority Register A For Slave 3

Name: MATRIX_PRAS3
Offset: 0x98
Reset: 0x00000002
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN7	M7PR[1:0]			LQOSEN6	M6PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN5	M5PR[1:0]			LQOSEN4	M4PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN3	M3PR[1:0]			LQOSEN2	M2PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN1	M1PR[1:0]			LQOSEN0	M0PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	1	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.19 Bus Matrix Priority Register B for Slave 3

Name: MATRIX_PRBS3
Offset: 0x9C
Reset: 0x00070000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN15	M15PR[1:0]			LQOSEN14	M14PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN13	M13PR[1:0]			LQOSEN12	M12PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		1	1	1

Bit	15	14	13	12	11	10	9	8
		LQOSEN11	M11PR[1:0]			LQOSEN10	M10PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN9	M9PR[1:0]			LQOSEN8	M8PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.20 Bus Matrix Priority Register A For Slave 4

Name: MATRIX_PRAS4
Offset: 0xA0
Reset: 0x00000020
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN7	M7PR[1:0]			LQOSEN6	M6PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN5	M5PR[1:0]			LQOSEN4	M4PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN3	M3PR[1:0]			LQOSEN2	M2PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN1	M1PR[1:0]			LQOSEN0	M0PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	1	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.21 Bus Matrix Priority Register B for Slave 4

Name: MATRIX_PRBS4
Offset: 0xA4
Reset: 0x00070000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN15	M15PR[1:0]			LQOSEN14	M14PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN13	M13PR[1:0]			LQOSEN12	M12PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		1	1	1

Bit	15	14	13	12	11	10	9	8
		LQOSEN11	M11PR[1:0]			LQOSEN10	M10PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN9	M9PR[1:0]			LQOSEN8	M8PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.22 Bus Matrix Priority Register A For Slave 5

Name: MATRIX_PRAS5
Offset: 0xA8
Reset: 0x00007220
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN7	M7PR[1:0]			LQOSEN6	M6PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN5	M5PR[1:0]			LQOSEN4	M4PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN3	M3PR[1:0]			LQOSEN2	M2PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		1	1	1		0	1	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN1	M1PR[1:0]			LQOSEN0	M0PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	1	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.23 Bus Matrix Priority Register B for Slave 5

Name: MATRIX_PRBS5
Offset: 0xAC
Reset: 0x00070000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN15	M15PR[1:0]			LQOSEN14	M14PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN13	M13PR[1:0]			LQOSEN12	M12PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		1	1	1

Bit	15	14	13	12	11	10	9	8
		LQOSEN11	M11PR[1:0]			LQOSEN10	M10PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN9	M9PR[1:0]			LQOSEN8	M8PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.24 Bus Matrix Priority Register A For Slave 6

Name: MATRIX_PRAS6
Offset: 0xB0
Reset: 0x00007000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN7	M7PR[1:0]			LQOSEN6	M6PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN5	M5PR[1:0]			LQOSEN4	M4PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN3	M3PR[1:0]			LQOSEN2	M2PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		1	1	1		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN1	M1PR[1:0]			LQOSEN0	M0PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.25 Bus Matrix Priority Register B for Slave 6

Name: MATRIX_PRBS6
Offset: 0xB4
Reset: 0x00070000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN15	M15PR[1:0]			LQOSEN14	M14PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN13	M13PR[1:0]			LQOSEN12	M12PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		1	1	1

Bit	15	14	13	12	11	10	9	8
		LQOSEN11	M11PR[1:0]			LQOSEN10	M10PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN9	M9PR[1:0]			LQOSEN8	M8PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.26 Bus Matrix Priority Register A For Slave x [x=7..15]

Name: MATRIX_PRASx
Offset: 0xB8 + (x-7)*0x08 [x=7..15]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN7	M7PR[1:0]			LQOSEN6	M6PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN5	M5PR[1:0]			LQOSEN4	M4PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN3	M3PR[1:0]			LQOSEN2	M2PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN1	M1PR[1:0]			LQOSEN0	M0PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.27 Bus Matrix Priority Register B for Slave x [x=7..15]

Name: MATRIX_PRBSx
Offset: 0xBC + (x-7)*0x08 [x=7..15]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Priority Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
		LQOSEN15	M15PR[1:0]			LQOSEN14	M14PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
		LQOSEN13	M13PR[1:0]			LQOSEN12	M12PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	15	14	13	12	11	10	9	8
		LQOSEN11	M11PR[1:0]			LQOSEN10	M10PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
		LQOSEN9	M9PR[1:0]			LQOSEN8	M8PR[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 2, 6, 10, 14, 18, 22, 26, 30 – LQOSENx Latency Quality of Service Enable for Master x

Value	Description
0	Disables propagation of Latency Quality of Service from the Master x to the Slave and apply MxPR priority for all access from Master x to the Slave.
1	Enables the propagation of Latency Quality of Service from the Master x to the Slave if supported by the Master x.

Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See section Arbitration Priority Scheme for details.
If LQOSENx bit is cleared, then this priority value is used as it for arbitration and downward propagation to the slave.
If LQOSENx bit is set, then this priority acts as the upper limit for the Latency Quality of Service from Master x.
For masters other than the CPU, the usual value of this field should be 0x0 if LQOSENx bit is cleared, and 0x1 if LQOSENx bit is set. For the CPU master, the usual value of this field should be 0x2.

17.11.28 Bus Matrix Master Remap Control Register

Name: MATRIX_MRCR
Offset: 0x0100
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RCB15	RCB14	RCB13	RCB12	RCB11	RCB10	RCB9	RCB8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RCB7	RCB6	RCB5	RCB4	RCB3	RCB2	RCB1	RCB0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – RCBx Remap Command Bit for Master x

Value	Description
0	Disables remapped address decoding for the selected Master.
1	Enables remapped address decoding for the selected Master.

17.11.29 MATRIX Special Function Register x [x=0..15]

Name: MATRIX_SFRx
Offset: 0x0110 + x*0x04 [x=0..15]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	SFR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	SFR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SFR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SFR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – SFR[31:0] Special Function Register Fields
The SFR fields are a set of D-type Flip-flops which are only connected to outputs of the MATRIX.
They are readable/writable from the User Interface and may be used to implement Configuration Registers which cannot be implemented in any of the other embedded peripherals of the product. Each bit of the SFR may be removed by hardware customizing at synthesis if not used.

17.11.30 Master Error Interrupt Enable Register

Name: MATRIX_MEIER
Offset: 0x0150
Reset: –
Property: Write-only

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	MERR15	MERR14	MERR13	MERR12	MERR11	MERR10	MERR9	MERR8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – MERRx Master x Access Error

Value	Description
0	No effect.
1	Enables Master x Access Error interrupt source.

17.11.31 Master Error Interrupt Disable Register

Name: MATRIX_MEIDR
Offset: 0x0154
Reset: –
Property: Write-only

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	MERR15	MERR14	MERR13	MERR12	MERR11	MERR10	MERR9	MERR8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – MERRx Master x Access Error

Value	Description
0	No effect.
1	Disables Master x Access Error interrupt source.

17.11.32 Master Error Interrupt Mask Register

Name: MATRIX_MEIMR
Offset: 0x0158
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	MERR15	MERR14	MERR13	MERR12	MERR11	MERR10	MERR9	MERR8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – MERRx Master x Access Error

Value	Description
0	Master x Access Error does not trigger any interrupt.
1	Master x Access Error triggers the Bus Matrix interrupt line.

17.11.33 Master Error Status Register

Name: MATRIX_MESR
Offset: 0x015C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	MERR15	MERR14	MERR13	MERR12	MERR11	MERR10	MERR9	MERR8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – MERRx Master x Access Error

Value	Description
0	No Master Access Error has occurred since the last read of the MATRIX_MESR.
1	At least one Master Access Error has occurred since the last read of the MATRIX_MESR.

17.11.34 MATRIX Master Error Address Register x [x=0..15]

Name: MATRIX_MEARx
Offset: 0x0160 + x*0x04 [x=0..15]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ERRADD[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ERRADD[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ERRADD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ERRADD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ERRADD[31:0] Master Error Address
Master Last Access Error Address

17.11.35 MATRIX Write Protection Mode Register

Name: MATRIX_WPMR
Offset: 0x01E4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CFGFRZ							WPEN
Access	R/W							R/W
Reset	0							0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x4D4154	PASSWD	Writing any other value in this field aborts the write operation of the WPEN and CFGFRZ bits. Always reads as 0.

Bit 7 – CFGFRZ Configuration Freeze

Value	Description
0	The MATRIX configuration is not frozen.
1	Freezes the MATRIX configuration until hardware reset. The registers that can be protected by the WPEN bit and the Write Protection Mode Register are no longer modifiable.

Bit 0 – WPEN Write Protection Enable
See section Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x4D4154 (“MAT” in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x4D4154 (“MAT” in ASCII).

17.11.36 Write Protection Status Register

Name: MATRIX_WPSR
Offset: 0x01E8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	WPVSRC[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 23:8 – WPVSRC[15:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protection violation has occurred since the last write of the MATRIX_WPMR.
1	A write protection violation has occurred since the last write of the MATRIX_WPMR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

17.11.37 Protection Slave Register 0

Name: MATRIX_PSR0
Offset: 0x0200
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 24, 25, 26, 27, 28, 29, 30, 31 – DPSOAx Downward Protection Split Address for HSELx Protected Region

Value	Description
0	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the base address of the slave protected region and up.
1	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the end address of the slave-protected region and down.

Bits 16, 17, 18, 19, 20, 21, 22, 23 – WRUSERHx Write USER for HSELx Protected Region
Privilege-eligible Area access rights:

WRUSERHx/RDUSERHx	User Access	Privileged Access
00	Denied	Write–Read
01	Read	Write–Read
10	Write	Write–Read
11	Write–Read	Write–Read

Value	Description
0	The HSELx AHB slave-protected region is split into one Write Privileged and one Write User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Write access.
1	The HSELx AHB slave-protected region is User for Write access.

Bits 8, 9, 10, 11, 12, 13, 14, 15 – RDUSERHx Read USER for HSELx Protected Region

Value	Description
0	The HSELx AHB slave-protected region is split into one Read Privileged and one Read User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Read access.
1	The HSELx AHB slave-protected region is User for Read access.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – LAUSERHx Low Area USER in HSELx Protected Region

Value	Description
0	The protection of the HSELx AHB slave area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured according to RDUSERHx and WRUSERHx. The whole remaining HSELx upper address space is configured as User access.
1	The HSELx AHB slave address area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured as User access, and the whole remaining upper address space according to RDUSERHx and WRUSERHx.

17.11.38 Protection Slave Register 1

Name: MATRIX_PSR1
Offset: 0x0204
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 24, 25, 26, 27, 28, 29, 30, 31 – DPSOAx Downward Protection Split Address for HSELx Protected Region

Value	Description
0	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the base address of the slave protected region and up.
1	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the end address of the slave-protected region and down.

Bits 16, 17, 18, 19, 20, 21, 22, 23 – WRUSERHx Write USER for HSELx Protected Region
Privilege-eligible Area access rights:

WRUSERHx/RDUSERHx	User Access	Privileged Access
00	Denied	Write–Read
01	Read	Write–Read
10	Write	Write–Read
11	Write–Read	Write–Read

Value	Description
0	The HSELx AHB slave-protected region is split into one Write Privileged and one Write User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Write access.
1	The HSELx AHB slave-protected region is User for Write access.

Bits 8, 9, 10, 11, 12, 13, 14, 15 – RDUSERHx Read USER for HSELx Protected Region

Value	Description
0	The HSELx AHB slave-protected region is split into one Read Privileged and one Read User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Read access.
1	The HSELx AHB slave-protected region is User for Read access.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – LAUSERHx Low Area USER in HSELx Protected Region

Value	Description
0	The protection of the HSELx AHB slave area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured according to RDUSERHx and WRUSERHx. The whole remaining HSELx upper address space is configured as User access.
1	The HSELx AHB slave address area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured as User access, and the whole remaining upper address space according to RDUSERHx and WRUSERHx.

17.11.39 Protection Slave Register 2

Name: MATRIX_PSR2
Offset: 0x0208
Reset: 0x00010101
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit	7	6	5	4	3	2	1	0
	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bits 24, 25, 26, 27, 28, 29, 30, 31 – DPSOAx Downward Protection Split Address for HSELx Protected Region

Value	Description
0	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the base address of the slave protected region and up.
1	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the end address of the slave-protected region and down.

Bits 16, 17, 18, 19, 20, 21, 22, 23 – WRUSERHx Write USER for HSELx Protected Region
Privilege-eligible Area access rights:

WRUSERHx/RDUSERHx	User Access	Privileged Access
00	Denied	Write–Read
01	Read	Write–Read
10	Write	Write–Read
11	Write–Read	Write–Read

Value	Description
0	The HSELx AHB slave-protected region is split into one Write Privileged and one Write User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Write access.
1	The HSELx AHB slave-protected region is User for Write access.

Bits 8, 9, 10, 11, 12, 13, 14, 15 – RDUSERHx Read USER for HSELx Protected Region

Value	Description
0	The HSELx AHB slave-protected region is split into one Read Privileged and one Read User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Read access.
1	The HSELx AHB slave-protected region is User for Read access.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – LAUSERHx Low Area USER in HSELx Protected Region

Value	Description
0	The protection of the HSELx AHB slave area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured according to RDUSERHx and WRUSERHx. The whole remaining HSELx upper address space is configured as User access.
1	The HSELx AHB slave address area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured as User access, and the whole remaining upper address space according to RDUSERHx and WRUSERHx.

17.11.40 Protection Slave Register 3

Name: MATRIX_PSR3
Offset: 0x020C
Reset: 0x00FFFFFF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 24, 25, 26, 27, 28, 29, 30, 31 – DPSOAx Downward Protection Split Address for HSELx Protected Region

Value	Description
0	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the base address of the slave protected region and up.
1	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the end address of the slave-protected region and down.

Bits 16, 17, 18, 19, 20, 21, 22, 23 – WRUSERHx Write USER for HSELx Protected Region
Privilege-eligible Area access rights:

WRUSERHx/RDUSERHx	User Access	Privileged Access
00	Denied	Write–Read
01	Read	Write–Read
10	Write	Write–Read
11	Write–Read	Write–Read

Value	Description
0	The HSELx AHB slave-protected region is split into one Write Privileged and one Write User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Write access.
1	The HSELx AHB slave-protected region is User for Write access.

Bits 8, 9, 10, 11, 12, 13, 14, 15 – RDUSERHx Read USER for HSELx Protected Region

Value	Description
0	The HSELx AHB slave-protected region is split into one Read Privileged and one Read User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Read access.
1	The HSELx AHB slave-protected region is User for Read access.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – LAUSERHx Low Area USER in HSELx Protected Region

Value	Description
0	The protection of the HSELx AHB slave area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured according to RDUSERHx and WRUSERHx. The whole remaining HSELx upper address space is configured as User access.
1	The HSELx AHB slave address area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured as User access, and the whole remaining upper address space according to RDUSERHx and WRUSERHx.

17.11.41 Protection Slave Register 4

Name: MATRIX_PSR4
Offset: 0x0210
Reset: 0x00010101
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit	7	6	5	4	3	2	1	0
	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bits 24, 25, 26, 27, 28, 29, 30, 31 – DPSOAx Downward Protection Split Address for HSELx Protected Region

Value	Description
0	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the base address of the slave protected region and up.
1	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the end address of the slave-protected region and down.

Bits 16, 17, 18, 19, 20, 21, 22, 23 – WRUSERHx Write USER for HSELx Protected Region
Privilege-eligible Area access rights:

WRUSERHx/RDUSERHx	User Access	Privileged Access
00	Denied	Write–Read
01	Read	Write–Read
10	Write	Write–Read
11	Write–Read	Write–Read

Value	Description
0	The HSELx AHB slave-protected region is split into one Write Privileged and one Write User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Write access.
1	The HSELx AHB slave-protected region is User for Write access.

Bits 8, 9, 10, 11, 12, 13, 14, 15 – RDUSERHx Read USER for HSELx Protected Region

Value	Description
0	The HSELx AHB slave-protected region is split into one Read Privileged and one Read User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Read access.
1	The HSELx AHB slave-protected region is User for Read access.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – LAUSERHx Low Area USER in HSELx Protected Region

Value	Description
0	The protection of the HSELx AHB slave area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured according to RDUSERHx and WRUSERHx. The whole remaining HSELx upper address space is configured as User access.
1	The HSELx AHB slave address area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured as User access, and the whole remaining upper address space according to RDUSERHx and WRUSERHx.

17.11.42 Protection Slave Register 5

Name: MATRIX_PSR5
Offset: 0x0214
Reset: 0x00010101
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit	7	6	5	4	3	2	1	0
	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bits 24, 25, 26, 27, 28, 29, 30, 31 – DPSOAx Downward Protection Split Address for HSELx Protected Region

Value	Description
0	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the base address of the slave protected region and up.
1	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the end address of the slave-protected region and down.

Bits 16, 17, 18, 19, 20, 21, 22, 23 – WRUSERHx Write USER for HSELx Protected Region
Privilege-eligible Area access rights:

WRUSERHx/RDUSERHx	User Access	Privileged Access
00	Denied	Write–Read
01	Read	Write–Read
10	Write	Write–Read
11	Write–Read	Write–Read

Value	Description
0	The HSELx AHB slave-protected region is split into one Write Privileged and one Write User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Write access.
1	The HSELx AHB slave-protected region is User for Write access.

Bits 8, 9, 10, 11, 12, 13, 14, 15 – RDUSERHx Read USER for HSELx Protected Region

Value	Description
0	The HSELx AHB slave-protected region is split into one Read Privileged and one Read User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Read access.
1	The HSELx AHB slave-protected region is User for Read access.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – LAUSERHx Low Area USER in HSELx Protected Region

Value	Description
0	The protection of the HSELx AHB slave area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured according to RDUSERHx and WRUSERHx. The whole remaining HSELx upper address space is configured as User access.
1	The HSELx AHB slave address area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured as User access, and the whole remaining upper address space according to RDUSERHx and WRUSERHx.

17.11.43 Protection Slave Register x [x=6..15]

Name: MATRIX_PSRx
Offset: 0x0218 + (x-6)*0x04 [x=6..15]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	DPSOA7	DPSOA6	DPSOA5	DPSOA4	DPSOA3	DPSOA2	DPSOA1	DPSOA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WRUSERH7	WRUSERH6	WRUSERH5	WRUSERH4	WRUSERH3	WRUSERH2	WRUSERH1	WRUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RDUSERH7	RDUSERH6	RDUSERH5	RDUSERH4	RDUSERH3	RDUSERH2	RDUSERH1	RDUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LAUSERH7	LAUSERH6	LAUSERH5	LAUSERH4	LAUSERH3	LAUSERH2	LAUSERH1	LAUSERH0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 24, 25, 26, 27, 28, 29, 30, 31 – DPSOAx Downward Protection Split Address for HSELx Protected Region

Value	Description
0	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the base address of the slave protected region and up.
1	For the HSELx AHB slave-protected region, the protected area split MATRIX_PASSR / PASPLITx defines the size of the slave-protected area starting from the end address of the slave-protected region and down.

Bits 16, 17, 18, 19, 20, 21, 22, 23 – WRUSERHx Write USER for HSELx Protected Region
Privilege-eligible Area access rights:

WRUSERHx/RDUSERHx	User Access	Privileged Access
00	Denied	Write–Read
01	Read	Write–Read
10	Write	Write–Read
11	Write–Read	Write–Read

Value	Description
0	The HSELx AHB slave-protected region is split into one Write Privileged and one Write User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Write access.
1	The HSELx AHB slave-protected region is User for Write access.

Bits 8, 9, 10, 11, 12, 13, 14, 15 – RDUSERHx Read USER for HSELx Protected Region

Value	Description
0	The HSELx AHB slave-protected region is split into one Read Privileged and one Read User area, according to LAUSERHx and MATRIX_PASSR / PASPLITx. That is, the so defined privilege-eligible high or low area is Privileged for Read access.
1	The HSELx AHB slave-protected region is User for Read access.

Bits 0, 1, 2, 3, 4, 5, 6, 7 – LAUSERHx Low Area USER in HSELx Protected Region

Value	Description
0	The protection of the HSELx AHB slave area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured according to RDUSERHx and WRUSERHx. The whole remaining HSELx upper address space is configured as User access.
1	The HSELx AHB slave address area laying below the corresponding MATRIX_PASSR / PASPLITx boundary is configured as User access, and the whole remaining upper address space according to RDUSERHx and WRUSERHx.

17.11.44 Protected Areas Split Slave Register 0

Name: MATRIX_PASSR0
Offset: 0x0240
Reset: 0x00000FFF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PASPLIT7[3:0]				PASPLIT6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PASPLIT5[3:0]				PASPLIT4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PASPLIT3[3:0]				PASPLIT2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	PASPLIT1[3:0]				PASPLIT0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PASPLITx Protected Areas Split for HSELx Protected Region
This field defines the boundary address offset where the HSELx AHB slave-protected region splits into two Protected Areas whose access is controlled according to the corresponding MATRIX_PSR.
If this size is set at or above the HSELx Region Size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx Protected Region.

PASPLITx	Split Offset	Protected Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes

1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.45 Protected Areas Split Slave Register 1

Name: MATRIX_PASSR1
Offset: 0x0244
Reset: 0x00000FFF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PASPLIT7[3:0]				PASPLIT6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PASPLIT5[3:0]				PASPLIT4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PASPLIT3[3:0]				PASPLIT2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	PASPLIT1[3:0]				PASPLIT0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PASPLITx Protected Areas Split for HSELx Protected Region
This field defines the boundary address offset where the HSELx AHB slave-protected region splits into two Protected Areas whose access is controlled according to the corresponding MATRIX_PSR.
If this size is set at or above the HSELx Region Size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx Protected Region.

PASPLITx	Split Offset	Protected Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes

1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.46 Protected Areas Split Slave Register 2

Name: MATRIX_PASSR2
Offset: 0x0248
Reset: 0x0000000F
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PASPLIT7[3:0]				PASPLIT6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PASPLIT5[3:0]				PASPLIT4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PASPLIT3[3:0]				PASPLIT2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PASPLIT1[3:0]				PASPLIT0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	1	1	1

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PASPLITx Protected Areas Split for HSELx Protected Region
This field defines the boundary address offset where the HSELx AHB slave-protected region splits into two Protected Areas whose access is controlled according to the corresponding MATRIX_PSR.
If this size is set at or above the HSELx Region Size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx Protected Region.

PASPLITx	Split Offset	Protected Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes

1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.47 Protected Areas Split Slave Register 3

Name: MATRIX_PASSR3
Offset: 0x024C
Reset: 0xFFFFFFFF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PASPLIT7[3:0]				PASPLIT6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	PASPLIT5[3:0]				PASPLIT4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	PASPLIT3[3:0]				PASPLIT2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	PASPLIT1[3:0]				PASPLIT0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PASPLITx Protected Areas Split for HSELx Protected Region
This field defines the boundary address offset where the HSELx AHB slave-protected region splits into two Protected Areas whose access is controlled according to the corresponding MATRIX_PSR.
If this size is set at or above the HSELx Region Size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx Protected Region.

PASPLITx	Split Offset	Protected Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes

1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.48 Protected Areas Split Slave Register 4

Name: MATRIX_PASSR4
Offset: 0x0250
Reset: 0x0000000F
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PASPLIT7[3:0]				PASPLIT6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PASPLIT5[3:0]				PASPLIT4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PASPLIT3[3:0]				PASPLIT2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PASPLIT1[3:0]				PASPLIT0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	1	1	1

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PASPLITx Protected Areas Split for HSELx Protected Region
This field defines the boundary address offset where the HSELx AHB slave-protected region splits into two Protected Areas whose access is controlled according to the corresponding MATRIX_PSR.
If this size is set at or above the HSELx Region Size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx Protected Region.

PASPLITx	Split Offset	Protected Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes

1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.49 Protected Areas Split Slave Register 5

Name: MATRIX_PASSR5
Offset: 0x0254
Reset: 0x0000000F
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PASPLIT7[3:0]				PASPLIT6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PASPLIT5[3:0]				PASPLIT4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PASPLIT3[3:0]				PASPLIT2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PASPLIT1[3:0]				PASPLIT0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	1	1	1

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PASPLITx Protected Areas Split for HSELx Protected Region
This field defines the boundary address offset where the HSELx AHB slave-protected region splits into two Protected Areas whose access is controlled according to the corresponding MATRIX_PSR.
If this size is set at or above the HSELx Region Size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx Protected Region.

PASPLITx	Split Offset	Protected Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes

1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.50 Protected Areas Split Slave Register 6

Name: MATRIX_PASSR6
Offset: 0x0258
Reset: 0x00000FFF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PASPLIT7[3:0]				PASPLIT6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PASPLIT5[3:0]				PASPLIT4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PASPLIT3[3:0]				PASPLIT2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	PASPLIT1[3:0]				PASPLIT0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PASPLITx Protected Areas Split for HSELx Protected Region
This field defines the boundary address offset where the HSELx AHB slave-protected region splits into two Protected Areas whose access is controlled according to the corresponding MATRIX_PSR.
If this size is set at or above the HSELx Region Size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx Protected Region.

PASPLITx	Split Offset	Protected Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes

1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.51 Protected Areas Split Slave Register 7

Name: MATRIX_PASSR7
Offset: 0x025C
Reset: 0x00000FFF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PASPLIT7[3:0]				PASPLIT6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PASPLIT5[3:0]				PASPLIT4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PASPLIT3[3:0]				PASPLIT2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	PASPLIT1[3:0]				PASPLIT0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PASPLITx Protected Areas Split for HSELx Protected Region
This field defines the boundary address offset where the HSELx AHB slave-protected region splits into two Protected Areas whose access is controlled according to the corresponding MATRIX_PSR.
If this size is set at or above the HSELx Region Size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx Protected Region.

PASPLITx	Split Offset	Protected Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes

1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.52 Protected Areas Split Slave Register x [x=8..15]

Name: MATRIX_PASSRx
Offset: 0x0260 + (x-8)*0x04 [x=8..15]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PASPLIT7[3:0]				PASPLIT6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PASPLIT5[3:0]				PASPLIT4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PASPLIT3[3:0]				PASPLIT2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PASPLIT1[3:0]				PASPLIT0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PASPLITx Protected Areas Split for HSELx Protected Region
This field defines the boundary address offset where the HSELx AHB slave-protected region splits into two Protected Areas whose access is controlled according to the corresponding MATRIX_PSR.
If this size is set at or above the HSELx Region Size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx Protected Region.

PASPLITx	Split Offset	Protected Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes

1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.53 MATRIX Protected Region Top Slave Register x [x=0..15]

Name: MATRIX_PRTSRx
Offset: 0x0280 + x*0x04 [x=0..15]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the MATRIX Protection registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	PRTOP7[3:0]				PRTOP6[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PRTOP5[3:0]				PRTOP4[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PRTOP3[3:0]				PRTOP2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PRTOP1[3:0]				PRTOP0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – PRTOPx HSELx Protected Region Top
This field defines the size of the HSELx protected region address space. Invalid sizes for the slave region must never be programmed. Valid sizes and number of protected regions are product, slave and slave configuration dependent.
Note: The slaves featuring multiple scalable contiguous protected regions have a single PRTOP0 field for all the protected regions.

If this HSELx protected region size is set at or below the MATRIX_PASSR / PASPLITx Protected Areas Split size, then the MATRIX_PSR register settings apply to the unique protected area then covering the whole HSELx protected region

PRTOPx	Top Offset	Protected Region Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes

.....continued

PRTOPx	Top Offset	Protected Region Size
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes
1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.11.54 Protected Peripheral Select 0 Register

Name: MATRIX_PPSELR0
Offset: 0x02C0
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	USERP31	USERP30	USERP29	USERP28	USERP27	USERP26	USERP25	USERP24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	USERP23	USERP22	USERP21	USERP20	USERP19	USERP18	USERP17	USERP16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	USERP15	USERP14	USERP13	USERP12	USERP11	USERP10	USERP9	USERP8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	USERP7	USERP6	USERP5	USERP4	USERP3	USERP2	USERP1	USERP0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – USERPy User PSELy Peripheral

Value	Description
0	The PSELx[y] APB Peripheral address space is configured as Privileged access.
1	The PSELx[y] APB Peripheral address space is configured as User access.

17.11.55 Protected Peripheral Select 1 Register

Name: MATRIX_PPSELR1
Offset: 0x02C4
Reset: 0x00000040
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	USERP31	USERP30	USERP29	USERP28	USERP27	USERP26	USERP25	USERP24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	USERP23	USERP22	USERP21	USERP20	USERP19	USERP18	USERP17	USERP16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	USERP15	USERP14	USERP13	USERP12	USERP11	USERP10	USERP9	USERP8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	USERP7	USERP6	USERP5	USERP4	USERP3	USERP2	USERP1	USERP0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	0	0	0	0	0	0

Bits 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 – USERPy User PSELy Peripheral

Value	Description
0	The PSELx[y] APB Peripheral address space is configured as Privileged access.
1	The PSELx[y] APB Peripheral address space is configured as User access.

17.11.56 Protected Peripheral Select 2 Register

Name: MATRIX_PPSELR2
Offset: 0x02C8
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Values in the Bus Matrix Protection Registers are product-dependent.

Bit	31	30	29	28	27	26	25	24
	USERP31	USERP30	USERP29	USERP28	USERP27	USERP26	USERP25	USERP24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	USERP23	USERP22	USERP21	USERP20	USERP19	USERP18	USERP17	USERP16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	USERP15	USERP14	USERP13	USERP12	USERP11	USERP10	USERP9	USERP8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	USERP7	USERP6	USERP5	USERP4	USERP3	USERP2	USERP1	USERP0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – USERPy User PSELy Peripheral

Value	Description
0	The PSELx[y] APB Peripheral address space is configured as Privileged access.
1	The PSELx[y] APB Peripheral address space is configured as User access.

18. Chip Identifier (CHIPID)

18.1 Description

Chip Identifier (CHIPID) registers are used to recognize the device and its revision. These registers provide the sizes and types of the on-chip memories, as well as the set of embedded peripherals.

Two CHIPID registers are embedded: Chip ID register (CHIPID_CIDR) and Chip ID Extension register (CHIPID_EXID). Both registers contain a hard-wired value that is read-only.

The CHIPID_CIDR register contains the following fields:

- VERSION: Identifies the revision of the silicon
- EPROC: Indicates the embedded ARM processor
- NVPTYP and NVPSIZ: Identify the type of embedded non-volatile memory and the size
- SRAMSIZ: Indicates the size of the embedded SRAM
- ARCH: Identifies the set of embedded peripherals
- EXT: Shows the use of the extension identifier register

The CHIPID_EXID register is device-dependent and reads ‘0’ if CHIPID_CIDR.EXT = 0.

18.2 Embedded Characteristics

- Chip ID Registers
  - Identification of the Device Revision, Sizes of the Embedded Memories, Set of Peripherals, Embedded Processor

18.3 Register Summary

Offset	Name	Bit Pos.								
0x00	CHIPID_CIDR	7:0	EPROC[2:0]			VERSION[4:0]				
		15:8	NVPSIZ2[3:0]				NVPSIZ[3:0]			
		23:16	ARCH[3:0]				SRAMSIZ[3:0]			
		31:24	EXT	NVPTYP[2:0]			ARCH[7:4]			
0x04	CHIPID_EXID	7:0	EXID[7:0]							
		15:8	EXID[15:8]							
		23:16	EXID[23:16]							
		31:24	EXID[31:24]							

18.3.1 Chip ID Register

Name: CHIPID_CIDR
Offset: 0x0
Reset: 0xA22F0700
Property: Read-only

Values not listed for bitfields must be considered as “reserved”

Bit	31	30	29	28	27	26	25	24
	EXT	NVPTYP[2:0]			ARCH[7:4]			
Access	R	R	R	R	R	R	R	R
Reset	1	0	1	0	0	0	1	0

Bit	23	22	21	20	19	18	17	16
	ARCH[3:0]				SRAMSIZ[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	1	0	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	NVPSIZ2[3:0]				NVPSIZ[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	1	1	1

Bit	7	6	5	4	3	2	1	0
	EPROC[2:0]			VERSION[4:0]				
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 31 – EXT Extension Flag

Value	Description
0	Chip ID has a single register definition without extension.
1	An extended Chip ID exists.

Bits 30:28 – NVPTYP[2:0] Non-volatile Program Memory Type

Value	Name	Description
2	FLASH	Embedded Flash Memory

Bits 27:20 – ARCH[7:0] Architecture Identifier

Value	Name	Description
34	SAMRH71	SAMRH71

Bits 19:16 – SRAMSIZ[3:0] Internal SRAM Size

Value	Name	Description
2	768K	768 Kbytes

Bits 15:12 – NVPSIZ2[3:0] Second Non-volatile Program Memory Size

Value	Name	Description
0	NONE	None

Bits 11:8 – NVPSIZ[3:0] Non-volatile Program Memory Size

Value	Name	Description
7	128K	128 Kbytes

Bits 7:5 – EPROC[2:0] Embedded Processor

Value	Name	Description
0	SAM x7	Cortex-M7

Bits 4:0 – VERSION[4:0] Version of the Device
Current version of the device.

18.3.2 Chip ID Extension Register

Name: CHIPID_EXID
Offset: 0x4
Reset: 0x00000001
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	EXID[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EXID[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EXID[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EXID[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	1

Bits 31:0 – EXID[31:0] Chip ID Extension
This field is cleared if CHIPID_CIDR.EXT = 0.

19. Hardened Embedded Flash Controller (HEFC)

19.1 Description

The Hardened Embedded Flash Controller (HEFC) provides a high-level interface between the internal Flash plane and the 32-bit internal bus.

Its 128-bit wide memory interface increases performance. It also manages the programming, erasing, locking and unlocking sequences of Flash memories using a full set of commands. One of the commands returns the embedded Flash descriptor definition that informs the system about the Flash organization, thus making the software generic.

This block embeds an ECC functionality to protect the embedded memory against cumulative radiation effects.

19.2 Embedded Characteristics

- Flash Plane Features Managed by HEFC
  - 128 calibration bits
  - 128 end-user fuse bits available
  - 32 lock bits, protecting each the 32 first pages against unwanted write or erase operations
  - 256 bytes of User Signature Area available
- Global HEFC Features
  - Commands protected by a keyword
  - Erases the entire Flash
  - Erases by group of pages (available group sizes in pages: 4, 8, 16, 32)
  - Erases by page
  - Supports erasing before programming
  - Register write protection
- HEFC Data Reliability
  - BCH(12) ECC module and extra parity implemented
    - Detects up to 3 errors per 32-bit word
    - Corrects up to 2 errors per 32-bit word
- Error notification by interruption or flag
- HEFC Power Management
  - Memory plane power ON/OFF control
  - Memory plane insulation

19.3 Product Dependencies

19.3.1 Power/Reset/Clock Management

The Hardened Embedded Flash Controller (HEFC) must be clocked by a 4 MHz clock at startup. Prior to any use (for boot, etc.), Flash modules driven by the HEFC must be properly powered-on and calibrated. The HEFC is responsible of this full power-up sequence based on the 4 MHz clock. Once this power-up sequence is done, the HEFC module notifies it.

Clocks provided to Flash modules must not exceed 20 MHz and are generated by the HEFC itself with a ratio of 3 ($CLK_{Flash} = CLK_{HEFC}/3$); CLK_{HEFC} must not exceed 60 MHz. Thus after the startup sequence, the 10 MHz clock may be changed but must not exceed 60 MHz.

Flash modules may be insulated and powered off during the application phase using the power management techniques embedded in the HEFC.

The write/erase clock is provided by the PMC. It must be configured at 2 MHz and enabled before using the erase/write feature on Flash memory.

19.3.2 Interrupt Sources

The HEFC interrupt lines are connected to the interrupt controller. Using the HEFC interrupts requires the interrupt controller to be programmed first. The HEFC interrupts are generated only if the corresponding value of Flash Mode register (HEFC_FMR) is ‘1’. For Flash Ready Interrupt, HEFC_FMR.FRDY must be set to ‘1’ and for Flash Power Status Interrupt, HEFC_FMR.FPSI must be set to ‘1’.

19.4 Functional Description

19.4.1 Aerospace Restrictions

The write and erase processes of the embedded Flash memories are not allowed during flights. Furthermore, for better data retention in a radiation environment, the HEFC offers the possibility for the end user to power-up or power-down (with automatic insulation process) the Flash memories.

19.4.2 Embedded Flash Organization

The embedded Flash interfaces directly with the internal bus. The embedded Flash is composed of the following:

- One memory plane organized in:
  - One write buffer (latch buffer) that manages page programming. The write buffer size is equal to the page size (256 bytes of data and 128 bytes of ECC). This buffer is write-only and accessible all along the 128-KByte address space, so that each word can be written to its final address.
  - One array of 512 pages (256 bytes of data and 128 bytes of ECC)
  - Two additional rows which contain:
    - Xrow[0]: 128 fuse bits, 32 lock bits and 32 NVM calibration bits (see Flash Commands for details on how they can be accessed)
    - Xrow[1] (User signature area): 1 page of 256 bytes and 128 bytes of ECC

  Note: Each 32 data bits (except for Xrow[0]) can be protected using 16 extra bits. The HEFC uses 13 extra bits to protect the 32 data bits on-the-fly. The 3 extra bits that are unused are padded to ‘0’ and erased to ‘1’.
- An ECC block to protect end user data against cumulative radiation events with a BCH code (32 bits of data protected by 13 bits of checksum). Per 32-databit word, one or two errors are corrected while three errors are detected. If a word contains 4 or more errors, the ECC block behaves unpredictably. All detected errors are monitored and notified.
- A Flash controller to interface the 128-Kbyte memory plane with the 32-bit interface AMBA bus. The memory plane responds natively to the different write/read AMBA commands:
  - When an AMBA READ command is received by the controller, a 128-bit READ operation is performed through the 128-Kbyte memory plane to collect the data. For more details, see Read Operations.
  - When an AMBA WRITE command is received by the controller, a LOAD operation is performed into the latch buffer (equivalent to one page). In a second step, a write command pushes these latched data to the Flash memories. For more details, see Write Commands.

  The other commands to control the memory plane are sent by the end user through the dedicated Flash Command register (HEFC_FCR). See Flash Commands for details.

  Thus, this block:
  - Manages high level requests from the end user
  - Generates the appropriate sequences to Flash memories
  - Routes the dataflow from end-user to Flash memories and from Flash memories to end user
  - Drives the insulation controller
- An insulation controller (driven by the Flash controller) to manage Flash memories' insulation and their power-up or power- down. For additional information, refer to Flash Power Management.

Figure 19-1. Embedded Flash Overview

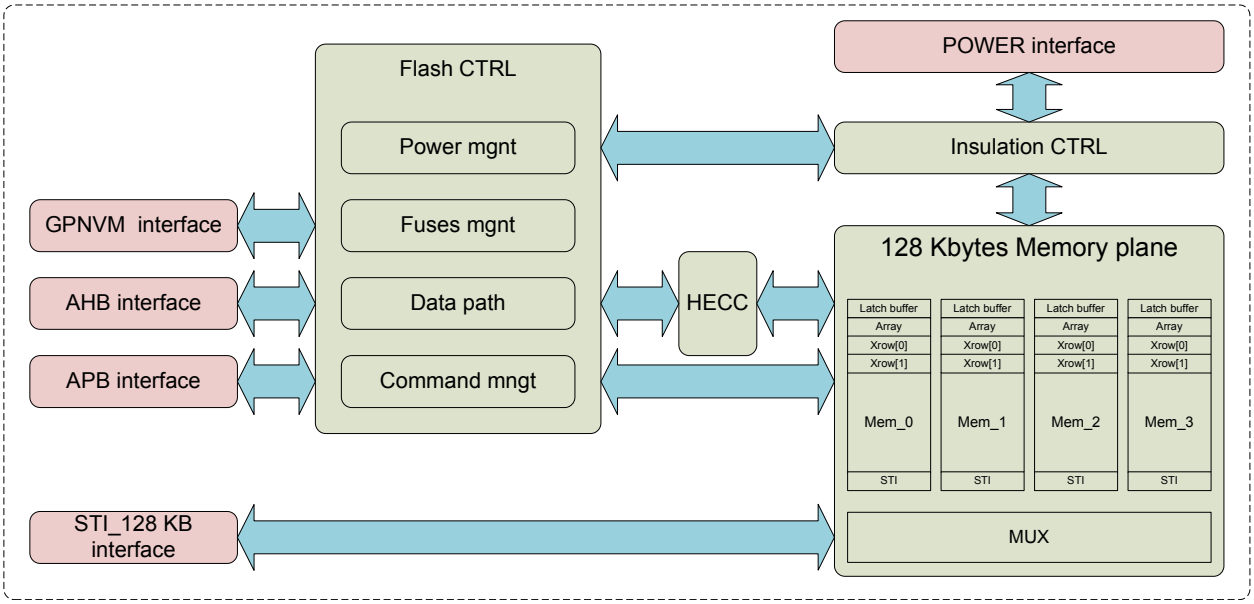
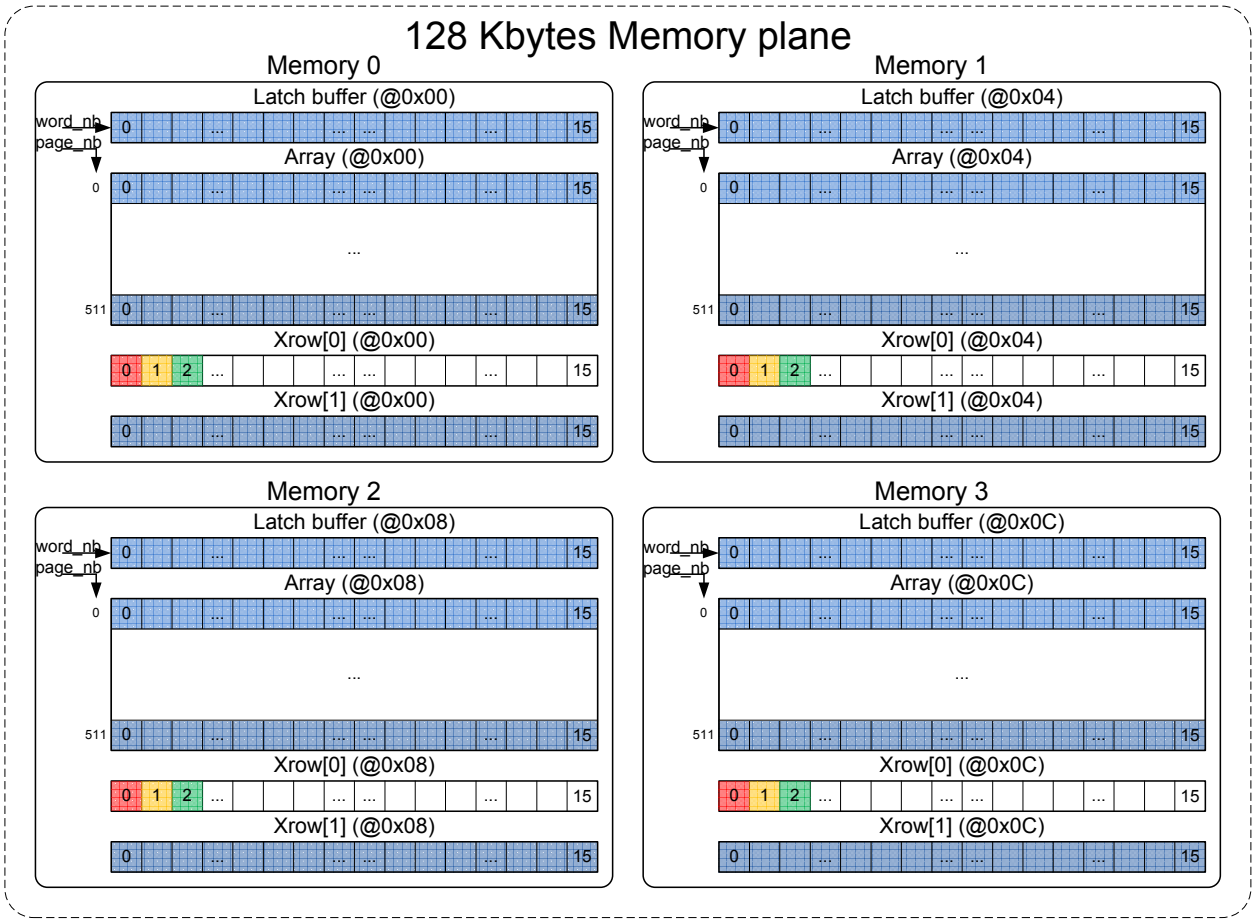


Figure 19-2. Flash Memory Plane Overview



Calibration bits (32 bits * 4) – each memory has its own calibration bits, thus 128 calibration bits are managed from memory plane's point of view

General Purpose bits (32 bits * 4) – each memory has 32 bits dedicated for general purpose (useful for storing chip configuration)

Lock bits (32 bits) – each memory as 32 lock bits to lock the 32 first pages, but all memories are seen as a single memory from end-user point of view, thus the 32 lock bits are managed in the same way at a time for all memories, thus, only 32 effective lock bits are seen.

User Data (48 bits) – each data (from array or Xrow[1] are 48 bits wide in each memory, HEFC manages these 48 bits to protect 32-bits of data with 13-bits of ECC (BCH + Parity). This means that each data could be corrected on-the-fly if one or two errors are detected.

Figure 19-3. Flash Memory Plane Areas

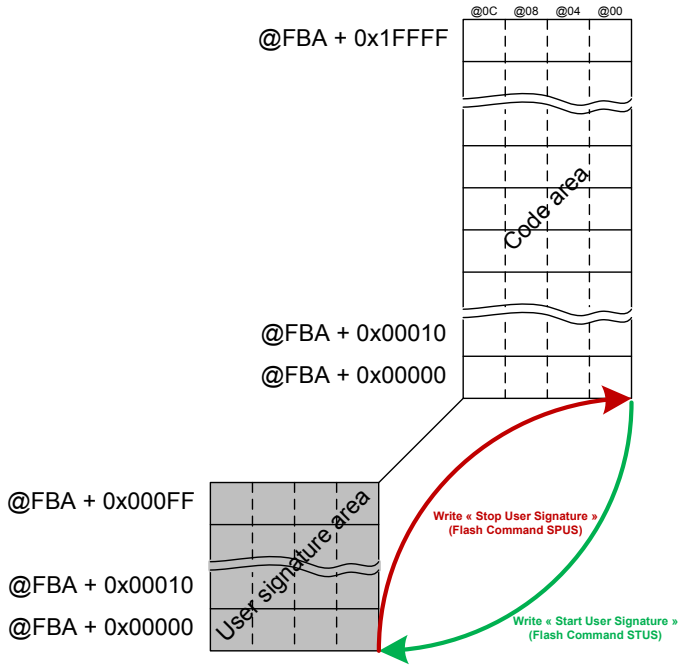
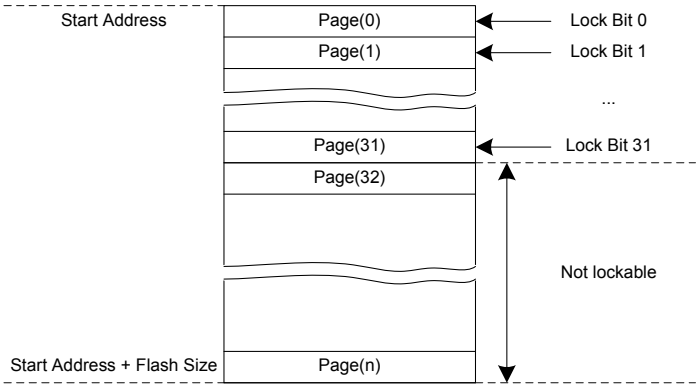


Figure 19-4. Flash Memory Plane Lockable Region



19.4.3 Read Operations

An optimized controller manages embedded Flash reads, thus increasing performance when the processor is running in Thumb2 mode by means of the 128-bit-wide memory interface.

The Flash memory is accessible through 8-, 16- and 32-bit reads.

As the Flash block size is smaller than the address space reserved for the internal memory area, the embedded Flash wraps around the address space and appears to be repeated within it.

19.4.4 Flash Commands

19.4.4.1 Introduction

The HEFC offers a set of commands to manage programming of Flash memory plane, locking and unlocking lockable regions, consecutive programming, full Flash erasing, etc.

The commands are listed in the following table:

Command	Value	Mnemonic
Get Flash descriptor	0x00	GETD

.....continued

Command	Value	Mnemonic
Write page	0x01	WP
Write page and lock	0x02	WPL
Erase all	0x05	EA
Erase page	0x06	EP
Erase pages	0x07	EPA
Set lock bit	0x08	SLB
Clear lock bit	0x09	CLB
Get lock bit	0x0A	GLB
Set GPNVM bit	0x0B	SGPB
Clear GPNVM bit	0x0C	CGPB
Get GPNVM bit	0x0D	GGPB
Get CALIB bit	0x10	GCALB
Write user signature	0x12	WUS
Erase user signature	0x13	EUS
Start read user signature	0x14	STUS
Stop read user signature	0x15	SPUS

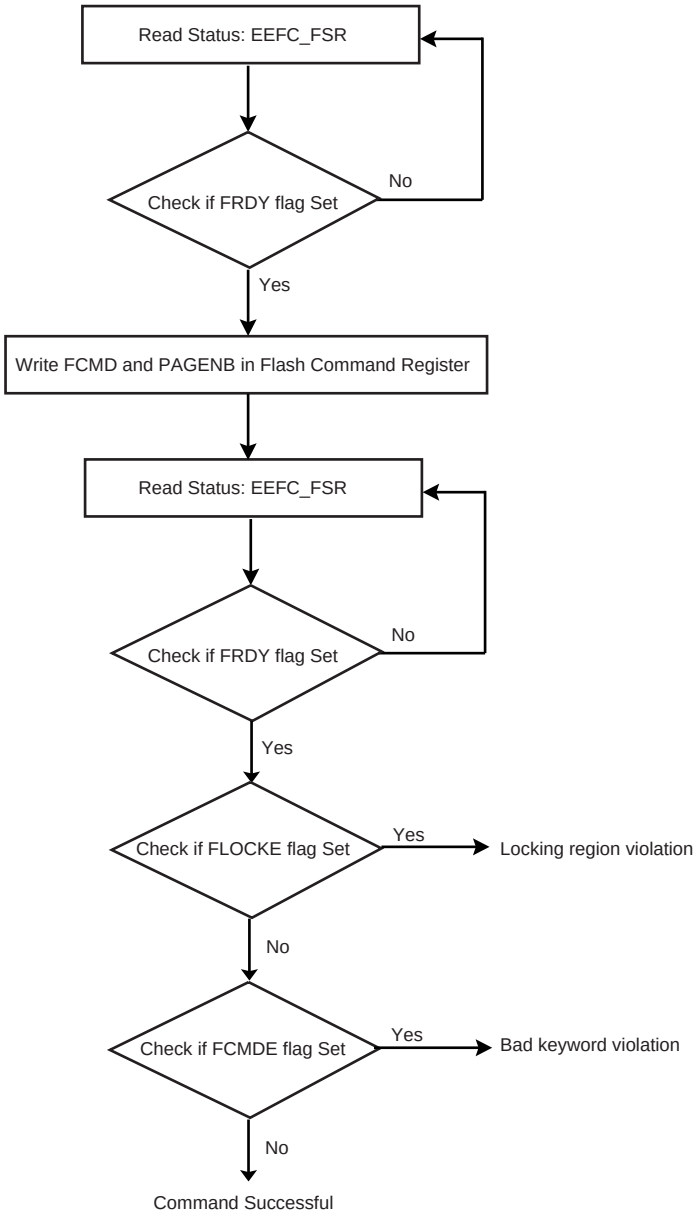
In order to execute one of these commands, select the required command using the FCMD field in the Flash Command register (HEFC_FCR). As soon as HEFC_FCR is written, the FRDY flag and the FVALUE field in the Flash Result register (HEFC_FRR) are automatically cleared. Once the current command has completed, the FRDY flag is automatically set. If an interrupt has been enabled by setting the bit HEFC_FMR.FRDY, the corresponding interrupt line of the interrupt controller is activated.

All the commands are protected by the same keyword, which must be written in the eight highest bits of HEFC_FCR.

Writing HEFC_FCR with data that does not contain the correct key and/or with an invalid command has no effect on the whole memory plane, but the FCMDE flag is set in the Flash Status register (HEFC_FSR). This flag is automatically cleared by a read access to HEFC_FSR.

When the current command writes or erases a page in a locked region, the command has no effect on the whole memory plane, but the FLOCKE flag is set in HEFC_FSR. This flag is automatically cleared by a read access to HEFC_FSR.

Figure 19-5. Command State Chart



19.4.4.2 Get Flash Descriptor Commands

This command provides the system with information on the Flash organization. The system can take full advantage of this information. For instance, a device could be replaced by one with more Flash capacity, and so the software is able to adapt itself to the new configuration.

To get the embedded Flash descriptor, the application writes the GETD command in HEFC_FCR. The first word of the descriptor can be read by the software application in HEFC_FRR as soon as the FRDY flag in HEFC_FSR rises. The next reads of HEFC_FRR provide the following word of the descriptor. If extra read operations to HEFC_FRR are done after the last word of the descriptor has been returned, the HEFC_FRR value is 0 until the next valid command.

Table 19-1. Flash Descriptor Definition

Symbol	Word Index	Description
FL_ID	0	Flash interface description

.....continued		
Symbol	**Word Index**	**Description**
FL_SIZE	1	Flash size in bytes
FL_PAGE_SIZE	2	Page size in bytes
FL_NB_PLANE	3	Number of planes
FL_PLANE[0]	4	Number of bytes in the plane
FL_NB_LOCK	4 + FL_NB_PLANE	Number of lock bits. A bit is associated with a lock region. A lock bit is used to prevent write or erase operations in the lock region.
FL_LOCK[0]	4 + FL_NB_PLANE + 1	Number of bytes in the first lock region

19.4.4.3 Write Commands

DMA write accesses must be 32-bit aligned. If a single byte is to be written in a 32-bit word, the rest of the word must be written with ones.

Several commands are used to program the Flash:

- Write Page (WP): one page is entirely written
- Write Page then Lock (WPL): one page is entirely written, then locked

Only 0 values can be programmed using Flash technology; 1 is the erased value. In order to program words in a page, the page must first be erased. Commands are available to erase the full memory plane or a given number of pages. With the EWP and EWPL commands, a page erase is done automatically before a page programming.

After programming, the page (if it is one of the first 32) can be locked to prevent miscellaneous write or erase sequences. The lock bit can be automatically set after page programming using the WPL command.

Data to be programmed in the Flash must be written in an internal latch buffer before writing the programming command in HEFC_FCR. Data can be written at their final destination address, as the latch buffer is mapped into the Flash memory address space and wraps around within this Flash address space.

Byte and half-word AHB accesses to the latch buffer are not allowed. Only 32-bit word accesses are supported.

32-bit words must be written continuously, in either ascending or descending order. Writing the latch buffer in a random order is not permitted. This prevents mapping a C-code structure to the latch buffer and accessing the data of the structure in any order. It is instead recommended to fill in a C-code structure in SRAM and copy it in the latch buffer in a continuous order.

The HEFC generates its own clock for Flash modules and ensures proper sequencing regarding clock ratio.

The latch buffer is automatically re-initialized, that is, written with logical 1, after execution of each programming command. The programming sequence is as follows:

1. Write the data to be programmed in the latch buffer.
2. Write the programming command in HEFC_FCR. This automatically clears HEFC_FSR.FRDY.
3. When Flash programming is completed, HEFC_FSR.FRDY rises. If an interrupt has been enabled by setting HEFC_FMR.FRDY, the interrupt line of the HEFC is activated.

The following two errors can be detected in HEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.
- Lock Error: The page to be programmed belongs to a locked region. A command must be run previously to unlock the corresponding region.

Only one page can be programmed at a time. It is only possible to program all the bits of a page (full page programming).

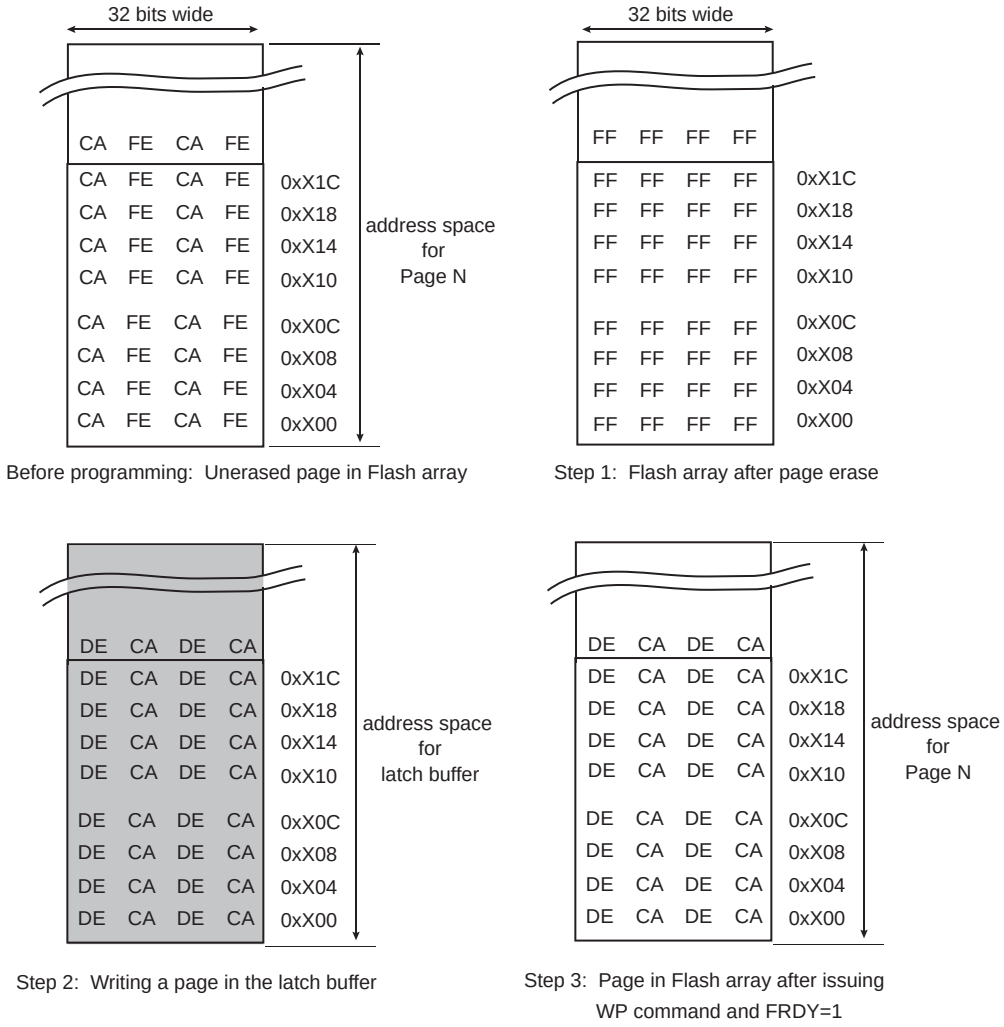
When a ‘Write Page’ (WP) command is issued, the HEFC starts the programming sequence and all the bits written at 0 in the latch buffer are cleared in the Flash memory array.

Note: During programming, that is, until HEFC_FSR.FDRY rises, access to the Flash is not allowed.

19.4.4.3.1 Full Page Programming

To program a full page, all the bits of the page must be erased before writing the latch buffer and issuing the WP command. The latch buffer must be written in ascending order, starting from the first address of the page.

Figure 19-6. Full Page Programming



19.4.4.3.2 Data Write Access into Latch Buffer

All write accesses to the latch buffer must be performed in 32-bit mode. The latch buffer must be fully written contiguously from beginning to end, prior to issuing any write commands. If no data is required at a given address in the latch buffer, then it must be written with ‘1’.

Below are examples of writing one value to latch buffers:

- 8-bit data (0xA5) at address 0x0001 in latch buffer:
  - Latch buffer must be erased first (all ‘1’)
  - Latch buffer must be filled, with a 32-bit access at offset 0x0000, Data = 0xFFFFA5FF (bit not required for data must be filled with ‘1’)
- 16-bit data (0xCAFE) at address 0x0002 in latch buffer:
  - Latch buffer must be erased first (all ‘1’)
  - Latch buffer must be filled, with a 32-bit access at offset 0x0000, Data = 0xCAFEFFFF (bit not required for data must be filled with ‘1’)
- 32-bit data (0xCAFEF00D) at address 0x0004 in latch buffer:
  - Latch buffer must be erased first (all ‘1’)

- Latch buffer must be filled, with a 32-bit access at offset 0x0004, Data = 0xCAFEF00D (all bits of the word are relevant)

Once all addresses of the latch buffer are loaded (with data or with '1'), a WP command must be launched with the page number argument to write it in Flash plane memories.

19.4.4.4 Erase Commands

Erase commands are allowed only on unlocked regions. Depending on the Flash memory, several commands can be used to erase the Flash:

- Erase All Memory (EA): All memory is erased. The processor must not fetch code from the Flash memory.
- Erase Page (EP): 1 page is erased in the Flash memory plane. The page to be erased is specified by the FARG field of the HEFC_FCR.
- Erase Pages (EPA): 4, 8, 16 or 32 pages are erased in the Flash memory plane. The first page to be erased is specified in the FARG[15:2] field of the HEFC_FCR and must be a multiple of 4, 8, 16 or 32 depending on the number of pages to erase at the same time.

If the processor is fetching code from the Flash memory while the EPA command is being executed, the processor accesses are stalled until the EPA command is completed. To avoid stalling the processor, the code can be run out of internal SRAM.

The erase sequence is the following:

1. Erase starts as soon as one of the erase commands and the FARG field are written in HEFC_FCR. For the EPA command, the two lowest bits of the FARG field define the number of pages to be erased (FARG[1:0]):

Table 19-2. HEFC_FCR.FARG Field for EPA Command

FARG[1:0]	Number of pages to be erased with EPA command
0	4 pages
1	8 pages
2	16 pages
3	32 pages

2. When erasing is completed, the bit HEFC_FSR.FRDY rises. If an interrupt has been enabled by setting the bit HEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.

Two errors can be detected in HEFC_FSR after an erasing sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.
- Lock Error: At least one page to be erased belongs to a locked region. The erase command has been refused, no page has been erased. A command must be run previously to unlock the corresponding region.

19.4.4.5 Lock Bit Protection

Each Lock bit is associated with one page in the embedded Flash memory plane. Only the 32 first pages are lockable by using the 32 available lock bits. Thus, one lock region is defined as one lock page in the embedded Flash memory plane. These regions prevent writing/erasing protected pages.

The following are lock bit commands:

- Set Lock Bit (SLB) – to lock a page
- Clear Lock Bit (CLB) – to unlock a page
- Get Lock Bit (CLB) – to get the status of a page

The following are lock sequence:

1. Execute the 'Set Lock Bit' command by writing HEFC_FCR.FCMD with the SLB command and HEFC_FCR.FARG with a page number to be protected.
2. When the locking completes, the bit HEFC_FSR.FRDY rises. If an interrupt has been enabled by setting HEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.
3. The result of the SLB command can be checked by running a 'Get Lock Bit' (GLB) command.

Note: The value of the FARG argument passed together with the SLB command must not exceed the higher lock bit index available in the product.

The following error may be detected in HEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.

It is possible to clear lock bits previously set. After the lock bits are cleared, the locked region can be erased or programmed. The unlock sequence is the following:

1. Execute the 'Clear Lock Bit' command by writing HEFC_FCR.FCMD with the CLB command and HEFC_FCR.FARG with a page number to be unprotected.
2. When the unlock operation completes, the bit HEFC_FSR.FRDY rises. If an interrupt has been enabled by setting HEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.

Note: The value of the FARG argument passed together with the CLB command must not exceed the higher lock bit index available in the product.

The following error may be detected in HEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.

The status of lock bits can be returned by the HEFC. The 'Get Lock Bit' sequence is the following:

1. Execute the 'Get Lock Bit' command by writing HEFC_FCR.FCMD with the GLB command. HEFC_FCR.FARG is meaningless.
2. Lock bits can be read by the software application in HEFC_FRR. The word read corresponds to the 32 lock bits. Extra reads to HEFC_FRR return 0.

For example, if the third bit of the first word read in HEFC_FRR is set, the third lock region is locked.

The following error may be detected in HEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.

Note: Once locked, pages are no longer accessible even in read (0xFFFFFFFF is read) .

19.4.4.6 GPNVM Bits

GPNVM bits do not interfere with the embedded Flash memory plane.

The GPNVM bits commands are the following one:

- Set GPNVM Bit (SGPB) – used to set a GPNVM bit
- Clear GPNVM Bit (CGPB) – used to clear a GPNVM bit
- Get GPNVM Bit (GGPB) – used to get the status of a GPNVM bit

The 'Set GPNVM Bit' sequence is the following:

1. Execute the 'Set GPNVM Bit' command by writing HEFC_FCR.FCMD with the SGPB command and HEFC_FCR.FARG with the number of GPNVM bits to be set.
2. When the GPNVM bit is set, the bit HEFC_FSR.FRDY rises. If an interrupt was enabled by setting the bit HEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.
3. The result of the SGPB command can be checked by running a 'Get GPNVM Bit' (GGPB) command.

Note: The value of the FARG argument passed together with the SGPB command must not exceed the higher GPNVM index available in the product. Flash data content is not altered if FARG exceeds the limit. Command Error is detected only if FARG is greater than 128.

The following error may be detected in HEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.

It is possible to clear GPNVM bits previously set. The 'Clear GPNVM Bit' sequence is the following:

1. Execute the 'Clear GPNVM Bit' command by writing HEFC_FCR.FCMD with the CGPB command and HEFC_FCR.FARG with the number of GPNVM bits to be cleared.
2. When the clear completes, the bit HEFC_FSR.FRDY rises. If an interrupt has been enabled by setting the bit HEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.

Note: The value of the FARG argument passed together with the CGPB command must not exceed the higher GPNVM index available in the product. Flash data content is not altered if FARG exceeds the limit. Command Error is detected only if FARG is greater than 128.

The following error may be detected in HEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.

The status of GPNVM bits can be returned by the HEFC. The sequence is the following:

1. Execute the 'Get GPNVM Bit' command by writing HEFC_FCR.FCMD with the GGPB command. Field HEFC_FCR.FARG is meaningless.
2. GPNVM bits can be read by the software application in HEFC_FRR. The first word read corresponds to the 32 first GPNVM bits; the reads that follow provide the next 32 GPNVM bits as long as it is meaningful. Extra reads to HEFC_FRR return 0.

For example, if the third bit of the first word read in HEFC_FRR is set, the third GPNVM bit is active.

The following error may be detected in HEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.

Note: Access to the Flash in read is permitted when a 'Set GPNVM Bit', 'Clear GPNVM Bit' or 'Get GPNVM Bit' command is executed.

Refer to General-purpose NVM (GPNVM) bits for a complete list of GPNVM bits and their functions.

19.4.4.7 User Signature Area

Each product contains a user signature area of 256 bytes of data (and 128 bytes of ECC). It can be used for storage. See the figure "Flash Memory Areas".

Write, Erase and Read the user signature area is allowed using the following commands:

- Write User Signature (WUS)
- Erase User Signature (EUS)
- Start Read User Signature (STUS)
- Stop Read User Signature (SPUS)

The sequence to write the user signature area is the following:

- Fill the write latch buffer, within the internal memory area address space.
- Execute the 'Write User Signature' command by writing HEFC_FCR.FCMD with the WUS command. Field HEFC_FCR.FARG is meaningless.
- When programming is completed, HEFC_FSR.FRDY rises. If an interrupt has been enabled by setting HEFC_FMR.FRDY, the corresponding interrupt line of the interrupt controller is activated.

The following error may be detected in HEFC_FSR after this sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.

The sequence to erase the user signature area is the following:

1. Execute the 'Erase User Signature' command by writing HEFC_FCR.FCMD with the EUS command. Field HEFC_FCR.FARG is meaningless.
2. When programming is completed, HEFC_FSR.FRDY rises. If an interrupt has been enabled by setting HEFC_FMR.FRDY, the corresponding interrupt line of the interrupt controller is activated.

The following error may be detected in HEFC_FSR after this sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.

The sequence to read the user signature area is the following:

1. Execute the 'Start Read User Signature' command by writing HEFC_FCR.FCMD with the STUS command. Field HEFC_FCR.FARG is meaningless.
2. Wait until HEFC_FSR.FRDY falls to read the user signature area. The user signature area is located in the first 256 bytes of the Flash memory mapping. The 'Start Read User Signature' command reuses some addresses of the memory plane but the user signature area is physically different from the memory plane.

3. To stop reading the user signature area, execute the ‘Stop Read User Signature’ command by writing HEFC_FCR.FCMD with the SPUS command. Field HEFC_FCR.FARG is meaningless.
4. When the SPUS command has been executed, HEFC_FSR.FRDY rises. If an interrupt was enabled by setting HEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.

Note that during the sequence, the software cannot be fetched from the Flash memory plane.

The following error may be detected in HEFC_FSR after this sequence:

- Command Error: A bad keyword has been written in HEFC_FCR.

19.4.5 Flash Power Management

The embedded Flash plane of 128 Kbytes is made of 4 Flash submodules of 32 Kbytes. The HEFC embeds a full power management mechanism that is partially configurable by the end user.

19.4.5.1 Initial Power-Up or Global Reset

To minimize the contribution of the power-up of Flash modules on the inrush current at initial power-up, the sequence described below is automatically applied:

Figure 19-7. Flash Modules Power-Up Sequence – [System]

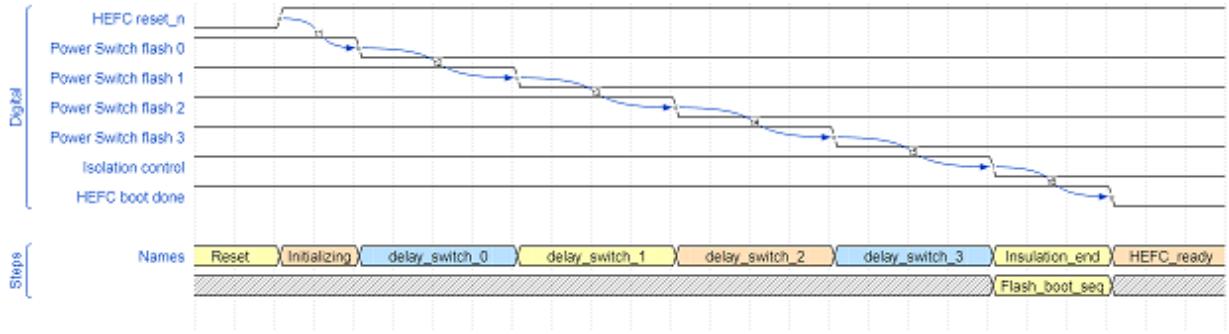


Table 19-3. Sequence Times (Flash Module Power-up Sequence Only)

Time	Duration	Description
t1	Some system clock periods	Once the HEFC global reset is released, logic starts to operate. After some initialization steps, the power switches sequence is launched.
t2	> 750 µs	Power switch 0 startup time.
t3	> 750 µs	Power switch 1 startup time.
t4	> 750 µs	Power switch 2 startup time.
t5	> 750 µs	Power switch 3 startup time.
t6	> 2100 µs	Once all Flash modules are started, the HEFC releases Flash signal insulation. A specific boot-up sequence is then applied on the Flash modules. At the end of this sequence, the HEFC and its Flash modules are ready to use.

Note: By default, the delay applied for power switch start-up phase is around 750 µs. The HEFC ensures this delay to be > 750 µs.

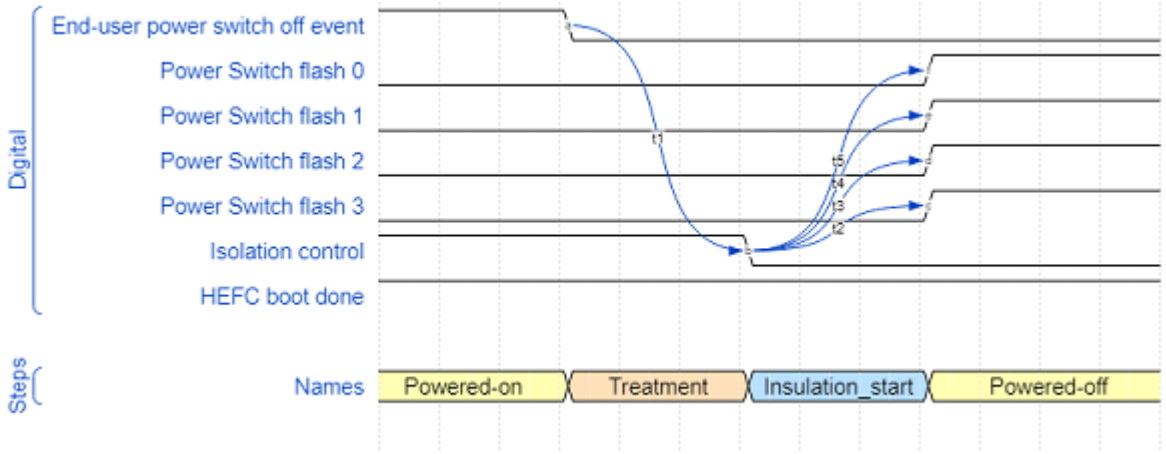
19.4.5.2 End User Power-Down

During the application run phase, the HEFC offers the possibility to the end user to power switched-off Flash modules when not used by clearing HEFC_FPMR.PWS_EN. Powering down Flash modules ensures a better data reliability in the context of space applications, as powered-down Flash modules are more immune to radiation effects.

Prior to being switched off and in order to not have any impact on the running system, Flash modules are insulated.

The timing diagram below shows the sequence applied on power-off request event.

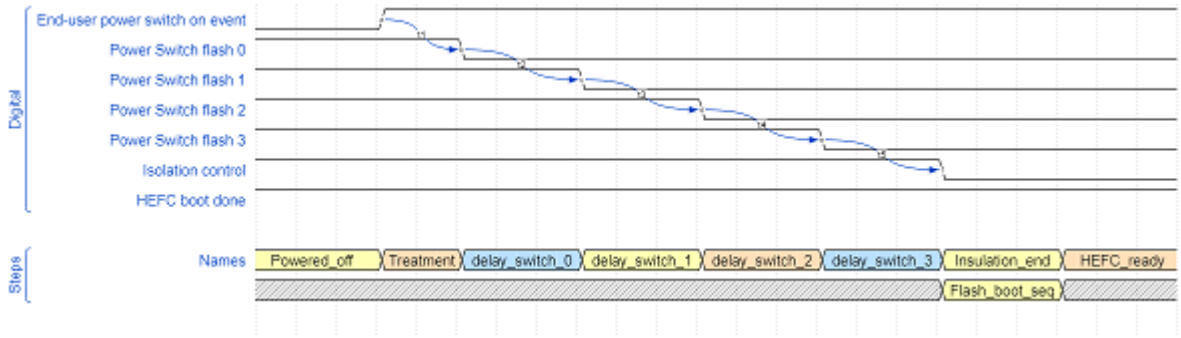
Figure 19-8. Flash Modules Power-Down Sequence – [End user]



19.4.5.3 End User Power-Up

Prior to powering up the Flash modules, the end user can configure the power switch delay applicable for each one. (See HEFC_FPMR.PWS_DLY for available values). Depending on this value, the end user must also properly set the value of HEFC_FPMR.VAR_FACTOR to handle the time required to set up the system clock. The end user power-up sequence is activated by setting HEFC_FPMR.PWS_EN.

Figure 19-9. Flash Modules Power-Up Sequence – [End user]



19.4.6 HECC Management

19.4.6.1 Overview

The Hardened Error Code Correction (HECC) is a module embedded to protect data against radiation effects. It is embedded next to the Hardened Flash Controller (HEFC).

19.4.6.2 Features

- BCH (32,12) Code + Extra Parity Code Embedded
  - To correct up to 2 errors
  - To detect up to 3 errors
- Interrupt Capabilities
  - 1 interrupt for unfixable error
  - 1 interrupt for fixable error
- Error Counter Available and for Statistics Reporting
  - 1 counter dedicated to fixable errors,
  - 1 counter dedicated to unfixable errors.
- Embedded HECC Testing Mode

19.4.6.3 Product Dependencies

19.4.6.3.1 Clocks Management

The Hardened Error Checker / Corrector (HECC) channels' encoders / decoders are fully asynchronous. Interface, control and status modules are system clock dependent. The HECC clock is provided by PMC module. Therefore, it shall be programmed first.

19.4.6.3.2 Interrupt Sources

The Hardened Error Checker / Corrector (HECC) handles fixable and unfixable errors. These two kinds of errors can generate their own interrupt. Using the HECC interrupt requires the interrupt controller to be programmed first.

19.4.6.4 Functional Description

19.4.6.4.1 Functional Block

The HECC block is made of:

- 1 encoder/decoder BCH(32,12)) + an extra parity bit
- An APB end user interface for configuration:
  - to enable/disable the HECC protection
  - to activate/deactivate the HECC test mode (independently in read or write mode),
  - to reset the HECC fixable and unfixable counters,
- An APB end user interface for errors status and reporting:
  - Indicate whether fixable error(s) has been detected and corrected on-the-fly, and quantifies them,
  - To indicate whether unfixable error(s) has been detected and quantifies them,
  - The indication can be done either by interruption or by polling,
  - To indicate the address and the status (type, size and origin) of the last HECC error detected.
- 4 channels to access the 4 Flash memories, which constitute the whole plane. (Only the channel 0 is relevant to set parameters, others channels (1 to 3) use the same parameters as channel 0).

19.4.6.4.2 Operation

When enabled (HECC_CR_X.ENABLE='1' and HECC_CR_X.ECC12_ENABLE='1'), the HECC operates on every access to the memories.

Nevertheless, the HECC detection and correction is performed only on read/fetch accesses, by comparing the HECC checksum stored in memory to the one locally computed from data read.

19.4.6.4.3 BCH (32,12) Code + extra parity bit

For each word, a 13-bit (12bit of checksum + 1 extra parity bit) is generated. With this correcting code, up to 2-bit error can be corrected (named fixable error later on). Up to 3-bit errors can be detected (named unfixable error later on).

19.4.6.4.4 Write Access

When the processor performs a write access to an HECC protected memory, it also writes the 13-bit checksum on the HECC protected memory. Other bits are driven low unless HECC testing is enabled.

19.4.6.4.5 Read Access

When the processor performs a read access to an HECC protected memory, it reads data and its associated checksum. The checksum read from HECC protected memory is compared against the checksum generated by the HECC from the same read data. Any discrepancy yields an error and a syndrome is computed to further qualify the error as fixable error or unfixable error.

19.4.6.4.6 Fixable Error

The correction is performed on-the-fly inside the processor during the current access and no timing penalty is incurred.

The fixable error event is reported in the Fail Address register (HECC_FAILAR) and in the Fail Status register (HECC_SR). If previously enabled, a fixable interrupt (HECC_ISR.FIX) is generated.

The fixable error remains in memory until a software-initiated rewrite is performed at the faulty memory location.

To permit later statistical study, a fixable error counter is available. It is incremented each time a fixable error is detected and until a counter overload is detected.

It is possible for the end-user to reset the fixable counter when it is needed.

19.4.6.4.7 Unfixable Error

The unfixable error event is reported in the Fail Address register (HECC_FAILAR) and in the Fail Status register (HECC_SR). If previously enabled, an unfixable interrupt (HECC_ISR.NOFIX) is generated.

To permit later statistical study, an unfixable error counter is available. It is incremented each time an unfixable error is detected and until a counter overload is detected.

It is possible for the end-user to reset the unfixable counter when it is needed.

Note: If more than three errors are detected, the BCH decoder with its extra parity bit cannot handle them properly (either fixable or unfixable error will be unpredictively raised).

19.4.6.4.8 Status Report

On error occurrences, following elements are stored in coordination with either **HECC_SR.MEM_FIX** or **HECC_SR.MEM_NOFIX** flags:

- **HECC_FAILAR** stores the failed address,
- **HECC_SR.MEM_ID** stores the bank identification,
- **HECC_SR.TYPE** stores the access type,
- **HECC_SR.HES** store the access size.

While not read, **HECC_SR** locks error flags and previous listed elements. Prior to reading **HECC_SR**, it is required to read **HECC_FAILAR** to get the failed address.

Reading the **HECC_SR** results in clearing pending flags.**HECC_FAILAR** as well as previously listed elements' contents become irrelevant.

19.4.6.4.9 HECC Testing

Operation of the HECC can be bypassed for testing purpose and is controlled in an HECC Configuration register (HECC_CR).

19.4.6.4.10 Write Test

When HECC write bypass is enabled (HECC_CR.TEST_MODE_WR = 1), the test checksum (HECC_TESTCB.tcb) replaces the HECC generated checksum during a data store.

19.4.6.4.11 Read Test

When HECC read diagnostic is enabled (HECC_CR.TEST_MODE_RD = 1), the test checksum (HECC_TESTCB.TCB) are updated with the read checksum during a data load or an instruction fetch.

Note: Parity is taken into account for error detection but only a 12-bit checksum is reported in HECC_TESTCB_X.TCB field.

19.4.7 Register Write Protection

To prevent any single software error from corrupting HEFC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the (HEFC_WPMR).

The following registers can be write-protected:

- HEFC Flash Mode Register
- HEFC Flash Power Management Register

19.5 Register Summary

Offset	Name	Bit Pos.								
0x00	HEFC_FMR	7:0							FPSI	FRDY
		15:8								
		23:16								ONE
		31:24								
0x04	HEFC_FCR	7:0	FCMD[7:0]							
		15:8	FARG[7:0]							
		23:16	FARG[15:8]							
		31:24	FKEY[7:0]							
0x08	HEFC_FSR	7:0				WREER		FLOCKE	FCMDE	FRDY
		15:8								
		23:16								
		31:24								
0x0C	HEFC_FRR	7:0	FVALUE[7:0]							
		15:8	FVALUE[15:8]							
		23:16	FVALUE[23:16]							
		31:24	FVALUE[31:24]							
0x10 ... 0x3F	Reserved									
0x40	HEFC_FPMR	7:0					PWS_DLY[1:0]		PWS_STAT	PWS_EN
		15:8			VAR_FACTOR[5:0]					
		23:16								FUNC_ISOL_CTRL_N
		31:24								
0x44 ... 0xE3	Reserved									
0xE4	HEFC_WPMR	7:0				USER	ERASEWP	LOCKWP	GPNVMWP	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0xE8 ... 0xFF	Reserved									
0x0100	HEFC_HECC_CR0	7:0		ECC12_ENABLE	RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ENABLE
		15:8								
		23:16								
		31:24								
0x0104	HEFC_HECC_CR1	7:0		ECC12_ENABLE	RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ENABLE
		15:8								
		23:16								
		31:24								
0x0108	HEFC_HECC_CR2	7:0		ECC12_ENABLE	RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ENABLE
		15:8								
		23:16								
		31:24								
0x010C	HEFC_HECC_CR3	7:0		ECC12_ENABLE	RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ENABLE
		15:8								
		23:16								
		31:24								
0x0110 ... 0x013F	Reserved									

.....continued

Offset	Name	Bit Pos.								
0x0140	HEFC_HECC_TESTCB0	7:0	TCB[7:0]							
		15:8	TCB[15:8]							
		23:16								
		31:24								
0x0144	HEFC_HECC_TESTCB1	7:0	TCB[7:0]							
		15:8	TCB[15:8]							
		23:16								
		31:24								
0x0148	HEFC_HECC_TESTCB2	7:0	TCB[7:0]							
		15:8	TCB[15:8]							
		23:16								
		31:24								
0x014C	HEFC_HECC_TESTCB3	7:0	TCB[7:0]							
		15:8	TCB[15:8]							
		23:16								
		31:24								
0x0150 ... 0x017F	Reserved									
0x0180	HEFC_HECC_SR	7:0	OVER_FIX	CPT_FIX[4:0]						MEM_FIX
		15:8	OVER_NOFIX	CPT_NOFIX[4:0]						MEM_NOFIX
		23:16								
		31:24			MEM_ID[1:0]		TYPE	HES[2:0]		
0x0184	HEFC_HECC_IER	7:0							MEM_NOFIX	MEM_FIX
		15:8								
		23:16								
		31:24								
0x0188	HEFC_HECC_IDR	7:0							MEM_NOFIX	MEM_FIX
		15:8								
		23:16								
		31:24								
0x018C	HEFC_HECC_IMR	7:0							MEM_NOFIX	MEM_FIX
		15:8								
		23:16								
		31:24								
0x0190	HEFC_HECC_FAILAR	7:0	ADDRESS[7:0]							
		15:8	ADDRESS[15:8]							
		23:16	ADDRESS[23:16]							
		31:24	ADDRESS[31:24]							

19.5.1 HEFC Flash Mode Register

Name: HEFC_FMR
Offset: 0x00
Reset: 0x00010400
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the “HEFC Write Protection Mode Register” .

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
								ONE
Access								R/W
Reset								1

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							FPSI	FRDY
Access							R/W	R/W
Reset							0	0

Bit 16 – ONE Must be written to 1

Bit 1 – FPSI Flash Power Status Interrupt Enable

Value	Description
0	Flash Power Status Enable does not generate an interrupt.
1	Flash Power Status Enable generates an interrupt.

Bit 0 – FRDY Flash Ready Interrupt Enable

Value	Description
0	Flash ready does not generate an interrupt.
1	Flash ready (to accept a new command) generates an interrupt.

19.5.2 HEFC Flash Command Register

Name: HEFC_FCR
Offset: 0x04
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	FKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	FARG[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	FARG[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	FCMD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 31:24 – FKEY[7:0] Flash Write Protection Key

Value	Name	Description
0x5A	PASSWD	The 0x5A value enables the command defined by the bits of the register. If the field is written with a different value, the write is not performed and no action is started.

Bits 23:8 – FARG[15:0] Flash Command Argument

GETD, GLB, GGPB, GCALB, WUS, EUS, STUS, SPUS, EA	Commands requiring no argument, including Erase all command	FARG is meaningless, must be written with 0
EPA	Erase pages command	FARG[1:0] defines the number of pages to be erased The start page must be written in FARG[15:2]. FARG[1:0] = 0: Four pages to be erased. FARG[15:2] = Page_Number / 4 FARG[1:0] = 1: Eight pages to be erased. FARG[15:3] = Page_Number / 8, FARG[2]=0 FARG[1:0] = 2: Sixteen pages to be erased. FARG[15:4] = Page_Number / 16, FARG[3:2]=0 FARG[1:0] = 3: Thirty-two pages to be erased. FARG[15:5] = Page_Number / 32, FARG[4:2]=0 Refer to the table “HEFC_FCR.FARG Field for EPA Command”.
WP, WPL, EP	Programming commands	FARG must be written with the page number to be programmed

SLB, CLB	Lock bit commands	FARG defines the page number to be locked or unlocked
SGPB, CGPB	GPNVM commands	FARG defines the GPNVM number to be programmed

Bits 7:0 – FCMD[7:0] Flash Command

Value	Name	Description
0x00	GETD	Get Flash descriptor
0x01	WP	Write page
0x02	WPL	Write page and lock
0x05	EA	Erase all
0x06	EP	Erase page
0x07	EPA	Erase pages
0x08	SLB	Set lock bit
0x09	CLB	Clear lock bit
0x0A	GLB	Get lock bit
0x0B	SGPB	Set GPNVM bit
0x0C	CGPB	Clear GPNVM bit
0x0D	GGPB	Get GPNVM bit
0x10	GCALB	Get CALIB bit
0x12	WUS	Write user signature
0x13	EUS	Erase user signature
0x14	STUS	Start read user signature
0x15	SPUS	Stop read user signature

19.5.3 HEFC Flash Status Register

Name: HEFC_FSR
Offset: 0x08
Reset: 0x00000001
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				WREER		FLOCKE	FCMDE	FRDY
Access				R		R	R	R
Reset				0		0	0	1

Bit 4 – WREER Write Register Error Status (cleared on read)
This flag is automatically cleared when HEFC_FSR is read or HEFC_FCR is written.

Value	Description
0	No register writing error has happened since the last read of HEFC_FSR.
1	Register writing error has happened since the last read of HEFC_FSR.

Bit 2 – FLOCKE Flash Lock Error Status (cleared on read)
This flag is automatically cleared when HEFC_FSR is read or HEFC_FCR is written.

Value	Description
0	No programming/erase of at least one locked region has happened since the last read of HEFC_FSR.
1	Programming/erase of at least one locked region has happened since the last read of HEFC_FSR.

Bit 1 – FCMDE Flash Command Error Status (cleared on read or by writing HEFC_FCR)

Value	Description
0	No invalid commands and no bad keywords were written in HEFC_FMR.
1	An invalid command and/or a bad keyword was/were written in HEFC_FMR.

Bit 0 – FRDY Flash Ready Status (cleared when Flash is busy)
When set, this flag triggers an interrupt if the FRDY flag is set in HEFC_FMR.
This flag is automatically cleared when the HEFC is busy.

Value	Description
0	The HEFC is busy.
1	The HEFC is ready to start a new command.

19.5.4 HEFC Flash Result Register

Name: HEFC_FRR
Offset: 0x0C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	FVALUE[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	FVALUE[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FVALUE[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FVALUE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – FVALUE[31:0] Flash Result Value
The result of a Flash command is returned in this register. If the size of the result is greater than 32 bits, the next resulting value is accessible at the next register read.

19.5.5 HEFC Flash Power Management Register

Name: HEFC_FPMR
Offset: 0x40
Reset: 0x0001320B
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the HEFC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
								FUNC_ISOL_CTRL_N
Access								R/W
Reset								1

Bit	15	14	13	12	11	10	9	8
			VAR_FACTOR[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	0	0	1	0

Bit	7	6	5	4	3	2	1	0
					PWS_DLY[1:0]		PWS_STAT	PWS_EN
Access					R/W	R/W	R/W	R/W
Reset					1	0	1	1

Bit 16 – FUNC_ISOL_CTRL_N Flash Insulation Control Status

Value	Description
0	Flash memory is insulated from the whole product.
1	Flash memory is connected to the whole product.

Bits 13:8 – VAR_FACTOR[5:0] Variation Factor
This field is a ratio to adapt the power switch delay versus the end user frequency.
The allowable range is 0 to 63.
VAR_FACTOR= 10 * (Freq_MCK / Freq_10MHZ)
As a consequence, the delay between the two power switches is t(freq_MCK,VAR_FACTOR)
= t(PWS_DELAY(10 MHz, 10)*VAR_FACTOR/freq_MCK).

Bits 3:2 – PWS_DLY[1:0] Power Switch Delay
This field specifies the delay to apply between each power switch (reference conditions RC=10 MHz, VAR_FACTOR=10).

Value	Name	Description
0	75US	Delay is set to 75 µs (RC=10 MHz, VAR_FACTOR=10)
1	150US	Delay is set to 150 µs (RC=10 MHz, VAR_FACTOR=10)
2	300US	Delay is set to 300 µs (RC=10 MHz, VAR_FACTOR=10)
3	600US	Delay is set to 600 µs (RC=10 MHz, VAR_FACTOR=10)

Bit 1 – PWS_STAT Power Switch Status

Value	Description
0	Power switch is off or power-up sequence is ongoing.
1	Power-up sequence is done.

Bit 0 – PWS_EN Power Switch Enable

Value	Description
0	Power switch is off.
1	Power switch is on.

19.5.6 HEFC Write Protection Mode Register

Name: HEFC_WPMR
Offset: 0xE4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				USER	ERASEWP	LOCKWP	GPNVMWP	WPEN
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x454643	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

Bit 4 – USER User Signature Write Protection

Value	Description
0	Write and Erase User signature commands selected by HEFC_FCR.FCMD are enabled.
1	Write and Erase User signature commands selected by HEFC_FCR.FCMD are disabled.

Bit 3 – ERASEWP Page, Sector and Plane Write Protection

Value	Description
0	Write and Erase Page commands selected by HEFC_FCR.FCMD are enabled.
1	Write and Erase Page commands selected by HEFC_FCR.FCMD are disabled.

Bit 2 – LOCKWP Lock Bit Write Protection

Value	Description
0	Set and Clear lock bit commands selected by HEFC_FCR.FCMD are enabled.
1	Set and Clear lock bit commands selected by HEFC_FCR.FCMD are disabled.

Bit 1 – GPNVMWP GPNVM Bit Write Protection

Value	Description
0	Set and Clear GPNVM commands selected by HEFC_FCR.FCMD are enabled.
1	Set and Clear GPNVM commands selected by HEFC_FCR.FCMD are disabled.

Bit 0 – WPEN Write Protection Enable
See Section 5.7 ”Flash Power Switch Control and Status” for the list of registers that can be protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x454643 (EFC in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x454643 (EFC in ASCII).

19.5.7 HECC Control Register 0

Name: HEFC_HECC_CR0
Offset: 0x100
Reset: 0x00000041
Property: Read/Write

The configuration of HEFC_HECC_CR0 is duplicated in all HEFC_HECC_CRx.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		ECC12_ENABLE	RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ENABLE
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		1	0	0		0	0	1

Bit 6 – ECC12_ENABLE BCH ECC Enable

Value	Description
0	8-bit ECC is enabled if ENABLE is at ‘0’.
1	12-bit ECC is enabled if ENABLE is at ‘1’. (only supported)

Bit 5 – RST_NOFIX_CPT Reset the Unfixable Error Counter

Value	Description
0	No effect.
1	Resets the ECC unfixable error counter.

Bit 4 – RST_FIX_CPT Reset the Fixable Error Counter

Value	Description
0	No effect.
1	Resets the ECC fixable error counter.

Bit 2 – TEST_MODE_WR Test Mode of ECC Protection - Write Mode
The ENABLE bit must be enabled to use this bit.

Value	Description
0	Test mode is deactivated.
1	Test mode is activated.

Bit 1 – TEST_MODE_RD Test Mode of ECC Protection - Read Mode
The ENABLE bit must be enabled to use this bit.

Value	Description
0	Test mode is deactivated.
1	Test mode is activated.

Bit 0 – ENABLE ECC Protection Enable

Value	Description
0	Protection is deactivated.
1	Protection is activated.

19.5.8 HECC Control Register x [x=1..3]

Name: HEFC_HECC_CRx
Offset: 0x0104 + (x-1)*0x04 [x=1..3]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		ECC12_ENABLE	RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	ENABLE
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit 6 – ECC12_ENABLE BCH ECC Enable

Value	Description
0	8-bit ECC is enabled if ENABLE is at ‘0’.
1	12-bit ECC is enabled if ENABLE is at ‘1’. (only supported)

Bit 5 – RST_NOFIX_CPT Reset the Unfixable Error Counter

Value	Description
0	No effect.
1	Resets the ECC unfixable error counter.

Bit 4 – RST_FIX_CPT Reset the Fixable Error Counter

Value	Description
0	No effect.
1	Resets the ECC fixable error counter.

Bit 2 – TEST_MODE_WR Test Mode of ECC Protection - Write Mode
The ENABLE bit must be enabled to use this bit.

Value	Description
0	Test mode is deactivated.
1	Test mode is activated.

Bit 1 – TEST_MODE_RD Test Mode of ECC Protection - Read Mode
The ENABLE bit must be enabled to use this bit.

Value	Description
0	Test mode is deactivated.
1	Test mode is activated.

Bit 0 – ENABLE ECC Protection Enable

Value	Description
0	Protection is deactivated.
1	Protection is activated.

19.5.9 HECC Test Mode Register x

Name: HEFC_HECC_TESTCBx
Offset: 0x0140 + x*0x04 [x=0..3]
Reset: 0x00000000
Property: Read/Write

The configuration of HEFC_HECC_CR0 is duplicated in the other HEFC_HECC_CRx.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TCB[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TCB[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – TCB[15:0] Test Check Bit (16 bits)
Check bit for testing purposes.
For BCH code, the checksum is made of the parity bit, located at TCB[12], and checkbit data, located at TCB[11:0].

19.5.10 HECC Status Register

Name: HEFC_HECC_SR
Offset: 0x180
Reset: 0x02000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
			MEM_ID[1:0]		TYPE	HES[2:0]		
Access			R	R	R	R	R	R
Reset			0	0	0	0	1	0

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	OVER_NOFIX	CPT_NOFIX[4:0]						MEM_NOFIX
Access	R	R	R	R	R	R		R
Reset	0	0	0	0	0	0		0

Bit	7	6	5	4	3	2	1	0
	OVER_FIX	CPT_FIX[4:0]						MEM_FIX
Access	R	R	R	R	R	R		R
Reset	0	0	0	0	0	0		0

Bits 29:28 – MEM_ID[1:0] Memory Identification Number
Indicates to which memory the error refers.

Value	Description
0	Memory 0
1	Memory 1
2	Memory 2
3	Memory 3

Bit 27 – TYPE Write or Read Access

Value	Description
0	Write access.
1	Read access.

Bits 26:24 – HES[2:0] Hardware Error Size
Identifies the size of the failed access.

Bit 15 – OVER_NOFIX Counter Overflow

Value	Description
0	No overflow has occurred since the last read.
1	One overflow has occurred since the last read.

Bits 14:10 – CPT_NOFIX[4:0] 5-bit Counter
Counts the number of unfixable errors detected for one type of memory during the mission.

Bit 8 – MEM_NOFIX Unfixable Error Status
An unfixable error has been detected.

Bit 7 – OVER_FIX Counter Overflow

Value	Description
0	No overflow has occurred since the last read.
1	One overflow has occurred since the last read.

Bits 6:2 – CPT_FIX[4:0] 5-bit Counter
Counts the number of fixable errors detected for one type of memory during the mission.

Bit 0 – MEM_FIX Fixable Error Status
A fixable error has been detected.

19.5.11 HECC Interrupt Enable Register

Name: HEFC_HECC_IER
Offset: 0x184
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							MEM_NOFIX	MEM_FIX
Access							W	W
Reset							–	–

Bit 1 – MEM_NOFIX Unfixable Error

Bit 0 – MEM_FIX Fixable Error

19.5.12 HECC Interrupt Disable Register

Name: HEFC_HECC_IDR
Offset: 0x188
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							MEM_NOFIX	MEM_FIX
Access							W	W
Reset							–	–

Bit 1 – MEM_NOFIX Unfixable Error

Bit 0 – MEM_FIX Fixable Error

19.5.13 HECC Interrupt Mask Register

Name: HEFC_HECC_IMR
Offset: 0x18C
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							MEM_NOFIX	MEM_FIX
Access							R	R
Reset							0	0

Bit 1 – MEM_NOFIX Unfixable error

Bit 0 – MEM_FIX Fixable error

19.5.14 HECC Fail Address register

Name: HEFC_HECC_FAILAR
Offset: 0x190
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ADDRESS[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDRESS[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDRESS[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDRESS[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDRESS[31:0] Address of the Error Detected
The address of the faulty data detected when a single or multiple error occurs.

20. Supply Controller (SUPC)

20.1 Description

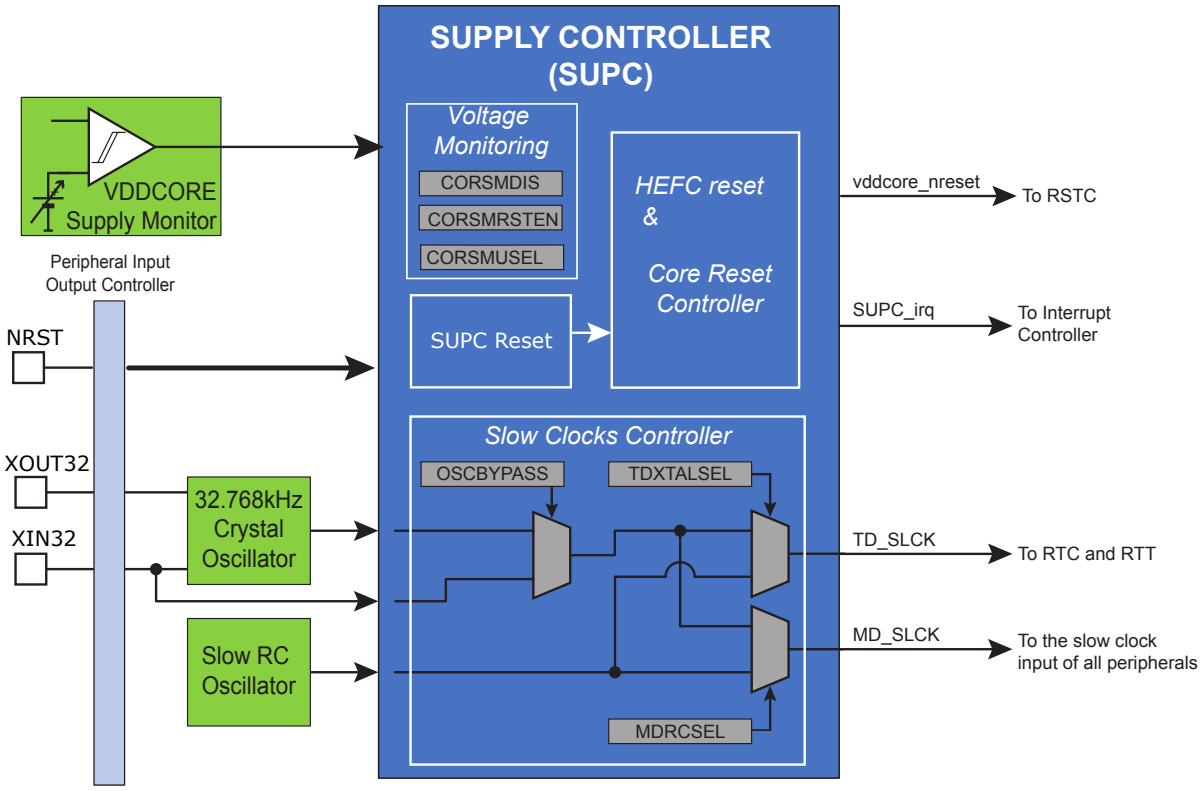
The Supply Controller (SUPC) is in charge of sequencing the internal circuits (oscillators, etc.) at device power-up, generating the slow clocks (MD_SLCK and TD_SLCK) and managing the Supply Monitor.

20.2 Embedded Characteristics

- Supply Monitor Detection on VDDCORE to Trigger a Core Reset
- Slow Clock Generation:
  - MD_SLCK — Monitoring domain slow clock
  - TD_CLK — Timing domain slow clock

20.3 Block Diagram

Figure 20-1. Supply Controller Block Diagram



20.4 Functional Description

20.4.1 Overview

From a power domain perspective, the device is classified as the following two power areas:

- VDDIO – Includes powered I/O pads and Flash memories

- VDDCORE –Powers the entire circuit, including logic and embedded memories

The SUPC includes the following functions:

- Internal circuits (oscillators and so on) sequencing at power-up
- Slow clock generation
- Power supply monitoring

At power-up (VDDIO and VDDCORE rise), the SUPC is maintained in reset until VDDCORE rises to its nominal value and the NRST pin is released. Once released from reset, the SUPC sequentially releases the reset of HEFC. Then, when the power-up sequence of the HEFC is completed, the release of the core logic reset signal (vddcore_nreset) is performed.

The SUPC slow clock generator provides the system with:

- TD_SLCK (Timing Domain Slow Clock) – This clock, generally sourced from the 32.768 kHz crystal oscillator, is routed to the RTC and RTT peripherals.
- MD_SLCK (Monitoring Domain Slow Clock) –This clock, generally sourced from the Slow RC oscillator, feeds safety-critical functions of the device (WDT, RSTC, SUPC, frequency monitors and detectors, PMC start-up time counters).

These clocks can be sourced either from the 32.768 kHz crystal oscillator or from the slow RC oscillator. At power-up, they both default to the slow RC oscillator.

The Supply Monitoring section of the SUPC includes a programmable supply monitor on VDDCORE. The output can either generate a software interrupt or a reset of the core logic.

At power-down, as soon as VDDCORE falls below the Supply Monitor threshold , the SUPC is reset. This leads to a complete loss of the SUPC configuration (slow clock sources, supply monitoring functions, etc.). In Run mode, the vddcore_nreset signal is immediately asserted.

20.4.2 Slow Clock Generator

As soon as VDDIO is supplied, both the 32.768 kHz crystal oscillator and the slow RC oscillator are powered. The slow RC oscillator is always enabled. The slow RC oscillator provides a faster start-up time than the 32.768 kHz crystal oscillator.

The user can select the 32.768 kHz crystal oscillator to be the source of the TD_SLCK, as it provides a more accurate frequency than the slow RC oscillator. The 32.768 kHz crystal oscillator is selected by setting the TDXTALSEL bit in the SUPC Control register (SUPC_CR).

For MD_SLCK, the slow RC oscillator is the only available clock source.

The following sequences must be followed to switch from the slow RC oscillator to the 32.768 kHz crystal oscillator:

Case 1: The 32.768 kHz crystal oscillator was not previously enabled.

1. Adjust the 32.768 kHz crystal oscillator start-up time in fields FXTALSTUP and SXTALSTUP of SUPC_MR.
2. Start and select the 32.768 kHz crystal oscillator as the source of TD_SLCK by setting TDXTALSEL in SUPC_CR.
3. The switching of TD_SLCK is effective when TDOSCSEL is set in SUPC_SR.
   Note: This flag is set at the end of the 32.768 kHz crystal oscillator start-up period, which can last from a few hundreds of millisecond to a few seconds.

Case 2: The 32.768 kHz crystal oscillator is already enabled and selected as the source of either MD_SLCK or TD_SLCK.

1. Select the 32.768 kHz crystal oscillator as the source of TD_SLCK by setting either TDXTALSEL in SUPC_CR.
2. The switching of TD_SLCK is effective when TDOSCSEL is set in SUPC_SR.

The switching time may vary depending on the slow RC oscillator output frequency. The switch of the slow clock source is glitch-free.

Reverting to the slow RC oscillator as the clock source of TD_SLCK is only possible by resetting the SUPC.

The user can also set the 32.768 kHz crystal oscillator in Bypass mode instead of connecting a crystal. In this case, the user has to provide the external clock signal on XIN32. The input characteristics of the XIN32 pin are given in the section Electrical Characteristics. To enter Bypass mode, SUPC_MR.OSCBYPASS must be set before setting TDXTALSEL.

20.4.3 Power-up, Power-down and SUPC Reset

The SUPC is powered by VDDCORE and is reset by the NRST pin. During reset, the system is in the following state:

- The SUPC is in reset state, and consequently,
- The core reset signal vddcore_nreset is asserted. The I/O pins default to their reset state as described in the section “Package and Pinout”.

The slow RC oscillator is the default oscillator of the SUPC at power-up.

As soon as both voltages are valid, the SUPC exits its reset state and, after five slow RC oscillator cycles, enables the VDDCORE supply monitor. When the output signal of this supply monitor (SMcore_out) indicates a valid VDDCORE voltage for at least one slow RC oscillator cycle, the core logic reset signal vddcore_nreset is de-asserted to the Reset Controller input.

At power-down (i.e., as soon as VDDCORE falls below its SM VTH- threshold), the SUPC is reset and its configuration is immediately lost. Without any sequencing, the core reset signal (vddcore_nreset) is asserted, the I/Os configuration is lost (they default to their reset state), the oscillators and PLLs are switched off. Considering the uncontrolled nature of this power-down, it is strongly recommended to have the device in an idle state before reaching the VTH- threshold of VDDCORE.

20.4.4 Core Reset

The Supply Controller manages the vddcore_nreset signal to the Reset Controller, as described in Power-up, Power-down and SUPC Reset. The vddcore_nreset signal is normally asserted before shutting down the core power supply and released as soon as the core power supply is correctly regulated.

The VDDCORE supply monitor detection source can be programmed to activate vddcore_nreset.

20.4.4.1 VDDCORE Supply Monitor Reset

The VDDCORE supply monitor provides the SMcore_out signal to the SUPC. If this signal is lost for longer than 1 slow clock period, the SUPC asserts vddcore_nreset if SUPC_MR.CORSMRSTEN is written to ‘1’.

If CORSMRSTEN is set, the vddcore_nreset signal is asserted for a minimum of one slow clock cycle and de-asserted as soon as SMcore_out indicates a valid VDDCORE voltage. SUPC_SR.CORSMRSTS indicates the source of the last reset.

20.4.5 Register Write Protection

To prevent any single software error from corrupting SYSC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the System Controller Write Protection Mode Register (SYSC_WPMR).

The following registers can be write-protected:

- RSTC Mode Register
- RTT Mode Register
- RTT Alarm Register
- RTC Control Register
- RTC Mode Register
- RTC Time Alarm Register
- RTC Calendar Alarm Register
- Supply Controller Control Register
- Supply Controller Supply Monitor Mode Register
- Supply Controller Mode Register
- Supply Controller Status Register

20.4.6 System Controller (SYSC) User Interface

Table 20-1. System Controller Registers

Offset	System Controller Peripheral	Name
0x00–0x0C	Reset Controller	RSTC
0x10–0x2C	Supply Controller	SUPC
0x30–0x3C	Real Time Timer	RTT
0x50–0x5C	Watchdog Timer	WDT
0x60–0x8C	Real Time Clock	RTC
0x90–0xE0	Reserved	–
0xE4	Write Protection Mode Register	SYSC_WPMR
0xE8–0xF8	Reserved	–

20.5 Register Summary

Offset	Name	Bit Pos.								
0x00	SUPC_CR	7:0			MDRCSEL		TDXTALSEL			
		15:8								
		23:16								
		31:24	KEY[7:0]							
0x04	SUPC_SMMR	7:0								
		15:8								
		23:16	CORESMUSEL							
		31:24								
0x08	SUPC_MR	7:0								
		15:8			CORSMDIS	CORSMRSTEN				
		23:16		FXTALSTUP		OSCBYPASS				
		31:24	KEY[7:0]							
0x0C ... 0x13	Reserved									
0x14	SUPC_SR	7:0	TDOSCSEL				CORSMRSTS			
		15:8								
		23:16								
		31:24								
0x18 ... 0x1B	Reserved									
0x1C	SUPC_PWR	7:0								
		15:8								
		23:16						MONSELS		
		31:24								
0x20 ... 0xD3	Reserved									
0xD4	SYSC_WPMR	7:0								WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

20.5.1 Supply Controller Control Register

Name: SUPC_CR
Offset: 0x00
Reset: –
Property: Write-only

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	KEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			MDRCSEL		TDXTALSEL			
Access			W		W			
Reset			–		–			

Bits 31:24 – KEY[7:0] Password

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation.

Bit 5 – MDRCSEL Monitoring Domain RC Oscillator Select
0 (NO_EFFECT): No effect.
1 (CRYSTAL_SEL): If KEY is correct, XTALSEL switches the slow clock of the monitoring domain (MD_SLCK) on the slow RC oscillator output.

Bit 3 – TDXTALSEL Timing Domain Crystal Oscillator Select
0 (NO_EFFECT): No effect.
1 (CRYSTAL_SEL): If KEY is correct, XTALSEL switches the slow clock of the timing domain (TD_SLCK) on the 32.768 kHz crystal oscillator output.

20.5.2 Supply Controller Supply Monitor Mode Register

Name: SUPC_SMMR
Offset: 0x04
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CORESMUSEL							
Access	R/W							
Reset	0							

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 23 – CORESMUSEL Core Supply Monitor User Selection

Value	Description
0	The VDDCORE supply monitor threshold is set to its default value. Refer to the section “Electrical Characteristics”.
1	Reserved.

20.5.3 Supply Controller Mode Register

Name: SUPC_MR
Offset: 0x08
Reset: 0x00004A00
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	KEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
		FXTALSTUP		OSCBYPASS				
Access		R/W		R/W				
Reset		0		0				

Bit	15	14	13	12	11	10	9	8
			CORSMDIS	CORSMRSTEN				
Access			R/W	R/W				
Reset			0	0				

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bits 31:24 – KEY[7:0] Password Key

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation.

Bit 22 – FXTALSTUP Fast Start-up 32.768 kHz Crystal Oscillator
FXTALSTUP can be written to ‘1’ only if the bit 17 is low. Once FXTALSTUP is set to ‘1’, bit 17 can be modified freely. This bit is located in the VDDIO domain.

Value	Description
0	Oscillator start-up time is 52800 slow RC periods.
1	Oscillator start-up time is 1 slow RC period.

Bit 20 – OSCBYPASS Oscillator Bypass
0 (NO_EFFECT): No effect. Clock selection depends on the value of XTALSEL (SUPC_CR).
1 (BYPASS): The 32.768 kHz crystal oscillator is bypassed if XTALSEL (SUPC_CR) is set. OSCBYPASS must be set prior to setting XTALSEL.

Bit 13 – CORSMDIS VDDCORE Supply Monitor Disable
0 (ENABLE): The VDDCORE supply monitor is enabled.
1 (DISABLE): The VDDCORE supply monitor is disabled.

Bit 12 – CORSMRSTEN VDDCORE Supply Monitor Reset Enable
0 (NOT_ENABLE): The core reset signal vddcore_nreset is not affected when a VDDCORE supply monitor detection occurs.
1 (ENABLE): The core reset signal, vddcore_nreset is asserted when a VDDCORE supply monitor detection occurs.

20.5.4 Supply Controller Status Register

Name: SUPC_SR
Offset: 0x14
Reset: 0x00000000
Property: Read-only

Because of the asynchronism between the Slow Clock (MD_SLCK) and the System Clock (MCK), the status register flag reset is taken into account only 2 slow clock cycles after the read of the SUPC_SR.

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	TDOSCSEL				CORSMRSTS			
Access	R				R			
Reset	0				0			

Bit 7 – TDOSCSEL 32 kHz Oscillator Selection Status
0 (RC): The timing domain slow clock, TD_SLCK, is generated by the slow RC oscillator.
1 (CRYST): The timing domain slow clock, TD_SLCK, is generated by the 32.768 kHz crystal oscillator.

Bit 3 – CORSMRSTS VDDCORE Supply Monitor Reset Status (cleared on read)
0 (NO): No VDDCORE Supply Monitor reset event has been detected since the last read of the SUPC_SR.
1 (PRESENT): At least one VDDCORE Supply Monitor reset event has been detected since the last read of the SUPC_SR.
When the voltage remains below the defined threshold, there is no VDDCORE Supply Monitor reset event at the output of the VDDCORE Supply Monitor detection cell. The VDDCORE Supply Monitor reset event occurs only when there is a voltage transition below the threshold.

20.5.5 Supply Controller Power Register

Name: SUPC_PWR
Offset: 0x1C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
						MONSELS		
Access						R		
Reset						0		

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 18 – MONSELS Monitoring Oscillator Selection Status
This bit is read-only.
0 (RC): The monitoring domain slow clock, MD_SLCK, is generated by the slow RC oscillator.
1 Reserved.

20.5.6 System Controller Write Protection Mode Register

Name: SYSC_WPMR
Offset: 0xD4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPEN
Access								R/W
Reset								0

Bits 31:8 – WPKEY[23:0] Write Protection Key.

Value	Name	Description
0x525443	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

Bit 0 – WPEN Write Protection Enable
See Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x525443 (“RTC” in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x525443 (“RTC” in ASCII).

21. Watchdog Timer (WDT)

21.1 Description

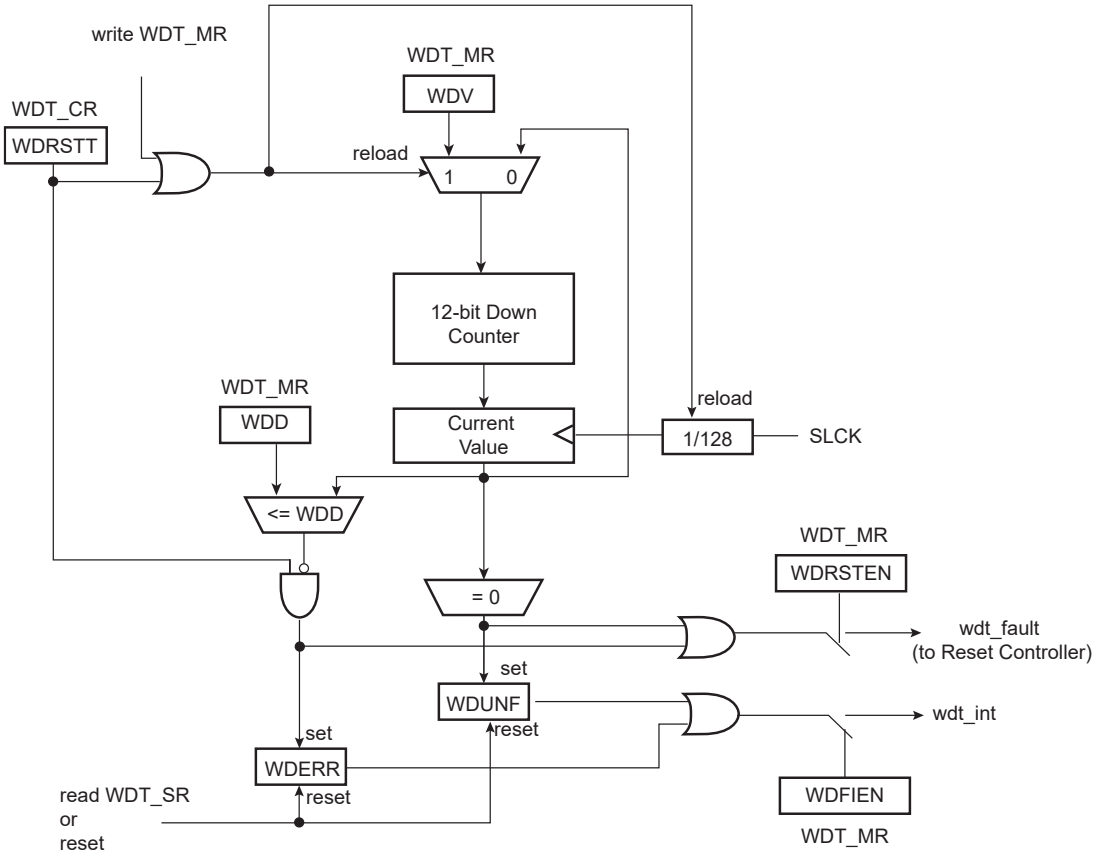
The Watchdog Timer (WDT) is used to prevent system lock-up if the software becomes trapped in a deadlock. It features a 12-bit down counter that allows a watchdog period of up to 16 seconds (slow clock around 32 kHz). It can generate a general reset or a processor reset only. In addition, it can be stopped while the processor is in Debug mode.

21.2 Embedded Characteristics

- 12-bit Key-protected Programmable Counter
- Watchdog Clock is Independent from Processor Clock
- Provides Reset or Interrupt Signals to the System
- Counter May Be Stopped while the Processor is in Debug State or in Idle Mode

21.3 Block Diagram

Figure 21-1. Watchdog Timer Block Diagram



21.4 Functional Description

The Watchdog Timer is used to prevent system lock-up if the software becomes trapped in a deadlock. It is supplied with VDDCORE. It restarts with initial values on processor reset.

The watchdog is built around a 12-bit down counter, which is loaded with the value defined in the field WDV of the Mode Register (WDT_MR). The Watchdog Timer uses the slow clock divided by 128 to establish the maximum watchdog period to be 16 seconds (with a typical slow clock of 32.768 kHz).

After a processor reset, the value of WDV is 0xFFF, corresponding to the maximum value of the counter with the external reset generation enabled (field WDRSTEN at 1 after a backup reset). This means that a default watchdog is running at reset, i.e., at power-up. The user can either disable the WDT by setting bit WDT_MR.WDDIS or reprogram the WDT to meet the maximum watchdog period the application requires.

When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.

If the watchdog is restarted by writing into the Control Register (WDT_CR), WDT_MR must not be programmed during a period of time of three slow clock periods following the WDT_CR write access. In any case, programming a new value in WDT_MR automatically initiates a restart instruction.

WDT_MR can be written only once. Only a processor reset resets it. Writing WDT_MR reloads the timer with the newly programmed mode parameters.

In normal operation, the user reloads the watchdog at regular intervals before the timer underflow occurs, by setting bit WDT_CR.WDRSTT. The watchdog counter is then immediately reloaded from WDT_MR and restarted, and the slow clock 128 divider is reset and restarted. WDT_CR is write-protected. As a result, writing WDT_CR without the correct hard-coded key has no effect. If an underflow does occur, the “wdt_fault” signal to the Reset Controller is asserted if bit WDT_MR.WDRSTEN is set. Moreover, the bit WDUNF is set in the Status Register (WDT_SR).

The reload of the watchdog must occur while the watchdog counter is within a window between 0 and WDD. WDD is defined in WDT_MR.

Any attempt to restart the watchdog while the watchdog counter is between WDV and WDD results in a watchdog error, even if the watchdog is disabled. The bit WDT_SR.WDERR is updated and the “wdt_fault” signal to the Reset Controller is asserted.

Note that this feature can be disabled by programming a WDD value greater than or equal to the WDV value. In such a configuration, restarting the Watchdog Timer is permitted in the whole range [0; WDV] and does not generate an error. This is the default configuration on reset (the WDD and WDV values are equal).

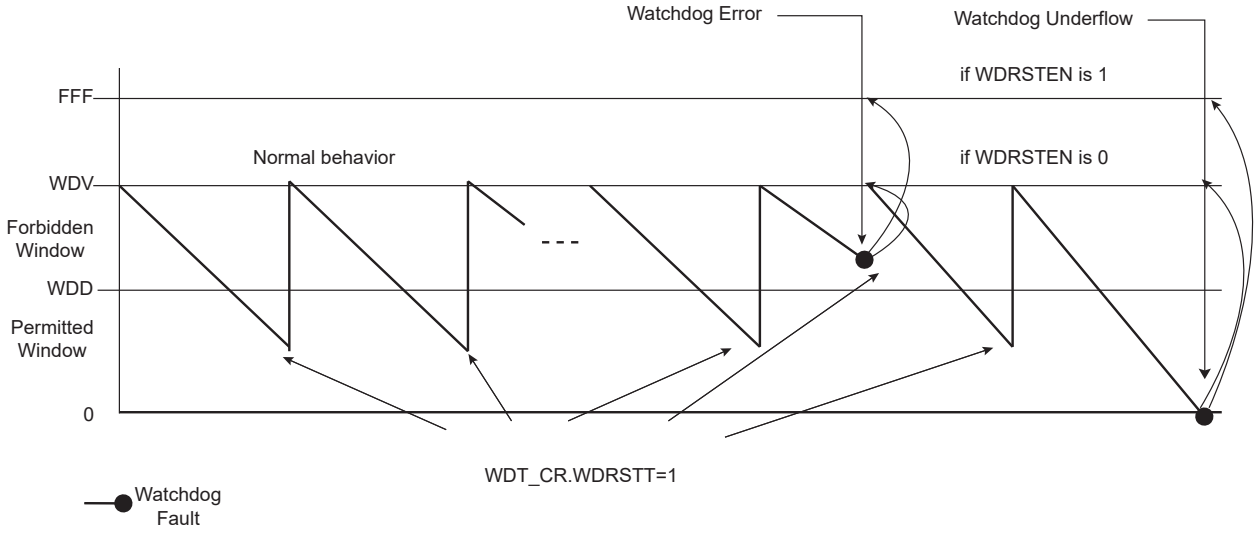
The status bits WDUNF (Watchdog Underflow) and WDERR (Watchdog Error) trigger an interrupt, provided the bit WDT_MR.WDFIEN is set. The signal “wdt_fault” to the Reset Controller causes a watchdog reset if the WDRSTEN bit is set as already explained in the Reset Controller documentation. In this case, the processor and the Watchdog Timer are reset, and the WDERR and WDUNF flags are reset.

If a reset is generated or if WDT_SR is read, the status bits are reset, the interrupt is cleared, and the “wdt_fault” signal to the reset controller is deasserted.

Writing WDT_MR reloads and restarts the down counter.

While the processor is in debug state, the counter may be stopped depending on the value programmed for the bits WDIDLEHLT and WDDBGHLT in WDT_MR.

Figure 21-2. Watchdog Behavior



21.5 Register Summary

Offset	Name	Bit Pos.								
0x00	WDT_CR	7:0								WDRSTT
		15:8								
		23:16								
		31:24	KEY[7:0]							
0x04	WDT_MR	7:0	WDV[7:0]							
		15:8	WDDIS	WDRPROC	WDRSTEN	WDFIEN	WDV[11:8]			
		23:16	WDD[7:0]							
		31:24			WDIDLEHLT	WDDBGHLT	WDD[11:8]			
0x08	WDT_SR	7:0							WDERR	WDUNF
		15:8								
		23:16								
		31:24								

21.5.1 Watchdog Timer Control Register

Name: WDT_CR
Offset: 0x00
Reset: –
Property: Write-only

The WDT_CR register values must not be modified within three slow clock periods following a restart of the watchdog performed by a write access in WDT_CR. Any modification will cause the watchdog to trigger an end of period earlier than expected.

Bit	31	30	29	28	27	26	25	24
	KEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								WDRSTT
Access								W
Reset								–

Bits 31:24 – KEY[7:0] Password

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation.

Bit 0 – WDRSTT Watchdog Restart

Value	Description
0	No effect.
1	Restarts the watchdog if KEY is written to 0xA5.

21.5.2 Watchdog Timer Mode Register

Name: WDT_MR
Offset: 0x04
Reset: 0x3FFF2FFF
Property: Read/Write Once

The first write access prevents any further modification of the value of this register. Read accesses remain possible.

The WDT_MR register values must not be modified within three slow clock periods following a restart of the watchdog performed by a write access in WDT_CR. Any modification will cause the watchdog to trigger an end of period earlier than expected.

Bit	31	30	29	28	27	26	25	24
			WDIDLEHLT	WDDBGHLT	WDD[11:8]			
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	WDD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	WDDIS	WDRPROC	WDRSTEN	WDFIEN	WDV[11:8]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	0	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	WDV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit 29 – WDIDLEHLT Watchdog Idle Halt

Value	Description
0	The watchdog runs when the system is in idle state.
1	The watchdog stops when the system is in idle state.

Bit 28 – WDDBGHLT Watchdog Debug Halt

Value	Description
0	The watchdog runs when the processor is in debug state.
1	The watchdog stops when the processor is in debug state.

Bits 27:16 – WDD[11:0] Watchdog Delta Value
Defines the permitted range for reloading the Watchdog Timer.
If the Watchdog Timer value is less than or equal to WDD, setting bit WDT_CR.WDRSTT restarts the timer.
If the Watchdog Timer value is greater than WDD, setting bit WDT_CR.WDRSTT causes a watchdog error.

Bit 15 – WDDIS Watchdog Disable
When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.

Value	Description
0	Enables the Watchdog Timer.
1	Disables the Watchdog Timer.

Bit 14 – WDRPROC Watchdog Reset Processor

Value	Description
0	If WDRSTEN is 1, a watchdog fault (underflow or error) activates all resets.

Value	Description
1	If WDRSTEN is 1, a watchdog fault (underflow or error) activates the processor reset.

Bit 13 – WDRSTEN Watchdog Reset Enable

Value	Description
0	A watchdog fault (underflow or error) has no effect on the resets.
1	A watchdog fault (underflow or error) triggers a watchdog reset.

Bit 12 – WDFIEN Watchdog Fault Interrupt Enable

Value	Description
0	A watchdog fault (underflow or error) has no effect on interrupt.
1	A watchdog fault (underflow or error) asserts interrupt.

Bits 11:0 – WDV[11:0] Watchdog Counter Value
Defines the value loaded in the 12-bit watchdog counter.

21.5.3 Watchdog Timer Status Register

Name: WDT_SR
Offset: 0x08
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							WDERR	WDUNF
Access							R	R
Reset							0	0

Bit 1 – WDERR Watchdog Error (cleared on read)

Value	Description
0	No watchdog error occurred since the last read of WDT_SR.
1	At least one watchdog error occurred since the last read of WDT_SR.

Bit 0 – WDUNF Watchdog Underflow (cleared on read)

Value	Description
0	No watchdog underflow occurred since the last read of WDT_SR.
1	At least one watchdog underflow occurred since the last read of WDT_SR.

22. Reset Controller (RSTC)

22.1 Description

The Reset Controller (RSTC), driven by software, external reset pin and peripheral events, handles all the resets of the system without any external components. It reports which reset occurred last.

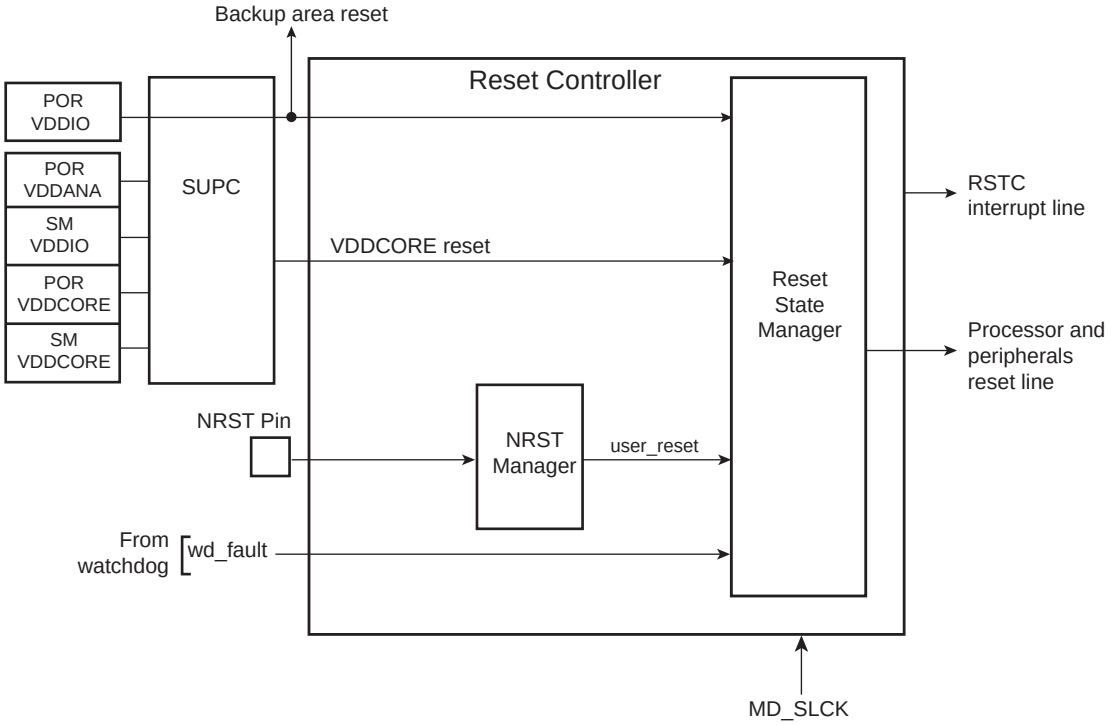
The RSTC also drives simultaneously the peripheral and processor resets. The peripheral resets can be controlled independently.

22.2 Embedded Characteristics

- Driven by Software, External Reset Pin and Peripheral Events
- Management of All System Resets, Including
  - Processor
  - Peripheral Set
- Reset Source Status
  - Status of the last reset
  - Either software reset, user reset, watchdog reset, 32.768 kHz crystal oscillator failure detection reset, CPU clock failure detection
- External Reset Signal Control and Shaping

22.3 Block Diagram

Figure 22-1. Reset Controller Block Diagram



22.4 Functional Description

22.4.1 Overview

The RSTC is made up of an NRST manager and a reset state manager. The RSTC clock is MD_SLCK (monitoring SLCK domain). The RSTC generates the following reset signals:

- proc_nreset: Processor reset line (also resets the Watchdog Timer)
- periph_nreset: Affects the full set of embedded peripherals
- periph_n_separate_reset[n]: Affects the embedded peripheral separately

Note: proc_nreset and periph_nreset are driven in the same way.

These reset signals are asserted by the RSTC, either on events generated by peripherals, events on NRST pin, or on software action. The reset state manager controls the generation of reset signals.

The RSTC Mode register (RSTC_MR), used to configure the RSTC, is powered with VDDCORE, so that its configuration is saved as long as VDDCORE is on.

22.4.2 NRST Manager

NRST behaves as an input and the system is held in reset if NRST is tied to GND by an external signal.

The NRST manager samples the NRST input pin.

22.4.2.1 NRST Signal or Interrupt

The NRST manager samples the NRST pin at MD_SLCK speed. When the NRST line is low for more than three clock cycles, a User Reset is reported to the reset state manager. The NRST pin must be asserted for at least 1 MD_SLCK clock cycle to ensure execution of a user reset.

However, the NRST manager can be programmed to not trigger a reset when an assertion of NRST occurs. Writing a ‘0’ to RSTC_MR.URSTEN disables the User Reset trigger.

The level of the pin NRST can be read at any time in the bit NRSTL in the RSTC Status Register (RSTC_SR). As soon as the NRST pin is asserted, RSTC_SR.URSTS is written to ‘1’. This bit is cleared only when the RSTC_SR is read.

The RSTC can also be programmed to generate an interrupt instead of generating a reset. To do so, RSTC_MR.URSTIEN must be set.

22.4.3 Reset States

The reset state manager handles the different reset sources and generates the internal reset signals. It reports the reset status in RSTTYP of the Status Register (RSTC_SR). The update of RSTC_SR.RSTTYP is performed when the processor reset is released.

22.4.3.1 General Reset

A general reset occurs:

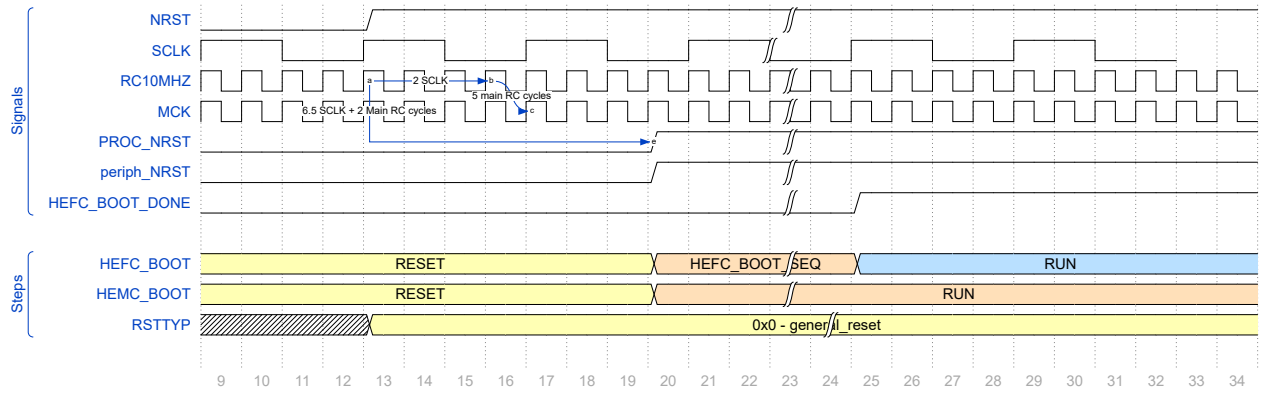
- when a VDDCORE fall is detected by the Supply Monitor (VDDCORE is not correctly powered externally)
- at an NRST pin request

The vddcore_nreset signal is asserted by the Supply Controller when a general reset occurs.

All the reset signals are released and RSTC_SR.RSTTYP reports a general reset. As the RSTC_MR is written to ‘0’, the NRST line rises two cycles after the vddcore_nreset.

The figure below shows how the general reset affects the reset signals.

Figure 22-2. General Reset Timing Diagram



22.4.3.2 32.768 kHz Crystal Oscillator Failure Detection Reset

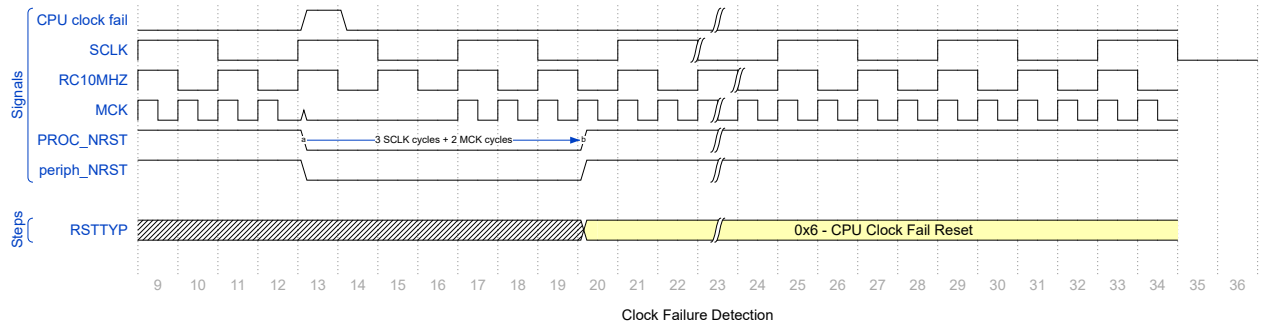
The 32.768 kHz crystal oscillator failure detection reset is done when the 32.768 kHz crystal oscillator frequency monitoring circuitry in the PMC detects a failure and RSTC_MR.SCKSW is written to '1'. This reset lasts three slow clock cycles.

When RSTC_MR.SCKSW is written to ‘0’, the 32.768 kHz crystal oscillator fault has no impact on the RSTC.

During the 32.768 kHz crystal oscillator failure detection reset, the processor reset and the peripheral reset are asserted.

When the 32.768 kHz crystal oscillator failure generates a VDDCORE reset, PMC_SR.XT32KERR is automatically cleared by the peripheral and core reset.

Figure 22-3. 32.768 kHz Crystal Oscillator Failure Detection Reset Timing Diagram



22.4.3.3 CPU Clock Failure Detection Reset

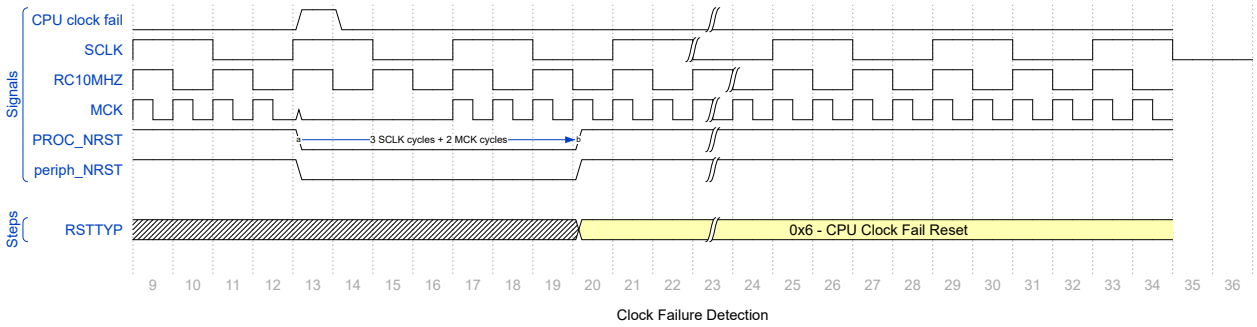
The CPU clock failure detection reset is done when the CPU frequency monitoring circuitry in the PMC detects a failure and RSTC_MR.SCKSW is written to ‘1’. This reset lasts three MD_SLCK cycles.

When RSTC_MR.SCKSW is written to ‘0’, the 32.768 crystal oscillator fault has no impact on the RSTC.

During a CPU clock failure detection reset, the processor reset and the peripheral reset are asserted.

When the CPU clock failure generates a VDDCORE reset, PMC_SR.XT32KERR is automatically cleared by the peripheral and core reset.

Figure 22-4. CPU Clock Failure Detection Reset Timing Diagram



22.4.3.4 Watchdog Reset

The watchdog reset is entered when a watchdog fault occurs. This reset lasts three MD_SLCK cycles.

When in watchdog reset, the processor reset and the peripheral reset are asserted.

The NWDTx pin indicates if the Watchdog x has expired or not:

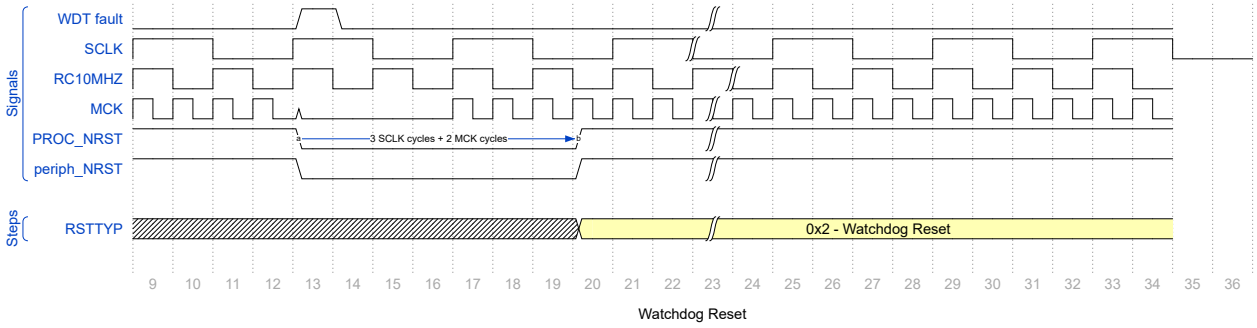
- If NWDTx is '1', the WDT expiration does not occur.
- If NWDTx is '0', the WDT expiration occurs.

The Watchdog Timer is reset by the proc_nreset signal. As the watchdog fault always causes a processor reset if WDT_MR.WDRSTEN is written to ‘1’, the Watchdog Timer is always reset after a watchdog reset, and the Watchdog is enabled by default and with a period set to a maximum.

When WDT_MR.WDRSTEN is written to ‘0’, the watchdog fault has no impact on the RSTC.

After a watchdog overflow occurs, the report on the RSTC_SR.RSTTYP may differ (either WDT_RST or USER_RST) depending on the external components driving the NRST pin. For example, if the NRST line is driven through a resistor and a capacitor (NRST pin debouncer), the reported value is USER_RST if the low to high transition is greater than one MD_SLCK cycle.

Figure 22-5. Watchdog Reset Timing Diagram



22.4.3.5 Software Reset

The RSTC offers commands to assert the different reset signals. These commands are performed by writing the Control register (RSTC_CR) with the following bits at ‘1’:

- RSTC_CR.PROCRST: Writing a ‘1’ to PROCRST resets the processor and all the embedded peripherals, including the memory system and, in particular, the Remap Command.
- RSTC_CR.PERIID and RSTC_CR.PERIIDON: Writing a '1' to PERIIDON and setting the peripheral identifier in PERIID resets the corresponding peripheral. To reset a peripheral, follow the steps below:
  1. Deactivate the different clocks of the peripheral to reset.
  2. Write a '1' to RSTC_CR.PERIIDON and set the peripheral identifier in RSTC_CR.PERIID.

  Then reactivate the clocks of the peripheral that has been reset.

⚠ CAUTION The SpaceWire (SpW) must have the clocks activated when the reset is launched.

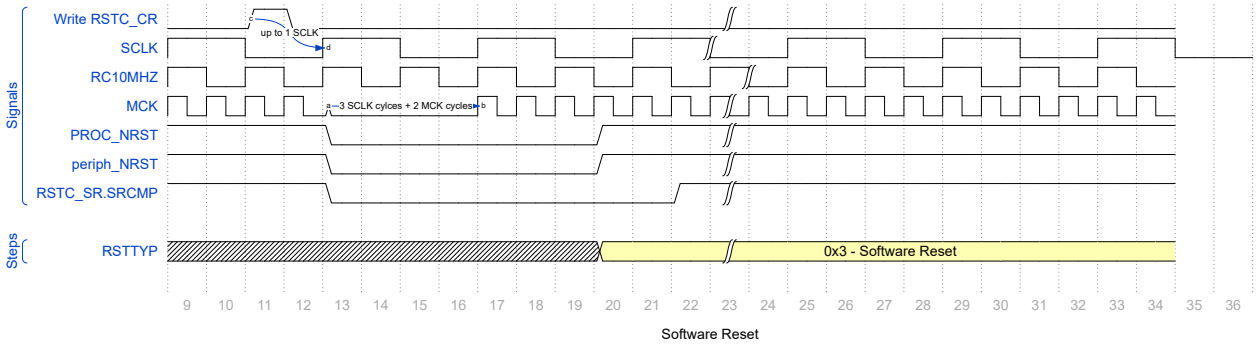
The software reset is entered if at least one of these bits is written to ‘1’ by the software. All these commands can be performed independently or simultaneously. The software reset lasts three MD_SLCK cycles.

The internal reset signals are asserted as soon as the register write is performed. This is detected on the Master Clock (MCK). They are released when the software reset has ended, i.e., synchronously to MD_SLCK.

If and only if the RSTC_CR.PROCRST is written to ‘1’, the RSTC reports the software status in field RSTC_SR.RSTTYP. Other software resets are not reported in RSTTYP.

As soon as a software operation is detected, RSTC_SR.SRCMP is written to ‘1’. SRCMP is cleared at the end of the software reset. No other software reset can be performed while SRCMP is written to ‘1’, and writing any value in the RSTC_CR has no effect.

Figure 22-6. Software Reset Timing Diagram



22.4.3.6 User Reset

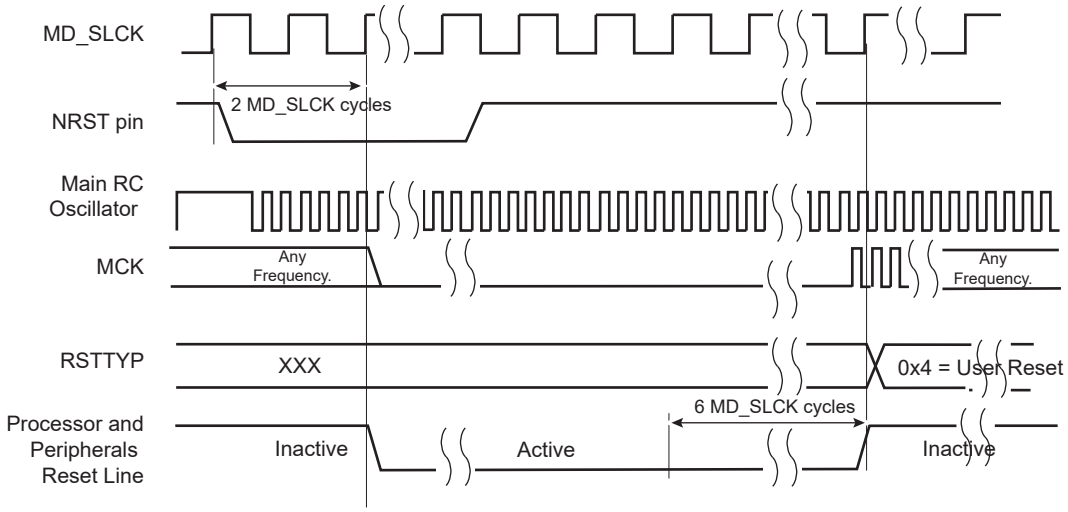
A user reset is generated when a low level is detected on the NRST pin and RSTC_MR.URSTEN is at ‘1’. The NRST input signal is resynchronized with MD_SLCK to ensure proper behavior of the system. Thus, the NRST pin must be asserted for at least 1 MD_SLCK clock cycle to ensure execution of a user reset.

The user reset is triggered 2 MD_SLCK cycles after a low level is detected on NRST. The processor reset and the peripheral reset are asserted.

The user reset ends when NRST rises, after a two-cycle resynchronization time and a three-cycle processor startup. The processor clock is reenabled as soon as NRST is confirmed high.

When the processor reset signal is released, RSTC_SR.RSTTYP is loaded with the value ‘4’, indicating a user reset.

Figure 22-7. User Reset Timing Diagram



22.4.4 Reset State Priorities

The reset state manager manages the priorities among the different reset sources. The resets are listed in order of priority as follows:

1. General reset
2. 32.768 kHz Crystal Failure Detection reset
3. CPU Clock Failure Detection reset
4. Watchdog reset
5. Software reset
6. User reset

Specific cases are listed below:

- When in user reset:
  - A watchdog event is impossible because the Watchdog Timer is being reset by the proc_nreset signal.
  - A software reset is impossible, since the processor reset is being activated.
- When in software reset:
  - A watchdog event has priority over the current state.
  - The NRST has no effect.
- When in watchdog reset:
  - The processor reset is active and so a software reset cannot be programmed.
  - A user reset cannot be entered.

22.5 Register Summary

Offset	Name	Bit Pos.								
0x00	RSTC_CR	7:0				PERIIDON	EXTRST			PROCRST
		15:8	PERIID[7:0]							
		23:16								
		31:24	KEY[7:0]							
0x04	RSTC_SR	7:0								URSTS
		15:8						RSTTYP[2:0]		
		23:16							SRCMP	NRSTL
		31:24								
0x08	RSTC_MR	7:0				URSTIEN	CPUFEN		SCKSW	URSTEN
		15:8								
		23:16								
		31:24	KEY[7:0]							

22.5.1 RSTC Control Register

Name: RSTC_CR
Offset: 0x00
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	KEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	PERIID[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
				PERIIDON	EXTRST			PROCRST
Access				W	W			W
Reset				–	–			–

Bits 31:24 – KEY[7:0] System Reset Key

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation.

Bits 15:8 – PERIID[7:0] Peripheral Identifier
Configures the identifier of peripheral to be reset.

Bit 4 – PERIIDON Peripheral Identifier Validation

Value	Description
0	No effect.
1	Validates the RSTC_CR.PERIID bit field value.

Bit 3 – EXTRST External Reset

Value	Description
0	No effect.
1	If KEY = 0xA5, asserts the NRST pin.

Bit 0 – PROCRST Processor Reset

Value	Description
0	No effect.
1	If KEY = 0xA5, resets the processor and all the embedded peripherals.

22.5.2 RSTC Status Register

Name: RSTC_SR
Offset: 0x04
Reset: 0x00010000
Property: Read-only

The reset value assumes that a general reset has been performed, subject to change if other types of reset are generated.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
							SRCMP	NRSTL
Access							R	R
Reset							0	1

Bit	15	14	13	12	11	10	9	8
						RSTTYP[2:0]		
Access						R	R	R
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
								URSTS
Access								R
Reset								0

Bit 17 – SRCMP Software Reset Command in Progress
When set, this bit indicates that a software reset command is in progress and that no further software reset should be performed until the end of the current one. This bit is automatically cleared at the end of the current software reset.

Value	Description
0	No software command is being performed by the RSTC. The RSTC is ready for a software command.
1	A software reset command is being performed by the RSTC. The RSTC is busy.

Bit 16 – NRSTL NRST Pin Level
Registers the NRST pin level sampled on each MCK rising edge.

Bits 10:8 – RSTTYP[2:0] Reset Type
This field reports the cause of the last processor reset. Reading this RSTC_SR does not reset this field.

Value	Name	Description
0	GENERAL_RST	First power-up reset
1	–	Reserved
2	WDT_RST	Watchdog fault occurred
3	SOFT_RST	Processor reset required by the software
4	USER_RST	NRST pin detected low
5	–	Reserved
6	CPU_FAIL_RST	CPU clock failure detection occurred
7	SLCK_XTAL_RST	32.768 kHz crystal failure detection fault occurred

Bit 0 – URSTS User Reset Status
A high-to-low transition of the NRST pin sets the URSTS. This transition is also detected on the MCK rising edge. If the user reset is disabled (URSTEN = 0 in RSTC_MR) and if the interrupt is enabled by RSTC_MR.URSTIEN, URSTS triggers an interrupt. Reading the RSTC_SR resets URSTS and clears the interrupt.

Value	Description
0	No high-to-low edge on NRST happened since the last read of RSTC_SR.
1	At least one high-to-low transition of NRST has been detected since the last read of RSTC_SR.

22.5.3 RSTC Mode Register

Name: RSTC_MR
Offset: 0x08
Reset: 0x00000001
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

Bit	31	30	29	28	27	26	25	24
	KEY[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				URSTIEN	CPUFEN		SCKSW	URSTEN
Access				R/W	R/W		R/W	R/W
Reset				0	0		0	1

Bits 31:24 – KEY[7:0] Write Access Password

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

Bit 4 – URSTIEN User Reset Interrupt Enable

Value	Description
0	RSTC_SR.USRTS at ‘1’ has no effect on the RSTC interrupt line.
1	RSTC_SR.USRTS at ‘1’ asserts the RSTC interrupt line if URSTEN = 0.

Bit 3 – CPUFEN CPU Fail Enable

Value	Description
0	The detection of a CPU clock failure has no effect.
1	The detection of a CPU clock failure resets the logic supplied by VDDCORE.

Bit 1 – SCKSW Slow Clock Switching

Value	Description
0	The detection of a 32.768 kHz crystal failure has no effect.
1	The detection of a 32.768 kHz crystal failure resets the logic supplied by VDDCORE.

Bit 0 – URSTEN User Reset Enable

Value	Description
0	The detection of a low level on the NRST pin does not generate a user reset.
1	The detection of a low level on the NRST pin triggers a user reset.

23. Real-time Clock (RTC)

23.1 Description

The Real-time Clock (RTC) peripheral is designed for very low power consumption. For optimal functionality, the RTC requires an accurate external 32.768 kHz clock, which can be provided by a crystal oscillator.

It combines a complete time-of-day clock with alarm and a Gregorian or Persian calendar, complemented by a programmable periodic interrupt. The alarm and calendar registers are accessed by a 32-bit data bus.

The time and calendar values are coded in binary-coded decimal (BCD) format. The time format can be 24-hour mode or 12-hour mode with an AM/PM indicator. The number of periods of 1/1024 seconds within 1 second is reported. This information allows a timestamping with an accuracy higher than 1 millisecond.

Updating time and calendar fields and configuring the alarm fields are performed by a parallel capture on the 32-bit data bus. An entry control is performed to avoid loading registers with incompatible BCD format data or with an incompatible date according to the current month/year/century.

A clock divider calibration circuitry can be used to compensate for crystal oscillator frequency variations.

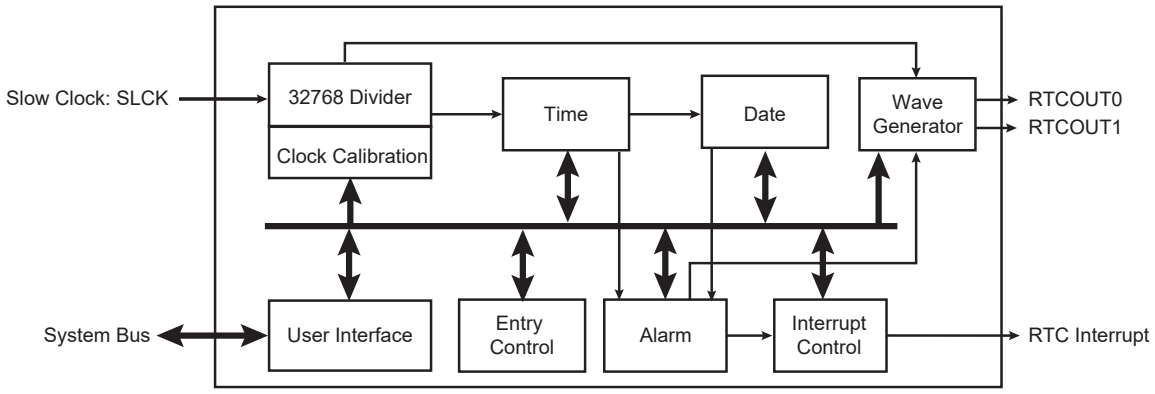
An RTC output can be programmed to generate several waveforms, including a prescaled clock derived from 32.768 kHz.

23.2 Embedded Characteristics

- Full Asynchronous Design for Ultra Low Power Consumption
- Gregorian and Persian Modes Supported
- Milliseconds Image Report
- Programmable Periodic Interrupt
- Safety/security Features:
  - Valid Time and Date Programming Check
  - On-The-Fly Time and Date Validity Check
- Counters Calibration Circuitry to Compensate for Crystal Oscillator Variations
- Waveform Generation
- Register Write Protection

23.3 Block Diagram

Figure 23-1. Real-time Clock Block Diagram



23.4 Product Dependencies

23.4.1 Power Management

The Real-time Clock is continuously clocked at 32.768 kHz. The Power Management Controller has no effect on RTC behavior.

23.4.2 Interrupt

Within the System Controller, the RTC interrupt is OR-wired with all the other module interrupts.

Only one System Controller interrupt line is connected on one of the internal sources of the interrupt controller.

RTC interrupt requires the interrupt controller to be programmed first.

When a System Controller interrupt occurs, the service routine must first determine the cause of the interrupt. This is done by reading each status register of the System Controller peripherals successively.

23.5 Functional Description

The RTC provides a full binary-coded decimal (BCD) clock that includes century (19/20), year (with leap years), month, date, day, hours, minutes and seconds reported in RTC Time Register (RTC_TIMR).

The MS field in the RTC Milliseconds Register (RTC_MSR) reports the number of periods of 1/1024 seconds elapsed within 1 second. The MS field is cleared at the beginning of each 1-second period.

The MS field can be used for timestamping with an accuracy higher than 1 ms. It can also be post-processed by software to provide the millisecond value. The data in milliseconds can be obtained by multiplying the value read in the MS field by 1000/1024 (for an optimal accuracy, the result must be rounded and not truncated).

The valid year range is up to 2099 in Gregorian mode (or 1300 to 1499 in Persian mode).

The RTC can operate in 24-hour mode or in 12-hour mode with an AM/PM indicator.

Corrections for leap years are included (all years divisible by 4 being leap years except 1900). This is correct up to the year 2099.

The RTC can generate configurable waveforms on RTCOUT0/1 outputs.

23.5.1 Reference Clock

The reference clock is the Slow Clock (SLCK) which can be driven internally or by an external 32.768 kHz crystal.

During low-power modes of the processor, the oscillator runs and power consumption is critical. The crystal selection must consider the current consumption for power saving and the frequency drift due to temperature effect on the circuit for time accuracy.

23.5.2 Timing

The RTC is updated in real time at one-second intervals in Normal mode for the counters of seconds, at one-minute intervals for the counter of minutes and so on.

Due to the asynchronous operation of the RTC with respect to the rest of the chip, to be certain that the value read in the RTC registers (century, year, month, date, day, hours, minutes, seconds) are valid and stable, it is necessary to read these registers twice. If the data is the same both times, then it is valid. Therefore, a minimum of two and a maximum of three accesses are required.

23.5.3 Alarm

The RTC has five programmable fields: month, date, hours, minutes and seconds.

Each of these fields can be enabled or disabled to match the alarm condition:

- If all the fields are enabled, an alarm flag is generated (the corresponding flag is asserted and an interrupt generated if enabled) at a given month, date, hour/minute/second.
- If only the “seconds” field is enabled, then an alarm is generated every minute.

Depending on the combination of fields enabled, a large number of possibilities are available to the user ranging from minutes to 365/366 days.

Hour, minute and second matching alarms (SECEN, MINEN, HOUREN) can be enabled independently of SEC, MIN, HOUR fields.

Note: To change one of the SEC, MIN, HOUR, DATE, MONTH fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_TIMALR or RTC_CALALR. The first access clears the enable corresponding to the field to change (SECEN, MINEN, HOUREN, DATEEN, MTHEN). If the field is already cleared, this access is not required. The second access performs the change of the value (SEC, MIN, HOUR, DATE, MONTH). The third access is required to re-enable the field by writing 1 in SECEN, MINEN, HOUREn, DATEEN, MTHEN fields.

23.5.4 Error Checking when Programming

Verification on user interface data is performed when accessing the century, year, month, date, day, hours, minutes, seconds and alarms. A check is performed on illegal BCD entries such as illegal date of the month with regard to the year and century configured.

If one of the time fields is not correct, the data is not loaded into the register/counter and a flag is set in the validity register. The user can not reset this flag. It is reset as soon as an acceptable value is programmed. This avoids any further side effects in the hardware. The same procedure is followed for the alarm.

The following checks are performed:

1. Century (check if it is in range 19–20 or 13–14 in Persian mode)
2. Year (BCD entry check)
3. Date (check range 01–31)
4. Month (check if it is in BCD range 01–12, check validity regarding “date”)
5. Day (check range 1–7)
6. Hour (BCD checks: in 24-hour mode, check range 00–23 and check that AM/PM flag is not set if RTC is set in 24-hour mode; in 12-hour mode check range 01–12)
7. Minute (check BCD and range 00–59)
8. Second (check BCD and range 00–59)

Note: If the 12-hour mode is selected by means of the RTC Mode Register (RTC_MR), a 12-hour value can be programmed and the returned value on RTC_TIMR will be the corresponding 24-hour value. The entry control checks the value of the AM/PM indicator (bit 22 of RTC_TIMR) to determine the range to be checked.

23.5.5 RTC Internal Free Running Counter Error Checking

To improve the reliability and security of the RTC, a permanent check is performed on the internal free running counters to report non-BCD or invalid date/time values.

An error is reported by TDERR bit in the status register (RTC_SR) if an incorrect value has been detected. The flag can be cleared by setting the TDERRCLR bit in the Status Clear Command Register (RTC_SCCR).

The TDERR error flag will be set again if the source of the error has not been cleared before clearing the TDERR flag. The clearing of the source of such error can be done by reprogramming a correct value on RTC_CALR and/or RTC_TIMR.

The RTC internal free running counters may automatically clear the source of TDERR due to their roll-over (i.e., every 10 seconds for SECONDS[3:0] field in RTC_TIMR). In this case the TDERR is held high until a clear command is asserted by TDERRCLR bit in RTC_SCCR.

23.5.6 Updating Time/Calendar

23.5.6.1 Description

The update of the time/calendar must be synchronized on a second periodic event by either polling the RTC_SR.SEC status bit or by enabling the SECEN interrupt in the RTC_IER register.

Once the second event occurs, the user must stop the RTC by setting the corresponding field in the Control Register (RTC_CR). Bit UPDTIM must be set to update time fields (hour, minute, second) and bit UPDCAL must be set to update calendar fields (century, year, month, date, day).

The ACKUPD bit must then be read to 1 by either polling the RTC_SR or by enabling the ACKUPD interrupt in the RTC_IER. Once ACKUPD is read to 1, it is mandatory to clear this flag by writing the corresponding bit in the RTC_SCCR, after which the user can write to the Time Register, the Calendar Register, or both.

Once the update is finished, the user must write UPDTIM and/or UPDCAL to 0 in the RTC_CR.

The timing sequence of the time/calendar update is described in the figure below.

When entering the programming mode of the calendar fields, the time fields remain enabled and both the time and the calendar fields are stopped. This is due to the location of the calendar logical circuity (downstream for low-power considerations). It is highly recommended to prepare all the fields to be updated before entering programming mode. In successive update operations, the user must wait for at least one second after resetting the UPDTIM/UPDCAL bit in the RTC_CR before setting these bits again. This is done by waiting for the SEC flag in the RTC_SR before setting the UPDTIM/UPDCAL bit. After resetting UPDTIM/UPDCAL, the SEC flag must also be cleared.

Figure 23-2. Time/Calendar Update Timing Diagram

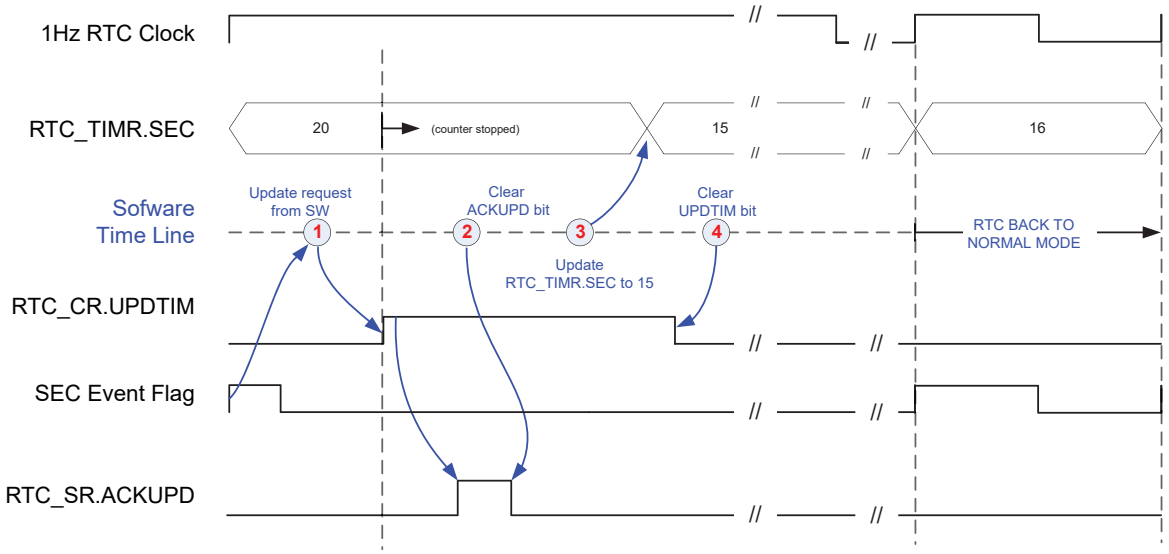
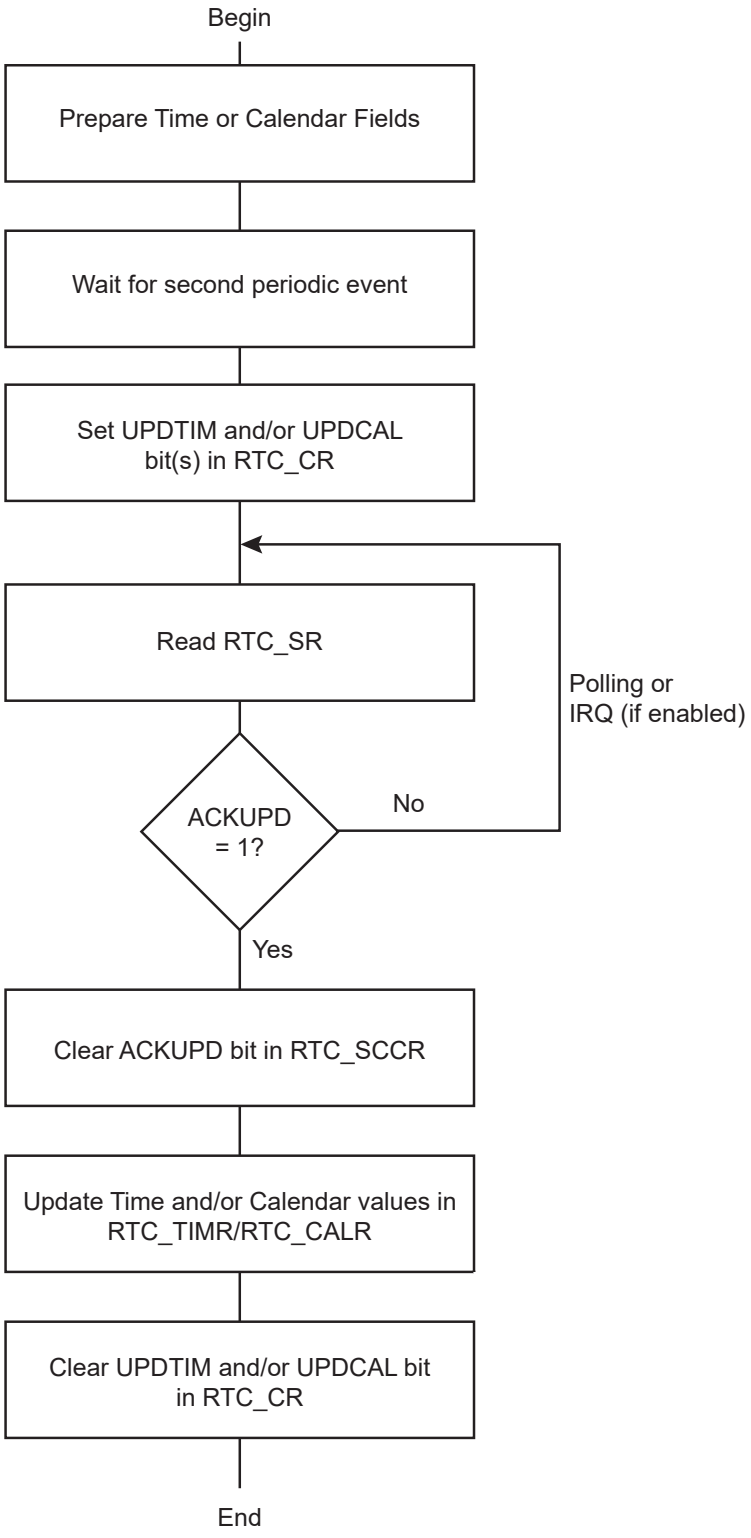


Figure 23-3. Gregorian and Persian Modes Update Sequence



23.5.7 RTC Accurate Clock Calibration

The crystal oscillator that drives the RTC may not be as accurate as expected mainly due to temperature variation. The RTC is equipped with circuitry able to correct slow clock crystal drift.

To compensate for possible temperature variations over time, this accurate clock calibration circuitry can be programmed on-the-fly and also programmed during application manufacturing, in order to correct the crystal frequency accuracy at room temperature (20–25°C). The typical clock drift range at room temperature is ±20 ppm.

In the device operating temperature range, the 32.768 kHz crystal oscillator clock inaccuracy can be up to -200 ppm.

The RTC clock calibration circuitry allows positive or negative correction in a range of 1.5 ppm to 1950 ppm.

The calibration circuitry is fully digital. Thus, the configured correction is independent of temperature, voltage, process, etc., and no additional measurement is required to check that the correction is effective.

If the correction value configured in the calibration circuitry results from an accurate crystal frequency measure, the remaining accuracy is bounded by the values listed below:

- Below 1 ppm, for an initial crystal drift between 1.5 ppm up to 20 ppm, and from 30 ppm to 90 ppm
- Below 2 ppm, for an initial crystal drift between 20 ppm up to 30 ppm, and from 90 ppm to 130 ppm
- Below 5 ppm, for an initial crystal drift between 130 ppm up to 200 ppm

The calibration circuitry does not modify the 32.768 kHz crystal oscillator clock frequency but it acts by slightly modifying the 1 Hz clock period from time to time. The correction event occurs every 1 + [(20 - (19 x HIGHPPM)) x CORRECTION] seconds. When the period is modified, depending on the sign of the correction, the 1 Hz clock period increases or reduces by around 4 ms. Depending on the CORRECTION, NEGPPM and HIGHPPM values configured in RTC_MR, the period interval between two correction events differs.

Figure 23-4. Calibration Circuitry

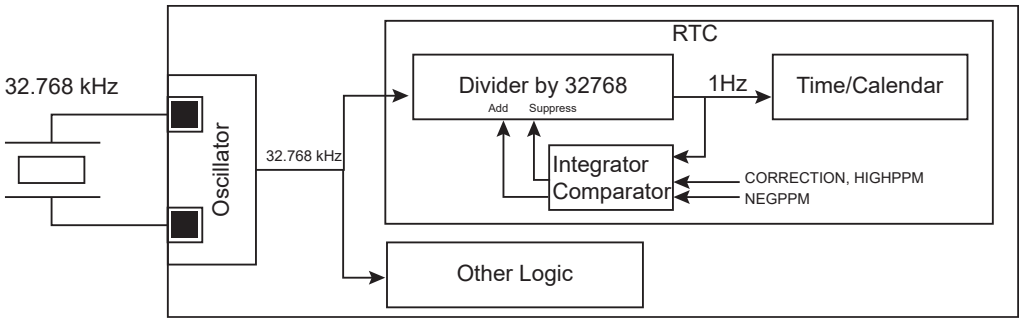
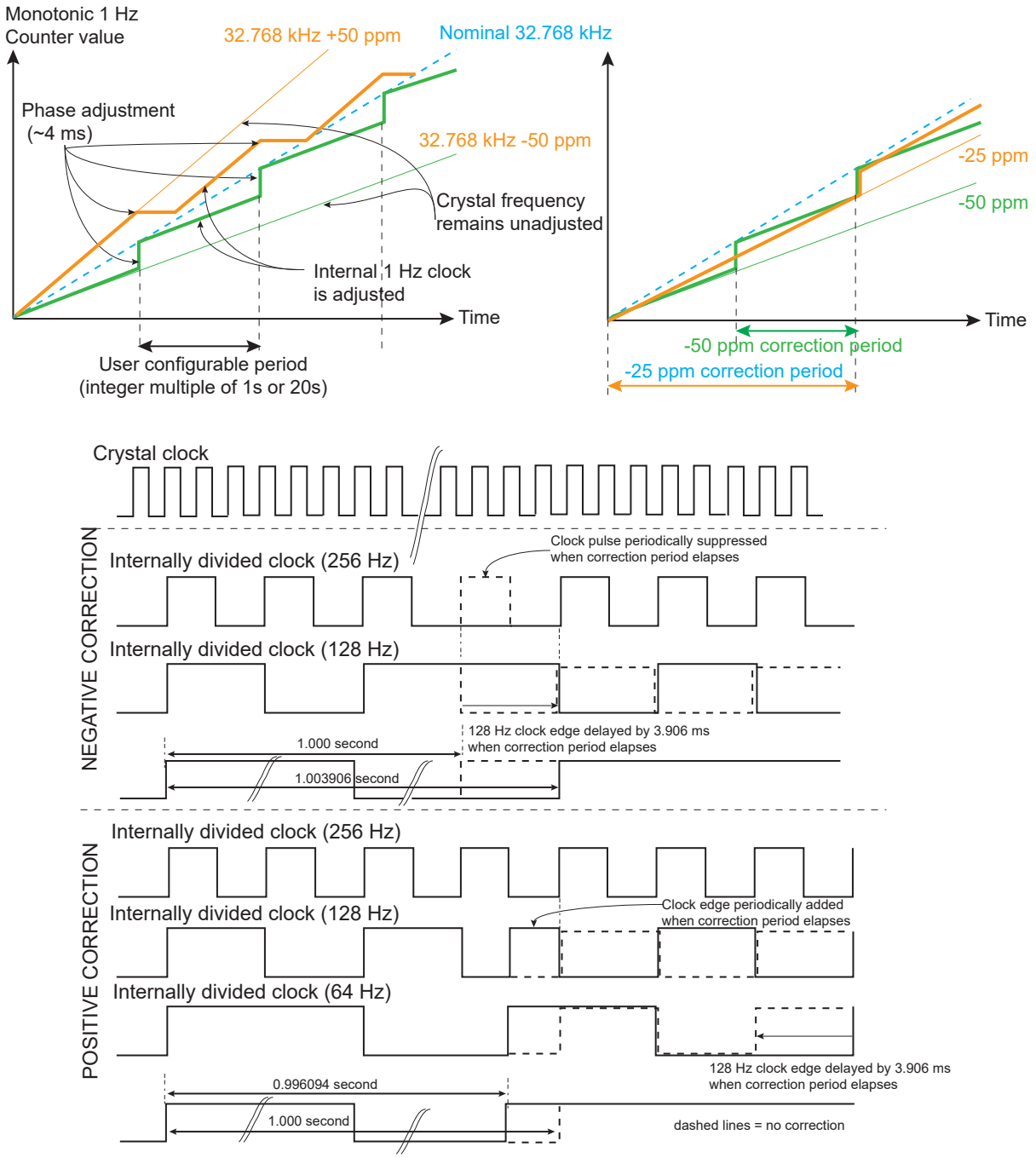


Figure 23-5. Calibration Circuitry Waveforms



The inaccuracy of a crystal oscillator at typical room temperature (±20 ppm at 20–25 °C) can be compensated if a reference clock/signal is used to measure such inaccuracy. This kind of calibration operation can be set up during the final product manufacturing by means of measurement equipment embedding such a reference clock. The correction of value must be programmed into the (RTC_MR), and this value is kept as long as the circuitry is powered (backup area). Removing the backup power supply cancels this calibration. This room temperature calibration can be further processed by means of the networking capability of the target application.

To ease the comparison of the inherent crystal accuracy with the reference clock/signal during manufacturing, an internal prescaled 32.768 kHz clock derivative signal can be assigned to drive RTC output. To accommodate the measure, several clock frequencies can be selected among 1 Hz, 32 Hz, 64 Hz, 512 Hz.

The clock calibration correction drives the internal RTC counters but can also be observed in the RTC output when one of the following three frequencies 1 Hz, 32 Hz or 64 Hz is configured. The correction is not visible in the RTC output if 512 Hz frequency is configured.

Note that this adjustment does not take into account the temperature variation.

The frequency drift (up to -200 ppm) due to temperature variation can be compensated using a reference time if the application can access such a reference. If a reference time cannot be used, a temperature sensor can be placed close to the crystal oscillator in order to get the operating temperature of the crystal oscillator. Once obtained, the temperature may be converted using a lookup table (describing the accuracy/temperature curve of the crystal oscillator used) and RTC_MR configured accordingly. The calibration can be performed on-the-fly. This adjustment method is not based on a measurement of the crystal frequency/drift and therefore can be improved by means of the networking capability of the target application.

If no crystal frequency adjustment has been done during manufacturing, it is still possible to do it. In the case where a reference time of the day can be obtained through LAN/WAN network, it is possible to calculate the drift of the application crystal oscillator by comparing the values read on RTC Time Register (RTC_TIMR) and programming the HIGHPPM and CORRECTION fields on RTC_MR according to the difference measured between the reference time and those of RTC_TIMR.

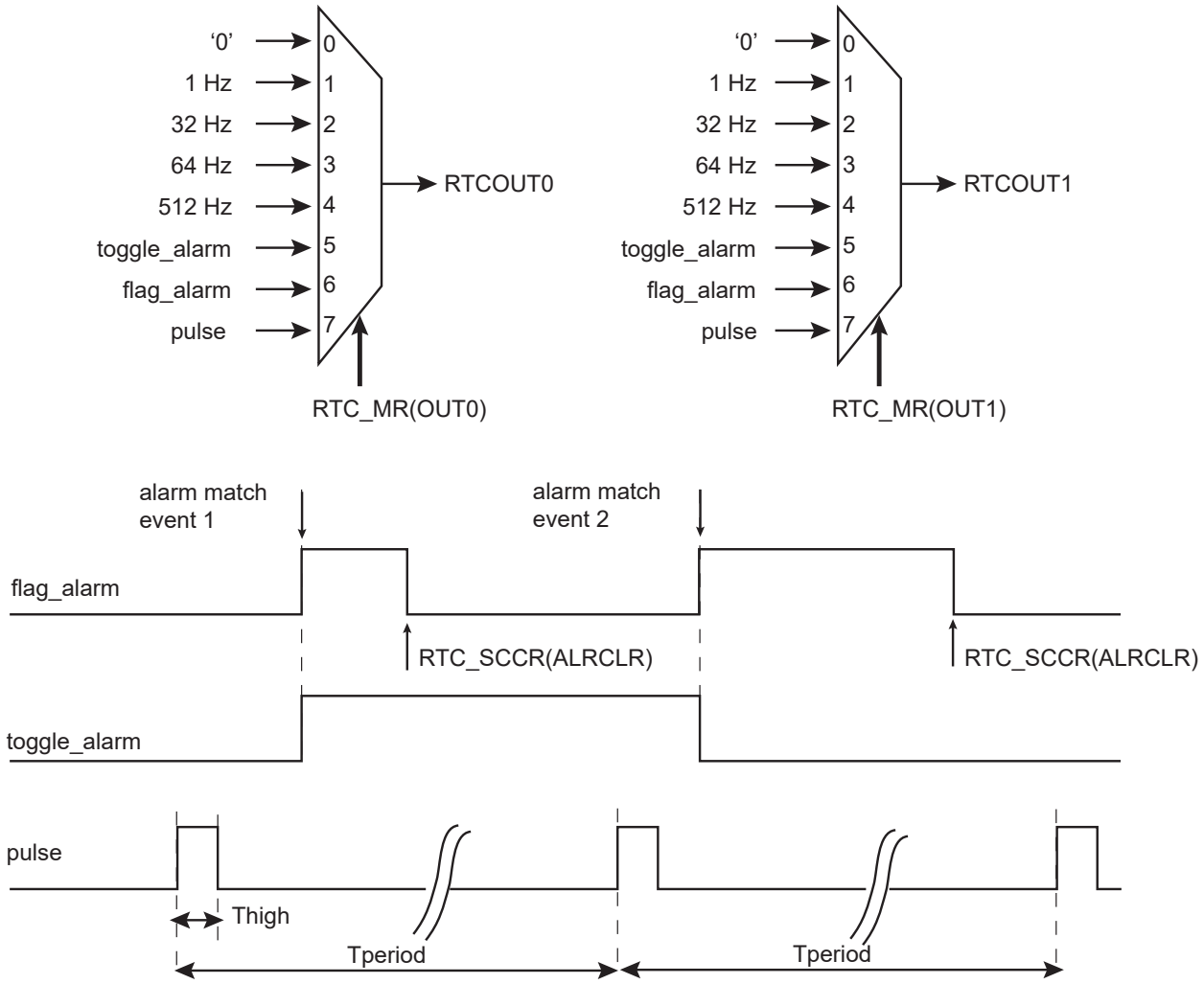
23.5.8 Waveform Generation

Waveforms can be generated in order to take advantage of the RTC inherent prescalers while the RTC is the only powered circuitry (Low-power mode of operation, Backup mode) or in any active mode. Entering Backup or Low-power operating modes does not affect the waveform generation outputs.

The outputs RTCOUT0 and RTCOUT1 can be configured to provide several types of waveforms. The figure below illustrates the different signals available to generate RTCOUT0 and RTCOUT1.

PIO lines associated to the RTC outputs automatically select these waveforms as soon as RTC_MR.OUT0/OUT1 differ from 0 and if RTC_MR.ENx has been written to 1.

Figure 23-6. Waveform Generation



23.6 Register Summary

Offset	Name	Bit Pos.								
0x00	RTC_CR	7:0							UPDCAL	UPDTIM
		15:8							TIMEVSEL[1:0]	
		23:16							CALEVSEL[1:0]	
		31:24								
0x04	RTC_MR	7:0				NEGPPM			PERSIAN	HRMOD
		15:8	HIGHPPM	CORRECTION[6:0]						
		23:16	EN1	OUT1[2:0]			EN0	OUT0[2:0]		
		31:24			TPERIOD[1:0]			THIGH[2:0]		
0x08	RTC_TIMR	7:0		SEC[6:0]						
		15:8		MIN[6:0]						
		23:16		AMPM	HOUR[5:0]					
		31:24								
0x0C	RTC_CALR	7:0		CENT[6:0]						
		15:8	YEAR[7:0]							
		23:16	DAY[2:0]			MONTH[4:0]				
		31:24			DATE[5:0]					
0x10	RTC_TIMALR	7:0	SECEN	SEC[6:0]						
		15:8	MINEN	MIN[6:0]						
		23:16	HOUREN	AMPM	HOUR[5:0]					
		31:24								
0x14	RTC_CALALR	7:0								
		15:8								
		23:16	MTHEN			MONTH[4:0]				
		31:24	DATEEN		DATE[5:0]					
0x18	RTC_SR	7:0			TDERR	CALEV	TIMEV	SEC	ALARM	ACKUPD
		15:8								
		23:16								
		31:24								
0x1C	RTC_SCCR	7:0			TDERRCLR	CALCLR	TIMCLR	SECCLR	ALRCLR	ACKCLR
		15:8								
		23:16								
		31:24								
0x20	RTC_IER	7:0			TDERREN	CALEN	TIMEN	SECEN	ALREN	ACKEN
		15:8								
		23:16								
		31:24								
0x24	RTC_IDR	7:0			TDERRDIS	CALDIS	TIMDIS	SECDIS	ALRDIS	ACKDIS
		15:8								
		23:16								
		31:24								
0x28	RTC_IMR	7:0			TDERR	CAL	TIM	SEC	ALR	ACK
		15:8								
		23:16								
		31:24								
0x2C	RTC_VER	7:0					NVCALALR	NVTIMALR	NVCAL	NVTIM
		15:8								
		23:16								
		31:24								
0x30 ... 0xCF	Reserved									
0xD0	RTC_MSR	7:0	MS[7:0]							
		15:8							MS[9:8]	
		23:16								
		31:24								

23.6.1 RTC Control Register

Name: RTC_CR
Offset: 0x00
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
							CALEVSEL[1:0]	
Access							R/W	R/W
Reset							0	0

Bit	15	14	13	12	11	10	9	8
							TIMEVSEL[1:0]	
Access							R/W	R/W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
							UPDCAL	UPDTIM
Access							R/W	R/W
Reset							0	0

Bits 17:16 – CALEVSEL[1:0] Calendar Event Selection
The event that generates the flag CALEV in RTC_SR depends on the value of CALEVSEL

Value	Name	Description
0	WEEK	Week change (every Monday at time 00:00:00)
1	MONTH	Month change (every 01 of each month at time 00:00:00)
2	YEAR	Year change (every January 1 at time 00:00:00)
3	YEAR	Reserved

Bits 9:8 – TIMEVSEL[1:0] Time Event Selection
The event that generates the flag TIMEV in RTC_SR depends on the value of TIMEVSEL.

Value	Name	Description
0	MINUTE	Minute change
1	HOUR	Hour change
2	MIDNIGHT	Every day at midnight
3	NOON	Every day at noon

Bit 1 – UPDCAL Update Request Calendar Register
Calendar counting consists of day, date, month, year and century counters. Calendar counters can be programmed once this bit is set and acknowledged by the bit ACKUPD of the RTC_SR.

Value	Description
0	No effect or, if UPDCAL has been previously written to 1, stops the update procedure.
1	Stops the RTC calendar counting.

Bit 0 – UPDTIM Update Request Time Register
Time counting consists of second, minute and hour counters. Time counters can be programmed once this bit is set and acknowledged by the bit ACKUPD of the RTC_SR.

Value	Description
0	No effect or, if UPDTIM has been previously written to 1, stops the update procedure.
1	Stops the RTC time counting.

23.6.2 RTC Mode Register

Name: RTC_MR
Offset: 0x04
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
			TPERIOD[1:0]			THIGH[2:0]		
Access			R/W	R/W		R/W	R/W	R/W
Reset			0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
	EN1	OUT1[2:0]			EN0	OUT0[2:0]		
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	HIGHPPM	CORRECTION[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				NEGPPM			PERSIAN	HRMOD
Access				R/W			R/W	R/W
Reset				0			0	0

Bits 29:28 – TPERIOD[1:0] Period of the Output Pulse

Value	Name	Description
0	P_1S	1 second
1	P_500MS	500 ms
2	P_250MS	250 ms
3	P_125MS	125 ms

Bits 26:24 – THIGH[2:0] High Duration of the Output Pulse

Value	Name	Description
0	H_31MS	31.2 ms
1	H_16MS	15.6 ms
2	H_4MS	3.91 ms
3	H_976US	976 µs
4	H_488US	488 µs
5	H_122US	122 µs
6	H_30US	30.5 µs
7	H_15US	15.2 µs

Bit 23 – EN1 PIO Line Enable
This bit is write-only.

Value	Description
0	The PIO line associated to RTCOUT1 is disabled.
1	The PIO line associated to RTCOUT1 is enabled.

Bits 22:20 – OUT1[2:0] RTCOUT1 Output Source Selection

Value	Name	Description
0	NO_WAVE	No waveform, stuck at ‘0’
1	FREQ1HZ	1 Hz square wave

Value	Name	Description
2	FREQ32HZ	32 Hz square wave
3	FREQ64HZ	64 Hz square wave
4	FREQ512HZ	512 Hz square wave
5	ALARM_TOGGLE	Output toggles when alarm flag rises
6	ALARM_FLAG	Output is a copy of the alarm flag
7	PROG_PULSE	Duty cycle programmable pulse

Bit 19 – EN0 PIO Line Enable
This bit is write-only.

Value	Description
0	The PIO line associated to RTCOUT0 is disabled.
1	The PIO line associated to RTCOUT0 is enabled.

Bits 18:16 – OUT0[2:0] RTCOUT0 OutputSource Selection

Value	Name	Description
0	NO_WAVE	No waveform, stuck at ‘0’
1	FREQ1HZ	1 Hz square wave
2	FREQ32HZ	32 Hz square wave
3	FREQ64HZ	64 Hz square wave
4	FREQ512HZ	512 Hz square wave
5	ALARM_TOGGLE	Output toggles when alarm flag rises
6	ALARM_FLAG	Output is a copy of the alarm flag
7	PROG_PULSE	Duty cycle programmable pulse

Bit 15 – HIGHPPM HIGH PPM Correction
If the absolute value of the correction to be applied is lower than 30 ppm, it is recommended to clear HIGHPPM.
HIGHPPM set to 1 is recommended for 30 ppm correction and above.
Formula:
If HIGHPPM = 0, then the clock frequency correction range is from 1.5 ppm up to 98 ppm. The RTC accuracy is less than 1 ppm for a range correction from 1.5 ppm up to 30 ppm.
The correction field must be programmed according to the required correction in ppm; the formula is as follows:

$$\mathrm{CORRECTION} = \frac{3906}{20 \times \mathrm{ppm}} - 1$$

The value obtained must be rounded to the nearest integer prior to being programmed into CORRECTION field.
If HIGHPPM = 1, then the clock frequency correction range is from 30.5 ppm up to 1950 ppm. The RTC accuracy is less than 1 ppm for a range correction from 30.5 ppm up to 90 ppm.
The correction field must be programmed according to the required correction in ppm; the formula is as follows:

$$\mathrm{CORRECTION} = \frac{3906}{\mathrm{ppm}} - 1$$

The value obtained must be rounded to the nearest integer prior to be programmed into CORRECTION field.
If NEGPPM is set to 1, the ppm correction is negative (used to correct crystals that are faster than the nominal 32.768 kHz).

Value	Description
0	Lower range ppm correction with accurate correction.
1	Higher range ppm correction with accurate correction.

Bits 14:8 – CORRECTION[6:0] Slow Clock Correction

Value	Description
0	No correction
1-127	The slow clock will be corrected according to the formula given in HIGHPPM description.

Bit 4 – NEGPPM Negative PPM Correction
See CORRECTION and HIGHPPM field descriptions.
NEGPPM must be cleared to correct a crystal slower than 32.768 kHz.

Value	Description
0	Positive correction (the divider will be slightly higher than 32768).
1	Negative correction (the divider will be slightly lower than 32768).

Bit 1 – PERSIAN PERSIAN Calendar

Value	Description
0	Gregorian calendar.
1	Persian calendar.

Bit 0 – HRMOD 12-/24-hour Mode

Value	Description
0	24-hour mode is selected.
1	12-hour mode is selected.

23.6.3 RTC Time Register

Name: RTC_TIMR
Offset: 0x08
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
		AMPM	HOUR[5:0]					
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
		MIN[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		SEC[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit 22 – AMPM Ante Meridiem Post Meridiem Indicator
This bit is the AM/PM indicator in 12-hour mode.

Value	Description
0	AM.
1	PM.

Bits 21:16 – HOUR[5:0] Current Hour
The range that can be set is 1–12 (BCD) in 12-hour mode or 0–23 (BCD) in 24-hour mode.

Bits 14:8 – MIN[6:0] Current Minute
The range that can be set is 0–59 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

Bits 6:0 – SEC[6:0] Current Second
The range that can be set is 0–59 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

23.6.4 RTC Calendar Register

Name: RTC_CALR
Offset: 0x0C
Reset: 0x01E11320
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
			DATE[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	23	22	21	20	19	18	17	16
	DAY[2:0]			MONTH[4:0]				
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
	YEAR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	1	1

Bit	7	6	5	4	3	2	1	0
		CENT[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	1	0	0	0	0	0

Bits 29:24 – DATE[5:0] Current Day in Current Month
The range that can be set is 01–31 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

Bits 23:21 – DAY[2:0] Current Day in Current Week
The range that can be set is 1–7 (BCD).
The coding of the number (which number represents which day) is user-defined as it has no effect on the date counter.

Bits 20:16 – MONTH[4:0] Current Month
The range that can be set is 01–12 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

Bits 15:8 – YEAR[7:0] Current Year
The range that can be set is 00–99 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

Bits 6:0 – CENT[6:0] Current Century
The range that can be set is 19–20 (Gregorian) or 13–14 (Persian) (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

23.6.5 RTC Time Alarm Register

Name: RTC_TIMALR
Offset: 0x10
Reset: 0x00000000
Property: Read/Write

To change one of the SEC, MIN, HOUR fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_TIMALR. The first access clears the enable corresponding to the field to change (SECEN, MINEN, HOUREN). If the field is already cleared, this access is not required. The second access performs the change of the value (SEC, MIN, HOUR). The third access is required to re-enable the field by writing 1 in SECEN, MINEN, HOUREN fields.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	HOUREN	AMPM	HOUR[5:0]					
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MINEN	MIN[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SECEN	SEC[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 23 – HOUREN Hour Alarm Enable

Value	Description
0	The hour-matching alarm is disabled.
1	The hour-matching alarm is enabled.

Bit 22 – AMPM AM/PM Indicator
This field is the alarm field corresponding to the BCD-coded hour counter.

Bits 21:16 – HOUR[5:0] Hour Alarm
This field is the alarm field corresponding to the BCD-coded hour counter.

Bit 15 – MINEN Minute Alarm Enable

Value	Description
0	The minute-matching alarm is disabled.
1	The minute-matching alarm is enabled.

Bits 14:8 – MIN[6:0] Minute Alarm
This field is the alarm field corresponding to the BCD-coded minute counter.

Bit 7 – SECEN Second Alarm Enable

Value	Description
0	The second-matching alarm is disabled.
1	The second-matching alarm is enabled.

Bits 6:0 – SEC[6:0] Second Alarm
This field is the alarm field corresponding to the BCD-coded second counter.

23.6.6 RTC Calendar Alarm Register

Name: RTC_CALALR
Offset: 0x14
Reset: 0x01010000
Property: Read/Write

To change one of the DATE, MONTH fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_CALALR. The first access clears the enable corresponding to the field to change (DATEEN, MTHEN). If the field is already cleared, this access is not required. The second access performs the change of the value (DATE, MONTH). The third access is required to re-enable the field by writing 1 in DATEEN, MTHEN fields.

Bit	31	30	29	28	27	26	25	24
	DATEEN		DATE[5:0]					
Access	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Reset	0		0	0	0	0	0	1

Bit	23	22	21	20	19	18	17	16
	MTHEN			MONTH[4:0]				
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 31 – DATEEN Date Alarm Enable

Value	Description
0	The date-matching alarm is disabled.
1	The date-matching alarm is enabled.

Bits 29:24 – DATE[5:0] Date Alarm
This field is the alarm field corresponding to the BCD-coded date counter.

Bit 23 – MTHEN Month Alarm Enable

Value	Description
0	The month-matching alarm is disabled.
1	The month-matching alarm is enabled.

Bits 20:16 – MONTH[4:0] Month Alarm
This field is the alarm field corresponding to the BCD-coded month counter.

23.6.7 RTC Status Register

Name: RTC_SR
Offset: 0x18
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TDERR	CALEV	TIMEV	SEC	ALARM	ACKUPD
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 5 – TDERR Time and/or Date Free Running Error

Value	Name	Description
0	CORRECT	The internal free running counters are carrying valid values since the last read of the Status Register (RTC_SR).
1	ERR_TIMEDATE	The internal free running counters have been corrupted (invalid date or time, non-BCD values) since the last read and/or they are still invalid.

Bit 4 – CALEV Calendar Event
The calendar event is selected in the CALEVSEL field in the Control Register (RTC_CR) and can be any one of the following events: week change, month change and year change.

Value	Name	Description
0	NO_CALEVENT	No calendar event has occurred since the last clear.
1	CALEVENT	At least one calendar event has occurred since the last clear.

Bit 3 – TIMEV Time Event
The time event is selected in the TIMEVSEL field in the Control Register (RTC_CR) and can be any one of the following events: minute change, hour change, noon, midnight (day change).

Value	Name	Description
0	NO_TIMEVENT	No time event has occurred since the last clear.
1	TIMEVENT	At least one time event has occurred since the last clear.

Bit 2 – SEC Second Event

Value	Name	Description
0	NO_SECEVENT	No second event has occurred since the last clear.
1	SECEVENT	At least one second event has occurred since the last clear.

Bit 1 – ALARM Alarm Flag

Value	Name	Description
0	NO_ALARMEVENT	No alarm matching condition occurred.

Value	Name	Description
1	ALARMEVENT	An alarm matching condition has occurred.

Bit 0 – ACKUPD Acknowledge for Update

Value	Name	Description
0	FREERUN	Time and calendar registers cannot be updated.
1	UPDATE	Time and calendar registers can be updated.

23.6.8 RTC Status Clear Command Register

Name: RTC_SCCR
Offset: 0x1C
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TDERRCLR	CALCLR	TIMCLR	SECCLR	ALRCLR	ACKCLR
Access			W	W	W	W	W	W
Reset			–	–	–	–	–	–

Bit 5 – TDERRCLR Time and/or Date Free Running Error Clear

Value	Description
0	No effect.
1	Clears corresponding status flag in the Status Register (RTC_SR).

Bit 4 – CALCLR Calendar Clear

Value	Description
0	No effect.
1	Clears corresponding status flag in the Status Register (RTC_SR).

Bit 3 – TIMCLR Time Clear

Value	Description
0	No effect.
1	Clears corresponding status flag in the Status Register (RTC_SR).

Bit 2 – SECCLR Second Clear

Value	Description
0	No effect.
1	Clears corresponding status flag in the Status Register (RTC_SR).

Bit 1 – ALRCLR Alarm Clear

Value	Description
0	No effect.
1	Clears corresponding status flag in the Status Register (RTC_SR).

Bit 0 – ACKCLR Acknowledge Clear

Value	Description
0	No effect.
1	Clears corresponding status flag in the Status Register (RTC_SR).

23.6.9 RTC Interrupt Enable Register

Name: RTC_IER
Offset: 0x20
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TDERREN	CALEN	TIMEN	SECEN	ALREN	ACKEN
Access			W	W	W	W	W	W
Reset			–	–	–	–	–	–

Bit 5 – TDERREN Time and/or Date Error Interrupt Enable

Value	Description
0	No effect.
1	The time and date error interrupt is enabled.

Bit 4 – CALEN Calendar Event Interrupt Enable

Value	Description
0	No effect.
1	The selected calendar event interrupt is enabled.

Bit 3 – TIMEN Time Event Interrupt Enable

Value	Description
0	No effect.
1	The selected time event interrupt is enabled.

Bit 2 – SECEN Second Event Interrupt Enable

Value	Description
0	No effect.
1	The second periodic interrupt is enabled.

Bit 1 – ALREN Alarm Interrupt Enable

Value	Description
0	No effect.
1	The alarm interrupt is enabled.

Bit 0 – ACKEN Acknowledge Update Interrupt Enable

Value	Description
0	No effect.
1	The acknowledge for update interrupt is enabled.

23.6.10 RTC Interrupt Disable Register

Name: RTC_IDR
Offset: 0x24
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TDERRDIS	CALDIS	TIMDIS	SECDIS	ALRDIS	ACKDIS
Access			W	W	W	W	W	W
Reset			–	–	–	–	–	–

Bit 5 – TDERRDIS Time and/or Date Error Interrupt Disable

Value	Description
0	No effect.
1	The time and date error interrupt is disabled.

Bit 4 – CALDIS Calendar Event Interrupt Disable

Value	Description
0	No effect.
1	The selected calendar event interrupt is disabled.

Bit 3 – TIMDIS Time Event Interrupt Disable

Value	Description
0	No effect.
1	The selected time event interrupt is disabled.

Bit 2 – SECDIS Second Event Interrupt Disable

Value	Description
0	No effect.
1	The second periodic interrupt is disabled.

Bit 1 – ALRDIS Alarm Interrupt Disable

Value	Description
0	No effect.
1	The alarm interrupt is disabled.

Bit 0 – ACKDIS Acknowledge Update Interrupt Disable

Value	Description
0	No effect.
1	The acknowledge for update interrupt is disabled.

23.6.11 RTC Interrupt Mask Register

Name: RTC_IMR
Offset: 0x28
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TDERR	CAL	TIM	SEC	ALR	ACK
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 5 – TDERR Time and/or Date Error Mask

Value	Description
0	The time and/or date error event is disabled.
1	The time and/or date error event is enabled.

Bit 4 – CAL Calendar Event Interrupt Mask

Value	Description
0	The selected calendar event interrupt is disabled.
1	The selected calendar event interrupt is enabled.

Bit 3 – TIM Time Event Interrupt Mask

Value	Description
0	The selected time event interrupt is disabled.
1	The selected time event interrupt is enabled.

Bit 2 – SEC Second Event Interrupt Mask

Value	Description
0	The second periodic interrupt is disabled.
1	The second periodic interrupt is enabled.

Bit 1 – ALR Alarm Interrupt Mask

Value	Description
0	The alarm interrupt is disabled.
1	The alarm interrupt is enabled.

Bit 0 – ACK Acknowledge Update Interrupt Mask

Value	Description
0	The acknowledge for update interrupt is disabled.
1	The acknowledge for update interrupt is enabled.

23.6.12 RTC Valid Entry Register

Name: RTC_VER
Offset: 0x2C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					NVCALALR	NVTIMALR	NVCAL	NVTIM
Access					R	R	R	R
Reset					0	0	0	0

Bit 3 – NVCALALR Non-valid Calendar Alarm

Value	Description
0	No invalid data has been detected in RTC_CALALR (Calendar Alarm Register).
1	RTC_CALALR has contained invalid data since it was last programmed.

Bit 2 – NVTIMALR Non-valid Time Alarm

Value	Description
0	No invalid data has been detected in RTC_TIMALR (Time Alarm Register).
1	RTC_TIMALR has contained invalid data since it was last programmed.

Bit 1 – NVCAL Non-valid Calendar

Value	Description
0	No invalid data has been detected in RTC_CALR (Calendar Register).
1	RTC_CALR has contained invalid data since it was last programmed.

Bit 0 – NVTIM Non-valid Time

Value	Description
0	No invalid data has been detected in RTC_TIMR (Time Register).
1	RTC_TIMR has contained invalid data since it was last programmed.

23.6.13 RTC Milliseconds Register

Name: RTC_MSR
Offset: 0xD0
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							MS[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	MS[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – MS[9:0] Number of 1/1024 seconds elapsed within 1 second
This field reports a multiple of 1/1024 seconds. The field is synchronized with the fields reported in RTC_TIMR. Thus the MS field is cleared after each elapsed second. This field is incremented monotonically over a period of 1 second and can be used for timestamping with an accuracy better than 1 millisecond.

24. Real-time Timer (RTT)

24.1 Description

The Real-time Timer (RTT) is built around a 32-bit counter used to count roll-over events of the programmable 16-bit prescaler driven from the 32-kHz slow clock source. It generates a periodic interrupt and/or triggers an alarm on a programmed value.

The RTT can also be configured to be driven by the 1Hz RTC signal, thus taking advantage of a calibrated 1Hz clock.

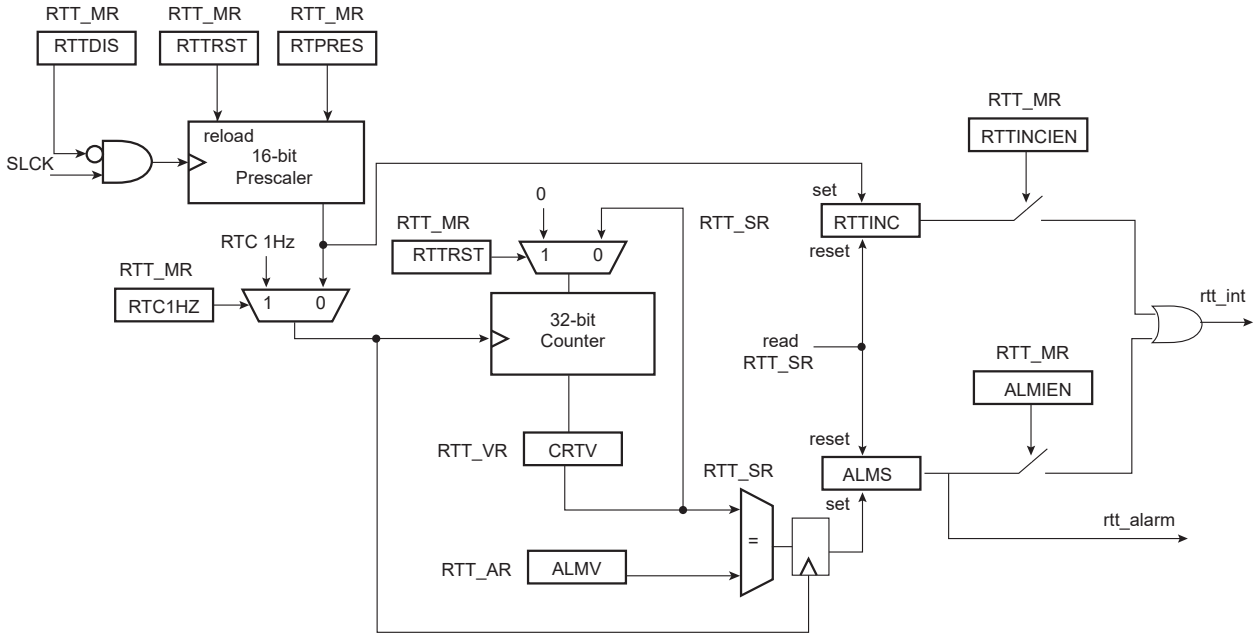
The slow clock source can be fully disabled to reduce power consumption when only an elapsed seconds count is required.

24.2 Embedded Characteristics

- 32-bit Free-running Counter on prescaled slow clock or RTC calibrated 1Hz clock
- 16-bit Configurable Prescaler
- Interrupt on Alarm or Counter Increment

24.3 Block Diagram

Figure 24-1. Real-time Timer Block Diagram



24.4 Functional Description

The programmable 16-bit prescaler value can be configured through the RTPRES field in the RTT Mode register (RTT_MR).

Configuring the RTPRES field value to 0x8000 (default value) corresponds to feeding the real-time counter with a 1Hz signal (if the slow clock is 32.768 kHz). The 32-bit counter can count up to 2^{32} seconds, corresponding to more than 136 years, then roll over to 0. Bit RTTINC in the RTT Status Register (RTT_SR) is set each time there is a prescaler roll-over.

The real-time 32-bit counter can also be supplied by the 1Hz RTC clock. This mode is interesting when the RTC 1Hz is calibrated (CORRECTION field ≠ 0 in RTC_MR) in order to guaranty the synchronism between RTC and RTT counters.

Setting the RTC1HZ bit in the RTT_MR drives the 32-bit RTT counter from the 1Hz RTC clock. In this mode, the RTPRES field has no effect on the 32-bit counter.

The prescaler roll-over generates an increment of the real-time timer counter if RTC1HZ = 0. Otherwise, if RTC1HZ = 1, the RTT counter is incremented every second. The RTTINC bit is set independently from the 32-bit counter increment.

The RTT can also be used as a free-running timer with a lower time-base. The best accuracy is achieved by writing RTPRES to 3 in RTT_MR.

Programming RTPRES to 1 or 2 is forbidden.

If the RTT is configured to trigger an interrupt, the interrupt occurs two slow clock cycles after reading the RTT_SR. To prevent several executions of the interrupt handler, the interrupt must be disabled in the interrupt handler and re-enabled when the RTT_SR is cleared.

The CRTV field can be read at any time in the RTT Value register (RTT_VR). As this value can be updated asynchronously with the Master Clock, the CRTV field must be read twice at the same value to read a correct value.

The current value of the counter is compared with the value written in the RTT Alarm register (RTT_AR). If the counter value matches the alarm, the ALMS bit in the RTT_SR is set. The RTT_AR is set to its maximum value (0xFFFFFFFF) after a reset.

The ALMS flag is always a source of the RTT alarm signal that may be used to exit the system from low power modes (see the Real-time Timer Block Diagram above).

The alarm interrupt must be disabled (ALMIEN must be cleared in RTT_MR) when writing a new ALMV value in the RTT_AR.

The RTTINC bit can be used to start a periodic interrupt, the period being one second when the RTPRES field value = 0x8000 and the slow clock = 32.768 kHz.

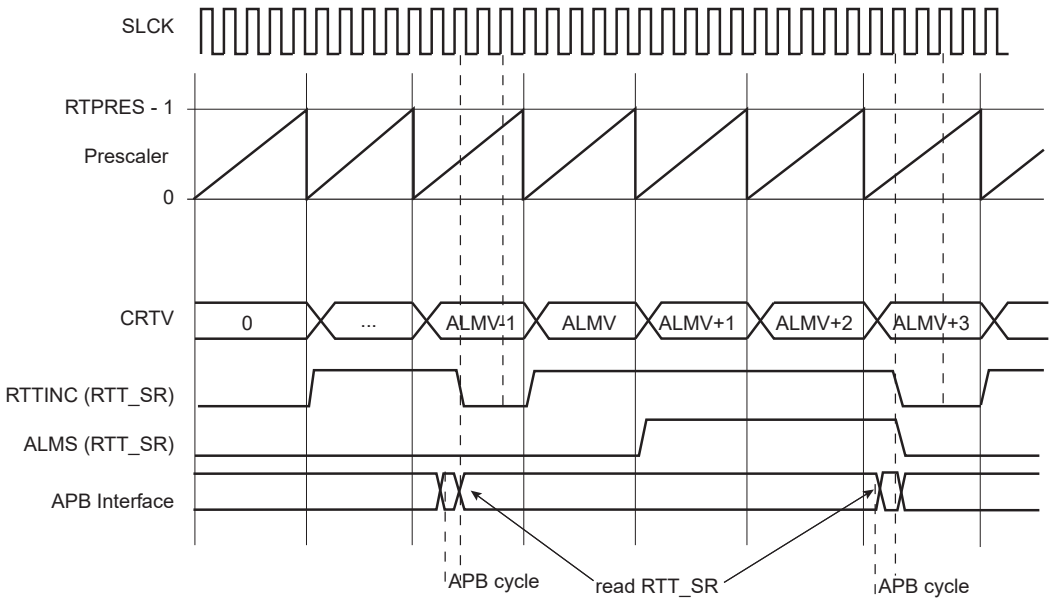
The RTTINCIEN bit must be cleared prior to writing a new RTPRES value in the RTT_MR.

Reading the RTT_SR automatically clears the RTTINC and ALMS bits.

Writing the RTTRST bit in the RTT_MR immediately reloads and restarts the clock divider with the new programmed value. This also resets the 32-bit counter.

When not used, the RTT can be disabled in order to suppress dynamic power consumption in this module. This can be achieved by setting the RTTDIS bit in the RTT_MR.

Figure 24-2. RTT Counting



24.5 Register Summary

Offset	Name	Bit Pos.								
0x00	RTT_MR	7:0	RTPRES[7:0]							
		15:8	RTPRES[15:8]							
		23:16				RTTDIS		RTTRST	RTTINCIEN	ALMIEN
		31:24								RTC1HZ
0x04	RTT_AR	7:0	ALMV[7:0]							
		15:8	ALMV[15:8]							
		23:16	ALMV[23:16]							
		31:24	ALMV[31:24]							
0x08	RTT_VR	7:0	CRTV[7:0]							
		15:8	CRTV[15:8]							
		23:16	CRTV[23:16]							
		31:24	CRTV[31:24]							
0x0C	RTT_SR	7:0							RTTINC	ALMS
		15:8								
		23:16								
		31:24								

24.5.1 Real-time Timer Mode Register

Name: RTT_MR
Offset: 0x00
Reset: 0x00008000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

Bit	31	30	29	28	27	26	25	24
								RTC1HZ
Access								R/W
Reset								0

Bit	23	22	21	20	19	18	17	16
				RTTDIS		RTTRST	RTTINCIEN	ALMIEN
Access				R/W		R/W	R/W	R/W
Reset				0		0	0	0

Bit	15	14	13	12	11	10	9	8
	RTPRES[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RTPRES[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 24 – RTC1HZ Real-Time Clock 1Hz Clock Selection

Value	Description
0	The RTT 32-bit counter is driven by the 16-bit prescaler roll-over events.
1	The RTT 32-bit counter is driven by the 1Hz RTC clock.

Bit 20 – RTTDIS Real-time Timer Disable

Value	Description
0	The RTT is enabled.
1	The RTT is disabled (no dynamic power consumption).

Bit 18 – RTTRST Real-time Timer Restart

Value	Description
0	No effect.
1	Reloads and restarts the clock divider with the new programmed value. This also resets the 32-bit counter.

Bit 17 – RTTINCIEN Real-time Timer Increment Interrupt Enable

Value	Description
0	The bit RTTINC in RTT_SR has no effect on interrupt.
1	The bit RTTINC in RTT_SR asserts interrupt.

Bit 16 – ALMIEN Alarm Interrupt Enable

Value	Description
0	The bit ALMS in RTT_SR has no effect on interrupt.
1	The bit ALMS in RTT_SR asserts interrupt.

Bits 15:0 – RTPRES[15:0] Real-time Timer Prescaler Value
Defines the number of SLCK periods required to increment the RTT. The RTTINCIEN bit must be cleared prior to writing a new RTPRES value.
RTPRES is defined as follows:

- RTPRES = 0: The prescaler period is equal to 2^{16} * SLCK periods.
- RTPRES = 1 or 2: forbidden.
- RTPRES ≠ 0,1 or 2: The prescaler period is equal to RTPRES * SLCK periods.

24.5.2 Real-time Timer Alarm Register

Name: RTT_AR
Offset: 0x04
Reset: 0xFFFFFFFF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

The alarm interrupt must be disabled (ALMIEN must be cleared in RTT_MR) when writing a new ALMV value.

Bit	31	30	29	28	27	26	25	24
	ALMV[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	ALMV[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	ALMV[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	ALMV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 31:0 – ALMV[31:0] Alarm Value
When the CRTV value in RTT_VR equals the ALMV field, the ALMS flag is set in RTT_SR. As soon as the ALMS flag rises, the CRTV value equals ALMV+1 (refer to the figure *RTT Counting* above).

24.5.3 Real-time Timer Value Register

Name: RTT_VR
Offset: 0x08
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	CRTV[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CRTV[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CRTV[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CRTV[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – CRTV[31:0] Current Real-time Value
Returns the current value of the RTT.
As CRTV can be updated asynchronously, it must be read twice at the same value.

24.5.4 Real-time Timer Status Register

Name: RTT_SR
Offset: 0x0C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							RTTINC	ALMS
Access							R	R
Reset							0	0

Bit 1 – RTTINC Prescaler Roll-over Status (cleared on read)

Value	Description
0	No prescaler roll-over occurred since the last read of the RTT_SR.
1	Prescaler roll-over occurred since the last read of the RTT_SR.

Bit 0 – ALMS Real-time Alarm Status (cleared on read)

Value	Description
0	The Real-time Alarm has not occurred since the last read of RTT_SR.
1	The Real-time Alarm occurred since the last read of RTT_SR.

25. Clock Generator

25.1 Description

The Clock Generator user interface is embedded within the Power Management Controller and is described in the Register Summary. However, the Clock Generator registers are named CKGR_.

25.2 Embedded Characteristics

The Clock Generator is made up of:

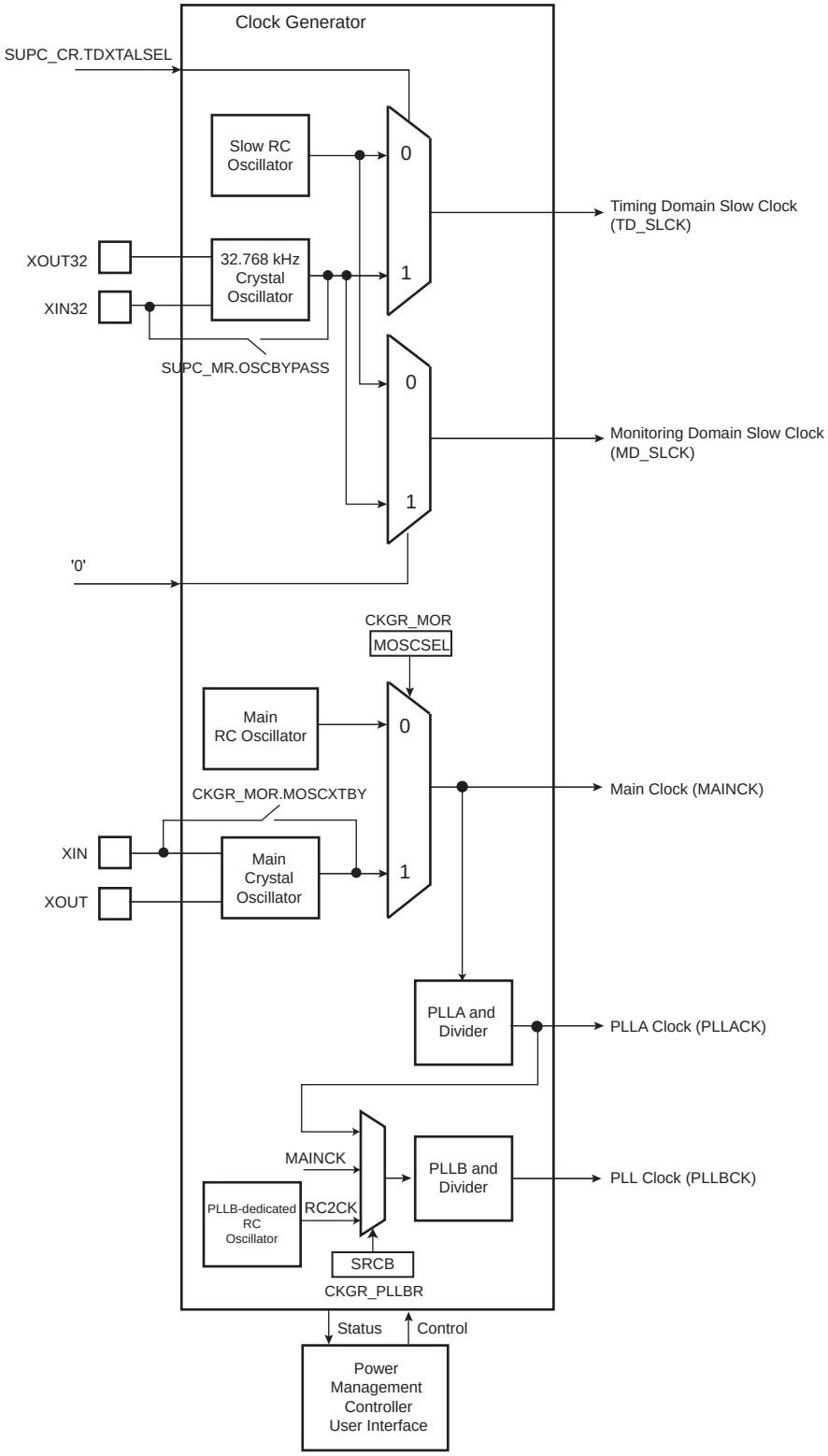
- A low-power 32.768 kHz Crystal Oscillator with Bypass Mode
- A low-power Slow RC Oscillator (32 kHz typical)
- A 3 MHz to 20 MHz Main Crystal Oscillator with Bypass Mode
- A Main RC Oscillator and a PLLB-dedicated RC Oscillator, each with Four Selectable Output Frequencies: 4/8/12/10 MHz. By default 4 MHz is selected.
- A 40 MHz to 200 MHz Programmable PLL (input from 8 MHz to 20 MHz)
- A 40 MHz to 200 MHz Programmable PLL (input from 8 MHz to 20 MHz)

It provides the following clocks:

- MD_SLCK— Monitoring Domain Slow clock. This clock, generally sourced from the Slow RC oscillator, feeds safety-critical functions of the device (WDT, RSTC, SUPC, frequency monitors and detectors, PMC start-up time counters).
- TD_CLK—Timing Domain Slow clock. This clock, generally sourced from the 32.768 kHz crystal oscillator, is routed to the RTC and RTT peripherals.
- MAINCK— Output of the Main clock oscillator selection: either the Main crystal oscillator or Main RC oscillator
- PLLACK— Output of the divider and the 40 MHz to 200 MHz programmable PLL (PLLA)
- PLLBCK— Output of the divider and the 40 MHz to 200 MHz programmable PLL (PLLB)

25.3 Block Diagram

Figure 25-1. Clock Generator Block Diagram



25.4 Slow Clock

The Supply Controller embeds a slow clock generator that is supplied with the VDDIO power supply. As soon as VDDIO is supplied, both the 32.768 kHz crystal oscillator and the Slow RC oscillator are powered, but only the Slow RC oscillator is enabled. This allows the monitoring Slow clock (MD_SLCK) to be valid in a short time (about 100 µs).

MD_SLCK is generated only by the Slow RC oscillator. TD_SLCK is generated either by the 32.768 kHz crystal oscillator or by the Slow RC oscillator.

To select the clock source, the selection is made via the TDXTALSEL bit in the Supply Controller Control register (SUPC_CR).

25.4.1 Slow RC Oscillator (32 kHz typical)

By default, the Slow RC oscillator is enabled and selected as a source of MD_SLCK and TD_SLCK. It is the only available source for MD_SCLK.

Compared to the 32.768 kHz crystal oscillator, this oscillator offers a faster start-up time and is less exposed to the external environment, as it is fully integrated. However, its output frequency is subject to larger variations with supply voltage, temperature and manufacturing process. Therefore, the user must take these variations into account when this oscillator is used as a time base (start-up counter, frequency monitor, etc.). Refer to the section “Electrical Characteristics”.

This oscillator is disabled by clearing the SUPC_CR.TDXTALSEL.

25.4.2 32.768 kHz Crystal Oscillator

By default, the 32.768 kHz oscillator is disabled. To use this oscillator, the XIN32 and XOUT32 pins must be connected to a 32.768 kHz crystal or to a ceramic resonator. Refer to the section “Electrical Characteristics” for appropriate loading capacitors selection on XIN32 and XOUT32.

Note that the user is not required to use the 32.768 kHz crystal oscillator and can use the Slow RC oscillator instead. Using the 32.768 kHz crystal oscillator provides a more accurate frequency than the Slow RC oscillator.

To select the 32.768 kHz crystal oscillator as the source of TD_SLCK, SUPC_CR.TDXTALSEL must be set. The crystal oscillator is directly connected to XIN32 and XOUT32. Note that if the source of TD_SLCK is the 32.768 kHz crystal oscillator, then the Slow RC oscillator is disabled.

If the user does not need the 32.768 kHz crystal oscillator, the XIN32 and XOUT32 pins must be pulled externally.

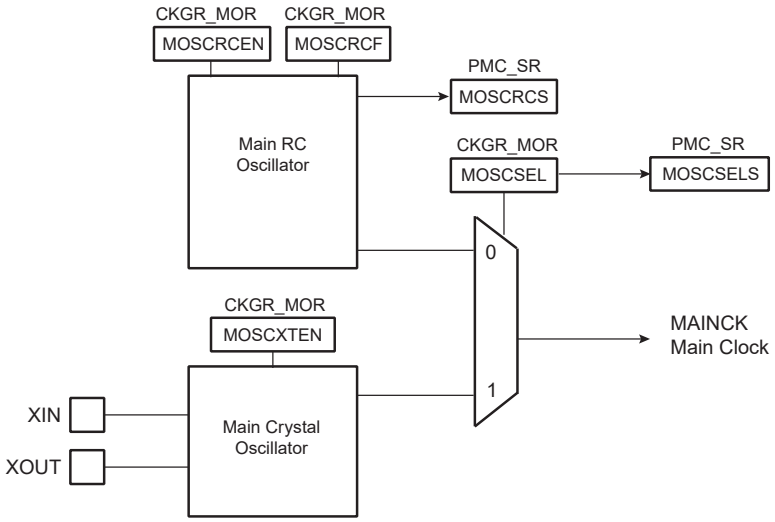
The user can also set the 32.768 kHz crystal oscillator in Bypass mode instead of connecting a crystal. In this case, the user must provide the external clock signal on XIN32. For input characteristics of the XIN32 pin, refer to the section “Electrical Characteristics”. To enter Bypass mode, the OSCBYPASS bit of the Supply Controller Mode register (SUPC_MR) must be set prior to setting SUPC_CR.TDXTALSEL.

25.5 Main Clock

The Main clock (MAINCK) has two sources:

- A Main RC oscillator (4/8/12/10 MHz) with a fast start-up time and that is selected by default to start the system. Its default frequency is 4 MHz.
- A Main crystal oscillator with Bypass mode

Figure 25-2. Main Clock (MAINCK) Block Diagram



25.5.1 Main RC Oscillator

After reset, the Main RC oscillator is enabled with the 4 MHz frequency selected. This oscillator is selected as the source of MAINCK. MAINCK is the default clock selected to start the system.

The software can disable or enable the Main RC oscillator with the MOSCRCEN bit in the Clock Generator Main Oscillator register (CKGR_MOR).

The output frequency of the Main RC oscillator can be selected among 4/8/12/10 MHz. Selection is done by configuring MOSCRCF in CKGR_MOR. When changing the frequency selection, the MOSCRCS bit in the Status register (PMC_SR) is automatically cleared and MAINCK is stopped until the oscillator is stabilized. Once the oscillator is stabilized, MAINCK restarts and PMC_SR.MOSCRCS is set. Note that enabling the Main RC oscillator (MOSCRCEN = 1) and changing its frequency (MOSCRCF) at the same time is not allowed.

This oscillator must be enabled first and its frequency changed in a second step.

When disabling the Main RC oscillator by clearing CKGR_MOR.MOSCRCEN, the PMC_SR.MOSCRCS bit is automatically cleared, indicating that the oscillator is OFF.

Setting the MOSCRCS bit in the Interrupt Enable register (PMC_IER) triggers an interrupt to the processor.

25.5.2 PLLB-dedicated RC Oscillator

The system embeds a second RC oscillator that is dedicated to driving the PLLB reference clock. It is enabled by setting the EN bit in the PMC_OSC2 register.

The output frequency of the PLLB-dedicated RC oscillator can be selected among 4/8/12/10 MHz. Selection is done by configuring the field OSCRCF in PMC_OSC2 register. The RC oscillator is enabled by default.

The frequency selection must not be performed while the PLLB is operating with this RC oscillator as reference clock. The PLLB must first be disabled to change this RC oscillator frequency.

25.5.3 Main RC Oscillator Frequency Adjustment

The Main RC oscillator frequencies of 4, 8, 12 and 10 MHz are factory-centered to the typical values by using Flash calibration bits (refer to the section "Electrical Characteristics"). The Flash calibration bit settings stored for the Main RC oscillator frequencies vary from device to device.

To get a starting point when changing the CAL4, CAL8, CAL10 or CAL12 fields in the PMC_OCR1 register, it is recommended to first read their corresponding Flash calibration bits in the Flash controller.

The user can adjust the value of the Main RC oscillator frequency by modifying the trimming values set in production for the Main RC Oscillator frequencies. This may be used to compensate frequency drifts due to temperature or voltage. Values written by the user application in the Oscillator Calibration register (PMC_OCR1) are reset after each

power-up or software reset. By default, the SELn bits are cleared, so the Main RC oscillator is driven with the factory-programmed Flash calibration bits which are programmed during chip production.

In order to calibrate the oscillator lower frequency of 4MHz , SEL4 must be set to `1' and a valid trim value must be configured in CAL4. Likewise, SEL8, SEL10 or SEL12 must be set to `1' and a trim value must be configured in CAL8, CAL10 or CAL12 in order to adjust the other frequencies of the oscillator.

It is not possible to adjust the oscillator frequency while operating from this oscillator.

When trimming one frequency, the selected frequency should not be the same as the one selected in CKGR_MOR.MOSCRCF. For instance, when writing SEL4 and CAL4 bits to modify the 4 MHz clock option, the selected frequency in CKGR_MOR.MOSCRCF must be either 8, 12 or 10 MHz

Prior to write the new trim value in CALn fields in PMC_OCR1 register, the bit EN_WR_CALIB in PMC_OSC2 must be set to '1' in order to enable write operation in calibration register.

At any time, the user can measure the main RC oscillator output frequency by means of the Main Frequency Counter (refer to Section 25.5.7 "Main Frequency Counter"). Once the frequency measurement is done, the main RC oscillator calibration fields (CALn) can be adjusted accordingly to correct this oscillator output frequency.

Trim values in the CALn bitfields are composed of two trim parameters: temperature and frequency. CALn[1:0] are used for RC temperature trim, while CALn[4:2] are used for frequency trim. CALn[6:5] are not used

25.5.4 Second RC Oscillator Frequency Adjustment

The Second RC oscillator frequencies of 4, 8, 12 and 10 MHz are factory-centered to the typical values by using Flash calibration bits (refer to the section "Electrical Characteristics"). The Flash calibration bit settings stored for the Second RC oscillator frequencies vary from device to device.

To get a starting point when changing the CAL4, CAL8, CAL10 or CAL12 fields, it is recommended to first read their corresponding Flash calibration bits in the Flash controller.

The user can adjust the value of the Second RC oscillator frequency by modifying the trimming values set in production for the Second RC Oscillator frequencies. This may be used to compensate frequency drifts due to temperature or voltage. Values written by the user application in the Oscillator Calibration register (PMC_OCR2) are reset after each power-up or software reset. By default, the SELn bits are cleared, so the Main RC oscillator is driven with the factory-programmed Flash calibration bits which are programmed during chip production.

In order to calibrate the oscillator lower frequency of 4MHz , SEL4 must be set to `1' and a valid trim value must be configured in CAL4. Likewise, SEL8, SEL10 or SEL12 must be set to `1' and a trim value must be configured in CAL8, CAL10 or CAL12 in order to adjust the other frequencies of the oscillator.

It is not possible to adjust the oscillator frequency while operating from this oscillator.

When trimming one frequency, the selected frequency should not be the same as the one selected in PMC_OSC2.OSCRCF. For instance, when writing SEL4 and CAL4 bits to modify the 4 MHz clock option, the selected frequency in PMC_OSC2.OSCRCF must be either 8, 12 or 10 MHz

Prior to write the new trim value in CALn fields in PMC_OCR2 register, the bit EN_WR_CALIB in PMC_OSC2 must be set to '1' in order to enable write operation in calibration register.

Trim values in the CALn bitfields are composed of two trim parameters: temperature and frequency. CALn[1:0] are used for RC temperature trim, while CALn[4:2] are used for frequency trim. CALn[6:5] are not used.

25.5.5 Main Crystal Oscillator

After reset, the Main crystal oscillator is disabled and is not selected as the source of MAINCK.

As the source of MAINCK, the Main crystal oscillator provides a very precise frequency. The software enables or disables this oscillator in order to reduce power consumption via CKGR_MOR.MOSCXTEN.

When disabling this oscillator by clearing the CKGR_MOR.MOSCXTEN, PMC_SR.MOSCXTS is automatically cleared, indicating the oscillator is off.

When enabling this oscillator, the user must initiate the start-up time counter. The start-up time depends on the characteristics of the external device connected to this oscillator.

When CKGR_MOR.MOSCXTEN and CKGR_MOR.MOSCXTST are written to enable this oscillator, PMC_SR.MOSCXTS is cleared and the counter starts counting down on MD_SLCK divided by 8 from the

CKGR_MOR.MOSCXTST value. Since the CKGR_MOR.MOSCXTST value is coded with 8 bits, the start-up time can be programmed up to 65536 MD_SLCK periods, corresponding to about 62 ms when running at 32.768 kHz.

When the start-up time counter reaches ‘0’, PMC_SR.MOSCXTS is set, indicating that the oscillator is stabilized. Setting the MOSCXTS bit in the Interrupt Mask register (PMC_IMR) can trigger an interrupt to the processor.

25.5.6 Main Clock Source Selection

The source of MAINCK can be selected from the following:

- the Main RC oscillator
- the Main crystal oscillator
- an external clock signal provided on the XIN input (Bypass mode of the Main crystal oscillator)

The advantage of the Main RC oscillator is its fast start-up time. By default, this oscillator is selected to start the system and it must be selected prior to entering Wait mode.

The advantage of the Main crystal oscillator is its high level of accuracy.

The selection of the oscillator is made with bit CKGR_MOR.MOSCSEL. The switchover of the MAINCK source is glitch-free, so there is no need to run MCK out of MD_SLCK, PLLACK or PLLBCK in order to change the selection. PMC_SR.MOSCSELS indicates when the switch sequence is done.

If a PLL is selected as the source of MCK, the system automatically switches the MCK source to MAINCK during the switching operation, then sets the MCK source back to the selected PLL after the switch is complete. This sequence happens under any one of the following cases:

- switching from the Main RC oscillator to the Main Crystal oscillator;
- switching from the Main Crystal oscillator to the Main RC oscillator;
- changing the output frequency of the Main RC oscillator.

Setting PMC_IMR.MOSCSELS triggers an interrupt to the processor.

25.5.7 Bypassing the Main Crystal Oscillator

Prior to bypassing the Main crystal oscillator, the external clock frequency provided on the XIN pin must be stable and within the values specified in the XIN Clock characteristics in the section “Electrical Characteristics”.

The sequence is as follows:

1. Ensure that an external clock is connected on XIN.
2. Enable the bypass by setting CKGR_MOR.MOSCXTBY.
3. Disable the Main crystal oscillator by clearing CKGR_MOR.MOSCXTEN.

25.5.8 Main Frequency Counter

The Main frequency counter measures the Main RC oscillator and the Main crystal oscillator against the MD_SLCK and is managed by CKGR_MCFR.

During the measurement period, the Main frequency counter increments at the speed of the clock defined by the bit CKGR_MCFR.CCSS.

A measurement is started in the following cases:

- When CKGR_MCFR.RCMEAS is written to ‘1’.
- When the Main RC oscillator is selected as the source of MAINCK and when this oscillator is stable (i.e., when the MOSCRCS bit is set)
- When the Main crystal oscillator is selected as the source of MAINCK and when this oscillator is stable (i.e., when the MOSCXTS bit is set)
- When MAINCK source selection is modified

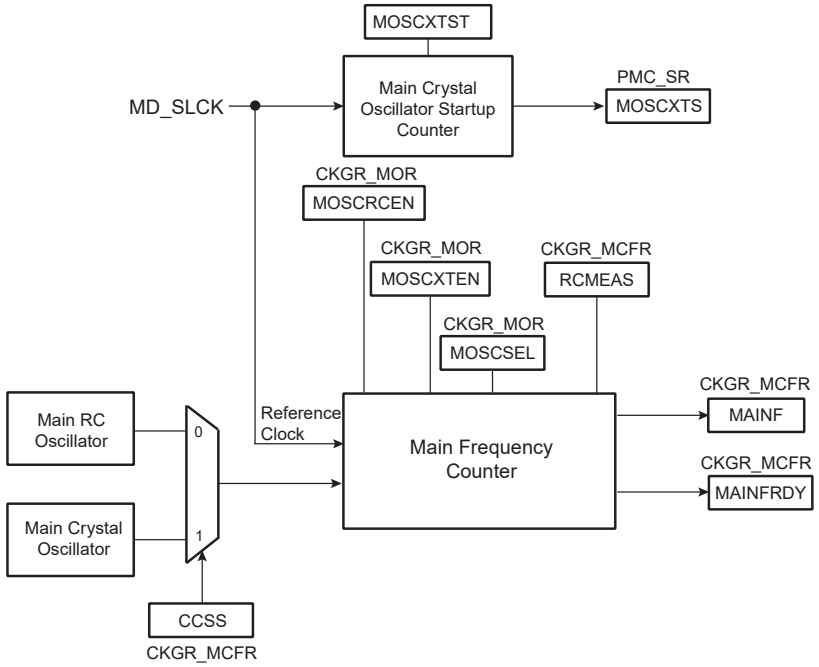
The measurement period ends at the 16th falling edge of MD_SLCK, the MAINFRDY bit in CKGR_MCFR is set and the counter stops counting. Its value can be read in the MAINF field of CKGR_MCFR and gives the number of clock cycles during 16 periods of MD_SLCK, so that the frequency of the Main RC oscillator or Main crystal oscillator can be determined.

If switching the source of MAINCK to the Main crystal oscillator from the Main RC oscillator, follow the programming sequence below to ensure that the oscillator is present and that its frequency is valid:

1. Enable the Main crystal oscillator by setting CKGR_MOR.MOSCXTEN. Configure the CKGR_MOR. MOSCXTST field with the Main crystal oscillator start-up time as defined in the section “Electrical Characteristics”.
2. Wait for PMC_SR.MOSCXTS flag to rise, indicating the end of a start-up period of the Main crystal oscillator.
3. Select the Main crystal oscillator as the source clock of the Main frequency counter by setting CKGR_MCFR.CCSS.
4. Initiate a frequency measurement by setting CKGR_MCFR.RCMEAS.
5. Read CKGR_MCFR.MAINFRDY until its value equals 1.
6. Read CKGR_MCFR.MAINF and compute the value of the Main crystal frequency.

If the MAINF value is valid, software can switch MAINCK to the Main crystal oscillator. Refer to Main Clock Source Selection.

Figure 25-3. Main Frequency Counter Block Diagram

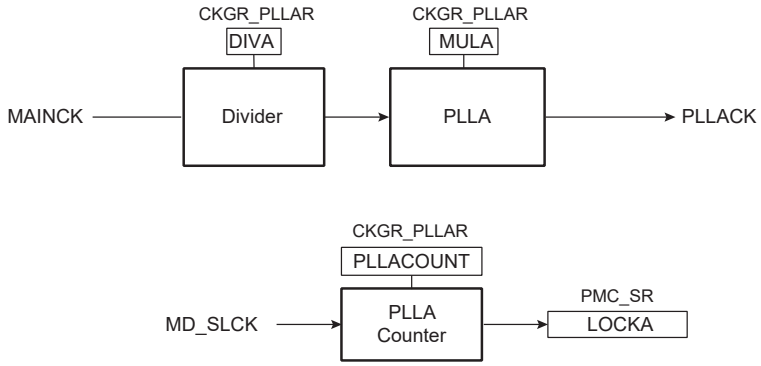


25.6 PLLA Clock

The PLLA clock (PLLACK) is generated from MAINCK by the PLLA and a predivider. This combination allows a wide range of frequencies.

The figure below shows the block diagram of the dividers and PLLA blocks.

Figure 25-4. Divider and PLLA Block Diagram

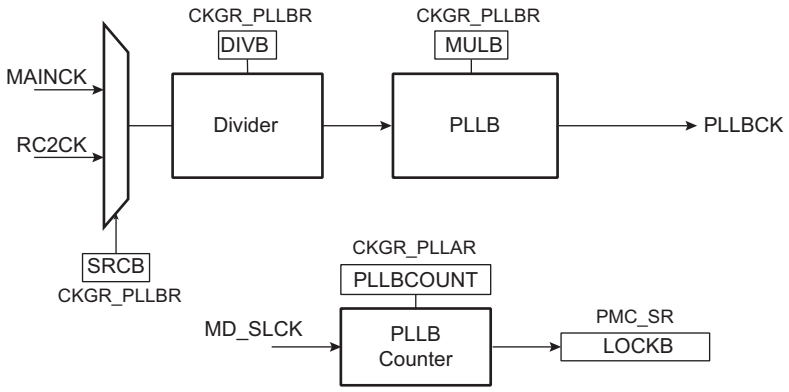


25.7 PLLB Clock

The PLLB clock (PLLBCK) is generated by the PLLB from MAINCK, or RC2CK and a predivider. This combination allows a wide range of frequencies.

The figure below shows the block diagram of the dividers and PLLB blocks.

Figure 25-5. Divider and PLLB Block Diagram



25.8 Divider and Phase Lock Loop Programming

The valid input frequency range of the PLLA is 8 MHz to 20 MHz and the valid output frequency range of the PLLA is 40 MHz to 200 MHz. The multiplier factor (MULA ratio) between the output and the input can be set in the range 1 to 25 in steps of 1.

The valid input frequency range of the PLLB is 8 MHz to 20 MHz and the valid output frequency range of the PLLA is 40 MHz to 200 MHz. The multiplier factor (MULB ratio) between the output and the input can be set in the range 1 to 25 in steps of 1.

The PLL (PLLA or PLLB) must first be correctly configured:

- Select the correct VCO frequency range depending on the output frequency requested (CKGR_PLLAR.FREQVCO and CKGR_PLLBR.FREQVCO)
- Select the correct value of current of the charge pump of PLL (PMC_PLLCFG.OUTCUR_PLLA and PMC_PLLCFG.OUTCUR_PLLB),
- Select the correct resistors and capacitors values embedded inside the PLL to guarantee its stability (PMC_PLLCFG.FLT_PLLA and PMC_PLLCFG.FLT_PLLB),

When a divider field (DIV) is cleared, the output of the corresponding divider and the PLL output is a continuous signal at level 0. On reset, each DIV field is cleared, thus the corresponding PLL input clock is stuck at ‘0’.

The PLL (PLLA or PLLB) allows multiplication of the divider's outputs. The PLL clock signal has a frequency that depends on the respective source signal frequency and on the parameters DIV (DIVA or DIVB) and MUL (MULA or MULB, in the range of 1 to 25). The factor applied to the source signal frequency is (MUL + 1)/DIV. When MUL is written to '0' or DIV = 0, the PLL is disabled and its power consumption is saved. Note that there is a delay of two MD_SLCK clock cycles between the disable command and the real disable of the PLL. Re-enabling the PLL can be performed by writing a value higher than '0' in the MUL field and DIV higher than '0'.

Whenever the PLL is re-enabled or one of its parameters is changed, the LOCK (LOCKA or LOCKB) bit in PMC_SR is automatically cleared. The values written in the PLLCOUNT field (PLLACOUNT or PLLBCOUNT) in CKGR_PLLR (CKGR_PLLAR or CKGR_PLLBR) are loaded in the PLL counter. The PLL counter then decrements at the speed of MD_SLCK until it reaches '0'. At this time, PMC_SR.LOCKA or PMC_SR.LOCKB is set and can trigger an interrupt to the processor. The user has to load the number of MD_SLCK cycles required to cover the PLL transient time into the PLLCOUNT field.

To avoid programming the PLL with a multiplication factor that is too high, the user can saturate the multiplication factor value sent to the PLL by setting the PLLA_MMAX and PLLB_MMAX field in the PLL Maximum Multiplier Value register (PMC_PMMR).

It is forbidden to change the MAINCK characteristics (oscillator selection, frequency adjustment of the Main RC oscillator) when:

- MAINCK is selected as the PLLA or PLLB clock source, and
- MCK is sourced from PLLA.

To change the MAINCK characteristics, the user must:

1. Switch the MCK source to MAINCK by writing a '1' to PMC_MCKR.CSS.
2. Change the Main RC oscillator frequency (MOSCRCF) or oscillator selection (MOSCSEL) in CKGR_MOR.
3. Wait for MOSCRCS (if frequency changes) or MOSCSELS (if oscillator selection changes) in PMC_SR.
4. Disable and then enable the PLL.
5. Wait for the LOCK flag in PMC_SR.
6. Switch back MCK to the PLLA by writing the appropriate value to PMC_MCKR.CSS.

26. Power Management Controller (PMC)

26.1 Description

The Power Management Controller (PMC) optimizes power consumption by controlling user peripheral clocks. The PMC enables/disables the clock inputs to many of the peripherals.

The Supply Controller selects the source of TD_SLCK of the real-time part (RTT/RTC) and the MD_SLCK of the rest of the System Controller (watchdog, etc.) separately. If the two sections are fed by the same clock, the unused oscillator is disabled automatically so that power consumption is optimized.

By default, at start-up, the chip runs out of MCK using the Main RC oscillator running at 4 MHz.

26.2 Embedded Characteristics

The Power Management Controller provides the following clocks:

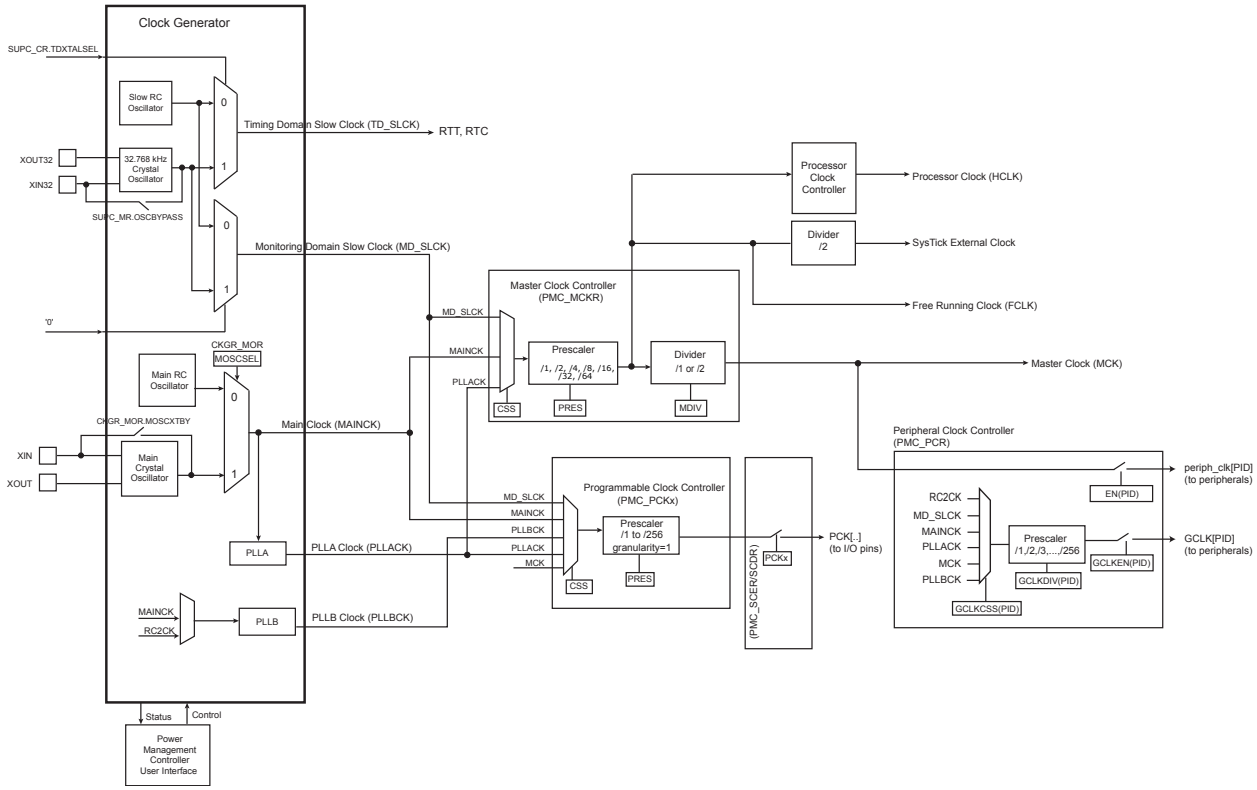
- Master Clock (MCK), programmable from a few hundred Hz to the maximum operating frequency of the device. It is available to the modules running permanently.
- Processor Clock (HCLK)
- Free-running processor Clock (FCLK)
- Cortex-M7 SysTick external clock
- Peripheral Clocks with independent ON/OFF control, provided to the peripherals
- Programmable Clock Outputs (PCKx), selected from the clock generator outputs to drive the device PCK pins
- Generic Clock (GCLK) with controllable division and ON/OFF control, dependent or not on MCK and HCLK, and provided to selected peripherals.

The Power Management Controller also provides the following features on clocks:

- A Main crystal oscillator failure detector
- A 32.768 kHz crystal oscillator frequency monitor
- A frequency counter on Main crystal oscillator or Main RC oscillator
- Two embedded RC oscillators with on-the-fly adjustable frequencies

26.3 Block Diagram

Figure 26-1. General Clock Distribution Block Diagram



Note: For clock frequencies, refer to the section “Electrical Characteristics”.

Note: The maximum frequency for HCLK is 100 MHz, and for MCK is 50 MHz.

Note: From 0 to 50 MHz, HCLK can either equal MCK or 2*MCK. Above 50 MHz, HCLK is always equal to 2*MCK .

26.4 Master Clock Controller

The Master Clock Controller provides the Master Clock (MCK) with the selection and division of the clock generator's output signals. MCK is the source clock of the peripheral clocks.

The clock to be selected between MD_SLCK, MAINCK, PLLACK is configured in PMC_MCKR.CSS. The prescaler supports the 1, 2, 3, 4, 8, 16, 32, 64 division factors and is configured using PMC_MCKR.PRES.

Each time PMC_MCKR is configured to define a new MCK, the MCKRDY bit is cleared in PMC_SR. It reads ‘0’ until MCK is established. Then, the MCKRDY bit is set and can trigger an interrupt to the processor. This feature is useful when switching from a high-speed clock to a lower one to inform the software when the change is completed.

Note: It is forbidden to modify MDIV and CSS at the same access. Each field must be modified separately with a wait for the MCKRDY flag between the first field modification and the second field modification.

26.5 SysTick External Clock

When the processor selects the SysTick external clock, the calibration value is fixed to 50000. This allows the generation of a time base of 1 ms with the SysTick clock at the maximum frequency on MCK divided by 2.

Refer to the section “ARM Cortex-M7 (ARM)” for details on selecting the SysTick external clock.

26.6 Core and Bus Independent Clocks for Peripherals

Some peripherals can operate while the core, bus and peripheral clock frequencies are modified, thus providing communications at a bit rate which is independent for the core/bus/peripheral clock. This mode of operation is possible by using the internally-generated independent clock sources. See Peripheral and Generic Clock Controller. For a list of peripherals supporting the independent clock (GCLK), refer to table “Peripheral Identifiers”, in section “Peripherals”.

26.7 Peripheral and Generic Clock Controller

The PMC controls the clocks of the embedded peripherals by means of the Peripheral Control register (PMC_PCR). With this register, the user can enable and disable the different clocks used by the peripherals:

- Peripheral clocks (periph_clk[PID]), routed to every peripheral and derived from the master clock (MCK), and
- Generic clocks (GCLK[PID]), routed to selected peripherals only (refer to the Peripheral Identifiers table in section Peripherals). These clocks are independent of the core and bus clocks (HCLK, MCK and periph_clk[PID]). They are generated by selection and division of the following sources: RC2CK, MD_SLCK, MAINCK, PLLBCK, PLLACK and MCK. Refer to the description of each peripheral for the limitation to be applied to GCLK[PID] compared to periph_clk[PID].

To configure a peripheral’s clocks, PMC_PCR.CMD must be written to ‘1’ and PMC_PCR.PID must be written with the index of the corresponding peripheral. All other configuration fields must be correctly set.

To read the current clock configuration of a peripheral, PMC_PCR.CMD must be written to ‘0’ and PMC_PCR.PID must be written with the index of the corresponding peripheral regardless of the values of other fields. This write does not modify the configuration of the peripheral. The PMC_PCR can then be read to know the configuration status of the corresponding PID.

The status of the peripheral clock activity can be read in the Peripheral Clock Status registers (PMC_CSRx).

When a periph_clk or a GCLK is disabled, it is immediately stopped.

It is forbidden to modify the source or the division ratio of a GCLK when it is in use. It must first be disabled.

To stop a peripheral clock, it is recommended that the system software wait until the peripheral has executed its last programmed operation before disabling the clock. This is to avoid data corruption or erroneous behavior of the system.

The bit number in PMC_CSRx is the Peripheral Identifier defined at the product level. The bit number corresponds to the interrupt source number assigned to the peripheral.

26.8 Free-running Processor Clock

The free-running Processor clock (FCLK) is used for sampling interrupts and clocking debug blocks.

26.9 Programmable Clock Output Controller

The PMC controls four signals to be output on the external pins PCKx. Each signal can be independently programmed via the Programmable Clock registers (PMC_PCKx).

PCKx can be independently selected between MD_SLCK, MAINCK, PLLACK, PLLBCK and MCK by configuring PMC_PCKx.CSS. Each output signal can also be divided by 1 to 256 by configuring PMC_PCKx.PRES.

Each output signal can be enabled and disabled by writing a ‘1’ to the corresponding bits PMC_SCER.PCKx and PMC_SCDR.PCKx, respectively. The status of the active programmable output clocks is given in PMC_SCSR.PCKx.

The status flag PMC_SR.PCKRDYx indicates that PCKx is actually what has been programmed in registers PMC_PCKx.

As the Programmable Clock Controller does not manage with glitch prevention when switching clocks, it is strongly recommended to disable PCKx before any configuration change and to re-enable it after the change is performed.

26.10 Main Crystal Oscillator Failure Detection

The Main crystal oscillator failure detector monitors the Main crystal oscillator against the Slow RC oscillator and provides an automatic switchover of the MAINCK source to the Main RC oscillator in case of failure detection.

The failure detector can be enabled or disabled by configuring CKGR_MOR.CFDEN. It is also disabled in either of the following cases:

- after a VDDCORE reset
- when the Main crystal oscillator is disabled (MOSCXTEN = 0)

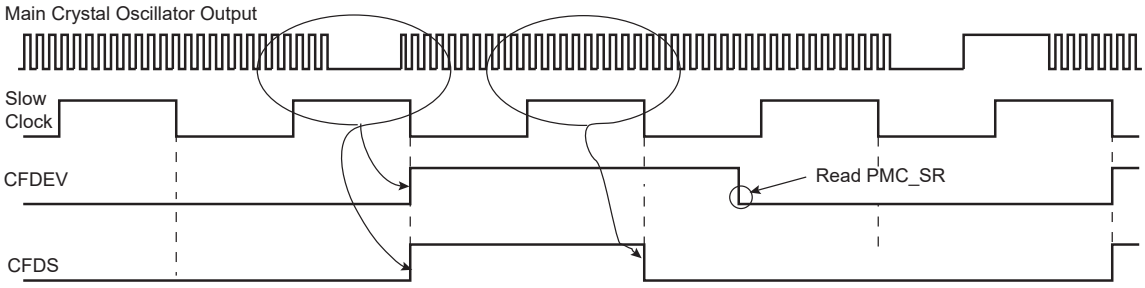
A failure is detected by means of a counter incrementing on the Main crystal oscillator output and detection logic is triggered by the Slow RC oscillator which is automatically enabled when CFDEN = 1.

The counter is cleared when the Slow RC oscillator clock signal is low and enabled when the signal is high. Thus, the failure detection time is one Slow RC oscillator period. If, during the high level period of the Slow RC oscillator clock signal, less than eight Main crystal oscillator clock periods have been counted, then a failure is reported. Note that when enabling the failure detector, up to two cycles of the Slow RC oscillator are needed to detect a failure of the Main crystal oscillator.

If a failure of Main crystal oscillator is detected, PMC_SR.CFDEV and PMC_SR.FOS both indicate a failure event. PMC_SR.CFDEV is cleared on read of PMC_SR, and PMC_SR.FOS is cleared by writing a ‘1’ to the FOCLR bit in the Fault Output Clear register (PMC_FOCR).

Only PMC_SR.CFDEV can generate an interrupt if the corresponding interrupt source is enabled in PMC_IER. The current status of the clock failure detection can be read at any time from PMC_SR.CFDS.

Figure 26-2. Clock Failure Detection Example



Note: Ratio of clock periods is for illustration purposes only.

If the Main crystal oscillator is selected as the source clock of MAINCK (CKGR_MOR.MOSCSEL = 1), and if the MCK source is PLLACK (CSS = 2), a clock failure detection automatically forces MAINCK to be the source clock for MCK. Then, regardless of the PMC configuration, a clock failure detection automatically forces the Main RC oscillator to be the source clock for MAINCK. If the Main RC oscillator is disabled when a clock failure detection occurs, it is automatically re-enabled by the clock failure detection mechanism.

Two Slow RC oscillator clock cycles are necessary to detect and switch from the Main crystal oscillator to the Main RC oscillator if the source of MCK is MAINCK, or three Slow RC oscillator clock cycles if the source of MCK is PLLACK.

A clock failure detection activates a fault output that is connected to the Pulse Width Modulator (PWM) Controller. With this connection, the PWM controller is able to force its outputs and to protect the driven device, if a clock failure is detected.

26.11 32.768 kHz Crystal Oscillator Frequency Monitor

The frequency of the 32.768 kHz crystal oscillator can be monitored by means of logic driven by the Main RC oscillator known as a reliable clock source. This function is enabled by configuring the XT32KFME bit of CKGR_MOR. Prior to enabling this frequency monitor, the 32.768 kHz crystal oscillator must be started and its start-up time be elapsed. Refer to details on the Slow clock generator in the section “Supply Controller (SUPC)”.

An error flag (XT32KERR in PMC_SR) is asserted when the 32.768 kHz crystal oscillator frequency is out of the ±10% nominal frequency value (i.e., 32.768 kHz). The error flag can be cleared only if the frequency monitor is disabled.

When the Main RC oscillator frequency is set to 4 MHz, the accuracy of the measurement is ±40% as this frequency is not trimmed during production. Therefore, ±10% accuracy is obtained only if the Main RC oscillator frequency is configured for 8 or 12 MHz.

The monitored clock frequency is declared invalid if at least 4 consecutive clock period measurement results are over the nominal period ±10%. Note that modifying the trimming values of the Main RC oscillator (PMC_OCR1) may impact the monitor accuracy and lead to inappropriate failure detection.

Due to the possible frequency variation of the Main RC oscillator acting as reference clock for the monitor logic, any 32.768 kHz crystal frequency deviation over ±10% of the nominal frequency is systematically reported as an error by means of PMC_SR.XT32KERR. Between -1% and -10% and +1% and +10%, the error is not systematically reported.

Thus only a crystal running at 32.768 kHz frequency ensures that the error flag will not be asserted. The permitted drift of the crystal is 10000 ppm (1%), which allows any standard crystal to be used.

If the Main RC oscillator frequency range needs to be changed while the frequency monitor is operating, the monitoring must be stopped prior to change the Main RC oscillator frequency. Then it can be re-enabled as soon as PMC_SR.MOSCRCS is set.

The error flag can be defined as an interrupt source of the PMC by setting PMC_IER.XT32KERR. This flag is also routed to the RSTC and may generate a reset of the device.

26.12 CPU Frequency Monitor

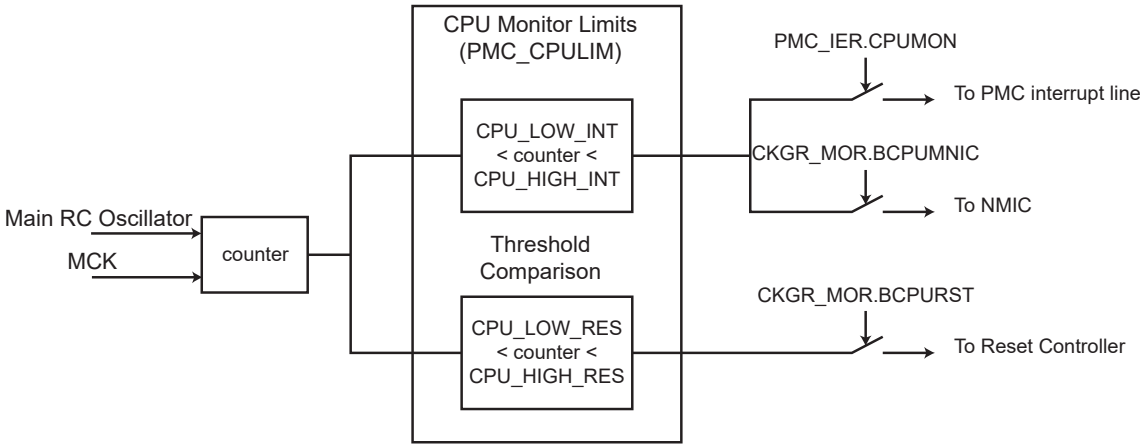
The frequency of the processor clock (HCLK) can be monitored through MCK with the Main RC oscillator. This monitoring can only be performed if the MCK frequency is at least 3 times faster than the embedded Main RC oscillator. This function is enabled by writing a ‘1’ to CKGR_MOR.BCPUNMIC, CKGR_MOR.BCPURST, or PMC_IER.CPUMON.

An error on the CPU frequency can lead to a PMC interrupt, an NMIC interrupt or a reset of the system (refer to sections “Non-Maskable Interrupt Controller (NMIC)” and/or “Reset Controller (RSTC)”).

When the corresponding PMC interrupt is enabled, the status of the CPU monitoring can be read on the PMC_SR.CPUMON. This status is cleared on read.

Once enabled, the monitor continuously counts the number of MCK cycles within 15 cycles of the embedded Main RC oscillator. The result is then compared to threshold values defined in the PMC_CPULIM register. Two levels of thresholds can be defined: a first couple of values defines thresholds to generate an interrupt and a second couple of values defines thresholds to generate a reset of the system.

Figure 26-3. CPU Frequency Monitor Block Diagram



26.13 Recommended Programming Sequence

Follow the steps below to program the PMC:

1. If the Main crystal oscillator is not required, the PLLA and divider can be directly configured (Step 6.) else this oscillator must be started (Step 2.).
2. Enable the Main crystal oscillator by setting CKGR_MOR.MOSCXTEN. The user can define a start-up time. This can be done by configuring CKGR_MOR.MOSCXTST. Once this register has been correctly configured, the user must wait for PMC_SR.MOSCXTS to be set. This can be done either by polling PMC_SR.MOSCXTS, or by waiting for the interrupt line to be raised if the associated interrupt source (MOSCXTS) has been enabled in PMC_IER.
3. Switch MAINCK to the Main crystal oscillator by setting CKGR_MOR.MOSCSEL.
4. Wait for PMC_SR.MOSCSELS to be set to ensure the switch is complete.
5. Check MAINCK frequency:
   This frequency can be measured via CKGR_MCFR.

   Read CKGR_MCFR until the MAINFRDY field is set, after which the user can read CKGR_MCFR.MAINF by performing an additional read. This provides the number of Main clock cycles that have been counted during a period of 16 MD_SLCK cycles.

   If MAINF = 0, switch MAINCK to the Main RC Oscillator by clearing CKGR_MOR.MOSCSEL. If MAINF ≠ 0, proceed to Step 6.
6. Set PLLA and Divider (if not required, proceed to Step 7.):
   All parameters needed to configure PLLA and the divider are located in CKGR_PLLAR.

   CKGR_PLLAR.DIVA is used to control the divider. This parameter can be programmed between 0 and 255. The divider output is divider input divided by DIVA. By default, DIVA is cleared, meaning that the divider and PLLA are turned off. The internal RC filter and the output current are located in PMC_PLL_CFG. They are configured depending on the required PLL frequency input and the frequency output to ensure PLL stability.

   CKGR_PLLAR.MULA is the PLLA multiplier factor. This parameter can be programmed between 0 and 56. If MULA is cleared, PLLA will be turned off, otherwise the PLLA output frequency is PLLA input frequency multiplied by (MULA + 1).

   CKGR_PLLAR.PLLACOUNT specifies the number of MD_SLCK cycles before PMC_SR.LOCKA is set after CKGR_PLLAR has been written.

   Once CKGR_PLLAR has been written, the user must wait for PMC_SR.LOCKA to be set. This can be done either by polling PMC_SR.LOCKA or by waiting for the interrupt line to be raised if the associated interrupt source (LOCKA) has been enabled in PMC_IER. All fields in CKGR_PLLAR can be programmed in a single write operation. If MULA or DIVA is modified, the LOCKA bit goes low to indicate that PLLA is not yet ready. When PLLA is locked, LOCKA is set again. The user must wait for the LOCKA bit to be set before using the PLLA output clock.
7. Select MCK and HCLK:
   MCK and HCLK are configurable via PMC_MCKR.

   CSS is used to select the clock source of MCK and HCLK. By default, the selected clock source is MAINCK.

   PRES is used to define the HCLK and MCK prescalers. The user can choose between 1, 2, 3, 4, 8, 16, 32, 64 division factors. Prescaler output is the selected clock source frequency divided by the PRES value.

   MDIV is used to define the MCK divider. It is possible to choose between different values (0 or 1). MCK output is the HCLK frequency divided by 1 or 2, depending on the value programmed in MDIV.

   By default, MDIV is cleared, which indicates that the HCLK is equal to MCK.

   Once the PMC_MCKR has been written, the user must wait for PMC_SR.MCKRDY to be set. This can be done either by polling PMC_SR.MCKRDY or by waiting for the interrupt line to be raised if the associated interrupt source (MCKRDY) has been enabled in PMC_IER. PMC_MCKR must not be programmed in a single write operation. The programming sequence for PMC_MCKR is as follows:

   If a new value for PMC_MCKR.CSS corresponds to any of the available PLL clocks:

   a. Program PMC_MCKR.PRES.

b. Wait for PMC_SR.MCKRDY to be set.

c. Program PMC_MCKR.MDIV.

d. Wait for PMC_SR.MCKRDY to be set.

e. Program PMC_MCKR.CSS.

f. Wait for PMC_SR.MCKRDY to be set.

If a new value for PMC_MCKR.CSS corresponds to MAINCK or MD_SLCK:

a. Program PMC_MCKR.CSS.

b. Wait for PMC_SR.MCKRDY to be set.

c. Program PMC_MCKR.PRES.

d. Wait for PMC_SR.MCKRDY to be set.

If CSS, MDIV or PRES are modified at any stage, the MCKRDY bit goes low to indicate that MCK and HCLK are not yet ready. The user must wait for MCKRDY bit to be set again before using MCK and HCLK.

Note: If PLLA clock was selected as MCK and the user decides to modify it by writing a new value into CKGR_PLLAR, the MCKRDY flag will go low while PLLA is unlocked. Once PLLA is locked again, LOCKA goes high and MCKRDY is set.
While PLLA is unlocked, MCK selection is automatically changed to MD_SLCK for PLLA. For further information, see Clock Switching Waveforms.
MCK is MAINCK divided by 2.

8. Select the Programmable clocks (PCKx):
PCKx are controlled via registers PMC_SCER, PMC_SCDR and PMC_SCSR.

   PCKx can be enabled and/or disabled via PMC_SCER and PMC_SCDR. Three PCKx can be used. PMC_SCSR indicates which PCKx is enabled. By default all PCKx are disabled.

   PMC_PCKx registers are used to configure PCKx.

   PMC_PCKx.CSS is used to select the PCKx divider source. Several clock options are available:

   – MAINCK
   – MD_SLCK
   – MCK
   – PLLACK
   – PLLBCK
   MD_SLCK is the default clock source.

   PMC_PCKx.PRES is used to control the PCKx prescaler. It is possible to choose between different values (1 to 256). PCKx output is prescaler input divided by PRES. By default, the PRES value is cleared which means that PCKx is equal to Slow clock.

   Once PMC_PCKx has been configured, the corresponding PCKx must be enabled and the user must wait for PMC_SR.PCKRDYx to be set. This can be done either by polling PMC_SR.PCKRDYx or by waiting for the interrupt line to be raised if the associated interrupt source (PCKRDYx) has been enabled in PMC_IER. All parameters in PMC_PCKx can be programmed in a single write operation.

   If the PMC_PCKx.CSS and PMC_PCKx.PRES parameters are to be modified, the corresponding PCKx must be disabled first. The parameters can then be modified. Once this has been done, the user must re-enable PCKx and wait for the PCKRDYx bit to be set.

9. Enable the peripheral clocks
Once all of the previous steps have been completed, the peripheral clocks can be enabled and/or disabled via register PMC_PCR.

26.14 Clock Switching Details

26.14.1 Master Clock Switching Timings

The tables below give the worst case timings required for MCK to switch from one selected clock to another one. This is in the event that the prescaler is deactivated. When the prescaler is activated, an additional time of 64 clock cycles of the newly selected clock has to be added.

Table 26-1. Clock Switching Timings (Worst Case)

From / To	MAINCK	MD_SLCK	PLL Clock
MAINCK	–	4 x MD_SLCK + 2.5 x MAINCK	3 x PLL Clock + 4 x MD_SLCK + 1 x MAINCK
MD_SLCK	0.5 x MAINCK + 4.5 x MD_SLCK	–	3 x PLL Clock + 5 x MD_SLCK
PLL Clock	0.5 x MAINCK + 4 x MD_SLCK + PLLCOUNT x MD_SLCK + 2.5 x PLL Clock	2.5 x PLL Clock + 5 x MD_SLCK + PLLCOUNT x MD_SLCK	See table below (Clock Switching Timings between Two PLLs (Worst Case)

Note:

1. PLL designates any available PLL of the Clock Generator.
2. PLLCOUNT designates either PLLACOUNT or PLLBCOUNT.

Table 26-2. Clock Switching Timings between Two PLLs (Worst Case)

From / To	PLLACK	PLLBCK
PLLACK	–	3 x PLLACK + 4 x MD_SLCK + 1.5 x PLLACK
PLLBCK	3 x PLLBCK + 4 x MD_SLCK + 1.5 x PLLBCK	–

26.14.2 Clock Switching Waveforms

In the following figures, Slow Clock designates MD_SLCK.

Figure 26-4. Switch Master Clock (MCK) from Slow Clock to PLLA Clock

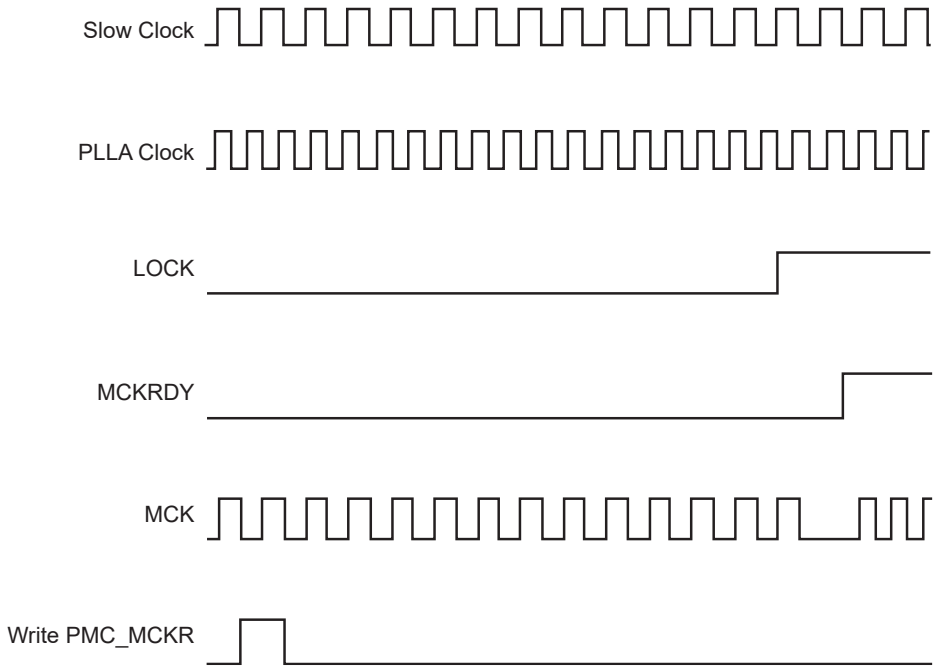


Figure 26-5. Switch Master Clock (MCK) from Main Clock (MAINCK) to Slow Clock

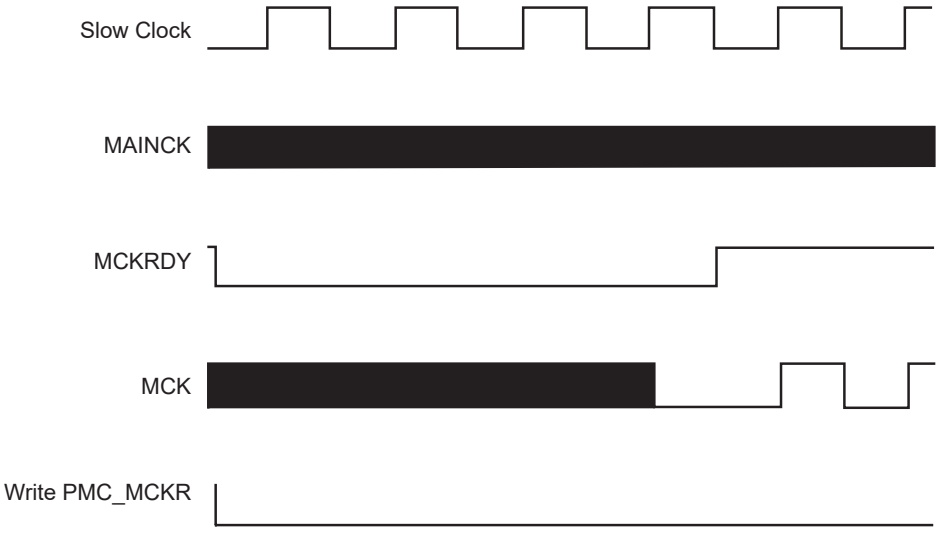


Figure 26-6. Change PLLA Programming

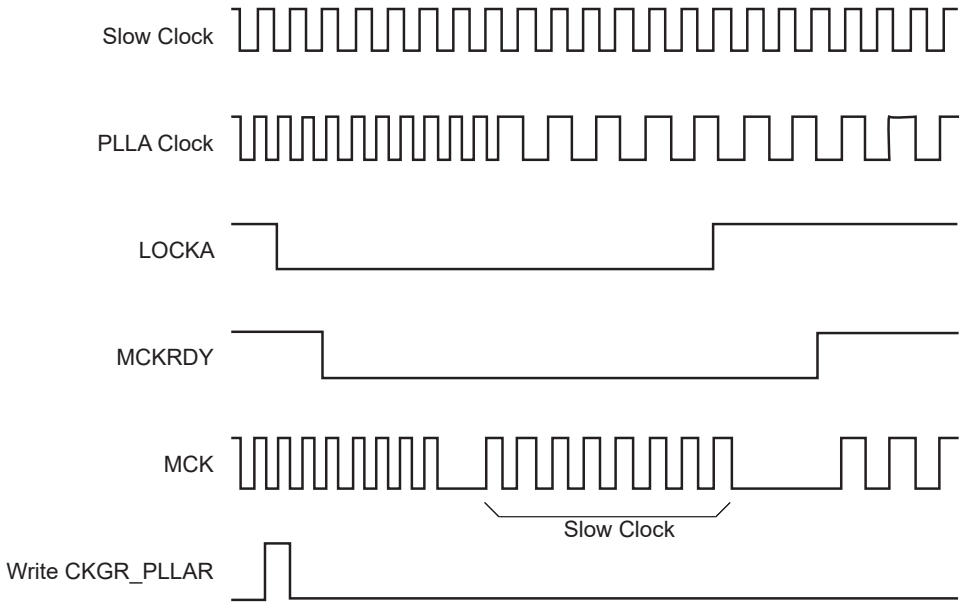
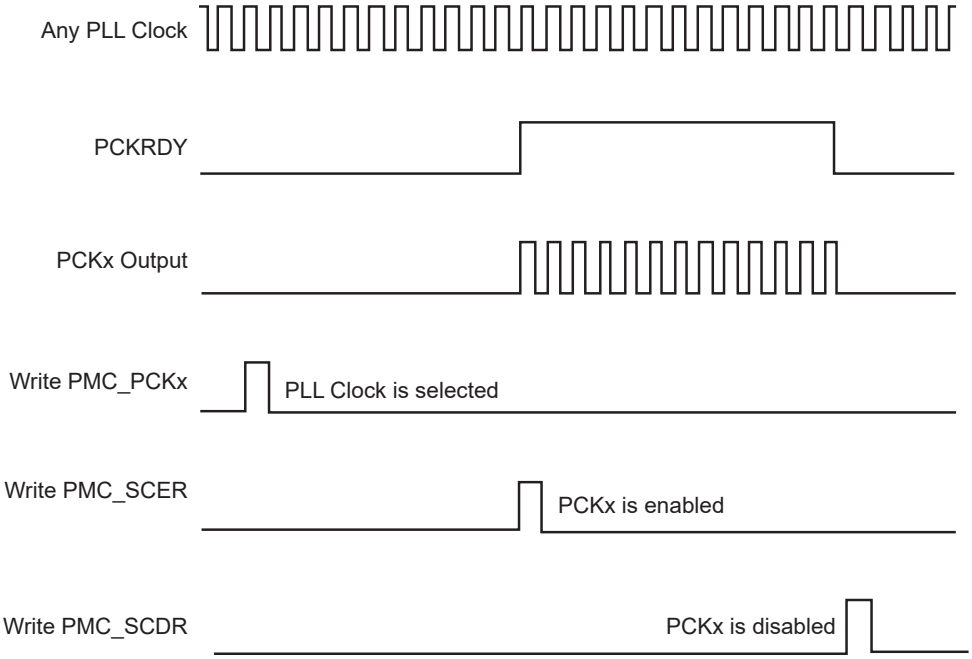


Figure 26-7. Programmable Clock Output Programming



26.15 Register Write Protection

To prevent any single software error from corrupting PMC behavior, certain registers in the address space can be write-protected by setting the WPEN bit or the WPITEN bit in the PMC Write Protection Mode Register (PMC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the PMC Write Protection Status Register (PMC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the PMC_WPSR.

The following registers are write-protected when the WPEN bit is set in PMC_WPMR:

- PMC System Clock Enable Register
- PMC System Clock Disable Register
- PMC Clock Generator Main Oscillator Register
- PMC Clock Generator Main Clock Frequency Register
- PMC Clock Generator PLLA Register
- PMC Clock Generator PLLB Register
- PMC Master Clock Register
- PMC Programmable Clock Register
- PMC PLL Configuration Register
- PMC Oscillator Calibration Register
- PLL Maximum Multiplier Value Register
- PMC Oscillator Calibration Register 2

The following interrupt registers are write-protected when the WPITEN bit is set in PMC_WPMR:

- PMC Interrupt Enable Register
- PMC Interrupt Disable Register

26.16 Register Summary

Offset	Name	Bit Pos.								
0x00	PMC_SCER	7:0								
		15:8					PCK3	PCK2	PCK1	PCK0
		23:16								
		31:24								
0x04	PMC_SCDR	7:0								
		15:8					PCK3	PCK2	PCK1	PCK0
		23:16								
		31:24								
0x08	PMC_SCSR	7:0								
		15:8					PCK3	PCK2	PCK1	PCK0
		23:16								
		31:24								
0x0C ... 0x1F	Reserved									
0x20	CKGR_MOR	7:0		MOSCRCF[2:0]			MOSCRCEN		MOSCXTBY	MOSCXTEN
		15:8	MOSCXTST[7:0]							
		23:16	KEY[7:0]							
		31:24				BCPUNMIC	BCPURST	XT32KFME	CFDEN	MOSCSEL
0x24	CKGR_MCFR	7:0	MAINF[7:0]							
		15:8	MAINF[15:8]							
		23:16				RCMEAS				MAINFRDY
		31:24								CCSS
0x28	CKGR_PLLAR	7:0	DIVA[7:0]							
		15:8	FREQ_VCO[1:0]		PLLACOUNT[5:0]					
		23:16	MULA[7:0]							
		31:24			ONE			MULA[10:8]		
0x2C	CKGR_PLLBR	7:0	DIVB[7:0]							
		15:8	FREQ_VCO[1:0]		PLLBCOUNT[5:0]					
		23:16	MULB[7:0]							
		31:24		SRCB[1:0]				MULB[10:8]		
0x30	PMC_MCKR	7:0		PRES[2:0]					CSS[1:0]	
		15:8			ZERO					MDIV
		23:16								
		31:24								
0x34 ... 0x3F	Reserved									
0x40	PMC_PCK0	7:0	PRES[3:0]					CSS[2:0]		
		15:8					PRES[7:4]			
		23:16								
		31:24								
0x44	PMC_PCK1	7:0	PRES[3:0]					CSS[2:0]		
		15:8					PRES[7:4]			
		23:16								
		31:24								
0x48	PMC_PCK2	7:0	PRES[3:0]					CSS[2:0]		
		15:8					PRES[7:4]			
		23:16								
		31:24								
0x4C	PMC_PCK3	7:0	PRES[3:0]					CSS[2:0]		
		15:8					PRES[7:4]			
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x50 ... 0x5F	Reserved									
0x60	PMC_IER	7:0					MCKRDY	LOCKB	LOCKA	MOSCXTS
		15:8		PCKRDY6	PCKRDY5	PCKRDY4	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
		23:16	CPUMON		XT32KERR			CFDEV	MOSCRCS	MOSCSELS
		31:24								
0x64	PMC_IDR	7:0					MCKRDY	LOCKB	LOCKA	MOSCXTS
		15:8		PCKRDY6	PCKRDY5	PCKRDY4	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
		23:16	CPUMON		XT32KERR			CFDEV	MOSCRCS	MOSCSELS
		31:24								
0x68	PMC_SR	7:0	OSCSELS				MCKRDY	LOCKB	LOCKA	MOSCXTS
		15:8		PCKRDY6	PCKRDY5	PCKRDY4	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
		23:16	CPUMON		XT32KERR	FOS	CFDS	CFDEV	MOSCRCS	MOSCSELS
		31:24								
0x6C	PMC_IMR	7:0					MCKRDY	LOCKB	LOCKA	MOSCXTS
		15:8		PCKRDY6	PCKRDY5	PCKRDY4	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
		23:16	CPUMON		XT32KERR			CFDEV	MOSCRCS	MOSCSELS
		31:24								
0x70 ... 0x77	Reserved									
0x78	PMC_FOCR	7:0								FOCLR
		15:8								
		23:16								
		31:24								
0x7C ... 0x7F	Reserved									
0x80	PMC_PLL_CFG	7:0					OUTCUR_PLLA[3:0]			
		15:8	SRA[1:0]		SCA[1:0]					
		23:16					OUTCUR_PLLB[3:0]			
		31:24	SRB[1:0]		SCB[1:0]					
0x84 ... 0xE3	Reserved									
0xE4	PMC_WPMR	7:0							WPITEN	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0xE8	PMC_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16	WPVSRC[15:8]							
		31:24								
0xEC ... 0x010B	Reserved									
0x010C	PMC_PCR	7:0		PID[6:0]						
		15:8				CMD	GCLKCSS[3:0]			
		23:16	GCLKDIV[3:0]							
		31:24			GCLKEN	EN	GCLKDIV[7:4]			
0x0110	PMC_OCR1	7:0	SEL4	CAL4[6:0]						
		15:8	SEL8	CAL8[6:0]						
		23:16	SEL10	CAL12[6:0]						
		31:24	SEL12	CAL10[6:0]						
0x0114 ... 0x012F	Reserved									

.....continued

Offset	Name	Bit Pos.								
0x0130	PMC_PMMR	7:0	PLLA_MMAX[7:0]							
		15:8						PLLA_MMAX[10:8]		
		23:16	PLLB_MMAX[7:0]							
		31:24						PLLB_MMAX[10:8]		
0x0134 ... 0x015F	Reserved									
0x0160	PMC_CPULIM	7:0	CPU_LOW_IT[7:0]							
		15:8	CPU_HIGH_IT[7:0]							
		23:16	CPU_LOW_RES[7:0]							
		31:24	CPU_HIGH_RES[7:0]							
0x0164 ... 0x016F	Reserved									
0x0170	PMC_CSR0	7:0	PID7	PID6	PID5	PID4	PID3	PID2	PID1	PID0
		15:8	PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
		23:16	PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
		31:24	PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
0x0174	PMC_CSR1	7:0	PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32
		15:8	PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
		23:16	PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
		31:24	PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
0x0178	PMC_CSR2	7:0	PID71	PID70	PID69	PID68	PID67	PID66	PID65	PID64
		15:8	PID79	PID78	PID77	PID76	PID75	PID74	PID73	PID72
		23:16	PID87	PID86	PID85	PID84	PID83	PID82	PID81	PID80
		31:24	PID95	PID94	PID93	PID92	PID91	PID90	PID89	PID88
0x017C	PMC_CSR3	7:0	PID103	PID102	PID101	PID100	PID99	PID98	PID97	PID96
		15:8	PID111	PID110	PID109	PID108	PID107	PID106	PID105	PID104
		23:16	PID119	PID118	PID117	PID116	PID115	PID114	PID113	PID112
		31:24	PID127	PID126	PID125	PID124	PID123	PID122	PID121	PID120
0x0180 ... 0x018F	Reserved									
0x0190	PMC_GCSR0	7:0	GPID7	GPID6	GPID5	GPID4	GPID3	GPID2	GPID1	GPID0
		15:8	GPID15	GPID14	GPID13	GPID12	GPID11	GPID10	GPID9	GPID8
		23:16	GPID23	GPID22	GPID21	GPID20	GPID19	GPID18	GPID17	GPID16
		31:24	GPID31	GPID30	GPID29	GPID28	GPID27	GPID26	GPID25	GPID24
0x0194	PMC_GCSR1	7:0	GPID39	GPID38	GPID37	GPID36	GPID35	GPID34	GPID33	GPID32
		15:8	GPID47	GPID46	GPID45	GPID44	GPID43	GPID42	GPID41	GPID40
		23:16	GPID55	GPID54	GPID53	GPID52	GPID51	GPID50	GPID49	GPID48
		31:24	GPID63	GPID62	GPID61	GPID60	GPID59	GPID58	GPID57	GPID56
0x0198	PMC_GCSR2	7:0	GPID71	GPID70	GPID69	GPID68	GPID67	GPID66	GPID65	GPID64
		15:8	GPID79	GPID78	GPID77	GPID76	GPID75	GPID74	GPID73	GPID72
		23:16	GPID87	GPID86	GPID85	GPID84	GPID83	GPID82	GPID81	GPID80
		31:24	GPID95	GPID94	GPID93	GPID92	GPID91	GPID90	GPID89	GPID88
0x019C	PMC_GCSR3	7:0	GPID103	GPID102	GPID101	GPID100	GPID99	GPID98	GPID97	GPID96
		15:8	GPID111	GPID110	GPID109	GPID108	GPID107	GPID106	GPID105	GPID104
		23:16	GPID119	GPID118	GPID117	GPID116	GPID115	GPID114	GPID113	GPID112
		31:24	GPID127	GPID126	GPID125	GPID124	GPID123	GPID122	GPID121	GPID120
0x01A0 ... 0x01AF	Reserved									
0x01B0	PMC_OSC2	7:0			OSCRCF[1:0]					EN
		15:8								EN_WR_CALIB
		23:16	KEY[7:0]							
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x01B4	PMC_OCR2	7:0	SEL4	CAL4[6:0]						
		15:8	SEL8	CAL8[6:0]						
		23:16	SEL10	CAL12[6:0]						
		31:24	SEL12	CAL10[6:0]						

26.16.1 PMC System Clock Enable Register

Name: PMC_SCER
Offset: 0x0000
Reset: –
Property: Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					PCK3	PCK2	PCK1	PCK0
Access					W	W	W	W
Reset					–	–	–	–

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bits 8, 9, 10, 11 – PCKx Programmable Clock x Output Enable

Value	Description
0	No effect.
1	Enables the corresponding Programmable Clock output.

26.16.2 PMC System Clock Disable Register

Name: PMC_SCDR
Offset: 0x0004
Reset: –
Property: Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					PCK3	PCK2	PCK1	PCK0
Access					W	W	W	W
Reset					–	–	–	–

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bits 8, 9, 10, 11 – PCKx Programmable Clock x Output Disable

Value	Description
0	No effect.
1	Disables the corresponding Programmable Clock output.

26.16.3 PMC System Clock Status Register

Name: PMC_SCSR
Offset: 0x0008
Reset: 0x00000001
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					PCK3	PCK2	PCK1	PCK0
Access					W	W	W	W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bits 8, 9, 10, 11 – PCKx Programmable Clock x Output Status

Value	Description
0	The corresponding Programmable Clock output is disabled.
1	The corresponding Programmable Clock output is enabled.

26.16.4 PMC Clock Generator Main Oscillator Register

Name: CKGR_MOR
Offset: 0x0020
Reset: 0x00000008
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
				BCPUNMIC	BCPURST	XT32KFME	CFDEN	MOSCSEL
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	KEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MOSCXTST[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		MOSCRCF[2:0]			MOSCRCEN		MOSCXTBY	MOSCXTEN
Access		R/W	R/W	R/W	R/W		R/W	R/W
Reset		0	0	0	1		0	0

Bit 28 – BCPUNMIC Bad CPU Clock Interrupt to NMIC Enable

Value	Description
0	A CPU clock failure has no effect on the Non Maskable Interrupt Controller (NMIC).
1	A CPU clock failure detection sends an interrupt signal to the NMIC.

Bit 27 – BCPURST Bad CPU Clock Reset Enable

Value	Description
0	A CPU clock failure detection cannot reset the system.
1	A CPU clock failure detection can reset the system if the reset controller is configured accordingly.

Bit 26 – XT32KFME 32.768 kHz Crystal Oscillator Frequency Monitoring Enable

Value	Description
0	The 32.768 kHz crystal oscillator frequency monitoring is disabled.
1	The 32.768 kHz crystal oscillator frequency monitoring is enabled.

Bit 25 – CFDEN Clock Failure Detector Enable

Value	Description
0	The clock failure detector is disabled.
1	The clock failure detector is enabled.

Bit 24 – MOSCSEL Main Clock Oscillator Selection

Value	Description
0	The Main RC oscillator is selected.
1	The Main crystal oscillator is selected.

Bits 23:16 – KEY[7:0] Write Access Password

Value	Name	Description
0x37	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

Bits 15:8 – MOSCXTST[7:0] Main Crystal Oscillator Start-up Time
Specifies the number of MD_SLCK cycles multiplied by 8 for the main crystal oscillator start-up time.

Bits 6:4 – MOSCRCF[2:0] Main RC Oscillator Frequency Selection
At start-up, the Main RC oscillator frequency is 4 MHz.
MOSCRCF must be changed only if MOSCRCS is set in the PMC_SR. Therefore MOSCRCF and MOSCRCEN cannot be changed at the same time.

Value	Name	Description
0	4_MHz	The Main RC oscillator frequency is at 4 MHz.
1	8_MHz	The Main RC oscillator frequency is at 8 MHz.
2	10_MHz	The Main RC oscillator frequency is at 10 MHz.
3	12_MHz	The Main RC oscillator frequency is at 12 MHz.

Bit 3 – MOSCRCEN Main RC Oscillator Enable
When MOSCRCEN is set, the MOSCRCS flag is set once the Main RC oscillator start-up time is achieved.

Value	Description
0	The Main RC oscillator is disabled.
1	The Main RC oscillator is enabled.

Bit 1 – MOSCXTBY Main Crystal Oscillator Bypass
When MOSCXTBY is set, the MOSCXTS flag in PMC_SR is automatically set.
Clearing MOSCXTEN and MOSCXTBY bits clears the MOSCXTS flag.
When the crystal oscillator bypass is disabled (MOSCXTBY = 0), the MOSCXTS flag must be read at ‘0’ in PMC_SR before enabling the crystal oscillator (MOSCXTEN = 1).

Value	Description
0	No effect.
1	The Main crystal oscillator is bypassed. MOSCXTEN must be cleared. An external clock must be connected on XIN.

Bit 0 – MOSCXTEN Main Crystal Oscillator Enable
A crystal must be connected between XIN and XOUT.
When MOSCXTEN is set, the MOSCXTS flag is set once the Main crystal oscillator start-up time is achieved.

Value	Description
0	The Main crystal oscillator is disabled.
1	The Main crystal oscillator is enabled. MOSCXTBY must be cleared.

26.16.5 PMC Clock Generator Main Clock Frequency Register

Name: CKGR_MCFR
Offset: 0x0024
Reset: 0x0001147F
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
								CCSS
Access								R/W
Reset								0

Bit	23	22	21	20	19	18	17	16
				RCMEAS				MAINFRDY
Access				R/W				R/W
Reset				0				1

Bit	15	14	13	12	11	10	9	8
	MAINF[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	1	0	0

Bit	7	6	5	4	3	2	1	0
	MAINF[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	1	1	1	1	1

Bit 24 – CCSS Counter Clock Source Selection

Value	Description
0	The measured clock of the MAINF counter is the Main RC oscillator.
1	The measured clock of the MAINF counter is the Main crystal oscillator.

Bit 20 – RCMEAS RC Oscillator Frequency Measure (write-only)
The measurement is performed on the main frequency (i.e., not limited to the Main RC oscillator only). If the source of MAINCK is the Main crystal oscillator, the restart of measurement may not be required because of the stability of crystal oscillators.

Value	Description
0	No effect.
1	Restarts measuring of the frequency of MAINCK. MAINF carries the new frequency as soon as a low-to-high transition occurs on the MAINFRDY flag.

Bit 16 – MAINFRDY Main Clock Frequency Measure Ready
To ensure that a correct value is read on the MAINF field, the MAINFRDY flag must be read at '1' then another read access must be performed on the register to get a stable value on the MAINF field.

Value	Description
0	MAINF value is not valid or the measured oscillator is disabled or a measure has just been started by means of RCMEAS.
1	The measured oscillator has been enabled previously and MAINF value is available.

Bits 15:0 – MAINF[15:0] Main Clock Frequency
Gives the number of cycles of the clock selected by the bit CCSS within 16 MD_SLCK periods. To calculate the frequency of the measured clock:
$f_{SELCLK} = (MAINF \times f_{MD\_SLCK})/16$
where frequency is in MHz.

26.16.6 PMC Clock Generator PLLA Register

Name: CKGR_PLLAR
Offset: 0x0028
Reset: 0x00003F00
Property: Read/Write

Possible limitations on PLLA input frequencies and multiplier factors should be checked before using the PMC.

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

WARNING Bit 29 must always be set to '1' when programming the CKGR_PLLAR.

Bit	31	30	29	28	27	26	25	24
			ONE			MULA[10:8]		
Access			R/W			R/W	R/W	R/W
Reset			0			0	0	0

Bit	23	22	21	20	19	18	17	16
	MULA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FREQ_VCO[1:0]		PLLACOUNT[5:0]					
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	DIVA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 29 – ONE Must Be Set to 1
Bit 29 must always be set to '1' when programming CKGR_PLLAR.

Bits 26:16 – MULA[10:0] PLLA Multiplier
Unlisted values are forbidden.

Value	Description
0	The PLLA is disabled (PLLA also disabled if DIVA = 0).
1 up to 25	PLLCK frequency is the PLLA input frequency multiplied by MULA + 1.
26 to 56	Rerserved.

Bits 15:14 – FREQ_VCO[1:0] VCO Frequency Configuratio

Value	Name	Description
0	VCO0	Frequency range: 40–80 MHz
1	VCO1	Frequency range: 70–150 MHz
2	VCO2	Frequency range: 125–275 MHz
3	VCO3	Frequency range: 250–450 MHz

Bits 13:8 – PLLACOUNT[5:0] PLLA Counter
Specifies the number of MD_SLCK cycles before the LOCKA bit is set in PMC_SR after CKGR_PLLAR is written.

Bits 7:0 – DIVA[7:0] PLLA Front End Divider

Value	Name	Description
0	0	PLLA is disabled.
1	BYPASS	Divider is bypassed (divide by 1) and PLLA is enabled.
2-64	DIV	Divider output is the selected clock divided by DIVA.
65-255	–	Reserved

26.16.7 PMC Clock Generator PLLB Register

Name: CKGR_PLLBR
Offset: 0x002C
Reset: 0x00003F00
Property: Read/Write

Possible limitations on PLLB input frequencies and multiplier factors should be checked before using the PMC.

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
		SRCB[1:0]				MULB[10:8]		
Access		R/W	R/W			R/W	R/W	R/W
Reset		0	0			0	0	0

Bit	23	22	21	20	19	18	17	16
	MULB[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FREQ_VCO[1:0]		PLLBCOUNT[5:0]					
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	DIVB[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 30:29 – SRCB[1:0] PLLB Source Clock Selection

Value	Name	Description
0	MAINCK	MAINCK is the source clock of PLLB.
1	RC2CK	RC2CK is the source clock of PLLB.
2	RC2CK	RC2CK is the source clock of PLLB.
3	RC2CK	RC2CK is the source clock of PLLB.

Bits 26:16 – MULB[10:0] PLLB Multiplier
Unlisted values are forbidden.

Value	Description
0	The PLLB is disabled (PLLB also disabled if DIVB = 0).
1 up to 25	PLLCK frequency is the PLLB input frequency multiplied by MULB + 1.
26 to 56	Rerserved.

Bits 15:14 – FREQ_VCO[1:0] VCO Frequency Configuration

Value	Name	Description
0	VCO0	Frequency range: 40–80 MHz
1	VCO1	Frequency range: 70–150 MHz
2	VCO2	Frequency range: 125–275 MHz
3	VCO3	Frequency range: 250–450 MHz

Bits 13:8 – PLLBCOUNT[5:0] PLLB Counter
Specifies the number of MD_SLCK cycles before the LOCKB bit is set in PMC_SR after CKGR_PLLBR is written.

Bits 7:0 – DIVB[7:0] PLLB Front End Divider

Value	Name	Description
0	0	PLLBis disabled.
1	BYPASS	Divider is bypassed (divide by 1) and PLLB is enabled.
2-64	DIV	Divider output is the selected clock divided by DIVB.
65-255	–	Reserved

26.16.8 PMC Master Clock Register

Name: PMC_MCKR
Offset: 0x0030
Reset: 0x00000001
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
			ZERO					MDIV
Access			R/W					R/W
Reset			0					0

Bit	7	6	5	4	3	2	1	0
		PRES[2:0]					CSS[1:0]	
Access		R/W	R/W	R/W			R/W	R/W
Reset		0	0	0			0	1

Bit 13 – ZERO Must be set to zero.
Bit 13 must always be set to '0' when programming PMC_MCKR.

Bit 8 – MDIV Master Clock Division

Value	Name	Description
0	EQ_PCK	MCK is FCLK divided by 1.
1	PCK_DIV2	MCK is FCLK divided by 2.

Bits 6:4 – PRES[2:0] Processor Clock Prescaler
Selected clock divided by PRES +1.

Value	Name	Description
0	CLK_1	Selected clock
1	CLK_2	Selected clock divided by 2
2	CLK_4	Selected clock divided by 4
3	CLK_8	Selected clock divided by 8
4	CLK_16	Selected clock divided by 16
5	CLK_32	Selected clock divided by 32
6	CLK_64	Selected clock divided by 64
7	–	Reserved

Bits 1:0 – CSS[1:0] Master Clock Source Selection

Value	Name	Description
0	SLOW_CLK	MD_SLCK is selected
1	MAIN_CLK	MAINCK is selected
2	PLLA_CLK	PLLACK is selected
3	–	Reserved

26.16.9 PMC Programmable Clock Register

Name: PMC_PCKx
Offset: 0x40 + x*0x04 [x=0..3]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					PRES[7:4]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PRES[3:0]					CSS[2:0]		
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset	0	0	0	0		0	0	0

Bits 11:4 – PRES[7:0] Programmable Clock Prescaler

Value	Description
0-255	Selected clock is divided by PRES+1.

Bits 2:0 – CSS[2:0] Programmable Clock Source Selection

Value	Name	Description
0	SLOW_CLK	MD_SLCK is selected
1	MAIN_CLK	MAINCK is selected
2	PLLA_CLK	PLLACK is selected
3	PLLB_CLK	PLLBCKDIV is selected
4	MCK	MCK is selected

26.16.10 PMC Interrupt Enable Register

Name: PMC_IER
Offset: 0x0060
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the PMC Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CPUMON		XT32KERR			CFDEV	MOSCRCS	MOSCSELS
Access	W		W			W	W	W
Reset	–		–			–	–	–

Bit	15	14	13	12	11	10	9	8
		PCKRDY6	PCKRDY5	PCKRDY4	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
Access		W	W	W	W	W	W	W
Reset		–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
					MCKRDY	LOCKB	LOCKA	MOSCXTS
Access					W	W	W	W
Reset					–	–	–	–

Bit 23 – CPUMON CPU Clock Monitor Interrupt Enable

Bit 21 – XT32KERR 32.768 kHz Crystal Oscillator Error Interrupt Enable

Bit 18 – CFDEV Clock Failure Detector Event Interrupt Enable

Bit 17 – MOSCRCS Main RC Oscillator Status Interrupt Enable

Bit 16 – MOSCSELS Main Clock Source Oscillator Selection Status Interrupt Enable

Bits 8, 9, 10, 11, 12, 13, 14 – PCKRDYx Programmable Clock Ready x Interrupt Enable

Bit 3 – MCKRDY Master Clock Ready Interrupt Enable

Bit 2 – LOCKB PLLB Lock Interrupt Enable

Bit 1 – LOCKA PLLA Lock Interrupt Enable

Bit 0 – MOSCXTS Main Crystal Oscillator Status Interrupt Enable

26.16.11 PMC Interrupt Disable Register

Name: PMC_IDR
Offset: 0x0064
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the PMC Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CPUMON		XT32KERR			CFDEV	MOSCRCS	MOSCSELS
Access	W		W			W	W	W
Reset	–		–			–	–	–

Bit	15	14	13	12	11	10	9	8
		PCKRDY6	PCKRDY5	PCKRDY4	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
Access		W	W	W	W	W	W	W
Reset		–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
					MCKRDY	LOCKB	LOCKA	MOSCXTS
Access					W	W	W	W
Reset					–	–	–	–

Bit 23 – CPUMON CPU Clock Monitor Interrupt Disable

Bit 21 – XT32KERR 32.768 kHz Crystal Oscillator Error Interrupt Disable

Bit 18 – CFDEV Clock Failure Detector Event Interrupt Disable

Bit 17 – MOSCRCS Main RC Status Interrupt Disable

Bit 16 – MOSCSELS Main Clock Source Oscillator Selection Status Interrupt Disable

Bits 8, 9, 10, 11, 12, 13, 14 – PCKRDYx Programmable Clock Ready x Interrupt Disable

Bit 3 – MCKRDY Master Clock Ready Interrupt Disable

Bit 2 – LOCKB PLLB Lock Interrupt Disable

Bit 1 – LOCKA PLLA Lock Interrupt Disable

Bit 0 – MOSCXTS Main Crystal Oscillator Status Interrupt Disable

26.16.12 PMC Status Register

Name: PMC_SR
Offset: 0x0068
Reset: 0x01030F08
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CPUMON		XT32KERR	FOS	CFDS	CFDEV	MOSCRCS	MOSCSELS
Access	R		R	R	R	R	R	R
Reset	0		0	0	0	0	1	1

Bit	15	14	13	12	11	10	9	8
		PCKRDY6	PCKRDY5	PCKRDY4	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
Access		W	W	W	W	W	W	W
Reset		0	0	0	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	OSCSELS				MCKRDY	LOCKB	LOCKA	MOSCXTS
Access	R				R	R	R	R
Reset	0				1	0	0	0

Bit 23 – CPUMON CPU Clock Monitor Error
This status is cleared on read.

Value	Description
0	The CPU clock is correct or the CPU clock monitor is disabled
1	The CPU clock is incorrect or has been incorrect for an elapsed period of time since the monitoring has been enabled.

Bit 21 – XT32KERR Slow Crystal Oscillator Error

Value	Description
0	The frequency of the 32.768 kHz crystal oscillator is correct (32.768 kHz±1%) or the monitoring is disabled.
1	The frequency of the 32.768 kHz crystal oscillator is incorrect or has been incorrect for an elapsed period of time since the monitoring has been enabled.

Bit 20 – FOS Clock Failure Detector Fault Output Status

Value	Description
0	The fault output of the clock failure detector is inactive.
1	The fault output of the clock failure detector is active. This status is cleared by writing a ‘1’ to FOCLR in PMC_FOCR.

Bit 19 – CFDS Clock Failure Detector Status

Value	Description
0	A clock failure of the Main crystal oscillator clock is not detected.
1	A clock failure of the Main crystal oscillator clock is detected.

Bit 18 – CFDEV Clock Failure Detector Event

Value	Description
0	No clock failure detection of the Main crystal oscillator clock has occurred since the last read of PMC_SR.

Value	Description
1	At least one clock failure detection of the Main crystal oscillator clock has occurred since the last read of PMC_SR.

Bit 17 – MOSCRCS Main RC Oscillator Status

Value	Description
0	Main RC oscillator is not stabilized.
1	Main RC oscillator is stabilized.

Bit 16 – MOSCSELS Main Clock Source Oscillator Selection Status

Value	Description
0	Selection is in progress.
1	Selection is done.

Bits 8, 9, 10, 11, 12, 13, 14 – PCKRDYx Programmable Clock Ready Status

Value	Description
0	Programmable Clock x is not ready.
1	Programmable Clock x is ready.

Bit 7 – OSCSELS Monitoring Domain Slow Clock Source Oscillator Selection

Value	Description
0	Slow RC oscillator is selected.
1	32.768 kHz crystal oscillator is selected.

Bit 3 – MCKRDY Master Clock Status

Value	Description
0	Master Clock is not ready.
1	Master Clock is ready.

Bit 2 – LOCKB PLLB Lock Status

Value	Description
0	PLLB is not locked
1	PLLB is locked.

Bit 1 – LOCKA PLLA Lock Status

Value	Description
0	PLLA is not locked
1	PLLA is locked.

Bit 0 – MOSCXTS Main Crystal Oscillator Status

Value	Description
0	Main crystal oscillator is not stabilized.
1	Main crystal oscillator is stabilized.

26.16.13 PMC Interrupt Mask Register

Name: PMC_IMR
Offset: 0x006C
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CPUMON		XT32KERR			CFDEV	MOSCRCS	MOSCSELS
Access	R		R			R	R	R
Reset	0		0			0	0	0

Bit	15	14	13	12	11	10	9	8
		PCKRDY6	PCKRDY5	PCKRDY4	PCKRDY3	PCKRDY2	PCKRDY1	PCKRDY0
Access		W	W	W	W	W	W	W
Reset		0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					MCKRDY	LOCKB	LOCKA	MOSCXTS
Access					R	R	R	R
Reset					0	0	0	0

Bit 23 – CPUMON CPU Clock Monitor Error Interrupt Mask

Bit 21 – XT32KERR 32.768 kHz Crystal Oscillator Error Interrupt Mask

Bit 18 – CFDEV Clock Failure Detector Event Interrupt Mask

Bit 17 – MOSCRCS Main RC Status Interrupt Mask

Bit 16 – MOSCSELS Main Clock Source Oscillator Selection Status Interrupt Mask

Bits 8, 9, 10, 11, 12, 13, 14 – PCKRDYx Programmable Clock Ready x Interrupt Mask

Bit 3 – MCKRDY Master Clock Ready Interrupt Mask

Bit 2 – LOCKB PLLB Lock Interrupt Mask

Bit 1 – LOCKA PLLA Lock Interrupt Mask

Bit 0 – MOSCXTS Main Crystal Oscillator Status Interrupt Mask

26.16.14 PMC Fault Output Clear Register

Name: PMC_FOCR
Offset: 0x0078
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								FOCLR
Access								W
Reset								–

Bit 0 – FOCLR Fault Output Clear
Clears the clock failure detector fault output.

26.16.15 PMC PLL Configuration Register

Name: PMC_PLL_CFG
Offset: 0x0080
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	SRB[1:0]		SCB[1:0]					
Access	R/W	R/W	R/W	R/W				
Reset	0	0	0	0				

Bit	23	22	21	20	19	18	17	16
					OUTCUR_PLLB[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SRA[1:0]		SCA[1:0]					
Access	R/W	R/W	R/W	R/W				
Reset	0	0	0	0				

Bit	7	6	5	4	3	2	1	0
					OUTCUR_PLLA[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 31:30 – SRB[1:0] Internal Filter PLL – Select Internal Resistor Value

Value	Name	Description
0	SR_VAL_24K	24 Ohms
1	SR_VAL_6K	6 Ohms
2	SR_VAL_3K	3 Ohms
3	SR_VAL_12K	12 Ohms

Bits 29:28 – SCB[1:0] Internal Filter PLL – Select Internal Capaticance Value

Value	Name	Description
0	SC_VAL_20p	20 pF
1	SC_VAL_40p	40 pF
2	SC_VAL_30p	30 pF
3	SC_VAL_60p	60 pF

Bits 19:16 – OUTCUR_PLLB[3:0] PLLB Output Current

Value	Name	Description
0	ICP0	0.5 mA
1	ICP1	0.75 mA
2	ICP2	1 mA
3	ICP3	1.25 mA

Bits 15:14 – SRA[1:0] Internal Filter PLL – Select Internal Resistor Value

Value	Name	Description
0	SR_VAL_24K	24 Ohms
1	SR_VAL_6K	6 Ohms
2	SR_VAL_3K	3 Ohms
3	SR_VAL_12K	12 Ohms

Bits 13:12 – SCA[1:0] Internal Filter PLL – Select Internal Capaticance Value

Value	Name	Description
0	SC_VAL_20p	20 pF
1	SC_VAL_40p	40 pF
2	SC_VAL_30p	30 pF
3	SC_VAL_60p	60 pF

Bits 3:0 – OUTCUR_PLLA[3:0] PLLA Output Current

Value	Name	Description
0	ICP0	0.5 mA
1	ICP1	0.75 mA
2	ICP2	1 mA
3	ICP3	1.25 mA

26.16.16 PMC Write Protection Mode Register

Name: PMC_WPMR
Offset: 0x00E4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
							WPITEN	WPEN
Access							R/W	R/W
Reset							0	0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x504D43	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

Bit 1 – WPITEN Write Protection Interrupt Enable
See Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection on interrupt registers if WPKEY corresponds to 0x504D43 (“PMC” in ASCII).
1	Enables the write protection on interrupt registers if WPKEY corresponds to 0x504D43 (“PMC” in ASCII).

Bit 0 – WPEN Write Protection Enable
See Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x504D43 (“PMC” in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x504D43 (“PMC” in ASCII).

26.16.17 PMC Write Protection Status Register

Name: PMC_WPSR
Offset: 0x00E8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	WPVSRC[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 23:8 – WPVSRC[15:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protection violation has occurred since the last read of the PMC_WPSR.
1	A write protection violation has occurred since the last read of the PMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

26.16.18 PMC Peripheral Control Register

Name: PMC_PCR
Offset: 0x010C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
			GCLKEN	EN	GCLKDIV[7:4]			
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	GCLKDIV[3:0]							
Access	R/W	R/W	R/W	R/W				
Reset	0	0	0	0				

Bit	15	14	13	12	11	10	9	8
				CMD	GCLKCSS[3:0]			
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		PID[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit 29 – GCLKEN Generic Clock Enable

Value	Description
0	The selected Generic clock is disabled.
1	The selected Generic clock is enabled.

Bit 28 – EN Enable

Value	Description
0	Selected Peripheral clock is disabled.
1	Selected Peripheral clock is enabled.

Bits 27:20 – GCLKDIV[7:0] Generic Clock Division Ratio
Generic clock is the selected clock period divided by GCLKDIV + 1.
GCLKDIV must not be changed while the peripheral selects GCLKx (e.g., bit rate, etc.).

Bit 12 – CMD Command

Value	Description
0	Read mode.
1	Write mode.

Bits 11:8 – GCLKCSS[3:0] Generic Clock Source Selection

Value	Name	Description
0	SLOW_CLK	MD_SLCK is selected
1	MAIN_CLK	MAINCK is selected
2	PLLA_CLK	PLLACK is selected
3	PLLB_CLK	PLLBCK is selected
4	MCK_CLK	MCK is selected
5	RC2CK	RC2CK is selected

Bits 6:0 – PID[6:0] Peripheral ID
Peripheral ID selection from PID2 to PID127.
“PID2 to PID127” refers to identifiers as defined in section “Peripheral Identifiers”.

26.16.19 PMC Oscillator Calibration Register

Name: PMC_OCR1
Offset: 0x0110
Reset: 0x12121212
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	SEL12	CAL10[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	1	0

Bit	23	22	21	20	19	18	17	16
	SEL10	CAL12[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	1	0

Bit	15	14	13	12	11	10	9	8
	SEL8	CAL8[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	1	0

Bit	7	6	5	4	3	2	1	0
	SEL4	CAL4[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	1	0

Bit 31 – SEL12 Selection of Main RC Oscillator Calibration Bits for 12 MHz

Value	Description
0	Factory-determined value stored in Flash memory.
1	Value written by user in CAL12 field of this register.

Bits 30:24 – CAL10[6:0] Main RC Oscillator Calibration Bits for 10 MHz
Calibration bits applied to the RC Oscillator when SEL10 is set.
The value read in this register depends on the value of SEL10:
If SEL10 is set to one, the calibration value stored in the Flash memory is read back
If SEL10 is set to zero, the last calibration value written in this field is read back

Bit 23 – SEL10 Selection of Main RC Oscillator Calibration Bits for 10 MHz

Value	Description
0	Factory-determined value stored in Flash memory.
1	Value written by user in CAL10 field of this register.

Bits 22:16 – CAL12[6:0] Main RC Oscillator Calibration Bits for 12 MHz
Calibration bits applied to the RC Oscillator when SEL12 is set.
The value read in this register depends on the value of SEL12:
If SEL12 is set to one, the calibration value stored in the Flash memory is read back
If SEL12 is set to zero, the last calibration value written in this field is read back

Bit 15 – SEL8 Selection of Main RC Oscillator Calibration Bits for 8 MHz

Value	Description
0	Factory-determined value stored in Flash memory.
1	Value written by user in CAL8 field of this register.

Bits 14:8 – CAL8[6:0] Main RC Oscillator Calibration Bits for 8 MHz
Calibration bits applied to the RC Oscillator when SEL8 is set to one.
The value read in this register depends on the value of SEL8:
If SEL8 is set to one, the calibration value stored in the Flash memory is read back
If SEL8 is set to zero, the last calibration value written in this field is read back

Bit 7 – SEL4 Selection of Main RC Oscillator Calibration Bits for 4 MHz

Value	Description
0	Default value stored in Flash memory.
1	Value written by user in CAL4 field of this register.

Bits 6:0 – CAL4[6:0] Main RC Oscillator Calibration Bits for 4 MHz
Calibration bits applied to the RC Oscillator when SEL4 is set to one.
The value read in this register depends on the value of SEL4:
If SEL4 is set to one, the calibration value stored in the Flash memory is read back
If SEL4 is set to zero, the last calibration value written in this field is read back

26.16.20 PLL Maximum Multiplier Value Register

Name: PMC_PMMR
Offset: 0x0130
Reset: 0x1FFF1FFF
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
						PLLB_MMAX[10:8]		
Access						R/W	R/W	R/W
Reset						1	1	1

Bit	23	22	21	20	19	18	17	16
	PLLB_MMAX[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
						PLLA_MMAX[10:8]		
Access						R/W	R/W	R/W
Reset						1	1	1

Bit	7	6	5	4	3	2	1	0
	PLLA_MMAX[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 26:16 – PLLB_MMAX[10:0] PLLB Maximum Allowed Multiplier Value
Defines the maximum value of multiplication factor that can be sent to PLLB. Any value of the MULB field (see PMC Clock Generator PLLB Register) above PLLB_MMAX is saturated to PLLB_MMAX. PLLB_MMAX write operation is cancelled in the following cases:

- The value of MULB is currently saturated by PLLB_MMAX.
- The user is trying to write a value of PLLB_MMAX that is smaller than the current value of MULB.

Bits 10:0 – PLLA_MMAX[10:0] PLLA Maximum Allowed Multiplier Value
Defines the maximum value of multiplication factor that can be sent to PLLA. Any value of the MULA field (see PMC Clock Generator PLLA Register) above PLLA_MMAX is saturated to PLLA_MMAX. PLLA_MMAX write operation is cancelled in the following cases:

- The value of MULA is currently saturated by PLLA_MMAX.
- The user is trying to write a value of PLLA_MMAX that is smaller than the current value of MULA.

26.16.21 PMC CPU Monitor Limits Register

Name: PMC_CPULIM
Offset: 0x0160
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	CPU_HIGH_RES[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CPU_LOW_RES[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CPU_HIGH_IT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CPU_LOW_IT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:24 – CPU_HIGH_RES[7:0] CPU Monitoring High Reset Limit
Beyond this limit, the CPU frequency monitor generates a reset.

Bits 23:16 – CPU_LOW_RES[7:0] CPU Monitoring Low RESET Limit
Below this limit, the CPU frequency monitor generates a reset.

Bits 15:8 – CPU_HIGH_IT[7:0] CPU Monitoring High IT Limit
Beyond this limit, the CPU frequency monitor generates an interrupt.

Bits 7:0 – CPU_LOW_IT[7:0] CPU Monitoring Low IT Limit
Below this limit, the CPU frequency monitor generates an interrupt.

26.16.22 PMC Peripheral Clock Status Register 0

Name: PMC_CSR0
Offset: 0x0170
Reset: 0x00000000
Property: Read-only

Depending on the product, not all PIDx may be implemented. For details, refer to the section "Peripheral Identifiers".

Bit	31	30	29	28	27	26	25	24
	PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PID7	PID6	PID5	PID4	PID3	PID2	PID1	PID0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – PIDx Peripheral Clock x Status

Value	Description
0	The corresponding peripheral clock is disabled.
1	The corresponding peripheral clock is enabled.

26.16.23 PMC Peripheral Clock Status Register 1

Name: PMC_CSR1
Offset: 0x0174
Reset: 0x10040000
Property: Read-only

Depending on the product, not all PIDx may be implemented. For details, refer to the section "Peripheral Identifiers".

Bit	31	30	29	28	27	26	25	24
	PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	1	0	0

Bit	15	14	13	12	11	10	9	8
	PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – PIDx Peripheral Clock x Status

Value	Description
0	The corresponding peripheral clock is disabled.
1	The corresponding peripheral clock is enabled.

26.16.24 PMC Peripheral Clock Status Register 2

Name: PMC_CSR2
Offset: 0x0178
Reset: 0x00000002
Property: Read-only

Depending on the product, not all PIDx may be implemented. For details, refer to the section "Peripheral Identifiers".

Bit	31	30	29	28	27	26	25	24
	PID95	PID94	PID93	PID92	PID91	PID90	PID89	PID88
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PID87	PID86	PID85	PID84	PID83	PID82	PID81	PID80
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PID79	PID78	PID77	PID76	PID75	PID74	PID73	PID72
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PID71	PID70	PID69	PID68	PID67	PID66	PID65	PID64
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	1	0

Bits 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 – PIDx Peripheral Clock x Status

Value	Description
0	The corresponding peripheral clock is disabled.
1	The corresponding peripheral clock is enabled.

26.16.25 PMC Peripheral Clock Status Register 3

Name: PMC_CSR3
Offset: 0x017C
Reset: 0x00000000
Property: Read-only

Depending on the product, not all PIDx may be implemented. For details, refer to the section "Peripheral Identifiers".

Bit	31	30	29	28	27	26	25	24
	PID127	PID126	PID125	PID124	PID123	PID122	PID121	PID120
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PID119	PID118	PID117	PID116	PID115	PID114	PID113	PID112
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PID111	PID110	PID109	PID108	PID107	PID106	PID105	PID104
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PID103	PID102	PID101	PID100	PID99	PID98	PID97	PID96
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – PIDx Peripheral Clock x Status

Value	Description
0	The corresponding peripheral clock is disabled.
1	The corresponding peripheral clock is enabled.

26.16.26 PMC Generic Clock Status Register 0

Name: PMC_GCSR0
Offset: 0x0190
Reset: 0x13C0E380
Property: Read-only

Depending on the product, not all GPIDx may be implemented. For details, refer to the section "Peripheral Identifiers".

Bit	31	30	29	28	27	26	25	24
	GPID31	GPID30	GPID29	GPID28	GPID27	GPID26	GPID25	GPID24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	0	0	1	1

Bit	23	22	21	20	19	18	17	16
	GPID23	GPID22	GPID21	GPID20	GPID19	GPID18	GPID17	GPID16
Access	R	R	R	R	R	R	R	R
Reset	1	1	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	GPID15	GPID14	GPID13	GPID12	GPID11	GPID10	GPID9	GPID8
Access	R	R	R	R	R	R	R	R
Reset	1	1	1	0	0	0	1	1

Bit	7	6	5	4	3	2	1	0
	GPID7	GPID6	GPID5	GPID4	GPID3	GPID2	GPID1	GPID0
Access	R	R	R	R	R	R	R	R
Reset	1	0	0	0	0	0	0	0

Bits 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 – GPIDx Generic Clock x Status

Value	Description
0	The corresponding Generic clock is disabled.
1	The corresponding Generic clock is enabled.

26.16.27 PMC Generic Clock Status Register 1

Name: PMC_GCSR1
Offset: 0x0194
Reset: 0x09206050
Property: Read-only

Depending on the product, not all GPIDx may be implemented. For details, refer to the section "Peripheral Identifiers".

Bit	31	30	29	28	27	26	25	24
	GPID63	GPID62	GPID61	GPID60	GPID59	GPID58	GPID57	GPID56
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	0	0	1

Bit	23	22	21	20	19	18	17	16
	GPID55	GPID54	GPID53	GPID52	GPID51	GPID50	GPID49	GPID48
Access	R	R	R	R	R	R	R	R
Reset	0	0	1	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	GPID47	GPID46	GPID45	GPID44	GPID43	GPID42	GPID41	GPID40
Access	R	R	R	R	R	R	R	R
Reset	0	1	1	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	GPID39	GPID38	GPID37	GPID36	GPID35	GPID34	GPID33	GPID32
Access	R	R	R	R	R	R	R	R
Reset	0	1	0	1	0	0	0	0

Bits 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 – GPIDx Generic Clock x Status

Value	Description
0	The corresponding Generic clock is disabled.
1	The corresponding Generic clock is enabled.

26.16.28 PMC Generic Clock Status Register 2

Name: PMC_GCSR2
Offset: 0x0198
Reset: 0x00000016
Property: Read-only

Depending on the product, not all GPIDx may be implemented. For details, refer to the section "Peripheral Identifiers".

Bit	31	30	29	28	27	26	25	24
	GPID95	GPID94	GPID93	GPID92	GPID91	GPID90	GPID89	GPID88
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	GPID87	GPID86	GPID85	GPID84	GPID83	GPID82	GPID81	GPID80
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	GPID79	GPID78	GPID77	GPID76	GPID75	GPID74	GPID73	GPID72
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	GPID71	GPID70	GPID69	GPID68	GPID67	GPID66	GPID65	GPID64
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	1	0	1	1	0

Bits 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 – GPIDx Generic Clock x Status

Value	Description
0	The corresponding Generic clock is disabled.
1	The corresponding Generic clock is enabled.

26.16.29 PMC Generic Clock Status Register 3

Name: PMC_GCSR3
Offset: 0x019C
Reset: 0x00000000
Property: Read-only

Depending on the product, not all GPIDx may be implemented. For details, refer to the section "Peripheral Identifiers".

Bit	31	30	29	28	27	26	25	24
	GPID127	GPID126	GPID125	GPID124	GPID123	GPID122	GPID121	GPID120
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	GPID119	GPID118	GPID117	GPID116	GPID115	GPID114	GPID113	GPID112
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	GPID111	GPID110	GPID109	GPID108	GPID107	GPID106	GPID105	GPID104
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	GPID103	GPID102	GPID101	GPID100	GPID99	GPID98	GPID97	GPID96
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – GPIDx Generic Clock x Status

Value	Description
0	The corresponding Generic clock is disabled.
1	The corresponding Generic clock is enabled.

26.16.30 PMC Oscillator Control Register 2

Name: PMC_OSC2
Offset: 0x01B0
Reset: 0x00000021
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	KEY[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
								EN_WR_CALIB
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
			OSCRCF[1:0]					EN
Access			R/W	R/W				R/W
Reset			1	0				1

Bits 23:16 – KEY[7:0] Calibration Register Write Key
Always reads as '0'.

Value	Name	Description
0x37	PASSWD	Unlock key to enable writes to calibration registers. Writing any other value to KEY will abort the Write operation.

Bit 8 – EN_WR_CALIB Enable Calibration Register Write

Value	Description
0	Disable write of the calibration register for the active frequency.
1	Enable write of the calibration register for the active frequency.

Bits 5:4 – OSCRCF[1:0] Second Oscillator Frequency Selection
At start-up, the second RC oscillator frequency is 10 MHz.

Value	Name	Description
0	4_MHz	The second RC oscillator frequency is at 4 MHz.
1	8_MHz	The second RC oscillator frequency is at 8 MHz.
2	10_MHz	The second RC oscillator frequency is at 10 MHz.
3	12_MHz	The second RC oscillator frequency is at 12 MHz.

Bit 0 – EN Enable

Value	Description
0	The 2nd fast oscillator is disabled.
1	The 2nd fast oscillator is enabled.

26.16.31 PMC Oscillator Calibration Register 2

Name: PMC_OCR2
Offset: 0x01B4
Reset: 0x12121212
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	SEL12	CAL10[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	1	0

Bit	23	22	21	20	19	18	17	16
	SEL10	CAL12[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	1	0

Bit	15	14	13	12	11	10	9	8
	SEL8	CAL8[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	1	0

Bit	7	6	5	4	3	2	1	0
	SEL4	CAL4[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	1	0

Bit 31 – SEL12 Selection of Second RC Oscillator Calibration Bits for 12 MHz

Value	Description
0	Factory-determined value stored in Flash memory.
1	Value written by user in CAL12 field of this register.

Bits 30:24 – CAL10[6:0] Second RC Oscillator Calibration Bits for 10 MHz
Calibration bits applied to the RC Oscillator when SEL10 is set.
The value read in this register depends on the value of SEL10:
If SEL10 is set to one, the calibration value stored in the Flash memory is read back
If SEL10 is set to zero, the last calibration value written in this field is read back

Bit 23 – SEL10 Selection of Second RC Oscillator Calibration Bits for 10 MHz

Value	Description
0	Factory-determined value stored in Flash memory.
1	Value written by user in CAL10 field of this register.

Bits 22:16 – CAL12[6:0] Second RC Oscillator Calibration Bits for 12 MHz
Calibration bits applied to the RC Oscillator when SEL12 is set.
The value read in this register depends on the value of SEL12:
If SEL12 is set to one, the calibration value stored in the Flash memory is read back
If SEL12 is set to zero, the last calibration value written in this field is read back

Bit 15 – SEL8 Selection of Second RC Oscillator Calibration Bits for 8 MHz

Value	Description
0	Factory-determined value stored in Flash memory.
1	Value written by user in CAL8 field of this register.

Bits 14:8 – CAL8[6:0] Second RC Oscillator Calibration Bits for 8 MHz
Calibration bits applied to the RC Oscillator when SEL8 is set to one.
The value read in this register depends on the value of SEL8:
If SEL8 is set to one, the calibration value stored in the Flash memory is read back
If SEL8 is set to zero, the last calibration value written in this field is read back

Bit 7 – SEL4 Selection of Second RC Oscillator Calibration Bits for 4 MHz

Value	Description
0	Default value stored in Flash memory.
1	Value written by user in CAL4 field of this register.

Bits 6:0 – CAL4[6:0] Second RC Oscillator Calibration Bits for 4 MHz
Calibration bits applied to the RC Oscillator when SEL4 is set to one.
The value read in this register depends on the value of SEL4:
If SEL4 is set to one, the calibration value stored in the Flash memory is read back
If SEL4 is set to zero, the last calibration value written in this field is read back

27. Parallel Input/Ouput Controller (PIO)

27.1 Description

The Parallel Input/Output Controller (PIO) manages up to 194 fully programmable input/output lines. Each I/O line may be dedicated as a general-purpose I/O or be assigned to a function of an embedded peripheral. This ensures effective optimization of the pins of the product.

Each I/O line of the PIO Controller features are as follows:

- An input change interrupt enabling level change detection on any I/O line
- Rising edge, falling edge, both edge, low-level or high-level detection on any I/O line
- Multi-Output Drive Capability
- Control of the pull-up and pull-down of the I/O line
- Input visibility and output control

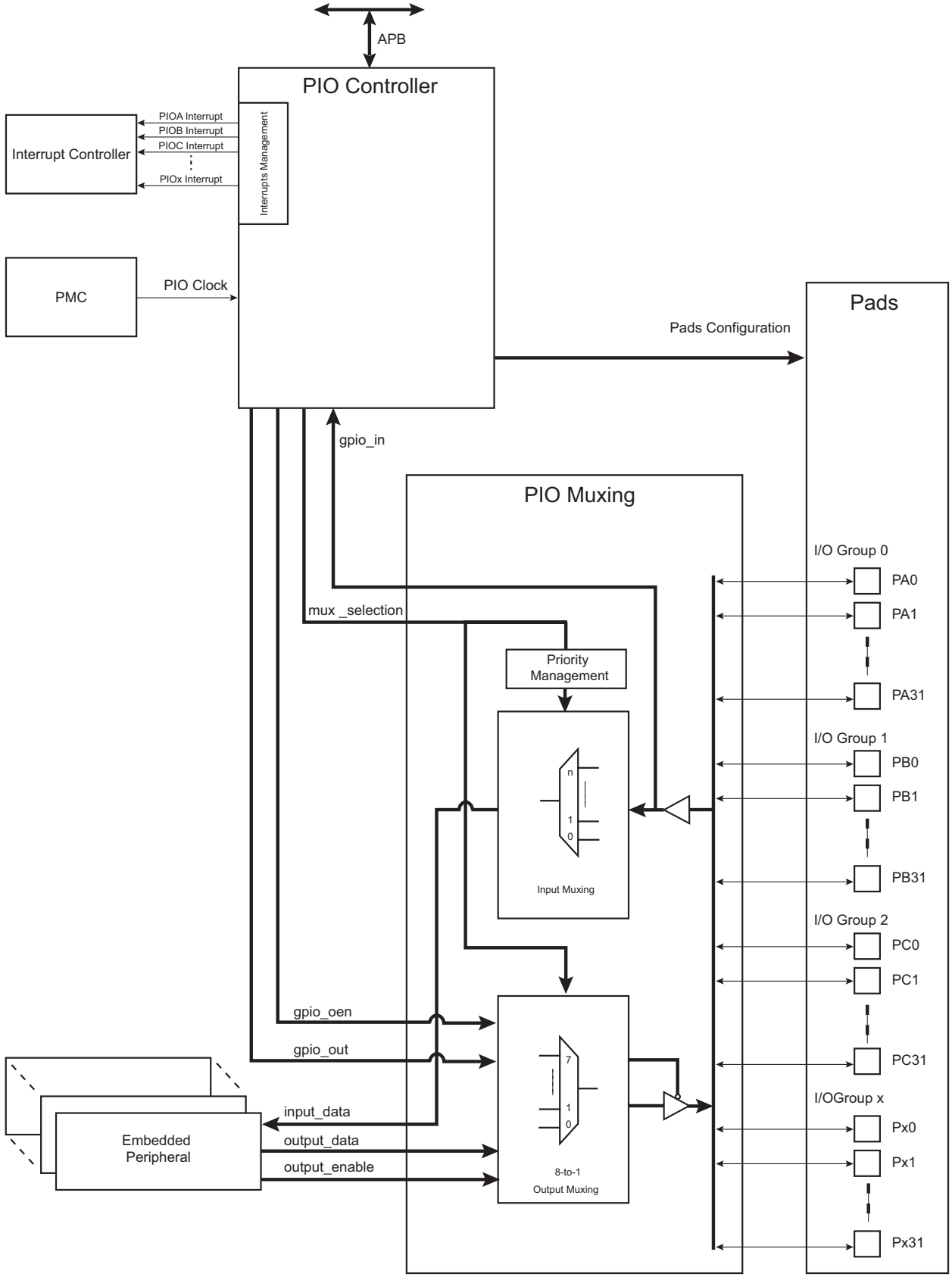
The PIO Controller also features a synchronous output providing up to 32 bits of data output in a single write operation.

27.2 Embedded Characteristics

- Up to 194 Programmable I/O Lines
- Multiplexing of up to Seven Peripheral Functions per I/O Line
- For Each I/O Line, Whether Assigned to a Peripheral or Used as General-purpose I/O
  - Input change interrupt
  - Multi-drive option
  - Programmable pull-up/pull-down on each I/O line
  - Pin data status register, supplies visibility of the level on the pin at any time
  - Programmable event: rising edge, falling edge, both edge, low-level or high-level
  - Configuration lock by the connected peripheral
  - Programmable configuration lock (active until next v_{DDCORE} reset) to protect against further software modifications (intentional or unintentional)
- Register Write Protection Against Unintentional Software Modifications:
  - One configuration bit to enable or disable protection of I/O line settings
  - One configuration bit to enable or disable protection of interrupt settings
- Synchronous Output, Possibility to Set or Clear Simultaneously up to 32 I/O Lines in a Single Write
- Programmable Schmitt Trigger Inputs
- Programmable I/O Drive

27.3 Block Diagram

Figure 27-1. Block Diagram



Note:

1. x = 6 (the number of I/O group is 7).
2. n depends on the number of I/O lines affected to the peripheral input.

27.4 Product Dependencies

27.4.1 Pin Multiplexing

Each pin is configurable, depending on the product, as either a general-purpose I/O line only, or as an I/O line multiplexed with up to 4 peripheral I/Os. As the multiplexing is hardware defined and thus product-dependent, the hardware designer and programmer must carefully determine the configuration of the PIO Controllers required by their application. When an I/O line is general-purpose only, i.e., not multiplexed with any peripheral I/O, programming of the PIO Controller regarding the assignment to a peripheral has no effect and only the PIO Controller can control how the pin is driven by the product.

27.4.2 Power Management

The Power Management Controller (PMC) controls the PIO Controller clock in order to save power. Writing any of the registers of the user interface does not require the PIO Controller clock to be enabled. This means that the configuration of the I/O lines does not require the PIO Controller clock to be enabled.

However, when the clock is disabled, not all of the features of the PIO Controller are available, including glitch filtering. Note that the input change interrupt, the interrupt modes on a programmable event and the read of the pin level require the clock to be validated.

After a hardware reset, the PIO clock is disabled by default.

The user must configure the PMC before any access to the input line information.

27.4.3 Interrupt Generation

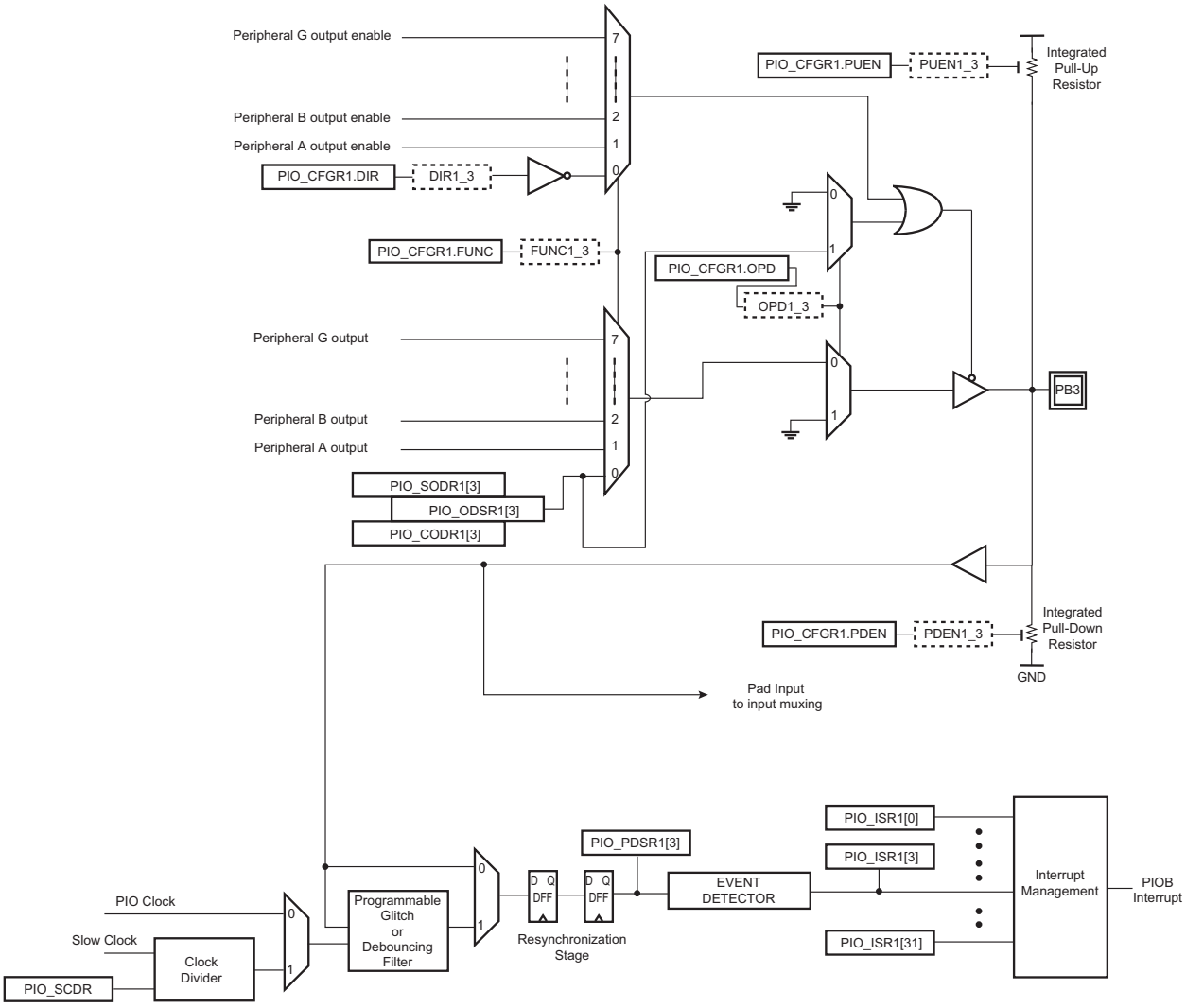
For interrupt handling, the PIO Controllers are considered as user peripherals. This means that the PIO Controller interrupt lines are connected among the interrupt sources. The PIO Controller supply one interrupt signal per I/O group. Refer to the PIO Controller peripheral identifier in the product description to identify the interrupt sources dedicated to the PIO Controller. Using the PIO Controller requires the Interrupt Controller to be programmed first.

The PIO Controller interrupt can be generated only if the PIO Controller clock is enabled.

27.5 Functional Description

The PIO Controller features up to 512 fully-programmable I/O lines. Most of the control logic associated to each I/O is represented in the figure below, where the I/O line 3 of the PIOB (PB3) is described as an example. In this description each signal shown represents one of up to 512 possible indexes.

Figure 27-2. I/O Line Control Logic



27.5.1 I/O Line Configuration Method

The user interface of the PIO Controller provides several sets of registers. Each set of registers interfaces with one I/O group.

Table 27-1. I/O Group List

I/O Group Number	PIO
0	PIOA
1	PIOB
...	...
6	PIOG

27.5.1.1 Programming I/O Line Configuration

The user must first define which I/O line in the group will be targeted by writing a 1 to the corresponding bit in the Mask register (PIO_MSKRx). Several I/O lines in an I/O group can be configured at the same time by setting the corresponding bits in PIO_MSKRx. Then, write the Configuration register (PIO_CFGRx) to apply the configuration to the I/O line(s) defined in PIO_MSKRx.

For more details concerning the I/O line configuration using PIO_MSKRx and PIO_CFGRx, see I/O Lines Programming Example.

27.5.1.2 Reading I/O Line Configuration

Reading configuration requires the user to first define which I/O line in the group x will be targeted by writing a 1 to the corresponding bit in PIO_MSKRx. The value of the targeted I/O line is read in PIO_CFGRx.

If several bits are set in PIO_MSKRx, then the read configuration in PIO_CFGRx is the configuration of the I/O line with the lowest index.

27.5.2 Pull-up and Pull-down Resistor Control

Each I/O line is designed with an embedded pull-up resistor and an embedded pull-down resistor.

The pull-up resistor on the I/O line(s) defined in PIO_MSKRx can be enabled by setting the PUEN bit in PIO_CFGRx. Clearing the PUEN bit in PIO_CFGRx disables the pull-up resistor of I/O lines defined in PIO_MSKRx.

The pull-down resistor on the I/O line(s) defined in PIO_MSKRx can be enabled by setting the PDEN bit in PIO_CFGRx. Clearing the PDEN bit in PIO_CFGRx disables the pull-down resistor of I/O lines defined in PIO_MSKRx.

If both PUEN and PDEN bit are set in PIO_CFGRx, only the pull-up resistor is enabled for I/O line(s) defined in PIO_MSKRx and the PDEN bit is discarded.

Control of the pull-up resistor is possible regardless of the configuration of the I/O line (Input, Output, Open-drain).

For more details concerning pull-up and pull-down configuration, see PIO_CFGR.

The reset value of PUEN and PDEN bits of each I/O line is defined at the product level and depends on the multiplexing of the device.

27.5.3 General-Purpose or Peripheral Function Selection

The PIO Controller provides multiplexing of up to 4 peripheral functions on a single pin. The selection is performed by writing the FUNC field in PIO_CFGRx. The selected function is applied to the I/O line(s) defined in PIO_MSKRx.

When FUNC is 0, no peripheral is selected and the General-Purpose PIO (GPIO) mode is selected (in this mode, the I/O line is controlled by the PIO Controller).

When FUNC is not 0, the peripheral selected to control the I/O line depends on the FUNC value.

For more details, see the PIO_CFGR.

Note that multiplexing of peripheral lines affects both input and output peripheral lines. When a peripheral is not selected on any I/O line, its inputs are assigned with constant default values defined at the product level. The user must ensure that only one I/O line is affected to a peripheral input at a time.

The reset value of the FUNC field of each I/O line is defined at the product level and depends on the multiplexing of the device.

27.5.4 Output Control

When the I/O line is assigned to a peripheral function, i.e., the corresponding FUNC field of the line configuration is not 0, the drive of the I/O line is controlled by the peripheral. According to the FUNC value, the selected peripheral determines whether the pin is driven or not.

When the FUNC field of a I/O line is 0, then the I/O line is set in General-Purpose mode and the I/O line can be configured to be driven by the PIO Controller instead of the peripheral.

If the DIR bit of the I/O line configuration (PIO_CFGRx) is set (OUTPUT) then the I/O line can be driven by the PIO Controller. The level driven on an I/O line can be determined by writing in the Set Output Data register (PIO_SODRx) and the Clear Output Data register (PIO_CODRx). These write operations, respectively, set and clear the Output Data Status register (PIO_ODSRx), which represents the data driven on the I/O lines. Writing PIO_ODSRx directly is possible and only affects the I/O line set to 1 in PIO_MSKRx (see Synchronous Data Output).

When the DIR bit of the I/O line configuration is at zero, the corresponding I/O line is used as an input only.

The DIR bit has no effect if the corresponding line is assigned to a peripheral function, but writing the DIR bit is managed whether the pin is configured to be controlled by the PIO Controller or assigned to a peripheral function. This enables configuration of the I/O line prior to setting it to be managed by the PIO Controller.

Similarly, writing in PIO_SODRx and PIO_CODRx affects PIO_ODSRx. This is important as it defines the first level driven on the I/O line.

27.5.5 Synchronous Data Output

Clearing one or more PIO line(s) and setting another one or more PIO line(s) synchronously cannot be done by using PIO_SODRx and PIO_CODRx. It requires two successive write operations into two different registers. To overcome this, the PIO Controller offers a direct control of PIO outputs by single write access to PIO_ODSRx. Only I/O lines set to 1 in PIO_MSKRx are written.

27.5.6 Open-Drain Mode

Each I/O can be independently programmed in Open-Drain mode. This feature permits several drivers to be connected on the I/O line which is driven low only by each device. An external pull-up resistor (or enabling of the internal one) is generally required to guarantee a high level on the line.

The Open-Drain mode is controlled by the OPD bit in the I/O line configuration (PIO_CFGRx). An I/O line is switched in Open-Drain mode by setting PIO_CFGRx.OPD. The Open-Drain mode can be selected if the I/O line is not controlled by a peripheral (the FUNC field must be cleared in PIO_CFGRx).

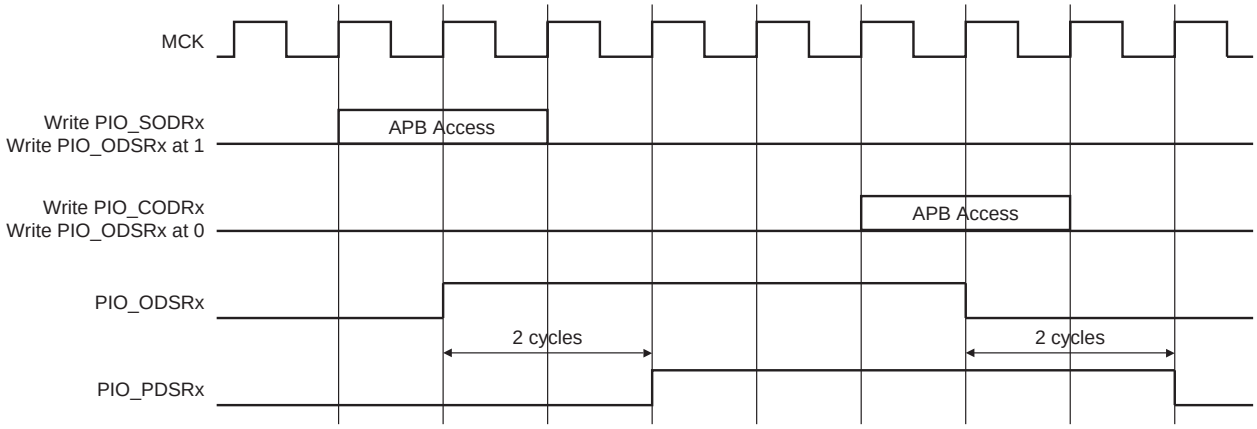
For more details on the Open-Drain mode, see the PIO_CFGR.

After reset, the OPD bit of each I/O line is defined at the product level and depends on the multiplexing of the device.

27.5.7 Output Line Timings

The figure below shows how the outputs are driven either by writing PIO_SODRx or PIO_CODRx, or by directly writing PIO_ODSRx. This last case is valid only if the corresponding bit in PIO_MSKRx is set. The figure below also shows when the feedback in the Pin Data Status Register (PIO_PDSRx) is available.

Figure 27-3. Output Line Timings



27.5.8 Inputs

The level on each I/O line of the I/O group x can be read through PIO_PDSRx. This register indicates the level of the I/O lines regardless of their configuration, whether uniquely as an input, or driven by the PIO Controller, or driven by a peripheral.

Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSRx reads the levels present on the I/O line at the time the clock was disabled.

27.5.9 Input Edge/Level Interrupt

Each I/O group can be programmed to generate an interrupt when it detects an edge or a level on an I/O line. The Input Edge/Level interrupts are controlled by writing the Interrupt Enable register (PIO_IERx) and the Interrupt Disable register (PIO_IDRx), which enable and disable the input change interrupt respectively by setting and clearing

the corresponding bit in the Interrupt Mask register (PIO_IMRx). As input change detection is possible only by comparing two successive samplings of the input of the I/O line, the PIO Controller clock must be enabled. The Input Change interrupt is available regardless of the configuration of the I/O line, i.e., configured as an input only, controlled by the PIO Controller or assigned to a peripheral function.

Each I/O group can generate an interrupt.

According to the EVTSEL field value in PIO_CFGRx, the interrupt signal of the I/O group x can be generated on the following occurrence:

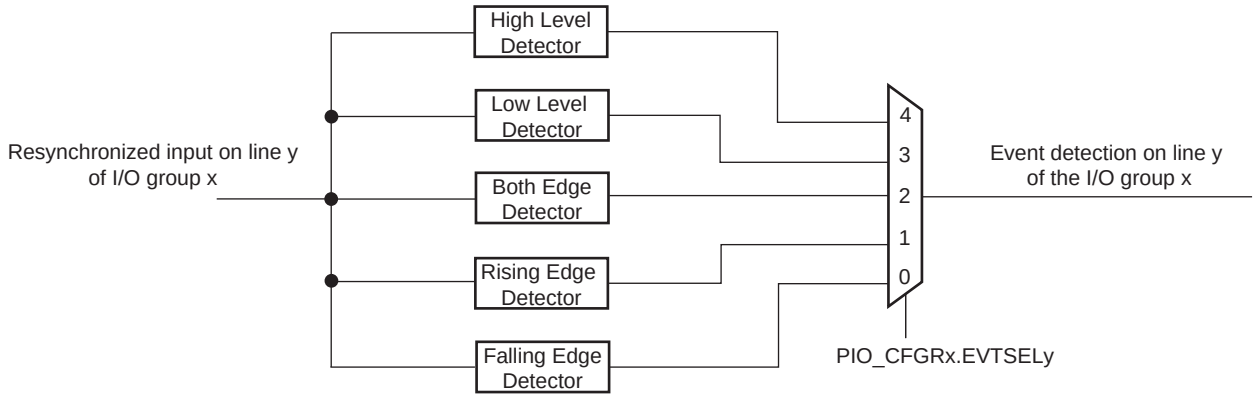
- PIO_CFGRx.EVTSELy = 0: The interrupt signal of the I/O group x is generated on the I/O line y falling edge detection (assuming that PIO_IMRx[y] = 1).
- PIO_CFGRx.EVTSELy = 1: The interrupt signal of the I/O group x is generated on the I/O line y rising edge detection (assuming that PIO_IMRx[y] = 1).
- PIO_CFGRx.EVTSELy = 2: The interrupt signal of the I/O group x is generated on the I/O line y both rising and falling edge detection (assuming that PIO_IMRx[y] = 1).
- PIO_CFGRx.EVTSELy = 3: The interrupt signal of the I/O group x is generated on the I/O line y low level detection (assuming that PIO_IMRx[y] = 1).
- PIO_CFGRx.EVTSELy = 4: The interrupt signal of the I/O group x is generated on the I/O line y high level detection (assuming that PIO_IMRx[y] = 1).

By default, the interrupt can be generated at any time a falling edge is detected on the input.

When an input edge or level is detected on an I/O line, the corresponding bit in the Interrupt Status register (PIO_ISRx) is set. If the corresponding bit in PIO_IMRx is set, the PIO Controller interrupt line of the I/O group x is asserted.

When the software reads PIO_ISRx, all the interrupts of the I/O group x are automatically cleared. This signifies that all the interrupts that are pending when PIO_ISRx is read must be handled. When an Interrupt is enabled on a “level”, the interrupt is generated as long as the interrupt source is not cleared, even if some read accesses in PIO_ISRx are performed.

Figure 27-4. Event Detector on Input Lines



Example of interrupt generation on following lines:

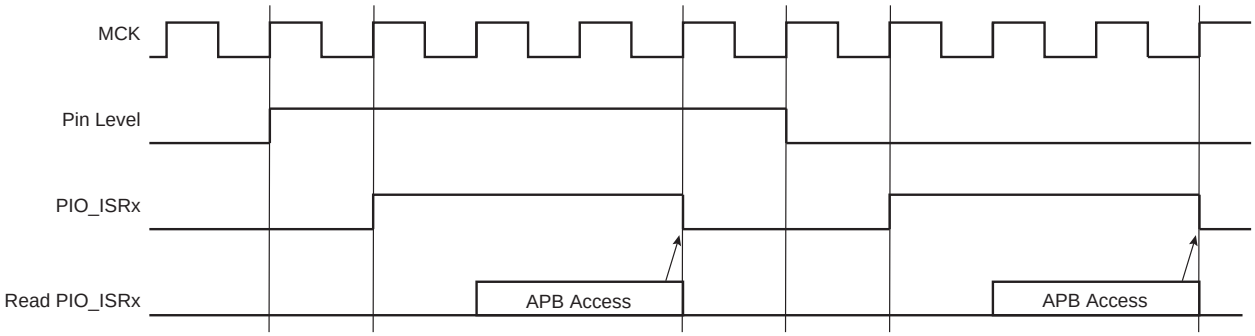
- Rising edge on the PIO line 0 of the I/O group 0 (PIOA)
- Low-level edge on the PIO line 1of the I/O group 0 (PIOA)
- Rising edge on the PIO line 2 of the I/O group 0 (PIOA)
- High-level on the PIO line 3 of the I/O group 0 (PIOA)
- Low-level on the PIO line 4 of the I/O group 0 (PIOA)
- High-level on the PIO line 0 of the I/O group 1 (PIOB)
- Falling edge on the PIO line 1 of the I/O group 1 (PIOB)
- Rising edge on the PIO line 2 of the I/O group 1 (PIOB)
- Any edge on the other lines of the I/O group 1 (PIOB)

The table below details the required configuration for this example.

Table 27-2. Configuration for Example Interrupt Generation

Configuration	Name
PIOA: Interrupt Mode	Enable interrupt sources for lines 0 to 4 of PIOA by writing 32'h0000_001F in PIO_IER0 (offset 0x20)
PIOA: Event Selection	Configure Rising Edge detection for lines 0 and 2: Write 32'h0000_0005 in PIO_MSKR0 (offset 0x0) Write 32'h0100_0000 in PIO_CFGR0 (offset 0x4)
	Configure Low Level detection for lines 1 and 4: Write 32'h0000_0012 in PIO_MSKR0 (offset 0x0) Write 32'h0300_0000 in PIO_CFGR0 (offset 0x4)
	Configure High Level detection for line 3: Write 32'h0000_0008 in PIO_MSKR0 (offset 0x0) Write 32'h0400_0000 in PIO_CFGR0 (offset 0x4)
PIOB: Interrupt Mode	Enable interrupt sources for all lines of PIOB by writing 32'hFFFF_FFFF in PIO_IER1 (offset 0x60)
PIOB: Event Selection	Configure High Level detection for line 0: Write 32'h0000_0001 in PIO_MSKR1 (offset 0x40) Write 32'h0400_0000 in PIO_CFGR1 (offset 0x44)
	Configure Falling Edge detection for line 1: Write 32'h0000_0002 in PIO_MSKR1 (offset 0x40) Write 32'h0000_0000 in PIO_CFGR1 (offset 0x44)
	Configure Rising Edge detection for line 2: Write 32'h0000_0004 in PIO_MSKR1 (offset 0x40) Write 32'h0100_000 in PIO_CFGR1 (offset 0x44)
	Configure Low Level detection for other lines: Write 32'hFFFF_FFF8 in PIO_MSKR1 (offset 0x40) Write 32'h0200_000 in PIO_CFGR1 (offset 0x44)

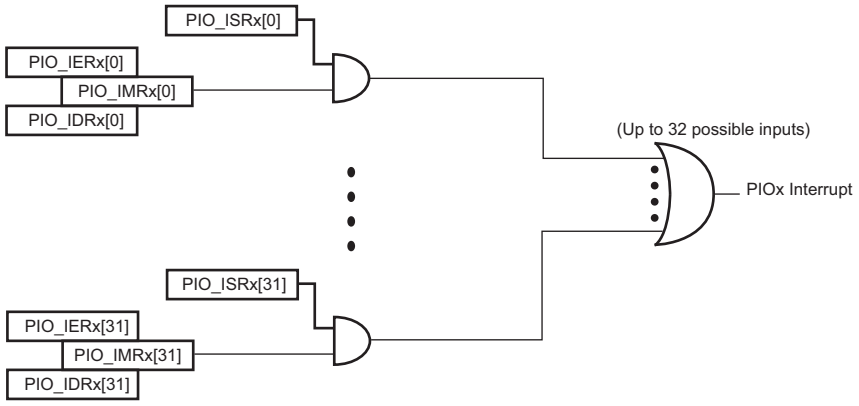
Figure 27-5. Input Change Interrupt Timings When No Additional Interrupt Modes



27.5.10 Interrupt Management

The PIO Controller can drive one interrupt signal per I/O group (see the figure below). Interrupt signals are connected to the interrupt controller of the system.

Figure 27-6. PIO Interrupt Management



27.5.11 I/O Lines Lock

When an I/O line is controlled by a peripheral (particularly the Pulse Width Modulation Controller (PWM)), it can become locked by the action of this peripheral via an input of the PIO Controller. When an I/O line is locked, the following fields in PIO_CFGRx are locked and cannot be modified:

- FUNC: Peripheral selection cannot be changed when the corresponding I/O line is locked.
- PUEN: Pull-Up configuration cannot be changed when the corresponding I/O line is locked.
- PDEN: Pull-Down configuration cannot be changed when the corresponding I/O line is locked.
- OPD: Open-Drain configuration cannot be changed when the corresponding I/O line is locked.

Writing to one of these fields while the corresponding I/O line is locked will have no effect.

The user can know at anytime which I/O line is locked by reading the Lock Status register (PIO_LOCKSR). Once an I/O line is locked, the only way to unlock it is to apply a hardware reset to the PIO Controller.

27.5.12 Programmable I/O Drive

It is possible to configure the I/O drive for pads PA0 to PG31. The I/O drive of the pad can be programmed by writing PIO_CFGRxDRVSTR. For details, refer to the section Electrical Characteristics.

27.5.13 Programmable Schmitt Trigger

It is possible to configure each input for the Schmitt trigger. The Schmitt trigger can be enabled by setting PIO_CFGRx.SCHMITT. By default the Schmitt trigger is active.

27.5.14 I/O Line Configuration Freeze

27.5.14.1 Software Freeze

Once the I/O line configuration is done, it can be frozen by using the I/O Freeze Configuration register (PIO_IOFRx) of the corresponding group.

27.5.14.1.1 Physical Freeze

Setting PIO_IOFRx.FPHY freezes the following fields of the I/O lines defined in PIO_MSKRx:

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable
- PDEN: Pull-Down Enable
- OPD: Open-Drain
- SCHMITT: Schmitt Trigger
- DRVSTR: Drive Strength

When the physical freeze is currently active on an I/O line, the PCFS flag is set when reading the PIO_CFGRx of the I/O line .

Only a hardware reset can release fields listed above.

27.5.14.1.2 Interrupt Freeze

Setting PIO_IOFRx.FINT freezes the EVTSEL field of the I/O lines defined in the PIO_MSKRx:

When the “Interrupt Freeze” is currently active on an I/O line, the ICFS flag is set when reading the PIO_CFGRx of the I/O line .

Only a hardware reset can release fields listed above.

27.5.15 Register Write Protection

To prevent any single software error from corrupting PIO behavior, certain registers in the address space can be write-protected by setting bits WPEN and WPITEN in the PIO Write Protection Mode Register (PIO_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the PIO Write Protection Status Register (PIO_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the PIO_WPSR.

The following registers are write-protected when WPEN is set in PIO_WPMR:

- PIO Mask Register
- PIO Configuration Register
- PIO Slow Clock Divider Debouncing Register

The following registers are write-protected when WPITEN is set in PIO_WPMR:

- PIO Interrupt Enable Register
- PIO Interrupt Disable Register

27.6 I/O Lines Programming Example

The programming example shown in the table below is used to obtain the following configurations:

- PIOA Configuration:
  - 4-bit output port on I/O lines 0 to 3, open-drain, with pull-up resistor
  - Four output signals on I/O lines 4 to 7 (to drive LEDs for example), driven high and low, no pull-up resistor, no pull-down resistor
  - I/O lines 16 to 19 assigned to peripheral A functions with pull-up resistor
  - I/O lines 20 to 23 assigned to peripheral B functions with pull-down resistor
- PIOB Configuration:
  - Four input signals on I/O lines 0 to 3 (to read push-button states for example), with pull-up resistors, glitch filters and input change interrupts
  - Four input signals on I/O lines 12 to 15 to read an external device status (polled, thus no input change interrupt), no pull-up resistor, no glitch filter
  - I/O lines 16 to 23 assigned to peripheral B functions with pull-down resistor
  - I/O lines 24 to 27 assigned to peripheral D with Input Change Interrupt, no pull-up resistor and no pull-down resistor

Table 27-3. Programming Example

Action	Register	Value to be Written
PIOA: 4-bit output port on I/O lines 0 to 3, open-drain, with pull-up resistor	PIO_MSKR0 (offset 0x0)	0x0000000F
	PIO_CFGR0 (offset 0x4)	0x00004300

.....continued

Action	Register	Value to be Written
PIOA: Four output signals on I/O lines 4 to 7 (to drive LEDs for example), driven high and low, no pull-up resistor, no pull-down resistor	PIO_MSKR0 (offset 0x0)	0x000000F0
	PIO_CFGR0 (offset 0x4)	0x00000100
PIOA: I/O lines 16 to 19 assigned to peripheral A functions with pull-up resistor	PIO_MSKR0 (offset 0x0)	0x000F0000
	PIO_CFGR0 (offset 0x4)	0x00000201
PIOA: I/O lines 20 to 23 assigned to peripheral B functions with pull-down resistor	PIO_MSKR0 (offset 0x0)	0x00F00000
	PIO_CFGR0 (offset 0x4)	0x00000402
PIOB: Four input signals on I/O lines 0 to 3 (to read push-button states for example), with pull-up resistors, glitch filters and interrupts on rising edge	PIO_MSKR1 (offset 0x40)	0x0000000F
	PIO_CFGR1 (offset 0x44)	0x01001200
PIOB: Four input signals on I/O line 12 to 15 to read an external device status (polled, thus no input change interrupt), no pull-up resistor, no glitch filter	PIO_MSKR1 (offset 0x40)	0x0000F000
	PIO_CFGR1 (offset 0x44)	0x01001200
PIOB: I/O lines 16 to 23 assigned to peripheral B functions with pull-down resistor	PIO_MSKR1 (offset 0x40)	0x0000000F
	PIO_CFGR1 (offset 0x44)	0x01001200
PIOB: I/O line 24 to 27 assigned to peripheral D with Input Interrupt on both edges, no pull-up resistor and no pull-down resistor	PIO_MSKR1 (offset 0x40)	0x0F000000
	PIO_CFGR1 (offset 0x44)	0x02000004
PIOB: Enable interrupt	PIO_IER1 (offset 0x60)	0x0F00000F

27.7 Register Summary

Each I/O line controlled by the PIO Controller is associated with a bit in each of the PIO Controller registers. Each register is 32 bits wide. If a parallel I/O line is not defined, writing to the corresponding bits has no effect. Undefined bits read zero.

Offset	Name	Bit Pos.								
0x00	PIO_MSKR0	7:0	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		15:8	MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
		23:16	MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
		31:24	MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24
0x04	PIO_CFGR0	7:0						FUNC[2:0]		
		15:8	SCHMITT	OPD				PDEN	PUEN	DIR
		23:16						DRVSTR[2:0]		
		31:24		ICFS	PCFS			EVTSEL[2:0]		
0x08	PIO_PDSR0	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0C	PIO_LOCKSR0	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x10	PIO_SODR0	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x14	PIO_CODR0	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x18	PIO_ODSR0	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x1C ... 0x1F	Reserved									
0x20	PIO_IER0	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x24	PIO_IDR0	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x28	PIO_IMR0	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x2C	PIO_ISR0	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x30 ... 0x3B	Reserved									

.....continued

Offset	Name	Bit Pos.								
0x3C	PIO_IOFR0	7:0							FINT	FPHY
		15:8	FRZKEY[7:0]							
		23:16	FRZKEY[15:8]							
		31:24	FRZKEY[23:16]							
0x40	PIO_MSKR1	7:0	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		15:8	MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
		23:16	MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
		31:24	MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24
0x44	PIO_CFGR1	7:0						FUNC[2:0]		
		15:8	SCHMITT	OPD				PDEN	PUEN	DIR
		23:16						DRVSTR[2:0]		
		31:24		ICFS	PCFS			EVTSEL[2:0]		
0x48	PIO_PDSR1	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x4C	PIO_LOCKSR1	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x50	PIO_SODR1	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x54	PIO_CODR1	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x58	PIO_ODSR1	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x5C ... 0x5F	Reserved									
0x60	PIO_IER1	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x64	PIO_IDR1	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x68	PIO_IMR1	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x6C	PIO_ISR1	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x70 ... 0x7B	Reserved									
0x7C	PIO_IOFR1	7:0							FINT	FPHY
		15:8	FRZKEY[7:0]							
		23:16	FRZKEY[15:8]							
		31:24	FRZKEY[23:16]							

.....continued

Offset	Name	Bit Pos.								
0x80	PIO_MSKR2	7:0	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		15:8	MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
		23:16	MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
		31:24	MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24
0x84	PIO_CFGR2	7:0						FUNC[2:0]		
		15:8	SCHMITT	OPD				PDEN	PUEN	DIR
		23:16						DRVSTR[2:0]		
		31:24		ICFS	PCFS			EVTSEL[2:0]		
0x88	PIO_PDSR2	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x8C	PIO_LOCKSR2	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x90	PIO_SODR2	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x94	PIO_CODR2	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x98	PIO_ODSR2	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x9C ... 0x9F	Reserved									
0xA0	PIO_IER2	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xA4	PIO_IDR2	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xA8	PIO_IMR2	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xAC	PIO_ISR2	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xB0 ... 0xBB	Reserved									
0xBC	PIO_IOFR2	7:0							FINT	FPHY
		15:8	FRZKEY[7:0]							
		23:16	FRZKEY[15:8]							
		31:24	FRZKEY[23:16]							
0xC0	PIO_MSKR3	7:0	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		15:8	MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
		23:16	MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
		31:24	MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24

.....continued

Offset	Name	Bit Pos.								
0xC4	PIO_CFGR3	7:0						FUNC[2:0]		
		15:8	SCHMITT	OPD				PDEN	PUEN	DIR
		23:16						DRVSTR[2:0]		
		31:24		ICFS	PCFS			EVTSEL[2:0]		
0xC8	PIO_PDSR3	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xCC	PIO_LOCKSR3	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xD0	PIO_SODR3	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xD4	PIO_CODR3	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xD8	PIO_ODSR3	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xDC ... 0xDF	Reserved									
0xE0	PIO_IER3	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xE4	PIO_IDR3	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xE8	PIO_IMR3	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xEC	PIO_ISR3	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0xF0 ... 0xFB	Reserved									
0xFC	PIO_IOFR3	7:0							FINT	FPHY
		15:8	FRZKEY[7:0]							
		23:16	FRZKEY[15:8]							
		31:24	FRZKEY[23:16]							
0x0100	PIO_MSKR4	7:0	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		15:8	MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
		23:16	MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
		31:24	MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24
0x0104	PIO_CFGR4	7:0						FUNC[2:0]		
		15:8	SCHMITT	OPD				PDEN	PUEN	DIR
		23:16						DRVSTR[2:0]		
		31:24		ICFS	PCFS			EVTSEL[2:0]		

.....continued

Offset	Name	Bit Pos.								
0x0108	PIO_PDSR4	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x010C	PIO_LOCKSR4	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0110	PIO_SODR4	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0114	PIO_CODR4	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0118	PIO_ODSR4	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x011C ... 0x011F	Reserved									
0x0120	PIO_IER4	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0124	PIO_IDR4	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0128	PIO_IMR4	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x012C	PIO_ISR4	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0130 ... 0x013B	Reserved									
0x013C	PIO_IOFR4	7:0							FINT	FPHY
		15:8	FRZKEY[7:0]							
		23:16	FRZKEY[15:8]							
		31:24	FRZKEY[23:16]							
0x0140	PIO_MSKR5	7:0	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		15:8	MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
		23:16	MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
		31:24	MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24
0x0144	PIO_CFGR5	7:0						FUNC[2:0]		
		15:8	SCHMITT	OPD				PDEN	PUEN	DIR
		23:16						DRVSTR[2:0]		
		31:24		ICFS	PCFS			EVTSEL[2:0]		
0x0148	PIO_PDSR5	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24

.....continued

Offset	Name	Bit Pos.								
0x014C	PIO_LOCKSR5	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0150	PIO_SODR5	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0154	PIO_CODR5	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0158	PIO_ODSR5	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x015C ... 0x015F	Reserved									
0x0160	PIO_IER5	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0164	PIO_IDR5	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0168	PIO_IMR5	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x016C	PIO_ISR5	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0170 ... 0x017B	Reserved									
0x017C	PIO_IOFR5	7:0							FINT	FPHY
		15:8	FRZKEY[7:0]							
		23:16	FRZKEY[15:8]							
		31:24	FRZKEY[23:16]							
0x0180	PIO_MSKR6	7:0	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		15:8	MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
		23:16	MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
		31:24	MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24
0x0184	PIO_CFGR6	7:0						FUNC[2:0]		
		15:8	SCHMITT	OPD				PDEN	PUEN	DIR
		23:16						DRVSTR[2:0]		
		31:24		ICFS	PCFS			EVTSEL[2:0]		
0x0188	PIO_PDSR6	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x018C	PIO_LOCKSR6	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24

.....continued

Offset	Name	Bit Pos.								
0x0190	PIO_SODR6	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0194	PIO_CODR6	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x0198	PIO_ODSR6	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x019C ... 0x019F	Reserved									
0x01A0	PIO_IER6	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x01A4	PIO_IDR6	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x01A8	PIO_IMR6	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x01AC	PIO_ISR6	7:0	P7	P6	P5	P4	P3	P2	P1	P0
		15:8	P15	P14	P13	P12	P11	P10	P9	P8
		23:16	P23	P22	P21	P20	P19	P18	P17	P16
		31:24	P31	P30	P29	P28	P27	P26	P25	P24
0x01B0 ... 0x01BB	Reserved									
0x01BC	PIO_IOFR6	7:0							FINT	FPHY
		15:8	FRZKEY[7:0]							
		23:16	FRZKEY[15:8]							
		31:24	FRZKEY[23:16]							
0x01C0 ... 0x04FF	Reserved									
0x0500	PIO_SCDR	7:0	DIV[7:0]							
		15:8			DIV[13:8]					
		23:16								
		31:24								
0x0504 ... 0x05DF	Reserved									
0x05E0	PIO_WPMR	7:0							WPITEN	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0x05E4	PIO_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16	WPVSRC[15:8]							
		31:24								

27.7.1 PIO Mask Register

Name: PIO_MSKRx
Offset: 0x00 + x*0x40 [x=0..6]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – MSKy PIO Line y Mask
These bits define the I/O lines to be configured when writing the PIO Configuration Register.
0 (DISABLED): Writing the PIO_CFGRx, PIO_ODSRx or PIO_IOFRx does not affect the corresponding I/O line configuration.
1 (ENABLED): Writing the PIO_CFGRx, PIO_ODSRx or PIO_IOFRx updates the corresponding I/O line configuration.

27.7.2 PIO Configuration Register

Name: PIO_CFGRx
Offset: 0x04 + x*0x40 [x=0..6]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

Writing this register will only affect I/O lines enabled in the PIO_MSKRx.

Bit	31	30	29	28	27	26	25	24
		ICFS	PCFS			EVTSEL[2:0]		
Access		R	R			R/W	R/W	R/W
Reset		0	0			0	0	0

Bit	23	22	21	20	19	18	17	16
						DRVSTR[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bit	15	14	13	12	11	10	9	8
	SCHMITT	OPD				PDEN	PUEN	DIR
Access	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

Bit	7	6	5	4	3	2	1	0
						FUNC[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bit 30 – ICFS Interrupt Configuration Freeze Status (read-only)
This bit gives information about the freeze state of the field EVTSEL of the read I/O line configuration.
0 (NOT_FROZEN): The field is not frozen and can be written for this I/O line.
1 (FROZEN): The field is frozen and cannot be written for this I/O line. Only a hardware reset can release the field.

Bit 29 – PCFS Physical Configuration Freeze Status (read-only)
This bit gives information about the freeze state of the following fields of the read I/O line configuration:

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable
- PDEN: Pull-Down Enable
- OPD: Open-Drain
- SCHMITT: Schmitt Trigger
- DRVSTR: Drive Strength

0 (NOT_FROZEN): The fields are not frozen and can be written for this I/O line.
1 (FROZEN): The fields are frozen and cannot be written for this I/O line. Only a hardware reset can release these fields.

Bits 26:24 – EVTSEL[2:0] Event Selection
This field defines the type of event to detect on the I/O lines of the I/O group x according to the PIO Mask Register.

Value	Name	Description
0	FALLING	Event detection on input falling edge
1	RISING	Event detection on input rising edge
2	BOTH	Event detection on input both edge
3	LOW	Event detection on low level input

Value	Name	Description
4	HIGH	Event detection on high level input
5	–	Reserved
6	–	Reserved
7	–	Reserved

Bits 18:16 – DRVSTR[2:0] Drive Strength
This field defines the drive strength of the I/O lines of the I/O group x according to the PIO Mask Register.

Value	Name	Description
0	OUT_2m	Output drive is 2 mA.
1	OUT_4m	Output drive is 4 mA.
2	OUT_8m	Output drive is 8 mA.
3	OUT_16m	Output drive is 16 mA.
4	OUT_24m	Output drive is 24 mA.
5	OUT_32m	Output drive is 32 mA.
6	OUT_40m	Output drive is 40 mA.
7	OUT_48m	Output drive is 48 mA.

Bit 15 – SCHMITT Schmitt Trigger
This bit defines the Schmitt trigger configuration of the I/O lines of the I/O group x according to the PIO Mask Register.
0 (ENABLED): Schmitt trigger is enabled for the selected I/O lines.
1 (DISABLED): Schmitt trigger is disabled for the selected I/O lines.

Bit 14 – OPD Open Drain
This bit defines the open drain configuration of the I/O lines of the I/O group x according to the PIO Mask Register.
0 (DISABLED): The open-drain is disabled for the selected I/O lines. I/O lines are driven at high- and low-level.
1 (ENABLED): The open-drain is enabled for the selected I/O lines. I/O lines are driven at low-level only.

Bit 10 – PDEN Pull-Down Enable
This bit defines the pull-down configuration of the I/O lines of the I/O group x according to the PIO Mask Register.
PDEN can be written to 1 only if PUEN is written to 0.
0 (DISABLED): Pull-down is disabled for the selected I/O lines.
1 (ENABLED): Pull-down is enabled for the selected I/O lines only if PUEN is 0.

Bit 9 – PUEN Pull-Up Enable
This bit defines the pull-up configuration of the I/O lines of the I/O group x according to the PIO Mask Register.
0 (DISABLED): Pull-up is disabled for the selected I/O lines.
1 (ENABLED): Pull-up is enabled for the selected I/O lines.

Bit 8 – DIR Direction
This bit defines the direction of the I/O lines of the I/O group x according to the PIO Mask Register.
0 (INPUT): The selected I/O lines are pure inputs.
1 (OUTPUT): The selected I/O lines are enabled in output.

Bits 2:0 – FUNC[2:0] I/O Line Function
This field defines the function for I/O lines of the I/O group x according to the PIO Mask Register.

Value	Name	Description
0	GPIO	Selects the PIO mode for the selected I/O lines.
1	PERIPH_A	Selects peripheral A for the selected I/O lines.
2	PERIPH_B	Selects peripheral B for the selected I/O lines.
3	PERIPH_C	Selects peripheral C for the selected I/O lines.
4	PERIPH_D	Selects peripheral D for the selected I/O lines.
5	Reserved	–
6	Reserved	–
7	Reserved	–

27.7.3 PIO Pin Data Status Register

Name: PIO_PDSRx
Offset: 0x08 + x*0x40 [x=0..6]
Reset: 0x00000000
Property: Read-only

Reset value of PIO_PDSR depends on the level of the I/O lines. Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSR reads the levels present on the I/O line at the time the clock was disabled.

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – Px Input Data Status

Value	Description
0	The I/O line of the I/O group x is at level 0.
1	The I/O line of the I/O group x is at level 1.

27.7.4 PIO Lock Status Register

Name: PIO_LOCKSRx
Offset: 0x0C + x*0x40 [x=0..6]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – Px Lock Status

Value	Description
0	The I/O line of the I/O group x is not locked.
1	The I/O line of the I/O group x is locked.

27.7.5 PIO Set Output Data Register

Name: PIO_SODRx
Offset: 0x10 + x*0x40 [x=0..6]
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – Px Set Output Data

Value	Description
0	No effect.
1	Sets the data to be driven on the I/O line of I/O group x.

27.7.6 PIO Clear Output Data Register

Name: PIO_CODRx
Offset: 0x14 + x*0x40 [x=0..6]
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – Px Clear Output Data

Value	Description
0	No effect.
1	Clears the data to be driven on the I/O line of the I/O group x.

27.7.7 PIO Output Data Status Register

Name: PIO_ODSRx
Offset: 0x18 + x*0x40 [x=0..6]
Reset: 0x00000000
Property: Read/Write

Writing this register will only affect I/O lines enabled in the PIO_MSKRx.

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – Px Output Data Status

Value	Description
0	The data to be driven on the I/O line of the I/O group x is 0.
1	The data to be driven on the I/O line of the I/O group x is 1.

27.7.8 PIO Interrupt Enable Register

Name: PIO_IERx
Offset: 0x20 + x*0x40 [x=0..6]
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the PIO Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – Px Input Change Interrupt Enable

Value	Description
0	No effect.
1	Enables the Input Change interrupt on the I/O line of the I/O group x.

27.7.9 PIO Interrupt Disable Register

Name: PIO_IDRx
Offset: 0x24 + x*0x40 [x=0..6]
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the PIO Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – Px Input Change Interrupt Disable

Value	Description
0	No effect.
1	Disables the Input Change interrupt on the I/O line of the I/O group x.

27.7.10 PIO Interrupt Mask Register

Name: PIO_IMRx
Offset: 0x28 + x*0x40 [x=0..6]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – Px Input Change Interrupt Mask

Value	Description
0	Input Change interrupt is disabled on the I/O line of the I/O group x.
1	Input Change interrupt is enabled on the I/O line of the I/O group x.

27.7.11 PIO Interrupt Status Register

Name: PIO_ISRx
Offset: 0x2C + x*0x40 [x=0..6]
Reset: 0x00000000
Property: Read-only

PIO_ISR is reset at 0x00000000. However, the first read of the register may read a different value as input changes may have occurred.

Bit	31	30	29	28	27	26	25	24
	P31	P30	P29	P28	P27	P26	P25	P24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	P23	P22	P21	P20	P19	P18	P17	P16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	P15	P14	P13	P12	P11	P10	P9	P8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	P7	P6	P5	P4	P3	P2	P1	P0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – Px Input Change Interrupt Status

Value	Description
0	No Input Change has been detected on the I/O line of the I/O group x since PIO_ISRx was last read or since reset.
1	At least one Input Change has been detected on the I/O line of the I/O group since PIO_ISRx was last read or since reset.

27.7.12 PIO I/O Freeze Configuration Register

Name: PIO_IOFRx
Offset: 0x3C + x*0x40 [x=0..6]
Reset: –
Property: Write-only

Writing this register will only affect I/O lines enabled in the PIO_MSKRx.

Bit	31	30	29	28	27	26	25	24
	FRZKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	FRZKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	FRZKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
							FINT	FPHY
Access							W	W
Reset							–	–

Bits 31:8 – FRZKEY[23:0] Freeze Key

Value	Name	Description
0x494F46	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit.

Bit 1 – FINT Freeze Interrupt Configuration
Only a hardware reset can reset the FINT bit.

Value	Description
0	No effect.
1	Freezes the EVTSEL field if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII):

Bit 0 – FPHY Freeze Physical Configuration
Only a hardware reset can reset the FPHY bit.

Value	Description
0	No effect.
1	Freezes the following configuration fields if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII): • FUNC: I/O Line Function • DIR: Direction • PUEN: Pull-Up Enable • PDEN: Pull-Down Enable • OPD: Open-Drain • SCHMITT: Schmitt Trigger • DRVSTR: Drive Strength

27.7.13 PIO Slow Clock Divider Debouncing Register

Name: PIO_SCDR
Offset: 0x500
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
			DIV[13:8]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DIV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 13:0 – DIV[13:0] Slow Clock Divider Selection for Debouncing
$t_{div\_slck} = ((DIV + 1) \times 2) \times t_{slck}$

27.7.14 PIO Write Protection Mode Register

Name: PIO_WPMR
Offset: 0x5E0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
							WPITEN	WPEN
Access							R/W	R/W
Reset							0	0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x50494F	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

Bit 1 – WPITEN Write Protection Interrupt Enable
See 27.5.15 Register Write Protection for the list of registers that can be protected.

Value	Description
0	Disables the write protection on interrupt registers if WPKEY corresponds to 0x50494F (“PIO” in ASCII).
1	Enables the write protection on interrupt registers if WPKEY corresponds to 0x50494F (“PIO” in ASCII).

Bit 0 – WPEN Write Protection Enable
See 27.5.15 Register Write Protection for the list of registers that can be protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII).

27.7.15 PIO Write Protection Status Register

Name: PIO_WPSR
Offset: 0x5E4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	WPVSRC[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 23:8 – WPVSRC[15:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protection violation has occurred since the last read of the PIO_WPSR.
1	A write protection violation has occurred since the last read of the PIO_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

28. Hardened External Memory Controller (HEMC)

28.1 Description

The Hardened External Memory Controller (HEMC) is designed to ensure the successful data transfer between several external devices and the embedded Memory Controller of an ARM-based device dedicated to aerospace applications.

The HEMC features two external memory controllers, a Hardened Static Memory Controller (HSMC) and a Hardened SDRAM Controller (HSDRAMC). These external memory controllers are capable of handling several types of external memory and peripheral devices, such as SRAM, PROM, Flash and SDRAM memories. The HEMC also supports an Hardened Error Correction Code (HECC) to manage data integrity on the external memories.

The flexible width of the memory data bus allows the end user to select between data transfer performance on the memory interface and optimized footprint of the memories.

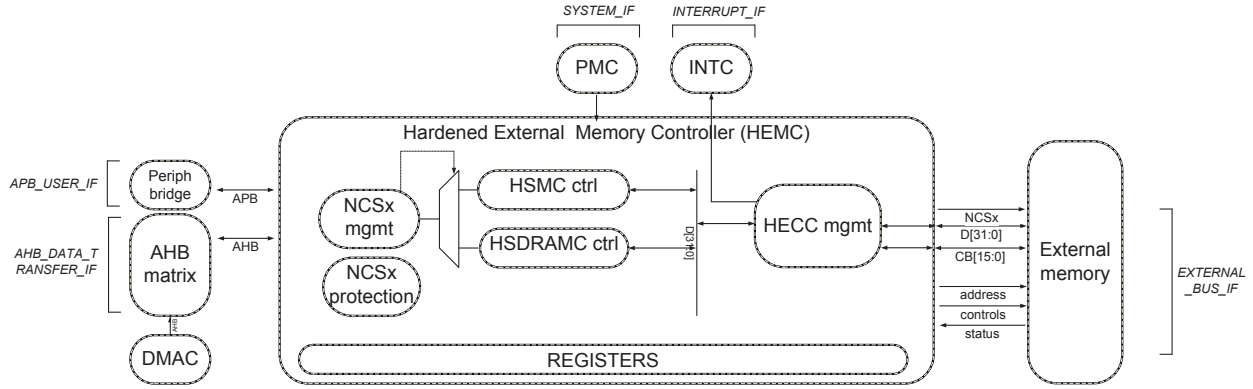
The HEMC handles data transfers with up to six external devices, each assigned to one of the six customizable address spaces. Data transfers are performed through a 8- to 48-bit data bus (including HECC checkbits), an external address bus of up to 28 bits, up to six chip select lines and several control pins that are generally multiplexed between the different external Memory Controllers.

28.2 Embedded Characteristics

- HSMC Controller
  - 8-bit, 16-bit and 32/40/48-bit PROM and Flash memories
  - 8-bit, 16-bit and 32/40/48-bit SRAM memories
- HSDRAMC Controller
  - 16-bit and 32/40/48-bit SDRAM Memories
- Memory Size
  - PROM, SRAM up to 1.5 GBytes
  - SDRAM up to 2 GBytes
- AMBA Compliant
- Read-Modify-Write (RMW)
- Word Burst, Word, Halfword and Byte Transfers
- Embedded Access Protection
  - AMBA read/write and user/superuser access can be accepted or denied by HEMC
- Interrupt Capabilities
- Six Independent Chip Selects (NCSx [x=0 to 5])
  - NCSx supports any type of memory
  - Configurable NCSx start address
  - NCSx memory sizes from 8 Kbytes to 512 Mbytes
- On-the-fly Hardened Error Correction Code (HECC)

28.3 Block Diagram

Figure 28-1. Organization of the HEMC



28.4 I/O Lines

The HEMC has interfaces to perform communication between external memories and the internal AMBA 32-bit bus. The signals are detailed in the table below.

Table 28-1. HEMC I/O Lines Description

Signal Name	Signal Direction	Signal Width	Signal Polarity	Signal Description
HEMC				
A[27:0]	Output	10	–	Common address line
D[31:0]	Input/Output	32	–	Common data bits
CB[15:0]	Input/Output	16	–	Common check bits
EXT_CTRL_BUFFER	Output	1	High	Common external buffer control signal
NCS[5:0]	Output	6	Low	Common chip selects lines signals
HSMC				
NRD	Output	1	Low	HSMC read signal
NWE	Output	1	Low	HSMC write enable signal
NWR[4:0]	Output	5	Low	HSMC write byte enable signals
NWAIT	Input	1	Low	HSMC external wait signal
HSDRAMC				
SDCK	Output	1	–	HSDRAMC clock signal
SDCKE	Output	1	High	HSDRAMC clock enable signal
SDNBS[4:0]	Output	5	Low	HSDRAMC byte mask signals
SDWE	Output	1	Low	HSDRAMC write enable signal
CAS	Output	1	Low	HSDRAMC column signal
RAS	Output	1	Low	HSDRAMC row signal
SDA10	Output	1	–	HSDRAMC address 10 line

BA[1:0]	Output	2	–	HSDRAMC bank select

28.5 Typical Applications

28.5.1 Hardware Interface

The HEMC supports PROM, Flash, SRAM and SDRAM memories, with data widths of 8, 16 or 32/40/48 bits.

Note: When HECC is enabled on 32-bit memories, the global width is either 40 bits ((Hamming 32,7) used) or 48 bits ((BCH(32,12) used).

Table 28-2. Typical Applications Supported – HSMC – 8/16-bit Modes

HEMC Signal Name	8-bit Mode Hamming(32,7) or BCH(32,12) ON or OFF	16-bit Mode Hamming(32,7) or BCH(32,12) ON or OFF	
	1 x 8-bit static devices	2 x 8-bit static devices	1 x 16-bit static devices
A[27:2]	A[27:2]	A[26:1]	A[26:1]
A[1]	A[1]	A[0]	A[0]
A[0]	A[0]	–	–
D[7:0]	D[7:0]	D[7:0]	D[7:0]
D[15:8]	–	D[7:0]	D[15:8]
D[23:16]	–	–	–
D[31:24]	–	–	–
CB[7:0]	–	–	–
CB[15:8]	–	–	–
NCS[0]	CS(1)	CS(1)	CS(1)
NCS[1]			
NCS[2]			
NCS[3]			
NCS[4]			
NCS[5]			
NRD	OE	OE	OE
NWE	WE(2)	–	WE
NWR[0]	WE(2)	WE	BLE(3)
NWR[1]	–	WE	BHE(3)
NWR[2]	–	–	–
NWR[3]	–	–	–
NWR[4]	–	–	–

Note:

1. CS can be connected to one of the NCS[5:0].
2. WE can be connected either to the global write enable NWE or to the byte write enable signal (NWR[0]).
3. Depending on the model of 16-bit memory used, to allow the memory to be read:
   – BLE shall be connected to AND(NRD, NWR[0]).
   – BHE shall be connected to AND(NRD, NWR[1]).

   See the memory datasheet for details.

Table 28-3. Typical Applications Supported – HSMC – 32/40/48-bit Mode

HEMC Signal Name	32-bit Mode Hamming(32,7) or BCH(32,12) OFF			40-bit Mode 32-bit Mode +Hamming(32,7) ON			48-bit Mode 32-bit Mode + BCH(32,12) ON			
	4 x 8-bit static devices	2 x 16-bit static devices	1 x 32-bit static devices	5 x 8-bit static devices	2 x 16-bit + 1 x 8-bit static devices	1 x 32-bit + 1 x 8-bit static devices	6 x 8-bit static devices	3 x 16-bit static devices	1 x 32-bit + 2 x 8-bit static devices	1 x 32-bit + 1 x 16-bit static devices
A[27:2]	A[25:0]	A[25:0]	A[25:0]	A[25:0]	A[25:0]	A[25:0]	A[25:0]	A[25:0]	A[25:0]	A[25:0]
A[1]	–	–	–	–	–	–	–	–	–	–
A[0]	–	–	–	–	–	–	–	–	–	–
D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]
D[15:8]	D[7:0]	D[15:8]	D[15:8]	D[7:0]	D[15:8]	D[15:8]	D[7:0]	D[15:8]	D[15:8]	D[15:8]
D[23:16]	D[7:0]	D[7:0]	D[23:16]	D[7:0]	D[7:0]	D[23:16]	D[7:0]	D[7:0]	D[23:16]	D[23:16]
D[31:24]	D[7:0]	D[15:8]	D[31:24]	D[7:0]	D[15:8]	D[31:24]	D[7:0]	D[15:8]	D[31:24]	D[31:24]
CB[7:0]	–	–	–	D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]	D[7:0]
CB[15:8]	–	–	–	–	–	–	D[7:0]	D[15:8]	D[7:0]	D [15:8]
NCS[0]	CS(1)	CS(1)	CS(1)	CS(1)	CS(1)	CS(1)	CS(1)	CS(1)	CS(1)	CS(1)
NCS[1]										
NCS[2]										
NCS[3]										
NCS[4]										
NCS[5]										
NRD	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE
NWE	–	WE	WE(2)	–	WE	WE(2)	–	WE	WE(2)	WE
NWR[0]	WE	BLE(3)	WE(2)	WE	BLE(3)	WE(2)	WE	BLE(3)	WE(2)	–
NWR[1]	WE	BHE(3)	WE(2)	WE	BHE(3)	WE(2)	WE	BHE(3)	WE(2)	BHE(3)
NWR[2]	WE	BLE(3)	WE(2)	WE	BLE (3)	WE(2)	WE	BLE(3)	WE(2)	–
NWR[3]	WE	BHE(3)	WE(2)	WE	BHE(3)	WE(2)	WE	BHE(3)	WE(2)	–

.....continued

HEMC Signal Name	32-bit Mode Hamming(32,7) or BCH(32,12) OFF			40-bit Mode 32-bit Mode +Hamming(32,7) ON			48-bit Mode 32-bit Mode + BCH(32,12) ON			
	4 x 8-bit static devices	2 x 16-bit static devices	1 x 32-bit static devices	5 x 8-bit static devices	2 x 16-bit + 1 x 8-bit static devices	1 x 32-bit + 1 x 8-bit static devices	6 x 8-bit static devices	3 x 16-bit static devices	1 x 32-bit + 2 x 8-bit static devices	1 x 32-bit + 1 x 16-bit static devices
NWR[4]	–	–	–	WE	WE	WE	WE	–	WE	BLE(3)

Note:

1. CS can be connected to one of the NCS[5:0].
2. Depending on the number of WE available (n) on a 32-bit memory:
   - if n=1, the WE signal must be connected to the global write enable NWE.
   - if n=4, each WE signal must be connected to the byte write enable signal (NWR[x]).
3. Depending on the model of 16-bit memory used, to allow the memory to be read:
   - BLE shall be connected to AND (NRD, NWR[0])
   - BHE shall be connected to AND (NRD, NWR[1])
   - BLE shall be connected to AND (NRD, NWR[2])
   - BHE shall be connected to AND (NRD, NWR[3])

   See the memory datasheet for details.

Table 28-4. Typical Applications Supported – HSDRAMC – 16-bit Mode (max size example)

HEMC Signal Name	16-bit Mode Hamming(32,7) or BCH(32,12) OFF/ON
	1 x 16-bit static device
A[0]	–
A[1]	A[0]
A[10:2]	A[9:1]
SDA10	A[10]
A[11]	–
A[14:12]	A[13:11]
A15	BA[0]
A16	BA[1]
D[7:0]	D[7:0]
D[15:8]	D[15:8]
D[23:16]	D[7:0]
D[31:24]	D[15:8]
CB[7:0]	–
CB[15:8]	–

.....continued	
HEMC Signal Name	**16-bit Mode Hamming(32,7) or BCH(32,12) OFF/ON**
	1 x 16-bit static device
NCS[0]	CS[1]
NCS[1]	
NCS[2]	
NCS[3]	
NCS[4]	
NCS[5]	
SDCK	CK
SDCKE	CKE
SDNBS[0]	DQM0
SDNBS[1]	DQM1
SDNBS[2]	–
SDNBS[3]	–
SDNBS[4]	–
SDWE	WE
CAS	CAS
RAS	RAS

Note:

1. CS can be connected to one of the NCS[5:0].

Table 28-5. Typical Applications Supported – HSDRAMC – 32/40/48-bit Mode (max size example)

HEMC Signal Name	32-bit Mode Hamming(32,7) or BCH(32,12) OFF	32-bit Mode Hamming(32,7) or BCH(32,12) ON
	2 x 16-bit static devices	**3 x 16-bit static devices**
A[0]	–	–
A[1]	–	–
A[11:2]	A[9:0]	A[9:0]
SDA10	A[10]	A[10]
A[12]	–	–
A[15:13]	A[13:11]	A[13:11]
A16	BA[0]	BA[0]
A17	BA[1]	BA[1]
D[7:0]	D[7:0]	D[7:0]
D[15:8]	D[15:8]	D[15:8]
D[23:16]	D[7:0]	D[7:0]

.....continued		
HEMC Signal Name	32-bit Mode Hamming(32,7) or BCH(32,12) OFF	32-bit Mode Hamming(32,7) or BCH(32,12) ON
	2 x 16-bit static devices	3 x 16-bit static devices
D[31:24]	D[15:8]	D[15:8]
CB[7:0]	–	D[7:0]
CB[15:8]	–	D[15:8]
NCS[0]	CS(1)	CS(1)
NCS[1]		
NCS[2]		
NCS[3]		
NCS[4]		
NCS[5]		
SDCK	CK	CK
SDCKE	CKE	CKE
SDNBS[0]	DQM0	DQM0
SDNBS[1]	DQM1	DQM1
SDNBS[2]	DQM0	DQM0
SDNBS[3]	DQM1	DQM1
SDNBS[4]	–	DQM0
SDWE	WE	WE
CAS	CAS	CAS
RAS	RAS	RAS

Note:

1. CS can be connected to one of the NCS[5:0].

28.6 Functional Description

The HEMC transfers data between the internal AMBA AHB Bus handled by the Bus Matrix and the external memories. It is composed of the following elements:

- HEMC NCSx management, to configure dynamically each NCSx memory area
- HEMC NCSx protection, to accept or reject internal AMBA AHB data transactions
- HSMC controller, to access external PROM/SRAM memories
- HSDRAMC controller, to access external SDRAM memories
- A chip select assignment feature that assigns an AHB address space to the external devices
- A multiplex controller circuit that shares the pins between the different memory controllers
- HECC management, to increase the reliability and to protect the data against radiation issues

All these elements are accessible via end user registers.

Note: For details, refer to the sections:

- Hardened Static Memory Controller (HSMC)
- Hardened SDRAM Controller (HSDRAMC)
- Hardened Error Correction Code (HECC) Controller

28.6.1 HEMC NCSx Management

28.6.1.1 Overview

The HEMC embeds six independent chip selects (NCS) that can be used for static or dynamic memories. The goal of the NCS management block is to configure each NCS memory area depending on the end user application needs.

28.6.1.2 Memory Type Selection

Each NCSx can support either static (SRAM, PROM, Flash) or dynamic (SDRAMC) memories. The configuration of the memory to support is performed through the HEMC_CR_NCSx register:

- To select static memories, clear the HEMC_CR_NCSx.TYPE.
- To select dynamic memories, set the HEMC_CR_NCSx.TYPE.

28.6.1.3 NCSx Bank Size Configuration

Each NCSx can support static memories ranging from from 8 Kbytes to 256 Mbytes. Dynamic memories up to 512 Mbytes are supported.

The bank size configuration is performed via HEMC_CR_NCSx.BANKSIZE.

The table below presents allowable bank size configurations and their associated code.

Table 28-6. Allowable Bank Size Values

Code	Size	Comments
0b00000	8 Kbytes	–
0b00001	16 Kbytes	–
0b00010	32 Kbytes	–
0b00011	64 Kbytes	–
0b00100	128 Kbytes	–
0b00101	256 Kbytes	–
0b00110	512 Kbytes	–
0b00111	1 MBytes	–
0b01000	2 MBytes	–
0b01001	4 Mbytes	–
0b01010	8 Mbytes	–
0b01011	16 Mbytes	–
0b01100	32 Mbytes	–
0b01101	64 Mbytes	–
0b01110	128 Mbytes	–
0b01111	256 Mbytes	–
0b10000	512 Mbytes	Valid for SDRAM Controller only
0b11111	NOT_USED	Configuration for a disabled chip select
Other values	RESERVED	–

Note: For the unused NCSx memory area, the corresponding HEMC_CR_NCSx.BANKSIZE must be set to 0x1F and HEMC_CR_NCSx.ADDBASE must be set to 0x3_FFFF.

28.6.1.4 NCSx Start Address

Each NCSx memory may be located anywhere inside the “Memories” section of the product memory mapping. To set up the location of the memory of an NCSx, the AHB relative(1) start address must be configured. This is done using the HEMC_CR_NCSx.ADDBASE register field(2).

Note:

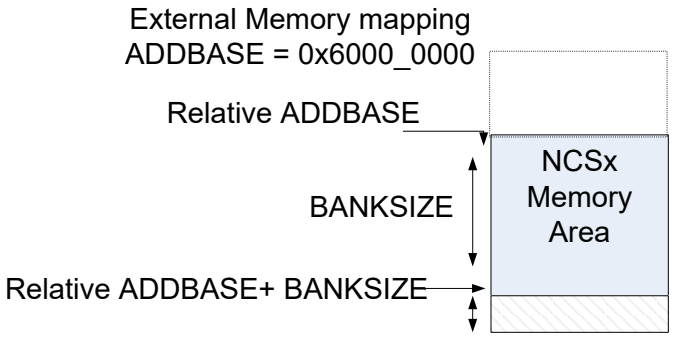
1. Because the authorized areas are inside the “Memories” section, all the AHB start address information is relative to the beginning of the “Memories” section of the product memory mapping. In our case, all the AHB start addresses are relative to 0x6000_0000.
2. Because the smallest allowable bank size is equal to 8 Kbytes, only the 18 MSB of the AHB relative start address are significant.
3. For the unused NCSx memory area, the corresponding HEMC_CR_NCSx.BANKSIZE must be set to 0x1F and HEMC_CR_NCSx.ADDBASE must be set to 0x3_FFFF.

28.6.1.5 NCSx Memory Area Definition

The NCSx memory area definition is performed by set up the previous parameters described in the three previous sections.

The figure below illustrates the NCSx memory area definition, for the relative start address and bank size parameters.

Figure 28-2. NCSx Memory Area Definition



28.6.1.6 Restrictions

Each NCSx memory area definition must be strictly in the range of the “Memories” section of the product mapping.

Moreover, if several NCSx memory areas are defined, they must not overlap each other. Otherwise, behavior is unpredictable.

The file “HEMC_Config_Manager.xlsm”, available on www.microchip.com, is provided to the end user to ensure correct configuration.

28.6.1.7 Example

The following example illustrates how to set up the HEMC_CR_NCSx register.

Example: NCS[1] shall cover the area 0x7000_00000 to 0x8000_0000 -1, with an SRAM.

1. The relative start address is 0x1000_0000. The 18-bit MSB are equal to 0x4_0000: HEMC_CR_NCS1.ADDBASE= 0x4_000.
2. The bank size is 0x8000_0000 – 0x7000_0000=0x1000_0000 = 256 Mbytes: HEMC_CR_NCS1.BANKSIZE= 0x10.
3. The memory type is HSMC: HEMC_CR_NCS1.TYPE= 0.

28.6.1.8 Boot Memory

If the application needs to boot on an external static PROM memory, the following constraints must be respected:

- Only the NCS0 has a boot capability. Therefore, the BOOT PROM memory must be connected on this chip select.

- In order to boot with the requested configuration, the NCS0 memory area is automatically configured from the values read on the following functions, when product reset is released:
  - CFG[1:0] configures NCS0 data width: 8-bit (‘00’),16-bit (‘01’) or 32-bit (‘10’)
  - CFG2 activates (‘1’) or not (‘0’) HECC
  - CFG3 selects either Hamming(32,7) (‘0’) or BCH(32,12) (‘1’) HECC

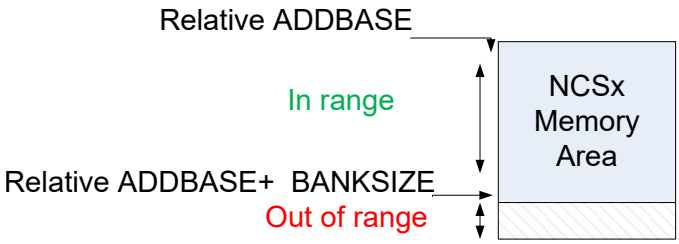
Note: Refer to the PIO multiplexing for the PIO of each CFGx function.

28.6.1.9 In Range/Out of Range Definition

When AHB data transfer occurs, the HEMC verifies whether the AHB address requested is in one of the NCSx memory areas previously defined (in range) or not (out of range).

The following figure presents the "in range" and "out of range" definition on an NCSx memory area.

Figure 28-3. Out of Range Definition



If the AHB address requested is in range, the current AHB transaction occurred normally.

If the AHB address requested is out of range:

- A flag Out_of_Range is raised automatically (HEMC_SR.OUTOFRANGE=’1’).
- An interrupt can also be raised (HEMC_ISR. OUTOFRANGE =’1’), if it is previously enabled (HEMC_IER. OUTOFRANGE =’1’).
- The current AHB transaction is aborted.

28.6.2 HEMC NCSx Protection

28.6.2.1 Overview

The HEMC embeds a protection feature with the following characteristics:

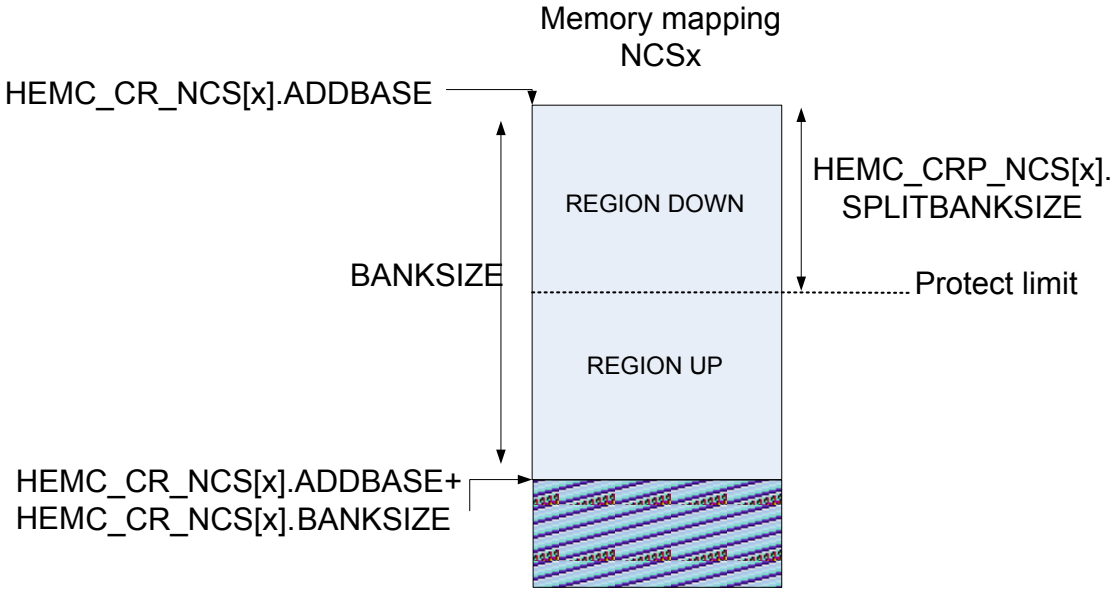
- Read (RD) or/and Write (WR) accesses are allowed or rejected on a part of each NCSx memory area depending on the HEMC NCSx protection applied on this area.
- User access can be allowed or rejected on a part of each NCSx memory area depending on the HEMC NCSx protection applied on this area.

Note: Superuser (or privilege) access is always allowed.

28.6.2.2 Memory Area Region Definition

A region is a part of an NCSx memory area. Two regions, named UP and DOWN in this document, make up one NCSx memory area and these regions are separated by the protect limit, as presented below:

Figure 28-4. Region in NCSx Memory Area Definition



The protect limit is defined by configuring the HEMC_CRP_NCSx.SPLITBANKSIZE. This limit can vary from 8 Kbytes to HEMC_CR_NCSx.BANKSIZE.

Note: If HEMC_CRP_NCSx.SPLITBANKSIZE is equal to HEMC_CR_NCSx.BANKSIZE, the region DOWN covers the entire NCS memory area.

28.6.2.3 Enabling/Disabling Protection

The protection can be enabled/disabled on each NCSx memory area by setting/clearing HEMC_CRP_NCSx.PROTECTON.

28.6.2.3.1 Access Types

The types of access are:

- Read (RD) access is allowed or rejected in a region if the HEMC_CRP_NCSx.RD is set or cleared, respectively.
- Write (WR) access is allowed or rejected in a region if the HEMC_CRP_NCSx.WR is set or cleared, respectively
- User access is allowed or rejected in a region if the HEMC_CRP_NCSx.SUPERUSER is cleared or set, respectively.

Note: Superuser (Privilege) access is always allowed in a region.

Note: HEMC_CRP_NCSx.PROTECTON must be set to enable protection.

28.6.2.3.2 Master Protection

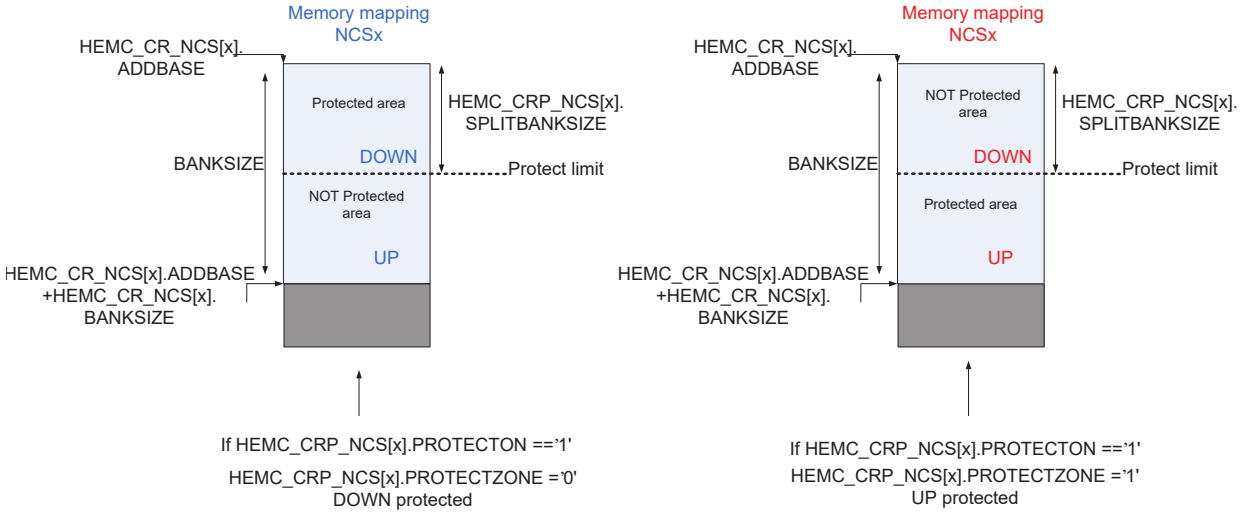
The protection rules described above can be applied to data coming from specific masters. By configuring HEMC_CRP_NCSx.MASTERNUMBER, data coming from masters can be submitted or not to the previous rules.

For the list of masters and their definitions, refer to the section “Bus Matrix”.

28.6.2.4 Region Protection

The protection can be applied on the REGION_DOWN or on the REGION_UP depending on the configuration of HEMC_CRP_NCSx.PROTECTZONE. Refer to the figures below.

Figure 28-5. Protection Region Applicability



28.6.2.5 Error Flags

When an unwanted AHB access from a specific master is done on an NCSx memory-protected region, (previously configured and enabled), the corresponding error flags are raised automatically.

- A flag RDERRORACCESS raises automatically (HEMC_SR. RDERRORACCESS =’1’) in a case of unwanted Read access.
- A flag WRERRORACCESS raises automatically (HEMC_SR. WRERRORACCESS =’1’) in a case of unwanted Write access.
- A flag USERERRORACCESS raises automatically (HEMC_SR. USERERRORACCESS =’1’) in a case of unwanted User access.

Note: Several flags can be raised in case of multiple violations on a protected area (unwanted Read and User access or unwanted Write and User access).

The corresponding interrupt (HEMC_ISR.RDERRORACCESS) is also raised if this interrupt is previously enabled (HEMC_IER. RDERRORACCESS).

The corresponding interrupt (HEMC_ISR. WRERRORACCESS) is also raised if this interrupt is previously enabled (HEMC_IER. WRERRORACCESS).

The corresponding interrupt (HEMC_ISR. USERERRORACCESS) is also raised if this interrupt is previously enabled (HEMC_IER. USERERRORACCESS).

Note: Several interrupts are raised if there are multiple violations on a protected area; for example, in the case of unwanted Read and User access or unwanted Write and User access.

28.6.3 External Control Buffer Signal

An additional signal can be used to control on-board external buffers if required.

The behavior of this signal is as follows:

- WR HSMC access: NOT(EXT_CTRL_BUFFER_DEFAULT_STATE)
- RD HSMC access: EXT_CTRL_BUFFER_DEFAULT_STATE
- WR HSDRAMC access: EXT_CTRL_BUFFER_DEFAULT_STATE
- RD HSDRAMC access: EXT_CTRL_BUFFER_DEFAULT_STATE
- NO access: EXT_CTRL_BUFFER_DEFAULT_STATE

and by default EXT_CTRL_BUFFER_DEFAULT_STATE is '1'.

The EXT_CTRL_BUFFER_DEFAULT_STATE value can be changed by configuring HEMC_CTRL.POL bit field.

- if HEMC_CTRL.POL is '0' then EXT_CTRL_BUFFER_DEFAULT_STATE is '1'.

- if HEMC_CTRL.POL is '1' then EXT_CTRL_BUFFER_DEFAULT_STATE is '0'.

28.7 Register Summary

Offset	Name	Bit Pos.								
0x00	HEMC_CR_NCS0	7:0	ADDBASE[0]	TYPE	BANKSIZE[4:0]					ZERO
		15:8	ADDBASE[8:1]							
		23:16	ADDBASE[16:9]							
		31:24	ECC12_ENABLE	ENABLE	WRITE_ECC_CONF					ADDBASE[17]
0x04	HEMC_CR_NCS1	7:0	ADDBASE[0]	TYPE	BANKSIZE[4:0]					ZERO
		15:8	ADDBASE[8:1]							
		23:16	ADDBASE[16:9]							
		31:24	ECC12_ENABLE	ENABLE						ADDBASE[17]
0x08	HEMC_CR_NCS2	7:0	ADDBASE[0]	TYPE	BANKSIZE[4:0]					ZERO
		15:8	ADDBASE[8:1]							
		23:16	ADDBASE[16:9]							
		31:24	ECC12_ENABLE	ENABLE						ADDBASE[17]
0x0C	HEMC_CR_NCS3	7:0	ADDBASE[0]	TYPE	BANKSIZE[4:0]					ZERO
		15:8	ADDBASE[8:1]							
		23:16	ADDBASE[16:9]							
		31:24	ECC12_ENABLE	ENABLE						ADDBASE[17]
0x10	HEMC_CR_NCS4	7:0	ADDBASE[0]	TYPE	BANKSIZE[4:0]					ZERO
		15:8	ADDBASE[8:1]							
		23:16	ADDBASE[16:9]							
		31:24	ECC12_ENABLE	ENABLE						ADDBASE[17]
0x14	HEMC_CR_NCS5	7:0	ADDBASE[0]	TYPE	BANKSIZE[4:0]					ZERO
		15:8	ADDBASE[8:1]							
		23:16	ADDBASE[16:9]							
		31:24	ECC12_ENABLE	ENABLE						ADDBASE[17]
0x18	HEMC_CTRL	7:0								POL
		15:8								
		23:16								
		31:24								
0x1C ... 0x1F	Reserved									
0x20	HEMC_CRP_NCS0	7:0	MASTERNUMBER[7:0]							
		15:8	MASTERNUMBER[15:8]							
		23:16	SPLITBANKSIZE[4:0]					RD	WR	SUPERUSER
		31:24							PROTECTON	PROTECTZONE
0x24	HEMC_CRP_NCS1	7:0	MASTERNUMBER[7:0]							
		15:8	MASTERNUMBER[15:8]							
		23:16	SPLITBANKSIZE[4:0]					RD	WR	SUPERUSER
		31:24							PROTECTON	PROTECTZONE
0x28	HEMC_CRP_NCS2	7:0	MASTERNUMBER[7:0]							
		15:8	MASTERNUMBER[15:8]							
		23:16	SPLITBANKSIZE[4:0]					RD	WR	SUPERUSER
		31:24							PROTECTON	PROTECTZONE

.....continued

Offset	Name	Bit Pos.								
0x2C	HEMC_CRP_NCS3	7:0	MASTERNUMBER[7:0]							
		15:8	MASTERNUMBER[15:8]							
		23:16	SPLITBANKSIZE[4:0]					RD	WR	SUPERUSER
		31:24							PROTECTON	PROTECTZONE
0x30	HEMC_CRP_NCS4	7:0	MASTERNUMBER[7:0]							
		15:8	MASTERNUMBER[15:8]							
		23:16	SPLITBANKSIZE[4:0]					RD	WR	SUPERUSER
		31:24							PROTECTON	PROTECTZONE
0x34	HEMC_CRP_NCS5	7:0	MASTERNUMBER[7:0]							
		15:8	MASTERNUMBER[15:8]							
		23:16	SPLITBANKSIZE[4:0]					RD	WR	SUPERUSER
		31:24							PROTECTON	PROTECTZONE
0x38	HEMC_IER	7:0				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
		15:8								
		23:16								
		31:24								
0x3C	HEMC_IDR	7:0				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
		15:8								
		23:16								
		31:24								
0x40	HEMC_IMR	7:0				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
		15:8								
		23:16								
		31:24								
0x44	HEMC_ISR	7:0				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
		15:8								
		23:16								
		31:24								
0x48	HEMC_SR	7:0				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
		15:8								
		23:16								
		31:24								

28.7.1 HEMC Control Register Chip Select 0

Name: HEMC_CR_NCS0
Offset: 0x00
Reset: 0x0000001E
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ECC12_ENABLE	ENABLE	WRITE_ECC_CONF					ADDBASE[17]
Access	R/W	R/W	R/W					R/W
Reset	x	x	0					0

Bit	23	22	21	20	19	18	17	16
	ADDBASE[16:9]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDBASE[8:1]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDBASE[0]	TYPE	BANKSIZE[4:0]					ZERO
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	1	1	1	1	0

Bit 31 – ECC12_ENABLE BCH ECC Enable
The reset value depends on CFG[3] value at reset.

Value	Description
0	Hamming(32,7) HECC is enabled if ENABLE is at '1'.
1	BCH(32,12) HECC is enabled if ENABLE is at '1'.

Bit 30 – ENABLE ECC Protection Enable
The reset value depends on CFG[2] value at reset.

Value	Description
0	Protection is deactivated.
1	Protection is activated.

Bit 29 – WRITE_ECC_CONF ECC Configuration Protection Enable

Value	Description
0	Changing the ECC configuration through this register is not permitted; it can only be done by sampling the states of CFG[2] and CFG[3] when Reset is released.
1	Changing the ECC configuration through the register is permitted.

Bits 24:7 – ADDBASE[17:0] Relative Base Address of NCS Area
18-bit MSB address to define the relative base address of the corresponding chip select. (all '1' is the default value, i.e 3_FFFF, meaning that the current area is not configured).

Bit 6 – TYPE Type of Memory Connected

Value	Name	Description
0	SMC	SMC (PROM or SRAM) is connected.
1	SDRAMC	SDRAMC is connected.

Bits 5:1 – BANKSIZE[4:0] Bank Size

Bank size by chip select, from 8 Kbytes to 512 Mbytes. Default is 256 Mbytes.

Value	Name	Description
00000	8KB	8 Kbytes
00001	16KB	16 Kbytes
00010	32KB	32 Kbytes
00011	64KB	64 Kbytes
00100	128KB	128 Kbytes
00101	256KB	256 Kbytes
00110	512KB	512 Kbytes
00111	1MB	1 Mbytes
01000	2MB	2 Mbytes
01001	4MB	4 Mbytes
01010	8MB	8 Mbytes
01011	16MB	16 Mbytes
01100	32MB	32 Mbytes
01101	64MB	64 Mbytes
01110	128MB	128 Mbytes
01111	256MB	256 Mbytes (Default)
10000	512MB	512 Mbytes
11111	NOT_USED	NCS not configured

Bit 0 – ZERO Fixed to 0

Value	Description
0	No continuity with previous block.

28.7.2 HEMC Control Register Chip Select x [x=1..5]

Name: HEMC_CR_NCSx
Offset: 0x04 + (x-1)*0x04 [x=1..5]
Reset: 0x01FFFFBE
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ECC12_ENABLE	ENABLE						ADDBASE[17]
Access	R/W	R/W						R/W
Reset	x	x						1

Bit	23	22	21	20	19	18	17	16
	ADDBASE[16:9]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	ADDBASE[8:1]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	ADDBASE[0]	TYPE	BANKSIZE[4:0]					ZERO
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	1	1	1	1	1	0

Bit 31 – ECC12_ENABLE BCH ECC Enable
The reset value depends on CFG[3] value at reset.

Value	Description
0	Hamming(32,7) HECC is enabled if ENABLE is at '1'.
1	BCH(32,12) HECC is enabled if ENABLE is at '1'.

Bit 30 – ENABLE ECC Protection Enable
The reset value depends on CFG[2] value at reset.

Value	Description
0	Protection is deactivated.
1	Protection is activated.

Bits 24:7 – ADDBASE[17:0] Relative Base Address of NCS Area
18-bit MSB address to define the relative base address of the corresponding chip select. (all '1' is the default value, i.e 3_FFFF, meaning that the current area is not configured).

Bit 6 – TYPE Type of Memory Connected

Value	Name	Description
0	SMC	SMC (PROM or SRAM) is connected.
1	SDRAMC	SDRAMC is connected.

Bits 5:1 – BANKSIZE[4:0] Bank Size
Bank size by chip select, from 8 Kbytes to 512 Mbytes. Default is 256 Mbytes.

Value	Name	Description
00000	8KB	8 Kbytes
00001	16KB	16 Kbytes
00010	32KB	32 Kbytes
00011	64KB	64 Kbytes

Value	Name	Description
00100	128KB	128 Kbytes
00101	256KB	256 Kbytes
00110	512KB	512 Kbytes
00111	1MB	1 Mbytes
01000	2MB	2 Mbytes
01001	4MB	4 Mbytes
01010	8MB	8 Mbytes
01011	16MB	16 Mbytes
01100	32MB	32 Mbytes
01101	64MB	64 Mbytes
01110	128MB	128 Mbytes
01111	256MB	256 Mbytes (Default)
10000	512MB	512 Mbytes
11111	NOT_USED	NCS not configured

Bit 0 – ZERO Fixed to 0

Value	Description
0	No continuity with previous block.

28.7.3 HEMC Polarity Control Register

Name: HEMC_CTRL
Offset: 0x18
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								POL
Access								R/W
Reset								0

Bit 0 – POL External Control Buffer Signal Polarity

Value	Description
0	ext_ctrl_buffer is '1' except for HSMC write cycle, where it is '0'
1	ext_ctrl_buffer is '0' except for HSMC write cycle, where it is '1'.

28.7.4 HEMC Control Register Protection Chip Select

Name: HEMC_CRP_NCSx
Offset: 0x20 + x*0x04 [x=0..5]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
							PROTECTON	PROTECTZONE
Access							R/W	R/W
Reset							0	0

Bit	23	22	21	20	19	18	17	16
	SPLITBANKSIZE[4:0]					RD	WR	SUPERUSER
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MASTERNUMBER[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MASTERNUMBER[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 25 – PROTECTON Protection Activation

Value	Description
0	Protection is OFF.
1	Protection is ON.

Bit 24 – PROTECTZONE Select Area Protected

Value	Description
0	DOWN is protected, UP is not protected.
1	DOWN is not protected, UP is protected.

Bits 23:19 – SPLITBANKSIZE[4:0] Bank Size Internal Separation
0b00000 to 0b10000: from 8 KBytes to 512 Mbytes with the constraint that SPLITBANKSIZE ≤ CR_NCS[x].BANKSIZE and CR_NCS[x].ADDBASE is valid.

Bit 18 – RD Read Access

Value	Description
0	Read access not allowed.
1	Read access allowed.

Bit 17 – WR Write Access

Value	Description
0	Write access not allowed.
1	Write access allowed.

Bit 16 – SUPERUSER User or Superuser Access

Value	Description
0	User and Superuser access allowed.
1	Only Superuser access allowed.

Bits 15:0 – MASTERNUMBER[15:0] Master Number ID
For each master on the AHB (from M0 to M15) (bit0 = M0...bit15 = M15)

Value	Description
0	Memory not protected against the master (no restriction).
1	Memory protected against the master (restriction).

28.7.5 HEMC Interrupt Enable Register

Name: HEMC_IER
Offset: 0x38
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
Access				W	W	W	W	
Reset				–	–	–	–	

Bit 4 – USERERRORACCESS User Access Interrupt Enable

Value	Description
0	No effect.
1	User error access interrupt enabled.

Bit 3 – WRERRORACCESS Write Access Interrupt Enable

Value	Description
0	No effect.
1	Write error access interrupt enabled.

Bit 2 – RDERRORACCESS Read Access Interrupt Enable

Value	Description
0	No effect.
1	Read error access interrupt enabled.

Bit 1 – OUTOFRANGE Out of Range Interrupt Enable

Value	Description
0	No effect.
1	Out of range interrupt enabled.

28.7.6 HEMC Interrupt Disable Register

Name: HEMC_IDR
Offset: 0x3C
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
Access				W	W	W	W	
Reset				–	–	–	–	

Bit 4 – USERERRORACCESS User Access Interrupt Disable

Value	Description
0	No effect.
1	User error access interrupt disabled.

Bit 3 – WRERRORACCESS Write Access Interrupt Disable

Value	Description
0	No effect.
1	Write error access interrupt disabled.

Bit 2 – RDERRORACCESS Read Access Interrupt Disable

Value	Description
0	No effect.
1	Read error access interrupt disabled.

Bit 1 – OUTOFRANGE Out of Range Interrupt Disable

Value	Description
0	No effect.
1	Out of range interrupt disabled.

28.7.7 HEMC Interrupt Mask Register

Name: HEMC_IMR
Offset: 0x40
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
Access				R	R	R	R	
Reset				0	0	0	0	

Bit 4 – USERERRORACCESS User Access Mask Interrupt

Value	Description
0	User error access interrupt disabled.
1	User error access interrupt enabled.

Bit 3 – WRERRORACCESS Write Access Error Mask Interrupt

Value	Description
0	Write error access interrupt disabled.
1	Write error access interrupt enabled.

Bit 2 – RDERRORACCESS Read Access Error Mask Interrupt

Value	Description
0	Read error access interrupt disabled.
1	Read error access interrupt enabled.

Bit 1 – OUTOFRANGE Out of Range Mask Interrupt

Value	Description
0	Out of range interrupt disabled.
1	Out of range interrupt enabled.

28.7.8 HEMC Interrupt Status Register

Name: HEMC_ISR
Offset: 0x44
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
Access				R	R	R	R	
Reset				0	0	0	0	

Bit 4 – USERERRORACCESS User Interrupt Pending

Value	Description
0	User error access interrupt not pending.
1	User error access interrupt pending.

Bit 3 – WRERRORACCESS Write Access Interrupt Pending

Value	Description
0	Write error access interrupt not pending.
1	Write error access interrupt pending.

Bit 2 – RDERRORACCESS Read Access Interrupt Pending

Value	Description
0	Read error access interrupt not pending.
1	Read error access interrupt pending.

Bit 1 – OUTOFRANGE Out of Range Interrupt Pending

Value	Description
0	Out of range interrupt not pending.
1	Out of range interrupt pending.

28.7.9 HEMC Status Register

Name: HEMC_SR
Offset: 0x48
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				USERERRORACCESS	WRERRORACCESS	RDERRORACCESS	OUTOFRANGE	
Access				R	R	R	R	
Reset				0	0	0	0	

Bit 4 – USERERRORACCESS User or Superuser Error Access

Value	Description
0	User error access not detected.
1	User error access detected.

Bit 3 – WRERRORACCESS Write Access Error

Value	Description
0	Write error access not detected.
1	Write error access detected.

Bit 2 – RDERRORACCESS Read Access Error

Value	Description
0	Read error access not detected.
1	Read error access detected.

Bit 1 – OUTOFRANGE Out of Range

Value	Description
0	Out of range not detected.
1	Out of range detected.

29. Hardened Error Correction Code (HECC) Controller

29.1 Description

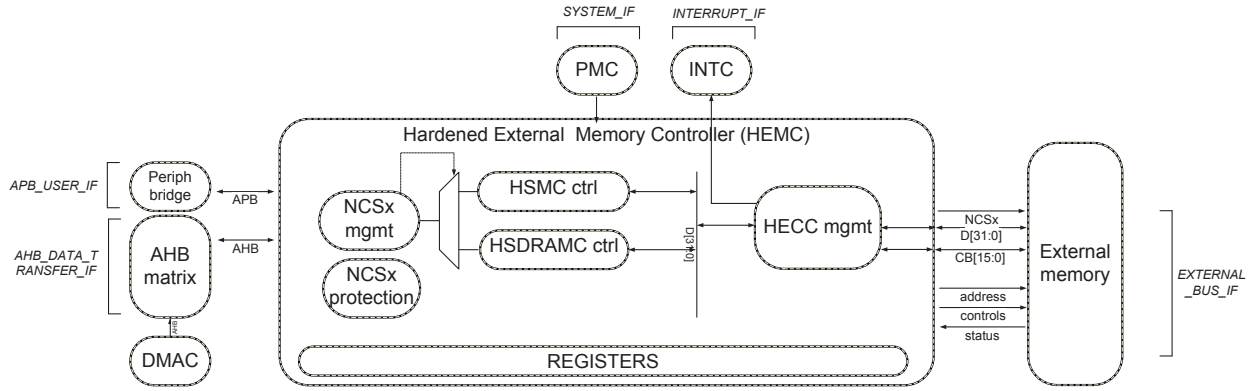
The Hardened Error Correction Code (HECC) Controller protects data against the effects of radiation. It is embedded in the Hardened External Memory Controller (HEMC).

29.2 Embedded Characteristics

- Three Channels Depending on the Memory Type and the Access Used
- Embedded Hamming (32,7) Code to Correct One Error and Detect up to Two Errors
- Embedded BCH (32,12) Code to Correct up to Two Errors and to Detect Three or More Errors
- Interrupt Capabilities
  - One interrupt dedicated to fixable errors
  - One interrupt dedicated to unfixable errors
- Error Counter Available for Statistical Reporting
  - One counter dedicated to fixable errors
  - One counter dedicated to unfixable errors
- Embedded HECC Testing Mode

29.3 Block Diagram

Figure 29-1. HECC Block Diagram



29.4 Functional Description

29.4.1 Overview

The HECC comprises:

- Two encoder/decoders (Hamming (32,7) or BCH(32,12)), end user selectable
- An APB end user interface used to configure:
  - the HECC protection
  - the encoder/decoder type used: (Hamming (32,7) or BCH(32,12))
  - the HECC test mode (independently in read or write mode)
  - the reset of the HECC fixable and unfixable counters
- An APB end user interface for errors status and reporting:

- Indicates whether fixable errors have been detected and corrected on-the-fly, and quantifies them
- Indicates whether unfixable errors have been detected and quantifies them
- The indication can be done either by interruption or by polling
- Indicates the address and the status (type, size and origin) of the last HECC error detected

29.4.2 Operation

Three channels are implemented in the HECC module. Each channel is dedicated to a memory type and/or the activation of the read-modify-write (RMW) feature.

- Channel 0 is used when accesses to the HSMC are performed.
- Channel 1 and 2 are used when accesses to the HSDRAMC are performed.

Note: When using HSDRAMC, it is mandatory to configure channels 1 and 2.

ECC protection is individually enabled or disable for each Chip Select by setting or clearing the HEMC_CR_NCSx.ENABLE bit (0≤x≤5). When enabled, the HECC operates at each access to the memories. However, the HECC detection and correction is performed only on read/fetch accesses, by comparing the HECC checksum stored in memory to the one locally computed from data read.

Note: Upon reset, ECC protection performed on the NCS0 memory area is configured from the value read on external pins CFG2 and CFG3. By driving CFG2 high (resp. low) during Reset, it is possible to enable (resp. disable) the HECC protection. By driving CFG3 high (resp. low), it is possible to select the appropriate encoder/decoder Hamming(32,7) (resp. BCH(32,12)).
The configuration can be changed through the HEMC_CR_NCS0 register by setting both the HEMC_CR_NCS0.ECC12_ENABLE and HEMC_CR_NCS0.ENABLE bits; however, this must be enabled by first setting HEMC_CR_NCS0.WRITE_ECC_CONF.

ECC protection performed on the other memory areas (NCSx, 1≤x≤5) is configured with the HEMC_CR_NCSx.ECC12_ENABLE and HEMC_CR_NCSx.ENABLE bits in the corresponding HEMC_CR_NCSx registers.

Note: When the HECC is enabled on the HEMC area, the read-modify-write feature must also be enabled so that the integrity of the HECC checkbits is preserved on sub-word writes.

29.4.3 Interrupt Sources

Using the HECC interrupt requires the interrupt controller to be programmed first.

29.5 Hamming (32, 7) Code

For each word, a 7-bit checksum is generated. With this correction code, errors of one bit can be corrected (fixable error) and errors of up to two bits (unfixable errors) can be detected.

29.5.1 Write Access

When the processor performs a write access to an HECC protected memory, it also writes the 7-bit HECC checksum on the HECC protected memory.

Note: CB[7] is always driven low unless HECC testing is enabled.

29.5.2 Read Access

When the processor performs a read access to an HECC protected memory, it reads data and the associated checksum. The checksum read from the HECC protected memory is compared against the checksum generated by the HECC from the same read data. Any discrepancy yields an error and a syndrome is computed to further qualify the error as a fixable error or unfixable error.

29.5.3 Fixable Error

The correction of a fixable error is performed on-the-fly inside the processor during the current access and no timing penalty is incurred.

The fixable error event is reported in the Status register (HEMC_HECC_SR) and the address of the corresponding error is reported in the Fail Address register (HEMC_HECC_FAILAR). If previously enabled, a fixable interrupt is generated.

The fixable error remains in memory until a software-initiated rewrite is performed at the faulty memory location.

To track the number of fixable errors, a fixable error counter is available. It is incremented each time a fixable error is detected and until a counter overload is detected.

It is possible for the end user to reset the fixable counter using RST_FIX_CPT in HEMC_HECC_CR.

29.5.4 Unfixable Error

The unfixable error event is reported in the Status register (HEMC_HECC_SR) and the address of the corresponding error is reported in the Fail Address register (HEMC_HECC_FAILAR). If previously enabled, an unfixable interrupt is generated.

To track the number of unfixable errors, an unfixable error counter is available. It is incremented each time an unfixable error is detected and until a counter overload is detected.

It is possible for the end user to reset the unfixable counter using RST_NOFIX_CPT in HEMC_HECC_CRx.

29.5.5 Hamming(32,7) on 8-bit Memories

When Hamming(32,7) is used, protection is performed as follows on 8-bit memories:

- D[7:0] is used to store/load 32-bit data or fetch instruction in a burst of four steps.
- D[7:0] is used to store/load checksum with an additional step.

29.5.5.1 Memory Repartition

The protected memory bank is partitioned as follows:

- 80% of the memory bank is available as program or data memory
- 20% of the memory bank is allocated to the HECC checksum

29.5.5.2 HECC Checkbits Address Calculation

The HECC checksum address is defined by the following equation:

@ecc=@bottom_memory –[@data - @ top_memory]/4

where

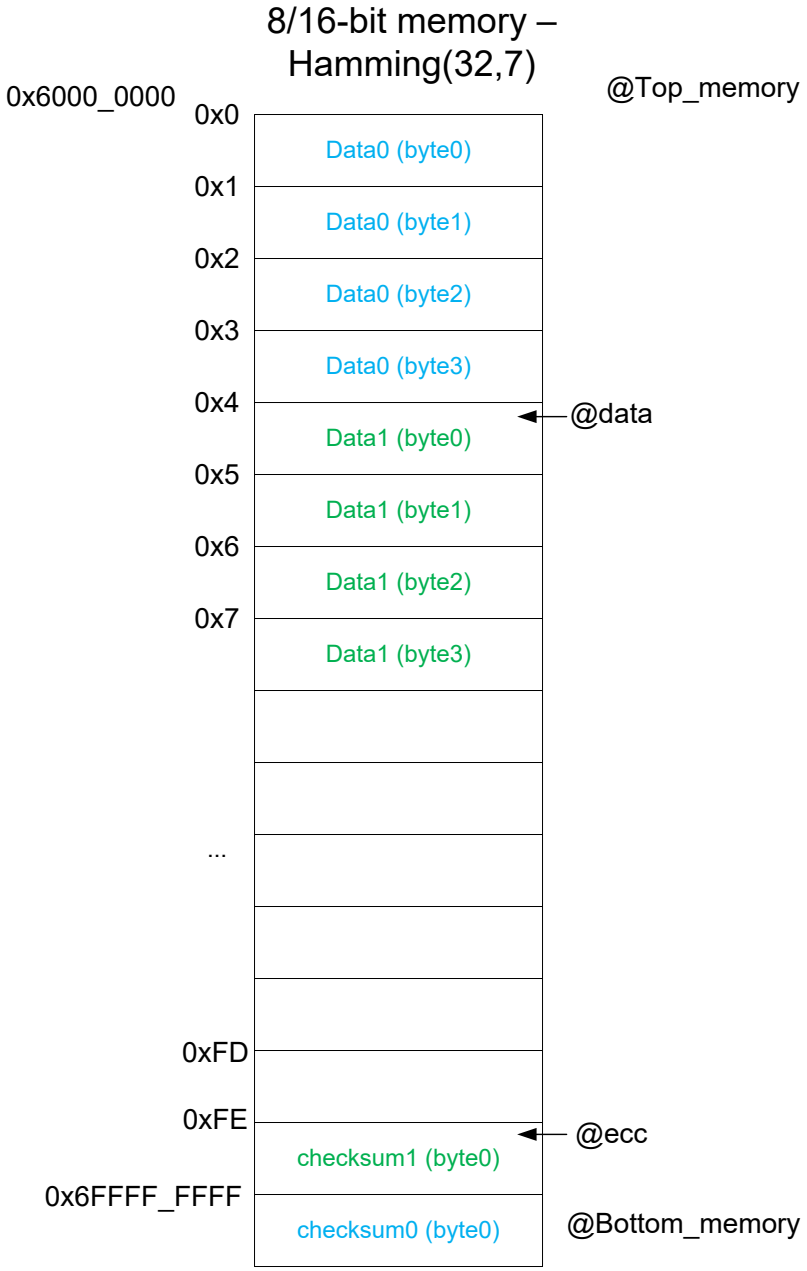
@ecc=address of HECC checksum

@bottom_memory = higher address cover by the memory range

@top_memory = lower address cover by the memory range

@data = address of data written

Figure 29-2. Hamming(32,7) Protection on 8-/16-bit Memory Organization (example with a memory of 256 MBytes)



29.5.6 Hamming(32,7) on 16-bit Memories

When Hamming(32,7) is used, protection is performed as follows on 16-bit memories:

- D[15:0] is used to store/load 32-bit data or fetch instruction in a burst of two steps.
- D[15:0] is used to store/load checksum with an additional step.

29.5.6.1 Memory Repartition

As shown in the figure above, the 16-bit memory is used as follows:

- 80% of the memory bank is available as program or data memory
- 20% of the memory bank is allocated to the HECC checksum

29.5.6.2 HECC Checkbits Address Calculation

The HECC checksum address is defined by the following equation:

@ecc=@bottom_memory –[@data - @ top_memory]/4

where

@ecc=address of HECC checksum

@bottom_memory = higher address cover by the memory range

@top_memory = lower address cover by the memory range

@data = address of data written

29.5.7 Hamming(32,7) on 32-bit Memories

When Hamming(32,7) is used, protection is performed as follows on 32-bit memories:

- D[31:0] is used to store/load 32-bit data or fetch instruction , in a single step
- CB[7:0] is used to store/load checksum in parallel of D[31:0]

29.5.7.1 Memory Repartition

The 32-bit memory is used as follows:

- 100% of the memory bank is available as program or data memory

29.5.7.2 HECC Checkbits Address Calculation

In this case, the HECC address is the same as the data address.

29.6 BCH (32,12) Code

For each word, a 12-bit checksum is generated. With this correcting code, an error of up to two bits, or fixable error, can be corrected. Errors of three bits or more, or unfixable errors, can be detected.

29.6.1 Write Access

When the processor performs a write access to an HECC protected memory, it also writes the 12-bit HECC checksum on the HECC protected memory.

Note: CB[15-12] is always driven low unless HECC testing is enabled.

29.6.2 Read Access

When the processor performs a read access to an HECC protected memory, it reads data and the associated checksum. The checksum read from the HECC protected memory is compared against the checksum generated by the HECC from the same read data. Any discrepancy yields an error and a syndrome is computed to further qualify the error as a fixable error or unfixable error.

29.6.3 Fixable Error

The correction of a fixable error is performed on-the-fly inside the processor during the current access and no timing penalty is incurred.

The fixable error event is reported in the Status register (HEMC_HECC_SR) and the address of the corresponding error is reported in the Fail Address register (HEMC_HECC_FAILAR). If previously enabled, a fixable interrupt is generated.

The fixable error remains in memory until a software-initiated rewrite is performed at the faulty memory location.

To track the number of fixable errors, a fixable error counter is available. It is incremented each time a fixable error is detected and until a counter overload is detected.

It is possible for the end user to reset the fixable counter using RST_FIX_CPT in HEMC_HECC_CR.

29.6.4 Unfixable Error

The unfixable error event is reported in the Status register (HEMC_HECC_SR) and the address of the corresponding error is reported in the Fail Address register (HEMC_HECC_FAILAR). If previously enabled, an unfixable interrupt is generated.

To track the number of unfixable errors, an unfixable error counter is available. It is incremented each time an unfixable error is detected and until a counter overload is detected.

It is possible for the end user to reset the unfixable counter using RST_NOFIX_CPT in HEMC_HECC_CRx.

29.6.5 BCH(32,12) on 8-bit Memories

When BCH(32,12) is used, protection is performed as follows on an 8-bit memory:

- D[7:0] is used to store/load 32-bit data or fetch instruction in a burst of four steps.
- D[7:0] is used to store/load checksum with an additional burst of two steps.

29.6.5.1 Memory Repartition

The protected memory bank is partitioned as follows:

- 66% of the memory bank is available as program or data memory
- 33% of the memory bank is allocated to the HECC checksum

29.6.5.2 HECC Checkbits Address Calculation

The HECC checksum address is defined by the following equation:

@ecc=@bottom_memory –([@data - @ top_memory]/2+1)

where

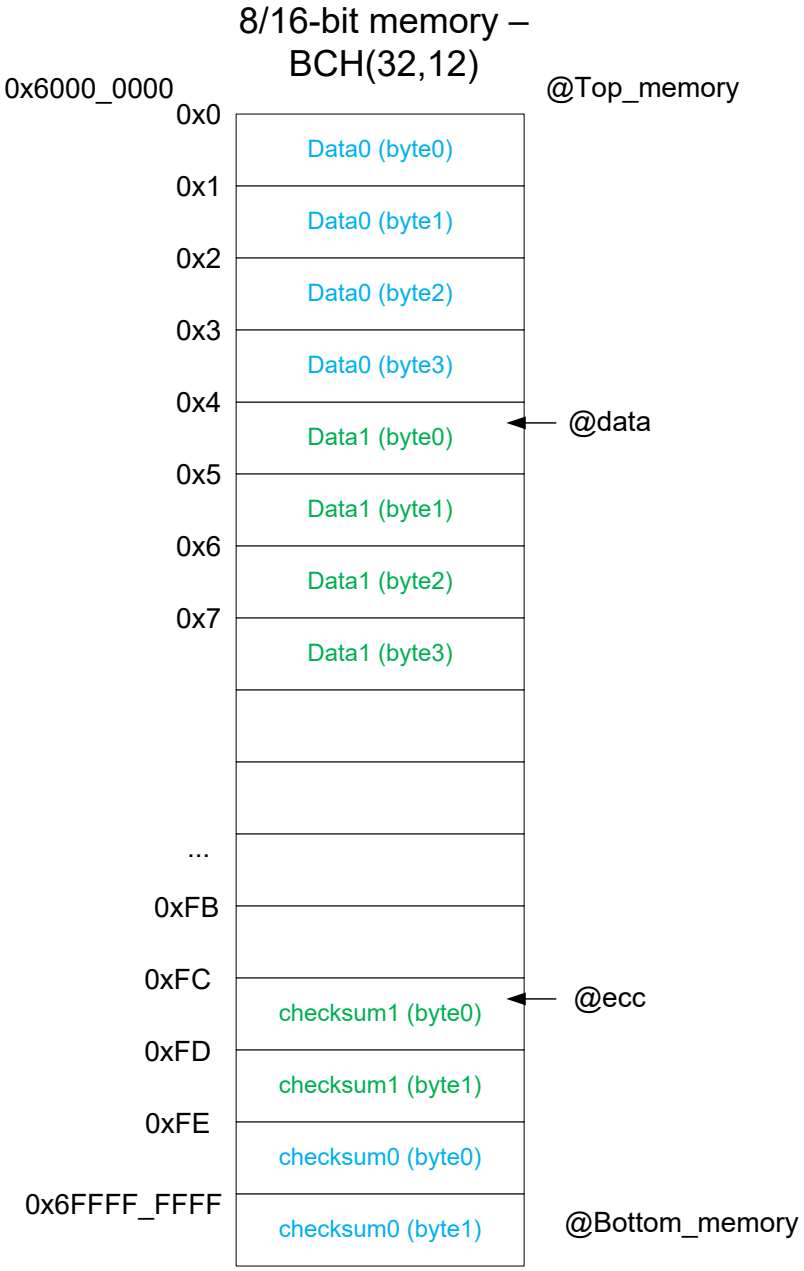
@ecc=address of HECC checksum

@bottom_memory = higher address cover by the memory range

@top_memory = lower address cover by the memory range

@data = address of data written

Figure 29-3. BCH(32,12) Protection on 8-/16-bit Memory Organization (example with a memory of 256 MBytes)



29.6.6 BCH(32,12) on 16-bit Memories

When BCH(32,12) is used, protection is performed as follows on 16-bit memory:

- D[15:0] is used to store/load 32-bit data or fetch instruction in a burst of two steps.
- D[15:0] is used to store/load checksum with one additional step.

29.6.6.1 Memory Repartition

As shown in the figure above, the protected memory bank is partitioned as follows:

- 66% of the memory bank is available as program or data memory
- 33% of the memory bank is allocated to the HECC checksum

29.6.6.2 HECC Checkbits Address Calculation

The HECC checksum address is defined by the following equation:

@ecc=@bottom_memory –([@data - @ top_memory]/2+1)

where

@ecc=address of HECC checksum

@bottom_memory = higher address cover by the memory range

@top_memory = lower address cover by the memory range

@data = address of data written

29.6.7 BCH(32,12) on 32-bit Memories

When BCH(32,12) is used, protection is performed as follows on 32-bit memory:

- D[31:0] is used to store/load 32-bit data or fetch instruction , in a single step
- CB[15:0] is used to store/load the checksum in parallel of D[31:0]

29.6.7.1 Memory Repartition

Therefore, the 32-bit memory is used as follows:

- 100% of the memory bank is available as program or data memory

29.6.7.2 HECC Checkbits Address Calculation

In this case, the HECC address is the same as the data address.

29.7 HECC Testing

Operation of the HECC can be bypassed for testing purposes and is controlled in the Control register (HEMC_HECC_CRx, x = 0..2). Each channel has its own test mode activation and set up.

29.7.1 Write Test

When HECC write bypass is enabled (HEMC_HECC_CRx.TEST_MODE_WR=1), the test checksum (HEMC_HECC_TESTCBx.TCB) replaces the HECC generated checksum during a data store on the corresponding channel.

29.7.2 Read Test

When the HECC read diagnostic is enabled (HEMC_HECC_CRx.TEST_MODE_RD=1), the test checksum (HEMC_HECC_TESTCBx.TCB) is updated with the read checksum during a data load or an instruction fetch on the corresponding channel.

Note: The HECC test routine must be executed entirely from the instruction cache (when activated) or from an area without the HECC activated and different from the one being accessed. Otherwise, the checksum read during instruction fetch will overwrite those from the area to be tested.

29.8 HECC Restrictions

When HECC is used to protect HSMC accesses, either the BCH or Hamming code may be selected.

When HECC is used to protect HSDRAMC accesses, the Hamming code may be selected.

As a consequence:

- Hamming or BCH code may be used for channel 0.
- Only Hamming code may be used for channel 1.
- Only Hamming code may be used for channel 2.

29.9 Register Summary

Offset	Name	Bit Pos.								
0x00 ... 0xFF	Reserved									
0x0100	HEMC_HECC_CR0	7:0			RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	
		15:8								
		23:16								
		31:24								
0x0104	HEMC_HECC_CR1	7:0			RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	
		15:8								
		23:16								
		31:24								
0x0108	HEMC_HECC_CR2	7:0			RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	
		15:8								
		23:16								
		31:24								
0x010C ... 0x013F	Reserved									
0x0140	HEMC_HECC_TESTCB0	7:0	TCB1[7:0]							
		15:8	TCB1[15:8]							
		23:16								
		31:24								
0x0144	HEMC_HECC_TESTCB1	7:0	TCB1[7:0]							
		15:8	TCB1[15:8]							
		23:16								
		31:24								
0x0148	HEMC_HECC_TESTCB2	7:0	TCB1[7:0]							
		15:8	TCB1[15:8]							
		23:16								
		31:24								
0x014C ... 0x017F	Reserved									
0x0180	HEMC_HECC_SR	7:0	OVER_FIX	CPT_FIX[4:0]						MEM_FIX
		15:8	OVER_NOFIX	CPT_NOFIX[4:0]						MEM_NOFIX
		23:16								
		31:24					TYPE	HES[2:0]		
0x0184	HEMC_HECC_IER	7:0							MEM_NOFIX	MEM_FIX
		15:8								
		23:16								
		31:24								
0x0188	HEMC_HECC_IDR	7:0							MEM_NOFIX	MEM_FIX
		15:8								
		23:16								
		31:24								
0x018C	HEMC_HECC_IMR	7:0							MEM_NOFIX	MEM_FIX
		15:8								
		23:16								
		31:24								
0x0190	HEMC_HECC_FAILAR	7:0	ADDRESS[7:0]							
		15:8	ADDRESS[15:8]							
		23:16	ADDRESS[23:16]							
		31:24	ADDRESS[31:24]							

29.9.1 HECC Control Register x

Name: HEMC_HECC_CRx
Offset: 0x0100 + x*0x04 [x=0..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			RST_NOFIX_CPT	RST_FIX_CPT		TEST_MODE_WR	TEST_MODE_RD	
Access			R/W	R/W		R/W	R/W	
Reset			0	0		0	0	

Bit 5 – RST_NOFIX_CPT Reset the Unfixable Error Counter
This bit automatically returns to ’0’ once the operation is done.

Value	Description
0	No effect.
1	Resets the ECC unfixable error counter.

Bit 4 – RST_FIX_CPT Reset the Fixable Error Counter
This bit automatically returns to ’0’ once the operation is done.

Value	Description
0	No effect.
1	Resets the ECC fixable error counter.

Bit 2 – TEST_MODE_WR Test Mode of ECC Protection - Write Mode
The ENABLE bit must be enabled to use TEST_MODE_WR.

Value	Description
0	Write test mode is deactivated.
1	Write test mode is activated.

Bit 1 – TEST_MODE_RD Test Mode of ECC Protection - Read Mode
The ENABLE bit must be enabled to use TEST_MODE_RD.

Value	Description
0	Read test mode is deactivated.
1	Read test mode is activated.

29.9.2 HECC Test Mode Register _x

Name: HEMC_HECC_TESTCBx
Offset: 0x0140 + x*0x04 [x=0..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TCB1[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TCB1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – TCB1[15:0] Test Check Bit (8- or 16-bit)
Check bit for test purposes.
When Hamming(32,7) is selected, only TCB1[7:0] is used.
When BCH(32,12) is selected, TCB1[15:0] is used.

29.9.3 HECC Status Register

Name: HEMC_HECC_SR
Offset: 0x180
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
					TYPE	HES[2:0]		
Access					R	R	R	R
Reset					0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	OVER_NOFIX	CPT_NOFIX[4:0]						MEM_NOFIX
Access	R	R	R	R	R	R		R
Reset	0	0	0	0	0	0		0

Bit	7	6	5	4	3	2	1	0
	OVER_FIX	CPT_FIX[4:0]						MEM_FIX
Access	R	R	R	R	R	R		R
Reset	0	0	0	0	0	0		0

Bit 27 – TYPE Write or Read Access

Value	Description
0	Write access.
1	Read access.

Bits 26:24 – HES[2:0] Hardware Error Size
Identifies the size of the failed access.

Bit 15 – OVER_NOFIX Counter Overflow

Value	Description
0	No overflow has occurred since the last read on unfixable counter.
1	An overflow has occurred since the last read on unfixable counter.

Bits 14:10 – CPT_NOFIX[4:0] 5-bit Counter
Counts the number of unfixable errors detected for one type of memory during the mission.

Bit 8 – MEM_NOFIX Unfixable Error Status

Value	Description
0	No unfixable error detected.
1	An unfixable error has been detected.

Bit 7 – OVER_FIX Counter Overflow

Value	Description
0	No overflow has occurred since the last read.
1	An overflow has occurred since the last read.

Bits 6:2 – CPT_FIX[4:0] 5-bit Counter
Counts the number of fixable errors detected for one type of memory during the mission.

Bit 0 – MEM_FIX Fixable Error Status
A fixable error has been detected.

Value	Description
0	No fixable error detected.
1	A fixable error has been detected.

29.9.4 HECC Interrupt Enable Register

Name: HEMC_HECC_IER
Offset: 0x184
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							MEM_NOFIX	MEM_FIX
Access							W	W
Reset							–	–

Bit 1 – MEM_NOFIX Unfixable Error

Bit 0 – MEM_FIX Fixable Error

29.9.5 HECC Interrupt Disable Register

Name: HEMC_HECC_IDR
Offset: 0x188
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							MEM_NOFIX	MEM_FIX
Access							W	W
Reset							–	–

Bit 1 – MEM_NOFIX Unfixable Error

Bit 0 – MEM_FIX Fixable Error

29.9.6 HECC Interrupt Mask Register

Name: HEMC_HECC_IMR
Offset: 0x18C
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							MEM_NOFIX	MEM_FIX
Access							R	R
Reset							0	0

Bit 1 – MEM_NOFIX Unfixable Error

Bit 0 – MEM_FIX Fixable Error

29.9.7 HECC Fail Address Register

Name: HEMC_HECC_FAILAR
Offset: 0x190
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ADDRESS[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDRESS[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDRESS[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDRESS[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDRESS[31:0] Address of the Error Detected
The address of the faulty data detected when a fixable or unfixable error occurs.

30. Hardened Static Memory Controller (HSMC)

30.1 Description

The Hardened External Memory Controller (HEMC) is designed to ensure the successful data transfer between several external devices and the ARM-based microcontroller. The Hardened Static Memory Controller (HSMC) is part of the HEMC.

The HSMC handles several types of external memory and peripheral devices, such as SRAM, PROM, EEPROM, LCD Module, NOR Flash.

The HSMC generates the signals that control the access to the external memory devices or peripheral devices. It has 6 chip selects, a 28-bit address bus and a configurable 8-, 16- or 32-bit data bus. Moreover, HECC can be applied on HSMC, leading to 40/48 bits of data. Separate read and write control signals allow for direct memory and peripheral interfacing. Read and write signal waveforms are fully adjustable.

The HSMC can manage wait requests from external devices to extend the current access.

30.2 Embedded Characteristics

- Customizable Address Space per Chip Select (from 8 Kbytes to 256 Mbytes)
- 8-bit, 16- or 32-bit Data Bus + 8- or 16-bit Additonal CB Bus
- Word, Halfword, Byte Transfers
- Byte Write Lines
- Programmable Setup, Pulse And Hold Time for Read Signals per Chip Select
- Programmable Setup, Pulse And Hold Time for Write Signals per Chip Select
- External Wait Request

30.3 I/O Lines Description

Table 30-1. I/O Lines Description

Signal Name	Signal Direction	Signal Width	SignalPolarity	Signal Description
Address				
A[27:0]	Out	28		Address Bit
Data				
D[31:0]	In/Out	32		Data Bits
CB[15/7:0]	In/Out	16/8		Check Bits
Controls/Status				
NWAIT	Out	1	Low	External Wait Signal
NCS[5:0]	Out	6	Low	Common Chip Select Lines Signals
NWR[4:0]	Out	5	Low	Write Byte Enable Signals
NWE	Out	1	Low	Write Enable Signal
NRD	Out	1	Low	Read Signal
EXT_CTRL_BUFFER	Out	1	High	External Buffer Control Signal

30.4 Product Dependencies

30.4.1 I/O Lines

The pins used for interfacing the HSMC are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the HSMC pins to their peripheral function. If I/O lines of the HSMC are not used by the application, they can be used for other purposes by the PIO Controller.

30.4.2 Power Management

The HSMC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the HSMC clock via the HEMC clock.

30.5 Connection to External Devices

30.5.1 Data Bus Width

A data bus width of 8, 16 or 32 bits can be selected for each chip select. This option is controlled by the bit DBW in the Mode register (HSMC_MODE) for the corresponding chip select.

30.5.2 Read & Write Access: Byte Access Supported

During a read access, all 32 bits are read and controlled by the read signal (NRD).

During a write access, all 32 bits or a part these bits (byte0, byte1, byte2 and/or byte3) can be written by using the following signals:

- NWR[4:0] signals are used to access one byte or several bytes in write mode.
  - NWR[0] controls D[0:7]
  - NWR[1] controls D[8:15]
  - NWR[2] controls D[16:23]
  - NWR[3] controls D[24:31]
    Note: NWR[4] has the same behavior as NWR[0].
- NWE signal is used to access the whole word in write mode

30.6 Read, Write and Read-Modify-Write Single Access

In the following sections:

- NWE represents either the NWE signal or one of the byte write lines (NWR0 to NWR4). NWR0 to NWR4 have the same timings and protocol as NWE.
- D[X:0] represents D[31:0] or D[15:0], or D[7:0], depending on data bus width selected. They all have the same timings.

Moreover, if HECC is activated in 32-bit mode:

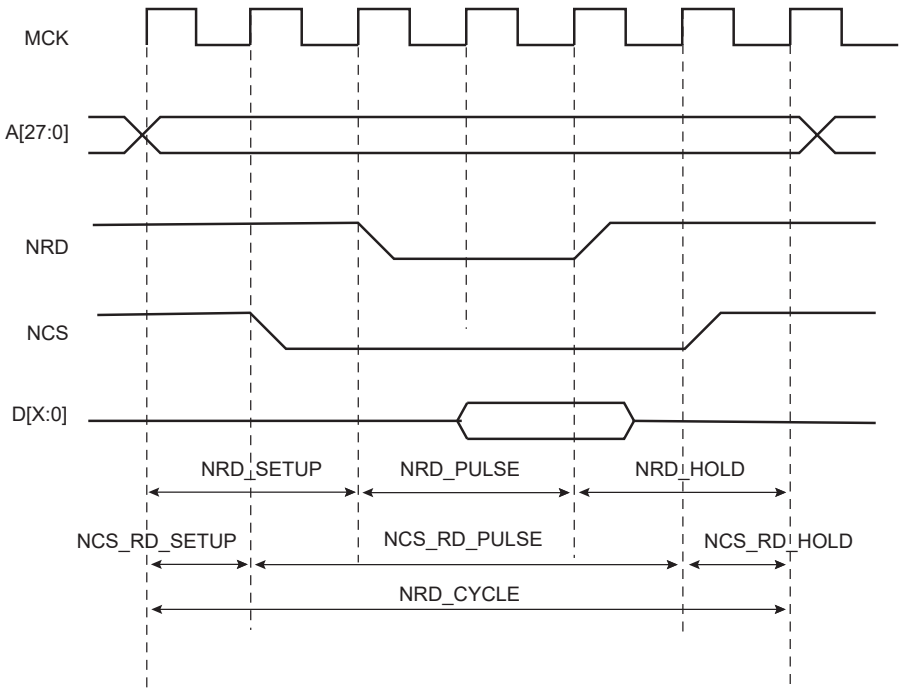
- CB[Y:0] represents CB[15:0] or CB[7:0], depending on the HECC selected. It is not represented in the following figures, but CB[Y:0] has the same timing as D[X:0].
- Similarly, NCSx represents one of the NCS[0..5] chip select lines.

30.6.1 Read Waveforms

The read cycle is shown in the following figure.

The read cycle starts with the address setting on the memory address bus.

Figure 30-1. Standard Read Cycle



30.6.1.1 NRD Waveform

The NRD signal is characterized by a setup timing, pulse width, and hold timing.

- nrd_setup — NRD setup time is defined as the setup of address before the NRD falling edge.
- nrd_pulse — NRD pulse length is the time between NRD falling edge and NRD rising edge.
- nrd_hold — NRD hold time is defined as the hold time of address after the NRD rising edge.

30.6.1.2 NCS Waveform

The NCS signal can be divided into a setup time, pulse length, and hold time:

- ncs_rd_setup — NCS setup time is defined as the setup time of address before the NCS falling edge.
- ncs_rd_pulse — NCS pulse length is the time between NCS falling edge and NCS rising edge.
- ncs_rd_hold — NCS hold time is defined as the hold time of address after the NCS rising edge.

30.6.2 Read Cycle

The NRD_CYCLE time is defined as the total duration of the read cycle, i.e., from the time where address is set on the address bus to the point where address may change. The total read cycle time is defined as:

NRD_CYCLE = NRD_SETUP + NRD_PULSE + NRD_HOLD,

as well as

NRD_CYCLE = NCS_RD_SETUP + NCS_RD_PULSE + NCS_RD_HOLD

All NRD and NCS timings are defined separately for each chip select as an integer number of Master Clock cycles. The NRD_CYCLE field is common to both the NRD and NCS signals, thus the timing period is of the same duration.

NRD_CYCLE, NRD_SETUP, and NRD_PULSE implicitly define the NRD_HOLD value as:

NRD_HOLD = NRD_CYCLE - NRD SETUP - NRD PULSE

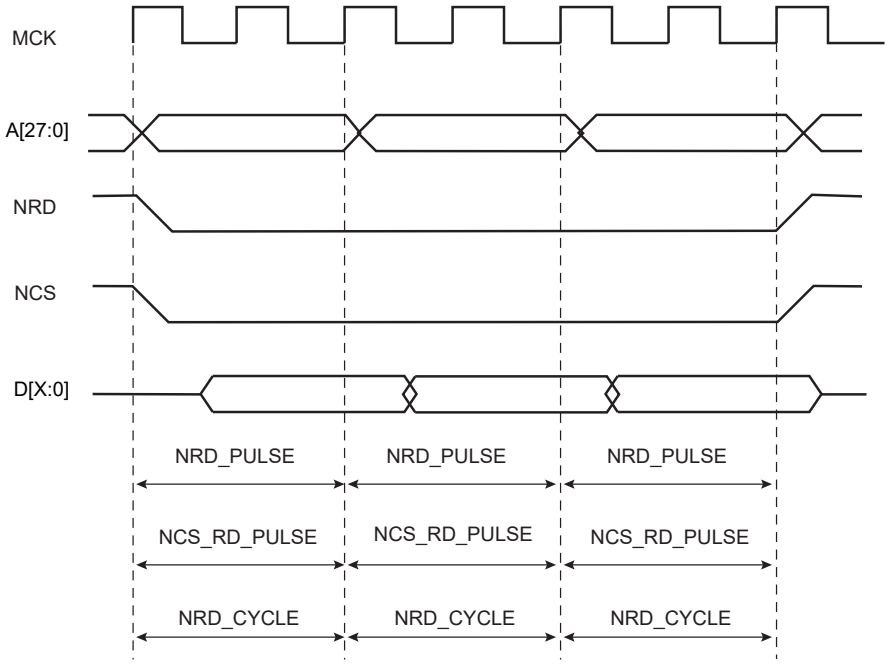
NRD_CYCLE, NCS_RD_SETUP, and NCS_RD_PULSE implicitly define the NCS_RD_HOLD value as:

NCS_RD_HOLD = NRD_CYCLE - NCS_RD_SETUP - NCS_RD_PULSE

30.6.3 Null Delay Setup and Hold

If null setup and hold parameters are programmed for NRD and/or NCS, NRD and NCS remain active continuously in case of consecutive read cycles in the same memory (see the following figure).

Figure 30-2. No Setup, No Hold on NRD and NCS Read Signals



30.6.4 Null Pulse

Programming a null pulse is not permitted. The pulse must be at least set to 1. A null value leads to unpredictable behavior.

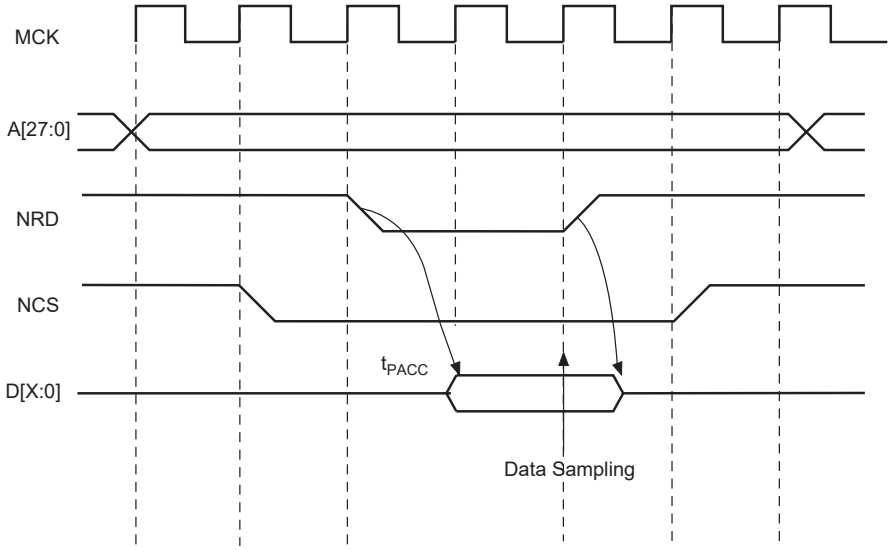
30.6.5 Read Mode

As NCS and NRD waveforms are defined independently of one other, the HSMC needs to know when the read data is available on the data bus. The HSMC does not compare NCS and NRD timings to know which signal rises first. The READ_MODE bit in the HSMC_MODE register of the corresponding chip select indicates which signal of NRD and NCS controls the read operation.

30.6.5.1 Read is Controlled by NRD (HSMC_MODE.READ_MODE = 1):

The following figure shows the waveforms of a read operation of a typical asynchronous RAM. The read data is available t_{PACC} after the falling edge of NRD, and turns to 'Z' after the rising edge of NRD. In this case, HSMC_MODE.READ_MODE must be set to 1 (read is controlled by NRD), to indicate that data is available with the rising edge of NRD. The HSMC samples the read data internally on the rising edge of Master Clock that generates the rising edge of NRD, whatever the programmed waveform of NCS may be.

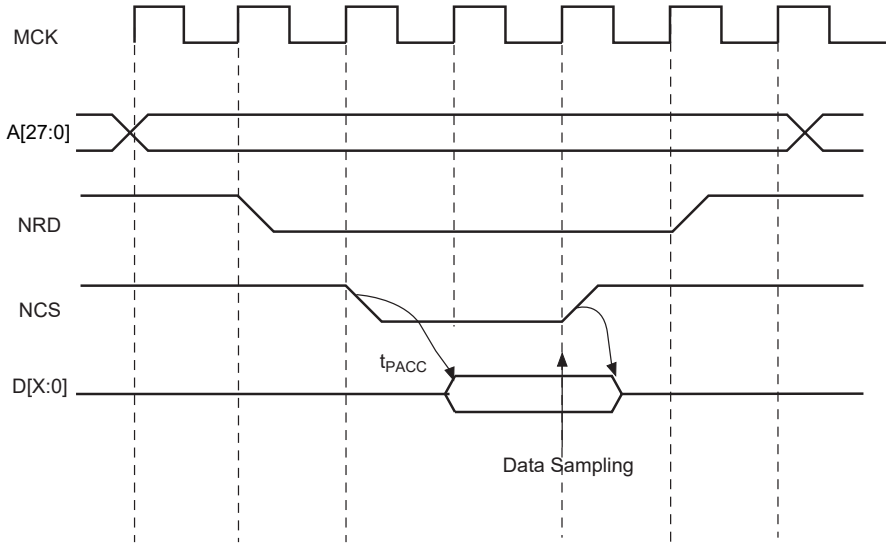
Figure 30-3. HSMC_MODE.READ_MODE = 1: Data is sampled by HSMC before the rising edge of NRD



30.6.5.2 Read is Controlled by NCS (HSMC_MODE.READ_MODE = 0)

The following figure shows the typical read cycle of an LCD module. The read data is valid t_{PACC} after the falling edge of the NCS signal and remains valid until the rising edge of NCS. Data must be sampled when NCS is raised. In this case, the HSMC_MODE.READ_MODE must be set to 0 (read is controlled by NCS): the HSMC internally samples the data on the rising edge of Master Clock that generates the rising edge of NCS, whatever the programmed waveform of NRD may be.

Figure 30-4. HSMC_MODE.READ_MODE = 0: Data is Sampled by HSMC Before the Rising Edge of NCS

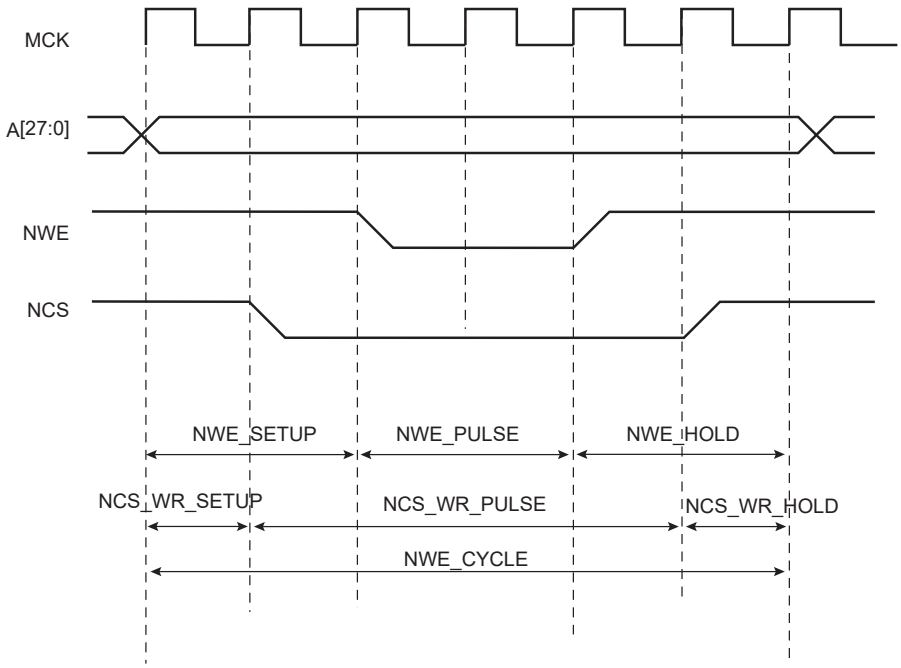


30.6.6 Write Waveforms

The write protocol is similar to the read protocol. It is depicted in the figure below. The write cycle starts with the address setting on the memory address bus.

- In 32-bit mode, the single access presented below is referenced ‘1 x WR 32 bits’ in this document.
- In 16-bit mode, the single access presented below is referenced ‘1 x WR 16 bits’ in this document.
- In 8-bit mode, the single access presented below is referenced ‘1 x WR 8 bits’ in this document.

Figure 30-5. Write Cycle



30.6.6.1 NWE Waveforms

The NWE signal is characterized by a setup timing, a pulse width and a hold timing.

- NWE_SETUP—the NWE setup time is defined as the setup of address and data before the NWE falling edge;
- NWE_PULSE—the NWE pulse length is the time between NWE falling edge and NWE rising edge;
- NWE_HOLD—the NWE hold time is defined as the hold time of address and data after the NWE rising edge.

30.6.6.2 NCS Waveforms

The NCS signal waveforms in write operation are not the same that those applied in read operations, but are separately defined:

- ncs_wr_setup—the NCS setup time is defined as the setup time of address before the NCS falling edge.
- ncs_wr_pulse—the NCS pulse length is the time between NCS falling edge and NCS rising edge;
- ncs_wr_hold—the NCS hold time is defined as the hold time of address after the NCS rising edge.

30.6.7 Write Cycle

The write_cycle time is defined as the total duration of the write cycle; that is, from the time where address is set on the address bus to the point where address may change. The total write cycle time is defined as:

NWE_CYCLE = NWE_SETUP + NWE_PULSE + NWE_HOLD,

as well as

NWE_CYCLE = NCS_WR_SETUP + NCS_WR_PULSE + NCS_WR_HOLD

All NWE and NCS (write) timings are defined separately for each chip select as an integer number of Master Clock cycles. The NWE_CYCLE field is common to both the NWE and NCS signals, thus the timing period is of the same duration.

NWE_CYCLE, NWE_SETUP, and NWE_PULSE implicitly define the NWE_HOLD value as:

NWE_HOLD = NWE_CYCLE - NWE_SETUP - NWE_PULSE

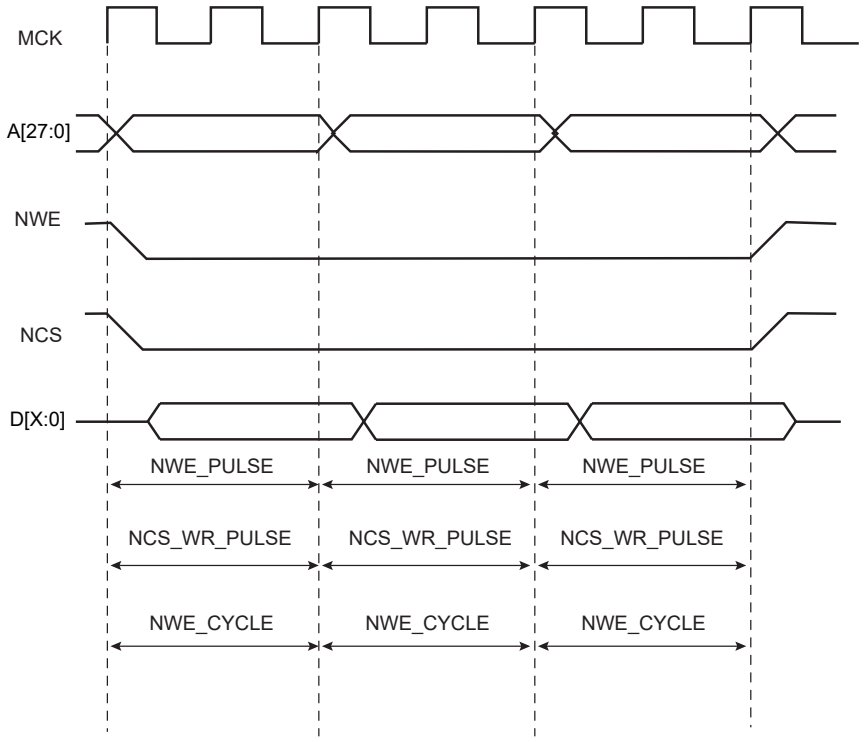
NWE_CYCLE, NCS_WR_SETUP, and NCS_WR_PULSE implicitly define the NCS_WR_HOLD value as:

NCS_WR_HOLD = NWE_CYCLE - NCS_WR_SETUP - NCS_WR_PULSE

30.6.8 Null Delay Setup and Hold

If null setup parameters are programmed for NWE and/or NCS, NWE and/or NCS remain active continuously in case of consecutive write cycles in the same memory (see the following figure). However, for devices that perform write operations on the rising edge of NWE or NCS, such as SRAM, either a setup or a hold must be programmed.

Figure 30-6. Null Setup and Hold Values of NCS and NWE in Write Cycle



30.6.9 Null Pulse

Programming null pulse is not permitted. Pulse must be at least set to 1. A null value leads to unpredictable behavior.

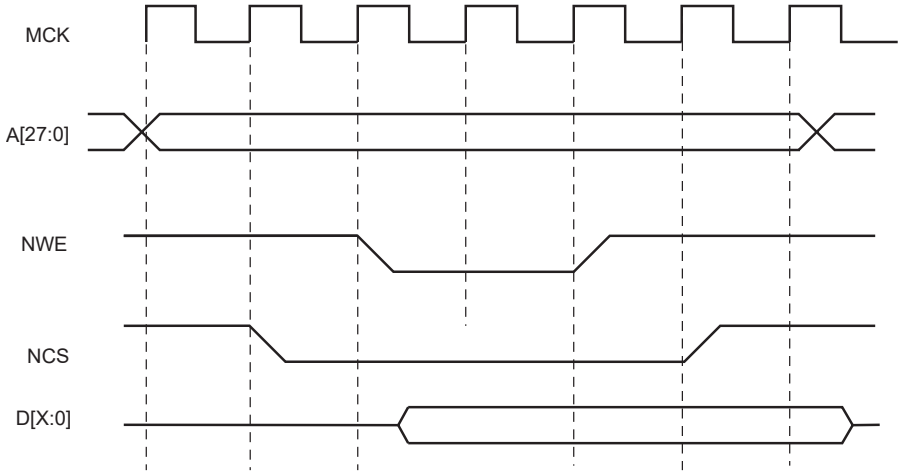
30.6.10 Write Mode

The bit WRITE_MODE in the HSMC_MODE register of the corresponding chip select indicates which signal controls the write operation.

30.6.10.1 Write is Controlled by NWE (HSMC.MODE.WRITE_MODE = 1):

The following figure shows the waveforms of a write operation with HSMC_MODE.WRITE_MODE set . The data is put on the bus during the pulse and hold steps of the NWE signal. The internal data buffers are switched to Output mode after the NWE_SETUP time, and until the end of the write cycle, regardless of the programmed waveform on NCS.

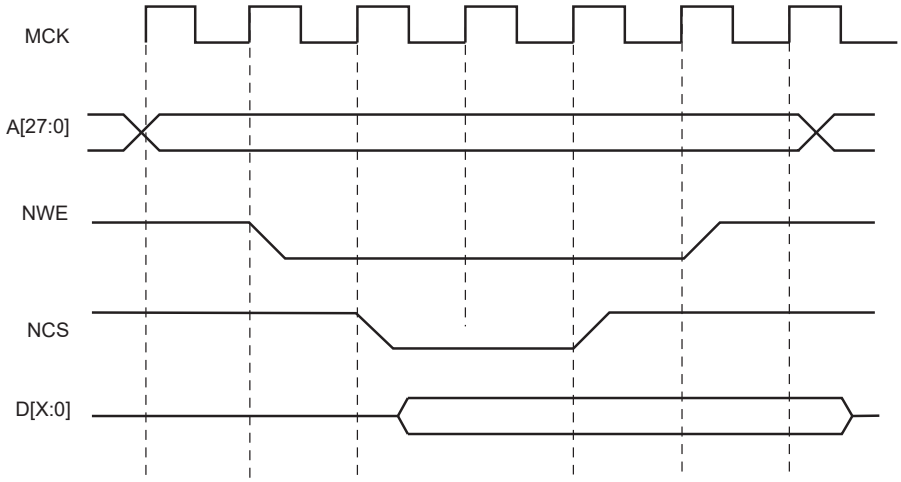
Figure 30-7. HSMC_MODE.WRITE_MODE = 1. Write Operation is Controlled by NWE



30.6.10.2 Write is Controlled by NCS (HSMC.MODE.WRITE_MODE = 0)

The following figure shows the waveforms of a write operation with HSMC_MODE.WRITE_MODE cleared. The data is put on the bus during the pulse and hold steps of the NCS signal. The internal data buffers are switched to Output mode after the NCS_WR_SETUP time, and until the end of the write cycle, regardless of the programmed waveform on NWE.

Figure 30-8. WRITE_MODE = 0. Write Operation is Controlled by NCS



30.6.11 Read Modify Write (RMW) Waveform

Instead of writing significant byte of a word, the user can ask to the processor to perform a Read Modify Write access. it consist in the reading of the whole 32-bit word, then modify it, then write back the whole 32-bit word.
Note: RMW (HSMC_MODE.RMW_ENABLE) is activated to ensure data integrity when HECC (HECC_CR.ENABLE) is activated.

Therefore, the waveform observed for such an access is the sum of the previous one, depending on the bus width and the HECC activation:

- In 32-bit mode, HECC ON/OFF: 1 x RD 32 bits followed by 1 x WR 32 bits.
- In 16-bit mode, HECC OFF: 2 x RD 16 bits followed by 2 x WR 16 bits.
- In 16-bit mode, HECC ON, Hamming (32,7) or BCH (32,12): 2 x RD 16 bits + 1 x RD 16 bits followed by 2 x WR 16 bits + 1 x WR 16 bit.
- In 8-bit mode, HECC OFF: 4 x RD 8 bits followed by 4 x WR 8 bits.
- In 8-bit mode, HECC ON, Hamming (32,7) code selected: 4 x RD 8 bits + 1 x RD 8 bits followed by 4 x WR 8 bits + 1 x WR 8 bits.

- In 8-bit mode, HECC ON, BCH (32,12) code selected: 4 x RD 8 bits + 2 x RD 8 bits followed by 4 x WR 8 bits + 2 x WR 8 bits.

30.6.12 Read Modify Write (RMW) Mode

Because RMW access is made of a Read access then a Write access, the RMW mode is configured by:

- Selecting an access mode for read cycle : HSMC_MR.READ_MODE.
- Selecting an access mode for write cycle : HSMC_MR.WRITE_MODE.

30.7 Register Write Protection

To prevent any single software error that may corrupt HSMC behavior, the registers listed below can be write-protected by setting the WPEN bit in the HSMC Write Protection Mode register (HSMC_WPMR).

If a write access in a write-protected register is detected, the WPVS flag in the HSMC Write Protection Status register (HSMC_WPSR) is set and the field WPVSRC indicates in which register the write access has been attempted.

The WPVS flag is automatically cleared after reading the HSMC_WPSR.

The following registers can be write-protected:

- HSMC Setup Register
- HSMC Pulse Register
- HSMC Cycle Register
- HSMC Mode Register

30.8 Coding Timing Parameters

All timing parameters are defined for one chip select and are grouped together in one register according to their type.

The HSMC_SETUP register groups the definition of all setup parameters:

- NRD_SETUP
- NCS_RD_SETUP
- NWE_SETUP
- NCS_WR_SETUP

The HSMC_PULSE register groups the definition of all pulse parameters:

- NRD_PULSE
- NCS_RD_PULSE
- NWE_PULSE
- NCS_WR_PULSE

The HSMC_CYCLE register groups the definition of all cycle parameters:

- NRD_CYCLE
- NWE_CYCLE

The following table shows how the timing parameters are coded and their permitted range.

Table 30-2. Coding and Range of Timing Parameters

Coded Value	Number of Bits	Effective Value	Permitted Range	
			Coded Value	Effective Value
setup [5:0]	6	128 × setup[5] + setup[4:0]	0 ≤ 31	0 ≤ 128+31
pulse [6:0]	7	256 × pulse[6] + pulse[5:0]	0 ≤ 63	0 ≤ 256+63

.....continued				
Coded Value	Number of Bits	Effective Value	Permitted Range	
			Coded Value	Effective Value
cycle [8:0]	9	256 × cycle[8:7] + cycle[6:0]	0 ≤ 127	0 ≤ 256+127 0 ≤ 512+127 0 ≤ 768+127

30.9 Usage Restriction

Note: The HSMC does not check the validity of the user-programmed parameters. If the sum of SETUP and PULSE parameters is larger than the corresponding CYCLE parameter, this leads to unpredictable behavior of the HSMC.

- For read operations:
  Null but positive setup and hold of address and NRD and/or NCS can not be guaranteed at the memory interface because of the propagation delay of theses signals through external logic and pads. If positive setup and hold values must be verified, then it is strictly recommended to program non-null values so as to cover possible skews between address, NCS and NRD signals.
- For write operations:
  If a null hold value is programmed on NWE, the HSMC can guarantee a positive hold of address and NCS signal after the rising edge of NWE. This is true for HSMC_MODE.WRITE_MODE = 1 only.
- For read and write operations:
  A null value for pulse parameters is forbidden and may lead to unpredictable behavior.

  In read and write cycles, the setup and hold time parameters are defined in reference to the address bus. For external devices that require setup and hold time between NCS and NRD signals (read), or between NCS and NWE signals (write), these setup and hold times must be converted into setup and hold times in reference to the address bus.

30.10 Read/Write Accesses Use Cases

On one hand, the HSMC can be configured in 8-, 16-, 32-/40-/48-bit mode, in order to access different widths of memories.
Note: When HECC protection is activated in 32-bit mode and Hamming(32,7) code is selected, the HSMC has a total data size of 40 bits.

Note: When HECC protection is activated in 32-bit mode and BCH(32,12) code is selected, the HSMC has a total data size of 48 bits.

On the other hand, the processor is a 32-bit one. Hence, the following AMBA requests that can be received by the HSMC are listed:

- Write 8-bit (WR_byte), Write 16-bit (WR_half_word), Write 32-bit (WR_word), Write n x 32-bit (WR_n_word)
- Read 8-bit (RD_byte), Read 16-bit (RD_half_word), Read 32-bit (RD_word), Read n x 32-bit (RD_n_word)

Therefore, this induces that different memory accesses are visible on the external memory interface, depending on the AMBA requests and the following options:

- Memory wide + HECC ON/OFF (8,16, 32/40/48 bits)
- RMW ON/OFF

Note: HECC ON implies the activation of RMW, to insure data integrity.

In the following sections, (//) means that the data are sent in parallel whereas (+) means that checksum are sent after data with additional cycles.

30.10.1 32-bit Mode

30.10.1.1 HECC Protection Deactivated (32-bit mode)

The table below sums up the different accesses on the external memory interface depending on the AMBA request, when 32-bit mode is selected and HECC protection is de-activated.

Table 30-3. Memory Access – 32-bit Mode – HECC OFF

AMBA Request	HSMC [32-bit] [HECC OFF] [RMW OFF]	HSMC [32-bit] [HECC OFF] [RMW ON]
RD_byte RD_half_word	1x RD 32 bits	1x RD 32 bits
WR_byte WR_half_word	1x WR 32 bits (1)	(RMW) (2)
RD_word	1x RD 32 bits	1x RD 32 bits
WR_word	1x WR 32 bits	1x WR 32 bits
RD_n_word	n x RD 32 bits (3)	n x RD 32 bits (3)
WR_n_word	n x WR 32 bits (3)	n x WR 32 bits (3)

Note:

1. Only the effective bytes are written depending the value of NWR[4:0]
2. (RMW)= 1x RD 32-bits then 1x WR 32 bits
3. The NCS and the control signals are not rising between the consecutives 1x RD 32 bits (or 1x WR 32 bits) access.

30.10.1.2 HECC Protection Activated (40-bit Mode (resp. 48-bit Mode))

The table below summarizes the different accesses on the external memory interface, depending on the AMBA request received, when 32-bit mode is selected and HECC protection is activated.

Table 30-4. Memory Access – 32-bit Mode – HECC ON

AMBA request	HSMC [32-bit] [HECC ON – Hamming(32,7)] [RMW ON] [data][sequence][checksum]	HSMC [32-bit] [HECC ON – BCH(32,12)] [RMW ON] [data][sequence][checksum]
RD_byte RD_half_word	(1x RD 32 bits) (//) (1 x RD 8 bits)	(1x RD 32 bits) (//) (1 x RD 16 bits)
WR_byte WR_half_word	(RMW_ECCON8) (1)	(RMW_ECCON16) (1)
RD_word	(1x RD 32 bits) (//) (1 x RD 8 bits)	(1x RD 32 bits) (//) (1 x RD 16 bits)
WR_word	(1x WR 32 bits) (//) (1 x WR 8 bits)	(1x WR 32 bits) (//) (1 x WR 16 bits)
RD_n_word	nx [(1 x RD 32 bits) (//) (1 x RD 8 bits)] (2)	nx [(1 x RD 32 bits) (//) (1 x RD 16 bits)] (2)
WR_n_word	nx [(1 x WR 32 bits) (//) (1 x WR 8 bits)] (2)	nx[(1 x WR 32 bits) (//) (1 x WR 16 bits)] (2)

Note:

1. (RMW_HECCONn)= [(1x RD 32 bits) (//) (1 x RD 8 bits) then [(1x WR 32 bits) (//) (1 x WR n bits)].
2. The NCS and the control signals are not rising between consecutive RD 32 bits (or WR 32 bits) accesses. (idem for RD 8/16 bits (or WR 8/16 bits) accesses.)

30.10.2 16-bit Mode

30.10.2.1 HECC Protection Deactivated

The table below sums up the different accesses visible on the external memory interface depending on the AMBA request, when 16-bit mode is selected and HECC protection is deactivated.

Table 30-5. Memory Access – 16-bit Mode – HECC OFF

AMBA Request	HSMC[16-bit] [HECC OFF] [RMW OFF]	HSMC[16-bit] [HECC OFF] [RMW ON]
RD_byte RD_half_word	1x RD 16 bits	1x RD 16 bits
WR_byte WR_half_word	1 x WR 16 bits(1)	(RMW)(2)
RD_word	2 x RD 16 bits(3)	2 x RD 16 bits(3)
WR_word	2 x WR 16 bits(3)	2 x WR 16 bits(3)
RD_n_word	nx [2 x RD 16 bits](3)	nx [2 x RD 16 bits](3)
WR_n_word	nx [2 x WR 16 bits](3)	nx [2 x WR 16 bits](3)

Note:

1. Only the effective bytes are written by using NWR[4:0].
2. (RMW)= [(2x RD 16 bits) then [(2x WR 16 bits).
3. The NCS and the control signals are not rising between the consecutive RD 16 bits (or WR 16 bits) access.

30.10.2.2 HECC Protection Activated

The table below sums up the different accesses on the external memory interface depending on the AMBA request, when 16-bit mode is selected and HECC protection is activated.

Table 30-6. Memory Access – 16-bit Mode – HECC ON

AMBA request	HSMC[16-bit] [HECC ON – Hamming(32,7)] [RMW ON] [data][sequence][checksum]	HSMC [16-bit] [HECC ON – BCH(32,12)] [RMW ON] [data][sequence][checksum]
RD_byte RD_half_word	(2 x RD 16 bits) (+) (1 x RD 16 bits) (1)	(2 x RD 16 bits) (+) (1 x RD 16 bits) (1)
WR_byte WR_half_word	(RMW_ECCON) (2)	(RMW_ECCON) (2)
RD_word	(2 x RD 16 bits) (+) (1 x RD 16 bits) (1)	(2 x RD 16 bits) (+) (1 x RD 16 bits) (1)
WR_word	[(2 x WR 16 bits) (+) (1 x WR 16 bits)] (1)	[(2 x WR 16 bits) (+) (1 x WR 16 bits)] (1)

.....continued		
AMBA request	**HSMC[16-bit] [HECC ON – Hamming(32,7)] [RMW ON] [data][sequence][checksum]**	**HSMC [16-bit] [HECC ON – BCH(32,12)] [RMW ON] [data][sequence][checksum]**
RD_n_word	nx [(2 x RD 16 bits) (+) (1 x RD 16 bits)] (1)	n x [(2 x RD 16 bits) (+) (1 x RD 16 bits)] (1)
WR_n_word	nx [(2 x WR 16 bits) (+) (1 x WR 16 bits)] (1)	nx [(2 x WR 16 bits) (+) (1 x WR 16 bits)] (1)

Note:

1. The NCS and the control signals are not rising between consecutive RD 16 bits (or WR 16 bits) accesses.
2. (RMW_ECCON) = [(2x RD 16 bits) (+) (1 x RD 16 bits) then [(2x WR 16 bits) (+) (1 x WR 16 bits)].

30.10.3 8-bit Mode

30.10.3.1 HECC Protection Deactivated

The table below sums up the different accesses on external memory interface, depending on the AMBA requests received, when the 8-bit mode is selected and HECC protection is deactivated.

Table 30-7. Memory Access – 8-bit mode – HECC OFF

AMBA request	HSMC[8-bit] [HECC OFF] [RMW OFF]	HSMC[8-bit] [HECC OFF] [RMW ON]
RD_byte	1 x RD 8 bits	1 x RD 8 bits
WR_byte	1 x WR 8 bits	RMW (1)
RD_half_word	2 x RD 8 bits (2)	2 x RD 8 bits (2)
WR_half_word	2 x WR 8 bits (1)	RMW (1)
RD_word	4 x RD 8 bits (2)	4 x RD 8 bits (2)
WR_word	4 x WR 8 bits (2)	4 x WR 8 bits (2)
RD_n_word	n x [4 x RD 8 bits] (2)	n x [4 x RD 8 bits] (2)
WR_n_word	n x [4 x WR 8 bits] (2)	n x [4 x WR 8 bits] (2)

Note:

1. (RMW)=(4x RD 8 bits) then (4x WR 8 bits).
2. The NCS and the control signals are not rising between the consecutive RD 8 bits (or WR 8 bits) access.

30.10.3.2 HECC Protection Activated

The table below sums up the different accesses on external memory interface, depending on the AMBA requests received, when 8 bit mode is selected and the HECC protection is activated.

Table 30-8. Memory Access – 8-bit mode – HECC ON

AMBA request	HSMC[8-bit] [HECC ON – Hamming(32,7)] [RMW ON] [data][sequence][checksum]	HSMC[8-bit] [HECC ON – BCH(32,12)] [RMW ON] [data][sequence][checksum]
RD_byte RD_half_word	(4 x RD 8 bits) (+) (1x RD 8 bits)(1)	(4 x RD 8 bits) (+) (2x RD 8 bits)(1)
WR_byte WR_half_word	RMW_ECCON8(2)	RMW_ECCON16(3)
RD_word	(4 x RD 8 bits) (+) (1x RD 8 bits) (1)	(4 x RD 8 bits) (+) (2x RD 8 bits)(1)
WR_word	(4 x WR 8 bits) (+) (1x WR 8 bits)(1)	(4 x WR 8 bits) (+) (2x WR 8 bits)(1)
RD_n_word	n x [(4 x RD 8 bits) (+) (1 x RD 8 bits)](1)	n x [(4 x RD 8 bits) (+) (2 x RD 8 bits)](1)
WR_n_word	n x [(4 x WR 8 bits) (+) (1 x WR 8 bits)](1)	n x [(4 x WR 8 bits) (+) (2 x WR 8 bits)](1)

Note:

1. The NCS and the control signals are not rising between the consecutive RD 8 bits (or WR 8 bits) access.
2. (RMW_HECCON8)= [(4 x RD 8 bits) (+) (1x RD 8 bits) then [(4 x WR 8 bits) (+) (1 x WR 8 bits).
3. (RMW_HECCON16)= [(4 x RD 8 bits) (+) (2 x RD 8 bits) then [(4 x WR 8 bits) (+) (2 x WR 8 bits)].

30.11 Automatic Wait States

Under certain circumstances, the HSMC automatically inserts idle cycles between accesses to avoid bus contention or operation conflict.

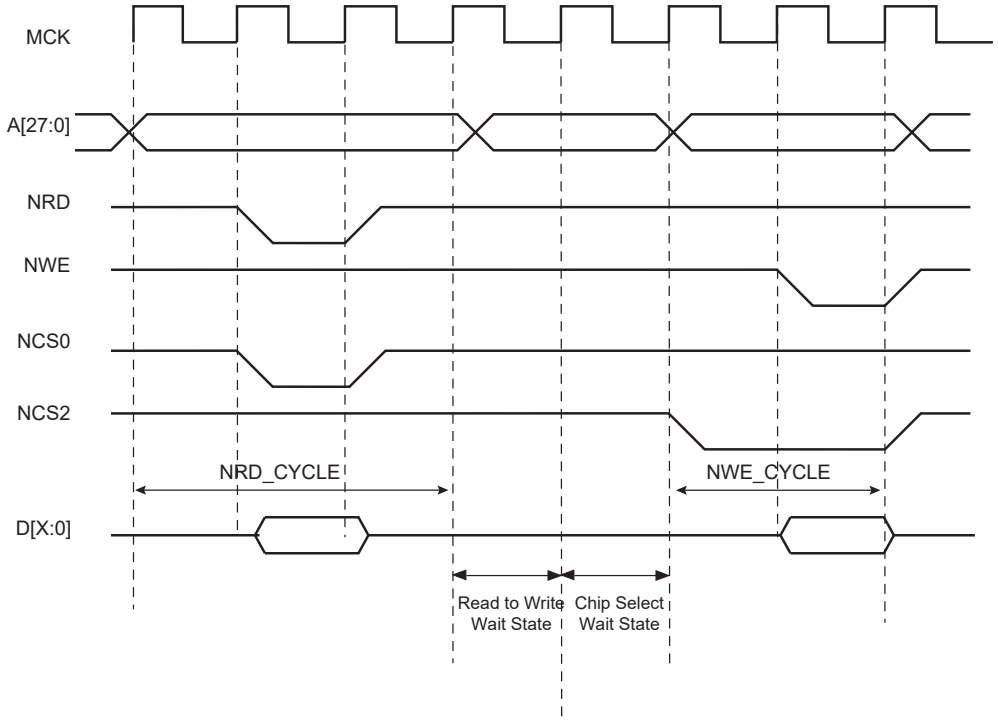
30.11.1 Chip Select Wait States

The HSMC always inserts an idle cycle between two transfers on separate Chip Selects. This idle cycle ensures that there is no bus contention between the deactivation of one device and the activation of the next one.

During Chip Select Wait state, all control lines are turned inactive: NWR, NCSx, NRD lines are all set to 1.

The following figure illustrates a Chip Select Wait state between access on Chip Select 0 and Chip Select 2.

Figure 30-9. Chip Select Wait State between a Read Access on NCS0 and a Write Access on NCS2



30.11.2 Early Read Wait State

In some cases, the HSMC inserts a wait state cycle between a write access and a read access to allow time for the write cycle to end before the subsequent read cycle begins. This wait state is not generated in addition to a chip select wait state. The early read cycle thus only occurs between a write and read access to the same memory device (same chip select).

An early read wait state is automatically inserted if at least one of the following conditions is valid:

- if the write controlling signal has no hold time and the read controlling signal has no setup time (Figure 30-10).
- in NCS Write controlled mode (HSMC_MODE.WRITE_MODE = 0), if there is no hold timing on the NCS signal and the NCS_RD_SETUP parameter is set to 0, regardless of the Read mode (Figure 30-11). The write operation must end with a NCS rising edge. Without an Early Read Wait State, the write operation could not complete properly.
- in NWE controlled mode (HSMC_MODE.WRITE_MODE = 1) and if there is no hold timing (NWE_HOLD = 0), the feedback of the write control signal is used to control address, data, and chip select lines. If the external write control signal is not inactivated as expected due to load capacitances, an Early Read Wait State is inserted and address, data and control signals are maintained one more cycle. See Figure 30-12.

Figure 30-10. Early Read Wait State: Write with No Hold Followed by Read with No Setup

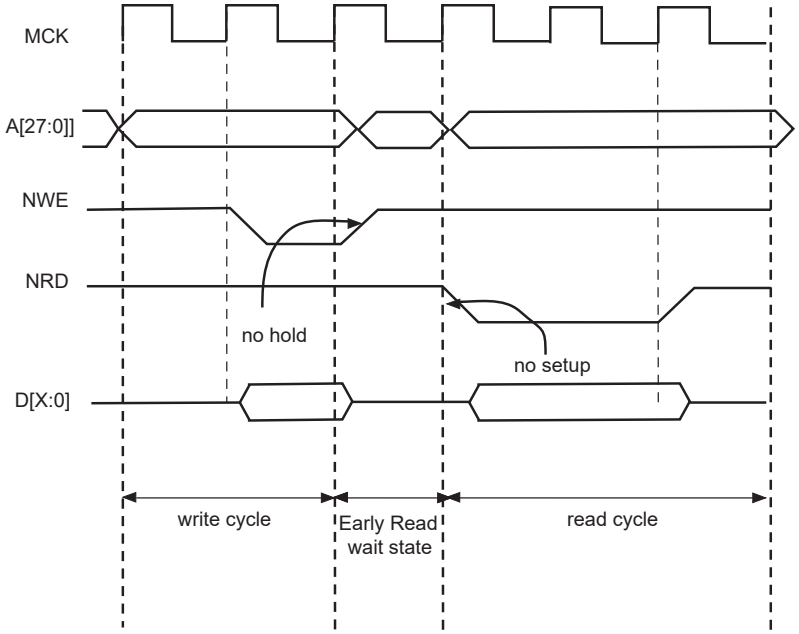


Figure 30-11. Early Read Wait State: NCS-controlled write with no hold followed by a read with no NCS setup

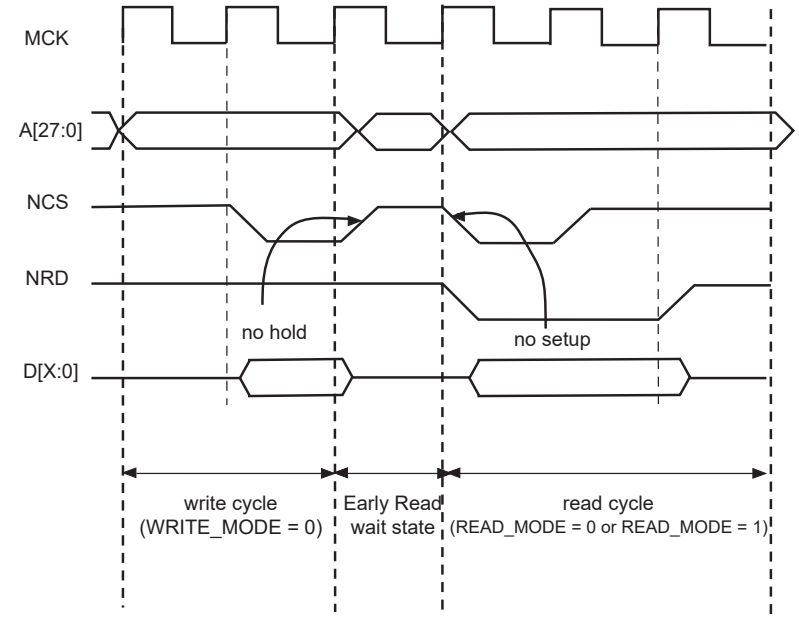
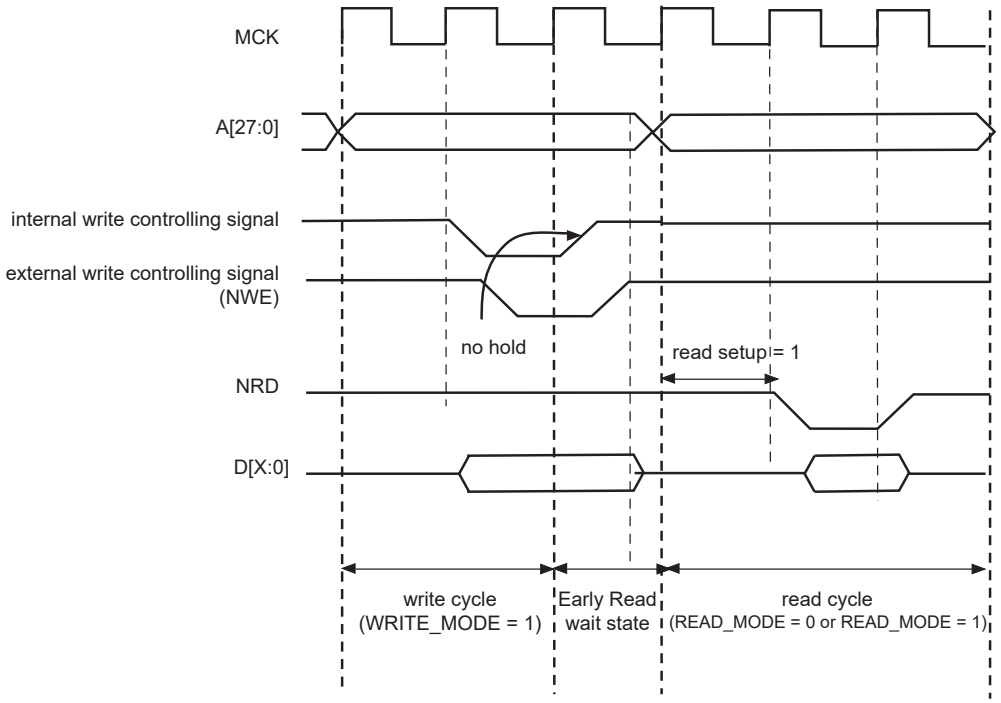


Figure 30-12. Early Read Wait State: NWE-controlled write with no hold followed by a read with one set-up cycle



30.11.3 Reload User Configuration Wait State

The user may change any of the configuration parameters by writing the HSMC user interface.

When detecting that a new user configuration has been written in the user interface, the HSMC inserts a wait state before starting the next access. This “reload user configuration wait state” is used by the HSMC to load the new set of parameters to apply to next accesses.

The reload configuration wait state is not applied in addition to the chip select wait state. If accesses before and after re-programming the user interface are made to different devices (chip selects), then one single chip select wait state is applied.

On the other hand, if accesses before and after writing the user interface are made to the same device, a reload configuration wait state is inserted, even if the change does not concern the current chip select.

30.11.3.1 User Procedure

To insert a reload configuration wait state, the HSMC detects a write access to any HSMC_MODE register of the user interface. If the user only modifies timing registers (HSMC_SETUP, HSMC_PULSE, HSMC_CYCLE registers) in the user interface, he must validate the modification by writing the HSMC_MODE, even if no change was made on the mode parameters.

The user must not change the configuration parameters of an HSMC chip select (Setup, Pulse, Cycle, Mode) if accesses are performed on this CS during the modification. Any change of the chip select parameters, while fetching the code from a memory connected on this CS, may lead to unpredictable behavior. The instructions used to modify the parameters of an HSMC chip select can be executed from the internal RAM or from a memory connected to another CS.

30.11.4 Read to Write Wait State

Due to an internal mechanism, a wait cycle is always inserted between consecutive read and write HSMC accesses.

This wait cycle is referred to as a read to write wait state in this document.

This wait cycle is applied in addition to chip select and reload user configuration wait states when they are to be inserted. See Figure 30-9.

30.12 External Wait

Any access can be extended by an external device using the NWAIT input signal of the HSMC. The HSMC_MODE.EXNW_MODE field on the corresponding chip select must be set ei ther to “10” (Frozen mode) or “11” (Ready mode). When HSMC_MODE.EXNW_MODE is set to “00” (disabled), the NWAIT signal is simply ignored on the corresponding chip select. The NWAIT signal delays the read or write operation in regards to the read or write controlling signal, depending on the Read and Write modes of the corresponding chip select.

30.12.1 Restriction

Important:
When HSMC_MODE.EXNW_MODE is enabled, it is mandatory to program at least one hold cycle for the read/write controlling signal.

The NWAIT signal is assumed to be a response of the external device to the read/write request of the HSMC. Then NWAIT is examined by the HSMC only in the pulse state of the read or write controlling signal. The assertion of the NWAIT signal outside the expected period has no impact on HSMC behavior.

30.12.2 Frozen Mode

When the external device asserts the NWAIT signal (active low), and after internal synchronization of this signal, the HSMC state is frozen, i.e., HSMC internal counters are frozen, and all control signals remain unchanged. When the resynchronized NWAIT signal is deasserted, the HSMC completes the access, resuming the access from the point where it was stopped. See Figure 30-13. This mode must be selected when the external device uses the NWAIT signal to delay the access and to freeze the HSMC.

The assertion of the NWAIT signal outside the expected period is ignored as illustrated in Figure 30-14.

Figure 30-13. Write Access with NWAIT Assertion in Frozen Mode (HSMC_MODE.EXNW_MODE = 10)

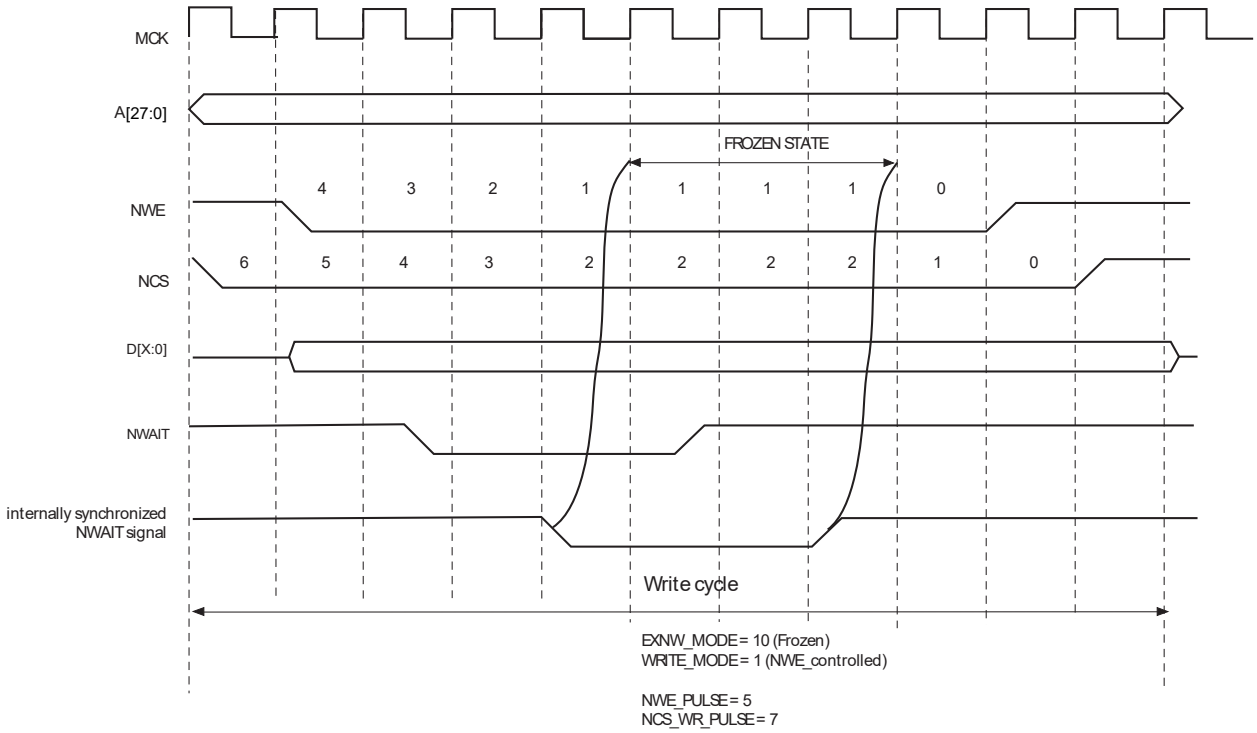
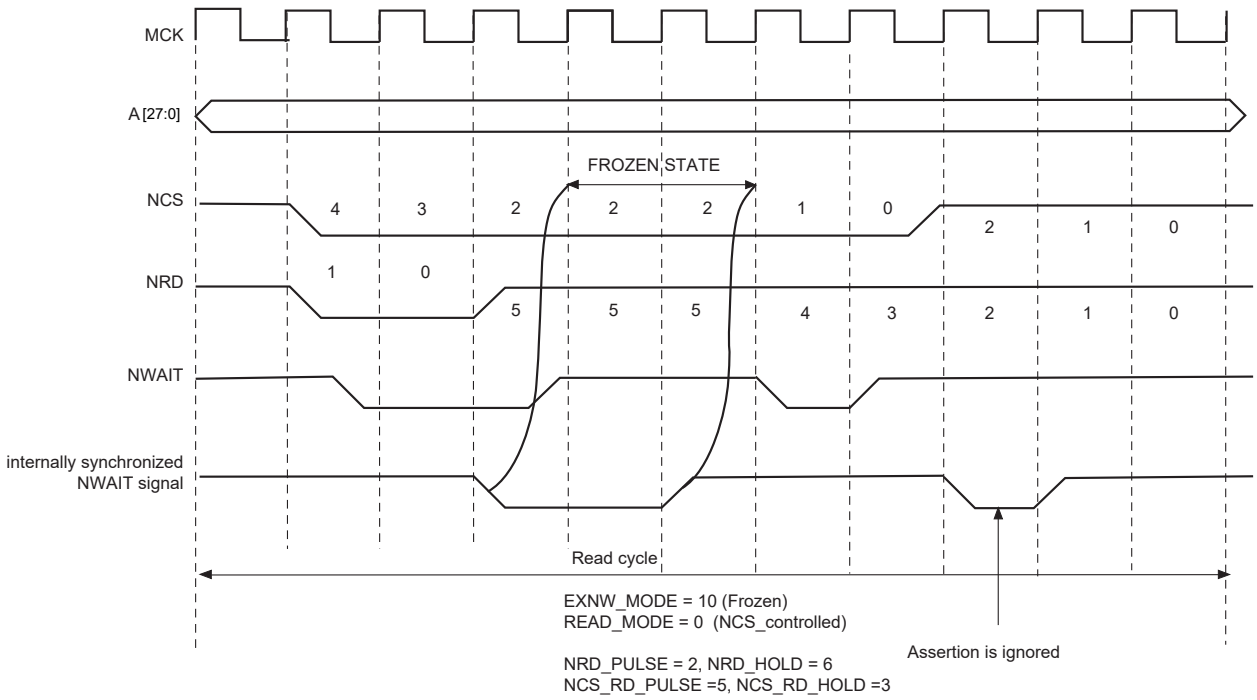


Figure 30-14. Read Access with NWAIT Assertion in Frozen Mode (HSMC_MODE.EXNW_MODE = 10)



30.12.3 Ready Mode

In Ready mode (HSMC_MODE.EXNW_MODE = 11), the HSMC behaves differently. Normally, the HSMC begins the access by down counting the setup and pulse counters of the read/write controlling signal. In the last cycle of the pulse phase, the resynchronized NWAIT signal is examined.

If asserted, the HSMC suspends the access as shown in the figures below. After deassertion, the access is completed: the hold step of the access is performed.

This mode must be selected when the external device uses deassertion of the NWAIT signal to indicate its ability to complete the read or write operation.

If the NWAIT signal is deasserted before the end of the pulse, or asserted after the end of the pulse of the controlling read/write signal, it has no impact on the access length as shown in Figure 30-16.

Figure 30-15. NWAIT Assertion in Write Access: Ready Mode (HSMC_MODE.EXNW_MODE = 11)

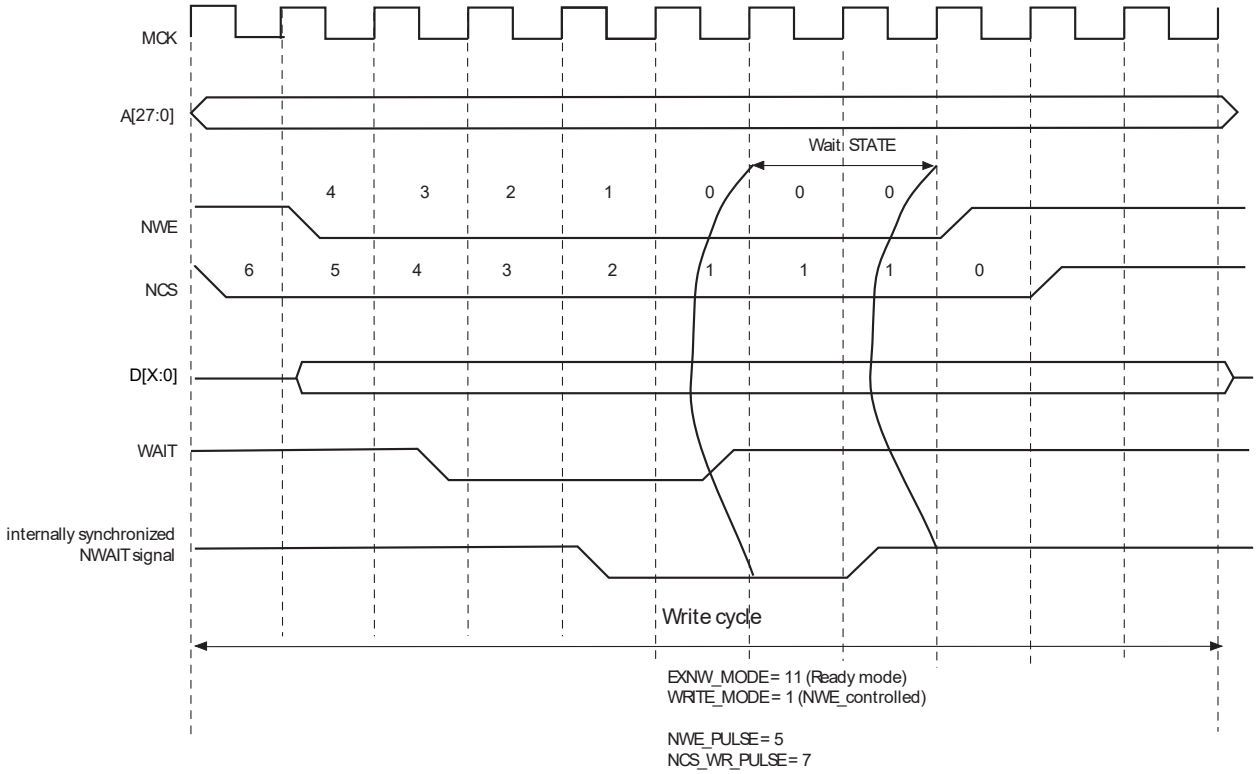
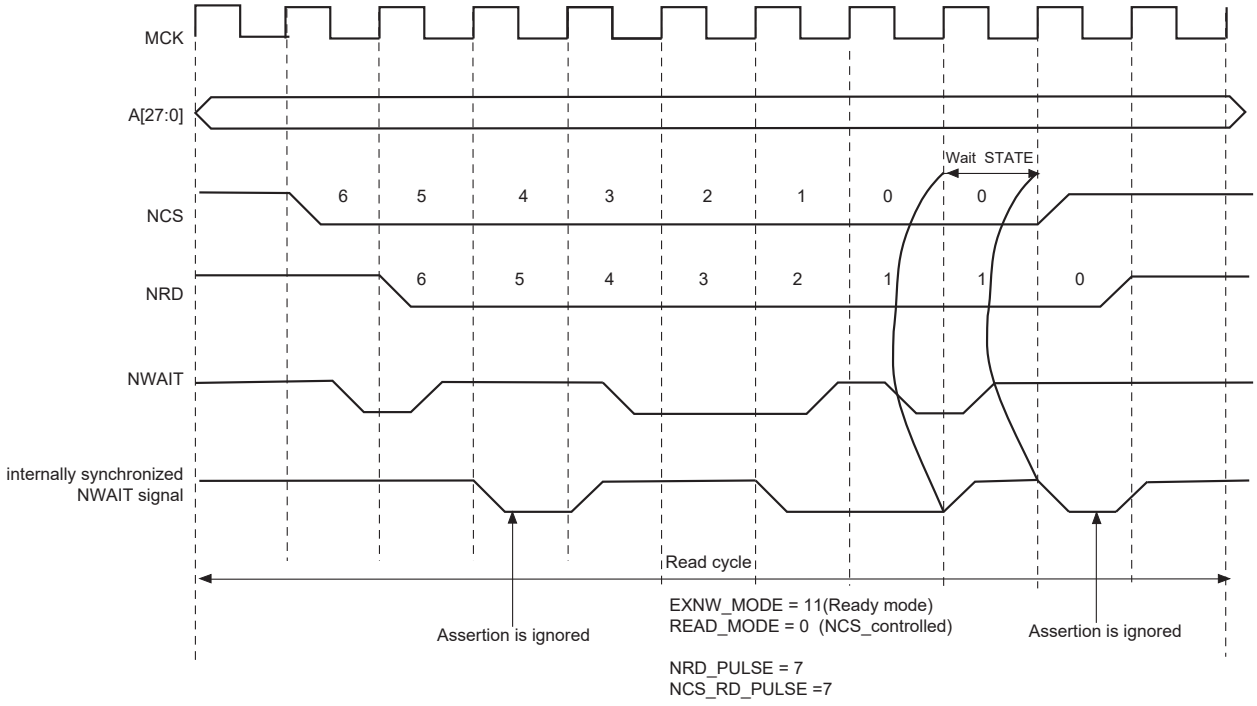


Figure 30-16. NWAIT Assertion in Read Access: Ready Mode (HSMC_MODE.EXNW_MODE = 11)



30.12.4 NWAIT Latency and Read/Write Timings

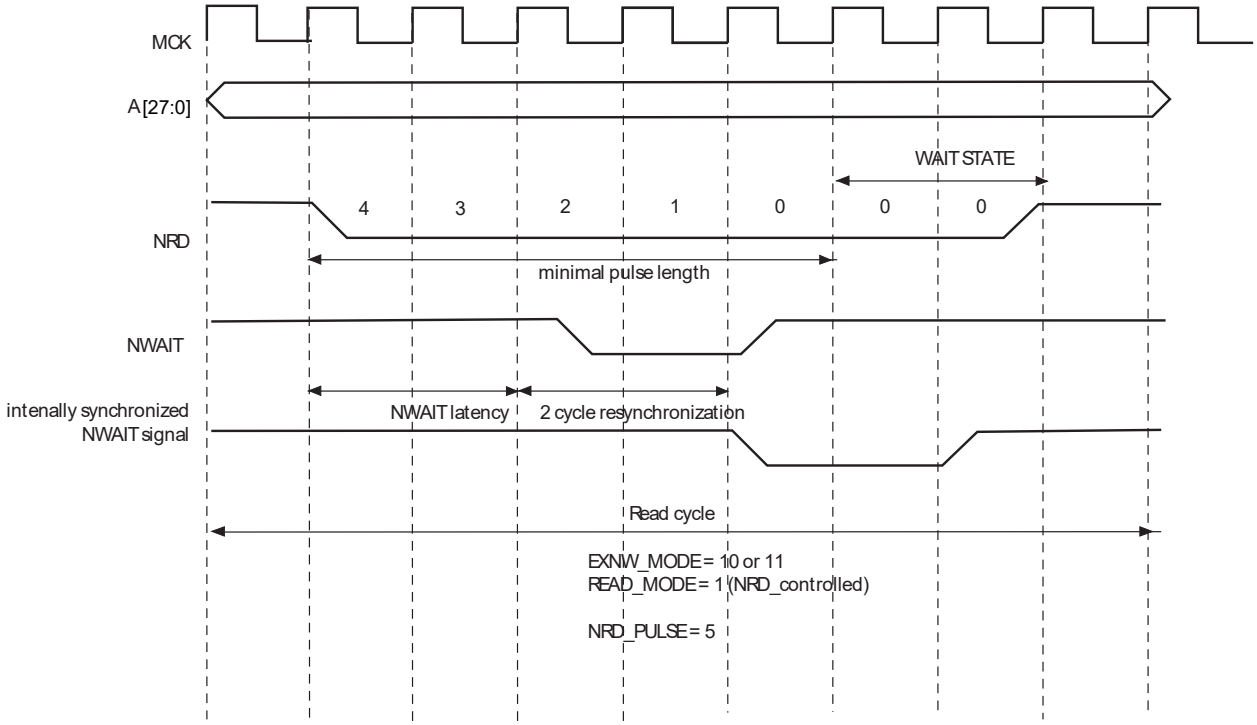
There may be a latency between the assertion of the read/write controlling signal and the assertion of the NWAIT signal by the device. The programmed pulse length of the read/write controlling signal must be at least equal to this latency plus the 2 cycles of resynchronization + one cycle. Otherwise, the HSMC may enter the hold state of the

access without detecting the NWAIT signal assertion. This is true in Frozen mode as well as in Ready mode. This is illustrated in the following figure.

When HSMC_MODE.EXNW_MODE is enabled (ready or frozen), the user must program a pulse length of the read and write controlling signal of at least:

Minimal pulse length = NWAIT latency + 2 resynchronization cycles + 1 cycle

Figure 30-17. NWAIT Latency



30.13 Register Summary

Offset	Name	Bit Pos.								
0x00	HSMC_SETUP0	7:0			NWE_SETUP[5:0]					
		15:8			NCS_WR_SETUP[5:0]					
		23:16			NRD_SETUP[5:0]					
		31:24			NCS_RD_SETUP[5:0]					
0x04	HSMC_PULSE0	7:0		NWE_PULSE[6:0]						
		15:8		NCS_WR_PULSE[6:0]						
		23:16		NRD_PULSE[6:0]						
		31:24		NCS_RD_PULSE[6:0]						
0x08	HSMC_CYCLE0	7:0	NWE_CYCLE[7:0]							
		15:8								NWE_CYCLE[8]
		23:16	NRD_CYCLE[7:0]							
		31:24								NRD_CYCLE[8]
0x0C	HSMC_MODE0	7:0			EXNW_MODE[1:0]			RMW_ENABLE	WRITE_MODE	READ_MODE
		15:8							DBW[1:0]	
		23:16								
		31:24								
0x10	HSMC_SETUP1	7:0			NWE_SETUP[5:0]					
		15:8			NCS_WR_SETUP[5:0]					
		23:16			NRD_SETUP[5:0]					
		31:24			NCS_RD_SETUP[5:0]					
0x14	HSMC_PULSE1	7:0		NWE_PULSE[6:0]						
		15:8		NCS_WR_PULSE[6:0]						
		23:16		NRD_PULSE[6:0]						
		31:24		NCS_RD_PULSE[6:0]						
0x18	HSMC_CYCLE1	7:0	NWE_CYCLE[7:0]							
		15:8								NWE_CYCLE[8]
		23:16	NRD_CYCLE[7:0]							
		31:24								NRD_CYCLE[8]
0x1C	HSMC_MODE1	7:0			EXNW_MODE[1:0]			RMW_ENABLE	WRITE_MODE	READ_MODE
		15:8							DBW[1:0]	
		23:16								
		31:24								
0x20	HSMC_SETUP2	7:0			NWE_SETUP[5:0]					
		15:8			NCS_WR_SETUP[5:0]					
		23:16			NRD_SETUP[5:0]					
		31:24			NCS_RD_SETUP[5:0]					
0x24	HSMC_PULSE2	7:0		NWE_PULSE[6:0]						
		15:8		NCS_WR_PULSE[6:0]						
		23:16		NRD_PULSE[6:0]						
		31:24		NCS_RD_PULSE[6:0]						
0x28	HSMC_CYCLE2	7:0	NWE_CYCLE[7:0]							
		15:8								NWE_CYCLE[8]
		23:16	NRD_CYCLE[7:0]							
		31:24								NRD_CYCLE[8]

.....continued

Offset	Name	Bit Pos.								
0x2C	HSMC_MODE2	7:0			EXNW_MODE[1:0]			RMW_ENABLE	WRITE_MODE	READ_MODE
		15:8							DBW[1:0]	
		23:16								
		31:24								
0x30	HSMC_SETUP3	7:0			NWE_SETUP[5:0]					
		15:8			NCS_WR_SETUP[5:0]					
		23:16			NRD_SETUP[5:0]					
		31:24			NCS_RD_SETUP[5:0]					
0x34	HSMC_PULSE3	7:0		NWE_PULSE[6:0]						
		15:8		NCS_WR_PULSE[6:0]						
		23:16		NRD_PULSE[6:0]						
		31:24		NCS_RD_PULSE[6:0]						
0x38	HSMC_CYCLE3	7:0	NWE_CYCLE[7:0]							
		15:8								NWE_CYCLE[8]
		23:16	NRD_CYCLE[7:0]							
		31:24								NRD_CYCLE[8]
0x3C	HSMC_MODE3	7:0			EXNW_MODE[1:0]			RMW_ENABLE	WRITE_MODE	READ_MODE
		15:8							DBW[1:0]	
		23:16								
		31:24								
0x40	HSMC_SETUP4	7:0			NWE_SETUP[5:0]					
		15:8			NCS_WR_SETUP[5:0]					
		23:16			NRD_SETUP[5:0]					
		31:24			NCS_RD_SETUP[5:0]					
0x44	HSMC_PULSE4	7:0		NWE_PULSE[6:0]						
		15:8		NCS_WR_PULSE[6:0]						
		23:16		NRD_PULSE[6:0]						
		31:24		NCS_RD_PULSE[6:0]						
0x48	HSMC_CYCLE4	7:0	NWE_CYCLE[7:0]							
		15:8								NWE_CYCLE[8]
		23:16	NRD_CYCLE[7:0]							
		31:24								NRD_CYCLE[8]
0x4C	HSMC_MODE4	7:0			EXNW_MODE[1:0]			RMW_ENABLE	WRITE_MODE	READ_MODE
		15:8							DBW[1:0]	
		23:16								
		31:24								
0x50	HSMC_SETUP5	7:0			NWE_SETUP[5:0]					
		15:8			NCS_WR_SETUP[5:0]					
		23:16			NRD_SETUP[5:0]					
		31:24			NCS_RD_SETUP[5:0]					
0x54	HSMC_PULSE5	7:0		NWE_PULSE[6:0]						
		15:8		NCS_WR_PULSE[6:0]						
		23:16		NRD_PULSE[6:0]						
		31:24		NCS_RD_PULSE[6:0]						
0x58	HSMC_CYCLE5	7:0	NWE_CYCLE[7:0]							
		15:8								NWE_CYCLE[8]
		23:16	NRD_CYCLE[7:0]							
		31:24								NRD_CYCLE[8]

.....continued

Offset	Name	Bit Pos.								
0x5C	HSMC_MODE5	7:0			EXNW_MODE[1:0]			RMW_ENABLE	WRITE_MODE	READ_MODE
		15:8							DBW[1:0]	
		23:16								
		31:24								
0x60 ... 0x6F	Reserved									
0x70	HSMC_WPMR	7:0								WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0x74	HSMC_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16	WPVSRC[15:8]							
		31:24								

30.13.1 HSMC Setup Register

Name: HSMC_SETUP
Offset: 0x00 + n*0x10 [n=0..5]
Reset: 0x02010102
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the 30.13.5 HSMC_WPMR.

Bit	31	30	29	28	27	26	25	24
			NCS_RD_SETUP[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	1	0

Bit	23	22	21	20	19	18	17	16
			NRD_SETUP[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8
			NCS_WR_SETUP[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	1

Bit	7	6	5	4	3	2	1	0
			NWE_SETUP[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	1	0

Bits 29:24 – NCS_RD_SETUP[5:0] NCS Setup Length in READ Access
In read access, the NCS signal setup length is defined as:
NCS setup length = (128* NCS_RD_SETUP[5] + NCS_RD_SETUP[4:0]) clock cycles

Bits 21:16 – NRD_SETUP[5:0] NRD Setup Length
The NRD signal setup length is defined in clock cycles as:
NRD setup length = (128* NRD_SETUP[5] + NRD_SETUP[4:0]) clock cycles

Bits 13:8 – NCS_WR_SETUP[5:0] NCS Setup Length in WRITE Access
In write access, the NCS signal setup length is defined as:
NCS setup length = (128* NCS_WR_SETUP[5] + NCS_WR_SETUP[4:0]) clock cycles

Bits 5:0 – NWE_SETUP[5:0] NWE Setup Length
The NWE signal setup length is defined as:
NWE setup length = (128* NWE_SETUP[5] + NWE_SETUP[4:0]) clock cycles

30.13.2 HSMC Pulse Register

Name: HSMC_PULSE
Offset: 0x04 + n*0x10 [n=0..5]
Reset: 0x2C2E2E2C
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the 30.13.5 HSMC_WPMR.

Bit	31	30	29	28	27	26	25	24
		NCS_RD_PULSE[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	1	0	1	1	0	0

Bit	23	22	21	20	19	18	17	16
		NRD_PULSE[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	1	0	1	1	1	0

Bit	15	14	13	12	11	10	9	8
		NCS_WR_PULSE[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	1	0	1	1	1	0

Bit	7	6	5	4	3	2	1	0
		NWE_PULSE[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	1	0	1	1	0	0

Bits 30:24 – NCS_RD_PULSE[6:0] NCS Pulse Length in READ Access
In standard read access, the NCS signal pulse length is defined as:
NCS pulse length = (256* NCS_RD_PULSE[6] + NCS_RD_PULSE[5:0]) clock cycles
The NCS pulse length must be at least 1 clock cycle.
In Page mode read access, the NCS_RD_PULSE parameter defines the duration of the first access to one page.

Bits 22:16 – NRD_PULSE[6:0] NRD Pulse Length
In standard read access, the NRD signal pulse length is defined in clock cycles as:
NRD pulse length = (256* NRD_PULSE[6] + NRD_PULSE[5:0]) clock cycles
The NRD pulse length must be at least 1 clock cycle.
In Page mode read access, the NRD_PULSE parameter defines the duration of the subsequent accesses in the page.

Bits 14:8 – NCS_WR_PULSE[6:0] NCS Pulse Length in WRITE Access
In write access, the NCS signal pulse length is defined as:
NCS pulse length = (256* NCS_WR_PULSE[6] + NCS_WR_PULSE[5:0]) clock cycles
The NCS pulse length must be at least 1 clock cycle.

Bits 6:0 – NWE_PULSE[6:0] NWE Pulse Length
The NWE signal pulse length is defined as:
NWE pulse length = (256* NWE_PULSE[6] + NWE_PULSE[5:0]) clock cycles
The NWE pulse length must be at least 1 clock cycle.

30.13.3 HSMC Cycle Register

Name: HSMC_CYCLE
Offset: 0x08 + n*0x10 [n=0..5]
Reset: 0x00300030
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the 30.13.5 HSMC_WPMR.

Bit	31	30	29	28	27	26	25	24
								NRD_CYCLE[8]
Access								R/W
Reset								0

Bit	23	22	21	20	19	18	17	16
	NRD_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	1	0	0	0	0

Bit	15	14	13	12	11	10	9	8
								NWE_CYCLE[8]
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
	NWE_CYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	1	0	0	0	0

Bits 24:16 – NRD_CYCLE[8:0] Total Read Cycle Length
The total read cycle length is the total duration in clock cycles of the read cycle. It is equal to the sum of the setup, pulse and hold steps of the NRD and NCS signals. It is defined as:
Read cycle length = (NRD_CYCLE[8:7]*256 + NRD_CYCLE[6:0]) clock cycles

Bits 8:0 – NWE_CYCLE[8:0] Total Write Cycle Length
The total write cycle length is the total duration in clock cycles of the write cycle. It is equal to the sum of the setup, pulse and hold steps of the NWE and NCS signals. It is defined as:
Write cycle length = (NWE_CYCLE[8:7]*256 + NWE_CYCLE[6:0]) clock cycles

30.13.4 HSMC Mode Register

Name: HSMC_MODE
Offset: 0x0C + n*0x10 [n=0..5]
Reset: 0x00000202
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the 30.13.5 HSMC_WPMR.

The user must confirm the HSMC configuration by writing any one of the HSMC_MODE registers.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							DBW[1:0]	
Access							R/W	R/W
Reset							1	0

Bit	7	6	5	4	3	2	1	0
			EXNW_MODE[1:0]			RMW_ENABLE	WRITE_MODE	READ_MODE
Access			R/W	R/W		R/W	R/W	R/W
Reset			0	0		0	1	0

Bits 9:8 – DBW[1:0] Data Bus Width

Value	Name	Description
0	8_BIT	8-bit data bus
1	16_BIT	16-bit data bus
2	32_BIT	32-bit data bus
3	–	Reserved

Bits 5:4 – EXNW_MODE[1:0] NWAIT Mode
The NWAIT signal is used to extend the current read or write signal. It is only taken into account during the pulse phase of the read and write controlling signal. When the use of NWAIT is enabled, at least one cycle hold duration must be programmed for the read and write controlling signal.

Value	Name	Description
0	DISABLED	Disabled–The NWAIT input signal is ignored on the corresponding chip select.
1	–	Reserved
2	FROZEN	Frozen Mode–If asserted, the NWAIT signal freezes the current read or write cycle. After deassertion, the read/write cycle is resumed from the point where it was stopped.
3	READY	Ready Mode–The NWAIT signal indicates the availability of the external device at the end of the pulse of the controlling read or write signal, to complete the access. If high, the access normally completes. If low, the access is extended until NWAIT returns high.

Bit 2 – RMW_ENABLE Read Modify Write Enable
Note: RMW must be enabled when the HECC is enabled.

Value	Description
0	Read Modify Write is disabled.
1	Read Modify Write is enabled.

Bit 1 – WRITE_MODE Write Mode

Value	Description
0	The write operation is controlled by the NCS signal. – If TDF optimization is enabled (TDF_MODE =1), TDF wait states will be inserted after the setup of NCS.
1	The write operation is controlled by the NWE signal. – If TDF optimization is enabled (TDF_MODE =1), TDF wait states will be inserted after the setup of NWE.

Bit 0 – READ_MODE Read Mode

Value	Description
0	The read operation is controlled by the NCS signal. – If TDF cycles are programmed, the external bus is marked busy after the rising edge of NCS. – If TDF optimization is enabled (TDF_MODE =1), TDF wait states are inserted after the setup of NCS.
1	The read operation is controlled by the NRD signal. – If TDF cycles are programmed, the external bus is marked busy after the rising edge of NRD. – If TDF optimization is enabled (TDF_MODE =1), TDF wait states are inserted after the setup of NRD.

30.13.5 HSMC Write Protection Mode Register

Name: HSMC_WPMR
Offset: 0x70
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPEN
Access								R/W
Reset								0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x534D43	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

Bit 0 – WPEN Write Protect Enable

See 30.7 Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x534D43 (“SMC” in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x534D43 (“SMC” in ASCII).

30.13.6 HSMC Write Protection Status Register

Name: HSMC_WPSR
Offset: 0x74
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	WPVSRC[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 23:8 – WPVSRC[15:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protection violation has occurred since the last read of the HSMC_WPSR register.
1	A write protection violation has occurred since the last read of the HSMC_WPSR register. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

31. Hardened SDRAM Controller (HSDRAMC)

31.1 Description

The Hardened SDRAM Controller (HSDRAMC) is part of the Hardened External Memory Controller (HEMC). The HEMC is designed to ensure the successful data transfer between several external devices and the ARM-based microcontroller.

The HSDRAMC extends the memory capabilities of a chip by providing the interface to external 16-bit and 32-bit SDRAM devices. The page size supports ranges from 2048 to 8192 and the number of columns from 256 to 4096. It supports byte (8-bit), half-word (16-bit) and word (32-bit) accesses.

The HSDRAMC supports a read or write burst length of one location. It keeps track of the active row in each bank, thus maximizing SDRAM performance, e.g., the application may be placed in one bank and data in the other banks. For optimized performance, it is advisable to avoid accessing different rows in the same bank.

The HSDRAMC supports a CAS latency of 1, 2 or 3 and optimizes the read access depending on the frequency.

31.2 Embedded Characteristics

- Customizable Address Space per Chip Select (from 8 Kbytes to 512 Mbytes)
- Numerous Configurations Supported
  - 2K, 4K, 8K, 16K row address memory parts
  - SDRAM with four internal banks
  - SDRAM with 16-bit and 32-bit data paths
- Word, Halfword, Byte Transfers
- Programming Facilities
  - Automatic Page break when memory boundary has been reached
  - Multibank ping-pong access
  - Timing parameters specified by software
  - Automatic refresh operation, refresh rate is programmable
- Error Detection
  - HECC protection can be enabled /disabled: Hamming(32,7) code implemented
  - Refresh error interrupt
- HSDRAM Power-up Initialization by Software
- CAS Latency of 1, 2, 3 Supported
- Auto Precharge Command Not Used

31.3 I/O Lines Description

Table 31-1. I/O Lines

Signal Name	Signal Direction	Signal Width	Signal Polarity	Signal Description
Address				
A[13:0]	Out	14		Address Bit
SDA10	Out	1		SDRAM Address 10 Line
Data				
D[31:0]	In/Out	32		Data Bits

.....continued				
Signal Name	**Signal Direction**	**Signal Width**	**Signal Polarity**	**Signal Description**
CB[7:0]	In/Out	8		Check Bits
Controls/ Status				
NCS[5:0]	Out	6	Low	Chip Select Lines Signals[see Note]
BA[1:0]	Out	1		HSDRAM Bank Address Signals
SDNBS[4:0]	Out	5	Low	HSDRAM Byte Mask Signals
SDNWE	Out	1	Low	HSDRAM Write Enable Signal
NCAS	Out	1	low	HSDRAM Column Signals
NRAS	Out	1	low	HSDRAM Row Signals
SDCK	Out	1		HSDRAM Clock Signal
SDCKE	Out	1	High	HSDRAM Clock Enable Signal

Note: For details, refer to the section "Hardened Embedded Memory Controller (HEMC)".

31.4 Software Interface/SDRAM Organization, Address Mapping

The HSDRAM address space is organized into banks, rows, and columns. The HSDRAMC allows mapping different memory types according to the values set in the Configuration register (HSDRAMC_CR).

The HSDRAMC makes the SDRAM device access protocol transparent to the user.

The tables below illustrate the SDRAM device memory mapping seen by the user in correlation with the device structure. Various configurations are illustrated.

31.4.1 SDRAM Address Mapping for 32-bit Memory Data Bus Width

Table 31-2. SDRAM Configuration Mapping: 2K Rows, 256/512/1024/2048/4096 Columns

CPU Address Line																														
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
									BK[1:0]		Row[10:0]											Column[7:0]								M[1:0]	
								BK[1:0]			Row[10:0]											Column[8:0]								M[1:0]	
							BK[1:0]				Row[10:0]											Column[9:0]								M[1:0]	
						BK[1:0]					Row[10:0]											Column[10:0]								M[1:0]	
					BK[1:0]						Row[10:0]											Column[11:0]								M[1:0]	

Table 31-3. SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048/4096 Columns

CPU Address Line																														
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
								BK[1:0]			Row[11:0]											Column[7:0]								M[1:0]	
							BK[1:0]				Row[11:0]											Column[8:0]								M[1:0]	
						BK[1:0]					Row[11:0]											Column[9:0]								M[1:0]	
					BK[1:0]						Row[11:0]											Column[10:0]								M[1:0]	

				BK[1:0]	Row[11:0]	Column[11:0]	M[1:0]

Table 31-4. SDRAM Configuration Mapping: 8K Rows, 256/512/1024/2048/4096 Columns

CPU Address Line																													
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
							BK[1:0]	Row [12:0]														Column[7:0]								M[1:0]	
						BK[1:0]	Row [12:0]															Column[8:0]								M[1:0]	
					BK[1:0]	Row [12:0]																Column[9:0]								M[1:0]	
				BK[1:0]	Row [12:0]																	Column[10:0]								M[1:0]	
			BK[1:0]	Row [12:0]																		Column[11:0]								M[1:0]	

Table 31-5. SDRAM Configuration Mapping: 16K Rows, 256/512/1024/2048/4096 Columns

CPU Address Line																													
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
						BK[1:0]	Row [13:0]															Column[7:0]								M[1:0]	
					BK[1:0]	Row [13:0]																Column[8:0]								M[1:0]	
				BK[1:0]	Row [13:0]																	Column[9:0]								M[1:0]	
			BK[1:0]	Row [13:0]																		Column[10:0]								M[1:0]	
		BK[1:0]	Row [13:0]																			Column[11:0]								M[1:0]	

Note:

1. M[1:0] is the byte address inside a 32-bit word.
2. Bk[1] = BA1, Bk[0] = BA0.

31.4.2 SDRAM Address Mapping for 16-bit Memory Data Bus Width

Table 31-6. SDRAM Configuration Mapping: 2K Rows, 256/512/1024/2048/4096 Columns

CPU Address Line																													
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
										BK[1:0]		Row[10:0]										Column[7:0]								M[0]	
									BK[1:0]		Row[10:0]											Column[8:0]								M[0]	
								BK[1:0]		Row[10:0]												Column[9:0]								M[0]	
							BK[1:0]		Row[10:0]													Column[10:0]								M[0]	
						BK[1:0]		Row[10:0]														Column[11:0]								M[0]	

Table 31-7. SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048/4096 Columns

CPU Address Line																													
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
									BK[1:0]		Row[11:0]											Column[7:0]								M[0]	
								BK[1:0]		Row[11:0]												Column[8:0]								M[0]	
							BK[1:0]		Row[11:0]													Column[9:0]								M[0]	
						BK[1:0]		Row[11:0]														Column[10:0]								M[0]	

					BK[1:0]	Row[11:0]															Column[11:0]										M[0]

Table 31-8. SDRAM Configuration Mapping: 8K Rows, 256/512/1024/2048/4096 Columns

CPU Address Line																															
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
								BK[1:0]		Row[12:0]													Column[7:0]								M[0]
							BK[1:0]		Row[12:0]														Column[8:0]								M[0]
						BK[1:0]		Row[12:0]																Column[9:0]							M[0]
					BK[1:0]		Row[12:0]																		Column[10:0]						M[0]
				BK[1:0]		Row[12:0]																				Column[11:0]					M[0]

Table 31-9. SDRAM Configuration Mapping: 16K Rows, 256/512/1024/2048/4096 Columns

CPU Address Line																															
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
							BK[1:0]		Row[13:0]														Column[7:0]								M[0]
						BK[1:0]		Row[13:0]																Column[8:0]							M[0]
					BK[1:0]		Row[13:0]																		Column[9:0]							M[0]
				BK[1:0]		Row[13:0]																				Column[10:0]						M[0]
			BK[1:0]		Row[13:0]																					Column[11:0]						M[0]

Note:

1. M0 is the byte address inside a 16-bit half-word.
2. Bk[1] = BA1, Bk[0] = BA0.

31.5 Product Dependencies

31.5.1 I/O Lines

The pins used for interfacing the HSDRAMC are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the HSDRAMC pins to their peripheral function. If I/O lines of the HSDRAMC are not used by the application, they can be used for other purposes by the PIO Controller.

31.5.2 Power Management

The HSDRAMC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the HSDRAMC clock.

The HSDRAM clock on pin SDCK is output as soon as the first access to the HSDRAM is made during the initialization phase.

The HSDRAMC clock must respect the following range:

- HSDRAMC_Min=MCK
- HSDRAMC_Max=4 * MCK

31.6 Interrupt Sources

The HSDRAMC interrupt is ORed internally to the HEMC interrupt. To enable it, ensure that the HEMC interrupt line is enabled.

31.7 Functional Description

31.7.1 SDRAM Device Initialization

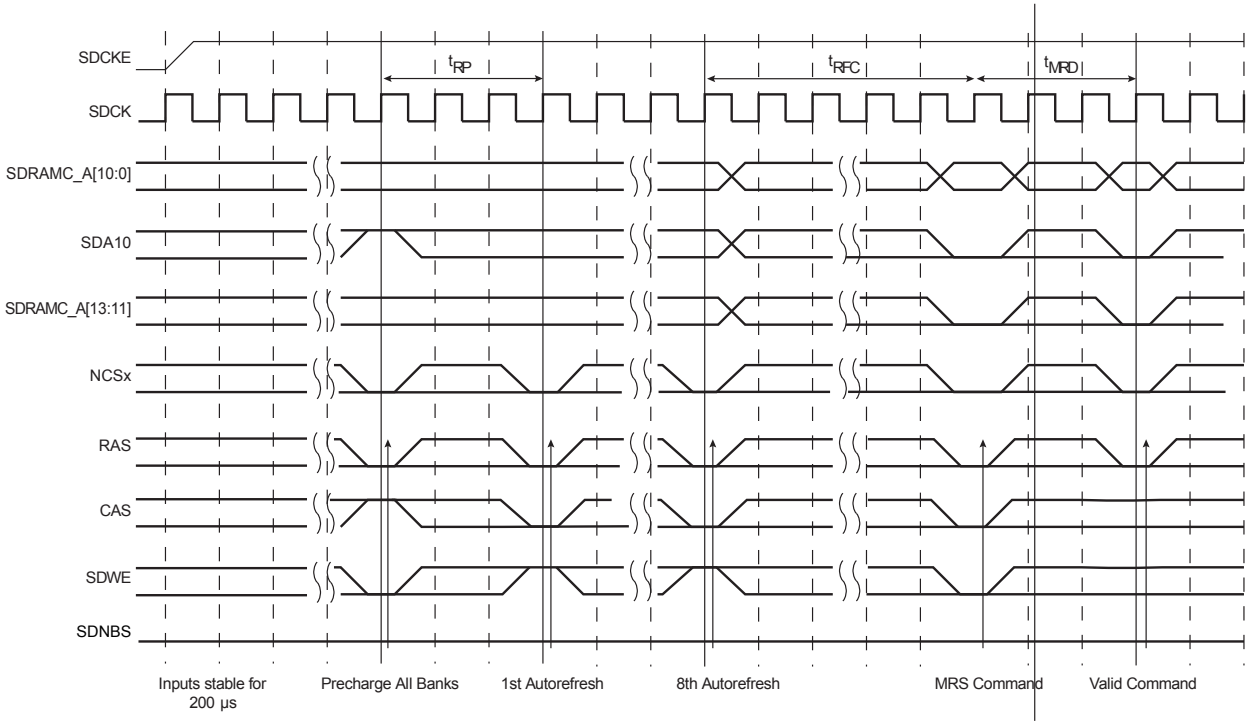
The initialization sequence is generated by software. The sequence to initialize SDRAM devices is the following:

1. Set the SDRAM features in the HSDRAMC_CR and HSDRAMC_SDR: asynchronous timings (TRC, TRAS, etc.), number of columns, number of rows, CAS latency, RMW feature and data bus width. Set UNAL bit in HSDRAMC_CFR1.
2. A pause of at least 200 µs must be observed before a signal toggle.
3. (1) A NOP command is issued to the SDRAM devices. The application must write a 1 to the MODE field in the Mode register (HSDRAMC_MR). Read the HSDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any SDRAM address.
4. An All Banks Precharge command is issued to the SDRAM. The application must write a 2 to the MODE field in the HSDRAMC_MR. Read the HSDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any SDRAM address.
5. Eight autorefresh (CBR) cycles are provided. The application must set the MODE field to 4 in the HSDRAMC_MR. Read the HSDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any SDRAM location eight times.
6. A Mode Register set (MRS) cycle is issued to program the parameters of the SDRAM, in particular CAS latency and burst length. The application must write a 3 to the MODE field in the HSDRAMC_MR. Read the HSDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the SDRAM. The write address must be chosen so that BA[1:0] are set to 0. For example, with a 16-bit 128 MB SDRAM (12 rows, 9 columns, 4 banks) bank address, the SDRAM write access should be done at the starting address of memory (0x7000_0000 as example)
7. The application must go into Normal mode. Configure MODE to 0 in the HSDRAMC_MR. Read the HSDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access at any location in the SDRAM.
8. Write the refresh rate into the COUNT field in the Refresh Timer register (HSDRAMC_TR). (Refresh rate = delay between refresh cycles). The SDRAM device requires a refresh every 15.625 µs or 7.81 µs. With a 100 MHz frequency, the Refresh Timer register must be set with the value 1562 (15.625 µs x 100 MHz) or 781 (7.81 µs x 100 MHz).
   After initialization, the SDRAM devices are fully functional.

Note:

1. The instructions stated in Step 3 of the initialization process must be respected to make sure the subsequent commands issued by the HSDRAMC are taken into account.

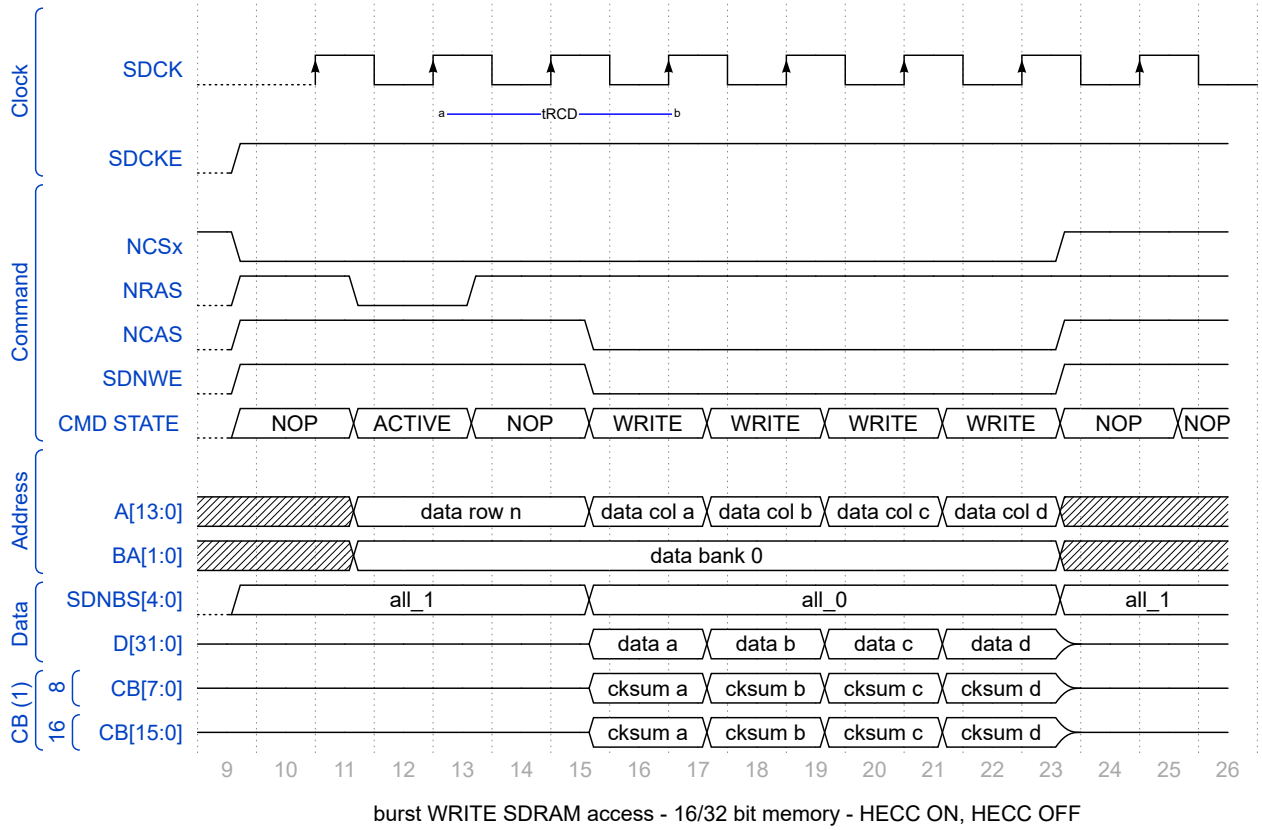
Figure 31-1. HSDRAM Device Initialization Sequence



31.7.2 Write Cycle

The HSDRAMC allows burst access or single access. In both cases, the HSDRAMC keeps track of the active row in each bank, thus maximizing performance. To initiate a burst access, the HSDRAMC uses the transfer type signal provided by the master requesting the access. If the next access is a sequential write access, writing to the SDRAM device is carried out. If the next access is a write-sequential access, but the current access is to a boundary page, or if the next access is in another row, then the HSDRAMC generates a precharge command, activates the new row and initiates a write command. To comply with SDRAM timing parameters, additional clock cycles are inserted between precharge and active commands (t_{RP}), and between active and write commands (t_{RCD}). For the definition of these timing parameters, refer to the HSDRAMC Configuration Registers (HSDRAMC_CR and HSDRAMC_SDR).

Figure 31-2. Write Burst SDRAM Access (16-bit/32-bit mode)



burst WRITE SDRAM access - 16/32 bit memory - HECC ON, HECC OFF

Note:

1. When HECC is ON, the CB buses are active (CB 8 in the case of Hamming(32,7).
2. In the case of 16-bit mode, D[31:0] is replaced by D[15:0].

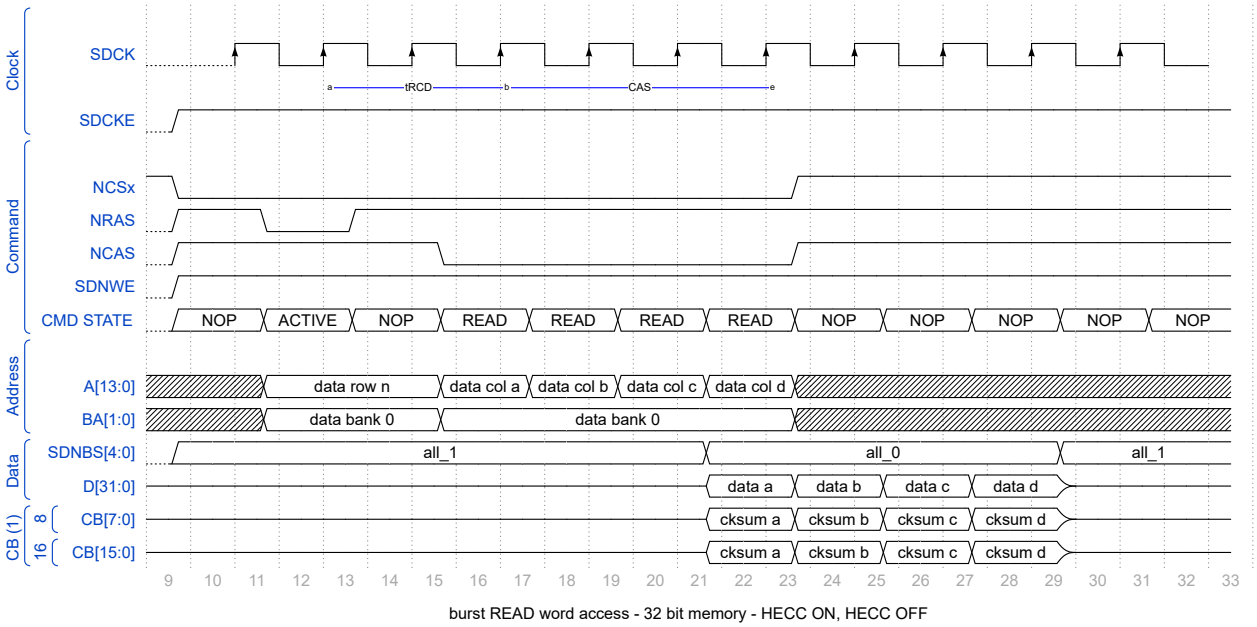
31.7.3 Read Cycle

The HSDRAMC allows burst access, incremental burst of unspecified length or single access. In all cases, the HSDRAMC keeps track of the active row in each bank, thus maximizing performance of the SDRAM. If row and bank addresses do not match the previous row/bank address, then the HSDRAMC automatically generates a precharge command, activates the new row and starts the read command. To comply with the SDRAM timing parameters, additional clock cycles on SDCK are inserted between precharge and active commands (t_{RP}), and between active and read commands (t_{RCD}). These two parameters are set in the HSDRAMC_SDR. After a read command, additional wait states are generated to comply with the CAS latency (1, 2 or 3 clock delays specified in the HSDRAMC_CR).

For a single access or an incremented burst of unspecified length, the HSDRAMC anticipates the next access. While the last value of the column is returned by the HSDRAMC on the bus, the HSDRAMC anticipates the read to the next column and thus anticipates the CAS latency. This reduces the effect of the CAS latency on the internal bus.

For burst access of specified length (4, 8, 16 words), access is not anticipated. This case leads to the best performance. If the burst is broken (border, Busy mode, etc.), the next access is handled as an incrementing burst of unspecified length.

Figure 31-3. Read Burst SDRAM Access (16/32-bit mode)



burst READ word access - 32 bit memory - HECC ON, HECC OFF

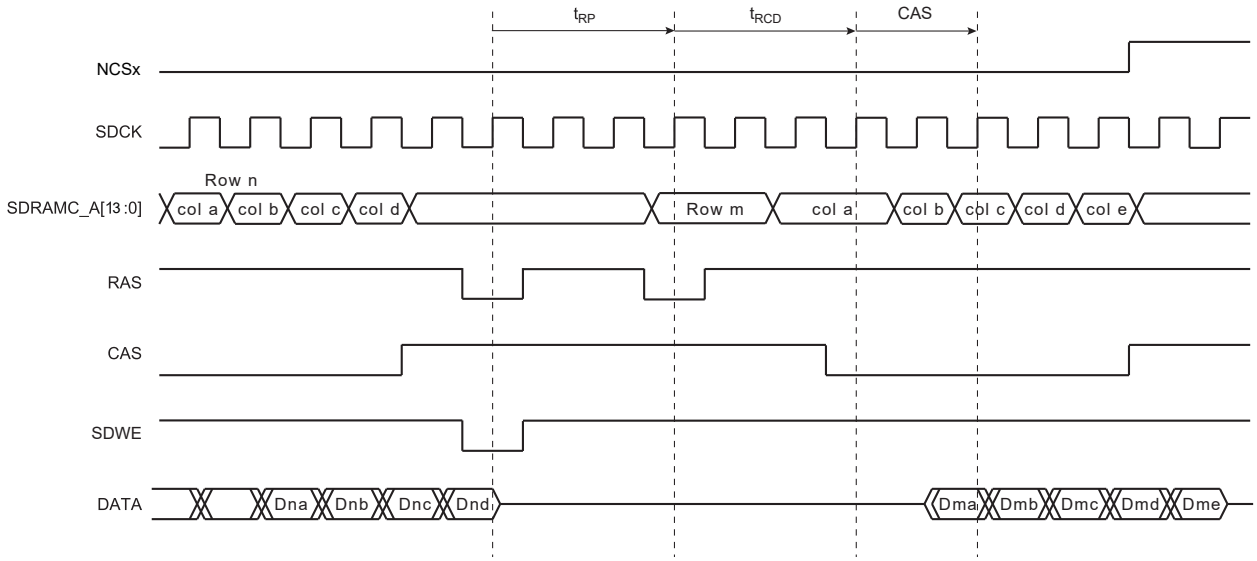
Note:

1. When HECC is ON, the CB buses are active (CB 8 in the case of Hamming(32,7).
2. In the case of 16-bit Mode, D[31:0] is replaced by D[15:0].

31.7.4 Border Management

When the memory row boundary has been reached, an automatic page break is inserted. In this case, the HSDRAMC generates a precharge command, activates the new row and initiates a read or write command. To comply with SDRAM timing parameters, an additional clock cycle is inserted between the precharge and the active command (t_{RP}) and between the active and the read command (t_{RCD}). Refer to the figure below.

Figure 31-4. Read Burst with Boundary Row Access

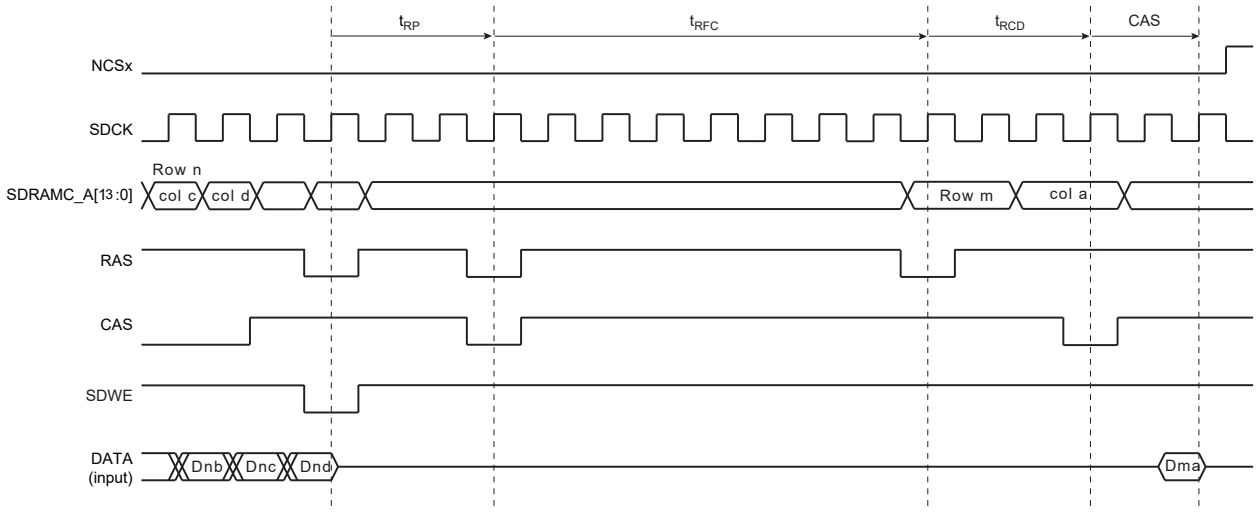


31.7.5 SDRAM Controller Refresh Cycles

An autorefresh command is used to refresh the SDRAM device. Refresh addresses are generated internally by the SDRAM device and incremented after each autorefresh automatically. The HSDRAMC generates these autorefresh commands periodically. An internal timer is loaded with the value in HSDRAMC_TR that indicates the number of

clock cycles between refresh cycles. A refresh error interrupt is generated when the previous autorefresh command did not perform. It is acknowledged by reading the Interrupt Status register (HSDRAMC_ISR). When the HSDRAMC initiates a refresh of the SDRAM device, internal memory accesses are not delayed. However, if the processor tries to access the SDRAM, the slave indicates that the device is busy and the master is held by a wait signal. Refer to the figure below.

Figure 31-5. Refresh Cycle Followed by a Read Access



31.8 Read/Write Access Use Cases

The following external accesses are visible depending on the HSDRAMC configuration and the different AMBA requests.

HSDRAMC configurations are as follows:

- Data bus width mode selected (16-bit or 32-bit)
- The activation or the de-activation of the HECC and the selection of the Error Correction Code (HECC) applied (Hamming (32,7)
- The activation or de-activation of the RMW feature

The AMBA requests on the 32-bit processor are as follows:

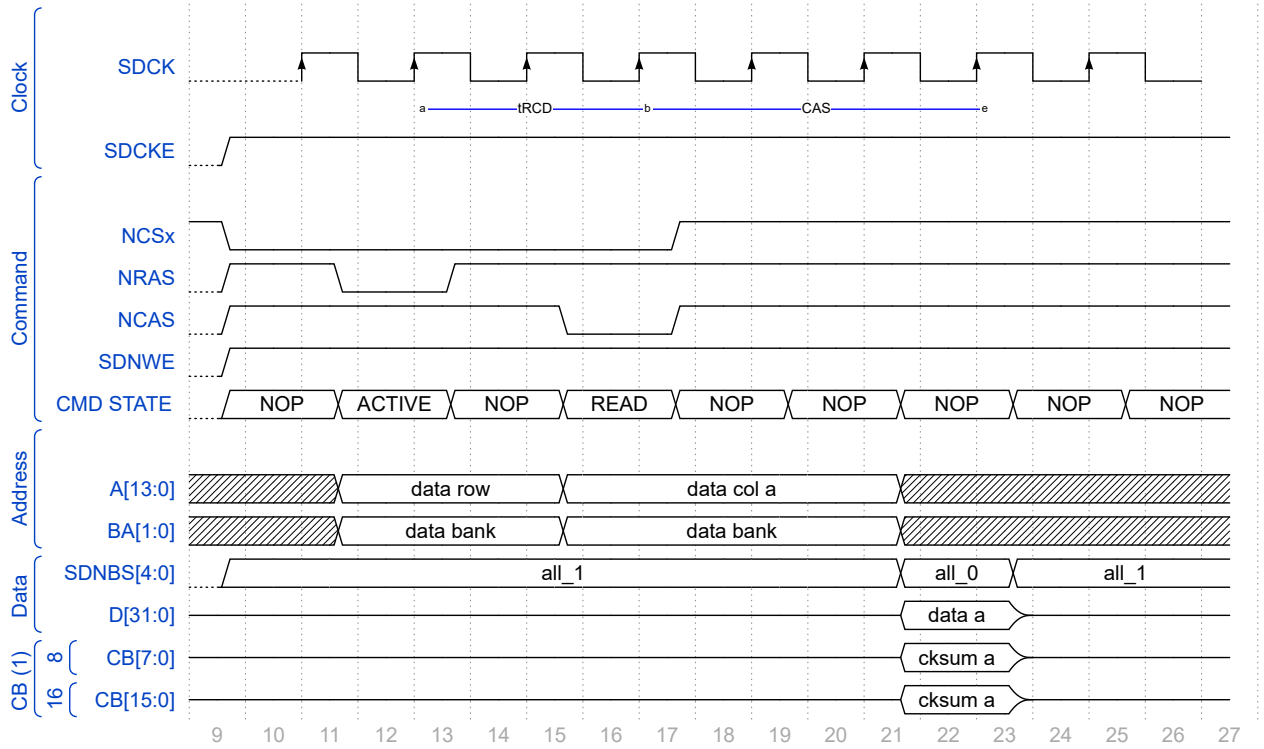
- Write 8 bits (WR_byte), Write 16 bits (WR_half_word), Write 32 bits (WR_word), Write n x 32 bits (WR_n_word)
- Read 8 bits (RD_byte), Read 16 bits (RD_half_word), Read 32 bits (RD_word), Read n x 32 bits (RD_n_word)

31.8.1 32-bit Mode

31.8.1.1 Single RD 32-bit Access

The figure below presents a single 32-bit read access in the case of a 32-bit wide data bus and HECC is OFF/ON. This is referred to as ‘RD Burst of 1 x 32 bits’ in this document.

Figure 31-6. RD Burst 1 x 32 bits – 32 bits wide, HECC OFF/ON



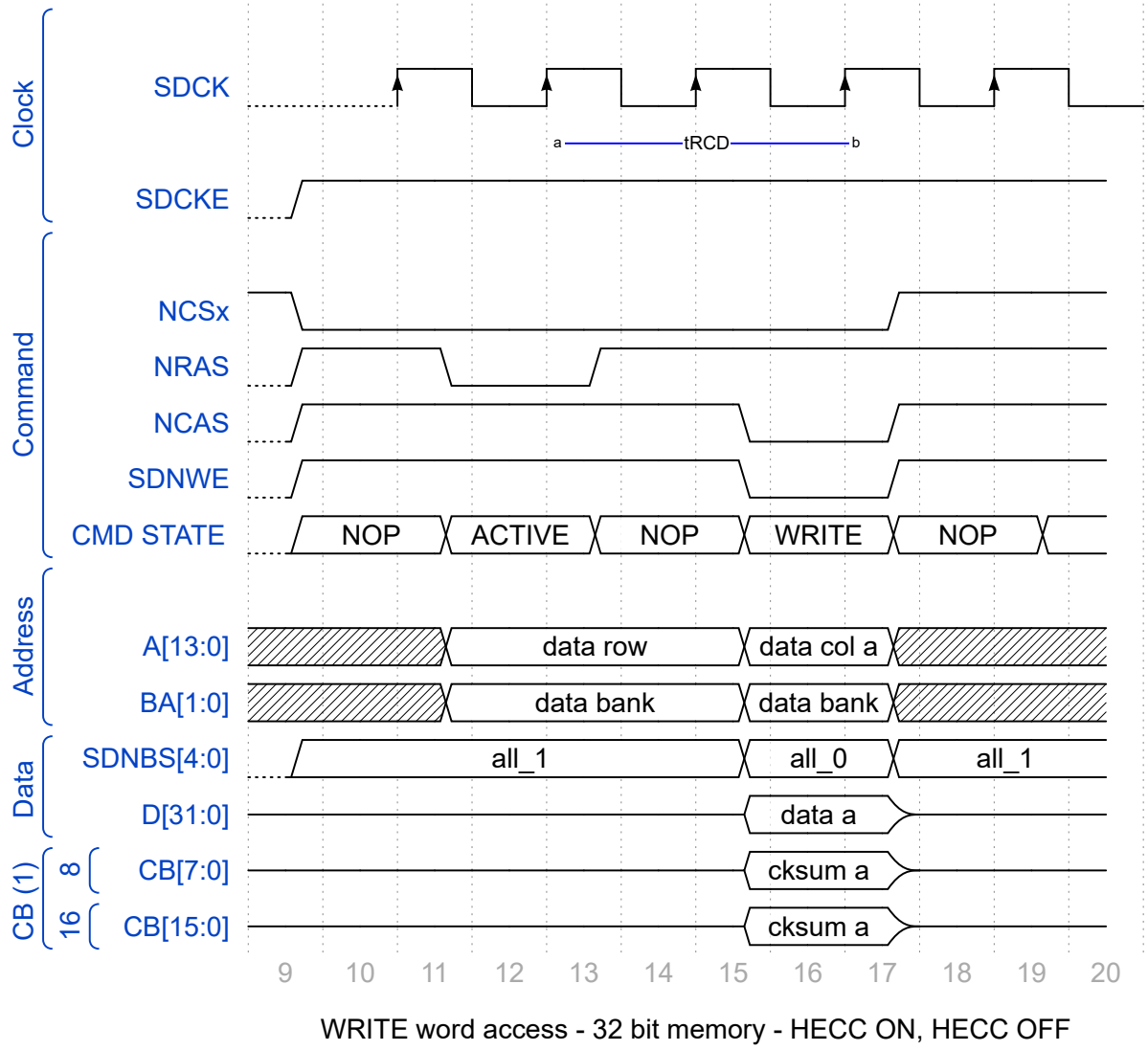
READ word access - 32 bit memory - HECC ON, HECC OFF

Note: When HECC is ON, the CB buses are active (CB 8 in the case of Hamming(32,7). Otherwise, they are not active.

31.8.1.2 Single WR 32-bit Access

The figure below presents a single 32-bit write access in the case of a 32-bit wide data bus and HECC is OFF/ON. This is referred to as ‘WR Burst of 1 x 32 bits’ in this document.

Figure 31-7. WR burst 1 x 32 bits – 32 bits wide, HECC OFF/ON



Note: When HECC is ON, the CB buses are active (CB 8 in the case of Hamming(32,7). Otherwise, they are not active.

31.8.1.3 HECC Protection Deactivated (32-bit mode)

The table below summarizes the different accesses visible on the external memory interface depending on the AMBA request, when 32-bit mode is selected and HECC protection is de-activated.

- RD Burst of 1 x 32 bits refers to Figure 31-6 without CB signals
- RD Burst of n x 32 bits refers to Figure 31-3 without CB signals
- WR Burst of 1 x 32 bits refers to Figure 31-7 without CB signals
- WR Burst of n x 32 bits refers to Figure 31-2 without CB signals

Table 31-10. Memory Access – 32-bit Mode – HECC OFF

AMBA Request	[HSDRAMC] [32-bit] [HECC OFF] [RMW OFF]	[HSDRAMC] [32-bit] [HECC OFF] [RMW ON]
RD_byte RD_half_word	RD Burst of 1 x 32 bits (1)	RD Burst of 1 x 32 bits(1)
WR_byte WR_half_word	WR Burst of 1 x 32 bits(1)	(RMW)(2)
RD_n_word (n≥1)	RD Burst of n x 32 bits	RD Burst of n x 32 bits
WR_n_word (n≥1)	WR Burst of n x 32 bits	WR Burst of n x 32 bits

Note:

1. SDNBS [4:0] will be used in order to access only significant bytes.
2. (RMW)=[(1 x RD burst of 1x 32 bits) then [(1x WR burst of 1 x 32 bits)].

31.8.1.4 HECC Protection Activated (40-bit Mode, resp. 48-bit Mode)

The table below summarizes the different access visible on the external memory interface, depending on the AMBA request received, when 32-bit mode is selected and the HECC protection is activated.

- (RD Burst of 1 x 32 bits) (//) (RD Burst of 1 x 8 bits) refers to Figure 31-6 with CB 8 signals used
- (RD Burst of 1 x 32 bits) (//) (RD Burst of 1 x 16 bits) refers to Figure 31-6 with CB 16 signals used
- (RD Burst of n x 32 bits) (//) (RD Burst of n x 8) bits refers to Figure 31-3 with CB 8 signals used
- (RD Burst of n x 32 bits) (//) (RD Burst of n x 16 bits) refers to Figure 31-3 with CB 16 signals used
- (WR Burst of 1 x 32 bits) (//) (WR Burst of 1 x 8 bits) refers to Figure 31-7 with CB 8 signals used
- (WR Burst of 1 x 32 bits) (//) (WR Burst of 1 x 16 bits) refers to Figure 31-7 with CB 16 signals used
- (WR Burst of n x 32 bits) (//) (WR Burst of n x 8 bits) refers to Figure 31-2 with CB 8 signals used
- (WR Burst of n x 32 bits) (//) (WR Burst of n x 16 bits) refers to Figure 31-2 with CB 16 signals used

Table 31-11. Memory Access – 32-bit Mode – HECC ON

AMBA Request	HSDRAMC [32-bit] [HECC ON – Hamming(32,7)] [RMW ON] [data][sequence][checksum]
RD_byte RD_half_word	(RD Burst of 1 x 32 bits) (//) (RD Burst of 1 x 8 bits)
WR_byte WR_half_word	(RMW_ECCON8) (1)
RD_n_word ((n≥1))	n x [(RD Burst of 1 x 32 bits) (//) (RD Burst of 1 x 8 bits)]
WR_n_word ((n≥1))	(WR Burst of n x 32 bits) (//) (WR Burst of n x 8 bits)

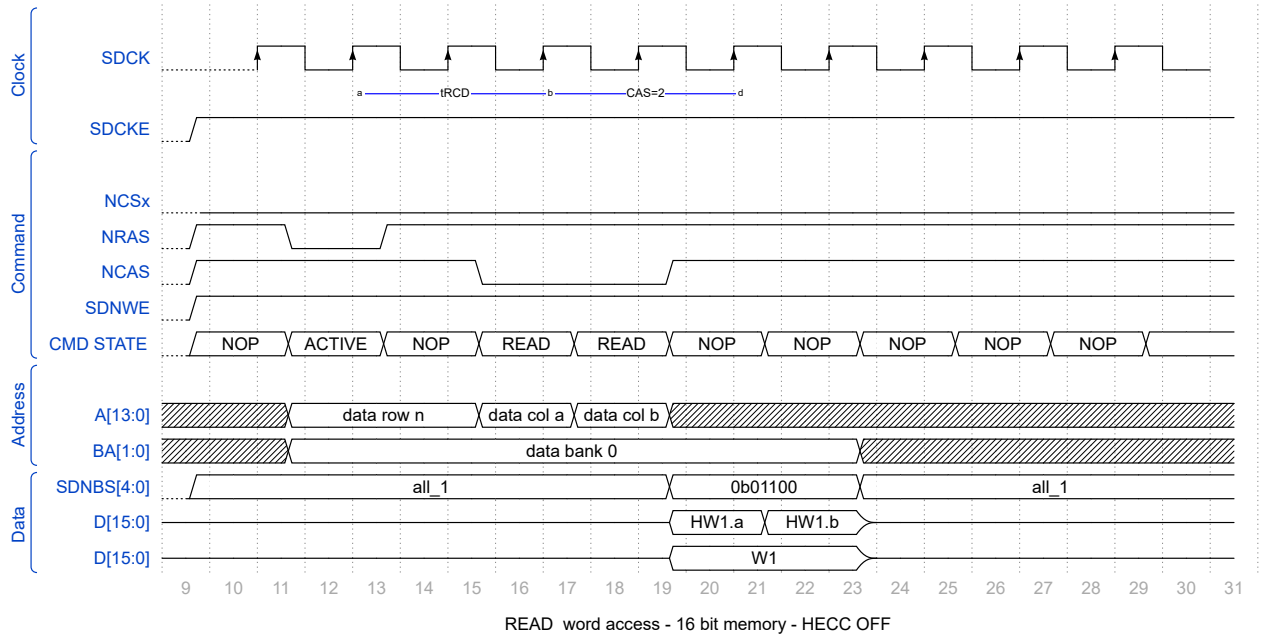
Note:

1. (RMW_ECCON8)= [(RD Burst of 1 x 32 bits) (//) (RD Burst of 1 x 8 bits)]+ [(WR Burst of 1 x 32 bits) (//) (WR Burst of 1 x 8 bits)]

31.8.2 16-bit Mode

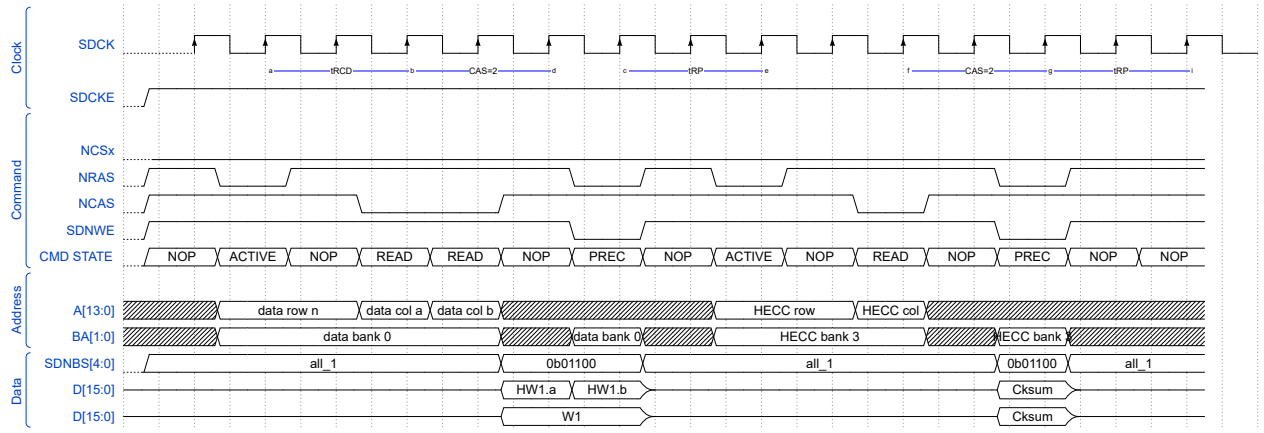
31.8.2.1 Single RD 32-bit Access

Figure 31-8. RD burst 1 x 32 bits – 16 bits wide, HECC OFF



READ word access - 16 bit memory - HECC OFF

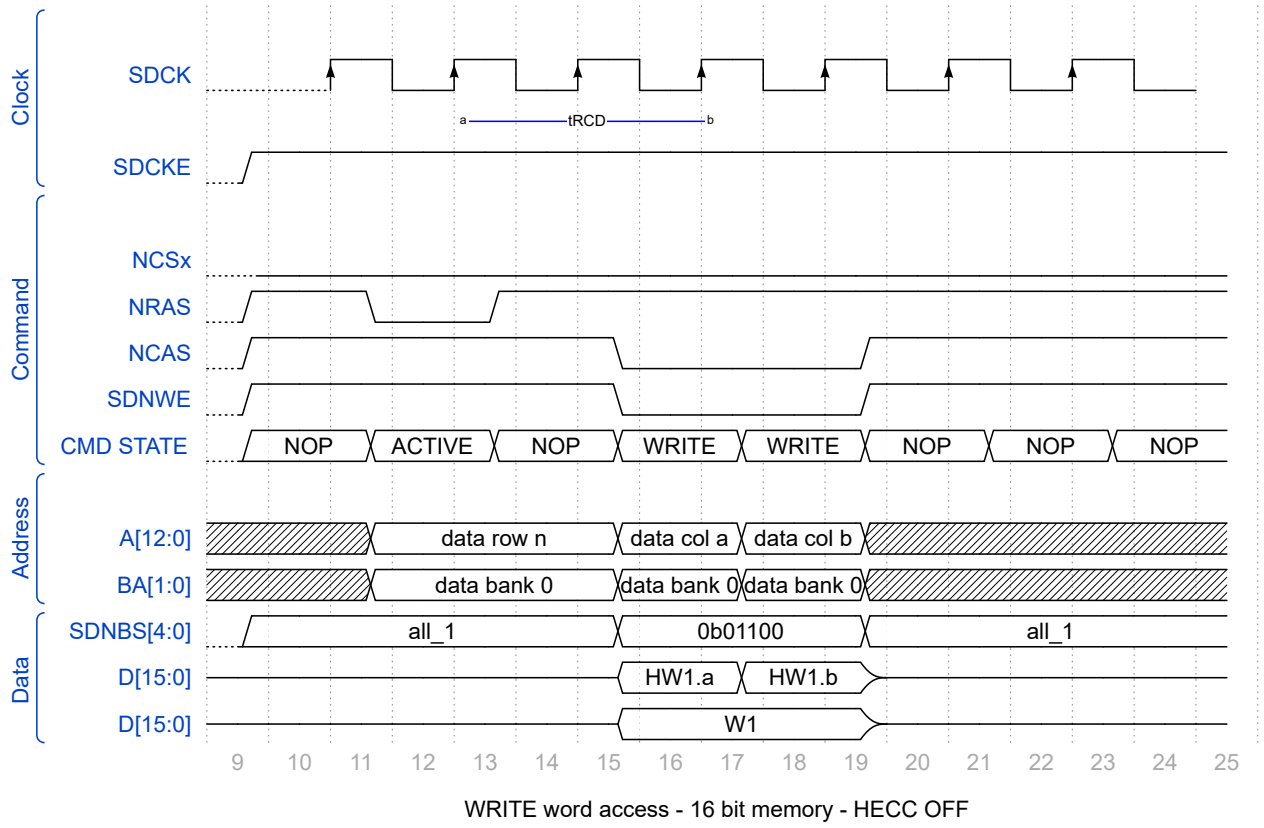
Figure 31-9. RD burst 1 x 32 bits – 16 bits wide, HECC ON



READ word access - 16 bit memory - HECC ON

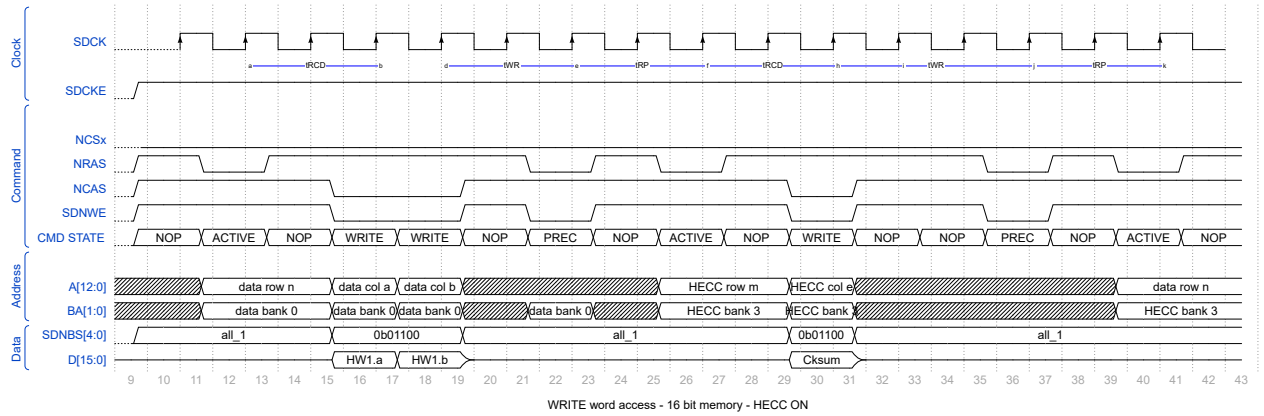
31.8.2.2 Single WR 32-bit Access

Figure 31-10. WR Burst 1 x 32 bits – 16 bits wide, HECC OFF



WRITE word access - 16 bit memory - HECC OFF

Figure 31-11. WR Burst 1 x 32 bits – 16 bits wide, HECC ON

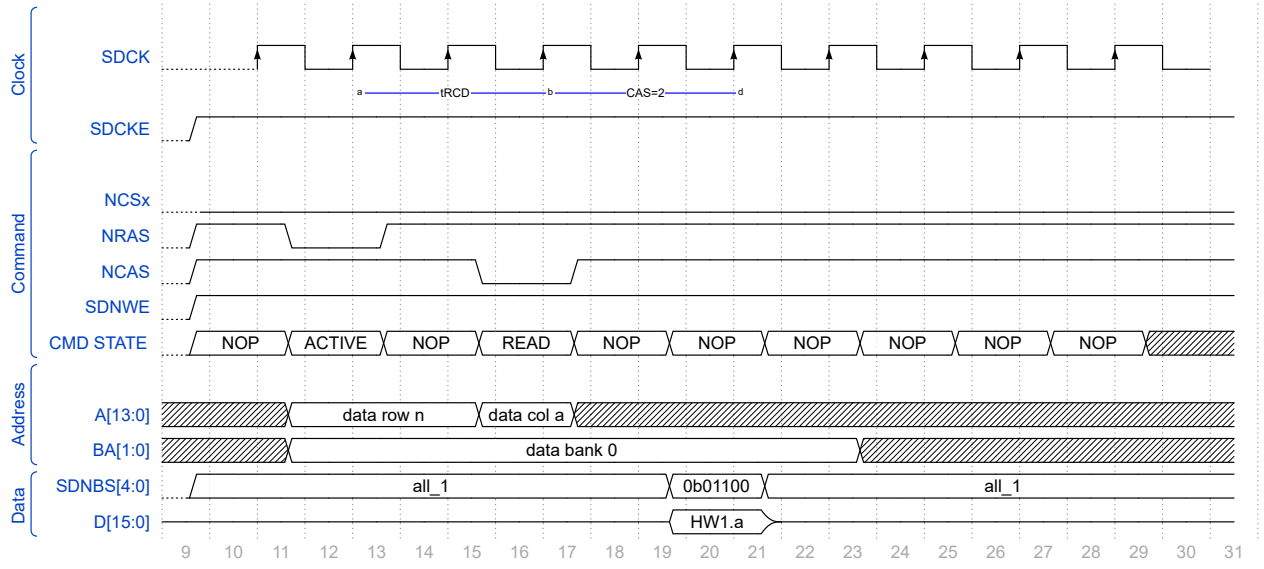


WRITE word access - 16 bit memory - HECC ON

31.8.2.3 HECC Protection Deactivated

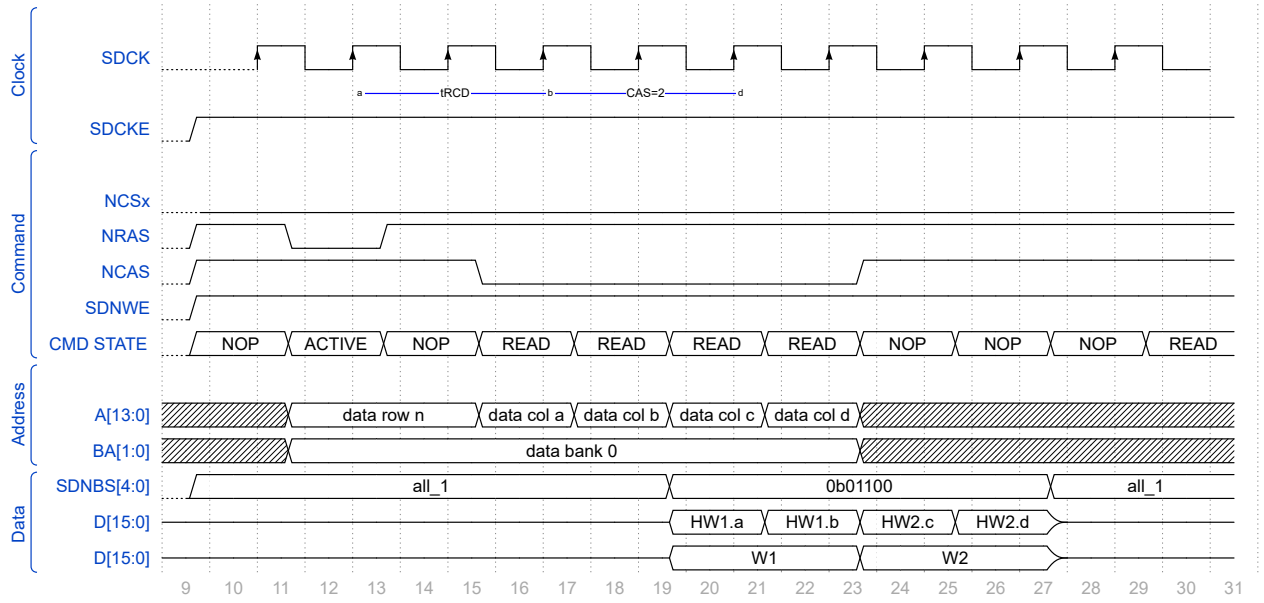
The following figures present the unitary waveforms used to constitute the whole access performed.

Figure 31-12. RD Burst of 1 x 16 bits



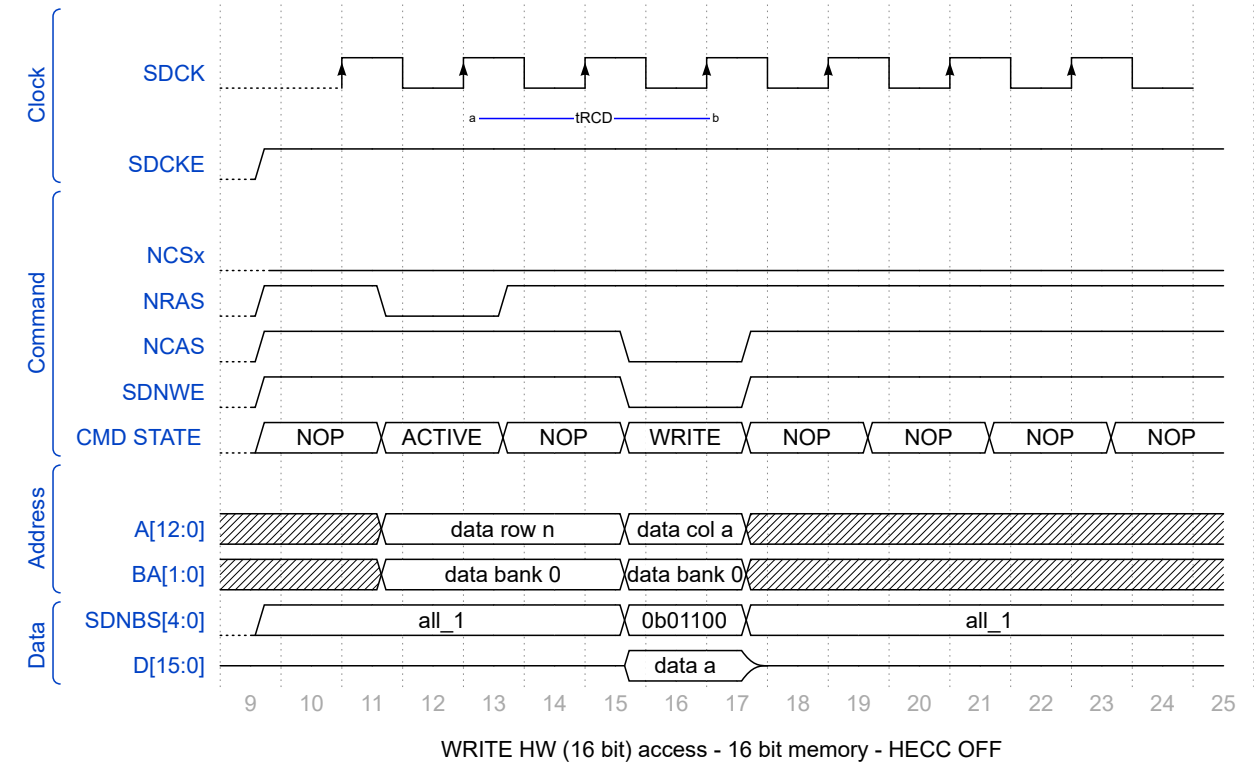
READ HW (16bit) access - 16 bit memory - HECC OFF

Figure 31-13. RD Burst of 2n x 16 bits



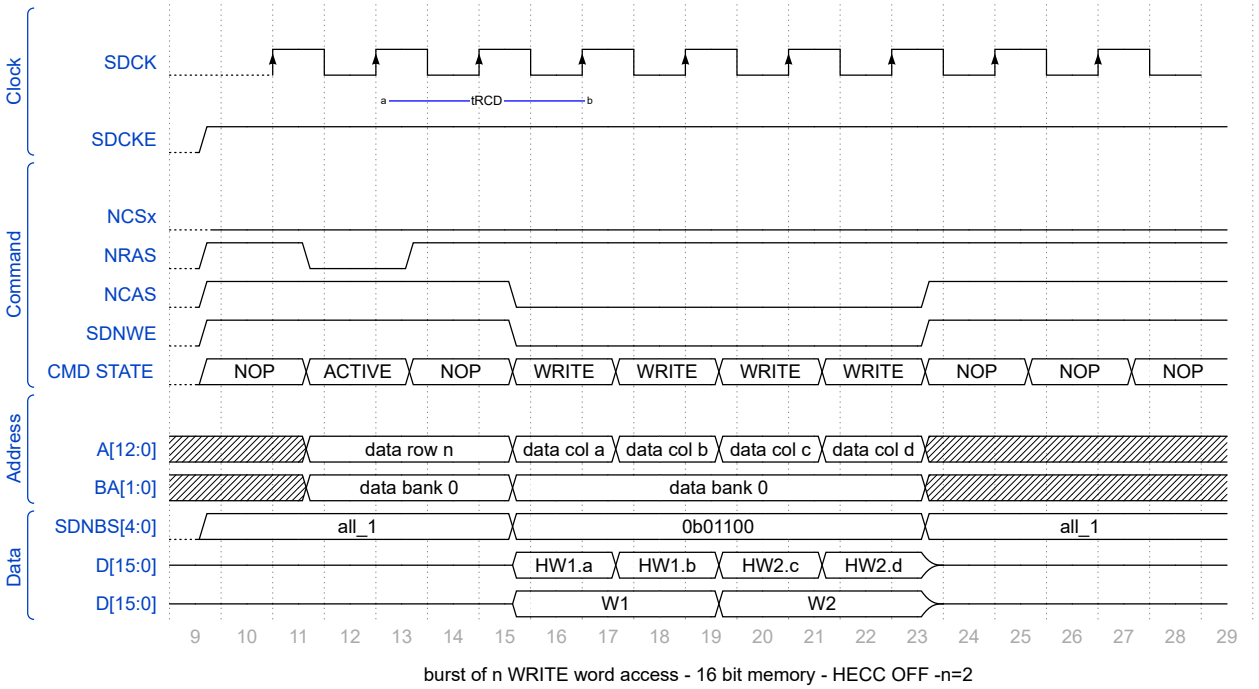
burst of n READ word access - 16 bit memory - HECC OFF -n=2

Figure 31-14. WR Burst of 1 x 16 bits



WRITE HW (16 bit) access - 16 bit memory - HECC OFF

Figure 31-15. WR Burst of 2n x 16 bits



burst of n WRITE word access - 16 bit memory - HECC OFF -n=2

The table below summarizes the different accesses visible on the external memory interface depending on the AMBA request, when 16-bit mode is selected and HECC protection is deactivated.

- RD Burst of 1 x 16 bits refers to Figure 31-12
- RD Burst of 2n x 16 bits refers to Figure 31-13

- WR Burst of 1 x 16 bits refers to Figure 31-14
- WR Burst of 2n x 16 bits refers to Figure 31-15

Table 31-12. Memory Access – 16-bit Mode – HECC OFF

AMBA Request	HSDRAMC [16-bit] [HECC OFF] [RMW OFF]	HSDRAMC [16-bit] [HECC OFF] [RMW ON]
RD_byte	(RD Burst of 1 x 16 bits) (1)	(RD Burst of 1 x 16 bits)
WR_byte	(WR Burst of 1 x 16 bits) (1)	(RMW) (2)
RD_half_word	(RD Burst of 1 x 16 bits)	(RD Burst of 1 x 16 bits)
WR_half_word	(WR Burst of 1 x 16 bits)	(RMW) (2)
RD_n_word, (n≥1)	(RD Burst of 2n x 16 bits)	(RD Burst of 2n x 16 bits)
WR_n_word, (n≥1)	(WR Burst of 2n x 16 bits)	(WR Burst of 2n x 16 bits)

Note:

1. SDNBS [4:0] will be used in order to acces only significant bytes.
2. (RMW)= [(RD Burst of 2 x 16 bits) then [(WR Burst of 2 x 16 bits)].

31.8.2.4 HECC Protection Activated

The table below summarizes the different accesses visible on the external memory interface depending on the AMBA request, when 16-bit mode is selected and HECC protection is activated.

- (RD Burst of 2 x 16 bits) (+) (RD Burst of 1 x 16 bits) refers to Figure 31-9
- (WR Burst of 2 x 16 bits) (+) (WR Burst of 1 x 16 bits) refers to Figure 31-11

Table 31-13. Memory Access – 16-bit Mode – HECC ON

AMBA Request	HSDRAMC [16-bit] [HECC ON – Hamming(32,7)] [RMW ON] [data][sequence][checksum]
RD_byte RD_half_word	(RD Burst of 2 x 16 bits) (+) (RD Burst of 1 x 16 bits)
WR_byte WR_half_word	(RMW_ECCON) (1)
RD_n_word (n≥1)	n x [(RD Burst of 2 x 16 bits) (+) (RD Burst of 1 x 16 bits)]
WR_n_word (n≥1)	n x [(WR Burst of 2 x 16 bits) (+) (WR Burst of 1 x 16 bits)]

Note:

1. (RMW_ECCON)= [(RD Burst of 2 x 16 bits) (+) (RD Burst of 1 x 16 bits) then (WR Burst of 2 x 16 bits) (+) (WR Burst of 1 x 16 bits)]

31.9 Register Write Protection

To prevent any single software error from corrupting HSDRAMC behavior, some registers in the address space can be write-protected by setting the WPEN bit in the HSDRAMC Write Protection Mode Register (HSDRAMC_WPMR). If a write access to a write-protected register is detected, the WPVS flag in the HSDRAMC Write Protection Status Register (HSDRAMC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the HSDRAMC_WPSR. The following registers can be write-protected:

- HSDRAMC Mode Register
- HSDRAMC Refresh Timer Register
- HSDRAMC Configuration Registers
- HSDRAMC High-Speed Register
- HSDRAMC Memory Device Register
- HSDRAMC Configuration Register 1

31.10 Register Summary

Offset	Name	Bit Pos.								
0x00	HSDRAMC_MR	7:0						MODE[2:0]		
		15:8								
		23:16								
		31:24								
0x04	HSDRAMC_TR	7:0	COUNT[7:0]							
		15:8					COUNT[11:8]			
		23:16								
		31:24								
0x08	HSDRAMC_CR	7:0	CAS[1:0]		NB	NR[1:0]		NC[2:0]		
		15:8							RMW	DBW
		23:16								
		31:24								
0x0C	HSDRAMC_SDR	7:0	TRC_TRFC[3:0]				TWR[3:0]			
		15:8	TRCD[3:0]				TRP[3:0]			
		23:16	TXSR[3:0]				TRAS[3:0]			
		31:24								
0x10	HSDRAMC_HSR	7:0								DA
		15:8								
		23:16								
		31:24								
0x14	HSDRAMC_IER	7:0								RES
		15:8								
		23:16								
		31:24								
0x18	HSDRAMC_IDR	7:0								RES
		15:8								
		23:16								
		31:24								
0x1C	HSDRAMC_IMR	7:0								RES
		15:8								
		23:16								
		31:24								
0x20	HSDRAMC_ISR	7:0								RES
		15:8								
		23:16								
		31:24								
0x24	HSDRAMC_CFR1	7:0					TMRD[3:0]			
		15:8								UNAL
		23:16								
		31:24								
0x28	HSDRAMC_WPMR	7:0								WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0x2C	HSDRAMC_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16								
		31:24								

31.10.1 HSDRAMC Mode Register

Name: HSDRAMC_MR
Offset: 0x0000
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the SDRAMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						MODE[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 2:0 – MODE[2:0] SDRAMC Command Mode
This field defines the command issued by the SDRAMC when the SDRAM device is accessed.

Value	Name	Description
0	NORMAL	Normal mode. Any access to the SDRAM is decoded normally. To activate this mode, the command must be followed by a write to the SDRAM.
1	NOP	The HSDRAMC issues a NOP command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the SDRAM.
2	ALLBANKS_PRECHARGE	The SDRAMC issues an “All Banks Precharge” command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the SDRAM.
3	LOAD_MODEREG	The HSDRAMC issues a “Load Mode Register” command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the SDRAM.
4	AUTO_REFRESH	The HSDRAMC issues an “Autorefresh” Command when the SDRAM device is accessed regardless of the cycle. Previously, an “All Banks Precharge” command must be issued. To activate this mode, the command must be followed by a write to the SDRAM.
5	EXT_LOAD_MODEREG	The HSDRAMC issues an “Extended Load Mode Register” command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the “Extended Load Mode Register” command must be followed by a write to the SDRAM. The write in the SDRAM must be done in the appropriate bank; most low-power SDRAM devices use the bank 1.
6-7	-	reserved

31.10.2 HSDRAMC Refresh Timer Register

Name: HSDRAMC_TR
Offset: 0x0004
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the SDRAMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					COUNT[11:8]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 11:0 – COUNT[11:0] HSDRAMC Refresh Timer Count
This 12-bit field is loaded into a timer that generates the refresh pulse. Each time the refresh pulse is generated, a refresh burst is initiated. The SDRAM device requires a refresh every 15.625 µs or 7.81 µs. With a 100 MHz frequency, the Refresh Timer Counter Register must be set with the value 1562 (15.625 µs x 100 MHz) or 781 (7.81 µs x 100 MHz).
To refresh the SDRAM device, this 12-bit field must be written. If this condition is not satisfied, no refresh command is issued and no refresh of the SDRAM device is carried out.

31.10.3 HSDRAMC Configuration Register

Name: HSDRAMC_CR
Offset: 0x0008
Reset: 0x000000C0
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the SDRAMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							RMW	DBW
Access							R/W	R/W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	CAS[1:0]		NB	NR[1:0]		NC[2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	0	0	0	0	0	0

Bit 9 – RMW Read Modify Write

Value	Description
0	RMW is disabled.
1	RMW is enabled.

Bit 8 – DBW Data Bus Width
Reset value is 16 bits.

Value	Description
0	Data bus width is 32 bits.
1	Data bus width is 16 bits.

Bits 7:6 – CAS[1:0] CAS Latency
Reset value is two cycles. In the HSDRAMC, only a CAS latency of one, two and three cycles are managed.

Value	Name	Description
0	Reserved	–
1	LATENCY1	1 cycle latency
2	LATENCY2	2 cycle latency
3	LATENCY3	3 cycle latency

Bit 5 – NB Number of Banks
Reset value is two banks.

Value	Name	Description
0	BANK2	2 banks
1	BANK4	4 banks

Bits 4:3 – NR[1:0] Number of Row Bits
Reset value is 11 row bits.

Value	Name	Description
0	ROW11	11 bits to define the row number, up to 2048 rows
1	ROW12	12 bits to define the row number, up to 4096 rows
2	ROW13	13 bits to define the row number, up to 8192 rows
3	Reserved	–

Bits 2:0 – NC[2:0] Number of Column Bits
Reset value is 8 column bits.

Value	Name	Description
0	COL8	8 bits to define the column number, up to 256 columns
1	COL9	9 bits to define the column number, up to 512 columns
2	COL10	10 bits to define the column number, up to 1024 columns
3	COL11	11 bits to define the column number, up to 2048 columns
4	COL12	12 bits to define the column number, up to 4096 columns

31.10.4 HSDRAMC Setup Delay Register

Name: HSDRAMC_SDR
Offset: 0x000C
Reset: 0x00852372
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the SDRAMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	TXSR[3:0]				TRAS[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	0	0	0	0	1	0	1

Bit	15	14	13	12	11	10	9	8
	TRCD[3:0]				TRP[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	0	0	0	1	1

Bit	7	6	5	4	3	2	1	0
	TRC_TRFC[3:0]				TWR[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	1	0	0	1	0

Bits 23:20 – TXSR[3:0] Exit Self-Refresh to Active Delay
Reset value is eight cycles.
This field defines the delay between SCKE set high and an Activate Command in number of cycles. Number of cycles is between 0 and 15.

Bits 19:16 – TRAS[3:0] Active to Precharge Delay
Reset value is five cycles.
This field defines the delay between an Activate Command and a Precharge Command in number of cycles. Number of cycles is between 0 and 15.

Bits 15:12 – TRCD[3:0] Row to Column Delay
Reset value is two cycles.
This field defines the delay between an Activate Command and a Read/Write Command in number of cycles. Number of cycles is between 0 and 15.

Bits 11:8 – TRP[3:0] Row Precharge Delay
Reset value is three cycles.
This field defines the delay between a Precharge Command and another Command in number of cycles. Number of cycles is between 0 and 15.

Bits 7:4 – TRC_TRFC[3:0] Row Cycle Delay and Row Refresh Cycle
Reset value is seven cycles.
This field defines two timings:
– the delay (t_{RFC}) between two Refresh commands and between a Refresh command and an Activate command
– and the delay (t_{RC}) between two Active commands in number of cycles.
The number of cycles is between 0 and 15. The end user must program max {t_{RC}, t_{RFC}}.

Bits 3:0 – TWR[3:0] Write Recovery Delay
Reset value is two cycles.
This field defines the Write Recovery Time in number of cycles. Number of cycles is between 0 and 15.

31.10.5 HSDRAMC High-Speed Register

Name: HSDRAMC_HSR
Offset: 0x0010
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the HSDRAMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								DA
Access								R/W
Reset								0

Bit 0 – DA Decode Cycle Enable
A decode cycle can be added on the addresses as soon as a non-sequential access is performed on the AHB bus. The addition of the decode cycle allows the HSDRAMC to gain time to access the SDRAM memory.

Value	Description
0	Decode cycle is disabled.
1	Decode cycle is enabled.

31.10.6 HSDRAMC Interrupt Enable Register

Name: HSDRAMC_IER
Offset: 0x0014
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								RES
Access								W
Reset								–

Bit 0 – RES Refresh Error Interrupt Enable

Value	Description
0	No effect.
1	Enables the refresh error interrupt.

31.10.7 HSDRAMC Interrupt Disable Register

Name: HSDRAMC_IDR
Offset: 0x0018
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								RES
Access								W
Reset								–

Bit 0 – RES Refresh Error Interrupt Disable

Value	Description
0	No effect.
1	Disables the refresh error interrupt.

31.10.8 HSDRAMC Interrupt Mask Register

Name: HSDRAMC_IMR
Offset: 0x001C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								RES
Access								R
Reset								0

Bit 0 – RES Refresh Error Interrupt Mask

Value	Description
0	The refresh error interrupt is disabled.
1	The refresh error interrupt is enabled.

31.10.9 HSDRAMC Interrupt Status Register

Name: HSDRAMC_ISR
Offset: 0x0020
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								RES
Access								R
Reset								0

Bit 0 – RES Refresh Error Status (cleared on read)

Value	Description
0	No refresh error has been detected since the register was last read.
1	A refresh error has been detected since the register was last read.

31.10.10 HSDRAMC Configuration Register 1

Name: HSDRAMC_CFR1
Offset: 0x0024
Reset: 0x00000002
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the HSDRAMC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								UNAL
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
					TMRD[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	1	0

Bit 8 – UNAL Support Unaligned Access
This mode is enabled with masters which have an AXI interface.

Value	Name	Description
0	UNSUPPORTED	Unaligned access is not supported.
1	SUPPORTED	Unaligned access is supported.

Bits 3:0 – TMRD[3:0] Load Mode Register Command to Active or Refresh Command
Reset value is 2 cycles.
This field defines the delay between a “Load Mode Register” command and an active or refresh command in number of cycles. Number of cycles is between 0 and 15.

31.10.11 HSDRAMC Write Protection Mode Register

Name: HSDRAMC_WPMR
Offset: 0x0028
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPEN
Access								R/W
Reset								0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x534452	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

Bit 0 – WPEN Write Protection Enable

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x534452 (SDR in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x534452 (SDR in ASCII).

31.10.12 HSDRAMC Write Protection Status Register

Name: HSDRAMC_WPSR
Offset: 0x002C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 15:8 – WPVSRC[7:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protection violation has occurred since the last read of the HSDRAMC_WPSR.
1	A write protection violation has occurred since the last read of the HSDRAMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

32. Special Function Registers (SFR)

32.1 Description

Special Function Registers (SFR) manage specific aspects of the integrated memory, bridge implementations, processor and other functionality not controlled elsewhere.

32.2 Embedded Characteristics

- 32-bit Special Function Registers Control Specific Behavior of the Product

32.3 Register Summary

Offset	Name	Bit Pos.								
0x00 ... 0x27	Reserved									
0x28	SFR_SECURE	7:0								ROM
		15:8								
		23:16								
		31:24								
0x2C ... 0x9F	Reserved									
0xA0	SFR_CAN0	7:0								
		15:8								
		23:16	EXT_MEM_ADDR[7:0]							
		31:24	EXT_MEM_ADDR[15:8]							
0xA4	SFR_CAN1	7:0								
		15:8								
		23:16	EXT_MEM_ADDR[7:0]							
		31:24	EXT_MEM_ADDR[15:8]							

32.3.1 Security Configuration Register

Name: SFR_SECURE
Offset: 0x28
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								ROM
Access								R/W
Reset								0

Bit 0 – ROM Disable Access to ROM Code
This bit is writable once only. When the ROM is secured, only a reset signal can clear this bit.

Value	Description
0	ROM is enabled.
1	ROM is disabled.

32.3.2 CAN0 MSB Address Register

Name: SFR_CAN0
Offset: 0xA0
Reset: 0x20000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	EXT_MEM_ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EXT_MEM_ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bits 31:16 – EXT_MEM_ADDR[15:0] MSB Base Address
Gives the 16-bit MSB of the CAN0 MSB base address. The 16-bit LSB must be programmed in the CAN0 user interface.

32.3.3 CAN1 MSB Address Register

Name: SFR_CAN1
Offset: 0xA4
Reset: 0x20000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	EXT_MEM_ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	1	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EXT_MEM_ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bits 31:16 – EXT_MEM_ADDR[15:0] MSB Base Address
Gives the 16-bit MSB of the CAN1 MSB base address. The 16-bit LSB must be programmed in the CAN1 user interface.

33. DMA Controller (XDMAC)

33.1 Description

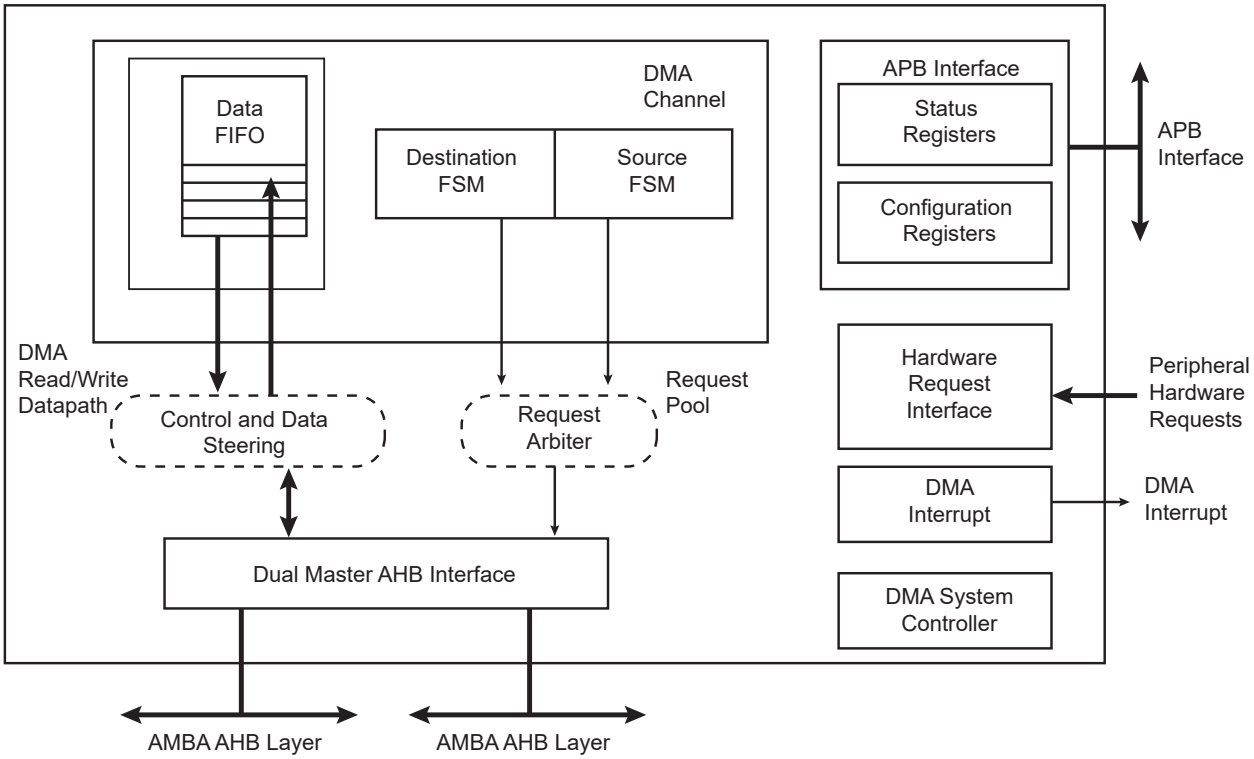
The DMA Controller (XDMAC) is a AHB-protocol central direct memory access controller. It performs peripheral data transfer and memory move operations over one or two bus ports through the unidirectional communication channel. Each channel is fully programmable and provides both peripheral or memory-to-memory transfers. The channel features are configurable at implementation.

33.2 Embedded Characteristics

- 2 AHB Master Interfaces
- 32 DMA Channels
- 44 Hardware Requests
- 4.125 Kbytes Embedded FIFO
- Supports Peripheral-to-Memory, Memory-to-Peripheral, or Memory-to-Memory Transfer Operations
- Peripheral DMA Operation Runs on Bytes (8-bit), Half-Word (16-bit) and Word (32-bit)
- Memory DMA Operation Runs on Bytes (8 bit), Half-Word (16-bit) and Word (32 -bit)
- Supports Hardware and Software Initiated Transfers
- Supports Linked List Operations
- Supports Incrementing or Fixed Addressing Mode
- Supports Programmable Independent Data Striding for Source and Destination
- Supports Programmable Independent Microblock Striding for Source and Destination
- Configurable Priority Group and Arbitration Policy
- Programmable AHB Burst Length
- Configuration Interface Accessible through APB Interface
- XDMAC Architecture Includes Multiport FIFO
- Supports Multiple View Channel Descriptor
- Automatic Flush of Channel Trailing Bytes
- Automatic Coarse-Grain and Fine-Grain Clock Gating
- Hardware Acceleration of Memset Pattern

33.3 Block Diagram

Figure 33-1. DMA Controller (XDMAC) Block Diagram



33.4 DMA Controller Peripheral Connections

33.5 Functional Description

33.5.1 Basic Definitions

Source Peripheral: Slave device, memory mapped on the interconnection network, from where the XDMAC reads data. The source peripheral teams up with a destination peripheral to form a channel. A data read operation is scheduled when the peripheral transfer request is asserted.

Destination Peripheral: Slave device, memory mapped on the interconnection network, to which the XDMAC writes. A write data operation is scheduled when the peripheral transfer request is asserted.

Channel: The data movement between source and destination creates a logical channel.

Transfer Type: The transfer is hardware-synchronized when it is paced by the peripheral hardware request, otherwise the transfer is self-triggered (memory to memory transfer).

33.5.2 Transfer Hierarchy Diagram

XDMAC Master Transfer: The Master Transfer is composed of a linked list of blocks. The channel address, control and configuration registers can be modified at the inter block boundary. The descriptor structure modifies the channel registers conditionally. Interrupts can be generated on a per block basis or when the end of linked list event occurs.

XDMAC Block: An XDMAC block is composed of a programmable number of microblocks. The channel configuration registers remain unchanged at the inter microblock boundary. The source and destination addresses are conditionally updated with a programmable signed number.

XDMAC Microblock: The microblock is composed of a programmable number of data. The channel configuration registers remain unchanged at the data boundary. The data address may be fixed (a FIFO location, a peripheral transmit or receive register), incrementing (a memory-mapped area) by a programmable signed number.

XDMAC Burst and Incomplete Burst: In order to improve the overall performance when accessing dynamic external memory, burst access is mandatory. Each data of the microblock is considered as a part of a memory burst. The programmable burst value indicates the largest memory burst allowed on a per channel basis. When the microblock length is not an integral multiple of the burst size, an incomplete burst is performed to read or write the last trailing bytes.

XDMAC Chunk and Incomplete Chunk: When a peripheral synchronized transfer is activated, the microblock splits into a number of data chunks. The chunk size is programmable. The larger the chunk is, the better the performance is. When the transfer size is not a multiple of the chunk size, the last chunk may be incomplete.

Figure 33-2. XDMAC Memory Transfer Hierarchy

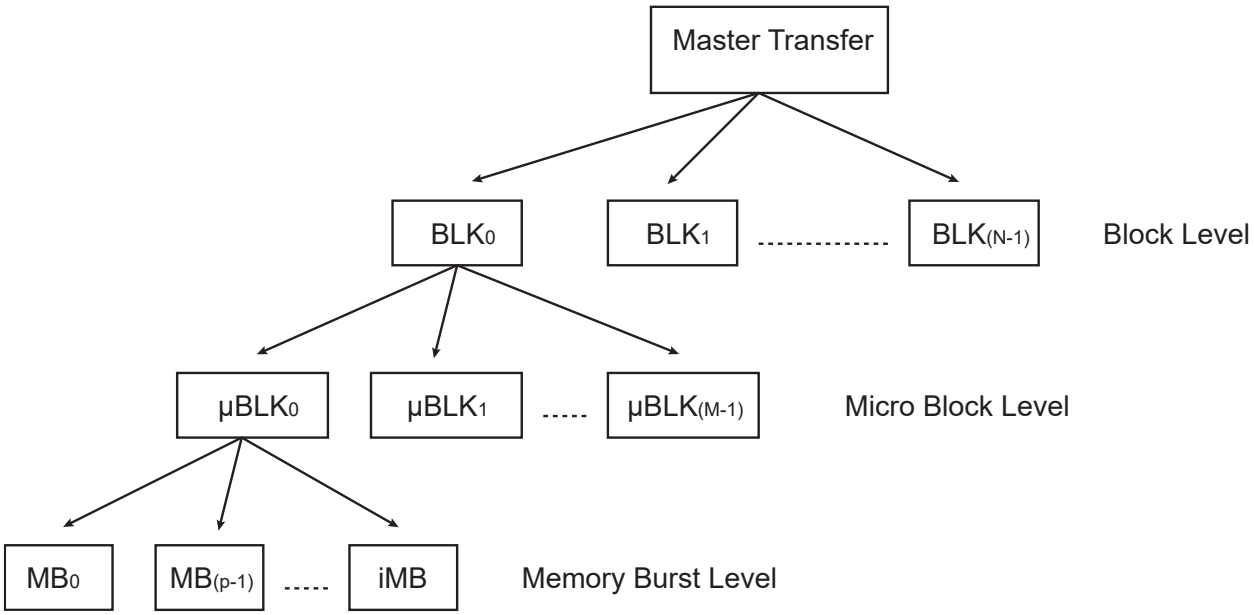
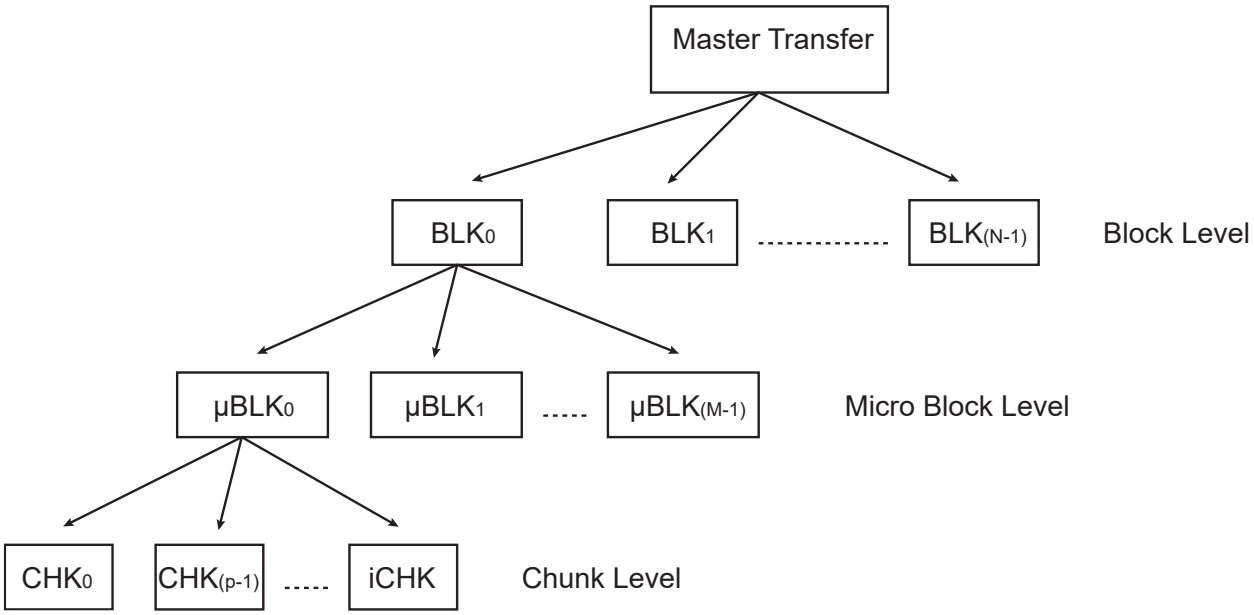


Figure 33-3. XDAMC Peripheral Transfer Hierarchy



33.5.3 Peripheral Synchronized Transfer

A peripheral hardware request interface is used to control the pace of the chunk transfer. When a peripheral is ready to transmit or receive a chunk of data, it asserts its request line and the DMA Controller transfers a data to or from the memory to the peripheral.

33.5.3.1 Software Triggered Synchronized Transfer

The Peripheral hardware request can be software controlled using the SWREQ field of the XDMAC Global Channel Software Request Register (XDMAC_GSWR). The peripheral synchronized transfer is paced using a processor write access in the XDMAC_GSWR. Each bit of that register triggers a transfer request. The XDMAC Global Channel Software Request Status Register (XDMAC_GSWS) indicates the status of the request; when set, the request is still pending.

33.5.4 XDMAC Transfer Software Operation

Note:
When a memory-to-memory transfer is performed, configure the field XDMAC_CCx.PERID (where ‘x’ is the index of the channel used for transfer) to an unused peripheral ID (refer to table “Peripheral Identifiers”).

33.5.4.1 Single Block Transfer With Single Microblock

1. Read the XDMAC Global Channel Status Register (XDMAC_GS) to select a free channel.
2. Clear the pending Interrupt Status bit(s) by reading the selected XDMAC Channel x Interrupt Status Register (XDMAC_CISx).
3. Write the XDMAC Channel x Source Address Register (XDMAC_CSAx) for channel x.
4. Write the XDMAC Channel x Destination Address Register (XDMAC_CDAx) for channel x.
5. Program field UBLEN in the XDMAC Channel x Microblock Control Register (XDMAC_CUBCx) with the number of data.
6. Program the XDMAC Channel x Configuration Register (XDMAC_CCx):
    6.1. Clear XDMAC_CCx.TYPE for a memory-to-memory transfer, otherwise set this bit.
    6.2. Configure XDMAC_CCx.MBSIZE to the memory burst size used.
    6.3. Configure XDMAC_CCx.SAM and DAM to Memory Addressing mode.
    6.4. Configure XDMAC_CCx.DSYNC to select the peripheral transfer direction.
    6.5. Configure XDMAC_CCx.CSIZE to configure the channel chunk size (only relevant for peripheral synchronized transfer).
    6.6. Configure XDMAC_CCx.DWIDTH to configure the transfer data width.
    6.7. Configure XDMAC_CCx.SIF, XDMAC_CCx.DIF to configure the master interface used to read data and write data, respectively.
    6.8. Configure XDMAC_CCx.PERID to select the active hardware request line (only relevant for a peripheral synchronized transfer).
    6.9. Set XDMAC_CCx.SWREQ to use a software request (only relevant for a peripheral synchronized transfer).
7. Clear the following five registers:
    – XDMAC Channel x Next Descriptor Control Register (XDMAC_CNDCx)
    – XDMAC Channel x Block Control Register (XDMAC_CBCx)
    – XDMAC Channel x Data Stride Memory Set Pattern Register (XDMAC_CDS_MSPx)
    – XDMAC Channel x Source Microblock Stride Register (XDMAC_CSUSx)
    – XDMAC Channel x Destination Microblock Stride Register (XDMAC_CDUSx)
    This indicates that the linked list is disabled, there is only one block and striding is disabled.
8. Enable the Microblock interrupt by writing a ‘1’ to bit BIE in the XDMAC Channel x Interrupt Enable Register (XDMAC_CIEx). Enable the Channel x Interrupt Enable bit by writing a ‘1’ to bit IEx in the XDMAC Global Interrupt Enable Register (XDMAC_GIE).
9. Enable channel x by writing a ‘1’ to bit ENx in the XDMAC Global Channel Enable Register (XDMAC_GE). XDMAC_GS.STx (XDMAC Channel x Status bit) is set by hardware.
10. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

33.5.4.2 Single Block Transfer With Multiple Microblock

1. Read the XDMAC_GS register to choose a free channel.
2. Clear the pending Interrupt Status bit by reading the chosen XDMAC_CISx register.
3. Write the XDMAC_CSAx register for channel x.
4. Write the XDMAC_CDAx register for channel x.
5. Program XDMAC_CUBCx.UBLEN with the number of data.
6. Program XDMAC_CCx register (see "Single Block Transfer With Single Microblock").
7. Program XDMAC_CBCx.BLEN with the number of microblocks of data.
8. Clear the following registers:
   - XDMAC_CNDCx
   - XDMAC_CDS_MSPx
   - XDMAC_CSUSx XDMAC_CDUSx
     This indicates that the linked list is disabled and striding is disabled.
9. Enable the Block interrupt by writing a '1' to XDMAC_CIEx.BIE, enable the Channel x Interrupt Enable bit by writing a '1' to XDMAC_GIEx.IEx.
10. Enable channel x by writing a '1' to the XDMAC_GE.ENx. XDMAC_GS.STx is set by hardware.
11. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

33.5.4.3 Master Transfer

1. Read the XDMAC_GS register to choose a free channel.
2. Clear the pending Interrupt Status bit by reading the chosen XDMAC_CISx register.
3. Build a linked list of transfer descriptors in memory. The descriptor view is programmable on a per descriptor basis. The linked list items structure must be word aligned. MBR_UBC.NDE must be configured to 0 in the last descriptor to terminate the list.
4. Configure field NDA in the XDMAC Channel x Next Descriptor Address Register (XDMAC_CNDAx) with the first descriptor address and bit XDMAC_CNDAx.NDAIF with the master interface identifier.
5. Configure the XDMAC_CNDCx register:
   5.1. Set XDMAC_CNDCx.NDE to enable the descriptor fetch.
   5.2. Set XDMAC_CNDCx.NDSUP to update the source address at the descriptor fetch time, otherwise clear this bit.
   5.3. Set XDMAC_CNDCx.NDDUP to update the destination address at the descriptor fetch time, otherwise clear this bit.
   5.4. Configure XDMAC_CNDCx.NDVIEW to define the length of the first descriptor.
6. Enable the End of Linked List interrupt by writing a '1' to XDMAC_CIEx.LIE.
7. Enable channel x by writing a '1' to XDMAC_GE.ENx. XDMAC_GS.STx is set by hardware.
8. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

33.5.4.4 Disabling A Channel Before Transfer Completion

Under normal operation, the software enables a channel by writing a '1' to XDMAC_GE.ENx, then the hardware disables a channel on transfer completion by clearing bit XDMAC_GS.STx. To disable a channel, write a '1' to bit XDMAC_GD.DIx and poll the XDMAC_GS register.

33.6 Linked List Descriptor Operation

33.6.1 Linked List Descriptor View

33.6.1.1 Channel Next Descriptor View 0–3 Structures

Table 33-1. Channel Next Descriptor View 0–3 Structures

Channel Next Descriptor	Offset	Structure member	Name
View 0 Structure	DSCR_ADDR+0x00	Next Descriptor Address Member	MBR_NDA
	DSCR_ADDR+0x04	Microblock Control Member	MBR_UBC
	DSCR_ADDR+0x08	Transfer Address Member	MBR_TA
View 1 Structure	DSCR_ADDR+0x00	Next Descriptor Address Member	MBR_NDA
	DSCR_ADDR+0x04	Microblock Control Member	MBR_UBC
	DSCR_ADDR+0x08	Source Address Member	MBR_SA
	DSCR_ADDR+0x0C	Destination Address Member	MBR_DA
View 2 Structure	DSCR_ADDR+0x00	Next Descriptor Address Member	MBR_NDA
	DSCR_ADDR+0x04	Microblock Control Member	MBR_UBC
	DSCR_ADDR+0x08	Source Address Member	MBR_SA
	DSCR_ADDR+0x0C	Destination Address Member	MBR_DA
	DSCR_ADDR+0x10	Configuration Register	MBR_CFG
View 3 Structure	DSCR_ADDR+0x00	Next Descriptor Address Member	MBR_NDA
	DSCR_ADDR+0x04	Microblock Control Member	MBR_UBC
	DSCR_ADDR+0x08	Source Address Member	MBR_SA
	DSCR_ADDR+0x0C	Destination Address Member	MBR_DA
	DSCR_ADDR+0x10	Configuration Member	MBR_CFG
	DSCR_ADDR+0x14	Block Control Member	MBR_BC
	DSCR_ADDR+0x18	Data Stride Member	MBR_DS
	DSCR_ADDR+0x1C	Source Microblock Stride Member	MBR_SUS
	DSCR_ADDR+0x20	Destination Microblock Stride Member	MBR_DUS

33.6.2 Descriptor Structure Members Description

33.6.2.1 Descriptor Structure Microblock Control Member

Name: MBR_UBC
Property: Read-only

Bit	31	30	29	28	27	26	25	24
				NVIEW[1:0]		NDEN	NSEN	NDE
Access				R	R	R	R	R
Reset								

Bit	23	22	21	20	19	18	17	16
	UBLEN[23:16]							
Access	R	R	R	R	R	R	R	R
Reset								

Bit	15	14	13	12	11	10	9	8
	UBLEN[15:8]							
Access	R	R	R	R	R	R	R	R
Reset								

Bit	7	6	5	4	3	2	1	0
	UBLEN[7:0]							
Access	R	R	R	R	R	R	R	R
Reset								

Bits 28:27 – NVIEW[1:0] Next Descriptor View

Value	Name	Description
0	NDV0	Next Descriptor View 0
1	NDV1	Next Descriptor View 1
2	NDV2	Next Descriptor View 2
3	NDV3	Next Descriptor View 3

Bit 26 – NDEN Next Descriptor Destination Update

Value	Description
0	Destination parameters remain unchanged.
1	Destination parameters are updated when the descriptor is retrieved.

Bit 25 – NSEN Next Descriptor Source Update

Value	Description
0	Source parameters remain unchanged.
1	Source parameters are updated when the descriptor is retrieved.

Bit 24 – NDE Next Descriptor Enable

Value	Description
0	Descriptor fetch is disabled.
1	Descriptor fetch is enabled.

Bits 23:0 – UBLEN[23:0] Microblock Length
This field indicates the number of data in the microblock. The microblock contains UBLEN data.

33.7 XDMAC Maintenance Software Operations

33.7.1 Disabling a Channel

A disable channel request occurs when a write operation is performed in the XDMAC_GD register. If the channel is source peripheral synchronized (bit XDMAC_CCx.TYPE is set and bit XDMAC_CCx.DSYNC is cleared), then pending bytes (bytes located in the FIFO) are written to memory and bit XDMAC_CISx.DIS is set. If the channel is not source peripheral synchronized, the current channel transaction (read or write) is terminated and XDMAC_CISx.DIS is set. XDMAC_GS.STx is cleared by hardware when the current transfer is completed. The channel is no longer active and can be reused.

33.7.2 Suspending a Channel

A disable channel request occurs when a write operation is performed in the XDMAC_GD register. If the channel is source peripheral synchronized (bit XDMAC_CCx.TYPE is set and bit XDMAC_CCx.DSYNC is cleared), then pending bytes (bytes located in the FIFO) are written to memory and bit XDMAC_CISx.DIS is set. If the channel is not source peripheral synchronized, the current channel transaction (read or write) is terminated and XDMAC_CISx.DIS is set. XDMAC_GS.STx is cleared by hardware when the current transfer is completed. The channel is no longer active and can be reused.

33.7.3 Flushing a Channel

A FIFO flush command is issued by writing to the XDMAC_SWF register. The content of the FIFO is written to memory. XDMAC_CISx.FIS (End of Flush Interrupt Status bit) is set when the last byte is successfully transferred to memory. The channel is not disabled. The flush operation is not blocking, meaning that read operation can be scheduled during the flush write operation. The flush operation is only relevant for peripheral to memory transfer where pending peripheral bytes are buffered into the channel FIFO.

33.7.4 Maintenance Operation Priority

33.7.4.1 Disable Operation Priority

- When a disable request occurs on a suspended channel, the XDMAC_GWS.WSx (Channel x Write Suspend bit) is cleared. If the transfer is source peripheral synchronized, the pending bytes are drained to memory. The bit XDMAC_CISx.DIS is set.
- When a disable request follows a flush request, if the flush last transaction is not yet scheduled, the flush request is discarded and the disable procedure is applied. Bit XDMAC_CISx.FIS is not set. Bit XDMAC_CISx.DIS is set when the disable request is completed. If the flush request transaction is already scheduled, the XDMAC_CISx.FIS is set. XDMAC_CISx.DIS is also set when the disable request is completed.

33.7.4.2 Flush Operation Priority

- When a flush request occurs on a suspended channel, if there are pending bytes in the FIFO, they are written out to memory, XDMAC_CISx.FIS is set. If the FIFO is empty, XDMAC_CISx.FIS is also set.
- If the flush operation is performed after a disable request, the flush command is ignored. XDMAC_CISx.FIS is not set.

33.7.4.3 Suspend Operation Priority

If the suspend operation is performed after a disable request, the write suspend operation is ignored.

33.8 XDMAC Software Requirements

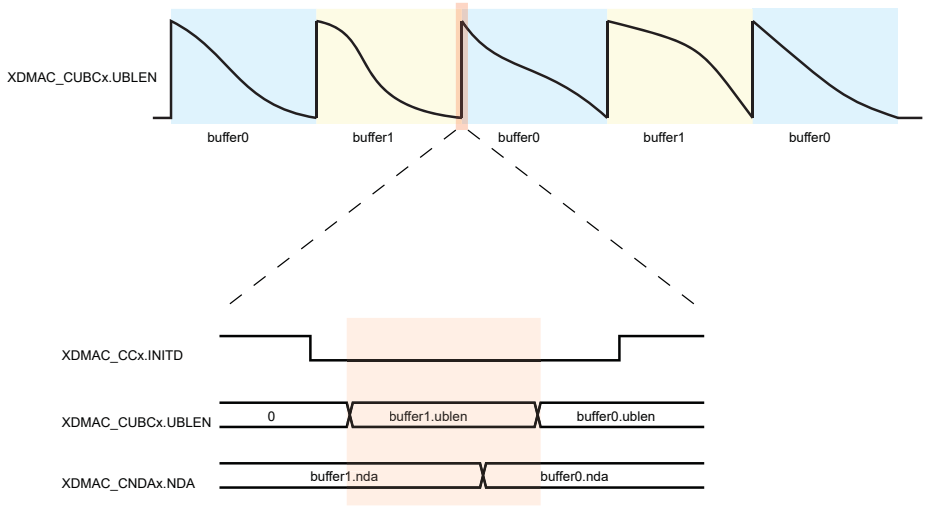
- Write operations to channel registers are not be performed in an active channel after the channel is enabled. If any channel parameters must be reprogrammed, this can only be done after disabling the XDMAC channel.
- XDMAC_CSAx and XDMAC_CDAx channel registers are to be programmed with a byte, half-word or word aligned address depending on the Channel x Data Width field (DWIDTH) of the XDMAC Channel x Configuration Register.
- When a memory-to-memory transfer is performed, configure the field XDMAC_CCx.PERID (where ‘x’ is the index of the channel used for the transfer) to an unused peripheral ID (refer to table “Peripheral Identifiers”).

- When XDMAC_CC.INITD is set to 0, XDMAC_CUBC.UBLEN and XDMAC_CNDA.NDA field values are unreliable when the descriptor is being updated. The following procedure applies to get the buffer descriptor identifier and the residual bytes:

```
Read XDMAC_CNDAx.NDA(nda0)
Read XDMAC_CCx.INITD(initd0)
Read XDMAC_CCx.INITD(initd0)
Read XDMAC_CUBCx.UBLEN(ublen)
Read XDMAC_CCx.INITD(initd1)
Read XDMA_CNDAx.NDA(nda1)
If (nda0 == nda1 && initd0 == 1 && initd1 == 1).
Then the ublen is correct, the buffer id is nda.
Else retry
```

See the figure below.

Figure 33-4. INITD Timing Diagram



33.9 Register Summary

Offset	Name	Bit Pos.								
0x00	XDMAC_GTYPE	7:0	FIFO_SZ[2:0]				NB_CH[4:0]			
		15:8	FIFO_SZ[10:3]							
		23:16		NB_REQ[6:0]						
		31:24								
0x04	XDMAC_GCFG	7:0					CGDISIF	CGDISFIFO	CGDISPIPE	CGDISREG
		15:8								BXKBEN
		23:16								
		31:24								
0x08	XDMAC_GWAC	7:0	PW1[3:0]				PW0[3:0]			
		15:8	PW3[3:0]				PW2[3:0]			
		23:16								
		31:24								
0x0C	XDMAC_GIE	7:0	IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0
		15:8	IE15	IE14	IE13	IE12	IE11	IE10	IE9	IE8
		23:16	IE23	IE22	IE21	IE20	IE19	IE18	IE17	IE16
		31:24	IE31	IE30	IE29	IE28	IE27	IE26	IE25	IE24
0x10	XDMAC_GID	7:0	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
		15:8	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
		23:16	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
		31:24	ID31	ID30	ID29	ID28	ID27	ID26	ID25	ID24
0x14	XDMAC_GIM	7:0	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
		15:8	IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8
		23:16	IM23	IM22	IM21	IM20	IM19	IM18	IM17	IM16
		31:24	IM31	IM30	IM29	IM28	IM27	IM26	IM25	IM24
0x18	XDMAC_GIS	7:0	IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0
		15:8	IS15	IS14	IS13	IS12	IS11	IS10	IS9	IS8
		23:16	IS23	IS22	IS21	IS20	IS19	IS18	IS17	IS16
		31:24	IS31	IS30	IS29	IS28	IS27	IS26	IS25	IS24
0x1C	XDMAC_GE	7:0	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
		15:8	EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8
		23:16	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
		31:24	EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24
0x20	XDMAC_GD	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x24	XDMAC_GS	7:0	ST7	ST6	ST5	ST4	ST3	ST2	ST1	ST0
		15:8	ST15	ST14	ST13	ST12	ST11	ST10	ST9	ST8
		23:16	ST23	ST22	ST21	ST20	ST19	ST18	ST17	ST16
		31:24	ST31	ST30	ST29	ST28	ST27	ST26	ST25	ST24
0x28	XDMAC_GRS	7:0	RS7	RS6	RS5	RS4	RS3	RS2	RS1	RS0
		15:8	RS15	RS14	RS13	RS12	RS11	RS10	RS9	RS8
		23:16	RS23	RS22	RS21	RS20	RS19	RS18	RS17	RS16
		31:24	RS31	RS30	RS29	RS28	RS27	RS26	RS25	RS24
0x2C	XDMAC_GWS	7:0	WS7	WS6	WS5	WS4	WS3	WS2	WS1	WS0
		15:8	WS15	WS14	WS13	WS12	WS11	WS10	WS9	WS8
		23:16	WS23	WS22	WS21	WS20	WS19	WS18	WS17	WS16
		31:24	WS31	WS30	WS29	WS28	WS27	WS26	WS25	WS24
0x30	XDMAC_GRWS	7:0	RWS7	RWS6	RWS5	RWS4	RWS3	RWS2	RWS1	RWS0
		15:8	RWS15	RWS14	RWS13	RWS12	RWS11	RWS10	RWS9	RWS8
		23:16	RWS23	RWS22	RWS21	RWS20	RWS19	RWS18	RWS17	RWS16
		31:24	RWS31	RWS30	RWS29	RWS28	RWS27	RWS26	RWS25	RWS24
0x34	XDMAC_GRWR	7:0	RWR7	RWR6	RWR5	RWR4	RWR3	RWR2	RWR1	RWR0
		15:8	RWR15	RWR14	RWR13	RWR12	RWR11	RWR10	RWR9	RWR8
		23:16	RWR23	RWR22	RWR21	RWR20	RWR19	RWR18	RWR17	RWR16
		31:24	RWR31	RWR30	RWR29	RWR28	RWR27	RWR26	RWR25	RWR24

.....continued

Offset	Name	Bit Pos.								
0x38	XDMAC_GSWR	7:0	SWREQ7	SWREQ6	SWREQ5	SWREQ4	SWREQ3	SWREQ2	SWREQ1	SWREQ0
		15:8	SWREQ15	SWREQ14	SWREQ13	SWREQ12	SWREQ11	SWREQ10	SWREQ9	SWREQ8
		23:16	SWREQ23	SWREQ22	SWREQ21	SWREQ20	SWREQ19	SWREQ18	SWREQ17	SWREQ16
		31:24	SWREQ31	SWREQ30	SWREQ29	SWREQ28	SWREQ27	SWREQ26	SWREQ25	SWREQ24
0x3C	XDMAC_GSWS	7:0	SWRS7	SWRS6	SWRS5	SWRS4	SWRS3	SWRS2	SWRS1	SWRS0
		15:8	SWRS15	SWRS14	SWRS13	SWRS12	SWRS11	SWRS10	SWRS9	SWRS8
		23:16	SWRS23	SWRS22	SWRS21	SWRS20	SWRS19	SWRS18	SWRS17	SWRS16
		31:24	SWRS31	SWRS30	SWRS29	SWRS28	SWRS27	SWRS26	SWRS25	SWRS24
0x40	XDMAC_GSWF	7:0	SWF7	SWF6	SWF5	SWF4	SWF3	SWF2	SWF1	SWF0
		15:8	SWF15	SWF14	SWF13	SWF12	SWF11	SWF10	SWF9	SWF8
		23:16	SWF23	SWF22	SWF21	SWF20	SWF19	SWF18	SWF17	SWF16
		31:24	SWF31	SWF30	SWF29	SWF28	SWF27	SWF26	SWF25	SWF24
0x44 ... 0x4F	Reserved									
0x50	XDMAC_CIE0	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x54	XDMAC_CID0	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x58	XDMAC_CIM0	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x5C	XDMAC_CIS0	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x60	XDMAC_CSA0	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x64	XDMAC_CDA0	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x68	XDMAC_CNDA0	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x6C	XDMAC_CNDC0	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x70	XDMAC_CUBC0	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x74	XDMAC_CBC0	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x78	XDMAC_CC0	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x7C	XDMAC_CDS_MSP0	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x80	XDMAC_CSUS0	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x84	XDMAC_CDUS0	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x88 ... 0x8F	Reserved									
0x90	XDMAC_CIE1	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x94	XDMAC_CID1	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x98	XDMAC_CIM1	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x9C	XDMAC_CIS1	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0xA0	XDMAC_CSA1	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0xA4	XDMAC_CDA1	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0xA8	XDMAC_CNDA1	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0xAC	XDMAC_CNDC1	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0xB0	XDMAC_CUBC1	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0xB4	XDMAC_CBC1	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0xB8	XDMAC_CC1	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0xBC	XDMAC_CDS_MSP1	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0xC0	XDMAC_CSUS1	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0xC4	XDMAC_CDUS1	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0xC8 ... 0xCF	Reserved									
0xD0	XDMAC_CIE2	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0xD4	XDMAC_CID2	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0xD8	XDMAC_CIM2	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0xDC	XDMAC_CIS2	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0xE0	XDMAC_CSA2	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0xE4	XDMAC_CDA2	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0xE8	XDMAC_CNDA2	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0xEC	XDMAC_CNDC2	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0xF0	XDMAC_CUBC2	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0xF4	XDMAC_CBC2	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0xF8	XDMAC_CC2	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0xFC	XDMAC_CDS_MSP2	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0100	XDMAC_CSUS2	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0104	XDMAC_CDUS2	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0108 ... 0x010F	Reserved									
0x0110	XDMAC_CIE3	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0114	XDMAC_CID3	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0118	XDMAC_CIM3	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x011C	XDMAC_CIS3	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0120	XDMAC_CSA3	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0124	XDMAC_CDA3	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0128	XDMAC_CNDA3	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x012C	XDMAC_CNDC3	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0130	XDMAC_CUBC3	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0134	XDMAC_CBC3	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0138	XDMAC_CC3	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x013C	XDMAC_CDS_MSP3	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0140	XDMAC_CSUS3	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0144	XDMAC_CDUS3	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0148 ... 0x014F	Reserved									
0x0150	XDMAC_CIE4	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0154	XDMAC_CID4	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0158	XDMAC_CIM4	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x015C	XDMAC_CIS4	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0160	XDMAC_CSA4	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0164	XDMAC_CDA4	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0168	XDMAC_CNDA4	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x016C	XDMAC_CNDC4	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0170	XDMAC_CUBC4	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0174	XDMAC_CBC4	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0178	XDMAC_CC4	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x017C	XDMAC_CDS_MSP4	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0180	XDMAC_CSUS4	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0184	XDMAC_CDUS4	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0188 ... 0x018F	Reserved									
0x0190	XDMAC_CIE5	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0194	XDMAC_CID5	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0198	XDMAC_CIM5	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x019C	XDMAC_CIS5	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x01A0	XDMAC_CSA5	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x01A4	XDMAC_CDA5	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x01A8	XDMAC_CNDA5	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x01AC	XDMAC_CNDC5	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x01B0	XDMAC_CUBC5	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x01B4	XDMAC_CBC5	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x01B8	XDMAC_CC5	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x01BC	XDMAC_CDS_MSP5	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x01C0	XDMAC_CSUS5	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x01C4	XDMAC_CDUS5	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x01C8 ... 0x01CF	Reserved									
0x01D0	XDMAC_CIE6	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x01D4	XDMAC_CID6	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x01D8	XDMAC_CIM6	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x01DC	XDMAC_CIS6	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x01E0	XDMAC_CSA6	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x01E4	XDMAC_CDA6	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x01E8	XDMAC_CNDA6	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x01EC	XDMAC_CNDC6	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x01F0	XDMAC_CUBC6	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x01F4	XDMAC_CBC6	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x01F8	XDMAC_CC6	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x01FC	XDMAC_CDS_MSP6	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0200	XDMAC_CSUS6	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0204	XDMAC_CDUS6	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0208 ... 0x020F	Reserved									
0x0210	XDMAC_CIE7	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0214	XDMAC_CID7	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0218	XDMAC_CIM7	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x021C	XDMAC_CIS7	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0220	XDMAC_CSA7	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0224	XDMAC_CDA7	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0228	XDMAC_CNDA7	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x022C	XDMAC_CNDC7	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0230	XDMAC_CUBC7	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0234	XDMAC_CBC7	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0238	XDMAC_CC7	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x023C	XDMAC_CDS_MSP7	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0240	XDMAC_CSUS7	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0244	XDMAC_CDUS7	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0248 ... 0x024F	Reserved									
0x0250	XDMAC_CIE8	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0254	XDMAC_CID8	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0258	XDMAC_CIM8	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x025C	XDMAC_CIS8	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0260	XDMAC_CSA8	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0264	XDMAC_CDA8	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0268	XDMAC_CNDA8	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x026C	XDMAC_CNDC8	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0270	XDMAC_CUBC8	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0274	XDMAC_CBC8	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0278	XDMAC_CC8	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x027C	XDMAC_CDS_MSP8	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0280	XDMAC_CSUS8	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0284	XDMAC_CDUS8	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0288 ... 0x028F	Reserved									
0x0290	XDMAC_CIE9	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0294	XDMAC_CID9	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0298	XDMAC_CIM9	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x029C	XDMAC_CIS9	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x02A0	XDMAC_CSA9	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x02A4	XDMAC_CDA9	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x02A8	XDMAC_CNDA9	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x02AC	XDMAC_CNDC9	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x02B0	XDMAC_CUBC9	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x02B4	XDMAC_CBC9	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x02B8	XDMAC_CC9	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x02BC	XDMAC_CDS_MSP9	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x02C0	XDMAC_CSUS9	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x02C4	XDMAC_CDUS9	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x02C8 ... 0x02CF	Reserved									
0x02D0	XDMAC_CIE10	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x02D4	XDMAC_CID10	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x02D8	XDMAC_CIM10	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x02DC	XDMAC_CIS10	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x02E0	XDMAC_CSA10	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x02E4	XDMAC_CDA10	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x02E8	XDMAC_CNDA10	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x02EC	XDMAC_CNDC10	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x02F0	XDMAC_CUBC10	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x02F4	XDMAC_CBC10	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x02F8	XDMAC_CC10	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x02FC	XDMAC_CDS_MSP10	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0300	XDMAC_CSUS10	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0304	XDMAC_CDUS10	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0308 ... 0x030F	Reserved									
0x0310	XDMAC_CIE11	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0314	XDMAC_CID11	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0318	XDMAC_CIM11	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x031C	XDMAC_CIS11	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0320	XDMAC_CSA11	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0324	XDMAC_CDA11	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0328	XDMAC_CNDA11	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x032C	XDMAC_CNDC11	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0330	XDMAC_CUBC11	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0334	XDMAC_CBC11	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0338	XDMAC_CC11	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x033C	XDMAC_CDS_MSP11	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0340	XDMAC_CSUS11	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0344	XDMAC_CDUS11	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0348 ... 0x034F	Reserved									
0x0350	XDMAC_CIE12	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0354	XDMAC_CID12	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0358	XDMAC_CIM12	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x035C	XDMAC_CIS12	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0360	XDMAC_CSA12	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0364	XDMAC_CDA12	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0368	XDMAC_CNDA12	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x036C	XDMAC_CNDC12	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0370	XDMAC_CUBC12	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0374	XDMAC_CBC12	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0378	XDMAC_CC12	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x037C	XDMAC_CDS_MSP12	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0380	XDMAC_CSUS12	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0384	XDMAC_CDUS12	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0388 ... 0x038F	Reserved									
0x0390	XDMAC_CIE13	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0394	XDMAC_CID13	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0398	XDMAC_CIM13	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x039C	XDMAC_CIS13	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x03A0	XDMAC_CSA13	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x03A4	XDMAC_CDA13	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x03A8	XDMAC_CNDA13	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x03AC	XDMAC_CNDC13	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x03B0	XDMAC_CUBC13	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x03B4	XDMAC_CBC13	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x03B8	XDMAC_CC13	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x03BC	XDMAC_CDS_MSP13	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x03C0	XDMAC_CSUS13	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x03C4	XDMAC_CDUS13	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x03C8 ... 0x03CF	Reserved									
0x03D0	XDMAC_CIE14	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x03D4	XDMAC_CID14	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x03D8	XDMAC_CIM14	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x03DC	XDMAC_CIS14	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x03E0	XDMAC_CSA14	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x03E4	XDMAC_CDA14	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x03E8	XDMAC_CNDA14	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x03EC	XDMAC_CNDC14	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x03F0	XDMAC_CUBC14	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x03F4	XDMAC_CBC14	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x03F8	XDMAC_CC14	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x03FC	XDMAC_CDS_MSP14	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0400	XDMAC_CSUS14	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0404	XDMAC_CDUS14	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0408 ... 0x040F	Reserved									
0x0410	XDMAC_CIE15	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0414	XDMAC_CID15	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0418	XDMAC_CIM15	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x041C	XDMAC_CIS15	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0420	XDMAC_CSA15	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0424	XDMAC_CDA15	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0428	XDMAC_CNDA15	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x042C	XDMAC_CNDC15	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0430	XDMAC_CUBC15	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0434	XDMAC_CBC15	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0438	XDMAC_CC15	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x043C	XDMAC_CDS_MSP15	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0440	XDMAC_CSUS15	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0444	XDMAC_CDUS15	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0448 ... 0x044F	Reserved									
0x0450	XDMAC_CIE16	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0454	XDMAC_CID16	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0458	XDMAC_CIM16	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x045C	XDMAC_CIS16	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0460	XDMAC_CSA16	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0464	XDMAC_CDA16	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0468	XDMAC_CNDA16	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x046C	XDMAC_CNDC16	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0470	XDMAC_CUBC16	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0474	XDMAC_CBC16	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0478	XDMAC_CC16	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x047C	XDMAC_CDS_MSP16	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0480	XDMAC_CSUS16	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0484	XDMAC_CDUS16	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0488 ... 0x048F	Reserved									
0x0490	XDMAC_CIE17	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0494	XDMAC_CID17	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0498	XDMAC_CIM17	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x049C	XDMAC_CIS17	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x04A0	XDMAC_CSA17	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x04A4	XDMAC_CDA17	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x04A8	XDMAC_CNDA17	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x04AC	XDMAC_CNDC17	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x04B0	XDMAC_CUBC17	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x04B4	XDMAC_CBC17	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x04B8	XDMAC_CC17	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x04BC	XDMAC_CDS_MSP17	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x04C0	XDMAC_CSUS17	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x04C4	XDMAC_CDUS17	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x04C8 ... 0x04CF	Reserved									
0x04D0	XDMAC_CIE18	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x04D4	XDMAC_CID18	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x04D8	XDMAC_CIM18	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x04DC	XDMAC_CIS18	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x04E0	XDMAC_CSA18	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x04E4	XDMAC_CDA18	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x04E8	XDMAC_CNDA18	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x04EC	XDMAC_CNDC18	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x04F0	XDMAC_CUBC18	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x04F4	XDMAC_CBC18	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x04F8	XDMAC_CC18	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x04FC	XDMAC_CDS_MSP18	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0500	XDMAC_CSUS18	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0504	XDMAC_CDUS18	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0508 ... 0x050F	Reserved									
0x0510	XDMAC_CIE19	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0514	XDMAC_CID19	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0518	XDMAC_CIM19	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x051C	XDMAC_CIS19	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0520	XDMAC_CSA19	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0524	XDMAC_CDA19	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0528	XDMAC_CNDA19	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x052C	XDMAC_CNDC19	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0530	XDMAC_CUBC19	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0534	XDMAC_CBC19	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0538	XDMAC_CC19	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x053C	XDMAC_CDS_MSP19	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0540	XDMAC_CSUS19	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0544	XDMAC_CDUS19	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0548 ... 0x054F	Reserved									
0x0550	XDMAC_CIE20	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0554	XDMAC_CID20	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0558	XDMAC_CIM20	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x055C	XDMAC_CIS20	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0560	XDMAC_CSA20	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0564	XDMAC_CDA20	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0568	XDMAC_CNDA20	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x056C	XDMAC_CNDC20	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0570	XDMAC_CUBC20	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0574	XDMAC_CBC20	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0578	XDMAC_CC20	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x057C	XDMAC_CDS_MSP20	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0580	XDMAC_CSUS20	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0584	XDMAC_CDUS20	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0588 ... 0x058F	Reserved									
0x0590	XDMAC_CIE21	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0594	XDMAC_CID21	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0598	XDMAC_CIM21	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x059C	XDMAC_CIS21	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x05A0	XDMAC_CSA21	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x05A4	XDMAC_CDA21	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x05A8	XDMAC_CNDA21	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x05AC	XDMAC_CNDC21	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x05B0	XDMAC_CUBC21	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x05B4	XDMAC_CBC21	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x05B8	XDMAC_CC21	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x05BC	XDMAC_CDS_MSP21	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x05C0	XDMAC_CSUS21	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x05C4	XDMAC_CDUS21	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x05C8 ... 0x05CF	Reserved									
0x05D0	XDMAC_CIE22	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x05D4	XDMAC_CID22	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x05D8	XDMAC_CIM22	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x05DC	XDMAC_CIS22	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x05E0	XDMAC_CSA22	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x05E4	XDMAC_CDA22	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x05E8	XDMAC_CNDA22	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x05EC	XDMAC_CNDC22	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x05F0	XDMAC_CUBC22	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x05F4	XDMAC_CBC22	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x05F8	XDMAC_CC22	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x05FC	XDMAC_CDS_MSP22	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0600	XDMAC_CSUS22	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0604	XDMAC_CDUS22	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0608 ... 0x060F	Reserved									
0x0610	XDMAC_CIE23	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0614	XDMAC_CID23	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0618	XDMAC_CIM23	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x061C	XDMAC_CIS23	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0620	XDMAC_CSA23	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0624	XDMAC_CDA23	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0628	XDMAC_CNDA23	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x062C	XDMAC_CNDC23	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0630	XDMAC_CUBC23	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0634	XDMAC_CBC23	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0638	XDMAC_CC23	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x063C	XDMAC_CDS_MSP23	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0640	XDMAC_CSUS23	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0644	XDMAC_CDUS23	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0648 ... 0x064F	Reserved									
0x0650	XDMAC_CIE24	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0654	XDMAC_CID24	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0658	XDMAC_CIM24	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x065C	XDMAC_CIS24	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0660	XDMAC_CSA24	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0664	XDMAC_CDA24	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0668	XDMAC_CNDA24	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x066C	XDMAC_CNDC24	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0670	XDMAC_CUBC24	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0674	XDMAC_CBC24	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0678	XDMAC_CC24	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x067C	XDMAC_CDS_MSP24	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0680	XDMAC_CSUS24	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0684	XDMAC_CDUS24	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0688 ... 0x068F	Reserved									
0x0690	XDMAC_CIE25	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0694	XDMAC_CID25	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0698	XDMAC_CIM25	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x069C	XDMAC_CIS25	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x06A0	XDMAC_CSA25	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x06A4	XDMAC_CDA25	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x06A8	XDMAC_CNDA25	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x06AC	XDMAC_CNDC25	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x06B0	XDMAC_CUBC25	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x06B4	XDMAC_CBC25	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x06B8	XDMAC_CC25	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x06BC	XDMAC_CDS_MSP25	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x06C0	XDMAC_CSUS25	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x06C4	XDMAC_CDUS25	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x06C8 ... 0x06CF	Reserved									
0x06D0	XDMAC_CIE26	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x06D4	XDMAC_CID26	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x06D8	XDMAC_CIM26	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x06DC	XDMAC_CIS26	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x06E0	XDMAC_CSA26	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x06E4	XDMAC_CDA26	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x06E8	XDMAC_CNDA26	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x06EC	XDMAC_CNDC26	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x06F0	XDMAC_CUBC26	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x06F4	XDMAC_CBC26	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x06F8	XDMAC_CC26	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x06FC	XDMAC_CDS_MSP26	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0700	XDMAC_CSUS26	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0704	XDMAC_CDUS26	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0708 ... 0x070F	Reserved									
0x0710	XDMAC_CIE27	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0714	XDMAC_CID27	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0718	XDMAC_CIM27	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x071C	XDMAC_CIS27	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0720	XDMAC_CSA27	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0724	XDMAC_CDA27	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0728	XDMAC_CNDA27	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x072C	XDMAC_CNDC27	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0730	XDMAC_CUBC27	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0734	XDMAC_CBC27	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0738	XDMAC_CC27	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x073C	XDMAC_CDS_MSP27	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0740	XDMAC_CSUS27	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0744	XDMAC_CDUS27	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0748 ... 0x074F	Reserved									
0x0750	XDMAC_CIE28	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0754	XDMAC_CID28	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0758	XDMAC_CIM28	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x075C	XDMAC_CIS28	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0760	XDMAC_CSA28	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0764	XDMAC_CDA28	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0768	XDMAC_CNDA28	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x076C	XDMAC_CNDC28	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0770	XDMAC_CUBC28	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0774	XDMAC_CBC28	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0778	XDMAC_CC28	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x077C	XDMAC_CDS_MSP28	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0780	XDMAC_CSUS28	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0784	XDMAC_CDUS28	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0788 ... 0x078F	Reserved									
0x0790	XDMAC_CIE29	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0794	XDMAC_CID29	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0798	XDMAC_CIM29	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x079C	XDMAC_CIS29	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x07A0	XDMAC_CSA29	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x07A4	XDMAC_CDA29	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x07A8	XDMAC_CNDA29	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x07AC	XDMAC_CNDC29	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x07B0	XDMAC_CUBC29	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x07B4	XDMAC_CBC29	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x07B8	XDMAC_CC29	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x07BC	XDMAC_CDS_MSP29	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x07C0	XDMAC_CSUS29	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x07C4	XDMAC_CDUS29	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x07C8 ... 0x07CF	Reserved									
0x07D0	XDMAC_CIE30	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x07D4	XDMAC_CID30	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x07D8	XDMAC_CIM30	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x07DC	XDMAC_CIS30	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x07E0	XDMAC_CSA30	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x07E4	XDMAC_CDA30	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x07E8	XDMAC_CNDA30	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x07EC	XDMAC_CNDC30	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x07F0	XDMAC_CUBC30	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x07F4	XDMAC_CBC30	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x07F8	XDMAC_CC30	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x07FC	XDMAC_CDS_MSP30	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0800	XDMAC_CSUS30	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0804	XDMAC_CDUS30	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								
0x0808 ... 0x080F	Reserved									
0x0810	XDMAC_CIE31	7:0		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
		15:8								
		23:16								
		31:24								
0x0814	XDMAC_CID31	7:0		ROID	WBEID	RBEID	FID	DID	LID	BID
		15:8								
		23:16								
		31:24								
0x0818	XDMAC_CIM31	7:0		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
		15:8								
		23:16								
		31:24								
0x081C	XDMAC_CIS31	7:0		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
		15:8								
		23:16								
		31:24								
0x0820	XDMAC_CSA31	7:0	SA[7:0]							
		15:8	SA[15:8]							
		23:16	SA[23:16]							
		31:24	SA[31:24]							
0x0824	XDMAC_CDA31	7:0	DA[7:0]							
		15:8	DA[15:8]							
		23:16	DA[23:16]							
		31:24	DA[31:24]							
0x0828	XDMAC_CNDA31	7:0	NDA[5:0]							NDAIF
		15:8	NDA[13:6]							
		23:16	NDA[21:14]							
		31:24	NDA[29:22]							
0x082C	XDMAC_CNDC31	7:0				NDVIEW[1:0]		NDDUP	NDSUP	NDE
		15:8								
		23:16								
		31:24								
0x0830	XDMAC_CUBC31	7:0	UBLEN[7:0]							
		15:8	UBLEN[15:8]							
		23:16	UBLEN[23:16]							
		31:24								
0x0834	XDMAC_CBC31	7:0	BLEN[7:0]							
		15:8					BLEN[11:8]			
		23:16								
		31:24								
0x0838	XDMAC_CC31	7:0	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
		15:8		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
		23:16	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
		31:24		PERID[6:0]						

.....continued

Offset	Name	Bit Pos.								
0x083C	XDMAC_CDS_MSP31	7:0	SDS_MSP[7:0]							
		15:8	SDS_MSP[15:8]							
		23:16	DDS_MSP[7:0]							
		31:24	DDS_MSP[15:8]							
0x0840	XDMAC_CSUS31	7:0	SUBS[7:0]							
		15:8	SUBS[15:8]							
		23:16	SUBS[23:16]							
		31:24								
0x0844	XDMAC_CDUS31	7:0	DUBS[7:0]							
		15:8	DUBS[15:8]							
		23:16	DUBS[23:16]							
		31:24								

33.9.1 XDMAC Global Type Register

Name: XDMAC_GTYPE
Offset: 0x00
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
		NB_REQ[6:0]						
Access		R	R	R	R	R	R	R
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FIFO_SZ[10:3]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FIFO_SZ[2:0]			NB_CH[4:0]				
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 22:16 – NB_REQ[6:0] Number of Peripheral Requests Minus One

Bits 15:5 – FIFO_SZ[10:0] Number of Bytes

Bits 4:0 – NB_CH[4:0] Number of Channels Minus One

33.9.2 XDMAC Global Configuration Register

Name: XDMAC_GCFG
Offset: 0x04
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								BXKBEN
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
					CGDISIF	CGDISFIFO	CGDISPIPE	CGDISREG
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit 8 – BXKBEN Boundary X Kilobyte Enable

Value	Description
0	The 1 Kbyte boundary is used.
1	The controller does not meet the AHB specification.

Bit 3 – CGDISIF Bus Interface Clock Gating Disable

Value	Description
0	The automatic clock gating is enabled for the system bus interface.
1	The automatic clock gating is disabled for the system bus interface.

Bit 2 – CGDISFIFO FIFO Clock Gating Disable

Value	Description
0	The automatic clock gating is enabled for the main FIFO.
1	The automatic clock gating is disabled for the main FIFO.

Bit 1 – CGDISPIPE Pipeline Clock Gating Disable

Value	Description
0	The automatic clock gating is enabled for the main pipeline.
1	The automatic clock gating is disabled for the main pipeline.

Bit 0 – CGDISREG Configuration Registers Clock Gating Disable

Value	Description
0	The automatic clock gating is enabled for the configuration registers.
1	The automatic clock gating is disabled for the configuration registers.

33.9.3 XDMAC Global Weighted Arbiter Configuration Register

Name: XDMAC_GWAC
Offset: 0x08
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	PW3[3:0]				PW2[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PW1[3:0]				PW0[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:12 – PW3[3:0] Pool Weight 3
This field indicates the weight of pool 3 in the arbitration scheme of the DMA scheduler.

Bits 11:8 – PW2[3:0] Pool Weight 2
This field indicates the weight of pool 2 in the arbitration scheme of the DMA scheduler.

Bits 7:4 – PW1[3:0] Pool Weight 1
This field indicates the weight of pool 1 in the arbitration scheme of the DMA scheduler.

Bits 3:0 – PW0[3:0] Pool Weight 0
This field indicates the weight of pool 0 in the arbitration scheme of the DMA scheduler.

33.9.4 XDMAC Global Interrupt Enable Register

Name: XDMAC_GIE
Offset: 0x0C
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	IE31	IE30	IE29	IE28	IE27	IE26	IE25	IE24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	IE23	IE22	IE21	IE20	IE19	IE18	IE17	IE16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	IE15	IE14	IE13	IE12	IE11	IE10	IE9	IE8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – IE
XDMAC Channel x Interrupt Enable

Value	Description
0	This bit has no effect. The Channel x Interrupt Mask bit (XDMAC_GIM.IMx) is not modified.
1	The corresponding mask bit is set. The XDMAC Channel x Interrupt Status register (XDMAC_GIS) can generate an interrupt.

33.9.5 XDMAC Global Interrupt Disable Register

Name: XDMAC_GID
Offset: 0x10
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	ID31	ID30	ID29	ID28	ID27	ID26	ID25	ID24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – ID XDMAC Channel x Interrupt Disable

Value	Description
0	This bit has no effect. The Channel x Interrupt Mask bit (XDMAC_GIM.IMx) is not modified.
1	The corresponding mask bit is reset. The Channel x Interrupt Status register interrupt (XDMAC_GIS) is masked.

33.9.6 XDMAC Global Interrupt Mask Register

Name: XDMAC_GIM
Offset: 0x14
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	IM31	IM30	IM29	IM28	IM27	IM26	IM25	IM24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	IM23	IM22	IM21	IM20	IM19	IM18	IM17	IM16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – IM XDMAC Channel x Interrupt Mask

Value	Description
0	This bit indicates that the channel x interrupt source is masked. The interrupt line is not raised.
1	This bit indicates that the channel x interrupt source is unmasked.

33.9.7 XDMAC Global Interrupt Status Register

Name: XDMAC_GIS
Offset: 0x18
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	IS31	IS30	IS29	IS28	IS27	IS26	IS25	IS24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	IS23	IS22	IS21	IS20	IS19	IS18	IS17	IS16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IS15	IS14	IS13	IS12	IS11	IS10	IS9	IS8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – IS
XDMAC Channel x Interrupt Status

Value	Description
0	This bit indicates that either the interrupt source is masked at the channel level or no interrupt is pending for channel x.
1	This bit indicates that an interrupt is pending for the channel x.

33.9.8 XDMAC Global Channel Enable Register

Name: XDMAC_GE
Offset: 0x1C
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	EN31	EN30	EN29	EN28	EN27	EN26	EN25	EN24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	EN23	EN22	EN21	EN20	EN19	EN18	EN17	EN16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – EN XDMAC Channel x Enable

Value	Description
0	This bit has no effect.
1	Enables channel n. This operation is permitted if the Channel x Status bit (XDMAC_GS.STx) was read as '0'.

33.9.9 XDMAC Global Channel Disable Register

Name: XDMAC_GD
Offset: 0x20
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – DI XDMAC Channel x Disable

Value	Description
0	This bit has no effect.
1	Disables channel x.

33.9.10 XDMAC Global Channel Status Register

Name: XDMAC_GS
Offset: 0x24
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ST31	ST30	ST29	ST28	ST27	ST26	ST25	ST24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ST23	ST22	ST21	ST20	ST19	ST18	ST17	ST16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ST15	ST14	ST13	ST12	ST11	ST10	ST9	ST8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ST7	ST6	ST5	ST4	ST3	ST2	ST1	ST0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – ST XDMAC Channel x Status

Value	Description
0	This bit indicates that the channel x is disabled.
1	This bit indicates that the channel x is enabled. If a channel disable request is issued, this bit remains asserted until pending transaction is completed.

33.9.11 XDMAC Global Channel Read Suspend Register

Name: XDMAC_GRS
Offset: 0x28
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	RS31	RS30	RS29	RS28	RS27	RS26	RS25	RS24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RS23	RS22	RS21	RS20	RS19	RS18	RS17	RS16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RS15	RS14	RS13	RS12	RS11	RS10	RS9	RS8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RS7	RS6	RS5	RS4	RS3	RS2	RS1	RS0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – RSx XDMAC Channel x Read Suspend

Value	Description
0	The read channel is not suspended.
1	The source requests for channel n are no longer serviced by the system scheduler.

33.9.12 XDMAC Global Channel Write Suspend Register

Name: XDMAC_GWS
Offset: 0x2C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WS31	WS30	WS29	WS28	WS27	WS26	WS25	WS24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WS23	WS22	WS21	WS20	WS19	WS18	WS17	WS16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WS15	WS14	WS13	WS12	WS11	WS10	WS9	WS8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	WS7	WS6	WS5	WS4	WS3	WS2	WS1	WS0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – WSx XDMAC Channel x Write Suspend

Value	Description
0	The write channel is not suspended.
1	Destination requests are no longer routed to the scheduler.

33.9.13 XDMAC Global Channel Read Write Suspend Register

Name: XDMAC_GRWS
Offset: 0x30
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	RWS31	RWS30	RWS29	RWS28	RWS27	RWS26	RWS25	RWS24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	RWS23	RWS22	RWS21	RWS20	RWS19	RWS18	RWS17	RWS16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	RWS15	RWS14	RWS13	RWS12	RWS11	RWS10	RWS9	RWS8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	RWS7	RWS6	RWS5	RWS4	RWS3	RWS2	RWS1	RWS0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – RWSx XDMAC Channel x Read Write Suspend

Value	Description
0	No effect.
1	Read and write requests are suspended.

33.9.14 XDMAC Global Channel Read Write Resume Register

Name: XDMAC_GRWR
Offset: 0x34
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	RWR31	RWR30	RWR29	RWR28	RWR27	RWR26	RWR25	RWR24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	RWR23	RWR22	RWR21	RWR20	RWR19	RWR18	RWR17	RWR16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	RWR15	RWR14	RWR13	RWR12	RWR11	RWR10	RWR9	RWR8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	RWR7	RWR6	RWR5	RWR4	RWR3	RWR2	RWR1	RWR0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – RWRx XDMAC Channel x Read Write Resume

Value	Description
0	No effect.
1	Read and write requests are serviced.

33.9.15 XDMAC Global Channel Software Request Register

Name: XDMAC_GSWR
Offset: 0x38
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	SWREQ31	SWREQ30	SWREQ29	SWREQ28	SWREQ27	SWREQ26	SWREQ25	SWREQ24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	SWREQ23	SWREQ22	SWREQ21	SWREQ20	SWREQ19	SWREQ18	SWREQ17	SWREQ16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	SWREQ15	SWREQ14	SWREQ13	SWREQ12	SWREQ11	SWREQ10	SWREQ9	SWREQ8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	SWREQ7	SWREQ6	SWREQ5	SWREQ4	SWREQ3	SWREQ2	SWREQ1	SWREQ0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – SWREQ XDMAC Channel x Software Request

Value	Description
0	No effect.
1	Requests a DMA transfer for channel x.

33.9.16 XDMAC Global Channel Software Request Status Register

Name: XDMAC_GSWS
Offset: 0x3C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	SWRS31	SWRS30	SWRS29	SWRS28	SWRS27	SWRS26	SWRS25	SWRS24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	SWRS23	SWRS22	SWRS21	SWRS20	SWRS19	SWRS18	SWRS17	SWRS16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SWRS15	SWRS14	SWRS13	SWRS12	SWRS11	SWRS10	SWRS9	SWRS8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SWRS7	SWRS6	SWRS5	SWRS4	SWRS3	SWRS2	SWRS1	SWRS0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – SWRS XDMAC Channel x Software Request Status

Value	Description
0	Channel x source request is serviced.
1	Channel x source request is pending.

33.9.17 XDMAC Global Channel Software Flush Request Register

Name: XDMAC_GSWF
Offset: 0x40
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	SWF31	SWF30	SWF29	SWF28	SWF27	SWF26	SWF25	SWF24
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	SWF23	SWF22	SWF21	SWF20	SWF19	SWF18	SWF17	SWF16
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	SWF15	SWF14	SWF13	SWF12	SWF11	SWF10	SWF9	SWF8
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	SWF7	SWF6	SWF5	SWF4	SWF3	SWF2	SWF1	SWF0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 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 – SWFx XDMAC Channel x Software Flush Request

Value	Description
0	No effect.
1	Requests a DMA transfer flush for channel x. This bit is only relevant when the transfer is source peripheral synchronized.

33.9.18 XDMAC Channel x Interrupt Enable Register [x=0..31]

Name: XDMAC_CIE
Offset: 0x50 + n*0x40 [n=0..31]
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE
Access		W	W	W	W	W	W	W
Reset		–	–	–	–	–	–	–

Bit 6 – ROIE Request Overflow Error Interrupt Enable Bit

Value	Description
0	No effect.
1	Enables request overflow error interrupt.

Bit 5 – WBIE Write Bus Error Interrupt Enable Bit

Value	Description
0	No effect.
1	Enables write bus error interrupt.

Bit 4 – RBIE Read Bus Error Interrupt Enable Bit

Value	Description
0	No effect.
1	Enables read bus error interrupt.

Bit 3 – FIE End of Flush Interrupt Enable Bit

Value	Description
0	No effect.
1	Enables end of flush interrupt.

Bit 2 – DIE End of Disable Interrupt Enable Bit

Value	Description
0	No effect.
1	Enables end of disable interrupt.

Bit 1 – LIE End of Linked List Interrupt Enable Bit

Value	Description
0	No effect.
1	Enables end of linked list interrupt.

Bit 0 – BIE End of Block Interrupt Enable Bit

Value	Description
0	No effect.
1	Enables end of block interrupt.

33.9.19 XDMAC Channel x Interrupt Disable Register [x = 0..31]

Name: XDMAC_CID
Offset: 0x54 + n*0x40 [n=0..31]
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		ROID	WBEID	RBEID	FID	DID	LID	BID
Access		W	W	W	W	W	W	W
Reset		–	–	–	–	–	–	–

Bit 6 – ROID Request Overflow Error Interrupt Disable Bit

Value	Description
0	No effect.
1	Disables request overflow error interrupt.

Bit 5 – WBEID Write Bus Error Interrupt Disable Bit

Value	Description
0	No effect.
1	Disables bus error interrupt.

Bit 4 – RBEID Read Bus Error Interrupt Disable Bit

Value	Description
0	No effect.
1	Disables bus error interrupt.

Bit 3 – FID End of Flush Interrupt Disable Bit

Value	Description
0	No effect.
1	Disables end of flush interrupt.

Bit 2 – DID End of Disable Interrupt Disable Bit

Value	Description
0	No effect.
1	Disables end of disable interrupt.

Bit 1 – LID End of Linked List Interrupt Disable Bit

Value	Description
0	No effect.
1	Disables end of linked list interrupt.

Bit 0 – BID End of Block Interrupt Disable Bit

Value	Description
0	No effect.
1	Disables end of block interrupt.

33.9.20 XDMAC Channel x Interrupt Mask Register [x = 0..31]

Name: XDMAC_CIM
Offset: 0x58 + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM
Access		R	R	R	R	R	R	R
Reset		0	0	0	0	0	0	0

Bit 6 – ROIM Request Overflow Error Interrupt Mask Bit

Value	Description
0	Request overflow interrupt is masked.
1	Request overflow interrupt is activated.

Bit 5 – WBEIM Write Bus Error Interrupt Mask Bit

Value	Description
0	Bus error interrupt is masked.
1	Bus error interrupt is activated.

Bit 4 – RBEIM Read Bus Error Interrupt Mask Bit

Value	Description
0	Bus error interrupt is masked.
1	Bus error interrupt is activated.

Bit 3 – FIM End of Flush Interrupt Mask Bit

Value	Description
0	End of flush interrupt is masked.
1	End of flush interrupt is activated.

Bit 2 – DIM End of Disable Interrupt Mask Bit

Value	Description
0	End of disable interrupt is masked.
1	End of disable interrupt is activated.

Bit 1 – LIM End of Linked List Interrupt Mask Bit

Value	Description
0	End of linked list interrupt is masked.
1	End of linked list interrupt is activated.

Bit 0 – BIM End of Block Interrupt Mask Bit

Value	Description
0	Block interrupt is masked.
1	Block interrupt is activated.

33.9.21 XDMAC Channel x Interrupt Status Register [x = 0..31]

Name: XDMAC_CIS
Offset: 0x5C + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS
Access		R	R	R	R	R	R	R
Reset		0	0	0	0	0	0	0

Bit 6 – ROIS Request Overflow Error Interrupt Status Bit

Value	Description
0	Overflow condition has not occurred.
1	Overflow condition has occurred at least once. (This information is only relevant for peripheral synchronized transfers.)

Bit 5 – WBEIS Write Bus Error Interrupt Status Bit

Value	Description
0	Write bus error condition has not occurred.
1	At least one bus error has been detected in a write access since the last read of the Status register.

Bit 4 – RBEIS Read Bus Error Interrupt Status Bit

Value	Description
0	Read bus error condition has not occurred.
1	At least one bus error has been detected in a read access since the last read of the Status register.

Bit 3 – FIS End of Flush Interrupt Status Bit

Value	Description
0	End of flush condition has not occurred.
1	End of flush condition has occurred since the last read of the Status register.

Bit 2 – DIS End of Disable Interrupt Status Bit

Value	Description
0	End of disable condition has not occurred.
1	End of disable condition has occurred since the last read of the Status register.

Bit 1 – LIS End of Linked List Interrupt Status Bit

Value	Description
0	End of linked list condition has not occurred.

Value	Description
1	End of linked list condition has occurred since the last read of the Status register.

Bit 0 – BIS End of Block Interrupt Status Bit

Value	Description
0	End of block interrupt has not occurred.
1	End of block interrupt has occurred since the last read of the Status register.

33.9.22 XDMAC Channel x Source Address Register [x = 0..31]

Name: XDMAC_CSA
Offset: 0x60 + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	SA[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	SA[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SA[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – SA[31:0] Channel x Source Address
Program this register with the source address of the DMA transfer.
A configuration error is generated when this address is not aligned with the transfer data size.

33.9.23 XDMAC Channel x Destination Address Register [x = 0..31]

Name: XDMAC_CDA
Offset: 0x64 + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	DA[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DA[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DA[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – DA[31:0] Channel x Destination Address
Program this register with the destination address of the DMA transfer.
A configuration error is generated when this address is not aligned with the transfer data size.

33.9.24 XDMAC Channel x Next Descriptor Address Register [x = 0..31]

Name: XDMAC_CNDA
Offset: 0x68 + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	NDA[29:22]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NDA[21:14]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NDA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NDA[5:0]							NDAIF
Access	R/W	R/W	R/W	R/W	R/W	R/W		R/W
Reset	0	0	0	0	0	0		0

Bits 31:2 – NDA[29:0] Channel x Next Descriptor Address
The 30-bit width of the NDA field represents the next descriptor address range 31:2. The descriptor is word-aligned and the two least significant register bits 1:0 are ignored.

Bit 0 – NDAIF Channel x Next Descriptor Interface

Value	Description
0	The channel descriptor is retrieved through system interface 0.
1	The channel descriptor is retrieved through system interface 1.

33.9.25 XDMAC Channel x Next Descriptor Control Register [x = 0..31]

Name: XDMAC_CNDC
Offset: 0x6C + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				NDVIEW[1:0]		NDDUP	NDSUP	NDE
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 4:3 – NDVIEW[1:0] Channel x Next Descriptor View

Value	Name	Description
0	NDV0	Next Descriptor View 0
1	NDV1	Next Descriptor View 1
2	NDV2	Next Descriptor View 2
3	NDV3	Next Descriptor View 3

Bit 2 – NDDUP Channel x Next Descriptor Destination Update
0 (DST_PARAMS_UNCHANGED): Destination parameters remain unchanged.
1 (DST_PARAMS_UPDATED): Destination parameters are updated when the descriptor is retrieved.

Bit 1 – NDSUP Channel x Next Descriptor Source Update
0 (SRC_PARAMS_UNCHANGED): Source parameters remain unchanged.
1 (SRC_PARAMS_UPDATED): Source parameters are updated when the descriptor is retrieved.

Bit 0 – NDE Channel x Next Descriptor Enable
0 (DSCR_FETCH_DIS): Descriptor fetch is disabled.
1 (DSCR_FETCH_EN): Descriptor fetch is enabled.

33.9.26 XDMAC Channel x Microblock Control Register [x = 0..31]

Name: XDMAC_CUBC
Offset: 0x70 + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	UBLEN[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	UBLEN[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	UBLEN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – UBLEN[23:0] Channel x Microblock Length
This field indicates the number of data in the microblock. The microblock contains UBLEN data.

33.9.27 XDMAC Channel x Block Control Register [x = 0..31]

Name: XDMAC_CBC
Offset: 0x74 + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					BLEN[11:8]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BLEN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 11:0 – BLEN[11:0] Channel x Block Length
The length of the block is (BLEN+1) microblocks.

33.9.28 XDMAC Channel x Configuration Register [x = 0..31]

Name: XDMAC_CC
Offset: 0x78 + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
		PERID[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WRIP	RDIP	INITD		DAM[1:0]		SAM[1:0]	
Access	R/W	R/W	R/W		R/W	R/W	R/W	R/W
Reset	0	0	0		0	0	0	0

Bit	15	14	13	12	11	10	9	8
		DIF	SIF	DWIDTH[1:0]		CSIZE[2:0]		
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MEMSET	SWREQ	PROT	DSYNC		MBSIZE[1:0]		TYPE
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset	0	0	0	0		0	0	0

Bits 30:24 – PERID[6:0] Channel x Peripheral Hardware Request Line Identifier
This field contains the peripheral hardware request line identifier. PERID refers to identifiers defined in "DMA Controller Peripheral Connections".
Note: When a memory-to-memory transfer is performed, configure PERID to an unused peripheral ID (refer to table "Peripheral Identifiers").

Bit 23 – WRIP Write in Progress (this bit is read-only)
0 (DONE): No active write transaction on the bus.
1 (IN_PROGRESS): A write transaction is in progress.

Bit 22 – RDIP Read in Progress (this bit is read-only)
0 (DONE): No active read transaction on the bus.
1 (IN_PROGRESS): A read transaction is in progress.

Bit 21 – INITD Channel Initialization Done (this bit is read-only)
0 (IN_PROGRESS): Channel initialization is in progress.
1 (TERMINATED): Channel initialization is completed.
Note: When set to 0, XDMAC_CUBC.UBLEN and XDMAC_CNDA.NDA field values are unreliable each time a descriptor is being updated. See 33.8 XDMAC Software Requirements.

Bits 19:18 – DAM[1:0] Channel x Destination Addressing Mode

Value	Name	Description
0	FIXED_AM	The address remains unchanged.
1	INCREMENTED_AM	The addressing mode is incremented (the increment size is set to the data size).
2	UBS_AM	The microblock stride is added at the microblock boundary.
3	UBS_DS_AM	The microblock stride is added at the microblock boundary; the data stride is added at the data boundary.

Bits 17:16 – SAM[1:0] Channel x Source Addressing Mode

Value	Name	Description
0	FIXED_AM	The address remains unchanged.
1	INCREMENTED_AM	The addressing mode is incremented (the increment size is set to the data size).
2	UBS_AM	The microblock stride is added at the microblock boundary.
3	UBS_DS_AM	The microblock stride is added at the microblock boundary, the data stride is added at the data boundary.

Bit 14 – DIF Channel x Destination Interface Identifier
0 (AHB_IF0): The data is written through system bus interface 0.
1 (AHB_IF1): The data is written though system bus interface 1.

Bit 13 – SIF Channel x Source Interface Identifier
0 (AHB_IF0): The data is read through system bus interface 0.
1 (AHB_IF1): The data is read through system bus interface 1.

Bits 12:11 – DWIDTH[1:0] Channel x Data Width

Value	Name	Description
0	BYTE	The data size is set to 8 bits
1	HALFWORD	The data size is set to 16 bits
2	WORD	The data size is set to 32 bits

Bits 10:8 – CSIZE[2:0] Channel x Chunk Size

Value	Name	Description
0	CHK_1	1 data transferred
1	CHK_2	2 data transferred
2	CHK_4	4 data transferred
3	CHK_8	8 data transferred
4	CHK_16	16 data transferred

Bit 7 – MEMSET Channel x Fill Block of Memory
0 (NORMAL_MODE): Memset is not activated.
1 (HW_MODE): Sets the block of memory pointed by DA field to the specified value. This operation is performed on 8-, 16- or 32-bit basis.

Bit 6 – SWREQ Channel x Software Request Trigger
0 (HWR_CONNECTED): Hardware request line is connected to the peripheral request line.
1 (SWR_CONNECTED): Software request is connected to the peripheral request line.

Bit 5 – PROT Channel x Protection
0 (PRIVILEGED): Channel uses Privileged mode.
1 (USER): Channel uses User mode.

Bit 4 – DSYNC Channel x Synchronization
0 (PER2MEM): Peripheral-to-memory transfer.
1 (MEM2PER): Memory-to-peripheral transfer.

Bits 2:1 – MBSIZE[1:0] Channel x Memory Burst Size

Value	Name	Description
0	SINGLE	The memory burst size is set to one.
1	FOUR	The memory burst size is set to four.
2	EIGHT	The memory burst size is set to eight.
3	SIXTEEN	The memory burst size is set to sixteen.

Bit 0 – TYPE Channel x Transfer Type
0 (MEM_TRAN): Self-triggered mode (memory-to-memory transfer).
1 (PER_TRAN): Synchronized mode (peripheral-to-memory or memory-to-peripheral transfer).

33.9.29 XDMAC Channel x Data Stride Memory Set Pattern Register [x = 0..31]

Name: XDMAC_CDS_MSP
Offset: 0x7C + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	DDS_MSP[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DDS_MSP[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SDS_MSP[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SDS_MSP[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:16 – DDS_MSP[15:0] Channel x Destination Data Stride or Memory Set Pattern
When XDMAC_CCx.MEMSET = 0, this field indicates the destination data stride.
When XDMAC_CCx.MEMSET = 1, this field indicates the memory set pattern.

Bits 15:0 – SDS_MSP[15:0] Channel x Source Data stride or Memory Set Pattern
When XDMAC_CCx.MEMSET = 0, this field indicates the source data stride.
When XDMAC_CCx.MEMSET = 1, this field indicates the memory set pattern.

33.9.30 XDMAC Channel x Source Microblock Stride Register [x = 0..31]

Name: XDMAC_CSUS
Offset: 0x80 + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	SUBS[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SUBS[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SUBS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – SUBS[23:0] Channel x Source Microblock Stride
Two’s complement microblock stride for channel x.

33.9.31 XDMAC Channel x Destination Microblock Stride Register [x = 0..31]

Name: XDMAC_CDUS
Offset: 0x84 + n*0x40 [n=0..31]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	DUBS[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DUBS[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DUBS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – DUBS[23:0] Channel x Destination Microblock Stride
Two’s complement microblock stride for channel x.

34. 1553 Interface (1553)

34.1 Description

The 1553 link is an interface commonly used in space applications. It features one nominal bus and one redundant bus to obtain reliable transfers and it is capable of carrying out data transfers at a bit rate of 1 Mbits/sec. A typical network is made of a Bus Controller (BC), a Bus Monitor (BM) and up to 31 Remote Terminals (RT).

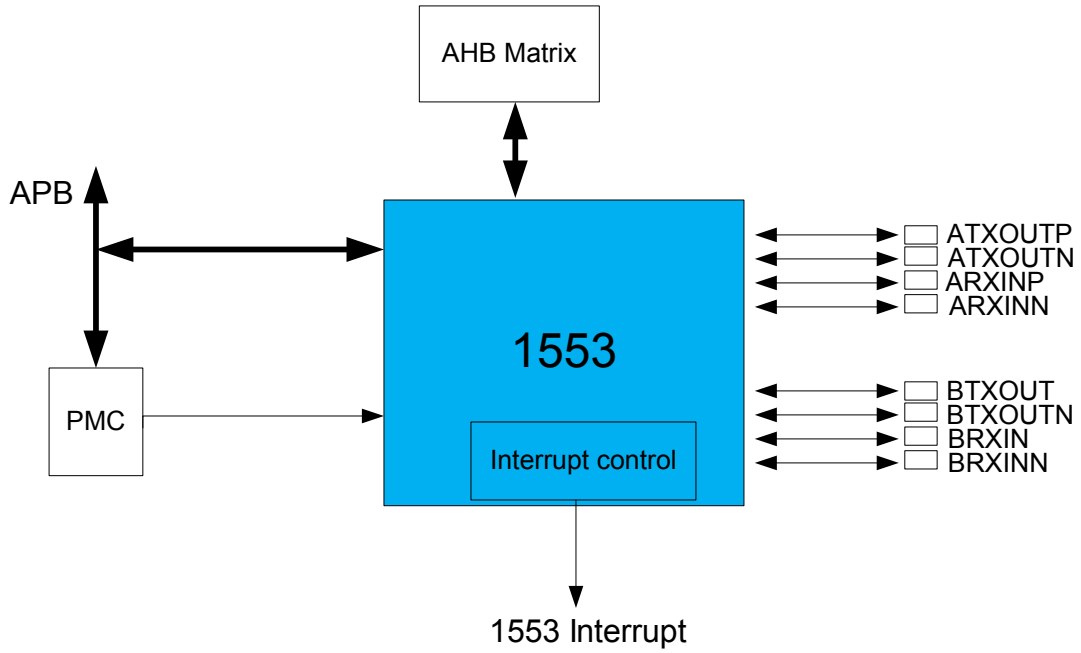
The 1553 interface embeds three configurations of the Bus Controller, Remote Terminal and Bus Monitor as defined in the MIL-STD-1553B standard.

34.2 Embedded Characteristics

- Compliant to MIL-STD-1553B Standard
- Bus Controller and Remote Terminal Configurations Available
- 1553 Data Rate: 1 Mbits/sec
- Command and Data Transactions Available
- Data Transaction: BC to RT, RT to BC, and RT to RT
- Secure Link with a Redundant Interface

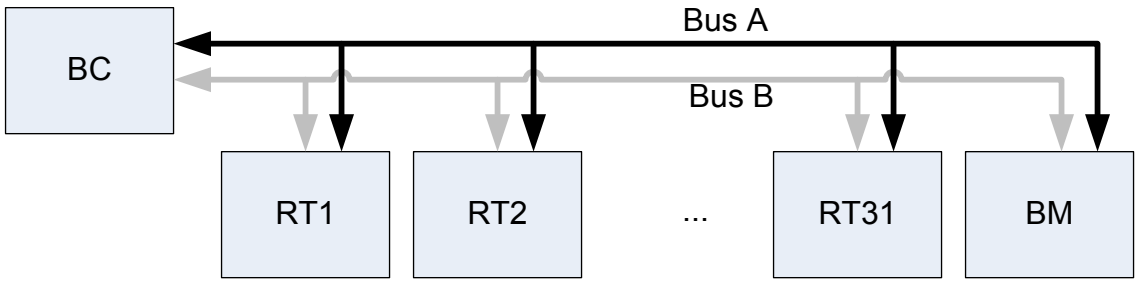
34.3 Block Diagram

Figure 34-1. Block Diagram



A typical 1553 network topology is presented in the figure below.

Figure 34-2. Network Diagram



34.4 Pin List

Table 34-1. Pin List

Pin Name	Description	Type
ARXINP	Nominal reception line (from transceiver)	Input
ARXINN	Complemented nominal reception line (from transceiver)	Input
BRXINP	Redundant reception line (from transceiver)	Input
BRXIXN	Complemented redundant reception line (from transceiver)	Input
ATXOUTP	Nominal transmission line (to transceiver)	Output
ATXOUTN	Complemented nominal transmission line (to transceiver)	Output
BTXOUTP	Redundant transmission line (to transceiver)	Output
BTXOUTN	Complemented redundant transmission line (to transceiver)	Output

34.5 IP1553 Dependencies

34.5.1 I/O Lines

There are no dependencies regarding PIO muxing.

34.5.2 Power Management

The 1553 is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the 1553 clock.

The 1553 peripheral must have a 12 MHz input frequency on peripheral GCLK to comply with 1553 protocol (1Mbit/s baud rate).

The MCK clock frequency should be at least twice the GCLK frequency.

34.5.3 Interrupt Sources

The 1553 has an interrupt line connected to the interrupt controller. Handling the 1553 interrupt requires programming the interrupt controller before configuring the 1553.

34.6 Functional Description

34.6.1 Modes of Operation

The 1553 device can be used as either of the following:

- Bus Controller (BC)
- Remote Terminal (RT)

Three types of transactions can be performed on the 1553 network at a data rate of 1 Mbits/s:

- Command frames (from BC to RT)
- Status frames (from RT to BC)
- Data frames (from BC to RT, RT to BC, or RT to RT)

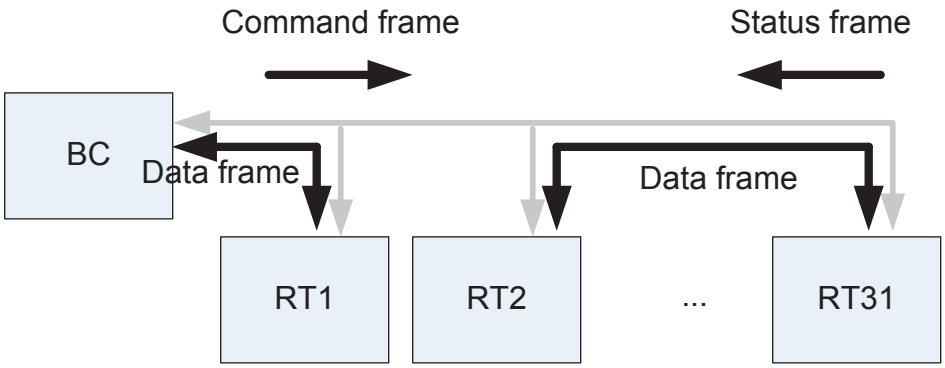
The BC always initiates transactions on the 1553 network by sending command frames. It also performs data frames.

The RT acknowledges the command frames by sending back a status frame. It also performs data frames.

The BC can be part of the transfer (BC to RT, RT to BC) or can only arbitrate the data transfer (RT to RT).

The different 1553 transactions are summarized in the following figure.

Figure 34-3. 1553 Frames



34.6.2 Memory Interface

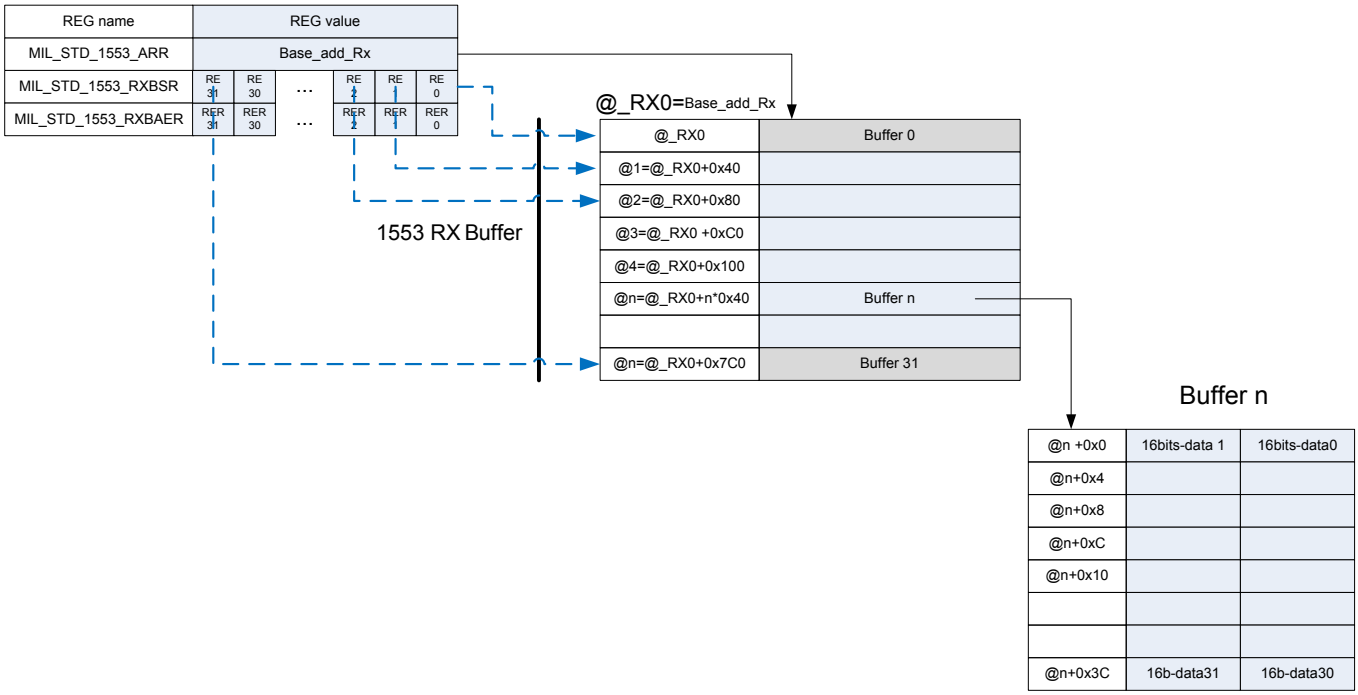
Each device has dedicated buffers to receive or transmit data.

Data, which are transferred from and to the 1553 link by packets, are stored in a 2-Kbyte 1553 RX buffer and an 2-Kbyte 1553 TX buffer. These buffers are located either in internal or external memory.

34.6.2.1 1553 RX Buffer Structure

The following figure describes the 1553 RX buffer and its management.

Figure 34-4. 1553 Rx Buffer



Note: Buffers 0 and 31 are never used for data purposes (see the 1553 protocol for details).

The 1553 RX buffer consists of 32 buffers. Each buffer can store up to 64 bytes of data (32 *16 bits data), which is the maximum amount of data transferred during a 1553 transaction.

34.6.2.2 1553 RX Buffer Management

Three registers are used to control the 1553 RX buffer:

- IP1553_ARR—used to set the base address of the 1553 RX buffer.
- IP1553_RXBSR—used to indicate the state of the buffer (‘1’: empty, ‘0’; full)
- IP1553_RXBAER—used to notify if errors are detected in the buffer (‘1’: errors, ‘0’: no error). See the section “Error Detection” for details.

34.6.2.3 1553 RX Buffer Setup

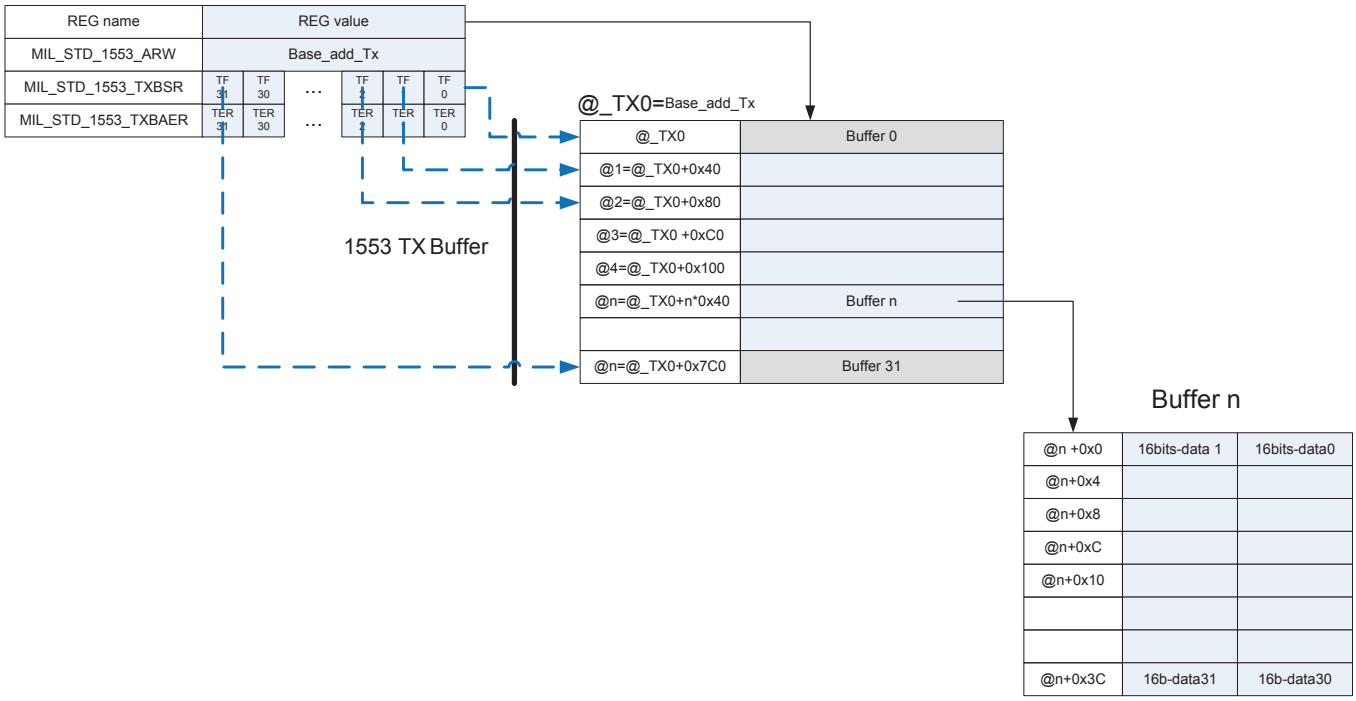
To set up the 1553 RX buffer, follow the steps below:

1. Specify the base address register of the 1553 RX buffer in the IP1553_ARR register.
2. Reset the content of the 32 buffers.
3. Clear the error register in reception: IP1553_RXBAER.
4. Set the status of each buffer in the IP1553_RXBSR register to ‘1’ to indicate that it is ready to receive data.
5. Enable the 1553 transfer.

34.6.2.4 1553 TX Buffer Structure

The following figure describes the 1553 TX buffer and its management.

Figure 34-5. 1553 Tx Buffer



Note: Buffers 0 and 31 are never used (see 1553 protocol for details).

The 1553 TX buffer consists of 32 buffers. Each buffer can store up to 64 bytes of data (32 *16 bits data), which is the maximum amount of data transferred during a 1553 single transfer.

34.6.2.5 1553 TX Buffer Management

Three registers are used to control the 1553 TX buffer:

- IP1553_ARW—used to set the base address of the 1553 TX buffer.
- IP1553_TXBSR—used to indicate the state of the buffer (‘0’: empty, ‘1’; data ready to send)
- IP1553_TXBAER— used to notify if errors are detected in the buffer (‘1’: errors, ‘0’: no error). See the section “Error Detection” for details.

34.6.2.6 1553 TX Buffer Setup

To set up the 1553 TX buffer, follow the steps below:

1. Specify the base address register of the 1553 TX buffer in the IP1553_ARW register.
2. Clear the error register in transmission: IP1553_TXBAER.
3. Set the status of each buffer in the IP1553_TXBSR register to ‘0’ to indicate that it is free.
4. Fill the buffer with the data to send.
5. Set the IP1553_TXBSR register to ‘1’ for the corresponding buffer to indicate that data are ready to send.
6. Enable the 1553 transfer.

34.6.3 Bus Controller (BC) Mode

Two registers are used to generate the command frames:

- IP1553_CMDR1—relative to the receiver device.
- IP1553_CMDR2—relative to the transmitter device.

The register IP1553_CMDR3 is used to configure the 1553 transaction:

- IP1553_CMDR3.BUS—defines the bus used
- IP1553_CMDR3.BCE—defines if the product is the emitter of data or not
- IP1553_CMDR3.BCR—defines if the product is the receiver of data or not

- IP1553_CMDR3.ER—defines if the transfer initiated is a command transfer or a data transfer. It shall be the last to be written

Note: The IP1553 can be configured as neither emitter nor receiver. In this case, an RT to RT transfer is performed.

One register is used to store the status frames:

- IP1553_CTRL1—stores the value of the first status word received during the 1553 transaction. The register stores also the value of the second status word received during the 1553 transaction.

34.6.3.1 Data Frame Transfer

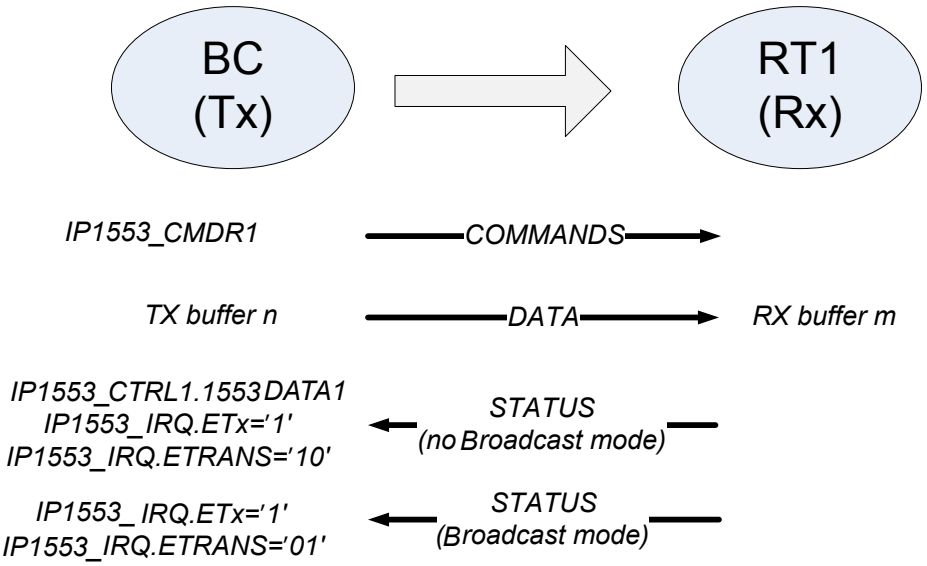
Three configurations are possible for a data frame transfer:

BC as transmitter and RT1 as receiver

Commands are sent from BC to RT1 via BC CMDR1 register.

Data are sent from BC to RT1 from TX buffer n to RX buffer m.

Once all data have been transmitted, the status is updated in return depending whether Broadcast mode is enabled or not.



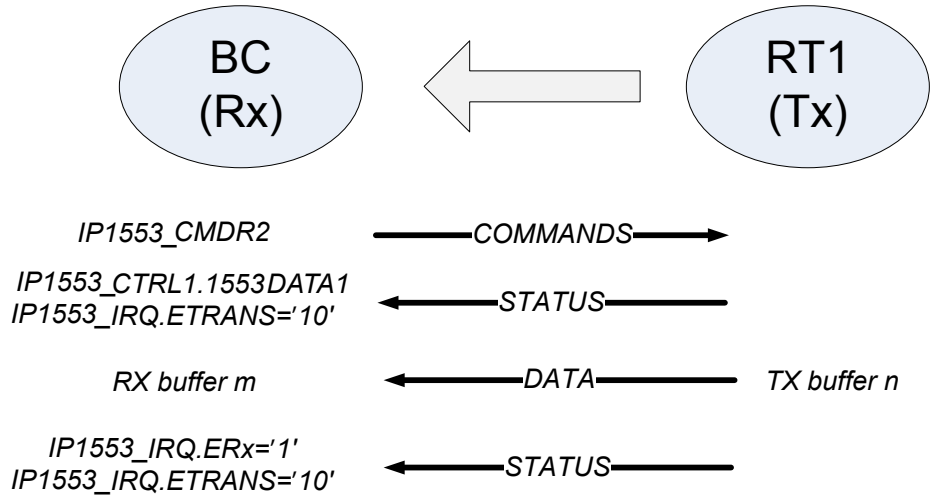
RT1 is considered as transmitter and BC as receiver

Commands are sent from RT1 to BC via the register IP1553_CMDR2.

Status is sent to from RT1 to BC in returns and stored in IP1553_CTRL1.IP1553DATA1.

Data are sent from RT1 to BC from TX buffer n to RX buffer m.

Once all data have been transmitted, the status is updated in return.



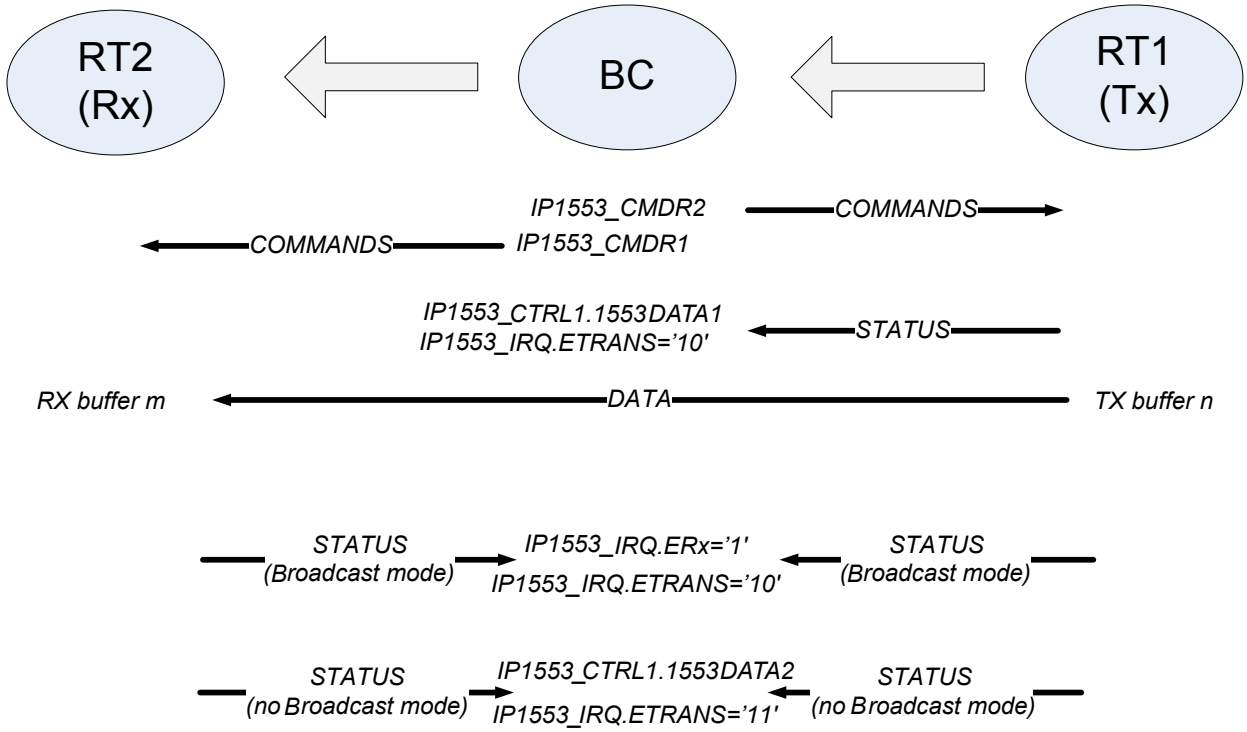
RT1 is considered as transmitter and RT2 as receiver

Commands are sent to RT2 via CMDR1 register and to RT1 via the register IP1553_CMDR2.

Status is sent to from RT1 to BC in returns and stored in IP1553_CTRL1.1553DATA1.

Data are sent from RT1 to RT2 from TX buffer n to RX buffer m.

Status is sent to from RT1 and RT2 to BC in return and stored in IP1553_CTRL1.1553DATA2 if Broadcast mode is not enabled.



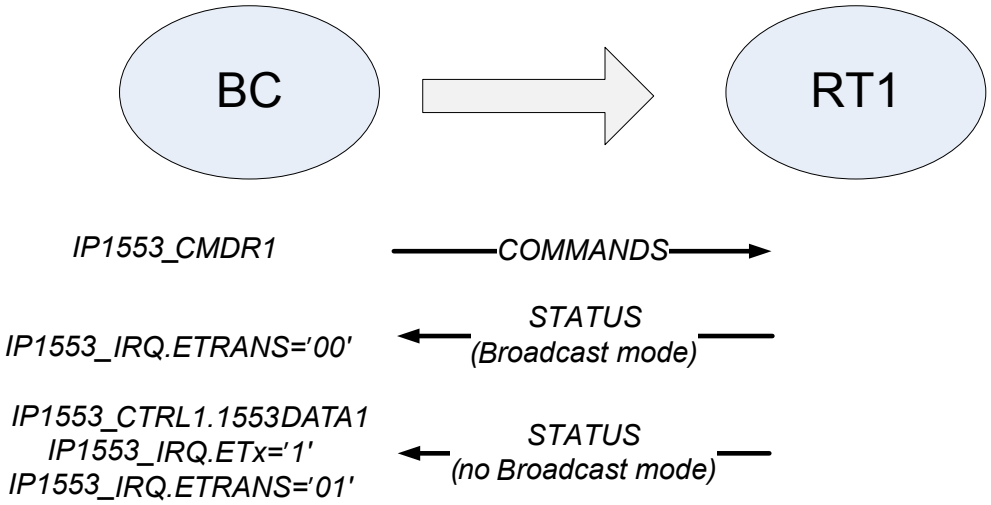
34.6.3.2 Command Frame Transfer

Three configurations are possible for a command frame transfer:

Command mode does not include data

The command is sent via the register IP1553_CMDR1.

Status is collected via IP1553_CRTL1.1553DATA1, if Broadcast mode is not enabled.

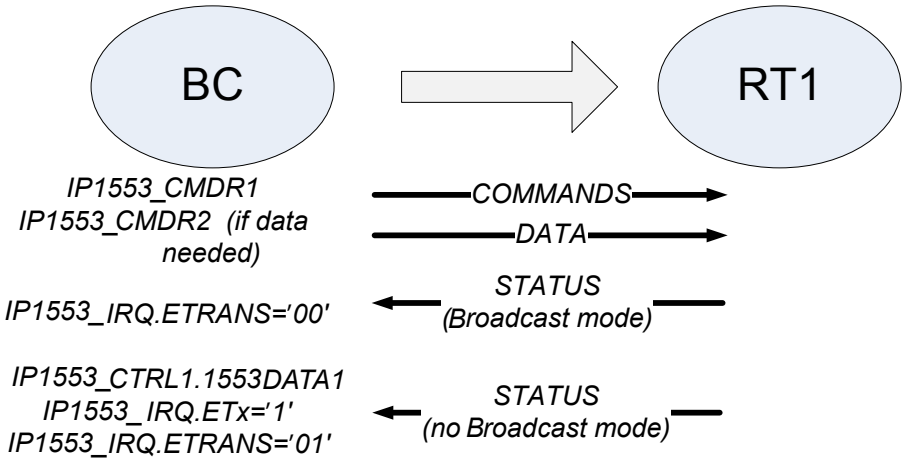


Command mode includes data to be sent to the receiver

The command is sent via the register IP1553_CMDR1.

Data is sent via the register IP1553_CMDR2.

Status is collected via IP1553_CRTL1.1553DATA1, if Broadcast mode is not enabled.



Command mode includes data to be sent by the receiver

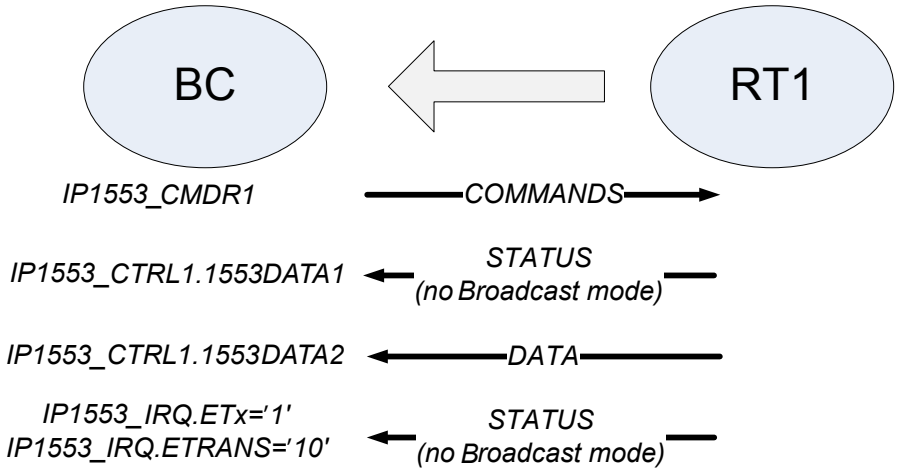
The command is sent via the register IP1553_CMDR1.

Status is collected via IP1553_CRTL1.1553DATA1.

Data is sent via the register IP1553_CMDR2.

Status is collected via IP1553_CRTL1.1553DATA1 if Broadcast mode is not enabled.

Data is collected via IP1553_CTRL1.1553DATA2 and the IRQ register is updated accordingly.



34.6.4 Remote Terminal (RT) Mode

34.6.4.1 Status Transfer

When configured in RT mode, the 1553 features a specific behavior depending on the command received from the 1553 network. In any case, it sends a status word in return for the command word previously sent.

After 4 µsec, the 1553 status word is automatically sent to the 1553 Bus Controller.

Table 34-2. 1553 Commands List

1553 Command Name	Consequence	1553 Status Word
Dynamic Bus control	IP1553_ISR.DBR=1 (see Note 1)	Dynamic Bus control acceptance OK
	IP1553_ISR.DBR=0 (see Note 2)	Dynamic Bus control acceptance KO
Synchronize without data word	IP1553_ISR.SWD=1	Status=OK
Synchronize with data word	IP1553_ISR.SDR=1	
	IP1553_CTRL1. IP1553DATA1=Data	Status=OK
Transmit status word	NA	Status= response to the last command previously sent (except for transmit status word or transmit last status word)
Initiate self test	IP1553_ISR.STR=1	Status=OK
Transmit built in test	NA	Status=OK+ BIT register content
Transmitter shutdown	Inhibits the Bus A if cmd is received on bus B Inhibits the Bus B if cmd is received on bus A IP1553_ISR.TSR=1	Status = OK
Override transmitter shutdown	Inhibits the Bus A if cmd is received on bus B Inhibits the Bus B if cmd is received on bus A IP1553_ISR.OSR=1	Status = OK

.....continued		
1553 Command Name	**Consequence**	**1553 Status Word**
Inhibit terminal flag bit	Resets the terminal flag bit in the Status frame. IP1553_ISR.ITF=1 IP1553_CR.TC is not relevant.	Status = OK
Override inhibit terminal flag bit	Activates the terminal flag bit in the Status frame. IP1553_ISR.OTF=1 IP1553_CR.TC is relevant	Status = OK
Reset remote terminal	IP1553_ISR.RRT=1	Status = OK
Transmit vector word	IP1553_ISR.TVR=1	Status + IP1553_VWR content
Transmit last vector word	IP1553_ISR.TVR=1	Status + last received command word.
Transmit command (see Note 1)	Once all data are transferred: IP1553_ISR.ETX=1 IP1553_TXBSR.TFn=0	Status + data coming from the 1553 TX buffer n.
	If buffer is not prepared: IP1553_ISR.TE=1 IP1553_TXBAER.TERn=1	Status only
Receive command (see Note 2)	After the reception of all data: IP1553_ISR.EMT=1 IP1553_RXBSR.REn=0	Status OK
	If an error appears: Buffer busy: IP1553_ISR.MTE=1 IP1553_RXBSR.RERn=1 During the reception of data: IP1553_ISR.TE=1	Status KO + send an invalid command

Note:

1. IP1553_CR.BEC shall be equal to 1.
2. IP1553_CR.BEC shall be equal to 0.

In both cases, an interrupt is generated if it has been previously enabled.

34.6.4.2 Broadcast Commands

A command received with a remote terminal address field equal to 0b11111 is considered as a broadcast command. The same operations are realized as described in the section “Status Transfer”. The differences are listed in the table below.

Table 34-3. 1553 Commands List – Broadcast Mode

1553 Command Name	Specificity of Broadcast Command	Status Word	
		Broadcast Bit	Message Error Bit
Dynamic Bus control	Not allowed	0b‘1’	0b‘1’
Transmit status word	Command discarded	–	–
No status send	Same	Same	–

.....continued

1553 Command Name	Specificity of Broadcast Command	Status Word	
		Broadcast Bit	Message Error Bit
Initiate self test	No status sent	0b‘1’	0b‘0’
Transmitter shutdown	No status sent	0b‘1’	0b‘0’
Override transmitter shutdown	No status sent	0b‘1’	0b‘0’
Inhibit terminal flag bit	No status sent	0b‘1’	0b‘0’
Override inhibit terminal flag bit	No status sent	0b‘1’	0b‘0’
Reset remote terminal	No status sent	0b‘0’	0b‘0’
Transmit vector word	Not allowed	0b‘1’	0b‘1’
Synchronize without/with data word	No status sent	0b‘1’	0b‘0’
Transmit last command	Command discarded	–	–
No status sends	Same	Same	–
Transmit BIT word	Not allowed	0b‘1’	0b‘1’
Transmitter shutdown	Command discarded	–	–
No status sends	Same	Same	–
Override transmitter shutdown	Command discarded	–	–
No status sends	Same	Same	–
Transmit command	Not allowed	0b‘1’	0b‘1’
Receive command	No status sent	0b‘1’	0b‘0’

34.6.5 Error Detection

The 1553 in RT or BC mode can detect different errors at different levels (physical and frame). In each case, a specific bit is set and an interrupt is generated when the corresponding interruption is previously enabled.

Table 34-4. Error List

Error Type	BC/RT	Behavior
Manchester code error	All	IP1553_ISR.TE=’1’
		IP1553_ISR.TCE=’1’
Parity error	All	IP1553_ISR.TE=’1’
		IP1553_ISR.TPE=’1’
Type of data error	All	IP1553_ISR.TE=’1’
		IP1553_ISR.TDE=’1’
Time out error	All	IP1553_ISR.TE=’1’
		IP1553_ISR.TTE=’1’
Error in number of word detected (not a time out error)	All	IP1553_ISR.TE=’1’
		IP1553_ISR.TWE=’1’

.....continued		
Error Type	**BC/RT**	**Behavior**
Illegal command	RT	IP1553_ISR.TE='1'
		Status frame has the message bit error set to '1'. (in broadcast mode, no status is sent back)
Invalid command (see Note 1)	RT	IP1553_ISR.TE='1'. Status frame has the message bit error set to '1'.
Illegal request (see Note 2)	BC	IP1553_ISR.TE='1' or IP1553_ISR.ITR='1'

Note:

1. An invalid command is a command with a valid command word but for which one or more data words are not compliant with parity, Manchester encoding, bit number, and sync field, or word number test.
2. A mode command transfer request is illegal when mode command word is not defined or defined as reserved in the 1553 standard.
   A data transfer request is illegal in any of the following cases:
   - IP1553_CMDR1.TR ='0'
   - The terminal address is set to '11111'
   - IP1553_CMDR2.TR='1'

34.6.6 1553 Interface Setup

To configure the 1553 network, the user must select the bus A or B for transfer:

- IP1553_CMDR3.BUS='0' for bus A
- IP1553_CMDR3.BUS='1' for bus B

The interruptions must be enabled to see the status of the various transactions.

34.6.6.1 Configuration in BC Mode

To configure the 1553 interface in BC mode, follow the steps below:

1. Set IP1553_CR.PT='1' to configure the terminal as a Bus Controller.
2. Specify the address of the remote device by configuring IP1553_CR.TA=0b'xxxxx'.
3. Specify that the BC device is a data emitter, a data transmitter or none:
   - IP1553_CMDR3.BCE='1' is a data emitter,
   - IP1553_CMDR3.BCR='1' is a data transmitter,
   - IP1553_CMDR3.BCE='0' and IP1553_CMDR3.BCR='0' for none.
4. Define the command frame to be sent:
   - IP1553_CMDR1 relative to data receiver device,
   - IP1553_CMDR2 relative to data transmitter device,
5. If data are to be sent:
   - Specify the base address of the 1553 TX buffer in the register IP1553_ARW.
   - Place the data to send in the 1553 TX buffer n.
   - Set the 1553 TX buffer n status (IP1553_TXBSR) to data ready to send.
6. If data are to be received:
   - Specify the base address of the 1553 RX buffer in the register IP1553_ARR.
   - Set the 1553 RX buffer n status (IP1553_RXBSR) to free to receive data.
7. Enable the transfer by specifying whether it is a data transfer or a command transfer:
   - IP1553_CMDR3.ER='0' for a data transfer
   - IP1553_CMDR3.ER='1' for a command transfer

34.6.6.2 Configuration in RT Mode

To configure the device in RT mode, follow the steps below:

1. Set IP1553_CR.PT='0' to configure the terminal as a Remote Terminal.
2. Specify the address of the remote device by configuring IP1553_CR.TA=0b'xxxxx'. (A soft reset shall be performed to update the change of BC or RT mode.)
3. If data are to be sent:
   – Specify the base address of the 1553 TX buffer in the register IP1553_ARW.
   – Place the data to send in the 1553 TX buffer n.
   – Set the 1553 TX buffer n status (IP1553_TXBSR) to data ready to send.
4. If data are to be received:
   – Specify the base address of the 1553 RX buffer in the register IP1553_ARR.
   – Set the 1553 RX buffer n status (IP1553_RXBSR) to free to receive data.
   It should then receive the frame from the 1553 network.

34.7 Register Summary

Offset	Name	Bit Pos.								
0x00	IP1553_CR	7:0	SC	TC			TA[4:0]			PT
		15:8					RST	BEC	SRC	BC
		23:16								
		31:24								
0x04 ... 0x0B	Reserved									
0x0C	IP1553_CMDR1	7:0	RTSUBADDRESS[2:0]				DATAWORDCOUNT[4:0]			
		15:8	RTADDRESS[4:0]					T_R	RTSUBADDRESS[4:3]	
		23:16								
		31:24								
0x10	IP1553_CMDR2	7:0	RTSUBADDRESS[2:0]				DATAWORDCOUNT[4:0]			
		15:8	RTADDRESS[4:0]					T_R	RTSUBADDRESS[4:3]	
		23:16								
		31:24								
0x14	IP1553_CMDR3	7:0					ER	BCR	BCE	BUS
		15:8								
		23:16								
		31:24								
0x18	IP1553_BITR	7:0	TOBITWORD[7:0]							
		15:8	TOBITWORD[15:8]							
		23:16								
		31:24								
0x1C	IP1553_VWR	7:0	TOVECTORWORD[7:0]							
		15:8	TOVECTORWORD[15:8]							
		23:16								
		31:24								
0x20	IP1553_IER	7:0	TCE	TE	ETRANS[1:0]		ETX	ERX	MTE	EMT
		15:8	DBR	TVR	ITR	BE	TWE	TTE	TDE	TPE
		23:16	OTF	ITF	RRT	SWD	SDR	OSR	TSR	STR
		31:24								IPB
0x24	IP1553_IDR	7:0	TCE	TE	ETRANS[1:0]		ETX	ERX	MTE	EMT
		15:8	DBR	TVR	ITR	BE	TWE	TTE	TDE	TPE
		23:16	OTF	ITF	RRT	SWD	SDR	OSR	TSR	STR
		31:24								IPB
0x28	IP1553_IMR	7:0	TCE	TE	ETRANS[1:0]		ETX	ERX	MTE	EMT
		15:8	DBR	TVR	ITR	BE	TWE	TTE	TDE	TPE
		23:16	OTF	ITF	RRT	SWD	SDR	OSR	TSR	STR
		31:24								IPB
0x2C	IP1553_ISR	7:0	TCE	TE	ETRANS[1:0]		ETX	ERX	MTE	EMT
		15:8	DBR	TVR	ITR	BE	TWE	TTE	TDE	TPE
		23:16	OTF	ITF	RRT	SWD	SDR	OSR	TSR	STR
		31:24								IPB
0x30	IP1553_CTRL1	7:0	IP1553DATA1[7:0]							
		15:8	IP1553DATA1[15:8]							
		23:16	IP1553DATA2[7:0]							
		31:24	IP1553DATA2[15:8]							
0x34	IP1553_CTRL2	7:0	FROMVECTORWORD[3:0]				FROMCOMMANDREG[3:0]			
		15:8	FROMVECTORWORD[11:4]							
		23:16					FROMVECTORWORD[15:12]			
		31:24								
0x38	IP1553_CTRL3	7:0	FROMSTATUSWORD [7:0]							
		15:8	FROMSTATUSWORD [15:8]							
		23:16	FROMBITWORD [7:0]							
		31:24	FROMBITWORD [15:8]							

.....continued

Offset	Name	Bit Pos.								
0x3C	IP1553_ARW	7:0	REG_ADDR_APB_W[7:0]							
		15:8	REG_ADDR_APB_W[15:8]							
		23:16	REG_ADDR_APB_W[23:16]							
		31:24	REG_ADDR_APB_W[31:24]							
0x40	IP1553_ARR	7:0	REG_ADDR_APB_R[7:0]							
		15:8	REG_ADDR_APB_R[15:8]							
		23:16	REG_ADDR_APB_R[23:16]							
		31:24	REG_ADDR_APB_R[31:24]							
0x44	IP1553_RXBSR	7:0	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0
		15:8	RE15	RE14	RE13	RE12	RE11	RE10	RE9	RE8
		23:16	RE23	RE22	RE21	RE20	RE19	RE18	RE17	RE16
		31:24	RE31	RE30	RE29	RE28	RE27	RE26	RE25	RE24
0x48	IP1553_RXBAER	7:0	RER7	RER6	RER5	RER4	RER3	RER2	RER1	RER0
		15:8	RER15	RER14	RER13	RER12	RER11	RER10	RER9	RER8
		23:16	RER23	RER22	RER21	RER20	RER19	RER18	RER17	RER16
		31:24	RER31	RER30	RER29	RER28	RER27	RER26	RER25	RER24
0x4C	IP1553_TXBSR	7:0	TF7	TF6	TF5	TF4	TF3	TF2	TF1	TF0
		15:8	TF15	TF14	TF13	TF12	TF11	TF10	TF9	TF8
		23:16	TF23	TF22	TF21	TF20	TF19	TF18	TF17	TF16
		31:24	TF31	TF30	TF29	TF28	TF27	TF26	TF25	TF24
0x50	IP1553_TXBAER	7:0	TER7	TER6	TER5	TER4	TER3	TER2	TER1	TER0
		15:8	TER15	TER14	TER13	TER12	TER11	TER10	TER9	TER8
		23:16	TER23	TER22	TER21	TER20	TER19	TER18	TER17	TER16
		31:24	TER31	TER30	TER29	TER28	TER27	TER26	TER25	TER24

34.7.1 Configuration Register

Name: IP1553_CR
Offset: 0x00
Reset: 0x00000401
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					RST	BEC	SRC	BC
Access					R/W	R/W	R/W	R/W
Reset					0	1	0	0

Bit	7	6	5	4	3	2	1	0
	SC	TC	TA[4:0]					PT
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit 11 – RST Soft Reset
RST is used when changing the functioning mode from BC to RT, or from RT to BC, or any time a reset is needed.

Value	Description
1	Resets the 1553.

Bit 10 – BEC BCEnableCmd
When the terminal receives a valid Dynamic Bus Control mode command, it checks the BCEnableCmd and accepts/rejects the control. The acceptance/rejection is indicated in the Dynamic Bus Control bit of the status word which is sent in response to the mode command.

Value	Description
0	The terminal rejects the bus control.
1	The terminal accepts the bus control.

Bit 9 – SRC SREQBitCmd
Indicates the value of the Service Request bit to be returned in status word transfers.

Bit 8 – BC BusyBitCmd
Indicates the value of the busy bit to be returned in status word transfers.

Value	Description
1	Inhibits the transmission of the data words in response to a transmit command and its corresponding interrupt.

Bit 7 – SC SSBitCmd
Indicates the value of the Subsystem Flag bit to be returned in status word transfers.

Bit 6 – TC TRBitCmd
Indicates the value of the T/F bit to be returned in status word transfers.
Note: This bit is used by the terminal to claim that it is the default terminal (value '1'). It should be equal to '0' otherwise.

After reception of a valid ‘Inhibit T/F Bit’ command:

- the TRBitCmd is masked
- the T/F bit is maintained at logic level ‘0’

After reception of a valid ‘Override Inhibit T/F bit’:

- the TRBitCmd is unmasked
- the content of the T/F bit is application dependent and is not defined by the standard

Bits 5:1 – TA[4:0] TermAddressConf
Address of the remote terminal on the 1553 bus.
’00000’ and ‘11111’ are reserved by the protocol and shall never be used

Bit 0 – PT POTermConf
Configuration of the terminal after a hardware reset of the 1553.

Value	Description
0	Configures the terminal as a Remote Terminal.
1	Configures the terminal as a Bus Controller. (default state)

34.7.2 Command Register 1

Name: IP1553_CMDR1
Offset: 0xC
Reset: 0x00000000
Property: Write-only

Command Register 1 is only used in BC mode. It is used with Command Register 2 and Command Register 3 in order to initiate a transfer on the 1553 bus.

These commands are relative to the receiver terminal.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RTADDRESS[4:0]					T_R	RTSUBADDRESS[4:3]	
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RTSUBADDRESS[2:0]			DATAWORDCOUNT[4:0]				
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 15:11 – RTADDRESS[4:0] RT Address
RTADDRESS is the command if it is relevant.
RTADDRESS is the address of the receiver of the data.

Bit 10 – T_R T/R
Data transfer: Field T_R is equal to '0'.
Command mode: Relies on the command itself.

Bits 9:5 – RTSUBADDRESS[4:0] RT SUBADDRESS
When RTSUBADDRESS=0x0 or 0x1F: Data word count is a mode code.
Other cases: RTSUBADDRESS is the sub-address inside the receiver of the data

Bits 4:0 – DATAWORDCOUNT[4:0] Data Word Count
When RTSUBADDRESS equals 0x'0' or 0x'1F': DATAWORDCOUNT is the mode code.
Note: Command word relative to the receiver terminal.

Other cases: DATAWORDCOUNT is the amount of data to be transferred.

34.7.3 Command Register 2

Name: IP1553_CMDR2
Offset: 0x10
Reset: 0x00000000
Property: Write-only

Command Register 2 is only used in BC mode. It is used with Command Register 1 and Command Register 3 in order to initiate a transfer on the 1553 bus Command word.

These commands are relative to the transmitter terminal.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RTADDRESS[4:0]					T_R	RTSUBADDRESS[4:3]	
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RTSUBADDRESS[2:0]			DATAWORDCOUNT[4:0]				
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 15:11 – RTADDRESS[4:0] RT Address
RTADDRESS is the command parameter (bits 15 to 11) if relevant.
RTADDRESS is the address of the sender of the data.

Bit 10 – T_R R/T
Data transfer is RT to RT: Field T_R is equal to ‘1’.
Command mode: Relies on the command itself (bit 10).

Bits 9:5 – RTSUBADDRESS[4:0] RT SUBADDRESS
RTSUBADDRESS is the command parameter (bits 9 to 5) if relevant.
RTSUBADDRESS is the sub-address inside the sender of the data.

Bits 4:0 – DATAWORDCOUNT[4:0] Data Word Count
DATAWORDCOUNT is the amount of data to be transferred, or the command parameter (bits 4 to 0) if relevant.

34.7.4 Command Register 3

Name: IP1553_CMDR3
Offset: 0x14
Reset: 0x00000000
Property: Write-only

Command Register 3 is only used in BC mode. It is used with Command Register 1 and Command Register 2 in order to initiate a transfer on the 1553 bus

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					ER	BCR	BCE	BUS
Access					W	W	W	W
Reset					0	0	0	0

Bit 3 – ER Data Or Command Transfer
It is permitted to have both BCR and BCE set to ‘0’. This corresponds to an RT-to-RT transfer, where the bus controller is neither transmitter nor receiver.
It is illegal to have both BCR and BCE set to ‘1’ since a BC cannot be emitter and receiver at the same time.

Value	Description
0	Initiates a data transfer on the bus.
1	Initiates a mode command transfer on the bus.

Bit 2 – BCR 1553 Receiver

Value	Description
0	Indicates that the product is not the receiver of the data.
1	Indicates that the product is the receiver of the data.

Bit 1 – BCE 1553 Emitter

Value	Description
0	Indicates that the product is not the emitter of the data.
1	Indicates that the product is the emitter of the data.

Bit 0 – BUS Bus Used

Value	Description
0	Indicates that the transfer uses BUS A.
1	Indicates that the transfer uses BUS B.

34.7.5 BIT Register

Name: IP1553_BITR
Offset: 0x18
Reset: 0x00000000
Property: Write-only

This register is only used in RT mode. The BIT register is used to store the built-in self test results. It contains the value to be sent by the terminal in response to a ‘Transmit Built-In Test’ command

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TOBITWORD[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TOBITWORD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – TOBITWORD[15:0] To Bit Word
The register stores the data used in Built-In Test mode.

34.7.6 Vector Word Register

Name: IP1553_VWR
Offset: 0x1C
Reset: 0x00000000
Property: Write-only

This register is only used in RT mode. The Vector Word register stores the value to be sent by the terminal in response to a ‘Transmit Vector Word’ command. Refer to the section “Broadcast Commands”.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TOVECTORWORD[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TOVECTORWORD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – TOVECTORWORD[15:0] To Vector Word
The register stores the data used in Vector Word mode.

34.7.7 IRQ Mask Enable Register

Name: IP1553_IER
Offset: 0x20
Reset: 0x00000000
Property: Write-only

The bits can be set individually by writing a ‘1’ in the corresponding position of the IRQ Mask Enable register. See the section “IRQ Status Register” for more details.

Bit	31	30	29	28	27	26	25	24
								IPB
Access								W
Reset								0

Bit	23	22	21	20	19	18	17	16
	OTF	ITF	RRT	SWD	SDR	OSR	TSR	STR
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DBR	TVR	ITR	BE	TWE	TTE	TDE	TPE
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TCE	TE	ETRANS[1:0]		ETX	ERX	MTE	EMT
Access	W		W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit 24 – IPB IPBusy

Bit 23 – OTF OvInhibitTermFlagReq

Bit 22 – ITF InhibitTermFlagReq

Bit 21 – RRT ResetRTReq

Bit 20 – SWD SyncWithoutDataReq

Bit 19 – SDR SyncWithDataReq

Bit 18 – OSR OvTranShutdownReq

Bit 17 – TSR TranShutdownReq

Bit 16 – STR InitSelfTestReq

Bit 15 – DBR DynamicBusContReq

Bit 14 – TVR TransVecWordReq

Bit 13 – ITR IllegalTransferReq

Bit 12 – BE BufIFErr

Bit 11 – TWE TransWordCounterErr

Bit 10 – TTE TransTimeOutErr

Bit 9 – TDE TransDataTypeErr

Bit 8 – TPE TransParityErr

Bit 7 – TCE TransCodingErr

Bit 6 – TE TransErr

Bits 5:4 – ETRANS[1:0] EndTransfer

Bit 3 – ETX EndTransmission

Bit 2 – ERX End reception

Bit 1 – MTE MemTransferErr

Bit 0 – EMT EndMemTransfer

34.7.8 IRQ Mask Disable Register

Name: IP1553_IDR
Offset: 0x24
Reset: 0x00000000
Property: Write-only

The bits can be reset individually by writing a ‘1’ in the corresponding position of the IRQ Mask Disable Register. See the section “IRQ Status Register” for more details.

Bit	31	30	29	28	27	26	25	24
								IPB
Access								W
Reset								0

Bit	23	22	21	20	19	18	17	16
	OTF	ITF	RRT	SWD	SDR	OSR	TSR	STR
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DBR	TVR	ITR	BE	TWE	TTE	TDE	TPE
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TCE	TE	ETRANS[1:0]		ETX	ERX	MTE	EMT
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit 24 – IPB IPBusy

Bit 23 – OTF OvInhibitTermFlagReq

Bit 22 – ITF InhibitTermFlagReq

Bit 21 – RRT ResetRTReq

Bit 20 – SWD SyncWithoutDataReq

Bit 19 – SDR SyncWithDataReq

Bit 18 – OSR OvTranShutdownReq

Bit 17 – TSR TranShutdownReq

Bit 16 – STR InitSelfTestReq

Bit 15 – DBR DynamicBusContReq

Bit 14 – TVR TransVecWordReq

Bit 13 – ITR IllegalTransferReq

Bit 12 – BE BufIFErr

Bit 11 – TWE TransWordCounterErr

Bit 10 – TTE TransTimeOutErr

Bit 9 – TDE TransDataTypeErr

Bit 8 – TPE TransParityErr

Bit 7 – TCE TransCodingErr

Bit 6 – TE TransErr

Bits 5:4 – ETRANS[1:0] EndTransfer

Bit 3 – ETX EndTransmission

Bit 2 – ERX End reception

Bit 1 – MTE MemTransferErr

Bit 0 – EMT EndMemTransfer

34.7.9 IRQ Mask register

Name: IP1553_IMR
Offset: 0x28
Reset: 0x00000000
Property: Read-only

The IRQ Mask Register content can be read at address 0x0028. Each bit corresponds to an interrupt source. See the section “IRQ Status Register” for more details.

Bit	31	30	29	28	27	26	25	24
								IPB
Access								R
Reset								0

Bit	23	22	21	20	19	18	17	16
	OTF	ITF	RRT	SWD	SDR	OSR	TSR	STR
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DBR	TVR	ITR	BE	TWE	TTE	TDE	TPE
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TCE	TE	ETRANS[1:0]		ETX	ERX	MTE	EMT
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 24 – IPB IPBusy

Bit 23 – OTF OvInhibitTermFlagReq

Bit 22 – ITF InhibitTermFlagReq

Bit 21 – RRT ResetRTReq

Bit 20 – SWD SyncWithoutDataReq

Bit 19 – SDR SyncWithDataReq

Bit 18 – OSR OvTranShutdownReq

Bit 17 – TSR TranShutdownReq

Bit 16 – STR InitSelfTestReq

Bit 15 – DBR DynamicBusContReq

Bit 14 – TVR TransVecWordReq

Bit 13 – ITR IllegalTransferReq

Bit 12 – BE BufIFErr

Bit 11 – TWE TransWordCounterErr

Bit 10 – TTE TransTimeOutErr

Bit 9 – TDE TransDataTypeErr

Bit 8 – TPE TransParityErr

Bit 7 – TCE TransCodingErr

Bit 6 – TE TransErr

Bits 5:4 – ETRANS[1:0] EndTransfer

Bit 3 – ETX EndTransmission

Bit 2 – ERX End reception

Bit 1 – MTE MemTransferErr

Bit 0 – EMT EndMemTransfer

34.7.10 IRQ Status Register

Name: IP1553_ISR
Offset: 0x2C
Reset: 0x00000000
Property: Cleared on read

The IRQ Status register is used to report the end of transfers and errors that are detected on the bus during operation. The use of the different bits of this register are presented in detail in the functional description of the block. All of the interrupts are active high.

Bit	31	30	29	28	27	26	25	24
								IPB
Access								COR
Reset								0

Bit	23	22	21	20	19	18	17	16
	OTF	ITF	RRT	SWD	SDR	OSR	TSR	STR
Access	COR	COR	COR	COR	COR	COR	COR	COR
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DBR	TVR	ITR	BE	TWE	TTE	TDE	TPE
Access	COR	COR	COR	COR	COR	COR	COR	COR
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TCE	TE	ETRANS[1:0]		ETX	ERX	MTE	EMT
Access	COR	COR	COR	COR	COR	COR	COR	COR
Reset	0	0	0	0	0	0	0	0

Bit 24 – IPB IPBusy
This signal is only useful in BC mode.

Value	Description
1	Indicates that the macrocell does not accept a new command.

Bit 23 – OTF OvInhibitTermFlagReq

Value	Description
1	Indicates that a valid ‘Override Inhibit Terminal Flag bit’ has been received.

Bit 22 – ITF InhibitTermFlagReq

Value	Description
1	Indicates that a valid ‘Inhibit Terminal Flag bit’ has been received.

Bit 21 – RRT ResetRTReq

Value	Description
1	Indicates that a valid ‘Reset Remote Terminal’ has been received.

Bit 20 – SWD SyncWithoutDataReq

Value	Description
1	Indicates that a valid ’Synchronize Without Data Word’ has been received.

Bit 19 – SDR SyncWithDataReq

Value	Description
1	Indicates that a valid ‘Synchronize With Data Word’ has been received. The data is then available on the IP1553Data1 bus.

Bit 18 – OSR OvTranShutdownReq

Value	Description
1	Indicates that a valid ‘Override Selected Transmitter Shutdown’ has been received.

Bit 17 – TSR TranShutdownReq

Value	Description
1	Indicates that a valid ‘Selected Transmitter Shutdown’ has been received.

Bit 16 – STR InitSelfTestReq

Value	Description
1	Indicates that a valid ‘Initiate Self-Test’ has been received.

Bit 15 – DBR DynamicBusContReq

Value	Description
1	Indicates that a valid ‘Dynamic Bus Control’ has been received. The level of the BCEnableCmd signal has no effect on this signal.

Bit 14 – TVR TransVecWordReq

Value	Description
1	Indicates that a valid ‘Transmit Vector Word’ has been received.

Bit 13 – ITR IllegalTransferReq

Value	Description
1	Indicates that the transfer which has been commanded via the command IF is not legal and will not be performed.

Bit 12 – BE BufIFErr

Value	Description
1	Indicates that a data word transmission has been stopped because data have not been provided in time on the Buffer interface.

Bit 11 – TWE TransWordCounterErr

Value	Description
1	Indicates that the number of words received does not correspond to the number of words expected.

Bit 10 – TTE TransTimeOutErr

Value	Description
1	Indicates the response time of the addressed terminal is greater than expected or that the response is missing.

Bit 9 – TDE TransDataTypeErr

Value	Description
1	Indicates that a data word has been received when a command word was expected and vice-versa.

Bit 8 – TPE TransParityErr

Value	Description
1	Indicates that a parity error has been detected on a received word.

Bit 7 – TCE TransCodingErr

Value	Description
1	Indicates that a Manchester code error has been detected on a word that has been received.

Bit 6 – TE TransErr

Value	Description
1	Indicates that an error has occurred during processing of the reception, transmission, or transfer.

Bits 5:4 – ETRANS[1:0] EndTransfer
When set to 0b'00', 0b'01' or 0b'10', this signal indicates that the data or mode command transfer has ended:

- 0b'00': No status word has been received.
- 0b'01': Indicates that one status word has been received and is available in IP1553_CTRL1.IP1553DATA1.
- 0b'10': Indicates that two status words have been received and are available in IP1553_CTRL1.IP1553DATA1 and IP1553_CTRL1.IP1553DATA2.
- 0b'11': Not used.

Bit 3 – ETX EndTransmission

Value	Description
1	Indicates that a data transmission has ended.

Bit 2 – ERX End reception

Value	Description
1	Indicates that a data reception is ended.

Bit 1 – MTE MemTransferErr

Value	Description
1	Indicates that a R/W memory transfer error has occurred.

Bit 0 – EMT EndMemTransfer

Value	Description
1	Indicates that a R/W memory transfer has succeeded.

34.7.11 Control Register 1

Name: IP1553_CTRL1
Offset: 0x30
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	IP1553DATA2[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	IP1553DATA2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IP1553DATA1[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IP1553DATA1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:16 – IP1553DATA2[15:0] IP1553 Data Value 2
In BC mode, this bus gives the value of the second status word received during the exchange of data words.

Bits 15:0 – IP1553DATA1[15:0] IP1553 Data Value 1
In RT mode, this bus gives the value of the data word received in a ‘Synchronize with Data Word’ mode command.
In BC mode, this bus gives the value of the first status word received during the exchange of a mode command or of data words.

34.7.12 Control Register 2

Name: IP1553_CTRL2
Offset: 0x34
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					FROMVECTORWORD[15:12]			
Access					R	R	R	R
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FROMVECTORWORD[11:4]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FROMVECTORWORD[3:0]				FROMCOMMANDREG[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 19:4 – FROMVECTORWORD[15:0] From Vector Word Register
Provides the content of the Vector Word register.

Bits 3:0 – FROMCOMMANDREG[3:0] From Command Register 3
Provides the content of the Command register 3.

34.7.13 Control Register 3

Name: IP1553_CTRL3
Offset: 0x38
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	FROMBITWORD [15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	FROMBITWORD [7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FROMSTATUSWORD [15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FROMSTATUSWORD [7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:16 – FROMBITWORD [15:0] Content of BIT Register
Provides the content of the BIT register.

Bits 15:0 – FROMSTATUSWORD [15:0] Content of Status Word Register
Provides the content of the Status Word register. The Status Word register contains the value of the status command if it is transmitted.

34.7.14 Address Register Read Register

Name: IP1553_ARR
Offset: 0x40
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	REG_ADDR_APB_R[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	REG_ADDR_APB_R[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	REG_ADDR_APB_R[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	REG_ADDR_APB_R[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – REG_ADDR_APB_R[31:0] RX Base Address
Stores the memory base address for RX.

34.7.15 Address Register Write Register

Name: IP1553_ARW
Offset: 0x3C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	REG_ADDR_APB_W[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	REG_ADDR_APB_W[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	REG_ADDR_APB_W[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	REG_ADDR_APB_W[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – REG_ADDR_APB_W[31:0] TX Base Address
Stores the memory base address for TX.

34.7.16 Rx Buffer Status Register

Name: IP1553_RXBSR
Offset: 0x44
Reset: 0x00000000
Property: Read/Write

This 32-bit register indicates for each of the 32 Rx Buffers whether they are free to receive data or not.

Bit	31	30	29	28	27	26	25	24
	RE31	RE30	RE29	RE28	RE27	RE26	RE25	RE24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RE23	RE22	RE21	RE20	RE19	RE18	RE17	RE16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RE15	RE14	RE13	RE12	RE11	RE10	RE9	RE8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – REx Receive Buffer x

Value	Description
0	Receive Buffer X is full. Writing a ‘0' is has no effect.
1	Receive Buffer X is empty. Writing a ‘1' to REx sets the bit to '1'.

34.7.17 Rx Buffer Access Error Register

Name: IP1553_RXBAER
Offset: 0x48
Reset: 0x00000000
Property: Read/Write

This 32-bit register indicates for each of the 32 Rx Buffers whether they are free to receive data or not.

Bit	31	30	29	28	27	26	25	24
	RER31	RER30	RER29	RER28	RER27	RER26	RER25	RER24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RER23	RER22	RER21	RER20	RER19	RER18	RER17	RER16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RER15	RER14	RER13	RER12	RER11	RER10	RER9	RER8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RER7	RER6	RER5	RER4	RER3	RER2	RER1	RER0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – RERx Receive Error x

Value	Description
0	No receive error detected for Buffer X. Writing a '0' has no effect.
1	Receive error detected for Buffer X. Writing a '1' to RERx resets the bit to '0'.

34.7.18 Tx Buffer Status Register

Name: IP1553_TXBSR
Offset: 0x4C
Reset: 0x00000000
Property: Read/Write

This 32-bit register indicates for each of the 32 Tx Buffers whether they are free to receive data or not.

Bit	31	30	29	28	27	26	25	24
	TF31	TF30	TF29	TF28	TF27	TF26	TF25	TF24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TF23	TF22	TF21	TF20	TF19	TF18	TF17	TF16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TF15	TF14	TF13	TF12	TF11	TF10	TF9	TF8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TF7	TF6	TF5	TF4	TF3	TF2	TF1	TF0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – TFx Transmit Buffer x

Value	Description
0	Transmit Buffer X is empty. Writing a ‘0' has no effect.
1	Transmit Buffer X is ready to be sent. Writing a ‘1' to TFx sets the bit to '1'.

34.7.19 Tx Buffer Access Error Register

Name: IP1553_TXBAER
Offset: 0x50
Reset: 0x00000000
Property: Read/Write

This 32-bit register indicates for each of the 32 Tx Buffers whether they are free to receive data or not.

Bit	31	30	29	28	27	26	25	24
	TER31	TER30	TER29	TER28	TER27	TER26	TER25	TER24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TER23	TER22	TER21	TER20	TER19	TER18	TER17	TER16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TER15	TER14	TER13	TER12	TER11	TER10	TER9	TER8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TER7	TER6	TER5	TER4	TER3	TER2	TER1	TER0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – TERx Transmit Error x

Value	Description
0	No receive error detected for Buffer X. Writing a ‘0' has no effect.
1	Receive error detected for Buffer X. Writing a ‘1' to TERx resets the bit to '0'.

35. SpaceWire (SpW)

35.1 Description

The SpaceWire (SpW) module is used in aerospace applications as a means of communication. The SpW module performs the interface between the SpaceWire network and AHB transactions. An embedded router is included in the SpW module to perform the mapping between the two SpaceWire links connected to the SpaceWire network and one transmitter packet (Tx), one receiver packet (Rx) and one Remote Memory Acces Protocol (RMAP) block connected to the internal AHB network.

The SpW module provides support for transmission of any type of protocol or data structure using SpaceWire packets. It has hardware support for logical addressing to different receivers, for execution of incoming RMAP commands, and for reception/generation of Escape Characters (Broadcast Codes) and Time Codes.

35.2 Embedded Characteristics

- Compliant with ECSS standards ECSS-E-ST-50, 51, 52, 53C
- SpaceWire Links: Digital Input/Ouput and LVDS Supported
- Two SpaceWire Interfaces Up to 200 Mbits/s Each
- One Received Packet (Rx), one Transmit Packet (Tx) and one RMAP Block Connected to the AHB Interface
- One SpaceWire Router to Transmit/Receive the SpaceWire Packets from Rx, Tx and RMAP Blocks to SpaceWire Link
- Fallback Scheme Implemented
- Support for RMAP Protocol (ECSS-E-ST-52)
- Support for CCSDS Packet Transfer Protocol (ECSS-E-ST-53)
- SpaceWire D Capabilities
- Support for Reception/Generation of Escape Characters (Broadcast Codes) and Time Codes

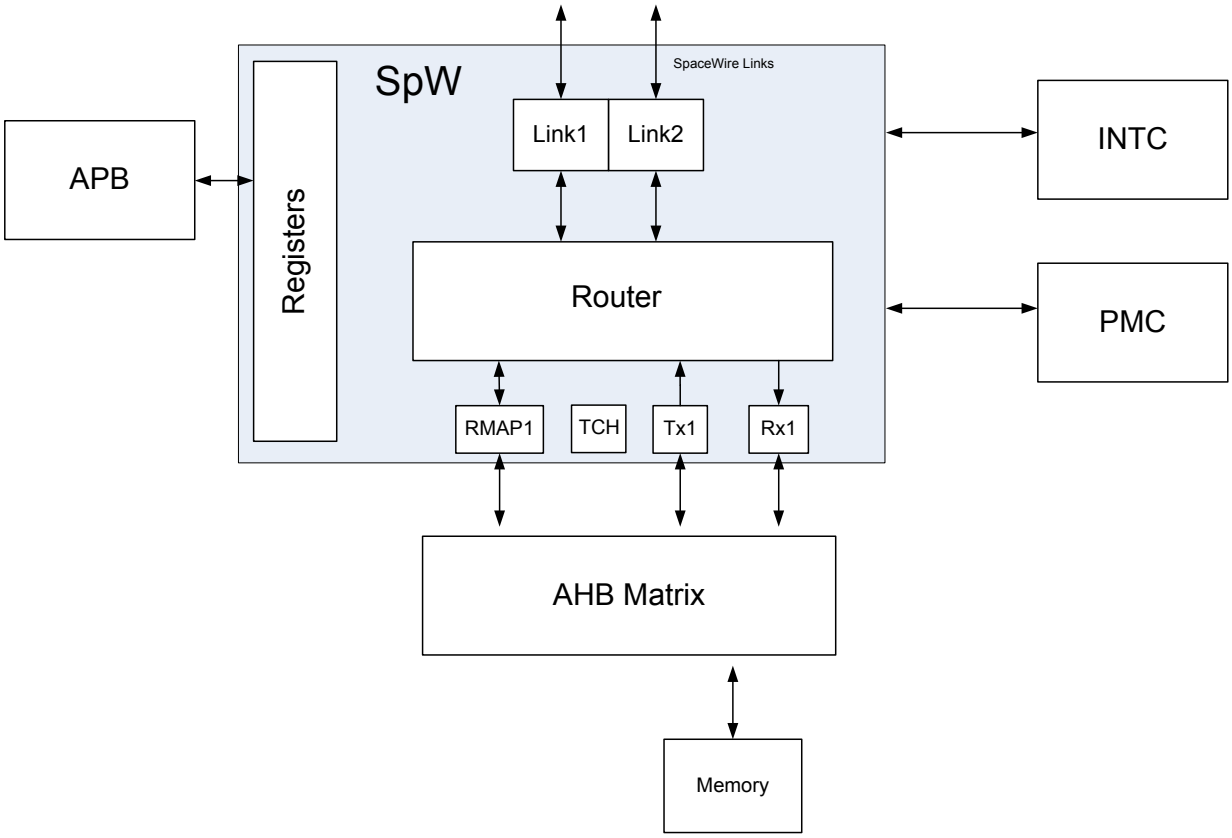
35.3 Reference Documents

The SpaceWire (SpW) standard is described in the following documents:

- ECSS-E-ST-50-12C describes the links, nodes, routers and networks
- ECSS-E-ST-50-51C describes the SpW protocol identification
- ECSS-E-ST-50-52C describes the RMAP protocol
- ECSS-E-ST-50-53C describes the CCSDS protocol

35.4 Block Diagram

Figure 35-1. Block Diagram



35.5 Product Dependencies

35.5.1 Power Management

The SpW is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SpW clock and the timetick clock if required.

On transmission, the SpaceWire bitrate is defined from the GCLK clock frequency (see the Speed Link section for more details). The GLCK frequency must be less than or equal to twice the MCK clock frequency; that is, GCLK ≤ 2*MCK

On reception, the SpaceWire bitrate is linked to the MCK clock frequency. The bit rate must be less than or equal to four times MCK; that is, RX_SpaceWire_bitrate ≤ 4*MCK.

35.5.2 Interrupt Sources

The SpW interrupt lines are connected to the interrupt controller. Using the SpW interrupts requires the interrupt controller to be programmed first. The SpW interrupts are generated only if the corresponding bit has been enabled in the SpW Link x Interrupt Mask (SPW_LINKx_IM) register.

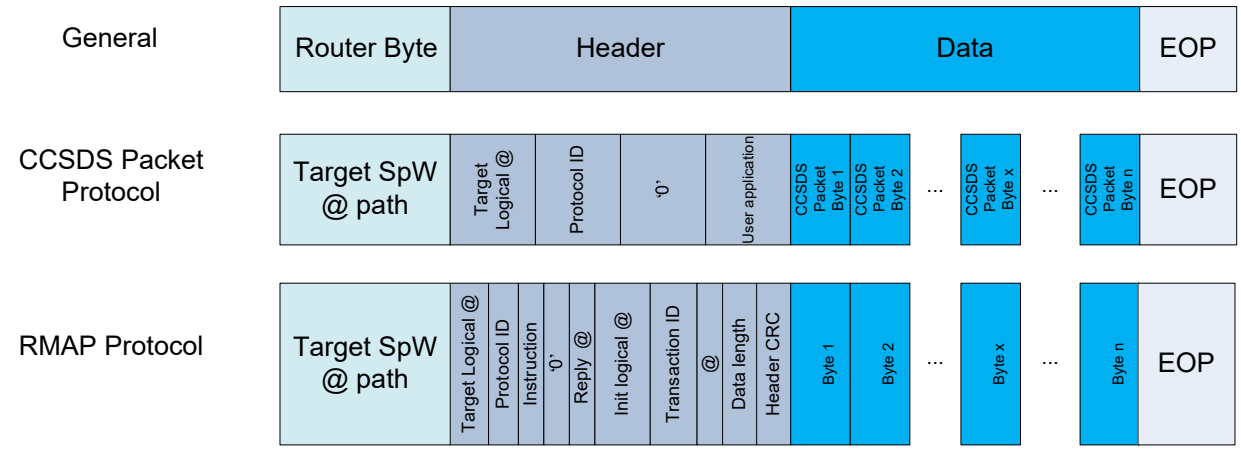
35.6 SpaceWire Protocol

The reference documents listed above detail the different protocols supported over SpaceWire links.

The figure below presents the SpaceWire protocol, split into four categories:

- Router byte
- Header
- Data
- End Of Packet (EOP)

Figure 35-2. SpaceWire Packet: Protocol Format Supported



Each part has the following function:

- Router bytes describe the path of a space wire packet
- Headers describes the protocol to used, the logical address targeted, etc.
- Data contains effective data to transmit/receive over SpaceWire
- EOP indicates the End Of Packet

35.7 Functional Description

The SpW module comprises:

- Two SpaceWire links
- A transmitter block
- A receiver block
- A router block to redirect SpaceWire packets from a source (SpaceWire link, RMAP, transmitter) to a destination (SpaceWire link, RMAP, receiver)
- An RMAP block to support the SpaceWire RMAP protocol
- A TCH block to support SpaceWire time-coded frames

35.7.1 SpaceWire Link

35.7.1.1 Speed Link

To be compliant with the SpaceWire standard, the initial bit rate of a SpaceWire link is 10 Mbits/s. Hence, the SpaceWire peripheral clock (SpWCLK), coming from the PMC, must be internally adapted to respect this constraint, by adjusting the ratio in SPW_LINKx_CLKDIV.TXINITDIV.

As soon as the SpaceWire connection link is established, the SpaceWire speed link can be increased up to 200 Mbits/s. To do so, the SpaceWire peripheral clock (SpWCLK) must be adjusted by adapting the ratio in SPW_LINKx_CLKDIV.TXOPERDIV.

The SpaceWire bit rates (initial and operational) are calculated from the following formulas:

SpWInitbitrate= (SpWCLK) / (0.5 (TXINITDIV +1))

SpWOperbitrate= (SpWCLK) / (0.5 (TXOPERDIV +1))

35.7.1.2 SpaceWire Link Configuration

After reset, the SpaceWire link is configured in Autostart mode. The SpaceWire link is configured using the register SPW_LINKx_CFG.

- If link disable is set, the link goes from run state to errorstate state.
- If autostart is set, the link remains in ready state until NULL are received.
- If link start is set, the link proceeds to started state.

Refer to the figure *State diagram for SpaceWire link interface*, available in ECSS-E-ST-50-12C (31 July 2008), for more details.

35.7.1.3 Link Status

The status of the SpaceWire CODEC link state is visible via SPWLINK_STATUS register.

35.7.1.4 Link Error Interrupts

The SpaceWire link block has three sets of interrupt registers:

- SpaceWire Link Pending Interrupt registers (SPW_LINKx_PI_yy) contain interrupts for the SpaceWire CODEC errors, configurable escape character, EEP transmitted/received and if a packet was aborted before fully transmitted.
- SpaceWire Distributed Interrupt Pending Interrupt registers (SPW_LINKx_DISTINTIPI_yy) contain interrupts for SpaceWire distributed interrupts. An incoming escape character with the three MSB set to “100” is interpreted as a distributed interrupt. The remaining five bits represent the interrupt number.
- SpaceWire Distributed Interrupt Acknowledge Pending Interrupt registers (SPW_LINKx_DISACKPI_xx) contain interrupts for SpaceWire distributed interrupt acknowledge. An incoming escape character with the three MSB set to “101” is interpreted as a distributed interrupt acknowledge. The remaining five bits represent the interrupt acknowledge number.

For details on set-up and use, refer to the register tables.

35.7.2 Embedded SpaceWire Router

The embedded SpaceWire router:

- directs the SpaceWire packet from source to destination
- maps the logical address to physical one, in the case of SpaceWire packets using logical address

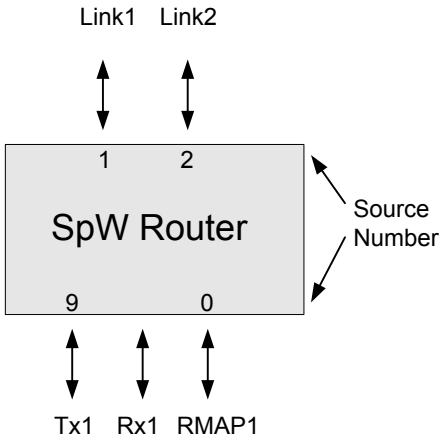
35.7.2.1 Sources

The different sources available, and their associated numbers, are listed below.

Table 35-1. Source Available and Associated Number

Address	Type of Address
0	RMAP 1
1	SpW Link 1
2	SpW Link 2
3–8	Reserved
9	Pkt Transmitter 1
10-17	Reserved

Figure 35-3. Source and Physical Address



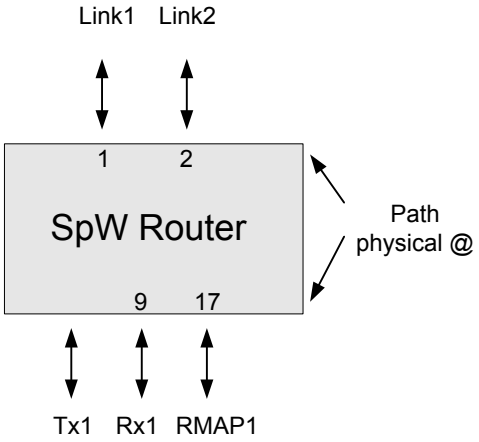
35.7.2.2 Destinations

The different destinations available, and their associated physical addresses, are listed below.

Table 35-2. Destination Number and Associated Physical Addresses

Path/Physical Address	Destination
0	Unused
1	SpW Link 1
2	SpW Link 2
3–8	Reserved
9	Pkt Receiver 1
10-16	Reserved
17	RMAP 1
18-31	Unused

Figure 35-4. Destination and Physical Address



35.7.2.3 Logical to Physical Addressing Mapping

Logical and physical address ranges are defined below.

Table 35-3. Address Repartition

Address	Type of Address
0	Disable logical address
1-30	Physical address or path (only 1-2, 9,17 used in this case)
31	Loopback for SpW link 1-2
32-255	Logical address

SpaceWire packets can contain logical addresses instead of physical paths. To correctly direct such a SpaceWire packet, a logical to physical mapping must be defined first.

This is done by configuring SPW_ROUTER_TABLE[logicaladdnb].

- SPW_ROUTER_TABLE[logicaladdnb].ADDR maps the logical address “logicaladdnb” to the physical one.
- SPW_ROUTER_TABLE[logicaladdnb].DELHEAD defines if router bytes are kept or deleted when the SpaceWire packet is passed on.

Then, the logical address must be authorized. To do so, SPW_ROUTER_CFG.LAENA must be set.

Note: logicaladdnb is in the range from 32 to 255.

35.7.2.4 Logical to Physical Addressing Mapping

SpaceWire packets may contain a logical address instead of a physical path. To correctly direct such a SpaceWire packet, a logical to physical mapping must be defined.

This is done by configuring SPW_ROUTER_TABLE[logicaladdnb].

- SPW_ROUTER_TABLE[logicaladdnb].ADDR maps the logical address “logicaladdnb” to the physical one.
- SPW_ROUTER_TABLE[logicaladdnb].DELHEAD defines if router bytes are kept or deleted when the SpaceWire packet is passed on.

Note: logicaladdnb is in the range from 32 to 255.

Then, the logical address must be authorized. To do so, SPW_ROUTER_CFG.LAENA must be set.

Note: It is recommended to write the table with all zeroes before configuring the router in order to obtain a correct initialization.

35.7.2.5 Fallback Routing

If SPW_ROUTER CFG.FALLBACK is set, an alternative destination can be picked if the addressed destination is unavailable. If the physical port N is unavailable (SpW link not running), the fallback is selected as follows:

- for an odd port, select N+1
- for an even port, select N-1

If the fallback port is unavailable, it is treated and reported as an illegal address.

Fallback is only applicable when using logical addressing. Path addressing is unaffected by fallback.

35.7.2.6 Timeout

If a timeout event is detected, the SpaceWire router destination is automatically flushed. The physical address is visible in the SPW_ROUTER_TIMEOUT register. The register is locked until it has been read by the user.

The timeout is enabled by default. To disable it, SPW_ROUTER_CFG.DISTIMEOUT must be set.

35.7.2.7 Error Router Status

If a routed packet has an illegal destination, the packet is discarded.

Error routing information is visible in the dedicated register SPW_ROUTER_STS.

When errors occur, it is possible to visualize:

- the number of packets dropped in SPW_ROUTER_STS.COUNT
- the source of the first dropped packet in SPW_ROUTER_STS.SOURCE

- the router byte of the first packet dropped in SPW_ROUTER_STS.BYTE
- the destination of the first packet dropped in SPW_ROUTER_STS.DEST

35.7.3 Packet Transmit

35.7.3.1 Overview

SpaceWire packets are transmitted by the packet transmit block to the SpaceWire network, passing through the router block.

35.7.3.2 Transmit (TX) Packet

Send list are used to transfer data SpaceWire packets over the SpaceWire network.

A send list is made of several send list entries.

The send list is made of a status part, a routing part, a header part and a data part.

32-bit Word	Bit Number																															
	31	30	29		27			24	23					18		16	15							8	7							0
0	Skip	Entry		-	RSize				-					Start																		
1	RB1								RB2								RB3								RB4							
2	RB5								RB6								RB7								RB8							
3	-				EscMask				EscChar								-					HCrc			HSize							
4	HAddr																															
5	-							DCRC	DSize																							
6	DAddr																															
7	-																								Timeout							

Field	Definition	Size	Value	Description
Skip	Skip	1	0	Use this entry.
			1	Skip this entry. All other fields are ignored.
Entry	Entry Type	2	0	Packet Data
			1	Escape Character
			others	Invalid
RSize	Router Byte Size	4	0-8	Number of Router Bytes to prepend to packet.
			others	Invalid
Start	Start Time	19	Any	Delay from send list start. Time Unit is environment dependent.
RB1	Router Byte 1	8	Any	Byte 1 to prepend, if used.
RB2	Router Byte 2	8	Any	Byte 2 to prepend, if used.
RB3	Router Byte 3	8	Any	Byte 3 to prepend, if used.
RB4	Router Byte 4	8	Any	Byte 4 to prepend, if used.
RB5	Router Byte 5	8	Any	Byte 5 to prepend, if used.
RB6	Router Byte 6	8	Any	Byte 6 to prepend, if used.
RB7	Router Byte 7	8	Any	Byte 7 to prepend, if used.
RB8	Router Byte 8	8	Any	Byte 8 to prepend, if used.

.....continued

Field	Definition	Size	Value	Description
EscMask	Escape Char Mask	4	Any	Defines where to send Escape Character.
EscChar	Escape Char	8	Any	Escape Character to send if Entry Type is Escape Char.
HCrc	Header Crc	1	0	Send as is.
			1	Calculate CRC and attach at end.
HSize	Header Size	8	Any	Header Size, in bytes
HAddr	Header Addr	32	Any	Header Start Address
DCrc	Data Crc	1	0	Send as is
			1	Calculate CRC and attach at end
DSize	Data Size	24	Any	Data Size, in bytes
DAddr	Data Addr	32	Any	Data Start Address
Timeout	Timeout	19	0	No Timeout
			others	Abort send list if time from send list start exceeds this value.

Each send list entry can be used or skipped, depending on its SKIP bit status.

- If the SKIP bitfield is ‘0’: the send list entry is sent.
- If the SKIP bitfield is ‘1’: the send list entry is skipped.

Each Packet Transmitter is able to transmit escape characters on up to four links. The table below shows how the EscMask bits in a Send List Entry map to links for each Packet Transmitter.

PktTx	EscMask(3)	EscMask(2)	EscMask(1)	EscMask(0)
Tx1	Link 1	Link 2	Not used	Not used

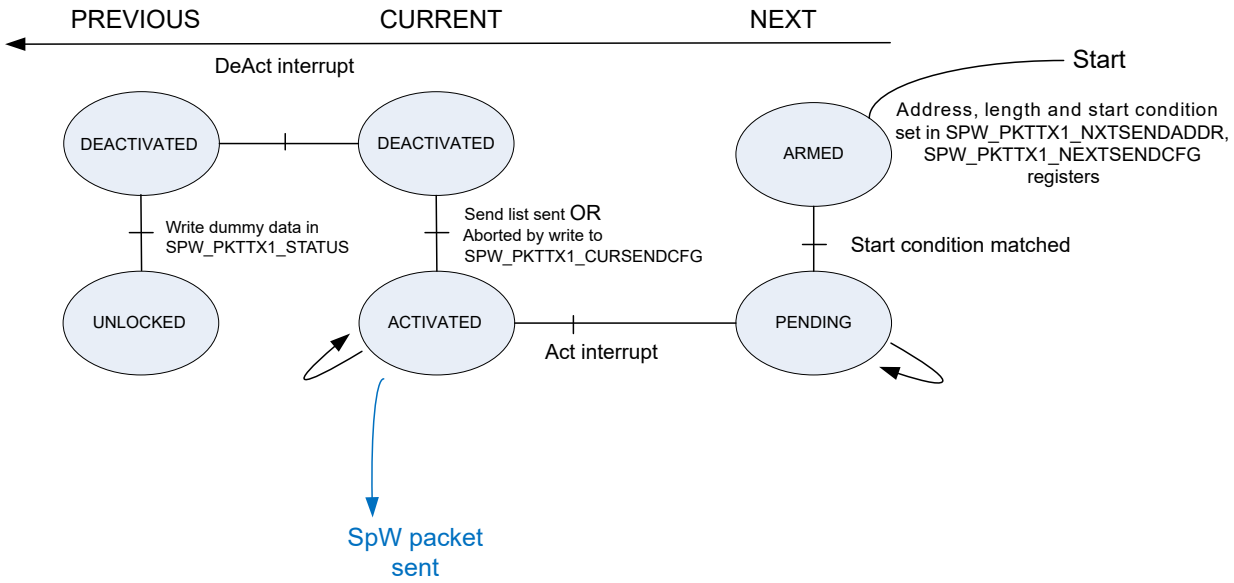
35.7.3.3 Send List State

Send lists go through three statuses, as listed below:

- Next status: The transmit send lists are set up.
- Current status: The send list is sent over the SpaceWire network.
- Previous status: All the data has been transferred to the network and the status of the send list can be read.

The flowchart below illustrates the different states of the send list.

Figure 35-5. Send List State



35.7.3.3.1 Next Status

In Next status, two states are possible:

- ARMED: The registers are initialized with send list information (starting address, length and starting condition). Refer to the section Send List Initialization.
- PENDING: Once the start condition has been detected, the send list proceeds to PENDING state. It then goes to ACTIVATED state.

35.7.3.3.2 Current Status

In Current status, two states are possible:

- ACTIVATED: The data transfer to the router block is pending.
  It will stay in this state unless:
  - The send list has been completely transferred.
  - The user requests that the current transmission (SPW_PKTTX_CURSENDCFG) be aborted.
- DEACTIVATED: The data transfer to the router block is finished.

35.7.3.3.3 Previous Status

In Previous status, one state is possible:

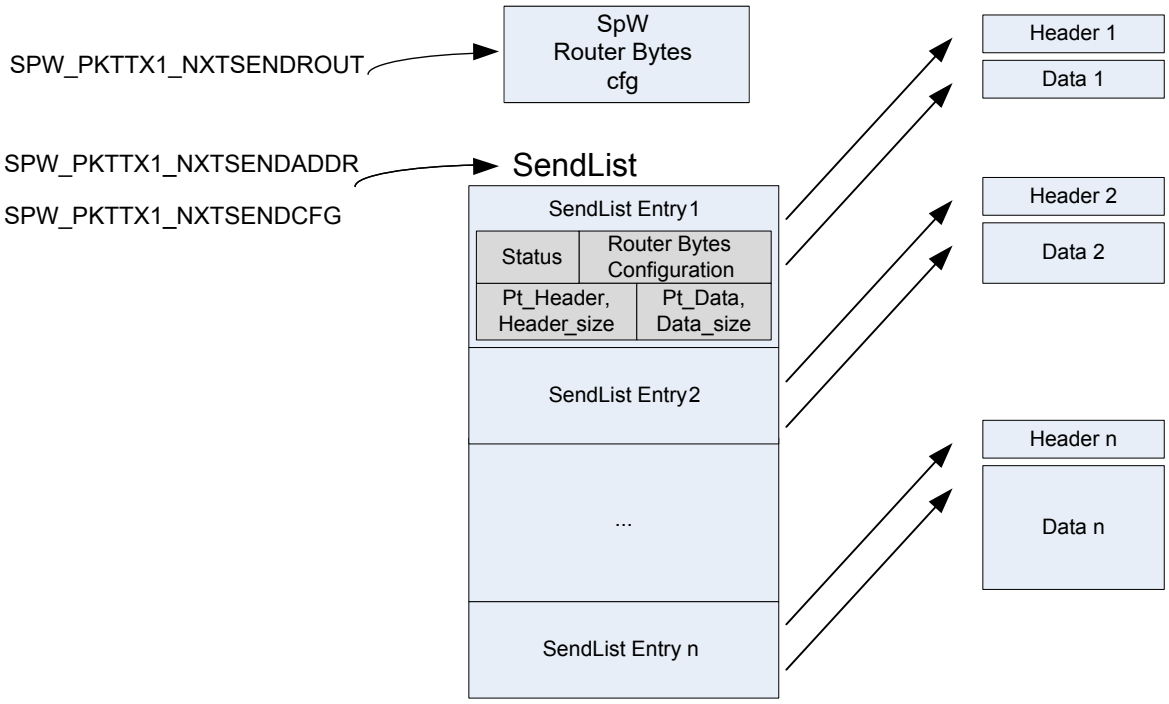
- DEACTIVATED: Until SPW_PKTTX1_STATUS is written, the status is locked and any following send lists cannot deactivate. SPW_PKTTX1_STATUS must be written with any dummy data to unlock the status after deactivation.

35.7.3.4 Send List Initialization

Send list initialization is performed during the ARMED state.

The figure below presents the global view of a send list and its content.

Figure 35-6. Send List Overview



To initialize a send list:

1. The start address of the send list must be indicated in the SPW_PKTTX1_NXTSENDADDR register.
2. The number of send list entries contained in the sendlist must be indicated in the SPW_PKTTX1_NXTSENDCFG register.
3. The destination path must be indicated either in the SPW_PKTTX1_NXTSENDROUT register or in the send list entry (R1 –R8). If the router byte is indicated:In the SPW_PKTTX1_NXTSENDROUT register, the router bytes are the same for all send list entries.Directly in the send list entry (R1 –R8), the router bytes are specific to the send list entry.
4. The number of send list entries contained in the send list must be indicated in the SPW_PKTTX1_NXTSENDCFG register.
5. The starting condition to transmit the send list must be written in the SPW_PKTTX1_NXTSENDCFG register.

Each send list entry must be filled as follows:

- **Status and configuration**
  - Skip bit: If one, the send list entry is ignored.
  - Entry: Indicates if dat or specific character are sent.
  - Start: Delay before sending the send list; this is defined by the START bitfield of the SPW_PKTTX1_NXTSENDCFG register.
  - EscChar and EscMask : Specific if escape characters are sent. EscChar defines which escape character to send, and the EscMask: defines which interface should receive it.
  - Timeout: Timeout to abort the send list transfer after a certain period.
- **Router bytes configuration**

– Rsize and RB1-8: Number of router bytes used in the send list and the router bytes (i.e., path of the SpaceWire packet to send).

Note: Up to four router bytes can be specified in the SPW_PKTTX1_NXTSENDROUT register. These bytes are the start of each packet in the send list.

Note: Up to eight router bytes can be specified directly in the Send List Entry. These bytes are the next part of the packet.

- **Header**
  – HCRC: Add or not a CRC at the end of the header part for reliability
  – Hsize: Size in bytes of the header part
  – Haddr: Starting address of the header part, stored in memory
- **Data**
  – DCRC: Add or not a CRC at the end of the data part for reliability
  – Dsize: Size in bytes of the data part
  – Ddaddr: Starting address of the data part, stored in memory

The header and data parts are set up with an address and a length defining a buffer in memory. These parts also may add the CRC byte after each block. Note that an added CRC is not included in Hsize or Dsize. The CRC is to support RMAP.

35.7.3.5 Current Send List Parameters

The router bytes of the current send list are visible in the SPW_PKTTX1_CURSENDROUT register.

The staring address of the current send list is visible in the SPW_PKTTX1_CURSENDADDR register

The number of the current send list entry is visible in the SPW_PKTTX1_CURSENDCFG register.

It is possible to abort any on-going transfer by setting the bit SPW_PKTTX1_CURSENDCFG. ABORT.

35.7.3.5.1 Send List Status

The status of the current send list is visible in the SPW_PKTTX1_STATUS register: deactivated, pending active or armed.

The origin of the previous send list deactivated is also visible in this register.

35.7.3.5.2 Abort a Send List

There are two ways to abort a send list transfer:

- To stop the transfer immediately: Set the bit SPW_PKTTX1_CURSENDCFG.ABORT.
- To stop the transfer at the next packet boundary: Write all zeroes in the field SPW_PKTTX1_CURSENDCFG.LEN.

35.7.3.6 Transmitter Interruptions

If previously enabled (SPW_PKTTX1_IM register), interruptions can be launched when a send list is activated or deactivated, and when EEP or EOP are sent. All the interruptions are pending on the SPW_PKTTX1_PI register.

35.7.3.7 Reset Transmitter

The packet transmitter can be reset by software by writing the correct pattern in SPW_PKTTX1_SWRESET.

35.7.4 Packet Received

35.7.4.1 Overview

Each receiver can receive SpaceWire packets if the routing mechanism is set up to reach them, i.e., if the logical address or logical path is correctly set.

Packets received are stored in Receive Buffers, and can be divided over buffers if necessary.

35.7.4.2 Received (RX) Packets

The packets received are divided into two parts.

- The first part is for packet data where incoming packet data is stored as received.

- The second part is for packet information such as length, receive time and status. For details, refer to the tables below.

32-bit Word	Bit number																													
	31							23				18		16												3	2	1	0
0	-							Crc						-												Split	Cont	Eep	Eop
1	DAddr																													
2	-							DSize																						
3	-											Etime																		

Field	Definition	Size	Value	Description
Crc	Crc	8	Any	8 bit CRC calculated over all packet bytes
Split	Split	1	Any	Packet is split, and continues in next entry/buffer
Cont	Continued	1	Any	Packet is continued from previous
EEP	Eep	1	Any	Error End of Packet
EOP	Eop	1	Any	End of Packet
DAddr	Data Addr	32	Any	Start Address of received packet. Will always be word aligned.
DSize	Data Size	24	Any	Length of received packet, in bytes
Etime	End Time	19	Any	Time at end of packet, from receive buffer start. Time Unit is environment dependent.

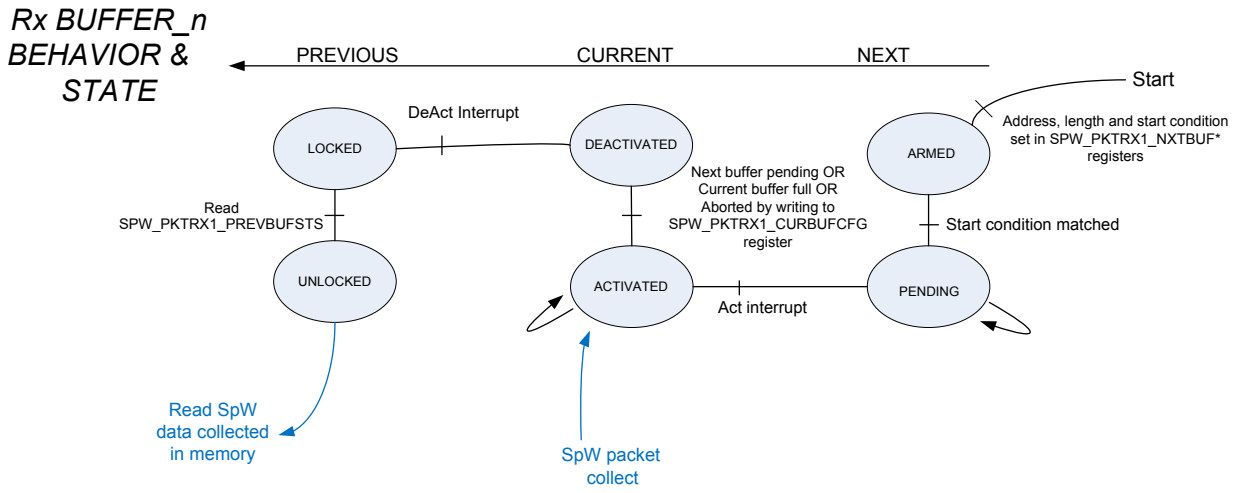
35.7.4.3 Received Buffers State

Received buffers three statuses, as listed below:

- Next status: The Rx buffers are set up.
- Current status: The Rx buffers collect SpaceWire packets.
- Previous status: The data collected and stored in memory can be read.

The flowchart below presents the Rx buffer states.

Figure 35-7. Received (Rx) Buffers State



35.7.4.3.1 Next Buffer

In Next Buffer state, two modes are possible:

- ARMED: The registers are initialized with buffer information (starting address, length and starting condition). Refer to the section “Received Buffer Set Up”.
- PENDING: Once the initialization is done and the start condition is detected, the buffer goes into the PENDING state. Then its behavior changes to Current Buffer state and its mode is ACTIVATED.

35.7.4.3.2 Current Buffer

In Current Buffer state, two modes are possible:

- ACTIVATED: All the information is passing through the next register to the current one. In this mode, the Rx buffer collect space packets received from the SpaceWire network. It will stay in this mode unless:
  - the current buffer (information) is full and the previous one is not locked
  - the current buffer (data) is full and the previous one is not locked. In this case, the on-going packet is split.
  - a split request is received and the previous one is not locked
  - the starting condition of the next buffer is matched and the previous one is not locked
  - the user requests an abort of the current reception (SPW_PKTRX1_CURBUFCFG registers).

  Note: When a packet is split, the first one is marked as ‘split’ and the second one is marked as ‘continue’.
- DEACTIVATED: The Rx buffers do not collect data. They pass directly to LOCKED mode, associated with the behavior PREVIOUS.

Depending on the SPW_PKTRX1_CFG register, the incoming data could be stalled or discarded.

35.7.4.3.3 Previous Buffer

In Previous Buffer state, two modes are possible:

- LOCKED: All the information is transferred to the previous buffer registers. It will remain in this mode until SPW_PKTRX1_PREVBUFSTS is read.
- UNLOCKED: In this mode, the data collected can be read from memory. To do so, the SPW_PKTRX1_PREVBUFSTS.CNT indicates how many received packet info entries have been written in memory.

Note: While the Previous Buffer is locked, a Current Buffer cannot be deactivated.

35.7.4.3.4 Status Buffer

The status of a receiver is read in the SPW_PKTRX1_STATUS (and resp. SPW_PKTRX1_PREVBUFSTS) registers and the number of packets received in the active buffer (resp. previous) as well.

35.7.4.4 Packet Information Structure

Table 35-4. Packet Information Definition

32-bit Word	Bit number																													
	31								23				18		16												3	2	1	0
0	-								Crc							-											Split	Cont	Eep	Eop
1	DAddr																													
2	-								DSize																					
3	-																				Etime									

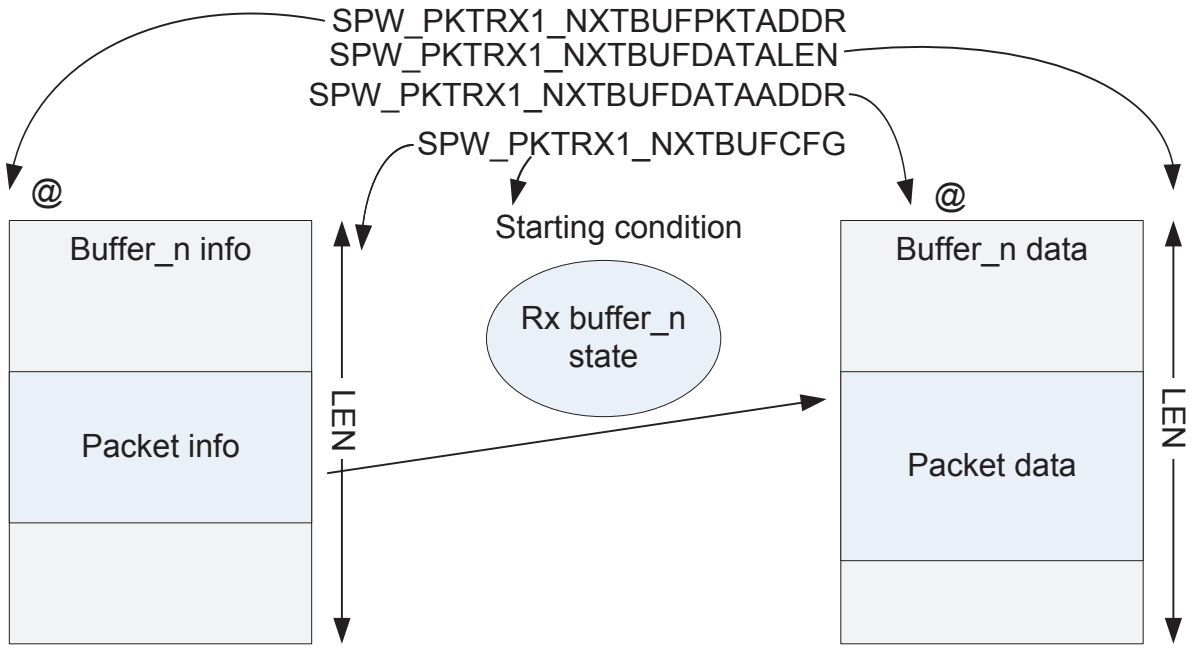
Field	Definition	Size	Value	Description
Crc	Crc	8	Any	8-bit CRC calculated over all packet bytes
Split	Split	1	Any	Packet is split, and continues in next entry/buffer
Cont	Continued	1	Any	Packet is continued from previous
Eep	Eep	1	Any	Error End of Packet
Eop	Eop	1	Any	End of Packet
DAddr	Data Addr	32	Any	Start Address of received packet. Will always be word aligned.

.....continued				
Field	**Definition**	**Size**	**Value**	**Description**
DSize	Data Size	24	Any	Length of received packet, in bytes
Etime	End Time	19	Any	Time at end of packet, from receive buffer start. Time Unit is environment dependent.

35.7.4.5 Received Buffers Setup

The figure below provides an overview of the received buffers.

Figure 35-8. Received (Rx) Buffers Overview



To correctly initialize the Rx buffers, the following steps must be performed:

- The starting address of the buffer which contains the received data is written in the SPW_PKTRX1_NXTBUFDATAADDR register. Starting address must be word-aligned.
- The length of the buffer which contains the received data is written in the SPW_PKTRX1_NXTBUFDATALEN register. When setting the data buffer length, each expected packet size should be rounded up to word size.
- The starting address of the buffer which contains the received packet information is written in the SPW_PKTRX1_NXTBUFPKTADDR register. Starting address must be word-aligned.
- The number of received packet information is written in the SPW_PKTRX1_NXTBUFCFG register.
- The starting condition of the buffer is written in the SPW_PKTRX1_NXTBUFCFG register.
- The split condition activation of the buffer is written in the SPW_PKTRX1_NXTBUFCFG register.

Note: All RX buffers must be word-aligned.

The table below indicates the possible conditions to start the Rx buffers, that is, become next to current buffers.

Table 35-5. Possible Starting Modes

Start	Mode	Description
0	Event	Start if any bit in Value matches an incoming event
1	Immediately	Start immediately
2	Time Code 1	Start if current time code in Time Code Handler 1 is equal to Value
3	Time Code 2	Not used

.....continued		
Start	**Mode**	**Description**
4	Later	Start if Current Buffer is deactivated, e.g. by becoming full

When the next buffer is activated (becomes the current buffer), all starting conditions in the next buffer are cleared).

35.7.4.6 Packet CRC

For each packet received, a CRC is calculated over the incoming bytes and stored in the Receive Packet Info structure for the packet. The CRC uses the same algorithm as RMAP. When calculating CRC over a complete RMAP packet (header, header CRC, data and data CRC), the result must be '0' if no errors have been detected.

If a packet is split, the first entry will show the CRC up to that point. The CRC in the continued entry will be over the entire packet.

35.7.4.7 Receiver Interruptions

Interruptions can be pending on SPW_PKTRX1_PI registers if they are previously enabled (SPW_PKTRX1_IM registers):

- to have an overview of the receive buffer state (ACTIVATED or DEACTIVATED)
- to detect the EEP or EOP character
- to detect if an incoming packet is discarded

35.7.4.8 Reset Receiver

The packet receiver can be reset by software by writing the correct pattern in SPW_PKTRX1_SWRESET.

35.7.5 RMAP

35.7.5.1 Overview

The RMAP Handler services incoming RMAP commands in hardware.

SpaceWire packets addressed to RMAP are interpreted according to the packet's Protocol ID.

RMAP packets are handled according to the RMAP standard. The table below lists the supported RMAP features.

Table 35-6. Supported RMAP Features

Feature	Command
Command Codes supported	3, 9, 11
Target Logical Address	Must be 0xFE or RMAP's configured LA
Key field value	Defined by SPW_RMAPx_CONFIG.DESTKEY
Single Address commands	Not supported, only sequential is supported. (Increment bit must be 1)
Read and Write commands	Supported
Read-Modify-Write command	Not supported
Write with reply	Supported
Verified write	Not supported (Verify data bit must be 0)
Reply address	Supported up to 12 bytes.

If an RMAP Handler is accessed from a SpW Link in the same group, it will automatically add a path router byte to route the reply back to the same link.

35.7.5.2 RMAP Setup

To correctly activate the RMAP protocol and accept the RMAP frames, the user must:

- enable the RMAP access by setting SPW_RMAP_CFG.RMAPENA.
- configure the target logical address in SPW_RMAP_CFG.TLA to accept the RMAP frame with this logical address.

- configure the destination key in SPW_RMAP_CFG.DESTKEY to accept the RMAP command which contains this value.

35.7.5.3 RMAP Status

The error information from RMAP is visible in the SPW_RMAP_STS register.

If valid but set, the corresponding error (incoherent CRC, non-matching TLA, Destination key error, etc.) can be retrieved.

35.7.6 Time Code

35.7.6.1 Overview

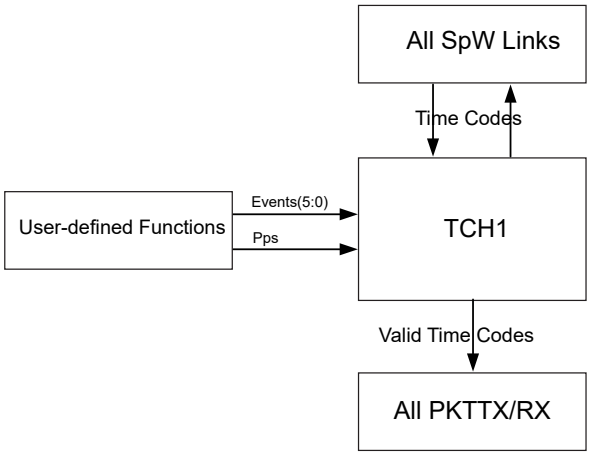
The time code handler (TCH1) receives and transmits time codes over multiple SpaceWire interfaces. It operates either in Master mode where it is the source of time codes, or in Slave mode where it is driven by incoming time codes. In Slave mode it can also act as a repeater and redistribute the time codes over other interfaces.

Time codes range from 0–63, are auto-incrementing and wrap around. A valid time code is defined as being equal to SPW_TCH_LASTTIMECODE.VALUE + 1 modulo 64. Since any incoming time code updates the register, only the first of multiple equal incoming time codes will be valid.

Note that the hardware does not differentiate between Master and Slave modes. In some special cases, they can be mixed by configuring sources in both SPW_TCH_CFGLISTEN and SPW_TCH_CFG.EVENT.

The TCH1 is connected to all SpW Link Handlers, and thus can drive or listen to any subset of links.

Figure 35-9. Time Code Handler Block



The TCH1 sends Time Synchronization events with the current time code to all Packet Transmitters and Packet Receivers to be used for transmission start.

The time code sources used are RTCOUT0/1 signals. They are internally connected to the SpaceWire block.

The table below shows the mapping between SpW links and the bits in SPW_TCH_CFGLISTEN and SPW_TCH_CFGSEND registers.

Table 35-7. Mapping between SpaceWire Link and TCH1 Configuration

Register Bit	1	0
Link	2	1

35.7.6.2 Operation/Usage

35.7.6.2.1 Master Mode

To run in Master mode, the following set-up must be done:

- Clear SPW_TCH_CFGLISTEN.
- Configure SPW_TCH_CFGSEND so that all interfaces where time codes should be transmitted are selected.

- Configure SPW_TCH_CFG.EVENT to select the event source that drives the time codes. The actual event sources are system-dependent and could be, e.g., RTCOUT0/1 signals.

It is possible to run Master mode manually from software without an event source by setting SPW_TCH_LASTTIMECODE.SEND each time a time code should be sent.

Selecting Time Code
Each configured event SPW_TCH_LASTTIMECODE.VALUE will be increased by 1 (modulo 64) and then transmitted.

To select Time Code N to be transmitted, write N-1 to SPW_TCH_LASTTIMECODE.VALUE before the configured event.

Synchronized Time Code Loop
To keep the full time code loop synchronized to an event (e.g. RTCOUT0/1), the SPW_TCH_CFGRESTART register can be set up to regularly set the time code value.

The events that can be used to trigger this are the same events as in the SPW_TCH_CONFIG.EVENT field. The SPW_TCH_CFGRESTART register also has an additional RTCOUT0/1 event which a system can connect to some source not shared by the Event sources.

When the configure event triggers, there is a delay of 2–3 SpWclk pulses, after which SPW_TCH_CFGRESTART.VALUE is copied to SPW_TCH_LASTTIMECODE.VALUE. This delay is to ensure that the Master mode trigger occurs before the Restart trigger if both have the same source or sources close in time.

If SPW_TCH_CFGRESTART.ONESHOT is set, this will only occur once. This can be used to prepare a change to SPW_TCH_LASTTIMECODE.VALUE with fewer timing constraints.

35.7.6.2.2 Slave Mode

To run in Slave mode, set up the following:

- The SPW_TCH_CFGLISTEN register must enable all SpW links where time codes may come from.
- The SPW_TCH_CFG register must be cleared.

If this is a pure slave node in a system, the SPW_TCH_CFGSEND register should be cleared. If this is used as a router/repeater, the SPW_TCH_CFGSEND register should select all interfaces where time codes should be retransmitted.

35.7.6.2.3 Watchdog

A watchdog notification can be set up to detect whether a time code arrives in the expected time window. This is typically only used in Slave mode.

To be notified about a time code coming too early, set up the SPW_TCH_CFGWD.EARLY register. If a time code arrives before that, the EARLYWD interrupt is set.

To be notified about a time code coming too late, set up the SPW_TCH_CFGWD.LATE register. If no time code has arrived by then, the LATEWD interrupt is set.

The watchdogs are reset when a valid time code is received, and only valid time codes are considered by the watchdog.

35.7.6.2.4 Notifications

To react on any valid time code, the TIMECODE interrupt can be used. This is used for both Master and Slave modes.

Use the SPW_TCH_CFGTCEVENT register to setup a trigger on specific timecode(s). For any bit set in SPW_TCH_CFGTCEVENT.MASK the corresponding bit in the time code and SPW_TCH_CFGTCEVENT.VALUE must be equal. When this triggers it gives the TCEVENT interrupt.

35.7.7 Interrupts

Interrupts are available to indicate the following events:

On the packet receiver side:

- the activation/deactivation of receive buffer
- an incoming character (EOP/EEP) detected

- an incoming packet discarded.

On the packet transmitter side:

- the activation/deactivation of a send list
- an incoming character (EOP/EEP) sent

On the TCH block:

- an early/late watchdog event triggered,
- a time code arrived,
- time synchronization event detected,
- a tc event detected

On each SpW link:

- incoming escape character which matches with ESCEVENT1 or 2
- a transmit packet discarded
- EEP received
- EEP transmitted
- the link state status
- link interface errors (credit error, ESC error, parity error, disconnection error)

35.7.7.1 Interrupt Register Access Definition

Each Pending Interrupt register has multiple access types to affect it in different ways. Each access type is set at a specific address.

Each Mask Interrupt register has multiple access types to affect it on different ways. Each access type is set at a specific address.

The following table describes the access types possible and the effect of reading and writing on them.

Register Name	Access Type	Read	Write
Pending Interrupt	RM	Read Pending Interrupt register Read Mask Interrupt register	–
	RCM	Read Pending Interrupt register and clear Pending Interrupt register	–
	R	Read Pending Interrupt register	–
	RC/S	Read and clear Interrupt register	Set Pending Interrupt register
	C	–	Clear Pending Interrupt register
Mask Interrupt	R/W	Read Mask Interrupt register	Write Mask Interrupt register
	S	–	Set Mask Interrupt
	C	–	Clear Mask Interrupt

Note: The pending interrupt can be forced for testing purposes by writing the Pending Interrupt register to ‘1’.

35.7.7.2 Interrupt Usage

The interrupts must be configured before using them.

To do so, the corresponding bit must be set in the SPW_PKTRX1, SPW_PKTTX1, SPW_TCH or SPW_LINKx Interrupt Mask registers (access type R/W or S).

When an interrupt is detected:

- The block origin of the interruption is found by reading the SpaceWire group Irq status 1 or 2 registers.
- More detail of the origin is then available by reading (access type RM, RCM, RC/S) the SPW_PKTRX1, SPW_PKTTX1, SPW_TCH or SPW_LINKx Pending Interrupt registers where the corresponding bit is pending.

There are several ways to clear a pending interrupt:

- Read and clear the Pending Interrupt register (access type RCM).
- Write to ‘0’ (Clear) the corresponding Pending Interrupt register. (access type C).

35.8 Register Summary

Offset	Name	Bit Pos.								
0x00	SPW_ROUTER_STS	7:0				DEST[4:0]				
		15:8				SOURCE[4:0]				
		23:16	BYTE[7:0]							
		31:24	COUNT[7:0]							
0x04	SPW_ROUTER_CFG	7:0						DISTIMEOUT	FALLBACK	LAENA
		15:8								
		23:16								
		31:24								
0x08	SPW_ROUTER_TIMEOUT	7:0				ADDR[4:0]				
		15:8								
		23:16								
		31:24	LOCKED							
0x0C ... 0x7F	Reserved									
0x80	SPW_ROUTER_TABLE 32	7:0				ADDR[4:0]				
		15:8								DELHEAD
		23:16								
		31:24								
...										
0x03FC	SPW_ROUTER_TABLE255	7:0				ADDR[4:0]				
		15:8								DELHEAD
		23:16								
		31:24								
0x0400	SPW_LINK1_PI_RM	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x0404	SPW_LINK1_PI_RCM	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x0408	SPW_LINK1_PI_R	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x040C	SPW_LINK1_PI_RCS	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x0410	SPW_LINK1_IM	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x0414	SPW_LINK1_PI_C	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x0418	SPW_LINK1_IM_S	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x041C	SPW_LINK1_IM_C	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0420	SPW_LINK1_CFG	7:0							COMMAND[1:0]	
		15:8								
		23:16								
		31:24								
0x0424	SPW_LINK1_CLKDIV	7:0				TXOPERDIV[4:0]				
		15:8								
		23:16				TXINITDIV[4:0]				
		31:24								
0x0428	SPW_LINK1_STATUS	7:0	TXDEFDIV[3:0]					LINKSTATE[2:0]		
		15:8								TXDEFDIV[4]
		23:16	SEEN3	SEEN2	SEEN1	SEEN0	GOTNCHAR	GOTFCT	GOTNULL	TXEMPTY
		31:24							SEEN5	SEEN4
0x042C	SPW_LINK1_SWRESET	7:0	PATTERN[7:0]							
		15:8	PATTERN[15:8]							
		23:16	PATTERN[23:16]							
		31:24	PATTERN[31:24]							
0x0430	SPW_LINK1_ESCCHAREVENT1	7:0	VALUE[7:0]							
		15:8	MASK[7:0]							
		23:16							HWEVENT	ACTIVE
		31:24								
0x0434	SPW_LINK1_ESCCHAREVENT2	7:0	VALUE[7:0]							
		15:8	MASK[7:0]							
		23:16							HWEVENT	ACTIVE
		31:24								
0x0438	SPW_LINK1_ESCCHARSTS	7:0	CHAR1[7:0]							
		15:8	CHAR2[7:0]							
		23:16								
		31:24								
0x043C	SPW_LINK1_TRANSESC	7:0	CHAR[7:0]							
		15:8								
		23:16								
		31:24								
0x0440	SPW_LINK1_DISTINTPI_RM	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x0444	SPW_LINK1_DISTINTPI_RCM	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x0448	SPW_LINK1_DISTINTPI_R	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x044C	SPW_LINK1_DISTINTPI_RCS	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x0450	SPW_LINK1_DISTINTIM	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x0454	SPW_LINK1_DISTINTPI_C	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24

.....continued

Offset	Name	Bit Pos.								
0x0458	SPW_LINK1_DISTINTIM_S	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x045C	SPW_LINK1_DISTINTIM_C	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x0460	SPW_LINK1_DISTACKPI_RM	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x0464	SPW_LINK1_DISTACKPI_RCM	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x0468	SPW_LINK1_DISTACKPI_R	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x046C	SPW_LINK1_DISTACKPI_RCS	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x0470	SPW_LINK1_DISTACKIM	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x0474	SPW_LINK1_DISTACKPI_C	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x0478	SPW_LINK1_DISTACKIM_S	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x047C	SPW_LINK1_DISTACKIM_C	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x0480	SPW_LINK2_PI_RM	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x0484	SPW_LINK2_PI_RCM	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x0488	SPW_LINK2_PI_R	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x048C	SPW_LINK2_PI_RCS	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0490	SPW_LINK2_IM	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x0494	SPW_LINK2_PI_C	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x0498	SPW_LINK2_IM_S	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x049C	SPW_LINK2_IM_C	7:0	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
		15:8							ESCEVENT1	ESCEVENT2
		23:16								
		31:24								
0x04A0	SPW_LINK2_CFG	7:0							COMMAND[1:0]	
		15:8								
		23:16								
		31:24								
0x04A4	SPW_LINK2_CLKDIV	7:0				TXOPERDIV[4:0]				
		15:8								
		23:16				TXINITDIV[4:0]				
		31:24								
0x04A8	SPW_LINK2_STATUS	7:0	TXDEFDIV[3:0]					LINKSTATE[2:0]		
		15:8								TXDEFDIV[4]
		23:16	SEEN3	SEEN2	SEEN1	SEEN0	GOTNCHAR	GOTFCT	GOTNULL	TXEMPTY
		31:24							SEEN5	SEEN4
0x04AC	SPW_LINK2_SWRESET	7:0	PATTERN[7:0]							
		15:8	PATTERN[15:8]							
		23:16	PATTERN[23:16]							
		31:24	PATTERN[31:24]							
0x04B0	SPW_LINK2_ESCCHAREVENT1	7:0	VALUE[7:0]							
		15:8	MASK[7:0]							
		23:16							HWEVENT	ACTIVE
		31:24								
0x04B4	SPW_LINK2_ESCCHAREVENT2	7:0	VALUE[7:0]							
		15:8	MASK[7:0]							
		23:16							HWEVENT	ACTIVE
		31:24								
0x04B8	SPW_LINK2_ESCCHARSTS	7:0	CHAR1[7:0]							
		15:8	CHAR2[7:0]							
		23:16								
		31:24								
0x04BC	SPW_LINK2_TRANSESC	7:0	CHAR[7:0]							
		15:8								
		23:16								
		31:24								
0x04C0	SPW_LINK2_DISTINTPI_RM	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x04C4	SPW_LINK2_DISTINTPI_RCM	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24

.....continued

Offset	Name	Bit Pos.								
0x04C8	SPW_LINK2_DISTINTPI_R	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x04CC	SPW_LINK2_DISTINTPI_RCS	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x04D0	SPW_LINK2_DISTINTIM	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x04D4	SPW_LINK2_DISTINTPI_C	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x04D8	SPW_LINK2_DISTINTIM_S	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x04DC	SPW_LINK2_DISTINTIM_C	7:0	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
		15:8	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
		23:16	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
		31:24	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
0x04E0	SPW_LINK2_DISTACKPI_RM	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x04E4	SPW_LINK2_DISTACKPI_RCM	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x04E8	SPW_LINK2_DISTACKPI_R	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x04EC	SPW_LINK2_DISTACKPI_RCS	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x04F0	SPW_LINK2_DISTACKIM	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x04F4	SPW_LINK2_DISTACKPI_C	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x04F8	SPW_LINK2_DISTACKIM_S	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x04FC	SPW_LINK2_DISTACKIM_C	7:0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
		15:8	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
		23:16	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
		31:24	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
0x0500 ... 0x07FF	Reserved									

.....continued

Offset	Name	Bit Pos.									
0x0800	SPW_PKTRX1_PIRM	7:0				ACT	DISCARD	EEP	EOP	DEACT
		15:8								
		23:16								
		31:24								
0x0804	SPW_PKTRX1_PIRCM	7:0				ACT	DISCARD	EEP	EOP	DEACT
		15:8								
		23:16								
		31:24								
0x0808	SPW_PKTRX1_PIR	7:0				ACT	DISCARD	EEP	EOP	DEACT
		15:8								
		23:16								
		31:24								
0x080C	SPW_PKTRX1_PIRCS	7:0				ACT	DISCARD	EEP	EOP	DEACT
		15:8								
		23:16								
		31:24								
0x0810	SPW_PKTRX1_IM	7:0				ACT	DISCARD	EEP	EOP	DEACT
		15:8								
		23:16								
		31:24								
0x0814	SPW_PKTRX1_PIC	7:0				ACT	DISCARD	EEP	EOP	DEACT
		15:8								
		23:16								
		31:24								
0x0818	SPW_PKTRX1_IMS	7:0				ACT	DISCARD	EEP	EOP	DEACT
		15:8								
		23:16								
		31:24								
0x081C	SPW_PKTRX1_IMC	7:0				ACT	DISCARD	EEP	EOP	DEACT
		15:8								
		23:16								
		31:24								
0x0820	SPW_PKTRX1_CFG	7:0								DISCARD
		15:8								
		23:16								
		31:24								
0x0824	SPW_PKTRX1_STATUS	7:0	COUNT[7:0]							
		15:8	COUNT[15:8]							
		23:16			DEACT	PENDING	ACT	ARM	LOCKED	PACKET
		31:24								
0x0828 ... 0x082F	Reserved									
0x0830	SPW_PKTRX1_NXTBUFDATAADDR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x0834	SPW_PKTRX1_NXTBUFDATALEN	7:0	LEN[7:0]							
		15:8	LEN[15:8]							
		23:16	LEN[23:16]							
		31:24								
0x0838	SPW_PKTRX1_NXTBUFPKTADDR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x083C	SPW_PKTRX1_NXTBUFCFG	7:0	MAXCNT[7:0]							
		15:8	MAXCNT[15:8]							
		23:16	START[2:0]		VALUE[5:0]					
		31:24		SPLIT						START[2:0]

.....continued

Offset	Name	Bit Pos.								
0x0840	SPW_PKTRX1_CURBUFDATAADDR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x0844	SPW_PKTRX1_CURBUFDATALEN	7:0	LEN[7:0]							
		15:8	LEN[15:8]							
		23:16	LEN[23:16]							
		31:24								
0x0848	SPW_PKTRX1_CURBUFPKTADDR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x084C	SPW_PKTRX1_CURBUFCFG	7:0	MAXCNT[7:0]							
		15:8	MAXCNT[15:8]							
		23:16								
		31:24	ABORT	SPLIT						
0x0850	SPW_PKTRX1_PREVBUFDATALEN	7:0	LEN[7:0]							
		15:8	LEN[15:8]							
		23:16	LEN[23:16]							
		31:24								
0x0854	SPW_PKTRX1_PREVBUFSTS	7:0	CNT[7:0]							
		15:8	CNT[15:8]							
		23:16					DMAERR	FULLD	FULLI	EEP
		31:24	LOCKED							
0x0858 ... 0x087B	Reserved									
0x087C	SPW_PKTRX1_SWRESET	7:0	PATTERN[7:0]							
		15:8	PATTERN[15:8]							
		23:16	PATTERN[23:16]							
		31:24	PATTERN[31:24]							
0x0880 ... 0x0BFF	Reserved									
0x0C00	SPW_PKTTX1_PI_RM	7:0					EEP	EOP	ACT	DEACT
		15:8								
		23:16								
		31:24								
0x0C04	SPW_PKTTX1_PI_RCM	7:0					EEP	EOP	ACT	DEACT
		15:8								
		23:16								
		31:24								
0x0C08	SPW_PKTTX1_PI_R	7:0					EEP	EOP	ACT	DEACT
		15:8								
		23:16								
		31:24								
0x0C0C	SPW_PKTTX1_PI_RCS	7:0					EEP	EOP	ACT	DEACT
		15:8								
		23:16								
		31:24								
0x0C10	SPW_PKTTX1_IM	7:0					EEP	EOP	ACT	DEACT
		15:8								
		23:16								
		31:24								
0x0C14	SPW_PKTTX1_PI_C	7:0					EEP	EOP	ACT	DEACT
		15:8								
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0C18	SPW_PKTTX1_IM_S	7:0					EEP	EOP	ACT	DEACT
		15:8								
		23:16								
		31:24								
0x0C1C	SPW_PKTTX1_IM_C	7:0					EEP	EOP	ACT	DEACT
		15:8								
		23:16								
		31:24								
0x0C20	SPW_PKTTX1_STATUS	7:0					DEACT	PENDING	ACT	ARM
		15:8								
		23:16						PREV[2:0]		
		31:24								
0x0C24	SPW_PKTTX1_NXTSENDROUT	7:0	BYTE4[7:0]							
		15:8	BYTE3[7:0]							
		23:16	BYTE2[7:0]							
		31:24	BYTE1[7:0]							
0x0C28	SPW_PKTTX1_NXTSENDADDR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x0C2C	SPW_PKTTX1_NXTSENDCFG	7:0	LEN[7:0]							
		15:8	LEN[15:8]							
		23:16	START[1:0]							VALUE
		31:24			ABORT					
0x0C30	SPW_PKTTX1_CURSENDROUT	7:0	BYTE4[7:0]							
		15:8	BYTE3[7:0]							
		23:16	BYTE2[7:0]							
		31:24	BYTE1[7:0]							
0x0C34	SPW_PKTTX1_CURSENDADDR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x0C38	SPW_PKTTX1_CURSENDCFG	7:0	LEN[7:0]							
		15:8	LEN[15:8]							
		23:16								
		31:24	ABORT							
0x0C3C	SPW_PKTTX1_SWRESET	7:0	PATTERN[7:0]							
		15:8	PATTERN[15:8]							
		23:16	PATTERN[23:16]							
		31:24	PATTERN[31:24]							
0x0C40 ... 0x0DFF	Reserved									
0x0E00	SPW_RMAP1_CFG	7:0	DESTKEY[7:0]							
		15:8	TLA[7:0]							
		23:16								RMAPENA
		31:24								
0x0E04	SPW_RMAP1_STS_RC	7:0	ERRCODE[7:0]							
		15:8								VALID
		23:16								
		31:24								
0x0E08	SPW_RMAP1_STS	7:0	ERRCODE[7:0]							
		15:8								VALID
		23:16								
		31:24								
0x0E0C ... 0x0E7F	Reserved									

.....continued

Offset	Name	Bit Pos.									
0x0E80	SPW_TCH_PI_RM	7:0				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
		15:8								
		23:16								
		31:24								
0x0E84	SPW_TCH_PI_RCM	7:0				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
		15:8								
		23:16								
		31:24								
0x0E88	SPW_TCH_PI_R	7:0				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
		15:8								
		23:16								
		31:24								
0x0E8C	SPW_TCH_PI_RCS	7:0				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
		15:8								
		23:16								
		31:24								
0x0E90	SPW_TCH_IM	7:0				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
		15:8								
		23:16								
		31:24								
0x0E94	SPW_TCH_PI_C	7:0				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
		15:8								
		23:16								
		31:24								
0x0E98	SPW_TCH_IM_S	7:0				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
		15:8								
		23:16								
		31:24								
0x0E9C	SPW_TCH_IM_C	7:0				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
		15:8								
		23:16								
		31:24								
0x0EA0	SPW_TCH_CFGLISTEN	7:0							L1	L0
		15:8								
		23:16								
		31:24								
0x0EA4	SPW_TCH_CFGSEND	7:0							S1	S0
		15:8								
		23:16								
		31:24								
0x0EA8	SPW_TCH_CFG	7:0			EVENT[5:0]					
		15:8								
		23:16								
		31:24								
0x0EAC	SPW_TCH_CFGRESTART	7:0			VALUE[5:0]					
		15:8	ONESHOT	PPS						EVENT
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0EB0	SPW_TCH_CFGTCEVENT	7:0			VALUE[5:0]					
		15:8			MASK[5:0]					
		23:16								
		31:24								
0x0EB4	SPW_TCH_CFGWD	7:0	LATE[7:0]							
		15:8	LATE[15:8]							
		23:16	EARLY[7:0]							
		31:24	EARLY[15:8]							
0x0EB8	SPW_TCH_LASTTIMECODE	7:0			VALUE[5:0]					
		15:8								SEND
		23:16								
		31:24								
0x0EBC	SPW_TCH_SWRESET	7:0	PATTERN[7:0]							
		15:8	PATTERN[15:8]							
		23:16	PATTERN[23:16]							
		31:24	PATTERN[31:24]							
0x0EC0 ... 0x0EFF	Reserved									
0x0F00	SPW_GROUP_IRQSTS1	7:0	TX1							
		15:8	RX1							
		23:16								TCH
		31:24								
0x0F04	SPW_GROUP_IRQSTS2	7:0								
		15:8								
		23:16	Di1	Dia1	Link1	Di2	Dia2	Link2		
		31:24								
0x0F08 ... 0x0F0B	Reserved									
0x0F0C	SPW_GROUP_EDACSTS	7:0	CORR[7:0]							
		15:8	UNCORR[7:0]							
		23:16								
		31:24								

35.8.1 SpW Router Status

Name: SPW_ROUTER_STS
Offset: 0x00
Reset: 0x00000000
Property: Read-only

Note: This register is cleared on read.

Bit	31	30	29	28	27	26	25	24
	COUNT[7:0]							
Access								
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BYTE[7:0]							
Access								
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				SOURCE[4:0]				
Access								
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
				DEST[4:0]				
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bits 31:24 – COUNT[7:0] Packet Count
Number of dropped packets. Saturates at max.

Bits 23:16 – BYTE[7:0] Router byte
Router byte of first dropped packet.

Bits 12:8 – SOURCE[4:0] Source Address
Source of first dropped packet.

Bits 4:0 – DEST[4:0] Destination Address
Destination from table of first dropped packet.

35.8.2 SpW Router Config

Name: SPW_ROUTER_CFG
Offset: 0x04
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						DISTIMEOUT	FALLBACK	LAENA
Access								R
Reset						0	0	0

Bit 2 – DISTIMEOUT Disable Timeout

Value	Description
1	Disable timeout.

Bit 1 – FALLBACK Fallback Routing

Value	Description
1	Enable fallback routing.

Bit 0 – LAENA LA Routing Enable

Value	Description
1	Enable logical address routing.

35.8.3 SpW Router Timeout

Name: SPW_ROUTER_TIMEOUT
Offset: 0x08
Reset: 0x00000000
Property: Read-only

Note: This register is cleared on read.

Bit	31	30	29	28	27	26	25	24
	LOCKED							
Access	R							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				ADDR[4:0]				
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit 31 – LOCKED Locked

Value	Description
1	A timeout has occurred.

Bits 4:0 – ADDR[4:0] Physical Address
Identifies destination that stalled.

35.8.4 SpW Router Table x

Name: SPW_ROUTER_TABLEx
Offset: 0x80 + (x- 32)*0x04 [x= 32..255]
Reset: 0x00000000
Property: Read/Write

Note: There are 224 instances of this register in the user interface.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								DELHEAD
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
				ADDR[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 8 – DELHEAD Delete Header Byte

Value	Description
1	Discard router byte for this logical address.

Bits 4:0 – ADDR[4:0] Address

Value	Description
0	Logical address is unused.
1-18	Physical address.
Others	Invalid

35.8.5 SpW Link x Pending Read Masked Interrupt

Name: SPW_LINKx_PI_RM
Offset: 0x0400 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							ESCEVENT1	ESCEVENT2
Access							R/W	R/W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Reset	0	0	0	0	0	0	0	0

Bit 9 – ESCEVENT1 Escape Event 1

Value	Description
1	Incoming Escape Character matched SPW_LINKx_EscCharEvent1.

Bit 8 – ESCEVENT2 Escape Event 2

Value	Description
1	Incoming Escape Character matched SPW_LINKx_EscCharEvent2.

Bit 7 – DISCARD Discard

Value	Description
1	Transmit packet discarded.

Bit 6 – EEPREC EEP Received

Value	Description
1	EEP received.

Bit 5 – EEPTRANS EEP Transmitted

Value	Description
1	EEP transmitted. If the link is not running it might not have been transmitted on the link, just passed the interface to the codec.

Bit 4 – LINKABORT LinkAbort

Value	Description
1	SpW Link x Status.LinkState has made a transition from Run to Error Reset.

Bit 3 – CRERR CrErr

Value	Description
1	Link interface credit error detected.

Bit 2 – ESCERR ESCErr

Value	Description
1	Link interface ESC error detected.

Bit 1 – PARERR ParErr

Value	Description
1	Link interface parity error detected.

Bit 0 – DISERR DisErr

Value	Description
1	Link interface disconnection error detected.

35.8.6 SpW Link x Pending Read and Clear Masked Interrupt

Name: SPW_LINKx_PI_RCM
Offset: 0x0404 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							ESCEVENT1	ESCEVENT2
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 9 – ESCEVENT1 Escape Event 1

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT1.

Bit 8 – ESCEVENT2 Escape Event 2

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT2.

Bit 7 – DISCARD Discard

Value	Description
1	Transmit packet discarded.

Bit 6 – EEPREC EEP Received

Value	Description
1	EEP received.

Bit 5 – EEPTRANS EEP Transmitted

Value	Description
1	EEP transmitted. If the link is not running it might not have been transmitted on the link, just passed the interface to the codec.

Bit 4 – LINKABORT LinkAbort

Value	Description
1	SPW_LINKx_STATUS.LINKSTATE has made a transition from Run to Error Reset.

Bit 3 – CRERR CrErr

Value	Description
1	Link interface credit error detected.

Bit 2 – ESCERR ESCErr

Value	Description
1	Link interface ESC error detected.

Bit 1 – PARERR ParErr

Value	Description
1	Link interface parity error detected.

Bit 0 – DISERR DisErr

Value	Description
1	Link interface disconnection error detected.

35.8.7 SpW Link x Pending Read Interrupt

Name: SPW_LINKx_PI_R
Offset: 0x0408 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							ESCEVENT1	ESCEVENT2
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 9 – ESCEVENT1 Escape Event 1

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT1.

Bit 8 – ESCEVENT2 Escape Event 2

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT2.

Bit 7 – DISCARD Discard

Value	Description
1	Transmit packet discarded.

Bit 6 – EEPREC EEP Received

Value	Description
1	EEP received.

Bit 5 – EEPTRANS EEP Transmitted

Value	Description
1	EEP transmitted. If the link is not running it might not have been transmitted on the link, just passed the interface to the codec.

Bit 4 – LINKABORT LinkAbort

Value	Description
1	SPW_LINKx_STATUS.LINKSTATE has made a transition from Run to Error Reset.

Bit 3 – CRERR CrErr

Value	Description
1	Link interface credit error detected.

Bit 2 – ESCERR ESCErr

Value	Description
1	Link interface ESC error detected.

Bit 1 – PARERR ParErr

Value	Description
1	Link interface parity error detected.

Bit 0 – DISERR DisErr

Value	Description
1	Link interface disconnection error detected.

35.8.8 SpW Link x Pending Read and Clear Interrupt

Name: SPW_LINKx_PI_RCS
Offset: 0x040C + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							ESCEVENT1	ESCEVENT2
Access							R/W	R/W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 9 – ESCEVENT1 Escape Event 1

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT1.

Bit 8 – ESCEVENT2 Escape Event 2

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT2.

Bit 7 – DISCARD Discard

Value	Description
1	Transmit packet discarded.

Bit 6 – EEPREC EEP Received

Value	Description
1	EEP received.

Bit 5 – EEPTRANS EEP Transmitted

Value	Description
1	EEP transmitted. If the link is not running it might not have been transmitted on the link, just passed the interface to the codec.

Bit 4 – LINKABORT LinkAbort

Value	Description
1	SPW_LINKx_STATUS.LINKSTATE has made a transition from Run to Error Reset.

Bit 3 – CRERR CrErr

Value	Description
1	Link interface credit error detected.

Bit 2 – ESCERR ESCErr

Value	Description
1	Link interface ESC error detected.

Bit 1 – PARERR ParErr

Value	Description
1	Link interface parity error detected.

Bit 0 – DISERR DisErr

Value	Description
1	Link interface disconnection error detected.

35.8.9 SpW Link x Interrupt Mask

Name: SPW_LINKx_IM
Offset: 0x0410 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							ESCEVENT1	ESCEVENT2
Access							R/W	R/W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 9 – ESCEVENT1 Escape Event 1

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT1.

Bit 8 – ESCEVENT2 Escape Event 2

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT2.

Bit 7 – DISCARD Discard

Value	Description
1	Transmit packet discarded.

Bit 6 – EEPREC EEP Received

Value	Description
1	EEP received

Bit 5 – EEPTRANS EEP Transmitted

Value	Description
1	EEP transmitted. If the link is not running it might not have been transmitted on the link, just passed the interface to the codec.

Bit 4 – LINKABORT LinkAbort

Value	Description
1	SPW_LINKx_STATUS.LINKSTATE has made a transition from Run to Error Reset.

Bit 3 – CRERR CrErr

Value	Description
1	Link interface credit error detected.

Bit 2 – ESCERR ESCErr

Value	Description
1	Link interface ESC error detected.

Bit 1 – PARERR ParErr

Value	Description
1	Link interface parity error detected.

Bit 0 – DISERR DisErr

Value	Description
1	Link interface disconnection error detected.

35.8.10 SpW Link x Clear Pending Interrupt

Name: SPW_LINKx_PI_C
Offset: 0x0414 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							ESCEVENT1	ESCEVENT2
Access							W	W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit 9 – ESCEVENT1 Escape Event 1

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT1.

Bit 8 – ESCEVENT2 Escape Event 2

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT2.

Bit 7 – DISCARD Discard

Value	Description
1	Transmit packet discarded.

Bit 6 – EEPREC EEP Received

Value	Description
1	EEP received.

Bit 5 – EEPTRANS EEP Transmitted

Value	Description
1	EEP transmitted. If the link is not running it might not have been transmitted on the link, just passed the interface to the codec.

Bit 4 – LINKABORT LinkAbort

Value	Description
1	SPW_LINKx_STATUS.LINKSTATE has made a transition from Run to Error Reset.

Bit 3 – CRERR CrErr

Value	Description
1	Link interface credit error detected.

Bit 2 – ESCERR ESCErr

Value	Description
1	Link interface ESC error detected.

Bit 1 – PARERR ParErr

Value	Description
1	Link interface parity error detected.

Bit 0 – DISERR DisErr

Value	Description
1	Link interface disconnection error detected.

35.8.11 SpW Link x Interrupt Set Mask

Name: SPW_LINKx_IM_S
Offset: 0x0418 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							ESCEVENT1	ESCEVENT2
Access							W	W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit 9 – ESCEVENT1 Escape Event 1

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT1.

Bit 8 – ESCEVENT2 Escape Event 2

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT2.

Bit 7 – DISCARD Discard

Value	Description
1	Transmit packet discarded.

Bit 6 – EEPREC EEP Received

Value	Description
1	EEP received.

Bit 5 – EEPTRANS EEP Transmitted

Value	Description
1	EEP transmitted. If the link is not running it might not have been transmitted on the link, just passed the interface to the codec.

Bit 4 – LINKABORT LinkAbort

Value	Description
1	SPW_LINKx_STATUS.LINKSTATE has made a transition from Run to Error Reset.

Bit 3 – CRERR CrErr

Value	Description
1	Link interface credit error detected.

Bit 2 – ESCERR ESCErr

Value	Description
1	Link interface ESC error detected.

Bit 1 – PARERR ParErr

Value	Description
1	Link interface parity error detected.

Bit 0 – DISERR DisErr

Value	Description
1	Link interface disconnection error detected.

35.8.12 SpW Link x Interrupt Clear Mask

Name: SPW_LINKx_IM_C
Offset: 0x041C + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							ESCEVENT1	ESCEVENT2
Access							W	W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	DISCARD	EEPREC	EEPTRANS	LINKABORT	CRERR	ESCERR	PARERR	DISERR
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit 9 – ESCEVENT1 Escape Event 1

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT1.

Bit 8 – ESCEVENT2 Escape Event 2

Value	Description
1	Incoming Escape Character matched SPW_LINKx_ESCCHAREVENT2.

Bit 7 – DISCARD Discard

Value	Description
1	Transmit packet discarded.

Bit 6 – EEPREC EEP Received

Value	Description
1	EEP received.

Bit 5 – EEPTRANS EEP Transmitted

Value	Description
1	EEP transmitted. If the link is not running it might not have been transmitted on the link, just passed the interface to the codec.

Bit 4 – LINKABORT LinkAbort

Value	Description
1	SPW_LINKx_STATUS.LINKSTATE has made a transition from Run to Error Reset.

Bit 3 – CRERR CrErr

Value	Description
1	Link interface credit error detected.

Bit 2 – ESCERR ESCErr

Value	Description
1	Link interface ESC error detected.

Bit 1 – PARERR ParErr

Value	Description
1	Link interface parity error detected.

Bit 0 – DISERR DisErr

Value	Description
1	Link interface disconnection error detected.

35.8.13 SpW Link x Config

Name: SPW_LINKx_CFG
Offset: 0x0420 + (x-1)*0x80 [x=1..2]
Reset: 0x00000002
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							COMMAND[1:0]	
Access							R/W	R/W
Reset							1	0

Bits 1:0 – COMMAND[1:0] Command

Value	Name	Description
0	Link Disable	The link proceeds directly to the ErrorReset state when reaching the Run state.
1	No command	State is not actively changed.
2	Auto Start	The Codec waits in state Ready until the first NULL character is received.
3	Link Start	SpaceWire link can proceed to Started state.

35.8.14 SpW Link x Clock Division

Name: SPW_LINKx_CLKDIV
Offset: 0x0424 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
				TXINITDIV[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				TXOPERDIV[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 20:16 – TXINITDIV[4:0] TxInitDiv
The initial transmitter signalling bitrate is SpwClk / (0.5 x (TxInitDiv + 1)).

Bits 4:0 – TXOPERDIV[4:0] TxOperDiv
The operating transmitter signalling bitrate is SpwClk / (0.5 x (TxOperDiv + 1)) x (Double Data Rate).

35.8.15 SpW Link x Status

Name: SPW_LINKx_STATUS
Offset: 0x0428 + (x-1)*0x80 [x=1..2]
Reset: 0x00710002
Property: Read-only

Bit	31	30	29	28	27	26	25	24
							SEEN5	SEEN4
Access							R	R
Reset							0	0

Bit	23	22	21	20	19	18	17	16
	SEEN3	SEEN2	SEEN1	SEEN0	GOTNCHAR	GOTFCT	GOTNULL	TXEMPTY
Access	R	R	R	R	R	R	R	R
Reset	0	1	1	1	0	0	0	1

Bit	15	14	13	12	11	10	9	8
								TXDEFDIV[4]
Access								R
Reset								0

Bit	7	6	5	4	3	2	1	0
	TXDEFDIV[3:0]					LINKSTATE[2:0]		
Access	R	R	R	R		R	R	R
Reset	0	0	0	0		0	1	0

Bit 25 – SEEN5 SEEN5

Value	Description
1	State Run seen.

Bit 24 – SEEN4 SEEN4

Value	Description
1	State Connecting seen.

Bit 23 – SEEN3 SEEN3

Value	Description
1	State Started seen.

Bit 22 – SEEN2 SEEN2

Value	Description
1	State Ready seen.

Bit 21 – SEEN1 SEEN1

Value	Description
1	State ErrorWait seen.

Bit 20 – SEEN0 SEEN0

Value	Description
1	State ErrorReset seen.

Bit 19 – GOTNCHAR GotNChar

Value	Description
1	Receiver got N–chars.

Bit 18 – GOTFCT GotFCT

Value	Description
1	Receiver got FCT.

Bit 17 – GOTNULL GotNull

Value	Description
1	Receiver got NULL.

Bit 16 – TXEMPTY TxEmpty

Value	Description
0	The transmit buffer is not empty.
1	The transmit buffer is empty.

Bits 8:4 – TXDEFDIV[4:0] TxDefDiv
The strapped reset value for transmitter bitrates.

Bits 2:0 – LINKSTATE[2:0] LinkState

Value	Name	Description
0	ERRORRESET	CODEC link state machine in ErrorReset state
1	ERRORWAIT	CODEC link state machine in ErrorWait state
2	READY	CODEC link state machine in Ready state
3	STARTED	CODEC link state machine in Started state
4	CONNECTING	CODEC link state machine in Connecting state
5	RUN	CODEC link state machine in Run state

35.8.16 SpW Link x Software Reset

Name: SPW_LINKx_SWRESET
Offset: 0x042C + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	PATTERN[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PATTERN[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PATTERN[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PATTERN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – PATTERN[31:0] Pattern

Value	Description
4D61_6A6F	Arm reset
7254_6F6D	Trigger reset, if armed.
Others	Clear Arm

35.8.17 SpW Link x Escape Character Event 1

Name: SPW_LINKx_ESCCHAREVENT1
Offset: 0x0430 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
							HWEVENT	ACTIVE
Access							R/W	R/W
Reset							0	0

Bit	15	14	13	12	11	10	9	8
	MASK[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	VALUE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 17 – HWEVENT HwEvent

Value	Description
1	Generate Hardware Event.

Bit 16 – ACTIVE Active

Value	Description
1	Event is active.

Bits 15:8 – MASK[7:0] Mask
Bits of new Time Code to check.

Bits 7:0 – VALUE[7:0] Value
Value used to check the incoming Esc Char.

35.8.18 SpW Link x Escape Character Event 2

Name: SPW_LINKx_ESCCHAREVENT2
Offset: 0x0434 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
							HWEVENT	ACTIVE
Access							R/W	R/W
Reset							0	0

Bit	15	14	13	12	11	10	9	8
	MASK[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	VALUE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 17 – HWEVENT HwEvent

Value	Description
1	Generate Hardware Event.

Bit 16 – ACTIVE Active

Value	Description
1	Event is active.

Bits 15:8 – MASK[7:0] Mask
Bits of new Time Code to check.

Bits 7:0 – VALUE[7:0] Value
Value used to check incoming Esc Char.

35.8.19 SpW Link x Escape Character Status

Name: SPW_LINKx_ESCCHARSTS
Offset: 0x0438 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	CHAR2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CHAR1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:8 – CHAR2[7:0] Esc Char 2
Latest Received Escape Character matching Event 2.

Bits 7:0 – CHAR1[7:0] Esc Char 1
Latest Received Escape Character matching Event 1.

35.8.20 SpW Link x Transmit Escape Character

Name: SPW_LINKx_TRANSESC
Offset: 0x043C + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	CHAR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – CHAR[7:0] Character
Escape Character to transmit.

35.8.21 SpW Link x Distributed Interrupt Pending Read Masked Interrupt

Name: SPW_LINKx_DISTINTPI_RM
Offset: 0x0440 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – DIx Distributed Interrupt x

Value	Description
1	Distributed Interrupt received.

35.8.22 SpW Link x Distributed Interrupt Pending Read and Clear Masked Interrupt

Name: SPW_LINKx_DISTINTPI_RCM
Offset: 0x0444 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – DIx Distributed Interrupt x

Value	Description
1	Distributed Interrupt received

35.8.23 SpW Link x Distributed Interrupt Pending Read Interrupt

Name: SPW_LINKx_DISTINTPI_R
Offset: 0x0448 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – DIx Distributed Interrupt x

Value	Description
1	Distributed Interrupt received

35.8.24 SpW Link x Distributed Interrupt Pending Read and Clear Interrupt

Name: SPW_LINKx_DISTINTPI_RCS
Offset: 0x044C + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DIx Distributed Interrupt x

Value	Description
1	Distributed Interrupt received

35.8.25 SpW Link x Distributed Interrupt Mask

Name: SPW_LINKx_DISTINTIM
Offset: 0x0450 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DIx Distributed Interrupt Mask x

Value	Description
1	Distributed Interrupt received mask

35.8.26 SpW Link x Distributed Interrupt Clear Pending Interrupt

Name: SPW_LINKx_DISTINTPI_C
Offset: 0x0454 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DIx Distributed Interrupt x

Value	Description
1	Distributed Interrupt received

35.8.27 SpW Link x Distributed Set Interrupt Mask

Name: SPW_LINKx_DISTINTIM_S
Offset: 0x0458 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DIx Distributed Interrupt x

Value	Description
1	Distributed Interrupt received mask.

35.8.28 SpW Link x Distributed Clear Interrupt Mask

Name: SPW_LINKx_DISTINTIM_C
Offset: 0x045C + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	DI31	DI30	DI29	DI28	DI27	DI26	DI25	DI24
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DI23	DI22	DI21	DI20	DI19	DI18	DI17	DI16
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DIx Distributed Interrupt Mask x

Value	Description
1	Distributed Interrupt received mask.

35.8.29 SpW Link x Distributed Interrupt Acknowledge Pending Read Masked Interrupt

Name: SPW_LINKx_DISTACKPI_RM
Offset: 0x0460 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – DAx Distributed Acknowledge x

Value	Description
1	Distributed Acknowledge received.

35.8.30 SpW Link x Distributed Interrupt Acknowledge Pending Read and Clear Masked Interrupt

Name: SPW_LINKx_DISTACKPI_RCM
Offset: 0x0464 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – DAx Distributed Acknowledge x

Value	Description
1	Distributed Acknowledge received.

35.8.31 SpW Link x Distributed Interrupt Acknowledge Pending Read Interrupt

Name: SPW_LINKx_DISTACKPI_R
Offset: 0x0468 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – DAx Distributed Acknowledge x

Value	Description
1	Distributed Acknowledge received.

35.8.32 SpW Link x Distributed Interrupt Acknowledge Pending Read and Clear Interrupt

Name: SPW_LINKx_DISTACKPI_RCS
Offset: 0x046C + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DAx Distributed Acknowledge x

Value	Description
1	Distributed Acknowledge received.

35.8.33 SpW Link x Distributed Interrupt Acknowledge Mask

Name: SPW_LINKx_DISTACKIM
Offset: 0x0470 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DAx Distributed Acknowledge Mask x

Value	Description
1	Distributed Acknowledge received mask.

35.8.34 SpW Link x Distributed Interrupt Acknowledge Clear Pending Interrupt

Name: SPW_LINKx_DISTACKPI_C
Offset: 0x0474 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DAx Distributed Acknowledge x

Value	Description
1	Distributed Acknowledge received.

35.8.35 SpW Link x Distributed Interrupt Acknowledge Set Mask

Name: SPW_LINKx_DISTACKIM_S
Offset: 0x0478 + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DAx Distributed Acknowledge Mask x

Value	Description
1	Distributed Acknowledge received mask.

35.8.36 SpW Link x Distributed Interrupt Acknowledge Clear Mask

Name: SPW_LINKx_DISTACKIM_C
Offset: 0x047C + (x-1)*0x80 [x=1..2]
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bits 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 – DAx Distributed Acknowledge Mask x

Value	Description
1	Distributed Acknowledge received mask.

35.8.37 SpW PktRx 1 Pending Read Masked Interrupt

Name: SPW_PKTRX1_PI_RM
Offset: 0x0800
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				ACT	DISCARD	EEP	EOP	DEACT
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit 4 – ACT Activated
Note: The content of the register is automatically ANDed with the corresponding masked register to obtain the masked information directly.

Value	Description
1	Received buffer activated.

Bit 3 – DISCARD Packet Discard

Value	Description
1	An incoming packet was discarded.

Bit 2 – EEP EEP Seen

Value	Description
1	An incoming EEP detected.

Bit 1 – EOP EOP Seen

Value	Description
1	An incoming EOP detected.

Bit 0 – DEACT Deactivated

Value	Description
1	Received buffer deactivated.

35.8.38 SpW PktRx 1 Pending Read and Clear Masked Interrupt

Name: SPW_PKTRX1_PI_RCM
Offset: 0x0804
Reset: 0x00000000
Property: Read-only

Note: This register is cleared on read.

Note: The register contents are automatically ANDed with the corresponding masked register to obtain the masked information directly.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				ACT	DISCARD	EEP	EOP	DEACT
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit 4 – ACT Activated

Value	Description
1	Received buffer activated.

Bit 3 – DISCARD Packet Discard

Value	Description
1	An incoming packet was discarded.

Bit 2 – EEP EEP seen

Value	Description
1	An incoming EEP detected.

Bit 1 – EOP EOP seen

Value	Description
1	An incoming EOP detected.

Bit 0 – DEACT Deactivated

Value	Description
1	Received buffer deactivated.

35.8.39 SpW PktRx 1 Pending Read Interrupt

Name: SPW_PKTRX1_PI_R
Offset: 0x0808
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				ACT	DISCARD	EEP	EOP	DEACT
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 4 – ACT Activated

Value	Description
1	Received buffer activated.

Bit 3 – DISCARD Packet Discard

Value	Description
1	An incoming packet was discarded.

Bit 2 – EEP EEP seen

Value	Description
1	An incoming EEP detected.

Bit 1 – EOP EOP seen

Value	Description
1	An incoming EOP detected.

Bit 0 – DEACT Deactivated

Value	Description
1	Received buffer deactivated.

35.8.40 SpW PktRx 1 Pending Read and Clear Interrupt

Name: SPW_PKTRX1_PI_RCS
Offset: 0x080C
Reset: 0x00000000
Property: Read/Write

Note: This register is cleared on read.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				ACT	DISCARD	EEP	EOP	DEACT
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 4 – ACT Activated

Value	Description
1	Received buffer activated.

Bit 3 – DISCARD Packet Discard

Value	Description
1	An incoming packet was discarded.

Bit 2 – EEP EEP seen

Value	Description
1	An incoming EEP detected.

Bit 1 – EOP EOP seen

Value	Description
1	An incoming EOP detected.

Bit 0 – DEACT Deactivated

Value	Description
1	Received buffer deactivated.

35.8.41 SpW PktRx 1 Interrupt Mask

Name: SPW_PKTRX1_IM
Offset: 0x0810
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				ACT	DISCARD	EEP	EOP	DEACT
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 4 – ACT Activated

Value	Description
1	Received buffer activated.

Bit 3 – DISCARD Packet Discard

Value	Description
1	An incoming packet was discarded.

Bit 2 – EEP EEP seen

Value	Description
1	An incoming EEP detected.

Bit 1 – EOP EOP seen

Value	Description
1	An incoming EOP detected.

Bit 0 – DEACT Deactivated

Value	Description
1	Received buffer deactivated.

35.8.42 SpW PktRx 1 Clear Pending Interrupt

Name: SPW_PKTRX1_PI_C
Offset: 0x0814
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				ACT	DISCARD	EEP	EOP	DEACT
Access				W	W	W	W	W
Reset				0	0	0	0	0

Bit 4 – ACT Activated

Value	Description
1	Received buffer activated.

Bit 3 – DISCARD Packet Discard

Value	Description
1	An incoming packet was discarded.

Bit 2 – EEP EEP seen

Value	Description
1	An incoming EEP detected.

Bit 1 – EOP EOP seen

Value	Description
1	An incoming EOP detected.

Bit 0 – DEACT Deactivated

Value	Description
1	Received buffer deactivated.

35.8.43 SpW PktRx 1 Interrupt Set Mask

Name: SPW_PKTRX1_IM_S
Offset: 0x0818
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				ACT	DISCARD	EEP	EOP	DEACT
Access				W	W	W	W	W
Reset				0	0	0	0	0

Bit 4 – ACT Activated

Value	Description
1	Received buffer activated.

Bit 3 – DISCARD Packet Discard

Value	Description
1	An incoming packet was discarded.

Bit 2 – EEP EEP seen

Value	Description
1	An incoming EEP detected.

Bit 1 – EOP EOP seen

Value	Description
1	An incoming EOP detected.

Bit 0 – DEACT Deactivated

Value	Description
1	Received buffer deactivated

35.8.44 SpW PktRx 1 Interrupt Clear Mask

Name: SPW_PKTRX1_IM_C
Offset: 0x081C
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				ACT	DISCARD	EEP	EOP	DEACT
Access				W	W	W	W	W
Reset				0	0	0	0	0

Bit 4 – ACT Activated

Value	Description
1	Received buffer activated.

Bit 3 – DISCARD Packet Discard

Value	Description
1	An incoming packet was discarded.

Bit 2 – EEP EEP seen

Value	Description
1	An incoming EEP detected

Bit 1 – EOP EOP seen

Value	Description
1	An incoming EOP detected.

Bit 0 – DEACT Deactivated

Value	Description
1	Received buffer deactivated.

35.8.45 SpW PktRx 1 Config

Name: SPW_PKTRX1_CFG
Offset: 0x0820
Reset: 0x00000001
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								DISCARD
Access								R/W
Reset								1

Bit 0 – DISCARD Discard

Value	Description
0	Stall incoming if inactive.
1	Discard incoming if inactive.

35.8.46 SpW PktRx 1 Status

Name: SPW_PKTRX1_STATUS
Offset: 0x0824
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			DEACT	PENDING	ACT	ARM	LOCKED	PACKET
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 21 – DEACT Deactivating

Value	Description
1	Current Receive Buffer is ready to deactivate.

Bit 20 – PENDING Pending

Value	Description
1	Next Receive Buffer is pending.

Bit 19 – ACT Active

Value	Description
1	Current receive buffer active.

Bit 18 – ARM Armed

Value	Description
1	Next buffer is armed.

Bit 17 – LOCKED Locked

Value	Description
1	Previous buffer is locked.

Bit 16 – PACKET Packet

Value	Description
1	Incoming Packet ongoing.

Bits 15:0 – COUNT[15:0] Packet Count
Number of packets in active buffer.

35.8.47 SpW PktRx 1 Next Buffer Data Address

Name: SPW_PKTRX1_NXTBUFDATAADDR
Offset: 0x0830
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Address
Next Buffer data address.

35.8.48 SpW PktRx 1 Next Buffer Data Length

Name: SPW_PKTRX1_NXTBUFDATALEN
Offset: 0x0834
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	LEN[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	LEN[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LEN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – LEN[23:0] Length
Next Buffer Data Length, in bytes.

35.8.49 SpW PktRx 1 Next Buffer Packet Address

Name: SPW_PKTRX1_NXTBUFPKTADDR
Offset: 0x0838
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Address
Next Buffer Packet address.

35.8.50 SpW PktRx 1 Next Buffer Config

Name: SPW_PKTRX1_NXTBUFCFG
Offset: 0x083C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
		SPLIT						START[2:0]
Access		R/W						R/W
Reset		0						0

Bit	23	22	21	20	19	18	17	16
	START[2:0]		VALUE[5:0]					
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MAXCNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MAXCNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 30 – SPLIT Split Pkt
Select how Start may switch buffers.

Value	Description
0	No effect.
1	If a start condition is met, split any ongoing packet and deactivate.

Bits 24:22 – START[2:0] Start Mode

Value	Name	Description
0	STARTEVENT	Start if any bit in Start Value matches an incoming event
1	STARTNOW	Start immediately. Request a deactivation on next packet boundary.
2	STARTTCH1	Start if Start Value matches an incoming Time Code from Time Code Handler 1
3	Reserved	Reserved
4	STARTLATER	Start when current buffer is deactivated (e.g., by buffer becoming full).

Bits 21:16 – VALUE[5:0] Start Value
Activate on matching Start Value. The interpretation depends on the START field.

Bits 15:0 – MAXCNT[15:0] Max Count
Next Buffer Packet Length, measured in number of packets, i.e. entries of 16 bytes. If MaxCnt is zero, the entire register is cleared.

35.8.51 SpW PktRx 1 Current Buffer Data Address

Name: SPW_PKTRX1_CURBUFDATAADDR
Offset: 0x0840
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Address
Current start address to buffer for received packet data.

35.8.52 SpW PktRx 1 Current Buffer Data Length

Name: SPW_PKTRX1_CURBUFDATALEN
Offset: 0x0844
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	LEN[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	LEN[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LEN[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – LEN[23:0] Length
Current buffer data length, in bytes.

35.8.53 SpW PktRx 1 Current Buffer Packet Address

Name: SPW_PKTRX1_CURBUFPKTADDR
Offset: 0x0848
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Address
Current buffer packet start address.

35.8.54 SpW PktRx 1 Current Buffer Config

Name: SPW_PKTRX1_CURBUFCFG
Offset: 0x084C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ABORT	SPLIT						
Access	R/W	R/W						
Reset	0	0						

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	MAXCNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MAXCNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – ABORT Abort

Value	Description
0	No abort.
1	Immediately abort and discard any ongoing packet.

Bit 30 – SPLIT Split

Value	Description
0	No split.
1	Immediately split any ongoing packet.

Bits 15:0 – MAXCNT[15:0] Max Count
Current buffer packet length, measured in number of packets, i.e., entries of 16 bytes.

35.8.55 SpW PktRx 1 Previous Buffer Data Length

Name: SPW_PKTRX1_PREVBUFDATALEN
Offset: 0x0850
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	LEN[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	LEN[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LEN[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – LEN[23:0] Length
Previous buffer data length, in bytes.

35.8.56 SpW PktRx 1 Previous Buffer Status

Name: SPW_PKTRX1_PREVBUFSTS
Offset: 0x0854
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	LOCKED							
Access	R/W							
Reset	0							

Bit	23	22	21	20	19	18	17	16
					DMAERR	FULLD	FULLI	EEP
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – LOCKED Locked

Value	Description
1	Previous buffer is locked. This bit is cleared when read.

Bit 19 – DMAERR DMA Error

Value	Description
1	DMA Error during packet write.

Bit 18 – FULLD Buffer Data Full

Value	Description
1	Receive Buffer data has become full.

Bit 17 – FULLI Buffer Info Full

Value	Description
1	Receive Buffer information has become full.

Bit 16 – EEP EEP seen

Value	Description
1	An incoming EEP detected in this buffer.

Bits 15:0 – CNT[15:0] Count
Number of packets received in previous buffer, i.e. number of Receive Packet Info entries stored.

35.8.57 SpW PktRx 1 Software Reset

Name: SPW_PKTRX1_SWRESET
Offset: 0x087C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	PATTERN[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PATTERN[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PATTERN[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PATTERN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – PATTERN[31:0] Pattern
4D61_6A6F: Arm reset.
7254_6F6D: Trigger reset, if armed.
Others: Clear Arm

35.8.58 SpW PktTx 1 Pending Read Masked Interrupt

Name: SPW_PKTTX1_PI_RM
Offset: 0x0C00
Reset: 0x00000000
Property: Read-only

Note: The register contents are automatically ANDed with the corresponding masked register to obtain the masked information directly.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					EEP	EOP	ACT	DEACT
Access					R	R	R	R
Reset					0	0	0	0

Bit 3 – EEP EEP Sent
Note: the content of the register is automatically ANDed with the corresponding masked register to obtain directly the masked information

Value	Description
1	An EEP was sent, i.e., an outgoing packet was discarded.

Bit 2 – EOP EOP Sent

Value	Description
1	An EOP was sent.

Bit 1 – ACT Activated

Value	Description
1	Send List activated.

Bit 0 – DEACT Deactivated

Value	Description
1	Send List deactivated.

35.8.59 SpW PktTx 1 Pending Read and Clear Masked Interrupt

Name: SPW_PKTTX1_PI_RCM
Offset: 0x0C04
Reset: 0x00000000
Property: Read-only

Note: This register is cleared on read.

Note: The register contents are automatically ANDed with the corresponding masked register to obtain the masked information directly.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					EEP	EOP	ACT	DEACT
Access					R	R	R	R
Reset					0	0	0	0

Bit 3 – EEP EEP Sent

Value	Description
1	An EEP was sent, i.e., an outgoing packet was discarded.

Bit 2 – EOP EOP Sent

Value	Description
1	An EOP was sent.

Bit 1 – ACT Activated

Value	Description
1	Send List activated.

Bit 0 – DEACT Deactivated

Value	Description
1	Send List deactivated.

35.8.60 SpW PktTx 1 Pending Read Interrupt

Name: SPW_PKTTX1_PI_R
Offset: 0x0C08
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					EEP	EOP	ACT	DEACT
Access					R	R	R	R
Reset					0	0	0	0

Bit 3 – EEP EEP Sent

Value	Description
1	An EEP was sent, i.e., an outgoing packet was discarded.

Bit 2 – EOP EOP Sent

Value	Description
1	An EOP was sent.

Bit 1 – ACT Activated

Value	Description
1	Send List activated.

Bit 0 – DEACT Deactivated

Value	Description
1	Send List deactivated.

35.8.61 SpW PktTx 1 Pending Read and Clear Interrupt

Name: SPW_PKTTX1_PI_RCS
Offset: 0x0C0C
Reset: 0x00000000
Property: Read/Write

Note: This register is cleared on read.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					EEP	EOP	ACT	DEACT
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit 3 – EEP EEP Sent

Value	Description
1	An EEP was sent, i.e., an outgoing packet was discarded.

Bit 2 – EOP EOP Sent

Value	Description
1	An EOP was sent.

Bit 1 – ACT Activated

Value	Description
1	Send List activated.

Bit 0 – DEACT Deactivated

Value	Description
1	Send List deactivated.

35.8.62 SpW PktTx 1 Interrupt Mask

Name: SPW_PKTTX1_IM
Offset: 0x0C10
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					EEP	EOP	ACT	DEACT
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit 3 – EEP EEP Sent

Value	Description
1	An EEP was sent, i.e., an outgoing packet was discarded.

Bit 2 – EOP EOP Sent

Value	Description
1	An EOP was sent.

Bit 1 – ACT Activated

Value	Description
1	Send List activated.

Bit 0 – DEACT Deactivated

Value	Description
1	Send List deactivated.

35.8.63 SpW PktTx 1 Clear Pending Interrupt

Name: SPW_PKTTX1_PI_C
Offset: 0x0C14
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					EEP	EOP	ACT	DEACT
Access								
Reset					0	0	0	0

Bit 3 – EEP EEP Sent

Value	Description
1	An EEP was sent, i.e., an outgoing packet was discarded.

Bit 2 – EOP EOP Sent

Value	Description
1	An EOP was sent.

Bit 1 – ACT Activated

Value	Description
1	Send List activated.

Bit 0 – DEACT Deactivated

Value	Description
1	Send List deactivated.

35.8.64 SpW PktTx 1 Interrupt Set Mask

Name: SPW_PKTTX1_IM_S
Offset: 0x0C18
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					EEP	EOP	ACT	DEACT
Access					W	W	W	W
Reset					0	0	0	0

Bit 3 – EEP EEP Sent

Value	Description
1	An EEP was sent, i.e., an outgoing packet was discarded.

Bit 2 – EOP EOP sent

Value	Description
1	An EOP was sent.

Bit 1 – ACT Activated

Value	Description
1	Send List activated.

Bit 0 – DEACT Deactivated

Value	Description
1	Send List deactivated.

35.8.65 SpW PktTx 1 Interrupt Clear Mask

Name: SPW_PKTTX1_IM_C
Offset: 0x0C1C
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					EEP	EOP	ACT	DEACT
Access					W	W	W	W
Reset					0	0	0	0

Bit 3 – EEP EEP Sent

Value	Description
1	An EEP was sent, i.e., an outgoing packet was discarded.

Bit 2 – EOP EOP Sent

Value	Description
1	An EOP was sent.

Bit 1 – ACT Activated

Value	Description
1	Send List activated.

Bit 0 – DEACT Deactivated

Value	Description
1	Send List deactivated.

35.8.66 SpW PktTx 1 Status

Name: SPW_PKTTX1_STATUS
Offset: 0x0C20
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
						PREV[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					DEACT	PENDING	ACT	ARM
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 18:16 – PREV[2:0] Previous
Describes status of last deactivated send list. This is set simultaneously with the Deact interrupt. The field is locked when non–zero, and any active send list will not be able to deactivate.
Writing any value to this register clears the PREV field.

Value	Name	Description
0	NOINFO	No information. Field not locked.
1	LASTSENDLISTOK	Last send list fully done
2	ABORTEDMEMERR	Aborted due to memory access error.
3	ABORTEDNEWSD	Aborted by new send list.
4	ABORTEDUSERCMD	Aborted by direct user command.
5	ABORTEDTIMEOUT	Aborted by timeout.
others	OTHERS	INVALID

Bit 3 – DEACT Deactivating

Value	Description
1	Current Send list is ready to deactivate.

Bit 2 – PENDING Pending

Value	Description
1	Next Send list is pending.

Bit 1 – ACT Active

Value	Description
1	Current Send list active.

Bit 0 – ARM Armed

Value	Description
1	Next Send list armed.

35.8.67 SpW PktTx 1 Next Send List Router Bytes

Name: SPW_PKTTX1_NXTSENDROUT
Offset: 0x0C24
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	BYTE1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BYTE2[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	BYTE3[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BYTE4[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:24 – BYTE1[7:0] Byte1
First byte to prepend, if non–zero.

Bits 23:16 – BYTE2[7:0] Byte2
Second byte to prepend, if non–zero.

Bits 15:8 – BYTE3[7:0] Byte3
Third byte to prepend, if non–zero.

Bits 7:0 – BYTE4[7:0] Byte4
Fourth byte to prepend, if non–zero.

35.8.68 SpW PktTx 1 Next Send List Address

Name: SPW_PKTTX1_NXTSENDADDR
Offset: 0x0C28
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Address
Next Send list address.

35.8.69 SpW PktTx 1 Next Send List Config

Name: SPW_PKTTX1_NXTSENDCFG
Offset: 0x0C2C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
			ABORT					
Access			R/W					
Reset			0					

Bit	23	22	21	20	19	18	17	16
	START[1:0]							VALUE
Access	R/W	R/W						R/W
Reset	0	0						0

Bit	15	14	13	12	11	10	9	8
	LEN[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LEN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 29 – ABORT Abort

Value	Description
1	Abort ongoing Send list when this Send list wants to start.

Bits 23:22 – START[1:0] Start Mode

Value	Name	Description
0	STARTEVENT	Start if any bit in Start Value matches an incoming event
1	STARTNOW	Start immediately, if possible
2	STARTTCH1	Start if Start Value matches an incoming Time Code from Time Code Handler 1
3	Reserved	Reserved

Bit 16 – VALUE Start Value
Activate on matching Start value. The interpretation depends on the START field.

Bits 15:0 – LEN[15:0] Length
Number of entries in Send list.

35.8.70 SpW PktTx 1 Current Send List Router Bytes

Name: SPW_PKTTX1_CURSENDROUT
Offset: 0x0C30
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	BYTE1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BYTE2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	BYTE3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BYTE4[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:24 – BYTE1[7:0] Byte1
First byte to prepend, if non–zero.

Bits 23:16 – BYTE2[7:0] Byte2
Second byte to prepend, if non–zero.

Bits 15:8 – BYTE3[7:0] Byte3
Third byte to prepend, if non–zero.

Bits 7:0 – BYTE4[7:0] Byte4
Fourth byte to prepend, if non–zero.

35.8.71 SpW PktTx 1 Current Send List Address

Name: SPW_PKTTX1_CURSENDADDR
Offset: 0x0C34
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Address
Current Send list address.

35.8.72 SpW PktTx 1 Current Send List Config

Name: SPW_PKTTX1_CURSENDCFG
Offset: 0x0C38
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ABORT							
Access	R/W							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	LEN[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LEN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – ABORT Abort

Value	Description
0	No immediate Abort.
1	Immediately Abort any ongoing packet.

Bits 15:0 – LEN[15:0] Length
Number of entries in Send list.

35.8.73 SpW PktTx 1 Software Reset

Name: SPW_PKTTX1_SWRESET
Offset: 0x0C3C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	PATTERN[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PATTERN[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PATTERN[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PATTERN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – PATTERN[31:0] Pattern
4D61_6A6F: Arm reset.
7254_6F6D: Trigger reset, if armed.
Others: Clear Arm

35.8.74 Spw RMAP 1 Config

Name: SPW_RMAP1_CFG
Offset: 0x00000E00
Reset: 0x00010000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
								RMAPENA
Access								R/W
Reset								1

Bit	15	14	13	12	11	10	9	8
	TLA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DESTKEY[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 16 – RMAPENA RMAP Enable

Value	Description
0	Disable Access to this RMAP.
1	Enable Access to this RMAP.

Bits 15:8 – TLA[7:0] Address
Target Logical Address. Any incoming RMAP command must have a TLA matching this value, or TLA = 0xFE, in order to be accepted.

Bits 7:0 – DESTKEY[7:0] DestKey
RMAP Destination Key. Any incoming RMAP command must have a matching Destination Key in order to be accepted. If not, the complete packet is rejected during header check.

35.8.75 Spw RMAP 1 Read and Clear Status

Name: SPW_RMAP1_STS_RC
Offset: 0x0E04
Reset: 0x00000000
Property: Read-only

Note: This register is cleared on read.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								R
Reset								0

Bit	7	6	5	4	3	2	1	0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 8 – VALID Valid

Value	Description
0	No new error detected.
1	New error detected.

Bits 7:0 – ERRCODE[7:0] Error Code

Value	Name	Description
0	NOERROR	No error detected.
1	DMAERROR	Error while DMA accessing the internal bus, e.g., illegal address.
2	RMAPERROR	Unused RMAP command according to [RMAP].
3	DESTKEYERROR	Destination key error.
4	DATACRCERROR	Data CRC error.
5	EOPERROR	Early EOP in header or data, i.e., EOP has been received with less data than expected from the RMAP command header.
6	CARGOERROR	Cargo too large. Late EOP or EEP in data, i.e., EOP/EEP has been received with more data than expected from the RMAP command header.
7	EEPERROR	Early EEP in data for RMAP commands. EEP has been received with less data or exactly as much as expected from the RMAP command header.
10	CMDERROR	Authorization error: Invalid or unsupported command.
12	TLAERROR	Non-matching Target Logical Address.
16	HEADERCRCERROR	Incorrect header CRC.
17	PROTOCOLIDERROR	Protocol Identifier not supported.
18	REPLYADDERROR	Unsupported reply address length.

35.8.76 Spw RMAP 1 Read Status

Name: SPW_RMAP1_STS
Offset: 0x0E08
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								VALID
Access								R
Reset								0

Bit	7	6	5	4	3	2	1	0
	ERRCODE[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 8 – VALID Valid

Value	Description
0	No new error detected.
1	New error detected.

Bits 7:0 – ERRCODE[7:0] Error Code

Value	Name	Description
0	NOERROR	No error detected.
1	DMAERROR	Error while DMA accessing the internal bus, e.g., illegal address.
2	RMAPERROR	Unused RMAP command according to [RMAP].
3	DESTKEYERROR	Destination key error.
4	DATACRCERROR	Data CRC error.
5	EOPERROR	Early EOP in header or data, i.e., EOP has been received with less data than expected from the RMAP command header.
6	CARGOERROR	Cargo too large. Late EOP or EEP in data, i.e., EOP/EEP has been received with more data than expected from the RMAP command header.
7	EEPERROR	Early EEP in data for RMAP commands. EEP has been received with less data or exactly as much as expected from the RMAP command header.
10	CMDERROR	Authorization error: Invalid or unsupported command.
12	TLAERROR	Non-matching Target Logical Address.
16	HEADERCRCERROR	Incorrect header CRC.
17	PROTOCOLIDERROR	Protocol Identifier not supported.
18	REPLYADDERROR	Unsupported reply address length.

35.8.77 Spw Tch Pending Read Masked Interrupt

Name: SPW_TCH_PI_RM
Offset: 0x0E80
Reset: 0x00010000
Property: Read-only

Note: The register contents are automatically ANDed with the corresponding masked register to obtain the masked information directly.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit 4 – EARLYWD Early Watchdog

Value	Description
1	Early Watchdog triggered.

Bit 3 – LATEWD Late Watchdog

Value	Description
1	Late Watchdog triggered.

Bit 2 – ANYTIMECODE Any Time Code

Value	Description
1	Any Time Code arrived.

Bit 1 – TIMECODE Time Code

Value	Description
1	Time synchronization event, i.e., valid Time Code received or internal event triggered time code to send.

Bit 0 – TCEVENT TcEvent

Value	Description
1	Tc Event triggered.

35.8.78 Spw Tch Pending Read and Clear Masked Interrupt

Name: SPW_TCH_PI_RCM
Offset: 0x0E84
Reset: 0x00000000
Property: Read-only

Note: This register is cleared on read.

Note: The register contents are automatically ANDed with the corresponding masked register to obtain the masked information directly.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit 4 – EARLYWD Early Watchdog

Value	Description
1	Early Watchdog triggered.

Bit 3 – LATEWD Late Watchdog

Value	Description
1	Late Watchdog triggered.

Bit 2 – ANYTIMECODE Any Time Code

Value	Description
1	Any Time Code arrived.

Bit 1 – TIMECODE Time Code

Value	Description
1	Time synchronization event, i.e., valid Time Code received or internal event triggered time code to send.

Bit 0 – TCEVENT TcEvent

Value	Description
1	Tc Event triggered.

35.8.79 Spw Tch Pending Read Interrupt

Name: SPW_TCH_PI_R
Offset: 0x0E88
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit 4 – EARLYWD Early Watchdog

Value	Description
1	Early Watchdog triggered.

Bit 3 – LATEWD Late Watchdog

Value	Description
1	Late Watchdog triggered.

Bit 2 – ANYTIMECODE Any Time Code

Value	Description
1	Any Time Code arrived.

Bit 1 – TIMECODE Time Code

Value	Description
1	Time synchronization event, i.e., valid Time Code received or internal event triggered time code to send.

Bit 0 – TCEVENT TcEvent

Value	Description
1	Tc Event triggered.

35.8.80 Spw Tch Pending Read and Clear Interrupt

Name: SPW_TCH_PI_RCS
Offset: 0x0E8C
Reset: 0x00000000
Property: Read/Write

Note: This register is cleared on read.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 4 – EARLYWD Early Watchdog

Value	Description
1	Early Watchdog triggered.

Bit 3 – LATEWD Late Watchdog

Value	Description
1	Late Watchdog triggered.

Bit 2 – ANYTIMECODE Any Time Code

Value	Description
1	Any Time Code arrived.

Bit 1 – TIMECODE Time Code

Value	Description
1	Time synchronization event, i.e., valid Time Code received or internal event triggered time code to send.

Bit 0 – TCEVENT TcEvent

Value	Description
1	Tc Event triggered.

35.8.81 Spw Tch Interrupt Mask

Name: SPW_TCH_IM
Offset: 0x0E90
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bit 4 – EARLYWD Early Watchdog

Value	Description
1	Early Watchdog triggered.

Bit 3 – LATEWD Late Watchdog

Value	Description
1	Late Watchdog triggered.

Bit 2 – ANYTIMECODE Any Time Code

Value	Description
1	Any Time Code arrived.

Bit 1 – TIMECODE Time Code

Value	Description
1	Time synchronization event, i.e., valid Time Code received or internal event triggered time code to send.

Bit 0 – TCEVENT TcEvent

Value	Description
1	Tc Event triggered.

35.8.82 Spw Tch Clear Pending Interrupt

Name: SPW_TCH_PI_C
Offset: 0x0E94
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
Access				W	W	W	W	W
Reset				0	0	0	0	0

Bit 4 – EARLYWD Early Watchdog

Value	Description
1	Early Watchdog triggered.

Bit 3 – LATEWD Late Watchdog

Value	Description
1	Late Watchdog triggered.

Bit 2 – ANYTIMECODE Any Time Code

Value	Description
1	Any Time Code arrived.

Bit 1 – TIMECODE Time Code

Value	Description
1	Time synchronization event, i.e., valid Time Code received or internal event triggered time code to send.

Bit 0 – TCEVENT TcEvent

Value	Description
1	Tc Event triggered.

35.8.83 Spw Tch Interrupt Set Mask

Name: SPW_TCH_IM_S
Offset: 0x0E98
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
Access				W	W	W	W	W
Reset				0	0	0	0	0

Bit 4 – EARLYWD Early Watchdog

Value	Description
1	Early Watchdog triggered.

Bit 3 – LATEWD Late Watchdog

Value	Description
1	Late Watchdog triggered.

Bit 2 – ANYTIMECODE Any Time Code

Value	Description
1	Any Time Code arrived.

Bit 1 – TIMECODE Time Code

Value	Description
1	Time synchronization event, i.e., valid Time Code received or internal event triggered time code to send.

Bit 0 – TCEVENT TcEvent

Value	Description
1	Tc Event triggered.

35.8.84 Spw Tch Interrupt Clear Mask

Name: SPW_TCH_IM_C
Offset: 0x0E9C
Reset: 0x00000000
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				EARLYWD	LATEWD	ANYTIMECODE	TIMECODE	TCEVENT
Access				W	W	W	W	W
Reset				0	0	0	0	0

Bit 4 – EARLYWD Early Watchdog

Value	Description
1	Early Watchdog triggered.

Bit 3 – LATEWD Late Watchdog

Value	Description
1	Late Watchdog triggered.

Bit 2 – ANYTIMECODE Any Time Code

Value	Description
1	Any Time Code arrived.

Bit 1 – TIMECODE Time Code

Value	Description
1	Time synchronization event, i.e., valid Time Code received or internal event triggered time code to send.

Bit 0 – TCEVENT TcEvent

Value	Description
1	Tc Event triggered.

35.8.85 Spw Tch Config Listener

Name: SPW_TCH_CFGLISTEN
Offset: 0x0EA0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							L1	L0
Access							R/W	R/W
Reset							0	0

Bits 0, 1 – Lx Listen x

Value	Description
1	Listen to corresponding link.

35.8.86 Spw Tch Config Sender

Name: SPW_TCH_CFGSEND
Offset: 0x0EA4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							S1	S0
Access							R/W	R/W
Reset							0	0

Bits 0, 1 – Sx Send x

Value	Description
1	Send to corresponding link.

35.8.87 Spw Tch Config

Name: SPW_TCH_CFG
Offset: 0x0EA8
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			EVENT[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bits 5:0 – EVENT[5:0] Event
Event source for Master mode.

35.8.88 Spw Tch Config Restart

Name: SPW_TCH_CFGRESTART
Offset: 0x0EAC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	ONESHOT	PPS						EVENT
Access	R/W	R/W						R/W
Reset	0	0						0

Bit	7	6	5	4	3	2	1	0
			VALUE[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 15 – ONESHOT One Shot

Value	Description
1	Clears this register after restart event.

Bit 14 – PPS Pps

Value	Description
1	Uses Pps input as event source for restart.

Bit 8 – EVENT Event

Value	Description
1	Uses Event input as event source for restart.

Bits 5:0 – VALUE[5:0] Value
Time Code value to set at restart event.

35.8.89 Spw Tch Config Tc Event

Name: SPW_TCH_CFGTCEVENT
Offset: 0x0EB0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
			MASK[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
			VALUE[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bits 13:8 – MASK[5:0] Mask
Bits of new Time Code to check.

Bits 5:0 – VALUE[5:0] Value
Value used to check the new Time Code.

35.8.90 Spw Tch Config Watchdog

Name: SPW_TCH_CFGWD
Offset: 0x0EB4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	EARLY[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EARLY[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	LATE[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LATE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:16 – EARLY[15:0] Early
N: Time code before N ticks triggers watchdog.

Value	Description
0	Early watchdog disabled.

Bits 15:0 – LATE[15:0] Late
N: No Time code before N ticks triggers watchdog.

Value	Description
0	Late watchdog disabled.

35.8.91 Spw Tch Last Time Code

Name: SPW_TCH_LASTTIMECODE
Offset: 0x0EB8
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								SEND
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
			VALUE[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 8 – SEND Send Now

Value	Description
0	Just change value when written.
1	Send Time Code value when written.

Bits 5:0 – VALUE[5:0] Value
Last Time Code distributed.

35.8.92 Spw Tch Software Reset

Name: SPW_TCH_SWRESET
Offset: 0x0EBC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	PATTERN[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PATTERN[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PATTERN[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PATTERN[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – PATTERN[31:0] Pattern
4D61_6A6F: Arm reset.
7254_6F6D: Trigger reset, if armed.
Others: Clear Arm

35.8.93 SpW Group Interrupt Status 1

Name: SPW_GROUP_IRQSTS1
Offset: 0x0F00
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
								TCH
Access								
Reset								0

Bit	15	14	13	12	11	10	9	8
	RX1							
Access	R							
Reset	0							

Bit	7	6	5	4	3	2	1	0
	TX1							
Access	R							
Reset	0							

Bit 16 – TCH Time Code Handler

Value	Description
1	Time Code Handler

Bit 15 – RX1

Value	Description
1	RXx

Bit 7 – TX1

Value	Description
1	TXx

35.8.94 SpW Group Interrupt Status 2

Name: SPW_GROUP_IRQSTS2
Offset: 0x0F04
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	Di1	Dia1	Link1	Di2	Dia2	Link2		
Access	R	R	R	R	R	R		
Reset	0	0	0	0	0	0		

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 23 – Di1 Distributed Interrupt 1

Value	Description
1	Distributed Interrupt, Link 1.

Bit 22 – Dia1 Distributed Interrupt Acknowledge, Link 1

Value	Description
1	Distributed Interrupt Acknowledge, Link 1.

Bit 21 – Link1 Link 1

Value	Description
1	Link handler, Link 1

Bit 20 – Di2 Distributed Interrupt 2

Value	Description
1	Distributed Interrupt, Link 2.

Bit 19 – Dia2 Distributed Interrupt Acknowledge, Link 2

Value	Description
1	Distributed Interrupt Acknowledge, Link 2.

Bit 18 – Link2 Link 2

Value	Description
1	Link handler, Link 2.

35.8.95 SpW Group EDAC Status

Name: SPW_GROUP_EDACSTS
Offset: 0x0F0C
Reset: 0x00000000
Property: Read-only

Note: This register is cleared on read.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	UNCORR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CORR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:8 – UNCORR[7:0] Uncorrectable Error
An uncorrectable error has been detected.

Bits 7:0 – CORR[7:0] Correction Count
Number of EDAC corrections made. Saturates at max.

36. Ethernet MAC (GMAC)

36.1 Description

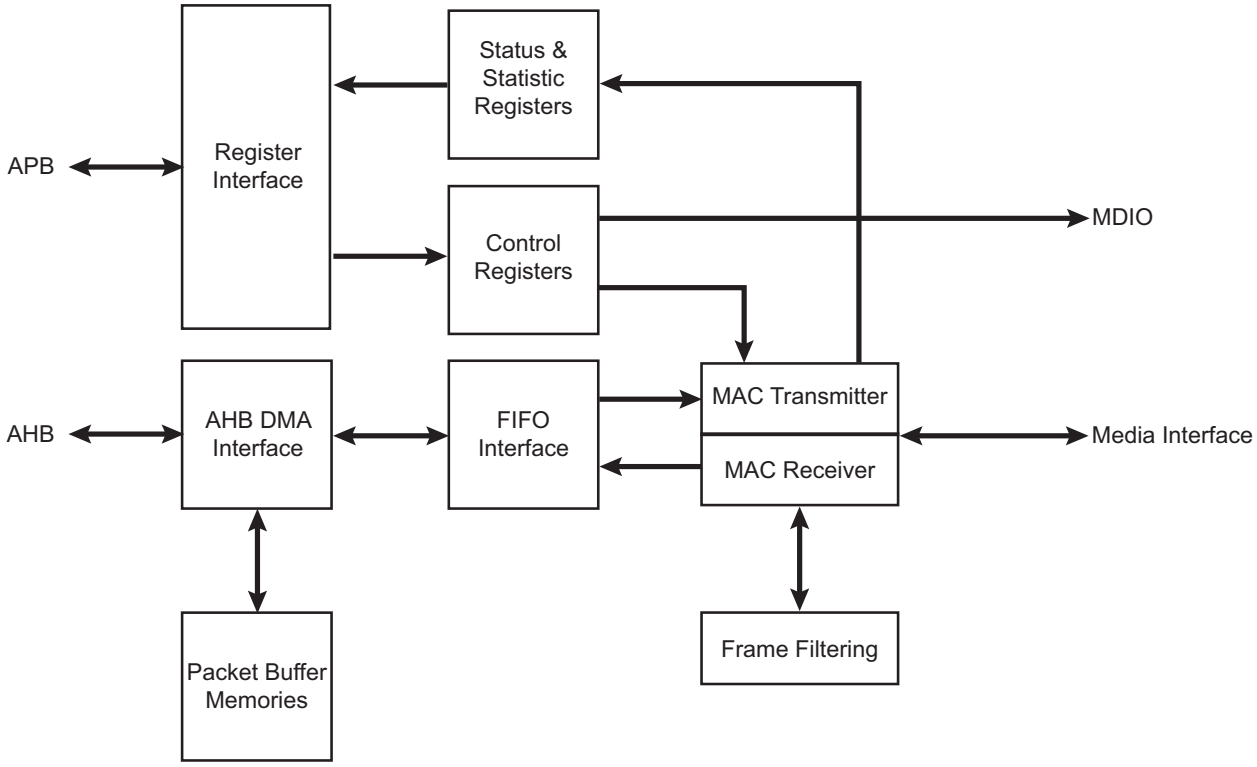
The Ethernet MAC (GMAC) module implements a 10/100 Mbps Ethernet MAC compatible with the IEEE 802.3 standard. The GMAC can operate in either half or full duplex mode at all supported speeds. The GMAC Network Configuration Register is used to select the speed, duplex mode and interface type (MII, RMII).

36.2 Embedded Characteristics

- Compatible with IEEE Standard 802.3
- 10, 100 Mbps Operation
- Full and Half Duplex Operation at all Supported Speeds of Operation
- Statistics Counter Registers for RMON/MIB
- MII/RMII Interface to the Physical Layer
- Integrated Physical Coding
- Direct Memory Access (DMA) Interface to External Memory
- Support for 6 Priority Queues
- 8 Kbytes Transmit RAM and 4 Kbytes Receive RAM (refer to Table 36-4 for queue-specific sizes)
- Programmable Burst Length and Endianism for DMA
- Interrupt Generation to Signal Receive and Transmit Completion, Errors or Other Events
- Automatic Pad and Cyclic Redundancy Check (CRC) Generation on Transmitted Frames
- Automatic Discard of Frames Received with Errors
- Receive and Transmit IP, TCP and UDP Checksum Offload. Both IPv4 and IPv6 Packet Types Supported
- Address Checking Logic for Four Specific 48-bit Addresses, Four Type IDs, Promiscuous Mode, Hash Matching of Unicast and Multicast Destination Addresses and Wake-on-LAN
- Management Data Input/Output (MDIO) Interface for Physical Layer Management
- Support for Jumbo Frames up to 10240 Bytes
- Full Duplex Flow Control with Recognition of Incoming Pause Frames and Hardware Generation of Transmitted Pause Frames
- Half Duplex Flow Control by Forcing Collisions on Incoming Frames
- Support for 802.1Q VLAN Tagging with Recognition of Incoming VLAN and Priority Tagged Frames
- Support for 802.1Qbb Priority-based Flow Control
- Programmable Inter Packet Gap (IPG) Stretch
- Recognition of IEEE 1588 PTP Frames
- IEEE 1588 Timestamp Unit (TSU)
- Support for 802.1AS Timing and Synchronization
- Supports 802.1Qav Traffic Shaping on Two Highest Priority Queues

36.3 Block Diagram

Figure 36-1. Block Diagram



36.4 Signal Interfaces

The GMAC includes the following signal interfaces:

- MII, RMII to an external PHY
- MDIO interface for external PHY management
- Slave APB interface for accessing GMAC registers
- Master AHB interface for memory access
- GTSUCOMP signal for TSU timer count value comparison

Table 36-1. GMAC Connections in Different Modes

Signal Name	Function	MII	RMII
GTXCK[1]	Transmit Clock or Reference Clock	TXCK	REFCK
GTXEN	Transmit Enable	TXEN	TXEN
GTX[3..0]	Transmit Data	TXD[3:0]	TXD[1:0]
GTXER	Transmit Coding Error	TXER	Not Used
GRXCK	Receive Clock	RXCK	Not Used
GRXDV	Receive Data Valid	RXDV	CRSDV
GRX[3..0]	Receive Data	RXD[3:0]	RXD[1:0]
GRXER	Receive Error	RXER	RXER

.....continued

Signal Name	Function	MII	RMII
GCRS	Carrier Sense and Data Valid	CRS	Not Used
GCOL	Collision Detect	COL	Not Used
GMDC	Management Data Clock	MDC	MDC
GMDIO	Management Data Input/Output	MDIO	MDIO

Note:

1. Input only. GTXCK must be provided with a 25 MHz/50 MHz external crystal oscillator for MII / RMII interfaces, respectively.

36.5 Product Dependencies

36.5.1 I/O Lines

The pins used for interfacing the GMAC may be multiplexed with PIO lines. The programmer must first program the PIO Controller to assign the pins to their peripheral function. If I/O lines of the GMAC are not used by the application, they can be used for other purposes by the PIO Controller.

36.5.2 Power Management

The GMAC is not continuously clocked. The user must first enable the GMAC clock in the Power Management Controller before using it.

36.5.3 Interrupt Sources

The GMAC interrupt line is connected to one of the internal sources of the interrupt controller. Using the GMAC interrupt requires prior programming of the interrupt controller.

The GMAC features 6 interrupt sources. Refer to the table “Peripheral Identifiers” in the section “Peripherals” for the interrupt numbers for GMAC priority queues.

36.6 Functional Description

36.6.1 Media Access Controller

The Media Access Controller (MAC) transmit block takes data from FIFO, adds preamble and, if necessary, pad and frame check sequence (FCS). Both half duplex and full duplex Ethernet modes of operation are supported. When operating in half duplex mode, the MAC transmit block generates data according to the carrier sense multiple access with collision detect (CSMA/CD) protocol. The start of transmission is deferred if carrier sense (CRS) is active. If collision (COL) becomes active during transmission, a jam sequence is asserted and the transmission is retried after a random backoff. The CRS and COL signals have no effect in full duplex mode.

The MAC receive block checks for valid preamble, FCS, alignment and length, and presents received frames to the MAC address checking block and FIFO. Software can configure the GMAC to receive jumbo frames up to 10240 bytes. It can optionally strip CRC from the received frame prior to transfer to FIFO.

The address checker recognizes four specific 48-bit addresses, can recognize four different type ID values, and contains a 64-bit Hash register for matching multicast and unicast addresses as required. It can recognize the broadcast address of all ones and copy all frames. The MAC can also reject all frames that are not VLAN tagged and recognize Wake on LAN events.

The MAC receive block supports offloading of IP, TCP and UDP checksum calculations (both IPv4 and IPv6 packet types supported), and can automatically discard bad checksum frames.

36.6.2 1588 Timestamp Unit

The 1588 timestamp unit (TSU) is implemented as a 94-bit timer.

The 48 upper bits [93:46] of the timer count seconds and are accessible in the GMAC 1588 Timer Seconds High Register (GMAC_TSH) and GMAC 1588 Timer Seconds Low Register (GMAC_TSL). The 30 lower bits [45:16] of the timer count nanoseconds and are accessible in the GMAC 1588 Timer Nanoseconds Register (GMAC_TN). The lowest 16 bits [15:0] of the timer count sub-nanoseconds.

The 46 lower bits roll over when they have counted to one second. The timer increments by a programmable period (to approximately 15.2 femtoseconds resolution) with each MCK period and can also be adjusted in 1ns resolution (incremented or decremented) through APB register accesses.

36.6.3 AHB Direct Memory Access Interface

The GMAC DMA controller is connected to the MAC FIFO interface and provides a scatter-gather type capability for packet data storage.

The DMA implements packet buffering where dual-port memories are used to buffer multiple frames.

36.6.3.1 Packet Buffer DMA

- Easier to guarantee maximum line rate due to the ability to store multiple frames in the packet buffer, where the number of frames is limited by the amount of packet buffer memory and Ethernet frame size
- Full store and forward, or partial store and forward programmable options (partial store will cater for shorter latency requirements)
- Support for Transmit TCP/IP checksum offload
- Support for priority queueing
- When a collision on the line occurs during transmission, the packet will be automatically replayed directly from the packet buffer memory rather than having to re-fetch through the AHB (full store and forward ONLY)
- Received error packets are automatically dropped before any of the packet is presented to the AHB (full store and forward ONLY), thus reducing AHB activity
- Supports manual RX packet flush capabilities
- Optional RX packet flush when there is lack of AHB resource

36.6.3.2 Partial Store and Forward Using Packet Buffer DMA

The DMA uses SRAM-based packet buffers, and can be programmed into a low latency mode, known as Partial Store and Forward. This allows for a reduced latency as the full packet is not buffered before forwarding. Note that this option is only available when the device is configured for full duplex operation.

This feature is enabled via the programmable TX and RX Partial Store and Forward registers. When the transmit Partial Store and Forward mode is activated, the transmitter will only begin to forward the packet to the MAC when there is enough packet data stored in the packet buffer. Likewise, when the receive Partial Store and Forward mode is activated, the receiver will only begin to forward the packet to the AHB when enough packet data is stored in the packet buffer. The amount of packet data required to activate the forwarding process is programmable via watermark registers which are located at the same address as the partial store and forward enable bits.

Note that the minimum operational value for the TX partial store and forward watermark is 20. There is no operational limit for the RX partial store and forward watermark. Enabling partial store and forward is a useful means to reduce latency, but there are performance implications.

The GMAC DMA uses separate transmit and receive lists of buffer descriptors, with each descriptor describing a buffer area in memory. This allows Ethernet packets to be broken up and scattered around the AHB memory space.

36.6.3.3 Receive AHB Buffers

Received frames, optionally including FCS, are written to receive AHB buffers stored in memory. The receive buffer depth is programmable in the range of 64 bytes to 16 Kbytes through the DMA Configuration register, with the default being 128 bytes.

The start location for each receive AHB buffer is stored in memory in a list of receive buffer descriptors at an address location pointed to by the receive buffer queue pointer. The base address for the receive buffer queue pointer is configured in software using the Receive Buffer Queue Base Address register.

Each list entry consists of two words. The first is the address of the receive AHB buffer and the second the receive status. If the length of a receive frame exceeds the AHB buffer length, the status word for the used buffer is written with zeroes except for the “start of frame” bit, which is always set for the first buffer in a frame. Bit zero of the address field is written to 1 to show the buffer has been used. The receive buffer manager then reads the location of the next

receive AHB buffer and fills that with the next part of the received frame data. AHB buffers are filled until the frame is complete and the final buffer descriptor status word contains the complete frame status. Refer to the table below for details of the receive buffer descriptor list.

Each receive AHB buffer start location is a word address. The start of the first AHB buffer in a frame can be offset by up to three bytes, depending on the value written to bits 14 and 15 of the Network Configuration register. If the start location of the AHB buffer is offset, the available length of the first AHB buffer is reduced by the corresponding number of bytes.

Table 36-2. Receive Buffer Descriptor Entry

Bit	Function
Word 0	
31:2	Address of beginning of buffer
1	Wrap—marks last descriptor in receive buffer descriptor list.
0	Ownership—needs to be zero for the GMAC to write data to the receive buffer. The GMAC sets this to one once it has successfully written a frame to memory. Software has to clear this bit before the buffer can be used again.
Word 1	
31	Global all ones broadcast address detected
30	Multicast hash match
29	Unicast hash match
28	–
27	Specific Address Register match found, bit 25 and bit 26 indicate which Specific Address Register causes the match.
26:25	Specific Address Register match. Encoded as follows: 00: Specific Address Register 1 match 01: Specific Address Register 2 match 10: Specific Address Register 3 match 11: Specific Address Register 4 match If more than one specific address is matched only one is indicated with priority 4 down to 1.
24	This bit has a different meaning depending on whether RX checksum offloading is enabled. **With RX checksum offloading disabled:** (bit 24 clear in Network Configuration Register) Type ID register match found, bit 22 and bit 23 indicate which type ID register causes the match. **With RX checksum offloading enabled:** (bit 24 set in Network Configuration Register) 0: The frame was not SNAP encoded and/or had a VLAN tag with the Canonical Format Indicator (CFI) bit set. 1: The frame was SNAP encoded and had either no VLAN tag or a VLAN tag with the CFI bit not set.

.....continued

Bit	Function
23:22	This bit has a different meaning depending on whether RX checksum offloading is enabled. **With RX checksum offloading disabled:** (bit 24 clear in Network Configuration) Type ID register match. Encoded as follows: 00: Type ID register 1 match 01: Type ID register 2 match 10: Type ID register 3 match 11: Type ID register 4 match If more than one Type ID is matched only one is indicated with priority 4 down to 1. **With RX checksum offloading enabled:** (bit 24 set in Network Configuration Register) 00: Neither the IP header checksum nor the TCP/UDP checksum was checked. 01: The IP header checksum was checked and was correct. Neither the TCP nor UDP checksum was checked. 10: Both the IP header and TCP checksum were checked and were correct. 11: Both the IP header and UDP checksum were checked and were correct.
21	VLAN tag detected—type ID of 0x8100. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a type ID of 0x8100
20	Priority tag detected—type ID of 0x8100 and null VLAN identifier. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a type ID of 0x8100 and a null VLAN identifier.
19:17	VLAN priority—only valid if bit 21 is set.
16	Canonical format indicator (CFI) bit (only valid if bit 21 is set).
15	End of frame—when set the buffer contains the end of a frame. If end of frame is not set, then the only valid status bit is start of frame (bit 14).
14	Start of frame—when set the buffer contains the start of a frame. If both bits 15 and 14 are set, the buffer contains a whole frame.
13	This bit has a different meaning depending on whether jumbo frames and ignore FCS modes are enabled. If neither mode is enabled this bit will be zero. **With jumbo frame mode enabled:** (bit 3 set in Network Configuration Register) Additional bit for length of frame (bit[13]), that is concatenated with bits[12:0] **With ignore FCS mode enabled and jumbo frames disabled:** (bit 26 set in Network Configuration Register and bit 3 clear in Network Configuration Register) This indicates per frame FCS status as follows: 0: Frame had good FCS 1: Frame had bad FCS, but was copied to memory as ignore FCS enabled.
12:0	These bits represent the length of the received frame which may or may not include FCS depending on whether FCS discard mode is enabled. **With FCS discard mode disabled:** (bit 17 clear in Network Configuration Register) Least significant 12 bits for length of frame including FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit[13] of the descriptor above. **With FCS discard mode enabled:** (bit 17 set in Network Configuration Register) Least significant 12 bits for length of frame excluding FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit[13] of the descriptor above.

To receive frames, the AHB buffer descriptors must be initialized by writing an appropriate address to bits 31:2 in the first word of each list entry. Bit 0 must be written with zero. Bit 1 is the wrap bit and indicates the last entry in the buffer descriptor list.

The start location of the receive buffer descriptor list must be written with the receive buffer queue base address before reception is enabled (receive enable in the Network Control register). Once reception is enabled, any writes to the Receive Buffer Queue Base Address register are ignored. When read, it will return the current pointer position in the descriptor list, though this is only valid and stable when receive is disabled.

If the filter block indicates that a frame should be copied to memory, the receive data DMA operation starts writing data into the receive buffer. If an error occurs, the buffer is recovered.

An internal counter within the GMAC represents the receive buffer queue pointer and it is not visible through the CPU interface. The receive buffer queue pointer increments by two words after each buffer has been used. It re-initializes to the receive buffer queue base address if any descriptor has its wrap bit set.

As receive AHB buffers are used, the receive AHB buffer manager sets bit zero of the first word of the descriptor to logic one indicating the AHB buffer has been used.

Software should search through the “used” bits in the AHB buffer descriptors to find out how many frames have been received, checking the start of frame and end of frame bits.

When the DMA is configured in the packet buffer Partial Store And Forward mode, received frames are written out to the AHB buffers as soon as enough frame data exists in the packet buffer. For both cases, this may mean several full AHB buffers are used before some error conditions can be detected. If a receive error is detected the receive buffer currently being written will be recovered. Previous buffers will not be recovered. As an example, when receiving frames with cyclic redundancy check (CRC) errors or excessive length, it is possible that a frame fragment might be stored in a sequence of AHB receive buffers. Software can detect this by looking for start of frame bit set in a buffer following a buffer with no end of frame bit set.

To function properly, a 10/100/1000 Ethernet system should have no excessive length frames or frames greater than 128 bytes with CRC errors. Collision fragments will be less than 128 bytes long, therefore it will be a rare occurrence to find a frame fragment in a receive AHB buffer, when using the default value of 128 bytes for the receive buffers size.

When in packet buffer full store and forward mode, only good received frames are written out of the DMA, so no fragments will exist in the AHB buffers due to MAC receiver errors. There is still the possibility of fragments due to DMA errors, for example used bit read on the second buffer of a multi-buffer frame.

If bit zero of the receive buffer descriptor is already set when the receive buffer manager reads the location of the receive AHB buffer, then the buffer has been already used and cannot be used again until software has processed the frame and cleared bit zero. In this case, the “buffer not available” bit in the Receive Status register is set and an interrupt triggered. The Receive Resource Error statistics register is also incremented.

When the DMA is configured in the packet buffer full store and forward mode, the user can optionally select whether received frames should be automatically discarded when no AHB buffer resource is available. This feature is selected via bit 24 of the DMA Configuration register (by default, the received frames are not automatically discarded). If this feature is off, then received packets will remain to be stored in the SRAM-based packet buffer until AHB buffer resource next becomes available. This may lead to an eventual packet buffer overflow if packets continue to be received when bit zero (used bit) of the receive buffer descriptor remains set. Note that after a used bit has been read, the receive buffer manager will re-read the location of the receive buffer descriptor every time a new packet is received. When the DMA is not configured in the packet buffer full store and forward mode and a used bit is read, the frame currently being received will be automatically discarded.

When the DMA is configured in the packet buffer full store and forward mode, a receive overrun condition occurs when the receive SRAM-based packet buffer is full, or because HRESP was not OK. In all other modes, a receive overrun condition occurs when either the AHB bus was not granted quickly enough, or because HRESP was not OK, or because a new frame has been detected by the receive block, but the status update or write back for the previous frame has not yet finished. For a receive overrun condition, the receive overrun interrupt is asserted and the buffer currently being written is recovered. The next frame that is received whose address is recognized reuses the buffer.

In any packet buffer mode, a write to bit 18 of GMAC_NCR will force a packet from the external SRAM-based receive packet buffer to be flushed. This feature is only acted upon when the RX DMA is not currently writing packet data out to AHB, i.e., it is in an IDLE state. If the RX DMA is active, a write to this bit is ignored.

36.6.3.4 Transmit AHB Buffers

Frames to transmit are stored in one or more transmit AHB buffers. Transmit frames can be between 1 and 16384 bytes long, so it is possible to transmit frames longer than the maximum length specified in the IEEE 802.3 standard. It should be noted that zero length AHB buffers are allowed and that the maximum number of buffers permitted for each transmit frame is 128.

The start location for each transmit AHB buffer is stored in memory in a list of transmit buffer descriptors at a location pointed to by the transmit buffer queue pointer. The base address for this queue pointer is set in software using the Transmit Buffer Queue Base Address register. Each list entry consists of two words. The first is the byte address of the transmit buffer and the second containing the transmit control and status. For the packet buffer DMA, the start location for each AHB buffer is a byte address, the bottom bits of the address being used to offset the start of the data from the data-word boundary (i.e., bits 2,1 and 0 are used to offset the address for 64-bit datapaths).

Frames can be transmitted with or without automatic CRC generation. If CRC is automatically generated, pad will also be automatically generated to take frames to a minimum length of 64 bytes. When CRC is not automatically generated (as defined in word 1 of the transmit buffer descriptor), the frame is assumed to be at least 64 bytes long and pad is not generated.

An entry in the transmit buffer descriptor list is described in the table below.

To transmit frames, the buffer descriptors must be initialized by writing an appropriate byte address to bits [31:0] in the first word of each descriptor list entry.

The second word of the transmit buffer descriptor is initialized with control information that indicates the length of the frame, whether or not the MAC is to append CRC and whether the buffer is the last buffer in the frame.

After transmission the status bits are written back to the second word of the first buffer along with the used bit. Bit 31 is the used bit which must be zero when the control word is read if transmission is to take place. It is written to one once the frame has been transmitted. Bits[29:20] indicate various transmit error conditions. Bit 30 is the wrap bit which can be set for any buffer within a frame. If no wrap bit is encountered the queue pointer continues to increment.

The Transmit Buffer Queue Base Address register can only be updated while transmission is disabled or halted; otherwise any attempted write will be ignored. When transmission is halted the transmit buffer queue pointer will maintain its value. Therefore when transmission is restarted the next descriptor read from the queue will be from immediately after the last successfully transmitted frame. while transmit is disabled (bit 3 of the Network Control register set low), the transmit buffer queue pointer resets to point to the address indicated by the Transmit Buffer Queue Base Address register. Note that disabling receive does not have the same effect on the receive buffer queue pointer.

Once the transmit queue is initialized, transmit is activated by writing to the transmit start bit (bit 9) of the Network Control register. Transmit is halted when a buffer descriptor with its used bit set is read, a transmit error occurs, or by writing to the transmit halt bit of the Network Control register. Transmission is suspended if a pause frame is received while the pause enable bit is set in the Network Configuration register. Rewriting the start bit while transmission is active is allowed. This is implemented with TXGO variable which is readable in the Transmit Status register at bit location 3. The TXGO variable is reset when:

- Transmit is disabled.
- A buffer descriptor with its ownership bit set is read.
- Bit 10, THALT, of the Network Control register is written.
- There is a transmit error such as too many retries or a transmit underrun.

To set TXGO, write TSTART to the bit 9 of the Network Control register. Transmit halt does not take effect until any ongoing transmit finishes.

If the DMA is configured for packet buffer Partial Store and Forward mode and a collision occurs during transmission of a multi-buffer frame, transmission will automatically restart from the first buffer of the frame. For packet buffer mode, the entire contents of the frame are read into the transmit packet buffer memory, so the retry attempt will be replayed directly from the packet buffer memory rather than having to re-fetch through the AHB.

If a used bit is read midway through transmission of a multi-buffer frame, this is treated as a transmit error. Transmission stops, GTXER is asserted and the FCS will be bad.

If transmission stops due to a transmit error or a used bit being read, transmission restarts from the first buffer descriptor of the frame being transmitted when the transmit start bit is rewritten.

Table 36-3. Transmit Buffer Descriptor Entry

Bit	Function
Word 0	
31:0	Byte address of buffer
Word 1	
31	Used—must be zero for the GMAC to read data to the transmit buffer. The GMAC sets this to one for the first buffer of a frame once it has been successfully transmitted. Software must clear this bit before the buffer can be used again.
30	Wrap—marks last descriptor in transmit buffer descriptor list. This can be set for any buffer within the frame.
29	Retry limit exceeded, transmit error detected
28	Reserved.
27	Transmit frame corruption due to AHB error—set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and GTXER asserted). Also set if single frame is too large for configured packet buffer memory size.
26	Late collision, transmit error detected.
25:23	Reserved
22:20	Transmit IP/TCP/UDP checksum generation offload errors: 000: No Error. 001: The Packet was identified as a VLAN type, but the header was not fully complete, or had an error in it. 010: The Packet was identified as a SNAP type, but the header was not fully complete, or had an error in it. 011: The Packet was not of an IP type, or the IP packet was invalidly short, or the IP was not of type IPv4/IPv6. 100: The Packet was not identified as VLAN, SNAP or IP. 101: Non supported packet fragmentation occurred. For IPv4 packets, the IP checksum was generated and inserted. 110: Packet type detected was not TCP or UDP. TCP/UDP checksum was therefore not generated. For IPv4 packets, the IP checksum was generated and inserted. 111: A premature end of packet was detected and the TCP/UDP checksum could not be generated.
19:17	Reserved
16	No CRC to be appended by MAC. When set, this implies that the data in the buffers already contains a valid CRC, hence no CRC or padding is to be appended to the current frame by the MAC. This control bit must be set for the first buffer in a frame and will be ignored for the subsequent buffers of a frame. Note that this bit must be clear when using the transmit IP/TCP/UDP checksum generation offload, otherwise checksum generation and substitution will not occur.
15	Last buffer, when set this bit will indicate the last buffer in the current frame has been reached.
14	Reserved
13:0	Length of buffer

36.6.3.5 DMA Bursting on the AHB

The DMA will always use SINGLE, or INCR type AHB accesses for buffer management operations. When performing data transfers, the AHB burst length used can be programmed using bits 4:0 of the DMA Configuration register so that either SINGLE, INCR or fixed length incrementing bursts (INCR4, INCR8 or INCR16) are used where possible.

When there is enough space and enough data to be transferred, the programmed fixed length bursts will be used. If there is not enough data or space available, for example when at the beginning or the end of a buffer, SINGLE type accesses are used. Also SINGLE type accesses are used at 1024 byte boundaries, so that the 1 Kbyte boundaries are not burst over as per AHB requirements.

The DMA will not terminate a fixed length burst early, unless an error condition occurs on the AHB or if receive or transmit are disabled in the Network Control register.

36.6.3.6 DMA Packet Buffer

The DMA uses packet buffers for both transmit and receive paths. This mode allows multiple packets to be buffered in both transmit and receive directions. This allows the DMA to withstand far greater access latencies on the AHB and make more efficient use of the AHB bandwidth. There are two modes of operation—Full Store and Forward and Partial Store and Forward.

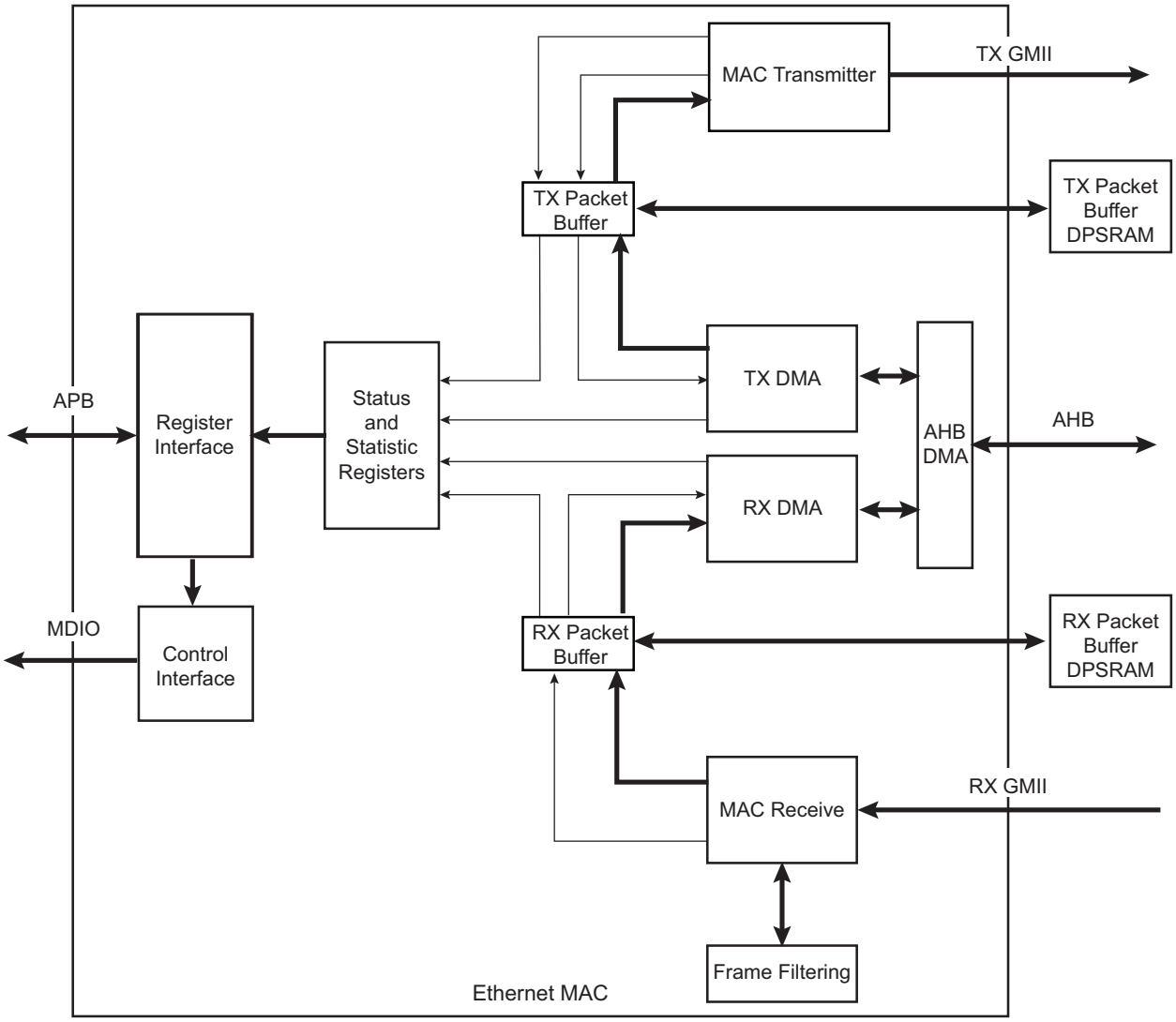
As described in section Partial Store and Forward Using Packet Buffer DMA, the DMA can be programmed into a low latency mode, known as Partial Store and Forward.

When the DMA is in full store and forward mode, full packets are buffered which provides the possibility to:

- Discard packets with error on the receive path before they are partially written out of the DMA, thus saving AHB bus bandwidth and driver processing overhead,
- Retry collided transmit frames from the buffer, thus saving AHB bus bandwidth,
- Implement transmit IP/TCP/UDP checksum generation offload.

With the packet buffers included, the structure of the GMAC data paths is shown in the figure below.

Figure 36-2. Data Paths with Packet Buffers Included



36.6.3.7 Transmit Packet Buffer

The transmitter packet buffer will continue attempting to fetch frame data from the AHB system memory until the packet buffer itself is full, at which point it will attempt to maintain its full level.

To accommodate the status and statistics associated with each frame, three words per packet are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet. Storing the status in the DPRAM is required in order to decouple the DMA interface of the buffer from the MAC interface, to update the MAC status/statistics and to generate interrupts in the order in which the packets that they represent were fetched from the AHB memory.

If any errors occur on the AHB while reading the transmit frame, the fetching of packet data from AHB memory is halted. The MAC transmitter will continue to fetch packet data, thereby emptying the packet buffer and allowing any good non-errored frames to be transmitted successfully. Once these have been fully transmitted, the status/statistics for the errored frame will be updated and software will be informed via an interrupt that an AHB error occurred. This way, the error is reported in the correct packet order.

The transmit packet buffer will only attempt to read more frame data from the AHB when space is available in the packet buffer memory. If space is not available it must wait until the a packet fetched by the MAC completes transmission and is subsequently removed from the packet buffer memory. Note that if full store and forward mode is active and if a single frame is fetched that is too large for the packet buffer memory, the frame is flushed and the DMA halted with an error status. This is because a complete frame must be written into the packet buffer before

transmission can begin, and therefore the minimum packet buffer memory size should be chosen to satisfy the maximum frame to be transmitted in the application.

In full store and forward mode, once the complete transmit frame is written into the packet buffer memory, a trigger is sent across to the MAC transmitter, which will then begin reading the frame from the packet buffer memory. Since the whole frame is present and stable in the packet buffer memory an underflow of the transmitter is not possible. The frame is kept in the packet buffer until notification is received from the MAC that the frame data has either been successfully transmitted or can no longer be retransmitted (too many retries in half duplex mode). When this notification is received the frame is flushed from memory to make room for a new frame to be fetched from AHB system memory.

In Partial Store and Forward mode, a trigger is sent across to the MAC transmitter as soon as sufficient packet data is available, which will then begin fetching the frame from the packet buffer memory. If, after this point, the MAC transmitter is able to fetch data from the packet buffer faster than the AHB DMA can fill it, an underflow of the transmitter is possible. In this case, the transmission is terminated early, and the packet buffer is completely flushed. Transmission can only be restarted by writing to the transmit START bit.

In half duplex mode, the frame is kept in the packet buffer until notification is received from the MAC that the frame data has either been successfully transmitted or can no longer be retransmitted (too many retries in half duplex mode). When this notification is received the frame is flushed from memory to make room for a new frame to be fetched from AHB system memory.

In full duplex mode, the frame is removed from the packet buffer on the fly.

Other than underflow, the only MAC related errors that can occur are due to collisions during half duplex transmissions. When a collision occurs the frame still exists in the packet buffer memory so can be retried directly from there. Only once the MAC transmitter has failed to transmit after sixteen attempts is the frame finally flushed from the packet buffer.

36.6.3.8 Receive Packet Buffer

The receive packet buffer stores frames from the MAC receiver along with their status and statistics. Frames with errors are flushed from the packet buffer memory, while good frames are pushed onto the DMA AHB interface.

The receiver packet buffer monitors the FIFO write interface from the MAC receiver and translates the FIFO pushes into packet buffer writes. At the end of the received frame the status and statistics are buffered so that the information can be used when the frame is read out. When programmed in full store and forward mode, if the frame has an error the frame data is immediately flushed from the packet buffer memory allowing subsequent frames to utilise the freed up space. The status and statistics for bad frames are still used to update the GMAC registers.

To accommodate the status and statistics associated with each frame, three words per packet (or two if configured in 64-bit datapath mode) are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet.

The receiver packet buffer will also detect a full condition so that an overflow condition can be detected. If this occurs, subsequent packets are dropped and an RX overflow interrupt is raised.

For full store and forward, the DMA only begins packet fetches once the status and statistics for a frame are available. If the frame has a bad status due to a frame error, the status and statistics are passed on to the GMAC registers. If the frame has a good status, the information is used to read the frame from the packet buffer memory and burst onto the AHB using the DMA buffer management protocol. Once the last frame data has been transferred to the packet buffer, the status and statistics are updated to the GMAC registers.

If Partial Store and Forward mode is active, the DMA will begin fetching the packet data before the status is available. As soon as the status becomes available, the DMA will fetch this information as soon as possible before continuing to fetch the remainder of the frame. Once the last frame data has been transferred to the packet buffer, the status and statistics are updated to the GMAC registers.

36.6.3.9 Priority Queueing in the DMA

The DMA by default uses a single transmit and receive queue. This means the list of transmit/receive buffer descriptors point to data buffers associated with a single transmit/receive data stream. The GMAC can select up to 6 priority queues. Each queue has an independent list of buffer descriptors pointing to separate data streams.

The table below gives the DPRAM size associated with each queue:

Table 36-4. Queue Size

Queue Number	Queue Size
5 (highest priority)	1 KB
4	2 KB
3	2 KB
2	512 bytes
1	512 bytes
0 (lowest priority)	2 KB

In the transmit direction, higher priority queues are always serviced before lower priority queues, with Q0 as lowest priority and Q5 as highest priority. This strict priority scheme requires the user to ensure that high priority traffic is constrained so that lower priority traffic will have required bandwidth. The GMAC DMA will determine the next queue to service by initiating a sequence of buffer descriptor reads interrogating the ownership bits of each. The buffer descriptor corresponding to the highest priority queue is read first. As an example, if the ownership bit of this descriptor is set, then the DMA will progress to reading the 2nd highest priority queue's descriptor. If that ownership bit read of this lower priority queue is set, then the DMA will read the 3rd highest priority queue's descriptor. If all the descriptors return an ownership bit set, then a resource error has occurred, an interrupt is generated and transmission is automatically halted. Transmission can only be restarted by setting the START bit in the Network Control register. The GMAC DMA will need to identify the highest available queue to transmit from when the START bit in the Network Control register is written to and the TX is in a halted state, or when the last word of any packet has been fetched from external AHB memory.

The GMAC transmit DMA maximizes the effectiveness of priority queuing by ensuring that high priority traffic be transmitted as early as possible after being fetched from AHB. High priority traffic fetched from AHB will be pushed to the MAC layer, depending on traffic shaping being enabled and the associated credit value for that queue, before any lower priority traffic that may pre-exist in the transmit SRAM-based packet buffer. This is achieved by separating the transmit SRAM-based packet buffer into regions, one region per queue. The size of each region determines the amount of SRAM space allocated per queue.

For each queue, there is an associated Transmit Buffer Queue Base Address register. For the lowest priority queue (or the only queue when only one queue is selected), the Transmit Buffer Queue Base Address is located at address 0x1C. For all other queues, the Transmit Buffer Queue Base Address registers are located at sequential addresses starting at address 0x440.

In the receive direction each packet is written to AHB data buffers in the order that it is received. For each queue, there is an independent set of receive AHB buffers for each queue. There is therefore a separate Receive Buffer Queue Base Address register for each queue. For the lowest priority queue (or the only queue when only one queue is selected), the Receive Buffer Queue Base Address is located at address 0x18. For all other queues, the Receive Buffer Queue Base Address registers are located at sequential addresses starting at address 0x480. Every received packet will pass through a programmable screening algorithm which will allocate a particular queue to that frame. The user interface to the screeners is through two types of programmable registers:

- Screening Type 1 registers—The module features 4 Screening Type 1 registers. Screening Type 1 registers hold values to match against specific IP and UDP fields of the received frames. The fields matched against are DS (Differentiated Services field of IPv4 frames), TC (Traffic class field of IPv6 frames) and/or the UDP destination port.
- Screening Type 2 registers—The module features 8 Screening Type 2 registers GMAC_ST2RPQ. Screening Type 2 registers operate independently of Screening Type 1 registers and offer additional match capabilities. Screening Type 2 allows a screen to be configured that is the combination of all or any of the following comparisons:

1. An enable bit VLAN priority, VLANE. A VLAN priority match will be performed if the VLAN priority enable is set. The extracted priority field in the VLAN header is compared against VLANP in the GMAC_ST2RPQ register itself.

2. An enable bit EtherType, ETHE. The EtherType field I2ETH inside GMAC_ST2RPQ maps to one of 4 EtherType match registers, GMAC_ST2ER. The extracted EtherType is compared against GMAC_ST2ER designated by this EtherType field.
3. An enable bit Compare A, COMPAE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.
4. An enable bit Compare B, COMPBE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.
5. An enable bit Compare C, COMPCE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.

Each screener type has an enable bit, a match pattern and a queue number. If a received frame matches on an enabled Screening register, then the frame will be tagged with the queue value in the associated Screening register, and forwarded onto the DMA and subsequently into the external memory associated with that queue. If two screeners are matched, then the one which resides at the lowest register address will take priority so care must be taken on the selection of the screener location.

When the priority queuing feature is enabled, the number of interrupt outputs from the GMAC core is increased to match the number of supported queues. The number of Interrupt Status registers is increased by the same number. Only DMA related events are reported using the individual interrupt outputs, as the GMAC can relate these events to specific queues. All other events generated within the GMAC are reported in the interrupt associated with the lowest priority queue. For the lowest priority queue (or the only queue when only 1 queue is selected), the Interrupt Status register is located at address 0x24. For all other queues, the Interrupt Status register is located at sequential addresses starting at address 0x400.

Note: The address matching is the first level of filtering. If there is a match, the screeners are the next level of filtering for routing the data to the appropriate queue. See MAC Filtering Block for more details.

The additional screening done by the functions Compare A, B, and C each have an enable bit and compare register field. COMPA, COMPB and COMPC in GMAC_ST2RPQ are pointers to a configured offset (OFFSVAL), value (COMPVAL), and mask (MASKVAL). If enabled, the compare is true if the data at the offset into the frame, ANDed with MASKVAL, is equal to the value of COMPVAL ANDed with MASKVAL. A 16-bit word comparison is done. The byte at the offset number of bytes from the index start is compared to bits 7:0 of the configured COMPVAL and MASKVAL. The byte at the offset number of bytes + 1 from the index start is compared to bits 15:8 of the configured COMPVAL and MASKVAL.

The offset value in bytes, OFFSVAL, ranges from 0 to 127 bytes from either the start of the frame, the byte after the EtherType field, the byte after the IP header (IPv4 or IPv6) or the byte after the TCP/UDP header. Note the logic to decode the IP header or the TCP/UDP header is reused from the TCP/UDP/IP checksum offload logic and therefore has the same restrictions on use (the main limitation is that IP fragmentation is not supported). Refer to the Checksum Offload for IP, TCP and UDP section of this documentation for further details.

Compare A, B, and C use a common set of 24 GMAC_ST2CW0/1 registers, thus all COMPA, COMPB and COMPC fields in the registers GMAC_ST2RPQ point to a single pool of 24 GMAC_ST2CW0/1 registers.

Note that Compare A, B and C together allow matching against an arbitrary 48 bits of data and so can be used to match against a MAC address.

All enabled comparisons are ANDed together to form the overall type 2 screening match.

36.6.4 MAC Transmit Block

The MAC transmitter can operate in either half duplex or full duplex mode and transmits frames in accordance with the Ethernet IEEE 802.3 standard. In half duplex mode, the CSMA/CD protocol of the IEEE 802.3 specification is followed.

A small input buffer receives data through the FIFO interface which will extract data in 32-bit form. All subsequent processing prior to the final output is performed in bytes.

Transmit data can be output using the MII interface.

Frame assembly starts by adding preamble and the start frame delimiter. Data is taken from the transmit FIFO interface a word at a time.

If necessary, padding is added to take the frame length to 60 bytes. CRC is calculated using an order 32-bit polynomial. This is inverted and appended to the end of the frame taking the frame length to a minimum of 64 bytes.

If the no CRC bit is set in the second word of the last buffer descriptor of a transmit frame, neither pad nor CRC are appended. The no CRC bit can also be set through the FIFO interface.

In full duplex mode (at all data rates), frames are transmitted immediately. Back to back frames are transmitted at least 96 bit times apart to guarantee the interframe gap.

In half duplex mode, the transmitter checks carrier sense. If asserted, the transmitter waits for the signal to become inactive, and then starts transmission after the interframe gap of 96 bit times. If the collision signal is asserted during transmission, the transmitter will transmit a jam sequence of 32 bits taken from the data register and then retry transmission after the backoff time has elapsed. If the collision occurs during either the preamble or Start Frame Delimiter (SFD), then these fields will be completed prior to generation of the jam sequence.

The backoff time is based on an XOR of the 10 least significant bits of the data coming from the transmit FIFO interface and a 10-bit pseudo random number generator. The number of bits used depends on the number of collisions seen. After the first collision 1 bit is used, then the second 2 bits and so on up to the maximum of 10 bits. All 10 bits are used above ten collisions. An error will be indicated and no further attempts will be made if 16 consecutive attempts cause collision. This operation is compliant with the description in Clause 4.2.3.2.5 of the IEEE 802.3 standard which refers to the truncated binary exponential backoff algorithm.

In 10/100 mode, both collisions and late collisions are treated identically, and backoff and retry will be performed up to 16 times. This condition is reported in the transmit buffer descriptor word 1 (late collision, bit 26) and also in the Transmit Status register (late collision, bit 7). An interrupt can also be generated (if enabled) when this exception occurs, and bit 5 in the Interrupt Status register will be set.

In all modes of operation, if the transmit DMA underruns, a bad CRC is automatically appended using the same mechanism as jam insertion and the GTXER signal is asserted. For a properly configured system this should never happen and also it is impossible if configured to use the DMA with packet buffers, as the complete frame is buffered in local packet buffer memory.

By setting when bit 28 is set in the Network Configuration register, the Inter Packet Gap (IPG) may be stretched beyond 96 bits depending on the length of the previously transmitted frame and the value written to the IPG Stretch register (GMAC_IPGS). The least significant 8 bits of the IPG Stretch register multiply the previous frame length (including preamble). The next significant 8 bits (+1 so as not to get a divide by zero) divide the frame length to generate the IPG. IPG stretch only works in full duplex mode and when bit 28 is set in the Network Configuration register. The IPG Stretch register cannot be used to shrink the IPG below 96 bits.

36.6.5 MAC Receive Block

All processing within the MAC receive block is implemented using a 16-bit data path. The MAC receive block checks for valid preamble, FCS, alignment and length, presents received frames to the FIFO interface and stores the frame destination address for use by the address checking block.

If, during the frame reception, the frame is found to be too long, a bad frame indication is sent to the FIFO interface. The receiver logic ceases to send data to memory as soon as this condition occurs.

At end of frame reception the receive block indicates to the DMA block whether the frame is good or bad. The DMA block will recover the current receive buffer if the frame was bad.

Ethernet frames are normally stored in DMA memory complete with the FCS. Setting the FCS remove bit in the network configuration (bit 17) causes frames to be stored without their corresponding FCS. The reported frame length field is reduced by four bytes to reflect this operation.

The receive block signals to the register block to increment the alignment, CRC (FCS), short frame, long frame, jabber or receive symbol errors when any of these exception conditions occur.

If bit 26 is set in the network configuration, CRC errors will be ignored and CRC errored frames will not be discarded, though the Frame Check Sequence Errors statistic register will still be incremented. Additionally, if not enabled for jumbo frames mode, then bit[13] of the receiver descriptor word 1 will be updated to indicate the FCS validity for the particular frame. This is useful for applications such as EtherCAT whereby individual frames with FCS errors must be identified.

Received frames can be checked for length field error by setting the length field error frame discard bit of the Network Configuration register (bit-16). When this bit is set, the receiver compares a frame's measured length with the length field (bytes 13 and 14) extracted from the frame. The frame is discarded if the measured length is shorter. This checking procedure is for received frames between 64 bytes and 1518 bytes in length.

Each discarded frame is counted in the 10-bit Length Field Frame Error statistics register. Frames where the length field is greater than or equal to 0x0600 hex will not be checked.

36.6.6 Checksum Offload for IP, TCP and UDP

The GMAC can be programmed to perform IP, TCP and UDP checksum offloading in both receive and transmit directions, which is enabled by setting bit 24 in the Network Configuration register for receive and bit 11 in the DMA Configuration register for transmit.

IPv4 packets contain a 16-bit checksum field, which is the 16-bit 1's complement of the 1's complement sum of all 16-bit words in the header. TCP and UDP packets contain a 16-bit checksum field, which is the 16-bit 1's complement of the 1's complement sum of all 16-bit words in the header, the data and a conceptual IP pseudo header.

To calculate these checksums in software requires each byte of the packet to be processed. For TCP and UDP this can use a large amount of processing power. Offloading the checksum calculation to hardware can result in significant performance improvements.

For IP, TCP or UDP checksum offload to be useful, the operating system containing the protocol stack must be aware that this offload is available so that it can make use of the fact that the hardware can either generate or verify the checksum.

36.6.6.1 Receiver Checksum Offload

When receive checksum offloading is enabled in the GMAC, the IPv4 header checksum is checked as per RFC 791, where the packet meets the following criteria:

- If present, the VLAN header must be four octets long and the CFI bit must not be set.
- Encapsulation must be RFC 894 Ethernet Type Encoding or RFC 1042 SNAP Encoding.
- IPv4 packet
- IP header is of a valid length

The GMAC also checks the TCP checksum as per RFC 793, or the UDP checksum as per RFC 768, if the following criteria are met:

- IPv4 or IPv6 packet
- Good IP header checksum (if IPv4)
- No IP fragmentation
- TCP or UDP packet

When an IP, TCP or UDP frame is received, the receive buffer descriptor gives an indication if the GMAC was able to verify the checksums. There is also an indication if the frame had SNAP encapsulation. These indication bits will replace the type ID match indication bits when the receive checksum offload is enabled. For details of these indication bits, refer to Receive Buffer Descriptor Entry.

If any of the checksums are verified as incorrect by the GMAC, the packet is discarded and the appropriate statistics counter incremented.

36.6.6.2 Transmitter Checksum Offload

The transmitter checksum offload is only available if the full store and forward mode is enabled. This is because the complete frame to be transmitted must be read into the packet buffer memory before the checksum can be calculated and written back into the headers at the beginning of the frame.

Transmitter checksum offload is enabled by setting bit [11] in the DMA Configuration register. When enabled, it will monitor the frame as it is written into the transmitter packet buffer memory to automatically detect the protocol of the frame. Protocol support is identical to the receiver checksum offload.

For transmit checksum generation and substitution to occur, the protocol of the frame must be recognized and the frame must be provided without the FCS field, by making sure that bit [16] of the transmit descriptor word 1 is clear. If the frame data already had the FCS field, this would be corrupted by the substitution of the new checksum fields.

If these conditions are met, the transmit checksum offload engine will calculate the IP, TCP and UDP checksums as appropriate. Once the full packet is completely written into packet buffer memory, the checksums will be valid and the relevant DPRAM locations will be updated for the new checksum fields as per standard IP/TCP and UDP packet structures.

If the transmitter checksum engine is prevented from generating the relevant checksums, bits [22:20] of the transmitter DMA writeback status will be updated to identify the reason for the error. Note that the frame will still be transmitted but without the checksum substitution, as typically the reason that the substitution did not occur was that the protocol was not recognized.

36.6.7 MAC Filtering Block

The filter block determines which frames should be written to the FIFO interface and on to the DMA.

Whether a frame is passed depends on what is enabled in the Network Configuration register, the state of the external matching pins, the contents of the specific address, type and Hash registers and the frame's destination address and type field.

If bit 25 of the Network Configuration register is not set, a frame will not be copied to memory if the GMAC is transmitting in half duplex mode at the time a destination address is received.

Ethernet frames are transmitted a byte at a time, least significant bit first. The first six bytes (48 bits) of an Ethernet frame make up the destination address. The first bit of the destination address, which is the LSB of the first byte of the frame, is the group or individual bit. This is one for multicast addresses and zero for unicast. The all ones address is the broadcast address and a special case of multicast.

The GMAC supports recognition of four specific addresses. Each specific address requires two registers, Specific Address Bottom register and Specific Address Top register. Specific Address Bottom register stores the first four bytes of the destination address and Specific Address Top register contains the last two bytes. The addresses stored can be specific, group, local or universal.

The destination address of received frames is compared against the data stored in the Specific Address registers once they have been activated. The addresses are deactivated at reset or when their corresponding Specific Address Bottom register is written. They are activated when Specific Address Top register is written. If a receive frame address matches an active address, the frame is written to the FIFO interface and on to DMA memory.

Frames may be filtered using the type ID field for matching. Four type ID registers exist in the register address space and each can be enabled for matching by writing a one to the MSB (bit 31) of the respective register. When a frame is received, the matching is implemented as an OR function of the various types of match.

The contents of each type ID register (when enabled) are compared against the length/type ID of the frame being received (e.g., bytes 13 and 14 in non-VLAN and non-SNAP encapsulated frames) and copied to memory if a match is found. The encoded type ID match bits (Word 0, Bit 22 and Bit 23) in the receive buffer descriptor status are set indicating which type ID register generated the match, if the receive checksum offload is disabled.

The reset state of the type ID registers is zero, hence each is initially disabled.

The following example illustrates the use of the address and type ID match registers for a MAC address of 21:43:65:87:A9:CB:

Preamble	55
SFD	D5
DA (Octet 0 - LSB)	21
DA (Octet 1)	43
DA (Octet 2)	65
DA (Octet 3)	87
DA (Octet 4)	A9
DA (Octet 5 - MSB)	CB
SA (LSB)	00(1)
SA	00(1)
SA	00(1)
SA	00(1)

SA	00[1]
SA (MSB)	00[1]
Type ID (MSB)	43
Type ID (LSB)	21

Note:

1. Contains the address of the transmitting device.

The sequence above shows the beginning of an Ethernet frame. Byte order of transmission is from top to bottom as shown. For a successful match to specific address 1, the following address matching registers must be set up:

- Specific Address 1 Bottom register (GMAC_SAB1) (Address 0x088) 0x87654321
- Specific Address 1 Top register (GMAC_SAT1) (Address 0x08C) 0x0000CBA9

For a successful match to the type ID, the following Type ID Match 1 register must be set up:

- Type ID Match 1 register (GMAC_TIDM1) (Address 0x0A8) 0x80004321

36.6.8 Broadcast Address

Frames with the broadcast address of 0xFFFFFFFFFFFF are stored to memory only if the 'no broadcast' bit in the Network Configuration register is set to zero.

36.6.9 Hash Addressing

The hash address register is 64 bits long and takes up two locations in the memory map. The least significant bits are stored in Hash Register Bottom and the most significant bits in Hash Register Top.

The unicast hash enable and the multicast hash enable bits in the Network Configuration register enable the reception of hash matched frames. The destination address is reduced to a 6-bit index into the 64-bit Hash register using the following hash function: The hash function is an XOR of every sixth bit of the destination address.

```
hash_index[05] = da[05] ^ da[11] ^ da[17] ^ da[23] ^ da[29] ^ da[35] ^ da[41] ^ da[47]
hash_index[04] = da[04] ^ da[10] ^ da[16] ^ da[22] ^ da[28] ^ da[34] ^ da[40] ^ da[46]
hash_index[03] = da[03] ^ da[09] ^ da[15] ^ da[21] ^ da[27] ^ da[33] ^ da[39] ^ da[45]
hash_index[02] = da[02] ^ da[08] ^ da[14] ^ da[20] ^ da[26] ^ da[32] ^ da[38] ^ da[44]
hash_index[01] = da[01] ^ da[07] ^ da[13] ^ da[19] ^ da[25] ^ da[31] ^ da[37] ^ da[43]
hash_index[00] = da[00] ^ da[06] ^ da[12] ^ da[18] ^ da[24] ^ da[30] ^ da[36] ^ da[42]
da[0]
```

represents the least significant bit of the first byte received, that is, the multicast/unicast indicator, and `da[47]` represents the most significant bit of the last byte received.

If the hash index points to a bit that is set in the Hash register then the frame will be matched according to whether the frame is multicast or unicast.

A multicast match will be signalled if the multicast hash enable bit is set, `da[0]` is logic 1 and the hash index points to a bit set in the Hash register.

A unicast match will be signalled if the unicast hash enable bit is set, `da[0]` is logic 0 and the hash index points to a bit set in the Hash register.

To receive all multicast frames, the Hash register should be set with all ones and the multicast hash enable bit should be set in the Network Configuration register.

36.6.10 Copy all Frames (Promiscuous Mode)

If the Copy All Frames bit is set in the Network Configuration register then all frames except those that are too long, too short, have FCS errors or have GRXER asserted during reception will be copied to memory. Frames with FCS errors will be copied if bit 26 is set in the Network Configuration register.

36.6.11 Disable Copy of Pause Frames

Pause frames can be prevented from being written to memory by setting the disable copying of pause frames control bit 23 in the Network Configuration register. When set, pause frames are not copied to memory regardless of the

Copy All Frames bit, whether a hash match is found, a type ID match is identified or if a destination address match is found.

36.6.12 VLAN Support

The following table describes an Ethernet encoded 802.1Q VLAN tag.

Table 36-5. 802.1Q VLAN Tag

TPID (Tag Protocol Identifier) 16 bits	TCI (Tag Control Information) 16 bits
0x8100	First 3 bits priority, then CFI bit, last 12 bits VID

The VLAN tag is inserted at the 13th byte of the frame adding an extra four bytes to the frame. To support these extra four bytes, the GMAC can accept frame lengths up to 1536 bytes by setting bit 8 in the Network Configuration register.

If the VID (VLAN identifier) is null (0x000) this indicates a priority-tagged frame.

The following bits in the receive buffer descriptor status word give information about VLAN tagged frames:-

- Bit 21 set if receive frame is VLAN tagged (i.e., type ID of 0x8100).
- Bit 20 set if receive frame is priority tagged (i.e., type ID of 0x8100 and null VID). (If bit 20 is set, bit 21 will be set also.)
- Bit 19, 18 and 17 set to priority if bit 21 is set.
- Bit 16 set to CFI if bit 21 is set.

The GMAC can be configured to reject all frames except VLAN tagged frames by setting the discard non-VLAN frames bit in the Network Configuration register.

36.6.13 IEEE 1588 Support

IEEE 1588 is a standard for precision time synchronization in local area networks. It works with the exchange of special Precision Time Protocol (PTP) frames. The PTP messages can be transported over IEEE 802.3/Ethernet, over Internet Protocol Version 4 or over Internet Protocol Version 6 as described in the annex of IEEE P1588.D2.1.

The GMAC indicates the message timestamp point (asserted on the start packet delimiter and de-asserted at end of frame) for all frames and the passage of PTP event frames (asserted when a PTP event frame is detected and de-asserted at end of frame).

IEEE 802.1AS is a subset of IEEE 1588. One difference is that IEEE 802.1AS uses the Ethernet multicast address 0180C200000E for sync frame recognition whereas IEEE 1588 does not. GMAC is designed to recognize sync frames with both IEEE 802.1AS and IEEE 1588 addresses and so can support both 1588 and 802.1AS frame recognition simultaneously.

Synchronization between master and slave clocks is a two stage process.

First, the offset between the master and slave clocks is corrected by the master sending a sync frame to the slave with a follow up frame containing the exact time the sync frame was sent. Hardware assist modules at the master and slave side detect exactly when the sync frame was sent by the master and received by the slave. The slave then corrects its clock to match the master clock.

Second, the transmission delay between the master and slave is corrected. The slave sends a delay request frame to the master which sends a delay response frame in reply. Hardware assist modules at the master and slave side detect exactly when the delay request frame was sent by the slave and received by the master. The slave will now have enough information to adjust its clock to account for delay. For example, if the slave was assuming zero delay, the actual delay will be half the difference between the transmit and receive time of the delay request frame (assuming equal transmit and receive times) because the slave clock will be lagging the master clock by the delay time already.

The timestamp is taken when the message timestamp point passes the clock timestamp point. This can generate an interrupt if enabled (GMAC_IER). However, MAC Filtering configuration is needed to actually ‘copy’ the message to memory. For Ethernet, the message timestamp point is the SFD and the clock timestamp point is the MII interface. (The IEEE 1588 specification refers to sync and delay_req messages as event messages as these require timestamping. These events are captured in the registers GMAC_EFTx and GMAC_EFRx, respectively. Follow up, delay response and management messages do not require timestamping and are referred to as general messages.)

1588 version 2 defines two additional PTP event messages. These are the peer delay request (Pdelay_Req) and peer delay response (Pdelay_Resp) messages. These events are captured in the registers GMAC_PEFTx and GMAC_PEFRx, respectively. These messages are used to calculate the delay on a link. Nodes at both ends of a link send both types of frames (regardless of whether they contain a master or slave clock). The Pdelay_Resp message contains the time at which a Pdelay_Req was received and is itself an event message. The time at which a Pdelay_Resp message is received is returned in a Pdelay_Resp_Follow_Up message.

1588 version 2 introduces transparent clocks of which there are two kinds, peer-to-peer (P2P) and end-to-end (E2E). Transparent clocks measure the transit time of event messages through a bridge and amend a correction field within the message to allow for the transit time. P2P transparent clocks additionally correct for the delay in the receive path of the link using the information gathered from the peer delay frames. With P2P transparent clocks delay_req messages are not used to measure link delay. This simplifies the protocol and makes larger systems more stable.

The GMAC recognizes four different encapsulations for PTP event messages:

1. 1588 version 1 (UDP/IPv4 multicast)
2. 1588 version 2 (UDP/IPv4 multicast)
3. 1588 version 2 (UDP/IPv6 multicast)
4. 1588 version 2 (Ethernet multicast)

Table 36-6. Example of Sync Frame in 1588 Version 1 Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	–
SA (Octets 6–11)	–
Type (Octets 12–13)	0800
IP stuff (Octets 14–22)	–
UDP (Octet 23)	11
IP stuff (Octets 24–29)	–
IP DA (Octets 30–32)	E00001
IP DA (Octet 33)	81 or 82 or 83 or 84
Source IP port (Octets 34–35)	–
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–42)	–
Version PTP (Octet 43)	01
Other stuff (Octets 44–73)	–
Control (Octet 74)	00
Other stuff (Octets 75–168)	–

Table 36-7. Example of Delay Request Frame in 1588 Version 1 Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	–
SA (Octets 6–11)	–
Type (Octets 12–13)	0800
IP stuff (Octets 14–22)	–

.....continued	
Frame Segment	**Value**
UDP (Octet 23)	11
IP stuff (Octets 24–29)	–
IP DA (Octets 30–32)	E00001
IP DA (Octet 33)	81 or 82 or 83 or 84
Source IP port (Octets 34–35)	–
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–42)	–
Version PTP (Octet 43)	01
Other stuff (Octets 44–73)	–
Control (Octet 74)	01
Other stuff (Octets 75–168)	–

For 1588 version 1 messages, sync and delay request frames are indicated by the GMAC if the frame type field indicates TCP/IP, UDP protocol is indicated, the destination IP address is 224.0.1.129/130/131 or 132, the destination UDP port is 319 and the control field is correct.

The control field is 0x00 for sync frames and 0x01 for delay request frames.

For 1588 version 2 messages, the type of frame is determined by looking at the message type field in the first byte of the PTP frame. Whether a frame is version 1 or version 2 can be determined by looking at the version PTP field in the second byte of both version 1 and version 2 PTP frames.

In version 2 messages sync frames have a message type value of 0x0, delay_req have 0x1, Pdelay_Req have 0x2 and Pdelay_Resp have 0x3.

Table 36-8. Example of Sync Frame in 1588 Version 2 (UDP/IPv4) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	–
SA (Octets 6–11)	–
Type (Octets 12–13)	0800
IP stuff (Octets 14–22)	–
UDP (Octet 23)	11
IP stuff (Octets 24–29)	–
IP DA (Octets 30–33)	E0000181
Source IP port (Octets 34–35)	–
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–41)	–
Message type (Octet 42)	00
Version PTP (Octet 43)	02

Table 36-9. Example of Pdelay_Req Frame in 1588 Version 2 (UDP/IPv4) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	–
SA (Octets 6–11)	–
Type (Octets 12–13)	0800
IP stuff (Octets 14–22)	–
UDP (Octet 23)	11
IP stuff (Octets 24–29)	–
IP DA (Octets 30–33)	E000006B
Source IP port (Octets 34–35)	–
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–41)	–
Message type (Octet 42)	02
Version PTP (Octet 43)	02

Table 36-10. Example of Sync Frame in 1588 Version 2 (UDP/IPv6) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	–
SA (Octets 6–11)	–
Type (Octets 12–13)	86dd
IP stuff (Octets 14–19)	–
UDP (Octet 20)	11
IP stuff (Octets 21–37)	–
IP DA (Octets 38–53)	FF0X00000000018
Source IP port (Octets 54–55)	–
Dest IP port (Octets 56–57)	013F
Other stuff (Octets 58–61)	–
Message type (Octet 62)	00
Other stuff (Octets 63–93)	–
Version PTP (Octet 94)	02

Table 36-11. Example of Pdelay_Resp Frame in 1588 Version 2 (UDP/IPv6) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	–
SA (Octets 6–11)	–

.....continued	
Frame Segment	**Value**
Type (Octets 12–13)	86dd
IP stuff (Octets 14–19)	–
UDP (Octet 20)	11
IP stuff (Octets 21–37)	–
IP DA (Octets 38–53)	FF0200000000006B
Source IP port (Octets 54–55)	–
Dest IP port (Octets 56–57)	013F
Other stuff (Octets 58–61)	–
Message type (Octet 62)	03
Other stuff (Octets 63–93)	–
Version PTP (Octet 94)	02

For the multicast address 011B19000000 sync and delay request frames are recognized depending on the message type field, 00 for sync and 01 for delay request.

Table 36-12. Example of Sync Frame in 1588 Version 2 (Ethernet Multicast) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	011B19000000
SA (Octets 6–11)	–
Type (Octets 12–13)	88F7
Message type (Octet 14)	00
Version PTP (Octet 15)	02

Pdelay request frames need a special multicast address so they can pass through ports blocked by the spanning tree protocol. For the multicast address 0180C200000E sync, Pdelay_Req and Pdelay_Resp frames are recognized depending on the message type field, 00 for sync, 02 for pdelay request and 03 for pdelay response.

Table 36-13. Example of Pdelay_Req Frame in 1588 Version 2 (Ethernet Multicast) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	0180C200000E
SA (Octets 6–11)	–
Type (Octets 12–13)	88F7
Message type (Octet 14)	00
Version PTP (Octet 15)	02

36.6.14 Timestamp Unit

The TSU consists of a timer and registers to capture the time at which PTP event frames cross the message timestamp point. An interrupt is issued when a capture register is updated.

The timer is implemented as a 94-bit register with the upper 48 bits counting seconds, the next 30 bits counting nanoseconds and the lowest 16 bits counting sub-nanoseconds. The lower 46 bits rolls over when they have counted to one second. An interrupt is generated when the seconds increment. The timer value can be read, written and adjusted through the APB interface. The timer is clocked by MCK.

The amount by which the timer increments each clock cycle is controlled by the timer increment registers (GMAC_TI). Bits 7:0 are the default increment value in nanoseconds and an additional 16 bits of sub-nanosecond resolution are available using the Timer Increment Sub-nanoseconds register (GMAC_TISUBN). If the rest of the register is written with zero, the timer increments by the value in [7:0], plus the value of GMAC_TISUBN, each clock cycle.

The GMAC_TISUBN register allows a resolution of approximately 15 femtoseconds.

Bits 15:8 of the increment register are the alternative increment value in nanoseconds and bits 23:16 are the number of increments after which the alternative increment value is used. If 23:16 are zero then the alternative increment value will never be used.

Taking the example of 10.2 MHz, there are 102 cycles every ten microseconds or 51 every five microseconds. So a timer with a 10.2 MHz clock source is constructed by incrementing by 98 ns for fifty cycles and then incrementing by 100 ns (98 × 50 + 100 = 5000). This is programmed by setting the 1588 Timer Increment register to 0x00326462.

For a 49.8 MHz clock source it would be 20 ns for 248 cycles followed by an increment of 40 ns (20 × 248 + 40 = 5000) programmed as 0x00F82814.

Having eight bits for the “number of increments” field allows frequencies up to 50 MHz to be supported with 200 kHz resolution.

Without the alternative increment field the period of the clock would be limited to an integer number of nanoseconds, resulting in supported clock frequencies of 8, 10, 20, 25, 40, 50, 100, 125, 200 and 250 MHz.

There are eight additional 80-bit registers that capture the time at which PTP event frames are transmitted and received. An interrupt is issued when these registers are updated. The TSU timer count value can be compared to a programmable comparison value. For the comparison, the 48 bits of the seconds value and the upper 22 bits of the nanoseconds value are used. A signal (GTSUCOMP) is provided to indicate when the TSU timer count value is equal to the comparison value stored in the TSU timer comparison value registers (0x0DC, 0x0E0, and 0x0E4). The GTSUCOMP signal can be routed to the Timer peripheral to automatically toggle pin TOIA11/PD21. This can be used as the reference clock for an external PLL to regenerate the audio clock in Ethernet AVB. An interrupt can also be generated (if enabled) when the TSU timer count value and comparison value are equal, mapped to bit 29 of the Interrupt Status register.

36.6.15 MAC 802.3 Pause Frame Support

Note: See Clause 31, and Annex 31A and 31B of the IEEE standard 802.3 for a full description of MAC 802.3 pause operation.

The following table shows the start of a MAC 802.3 pause frame.

Table 36-14. Start of an 802.3 Pause Frame

Address		Type (MAC Control Frame)	Pause	
Destination	Source		Opcode	Time
0x0180C2000001	6 bytes	0x8808	0x0001	2 bytes

The GMAC supports both hardware controlled pause of the transmitter, upon reception of a pause frame, and hardware generated pause frame transmission.

36.6.15.1 802.3 Pause Frame Reception

Bit 13 of the Network Configuration register is the pause enable control for reception. If this bit is set, transmission pauses if a non zero pause quantum frame is received.

If a valid pause frame is received, then the Pause Time register is updated with the new frame's pause time, regardless of whether a previous pause frame is active or not. An interrupt (either bit 12 or bit 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled (bit 12 and bit 13 of the Interrupt Mask register). Pause frames received with non zero quantum are indicated through the interrupt

bit 12 of the Interrupt Status register. Pause frames received with zero quantum are indicated on bit 13 of the Interrupt Status register.

Once the Pause Time register is loaded and the frame currently being transmitted has been sent, no new frames are transmitted until the pause time reaches zero. The loading of a new pause time, and hence the pausing of transmission, only occurs when the GMAC is configured for full duplex operation. If the GMAC is configured for half duplex there will be no transmission pause, but the pause frame received interrupt will still be triggered. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address 1 register or if it matches the reserved address of 0x0180C2000001. It must also have the MAC control frame type ID of 0x8808 and have the pause opcode of 0x0001.

Pause frames that have frame check sequence (FCS) or other errors will be treated as invalid and will be discarded. 802.3 Pause frames that are received after Priority-based Flow Control (PFC) has been negotiated will also be discarded. Valid pause frames received will increment the Pause Frames Received statistic register.

The Pause Time register decrements every 512 bit times once transmission has stopped. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration register) which causes the Pause Time register to decrement every GTXCK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Pause Time register decrements to zero (assuming it has been enabled by bit 13 in the Interrupt Mask register). This interrupt is also set when a zero quantum pause frame is received.

36.6.15.2 802.3 Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit pause frame bits of the Network Control register. If either bit 11 or bit 12 of the Network Control register is written with logic 1, an 802.3 pause frame will be transmitted, providing full duplex is selected in the Network Configuration register and the transmit block is enabled in the Network Control register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise the following:

- A destination address of 01-80-C2-00-00-01
- A source address taken from Specific Address 1 register
- A type ID of 88-08 (MAC control frame)
- A pause opcode of 00-01
- A Pause Quantum register
- Fill of 00 to take the frame to minimum frame length
- Valid FCS

The pause quantum used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 11 is written with a one, the pause quantum will be taken from the Transmit Pause Quantum register. The Transmit Pause Quantum register resets to a value of 0xFFFF giving maximum pause quantum as default.
- If bit 12 is written with a one, the pause quantum will be zero.

After transmission, a pause frame transmitted interrupt will be generated (bit 14 of the Interrupt Status register) and the only the statistics register Pause Frames Transmitted is incremented.

Pause frames can also be transmitted by the MAC using normal frame transmission methods.

36.6.16 MAC PFC Priority-based Pause Frame Support

Note: Refer to the 802.1Qbb standard for a full description of priority-based pause operation.

The following table shows the start of a Priority-based Flow Control (PFC) pause frame.

Table 36-15. Start of a PFC Pause Frame

Address		Type (Mac Control Frame)	Pause Opcode	Priority Enable Vector	Pause Time
Destination	Source				
0x0180C2000001	6 bytes	0x8808	0x1001	2 bytes	8 × 2 bytes

The GMAC supports PFC priority-based pause transmission and reception. Before PFC pause frames can be received, bit 16 of the Network Control register must be set.

36.6.16.1 PFC Pause Frame Reception

The ability to receive and decode priority-based pause frames is enabled by setting bit 16 of the Network Control register. When this bit is set, the GMAC will match either classic 802.3 pause frames or PFC priority-based pause frames. Once a priority-based pause frame has been received and matched, then from that moment on the GMAC will only match on priority-based pause frames (this is an 802.1Qbb requirement, known as PFC negotiation). Once priority-based pause has been negotiated, any received 802.3x format pause frames will not be acted upon.

If a valid priority-based pause frame is received then the GMAC will decode the frame and determine which, if any, of the eight priorities require to be paused. Up to eight Pause Time registers are then updated with the eight pause times extracted from the frame regardless of whether a previous pause operation is active or not. An interrupt (either bit 12 or bit 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled (bit 12 and bit 13 of the Interrupt Mask register). Pause frames received with non zero quantum are indicated through the interrupt bit 12 of the Interrupt Status register. Pause frames received with zero quantum are indicated on bit 13 of the Interrupt Status register. The loading of a new pause time only occurs when the GMAC is configured for full duplex operation. If the GMAC is configured for half duplex, the pause time counters will not be loaded, but the pause frame received interrupt will still be triggered. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address 1 register or if it matches the reserved address of 0x0180C2000001. It must also have the MAC control frame type ID of 0x8808 and have the pause opcode of 0x0101.

Pause frames that have frame check sequence (FCS) or other errors will be treated as invalid and will be discarded. Valid pause frames received will increment the Pause Frames Received Statistic register.

The Pause Time registers decrement every 512 bit times immediately following the PFC frame reception. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration register) which causes the Pause Time register to decrement every GRXCK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Pause Time register decrements to zero (assuming it has been enabled by bit 13 in the Interrupt Mask register). This interrupt is also set when a zero quantum pause frame is received.

36.6.16.2 PFC Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit priority-based pause frame bit of the Network Control register. If bit 17 of the Network Control register is written with logic 1, a PFC pause frame will be transmitted providing full duplex is selected in the Network Configuration register and the transmit block is enabled in the Network Control register. When bit 17 of the Network Control register is set, the fields of the priority-based pause frame will be built using the values stored in the Transmit PFC Pause register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise the following:

- A destination address of 01-80-C2-00-00-01
- A source address taken from Specific Address 1 register
- A type ID of 88-08 (MAC control frame)
- A pause opcode of 01-01
- A priority enable vector taken from Transmit PFC Pause register
- 8 Pause Quantum registers
- Fill of 00 to take the frame to minimum frame length
- Valid FCS

The Pause Quantum registers used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 17 of the Network Control register is written with a one, then the priority enable vector of the priority-based pause frame will be set equal to the value stored in the Transmit PFC Pause register [7:0]. For each entry equal to zero in the Transmit PFC Pause register [15:8], the pause quantum field of the pause frame associated with that entry will be taken from the Transmit Pause Quantum register. For each entry equal to one in the Transmit PFC Pause register [15:8], the pause quantum associated with that entry will be zero.
- The Transmit Pause Quantum register resets to a value of 0xFFFF giving maximum pause quantum as default.

After transmission, a pause frame transmitted interrupt will be generated (bit 14 of the Interrupt Status register) and the only statistics register that will be incremented will be the Pause Frames Transmitted register.

PFC Pause frames can also be transmitted by the MAC using normal frame transmission methods.

36.6.17 Energy-efficient Ethernet Support

IEEE 802.3az adds support for energy efficiency to Ethernet. These are the key features of 802.3az:

- Allows a system's transmit path to enter a low power mode if there is nothing to transmit.
- Allows a PHY to detect whether its link partner's transmit path is in low power mode, therefore allowing the system's receive path to enter low power mode.
- Link remains up during lower power mode and no frames are dropped.
- Asymmetric, one direction can be in low power mode while the other is transmitting normally.
- LPI (Low Power Idle) signaling is used to control entry and exit to and from low power modes.
- LPI signaling can only take place if both sides have indicated support for it through auto-negotiation.

These are the key features of 802.3az operation:

- Low power control is done at the MII (reconciliation sublayer).
- As an architectural convenience in writing the 802.3az it is assumed that transmission is deferred by asserting carrier sense, in practice it will not be done this way. This system will know when it has nothing to transmit and only enter low power mode when it is not transmitting.
- LPI should not be requested unless the link has been up for at least one second.
- LPI is signaled on the transmit path by asserting 0x01 on txd with tx_en low and tx_er high.
- A PHY on seeing LPI requested on the MII will send the sleep signal before going quiet. After going quiet it will periodically transmit refresh signals.
- LPI mode ends by transmitting normal idle for the wake time. There is a default time for this but it can be adjusted in software using the Link Layer Discovery Protocol (LLDP) described in Clause 79 of 802.3az.
- LPI is indicated at the receive side when sleep and refresh signaling has been detected.

36.6.18 802.1Qav Support - Credit-based Shaping

A credit-based shaping algorithm is available on the two highest priority queues and is defined in the standard 802.1Qav: Forwarding and Queuing Enhancements for Time-Sensitive Streams. This allows traffic on these queues to be limited and to allow other queues to transmit.

Traffic shaping is enabled via the CBS (Credit Based Shaping) Control register. This enables a counter which stores the amount of transmit 'credit', measured in bytes that a particular queue has. A queue may only transmit if it has non-negative credit. If a queue has data to send, but is held off from doing as another queue is transmitting, then credit will accumulate in the credit counter at the rate defined in the IdleSlope register (GMAC_CBSISQx) for that queue.

portTransmitRate is the transmission rate, in bits per second, that the underlying MAC service that supports transmission through the Port provides. The value of this parameter is determined by the operation of the MAC.

IdleSlope is the rate of change of increasing credit when waiting to transmit and must be less than the value of the portTransmitRate.

The max value of IdleSlope (or sendSlope) is (portTransmitRate / bits_per_MII_Clock).

In case of 100Mbps, maximum IdleSlope = (100Mbps / 4) = 0x17D7840.

When this queue is transmitting, the credit counter is decremented at the rate of sendSlope, which is defined as (portTransmitRate - IdleSlope). A queue can accumulate negative credit when transmitting which will hold off any

other transfers from that queue until credit returns to a non-negative value. No transfers are halted when a queue's credit becomes negative; it will accumulate negative credit until the transfer completes.

The highest priority queue always has priority regardless of which queue has the most credit.

36.6.19 LPI Operation in the GMAC

It is best to use firmware to control LPI. LPI operation happens at the system level. Firmware gives maximum control and flexibility of operation. LPI operation is straightforward and firmware should be capable of responding within the required timeframes.

Autonegotiation:

1. Indicate EEE capability using next page autonegotiation.

For the transmit path:

1. If the link has been up for 1 second and there is nothing being transmitted, write to the TXLPIEN bit in the Network Control register.
2. Wake up by clearing the TXLPIEN bit in the Network Control register.

For the receive path:

1. Enable RXLPISBC bit in GMAC_IER. The bit RXLPIS is set in Network Status Register triggering an interrupt.
2. Wait for an interrupt to indicate that LPI has been received.
3. Disable relevant parts of the receive path if desired.
4. The RXLPIS bit in Network Status Register gets cleared to indicate that regular idle has been received. This triggers an interrupt.
5. Re-enable the receive path.

36.6.20 PHY Interface

Different PHY interfaces are supported by the Ethernet MAC:

- MII
- RMII

The MII interface is provided for 10/100 operation and uses txd[3:0] and rxd[3:0]. The RMII interface is provided for 10/100 operation and uses txd[1:0] and rxd[1:0].

36.6.21 10/100 Operation

The 10/100Mbps speed bit in the Network Configuration register is used to select between 10Mbps and 100Mbps.

36.6.22 Jumbo Frames

The jumbo frames enable bit in the Network Configuration register allows the GMAC, in its default configuration, to receive jumbo frames up to 10240 bytes in size. This operation does not form part of the IEEE 802.3 specification and is normally disabled. When jumbo frames are enabled, frames received with a frame size greater than 10240 bytes are discarded.

36.7 Programming Interface

36.7.1 Initialization

36.7.1.1 Configuration

Initialization of the GMAC configuration (e.g., loop back mode, frequency ratios) must be done while the transmit and receive circuits are disabled. See the description of the Network Control register and Network Configuration register earlier in this document.

To change loop back mode, the following sequence of operations must be followed:

1. Write to Network Control register to disable transmit and receive circuits.

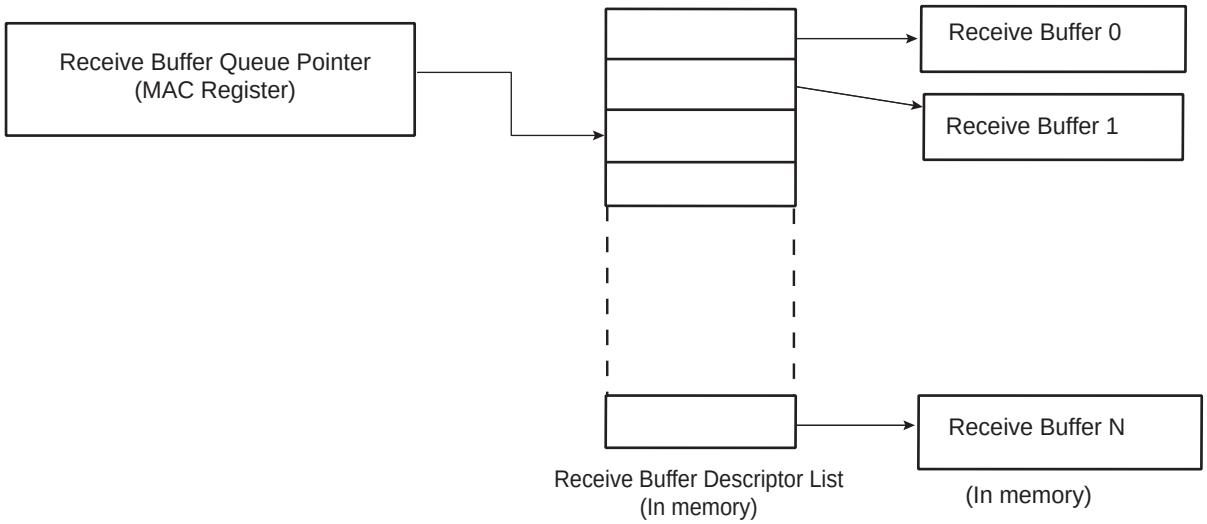
2. Write to Network Control register to change loop back mode.
3. Write to Network Control register to re-enable transmit or receive circuits.
   Note: These writes to the Network Control register cannot be combined in any way.

36.7.1.2 Receive Buffer List

Receive data is written to areas of data (i.e., buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (receive buffer queue) is a sequence of descriptor entries as defined in Receive Buffer Descriptor Entry.

The Receive Buffer Queue Pointer register points to this data structure.

Figure 36-3. Receive Buffer List



To create the list of buffers:

1. Allocate a number (N) of buffers of X bytes in system memory, where X is the DMA buffer length programmed in the DMA Configuration register.
2. Allocate an area 8N bytes for the receive buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GMAC, i.e., bit 0 of word 0 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 1 in word 0 set to 1).
4. Write address of receive buffer descriptor list and control information to GMAC register receive buffer queue pointer
5. The receive circuits can then be enabled by writing to the address recognition registers and the Network Control register.
   Note: The queue pointers must be initialized and point to USED descriptors for all queues including those not intended for use.

36.7.1.3 Transmit Buffer List

Transmit data is read from areas of data (the buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (Transmit Buffer Queue) is a sequence of descriptor entries as defined in Transmit Buffer Descriptor Entry.

The Transmit Buffer Queue Pointer register points to this data structure.

To create this list of buffers:

1. Allocate a number (N) of buffers of between 1 and 2047 bytes of data to be transmitted in system memory. Up to 128 buffers per frame are allowed.
2. Allocate an area 8N bytes for the transmit buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GMAC, i.e., bit 31 of word 1 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 30 in word 1 set to 1).
4. Write address of transmit buffer descriptor list and control information to GMAC register transmit buffer queue pointer.

5. The transmit circuits can then be enabled by writing to the Network Control register.
   Note: The queue pointers must be initialized and point to USED descriptors for all queues including those not intended for use.

36.7.1.4 Address Matching

The GMAC Hash register pair and the four Specific Address register pairs must be written with the required values. Each register pair comprises of a bottom register and top register, with the bottom register being written first. The address matching is disabled for a particular register pair after the bottom register has been written and re-enabled when the top register is written. Each register pair may be written at any time, regardless of whether the receive circuits are enabled or disabled.

As an example, to set Specific Address 1 register to recognize destination address 21:43:65:87:A9:CB, the following values are written to Specific Address 1 Bottom register and Specific Address 1 Top register:

- Specific Address 1 Bottom register bits 31:0 (0x98): 0x8765_4321.
- Specific Address 1 Top register bits 31:0 (0x9C): 0x0000_CBA9.

Note: The address matching is the first level of filtering. If there is a match, the screeners are the next level of filtering for routing the data to the appropriate queue. See Priority Queueing in the DMA for more details.

36.7.1.5 PHY Maintenance

The PHY Maintenance register is implemented as a shift register. Writing to the register starts a shift operation which is signalled as complete when bit two is set in the Network Status register (about 2000 MCK cycles later when bits 18:16 are set to 010 in the Network Configuration register). An interrupt is generated as this bit is set.

During this time, the MSB of the register is output on the GMDIO pin and the LSB updated from the GMDIO pin with each Management Data Clock (MDC) cycle. This causes the transmission of a PHY management frame on GMDIO. See section 22.2.4.5 of the IEEE 802.3 standard.

Reading during the shift operation will return the current contents of the shift register. At the end of the management operation the bits will have shifted back to their original locations. For a read operation the data bits are updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced.

The Management Data Clock (MDC) should not toggle faster than 2.5 MHz (minimum period of 400 ns), as defined by the IEEE 802.3 standard. MDC is generated by dividing down MCK. Three bits in the Network Configuration register determine by how much MCK should be divided to produce MDC.

36.7.1.6 Interrupts

There are 18 interrupt conditions that are detected within the GMAC. The conditions are ORed to make multiple interrupts. Depending on the overall system design this may be passed through a further level of interrupt collection (interrupt controller). On receipt of the interrupt signal, the CPU enters the interrupt handler. Refer to the device interrupt controller documentation to identify that it is the GMAC that is generating the interrupt. To ascertain which interrupt, read the Interrupt Status register. Note that in the default configuration this register will clear itself after being read, though this may be configured to be write-one-to-clear if desired.

At reset all interrupts are disabled. To enable an interrupt, write to Interrupt Enable register with the pertinent interrupt bit set to 1. To disable an interrupt, write to Interrupt Disable register with the pertinent interrupt bit set to 1. To check whether an interrupt is enabled or disabled, read Interrupt Mask register. If the bit is set to 1, the interrupt is disabled.

36.7.1.7 Transmitting Frames

The procedure to set up a frame for transmission is the following:

1. Enable transmit in the Network Control register.
2. Allocate an area of system memory for transmit data. This does not have to be contiguous, varying byte lengths can be used if they conclude on byte borders.
3. Set-up the transmit buffer list by writing buffer addresses to word zero of the transmit buffer descriptor entries and control and length to word one.
4. Write data for transmission into the buffers pointed to by the descriptors.
5. Write the address of the first buffer descriptor to transmit buffer descriptor queue pointer.
6. Enable appropriate interrupts.
7. Write to the transmit start bit (TSTART) in the Network Control register.

36.7.1.8 Receiving Frames

When a frame is received and the receive circuits are enabled, the GMAC checks the address and, in the following cases, the frame is written to system memory:

- If it matches one of the four Specific Address registers.
- If it matches one of the four Type ID registers.
- If it matches the hash address function.
- If it is a broadcast address (0xFFFFFFFFFFFF) and broadcasts are allowed.
- If the GMAC is configured to “copy all frames”.

The register receive buffer queue pointer points to the next entry in the receive buffer descriptor list and the GMAC uses this as the address in system memory to write the frame to.

Once the frame has been completely and successfully received and written to system memory, the GMAC then updates the receive buffer descriptor entry (see Receive Buffer Descriptor Entry) with the reason for the address match and marks the area as being owned by software. Once this is complete, a receive complete interrupt is set. Software is then responsible for copying the data to the application area and releasing the buffer (by writing the ownership bit back to 0).

If the GMAC is unable to write the data at a rate to match the incoming frame, then a receive overrun interrupt is set. If there is no receive buffer available, i.e., the next buffer is still owned by software, a receive buffer not available interrupt is set. If the frame is not successfully received, a statistics register is incremented and the frame is discarded without informing software.

36.7.2 Statistics Registers

Statistics registers are described in the User Interface beginning with GMAC Octets Transmitted Low Register and ending with GMAC UDP Checksum Errors Register.

The statistics register block begins at 0x100 and runs to 0x1B0, and comprises the registers listed below.

Octets Transmitted Low Register	Broadcast Frames Received Register
Octets Transmitted High Register	Multicast Frames Received Register
Frames Transmitted Register	Pause Frames Received Register
Broadcast Frames Transmitted Register	64 Byte Frames Received Register
Multicast Frames Transmitted Register	65 to 127 Byte Frames Received Register
Pause Frames Transmitted Register	128 to 255 Byte Frames Received Register
64 Byte Frames Transmitted Register	256 to 511 Byte Frames Received Register
65 to 127 Byte Frames Transmitted Register	512 to 1023 Byte Frames Received Register
128 to 255 Byte Frames Transmitted Register	1024 to 1518 Byte Frames Received Register
256 to 511 Byte Frames Transmitted Register	1519 to Maximum Byte Frames Received Register
512 to 1023 Byte Frames Transmitted Register	Undersize Frames Received Register
1024 to 1518 Byte Frames Transmitted Register	Oversize Frames Received Register
Greater Than 1518 Byte Frames Transmitted Register	Jabbers Received Register
Transmit Underruns Register	Frame Check Sequence Errors Register
Single Collision Frames Register	Length Field Frame Errors Register
Multiple Collision Frames Register	Receive Symbol Errors Register
Excessive Collisions Register	Alignment Errors Register
Late Collisions Register	Receive Resource Errors Register
Deferred Transmission Frames Register	Receive Overrun Register

Carrier Sense Errors Register	IP Header Checksum Errors Register
Octets Received Low Register	TCP Checksum Errors Register
Octets Received High Register	UDP Checksum Errors Register
Frames Received Register	

These registers reset to zero on a read and stick at all ones when they count to their maximum value. They should be read frequently enough to prevent loss of data.

The receive statistics registers are only incremented when the receive enable bit (RXEN) is set in the Network Control register.

Once a statistics register has been read, it is automatically cleared. When reading the Octets Transmitted and Octets Received registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

36.8 Register Summary

Offset	Name	Bit Pos.								
0x00	GMAC_NCR	7:0	WESTAT	INCSTAT	CLRSTAT	MPE	TXEN	RXEN	LBL	
		15:8	SRTSM			TXZQPF	TXPF	THALT	TSTART	BP
		23:16					TXLPIEN	FNP	TXPBPF	ENPBPR
		31:24								
0x04	GMAC_NCFGR	7:0	UNIHEN	MTIHEN	NBC	CAF	JFRAME	DNVLAN	FD	SPD
		15:8	RXBUFO[1:0]		PEN	RTY				MAXFS
		23:16	DCPF	DBW[1:0]		CLK[2:0]			RFCS	LFERD
		31:24		IRXER	RXBP	IPGSEN		IRXFCS	EFRHD	RXCOEN
0x08	GMAC_NSR	7:0	RXLPIS					IDLE	MDIO	
		15:8								
		23:16								
		31:24								
0x0C	GMAC_UR	7:0								RMII
		15:8								
		23:16								
		31:24								
0x10	GMAC_DCFGR	7:0	ESPA	ESMA		FBLDO[4:0]				
		15:8					TXCOEN	TXPBMS	RXBMS[1:0]	
		23:16	DRBS[7:0]							
		31:24								DDRP
0x14	GMAC_TSR	7:0			TXCOMP	TFC	TXGO	RLE	COL	UBR
		15:8								HRESP
		23:16								
		31:24								
0x18	GMAC_RBQB	7:0	ADDR[5:0]							
		15:8	ADDR[13:6]							
		23:16	ADDR[21:14]							
		31:24	ADDR[29:22]							
0x1C	GMAC_TBQB	7:0	ADDR[5:0]							
		15:8	ADDR[13:6]							
		23:16	ADDR[21:14]							
		31:24	ADDR[29:22]							
0x20	GMAC_RSR	7:0					HNO	RXOVR	REC	BNA
		15:8								
		23:16								
		31:24								
0x24	GMAC_ISR	7:0	TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS
		15:8		PFTR	PTZ	PFNZ	HRESP	ROVR		
		23:16	PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR		
		31:24			TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
0x28	GMAC_IER	7:0	TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS
		15:8	EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR		
		23:16	PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR		
		31:24			TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
0x2C	GMAC_IDR	7:0	TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS
		15:8	EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR		
		23:16	PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR		
		31:24			TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
0x30	GMAC_IMR	7:0	TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS
		15:8	EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR		
		23:16	PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR		
		31:24			TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT

.....continued

Offset	Name	Bit Pos.								
0x34	GMAC_MAN	7:0	DATA[7:0]							
		15:8	DATA[15:8]							
		23:16	PHYA[0]	REGA[4:0]					WTN[1:0]	
		31:24	WZO	CLTTO	OP[1:0]		PHYA[4:1]			
0x38	GMAC_RPQ	7:0	RPQ[7:0]							
		15:8	RPQ[15:8]							
		23:16								
		31:24								
0x3C	GMAC_TPQ	7:0	TPQ[7:0]							
		15:8	TPQ[15:8]							
		23:16								
		31:24								
0x40	GMAC_TPSF	7:0	TPB1ADR[7:0]							
		15:8					TPB1ADR[11:8]			
		23:16								
		31:24	ENTXP							
0x44	GMAC_RPSF	7:0	RPB1ADR[7:0]							
		15:8					RPB1ADR[11:8]			
		23:16								
		31:24	ENRXP							
0x48	GMAC_RJFML	7:0	FML[7:0]							
		15:8			FML[13:8]					
		23:16								
		31:24								
0x4C ... 0x7F	Reserved									
0x80	GMAC_HRB	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x84	GMAC_HRT	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x88	GMAC_SAB1	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x8C	GMAC_SAT1	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16								
		31:24								
0x90	GMAC_SAB2	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x94	GMAC_SAT2	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16								
		31:24								
0x98	GMAC_SAB3	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x9C	GMAC_SAT3	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0xA0	GMAC_SAB4	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0xA4	GMAC_SAT4	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16								
		31:24								
0xA8	GMAC_TIDM1	7:0	TID[7:0]							
		15:8	TID[15:8]							
		23:16								
		31:24	ENID1							
0xAC	GMAC_TIDM2	7:0	TID[7:0]							
		15:8	TID[15:8]							
		23:16								
		31:24	ENID2							
0xB0	GMAC_TIDM3	7:0	TID[7:0]							
		15:8	TID[15:8]							
		23:16								
		31:24	ENID3							
0xB4	GMAC_TIDM4	7:0	TID[7:0]							
		15:8	TID[15:8]							
		23:16								
		31:24	ENID4							
0xB8 ... 0xBB	Reserved									
0xBC	GMAC_IPGS	7:0	FL[7:0]							
		15:8	FL[15:8]							
		23:16								
		31:24								
0xC0	GMAC_SVLAN	7:0	VLAN_TYPE[7:0]							
		15:8	VLAN_TYPE[15:8]							
		23:16								
		31:24	ESVLAN							
0xC4	GMAC_TPFCP	7:0	PEV[7:0]							
		15:8	PQ[7:0]							
		23:16								
		31:24								
0xC8	GMAC_SAMB1	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0xCC	GMAC_SAMT1	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16								
		31:24								
0xD0 ... 0xDB	Reserved									
0xDC	GMAC_NSC	7:0	NANOSEC[7:0]							
		15:8	NANOSEC[15:8]							
		23:16			NANOSEC[21:16]					
		31:24								
0xE0	GMAC_SCL	7:0	SEC[7:0]							
		15:8	SEC[15:8]							
		23:16	SEC[23:16]							
		31:24	SEC[31:24]							

.....continued

Offset	Name	Bit Pos.								
0xE4	GMAC_SCH	7:0	SEC[7:0]							
		15:8	SEC[15:8]							
		23:16								
		31:24								
0xE8	GMAC_EFTSH	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16								
		31:24								
0xEC	GMAC_EFRSH	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16								
		31:24								
0xF0	GMAC_PEFTSH	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16								
		31:24								
0xF4	GMAC_PEFRSH	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16								
		31:24								
0xF8 ... 0xFF	Reserved									
0x0100	GMAC_OTLO	7:0	TXO[7:0]							
		15:8	TXO[15:8]							
		23:16	TXO[23:16]							
		31:24	TXO[31:24]							
0x0104	GMAC_OTHI	7:0	TXO[7:0]							
		15:8	TXO[15:8]							
		23:16								
		31:24								
0x0108	GMAC_FT	7:0	FTX[7:0]							
		15:8	FTX[15:8]							
		23:16	FTX[23:16]							
		31:24	FTX[31:24]							
0x010C	GMAC_BCFT	7:0	BFTX[7:0]							
		15:8	BFTX[15:8]							
		23:16	BFTX[23:16]							
		31:24	BFTX[31:24]							
0x0110	GMAC_MFT	7:0	MFTX[7:0]							
		15:8	MFTX[15:8]							
		23:16	MFTX[23:16]							
		31:24	MFTX[31:24]							
0x0114	GMAC_PFT	7:0	PFTX[7:0]							
		15:8	PFTX[15:8]							
		23:16								
		31:24								
0x0118	GMAC_BFT64	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x011C	GMAC_TBFT127	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0120	GMAC_TBFT255	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							

.....continued

Offset	Name	Bit Pos.								
0x0124	GMAC_TBFT511	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0128	GMAC_TBFT1023	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x012C	GMAC_TBFT1518	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0130	GMAC_GTBFT1518	7:0	NFTX[7:0]							
		15:8	NFTX[15:8]							
		23:16	NFTX[23:16]							
		31:24	NFTX[31:24]							
0x0134	GMAC_TUR	7:0	TXUNR[7:0]							
		15:8							TXUNR[9:8]	
		23:16								
		31:24								
0x0138	GMAC_SCF	7:0	SCOL[7:0]							
		15:8	SCOL[15:8]							
		23:16							SCOL[17:16]	
		31:24								
0x013C	GMAC_MCF	7:0	MCOL[7:0]							
		15:8	MCOL[15:8]							
		23:16							MCOL[17:16]	
		31:24								
0x0140	GMAC_EC	7:0	XCOL[7:0]							
		15:8							XCOL[9:8]	
		23:16								
		31:24								
0x0144	GMAC_LC	7:0	LCOL[7:0]							
		15:8							LCOL[9:8]	
		23:16								
		31:24								
0x0148	GMAC_DTF	7:0	DEFT[7:0]							
		15:8	DEFT[15:8]							
		23:16							DEFT[17:16]	
		31:24								
0x014C	GMAC_CSE	7:0	CSR[7:0]							
		15:8							CSR[9:8]	
		23:16								
		31:24								
0x0150	GMAC_ORLO	7:0	RXO[7:0]							
		15:8	RXO[15:8]							
		23:16	RXO[23:16]							
		31:24	RXO[31:24]							
0x0154	GMAC_ORHI	7:0	RXO[7:0]							
		15:8	RXO[15:8]							
		23:16								
		31:24								
0x0158	GMAC_FR	7:0	FRX[7:0]							
		15:8	FRX[15:8]							
		23:16	FRX[23:16]							
		31:24	FRX[31:24]							

.....continued

Offset	Name	Bit Pos.								
0x015C	GMAC_BCFR	7:0	BFRX[7:0]							
		15:8	BFRX[15:8]							
		23:16	BFRX[23:16]							
		31:24	BFRX[31:24]							
0x0160	GMAC_MFR	7:0	MFRX[7:0]							
		15:8	MFRX[15:8]							
		23:16	MFRX[23:16]							
		31:24	MFRX[31:24]							
0x0164	GMAC_PFR	7:0	PFRX[7:0]							
		15:8	PFRX[15:8]							
		23:16								
		31:24								
0x0168	GMAC_BFR64	7:0	NFRX[7:0]							
		15:8	NFRX[15:8]							
		23:16	NFRX[23:16]							
		31:24	NFRX[31:24]							
0x016C	GMAC_TBFR127	7:0	NFRX[7:0]							
		15:8	NFRX[15:8]							
		23:16	NFRX[23:16]							
		31:24	NFRX[31:24]							
0x0170	GMAC_TBFR255	7:0	NFRX[7:0]							
		15:8	NFRX[15:8]							
		23:16	NFRX[23:16]							
		31:24	NFRX[31:24]							
0x0174	GMAC_TBFR511	7:0	NFRX[7:0]							
		15:8	NFRX[15:8]							
		23:16	NFRX[23:16]							
		31:24	NFRX[31:24]							
0x0178	GMAC_TBFR1023	7:0	NFRX[7:0]							
		15:8	NFRX[15:8]							
		23:16	NFRX[23:16]							
		31:24	NFRX[31:24]							
0x017C	GMAC_TBFR1518	7:0	NFRX[7:0]							
		15:8	NFRX[15:8]							
		23:16	NFRX[23:16]							
		31:24	NFRX[31:24]							
0x0180	GMAC_TMXBFR	7:0	NFRX[7:0]							
		15:8	NFRX[15:8]							
		23:16	NFRX[23:16]							
		31:24	NFRX[31:24]							
0x0184	GMAC_UFR	7:0	UFRX[7:0]							
		15:8							UFRX[9:8]	
		23:16								
		31:24								
0x0188	GMAC_OFR	7:0	OFRX[7:0]							
		15:8							OFRX[9:8]	
		23:16								
		31:24								
0x018C	GMAC_JR	7:0	JRX[7:0]							
		15:8							JRX[9:8]	
		23:16								
		31:24								
0x0190	GMAC_FCSE	7:0	FCKR[7:0]							
		15:8							FCKR[9:8]	
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0194	GMAC_LFFE	7:0	LFER[7:0]							
		15:8							LFER[9:8]	
		23:16								
		31:24								
0x0198	GMAC_RSE	7:0	RXSE[7:0]							
		15:8							RXSE[9:8]	
		23:16								
		31:24								
0x019C	GMAC_AE	7:0	AER[7:0]							
		15:8							AER[9:8]	
		23:16								
		31:24								
0x01A0	GMAC_RRE	7:0	RXRER[7:0]							
		15:8	RXRER[15:8]							
		23:16							RXRER[17:16]	
		31:24								
0x01A4	GMAC_ROE	7:0	RXOVR[7:0]							
		15:8							RXOVR[9:8]	
		23:16								
		31:24								
0x01A8	GMAC_IHCE	7:0	HCKER[7:0]							
		15:8								
		23:16								
		31:24								
0x01AC	GMAC_TCE	7:0	TCKER[7:0]							
		15:8								
		23:16								
		31:24								
0x01B0	GMAC_UCE	7:0	UCKER[7:0]							
		15:8								
		23:16								
		31:24								
0x01B4 ... 0x01BB	Reserved									
0x01BC	GMAC_TISUBN	7:0	LSBTIR[7:0]							
		15:8	LSBTIR[15:8]							
		23:16								
		31:24								
0x01C0	GMAC_TSH	7:0	TCS[7:0]							
		15:8	TCS[15:8]							
		23:16								
		31:24								
0x01C4 ... 0x01CF	Reserved									
0x01D0	GMAC_TSL	7:0	TCS[7:0]							
		15:8	TCS[15:8]							
		23:16	TCS[23:16]							
		31:24	TCS[31:24]							
0x01D4	GMAC_TN	7:0	TNS[7:0]							
		15:8	TNS[15:8]							
		23:16	TNS[23:16]							
		31:24			TNS[29:24]					
0x01D8	GMAC_TA	7:0	ITDT[7:0]							
		15:8	ITDT[15:8]							
		23:16	ITDT[23:16]							
		31:24	ADJ		ITDT[29:24]					

.....continued

Offset	Name	Bit Pos.								
0x01DC	GMAC_TI	7:0	CNS[7:0]							
		15:8	ACNS[7:0]							
		23:16	NIT[7:0]							
		31:24								
0x01E0	GMAC_EFTSL	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16	RUD[23:16]							
		31:24	RUD[31:24]							
0x01E4	GMAC_EFTN	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16	RUD[23:16]							
		31:24			RUD[29:24]					
0x01E8	GMAC_EFRSL	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16	RUD[23:16]							
		31:24	RUD[31:24]							
0x01EC	GMAC_EFRN	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16	RUD[23:16]							
		31:24			RUD[29:24]					
0x01F0	GMAC_PEFTSL	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16	RUD[23:16]							
		31:24	RUD[31:24]							
0x01F4	GMAC_PEFTN	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16	RUD[23:16]							
		31:24			RUD[29:24]					
0x01F8	GMAC_PEFRSL	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16	RUD[23:16]							
		31:24	RUD[31:24]							
0x01FC	GMAC_PEFRN	7:0	RUD[7:0]							
		15:8	RUD[15:8]							
		23:16	RUD[23:16]							
		31:24			RUD[29:24]					
0x0200 ... 0x026F	Reserved									
0x0270	GMAC_RXLPI	7:0	COUNT[7:0]							
		15:8	COUNT[15:8]							
		23:16								
		31:24								
0x0274	GMAC_RXLPITIME	7:0	LPITIME[7:0]							
		15:8	LPITIME[15:8]							
		23:16	LPITIME[23:16]							
		31:24								
0x0278	GMAC_TXLPI	7:0	COUNT[7:0]							
		15:8	COUNT[15:8]							
		23:16								
		31:24								
0x027C	GMAC_TXLPITIME	7:0	LPITIME[7:0]							
		15:8	LPITIME[15:8]							
		23:16	LPITIME[23:16]							
		31:24								
0x0280 ... 0x03FF	Reserved									

.....continued

Offset	Name	Bit Pos.								
0x0400	GMAC_ISRPQ1	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0404	GMAC_ISRPQ2	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0408	GMAC_ISRPQ3	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x040C	GMAC_ISRPQ4	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0410	GMAC_ISRPQ5	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0414 ... 0x043F	Reserved									
0x0440	GMAC_TBQBAPQ1	7:0	TXBQBA[5:0]							
		15:8	TXBQBA[13:6]							
		23:16	TXBQBA[21:14]							
		31:24	TXBQBA[29:22]							
0x0444	GMAC_TBQBAPQ2	7:0	TXBQBA[5:0]							
		15:8	TXBQBA[13:6]							
		23:16	TXBQBA[21:14]							
		31:24	TXBQBA[29:22]							
0x0448	GMAC_TBQBAPQ3	7:0	TXBQBA[5:0]							
		15:8	TXBQBA[13:6]							
		23:16	TXBQBA[21:14]							
		31:24	TXBQBA[29:22]							
0x044C	GMAC_TBQBAPQ4	7:0	TXBQBA[5:0]							
		15:8	TXBQBA[13:6]							
		23:16	TXBQBA[21:14]							
		31:24	TXBQBA[29:22]							
0x0450	GMAC_TBQBAPQ5	7:0	TXBQBA[5:0]							
		15:8	TXBQBA[13:6]							
		23:16	TXBQBA[21:14]							
		31:24	TXBQBA[29:22]							
0x0454 ... 0x047F	Reserved									
0x0480	GMAC_RBQBAPQ1	7:0	RXBQBA[5:0]							
		15:8	RXBQBA[13:6]							
		23:16	RXBQBA[21:14]							
		31:24	RXBQBA[29:22]							
0x0484	GMAC_RBQBAPQ2	7:0	RXBQBA[5:0]							
		15:8	RXBQBA[13:6]							
		23:16	RXBQBA[21:14]							
		31:24	RXBQBA[29:22]							
0x0488	GMAC_RBQBAPQ3	7:0	RXBQBA[5:0]							
		15:8	RXBQBA[13:6]							
		23:16	RXBQBA[21:14]							
		31:24	RXBQBA[29:22]							

.....continued

Offset	Name	Bit Pos.								
0x048C	GMAC_RBQBAPQ4	7:0	RXBQBA[5:0]							
		15:8	RXBQBA[13:6]							
		23:16	RXBQBA[21:14]							
		31:24	RXBQBA[29:22]							
0x0490	GMAC_RBQBAPQ5	7:0	RXBQBA[5:0]							
		15:8	RXBQBA[13:6]							
		23:16	RXBQBA[21:14]							
		31:24	RXBQBA[29:22]							
0x0494 ... 0x049F	Reserved									
0x04A0	GMAC_RBSRPQ1	7:0	RBS[7:0]							
		15:8	RBS[15:8]							
		23:16								
		31:24								
0x04A4	GMAC_RBSRPQ2	7:0	RBS[7:0]							
		15:8	RBS[15:8]							
		23:16								
		31:24								
0x04A8	GMAC_RBSRPQ3	7:0	RBS[7:0]							
		15:8	RBS[15:8]							
		23:16								
		31:24								
0x04AC	GMAC_RBSRPQ4	7:0	RBS[7:0]							
		15:8	RBS[15:8]							
		23:16								
		31:24								
0x04B0	GMAC_RBSRPQ5	7:0	RBS[7:0]							
		15:8	RBS[15:8]							
		23:16								
		31:24								
0x04B4 ... 0x04BB	Reserved									
0x04BC	GMAC_CBSCR	7:0							QBE	QAE
		15:8								
		23:16								
		31:24								
0x04C0	GMAC_CBSISQA	7:0	IS[7:0]							
		15:8	IS[15:8]							
		23:16	IS[23:16]							
		31:24	IS[31:24]							
0x04C4	GMAC_CBSISQB	7:0	IS[7:0]							
		15:8	IS[15:8]							
		23:16	IS[23:16]							
		31:24	IS[31:24]							
0x04C8 ... 0x04FF	Reserved									
0x0500	GMAC_ST1RPQ0	7:0	DSTCM[3:0]					QNB[2:0]		
		15:8	UDPM[3:0]				DSTCM[7:4]			
		23:16	UDPM[11:4]							
		31:24			UDPE	DSTCE	UDPM[15:12]			
0x0504	GMAC_ST1RPQ1	7:0	DSTCM[3:0]					QNB[2:0]		
		15:8	UDPM[3:0]				DSTCM[7:4]			
		23:16	UDPM[11:4]							
		31:24			UDPE	DSTCE	UDPM[15:12]			

.....continued

Offset	Name	Bit Pos.								
0x0508	GMAC_ST1RPQ2	7:0	DSTCM[3:0]					QNB[2:0]		
		15:8	UDPM[3:0]				DSTCM[7:4]			
		23:16	UDPM[11:4]							
		31:24			UDPE	DSTCE	UDPM[15:12]			
0x050C	GMAC_ST1RPQ3	7:0	DSTCM[3:0]					QNB[2:0]		
		15:8	UDPM[3:0]				DSTCM[7:4]			
		23:16	UDPM[11:4]							
		31:24			UDPE	DSTCE	UDPM[15:12]			
0x0510 ... 0x053F	Reserved									
0x0540	GMAC_ST2RPQ0	7:0		VLANP[2:0]				QNB[2:0]		
		15:8	COMPA[2:0]			ETHE	I2ETH[2:0]			VLANE
		23:16	COMPB[4:0]					COMPAE	COMPA[4:3]	
		31:24		COMPCE	COMPC[4:0]					COMPBE
0x0544	GMAC_ST2RPQ1	7:0		VLANP[2:0]				QNB[2:0]		
		15:8	COMPA[2:0]			ETHE	I2ETH[2:0]			VLANE
		23:16	COMPB[4:0]					COMPAE	COMPA[4:3]	
		31:24		COMPCE	COMPC[4:0]					COMPBE
0x0548	GMAC_ST2RPQ2	7:0		VLANP[2:0]				QNB[2:0]		
		15:8	COMPA[2:0]			ETHE	I2ETH[2:0]			VLANE
		23:16	COMPB[4:0]					COMPAE	COMPA[4:3]	
		31:24		COMPCE	COMPC[4:0]					COMPBE
0x054C	GMAC_ST2RPQ3	7:0		VLANP[2:0]				QNB[2:0]		
		15:8	COMPA[2:0]			ETHE	I2ETH[2:0]			VLANE
		23:16	COMPB[4:0]					COMPAE	COMPA[4:3]	
		31:24		COMPCE	COMPC[4:0]					COMPBE
0x0550	GMAC_ST2RPQ4	7:0		VLANP[2:0]				QNB[2:0]		
		15:8	COMPA[2:0]			ETHE	I2ETH[2:0]			VLANE
		23:16	COMPB[4:0]					COMPAE	COMPA[4:3]	
		31:24		COMPCE	COMPC[4:0]					COMPBE
0x0554	GMAC_ST2RPQ5	7:0		VLANP[2:0]				QNB[2:0]		
		15:8	COMPA[2:0]			ETHE	I2ETH[2:0]			VLANE
		23:16	COMPB[4:0]					COMPAE	COMPA[4:3]	
		31:24		COMPCE	COMPC[4:0]					COMPBE
0x0558	GMAC_ST2RPQ6	7:0		VLANP[2:0]				QNB[2:0]		
		15:8	COMPA[2:0]			ETHE	I2ETH[2:0]			VLANE
		23:16	COMPB[4:0]					COMPAE	COMPA[4:3]	
		31:24		COMPCE	COMPC[4:0]					COMPBE
0x055C	GMAC_ST2RPQ7	7:0		VLANP[2:0]				QNB[2:0]		
		15:8	COMPA[2:0]			ETHE	I2ETH[2:0]			VLANE
		23:16	COMPB[4:0]					COMPAE	COMPA[4:3]	
		31:24		COMPCE	COMPC[4:0]					COMPBE
0x0560 ... 0x05FF	Reserved									
0x0600	GMAC_IERPQ1	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0604	GMAC_IERPQ2	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0608	GMAC_IERPQ3	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x060C	GMAC_IERPQ4	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0610	GMAC_IERPQ5	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0614 ... 0x061F	Reserved									
0x0620	GMAC_IDRPQ1	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0624	GMAC_IDRPQ2	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0628	GMAC_IDRPQ3	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x062C	GMAC_IDRPQ4	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0630	GMAC_IDRPQ5	7:0	TCOMP	TFC	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0634 ... 0x063F	Reserved									
0x0640	GMAC_IMRPQ1	7:0	TCOMP	AHB	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0644	GMAC_IMRPQ2	7:0	TCOMP	AHB	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0648	GMAC_IMRPQ3	7:0	TCOMP	AHB	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x064C	GMAC_IMRPQ4	7:0	TCOMP	AHB	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0650	GMAC_IMRPQ5	7:0	TCOMP	AHB	RLEX			RXUBR	RCOMP	
		15:8					HRESP	ROVR		
		23:16								
		31:24								
0x0654 ... 0x06DF	Reserved									

.....continued

Offset	Name	Bit Pos.								
0x06E0	GMAC_ST2ER0	7:0	COMPVAL[7:0]							
		15:8	COMPVAL[15:8]							
		23:16								
		31:24								
0x06E4	GMAC_ST2ER1	7:0	COMPVAL[7:0]							
		15:8	COMPVAL[15:8]							
		23:16								
		31:24								
0x06E8	GMAC_ST2ER2	7:0	COMPVAL[7:0]							
		15:8	COMPVAL[15:8]							
		23:16								
		31:24								
0x06EC	GMAC_ST2ER3	7:0	COMPVAL[7:0]							
		15:8	COMPVAL[15:8]							
		23:16								
		31:24								
0x06F0 ... 0x06FF	Reserved									
0x0700	GMAC_ST2CW00	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0704	GMAC_ST2CW10	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0708	GMAC_ST2CW01	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x070C	GMAC_ST2CW11	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0710	GMAC_ST2CW02	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0714	GMAC_ST2CW12	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0718	GMAC_ST2CW03	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x071C	GMAC_ST2CW13	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0720	GMAC_ST2CW04	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0724	GMAC_ST2CW14	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0728	GMAC_ST2CW05	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x072C	GMAC_ST2CW15	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0730	GMAC_ST2CW06	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0734	GMAC_ST2CW16	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0738	GMAC_ST2CW07	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x073C	GMAC_ST2CW17	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0740	GMAC_ST2CW08	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0744	GMAC_ST2CW18	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0748	GMAC_ST2CW09	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x074C	GMAC_ST2CW19	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0750	GMAC_ST2CW010	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0754	GMAC_ST2CW110	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0758	GMAC_ST2CW011	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x075C	GMAC_ST2CW111	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0760	GMAC_ST2CW012	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0764	GMAC_ST2CW112	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0768	GMAC_ST2CW013	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x076C	GMAC_ST2CW113	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0770	GMAC_ST2CW014	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0774	GMAC_ST2CW114	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0778	GMAC_ST2CW015	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x077C	GMAC_ST2CW115	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0780	GMAC_ST2CW016	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0784	GMAC_ST2CW116	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0788	GMAC_ST2CW017	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x078C	GMAC_ST2CW117	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x0790	GMAC_ST2CW018	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x0794	GMAC_ST2CW118	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0798	GMAC_ST2CW019	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x079C	GMAC_ST2CW119	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x07A0	GMAC_ST2CW020	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x07A4	GMAC_ST2CW120	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x07A8	GMAC_ST2CW021	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x07AC	GMAC_ST2CW121	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x07B0	GMAC_ST2CW022	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x07B4	GMAC_ST2CW122	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								
0x07B8	GMAC_ST2CW023	7:0	MASKVAL[7:0]							
		15:8	MASKVAL[15:8]							
		23:16	COMPVAL[7:0]							
		31:24	COMPVAL[15:8]							
0x07BC	GMAC_ST2CW123	7:0	OFFSSTRT[0]	OFFSVAL[6:0]						
		15:8								OFFSSTRT[1]
		23:16								
		31:24								

36.8.1 GMAC Network Control Register

Name: GMAC_NCR
Offset: 0x000
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					TXLPIEN	FNP	TXPBPF	ENPBPR
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SRTSM			TXZQPF	TXPF	THALT	TSTART	BP
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	WESTAT	INCSTAT	CLRSTAT	MPE	TXEN	RXEN	LBL	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	

Bit 19 – TXLPIEN Enable LPI Transmission
When set, LPI (low power idle) is immediately transmitted.

Bit 18 – FNP Flush Next Packet
Flush the next packet from the external RX DPRAM. Writing one to this bit will only have an effect if the DMA is not currently writing a packet already stored in the DPRAM to memory.

Bit 17 – TXPBPF Transmit PFC Priority-based Pause Frame
Takes the values stored in the Transmit PFC Pause Register.

Bit 16 – ENPBPR Enable PFC Priority-based Pause Reception
Enables PFC Priority Based Pause Reception capabilities. Setting this bit enables PFC negotiation and recognition of priority-based pause frames.

Bit 15 – SRTSM Store Receive Timestamp to Memory

Value	Description
0	Normal operation.
1	Causes the CRC of every received frame to be replaced with the value of the nanoseconds field of the 1588 timer that was captured as the receive frame passed the message timestamp point. Note that bit RFCS in register GMAC_NCFGR may not be set to 1 when the timer should be captured.

Bit 12 – TXZQPF Transmit Zero Quantum Pause Frame
Writing one to this bit causes a pause frame with zero quantum to be transmitted.

Bit 11 – TXPF Transmit Pause Frame
Writing one to this bit causes a pause frame to be transmitted.

Bit 10 – THALT Transmit Halt
Writing one to this bit halts transmission as soon as any ongoing frame transmission ends.

Bit 9 – TSTART Start Transmission
Writing one to this bit starts transmission.

Bit 8 – BP Back pressure
If set in 10M or 100M half duplex mode, forces collisions on all received frames.

Bit 7 – WESTAT Write Enable for Statistics Registers
Setting this bit to one makes the statistics registers writable for functional test purposes.

Bit 6 – INCSTAT Increment Statistics Registers
This bit is write-only. Writing a one increments all the statistics registers by one for test purposes.

Bit 5 – CLRSTAT Clear Statistics Registers
This bit is write-only. Writing a one clears the statistics registers.

Bit 4 – MPE Management Port Enable
Set to one to enable the management port. When zero, forces GMDIO to high impedance state and MDC low.

Bit 3 – TXEN Transmit Enable
When set, TXEN enables the GMAC transmitter to send data. When reset transmission will stop immediately, the transmit pipeline and control registers will be cleared and the Transmit Queue Pointer Register will reset to point to the start of the transmit descriptor list.

Bit 2 – RXEN Receive Enable
When set, RXEN enables the GMAC to receive data. When reset frame reception stops immediately and the receive pipeline will be cleared. The Receive Queue Pointer Register is unaffected.

Bit 1 – LBL Loop Back Local
Connects GTX to GRX, GTXEN to GRXDV and forces full duplex mode. GRXCK and GTXCK may malfunction as the GMAC is switched into and out of internal loop back. It is important that receive and transmit circuits have already been disabled when making the switch into and out of internal loop back.

36.8.2 GMAC Network Configuration Register

Name: GMAC_NCFGR
Offset: 0x004
Reset: 0x00080000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
		IRXER	RXBP	IPGSEN		IRXFCS	EFRHD	RXCOEN
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	23	22	21	20	19	18	17	16
	DCPF	DBW[1:0]		CLK[2:0]			RFCS	LFERD
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	0	0	0

Bit	15	14	13	12	11	10	9	8
	RXBUFO[1:0]		PEN	RTY				MAXFS
Access	R/W	R/W	R/W	R/W				R/W
Reset	0	0	0	0				0

Bit	7	6	5	4	3	2	1	0
	UNIHEN	MTIHEN	NBC	CAF	JFRAME	DNVLAN	FD	SPD
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 30 – IRXER Ignore IPG GRXER
When set, GRXER has no effect on the GMAC's operation when GRXDV is low.

Bit 29 – RXBP Receive Bad Preamble
When set, frames with non-standard preamble are not rejected.

Bit 28 – IPGSEN IP Stretch Enable
When set, the transmit IPG can be increased above 96 bit times depending on the previous frame length using the IPG Stretch Register.

Bit 26 – IRXFCS Ignore RX FCS
When set, frames with FCS/CRC errors will not be rejected. FCS error statistics will still be collected for frames with bad FCS and FCS status will be recorded in frame’s DMA descriptor. For normal operation this bit must be set to zero.

Bit 25 – EFRHD Enable Frames Received in Half Duplex
Enable frames to be received in half-duplex mode while transmitting.

Bit 24 – RXCOEN Receive Checksum Offload Enable
When set, the receive checksum engine is enabled. Frames with bad IP, TCP or UDP checksums are discarded.

Bit 23 – DCPF Disable Copy of Pause Frames
Set to one to prevent valid pause frames being copied to memory. When set, pause frames are not copied to memory regardless of the state of the Copy All Frames bit, whether a hash match is found or whether a type ID match is identified. If a destination address match is found, the pause frame will be copied to memory. Note that valid pause frames received will still increment pause statistics and pause the transmission of frames as required.

Bits 22:21 – DBW[1:0] Data Bus Width
Should always be written to ‘0’.

Bits 20:18 – CLK[2:0] MDC CLock Division
Set according to MCK speed. These three bits determine the number MCK will be divided by to generate Management Data Clock (MDC). For conformance with the 802.3 specification, MDC must not exceed 2.5 MHz (MDC is only active during MDIO read and write operations).

Value	Name	Description
0	MCK_8	MCK divided by 8 (MCK up to 20 MHz)
1	MCK_16	MCK divided by 16 (MCK up to 40 MHz)
2	MCK_32	MCK divided by 32 (MCK up to 80 MHz)
3	MCK_48	MCK divided by 48 (MCK up to 120 MHz)
4	MCK_64	MCK divided by 64 (MCK up to 160 MHz)
5	MCK_96	MCK divided by 96 (MCK up to 240 MHz)

Bit 17 – RFCS Remove FCS
Setting this bit will cause received frames to be written to memory without their frame check sequence (last 4 bytes). The frame length indicated will be reduced by four bytes in this mode.

Bit 16 – LFERD Length Field Error Frame Discard
Setting this bit causes frames with a measured length shorter than the extracted length field (as indicated by bytes 13 and 14 in a non-VLAN tagged frame) to be discarded. This only applies to frames with a length field less than 0x0600.

Bits 15:14 – RXBUFO[1:0] Receive Buffer Offset
Indicates the number of bytes by which the received data is offset from the start of the receive buffer

Bit 13 – PEN Pause Enable
When set, transmission will pause if a non-zero 802.3 classic pause frame is received and PFC has not been negotiated.

Bit 12 – RTY Retry Test
Must be set to zero for normal operation. If set to one the backoff between collisions will always be one slot time. Setting this bit to one helps test the too many retries condition. Also used in the pause frame tests to reduce the pause counter's decrement time from 512 bit times, to every GRXCK cycle.

Bit 8 – MAXFS 1536 Maximum Frame Size
Setting this bit means the GMAC will accept frames up to 1536 bytes in length. Normally the GMAC would reject any frame above 1518 bytes.

Bit 7 – UNIHEN Unicast Hash Enable
When set, unicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash Register.

Bit 6 – MTIHEN Multicast Hash Enable
When set, multicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash Register.

Bit 5 – NBC No Broadcast
When set to logic one, frames addressed to the broadcast address of all ones will not be accepted.

Bit 4 – CAF Copy All Frames
When set to logic one, all valid frames will be accepted.

Bit 3 – JFRAME Jumbo Frame Size
Set to one to enable jumbo frames up to 10240 bytes to be accepted. The default length is 10240 bytes.

Bit 2 – DNVLAN Discard Non-VLAN FRAMES
When set only VLAN tagged frames will be passed to the address matching logic.

Bit 1 – FD Full Duplex
If set to logic one, the transmit block ignores the state of collision and carrier sense and allows receive while transmitting.

Bit 0 – SPD Speed
Set to logic one to indicate 100 Mbps operation, logic zero for 10 Mbps.

36.8.3 GMAC Network Status Register

Name: GMAC_NSR
Offset: 0x008
Reset: see Note
Property: Read-only

Note: The register reset value depends on the status of the GMDIO input pin.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RXLPIS					IDLE	MDIO	
Access	R					R	R	
Reset	0					1	–	

Bit 7 – RXLPIS LPI Indication
Low power idle has been detected on receive. This bit is set when LPI is detected and reset when normal idle is detected. An interrupt is generated when the state of this bit changes.

Bit 2 – IDLE PHY Management Logic Idle
The PHY management logic is idle (i.e., has completed).

Bit 1 – MDIO MDIO Input Status
Returns status of the GMDIO pin.

36.8.4 GMAC User Register

Name: GMAC_UR
Offset: 0x00C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								RMII
Access								R/W
Reset								0

Bit 0 – RMII Reduced MII Mode

Value	Description
0	RMII mode is selected (default).
1	MII mode is selected.

36.8.5 GMAC DMA Configuration Register

Name: GMAC_DCFGR
Offset: 0x010
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
								DDRP
Access								R/W
Reset								0

Bit	23	22	21	20	19	18	17	16
	DRBS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
					TXCOEN	TXPBMS	RXBMS[1:0]	
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ESPA	ESMA		FBLDO[4:0]				
Access	R/W	R/W		R/W	R/W	R/W	R/W	R/W
Reset	0	0		0	0	0	0	0

Bit 24 – DDRP DMA Discard Receive Packets
When set, the GMAC DMA will automatically discard receive packets from the receiver packet buffer memory when no AHB resource is available.
When low, the received packets will remain to be stored in the SRAM based packet buffer until AHB buffer resource next becomes available.
A write to this bit is ignored if the DMA is not configured in the packet buffer full store and forward mode.

Bits 23:16 – DRBS[7:0] DMA Receive Buffer Size
DMA receive buffer size in AHB system memory. The value defined by these bits determines the size of buffer to use in main AHB system memory when writing received data.
The value is defined in multiples of 64 bytes, thus a value of 0x01 corresponds to buffers of 64 bytes, 0x02 corresponds to 128 bytes etc.
For example:
– 0x02: 128 bytes
– 0x18: 1536 bytes (1 × max length frame/buffer)
– 0xA0: 10240 bytes (1 × 10K jumbo frame/buffer)
Note that this value should never be written as zero.

Bit 11 – TXCOEN Transmitter Checksum Generation Offload Enable
Transmitter IP, TCP and UDP checksum generation offload enable. When set, the transmitter checksum generation engine is enabled to calculate and substitute checksums for transmit frames. When clear, frame data is unaffected.

Bit 10 – TXPBMS Transmitter Packet Buffer Memory Size Select
Having this bit at zero halves the amount of memory used for the transmit packet buffer. This reduces the amount of memory used by the GMAC. It is important to set this bit to one if the full configured physical memory is available.
The value in brackets below represents the size that would result for the default maximum configured memory size of 4 Kbytes.

Value	Description
0	Do not use top address bit (2 Kbytes).

Value	Description
1	Use full configured addressable space (4 Kbytes).

Bits 9:8 – RXBMS[1:0] Receiver Packet Buffer Memory Size Select
The default receive packet buffer size is 4 Kbytes. The table below shows how to configure this memory to FULL, HALF, QUARTER or EIGHTH of the default size.

Value	Name	Description
0	EIGHTH	4/8 Kbyte Memory Size
1	QUARTER	4/4 Kbytes Memory Size
2	HALF	4/2 Kbytes Memory Size
3	FULL	4 Kbytes Memory Size

Bit 7 – ESPA Endian Swap Mode Enable for Packet Data Accesses
When set, selects swapped endianism for AHB transfers. When clear, selects little endian mode.

Bit 6 – ESMA Endian Swap Mode Enable for Management Descriptor Accesses
When set, selects swapped endianism for AHB transfers. When clear, selects little endian mode.

Bits 4:0 – FBLDO[4:0] Fixed Burst Length for DMA Data Operations
Selects the burst length to attempt to use on the AHB when transferring frame data. Not used for DMA management operations and only used where space and data size allow. Otherwise SINGLE type AHB transfers are used.
One-hot priority encoding enforced automatically on register writes as follows, where ‘x’ represents don’t care:

Value	Name	Description
0	–	Reserved
1	SINGLE	00001: Always use SINGLE AHB bursts
2	–	Reserved
4	INCR4	001xx: Attempt to use INCR4 AHB bursts (Default)
8	INCR8	01xxx: Attempt to use INCR8 AHB bursts
16	INCR16	1xxxx: Attempt to use INCR16 AHB bursts

36.8.6 GMAC Transmit Status Register

Name: GMAC_TSR
Offset: 0x014
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								HRESP
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
			TXCOMP	TFC	TXGO	RLE	COL	UBR
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit 8 – HRESP HRESP Not OK
Set when the DMA block sees HRESP not OK. Writing a one clears this bit.

Bit 5 – TXCOMP Transmit Complete
Set when a frame has been transmitted. Writing a one clears this bit.

Bit 4 – TFC Transmit Frame Corruption Due to AHB Error
Transmit frame corruption due to AHB error. Set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and GTXER asserted).
Also set in DMA packet buffer mode if single frame is too large for configured packet buffer memory size.
Writing a one clears this bit.

Bit 3 – TXGO Transmit Go
Transmit go, if high transmit is active. When using the DMA interface, this bit represents the TXGO variable as specified in the transmit buffer description.

Bit 2 – RLE Retry Limit Exceeded
Writing a one clears this bit.

Bit 1 – COL Collision Occurred
Set by the assertion of collision. Writing a one clears this bit. When operating in 10/100 mode, this status indicates either a collision or a late collision.

Bit 0 – UBR Used Bit Read
Set when a transmit buffer descriptor is read with its used bit set. Writing a one clears this bit.

36.8.7 GMAC Receive Buffer Queue Base Address Register

Name: GMAC_RBQB
Offset: 0x018
Reset: 0x00000000
Property: Read/Write

This register holds the start address of the receive buffer queue (receive buffers descriptor list). The receive buffer queue base address must be initialized before receive is enabled through bit 2 of the Network Control Register. Once reception is enabled, any write to the Receive Buffer Queue Base Address Register is ignored. Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used. Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the “used” bits.

In terms of AMBA AHB operation, the descriptors are read from memory using a single 32-bit AHB access. The descriptors should be aligned at 32-bit boundaries and the descriptors are written to using two individual non sequential accesses.

Bit	31	30	29	28	27	26	25	24
	ADDR[29:22]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[21:14]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bits 31:2 – ADDR[29:0] Receive Buffer Queue Base Address
Written with the address of the start of the receive queue.

36.8.8 GMAC Transmit Buffer Queue Base Address Register

Name: GMAC_TBQB
Offset: 0x01C
Reset: 0x00000000
Property: Read/Write

This register holds the start address of the transmit buffer queue (transmit buffers descriptor list). The Transmit Buffer Queue Base Address Register must be initialized before transmit is started through bit 9 of the Network Control Register. Once transmission has started, any write to the Transmit Buffer Queue Base Address Register is illegal and therefore ignored.

Note that due to clock boundary synchronization, it takes a maximum of four MCK cycles from the writing of the transmit start bit before the transmitter is active. Writing to the Transmit Buffer Queue Base Address Register during this time may produce unpredictable results.

Reading this register returns the location of the descriptor currently being accessed. Since the DMA handles two frames at once, this may not necessarily be pointing to the current frame being transmitted.

In terms of AMBA AHB operation, the descriptors are written to memory using a single 32-bit AHB access. The descriptors should be aligned at 32-bit boundaries and the descriptors are read from memory using two individual non sequential accesses.

Bit	31	30	29	28	27	26	25	24
	ADDR[29:22]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[21:14]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bits 31:2 – ADDR[29:0] Transmit Buffer Queue Base Address
Written with the address of the start of the transmit queue.

36.8.9 GMAC Receive Status Register

Name: GMAC_RSR
Offset: 0x020
Reset: 0x00000000
Property: Read/Write

This register, when read, provides receive status details. Once read, individual bits may be cleared by writing a one to them. It is not possible to set a bit to 1 by writing to the register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					HNO	RXOVR	REC	BNA
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit 3 – HNO HRESP Not OK
Set when the DMA block sees HRESP not OK. Writing a one clears this bit.

Bit 2 – RXOVR Receive Overrun
This bit is set if the receive status was not taken at the end of the frame. This bit is also set if the packet buffer overflows. The buffer will be recovered if an overrun occurs. Writing a one clears this bit.

Bit 1 – REC Frame Received
One or more frames have been received and placed in memory. Writing a one clears this bit.

Bit 0 – BNA Buffer Not Available
An attempt was made to get a new buffer and the pointer indicated that it was owned by the processor. The DMA will re-read the pointer each time an end of frame is received until a valid pointer is found. This bit is set following each descriptor read attempt that fails, even if consecutive pointers are unsuccessful and software has in the mean time cleared the status flag. Writing a one clears this bit.

36.8.10 GMAC Interrupt Status Register

Name: GMAC_ISR
Offset: 0x024
Reset: 0x00000000
Property: Read-only

This register indicates the source of the interrupt. In order that the bits of this register read 1, the corresponding interrupt source must be enabled in the mask register. If any bit is set in this register, the GMAC interrupt signal will be asserted in the system.

Bit	31	30	29	28	27	26	25	24
			TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR		
Access	R	R	R	R	R	R		
Reset	0	0	0	0	0	0		

Bit	15	14	13	12	11	10	9	8
		PFTR	PTZ	PFNZ	HRESP	ROVR		
Access		R	R	R	R	R		
Reset		0	0	0	0	0		

Bit	7	6	5	4	3	2	1	0
	TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 29 – TSUTIMCOMP TSU Timer Comparison
Indicates when TSU timer count value is equal to programmed value. Cleared on read.

Bit 28 – WOL Wake On LAN
WOL interrupt. Indicates a WOL event has been received.

Bit 27 – RXLPISBC Receive LPI indication Status Bit Change
Receive LPI indication status bit change. Cleared on read.

Bit 26 – SRI TSU Seconds Register Increment
Indicates the register has incremented. Cleared on read.

Bit 25 – PDRSFT PDelay Response Frame Transmitted
Indicates a PTP pdelay_resp frame has been transmitted. Cleared on read.

Bit 24 – PDRQFT PDelay Request Frame Transmitted
Indicates a PTP pdelay_req frame has been transmitted. Cleared on read.

Bit 23 – PDRSFR PDelay Response Frame Received
Indicates a PTP pdelay_resp frame has been received. Cleared on read.

Bit 22 – PDRQFR PDelay Request Frame Received
Indicates a PTP pdelay_req frame has been received. Cleared on read.

Bit 21 – SFT PTP Sync Frame Transmitted
Indicates a PTP sync frame has been transmitted. Cleared on read.

Bit 20 – DRQFT PTP Delay Request Frame Transmitted
Indicates a PTP delay_req frame has been transmitted. Cleared on read.

Bit 19 – SFR PTP Sync Frame Received
Indicates a PTP sync frame has been received. Cleared on read.

Bit 18 – DRQFR PTP Delay Request Frame Received
Indicates a PTP delay_req frame has been received. Cleared on read.

Bit 14 – PFTR Pause Frame Transmitted
Indicates a pause frame has been successfully transmitted after being initiated from the Network Control register. Cleared on read.

Bit 13 – PTZ Pause Time Zero
Set when either the Pause Time register at address 0x38 decrements to zero, or when a valid pause frame is received with a zero pause quantum field. Cleared on read.

Bit 12 – PFNZ Pause Frame with Non-zero Pause Quantum Received
Indicates a valid pause has been received that has a non-zero pause quantum field. Cleared on read.

Bit 11 – HRESP HRESP Not OK
Set when the DMA block sees HRESP not OK. Cleared on read.

Bit 10 – ROVR Receive Overrun
Set when the receive overrun status bit is set. Cleared on read.

Bit 7 – TCOMP Transmit Complete
Set when a frame has been transmitted. Cleared on read.

Bit 6 – TFC Transmit Frame Corruption Due to AHB Error
Transmit frame corruption due to AHB error. Set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame.

Bit 5 – RLEX Retry Limit Exceeded
Transmit error. Cleared on read.

Bit 4 – TUR Transmit Underrun
This interrupt is set if the transmitter was forced to terminate a frame that it has already began transmitting due to further data being unavailable.
This interrupt is set if a transmitter status write back has not completed when another status write back is attempted.
This interrupt is also set when the transmit DMA has written the SOP data into the FIFO and either the AHB bus was not granted in time for further data, or because an AHB not OK response was returned, or because the used bit was read.

Bit 3 – TXUBR TX Used Bit Read
Set when a transmit buffer descriptor is read with its used bit set. Cleared on read.

Bit 2 – RXUBR RX Used Bit Read
Set when a receive buffer descriptor is read with its used bit set. Cleared on read.

Bit 1 – RCOMP Receive Complete
A frame has been stored in memory. Cleared on read.

Bit 0 – MFS Management Frame Sent
The PHY Maintenance Register has completed its operation. Cleared on read.

36.8.11 GMAC Interrupt Enable Register

Name: GMAC_IER
Offset: 0x028
Reset: –
Property: Write-only

This register is write-only and when read will return zero.

The following values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
			TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
Access			W	W	W	W	W	W
Reset			–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR		
Access	W	W	W	W	W	W		
Reset	–	–	–	–	–	–		

Bit	15	14	13	12	11	10	9	8
	EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR		
Access	W	W	W	W	W	W		
Reset	–	–	–	–	–	–		

Bit	7	6	5	4	3	2	1	0
	TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 29 – TSUTIMCOMP TSU Timer Comparison

Bit 28 – WOL Wake On LAN

Bit 27 – RXLPISBC Enable RX LPI Indication

Bit 26 – SRI TSU Seconds Register Increment

Bit 25 – PDRSFT PDelay Response Frame Transmitted

Bit 24 – PDRQFT PDelay Request Frame Transmitted

Bit 23 – PDRSFR PDelay Response Frame Received

Bit 22 – PDRQFR PDelay Request Frame Received

Bit 21 – SFT PTP Sync Frame Transmitted

Bit 20 – DRQFT PTP Delay Request Frame Transmitted

Bit 19 – SFR PTP Sync Frame Received

Bit 18 – DRQFR PTP Delay Request Frame Received

Bit 15 – EXINT External Interrupt

Bit 14 – PFTR Pause Frame Transmitted

Bit 13 – PTZ Pause Time Zero

Bit 12 – PFNZ Pause Frame with Non-zero Pause Quantum Received

Bit 11 – HRESP HRESP Not OK

Bit 10 – ROVR Receive Overrun

Bit 7 – TCOMP Transmit Complete

Bit 6 – TFC Transmit Frame Corruption Due to AHB Error

Bit 5 – RLEX Retry Limit Exceeded or Late Collision

Bit 4 – TUR Transmit Underrun

Bit 3 – TXUBR TX Used Bit Read

Bit 2 – RXUBR RX Used Bit Read

Bit 1 – RCOMP Receive Complete

Bit 0 – MFS Management Frame Sent

36.8.12 GMAC Interrupt Disable Register

Name: GMAC_IDR
Offset: 0x02C
Reset: –
Property: Write-only

This register is write-only and when read will return zero.

The following values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
			TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
Access			W	W	W	W	W	W
Reset			–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR		
Access	W	W	W	W	W	W		
Reset	–	–	–	–	–	–		

Bit	15	14	13	12	11	10	9	8
	EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR		
Access	W	W	W	W	W	W		
Reset	–	–	–	–	–	–		

Bit	7	6	5	4	3	2	1	0
	TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 29 – TSUTIMCOMP TSU Timer Comparison

Bit 28 – WOL Wake On LAN

Bit 27 – RXLPISBC Enable RX LPI Indication

Bit 26 – SRI TSU Seconds Register Increment

Bit 25 – PDRSFT PDelay Response Frame Transmitted

Bit 24 – PDRQFT PDelay Request Frame Transmitted

Bit 23 – PDRSFR PDelay Response Frame Received

Bit 22 – PDRQFR PDelay Request Frame Received

Bit 21 – SFT PTP Sync Frame Transmitted

Bit 20 – DRQFT PTP Delay Request Frame Transmitted

Bit 19 – SFR PTP Sync Frame Received

Bit 18 – DRQFR PTP Delay Request Frame Received

Bit 15 – EXINT External Interrupt

Bit 14 – PFTR Pause Frame Transmitted

Bit 13 – PTZ Pause Time Zero

Bit 12 – PFNZ Pause Frame with Non-zero Pause Quantum Received

Bit 11 – HRESP HRESP Not OK

Bit 10 – ROVR Receive Overrun

Bit 7 – TCOMP Transmit Complete

Bit 6 – TFC Transmit Frame Corruption Due to AHB Error

Bit 5 – RLEX Retry Limit Exceeded or Late Collision

Bit 4 – TUR Transmit Underrun

Bit 3 – TXUBR TX Used Bit Read

Bit 2 – RXUBR RX Used Bit Read

Bit 1 – RCOMP Receive Complete

Bit 0 – MFS Management Frame Sent

36.8.13 GMAC Interrupt Mask Register

Name: GMAC_IMR
Offset: 0x030
Reset: 0x07FFFFFF
Property: Read/Write

The Interrupt Mask Register is a read-only register indicating which interrupts are masked. All bits are set at reset and can be reset individually by writing to the Interrupt Enable Register or set individually by writing to the Interrupt Disable Register. Having separate address locations for enable and disable saves the need for performing a read modify write when updating the Interrupt Mask Register.

For test purposes there is a write-only function to this register that allows the bits in the Interrupt Status Register to be set or cleared, regardless of the state of the mask register. A write to this register directly affects the state of the corresponding bit in the Interrupt Status Register, causing an interrupt to be generated if a 1 is written.

The following values are valid for all listed bit names of this register when read:

0: The corresponding interrupt is enabled.

1: The corresponding interrupt is not enabled.

Bit	31	30	29	28	27	26	25	24
			TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	1	1	1

Bit	23	22	21	20	19	18	17	16
	PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR		
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	1	1	1	1	1	1		

Bit	15	14	13	12	11	10	9	8
	EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR		
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	1	1	1	1	1	1		

Bit	7	6	5	4	3	2	1	0
	TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit 29 – TSUTIMCOMP TSU Timer Comparison

Bit 28 – WOL Wake On LAN

Bit 27 – RXLPISBC Enable RX LPI Indication

Bit 26 – SRI TSU Seconds Register Increment

Bit 25 – PDRSFT PDelay Response Frame Transmitted

Bit 24 – PDRQFT PDelay Request Frame Transmitted

Bit 23 – PDRSFR PDelay Response Frame Received

Bit 22 – PDRQFR PDelay Request Frame Received

Bit 21 – SFT PTP Sync Frame Transmitted

Bit 20 – DRQFT PTP Delay Request Frame Transmitted

Bit 19 – SFR PTP Sync Frame Received

Bit 18 – DRQFR PTP Delay Request Frame Received

Bit 15 – EXINT External Interrupt

Bit 14 – PFTR Pause Frame Transmitted

Bit 13 – PTZ Pause Time Zero

Bit 12 – PFNZ Pause Frame with Non-zero Pause Quantum Received

Bit 11 – HRESP HRESP Not OK

Bit 10 – ROVR Receive Overrun

Bit 7 – TCOMP Transmit Complete

Bit 6 – TFC Transmit Frame Corruption Due to AHB Error

Bit 5 – RLEX Retry Limit Exceeded

Bit 4 – TUR Transmit Underrun

Bit 3 – TXUBR TX Used Bit Read

Bit 2 – RXUBR RX Used Bit Read

Bit 1 – RCOMP Receive Complete

Bit 0 – MFS Management Frame Sent

36.8.14 GMAC PHY Maintenance Register

Name: GMAC_MAN
Offset: 0x034
Reset: 0x00000000
Property: Read/Write

The PHY Maintenance Register is implemented as a shift register. Writing to the register starts a shift operation which is signalled as complete when bit 2 is set in the Network Status Register. It takes about 2000 MCK cycles to complete, when MDC is set for MCK divide by 32 in the Network Configuration Register. An interrupt is generated upon completion.

During this time, the MSB of the register is output on the GMDIO pin and the LSB updated from the GMDIO pin with each MDC cycle. This causes transmission of a PHY management frame on the GMDIO pin. See Section 22.2.4.5 of the IEEE 802.3 standard.

Reading during the shift operation returns the current contents of the shift register. At the end of management operation, the bits will have shifted back to their original locations. For a read operation, the data bits are updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced.

The MDIO interface can read IEEE 802.3 clause 45 PHYs as well as clause 22 PHYs. To read clause 45 PHYs, bit 30 should be written with a 0 rather than a 1. To write clause 45 PHYs, bits 31:28 should be written as 0x0001. See the table below.

Table 36-16. Clause 22/Clause 45 PHYs Read/Write Access Configuration (GMAC_MAN Bits 31:28)

PHY	Access	Bit Value			
		WZO	CLTTO	OP[1]	OP[0]
Clause 22	Read	0	1	1	0
	Write	0	1	0	1
Clause 45	Read	0	0	1	1
	Write	0	0	0	1
	Read + Address	0	0	1	0

For a description of MDC generation, see GMAC Network Configuration Register.

Bit	31	30	29	28	27	26	25	24
	WZO	CLTTO	OP[1:0]		PHYA[4:1]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	PHYA[0]	REGA[4:0]					WTN[1:0]	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DATA[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – WZO Write ZERO
Must be written with 0.

Bit 30 – CLTTO Clause 22 Operation

Value	Description
0	Clause 45 operation
1	Clause 22 operation

Bits 29:28 – OP[1:0] Operation

Value	Description
01	Write
10	Read

Bits 27:23 – PHYA[4:0] PHY Address

Bits 22:18 – REGA[4:0] Register Address
Specifies the register in the PHY to access.

Bits 17:16 – WTN[1:0] Write Ten
Must be written to 10.

Bits 15:0 – DATA[15:0] PHY Data
For a write operation this field is written with the data to be written to the PHY. After a read operation this field contains the data read from the PHY.

36.8.15 GMAC Receive Pause Quantum Register

Name: GMAC_RPQ
Offset: 0x038
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RPQ[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RPQ[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – RPQ[15:0] Received Pause Quantum
Stores the current value of the Receive Pause Quantum Register which is decremented every 512 bit times.

36.8.16 GMAC Transmit Pause Quantum Register

Name: GMAC_TPQ
Offset: 0x03C
Reset: 0x0000FFFF
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TPQ[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	TPQ[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 15:0 – TPQ[15:0] Transmit Pause Quantum
Written with the pause quantum value for pause frame transmission.

36.8.17 GMAC TX Partial Store and Forward Register

Name: GMAC_TPSF
Offset: 0x040
Reset: 0x00000FFF
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ENTXP							
Access	R/W							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					TPB1ADR[11:8]			
Access					R/W	R/W	R/W	R/W
Reset					1	1	1	1

Bit	7	6	5	4	3	2	1	0
	TPB1ADR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit 31 – ENTXP Enable TX Partial Store and Forward Operation

Bits 11:0 – TPB1ADR[11:0] Transmit Partial Store and Forward Address Watermark value. Reset = 1.

36.8.18 GMAC RX Partial Store and Forward Register

Name: GMAC_RPSF
Offset: 0x044
Reset: 0x00000FFF
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ENRXP							
Access	R/W							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					RPB1ADR[11:8]			
Access					R/W	R/W	R/W	R/W
Reset					1	1	1	1

Bit	7	6	5	4	3	2	1	0
	RPB1ADR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit 31 – ENRXP Enable RX Partial Store and Forward Operation

Bits 11:0 – RPB1ADR[11:0] Receive Partial Store and Forward Address Watermark value. Reset = 1.

36.8.19 GMAC RX Jumbo Frame Max Length Register

Name: GMAC_RJFML
Offset: 0x048
Reset: 0x00003FFF
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
			FML[13:8]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	FML[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 13:0 – FML[13:0] Frame Max Length
Rx jumbo frame maximum length.

36.8.20 GMAC Hash Register Bottom

Name: GMAC_HRB
Offset: 0x080
Reset: 0x00000000
Property: Read/Write

The unicast hash enable (UNIHEN) and the multicast hash enable (MITIHEN) bits in the Network Configuration Register (GMAC Network Configuration Register) enable the reception of hash matched frames. See Hash Addressing.

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Hash Address
The first 32 bits of the Hash Address Register.

36.8.21 GMAC Hash Register Top

Name: GMAC_HRT
Offset: 0x084
Reset: 0x00000000
Property: Read/Write

The unicast hash enable (UNIHEN) and the multicast hash enable (MITIHEN) bits in the GMAC Network Configuration Register enable the reception of hash matched frames. See Hash Addressing.

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Hash Address
Bits 63 to 32 of the Hash Address Register.

36.8.22 GMAC Specific Address 1 Bottom Register

Name: GMAC_SAB1
Offset: 0x088
Reset: 0x00000000
Property: Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Specific Address 1
Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

36.8.23 GMAC Specific Address 1 Top Register

Name: GMAC_SAT1
Offset: 0x08C
Reset: 0x00000000
Property: Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – ADDR[15:0] Specific Address 1
The most significant bits of the destination address, that is, bits 47:32.

36.8.24 GMAC Specific Address 2 Bottom Register

Name: GMAC_SAB2
Offset: 0x090
Reset: 0x00000000
Property: Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Specific Address 2
Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

36.8.25 GMAC Specific Address 2 Top Register

Name: GMAC_SAT2
Offset: 0x094
Reset: 0x00000000
Property: Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – ADDR[15:0] Specific Address 2
The most significant bits of the destination address, that is, bits 47:32.

36.8.26 GMAC Specific Address 3 Bottom Register

Name: GMAC_SAB3
Offset: 0x098
Reset: 0x00000000
Property: Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Specific Address 3
Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

36.8.27 GMAC Specific Address 3 Top Register

Name: GMAC_SAT3
Offset: 0x09C
Reset: 0x00000000
Property: Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – ADDR[15:0] Specific Address 3
The most significant bits of the destination address, that is, bits 47:32.

36.8.28 GMAC Specific Address 4 Bottom Register

Name: GMAC_SAB4
Offset: 0x0A0
Reset: 0x00000000
Property: Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Specific Address 4
Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

36.8.29 GMAC Specific Address 4 Top Register

Name: GMAC_SAT4
Offset: 0x0A4
Reset: 0x00000000
Property: Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – ADDR[15:0] Specific Address 4
The most significant bits of the destination address, that is, bits 47:32.

36.8.30 GMAC Type ID Match 1 Register

Name: GMAC_TIDM1
Offset: 0x0A8
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ENID1							
Access	R/W							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TID[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TID[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – ENID1 Enable Copying of TID Matched Frames

Value	Description
0	TID is not part of the comparison match.
1	TID is processed for the comparison match.

Bits 15:0 – TID[15:0] Type ID Match 1
For use in comparisons with received frames type ID/length frames.

36.8.31 GMAC Type ID Match 2 Register

Name: GMAC_TIDM2
Offset: 0x0AC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ENID2							
Access	R/W							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TID[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TID[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – ENID2 Enable Copying of TID Matched Frames

Value	Description
0	TID is not part of the comparison match.
1	TID is processed for the comparison match.

Bits 15:0 – TID[15:0] Type ID Match 2
For use in comparisons with received frames type ID/length frames.

36.8.32 GMAC Type ID Match 3 Register

Name: GMAC_TIDM3
Offset: 0x0B0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ENID3							
Access	R/W							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TID[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TID[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – ENID3 Enable Copying of TID Matched Frames

Value	Description
0	TID is not part of the comparison match.
1	TID is processed for the comparison match.

Bits 15:0 – TID[15:0] Type ID Match 3
For use in comparisons with received frames type ID/length frames.

36.8.33 GMAC Type ID Match 4 Register

Name: GMAC_TIDM4
Offset: 0x0B4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ENID4							
Access	R/W							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TID[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TID[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – ENID4 Enable Copying of TID Matched Frames

Value	Description
0	TID is not part of the comparison match.
1	TID is processed for the comparison match.

Bits 15:0 – TID[15:0] Type ID Match 4
For use in comparisons with received frames type ID/length frames.

36.8.34 GMAC IPG Stretch Register

Name: GMAC_IPGS
Offset: 0x0BC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	FL[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – FL[15:0] Frame Length
Bits 7:0 are multiplied with the previously transmitted frame length (including preamble). Bits 15:8 +1 divide the frame length. If the resulting number is greater than 96 and bit 28 is set in the Network Configuration Register then the resulting number is used for the transmit inter-packet-gap. 1 is added to bits 15:8 to prevent a divide by zero. See MAC Transmit Block.

36.8.35 GMAC Stacked VLAN Register

Name: GMAC_SVLAN
Offset: 0x0C0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ESVLAN							
Access	R/W							
Reset	0							

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	VLAN_TYPE[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	VLAN_TYPE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – ESVLAN Enable Stacked VLAN Processing Mode

Value	Description
0	Disable the stacked VLAN processing mode
1	Enable the stacked VLAN processing mode

Bits 15:0 – VLAN_TYPE[15:0] User Defined VLAN_TYPE Field
User defined VLAN_TYPE field. When Stacked VLAN is enabled, the first VLAN tag in a received frame will only be accepted if the VLAN type field is equal to this user defined VLAN_TYPE, OR equal to the standard VLAN type (0x8100). Note that the second VLAN tag of a Stacked VLAN packet will only be matched correctly if its VLAN_TYPE field equals 0x8100.

36.8.36 GMAC Transmit PFC Pause Register

Name: GMAC_TPFCP
Offset: 0x0C4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	PQ[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PEV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:8 – PQ[7:0] Pause Quantum
If bit 17 of the Network Control Register is written with a one then for each entry equal to zero in the Transmit PFC Pause Register[15:8], the PFC pause frame's pause quantum field associated with that entry will be taken from the Transmit Pause Quantum Register. For each entry equal to one in the Transmit PFC Pause Register [15:8], the pause quantum associated with that entry will be zero.

Bits 7:0 – PEV[7:0] Priority Enable Vector
If bit 17 of the Network Control Register is written with a one then the priority enable vector of the PFC priority based pause frame will be set equal to the value stored in this register [7:0].

36.8.37 GMAC Specific Address 1 Mask Bottom Register

Name: GMAC_SAMB1
Offset: 0x0C8
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Specific Address 1 Mask
Setting a bit to one masks the corresponding bit in the Specific Address 1 Register.

36.8.38 GMAC Specific Address Mask 1 Top Register

Name: GMAC_SAMT1
Offset: 0x0CC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – ADDR[15:0] Specific Address 1 Mask
Setting a bit to one masks the corresponding bit in the Specific Address 1 Register.

36.8.39 GMAC 1588 Timer Nanosecond Comparison Register

Name: GMAC_NSC
Offset: 0x0DC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			NANOSEC[21:16]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NANOSEC[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NANOSEC[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 21:0 – NANOSEC[21:0] 1588 Timer Nanosecond Comparison Value
Value is compared to the bits [45:24] of the TSU timer count value (upper 22 bits of nanosecond value).

36.8.40 GMAC 1588 Timer Second Comparison Low Register

Name: GMAC_SCL
Offset: 0x0E0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	SEC[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	SEC[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SEC[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SEC[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – SEC[31:0] 1588 Timer Second Comparison Value
Value is compared to seconds value bits [31:0] of the TSU timer count value.

36.8.41 GMAC 1588 Timer Second Comparison High Register

Name: GMAC_SCH
Offset: 0x0E4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	SEC[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SEC[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – SEC[15:0] 1588 Timer Second Comparison Value
Value is compared to the top 16 bits (most significant 16 bits [47:32] of seconds value) of the TSU timer count value.

36.8.42 GMAC PTP Event Frame Transmitted Seconds High Register

Name: GMAC_EFTSH
Offset: 0x0E8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – RUD[15:0] Register Update
The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.43 GMAC PTP Event Frame Received Seconds High Register

Name: GMAC_EFRSH
Offset: 0x0EC
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – RUD[15:0] Register Update
The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.44 GMAC PTP Peer Event Frame Transmitted Seconds High Register

Name: GMAC_PEFTSH
Offset: 0x0F0
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – RUD[15:0] Register Update
The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.45 GMAC PTP Peer Event Frame Received Seconds High Register

Name: GMAC_PEFRSH
Offset: 0x0F4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – RUD[15:0] Register Update
The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.46 GMAC Octets Transmitted Low Register

Name: GMAC_OTLO
Offset: 0x100
Reset: 0x00000000
Property: Read-only

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

Bit	31	30	29	28	27	26	25	24
	TXO[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TXO[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TXO[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TXO[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – TXO[31:0] Transmitted Octets
Transmitted octets in frame without errors [31:0]. The number of octets transmitted in valid frames of any type. This counter is 48-bits, and is read through two registers. This count does not include octets from automatically generated pause frames.

36.8.47 GMAC Octets Transmitted High Register

Name: GMAC_OTHI
Offset: 0x104
Reset: 0x00000000
Property: Read-only

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TXO[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TXO[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – TXO[15:0] Transmitted Octets
Transmitted octets in frame without errors [47:32]. The number of octets transmitted in valid frames of any type. This counter is 48-bits, and is read through two registers. This count does not include octets from automatically generated pause frames.

36.8.48 GMAC Frames Transmitted Register

Name: GMAC_FT
Offset: 0x108
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	FTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	FTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – FTX[31:0] Frames Transmitted without Error
Frames transmitted without error. This register counts the number of frames successfully transmitted, i.e., no underrun and not too many retries. Excludes pause frames.

36.8.49 GMAC Broadcast Frames Transmitted Register

Name: GMAC_BCFT
Offset: 0x10C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	BFTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BFTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	BFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – BFTX[31:0] Broadcast Frames Transmitted without Error
Broadcast frames transmitted without error. This register counts the number of broadcast frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

36.8.50 GMAC Multicast Frames Transmitted Register

Name: GMAC_MFT
Offset: 0x110
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	MFTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	MFTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – MFTX[31:0] Multicast Frames Transmitted without Error
This register counts the number of multicast frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

36.8.51 GMAC Pause Frames Transmitted Register

Name: GMAC_PFT
Offset: 0x114
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	PFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – PFTX[15:0] Pause Frames Transmitted Register
This register counts the number of pause frames transmitted. Only pause frames triggered by the register interface or through the external pause pins are counted as pause frames. Pause frames received through the FIFO interface are counted in the frames transmitted counter.

36.8.52 GMAC 64 Byte Frames Transmitted Register

Name: GMAC_BFT64
Offset: 0x118
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFTX[31:0] 64 Byte Frames Transmitted without Error
This register counts the number of 64 byte frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

36.8.53 GMAC 65 to 127 Byte Frames Transmitted Register

Name: GMAC_TBFT127
Offset: 0x11C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFTX[31:0] 65 to 127 Byte Frames Transmitted without Error
This register counts the number of 65 to 127 byte frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

36.8.54 GMAC 128 to 255 Byte Frames Transmitted Register

Name: GMAC_TBFT255
Offset: 0x120
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFTX[31:0] 128 to 255 Byte Frames Transmitted without Error
This register counts the number of 128 to 255 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

36.8.55 GMAC 256 to 511 Byte Frames Transmitted Register

Name: GMAC_TBFT511
Offset: 0x124
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFTX[31:0] 256 to 511 Byte Frames Transmitted without Error
This register counts the number of 256 to 511 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

36.8.56 GMAC 512 to 1023 Byte Frames Transmitted Register

Name: GMAC_TBFT1023
Offset: 0x128
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFTX[31:0] 512 to 1023 Byte Frames Transmitted without Error
This register counts the number of 512 to 1023 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

36.8.57 GMAC 1024 to 1518 Byte Frames Transmitted Register

Name: GMAC_TBFT1518
Offset: 0x12C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFTX[31:0] 1024 to 1518 Byte Frames Transmitted without Error
This register counts the number of 1024 to 1518 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

36.8.58 GMAC Greater Than 1518 Byte Frames Transmitted Register

Name: GMAC_GTBFT1518
Offset: 0x130
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFTX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFTX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFTX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFTX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFTX[31:0] Greater than 1518 Byte Frames Transmitted without Error
This register counts the number of 1518 or above byte frames successfully transmitted without error i.e., no underrun and not too many retries.

36.8.59 GMAC Transmit Underruns Register

Name: GMAC_TUR
Offset: 0x134
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							TXUNR[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	TXUNR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – TXUNR[9:0] Transmit Underruns
This register counts the number of frames not transmitted due to a transmit underrun. If this register is incremented then no other statistics register is incremented.

36.8.60 GMAC Single Collision Frames Register

Name: GMAC_SCF
Offset: 0x138
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
							SCOL[17:16]	
Access							R	R
Reset							0	0

Bit	15	14	13	12	11	10	9	8
	SCOL[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SCOL[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 17:0 – SCOL[17:0] Single Collision
This register counts the number of frames experiencing a single collision before being successfully transmitted i.e., no underrun.

36.8.61 GMAC Multiple Collision Frames Register

Name: GMAC_MCF
Offset: 0x13C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
							MCOL[17:16]	
Access							R	R
Reset							0	0

Bit	15	14	13	12	11	10	9	8
	MCOL[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MCOL[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 17:0 – MCOL[17:0] Multiple Collision
This register counts the number of frames experiencing between two and fifteen collisions prior to being successfully transmitted, i.e., no underrun and not too many retries.

36.8.62 GMAC Excessive Collisions Register

Name: GMAC_EC
Offset: 0x140
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							XCOL[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	XCOL[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – XCOL[9:0] Excessive Collisions
This register counts the number of frames that failed to be transmitted because they experienced 16 collisions.

36.8.63 GMAC Late Collisions Register

Name: GMAC_LC
Offset: 0x144
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							LCOL[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	LCOL[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – LCOL[9:0] Late Collisions
This register counts the number of late collisions occurring after the slot time (512 bits) has expired. In 10/100 mode, late collisions are counted twice i.e., both as a collision and a late collision.

36.8.64 GMAC Deferred Transmission Frames Register

Name: GMAC_DTF
Offset: 0x148
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
							DEFT[17:16]	
Access							R	R
Reset							0	0

Bit	15	14	13	12	11	10	9	8
	DEFT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DEFT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 17:0 – DEFT[17:0] Deferred Transmission
This register counts the number of frames experiencing deferral due to carrier sense being active on their first attempt at transmission. Frames involved in any collision are not counted nor are frames that experienced a transmit underrun.

36.8.65 GMAC Carrier Sense Errors Register

Name: GMAC_CSE
Offset: 0x14C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							CSR[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	CSR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – CSR[9:0] Carrier Sense Error
This register counts the number of frames transmitted where carrier sense was not seen during transmission or where carrier sense was deasserted after being asserted in a transmit frame without collision (no underrun). Only incremented in half duplex mode. The only effect of a carrier sense error is to increment this register. The behavior of the other statistics registers is unaffected by the detection of a carrier sense error.

36.8.66 GMAC Octets Received Low Register

Name: GMAC_ORLO
Offset: 0x150
Reset: 0x00000000
Property: Read-only

When reading the Octets Transmitted and Octets Received Registers, bits [31:0] should be read prior to bits [47:32] to ensure reliable operation.

Bit	31	30	29	28	27	26	25	24
	RXO[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RXO[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RXO[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RXO[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RXO[31:0] Received Octets
Received octets in frame without errors [31:0]. The number of octets received in valid frames of any type. This counter is 48-bits and is read through two registers. This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.67 GMAC Octets Received High Register

Name: GMAC_ORHI
Offset: 0x154
Reset: 0x00000000
Property: Read-only

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RXO[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RXO[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – RXO[15:0] Received Octets
Received octets in frame without errors [47:32]. The number of octets received in valid frames of any type. This counter is 48-bits and is read through two registers. This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.68 GMAC Frames Received Register

Name: GMAC_FR
Offset: 0x158
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	FRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	FRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – FRX[31:0] Frames Received without Error
Frames received without error. This register counts the number of frames successfully received. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.69 GMAC Broadcast Frames Received Register

Name: GMAC_BCFR
Offset: 0x15C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	BFRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BFRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	BFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – BFRX[31:0] Broadcast Frames Received without Error
Broadcast frames received without error. This register counts the number of broadcast frames successfully received. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.70 GMAC Multicast Frames Received Register

Name: GMAC_MFR
Offset: 0x160
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	MFRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	MFRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – MFRX[31:0] Multicast Frames Received without Error
This register counts the number of multicast frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.71 GMAC Pause Frames Received Register

Name: GMAC_PFR
Offset: 0x164
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	PFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – PFRX[15:0] Pause Frames Received Register
This register counts the number of pause frames received without error.

36.8.72 GMAC 64 Byte Frames Received Register

Name: GMAC_BFR64
Offset: 0x168
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFRX[31:0] 64 Byte Frames Received without Error
This register counts the number of 64 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.73 GMAC 65 to 127 Byte Frames Received Register

Name: GMAC_TBFR127
Offset: 0x16C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFRX[31:0] 65 to 127 Byte Frames Received without Error
This register counts the number of 65 to 127 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.74 GMAC 128 to 255 Byte Frames Received Register

Name: GMAC_TBFR255
Offset: 0x170
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFRX[31:0] 128 to 255 Byte Frames Received without Error
This register counts the number of 128 to 255 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.75 GMAC 256 to 511 Byte Frames Received Register

Name: GMAC_TBFR511
Offset: 0x174
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFRX[31:0] 256 to 511 Byte Frames Received without Error
This register counts the number of 256 to 511 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.76 GMAC 512 to 1023 Byte Frames Received Register

Name: GMAC_TBFR1023
Offset: 0x178
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFRX[31:0] 512 to 1023 Byte Frames Received without Error
This register counts the number of 512 to 1023 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

36.8.77 GMAC 1024 to 1518 Byte Frames Received Register

Name: GMAC_TBFR1518
Offset: 0x17C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFRX[31:0] 1024 to 1518 Byte Frames Received without Error
This register counts the number of 1024 to 1518 byte frames successfully received without error, i.e., no underrun and not too many retries.

36.8.78 GMAC 1519 to Maximum Byte Frames Received Register

Name: GMAC_TMXBFR
Offset: 0x180
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	NFRX[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	NFRX[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NFRX[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	NFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – NFRX[31:0] 1519 to Maximum Byte Frames Received without Error
This register counts the number of 1519 byte or above frames successfully received without error. Maximum frame size is determined by the Network Configuration Register bit 8 (1536 maximum frame size) or bit 3 (jumbo frame size). Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory. See GMAC Network Configuration Register.

36.8.79 GMAC Undersized Frames Received Register

Name: GMAC_UFR
Offset: 0x184
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							UFRX[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	UFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – UFRX[9:0] Undersize Frames Received
This register counts the number of frames received less than 64 bytes in length (10/100 mode, full duplex) that do not have either a CRC error or an alignment error.

36.8.80 GMAC Oversized Frames Received Register

Name: GMAC_OFR
Offset: 0x188
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							OFRX[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	OFRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – OFRX[9:0] Oversized Frames Received
This register counts the number of frames received exceeding 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) in length but do not have either a CRC error, an alignment error nor a receive symbol error.
See GMAC Network Configuration Register.

36.8.81 GMAC Jabbers Received Register

Name: GMAC_JR
Offset: 0x18C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							JRX[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	JRX[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – JRX[9:0] Jabbers Received
The register counts the number of frames received exceeding 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register) and have either a CRC error, an alignment error or a receive symbol error. See GMAC Network Configuration Register.

36.8.82 GMAC Frame Check Sequence Errors Register

Name: GMAC_FCSE
Offset: 0x190
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							FCKR[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	FCKR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – FCKR[9:0] Frame Check Sequence Errors
The register counts frames that are an integral number of bytes, have bad CRC and are between 64 and 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register). This register is also incremented if a symbol error is detected and the frame is of valid length and has an integral number of bytes.
This register is incremented for a frame with bad FCS, regardless of whether it is copied to memory due to ignore FCS mode being enabled in bit 26 of the Network Configuration Register. See GMAC Network Configuration Register.

36.8.83 GMAC Length Field Frame Errors Register

Name: GMAC_LFFE
Offset: 0x194
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							LFER[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	LFER[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – LFER[9:0] Length Field Frame Errors
This register counts the number of frames received that have a measured length shorter than that extracted from the length field (bytes 13 and 14). This condition is only counted if the value of the length field is less than 0x0600, the frame is not of excessive length and checking is enabled through bit 16 of the Network Configuration Register. See GMAC Network Configuration Register.

36.8.84 GMAC Receive Symbol Errors Register

Name: GMAC_RSE
Offset: 0x198
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							RXSE[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	RXSE[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – RXSE[9:0] Receive Symbol Errors
This register counts the number of frames that had GRXER asserted during reception. For 10/100 mode symbol errors are counted regardless of frame length checks. Receive symbol errors will also be counted as an FCS or alignment error if the frame is between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register). If the frame is larger it will be recorded as a jabber error. See GMAC Network Configuration Register.

36.8.85 GMAC Alignment Errors Register

Name: GMAC_AE
Offset: 0x19C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							AER[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	AER[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – AER[9:0] Alignment Errors
This register counts the frames that are not an integral number of bytes long and have bad CRC when their length is truncated to an integral number of bytes and are between 64 and 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register). This register is also incremented if a symbol error is detected and the frame is of valid length and does not have an integral number of bytes. See GMAC Network Configuration Register.

36.8.86 GMAC Receive Resource Errors Register

Name: GMAC_RRE
Offset: 0x1A0
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
							RXRER[17:16]	
Access							R	R
Reset							0	0

Bit	15	14	13	12	11	10	9	8
	RXRER[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RXRER[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 17:0 – RXRER[17:0] Receive Resource Errors
Counts the frames that were successfully received by the MAC but could not be copied to memory because no receive buffer was available. This occurs when the GMAC reads a buffer descriptor with its ownership (or used) bit set.

36.8.87 GMAC Receive Overruns Register

Name: GMAC_ROE
Offset: 0x1A4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							RXOVR[9:8]	
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	RXOVR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 9:0 – RXOVR[9:0] Receive Overruns
This register counts the number of frames that are address recognized but were not copied to memory due to a receive overrun.

36.8.88 GMAC IP Header Checksum Errors Register

Name: GMAC_IHCE
Offset: 0x1A8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	HCKER[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – HCKER[7:0] IP Header Checksum Errors
This register counts the number of frames discarded due to an incorrect IP header checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) and do not have a CRC error, an alignment error, nor a symbol error.

36.8.89 GMAC TCP Checksum Errors Register

Name: GMAC_TCE
Offset: 0x1AC
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	TCKER[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – TCKER[7:0] TCP Checksum Errors
This register counts the number of frames discarded due to an incorrect TCP checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) and do not have a CRC error, an alignment error, nor a symbol error.

36.8.90 GMAC UDP Checksum Errors Register

Name: GMAC_UCE
Offset: 0x1B0
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	UCKER[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – UCKER[7:0] UDP Checksum Errors
This register counts the number of frames discarded due to an incorrect UDP checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) and do not have a CRC error, an alignment error, nor a symbol error.

36.8.91 GMAC 1588 Timer Increment Sub-nanoseconds Register

Name: GMAC_TISUBN
Offset: 0x1BC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	LSBTIR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LSBTIR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – LSBTIR[15:0] Lower Significant Bits of Timer Increment Register
Lower significant bits of Timer Increment Register[15:0] giving a 24-bit timer_increment counter. These bits are the sub-ns value which the 1588 timer will be incremented each clock cycle. Bit n = $2^{(n-16)}$ nsec giving a resolution of approximately 15.2E^{-15} sec.

36.8.92 GMAC 1588 Timer Seconds High Register

Name: GMAC_TSH
Offset: 0x1C0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TCS[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TCS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – TCS[15:0] Timer Count in Seconds
This register is writable. It increments by one when the 1588 nanoseconds counter counts to one second. It may also be incremented when the Timer Adjust Register is written.

36.8.93 GMAC 1588 Timer Seconds Low Register

Name: GMAC_TSL
Offset: 0x1D0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	TCS[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TCS[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TCS[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TCS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – TCS[31:0] Timer Count in Seconds
This register is writable. It increments by one when the 1588 nanoseconds counter counts to one second. It may also be incremented when the Timer Adjust Register is written.

36.8.94 GMAC 1588 Timer Nanoseconds Register

Name: GMAC_TN
Offset: 0x1D4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
			TNS[29:24]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TNS[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TNS[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TNS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 29:0 – TNS[29:0] Timer Count in Nanoseconds
This register is writable. It can also be adjusted by writes to the 1588 Timer Adjust Register. It increments by the value of the 1588 Timer Increment Register each clock cycle.

36.8.95 GMAC 1588 Timer Adjust Register

Name: GMAC_TA
Offset: 0x1D8
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	ADJ		ITDT[29:24]					
Access	W		W	W	W	W	W	W
Reset	–		–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	ITDT[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	ITDT[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	ITDT[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 31 – ADJ Adjust 1588 Timer
Write as one to subtract from the 1588 timer. Write as zero to add to it.

Bits 29:0 – ITDT[29:0] Increment/Decrement
The number of nanoseconds to increment or decrement the 1588 Timer Nanoseconds Register. If necessary, the 1588 Seconds Register will be incremented or decremented.

36.8.96 GMAC 1588 Timer Increment Register

Name: GMAC_TI
Offset: 0x1DC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	NIT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ACNS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CNS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:16 – NIT[7:0] Number of Increments
The number of increments after which the alternative increment is used.

Bits 15:8 – ACNS[7:0] Alternative Count Nanoseconds
Alternative count of nanoseconds by which the 1588 Timer Nanoseconds Register will be incremented each clock cycle.

Bits 7:0 – CNS[7:0] Count Nanoseconds
A count of nanoseconds by which the 1588 Timer Nanoseconds Register will be incremented each clock cycle.

36.8.97 GMAC PTP Event Frame Transmitted Seconds Low Register

Name: GMAC_EFTSL
Offset: 0x1E0
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	RUD[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RUD[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RUD[31:0] Register Update
The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.98 GMAC PTP Event Frame Transmitted Nanoseconds Register

Name: GMAC_EFTN
Offset: 0x1E4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
			RUD[29:24]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RUD[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 29:0 – RUD[29:0] Register Update
The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.99 GMAC PTP Event Frame Received Seconds Low Register

Name: GMAC_EFRSL
Offset: 0x1E8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	RUD[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RUD[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RUD[31:0] Register Update
The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.100 GMAC PTP Event Frame Received Nanoseconds Register

Name: GMAC_EFRN
Offset: 0x1EC
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
			RUD[29:24]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RUD[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 29:0 – RUD[29:0] Register Update
The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.101 GMAC PTP Peer Event Frame Transmitted Seconds Low Register

Name: GMAC_PEFTSL
Offset: 0x1F0
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	RUD[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RUD[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RUD[31:0] Register Update
The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.102 GMAC PTP Peer Event Frame Transmitted Nanoseconds Register

Name: GMAC_PEFTN
Offset: 0x1F4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
			RUD[29:24]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RUD[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 29:0 – RUD[29:0] Register Update
The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.103 GMAC PTP Peer Event Frame Received Seconds Low Register

Name: GMAC_PEFRSL
Offset: 0x1F8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	RUD[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RUD[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RUD[31:0] Register Update
The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.104 GMAC PTP Peer Event Frame Received Nanoseconds Register

Name: GMAC_PEFRN
Offset: 0x1FC
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
			RUD[29:24]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RUD[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RUD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RUD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 29:0 – RUD[29:0] Register Update
The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

36.8.105 GMAC Received LPI Transitions

Name: GMAC_RXLPI
Offset: 0x270
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – COUNT[15:0] Count of RX LPI transitions (cleared on read)
A count of the number of times there is a transition from receiving normal idle to receiving low power idle.

36.8.106 GMAC Received LPI Time

Name: GMAC_RXLPITIME
Offset: 0x274
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	LPITIME[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	LPITIME[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LPITIME[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – LPITIME[23:0] Time in LPI (cleared on read)
This field increments once every 16 MCK cycles when the bit LPI Indication (bit 7) is set in the Network Status register.

36.8.107 GMAC Transmit LPI Transitions

Name: GMAC_TXLPI
Offset: 0x278
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	COUNT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COUNT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – COUNT[15:0] Count of LPI transitions (cleared on read)
A count of the number of times the bit Enable LPI Transmission (bit 19) goes from low to high in the Network Control register.

36.8.108 GMAC Transmit LPI Time

Name: GMAC_TXLPITIME
Offset: 0x27C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	LPITIME[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	LPITIME[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LPITIME[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – LPITIME[23:0] Time in LPI (cleared on read)
This field increments once every 16 MCK cycles when the bit Enable LPI Transmission (bit 19) is set in the Network Control register.

36.8.109 GMAC Interrupt Status Register Priority Queue x

Name: GMAC_ISRPQx
Offset: 0x0400 + (x-1)*0x04 [x=1..5]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					HRESP	ROVR		
Access					R/W	R/W		
Reset					0	0		

Bit	7	6	5	4	3	2	1	0
	TCOMP	TFC	RLEX			RXUBR	RCOMP	
Access	R/W	R/W	R/W			R/W	R/W	
Reset	0	0	0			0	0	

Bit 11 – HRESP HRESP Not OK

Bit 10 – ROVR Receive Overrun

Bit 7 – TCOMP Transmit Complete

Bit 6 – TFC Transmit Frame Corruption Due to AHB Error
Transmit frame corruption due to AHB error—set if an error occurs whilst midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame.

Bit 5 – RLEX Retry Limit Exceeded or Late Collision

Bit 2 – RXUBR RX Used Bit Read

Bit 1 – RCOMP Receive Complete

36.8.110 GMAC Transmit Buffer Queue Base Address Register Priority Queue x

Name: GMAC_TBQBAPQx
Offset: 0x0440 + (x-1)*0x04 [x=1..5]
Reset: 0x00000000
Property: Read/Write

These registers hold the start address of the transmit buffer queues (transmit buffers descriptor lists) for the additional queues and must be initialized to the address of valid descriptors, even if the priority queues are not used.

Bit	31	30	29	28	27	26	25	24
	TXBQBA[29:22]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TXBQBA[21:14]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TXBQBA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TXBQBA[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bits 31:2 – TXBQBA[29:0] Transmit Buffer Queue Base Address
Written with the address of the start of the transmit queue.

36.8.111 GMAC Receive Buffer Queue Base Address Register Priority Queue x

Name: GMAC_RBQBAPQx
Offset: 0x0480 + (x-1)*0x04 [x=1..5]
Reset: 0x00000000
Property: Read/Write

These registers hold the start address of the receive buffer queues (receive buffers descriptor lists) for the additional queues and must be initialized to the address of valid descriptors, even if the priority queues are not used.

Bit	31	30	29	28	27	26	25	24
	RXBQBA[29:22]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RXBQBA[21:14]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RXBQBA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RXBQBA[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bits 31:2 – RXBQBA[29:0] Receive Buffer Queue Base Address
Written with the address of the start of the receive queue.

36.8.112 GMAC Receive Buffer Size Register Priority Queue x

Name: GMAC_RBSRPQx
Offset: 0x04A0 + (x-1)*0x04 [x=1..5]
Reset: 0x00000002
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RBS[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RBS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	1	0

Bits 15:0 – RBS[15:0] Receive Buffer Size
DMA receive buffer size in AHB system memory. The value defined by these bits determines the size of buffer to use in main AHB system memory when writing received data.
The value is defined in multiples of 64 bytes such that a value of 0x01 corresponds to buffers of 64 bytes, 0x02 corresponds to 128 bytes etc.
For example:
0x02: 128bytes
0x18: 1536bytes (1 × max length frame/buffer)
0xA0: 10240bytes (1 × 10K jumbo frame/buffer)
Note that this value should never be written as zero.

36.8.113 GMAC Credit-Based Shaping Control Register

Name: GMAC_CBSCR
Offset: 0x4BC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							QBE	QAE
Access							R/W	R/W
Reset							0	0

Bit 1 – QBE Queue B CBS Enable

Value	Description
0	Credit-based shaping on the highest priority queue (queue B) is disabled.
1	Credit-based shaping on the highest priority queue (queue B) is enabled.

Bit 0 – QAE Queue A CBS Enable

Value	Description
0	Credit-based shaping on the second highest priority queue (queue A) is disabled.
1	Credit-based shaping on the second highest priority queue (queue A) is enabled.

36.8.114 GMAC Credit-Based Shaping IdleSlope Register for Queue A

Name: GMAC_CBSISQA
Offset: 0x4C0
Reset: 0x00000000
Property: Read/Write

Credit-based shaping must be disabled in GMAC_CBSCR before updating this register.

Bit	31	30	29	28	27	26	25	24
	IS[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	IS[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IS[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – IS[31:0] IdleSlope
IdleSlope value for queue A in bytes/second.
The IdleSlope value is defined as the rate of change of credit when a packet is waiting to be sent. This must not exceed the port transmit rate which is dependent on the speed of operation, e.g., 100 Mb/second = 32'h017D7840 If 50% of bandwidth was to be allocated to a particular queue in 100 Mb/second mode, then the IdleSlope value for that queue would be calculated as 32'h017D7840 / 2.

36.8.115 GMAC Credit-Based Shaping IdleSlope Register for Queue B

Name: GMAC_CBSISQB
Offset: 0x4C4
Reset: 0x00000000
Property: Read/Write

Credit-based shaping must be disabled in GMAC_CBSCR before updating this register.

Bit	31	30	29	28	27	26	25	24
	IS[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	IS[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IS[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – IS[31:0] IdleSlope
IdleSlope value for queue B in bytes/second.
The IdleSlope value is defined as the rate of change of credit when a packet is waiting to be sent. This must not exceed the port transmit rate which is dependent on the speed of operation, e.g., 100 Mb/second = 32'h017D7840. If 50% of bandwidth was to be allocated to a particular queue in 100 Mb/second mode, then the IdleSlope value for that queue would be calculated as 32'h017D7840 / 2.

36.8.116 GMAC Screening Type 1 Register x Priority Queue

Name: GMAC_ST1RPQx
Offset: 0x0500 + x*0x04 [x=0..3]
Reset: 0x00000000
Property: Read/Write

Screening type 1 registers are used to allocate up to 6 priority queues to received frames based on certain IP or UDP fields of incoming frames.

Bit	31	30	29	28	27	26	25	24
			UDPE	DSTCE	UDPM[15:12]			
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	UDPM[11:4]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	UDPM[3:0]				DSTCM[7:4]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DSTCM[3:0]					QNB[2:0]		
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset	0	0	0	0		0	0	0

Bit 29 – UDPE UDP Port Match Enable
When UDP port match enable is set (bit 29), the UDP Destination Port of the received UDP frame is matched against bits 27:12.

Bit 28 – DSTCE Differentiated Services or Traffic Class Match Enable
When DS/TC match enable is set (bit 28), the DS (differentiated services) field of the received IPv4 header or TC field (traffic class) of IPv6 headers are matched against bits 11:4.

Bits 27:12 – UDPM[15:0] UDP Port Match
When UDP port match enable is set (bit 29), the UDP Destination Port of the received UDP frame is matched against bits 27:12.

Bits 11:4 – DSTCM[7:0] Differentiated Services or Traffic Class Match
When DS/TC match enable is set (bit 28), the DS (differentiated services) field of the received IPv4 header or TC field (traffic class) of IPv6 headers are matched against bits 11:4.

Bits 2:0 – QNB[2:0] Queue Number (0–5)
If a match is successful, then the queue value programmed in bits 2:0 is allocated to the frame.

36.8.117 GMAC Screening Type 2 Register x Priority Queue

Name: GMAC_ST2RPQx
Offset: 0x0540 + x*0x04 [x=0..7]
Reset: 0x00000000
Property: Read/Write

Screening type 2 registers are used to allocate up to 6 priority queues to received frames based on the VLAN priority field of received ethernet frames.

Bit	31	30	29	28	27	26	25	24
		COMPCE	COMPC[4:0]					COMPBE
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	COMPB[4:0]					COMPAE	COMPA[4:3]	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	COMPA[2:0]			ETHE	I2ETH[2:0]			VLANE
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		VLANP[2:0]				QNB[2:0]		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit 30 – COMPCE Compare C Enable

Value	Description
0	Comparison via the register designated by index COMPC is disabled.
1	Comparison via the register designated by index COMPC is enabled.

Bits 29:25 – COMPC[4:0] Index of Screening Type 2 Compare Word 0/Word 1 register x
COMPC is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPCE is set, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

Bit 24 – COMPBE Compare B Enable

Value	Description
0	Comparison via the register designated by index COMPB is disabled.
1	Comparison via the register designated by index COMPB is enabled.

Bits 23:19 – COMPB[4:0] Index of Screening Type 2 Compare Word 0/Word 1 register x
COMPB is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPBE is set, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

Bit 18 – COMPAE Compare A Enable

Value	Description
0	Comparison via the register designated by index COMPA is disabled.
1	Comparison via the register designated by index COMPA is enabled.

Bits 17:13 – COMPA[4:0] Index of Screening Type 2 Compare Word 0/Word 1 register x
COMPA is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPAE is set, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

Bit 12 – ETHE EtherType Enable

Value	Description
0	EtherType match with bits 15:0 in the register designated by the value of I2ETH is disabled.
1	EtherType match with bits 15:0 in the register designated by the value of I2ETH is enabled.

Bits 11:9 – I2ETH[2:0] Index of Screening Type 2 EtherType register x
When ETHE is set (bit 12), the field EtherType (last EtherType in the header if the frame is VLAN tagged) is compared with bits 15:0 in the register designated by the value of I2ETH.

Bit 8 – VLANE VLAN Enable

Value	Description
0	VLAN match is disabled.
1	VLAN match is enabled.

Bits 6:4 – VLANP[2:0] VLAN Priority
When VLAN match enable is set (bit 8), the VLAN priority field of the received frame is matched against bits 7:4 of this register.

Bits 2:0 – QNB[2:0] Queue Number (0–5)
If a match is successful, then the queue value programmed in QNB is allocated to the frame.

36.8.118 GMAC Interrupt Enable Register Priority Queue x

Name: GMAC_IERPQx
Offset: 0x0600 + (x-1)*0x04 [x=1..5]
Reset: –
Property: Write-only

The following values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					HRESP	ROVR		
Access					W	W		
Reset					–	–		

Bit	7	6	5	4	3	2	1	0
	TCOMP	TFC	RLEX			RXUBR	RCOMP	
Access	W	W	W			W	W	
Reset	–	–	–			–	–	

Bit 11 – HRESP HRESP Not OK

Bit 10 – ROVR Receive Overrun

Bit 7 – TCOMP Transmit Complete

Bit 6 – TFC Transmit Frame Corruption Due to AHB Error

Bit 5 – RLEX Retry Limit Exceeded or Late Collision

Bit 2 – RXUBR RX Used Bit Read

Bit 1 – RCOMP Receive Complete

36.8.119 GMAC Interrupt Disable Register Priority Queue x

Name: GMAC_IDRPQx
Offset: 0x0620 + (x-1)*0x04 [x=1..5]
Reset: –
Property: Write-only

The following values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					HRESP	ROVR		
Access					W	W		
Reset					–	–		

Bit	7	6	5	4	3	2	1	0
	TCOMP	TFC	RLEX			RXUBR	RCOMP	
Access	W	W	W			W	W	
Reset	–	–	–			–	–	

Bit 11 – HRESP HRESP Not OK

Bit 10 – ROVR Receive Overrun

Bit 7 – TCOMP Transmit Complete

Bit 6 – TFC Transmit Frame Corruption Due to AHB Error

Bit 5 – RLEX Retry Limit Exceeded or Late Collision

Bit 2 – RXUBR RX Used Bit Read

Bit 1 – RCOMP Receive Complete

36.8.120 GMAC Interrupt Mask Register Priority Queue x

Name: GMAC_IMRPQx
Offset: 0x0640 + (x-1)*0x04 [x=1..5]
Reset: 0x00000000
Property: Read/Write

A read of this register returns the value of the receive complete interrupt mask.

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Register, causing an interrupt to be generated if a 1 is written.

The following values are valid for all listed bit names of this register:

0: Corresponding interrupt is enabled.

1: Corresponding interrupt is disabled.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					HRESP	ROVR		
Access					R/W	R/W		
Reset					0	0		

Bit	7	6	5	4	3	2	1	0
	TCOMP	AHB	RLEX			RXUBR	RCOMP	
Access	R/W	R/W	R/W			R/W	R/W	
Reset	0	0	0			0	0	

Bit 11 – HRESP HRESP Not OK

Bit 10 – ROVR Receive Overrun

Bit 7 – TCOMP Transmit Complete

Bit 6 – AHB AHB Error

Bit 5 – RLEX Retry Limit Exceeded or Late Collision

Bit 2 – RXUBR RX Used Bit Read

Bit 1 – RCOMP Receive Complete

36.8.121 GMAC Screening Type 2 EtherType Register x

Name: GMAC_ST2ERx
Offset: 0x06E0 + x*0x04 [x=0..3]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	COMPVAL[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	COMPVAL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – COMPVAL[15:0] Ethertype Compare Value
When the bit GMAC_ST2RPQ.ETHE is enabled, the EtherType (last EtherType in the header if the frame is VLAN tagged) is compared with bits 15:0 in the register designated by GMAC_ST2RPQ.I2ETH.

36.8.122 GMAC Screening Type 2 Compare Word 0 Register x

Name: GMAC_ST2CW0x
Offset: 0x0700 + x*0x08 [x=0..23]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	COMPVAL[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	COMPVAL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MASKVAL[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MASKVAL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:16 – COMPVAL[15:0] Compare Value
The byte stored in bits [23:16] is compared against the first byte of the 2 bytes extracted from the frame.
The byte stored in bits [31:24] is compared against the second byte of the 2 bytes extracted from the frame.

Bits 15:0 – MASKVAL[15:0] Mask Value
The value of MASKVAL ANDed with the 2 bytes extracted from the frame is compared to the value of MASKVAL ANDed with the value of COMPVAL.

36.8.123 GMAC Screening Type 2 Compare Word 1 Register x

Name: GMAC_ST2CW1x
Offset: 0x0704 + x*0x08 [x=0..23]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								OFFSSTRT[1]
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
	OFFSSTRT[0]	OFFSVAL[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 8:7 – OFFSSTRT[1:0] Ethernet Frame Offset Start

Value	Name	Description
0	FRAMESTART	Offset from the start of the frame
1	ETHERTYPE	Offset from the byte after the EtherType field
2	IP	Offset from the byte after the IP header field
3	TCP_UDP	Offset from the byte after the TCP/UDP header field

Bits 6:0 – OFFSVAL[6:0] Offset Value in Bytes
The value of OFFSVAL ranges from 0 to 127 bytes, and is counted from either the start of the frame, the byte after the EtherType field (last EtherType in the header if the frame is VLAN tagged), the byte after the IP header (IPv4 or IPv6) or the byte after the TCP/UDP header.

37. Quad Serial Peripheral Interface (QSPI)

37.1 Description

The Quad Serial Peripheral Interface (QSPI) is a synchronous serial data link that provides communication with external devices in Master mode.

The QSPI can be used in SPI mode to interface to serial peripherals such as ADCs, DACs, LCD controllers, CAN controllers and sensors, or in Serial Memory mode to interface to serial Flash memories.

The QSPI allows the system to execute code directly from a serial Flash memory (XIP) without code shadowing to RAM. The serial Flash memory mapping is seen in the system as other memories such as ROM, SRAM, DRAM, embedded Flash memory, etc.

With the support of the Quad SPI protocol, the QSPI allows the system to use high-performance serial Flash memories which are small and inexpensive, in place of larger and more expensive parallel Flash memories.

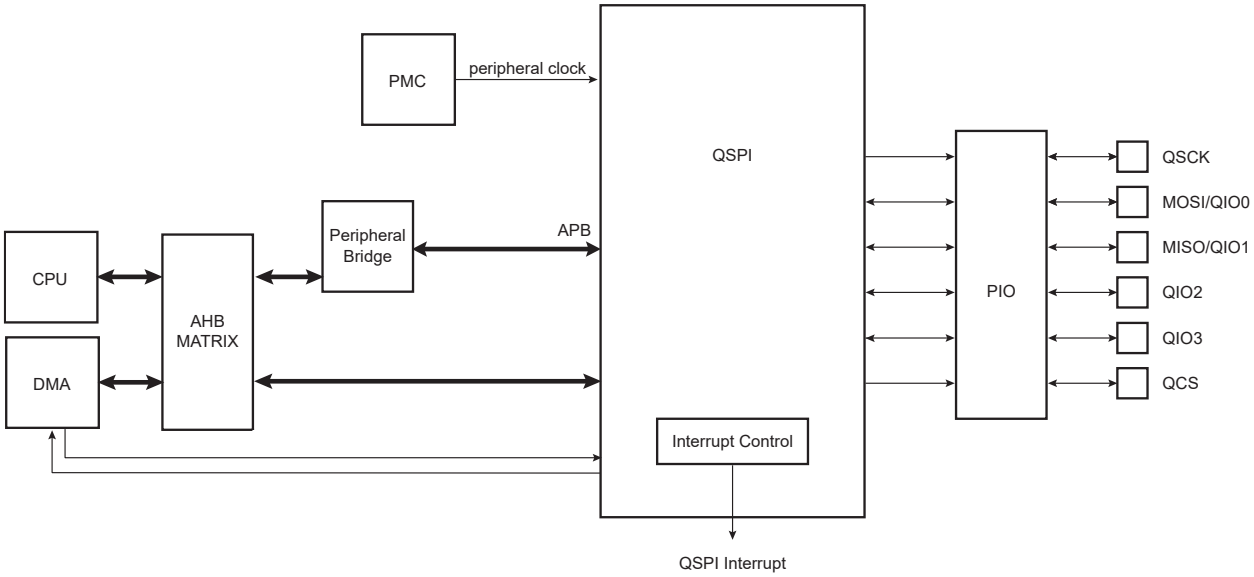
Note: Stacked devices with a rollover in the memory address space at each die boundary are not supported.

37.2 Embedded Characteristics

- Master SPI Interface
  - Programmable clock phase and clock polarity
  - Programmable transfer delays between consecutive transfers, between clock and data, between deactivation and activation of chip select
- SPI Mode
  - Interface to serial peripherals such as ADCs, DACs, LCD controllers, CAN controllers and sensors
  - 8-bit/16-bit/32-bit programmable data length
- Serial Memory Mode
  - Interface to serial Flash memories operating in Single-bit SPI, Dual SPI and Quad SPI
  - Interface to serial Flash Memories operating in Single Data Rate Mode
  - Supports “Execute In Place” (XIP)— code execution by the system directly from a serial Flash memory
  - Flexible instruction register for compatibility with all serial Flash memories
  - 32-bit address mode (default is 24-bit address) to support serial Flash memories larger than 128 Mbits
  - Continuous read mode
  - Scrambling/unscrambling “On-The-Fly”
- Connection to DMA Channel Capabilities Optimizes Data Transfers
  - One channel for the receiver, one channel for the transmitter
- Register Write Protection

37.3 Block Diagram

Figure 37-1. Block Diagram



37.4 Signal Description

Table 37-1. Signal Description

Pin Name	Pin Description	Type
QSCK	Serial Clock	Output
MOSI (QIO0) (1)(2)	Data Output (Data Input Output 0)	Output (Input/Output)
MISO (QIO1) (1)(2)	Data Input (Data Input Output 1)	Input (Input/Output)
QIO2 (3)	Data Input Output 2	Input/Output
QIO3 (3)	Data Input Output 3	Input/Output
QCS	Peripheral Chip Select	Output

Note:

1. MOSI and MISO are used for single-bit SPI operation.
2. QIO0–QIO1 are used for Dual SPI operation.
3. QIO0–QIO3 are used for Quad SPI operation.

37.5 Product Dependencies

37.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the QSPI pins to their peripheral functions.

37.5.2 Power Management

The QSPI may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the QSPI clock.

37.5.3 Interrupt Sources

The QSPI has an interrupt line connected to the Interrupt Controller. Handling the QSPI interrupt requires programming the interrupt controller before configuring the QSPI.

37.5.4 Direct Memory Access Controller (DMA)

The QSPI can be used in conjunction with the Direct Memory Access Controller (DMA) in order to reduce processor overhead. For a full description of the DMA, refer to the section “DMA Controller (XDMAC)”.

Note: DMA write accesses must be 32-bit aligned. If a single byte is to be written in a 32-bit word, the rest of the word must be filled with ones.

37.6 Functional Description

37.6.1 Serial Clock Baud Rate

The QSPI baud rate clock is generated by dividing the peripheral clock by a value between 1 and 256.

37.6.2 Serial Clock Phase and Polarity

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the QSPI Serial Clock register (QSPI_SCR). The CPHA bit in the QSPI_SCR programs the clock phase. These two parameters determine the edges of the clock signal on which data is driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Thus, the interfaced slave must use the same parameter values to communicate.

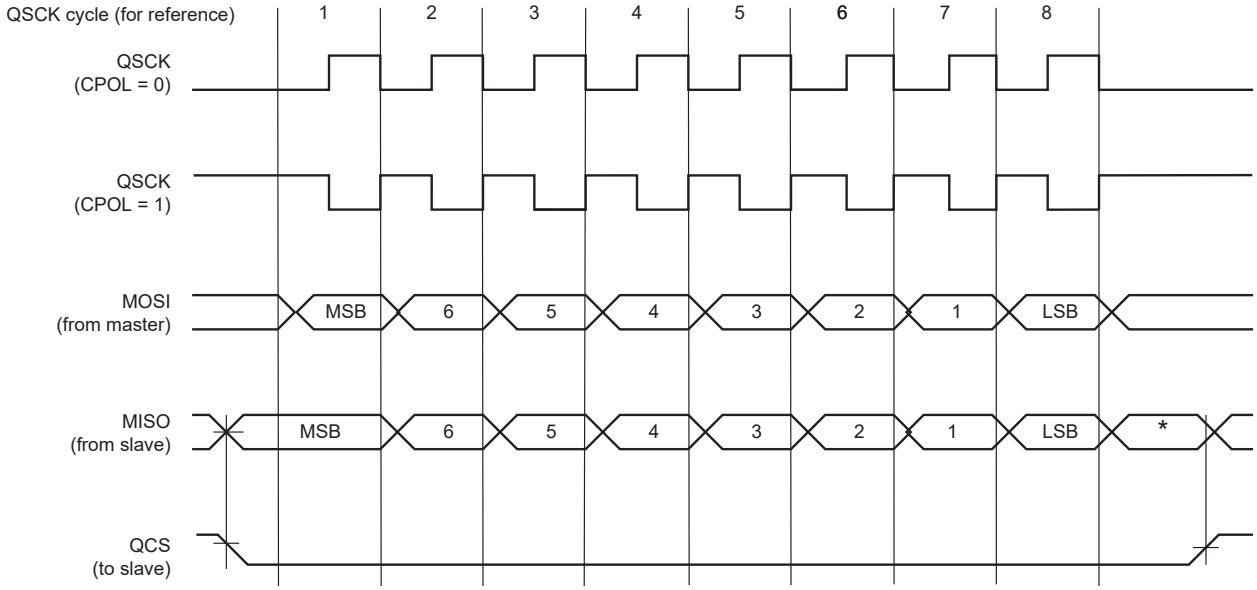
The table below shows the four modes and corresponding parameter settings.

Table 37-2. QSPI Bus Clock Modes

QSPI Clock Mode	QSPI_SCR.CPOL	QSPI_SCR.CPHA	Shift QSCK Edge	Capture QSCK Edge	QSCK Inactive Level
0	0	0	Falling	Falling	Low
1	0	1	Rising	Rising	Low
2	1	0	Rising	Rising	High
3	1	1	Falling	Falling	High

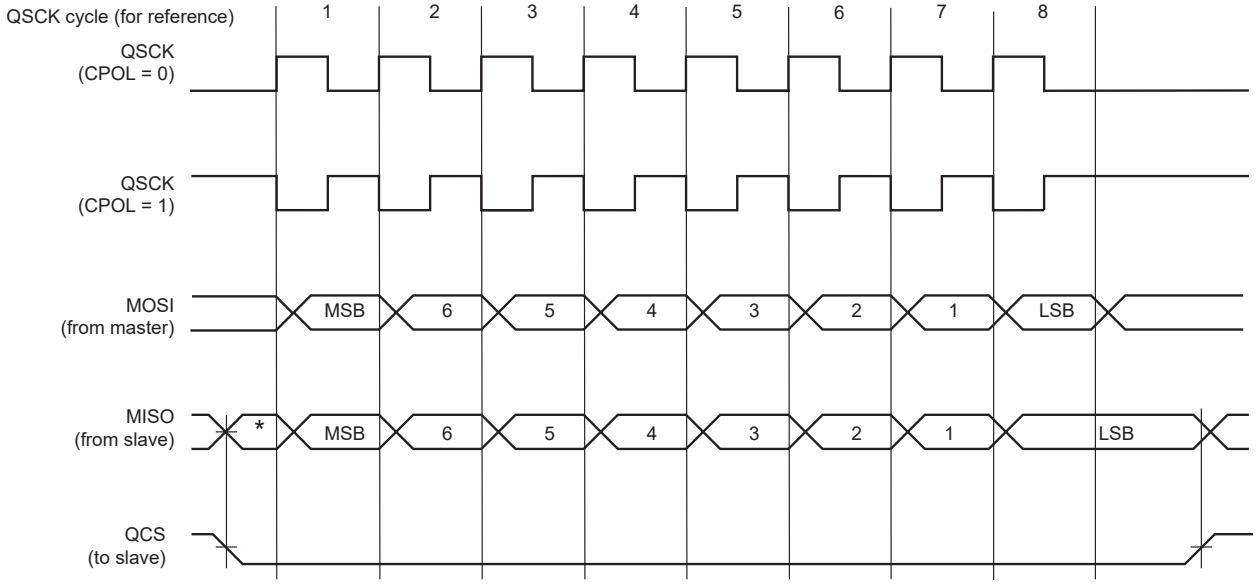
The following figures show examples of data transfers.

Figure 37-2. QSPI Transfer Format (QSPI_SCR.CPHA = 0, 8 bits per transfer)



* Not defined, but normally MSB of previous character received.

Figure 37-3. QSPI Transfer Format (QSPI_SCR.CPHA = 1, 8 bits per transfer)



* Not defined but normally LSB of previous character transmitted.

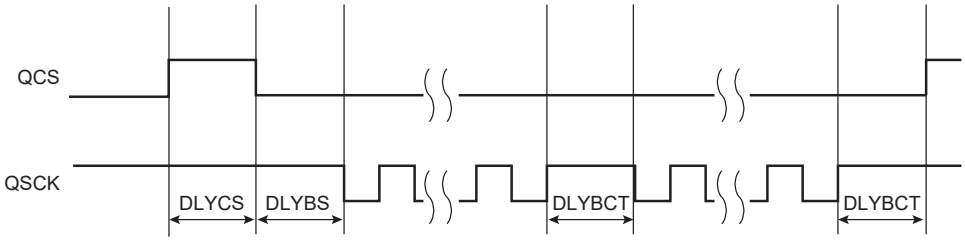
37.6.3 Transfer Delays

The figure below shows several consecutive transfers while the chip select is active. Three delays can be programmed to modify the transfer waveforms:

- The delay between the deactivation and the activation of QCS, programmed by writing QSPI_MR.DLYCS. Allows to adjust the minimum time of QCS at high level.
- The delay before QSCK, programmed by writing QSPI_SR.DLYBS. Allows the start of QSCK to be delayed after the chip select has been asserted.
- The delay between consecutive transfers, programmed by writing QSPI_MR.DLYBCT. Allows insertion of a delay between two consecutive transfers. In Serial Memory mode, this delay is not programmable and DLYBCT is ignored. In this mode, DLYBCT must be written to ‘0’.

These delays allow the QSPI to be adapted to the interfaced peripherals and their speed and bus release time.

Figure 37-4. Programmable Delays



37.6.4 QSPI SPI Mode

In SPI mode, the QSPI acts as a standard SPI Master.

To activate this mode, QSPI_MR.SMM must be written to ‘0’ in QSPI_MR.

37.6.4.1 SPI Mode Operations

The QSPI in standard SPI mode operates on the clock generated by the internal programmable baud rate generator. It fully controls the data transfers to and from the slave connected to the SPI bus. The QSPI drives the chip select line to the slave (QCS) and the serial clock signal (QSCK).

The QSPI features two holding registers, the Transmit Data register (QSPI_TDR) and the Receive Data register (QSPI_RDR), and a single internal shift register. The holding registers maintain the data flow at a constant rate.

After enabling the QSPI, a data transfer begins when the processor writes to the QSPI_TDR. The written data is immediately transferred to the internal shift register and transfer on the SPI bus starts. While the data in the internal shift register is shifted on the MOSI line, the MISO line is sampled and shifted to the internal shift register. Receiving data cannot occur without transmitting data. If receiving mode is not needed, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the Status register (QSPI_SR) can be discarded.

If new data is written in QSPI_TDR during the transfer, it is retained there until the current transfer is completed. Then, the received data is transferred from the internal shift register to the QSPI_RDR, the data in QSPI_TDR is loaded in the internal shift register and a new transfer starts.

The transfer of a data written in QSPI_TDR in the internal shift register is indicated by the Transmit Data Register Empty (TDRE) bit in the QSPI_SR. When new data is written in QSPI_TDR, this bit is cleared. QSPI_SR.TDRE is used to trigger the Transmit DMA channel.

The end of transfer is indicated by the TXEMPTY flag in the QSPI_SR. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, QSPI_SR.TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

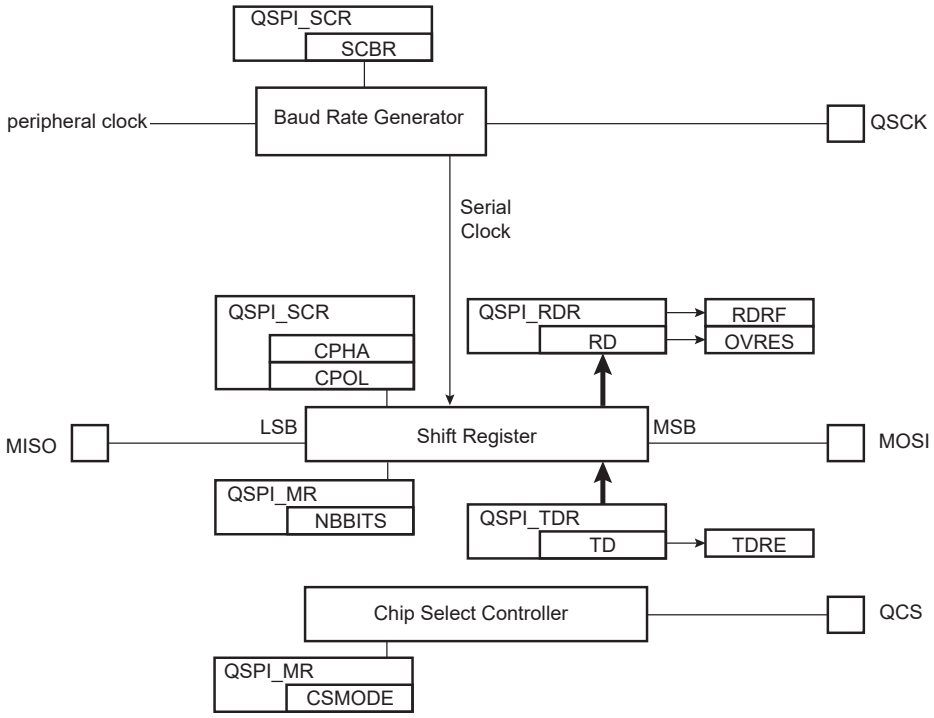
The transfer of received data from the internal shift register in QSPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in the QSPI_SR. When the received data is read, QSPI_SR.RDRF bit is cleared.

If the QSPI_RDR has not been read before new data is received, the Overrun Error Status (OVRES) bit in QSPI_SR is set. As long as this flag is set, data is loaded in QSPI_RDR. The user must read the QSPI_SR to clear the OVRES bit.

The following figures show, respectively, a block diagram of the SPI when operating in Master mode, and a flow chart describing how transfers are handled.

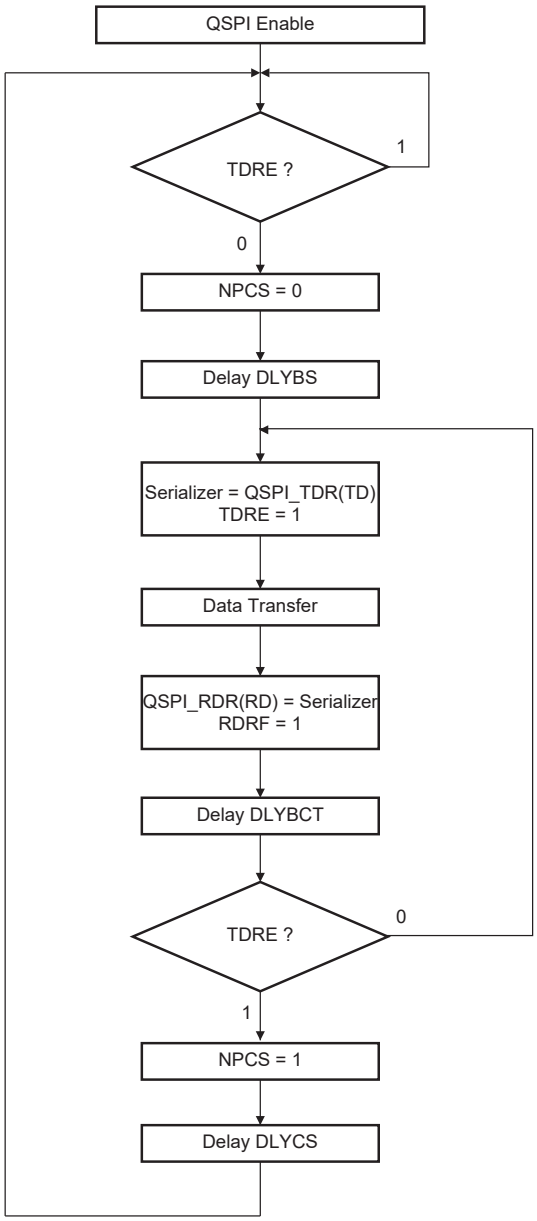
37.6.4.2 SPI Mode Block Diagram

Figure 37-5. SPI Mode Block Diagram



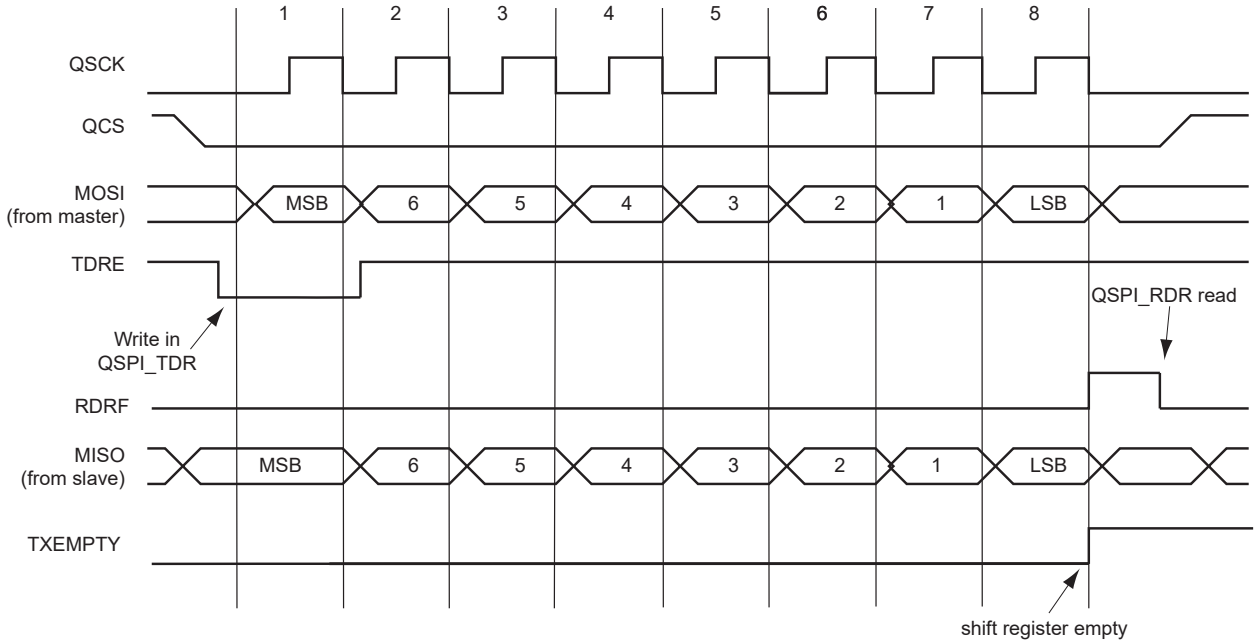
37.6.4.3 SPI Mode Flow Diagram

Figure 37-6. SPI Mode Flow Diagram



The figure below shows Transmit Data Register Empty (TDRE), Receive Data Register Full (RDRF) and Transmission Register Empty (TXEMPTY) status flags behavior within the QSPI_SR during an 8-bit data transfer in Fixed mode, without DMA.

Figure 37-7. Status Register Flags Behavior



37.6.4.4 Peripheral Deselection without DMA

During a transfer of more than one data on a Chip Select without the DMA, the QSPI_TDR is loaded by the processor and the flag TDRE rises as soon as the content of the QSPI_TDR is transferred into the internal shift register. When this flag is detected high, the QSPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer, the Chip Select is not deasserted between the two transfers. Depending on the application software handling the QSPI_SR flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload the QSPI_TDR in time to keep the chip select active (low). A null Delay Between Consecutive Transfer (DLYBCT) value in the QSPI_MR gives even less time for the processor to reload the QSPI_TDR. With some SPI slave peripherals, requiring the chip select line to remain active (low) during a full set of transfers may lead to communication errors.

To facilitate interfacing with such devices, QSPI_MR.CSMODE may be configured to ‘1’. This allows the chip select lines to remain in their current state (low = active) until the end of transfer is indicated by the Last Transfer (LASTXFER) bit in the Control register (QSPI_CR). Even if the QSPI_TDR is not reloaded, the chip select remains active. To have the chip select line rise at the end of the last data transfer, QSPI_CR.LASTXFER must be written to ‘1’ at the same time or after writing the last data to transmit into the QSPI_TDR.

37.6.4.5 Peripheral Deselection with DMA

When the DMA Controller is used, the Chip Select line remains low during the transfer since the TDRE flag is managed by the DMA itself. Reloading the QSPI_TDR by the DMA is done as soon as the TDRE flag is set. In this case, writing QSPI_MR.CSMODE to ‘1’ may not be needed. However, when other DMA channels connected to other peripherals are also in use, the QSPI DMA could be delayed by another DMA with a higher priority on the bus. Having DMA buffers in slower memories like Flash memory or SDRAM compared to fast internal SRAM, may lengthen the reload time of the QSPI_TDR by the DMA as well. This means that the QSPI_TDR might not be reloaded in time to keep the chip select line low. In this case, the chip select line may toggle between data transfer and according to some SPI Slave devices, the communication might get lost. It may be necessary to configure QSPI_MR.CSMODE to ‘1’.

When QSPI_MR.CSMODE is configured to ‘0’, the QCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a Chip Select, the flag TDRE rises as soon as the content of the QSPI_TDR is transferred into the internal shifter. When this flag is detected, the QSPI_TDR can be reloaded. If this reload occurs before the end of the current transfer, the Chip Select is not deasserted between the two transfers. This might lead to difficulties for interfacing with some serial peripherals requiring the chip select to be deasserted after each transfer. To facilitate interfacing with such devices, the QSPI_MR may be configured with QSPI_MR.CSMODE at ‘2’.

37.6.5 QSPI Serial Memory Mode

In Serial Memory mode, the QSPI acts as a serial Flash memory controller. The QSPI can be used to read data from the serial Flash memory allowing the CPU to execute code from it (XIP execute in place). The QSPI can also be used to control the serial Flash memory (Program, Erase, Lock, etc.) by sending specific commands. In this mode, the QSPI is compatible with single-bit SPI, Dual SPI and Quad SPI protocols.

To activate this mode, QSPI_MR.SMM must be written to '1'.

In Serial Memory mode, data is transferred either by QSPI_TDR and QSPI_RDR or by writing or read in the QSPI memory space (0x1800_0000) depending on TFRTYP and SMRM configuration.

37.6.5.1 Instruction Frame

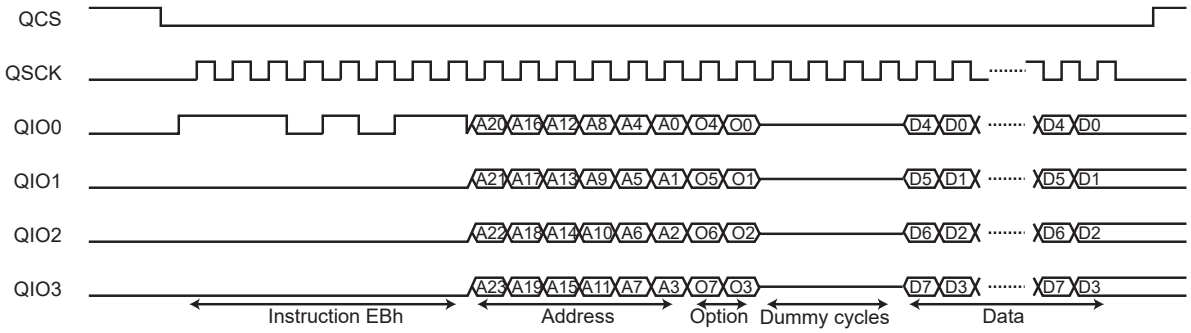
In order to control serial Flash memories, the QSPI is able to send instructions via the SPI bus (ex: READ, PROGRAM, ERASE, LOCK, etc.). Because the instruction set implemented in serial Flash memories is memory vendor-dependent, the QSPI includes a complete Instruction Frame register (QSPI_IFR), which makes it very flexible and compatible with all serial Flash memories.

An instruction frame includes:

- An instruction code (size: 8 bits). The instruction is optional in some cases (see section Continuous Read mode).
- An address (size: 24 bits or 32 bits). The address is optional but is required by instructions such as READ, PROGRAM, ERASE, LOCK. By default the address is 24 bits long, but it can be 32 bits long to support serial Flash memories larger than 128 Mbits (16 Mbytes).
- An option code (size: 1/2/4/8 bits). The option code is not required, but it is useful to activate the XIP mode or the Continuous Read mode (see section Continuous Read mode) for READ instructions, in some serial Flash memory devices. These modes improve the data read latency.
- Dummy cycles. Dummy cycles are optional but required by some READ instructions.
- Data bytes are optional. Data bytes are present for data transfer instructions such as READ or PROGRAM.

The instruction code, the address/option and the data can be sent with Single-bit SPI, Dual SPI or Quad SPI protocols.

Figure 37-8. Instruction Frame



37.6.5.2 Instruction Frame Transmission

To send an instruction frame, the user must first configure the address to send by writing the field ADDR in the Instruction Address register (QSPI_IAR). This step is required if the instruction frame includes an address and no data. When data is present, the address of the instruction is defined by the address of the data accesses in the QSPI memory space, not by QSPI_IAR.

If the instruction frame includes the instruction code and/or the option code, the user must configure the instruction code and/or the option code to send by writing the fields INST and OPT in the Instruction Code register (QSPI_ICR).

Then, the user must write QSPI_IFR to configure the instruction frame depending on which instruction must be sent. If the instruction frame does not include data, writing in this register triggers the send of the instruction frame in the QSPI. If the instruction frame includes data, the send of the instruction frame is triggered by the first data access in the QSPI memory space.

The instruction frame is configured by the following bits and fields of QSPI_IFR:

- WIDTH field—used to configure which data lanes are used to send the instruction code, the address, the option code and to transfer the data. It is possible to use two unidirectional data lanes (MISO-MOSI Single-bit SPI), two bidirectional data lanes (QIO0-QIO1 Dual SPI) or four bidirectional data lanes (QIO0–QIO3 Quad SPI).
- INSTEN bit—used to enable the send of an instruction code.
- ADDREN bit—used to enable the send of an address after the instruction code.
- OPTEN bit—used to enable the send of an option code after the address.
- DATAEN bit—used to enable the transfer of data (READ or PROGRAM instruction).
- OPTL field—used to configure the option code length. The value written in OPTL must be consistent with the value written in the field WIDTH. For example: OPTL = 0 (1-bit option code) is not consistent with WIDTH = 6 (option code sent with QuadSPI protocol, thus the minimum length of the option code is 4 bits).
- ADDRL bit—used to configure the address length.
- TFRTYP field—used to define which type of data transfer must be performed.
- NBDUM field—used to configure the number of dummy cycles when reading data from the serial Flash memory. Between the address/option and the data, with some instructions, dummy cycles are inserted by the serial Flash memory.

See 37.7.12 QSPI_IFR.

If data transfer is enabled, the user can access the serial memory by reading or writing the QSPI memory space:

- To read in the serial memory, but not a memory data, for example a JEDEC-ID or the QSPI_SR, QSPI_IFR.TFRTYP must be written to ‘0’.
- To read in the serial memory, and particularly a memory data, TFRTYP must be written to ‘1’.
- To write in the serial memory, but not a memory data, for example writing the configuration or the QSPI_SR, TFRTYP must be written to ‘2’.
- If the user wants to write in the serial memory in particular to program a memory data, TFRTYP must be written to ‘3’ .

If QSPI_IFR.TFRTYP has a value other than ‘1’ and QSPI_MR.SMRM = 0, the address sent in the instruction frame is the address of the first system bus accesses. The addresses of the next accesses are not used by the QSPI. At each system bus access, an SPI transfer is performed with the same size. For example, a halfword system bus access leads to a 16-bit SPI transfer, and a byte system bus access leads to an 8-bit SPI transfer.

If SMRM = 1 and TFRTYP= (0 or 2), accesses are made via the QSPI registers and the address sent in the instruction frame is the address defined in QSPI_IAR.

Each time QSPI_IFR is written (in case of read access), or each time QSPI_TDR is written (in case of write transfer), an SPI transfer is performed with a byte size. Another byte is read each time QSPI_RDR is read (flag RDRF shows when a data can be read in QSPI_RDR) or written each time QSPI_TDR is written (flag TDRE shows when a new data can be written). The SPI transfer ends by writing QSPI_CR.LASTXFER.

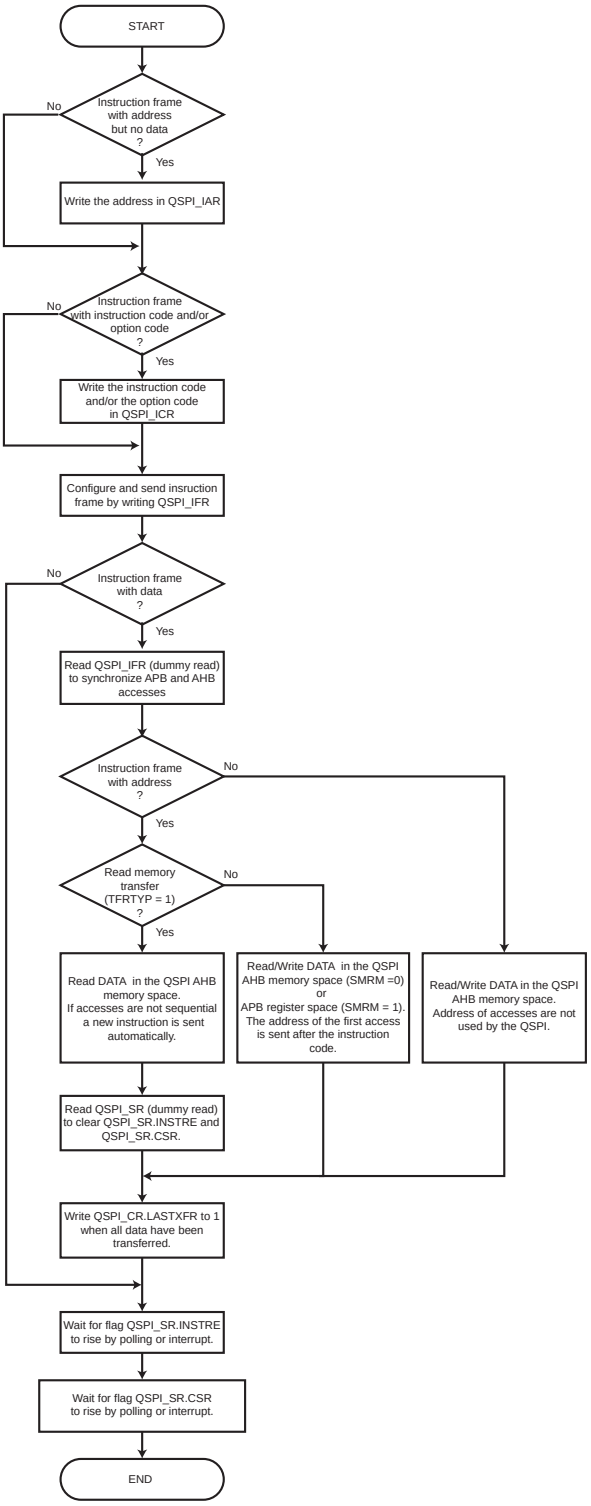
If TFRTYP = 1, the address of the first instruction frame is the one of the first read access in the QSPI memory space. Each time the read accesses become nonsequential (addresses are not consecutive), a new instruction frame is sent with the last system bus access address. In this way, the system can read data at a random location in the serial memory. The size of the SPI transfers may differ from the size of the system bus read accesses.

When data transfer is not enabled, the end of the instruction frame is indicated when QSPI_SR.INSTRE rises. (The QSPI_SR.CSR flag indicates when chip select rises. A delay between these flags may exist in case of high clock division or a high DLYBCT value).

When data transfer is enabled, the user must indicate when the data transfer is completed in the QSPI memory space by setting QSPI_CR.LASTXFR. The end of the instruction frame is indicated when QSPI_SR.INSTRE rises.

The following figure illustrates instruction transmission management.

Figure 37-9. Instruction Transmission Flow Diagram



37.6.5.3 Read Memory Transfer

The user can access the data of the serial memory by sending an instruction with QSPI_IFR.DATAEN = 1 and QSPI_IFR.TFRTYP = 1.

In this mode, the QSPI is able to read data at random address into the serial Flash memory, allowing the CPU to execute code directly from it (XIP execute-in-place).

In order to fetch data, the user must first configure the instruction frame by writing the QSPI_IFR. Then data can be read at any address in the QSPI address space mapping. The address of the system bus read accesses match the address of the data inside the serial Flash memory.

When Fetch mode is enabled, several instruction frames can be sent before writing QSPI_CR.LASTXFR. Each time the system bus read accesses become nonsequential (addresses are not consecutive), a new instruction frame is sent with the corresponding address.

37.6.5.4 Continuous Read Mode

The QSPI is compatible with the Continuous Read mode which is implemented in some serial Flash memories.

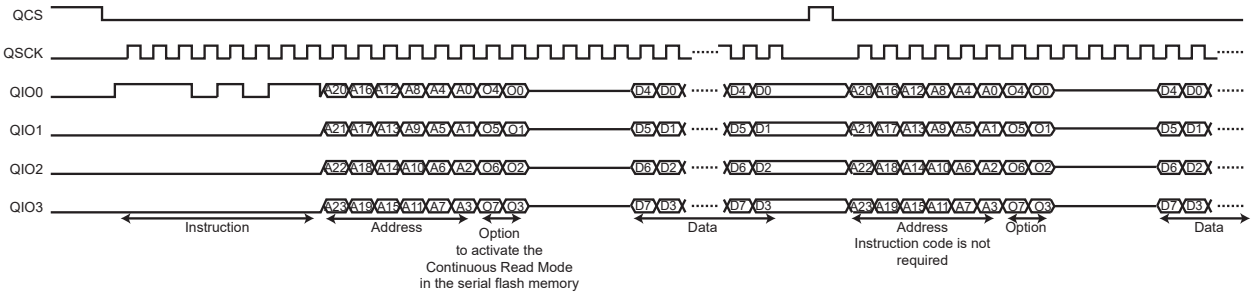
In Continuous Read mode, the instruction overhead is reduced by excluding the instruction code from the instruction frame. When the Continuous Read mode is activated in a serial Flash memory by a specific option code, the instruction code is stored in the memory. For the next instruction frames, the instruction code is not required as the memory uses the stored one.

In the QSPI, Continuous Read mode is used when reading data from the memory (QSPI_IFR.TFRTYP = 1). The addresses of the system bus read accesses are often nonsequential and this leads to many instruction frames that have the same instruction code. By disabling the send of the instruction code, the Continuous Read mode reduces the access time of the data.

To be functional, this mode must be enabled in both the QSPI and the serial Flash memory. The Continuous Read mode is enabled in the QSPI by writing CRM to ‘1’ in the QSPI_IFR (TFRTYP must equal 1). The Continuous Read mode is enabled in the serial Flash memory by sending a specific option code.

⚠ CAUTION If the Continuous Read mode is not supported by the serial Flash memory or disabled, CRM bit must not be written to ‘1’, otherwise data read out of the serial Flash memory is unpredictable.

Figure 37-10. Continuous Read Mode



37.6.5.5 Instruction Frame Transmission Examples

All waveforms in the following examples describe SPI transfers in SPI Clock mode 0 (QSPI_SCR.CPOL = 0 and QSPI_SCR.CPHA = 0; see section Serial Clock Phase and Polarity).

All system bus accesses described below refer to the system bus address phase. System bus wait cycles and system bus data phases are not shown.

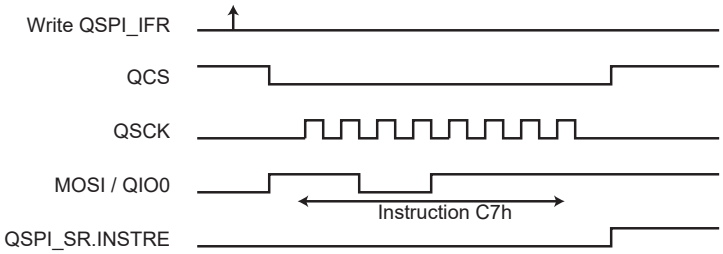
Example 1:

Instruction in Single-bit SPI, without address, without option, without data.

Command: CHIP ERASE (C7h).

- Write 0x0000_00C7 in QSPI_ICR.
- Write 0x0000_0010 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 37-11. Instruction Transmission Waveform 1



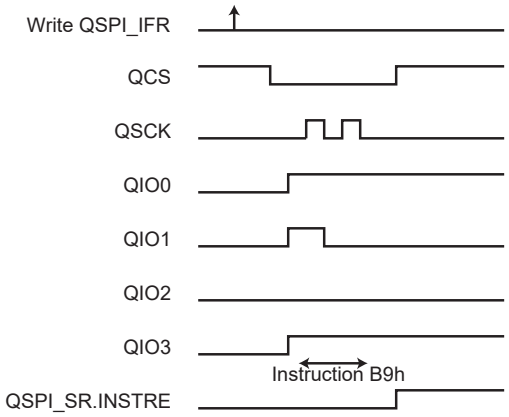
Example 2:

Instruction in Quad SPI, without address, without option, without data.

Command: POWER DOWN (B9h)

- Write 0x0000_00B9 in QSPI_ICR.
- Write 0x0000_0016 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 37-12. Instruction Transmission Waveform 2



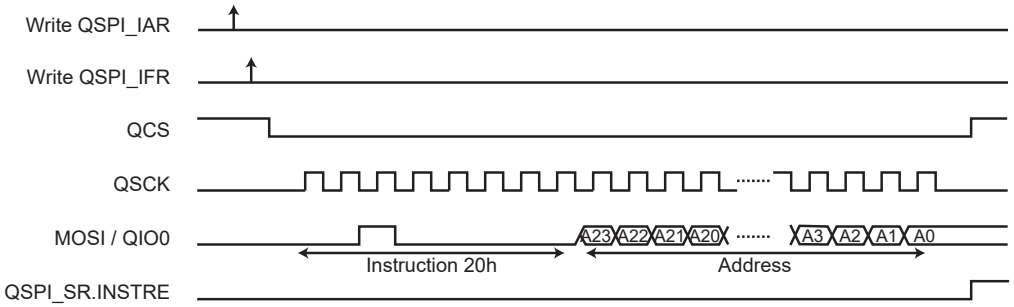
Example 3:

Instruction in Single-bit SPI, with address in Single-bit SPI, without option, without data.

Command: BLOCK ERASE (20h)

- Write the address (of the block to erase) in QSPI_AR.
- Write 0x0000_0020 in QSPI_ICR.
- Write 0x0000_0030 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 37-13. Instruction Transmission Waveform 3



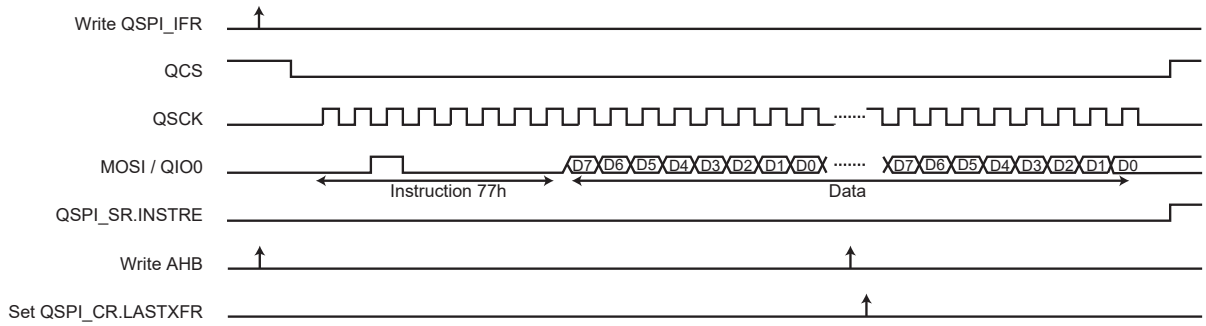
Example 4:

Instruction in Single-bit SPI, without address, without option, with data write in Single-bit SPI.

Command: SET BURST (77h)

- Write 0x0000_0077 in QSPI_ICR.
- Write 0x0000_2090 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Write data in the system bus memory space (0x1800_0000).
  The address of system bus write accesses is not used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 37-14. Instruction Transmission Waveform 4



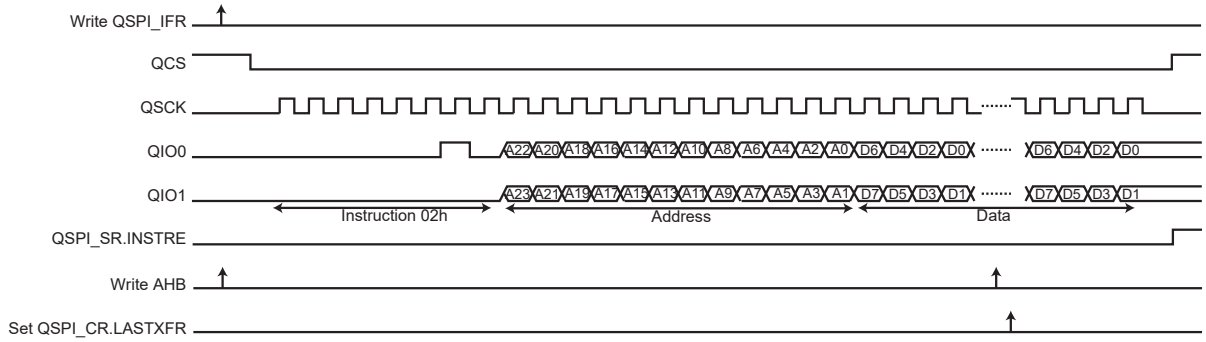
Example 5:

Instruction in Single-bit SPI, with address in Dual SPI, without option, with data write in Dual SPI.

Command: BYTE/PAGE PROGRAM (02h)

- Write 0x0000_0002 in QSPI_ICR.
- Write 0x0000_30B3 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Write data in the QSPI system bus memory space (0x1800_0000).
  The address of the first system bus write access is sent in the instruction frame.
  The address of the next system bus write accesses is not used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 37-15. Instruction Transmission Waveform 5



Example 6:

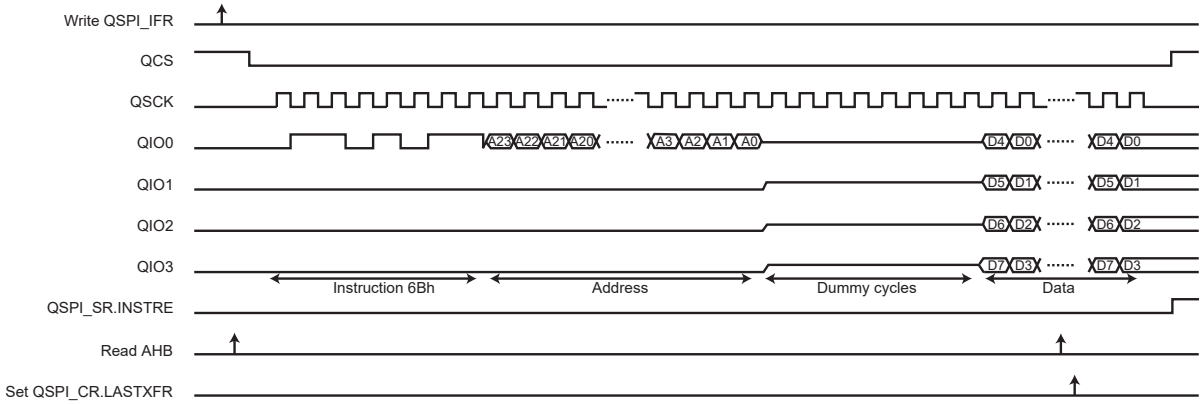
Instruction in Single-bit SPI, with address in Single-bit SPI, without option, with data read in Quad SPI, with eight dummy cycles.

Command: QUAD_OUTPUT READ ARRAY (6Bh)

- Write 0x0000_006B in QSPI_ICR.
- Write 0x0008_10B2 in QSPI_IFR.
- Read QSPI_IR (dummy read) to synchronize system bus accesses.

- Read data in the QSPI system bus memory space (0x1800_0000).
  The address of the first system bus read access is sent in the instruction frame.
  The address of the next system bus read accesses is not used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 37-16. Instruction Transmission Waveform 6



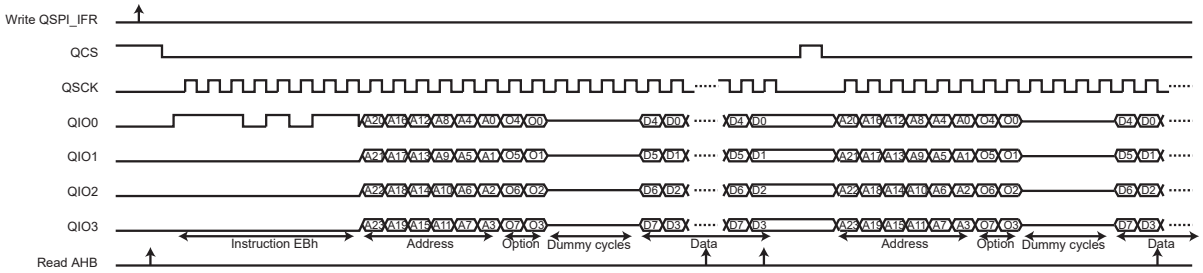
Example 7:

Instruction in Single-bit SPI, with address and option in Quad SPI, with data read in Quad SPI, with four dummy cycles, with fetch and continuous read.

Command: FAST READ QUAD I/O (EBh) - 8-BIT OPTION (0x30h)

- Write 0x0030_00EB in QSPI_ICR.
- Write 0x0004_33F4 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x1800_0000).
  Fetch is enabled, the address of the system bus read accesses is always used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 37-17. Instruction Transmission Waveform 7



Example 8:

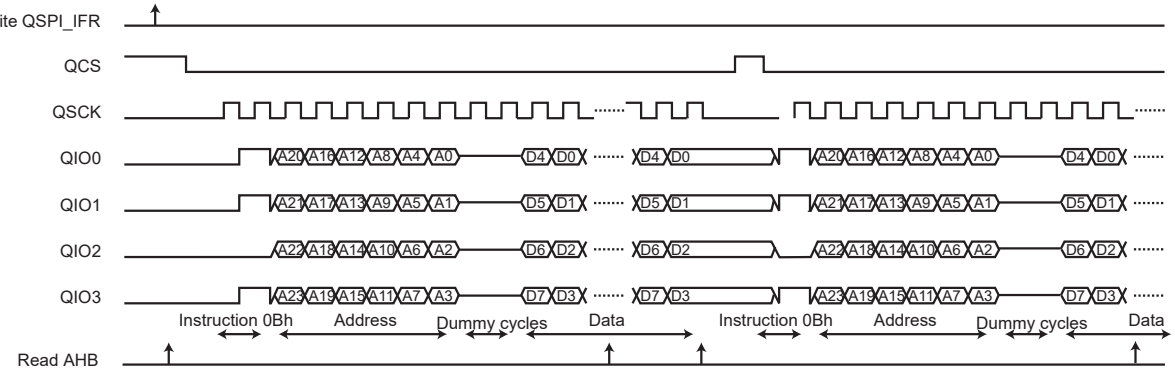
Instruction in Quad SPI, with address in Quad SPI, without option, with data read in Quad SPI, with two dummy cycles, with fetch.

Command: HIGH-SPEED READ (0Bh)

- Write 0x0000_000B in QSPI_ICR.
- Write 0x0002_20B6 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x1800_0000).
  Fetch is enabled, the address of the system bus read accesses is always used.
- Write a ‘1’ to QSPI_CR.LASTXFR.

- Wait for QSPI_SR.INSTRE to rise.

Figure 37-18. Instruction Transmission Waveform 8



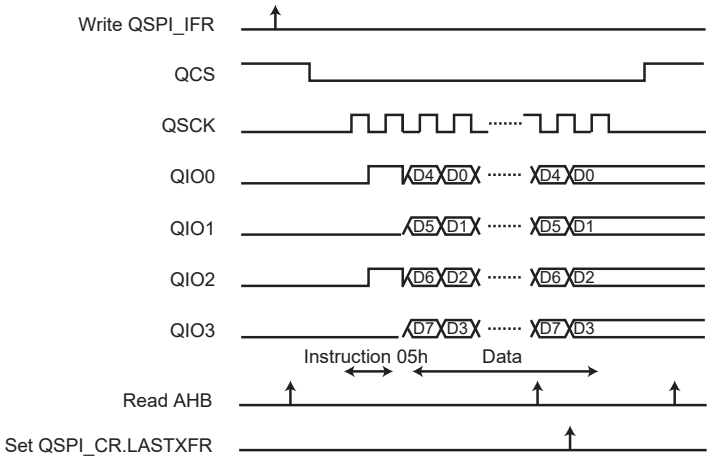
Example 9:

Instruction in Quad SPI, without address, without option, with data read in Quad SPI, without dummy cycles, without fetch.

Command: HIGH-SPEED READ (05h)

- Write 0x0000_0005 in QSPI_ICR.
- Write 0x0000_0096 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x1800_0000).
  Fetch is disabled.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 37-19. Instruction Transmission Waveform 9



Example 10:

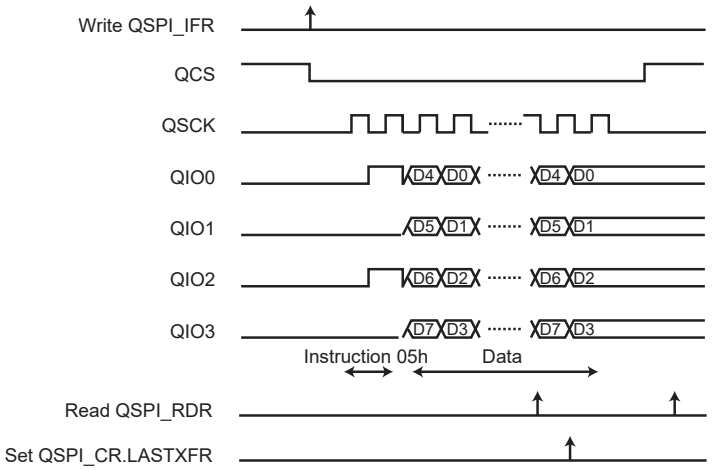
Instruction in Quad SPI, without address, without option, with data read in Quad SPI, without dummy cycles, without fetch, read launched through APB interface.

Command: HIGH-SPEED READ (05h)

- Set SMRM to ‘1’ in QSPI_MR
- Write 0x0000_0005 in QSPI_ICR.
- Write 0x0100_0096 in QSPI_IFR (will start the transfer).
- Wait flag RDRF and Read data in the QSPI_RDR register
  Fetch is disabled.

- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 37-20. Instruction Transmission Waveform 10



37.6.6 Scrambling/Unscrambling Function

The scrambling/unscrambling function cannot be performed on devices other than memories. Data is scrambled when written to memory and unscrambled when data is read.

The external data lines can be scrambled in order to prevent intellectual property data located in off-chip memories from being easily recovered by analyzing data at the package pin level of either the microcontroller or the QSPI slave device (e.g., memory).

The scrambling/unscrambling function can be enabled by writing a ‘1’ to the SCREN bit in the QSPI Scrambling Mode Register (QSPI_SMR).

The scrambling and unscrambling are performed on-the-fly without impacting the throughput.

The scrambling method depends on the user-configurable user scrambling key (field USRK) in the QSPI Scrambling Key Register (QSPI_SKR). QSPI_SKR is only accessible in Write mode.

If QSPI_SMR.RVDIS is written to ‘0’, the scrambling/unscrambling algorithm includes the user scrambling key plus a random value depending on device processing characteristics. Data scrambled by a given microcontroller cannot be unscrambled by another.

If QSPI_SMR.RVDIS is written to ‘1’, the scrambling/unscrambling algorithm includes only the user scrambling key. No random value is part of the key.

The user scrambling key or the seed for key generation must be securely stored in a reliable nonvolatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

37.6.7 Register Write Protection

To prevent any single software error from corrupting QSPI behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the QSPI Write Protection Mode Register (QSPI_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the QSPI Write Protection Status Register (QSPI_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the QSPI_WPSR.

The following registers can be write-protected when WPEN is set in QSPI_WPMR:

- QSPI Mode Register
- QSPI Serial Clock Register
- QSPI Scrambling Mode Register
- QSPI Scrambling Key Register

The following registers can be write-protected when WPCREN is set in QSPI_WPMR:

- QSPI Control Register

The following registers can be write-protected when WPITEN is set in QSPI_WPMR:

- QSPI Interrupt Enable Register
- QSPI Interrupt Disable Register

37.7 Register Summary

Offset	Name	Bit Pos.								
0x00	QSPI_CR	7:0	SWRST						QSPIDIS	QSPIEN
		15:8								
		23:16								
		31:24								LASTXFER
0x04	QSPI_MR	7:0			CSMODE[1:0]		SMRM	WDRBT	LLB	SMM
		15:8					NBBITS[3:0]			
		23:16	DLYBCT[7:0]							
		31:24	DLYCS[7:0]							
0x08	QSPI_RDR	7:0	RD[7:0]							
		15:8	RD[15:8]							
		23:16								
		31:24								
0x0C	QSPI_TDR	7:0	TD[7:0]							
		15:8	TD[15:8]							
		23:16								
		31:24								
0x10	QSPI_SR	7:0					OVRES	TXEMPTY	TDRE	RDRF
		15:8						INSTRE	CSS	CSR
		23:16								
		31:24								QSPIENS
0x14	QSPI_IER	7:0					OVRES	TXEMPTY	TDRE	RDRF
		15:8						INSTRE	CSS	CSR
		23:16								
		31:24								
0x18	QSPI_IDR	7:0					OVRES	TXEMPTY	TDRE	RDRF
		15:8						INSTRE	CSS	CSR
		23:16								
		31:24								
0x1C	QSPI_IMR	7:0					OVRES	TXEMPTY	TDRE	RDRF
		15:8						INSTRE	CSS	CSR
		23:16								
		31:24								
0x20	QSPI_SCR	7:0							CPHA	CPOL
		15:8	SCBR[7:0]							
		23:16	DLYBS[7:0]							
		31:24								
0x24 ... 0x2F	Reserved									
0x30	QSPI_IAR	7:0	ADDR[7:0]							
		15:8	ADDR[15:8]							
		23:16	ADDR[23:16]							
		31:24	ADDR[31:24]							
0x34	QSPI_ICR	7:0	INST[7:0]							
		15:8								
		23:16	OPT[7:0]							
		31:24								
0x38	QSPI_IFR	7:0	DATAEN	OPTEN	ADDREN	INSTEN		WIDTH[2:0]		
		15:8		CRM	TFRTYP[1:0]			ADDRL	OPTL[1:0]	
		23:16				NBDUM[4:0]				
		31:24								
0x3C ... 0x3F	Reserved									

.....continued

Offset	Name	Bit Pos.								
0x40	QSPI_SMR	7:0							RVDIS	SCREN
		15:8								
		23:16								
		31:24								
0x44	QSPI_SKR	7:0	USRK[7:0]							
		15:8	USRK[15:8]							
		23:16	USRK[23:16]							
		31:24	USRK[31:24]							
0x48 ... 0xE3	Reserved									
0xE4	QSPI_WPMR	7:0						WPCREN	WPITEN	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0xE8	QSPI_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16								
		31:24								

37.7.1 QSPI Control Register

Name: QSPI_CR
Offset: 0x00
Reset: –
Property: Write-only

This register can only be written if the WPCREN bit is cleared in the QSPI Write Protection Mode Register .

Bit	31	30	29	28	27	26	25	24
								LASTXFER
Access								W
Reset								–

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	SWRST						QSPIDIS	QSPIEN
Access	W						W	W
Reset	–						–	–

Bit 24 – LASTXFER Last Transfer

Value	Description
0	No effect.
1	The chip select is deasserted after the character written in QSPI_TDR.TD has been transferred.

Bit 7 – SWRST QSPI Software Reset
DMA channels are not affected by software reset.

Value	Description
0	No effect.
1	Reset the QSPI. A software-triggered hardware reset of the QSPI interface is performed.

Bit 1 – QSPIDIS QSPI Disable
As soon as QSPIDIS is set, the QSPI finishes its transfer.
All pins are set in Input mode and no data is received or transmitted.
If a transfer is in progress, the transfer is finished before the QSPI is disabled.
If both QSPIEN and QSPIDIS are equal to one when QSPI_CR is written, the QSPI is disabled.

Value	Description
0	No effect.
1	Disables the QSPI.

Bit 0 – QSPIEN QSPI Enable

Value	Description
0	No effect.
1	Enables the QSPI to transfer and receive data.

37.7.2 QSPI Mode Register

Name: QSPI_MR
Offset: 0x04
Reset: 0x00000000
Property: Read/Write

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	DLYCS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DLYBCT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
					NBBITS[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
			CSMODE[1:0]		SMRM	WDRBT	LLB	SMM
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bits 31:24 – DLYCS[7:0] Minimum Inactive QCS Delay
This field defines the minimum delay between the deactivation and the activation of QCS. The DLYCS time guarantees the slave minimum deselect time.
If DLYCS written to ‘0’, one peripheral clock period is inserted by default.
Otherwise, the following equation determines the delay:
DLYCS = Minimum inactive × $f_{peripheral\ clock}$

Bits 23:16 – DLYBCT[7:0] Delay Between Consecutive Transfers
This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.
When DLYBCT is written to ‘0’, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers. In Serial Memory mode (SMM = 1), DLYBCT must be written to ‘0’ and no delay is inserted.
Otherwise, the following equation determines the delay:
DLYBCT = (Delay Between Consecutive Transfers × $f_{peripheral\ clock}$) / 32

Bits 11:8 – NBBITS[3:0] Number Of Bits Per Transfer
NBBITS is used only when SMM is set to '0'.

Value	Name	Description
0	8_BIT	8 bits for transfer
8	16_BIT	16 bits for transfer

Bits 5:4 – CSMODE[1:0] Chip Select Mode
The CSMODE field determines how the chip select is deasserted
Note: This field is forced to LASTXFER when SMM is written to ‘1’.

Value	Name	Description
0	NOT_RELOADED	The chip select is deasserted if QSPI_TDR.TD has not been reloaded before the end of the current transfer.

Value	Name	Description
1	LASTXFER	The chip select is deasserted when the bit LASTXFER is written to '1' and the character written in QSPI_TDR.TD has been transferred.
2	SYSTEMATICALLY	The chip select is deasserted systematically after each transfer.

Bit 3 – SMRM Serial Memory Register Mode

Value	Description
0	Serial Memory registers are written via AHB access. See section Instruction Frame Transmission for details.
1	Serial Memory registers are written via APB access. See section Instruction Frame Transmission for details.

Bit 2 – WDRBT Wait Data Read Before Transfer
0 (DISABLED): No effect. In SPI mode, a transfer can be initiated whatever the state of the QSPI_RDR is.
1 (ENABLED): In SPI mode, a transfer can start only if the QSPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception.

Bit 1 – LLB Local Loopback Enable
0 (DISABLED): Local loopback path disabled.
1 (ENABLED): Local loopback path enabled.
LLB controls the local loopback on the data serializer for testing in SPI mode only. (MISO is internally connected on MOSI).

Bit 0 – SMM Serial Memory Mode
0 (SPI): The QSPI is in SPI mode.
1 (MEMORY): The QSPI is in Serial Memory mode.

37.7.3 QSPI Receive Data Register

Name: QSPI_RDR
Offset: 0x08
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – RD[15:0] Receive Data
Data received by the QSPI is stored in this register right-justified. Unused bits read zero.

37.7.4 QSPI Transmit Data Register

Name: QSPI_TDR
Offset: 0x0C
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TD[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 15:0 – TD[15:0] Transmit Data
Data to be transmitted by the QSPI is stored in this register. Information to be transmitted must be written to the Transmit Data register in a right-justified format.

37.7.5 QSPI Status Register

Name: QSPI_SR
Offset: 0x10
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
								QSPIENS
Access								R
Reset								0

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						INSTRE	CSS	CSR
Access						R	R	R
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
					OVRES	TXEMPTY	TDRE	RDRF
Access					R	R	R	R
Reset					0	0	0	0

Bit 24 – QSPIENS QSPI Enable Status

Value	Description
0	QSPI is disabled.
1	QSPI is enabled.

Bit 10 – INSTRE Instruction End Status (cleared on read)

Value	Description
0	No instruction end has been detected since the last read of QSPI_SR.
1	At least one instruction end has been detected since the last read of QSPI_SR.

Bit 9 – CSS Chip Select Status

Value	Description
0	The chip select is asserted.
1	The chip select is not asserted.

Bit 8 – CSR Chip Select Rise (cleared on read)

Value	Description
0	No chip select rise has been detected since the last read of QSPI_SR.
1	At least one chip select rise has been detected since the last read of QSPI_SR.

Bit 3 – OVRES Overrun Error Status (cleared on read)
An overrun occurs when QSPI_RDR is loaded at least twice from the serializer since the last read of the QSPI_RDR.

Value	Description
0	No overrun has been detected since the last read of QSPI_SR.
1	At least one overrun error has occurred since the last read of QSPI_SR.

Bit 2 – TXEMPTY Transmission Registers Empty (cleared by writing QSPI_TDR)

Value	Description
0	As soon as data is written in QSPI_TDR.

Value	Description
1	QSPI_TDR and the internal shifter are empty. If a transfer delay has been defined, TXEMPTY is set after the completion of such delay.

Bit 1 – TDRE Transmit Data Register Empty (cleared by writing QSPI_TDR)
TDRE equals zero when the QSPI is disabled or at reset. The QSPI enable command sets this bit to one.

Value	Description
0	Data has been written to QSPI_TDR and not yet transferred to the serializer.
1	The last data written in the QSPI_TDR has been transferred to the serializer.

Bit 0 – RDRF Receive Data Register Full (cleared by reading QSPI_RDR)

Value	Description
0	No data has been received since the last read of QSPI_RDR.
1	Data has been received and the received data has been transferred from the serializer to QSPI_RDR since the last read of QSPI_RDR.

37.7.6 QSPI Interrupt Enable Register

Name: QSPI_IER
Offset: 0x14
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the QSPI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						INSTRE	CSS	CSR
Access						W	W	W
Reset						–	–	–

Bit	7	6	5	4	3	2	1	0
					OVRES	TXEMPTY	TDRE	RDRF
Access					W	W	W	W
Reset					–	–	–	–

Bit 10 – INSTRE Instruction End Interrupt Enable

Bit 9 – CSS Chip Select Status Interrupt Enable

Bit 8 – CSR Chip Select Rise Interrupt Enable

Bit 3 – OVRES Overrun Error Interrupt Enable

Bit 2 – TXEMPTY Transmission Registers Empty Enable

Bit 1 – TDRE Transmit Data Register Empty Interrupt Enable

Bit 0 – RDRF Receive Data Register Full Interrupt Enable

37.7.7 QSPI Interrupt Disable Register

Name: QSPI_IDR
Offset: 0x18
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the QSPI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						INSTRE	CSS	CSR
Access						W	W	W
Reset						–	–	–

Bit	7	6	5	4	3	2	1	0
					OVRES	TXEMPTY	TDRE	RDRF
Access					W	W	W	W
Reset					–	–	–	–

Bit 10 – INSTRE Instruction End Interrupt Disable

Bit 9 – CSS Chip Select Status Interrupt Disable

Bit 8 – CSR Chip Select Rise Interrupt Disable

Bit 3 – OVRES Overrun Error Interrupt Disable

Bit 2 – TXEMPTY Transmission Registers Empty Disable

Bit 1 – TDRE Transmit Data Register Empty Interrupt Disable

Bit 0 – RDRF Receive Data Register Full Interrupt Disable

37.7.8 QSPI Interrupt Mask Register

Name: QSPI_IMR
Offset: 0x1C
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						INSTRE	CSS	CSR
Access						R	R	R
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
					OVRES	TXEMPTY	TDRE	RDRF
Access					R	R	R	R
Reset					0	0	0	0

Bit 10 – INSTRE Instruction End Interrupt Mask

Bit 9 – CSS Chip Select Status Interrupt Mask

Bit 8 – CSR Chip Select Rise Interrupt Mask

Bit 3 – OVRES Overrun Error Interrupt Mask

Bit 2 – TXEMPTY Transmission Registers Empty Mask

Bit 1 – TDRE Transmit Data Register Empty Interrupt Mask

Bit 0 – RDRF Receive Data Register Full Interrupt Mask

37.7.9 QSPI Serial Clock Register

Name: QSPI_SCR
Offset: 0x20
Reset: 0x00000000
Property: Read/Write

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	DLYBS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SCBR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
							CPHA	CPOL
Access							R/W	R/W
Reset							0	0

Bits 23:16 – DLYBS[7:0] Delay Before QSCK
This field defines the delay from QCS valid to the first valid QSCK transition.
When DLYBS equals zero, the QCS valid to QSCK transition is 1/2 the QSCK clock period.
Otherwise, the following equation determines the delay:
DLYBS = Delay Before QSCK × $f_{peripheral\ clock}$

Bits 15:8 – SCBR[7:0] Serial Clock Baud Rate
The QSPI uses a modulus counter to derive the QSCK baud rate from the peripheral clock. The baud rate is selected by writing a value from 0 to 255 in the SCBR field. The following equation determines the QSCK baud rate:
SCBR = ($f_{peripheral\ clock}$ / QSCK Baudrate) - 1

Bit 1 – CPHA Clock Phase
CPHA determines which edge of QSCK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

Value	Description
0	Data is captured on the leading edge of QSCK and changed on the following edge of QSCK.
1	Data is changed on the leading edge of QSCK and captured on the following edge of QSCK.

Bit 0 – CPOL Clock Polarity
CPOL is used to determine the inactive state value of the serial clock (QSCK). It is used with CPHA to produce the required clock/data relationship between master and slave devices.

Value	Description
0	The inactive state value of QSCK is logic level zero.
1	The inactive state value of QSCK is logic level one.

37.7.10 QSPI Instruction Address Register

Name: QSPI_IAR
Offset: 0x30
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ADDR[31:0] Address
Address to send to the serial Flash memory in the instruction frame.

37.7.11 QSPI Instruction Code Register

Name: QSPI_ICR
Offset: 0x34
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	OPT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	INST[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:16 – OPT[7:0] Option Code
Option code to send to the serial Flash memory.

Bits 7:0 – INST[7:0] Instruction Code
Instruction code to send to the serial Flash memory.

37.7.12 QSPI Instruction Frame Register

Name: QSPI_IFR
Offset: 0x38
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
				NBDUM[4:0]				
Access								
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
		CRM	TFRTYP[1:0]			ADDRL	OPTL[1:0]	
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bit	7	6	5	4	3	2	1	0
	DATAEN	OPTEN	ADDREN	INSTEN		WIDTH[2:0]		
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset	0	0	0	0		0	0	0

Bits 20:16 – NBDUM[4:0] Number Of Dummy Cycles
The NBDUM field defines the number of dummy cycles required by the serial Flash memory before data transfer.

Bit 14 – CRM Continuous Read Mode
0 (DISABLED): Continuous Read mode is disabled.
1 (ENABLED): Continuous Read mode is enabled.

Bits 13:12 – TFRTYP[1:0] Data Transfer Type

Value	Name	Description
0	TRSFR_READ	Read transfer from the serial memory. Scrambling is not performed. Read at random location (fetch) in the serial Flash memory is not possible.
1	TRSFR_READ_MEMORY	Read data transfer from the serial memory. If enabled, scrambling is performed. Read at random location (fetch) in the serial Flash memory is possible.
2	TRSFR_WRITE	Write transfer into the serial memory. Scrambling is not performed.
3	TRSFR_WRITE_MEMORY	Write data transfer into the serial memory. If enabled, scrambling is performed.

Bit 10 – ADDRL Address Length
The ADDRL bit determines the length of the address.
0 (24_BIT): The address is 24 bits long.
1 (32_BIT): The address is 32 bits long.

Bits 9:8 – OPTL[1:0] Option Code Length
The OPTL field determines the length of the option code. The value written in OPTL must be consistent with the value written in the field WIDTH. For example, OPTL = 0 (1-bit option code) is not consistent with WIDTH = 6 (option code sent with QuadSPI protocol, thus the minimum length of the option code is 4 bits).

Value	Name	Description
0	OPTION_1BIT	The option code is 1 bit long.
1	OPTION_2BIT	The option code is 2 bits long.
2	OPTION_4BIT	The option code is 4 bits long.
3	OPTION_8BIT	The option code is 8 bits long.

Bit 7 – DATAEN Data Enable

Value	Description
0	No data is sent/received to/from the serial Flash memory.
1	Data is sent/received to/from the serial Flash memory.

Bit 6 – OPTEN Option Enable

Value	Description
0	The option is not sent to the serial Flash memory.
1	The option is sent to the serial Flash memory.

Bit 5 – ADDREN Address Enable

Value	Description
0	The transfer address is not sent to the serial Flash memory.
1	The transfer address is sent to the serial Flash memory.

Bit 4 – INSTEN Instruction Enable

Value	Description
0	The instruction is not sent to the serial Flash memory.
1	The instruction is sent to the serial Flash memory.

Bits 2:0 – WIDTH[2:0] Width of Instruction Code, Address, Option Code and Data

Value	Name	Description
0	SINGLE_BIT_SPI	Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Single-bit SPI
1	DUAL_OUTPUT	Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Dual SPI
2	QUAD_OUTPUT	Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Quad SPI
3	DUAL_IO	Instruction: Single-bit SPI / Address-Option: Dual SPI / Data: Dual SPI
4	QUAD_IO	Instruction: Single-bit SPI / Address-Option: Quad SPI / Data: Quad SPI
5	DUAL_CMD	Instruction: Dual SPI / Address-Option: Dual SPI / Data: Dual SPI
6	QUAD_CMD	Instruction: Quad SPI / Address-Option: Quad SPI / Data: Quad SPI

37.7.13 QSPI Scrambling Mode Register

Name: QSPI_SMR
Offset: 0x40
Reset: 0x00000000
Property: Read/Write

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							RVDIS	SCREN
Access							R/W	R/W
Reset							0	0

Bit 1 – RVDIS Scrambling/Unscrambling Random Value Disable

Value	Description
0	The scrambling/unscrambling algorithm includes the user scrambling key plus a random value that may differ between devices.
1	The scrambling/unscrambling algorithm includes only the user scrambling key.

Bit 0 – SCREN Scrambling/Unscrambling Enable
0 (DISABLED): The scrambling/unscrambling is disabled.
1 (ENABLED): The scrambling/unscrambling is enabled.

37.7.14 QSPI Scrambling Key Register

Name: QSPI_SKR
Offset: 0x44
Reset: –
Property: Write-only

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	USRK[31:24]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	USRK[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	USRK[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	USRK[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 31:0 – USRK[31:0] User Scrambling Key

37.7.15 QSPI Write Protection Mode Register

Name: QSPI_WPMR
Offset: 0xE4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
						WPCREN	WPITEN	WPEN
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x515350	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

Bit 2 – WPCREN Write Protection Control Register Enable

Value	Description
0	Disables the write protection on the Control register if WPKEY corresponds to 0x515350.
1	Enables the write protection on the Control register if WPKEY corresponds to 0x515350.

Bit 1 – WPITEN Write Protection Interrupt Enable

Value	Description
0	Disables the write protection on Interrupt registers if WPKEY corresponds to 0x515350.
1	Enables the write protection on Interrupt registers if WPKEY corresponds to 0x515350.

Bit 0 – WPEN Write Protection Enable

See section Register Write Protection for the list of registers that can be protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x515350 (QSP in ASCII)
1	Enables the write protection if WPKEY corresponds to 0x515350 (QSP in ASCII)

37.7.16 QSPI Write Protection Status Register

Name: QSPI_WPSR
Offset: 0xE8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 15:8 – WPVSRC[7:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protection violation has occurred since the last read of the QSPI_WPSR.
1	A write protection violation has occurred since the last read of the QSPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

38. Flexible Serial Communication Controller (FLEXCOM)

38.1 Description

The Flexible Serial Communication Controller (FLEXCOM) offers several serial communication protocols that are managed by the three submodules USART, SPI, and TWI.

The Universal Synchronous Asynchronous Receiver Transceiver (USART) provides one full-duplex universal synchronous asynchronous serial link. Data frame format is widely programmable (data length, parity, number of stop bits) to support a maximum of standards. The receiver implements parity error, framing error and overrun error detection. The receiver timeout enables handling variable-length frames and the transmitter timeguard facilitates communications with slow remote devices. Multidrop communications are also supported through address bit handling in reception and transmission.

The USART features three test modes: Remote Loopback, Local Loopback and Automatic Echo.

The USART supports specific operating modes providing interfaces on RS485, LIN, LON, , with ISO7816 T = 0 or T = 1 smart card slots, and infrared transceivers. The hardware handshaking feature enables an out-of-band flow control by automatic management of the pins RTS and CTS.

The USART supports the connection to the DMA Controller, which enables data transfers to the transmitter and from the receiver. The DMAC provides chained buffer management without any intervention of the processor.

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master” which controls the data flow, while the other devices act as “slaves” which have data shifted into and out by the master. Different CPUs can take turn being masters (multiple master protocol, contrary to single master protocol where one CPU is always the master while all of the others are always slaves). One master can simultaneously shift data into multiple slaves. However, only one slave can drive its output to write data back to the master at any given time.

A slave device is selected when the master asserts its NSS signal. If multiple slave devices exist, the master generates a separate slave select signal for each slave (NPCS).

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI)—This data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO)—This data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.
- Serial Clock (SPCK)—This control line is driven by the master and regulates the flow of the data bits. The master can transmit data at a variety of baud rates; there is one SPCK pulse for each bit that is transmitted.
- Slave Select (NSS)—This control line allows slaves to be turned on and off by hardware.

The Two-wire Interface (TWI) interconnects components on a unique two-wire bus, made up of one clock line and one data line with speeds of up to 400 kbits per second in Fast mode and up to 3.4 Mbits per second in High-speed Slave mode only, based on a byte-oriented transfer format. It can be used with any Atmel Two-wire Interface bus Serial EEPROM and I^2C-compatible devices, such as a Real-Time Clock (RTC), Dot Matrix/Graphic LCD Controller and temperature sensor. The TWI is programmable as a master or a slave with sequential or single-byte access. Multiple master capability is supported.

Arbitration of the bus is performed internally and puts the TWI in Slave mode automatically if the bus arbitration is lost.

A configurable baud rate generator permits the output data rate to be adapted to a wide range of core clock frequencies.

The following table lists the compatibility level of the Atmel Two-wire Interface in Master mode and a full I^2C compatible device.

Table 38-1. Atmel TWI Compatibility with I²C Standard

I²C Standard	Atmel TWI
Standard Mode Speed (100 kHz)	Supported
Fast Mode Speed (400 kHz)	Supported
High-speed Mode (Slave only, 3.4 MHz)	Supported
7- or 10-bit(1) Slave Addressing	Supported
Repeated Start (Sr) Condition	Supported
ACK and NACK Management	Supported
Input Filtering	Supported
Slope Control	Not Supported
Clock Stretching	Supported
Multi Master Capability	Supported

Notes: 1. 10-bit support in Master mode only

38.2 Embedded Characteristics

38.2.1 USART/UART Characteristics

- 8-byte Transmit and Receive FIFOs
- Programmable Baud Rate Generator
- Baud Rate can be Independent of the Processor/Peripheral Clock
- Comparison Function on Received Character
- 5-bit to 9-bit Full-duplex Synchronous or Asynchronous Serial Communications
  - 1, 1.5 or 2 Stop Bits in Asynchronous Mode or 1 or 2 Stop Bits in Synchronous Mode
  - Parity Generation and Error Detection
  - Framing Error Detection, Overrun Error Detection
  - Digital Filter on Receive Line
  - MSB- or LSB-first
  - Optional Break Generation and Detection
  - By 8 or by 16 Oversampling Receiver Frequency
  - Optional Hardware Handshaking RTS-CTS
  - Receiver Timeout and Transmitter Timeguard
  - Optional Multidrop Mode with Address Generation and Detection
- RS485 with Driver Control Signal
- ISO7816, T = 0 or T = 1 Protocols for Interfacing with Smart Cards
  - NACK Handling, Error Counter with Repetition and Iteration Limit
- IrDA Modulation and Demodulation
  - Communication at up to 115.2 kbit/s
- LIN Mode
  - Compliant with LIN 1.3 and LIN 2.0 specifications
  - Master or Slave
  - Processing of Frames with Up to 256 Data Bytes
  - Response Data Length can be Configurable or Defined Automatically by the Identifier
  - Self-synchronization in Slave Node Configuration
  - Automatic Processing and Verification of the “Synch Break” and the “Synch Field”

  - "Synch Break" Detection Even When Partially Superimposed with a Data Byte
  - Automatic Identifier Parity Calculation/Sending and Verification
  - Parity Sending and Verification Can be Disabled
  - Automatic Checksum Calculation/sending and Verification
  - Checksum Sending and Verification Can be Disabled
  - Support Both "Classic" and "Enhanced" Checksum Types
  - Full LIN Error Checking and Reporting
  - Frame Slot Mode: Master Allocates Slots to the Scheduled Frames Automatically
  - Generation of the Wakeup Signal
- LON Mode
  - Compliant with CEA-709 Specification
  - Full-layer 2 Implementation
  - Differential Manchester Encoding/Decoding (CDP)
  - Preamble Generation Including Bit- and Byte-sync Fields
  - LON Timings Handling (beta1, beta2, IDT, etc.)
  - CRC Generation and Checking
  - Automated Random Number Generation
  - Backlog Calculation and Update
  - Collision Detection Support
  - Supports Both comm_type = 1 and comm_type = 2 Modes
  - Clock Drift Tolerance Up to 16%
  - Optimal for Node-to-Node Communication (no embedded digital line filter)
- Test Modes
  - Remote Loopback, Local Loopback, Automatic Echo
- Supports Connection of:
  - Two DMA Controller (DMAC) Channels
  - Offers Buffer Transfer without Processor Intervention
- Register Write Protection

38.2.2 SPI Characteristics

- 16-byte Transmit and Receive FIFOs
- Master or Slave Serial Peripheral Bus Interface
  - 8-bit to 16-bit programmable data length per chip select
  - Programmable phase and polarity per chip select
  - Programmable transfer delay between consecutive transfers and delay before SPI clock per chip select
  - Programmable delay between chip selects
- Selectable Mode Fault Detection
- Master Mode Can Drive SPCK up to Peripheral Clock
- Master Mode Bit Rate Can Be Independent of the Processor/Peripheral Clock
- Slave Mode Operates on SPCK, Asynchronously with Core and Bus Clock
- Four Chip Selects with External Decoder Support Allow Communication with up to 15Peripherals
- Communication with Serial External Devices Supported
  - Serial memories, such as DataFlash and 3-wire EEPROMs
  - Serial peripherals, such as ADCs, DACs, LCD controllers, CAN controllers and sensors
  - External coprocessors
- Connection to DMA Channel Capabilities, Optimizing Data Transfers
  - One channel for the receiver
  - One channel for the transmitter
- Register Write Protection

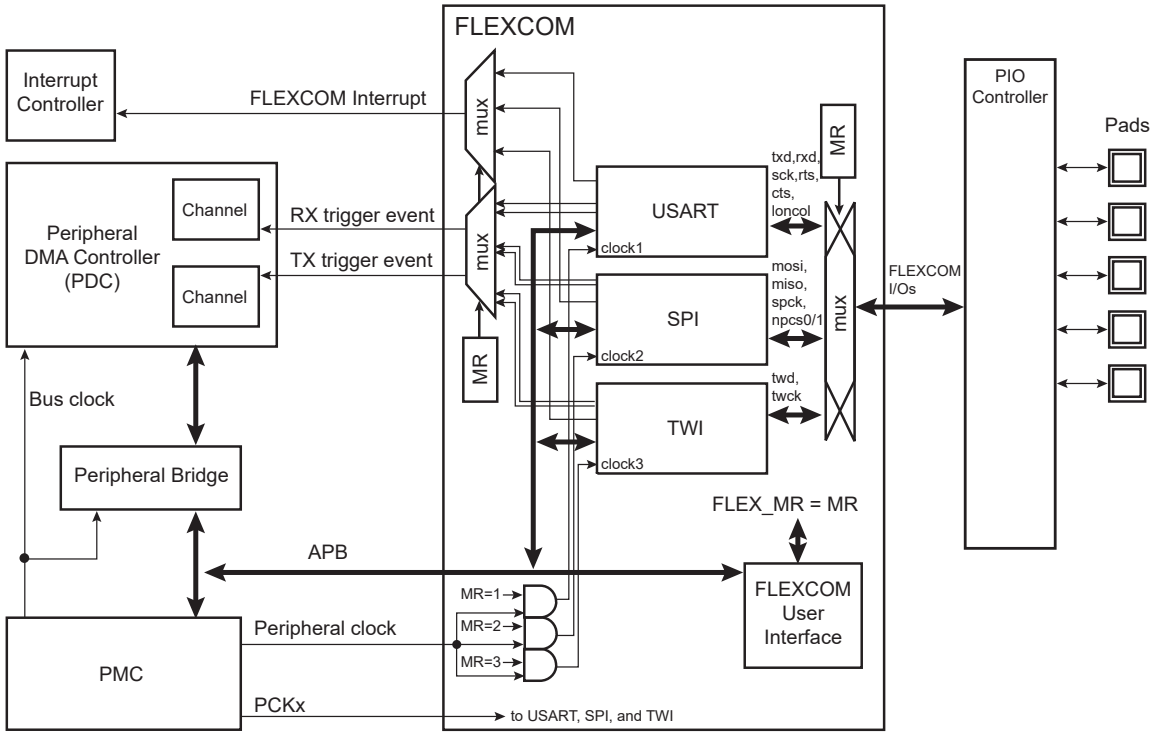
38.2.3 TWI/SMBus Characteristics

- 8-byte Transmit and Receive FIFOs
- Bit Rate: Up to 400 kbit/s in Fast Mode and 3.4 Mbit/s in High-Speed Mode (Slave Only)
- Bit Rate can be Independent of the Processor/Peripheral Clock
- SMBus Support
- Compatible with Atmel Two-wire Interface Serial Memory and I²C Compatible Devices(1)
- Master and Multi-Master Operation (Standard and Fast Mode Only)
- Slave Mode Operation (Standard, Fast and High-Speed Mode)
- One, Two or Three Bytes for Slave Address
- Sequential Read/Write Operations
- General Call Supported in Slave Mode
- Connection to DMA Controller Channels Optimizes Data Transfers
  - One Channel for the Receiver
  - One Channel for the Transmitter
- Register Write Protection

Note: 1. See table Atmel TWI Compatibility with I²C Standard for further details.

38.3 Block Diagram

Figure 38-1. FLEXCOM Block Diagram



38.4 I/O Lines Description

Table 38-2. I/O Lines Description

Name	Description			Type
	USART/UART	SPI	TWI	
FLEXCOM_IO0	TXD	MOSI	TWD	I/O
FLEXCOM_IO1	RXD	MISO	TWCK	I/O
FLEXCOM_IO2	SCK	SPCK	–	I/O
FLEXCOM_IO3	CTS	NPCS0/NSS	–	I/O
FLEXCOM_IO4	RTS	NPCS1	–	O
FLEXCOM_IO5	–	NPCS2	–	O
FLEXCOM_IO6	–	NPCS3	–	O
FLEXCOM_IO7	LONCOL	–	–	I

38.5 Product Dependencies

38.5.1 I/O Lines

The pins used for interfacing the FLEXCOM are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the desired FLEXCOM pins to their peripheral function. If I/O lines of the FLEXCOM are not used by the application, they can be used for other purposes by the PIO Controller.

38.5.2 Power Management

The peripheral clock is not continuously provided to the FLEXCOM. The programmer must first enable the FLEXCOM Clock in the Power Management Controller (PMC) before using the USART or SPI or TWI.

38.5.3 Interrupt Sources

The FLEXCOM interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the FLEXCOM interrupt requires the Interrupt Controller to be programmed first.

38.6 Register Accesses

Register accesses support 8-bit, 16-bit and 32-bit accesses, which means that only an 8-bit part of a 32-bit register can be written in one access, for instance. For this the access must be done with the right size at the right address.

8-bit, 16-bit and 32-bit accesses are supported for register accesses. However, a field in a register cannot be partially written (e.g., if a field is bigger than 8 bits, the whole field must be written).

This feature helps avoiding a read-modify-write process if only a small part of the register is to be modified.

38.7 USART Functional Description

38.7.1 Baud Rate Generator

The baud rate generator provides the bit period clock named “baud rate clock” to both the receiver and the transmitter.

Configuring the USCLKS field in FLEX_US_MR selects the baud rate generator clock from one of the following sources:

- the peripheral clock
- a division of the peripheral clock, the divider being product dependent, but generally set to 8
- a fully programmable generic clock (GCLK) provided by PMC and independent of processor/peripheral clock
- the external clock, available on the SCK pin

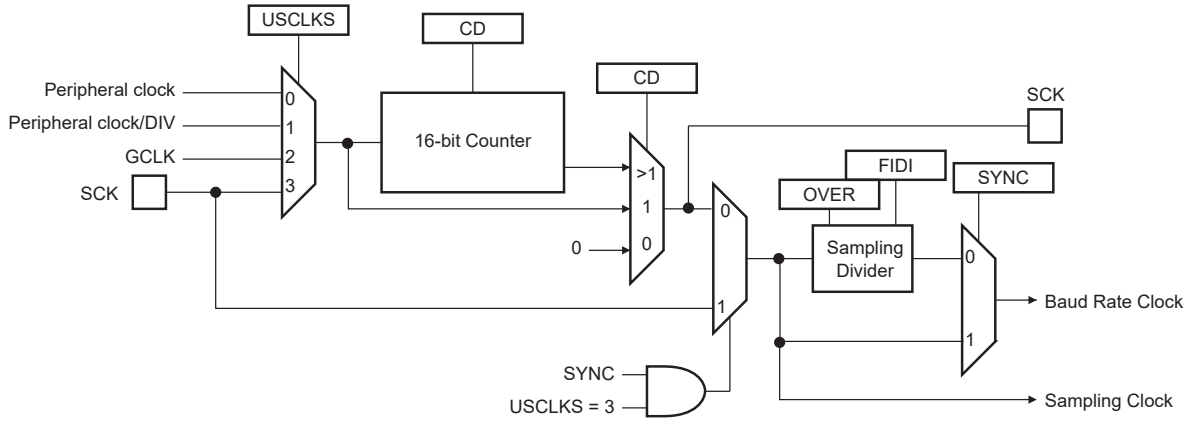
The baud rate generator is based upon a 16-bit divider, which is programmed with the CD field of the Baud Rate Generator Register (FLEX_US_BRGR). If a zero is written to CD, the baud rate generator does not generate any clock. If a one is written to CD, the divider is bypassed and becomes inactive.

If the external SCK clock is selected, the duration of the low and high levels of the signal provided on the SCK pin must be longer than a peripheral clock period. The frequency of the signal provided on SCK must be at least three times lower than peripheral clock .

If GCLK is selected, the baud rate is independent of the processor/peripheral clock and thus processor/peripheral clock frequency can be changed without affecting the USART transfer. The GCLK frequency must be at least three times lower than peripheral clock frequency.

If GCLK is selected (USCLKS = 2) and the SCK pin is driven (CLKO = 1), the CD field must be greater than 1.

Figure 38-2. Baud Rate Generator



38.7.1.1 Baud Rate in Asynchronous Mode

If the USART is programmed to operate in Asynchronous mode, the selected clock is first divided by CD, which is field-programmed in FLEX_US_BRGR. The resulting clock is provided to the receiver as a sampling clock and then divided by 16 or 8, depending on the programming of FLEX_US_MR.OVER.

If OVER is set, the receiver sampling is eight times higher than the baud rate clock. If OVER is cleared, the sampling is performed at 16 times the baud rate clock.

The baud rate is calculated as per the following formula:

$$\text{Baud rate} = \frac{\text{Selected Clock}}{(8(2 - \text{OVER})\text{CD})}$$

This gives a maximum baud rate of peripheral clock divided by 8, assuming that peripheral clock is the highest possible clock and that the OVER bit is set.

38.7.1.1.1 Baud Rate Calculation Example

The following table shows calculations of CD to obtain a baud rate at 38,400 bit/s for different source clock frequencies. It also shows the actual resulting baud rate and the error.

Table 38-3. Baud Rate Example (OVER = 0)

Source Clock (MHz)	Expected Baud Rate (bit/s)	Calculation Result	CD	Actual Baud Rate (bit/s)	Error
3,686,400	38,400	6.00	6	38,400.00	0.00%
4,915,200	38,400	8.00	8	38,400.00	0.00%

.....continued

Source Clock (MHz)	Expected Baud Rate (bit/s)	Calculation Result	CD	Actual Baud Rate (bit/s)	Error
5,000,000	38,400	8.14	8	39,062.50	1.70%
7,372,800	38,400	12.00	12	38,400.00	0.00%
8,000,000	38,400	13.02	13	38,461.54	0.16%
12,000,000	38,400	19.53	20	37,500.00	2.40%
12,288,000	38,400	20.00	20	38,400.00	0.00%
14,318,180	38,400	23.30	23	38,908.10	1.31%
14,745,600	38,400	24.00	24	38,400.00	0.00%
18,432,000	38,400	30.00	30	38,400.00	0.00%
24,000,000	38,400	39.06	39	38,461.54	0.16%
24,576,000	38,400	40.00	40	38,400.00	0.00%
25,000,000	38,400	40.69	40	38,109.76	0.76%
32,000,000	38,400	52.08	52	38,461.54	0.16%
32,768,000	38,400	53.33	53	38,641.51	0.63%
33,000,000	38,400	53.71	54	38,194.44	0.54%
40,000,000	38,400	65.10	65	38,461.54	0.16%
50,000,000	38,400	81.38	81	38,580.25	0.47%

The baud rate is calculated with the following formula:

Baud rate = MCK / CD × 16

The baud rate error is calculated with the following formula. It is not recommended to work with an error higher than 5%.

$$\text{Error} = 1 - \left(\frac{\text{Expected Baud Rate}}{\text{Actual Baud Rate}}\right)$$

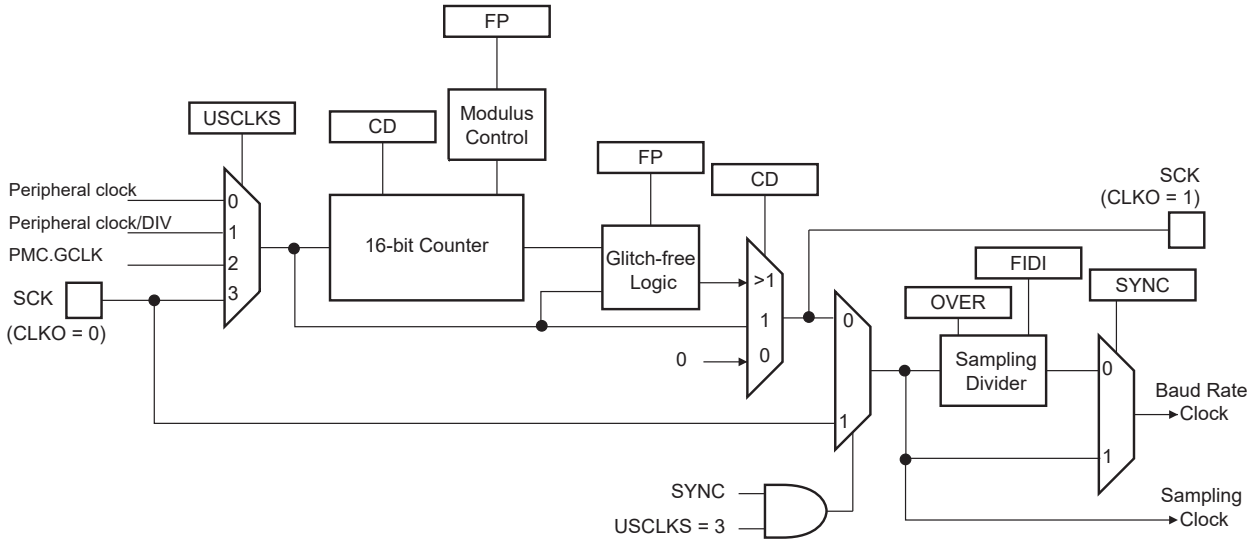
38.7.1.2 Fractional Baud Rate in Asynchronous Mode

The baud rate generator previously defined is subject to the following limitation: the output frequency changes by only integer multiples of the reference frequency. An approach to this problem is to integrate a fractional N clock generator that has a high resolution. The generator architecture is modified to obtain baud rate changes by a fraction of the reference source clock. This fractional part is programmed with the FP field in FLEX_US_BRGR. If FP is not 0, the fractional part is activated. The resolution is one eighth of the clock divider. The fractional baud rate is calculated using the following formula:

$$\text{Baud rate} = \frac{\text{Selected Clock}}{\left(8(2 - \text{OVER})\left(\text{CD} + \frac{\text{FP}}{8}\right)\right)}$$

The modified architecture is presented in the following figure.

Figure 38-3. Fractional Baud Rate Generator



WARNING When the value of field FP is greater than 0, the SCK (oversampling clock) generates nonconstant duty cycles. The SCK high duration is increased by "selected clock" period from time to time. The duty cycle depends on the value of the CD field.

38.7.1.3 Baud Rate in Synchronous Mode

If the USART is programmed to operate in Synchronous mode, the selected clock is simply divided by the CD field in FLEX_US_BRGR:

$$\text{Baud rate} = \frac{\text{Selected Clock}}{\text{CD}}$$

In Synchronous mode, if the external clock is selected (USCLKS = 3) and CLKO = 0 (Slave mode), the clock is provided directly by the signal on the USART SCK pin. No division is active. The value written in FLEX_US_BRGR has no effect. The external clock frequency must be at least three times lower than the system clock. In Synchronous mode master (USCLKS = 0 or 1, CLKO = 1), the receive part limits the SCK maximum frequency to $f_{peripheral\ clock}/3$.

When either the external clock SCK or the internal clock divided (peripheral clock/DIV or GCLK) is selected, the value programmed in CD must be even if the user has to ensure a 50:50 mark/space ratio on the SCK pin. If the peripheral clock is selected, the baud rate generator ensures a 50:50 duty cycle on the SCK pin, even if the value programmed in CD is odd.

38.7.1.4 Baud Rate in ISO 7816 Mode

The ISO7816 specification defines the bit rate with the following formula:

$$B = \frac{\text{Di}}{\text{Fi}} \times f$$

where:

- B is the bit rate
- Di is the bit rate adjustment factor
- Fi is the clock frequency division factor
- f is the ISO7816 clock frequency (Hz)

Di is a binary value encoded on a 4-bit field, named DI, as represented in the following table.

Table 38-4. Binary and Decimal Values for Di

DI field	0001	0010	0011	0100	0101	0110	1000	1001

Di (decimal)	1	2	4	8	16	32	12	20

Fi is a binary value encoded on a 4-bit field, named FI, as represented in the following table.

Table 38-5. Binary and Decimal Values for Fi

FI field	0000	0001	0010	0011	0100	0101	0110	1001	1010	1011	1100	1101
Fi (decimal)	372	372	558	744	1116	1488	1860	512	768	1024	1536	2048

The following table shows the resulting Fi/Di Ratio, which is the ratio between the ISO7816 clock and the baud rate clock.

Table 38-6. Possible Values for the Fi/Di Ratio

Fi/Di	372	558	744	1116	1488	1806	512	768	1024	1536	2048
1	372	558	744	1116	1488	1860	512	768	1024	1536	2048
2	186	279	372	558	744	930	256	384	512	768	1024
4	93	139.5	186	279	372	465	128	192	256	384	512
8	46.5	69.75	93	139.5	186	232.5	64	96	128	192	256
16	23.25	34.87	46.5	69.75	93	116.2	32	48	64	96	128
32	11.62	17.43	23.25	34.87	46.5	58.13	16	24	32	48	64
12	31	46.5	62	93	124	155	42.66	64	85.33	128	170.6
20	18.6	27.9	37.2	55.8	74.4	93	25.6	38.4	51.2	76.8	102.4

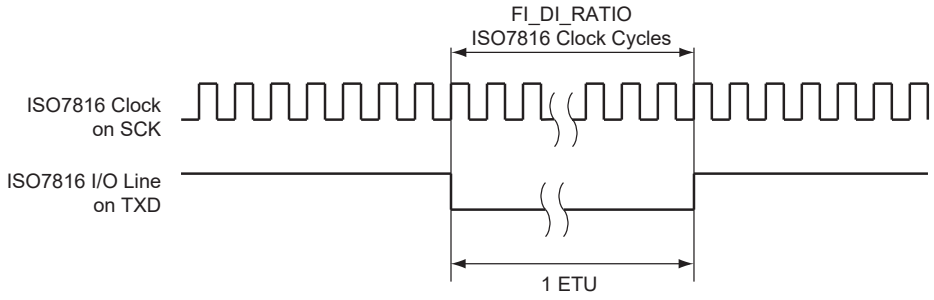
If the USART is configured in ISO7816 mode, the clock selected by the USCLKS field in FLEX_US_MR is first divided by the value programmed in field CD field in FLEX_US_BRGR. The resulting clock can be provided to the SCK pin to feed the smart card clock inputs. This means that FLEX_US_MR.CLKO can be set.

This clock is then divided by the value programmed in the FI_DI_RATIO field in the FI DI Ratio Register (FLEX_US_FIDI). This is performed by the Sampling Divider, which performs a division by up to 2047 in ISO7816 mode. The noninteger values of the Fi/Di Ratio are not supported and the user must program the FI_DI_RATIO field to a value as close as possible to the expected value.

The FI_DI_RATIO field resets to the value 0x174 (372 in decimal) and is the most common divider between the ISO7816 clock and the bit rate (Fi = 372, Di = 1).

The following figure shows the relation between the Elementary Time Unit, corresponding to a bit time, and the ISO 7816 clock.

Figure 38-4. Elementary Time Unit (ETU)



38.7.2 Receiver and Transmitter Control

After reset, the receiver is disabled. The user must enable the receiver by setting the RXEN bit in the USART Control Register (FLEX_US_CR). However, the receiver registers can be programmed before the receiver clock is enabled.

After reset, the transmitter is disabled. The user must enable it by setting the TXEN bit in FLEX_US_CR. However, the transmitter registers can be programmed before being enabled.

The receiver and the transmitter can be enabled together or independently.

At any time, the software can perform a reset on the receiver or the transmitter of the USART by setting the corresponding bit, RSTRX and RSTTX respectively, in FLEX_US_CR. The software resets clear the status flag and reset internal state machines but the user interface configuration registers hold the value configured prior to software reset. Regardless of what the receiver or the transmitter is performing, the communication is immediately stopped.

The user can also independently disable the receiver or the transmitter by setting RXDIS and TXDIS respectively in FLEX_US_CR. If the receiver is disabled during a character reception, the USART waits until the end of reception of the current character, then the reception is stopped. If the transmitter is disabled while it is operating, the USART waits the end of transmission of both the current character and character being stored in the USART Transmit Holding Register (FLEX_US_THR). If a timeguard is programmed, it is handled normally.

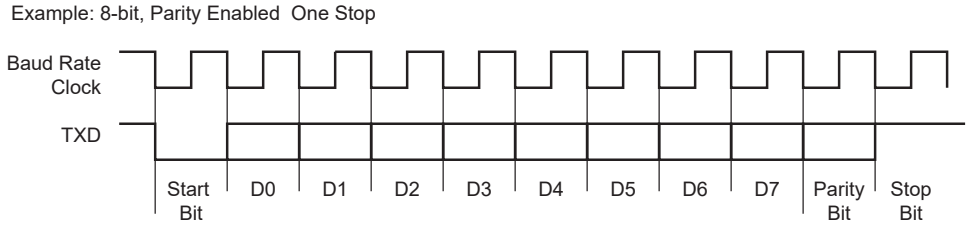
38.7.3 Synchronous and Asynchronous Modes

38.7.3.1 Transmitter Operations

The transmitter performs the same in both Synchronous and Asynchronous operating modes (SYNC = 0 or SYNC = 1). One start bit, up to 9 data bits, 1 optional parity bit and up to 2 stop bits are successively shifted out on the TXD pin at each falling edge of the programmed serial clock.

The number of data bits is selected by the CHRL field and the MODE9 bit in FLEX_US_MR. Nine bits are selected by setting the MODE9 bit regardless of the CHRL field. The parity bit is set according to the PAR field in FLEX_US_MR. The even, odd, space, marked or none parity bit can be configured. The MSBF bit in FLEX_US_MR configures which data bit is sent first. If written to 1, the most significant bit is sent first. If written to 0, the less significant bit is sent first. The number of stop bits is selected by the NBSTOP field in FLEX_US_MR. The 1.5 stop bit is supported in Asynchronous mode only.

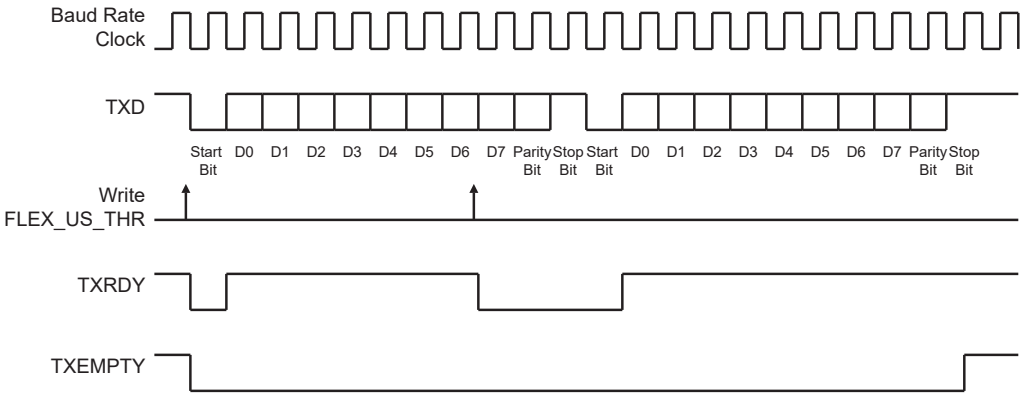
Figure 38-5. Character Transmit



The characters are sent by writing in FLEX_US_THR. The transmitter reports two status bits in the USART Channel Status Register (FLEX_US_CSR): TXRDY (Transmitter Ready), which indicates that FLEX_US_THR is empty and TXEMPTY, which indicates that all the characters written in FLEX_US_THR have been processed. When the current character processing is completed, the last character written in FLEX_US_THR is transferred into the shift register of the transmitter and FLEX_US_THR is emptied, thus TXRDY rises.

Both TXRDY and TXEMPTY bits are low when the transmitter is disabled. Writing a character in FLEX_US_THR while TXRDY is low has no effect and the written character is lost.

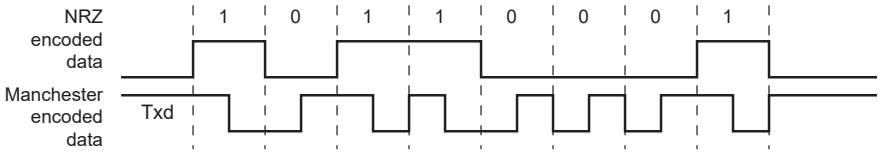
Figure 38-6. Transmitter Status



38.7.3.2 Manchester Encoder

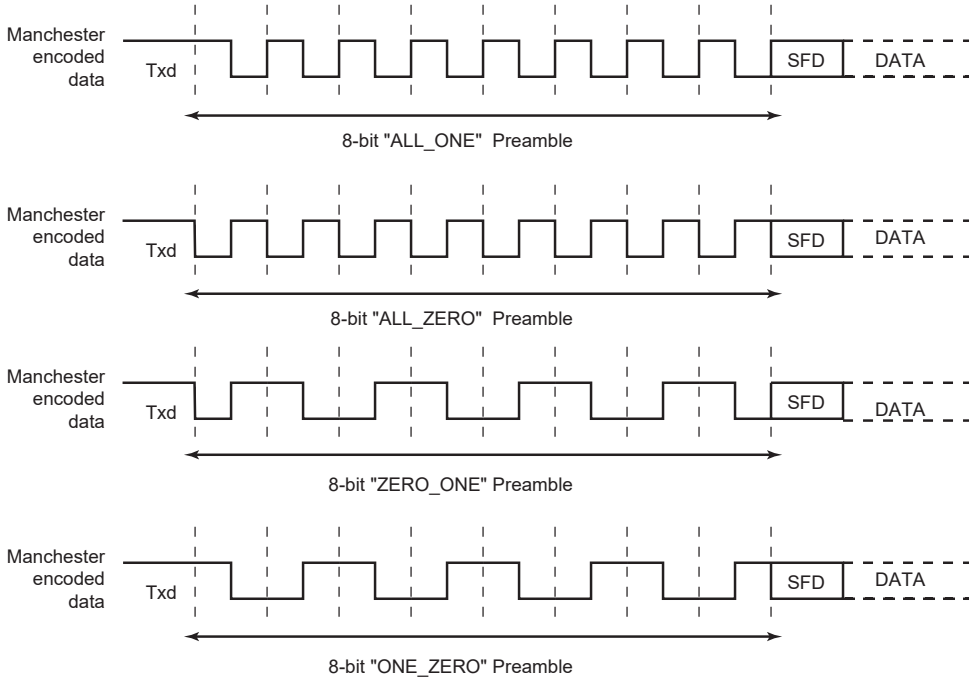
When the Manchester encoder is in use, characters transmitted through the USART are encoded based on biphase Manchester II format. To enable this mode, set the FLEX_US_MR.MAN bit to 1. Depending on polarity configuration, a logic level (zero or one), is transmitted as a coded signal one-to-zero or zero-to-one. Thus, a transition always occurs at the midpoint of each bit time. It consumes more bandwidth than the original NRZ signal (2x) but the receiver has more error control since the expected input must show a change at the center of a bit cell. An example of Manchester encoded sequence is: the byte 0xB1 or 10110001 encodes to 10 01 10 10 01 01 01 10, assuming the default polarity of the encoder. The following figure illustrates this coding scheme.

Figure 38-7. NRZ to Manchester Encoding



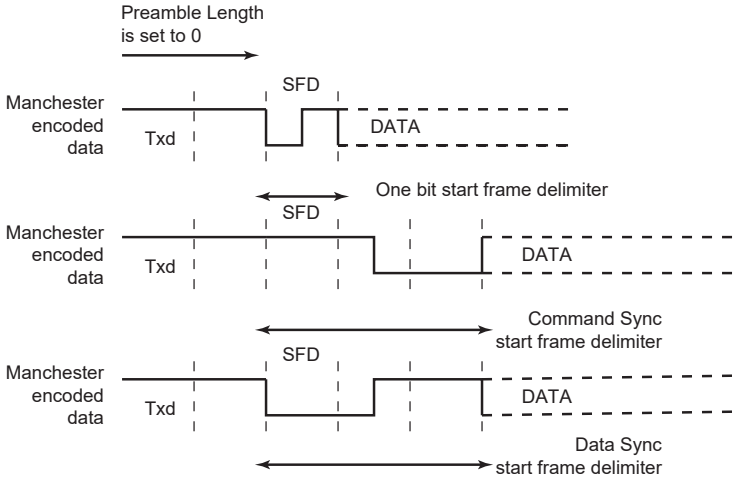
The Manchester encoded character can also be encapsulated by adding both a configurable preamble and a start frame delimiter pattern. Depending on the configuration, the preamble is a training sequence, composed of a predefined pattern with a programmable length from 1 to 15 bit times. If the preamble length is set to 0, the preamble waveform is not generated prior to any character. The preamble pattern is chosen among the following sequences: ALL_ONE, ALL_ZERO, ONE_ZERO or ZERO_ONE, writing the FLEX_US_MAN.TX_PP field. The TX_PL field is used to configure the preamble length. The following figure illustrates and defines the valid patterns. To improve flexibility, the encoding scheme can be configured using the FLEX_US_MAN.TX_MPOL bit. If the TX_MPOL bit is set to zero (default), a logic zero is encoded with a zero-to-one transition and a logic one is encoded with a one-to-zero transition. If the TX_MPOL bit is set to one, a logic one is encoded with a one-to-zero transition and a logic zero is encoded with a zero-to-one transition.

Figure 38-8. Preamble Patterns, Default Polarity Assumed



A start frame delimiter is to be configured using the FLEX_US_MR.ONEBIT bit. It consists of a user-defined pattern that indicates the beginning of a valid data. The following figure illustrates these patterns. If the start frame delimiter, also known as the start bit, is one bit, (ONEBIT = 1), a logic zero is Manchester encoded and indicates that a new character is being sent serially on the line. If the start frame delimiter is a synchronization pattern also referred to as sync (ONEBIT = 0), a sequence of three bit times is sent serially on the line to indicate the start of a new character. The sync waveform is in itself an invalid Manchester waveform as the transition occurs at the middle of the second bit time. Two distinct sync patterns are used: the command sync and the data sync. The command sync has a logic one level for one and a half bit times, then a transition to logic zero for the second one and a half bit times. If the FLEX_US_MR.MODSYNC bit is set to 1, the next character is a command. If it is set to 0, the next character is a data. When direct memory access is used, the MODSYNC bit can be immediately updated with a modified character located in memory. To enable this mode, the FLEX_US_MR.VAR_SYNC bit must be set. In this case, the FLEX_US_MR.MODSYNC bit is bypassed and the sync configuration is held in the FLEX_US_THR.TXSYNH bit. The USART character format is modified and includes sync information.

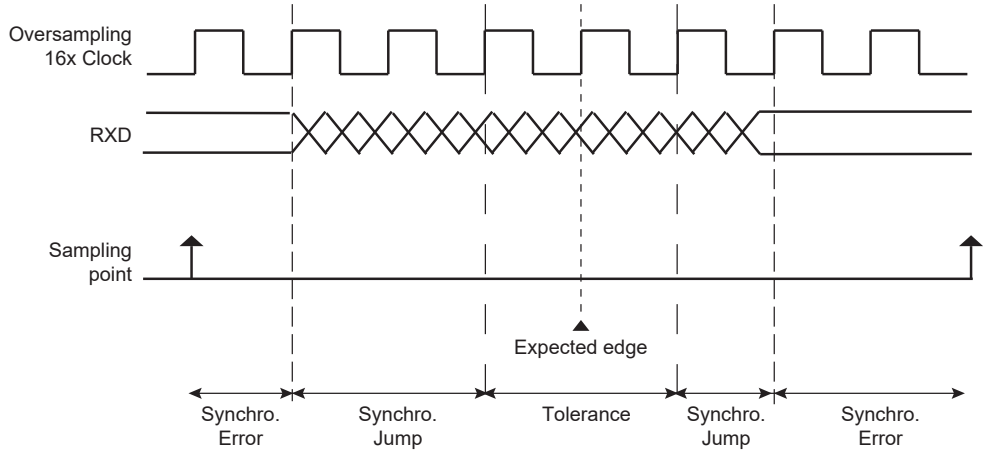
Figure 38-9. Start Frame Delimiter



38.7.3.2.1 Drift Compensation

Drift compensation is available only in 16X Oversampling mode. An hardware recovery system allows a larger clock drift. To enable the hardware system, the bit in the FLEX_US_MAN register must be set. If the RXD edge is one 16X clock cycle from the expected edge, this is considered as normal jitter and no corrective actions is taken. If the RXD event is between 4 and 2 clock cycles before the expected edge, then the current period is shortened by one clock cycle. If the RXD event is between 2 and 3 clock cycles after the expected edge, then the current period is lengthened by one clock cycle. These intervals are considered to be drift and so corrective actions are automatically taken.

Figure 38-10. Bit Resynchronization



38.7.3.3 Asynchronous Receiver

If the USART is programmed in Asynchronous operating mode (SYNC = 0), the receiver oversamples the RXD input line. The oversampling is either 16 or 8 times the baud rate clock, depending on the FLEX_US_MR.OVER bit.

The receiver samples the RXD line. If the line is sampled during one half of a bit time to 0, a start bit is detected and data, parity and stop bits are successively sampled on the bit rate clock.

If the oversampling is 16 (OVER = 0), a start is detected at the eighth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 16 oversampling clock cycles. If the oversampling is 8 (OVER = 1), a start bit is detected at the fourth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 8 oversampling clock cycles.

The number of data bits, first bit sent and Parity mode are selected by the same fields and bits as the transmitter, i.e., respectively CHRL, MODE9, MSBF and PAR. For the synchronization mechanism only, the number of stop bits has no effect on the receiver as it considers only one stop bit, regardless of the NBSTOP field, so that resynchronization between the receiver and the transmitter can occur. Moreover, as soon as the stop bit is sampled, the receiver starts looking for a new start bit so that resynchronization can also be accomplished when the transmitter is operating with one stop bit.

The following figures illustrate start detection and character reception when USART operates in Asynchronous mode.

Figure 38-11. Asynchronous Start Detection

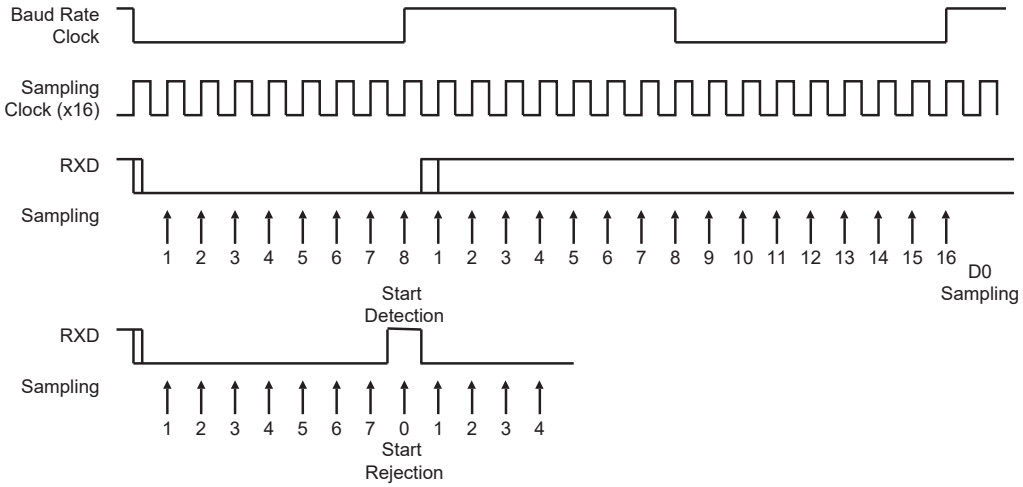
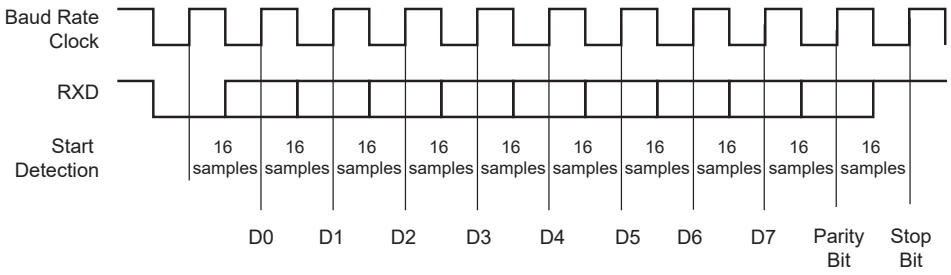


Figure 38-12. Asynchronous Character Reception

Example: 8-bit, Parity Enabled



38.7.3.4 Manchester Decoder

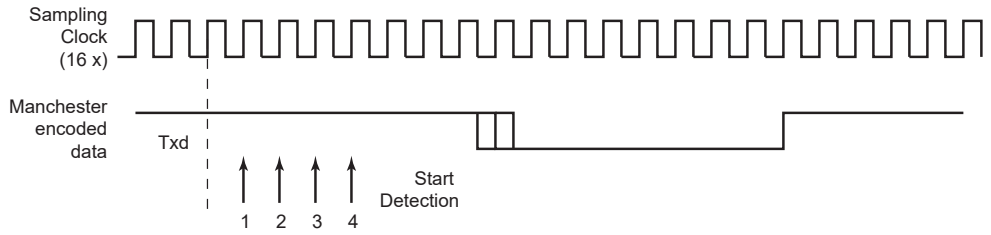
When the FLEX_US_MR.MAN bit is set, the Manchester decoder is enabled. The decoder performs both preamble and start frame delimiter detection. One input line is dedicated to Manchester encoded input data.

An optional preamble sequence can be defined. Its length is user-defined and totally independent of the transmitter side. Use the FLEX_US_MAN.RX_PL field to configure the length of the preamble sequence. If the length is set to 0, no preamble is detected and the function is disabled. In addition, the polarity of the input stream is programmable with the FLEX_US_MAN.RX_MPOL bit. Depending on the desired application, the preamble pattern matching is to be defined via the FLEX_US_MAN.RX_PP field. See figure Preamble Patterns, Default Polarity Assumed for available preamble patterns.

Unlike preamble, the start frame delimiter is shared between Manchester Encoder and Decoder. So, if ONEBIT bit = 1, only a zero encoded Manchester can be detected as a valid start frame delimiter. If ONEBIT = 0, only a sync pattern is detected as a valid start frame delimiter. Decoder operates by detecting transition on incoming stream. If RXD is sampled during one quarter of a bit time to zero, a start bit is detected. See the following figure. The sample pulse rejection mechanism applies.

The FLEX_US_MAN.RXIDLEV bit informs the USART of the receiver line idle state value (receiver line inactive). The user must define RXIDLEV to ensure reliable synchronization. By default, RXIDLEV is set to one (receiver line is at level 1 when there is no activity).

Figure 38-13. Asynchronous Start Bit Detection



The receiver is activated and starts preamble and frame delimiter detection, sampling the data at one quarter and then three quarters. If a valid preamble pattern or start frame delimiter is detected, the receiver continues decoding with the same synchronization. If the stream does not match a valid pattern or a valid start frame delimiter, the receiver resynchronizes on the next valid edge.The minimum time threshold to estimate the bit value is three quarters of a bit time.

If a valid preamble (if used) followed with a valid start frame delimiter is detected, the incoming stream is decoded into NRZ data and passed to USART for processing. The following figure illustrates Manchester pattern mismatch. When incoming data stream is passed to the USART, the receiver is also able to detect Manchester code violation. A code violation is a lack of transition in the middle of a bit cell. In this case, the MANE flag in FLEX_US_CSR is raised. It is cleared by writing a one to FLEX_US_CR.RSTSTA. See figure "Manchester Error Flag" below for an example of Manchester error detection during the data phase.

Figure 38-14. Preamble Pattern Mismatch

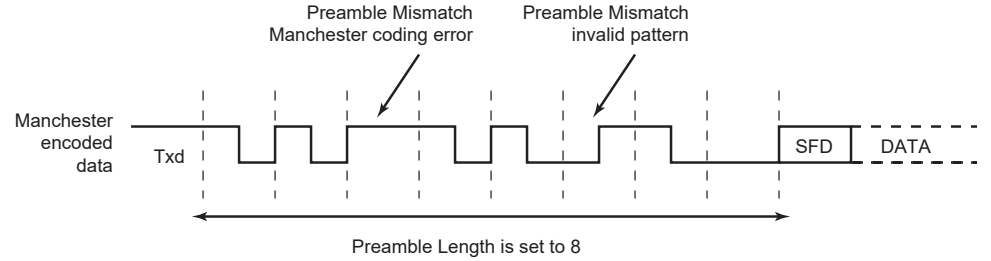
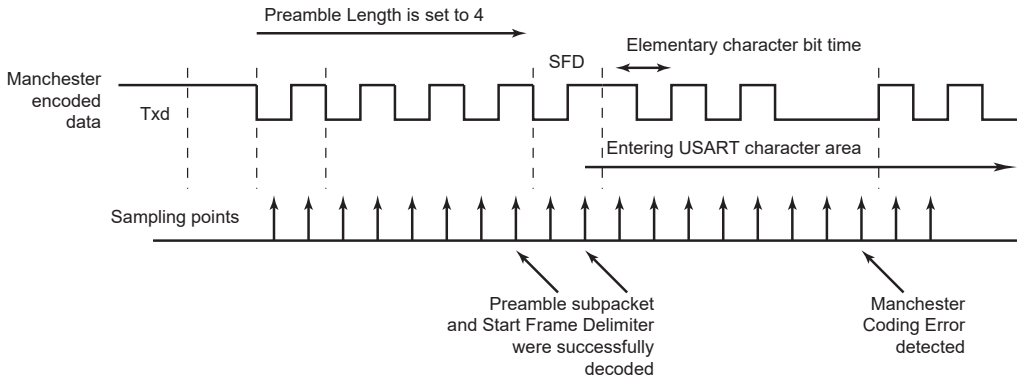


Figure 38-15. Manchester Error Flag



When the start frame delimiter is a sync pattern (ONEBIT = 0), both command and data delimiter are supported. If a valid sync is detected, the received character is written as RXCHR field in the Receive Holding Register (FLEX_US_RHR) and the RXSYNH is updated. RXCHR is set to 1 when the received character is a command, and it is set to 0 if the received character is a data. This mechanism alleviates and simplifies the direct memory access as the character contains its own sync field in the same register.

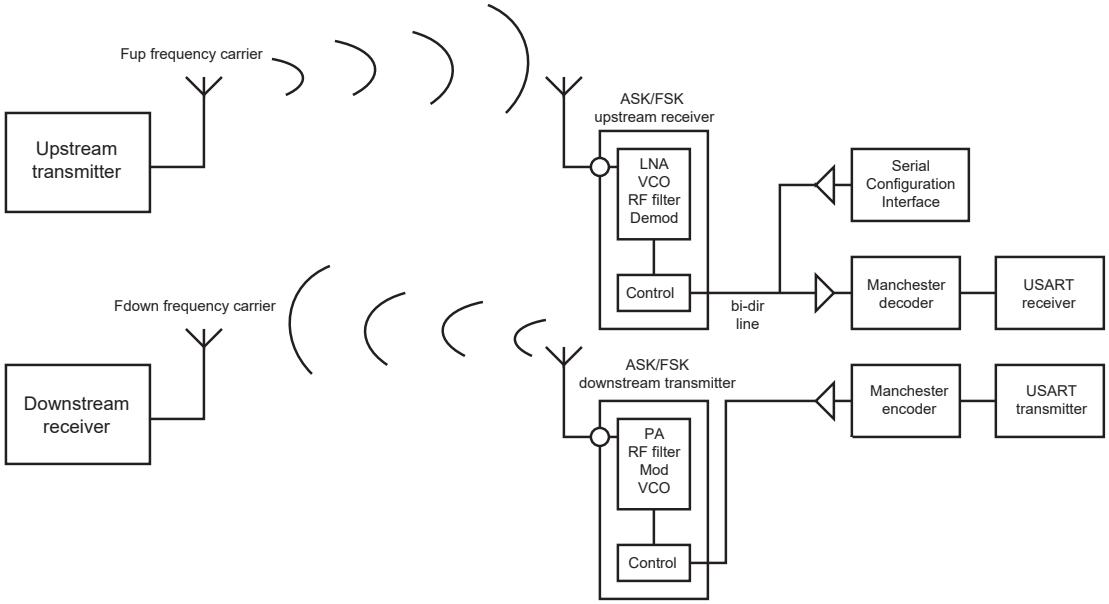
As the decoder is setup to be used in Unipolar mode, the first bit of the frame has to be a zero-to-one transition.

38.7.3.5 Radio Interface: Manchester Encoded USART Application

This section describes low data rate RF transmission systems and their integration with a Manchester encoded USART. These systems are based on transmitter and receiver ICs that support ASK and FSK modulation schemes.

The goal is to perform full-duplex radio transmission of characters using two different frequency carriers. See configuration in the following figure.

Figure 38-16. Manchester Encoded Characters RF Transmission



The USART peripheral is configured as a Manchester encoder/decoder. Looking at the downstream communication channel, Manchester encoded characters are serially sent to the RF transmitter. This may also include a user defined preamble and a start frame delimiter. Mostly, preamble is used in the RF receiver to distinguish between a valid data from a transmitter and signals due to noise. The Manchester stream is then modulated. See the following figure for an example of ASK modulation scheme. When a logic one is sent to the ASK modulator, the power amplifier, referred to as PA, is enabled and transmits an RF signal at downstream frequency. When a logic zero is transmitted, the RF signal is turned off. If the FSK modulator is activated, two different frequencies are used to transmit data. When a logic 1 is sent, the modulator outputs an RF signal at frequency F0 and switches to F1 if the data sent is a 0. See figure "FSK Modulator Output" below.

From the receiver side, another carrier frequency is used. The RF receiver performs a bit check operation examining demodulated data stream. If a valid pattern is detected, the receiver switches to Receiving mode. The demodulated stream is sent to the Manchester decoder. Because of bit checking inside RF IC, the data transferred to the microcontroller is reduced by a user-defined number of bits. The Manchester preamble length is to be defined in accordance with the RF IC configuration.

Figure 38-17. ASK Modulator Output

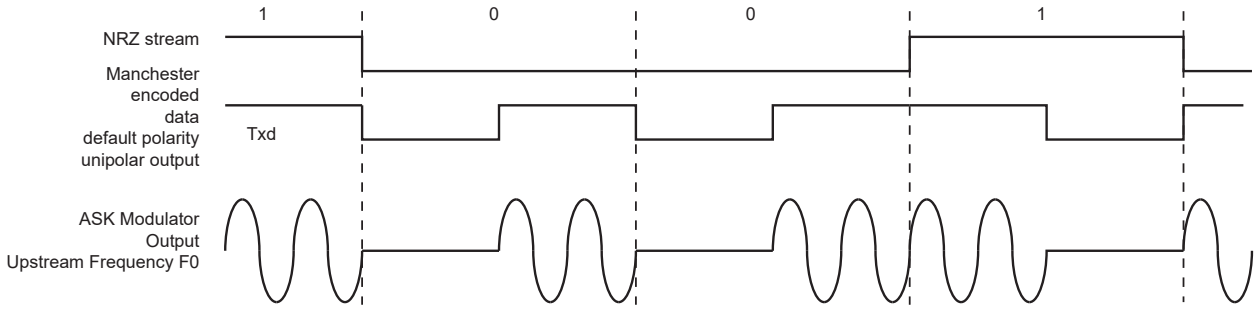
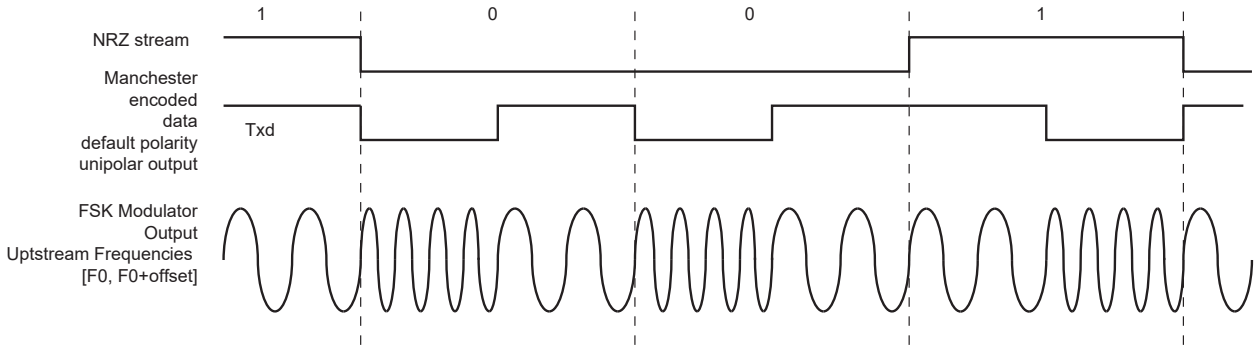


Figure 38-18. FSK Modulator Output



38.7.3.6 Synchronous Receiver

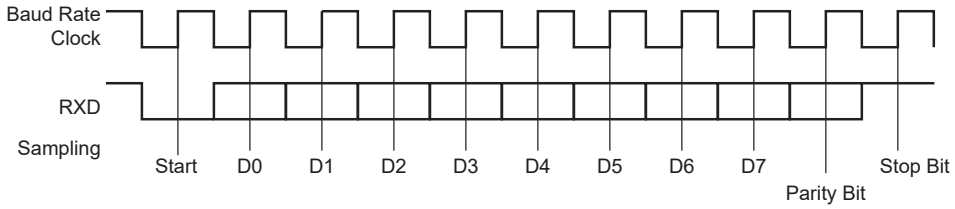
In Synchronous mode (SYNC = 1), the receiver samples the RXD signal on each rising edge of the baud rate clock. If a low level is detected, it is considered as a start. All data bits, the parity bit and the stop bits are sampled and the receiver waits for the next start bit. Synchronous mode operations provide a high-speed transfer capability.

Configuration fields and bits are the same as in Asynchronous mode.

The following figure illustrates a character reception in Synchronous mode.

Figure 38-19. Synchronous Mode Character Reception

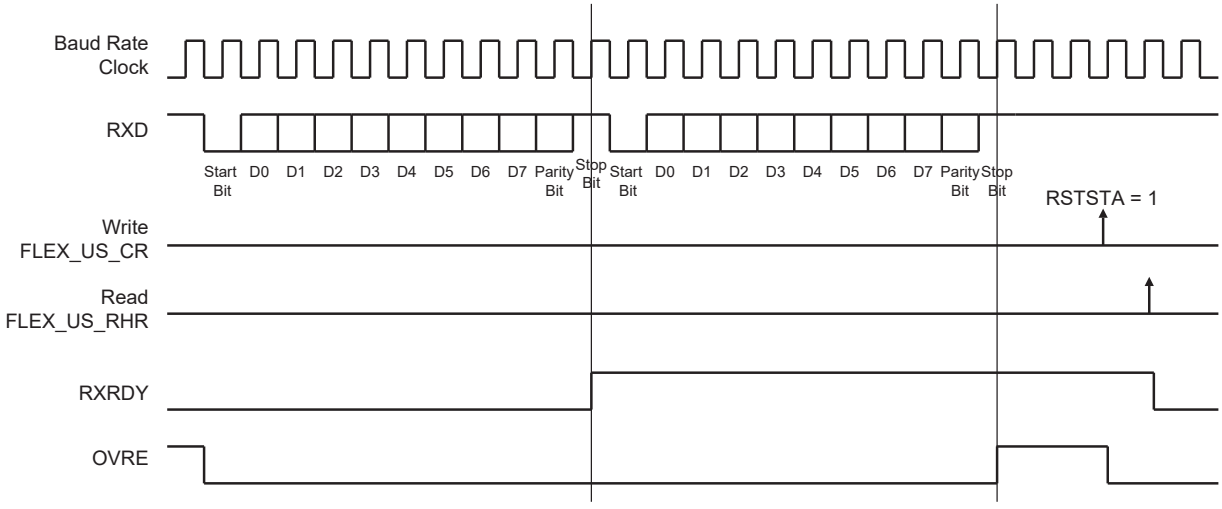
Example: 8-bit, Parity Enabled 1 Stop



38.7.3.7 Receiver Operations

When a character reception is completed, it is transferred to the Receive Holding Register (FLEX_US_RHR) and the FLEX_US_CSR.RXRDY bit is raised. If a character is completed while the RXRDY is set, the Overrun Error (OVRE) bit is set. The last character is transferred into FLEX_US_RHR and overwrites the previous one. The OVRE bit is cleared by writing a one to Reset Status bit FLEX_US_CR.RSTSTA.

Figure 38-20. Receiver Status



38.7.3.8 Parity

The USART supports five parity modes that are selected by writing to the FLEX_US_MR.PAR field. The PAR field also enables the Multidrop mode (see section Multidrop Mode). Even and odd parity bit generation and error detection are supported.

If even parity is selected, the parity generator of the transmitter drives the parity bit to 0 if a number of 1s in the character data bit is even, and to 1 if the number of 1s is odd. Accordingly, the receiver parity checker counts the number of received 1s and reports a parity error if the sampled parity bit does not correspond. If odd parity is selected, the parity generator of the transmitter drives the parity bit to 1 if a number of 1s in the character data bit is even, and to 0 if the number of 1s is odd. Accordingly, the receiver parity checker counts the number of received 1s and reports a parity error if the sampled parity bit does not correspond. If the mark parity is used, the parity generator of the transmitter drives the parity bit to 1 for all characters. The receiver parity checker reports an error if the parity bit is sampled to 0. If the space parity is used, the parity generator of the transmitter drives the parity bit to 0 for all characters. The receiver parity checker reports an error if the parity bit is sampled to 1. If parity is disabled, the transmitter does not generate any parity bit and the receiver does not report any parity error.

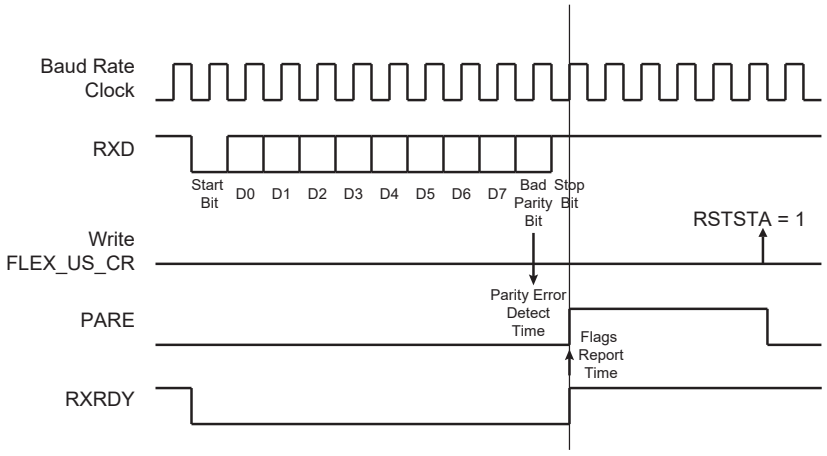
The following table shows an example of the parity bit for the character 0x41 (character ASCII “A”) depending on the configuration of the USART. Because there are two bits set to 1 in the character value, the parity bit is set to 1 when the parity is odd, or configured to 0 when the parity is even.

Table 38-7. Parity Bit Examples

Character	Hexadecimal	Binary	Parity Bit	Parity Mode
A	0x41	0100 0001	1	Odd
A	0x41	0100 0001	0	Even
A	0x41	0100 0001	1	Mark
A	0x41	0100 0001	0	Space
A	0x41	0100 0001	None	None

When the receiver detects a parity error, it sets the Parity Error bit FLEX_US_CSR.PARE. The PARE bit can be cleared by writing a one to the FLEX_US_CR.RSTSTA bit. The following figure illustrates the parity bit status setting and clearing.

Figure 38-21. Parity Error



38.7.3.9 Multidrop Mode

If the value 0x6 or 0x07 is written to the FLEX_US_MR.PAR field, the USART runs in Multidrop mode. This mode differentiates the data characters and the address characters. Data are transmitted with the parity bit to 0 and addresses are transmitted with the parity bit to 1.

If the USART is configured in Multidrop mode, the receiver sets the PARE parity error bit when the parity bit is high and the transmitter is able to send a character with the parity bit high when a one is written to the FLEX_US_CR.SENDA bit.

To handle parity error, the PARE bit is cleared by writing a one to the FLEX_US_CR.RSTSTA bit.

The transmitter sends an address byte (parity bit set) when the FLEX_US_CR.SENDA bit is written to 1. In this case, the next byte written to FLEX_US_THR is transmitted as an address. Any character written in FLEX_US_THR when the SENDA command is not written is transmitted normally with parity to 0.

38.7.3.10 Transmitter Timeguard

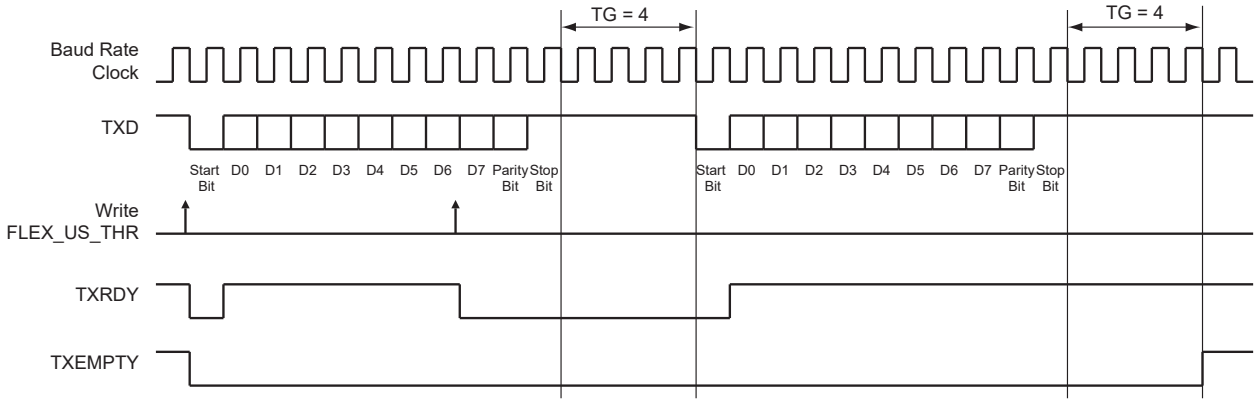
The timeguard feature enables the USART interface with slow remote devices.

The timeguard function enables the transmitter to insert an idle state on the TXD line between two characters. This idle state actually acts as a long stop bit.

The duration of the idle state is programmed in the TG field of the Transmitter Timeguard Register (FLEX_US_TTGR). When this field is written to zero, no timeguard is generated. Otherwise, the transmitter holds a high level on TXD after each transmitted byte during the number of bit periods programmed in TG in addition to the number of stop bits.

As illustrated in the following figure, the behavior of the TXRDY and TXEMPTY status bits is modified by the programming of a timeguard. TXRDY rises only when the start bit of the next character is sent, and thus remains to 0 during the timeguard transmission if a character has been written in FLEX_US_THR. TXEMPTY remains low until the timeguard transmission is completed as the timeguard is part of the current character being transmitted.

Figure 38-22. Timeguard Operations



The following table indicates the maximum length of a timeguard period that the transmitter can handle in relation to the function of the baud rate.

Table 38-8. Maximum Timeguard Length Depending on Baud Rate

Baud Rate (bit/s)	Bit Time (µs)	Timeguard (ms)
1,200	833	212.50
9,600	104	26.56
14,400	69.4	17.71
19,200	52.1	13.28
28,800	34.7	8.85
38,400	26	6.63
56,000	17.9	4.55
57,600	17.4	4.43
115,200	8.7	2.21

38.7.3.11 Receiver Timeout

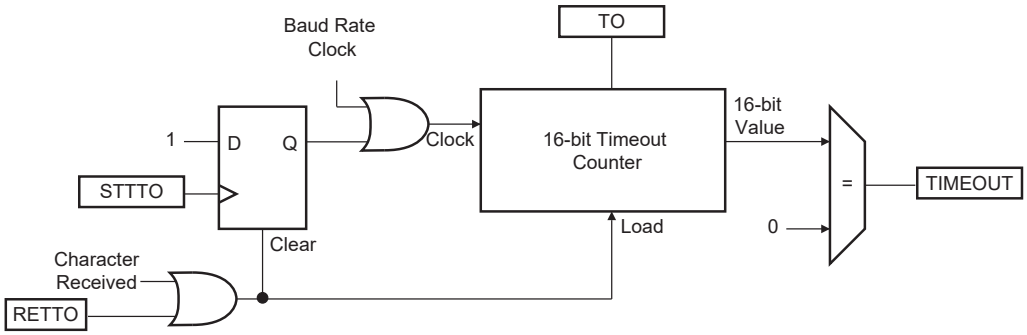
The Receiver Timeout provides support in handling variable-length frames. This feature detects an idle condition on the RXD line. When a timeout is detected, the FLEX_US_CSR.TIMEOUT bit rises and can generate an interrupt, thus indicating to the driver an end of frame.

The timeout delay period (during which the receiver waits for a new character) is programmed in the TO field of the Receiver Timeout Register (FLEX_US_RTOR). If the TO field is written to 0, the Receiver Timeout is disabled and no timeout is detected. The FLEX_US_CSR.TIMEOUT bit remains at 0. Otherwise, the receiver loads a 16-bit counter with the value programmed in TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, the FLEX_US_CSR.TIMEOUT bit rises. Then, the user can either:

- Stop the counter clock until a new character is received. This is performed by writing a ‘1’ to FLEX_US_CR.STTTO. In this case, the idle state on RXD before a new character is received does not provide a timeout. This prevents having to handle an interrupt before a character is received and enables waiting for the next idle state on RXD after a frame is received.
- Obtain an interrupt while no character is received. This is performed by writing a ‘1’ to FLEX_US_CR.RETTO. In this case, the counter starts counting down immediately from the value TO. This generates a periodic interrupt so that a user timeout can be handled, for example when no key is pressed on a keyboard.

The following figure shows the block diagram of the Receiver Timeout feature.

Figure 38-23. Receiver Timeout Block Diagram



The following table gives the maximum timeout period for some standard baud rates.

Table 38-9. Maximum Timeout Period

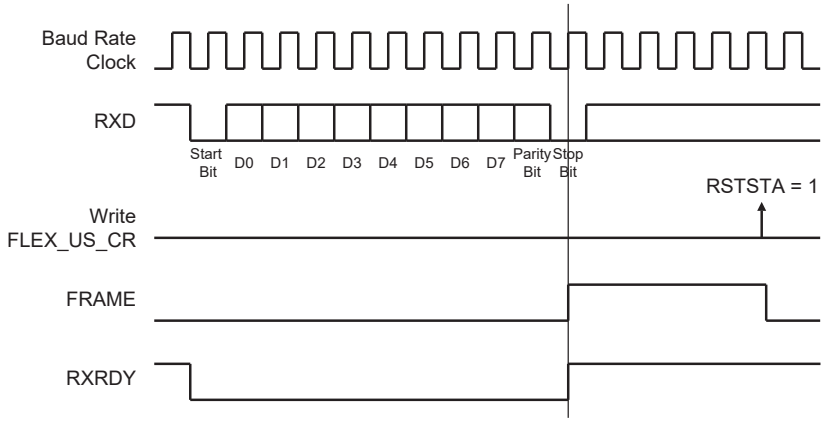
Baud Rate (bit/s)	Bit Time (µs)	Timeout (ms)
600	1,667	109,225
1,200	833	54,613
2,400	417	27,306
4,800	208	13,653
9,600	104	6,827
14,400	69	4,551
19,200	52	3,413
28,800	35	2,276
38,400	26	1,704
56,000	18	1,170
57,600	17	1,138
200,000	5	328

38.7.3.12 Framing Error

The receiver is capable of detecting framing errors. A framing error happens when the stop bit of a received character is detected at level 0. This can occur if the receiver and the transmitter are fully desynchronized.

A framing error is reported on the FLEX_US_CSR.FRAME bit. The FRAME bit is asserted in the middle of the stop bit as soon as the framing error is detected. It is cleared by writing a one to the FLEX_US_CR.RSTSTA bit.

Figure 38-24. Framing Error Status



38.7.3.13 Transmit Break

The user can request the transmitter to generate a break condition on the TXD line. A break condition drives the TXD line low during at least one complete character. It appears the same as a 0x00 character sent with the parity and the stop bits to 0. However, the transmitter holds the TXD line at least during one character until the user requests the break condition to be removed.

A break is transmitted by setting the FLEX_US_CR.STTBRK bit. This can be done at any time, either while the transmitter is empty (no character in either the shift register or in FLEX_US_THR) or when a character is being transmitted. If a break is requested while a character is being shifted out, the character is first completed before the TXD line is held low.

Once the Start Break command is requested, further Start Break commands are ignored until the end of the break is completed.

The break condition is removed by setting the FLEX_US_CR.STPBRK bit. If the Stop Break command is requested before the end of the minimum break duration (one character, including start, data, parity and stop bits), the transmitter ensures that the break condition completes.

The transmitter considers the break as though it is a character, i.e., the Start Break and Stop Break commands are processed only if the FLEX_US_CSR.TXRDY bit = 1 and the start of the break condition clears the TXRDY and TXEMPTY bits as if a character was processed.

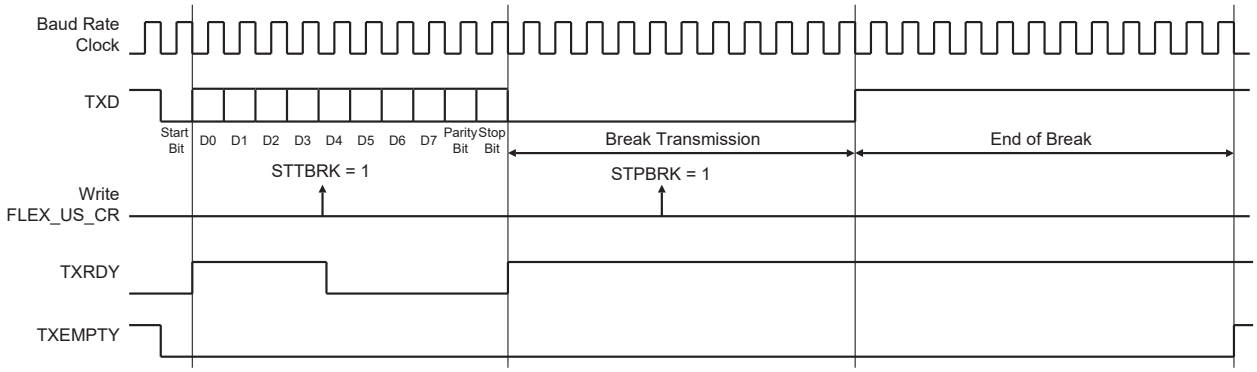
Setting both the FLEX_US_CR.STTBRK and FLEX_US_CR.STPBRK bits can lead to an unpredictable result. All Stop Break commands requested without a previous Start Break command are ignored. A byte written into the Transmit Holding register while a break is pending, but not started, is ignored.

After the break condition, the transmitter returns the TXD line to 1 for a minimum of 12 bit times. Thus, the transmitter ensures that the remote receiver detects correctly the end of break and the start of the next character. If the timeguard is programmed with a value higher than 12, the TXD line is held high for the timeguard period.

After holding the TXD line for this period, the transmitter resumes normal operations.

The following figure illustrates the effect of both the Start Break (STTBRK) and Stop Break (STPBRK) commands on the TXD line.

Figure 38-25. Break Transmission



38.7.3.14 Receive Break

The receiver detects a break condition when all data, parity and stop bits are low. This corresponds to detecting a framing error with data to 0x00, but FRAME remains low.

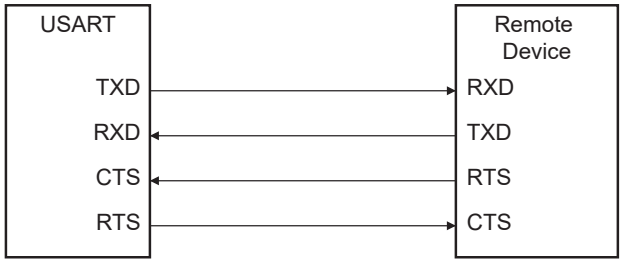
When the low stop bit is detected, the receiver asserts the FLEX_US_CSR.RXBRK bit. FLEX_US_CSR.RXBRK may be cleared by setting the FLEX_US_CR.RSTSTA bit.

An end of receive break is detected by a high level for at least 2/16ths of a bit period in Asynchronous operating mode or one sample at high level in Synchronous operating mode. The end of break detection also asserts the RXBRK bit.

38.7.3.15 Hardware Handshaking

The USART features a hardware handshaking out-of-band flow control. The RTS and CTS pins are used to connect with the remote device, as shown in the following figure.

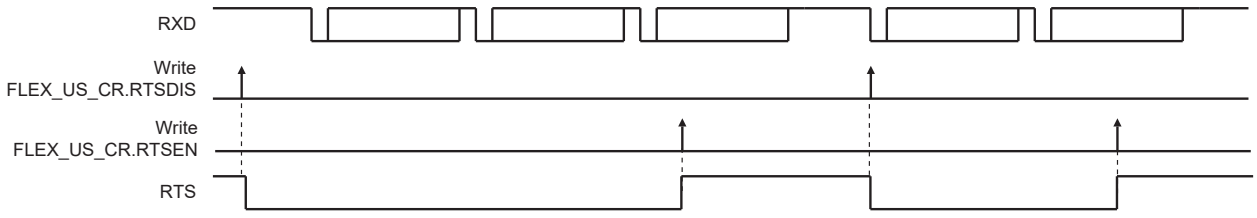
Figure 38-26. Connection with a Remote Device for Hardware Handshaking



Setting the USART to operate with hardware handshaking is performed by writing the FLEX_US_MR.USART_MODE field to the value 0x2.

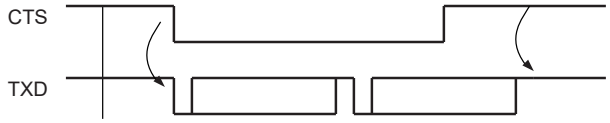
The USART behavior when hardware handshaking is enabled is the same as the behavior in standard Synchronous or Asynchronous mode, except that the receiver drives the RTS pin as described below and the level on the CTS pin modifies the behavior of the transmitter as described below. Using this mode requires using the DMAC channel for reception. The transmitter can handle hardware handshaking in any case.

Figure 38-27. RTS Line Software Control when FLEX_US_MR.USART_MODE = 2



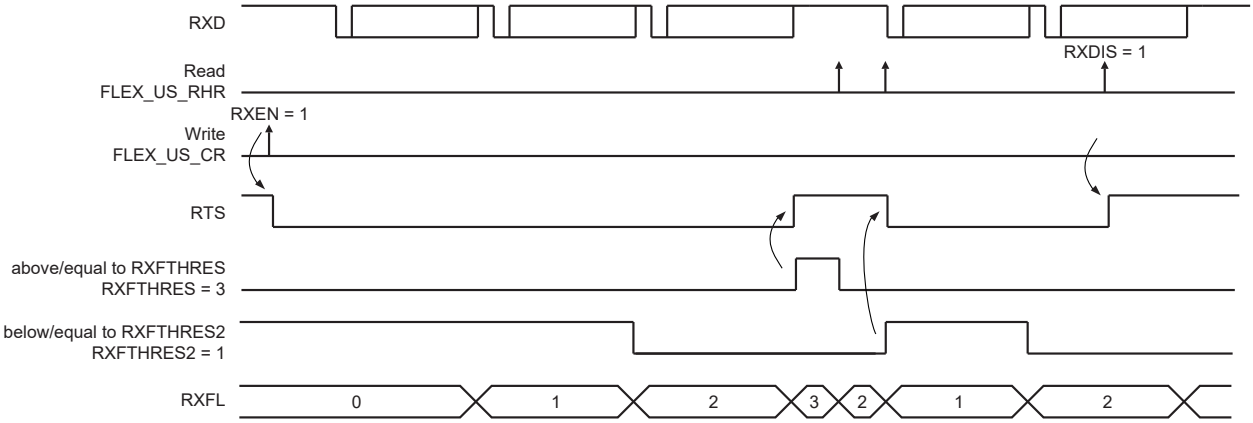
The following figure shows how the transmitter operates if hardware handshaking is enabled. The CTS pin disables the transmitter. If a character is being processed, the transmitter is disabled only after the completion of the current character and transmission of the next character happens as soon as the pin CTS falls.

Figure 38-28. Transmitter Behavior when Operating with Hardware Handshaking



If USART FIFOs are enabled (bit FLEX_US_CR.FIFOEN), the RTS pin can be controlled by the USART Receive FIFO thresholds. The RTS pin control through Receive FIFO thresholds can be activated with the FLEX_US_FMR.FRTSC bit. Once activated, the RTS pin will be controlled by Receive FIFO thresholds, set to level 1 each time RXFTHRES is reached and set to level '0' each time RXFTHRES2 is reached (and RXFTHRES is not reached).

Figure 38-29. Receiver Behavior When FIFO Enabled and FRTSC Set to '1'



Note: In this mode, RXFTHRES must be > RXFTHRES2.

38.7.4 ISO7816 Mode

The USART features an ISO7816-compatible operating mode. This mode permits interfacing with smart cards and Security Access Modules (SAM) communicating through an ISO7816 link. Both T = 0 and T = 1 protocols defined by the ISO7816 specification are supported.

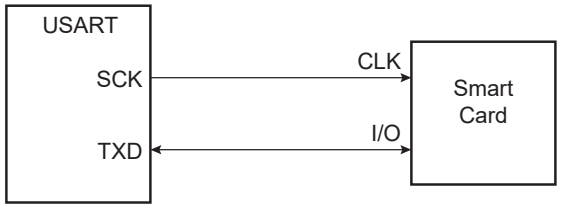
Setting the USART in ISO7816 mode is performed by writing the FLEX_US_MR.USART_MODE field to the value 0x4 for protocol T = 0 and to the value 0x6 for protocol T = 1.

38.7.4.1 ISO7816 Mode Overview

The ISO7816 is a half-duplex communication on only one bidirectional line. The baud rate is determined by a division of the clock provided to the remote device (see figure in section Baud Rate Generator).

The USART connects to a smart card as shown in the following figure. The TXD line becomes bidirectional and the baud rate generator feeds the ISO7816 clock on the SCK pin. As the TXD pin becomes bidirectional, its output remains driven by the output of the transmitter but only when the transmitter is active while its input is directed to the input of the receiver. The USART is considered as the master of the communication as it generates the clock.

Figure 38-30. Connection of a Smart Card to the USART



When operating in ISO7816, either in T = 0 or T = 1 modes, the character format is fixed. The configuration is 8 data bits, even parity and 1 or 2 stop bits, regardless of the values programmed in the CHRL, MODE9, PAR and CHMODE fields. MSBF can be used to transmit LSB or MSB first. Parity Bit (PAR) can be used to transmit in Normal or Inverse mode. See section 38.10.5 FLEX_US_MR and description of field PAR (Parity Type).

The USART cannot operate concurrently in both Receiver and Transmitter modes as the communication is unidirectional at a time. It has to be configured according to the required mode by enabling or disabling either the receiver or the transmitter as desired. Enabling both the receiver and the transmitter at the same time in ISO7816 mode may lead to unpredictable results.

The ISO7816 specification defines an inverse transmission format. Data bits of the character must be transmitted on the I/O line at their negative value.

38.7.4.2 Protocol T = 0

In T = 0 protocol, a character is made up of 1 start bit, 8 data bits, 1 parity bit and 1 guard time, which lasts two bit times. The transmitter shifts out the bits and does not drive the I/O line during the guard time.

If no parity error is detected, the I/O line remains at 1 during the guard time and the transmitter can continue with the transmission of the next character, as shown in the following figure.

If a parity error is detected by the receiver, it drives the I/O line to 0 during the guard time, as shown in figure "T = 0 Protocol with Parity Error" below. This error bit is also named NACK, for Non Acknowledge. In this case, the character lasts 1 bit time more, as the guard time length is the same and is added to the error bit time which lasts 1 bit time.

When the USART is the receiver and it detects an error, it does not load the erroneous character in the Receive Holding Register (FLEX_US_RHR). It appropriately sets the PARE bit in the Status Register (FLEX_US_CSR) so that the software can handle the error.

Figure 38-31. T = 0 Protocol without Parity Error

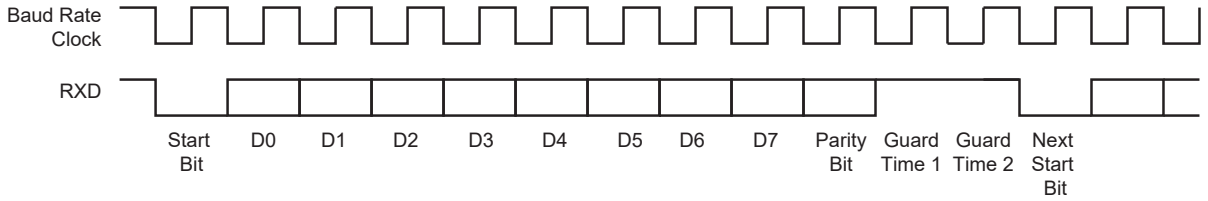
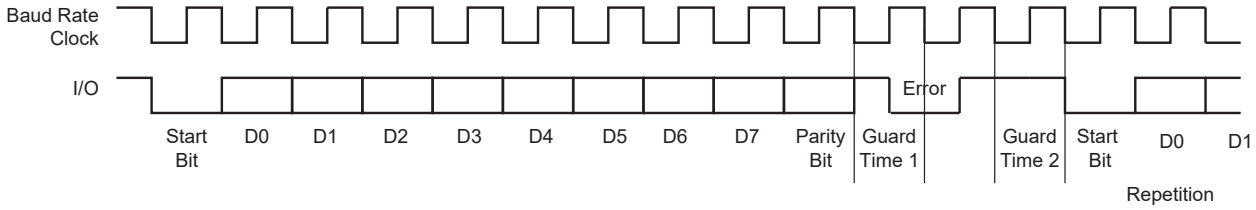


Figure 38-32. T = 0 Protocol with Parity Error



38.7.4.2.1 Receive Error Counter

The USART receiver also records the total number of errors. This can be read in the Number of Error (FLEX_US_NER) register. The NB_ERRORS field can record up to 255 errors. Reading FLEX_US_NER automatically clears the NB_ERRORS field.

38.7.4.2.2 Receive NACK Inhibit

The USART can be configured to inhibit an error. This is done by writing a ‘1’ to FLEX_US_MR.INACK. In this case, no error signal is driven on the I/O line even if a parity bit is detected.

Moreover, if INACK = 1, the erroneous received character is stored in the Receive Holding register as if no error occurred, and the RXRDY bit rises.

38.7.4.2.3 Transmit Character Repetition

When the USART is transmitting a character and gets a NACK, it can automatically repeat the character before moving on to the next one. Repetition is enabled by writing the FLEX_US_MR.MAX_ITERATION field at a value higher than 0. Each character can be transmitted up to eight times: the first transmission plus seven repetitions.

If MAX_ITERATION does not equal zero, the USART repeats the character as many times as the value loaded in MAX_ITERATION.

When the USART repetition number reaches MAX_ITERATION, and the last repeated character is not acknowledged, the FLEX_US_CSR.ITER bit is set. If the repetition of the character is acknowledged by the receiver, the repetitions are stopped and the iteration counter is cleared.

The FLEX_US_CSR.ITER bit can be cleared by writing the FLEX_US_CR.RSTIT bit to 1.

38.7.4.2.4 Disable Successive Receive NACK

The receiver can limit the number of successive NACKs sent back to the remote transmitter. This is programmed by setting the FLEX_US_MR.DSNACK bit. The maximum number of NACKs transmitted is programmed in the MAX_ITERATION field. As soon as MAX_ITERATION is reached, no error signal is driven on the I/O line and the FLEX_US_CSR.ITER bit is set.

38.7.4.3 Protocol T = 1

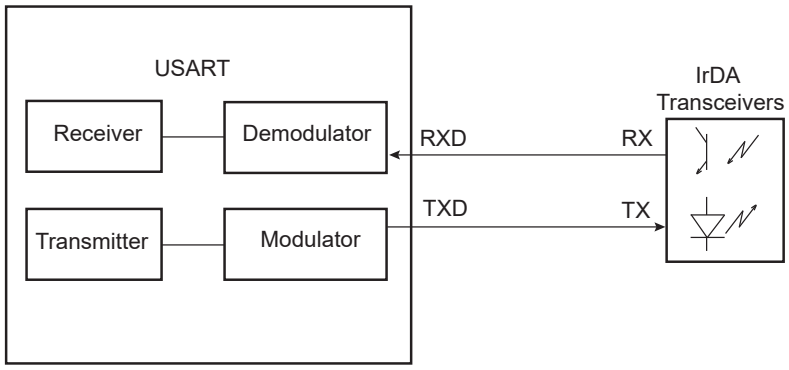
When operating in ISO7816 protocol T = 1, the transmission is similar to an asynchronous format with only one stop bit. The parity is generated when transmitting and checked when receiving. Parity error detection sets the FLEX_US_CSR.PARE bit.

38.7.5 IrDA Mode

The USART features an IrDA mode supplying half-duplex point-to-point wireless communication. It embeds the modulator and demodulator which allows a glueless connection to the infrared transceivers, as shown in the following figure. The modulator and demodulator are compliant with the IrDA specification version 1.1 and support data transfer speeds ranging from 2.4 kbit/s to 115.2 kbit/s.

The USART IrDA mode is enabled by setting the FLEX_US_MR.USART_MODE field to the value 0x8. The IrDA Filter Register (FLEX_US_IF) allows configuring the demodulator filter. The USART transmitter and receiver operate in a normal Asynchronous mode and all parameters are accessible. Note that the modulator and the demodulator are activated.

Figure 38-33. Connection to IrDA Transceivers



The receiver and the transmitter must be enabled or disabled according to the direction of the transmission to be managed.

To receive IrDA signals, the following needs to be done:

- Disable TX and Enable RX
- Configure the TXD pin as PIO and set it as an output to 0 (to avoid LED transmission). Disable the internal pullup (better for power consumption).
- Receive data

38.7.5.1 IrDA Modulation

For baud rates up to and including 115.2 kbit/s, the RZI modulation scheme is used. “0” is represented by a light pulse of 3/16th of a bit time. Some examples of signal pulse duration are shown in the following table.

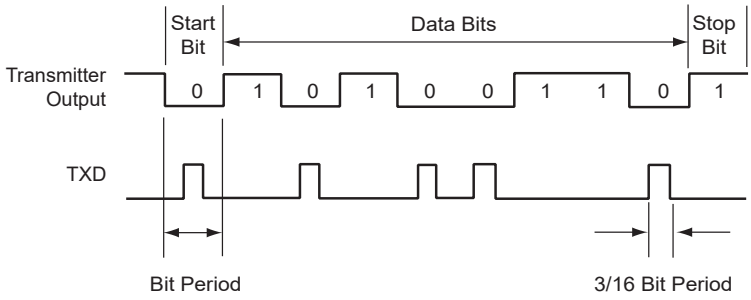
Table 38-10. IrDA Pulse Duration

Baud Rate	Pulse Duration (3/16)
2.4 kbit/s	78.13 µs

.....continued	
Baud Rate	**Pulse Duration (3/16)**
9.6 kbit/s	19.53 µs
19.2 kbit/s	9.77 µs
38.4 kbit/s	4.88 µs
57.6 kbit/s	3.26 µs
115.2 kbit/s	1.63 µs

The following figure shows an example of character transmission.

Figure 38-34. IrDA Modulation



38.7.5.2 IrDA Baud Rate

The following table gives some examples of CD values, baud rate error and pulse duration. Note that the requirement on the maximum acceptable error of ±1.87% must be met.

Table 38-11. IrDA Baud Rate Error

Peripheral Clock	Baud Rate (bit/s)	CD	Baud Rate Error	Pulse Time (µs)
3,686,400	115,200	2	0.00%	1.63
20,000,000	115,200	11	1.38%	1.63
32,768,000	115,200	18	1.25%	1.63
40,000,000	115,200	22	1.38%	1.63
3,686,400	57,600	4	0.00%	3.26
20,000,000	57,600	22	1.38%	3.26
32,768,000	57,600	36	1.25%	3.26
40,000,000	57,600	43	0.93%	3.26
3,686,400	38,400	6	0.00%	4.88
20,000,000	38,400	33	1.38%	4.88
32,768,000	38,400	53	0.63%	4.88
40,000,000	38,400	65	0.16%	4.88
3,686,400	19,200	12	0.00%	9.77
20,000,000	19,200	65	0.16%	9.77
32,768,000	19,200	107	0.31%	9.77
40,000,000	19,200	130	0.16%	9.77

.....continued

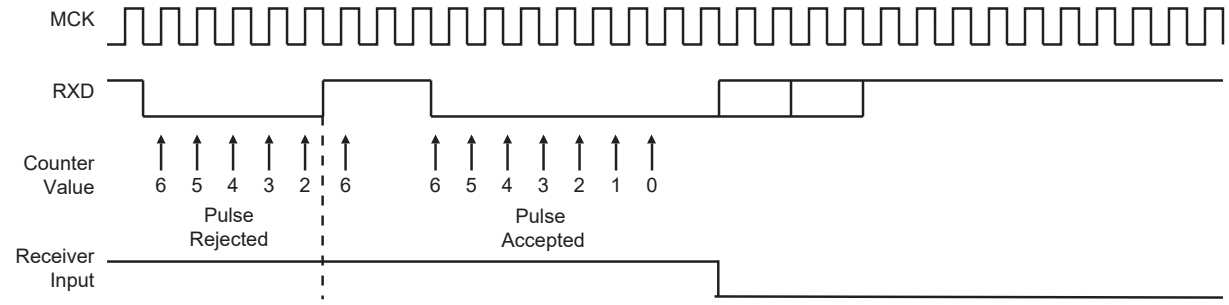
Peripheral Clock	Baud Rate (bit/s)	CD	Baud Rate Error	Pulse Time (µs)
3,686,400	9,600	24	0.00%	19.53
20,000,000	9,600	130	0.16%	19.53
32,768,000	9,600	213	0.16%	19.53
40,000,000	9,600	260	0.16%	19.53
3,686,400	2,400	96	0.00%	78.13
20,000,000	2,400	521	0.03%	78.13
32,768,000	2,400	853	0.04%	78.13

38.7.5.3 IrDA Demodulator

The demodulator is based on the IrDA Receive filter comprised of an 8-bit down counter which is loaded with the value programmed in FLEX_US_IF. When a falling edge is detected on the RXD pin, the Filter Counter starts counting down at the peripheral clock speed. If a rising edge is detected on the RXD pin, the counter stops and is reloaded with FLEX_US_IF. If no rising edge is detected when the counter reaches 0, the input of the receiver is driven low during one bit time.

The following figure illustrates the operations of the IrDA demodulator.

Figure 38-35. IrDA Demodulator Operations



The programmed value in the FLEX_US_IF register must always meet the following criteria:

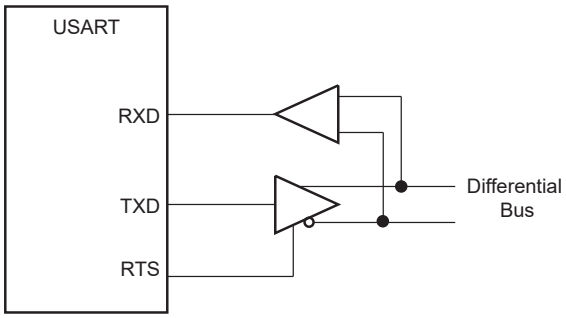
$t_{peripheral\ clock} \times (IRDA\_FILTER + 3) < 1.41\ \mu s$

As the IrDA mode uses the same logic as the ISO7816, note that the FLEX_US_FIDI.FI_DI_RATIO field must be set to a value higher than 0 to make sure IrDA communications operate correctly.

38.7.6 RS485 Mode

The USART features the RS485 mode to enable line driver control. While operating in RS485 mode, the USART behaves as though in Asynchronous or Synchronous mode and configuration of all the parameters is possible. The difference is that the RTS pin is driven high when the transmitter is operating. The behavior of the RTS pin is controlled by the TXEMPTY bit. A typical connection of the USART to an RS485 bus is shown in the following figure.

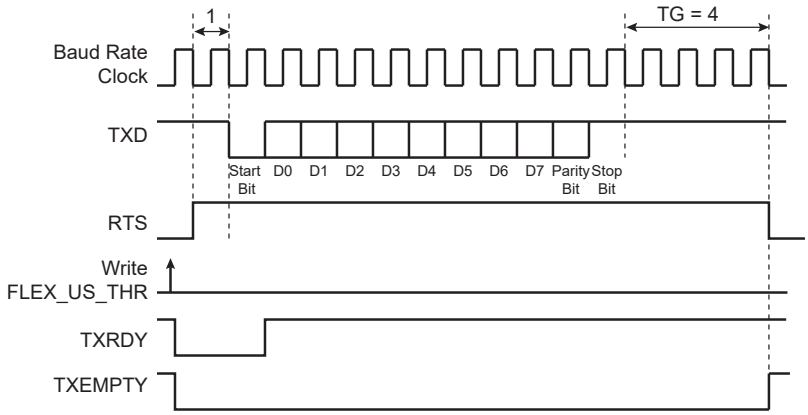
Figure 38-36. Typical Connection to an RS485 Bus



The USART is set in RS485 mode by writing the value 0x1 to the FLEX_US_MR.USART_MODE field.

The RTS pin is at a level inverse to the TXEMPTY bit. Significantly, the RTS pin remains high when a timeguard is programmed, so that the line can remain driven after the last character completion. The following figure gives an example of the RTS waveform during a character transmission when the timeguard is enabled.

Figure 38-37. Example of RTS Drive with Timeguard



38.7.7 USART Comparison Function on Received Character

The CMP flag in FLEX_US_CSR is set when the received character matches the conditions programmed in FLEX_US_CMPR. The CMP flag is set as soon as FLEX_US_RHR is loaded with the new received character. The CMP flag is cleared by writing a one to FLEX_US_CR.RSTSTA.

FLEX_US_CMPR can be programmed to provide different comparison methods:

- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if any received character equals VAL1 or VAL2.

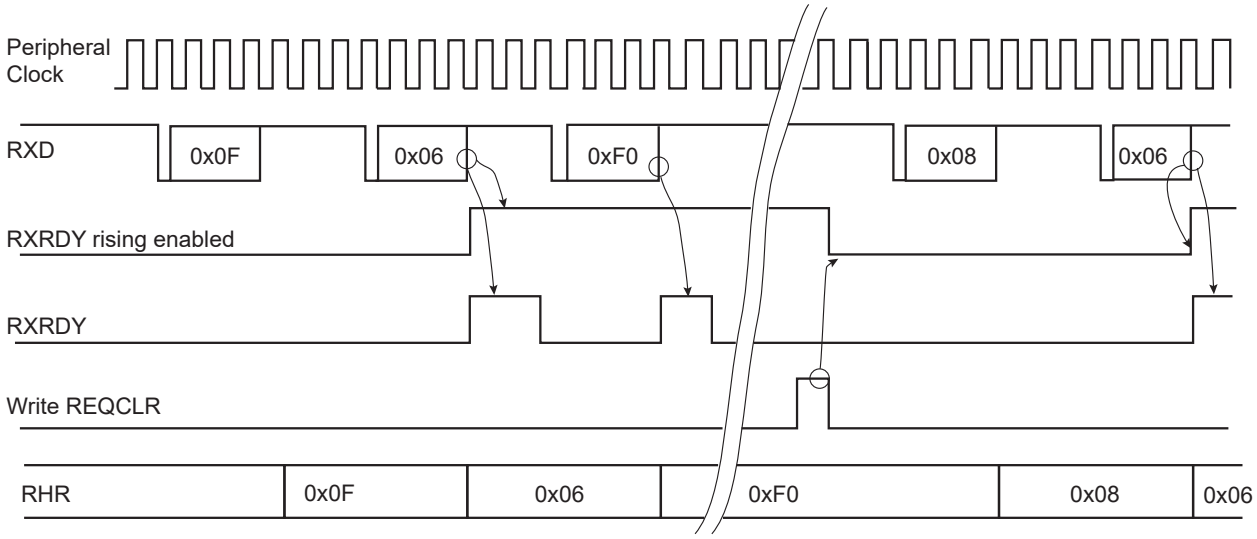
When the FLEX_US_CMPR.CMPMODE bit is set to FLAG_ONLY (value 0), all received data are loaded in FLEX_US_RHR and the CMP flag provides the status of the comparison result.

By programming the START_CONDITION.CMPMODE bit (value 1), the comparison function result triggers the start of the loading of FLEX_US_RHR (see the following figure). The trigger condition exists as soon as the received character value matches the condition defined by the programming of VAL1, VAL2 and CMPPAR in FLEX_US_CMPR. The comparison trigger event is restarted by writing a 1 to the FLEX_US_CR.REQCLR bit.

The value programmed in the VAL1 and VAL2 fields must not exceed the maximum value of the received character (see CHRL field in register 38.10.5 FLEX_US_MR).

Figure 38-38. Receive Holding Register Management

CMPMODE = 1, VAL1 = VAL2 = 0x06



38.7.8 LIN Mode

The LIN mode provides master node and slave node connectivity on a LIN bus.

The LIN (Local Interconnect Network) is a serial communication protocol which efficiently supports the control of mechatronic nodes in distributed automotive applications.

The main properties of the LIN bus are:

- Single master/multiple slaves concept
- Low-cost silicon implementation based on common UART/SCI interface hardware, an equivalent in software, or as a pure state machine.
- Self synchronization without quartz or ceramic resonator in the slave nodes
- Deterministic signal transmission
- Low cost single-wire implementation
- Speed up to 20 kbit/s

LIN provides cost efficient bus communication where the bandwidth and versatility of CAN are not required.

The LIN mode enables processing LIN frames with a minimum of action from the microprocessor.

38.7.8.1 Modes of Operation

The USART can act either as a LIN master node or as a LIN slave node.

The node configuration is chosen by setting the USART_MODE field in the USART Mode Register (FLEX_US_MR):

- LIN master node (USART_MODE = 0xA)
- LIN slave node (USART_MODE = 0xB)

In order to avoid unpredictable behavior, any change of the LIN node configuration must be followed by a software reset of the transmitter and of the receiver (except the initial node configuration after a hardware reset). (See section 38.7.2 Receiver and Transmitter Control.)

38.7.8.2 Baud Rate Configuration

See section Baud Rate in Asynchronous Mode.

- LIN master node: The baud rate is configured in FLEX_US_BRGR.
- LIN slave node: The initial baud rate is configured in FLEX_US_BRGR. This configuration is automatically copied in the LIN Baud Rate Register (FLEX_US_LINBRR) when writing FLEX_US_BRGR. After the synchronization procedure, the baud rate is updated in FLEX_US_LINBRR.

38.7.8.3 Receiver and Transmitter Control

See section 38.7.2 Receiver and Transmitter Control.

38.7.8.4 Character Transmission

See section Transmitter Operations.

38.7.8.5 Character Reception

See section Receiver Operations.

38.7.8.6 Header Transmission (Master Node Configuration)

All the LIN Frames start with a header which is sent by the master node and consists of a Synch Break Field, Synch Field and Identifier Field.

So in master node configuration, the frame handling starts with the sending of the header.

The header is transmitted as soon as the identifier is written in the LIN Identifier Register (FLEX_US_LINIR). At this moment the flag TXRDY falls.

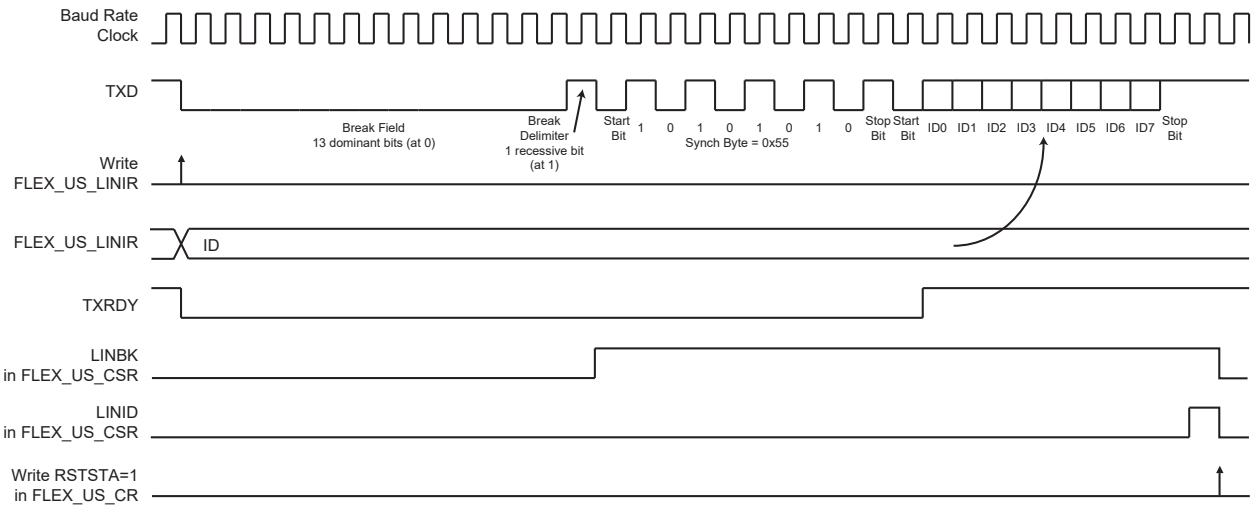
The Break Field, the Synch Field and the Identifier Field are sent automatically one after the other.

The Break Field consists of 13 dominant bits and 1 recessive bit, the Synch Field is the character 0x55 and the Identifier corresponds to the character written in the LIN Identifier Register (FLEX_US_LINIR). The Identifier parity bits can be automatically computed and sent (see section Identifier Parity).

The flag TXRDY rises when the identifier character is transferred into the shift register of the transmitter.

As soon as the Synch Break Field is transmitted, the FLEX_US_CSR.LINBK flag bit is set. Likewise, as soon as the Identifier Field is sent, the FLEX_US_CSR.LINID flag bit is set. These flags are reset by writing a one to the FLEX_US_CR.RSTSTA bit.

Figure 38-39. Header Transmission



38.7.8.7 Header Reception (Slave Node Configuration)

All the LIN frames start with a header which is sent by the master node and consists of a Synch Break Field, Synch Field and Identifier Field.

In slave node configuration, the frame handling starts with the reception of the header.

The USART uses a break detection threshold of 11 nominal bit times at the actual baud rate. At any time, if 11 consecutive recessive bits are detected on the bus, the USART detects a Break Field. As long as a Break Field has not been detected, the USART stays idle and the received data are not taken in account.

When a Break Field has been detected, the FLEX_US_CSR.LINBK flag is set and the USART expects the Synch Field character to be 0x55. This field is used to update the actual baud rate in order to remain synchronized (see section Slave Node Synchronization). If the received Synch character is not 0x55, an Inconsistent Synch Field error is generated (see section LIN Errors).

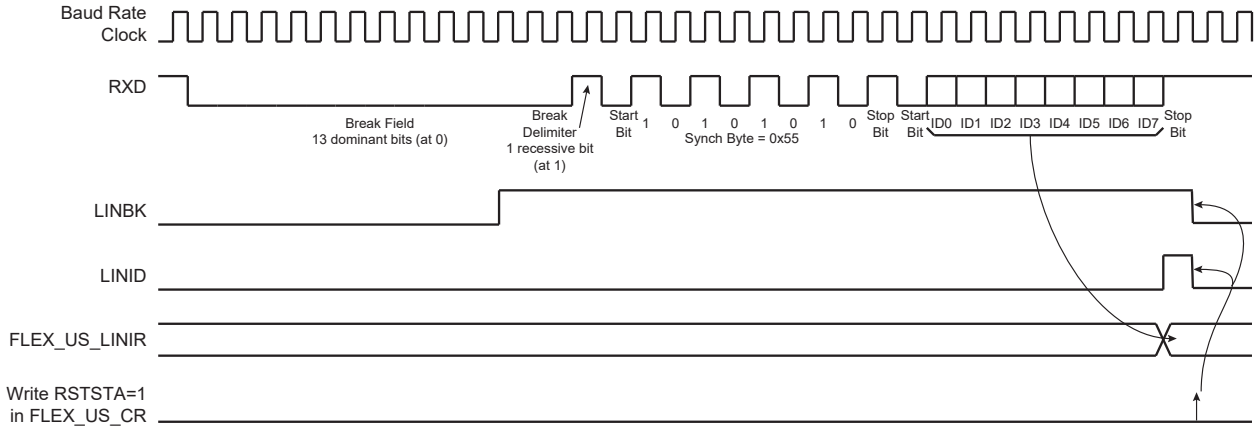
After receiving the Synch Field, the USART expects to receive the Identifier Field.

When the Identifier Field has been received, the FLEX_US_CSR.LINID flag bit is set. At this moment, the IDCHR field in the LIN Identifier Register (FLEX_US_LINIR) is updated with the received character. The Identifier parity bits can be automatically computed and checked (see section Identifier Parity).

If the header is not entirely received within the time given by the maximum length of the header $t_{Header\_Maximum}$, the FLEX_US_CSR.LINHTE error flag bit is set.

The flag bits LINID, LINBK and LINHTE are reset by writing a one to the FLEX_US_CR.RSTSTA bit.

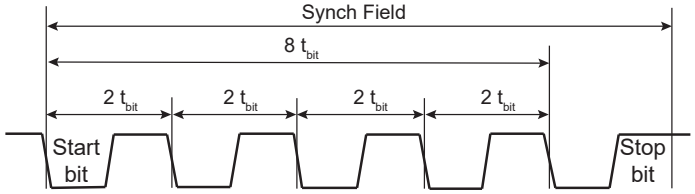
Figure 38-40. Header Reception



38.7.8.8 Slave Node Synchronization

The synchronization is done only in slave node configuration. The procedure is based on time measurement between the falling edges of the Synch Field. The falling edges are available in distances of 2, 4, 6 and 8 bit times.

Figure 38-41. Synch Field



The time measurement is made by a 19-bit counter driven by the sampling clock (see section Baud Rate Generator).

When the start bit of the Synch Field is detected, the counter is reset. Then during the next eight t_{bit} of the Synch Field, the counter is incremented. At the end of these eight t_{bit}, the counter is stopped. At this moment, the 16 most significant bits of the counter (value divided by 8) give the new clock divider (LINCD) and the 3 least significant bits of this value (the remainder) give the new fractional part (LINFP).

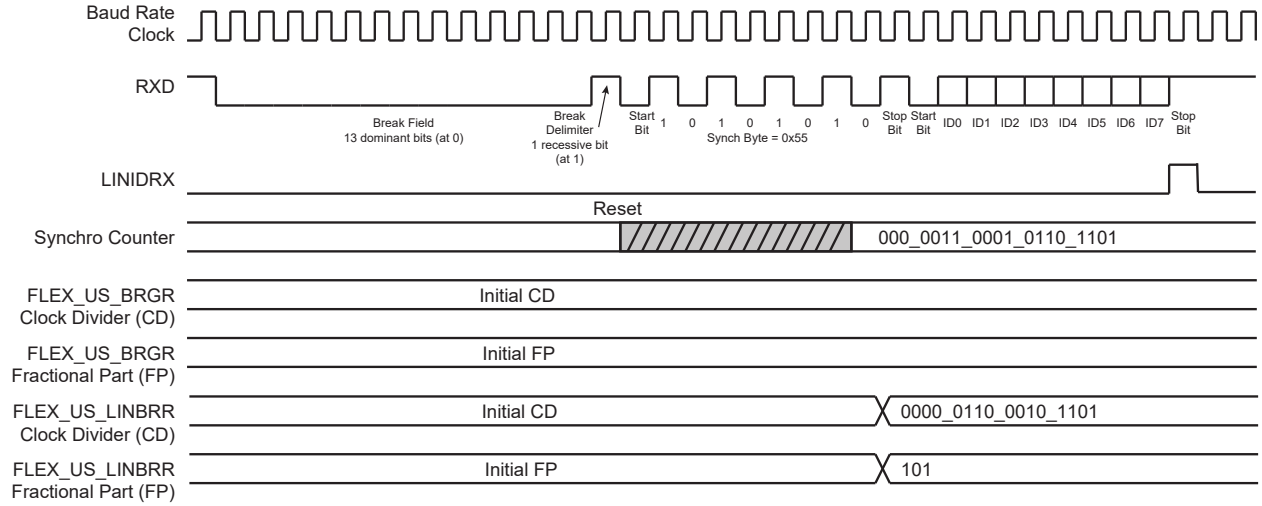
Once the Synch Field has been entirely received, the clock divider (LINCD) and the fractional part (LINFP) are updated in the LIN Baud Rate Register (FLEX_US_LINBRR) with the computed values, if the Synchronization is not disabled by the SYNCDIS bit in the LIN Mode Register (FLEX_US_LINMR).

After reception of the Synch Field:

- If it appears that the computed baud rate deviation compared to the initial baud rate is superior to the maximum tolerance FTol_Unsynch (±15%), then the clock divider (LINCD) and the fractional part (LINFP) are not updated, and the FLEX_US_CSR.LINSTE error flag bit is set.
- If it appears that the sampled Synch character is not equal to 0x55, then the clock divider (LINCD) and the fractional part (LINFP) are not updated, and the FLEX_US_CSR.LINISFE error flag bit is set.

Flags LINSTE and LINISFE are reset by writing a one to the FLEX_US_CR.RSTSTA bit.

Figure 38-42. Slave Node Synchronization



The synchronization accuracy depends on several parameters:

- The nominal clock frequency (f_{Nom}) (the theoretical slave node clock frequency)
- The baud rate
- The oversampling (OVER = 0 => 16X or OVER = 1 => 8X)

The following formula is used to compute the deviation of the slave bit rate relative to the master bit rate after synchronization (f_{SLAVE} is the real slave node clock frequency).

$$\text{Baud rate deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{Over}) + \beta] \times \text{Baud rate}}{8 \times f_{SLAVE}}\right)\%$$

$$\text{Baud rate deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{Over}) + \beta] \times \text{Baud rate}}{8 \times \left(\frac{f_{TOL\_UNSYNCH}}{100}\right) \times f_{Nom}}\right)\%$$

$-0.5 \leq \alpha \leq +0.5 \quad -1 < \beta < +1$

$f_{TOL\_UNSYNCH}$ is the deviation of the real slave node clock from the nominal clock frequency. The LIN Standard imposes that it must not exceed ±15%. The LIN Standard imposes also that for communication between two nodes, their bit rate must not differ by more than ±2%. This means that the baud rate deviation must not exceed ±1%.

Therefore, a minimum value for the nominal clock frequency can be computed as follows:

$$f_{Nom}(\text{min}) = \left(100 \times \frac{[0.5 \times 8 \times (2 - \text{Over}) + 1] \times \text{Baud rate}}{8 \times \left(\frac{-15}{100} + 1\right) \times 1\%}\right)\text{Hz}$$

Examples:

- Baud rate = 20 kbit/s, OVER = 0 (Oversampling 16X) => f_{Nom}(min) = 2.64 MHz
- Baud rate = 20 kbit/s, OVER = 1 (Oversampling 8X) => f_{Nom}(min) = 1.47 MHz
- Baud rate = 1 kbit/s, OVER = 0 (Oversampling 16X) => f_{Nom}(min) = 132 kHz
- Baud rate = 1 kbit/s, OVER = 1 (Oversampling 8X) => f_{Nom}(min) = 74 kHz

38.7.8.9 Identifier Parity

A protected identifier consists of two subfields: the identifier and the identifier parity. Bits 0 to 5 are assigned to the identifier, and bits 6 and 7 are assigned to the parity.

The USART interface can generate/check these parity bits, but this feature can also be disabled. The user can choose between two modes via the FLEX_US_LINMR.PARDIS bit:

- PARDIS = 0:

  - During header transmission, the parity bits are computed and sent with the six least significant bits of the IDCHR field of the LIN Identifier Register (FLEX_US_LINIR). Bits 6 and 7 of this register are discarded.
  - During header reception, the parity bits of the identifier are checked. If the parity bits are wrong, an Identifier Parity error occurs (see section Parity). Only the six least significant bits of the IDCHR field are updated with the received Identifier. Bits 6 and 7 are stuck to 0.
- PARDIS = 1:
  - During header transmission, all the bits of the IDCHR field of the LIN Identifier Register (FLEX_US_LINIR) are sent on the bus.
  - During header reception, all the bits of the IDCHR field are updated with the received Identifier.

38.7.8.10 Node Action

Depending on the identifier, the node is affected—or not—by the LIN response. Consequently, after sending or receiving the identifier, the USART must be configured. There are three possible configurations:

- PUBLISH: the node sends the response.
- SUBSCRIBE: the node receives the response.
- IGNORE: the node is not concerned by the response, it does not send and does not receive the response.

This configuration is made by the LIN Node Action (NACT) field in USART LIN Mode Register (FLEX_US_LINMR).

Example: a LIN cluster that contains a master and two slaves:

- Data transfer from the master to slave 1 and to slave 2:

NACT(master) = PUBLISH

NACT(slave 1) = SUBSCRIBE

NACT(slave 2) = SUBSCRIBE

- Data transfer from the master to slave 1 only:

NACT(master) = PUBLISH

NACT(slave 1) = SUBSCRIBE

NACT(slave 2) = IGNORE

- Data transfer from slave 1 to the master:

NACT(master) = SUBSCRIBE

NACT(slave 1) = PUBLISH

NACT(slave 2) = IGNORE

- Data transfer from slave 1 to slave 2:

NACT(master) = IGNORE

NACT(slave 1) = PUBLISH

NACT(slave 2) = SUBSCRIBE

- Data transfer from slave 2 to the master and to slave 1:

NACT(master) = SUBSCRIBE

NACT(slave 1) = SUBSCRIBE

NACT(slave 2) = PUBLISH

38.7.8.11 Response Data Length

The LIN response data length is the number of data fields (bytes) of the response excluding the checksum.

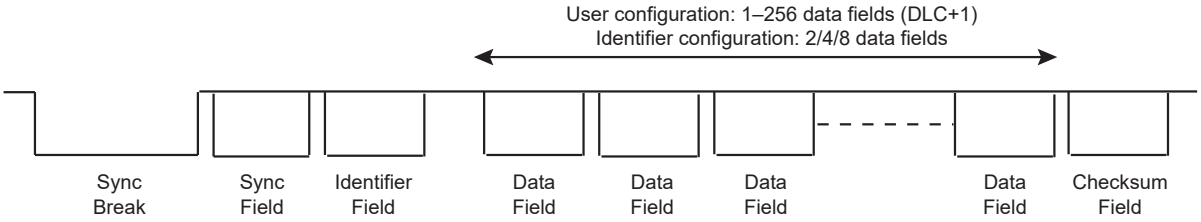
The response data length can either be configured by the user or be defined automatically by bits 4 and 5 of the Identifier (compatibility to LIN Specification 1.1). The user can choose between these two modes by the FLEX_US_LINMR.DLM bit:

- DLM = 0: The response data length is configured by the user via the FLEX_US_LINMR.DLC field. The response data length is equal to (DLC + 1) bytes. DLC can be programmed from 0 to 255, so the response can contain from 1 data byte up to 256 data bytes.
- DLM = 1: The response data length is defined by the Identifier (IDCHR in FLEX_US_LINIR) according to the table below. The FLEX_US_LINMR.DLC field is discarded. The response can contain 2 or 4 or 8 data bytes.

Table 38-12. Response Data Length if DLM = 1

IDCHR[5]	IDCHR[4]	Response Data Length (bytes)
0	0	2
0	1	2
1	0	4
1	1	8

Figure 38-43. Response Data Length



38.7.8.12 Checksum

The last field of a frame is the checksum. The checksum contains the inverted 8- bit sum with carry, over all data bytes or all data bytes and the protected identifier. Checksum calculation over the data bytes only is called classic checksum and it is used for communication with LIN 1.3 slaves. Checksum calculation over the data bytes and the protected identifier byte is called enhanced checksum and it is used for communication with LIN 2.0 slaves.

The USART can be configured to:

- Send/Check an Enhanced checksum automatically (CHKDIS = 0 & CHKTYP = 0)
- Send/Check a Classic checksum automatically (CHKDIS = 0 & CHKTYP = 1)
- Not send/check a checksum (CHKDIS = 1)

This configuration is made by the Checksum Type (CHKTYP) and Checksum Disable (CHKDIS) bits of FLEX_US_LINMR.

If the checksum feature is disabled, the user can send it manually all the same, by considering the checksum as a normal data byte and by adding 1 to the response data length (see section Response Data Length).

38.7.8.13 Frame Slot Mode

This mode is useful only for master nodes. It respects the following rule: each frame slot shall be longer than or equal to $t_{Frame\_Maximum}$.

If the Frame Slot mode is enabled (FSDIS = 0) and a frame transfer has been completed, the TXRDY flag is set again only after $t_{Frame\_Maximum}$ delay, from the start of frame. So the master node cannot send a new header if the frame slot duration of the previous frame is inferior to $t_{Frame\_Maximum}$.

If the Frame Slot mode is disabled (FSDIS = 1) and a frame transfer has been completed, the TXRDY flag is set again immediately.

The $t_{Frame\_Maximum}$ is calculated as follows:

If the Checksum is sent (CHKDIS = 0):

- $t_{Header\_Nominal} = 34 \times t_{bit}$
- $t_{Response\_Nominal} = 10 \times (NData + 1) \times t_{bit}$
- $t_{Frame\_Maximum} = 1.4 \times (t_{Header\_Nominal} + t_{Response\_Nominal} + 1)$(1)
- $t_{Frame\_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1 + 1) + 1) \times t_{bit}$

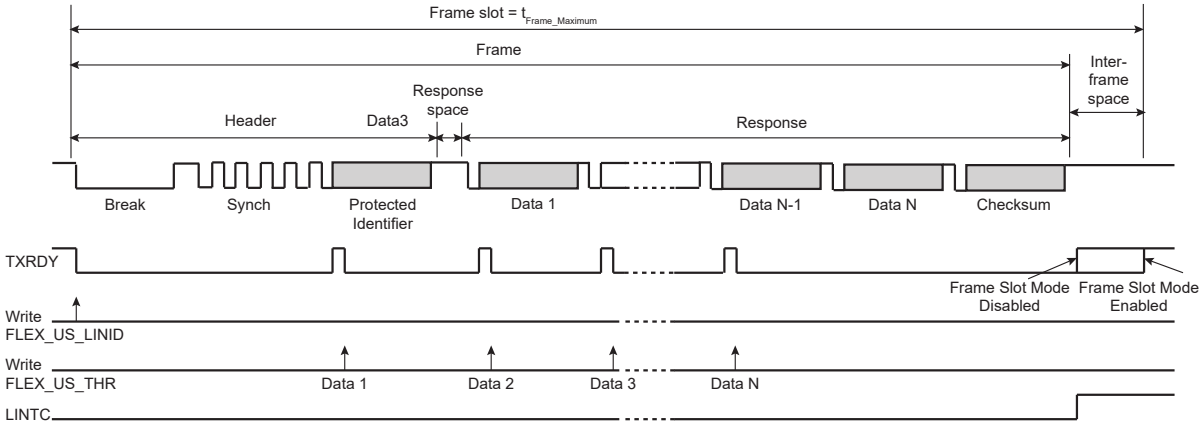
- $t_{Frame\_Maximum} = (77 + 14 \times DLC) \times t_{bit}$

If the Checksum is not sent (CHKDIS = 1):

- $t_{Header\_Nominal} = 34 \times t_{bit}$
- $t_{Response\_Nominal} = 10 \times NData \times t_{bit}$
- $t_{Frame\_Maximum} = 1.4 \times (t_{Header\_Nominal} + t_{Response\_Nominal} + 1)$(1)
- $t_{Frame\_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1) + 1) \times t_{bit}$
- $t_{Frame\_Maximum} = (63 + 14 \times DLC) \times t_{bit}$

Note: 1. The term “+1” leads to an integer result for $t_{Frame\_Maximum}$ (LIN Specification 1.3).

Figure 38-44. Frame Slot Mode



38.7.8.14 LIN Errors

38.7.8.14.1 Bit Error

This error is generated in master of slave node configuration, when the USART is transmitting and if the transmitted value on the Tx line is different from the value sampled on the Rx line. If a bit error is detected, the transmission is aborted at the next byte border.

This error is reported by the FLEX_US_CSR.LINBE flag.

38.7.8.14.2 Inconsistent Synch Field Error

This error is generated in slave node configuration, if the Synch Field character received is other than 0x55.

This error is reported by the FLEX_US_CSR.LINISFE flag.

38.7.8.14.3 Identifier Parity Error

This error is generated in slave node configuration, if the parity of the identifier is wrong. This error can be generated only if the parity feature is enabled (PARDIS = 0).

This error is reported by the FLEX_US_CSR.LINIPE flag.

38.7.8.14.4 Checksum Error

This error is generated in master of slave node configuration, if the received checksum is wrong. This flag can be set to 1 only if the checksum feature is enabled (CHKDIS = 0).

This error is reported by the FLEX_US_CSR.LINCE flag.

38.7.8.14.5 Slave Not Responding Error

This error is generated in master of slave node configuration, when the USART expects a response from another node (NACT = SUBSCRIBE) but no valid message appears on the bus within the time given by the maximum length of the message frame, $t_{Frame\_Maximum}$ (see section Frame Slot Mode). This error is disabled if the USART does not expect any message (NACT = PUBLISH or NACT = IGNORE).

This error is reported by the FLEX_US_CSR.LINSNRE.

38.7.8.14.6 Synch Tolerance Error

This error is generated in slave node configuration if, after the clock synchronization procedure, it appears that the computed baud rate deviation compared to the initial baud rate is superior to the maximum tolerance FTol_Unsynch (±15%).

This error is reported by the FLEX_US_CSR.LINSTE flag.

38.7.8.14.7 Header Timeout Error

This error is generated in slave node configuration, if the Header is not entirely received within the time given by the maximum length of the Header, $t_{Header\_Maximum}$.

This error is reported by the FLEX_US_CSR.LINHTE flag.

38.7.8.15 LIN Frame Handling

38.7.8.15.1 Master Node Configuration

- Write FLEX_US_CR.TXEN and FLEX_US_CR.RXEN to enable both the transmitter and the receiver.
- Write FLEX_US_MR.USART_MODE to select the LIN mode and the master node configuration.
- Write FLEX_US_BRGR.CD and FLEX_US_BRGR.FP to configure the baud rate.
- Write NACT, PARDIS, CHKDIS, CHKTYPE, DLCM, FSDIS and DLC in FLEX_US_LINMR to configure the frame transfer.
- Check that FLEX_US_CSR. TXRDY is set to 1.
- Write FLEX_US_LINIR.IDCHR to send the header.

What comes next depends on the NACT configuration:

- Case 1: NACT = PUBLISH, the USART sends the response.
  - Wait until FLEX_US_CSR. TXRDY rises.
  - Write FLEX_US_THR.TCHR to send a byte.
  - If all the data have not been written, repeat the two previous steps.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.
- Case 2: NACT = SUBSCRIBE, the USART receives the response.
  - Wait until FLEX_US_CSR.RXRDY rises.
  - Read FLEX_US_RHR.RCHR.
  - If all the data have not been read, repeat the two previous steps.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.
- Case 3: NACT = IGNORE, the USART is not concerned by the response.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.

Figure 38-45. Master Node Configuration, NACT = PUBLISH

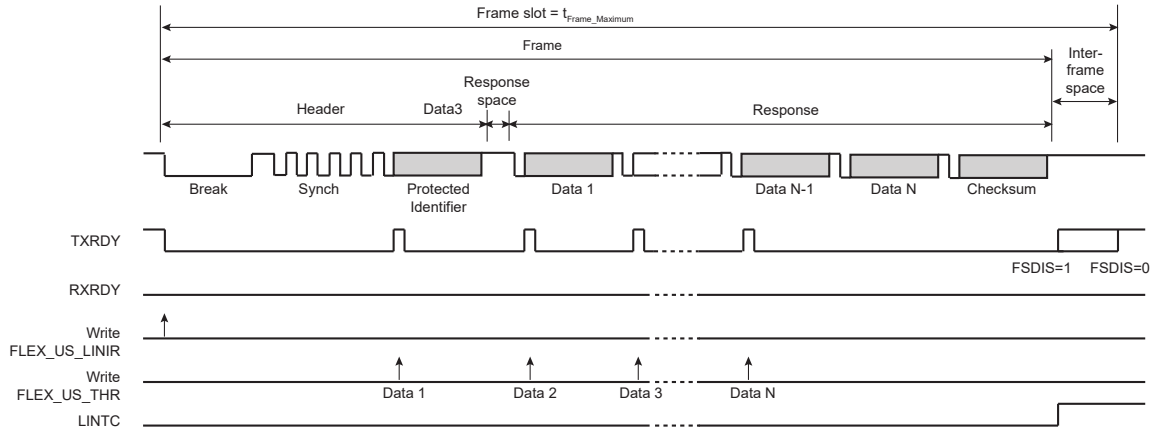


Figure 38-46. Master Node Configuration, NACT = SUBSCRIBE

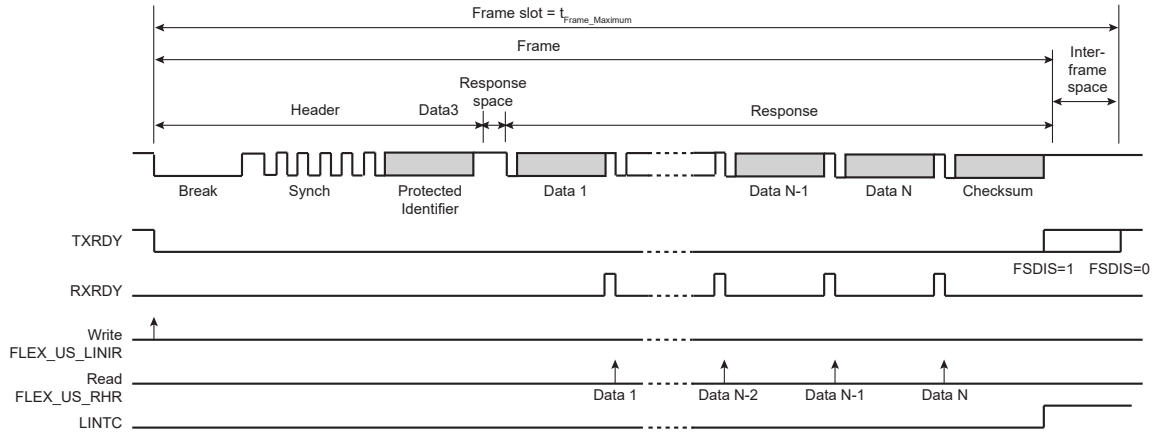
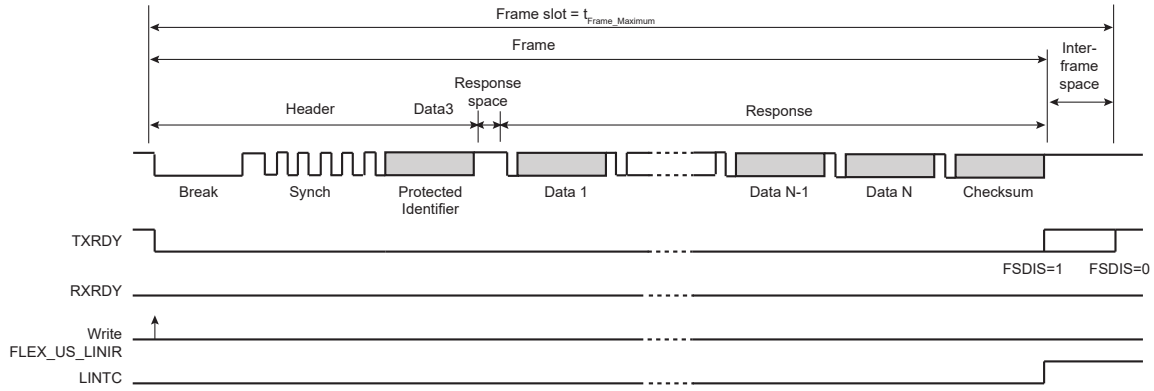


Figure 38-47. Master Node Configuration, NACT = IGNORE



38.7.8.15.2 Slave Node Configuration

- Write FLEX_US_CR.TXEN and FLEX_US_CR.RXEN to enable both the transmitter and the receiver.
- Write FLEX_US_MR.USART_MODE to select the LIN mode and the slave node configuration.
- Write FLEX_US_BRGR.CD and FLEX_US_BRGR.FP to configure the baud rate.
- Wait until FLEX_US_CSR.LINID rises.
- Check LINISFE and LINPE errors.
- Read FLEX_US_RHR.IDCHR.
- Write NACT, PARDIS, CHKDIS, CHKTYPE, DLCM and DLC in FLEX_US_LINMR to configure the frame transfer.

IMPORTANT: If the NACT configuration for this frame is PUBLISH, FLEX_US_LINMR must be written with NACT = PUBLISH even if this field is already correctly configured, in order to set the TXREADY flag and the corresponding write transfer request.

What comes next depends on the NACT configuration:

- Case 1: NACT = PUBLISH, the LIN controller sends the response.
  - Wait until FLEX_US_CSR.TXRDY rises.
  - Write FLEX_US_THR.TCHR to send a byte.
  - If all the data have not been written, repeat the two previous steps.
  - Wait until FLEX_US_CSR. LINTC rises.
  - Check the LIN errors.
- Case 2: NACT = SUBSCRIBE, the USART receives the response.
  - Wait until FLEX_US_CSR.RXRDY rises.
  - Read FLEX_US_RHR.RCHR.

  - If all the data have not been read, repeat the two previous steps.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.
- Case 3: NACT = IGNORE, the USART is not concerned by the response.
  - Wait until FLEX_US_CSR.LINTC rises.
  - Check the LIN errors.

Figure 38-48. Slave Node Configuration, NACT = PUBLISH

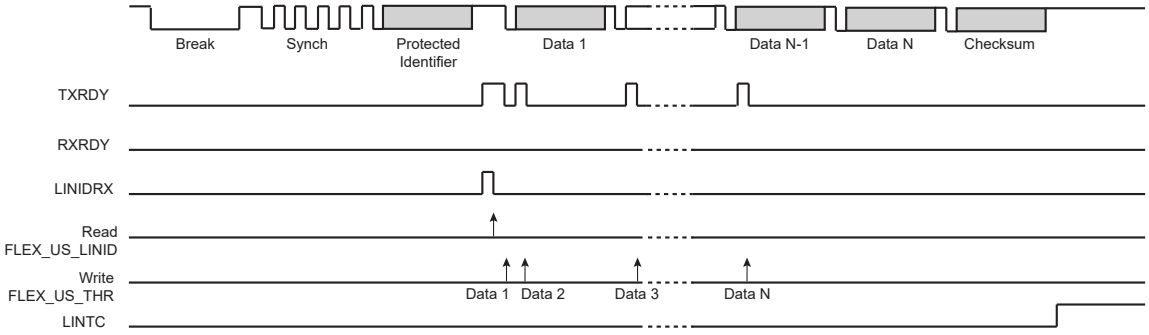


Figure 38-49. Slave Node Configuration, NACT = SUBSCRIBE

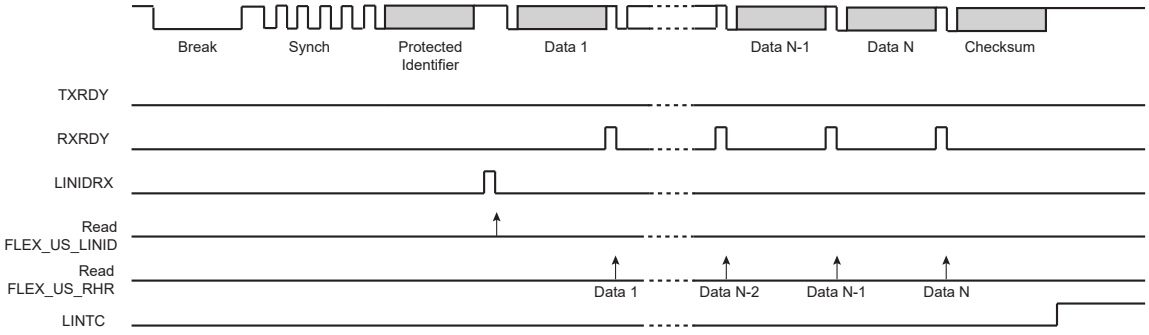
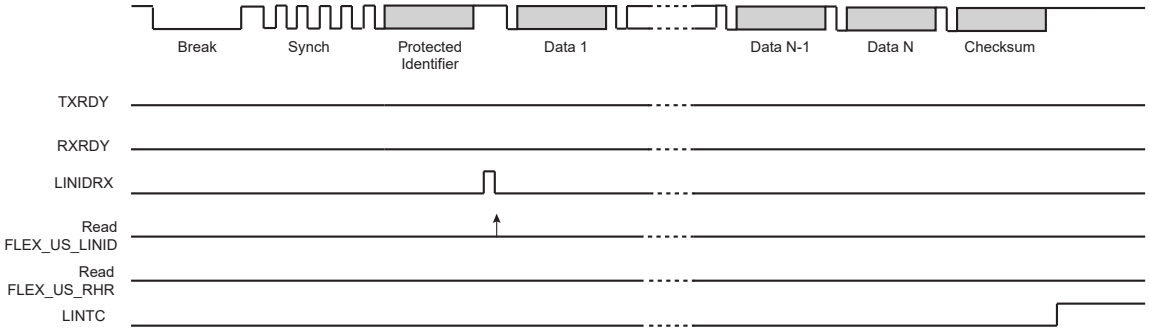


Figure 38-50. Slave Node Configuration, NACT = IGNORE



38.7.8.16 LIN Frame Handling with the DMAC

The USART can be used in association with the DMAC in order to transfer data directly into/from the on- and off-chip memories without any processor intervention.

The DMAC uses the trigger flags, TXRDY and RXRDY, to write or read into the USART. The DMAC always writes in the Transmit Holding Register (FLEX_US_THR) and it always reads in the Receive Holding Register (FLEX_US_RHR). The size of the data written or read by the DMAC in the USART is always a byte.

38.7.8.16.1 Master Node Configuration

The user can choose between two DMAC modes by configuring the FLEX_US_LINMR.PDCM bit:

- PDCM = 1: The LIN configuration is stored in the WRITE buffer and it is written by the DMAC in the Transmit Holding register FLEX_US_THR (instead of the LIN Mode register FLEX_US_LINMR). Because the DMAC transfer size is limited to a byte, the transfer is split into two accesses. During the first access, the NACT,

PARDIS, CHKDIS, CHKTYP, DLM and FSDIS bits are written. During the second access, the 8-bit DLC field is written.

- PDCM = 0: The LIN configuration is not stored in the WRITE buffer and it must be written by the user in FLEX_US_LINMR.

The WRITE buffer also contains the Identifier and the data, if the USART sends the response (NACT = PUBLISH).

The READ buffer contains the data if the USART receives the response (NACT = SUBSCRIBE).

Figure 38-51. Master Node with DMAC (PDCM = 1)

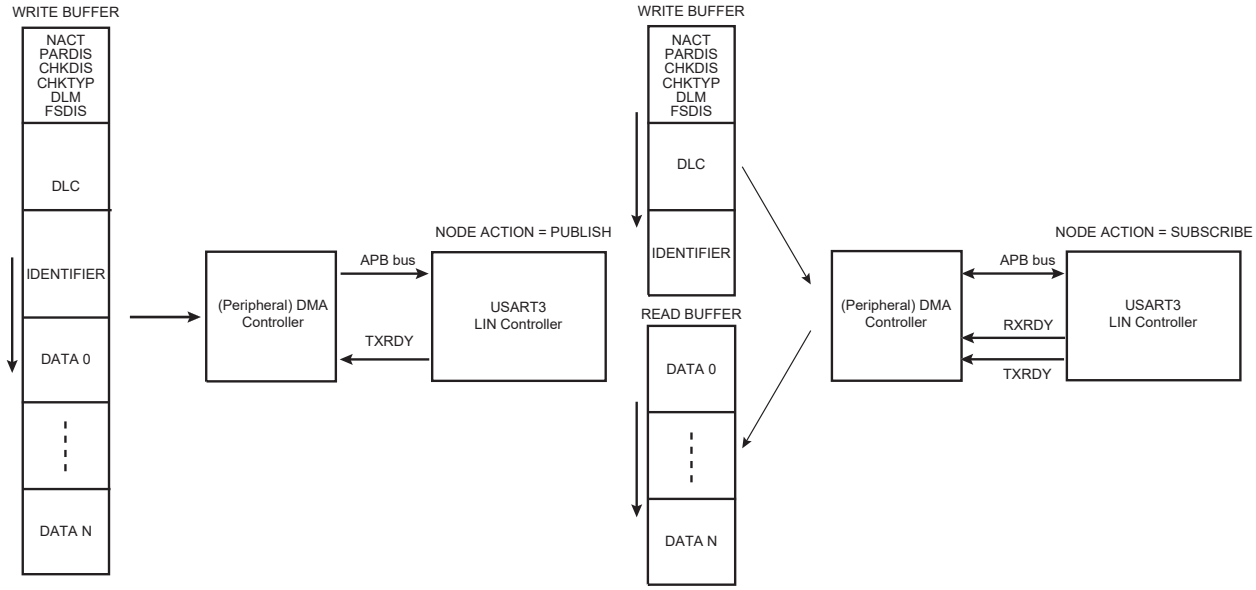
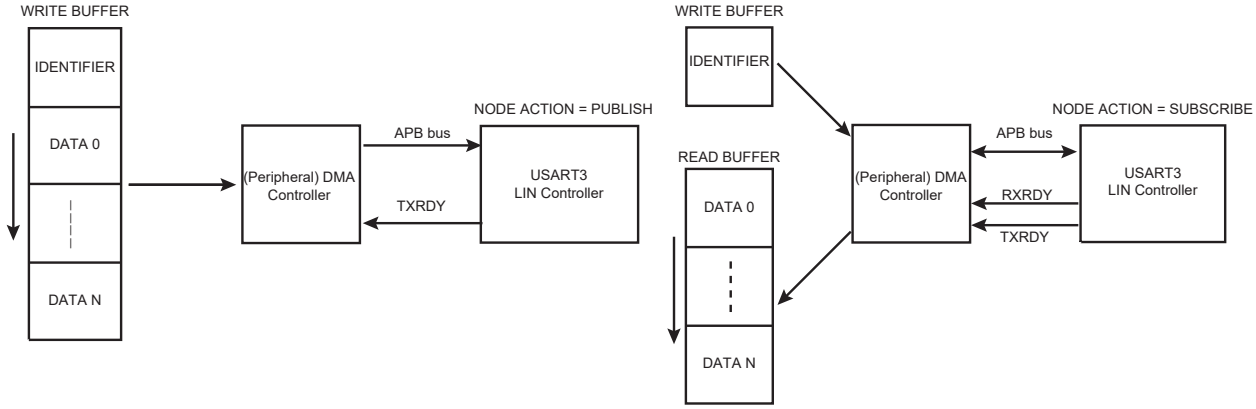


Figure 38-52. Master Node with DMAC (PDCM = 0)



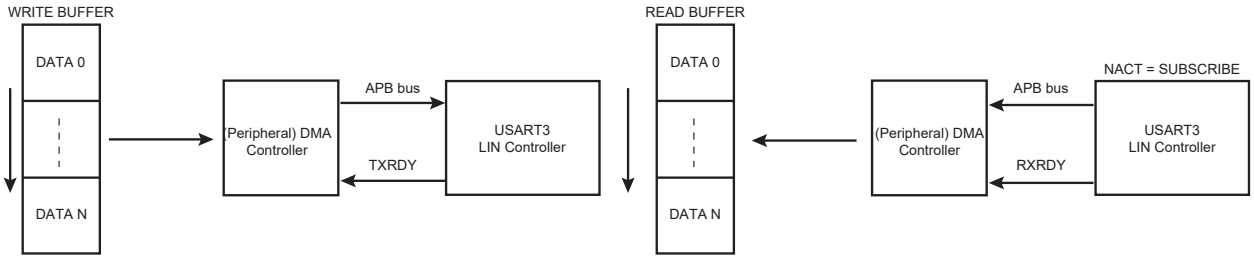
38.7.8.16.2 Slave Node Configuration

In this configuration, the DMAC transfers only the data. The identifier must be read by the user in the LIN Identifier Register (FLEX_US_LINIR). The LIN mode must be written by the user in FLEX_US_LINMR.

The WRITE buffer contains the data if the USART sends the response (NACT = PUBLISH).

The READ buffer contains the data if the USART receives the response (NACT = SUBSCRIBE).

Figure 38-53. Slave Node with DMAC



38.7.8.17 Wakeup Request

Any node in a sleeping LIN cluster may request a wakeup.

In the LIN 2.0 specification, the wakeup request is issued by forcing the bus to the dominant state from 250 µs to 5 ms. For this, it is necessary to send the character 0xF0 in order to impose five successive dominant bits. Whatever the baud rate is, this character respects the specified timings.

- Baud rate min = 1 kbit/s -> t_{bit} = 1 ms -> 5 t_{bit} = 5 ms
- Baud rate max = 20 kbit/s -> t_{bit} = 50 µs -> 5 t_{bit} = 250 µs

In the LIN 1.3 specification, the wakeup request should be generated with the character 0x80 in order to impose eight successive dominant bits.

Using the FLEX_US_LINMR.WKUPTYP bit, the user can choose to send either a LIN 2.0 wakeup request (WKUPTYP = 0) or a LIN 1.3 wakeup request (WKUPTYP = 1).

A wakeup request is transmitted by writing the FLEX_US_CR.LINWKUP bit to 1. Once the transfer is completed, the LINTC flag is asserted in the Status Register (FLEX_US_CSR). It is cleared by writing a one to the FLEX_US_CR.RSTSTA bit.

38.7.8.18 Bus Idle Timeout

If the LIN bus is inactive for a certain duration, the slave nodes shall automatically enter in Sleep mode. In the LIN 2.0 specification, this timeout is defined as 4 seconds. In the LIN 1.3 specification, it is defined as 25,000 t_{bit}.

In slave Node configuration, the receiver timeout detects an idle condition on the RXD line. When a timeout is detected, the FLEX_US_CSR.TIMEOUT bit rises and can generate an interrupt, thus indicating to the driver to go into Sleep mode.

The timeout delay period (during which the receiver waits for a new character) is programmed in the FLEX_US_RTOR.TO field. If a zero is written to the TO field, the Receiver Timeout is disabled and no timeout is detected. The FLEX_US_CSR.TIMEOUT bit remains at 0. Otherwise, the receiver loads a 17-bit counter with the value programmed in TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, the FLEX_US_CSR.TIMEOUT bit rises.

If STTTO is performed, the counter clock is stopped until a first character is received.

If RETTO is performed, the counter starts counting down immediately from the value TO.

Table 38-13. Receiver Timeout Programming

LIN Specification	Baud Rate	Timeout period	TO
2.0	1,000 bit/s	4s	4,000
	2,400 bit/s		9,600
	9,600 bit/s		38,400
	19,200 bit/s		76,800
	20,000 bit/s		80,000
1.3	–	25,000 t_{bit}	25,000

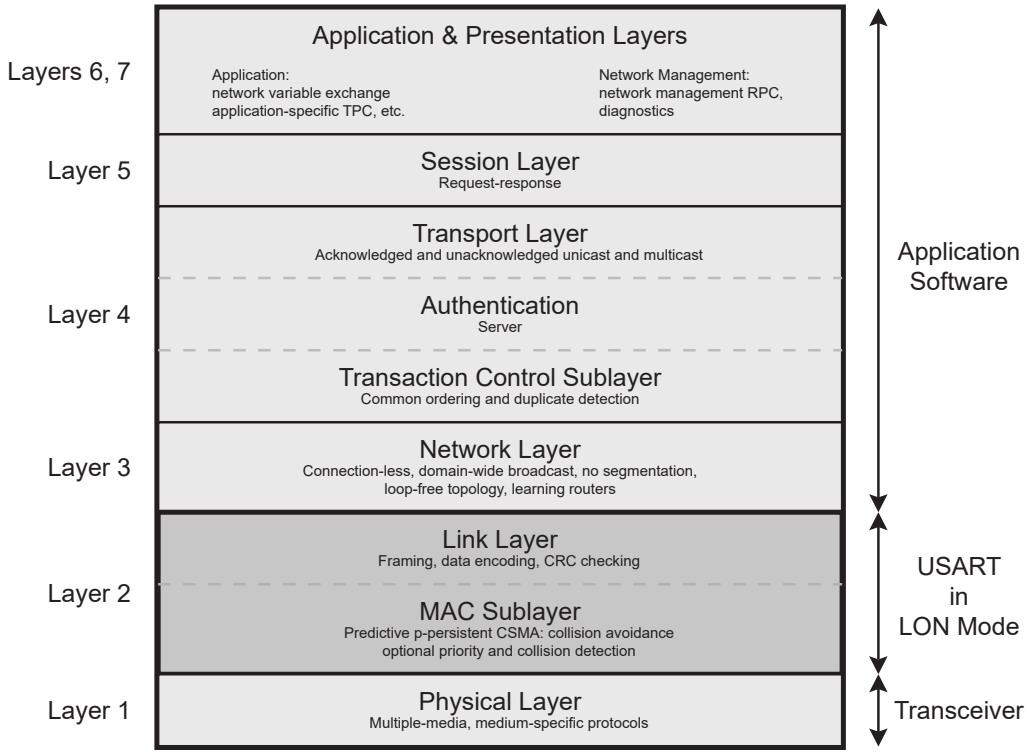
38.7.9 LON Mode

The LON mode provides connectivity to the local operating network (LON).

The LON standard covers all seven layers of the OSI (Open Systems Interconnect) reference model from the physical interfaces such as wired, power line, RF, and IP to the application layer and all layers in between.

The LON mode enables the transmission and reception of Physical Protocol Data Unit (PPDU) frames with minimum intervention from the microprocessor.

Figure 38-54. LON Protocol Layering



The USART configured in LON mode is a full-layer 2 implementation including standard timings handling, framing (transmit and receive PPDU frames), backlog estimation and other features. At the frame encoding/decoding level, differential Manchester encoding is used (also known as CDP). When configured in LON mode, there is no embedded digital line filter, thus the optimal usage is node-to-node communication.

38.7.9.1 Mode of Operation

To configure the USART to act as a LON node, the USART_MODE field of the USART Mode Register (FLEX_US_MR) must be set to 0x9.

To avoid unpredictable behavior, any change of the LON node configuration must be preceded by a software reset of the transmitter and the receiver (except the initial node configuration after a hardware reset) and followed by a transmitter/receiver enable. See section Receiver and Transmitter Control.

38.7.9.2 Receiver and Transmitter Control

See section Receiver and Transmitter Control.

38.7.9.3 Character Transmission

A LON frame is made up of a preamble, a data field (up to 256 bytes) and a 16-bit CRC field. The preamble and CRC fields are automatically generated and the LON node starts the transmission algorithm on a write to LON_L2HDR. See section Sending a Frame.

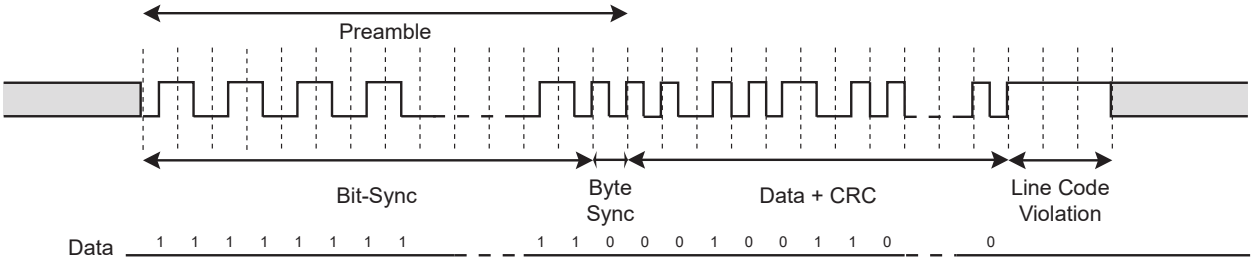
38.7.9.4 Character Reception

When receiving a LON frame, the Receive Holding Register (FLEX_US_RHR) is updated upon completed character reception and the RXRDY bit in the Status register rises. If a character is completed while the RXRDY bit is set, the

OVRE (Overrun Error) bit is set. The LON preamble field is only used for synchronization, therefore only the Data and CRC fields are transmitted to the Receive Holding Register (FLEX_US_RHR). See section Sending a Frame.

38.7.9.5 LON Frame

Figure 38-55. LON Framing



38.7.9.5.1 Encoding / Decoding

The USART configured in LON mode encodes transmitted data and decodes received data using differential Manchester encoding. In differential Manchester encoding, a '1' bit is indicated by making the first half of the signal equal to the last half of the previous bit's signal (no transition at the start of the bit-time). A '0' bit is indicated by making the first half of the signal opposite to the last half of the previous bit's signal (a zero bit is indicated by a transition at the beginning of the bit-time). As is the case with normal Manchester encoding, missing transition at the middle of bit-time represents a Manchester code violation.

The FLEX_US_MAN.RXIDLEV bit informs the USART of the receiver line idle state value (receiver line inactive) thus ensuring higher reliability of preamble synchronization. By default, RXIDLEV is set (receiver line is at level 1 when there is no activity).

Differential Manchester encoding is polarity-insensitive.

Figure 38-56. LON PPDU

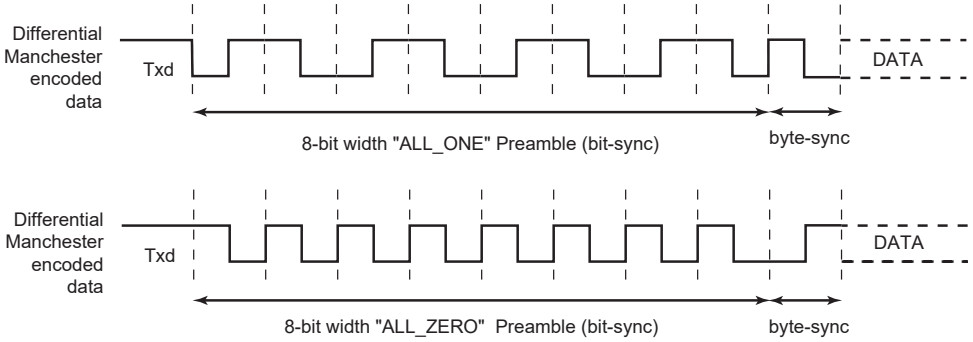
38.7.9.5.2 Preamble Transmission

Each LON frame begins with a preamble of variable length which consists of a bit-sync field and a byte-sync field. The LONPL field of the USART LON Preamble Register (FLEX_US_LONPR) defines the preamble length. Note that a preamble length of '0' is not allowed.

The LON implementation allows two different preamble patterns ALL_ONE and ALL_ZERO which can be configured through the TX_PL field of the USART Manchester Configuration Register (FLEX_US_MAN). The following figure illustrates and defines the valid patterns.

Other preamble patterns are not supported.

Figure 38-57. Preamble Patterns



38.7.9.5.3 Preamble Reception

LON received frames begin with a preamble of variable length. The receiving algorithm does not check the preamble length, although a minimum of length of 4 bits is required for the receiving algorithm to consider the received preamble as valid.

As is the case with LON preamble transmission, two preamble patterns (ALL_ONE and ALL_ZERO) are allowed and can be configured through the RX_PL field of the USART Manchester Configuration Register (FLEX_US_MAN). The above figure illustrates and defines the valid patterns.

Other preamble patterns are not supported.

38.7.9.5.4 Header Transmission

Each LON frame, after sending the preamble, starts with the frame header also called L2HDR according to CEA-709 specification. This header consist of the priority bit, the alternative path bit and the backlog increment. It is the first data to be sent.

In LON mode, the transmitting algorithm starts when FLEX_US_LONL2HDR is written (it is the first data to send).

38.7.9.5.5 Header Reception

Each LON frame, after receiving the preamble, receives the frame header also called L2HDR according to CEA-709 specification. This header consists of the priority bit, the alternative path bit, and the backlog increment.

The frame header is the first received data and the RXRDY bit rises as soon as the frame header as been received and stored in the Receive Holding Register (FLEX_US_RHR).

38.7.9.5.6 Data

Data are sent/received serially after the preamble transmission/reception. Data can be either sent/received MSB first or LSB first depending on the MSBF bit value in the USART Mode Register (FLEX_US_MR).

38.7.9.5.7 CRC

The two last bytes of LON frames are dedicated to CRC.

When transmitting, the CRC of the frame is automatically generated and sent when expected.

When receiving frames, the CRC is automatically checked and the FLEX_US_CSR.LCRCE flag is set if the calculated CRC does not match the received one. Note that the two received CRC bytes are seen as two additional data from the user point of view.

38.7.9.5.8 End Of Frame

The USART configured in LON mode terminates the frame with a three t_{bit} long Manchester code violation. After sending the last CRC bit, it maintains the data transitionless during three bit periods.

38.7.9.6 LON Operating Modes

38.7.9.6.1 Transmitting/Receiving Modules

According to the LON node configuration and LON network state, the transmitting module is activated if a transmission request has been made and access to the LON bus granted. It returns to idle state once the transmission ends.

According to the LON node configuration and LON network state, the receiving module is activated if a valid preamble is detected and the transmitting module is not activated.

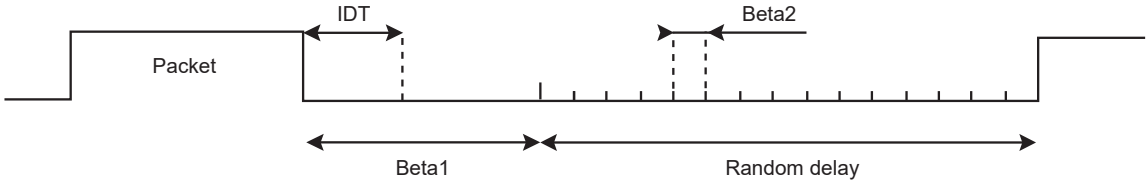
38.7.9.6.2 comm_type

In CEA-709 standard 2, communication configurations are defined and configurable through the comm_type variable. The comm_type variable value can be set in the USART LON Mode Register (FLEX_US_LONMR) through the COMMT bit. The selection of the comm_type determines the MAC behavior in the following ways:

- comm_type = 1:
  - An indeterminate time is defined during the Beta 1 period in which all transitions on the channel are ignored (see figure below).
  - The MAC sublayer ignores collisions occurring during the first 25% of the transmitted preamble. Optionally (according to the FLEX_US_LONMR.CDTAIL bit), it ignores collisions reported following the transmission of the CRC but prior to the end of transmission.
  - If a collision is detected during preamble transmission, the MAC sublayer can terminate the packet if so configured according to the FLEX_US_LONMR.TCOL bit. Collisions detected after the preamble have been sent do not terminate transmission.
- comm_type = 2:
  - No indeterminate time is defined at the MAC sublayer.

– The MAC sublayer always terminates the packet upon notification of a collision.

Figure 38-58. LON Indeterminate Time



38.7.9.6.3 Collision Detection

As an option of the CEA-709 standard collision detection is supported through an active low Collision Detect (CD) input from the transceiver.

The Collision Detection source can be either external (see section I/O Lines Description) or internal. The collision detection source selection is defined through the LCDS bit in the USART LON Mode Register (FLEX_US_LONMR).

The Collision Detection feature can be activated through the COLDET bit of the USART LON Mode Register. If the collision detection feature is enabled and CD signal goes low for at least half t_{bit} period then a collision is detected and reported as defined in section comm_type.

38.7.9.6.4 Collision Detection Mode

As defined in section comm_type, if comm_type = 1 the LON node can be configured to either not terminate transmission upon collision notification during preamble transmission or terminate transmission.

The FLEX_US_LONMR.TCOL bit allows to decide whether to terminate transmission or not upon collision notification during preamble transmission.

38.7.9.6.5 Collision Detection after CRC

As defined in section comm_type, if comm_type = 1 the LON node can be configured to ignore collisions reported after the CRC has been sent but prior to the end of the frame.

The FLEX_US_LONMR.CDTAIL bit can be used to decide whether such collision notifications must be considered or not.

38.7.9.6.6 Random Number Generation

The Predictive p-persistent CSMA algorithm defined in the CEA-709.1 Standard is based on a random number generation.

This random number is automatically generated by an internal algorithm.

In addition, a USART IC DIFF Register (FLEX_US_ICDIFF) is available to avoid that two same chips with the same software generate the same random number after reset. The value of this register is used by the internal algorithm to generate the random number. Therefore, putting a different value here for each chip ensures that the random number generated after a reset at the same time will not be the same. It is recommended to put the chip ID code here.

38.7.9.7 LON Node Backlog Estimation

As defined in CEA-709, the LON node maintains its own backlog estimation. The node backlog estimation is initially set to one, will always be greater than 1 and will never exceed 63. If the node backlog estimation exceeds the maximum backlog value, the backlog value is set to 63 and a backlog overflow error flag is set (LBLOVFE flag).

The node backlog estimation is incremented each time a frame is sent or received successfully. The increment to the backlog is encoded into the link layer header, and represents the number of messages that the packet shall cause to be generated upon reception.

The backlog decrements under one of the following conditions:

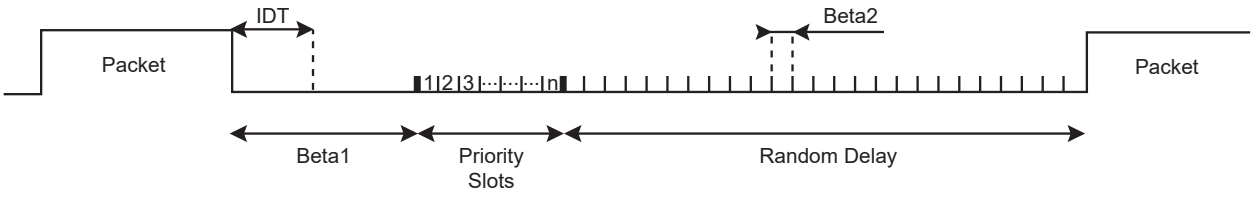
- On waiting to transmit: If Wbase randomizing slots go by without channel activity.
- On receive: If a packet is received with a backlog increment of ‘0’.
- On transmit: If a packet is transmitted with a backlog increment of ‘0’.
- On idle: If a packet cycle time expires without channel activity.

38.7.9.7.1 Optional Collision Detection Feature And Backlog Estimation

Each time a frame is transmitted and a collision occurred, the backlog is incremented by 1. In this case, the backlog increment encoded in the link layer is ignored.

38.7.9.8 LON Timings

Figure 38-59. LON Timings



38.7.9.8.1 Beta2

A node wishing to transmit generates a random delay T. This delay is an integer number of randomizing slots of duration Beta2.

The beta2 length (in t_{bit}) is configurable through FLEX_US_FIDI. Note that a length of '0' is not allowed.

38.7.9.8.2 Beta1 Tx/Rx

Beta1 is the period immediately following the end of a packet cycle (see the above figure). A node attempting to transmit monitors the state of the channel, and if it detects no transmission during the Beta1 period, it determines the channel to be idle.

The Beta1 value is different depending on the previous packet type (received packet or transmitted packet).

Beta1Rx and Beta1Tx length can be configured respectively through the USART LON Beta1 Rx Register (FLEX_US_LONB1RX) and the USART LON Beta1 Tx Register (FLEX_US_LONB1TX). Note that a length of '0' is not allowed.

38.7.9.8.3 Pcycle Timer

The packet cycle timer is reset to its initial value whenever the backlog is changed. It is started (begins counting down at its current value) whenever the MAC layer becomes idle. An idle MAC layer is defined as:

- Not receiving
- Not transmitting,
- Not waiting to transmit,
- Not timing Beta1,
- Not waiting for priority slots, and not waiting for the first Wbase randomizing window to complete.

On transition from idle to either transmit or receive, the packet cycle timer is halted.

The pcycle timer value can be configured in FLEX_US_TTGR. Note that '0' value is not allowed.

38.7.9.8.4 Wbase

The wbase timer represents the base windows size. Its duration, derived from Beta2, equals 16 Beta2 slots.

38.7.9.8.5 Priority Slots

On a channel by channel basis, the protocol supports optional priority. Priority slots, if any, follow immediately after the Beta1 period that follows the transmission of a packet (see the above figure). The number of priority slots per channel ranges from 0 to 127.

The number of priority slots in the LON network configuration is defined through the PSNB field of the USART LON Priority Register (FLEX_US_LONPRIO). And the priority slot affected to the LON node, if any, is defined through the FLEX_US_LONPRIO.NPS field.

38.7.9.8.6 Indeterminate Time

See section comm_type.

Like Beta1, the IDT value is different depending on whether the previous frame was transmitted or received.

IDTRx and IDTTx can be configured respectively through the USART LON IDT Rx Register (FLEX_US_LONIDTRX) and the USART LON IDT Tx Register (FLEX_US_LONIDTTX).

38.7.9.8.7 End of Frame Condition

The USART configured in LON mode terminates the frame with a three t_{bit} long Manchester code violation. After sending the last CRC bit, it maintains the data transitionless during three bit periods.

While receiving data, the USART configured in LON mode detects an end of frame condition after a t_{eof} transitionless Manchester code violation. The EOFS field in the USART LON Mode Register can configure t_{eof}.

38.7.9.9 LON Errors

All these flags can be read in the LON Channel Status Register (FLEX_US_CSR) and generate interrupts if configured in the LON Interrupt Enable Register (FLEX_US_IER).

These flags can be reset by writing a one to the FLEX_US_CR.RSTSTA bit.

38.7.9.9.1 Underrun Error

If the USART is in LON mode and if a character is sent while the Transmit Holding Register (FLEX_US_THR) is empty, the UNRE bit flag is set.

38.7.9.9.2 Collision Detection

The LCOL flag is set whenever a valid collision has been detected and the LON node is configured to report it (see section Collision Detection).

38.7.9.9.3 LON Frame Early Termination

The LFET flag is set whenever a LON frame has been terminated early due to collision detection.

38.7.9.9.4 Reception Error

The LCRCE flag is set if the received frame has an erroneous CRC and the flag LSFE is set if the received frame is too short (LON frames must be at least 8 bytes long).

These flags can be read in FLEX_US_CSR.

38.7.9.9.5 Backlog Overflow

The LBLOVFE flag is set if the LON node backlog estimation goes over 63, which is the maximum backlog value.

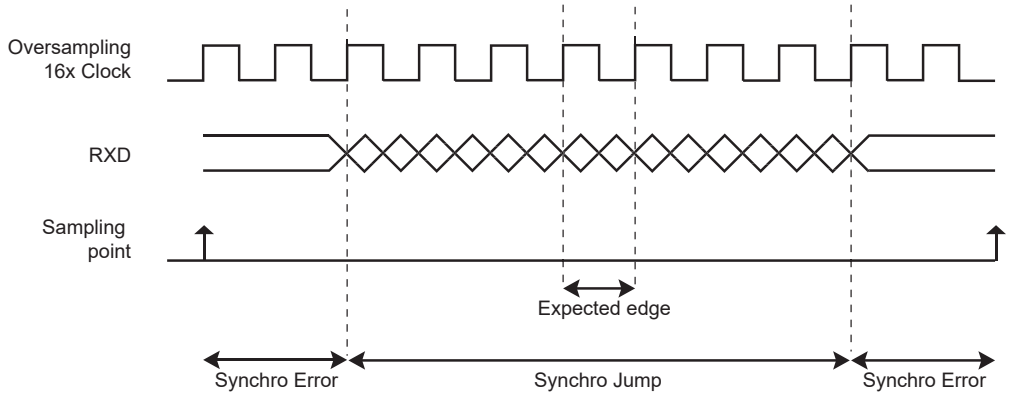
38.7.9.10 Drift Compensation

It may happen that while receiving a frame, the baud rate used by the sender is not exactly the one expected, due to sender clock drifting for instance. In such case, the hardware drift compensation algorithm is used to recover up to 16% clock drift (expected baud rate ±16% is supported).

Drift compensation is available only in 16X Oversampling mode. To enable the hardware system, the DRIFT bit of the FLEX_US_MAN register must be set. If the RXD edge is between one and three 16X clock cycles away from the expected edge, then the period is shortened or lengthened accordingly to center the RXD edge.

Drift compensation hardware feature allows up to 16% clock drift to be handled, provided system clock is fast enough compared to the selected baud rate.

Figure 38-60. Bit Resynchronization



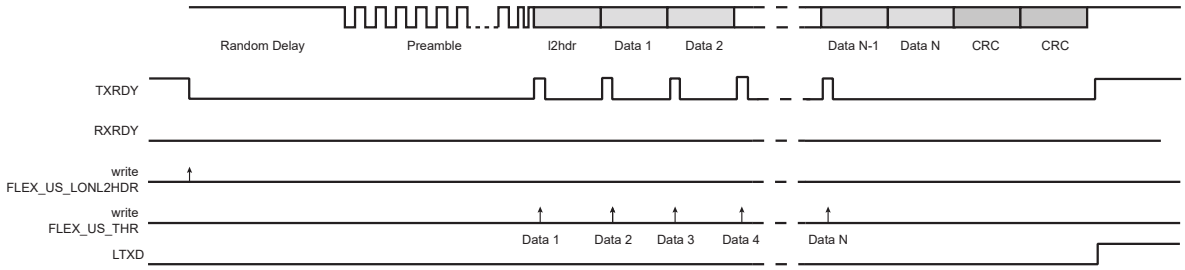
38.7.9.11 LON Frame Handling

38.7.9.11.1 Sending a Frame

1. Write FLEX_US_CR.TXEN and FLEX_US_CR.RXEN to enable both the transmitter and the receiver.
2. Write FLEX_US_MR.USART_MODE to select the LON mode configuration.
3. Write FLEX_US_BRGR.CD and FLEX_US_BRGR.FP to configure the baud rate.

4. Write COMMT, COLDET, TCOL, CDTAIL, RDMNBM and DMAM in FLEX_US_LONMR to configure the LON operating mode.
5. Write BETA2, BETA1TX, BETA1RX, PCYCLE, PSNB, NPS, IDTTX and ITDRX respectively in FLEX_US_FIDI, FLEX_US_LONB1TX, FLEX_US_LONB1RX, FLEX_US_TTGR, FLEX_US_LONPRIO, FLEX_US_LONIDTTX and FLEX_US_LONIDTRX to set the LON network configuration.
6. Write FLEX_US_MAN.TX_PL to select the preamble pattern to use.
7. Write LONPL and LONDL in FLEX_US_LONPR and FLEX_US_LONDL to set the frame transfer.
8. Check that FLEX_US_CSR.TXRDY is set to 1.
9. Write FLEX_US_LONL2HDR to send the header.
10. Wait until FLEX_US_CSR.TXRDY rises.
11. Write FLEX_US_THR.TCHR to send a byte.
12. If all the data have not been written, repeat the two previous steps.
13. Wait until FLEX_US_CSR.LTXD rises.
14. Check the LON errors.

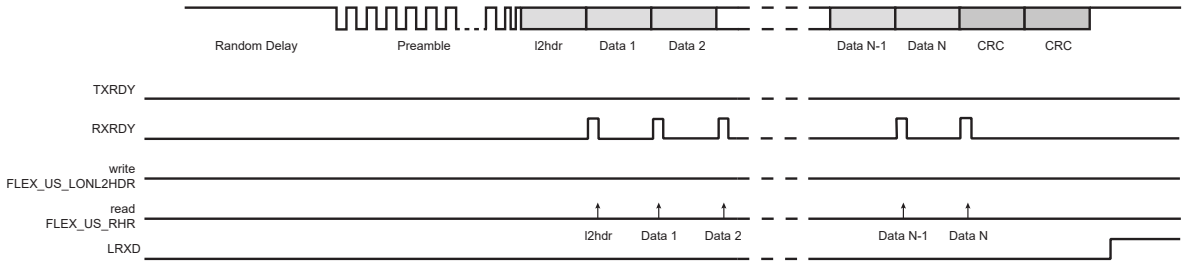
Figure 38-61. Tx Frame



38.7.9.11.2 Receiving a Frame

1. Write FLEX_US_CR.TXEN and FLEX_US_CR.RXEN to enable both the transmitter and the receiver.
2. Write FLEX_US_MR.USART_MODE to select the LON mode configuration.
3. Write FLEX_US_BRGR.CD and FLEX_US_BRGR.FP to configure the baud rate.
4. Write COMMT, COLDET, TCOL, CDTAIL, RDMNBM and DMAM in FLEX_US_LONMR to configure the LON operating mode.
5. Write BETA2, BETA1TX, BETA1RX, PCYCLE, PSNB, NPS, IDTTX and ITDRX respectively in FLEX_US_FIDI, FLEX_US_LONB1TX, FLEX_US_LONB1RX, FLEX_US_TTGR, FLEX_US_LONPRIO, FLEX_US_LONIDTTX and FLEX_US_LONIDTRX to set the LON network configuration.
6. Write FLEX_US_MAN.RXIDLEV and FLEX_US_MAN.RX_PL to indicate the receiver line value and select the preamble pattern to use.
7. Wait until FLEX_US_CSR.RXRDY rises.
8. Read FLEX_US_RHR.RCHR.
9. If all the data and the two CRC bytes have not been read, repeat the two previous steps.
10. Wait until FLEX_US_CSR.LRXD rises.
11. Check the LON errors.

Figure 38-62. Rx Frame



38.7.9.12 LON Frame Handling with the Peripheral DMA Controller

The USART can be used in association with the DMA Controller in order to transfer data directly into/from the on- and off-chip memories without any processor intervention.

The DMA uses the trigger flags, TXRDY and RXRDY, to write or read into the USART. The DMA always writes in the Transmit Holding Register (FLEX_US_THR) and it always reads in the Receive Holding Register (FLEX_US_RHR). The size of the data written or read by the DMA in the USART is always a byte.

38.7.9.12.1 Configuration

The user can choose between two DMA modes by the DMAM bit in the LON Mode register

(FLEX_US_LONMR):

- DMAM = 1: The LON frame data length (DATAL) is stored in the WRITE buffer and it is written by the DMA in the Transmit Holding register FLEX_US_THR (instead of the LON Data Length register FLEX_US_LONDL).
- DMAM = 0: The LON frame data length (DATAL) is not stored in the WRITE buffer and it must be written by the user in the LON Data Length Register (FLEX_US_LONDL).

In both DMA modes, L2HDR is considered as a data and its value must be stored in the WRITE buffer as the first data to write.

Figure 38-63. DMAM = 1

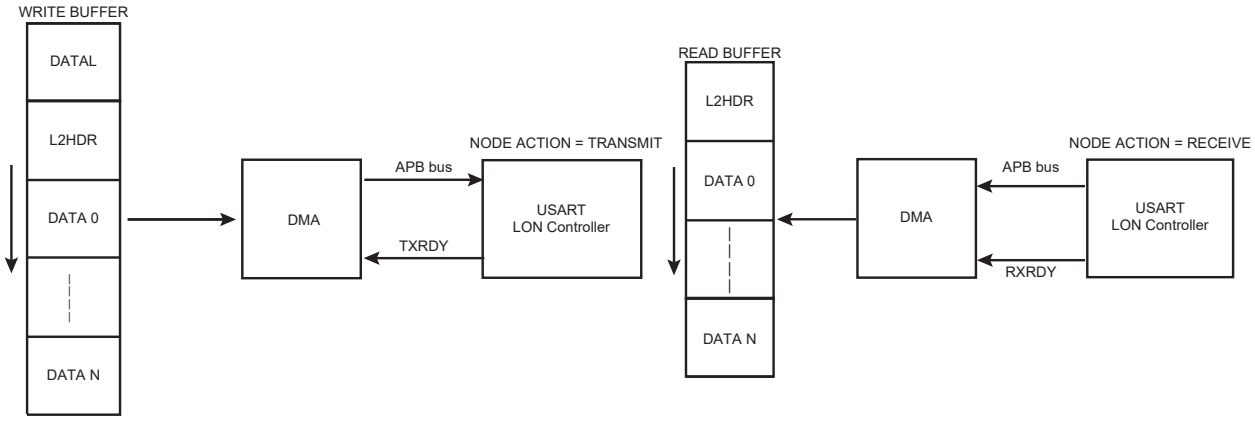
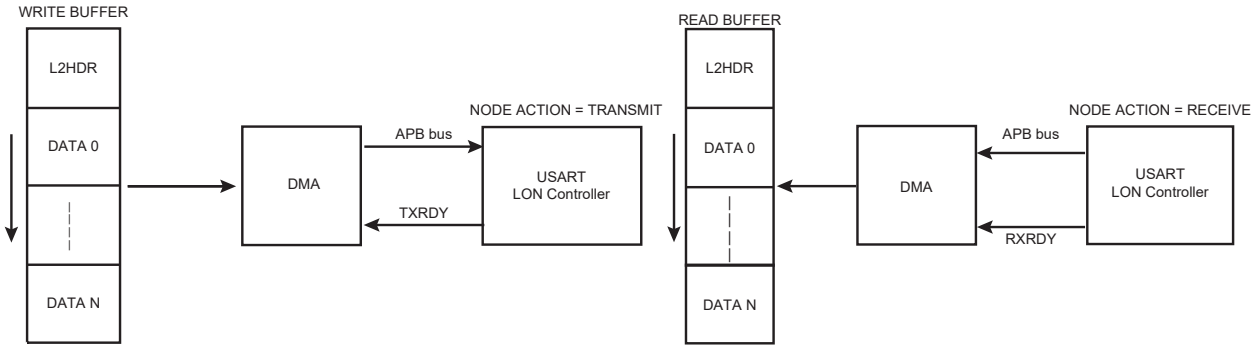


Figure 38-64. DMAM = 0



38.7.9.12.2 DMA and Collision Detection

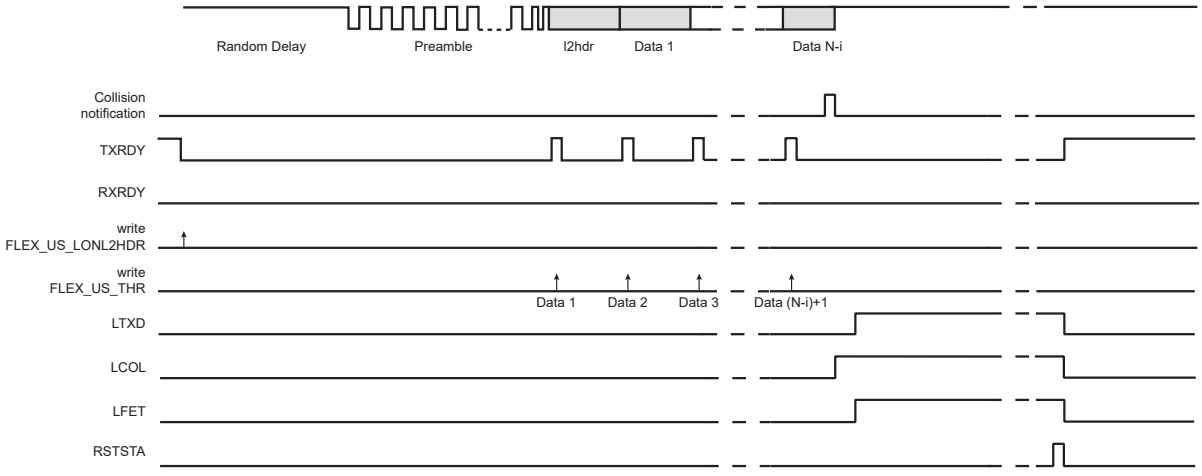
As explained in section comm_type, depending on LON configuration the transmission may be terminated early upon collision notification which means that the DMA transfer may be stopped before its end.

In case of early end of transmission due to collision detection, the USART in LON mode acts as follows:

- Send the end of frame trigger.
- Hold down TXRDY, thus avoiding any additional DMA transfer.
- Set the LTXD, LCOL and LFET flags in FLEX_US_CSR.
- Wait for the application to reconfigure the DMA.

- Wait until the LCOL and LFET flags are cleared through the FLEX_US_CR.RSTSTA bit (it will release the TXRDY signal).

Figure 38-65. DMA, Collision and Early Frame Termination



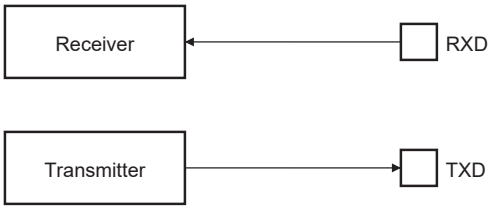
38.7.10 Test Modes

The USART can be programmed to operate in three different test modes. The internal loopback capability allows on-board diagnostics. In Loopback mode, the USART interface pins are disconnected or not and reconfigured for loopback internally or externally.

38.7.10.1 Normal Mode

Normal mode connects the RXD pin on the receiver input and the transmitter output on the TXD pin.

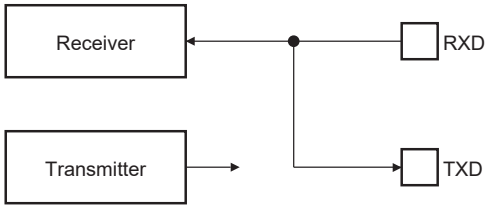
Figure 38-66. Normal Mode Configuration



38.7.10.2 Automatic Echo Mode

Automatic Echo mode allows bit-by-bit retransmission. When a bit is received on the RXD pin, it is sent to the TXD pin, as shown in the following figure. Programming the transmitter has no effect on the TXD pin. The RXD pin is still connected to the receiver input, thus the receiver remains active.

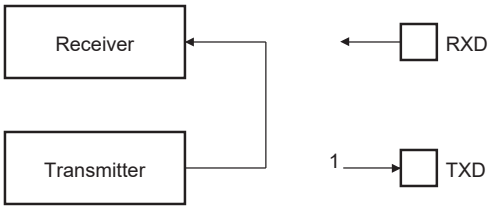
Figure 38-67. Automatic Echo Mode Configuration



38.7.10.3 Local Loopback Mode

Local Loopback mode connects the output of the transmitter directly to the input of the receiver, as shown in the following figure. The TXD and RXD pins are not used. The RXD pin has no effect on the receiver and the TXD pin is continuously driven high, as in idle state.

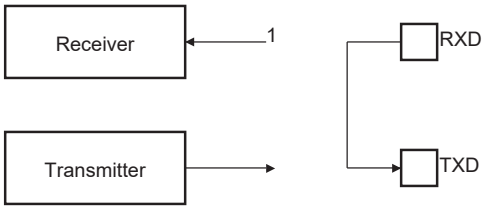
Figure 38-68. Local Loopback Mode Configuration



38.7.10.4 Remote Loopback Mode

Remote Loopback mode directly connects the RXD pin to the TXD pin, as shown in the following figure. The transmitter and the receiver are disabled and have no effect. This mode allows bit-by-bit retransmission.

Figure 38-69. Remote Loopback Mode Configuration



38.7.11 USART FIFOs

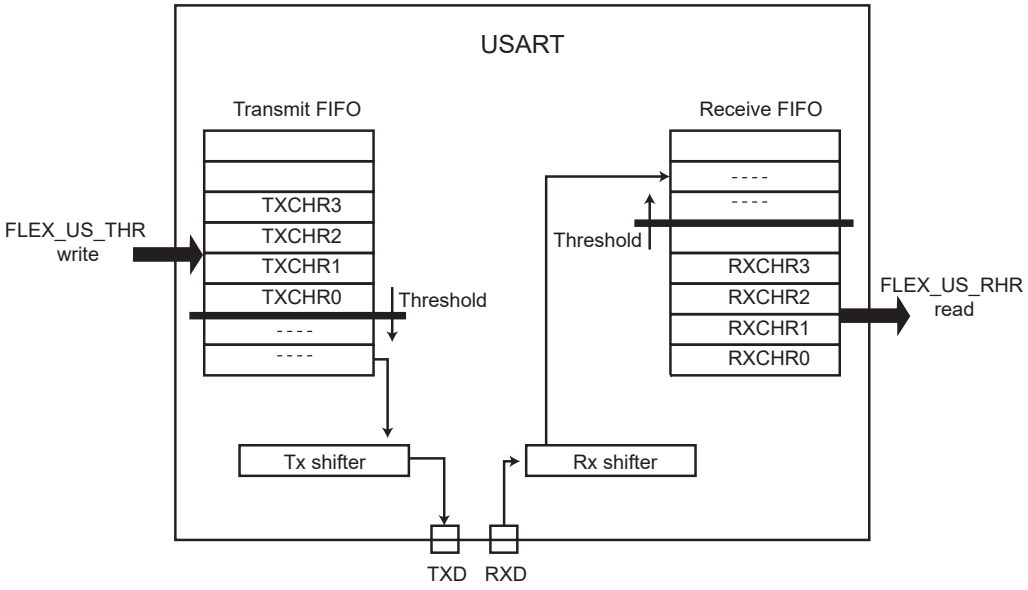
38.7.11.1 Overview

The USART includes two FIFOs which can be enabled/disabled using FLEX_US_CR.FIFOEN/FIFODIS. It is recommended to disable both the transmitter and the receiver before enabling or disabling the FIFOs, using the FLEX_US_CR.TXDIS/RXDIS bits.

Writing FLEX_US_CR.FIFOEN to ‘1’ enables a 8-data Transmit FIFO and a 8-data Receive FIFO.

It is possible to write or to read single or multiple data in the same access to FLEX_US_THR/RHR. See sections USART Single Data Mode and USART Multiple Data Mode.

Figure 38-70. FIFOs Block Diagram



38.7.11.2 Sending Data with FIFO Enabled

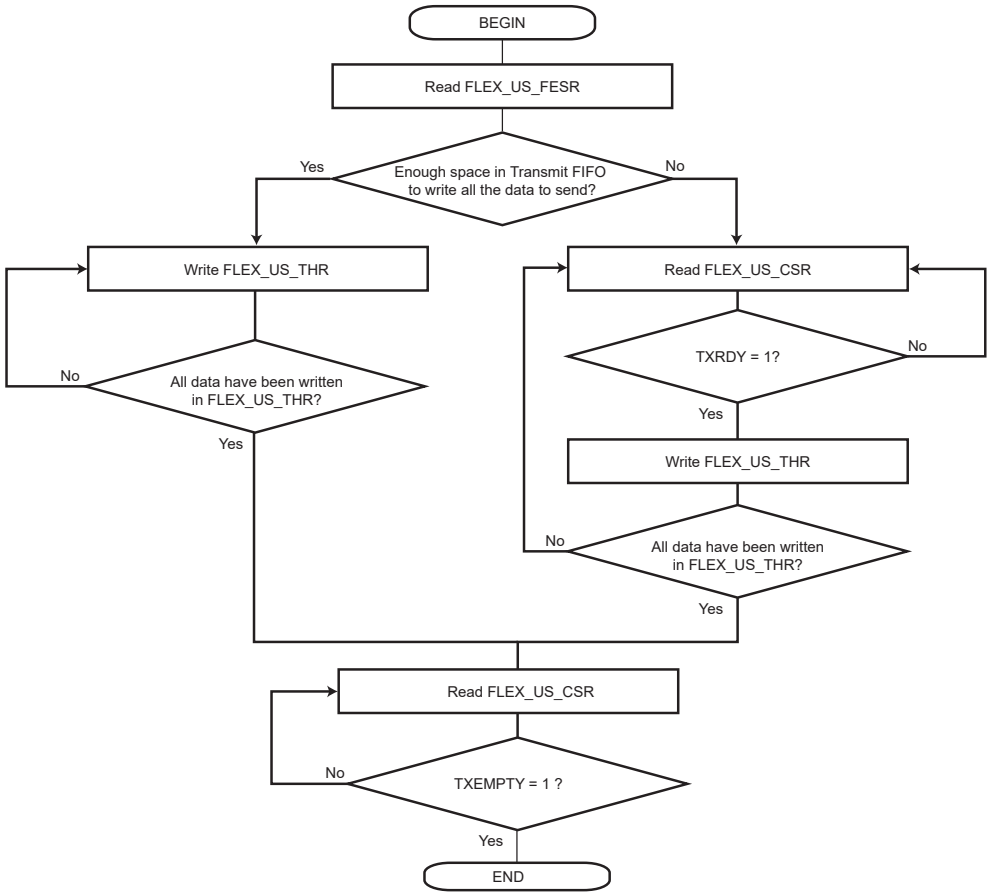
When the Transmit FIFO is enabled, write access to FLEX_US_THR loads the Transmit FIFO.

The FIFO level is provided in FLEX_US_FLR.TXFL. If the FIFO can accept the number of data to be transmitted, there is no need to monitor FLEX_US_CSR.TXRDY and the data can be successively written in FLEX_US_THR.

If the FIFO cannot accept the data due to insufficient space, wait for the TXRDY flag to be set before writing the data in FLEX_US_THR.

When the space in the FIFO allows only a portion of the data to be written, the TXRDY flag must be monitored before writing the remaining data.

Figure 38-71. Sending Data with FIFO Enabled

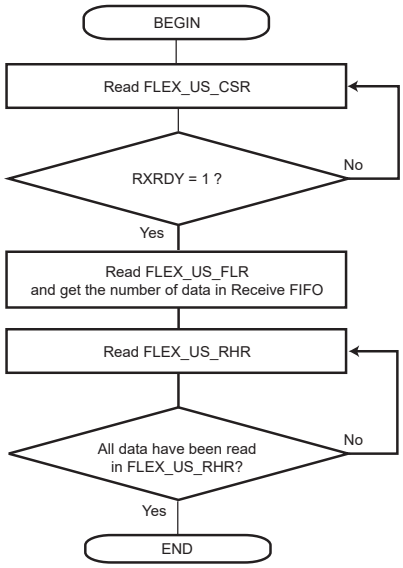


38.7.11.3 Receiving Data with FIFO Enabled

When the Receive FIFO is enabled, FLEX_US_RHR access reads the FIFO.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with FLEX_US_FLR.RXFL. All the data can be read successively in FLEX_US_RHR without checking the RXRDY flag between each access.

Figure 38-72. Receiving Data with FIFO Enabled



38.7.11.4 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using FLEX_US_CR.TXFCLR/RXFCLR.

38.7.11.5 TXEMPTY, TXRDY and RXRDY Behavior

FLEX_US_CSR.TXEMPTY, FLEX_US_CSR.TXRDY and FLEX_US_CSR.RXRDY flags display a specific behavior when FIFOs are enabled.

The TXEMPTY flag is cleared as long as there are characters in the Transmit FIFO or in the internal shift register. TXEMPTY is set when there are no characters in the Transmit FIFO and in the internal shift register.

TXRDY indicates if a data can be written in the Transmit FIFO. Thus the TXRDY flag is set as long as the Transmit FIFO can accept new data. See figure TXRDY in Single Data Mode and TXRDYM = 0.

RXRDY indicates if an unread data is present in the Receive FIFO. Thus the RXRDY flag is set as soon as one unread data is in the Receive FIFO. See figure RXRDY in Single Data Mode and RXRDYM = 0 below.

TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in the USART FIFO Mode Register (FLEX_US_FMR) to reduce the number of accesses to FLEX_US_RHR/THR. However, for some configurations, the following constraints apply:

- If FLEX_US_MR.MODE9 is set, FLEX_US_FMR.TXRDYM/RXRDYM must be cleared.
- If FLEX_US_MR.USART_MODE is set to either LON_MODE, LIN_MASTER or LIN_SLAVE, FLEX_US_FMR.TXRDYM/RXRDYM must be cleared.

See USART FIFO Mode Register for the FIFO configuration.

Figure 38-73. TXRDY in Single Data Mode and TXRDYM = 0

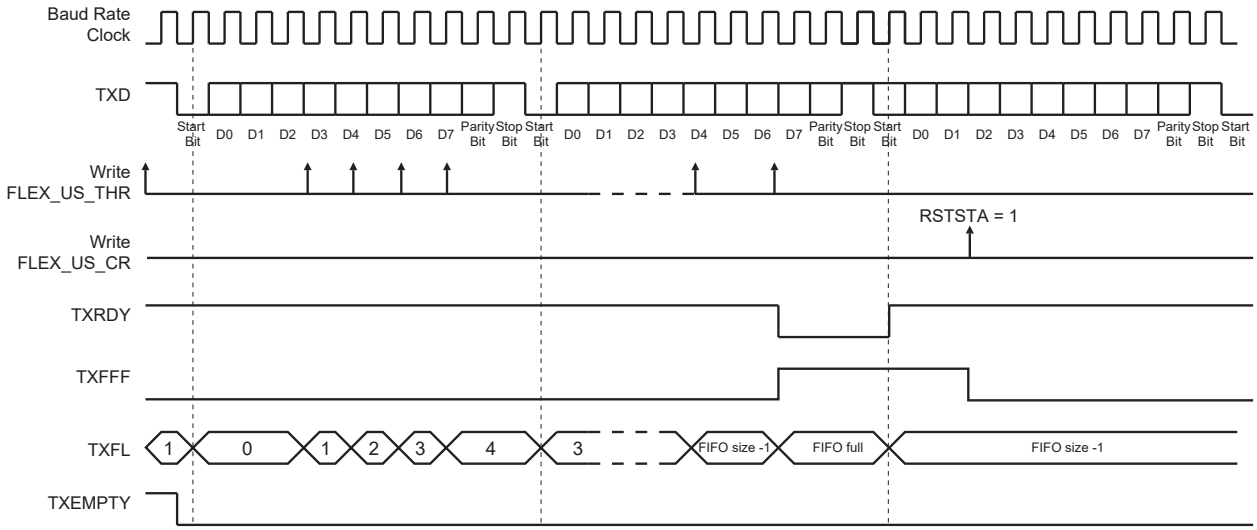
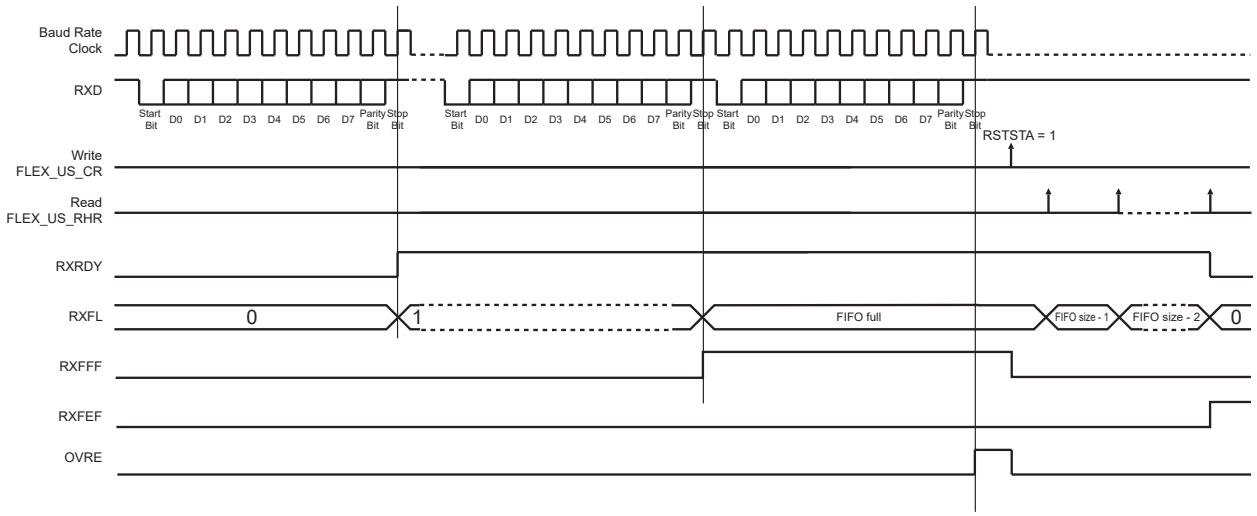


Figure 38-74. RXRDY in Single Data Mode and RXRDYM = 0



38.7.11.6 USART Single Data Mode

In Single Data mode, only one data is written every time FLEX_US_THR is accessed, and only one data is read every time FLEX_US_RHR is accessed.

When FLEX_US_FMR.TXRDYM = 0, the Transmit FIFO operates in Single Data mode.

When FLEX_US_FMR.RXRDYM = 0, the Receive FIFO operates in Single Data mode.

If FLEX_US_MR.MODE9 is set, or if FLEX_US_MR.USART_MODE is set to either LON_MODE, LIN_MASTER or LIN_SLAVE, the FIFOs must operate in Single Data mode.

See USART Receive Holding Register (FLEX_US_RHR) and USART Transmit Holding Register (FLEX_US_THR).

38.7.11.6.1 DMAC

The DMAC transfer type must be configured in bytes or halfwords when FIFOs operate in Single Data mode (the same applies when FIFOs are disabled).

38.7.11.7 USART Multiple Data Mode

Multiple Data mode minimizes the number of accesses by concatenating the data to send/read in one access.

When FLEX_US_FMR.TXRDYM > 0, the Transmit FIFO operates in Multiple Data mode.

When FLEX_US_FMR.RXRDYM > 0, the Receive FIFO operates in Multiple Data mode.

However, Multiple Data mode cannot be used for the following configurations:

- If FLEX_US_MR.MODE9 is set
- If FLEX_US_MR.USART_MODE is set to either LON_MODE, LIN_MASTER or LIN_SLAVE

In Multiple Data mode, it is possible to write/read up to four data in one FLEX_US_THR/FLEX_US_RHR access.

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read, if the access is a halfword size register access, then two data are written/read and, finally, if the access is a word-size register access, four data are written/read.

Written/read data are always right-aligned, as described in sections USART Receive Holding Register (FIFO Multi Data) and USART Transmit Holding Register (FIFO Multi Data).

As an example, if the Transmit FIFO is empty and there are six data to send, either of the following write accesses may be performed:

- six FLEX_US_THR-byte write accesses
- three FLEX_US_THR-halfword write accesses
- one FLEX_US_THR word write access and one FLEX_US_THR halfword write access

With a Receive FIFO containing six data, any of the following read accesses may be performed:

- six FLEX_US_RHR-byte read accesses
- three FLEX_US_RHR-halfword read accesses
- one FLEX_US_RHR-word read access and one FLEX_US_RHR-halfword read access

38.7.11.7.1 TXRDY and RXRDY Configuration

In Multiple Data mode, it is possible to write one or more data in the same FLEX_US_THR/FLEX_US_RHR access. The TXRDY flag indicates if one or more data can be written in the FIFO depending on the configuration of FLEX_US_FMR.TXRDYM/RXRDYM.

As an example, if four data are written each time in FLEX_US_THR, it is useful to configure the TXRDYM field to the value ‘2’ so that the TXRDY flag is at ‘1’ only when at least four data can be written in the Transmit FIFO.

In the same way, if four data are read each time in FLEX_US_RHR, it is useful to configure the RXRDYM field to the value ‘2’ so that the RXRDY flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

38.7.11.7.2 DMAC

When FIFOs operate in Multiple Data mode, the DMAC transfer type must be configured in byte, halfword or word depending on the FLEX_US_FMR.TXRDYM/RXRDYM settings.

38.7.11.8 Transmit FIFO Lock

- LIN Mode:

If a frame is aborted using the Abort LIN Transmission bit (FLEX_US_CR.LINABT), a lock is set on the Transmit FIFO, preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, resetting DMAC channels, etc., without any risk.

- LON Mode:

If a frame is terminated early due to collision, a lock is set on the Transmit FIFO preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, resetting DMAC channels, etc., without any risk.

The TXFLOCK bit in the USART FIFO Event Status Register (FLEX_US_FESR) is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared by setting FLEX_US_CR.TXFLCLR to ‘1’.

38.7.11.9 FIFO Pointer Error

A FIFO overflow is reported in FLEX_US_FESR.

If the Transmit FIFO is full and a write access is performed on FLEX_US_THR, it generates a Transmit FIFO pointer error and sets FLEX_US_FESR.TXFPTEF.

In Multiple Data mode, if the number of data written in FLEX_US_THR (according to the register access size) is greater than the free space in the Transmit FIFO, a Transmit FIFO pointer error is generated and FLEX_US_FESR.TXFPTEF is set.

A FIFO underflow is reported in FLEX_US_FESR.

In Multiple Data mode, if the number of data read in FLEX_US_RHR (according to the register access size) is greater than the number of unread data in the Receive FIFO, a Receive FIFO pointer error is generated and FLEX_US_FESR.RXFPTEF is set.

No pointer error occurs if the FIFO state/level is checked before writing/reading in FLEX_US_THR/FLEX_US_RHR. The FIFO state/level can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags may not behave as expected; their states should be ignored.

If a Transmit pointer error occurs, a transmitter reset must be performed using FLEX_US_CR.RSTTX. If a Receive pointer error occurs, a receiver reset must be performed using FLEX_US_CR.RSTRX.

38.7.11.10 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

The Transmit FIFO threshold can be set using the field FLEX_US_FMR.TXFTHRES. Each time the Transmit FIFO level goes from ‘above threshold’ to ‘equal to or below threshold’, the flag FLEX_US_FESR.TXFTHF is set. The application is warned that the Transmit FIFO has reached the defined threshold and that it can be reloaded.

The Receive FIFO threshold can be set using the field FLEX_US_FMR.RXFTHRES. Each time the Receive FIFO level goes from ‘below threshold’ to ‘equal to or above threshold’, the flag FLEX_US_FESR.RXFTHF is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The Receive FIFO threshold 2 can be set using the field FLEX_US_FMR.RXFTHRES2. Each time the Receive FIFO level goes from ‘above threshold 2’ to ‘equal to or below threshold 2’, the flag FLEX_US_FESR.RXFTHF2 is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The TXFTHF, RXFTHF and RXTHF2 flags can be configured to generate an interrupt using FLEX_US_FIER and FLEX_US_FIDR.

38.7.11.11 FIFO Flags

FIFOs come with a set of flags which can be configured to generate interrupts through FLEX_US_FIER and FLEX_US_FIDR.

FIFO flags state can be read in FLEX_US_FESR. They are cleared by writing FLEX_US_CR.RSTSTA to ‘1’.

38.7.12 USART Register Write Protection

The FLEXCOM operating mode (FLEX_MR.OPMODE) must be set to FLEX_MR_OPMODE_USART to enable access to the write protection registers.

To prevent any single software error from corrupting USART behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable) bit in the USART Write Protection Mode Register (FLEX_US_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the USART Write Protection Status Register (FLEX_US_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_US_WPSR.

The following registers can be write-protected when WPEN is set:

- USART Mode Register
- USART Baud Rate Generator Register
- USART Receiver Timeout Register
- USART Transmitter Timeguard Register
- USART FI DI RATIO Register
- USART IrDA FILTER Register
- USART Manchester Configuration Register

- USART LON Mode Register
- USART LON Beta1 Tx Register
- USART LON Beta1 Rx Register
- USART LON Priority Register
- USART LON IDT Tx Register
- USART LON IDT Rx Register
- USART IC DIFF Register
- USART Comparison Register

38.8 SPI Functional Description

38.8.1 Modes of Operation

The SPI operates in Master mode or in Slave mode.

- The SPI operates in Master mode by writing a 1 to the MSTR bit in the SPI Mode Register (FLEX_SPI_MR):
  - The pins NPCS0 to NPCS3 are all configured as outputs.
  - The SPCK pin is driven.
  - The MISO line is wired on the receiver input.
  - The MOSI line is driven as an output by the transmitter.
- The SPI operates in Slave mode if the MSTR bit in FLEX_SPI_MR is written to 0:
  - The MISO line is driven by the transmitter output.
  - The MOSI line is wired on the receiver input.
  - The SPCK pin is driven by the transmitter to synchronize the receiver.
  - The NPCS0 pin becomes an input, and is used as a slave select signal (NSS).
  - Pins NPCS1 to NPCS3 are not driven and can be used for other purposes.

The data transfers are identically programmable for both modes of operation. The bit rate generator is activated only in Master mode.

38.8.2 Data Transfer

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the SPI Chip Select Register (FLEX_SPI_CSR). The clock phase is programmed with the NCPHA bit. These two parameters determine the edges of the clock signal on which data are driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Consequently, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are connected and require different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

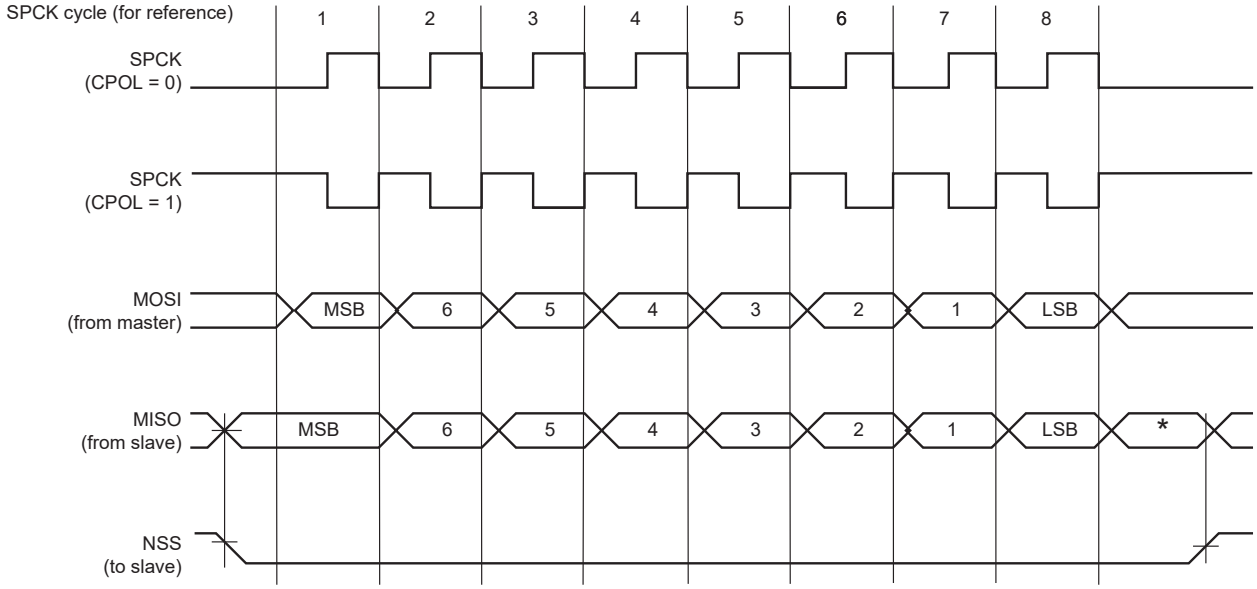
The following table shows the four modes and corresponding parameter settings.

Table 38-14. SPI Bus Protocol Mode

SPI Mode	CPOL	NCPHA	Shift SPCK Edge	Capture SPCK Edge	SPCK Inactive Level
0	0	1	Falling	Rising	Low
1	0	0	Rising	Falling	Low
2	1	1	Rising	Falling	High
3	1	0	Falling	Rising	High

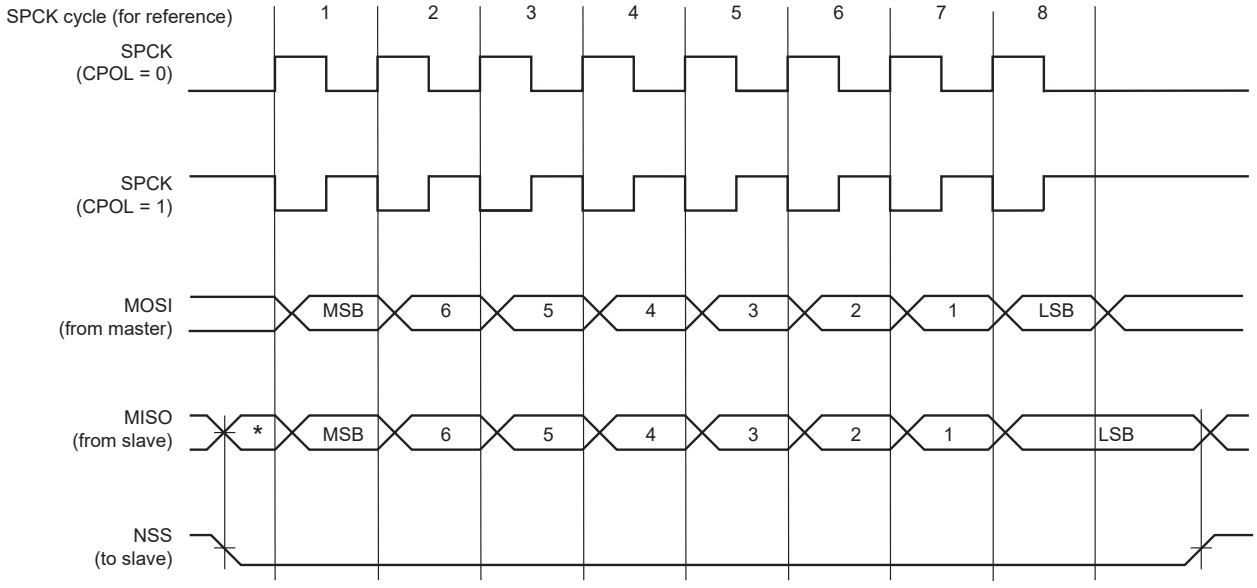
The following figures show examples of data transfers.

Figure 38-75. SPI Transfer Format (NCPHA = 1, 8 bits per transfer)



* Not defined.

Figure 38-76. SPI Transfer Format (NCPHA = 0, 8 bits per transfer)



* Not defined.

38.8.3 Master Mode Operations

When configured in Master mode, the SPI operates on the clock generated by the internal programmable bit rate generator. It fully controls the data transfers to and from the slave(s) connected to the SPI bus. The SPI drives the chip select line to the slave and the serial clock signal (SPCK).

The SPI features two holding registers, the Transmit Data Register (FLEX_SPI_TDR) and the Receive Data Register (FLEX_SPI_RDR), and a single shift register. The holding registers maintain the data flow at a constant rate.

After enabling the SPI, a data transfer starts when the processor writes to FLEX_SPI_TDR. The written data are immediately transferred in the shift register and the transfer on the SPI bus starts. While the data in the shift register is shifted on the MOSI line, the MISO line is sampled and shifted in the shift register. Data cannot be loaded in

FLEX_SPI_RDR without transmitting data. If there is no data to transmit, a dummy data can be used (FLEX_SPI_TDR filled with ones). When the WDRBT bit is set, a new data cannot be transmitted if FLEX_SPI_RDR has not been read. If Receiving mode is not required, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the SPI Status Register (FLEX_SPI_SR) can be discarded.

Before writing the TDR, the FLEX_SPI_MR.PCS field must be set in order to select a slave.

If new data are written in FLEX_SPI_TDR during the transfer, it is kept in FLEX_SPI_TDR until the current transfer is completed. Then, the received data are transferred from the shift register to FLEX_SPI_RDR, the data in FLEX_SPI_TDR is loaded in the shift register and a new transfer starts.

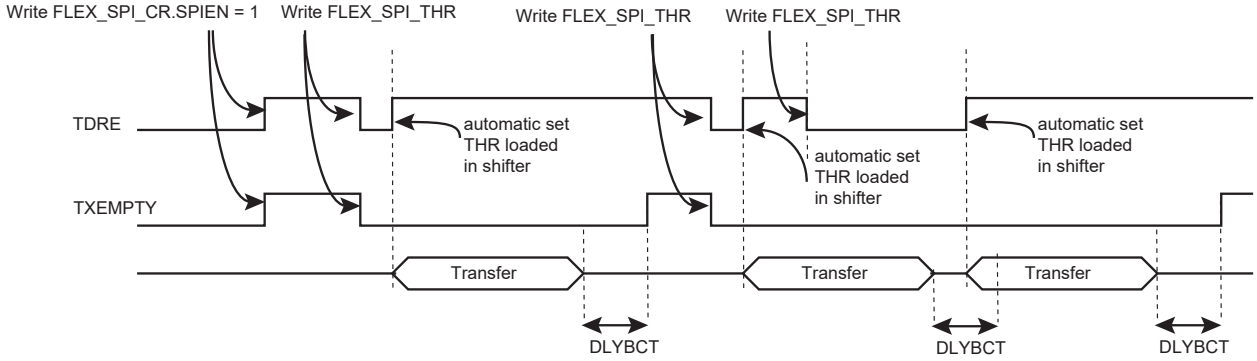
As soon as the FLEX_SPI_TDR is written, the Transmit Data Register Empty (TDRE) flag in FLEX_SPI_SR is cleared. When the data written in FLEX_SPI_TDR is loaded into the shift register, the FLEX_SPI_SR.TDRE flag is set. The TDRE bit is used to trigger the Transmit DMA channel (see figure below).

The end of transfer is indicated by FLEX_SPI_SR.TXEMPTY. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

Note:

1. When the SPI is enabled, the TDRE and TXEMPTY flags are set.
2. The TXEMPTY flag alone cannot be used to detect the end of the buffer DMA transfer.

Figure 38-77. TDRE and TXEMPTY Flag Behavior



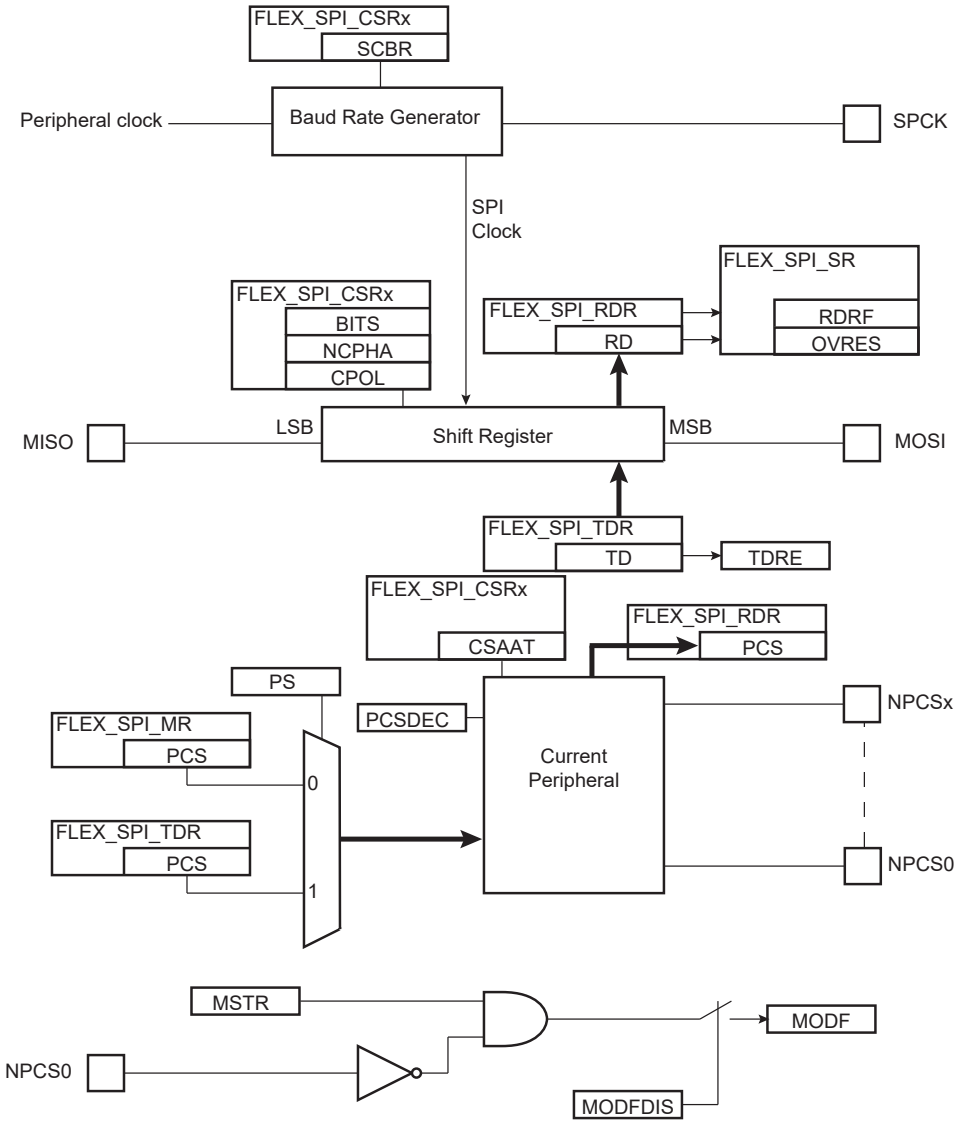
The transfer of received data from the shift register to FLEX_SPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in FLEX_SPI_SR. When the received data are read, the RDRF bit is cleared.

If FLEX_SPI_RDR has not been read before new data are received, the Overrun Error bit (OVRES) in FLEX_SPI_SR is set. As long as this flag is set, data are loaded in FLEX_SPI_RDR. The user has to read the status register to clear the OVRES bit.

The following figures show, respectively, a block diagram of the SPI when operating in Master mode and a flow chart describing how transfers are handled.

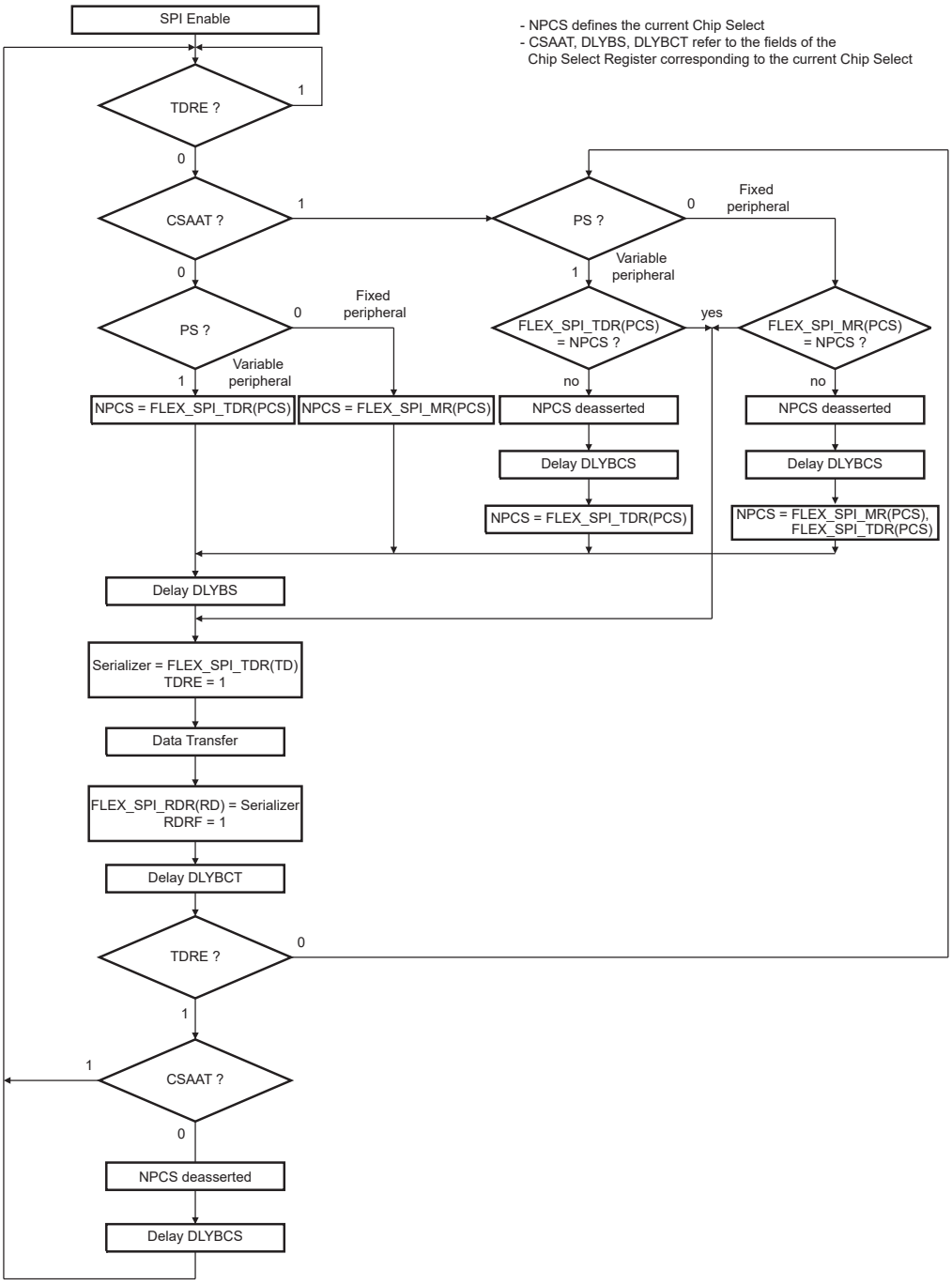
38.8.3.1 Master Mode Block Diagram

Figure 38-78. Master Mode Block Diagram



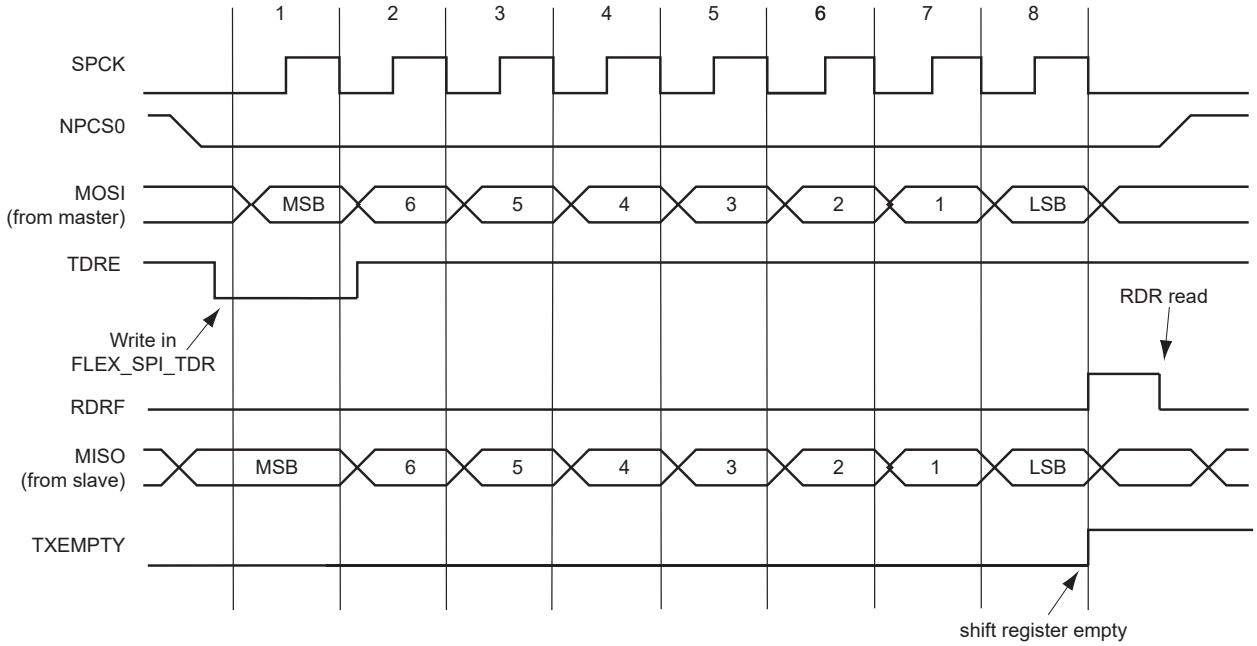
38.8.3.2 Master Mode Flowchart

Figure 38-79. Master Mode



The following figure shows the behavior of Transmit Data Register Empty (TDRE), Receive Data Register (RDRF) and Transmission Register Empty (TXEMPTY) status flags within FLEX_SPI_SR during an 8-bit data transfer in Fixed mode without the DMAC involved.

Figure 38-80. Status Register Flags Behavior



38.8.3.3 Clock Generation

The SPI bit rate clock is generated by dividing a source clock which can be the peripheral clock or a programmable clock from the GCLK. The divider can be a value between 1 and 255.

If the SCBR field is programmed to 1 and the clock source is GCLK, the operating bit rate is peripheral clock (refer to the section “Electrical Characteristics” for the SPCK maximum frequency). Triggering a transfer while SCBR is at 0 can lead to unpredictable results.

At reset, SCBR is 0 and the user has to program it to a valid value before performing the first transfer.

The divisor can be defined independently for each chip select, as it has to be programmed in the FLEX_SPI_CSR.SCBR field. This allows the SPI to automatically adapt the bit rate for each interfaced peripheral without reprogramming.

If GCLK is selected as source clock (FLEX_SPI_MR.BRSRCCLK = 1), the bit rate is independent of the processor/bus clock. Thus, the processor clock can be changed while SPI is enabled. The processor clock frequency changes must be performed only by programming the PMC_MCKR.PRES field (refer to the section “Power Management Controller” (PMC)). Any other method to modify the processor/bus clock frequency (PLL multiplier, etc.) is forbidden when SPI is enabled.

The peripheral clock frequency must be at least three times higher than GCLK.

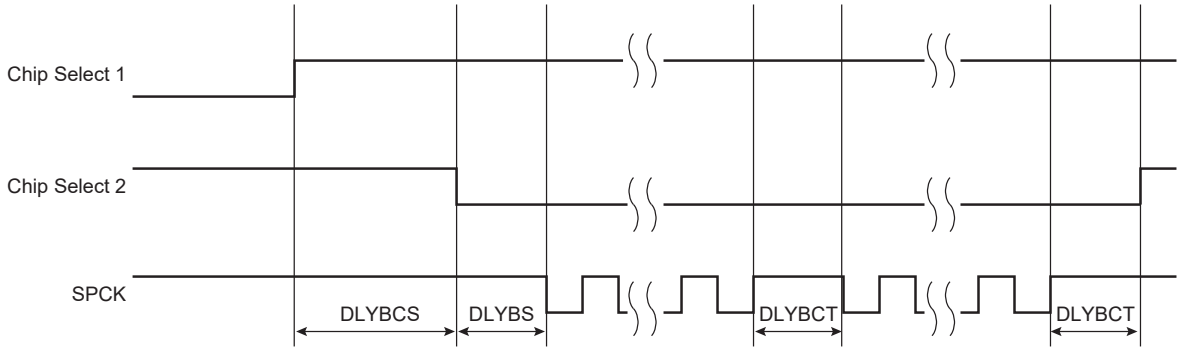
38.8.3.4 Transfer Delays

The figure below shows a chip select transfer change and consecutive transfers on the same chip select. Three delays can be programmed to modify the transfer waveforms:

- The delay between the chip selects. It is programmable only once for all chip selects by writing the FLEX_SPI_MR.DLYBCS field. The SPI slave device deactivation delay is managed through DLYBCS. If there is only one SPI slave device connected to the master, the DLYBCS field does not need to be configured. If several slave devices are connected to a master, DLYBCS must be configured depending on the highest deactivation delay. Refer to the SPI slave device electrical characteristics.
- The delay before SPCK, independently programmable for each chip select by writing the DLYBS field. The SPI slave device activation delay is managed through DLYBS. Refer to the SPI slave device electrical characteristics to define DLYBS.
- The delay between consecutive transfers, independently programmable for each chip select by writing the DLYBCT field. The time required by the SPI slave device to process received data is managed through DLYBCT. This time depends on the SPI slave system activity.

These delays allow the SPI to be adapted to the interfaced peripherals and their speed and bus release time.

Figure 38-81. Programmable Delays



38.8.3.5 Peripheral Selection

The serial peripherals are selected through the assertion of the NPCS0 to NPCS3 signals. By default, all NPCS signals are high before and after each transfer.

- Fixed Peripheral Select Mode: SPI exchanges data with only one peripheral.
  Fixed Peripheral Select mode is enabled by writing the FLEX_SPI_MR.PS bit to zero. In this case, the current peripheral is defined by the FLEX_SPI_MR.PCS field, and the FLEX_SPI_TDR. PCS field has no effect.
- Variable Peripheral Select Mode: Data can be exchanged with more than one peripheral without having to reprogram FLEX_SPI_MR.PCS.
  Variable Peripheral Select Mode is enabled by setting the FLEX_SPI_MR.PS bit to one. The FLEX_SPI_TDR.PCS field is used to select the current peripheral. This means that the peripheral selection can be defined for each new data. The value must be written in a single access to FLEX_SPI_TDR in the following format:

  [xxxxxxx(7-bit) + LASTXFER(1-bit)(1)+ xxxx(4-bit) + PCS (4-bit) + TD (8 to 16-bit data)]
  with LASTXFER at 0 or 1 depending on the CSAAT bit, and PCS equal to the chip select to assert, as defined in section SPI Transmit Data Register (FLEX_SPI_TDR).
  Note: 1. Optional

  The CSAAT, LASTXFER and CSNAAT bits are discussed in section Peripheral Deselection with DMA.

  If LASTXFER is used, the command must be issued after writing the last character. Instead of LASTXFER, the user can use the SPIDIS command. After the end of the DMA transfer, it is necessary to wait for the TXEMPTY flag and then write SPIDIS into the SPI Control Register (FLEX_SPI_CR). This does not change the configuration register values). The NPCS is disabled after the last character transfer. Then, another DMA transfer can be started if the FLEX_SPI_CR.SPIEN bit has previously been written.

38.8.3.6 SPI Direct Access Memory Controller (DMAC)

In both Fixed and Variable modes, the Direct Memory Access Controller (DMAC) can be used to reduce processor overhead.

The fixed peripheral selection allows buffer transfers with a single peripheral. Using the DMAC is an optimal means, as the size of the data transfer between the memory and the SPI is either 8 bits or 16 bits. However, if the peripheral selection is modified, FLEX_SPI_MR must be reprogrammed.

The variable peripheral selection allows buffer transfers with multiple peripherals without reprogramming FLEX_SPI_MR. Data written in FLEX_SPI_TDR is 32 bits wide and defines the real data to be transmitted and the destination peripheral. Using the DMAC in this mode requires 32-bit wide buffers, with the data in the LSBs and the PCS and LASTXFER fields in the MSBs. However, the SPI still controls the number of bits (8 to 16) to be transferred through MISO and MOSI lines with the chip select configuration registers. This is not the optimal means in terms of memory size for the buffers, but it provides a very effective means to exchange data with several peripherals without any intervention of the processor.

38.8.3.7 Peripheral Chip Select Decoding

The user can program the SPI to operate with up to 15 slave peripherals by decoding the four chip select lines, NPCS0 to NPCS3 with an external decoder/demultiplexer (see the following figure). This can be enabled by setting the FLEX_SPI_MR.PCSDEC bit.

When operating without decoding, the SPI makes sure that in any case only one chip select line is activated, i.e., one NPCS line driven low at a time. If two bits are defined low in a PCS field, only the lowest numbered chip select is driven low.

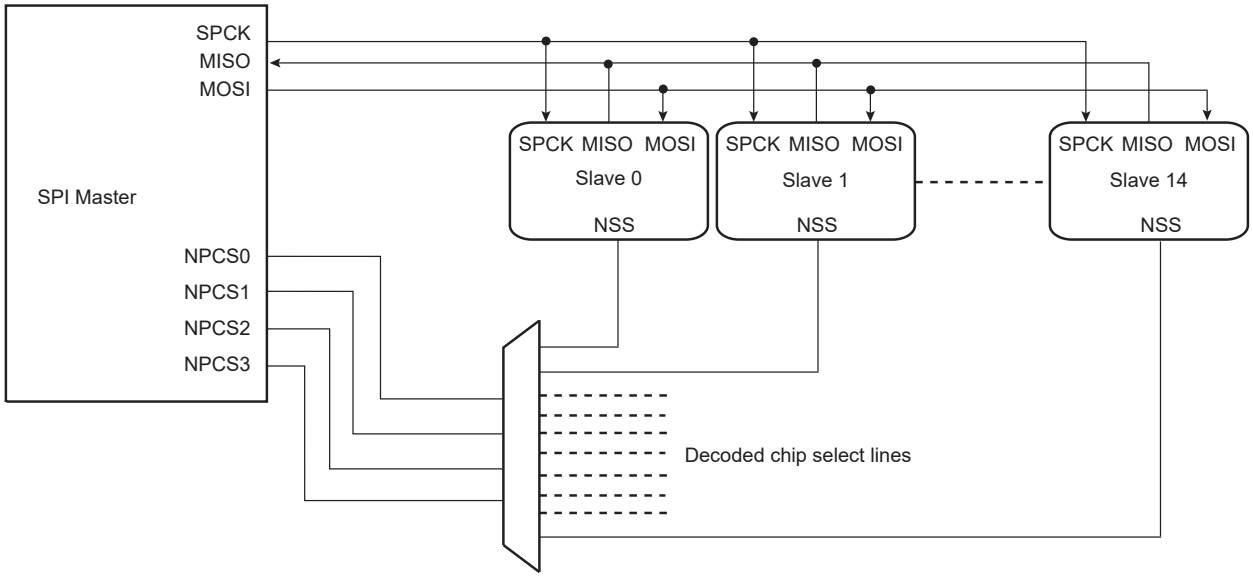
When operating with decoding, the SPI directly outputs the value defined by the PCS field on the NPCS lines of either FLEX_SPI_MR or FLEX_SPI_TDR (depending on PS).

As the SPI sets a default value of 0xF on the chip select lines (i.e., all chip select lines at 1) when not processing any transfer, only 15 peripherals can be decoded.

The SPI has only four Chip Select registers. As a result, when external decoding is activated, each NPCS chip select defines the characteristics of up to four peripherals. As an example, FLEX_SPI_CRS0 defines the characteristics of the externally decoded peripherals 0 to 3, corresponding to the PCS values 0x0 to 0x3. Consequently, the user has to make sure to connect compatible peripherals on the decoded chip select lines 0 to 3, 4 to 7, 8 to 11 and 12 to 14. The following figure shows this type of implementation.

If the CSAAT bit is used, with or without the DMAC, the mode fault detection for NPCS0 line must be disabled. This is not needed for all other chip select lines since mode fault detection is only on NPCS0.

Figure 38-82. Chip Select Decoding Application Block Diagram: Single Master/Multiple Slave Implementation



38.8.3.8 Peripheral Deselection without DMA

During a transfer of more than one data on a Chip Select without the DMA, FLEX_SPI_TDR is loaded by the processor, the TDRE flag rises as soon as the content of FLEX_SPI_TDR is transferred into the internal shift register. When this flag is detected high, FLEX_SPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer, and if the next transfer is performed on the same chip select as the current transfer, the Chip Select is not deasserted between the two transfers. But depending on the application software handling the SPI status register flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload FLEX_SPI_TDR in time to keep the chip select active (low). A null DLYBCT value (delay between consecutive transfers) in FLEX_SPI_CSR, gives even less time for the processor to reload FLEX_SPI_TDR. With some SPI slave peripherals, if the chip select line must remain active (low) during a full set of transfers, communication errors can occur.

To facilitate interfacing with such devices, the Chip Select registers [CSR0...CSR3] can be programmed with the Chip Select Active After Transfer (CSAAT) bit to 1. This allows the chip select lines to remain in their current state (low = active) until a transfer to another chip select is required. Even if FLEX_SPI_TDR is not reloaded, the chip select

remains active. To de-assert the chip select line at the end of the transfer, the Last Transfer (LASTXFER) bit in FLEX_SPI_CR must be set after writing the last data to transmit into FLEX_SPI_TDR.

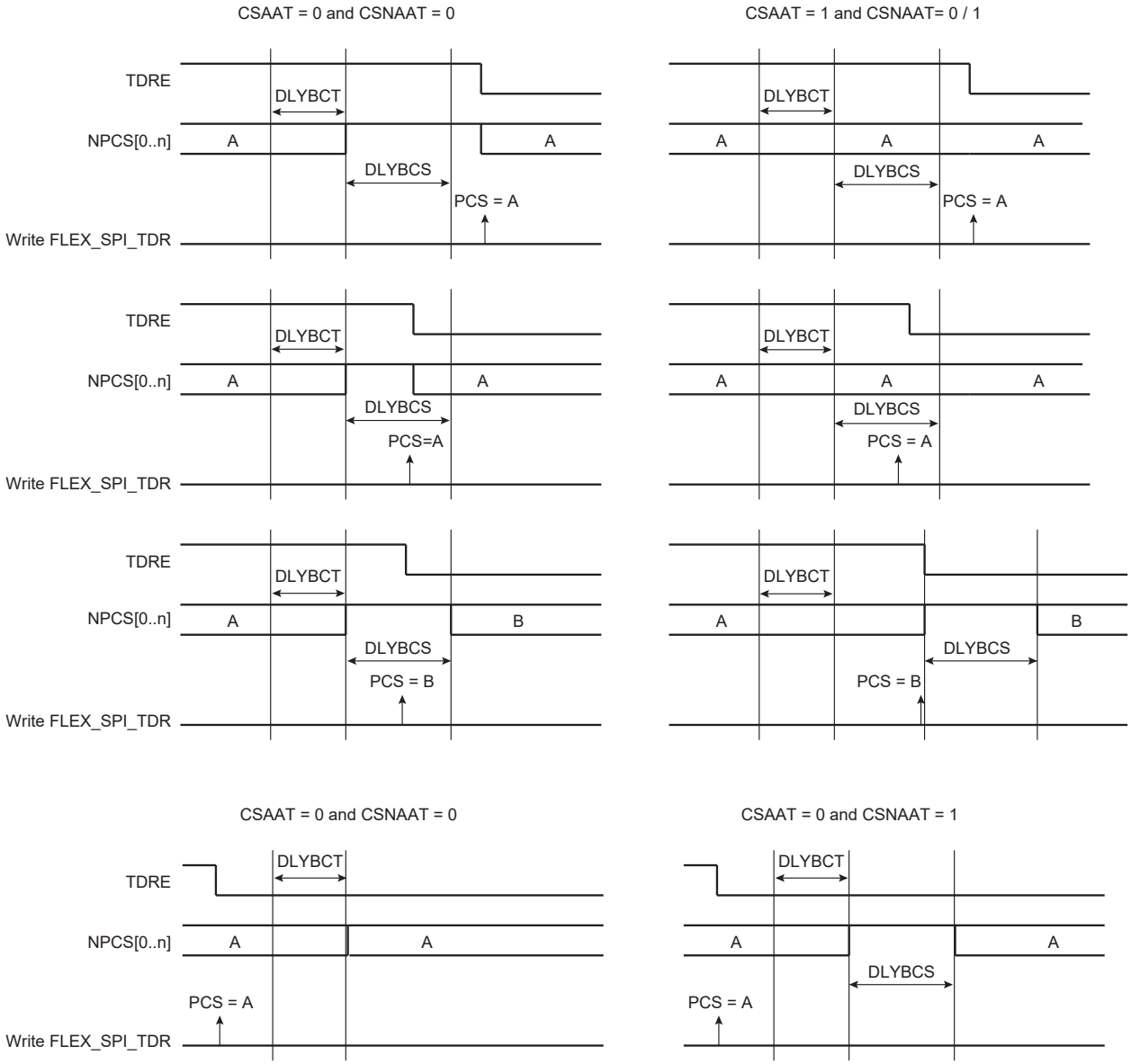
38.8.3.9 Peripheral Deselection with DMA

DMA provides faster reloads of FLEX_SPI_TDR compared to software. However, depending on the system activity, it is not guaranteed that FLEX_SPI_TDR is written with the next data before the end of the current transfer. Consequently, a data can be lost by the deassertion of the NPCS line for SPI slave peripherals requiring the chip select line to remain active between two transfers. The only way to guarantee a safe transfer in this case is the use of the CSAAT and LASTXFER bits.

When the CSAAT bit is cleared, the NPCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a Chip Select, the TDRE flag rises as soon as the content of FLEX_SPI_TDR is transferred into the internal shift register. When this flag is detected, FLEX_SPI_TDR can be reloaded. If this reload occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the Chip Select is not deasserted between the two transfers. This can lead to difficulties to interface with some serial peripherals requiring the chip select to be deasserted after each transfer. To facilitate interfacing with such devices, FLEX_SPI_CSR can be programmed with the Chip Select Not Active After Transfer (CSNAAT) bit to 1. This allows the chip select lines to be deasserted systematically during a time “DLYBCS” (the value of the CSNAAT bit is processed only if the CSAAT bit is cleared for the same chip select).

The following figure shows different peripheral deselection cases and the effect of the CSAAT and CSNAAT bits.

Figure 38-83. Peripheral Deselection



38.8.3.10 Mode Fault Detection

The SPI has the capability to operate in multi-master environment. Consequently, the NPCS0/NSS line must be monitored. If one of the masters on the SPI bus is currently transmitting, the NPCS0/NSS line is low and the SPI must not transmit a data. A mode fault is detected when the SPI is programmed in Master mode and a low level is driven by an external master on the NPCS0/NSS signal. In multi-master environment, NPCS0, MOSI, MISO and SPCK pins must be configured in open drain (through the PIO controller). When a mode fault is detected, the FLEX_SPI_SR.MODF bit is set until FLEX_SPI_SR is read and the SPI is automatically disabled until it is re-enabled by writing the FLEX_SPI_CR.SPIEN bit to 1.

By default, the mode fault detection is enabled. The user can disable it by setting the FLEX_SPI_MR.MODFDIS bit.

38.8.4 SPI Slave Mode

When operating in Slave mode, the SPI processes data bits on the clock provided on the SPI clock pin (SPCK).

The SPI waits until NSS goes active before receiving the serial clock from an external master. When NSS falls, the clock is validated and the data are loaded in FLEX_SPI_RDR according to the configuration value of the FLEX_SPI_CSR0.BITS field. These bits are processed following a phase and a polarity defined respectively by the

FLEX_SPI_CSR0.NCPHA and FLEX_SPI_CSR0.CPOL bits. Note that the BITS field, CPOL bit and NCPHA bit of the other Chip Select registers have no effect when the SPI is programmed in Slave mode.

The bits are shifted out on the MISO line and sampled on the MOSI line.

Note: For more information on the BITS field, see also the note below the FLEX_SPI_CSRx register bitmap in section SPI Chip Select Register

When all bits are processed, the received data are transferred in FLEX_SPI_RDR and the RDRF bit rises. If FLEX_SPI_RDR has not been read before new data are received, the Overrun Error bit (OVRES) in FLEX_SPI_SR is set. As long as this flag is set, data are loaded in FLEX_SPI_RDR. The user must read FLEX_SPI_SR to clear the OVRES bit.

When a transfer starts, the data shifted out is the data present in the shift register. If no data has been written in FLEX_SPI_TDR, the last data received is transferred. If no data has been received since the last reset, all bits are transmitted low, as the shift register resets to 0.

When a first data is written in FLEX_SPI_TDR, it is transferred immediately in the shift register and the TDRE flag rises. If new data is written, it remains in FLEX_SPI_TDR until a transfer occurs, i.e., NSS falls and there is a valid clock on the SPCK pin. When the transfer occurs, the last data written in FLEX_SPI_TDR is transferred in the shift register and the TDRE flag rises. This enables frequent updates of critical variables with single transfers.

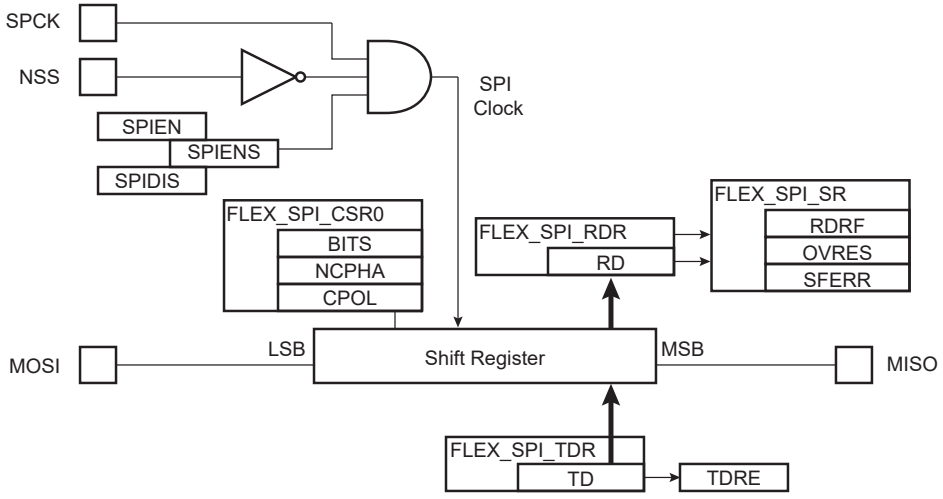
Then, a new data is loaded in the shift register from FLEX_SPI_TDR. If no character is ready to be transmitted, i.e., no character has been written in FLEX_SPI_TDR since the last load from FLEX_SPI_TDR to the shift register, FLEX_SPI_TDR is retransmitted. In this case the Underrun Error Status Flag (UNDES) is set in FLEX_SPI_SR.

If NSS rises between two characters, it must be kept high for two MCK clock periods or more and the next SPCK capture edge must not occur less than four MCK periods after NSS rise.

In slave mode, if the NSS line rises and the received character length does not match the configuration defined in FLEX_SPI_CSR0.BITS, the flag SFERR is set in FLEX_SPI_SR.

The following figure shows a block diagram of the SPI when operating in Slave mode.

Figure 38-84. Slave Mode Functional Block Diagram



38.8.5 SPI Comparison Function on Received Character

The comparison is only relevant for SPI Slave mode (MSTR = 0 in FLEX_US_MR).

In Active mode, the CMP flag in FLEX_SPI_SR is raised. It is set when the received character matches the conditions programmed in the SPI Comparison Register (FLEX_SPI_CMPR). The CMP flag is set as soon as FLEX_SPI_RDR is loaded with the new received character. The CMP flag is cleared by reading FLEX_SPI_SR.

The SPI Comparison Register (FLEX_SPI_CMPR) can be programmed to provide different comparison methods. These are listed below:

- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.

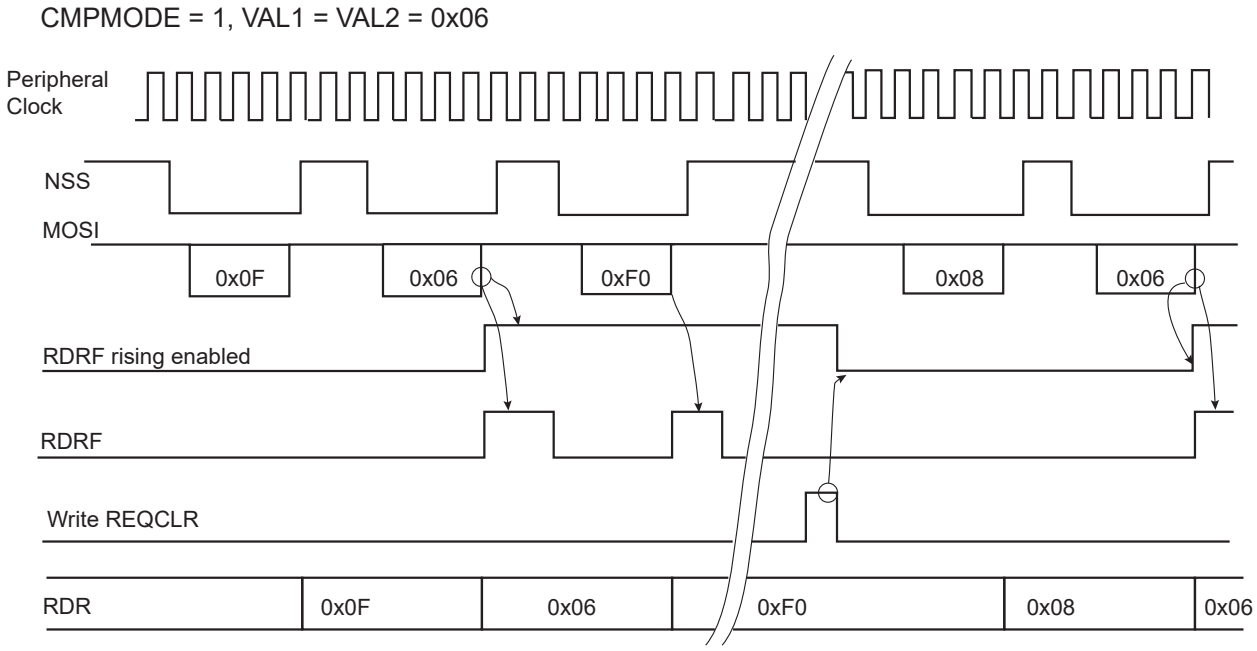
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if any received character equals VAL1 or VAL2.

When FLEX_SPI_MR.CMPMODE is cleared, all received data is loaded in FLEX_SPI_RDR and the CMP flag provides the status of the comparison result.

By setting the CMPMODE bit, the comparison result triggers the start of FLEX_SPI_RDR loading (see figure below). The trigger condition exists as soon as the received character value matches the conditions defined by VAL1 and VAL2 in FLEX_SPI_CMPR. The comparison trigger event is restarted by writing a 1 to the FLEX_SPI_CR.REQCLR bit.

The value programmed in VAL1 and VAL2 fields must not exceed the maximum value of the received character (see BITS field in SPI Chip Select Register (FLEX_SPI_CSR)).

Figure 38-85. Receive Data Register Management



38.8.6 SPI FIFOs

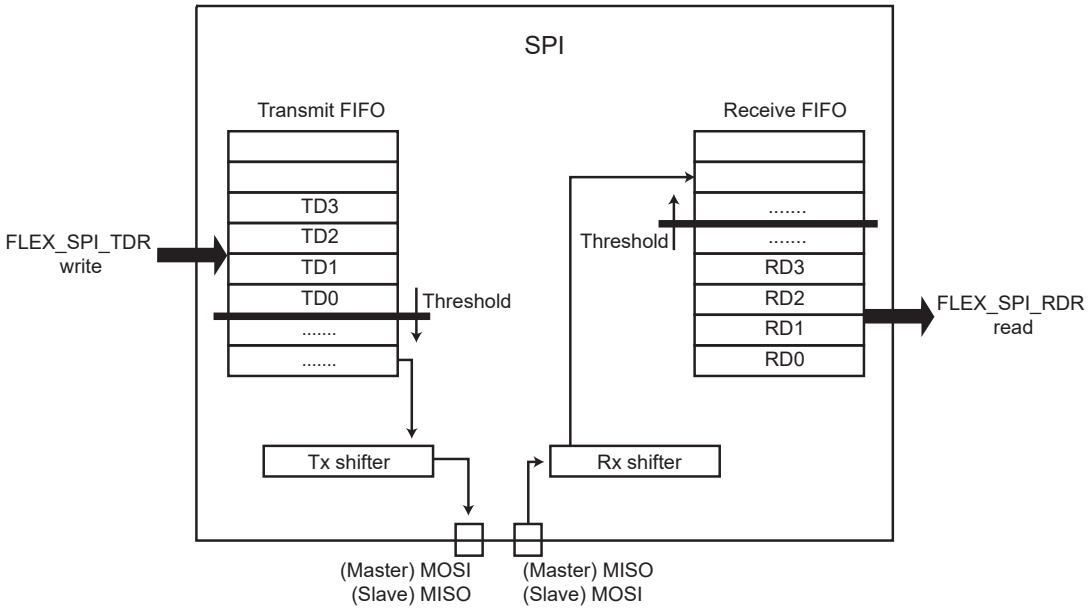
38.8.6.1 Overview

The SPI includes two FIFOs which can be enabled/disabled using the FLEX_SPI_CR.FIFOEN/FIFODIS. It is recommended to disable the SPI module before enabling or disabling the SPI FIFOs (FLEX_SPI_CR.SPIDIS).

Writing FLEX_SPI_CR.FIFOEN to ‘1’ enables a FIFO_DEPTH-data Transmit FIFO and a FIFO_DEPTH-data Receive FIFO.

It is possible to write or to read single or multiple data in the same access to FLEX_SPI_TDR/RDR. See sections SPI Single Data Mode and SPI Multiple Data Mode.

Figure 38-86. FIFOs Block Diagram



38.8.6.2 Sending Data with FIFO Enabled

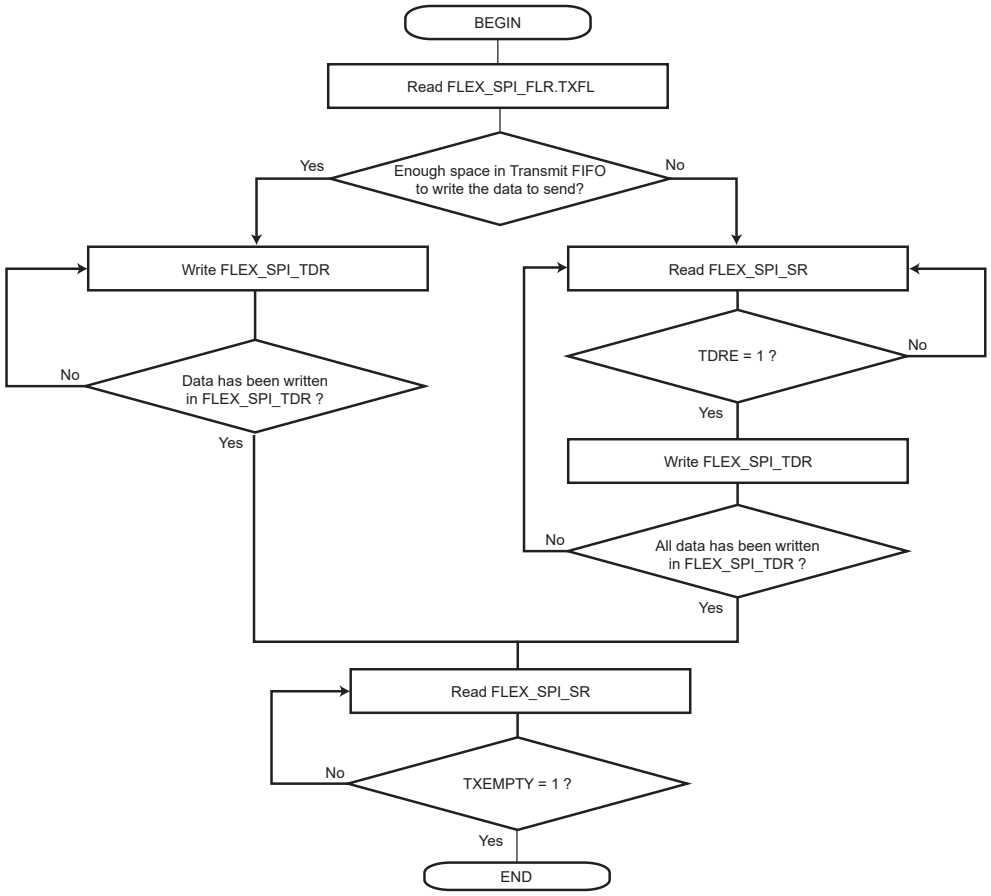
When the Transmit FIFO is enabled, write access to FLEX_SPI_TDR loads the Transmit FIFO.

The FIFO level is provided in FLEX_SPI_FLR.TXFL. If the FIFO can accept the number of data to be transmitted, there is no need to monitor FLEX_SPI_SR.TDRE and the data can be successively written in FLEX_SPI_TDR.

If the FIFO cannot accept the data due to insufficient space, wait for the TDRE flag to be set before writing the data in FLEX_SPI_TDR.

When the space in the FIFO allows only a portion of the data to be written, the TDRE flag must be monitored before writing the remaining data.

Figure 38-87. Sending Data with FIFO Enabled

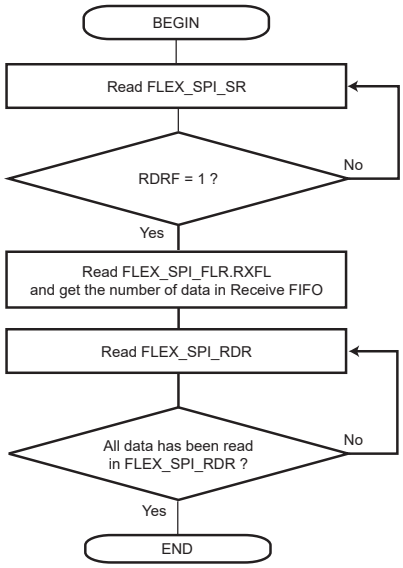


38.8.6.3 Receiving Data with FIFO Enabled

When the Receive FIFO is enabled, FLEX_SPI_RDR access reads the FIFO.

When data are present in the Receive FIFO (RDRF flag set to ‘1’), the exact number of data can be checked with FLEX_SPI_FLR.RXFL. All the data can be read successively in FLEX_SPI_RDR without checking the RDRF flag between each access.

Figure 38-88. Receiving Data with FIFO Enabled



38.8.6.4 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using FLEX_SPI_CR.TXFCLR/RXFCLR.

38.8.6.5 TXEMPTY, TDRE and RDRF Behavior

FLEX_SPI_SR.TXEMPTY, FLEX_SPI_SR.TDRE and FLEX_SPI_SR.RDRF flags display a specific behavior when FIFOs are enabled.

The TXEMPTY flag is cleared as long as there are characters in the Transmit FIFO or in the internal shift register. TXEMPTY is set when there are no characters in the Transmit FIFO and in the internal shift register.

TDRE indicates if a data can be written in the Transmit FIFO. Thus the TDRE flag is set as long as the Transmit FIFO can accept new data. See figure TDRE in Single Data Mode and TXRDYM = 0.

RDRF indicates if an unread data is present in the Receive FIFO. Thus the RDRF flag is set as soon as one unread data is in the Receive FIFO. See figure RDRF in Single Data Mode and RXRDYM = 0.

TDRE and RDRF behavior can be modified using the TXRDYM and RXRDYM fields in the SPI FIFO Mode Register (FLEX_SPI_FMR) to reduce the number of accesses to FLEX_SPI_TDR/RDR. However, for some configurations, the following constraints apply:

- When the Variable Peripheral Select mode is used (FLEX_SPI_MR.PS=1), TXRDYM/RXRDYM must be cleared.
- In Master mode (FLEX_SPI_MR.MSTR=1), RXRDYM must be cleared.

As an example, in Master mode, the Transmit FIFO can be loaded with multiple data in the same access by configuring TXRDYM>0.

See SPI FIFO Mode Register (FLEX_SPI_FMR) for the FIFO configuration.

Figure 38-89. TDRE in Single Data Mode and TXRDYM = 0

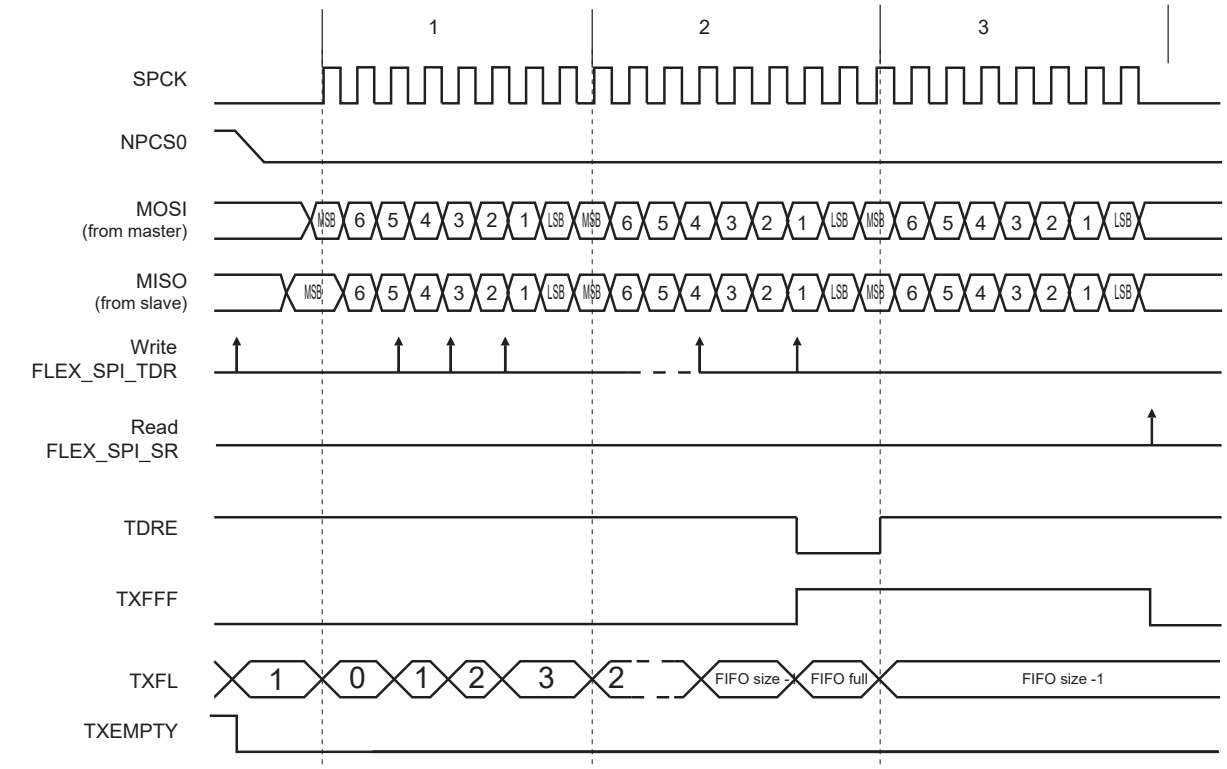
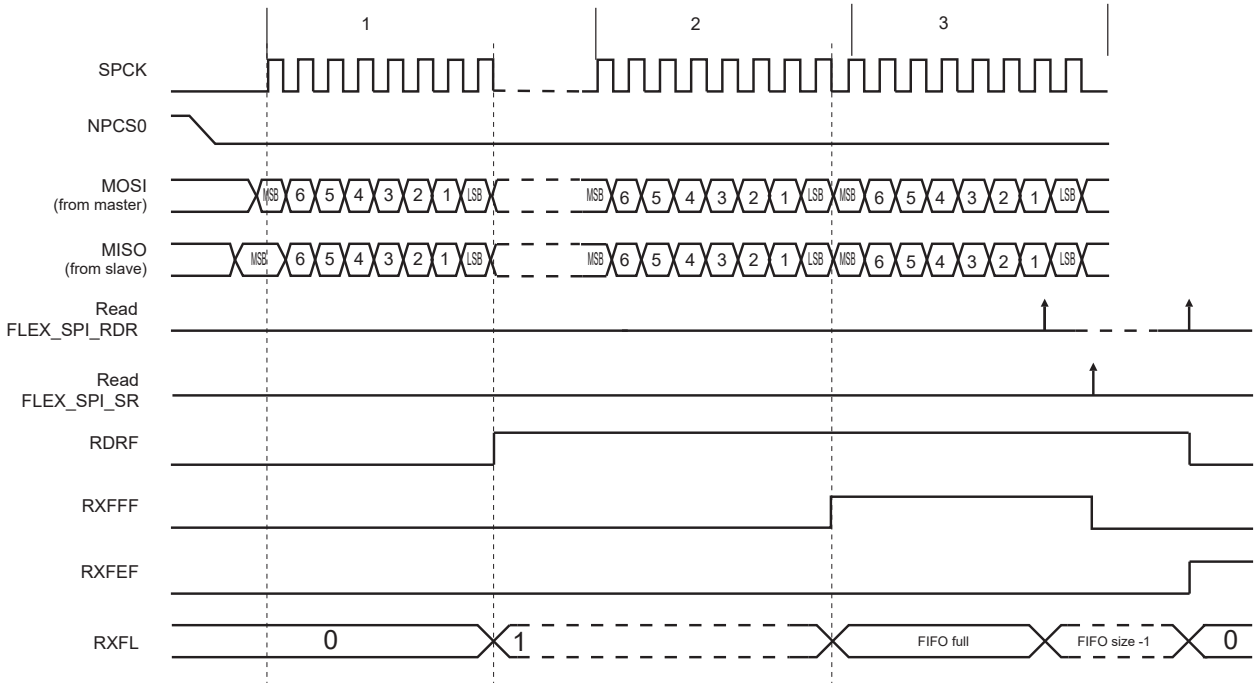


Figure 38-90. RDRF in Single Data Mode and RXRDYM = 0



38.8.6.6 SPI Single Data Mode

In Single Data mode, only one data is written every time FLEX_SPI_TDR is accessed, and only one data is read every time FLEX_SPI_RDR is accessed.

When FLEX_SPI_FMR.TXRDYM = 0, the Transmit FIFO operates in Single Data mode.

When FLEX_SPI_FMR.RXRDYM = 0, the Receive FIFO operates in Single Data mode.

If Master mode is used (FLEX_SPI_MR.MSTR=1), the Receive FIFO must operate in Single Data mode.

If Variable Peripheral Select mode is used (FLEX_SPI_MR.PS=1), the Transmit FIFO must operate in Single Data mode.

See sections SPI Transmit Data Register and SPI Receive Data Register.

38.8.6.6.1 DMAC

When FIFOs operate in Single Data mode, the DMAC transfer type must be configured either in bytes, halfwords or words depending on FLEX_SPI_MR.PS bit value and FLEX_SPI_CSRx.BITS field value.

The same applies when FIFOs are disabled.

38.8.6.7 SPI Multiple Data Mode

Multiple Data mode minimizes the number of accesses by concatenating the data to send/read in one access.

When FLEX_SPI_FMR.TXRDYM > 0, the Transmit FIFO operates in Multiple Data mode.

When FLEX_SPI_FMR.RXRDYM > 0, the Receive FIFO operates in Multiple Data mode.

Multiple data can be read from the Receive FIFO only in Slave mode (FLEX_SPI_MR.MSTR=0).

The Transmit FIFO can be loaded with multiple data in the same access by configuring TXRDYM>0 and when FLEX_SPI_MR.PS=0.

In Multiple Data mode, up to two data can be written in one FLEX_SPI_TDR write access. It is also possible to read up to four data in one FLEX_SPI_RDR access if FLEX_SPI_CSRx.BITS is configured to ‘0’ (8-bit data size) and up to two data if FLEX_SPI_CSRx.BITS is configured to a value other than ‘0’ (more than 8-bit data size).

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read. If the access is a halfword size register access, then up to two data are read and only one data is written. Lastly, if the access is a word-size register access, then up to four data are read and up to two data are written.

Written/read data are always right-aligned, as described in sections SPI Receive Data Register (FIFO Multiple Data, 8-bit), SPI Receive Data Register (FIFO Multiple Data, 16-bit) and SPI Transmit Data Register (FIFO Multiple Data, 8- to 16-bit).

As an example, if the Transmit FIFO is empty and there are six data to send, either of the following write accesses may be performed:

- six FLEX_SPI_TDR-byte write accesses
- three FLEX_SPI_TDR-halfword write accesses

With a Receive FIFO containing six data, any of the following read accesses may be performed:

- six FLEX_SPI_RDR-byte read accesses
- three FLEX_SPI_RDR-halfword read accesses
- one FLEX_SPI_RDR-word read access and one FLEX_SPI_RDR-halfword read access

38.8.6.7.1 TDRE and RDRF Configuration

In Multiple Data mode, it is possible to write one or more data in the same FLEX_SPI_TDR/RDR access. The TDRE flag indicates if one or more data can be written in the FIFO depending on the configuration of FLEX_SPI_FMR.TXRDYM/RXRDYM.

As an example, if two data are written each time in FLEX_SPI_TDR, it is useful to configure the TXRDYM field to the value ‘1’ so that the TDRE flag is at ‘1’ only when at least two data can be written in the Transmit FIFO.

Similarly, if four data are read each time in FLEX_SPI_RDR, it is useful to configure the RXRDYM field to the value ‘2’ so that the RDRF flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

38.8.6.7.2 DMAC

It is mandatory to configure DMAC channel size (byte, halfword or word) according to FLEX_SPI_FMR.TXRDYM/RXRDYM configuration. See section SPI Multiple Data Mode for constraints.

38.8.6.8 FIFO Pointer Error

A FIFO overflow is reported in FLEX_SPI_SR.

If the Transmit FIFO is full and a write access is performed on FLEX_SPI_TDR, it generates a Transmit FIFO pointer error and sets FLEX_SPI_SR.TXFPTEF.

In Multiple Data mode, if the number of data written in FLEX_SPI_TDR (according to the register access size) is greater than the free space in the Transmit FIFO, a Transmit FIFO pointer error is generated and FLEX_SPI_SR.TXFPTEF is set.

A FIFO underflow is reported in FLEX_SPI_SR.

In Multiple Data mode, if the number of data read in FLEX_SPI_RDR (according to the register access size) is greater than the number of unread data in the Receive FIFO, a Receive FIFO pointer error is generated and FLEX_SPI_SR.RXFPTEF is set.

No pointer error occurs if the FIFO state/level is checked before writing/reading in FLEX_SPI_TDR/SPI_RDR. The FIFO state/level can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags may not behave as expected; their states should be ignored.

If a pointer error occurs, a software reset must be performed using FLEX_SPI_CR.SWRST (configuration will be lost).

38.8.6.9 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

The Transmit FIFO threshold can be set using the field FLEX_SPI_FMR.TXFTHRES. Each time the Transmit FIFO level goes from ‘above threshold’ to ‘equal to or below threshold’, the flag FLEX_SPI_SR.TXFTHF is set. The application is warned that the Transmit FIFO has reached the defined threshold and that it can be reloaded.

The Receive FIFO threshold can be set using the field FLEX_SPI_FMR.RXFTHRES. Each time the Receive FIFO level goes from ‘below threshold’ to ‘equal to or above threshold’, the flag FLEX_SPI_SR.RXFTHF is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The TXFTHF and RXFTHF flags can be configured to generate an interrupt using FLEX_SPI_IER and FLEX_SPI_IDR.

38.8.6.10 FIFO Flags

FIFOs come with a set of flags which can be configured to generate interrupts through FLEX_SPI_IER and FLEX_SPI_IDR.

FIFO flags state can be read in FLEX_SPI_SR. They are cleared when FLEX_SPI_SR is read.

38.8.7 SPI Register Write Protection

The FLEXCOM operating mode (FLEX_MR.OPMODE) must be set to FLEX_MR_OPMODE_SPI to enable access to the write protection registers.

To prevent any single software error from corrupting SPI behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable), WPITEN (Write Protection Interrupt Enable), and/or WPCREN (Write Protection Control Enable) bits in the SPI Write Protection Mode Register (FLEX_SPI_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the SPI Write Protection Status Register (FLEX_SPI_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_SPI_WPSR.

The following registers can be write-protected when WPEN is set:

- SPI Mode Register
- SPI Chip Select Register
- SPI Comparison Register

The following registers can be write-protected when WPITEN is set:

- SPI Interrupt Enable Register
- SPI Interrupt Disable Register

The following register can be write-protected when WPCREN is set:

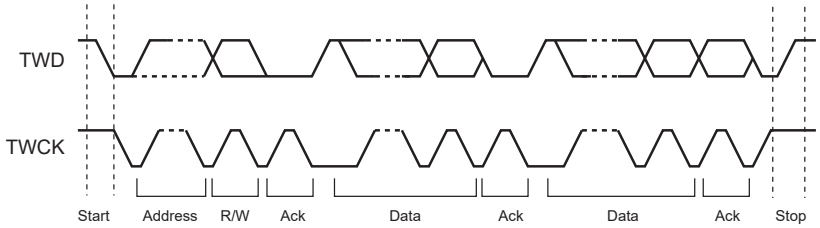
- SPI Control Register

38.9 TWI Functional Description

38.9.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data are transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited (see figure below).

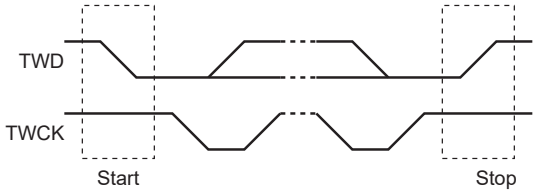
Figure 38-91. Transfer Format



Each transfer begins with a START condition and terminates with a STOP condition (see figure below).

- A high-to-low transition on the TWD line while TWCK is high defines the START condition.
- A low-to-high transition on the TWD line while TWCK is high defines a STOP condition.

Figure 38-92. START and STOP Conditions



38.9.2 Modes of Operation

The TWI has different modes of operation:

- Master Transmitter mode (Standard and Fast modes only)
- Master Receiver mode (Standard and Fast modes only)
- Multi-master Transmitter mode (Standard and Fast modes only)
- Multi-master Receiver mode (Standard and Fast modes only)
- Slave Transmitter mode (Standard, Fast and High-speed modes)
- Slave Receiver mode (Standard, Fast and High-speed modes)

These modes are described in the following sections.

38.9.3 Master Mode

38.9.3.1 Definition

The master is the device that starts a transfer, generates a clock and stops it. This operating mode is not available if High-speed mode is selected.

38.9.3.2 Programming Master Mode

The following fields must be programmed before entering Master mode:

1. DADR (+ IADRSZ + IADR if a 10-bit device is addressed): The device address is used to access slave devices in Read or Write mode.

2. CWGR + CKDIV + CHDIV + CLDIV: Clock waveform.
3. SVDIS: Disables Slave mode.
4. MSEN: Enables Master mode.
   Note: If the TWI is already in Master mode, the device address (DADR) can be configured without disabling the Master mode.

38.9.3.3 Transfer Speed/Bit Rate

The TWI speed is defined in FLEX_TWI_CWGR. The TWI bit rate can be based either on the peripheral clock if the BRSRCCLK bit value is 0 or on a programmable clock source provided by the GCLK if the BRSRCCLK bit value is 1.

If BRSRCCLK = 1, the bit rate is independent of the processor/peripheral clock and thus processor/peripheral clock frequency can be changed without affecting the TWI transfer rate.

The GCLK frequency must be at least three times lower than the peripheral clock frequency.

38.9.3.4 Master Transmitter Mode

This operating mode is not available if High-speed mode is selected.

After the master initiates a START condition when writing into the Transmit Holding register FLEX_TWI_THR, it sends a 7-bit slave address, configured in the Master Mode Register (DADR in FLEX_TWI_MMR), to notify the slave device. The bit following the slave address indicates the transfer direction, 0 in this case (FLEX_TWI_MMR.MREAD = 0).

The TWI transfers require the slave to acknowledge each received byte. During the acknowledge clock pulse (ninth pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. If the slave does not acknowledge the byte, then the Not Acknowledge flag (NACK) is set in the TWI Status Register (FLEX_TWI_SR) of the master and a STOP condition is sent. Alternatively, if the FLEX_TWI_MMR.NOAP bit is set, no stop condition will be sent and a START or STOP condition must be triggered manually through the FLEX_TWI_CR.START or FLEX_TWI_CR.STOP bit once the software is ready for the transmission of the condition. The NACK flag must be cleared by reading the TWI Status Register (FLEX_TWI_SR) before the next write into the TWI Transmit Holding Register (FLEX_TWI_THR). As with the other status bits, an interrupt can be generated if enabled in the interrupt enable Register (FLEX_TWI_IER). If the slave acknowledges the byte, the data written in FLEX_TWI_THR is then shifted in the internal shifter and transferred. When an acknowledge is detected, the TXRDY bit is set until a new write in FLEX_TWI_THR.

TXRDY is used as transmit ready for the DMA transmit channel.

Note: To clear the TXRDY flag in Master mode, write the FLEX_TWI_CR.MSDIS bit to 1, then write the FLEX_TWI_CR.MSEN bit to 1.

While no new data is written in FLEX_TWI_THR, the serial clock line is tied low. When new data is written in FLEX_TWI_THR, the SCL is released and the data is sent. To generate a STOP event, the STOP command must be performed by writing in the STOP field of the TWI Control Register (FLEX_TWI_CR).

After a master write transfer, the Serial Clock line is stretched (tied low) while no new data is written in FLEX_TWI_THR or until a STOP command is performed.

See the following figures.

Figure 38-93. Master Write with One Data Byte

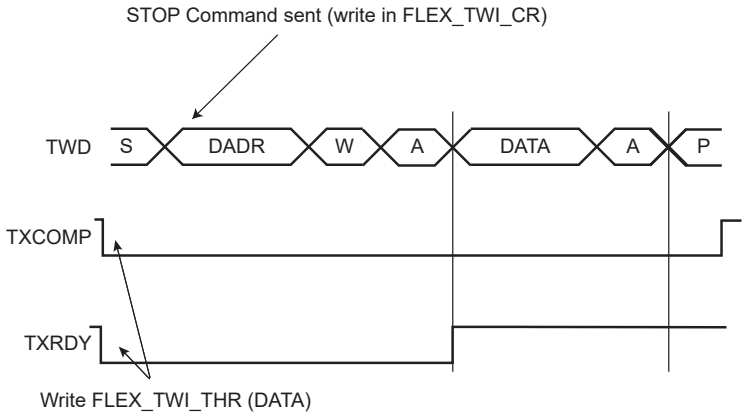


Figure 38-94. Master Write with Multiple Data Bytes

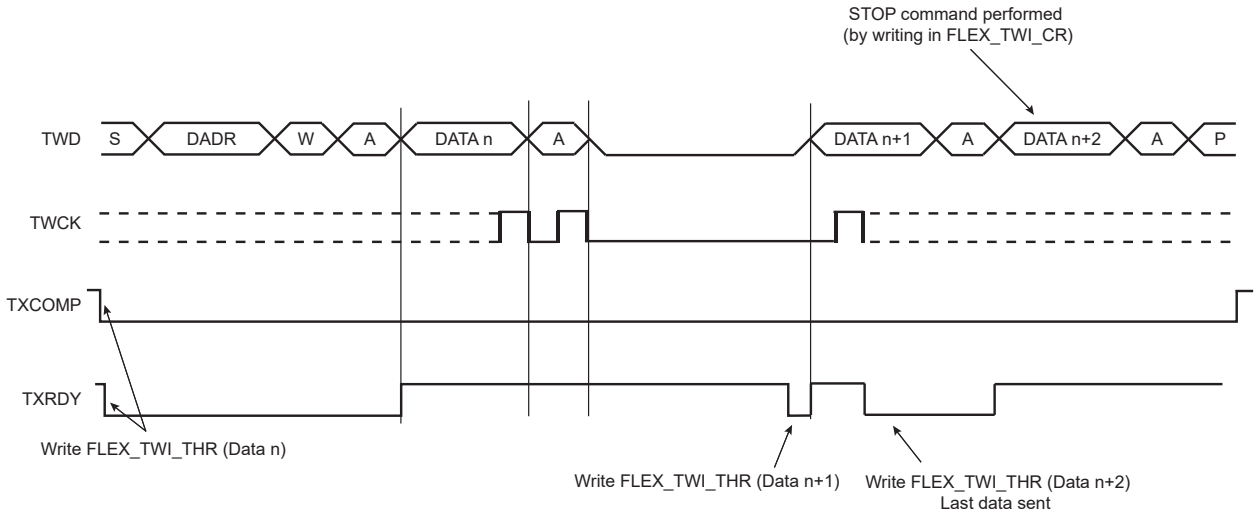
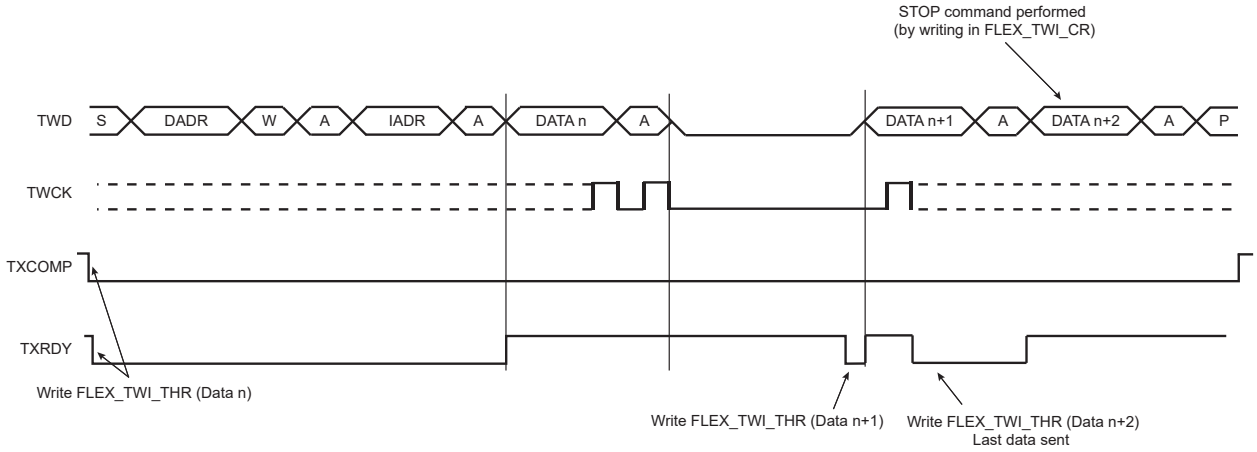


Figure 38-95. Master Write with One Byte Internal Address and Multiple Data Bytes



38.9.3.5 Master Receiver Mode

Master Receiver mode is not available if High-speed mode is selected.

The read sequence begins by setting the START bit. After the start condition has been sent, the master sends a 7-bit slave address to notify the slave device. The bit following the slave address indicates the transfer direction, 1 in this case (FLEX_TWI_MMR.MREAD = 1). During the acknowledge clock pulse (9th pulse), the master releases the data

line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. The master polls the data line during this clock pulse and sets the FLEX_TWI_SR.NACK bit if the slave does not acknowledge the byte.

If an acknowledge is received, the master is then ready to receive data from the slave. After data has been received, the master sends an acknowledge condition to notify the slave that the data has been received except for the last data (see figure "Master Read with One Data Byte" below). When the FLEX_TWI_SR.RXRDY bit is set, a character has been received in the Receive Holding Register (FLEX_TWI_RHR). The RXRDY bit is reset when reading FLEX_TWI_RHR.

When a single data byte read is performed, with or without internal address (IADR), the START and STOP bits must be set at the same time. See figure "Master Read with One Data Byte" below. When a multiple data byte read is performed, with or without internal address (IADR), the STOP bit must be set after the next-to-last data received (same condition applies for START bit to generate a repeated start). See figure "Master Read with Multiple Data Bytes" below. For internal address usage, see section Internal Address.

If FLEX_TWI_RHR is full (RXRDY high) and the master is receiving data, the serial clock line will be tied low before receiving the last bit of the data and until FLEX_TWI_RHR is read. Once FLEX_TWI_RHR is read, the master will stop stretching the serial clock line and end the data reception. See figure "Master Read Clock Stretching with Multiple Data Bytes" below.

⚠WARNING When receiving multiple bytes in Master Read mode, if the next-to-last access is not read (the RXRDY flag remains high), the last access will not be completed until FLEX_TWI_RHR is read. The last access stops on the next-to-last bit (clock stretching). When FLEX_TWI_RHR is read there is only half a bit period to send the STOP bit (or START bit) command, else another read access might occur (spurious access).

A possible workaround is to set the STOP bit (or START bit) before reading FLEX_TWI_RHR on the next-to-last access (within IT handler).

Figure 38-96. Master Read with One Data Byte

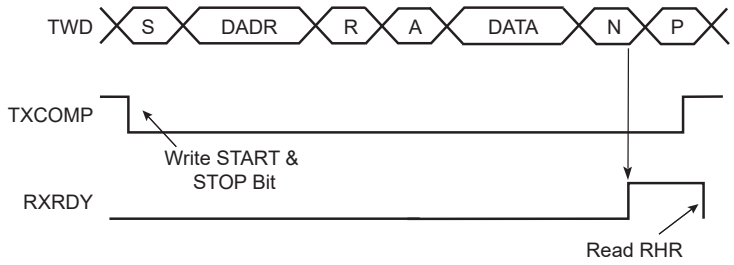


Figure 38-97. Master Read with Multiple Data Bytes

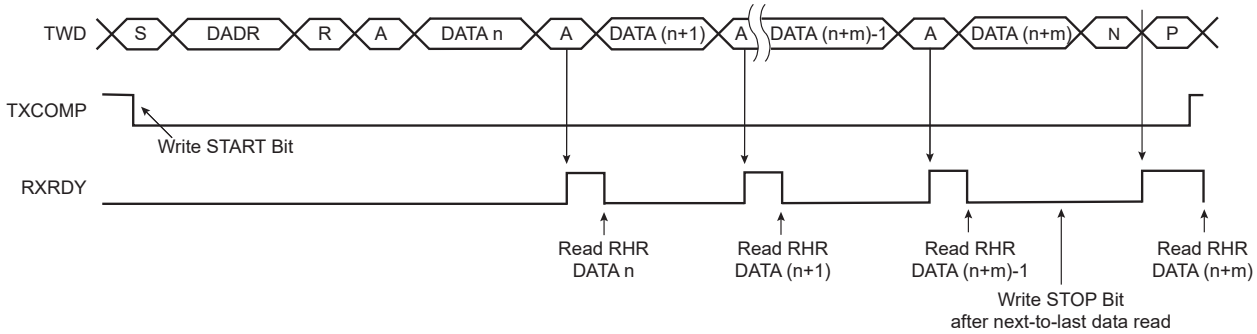
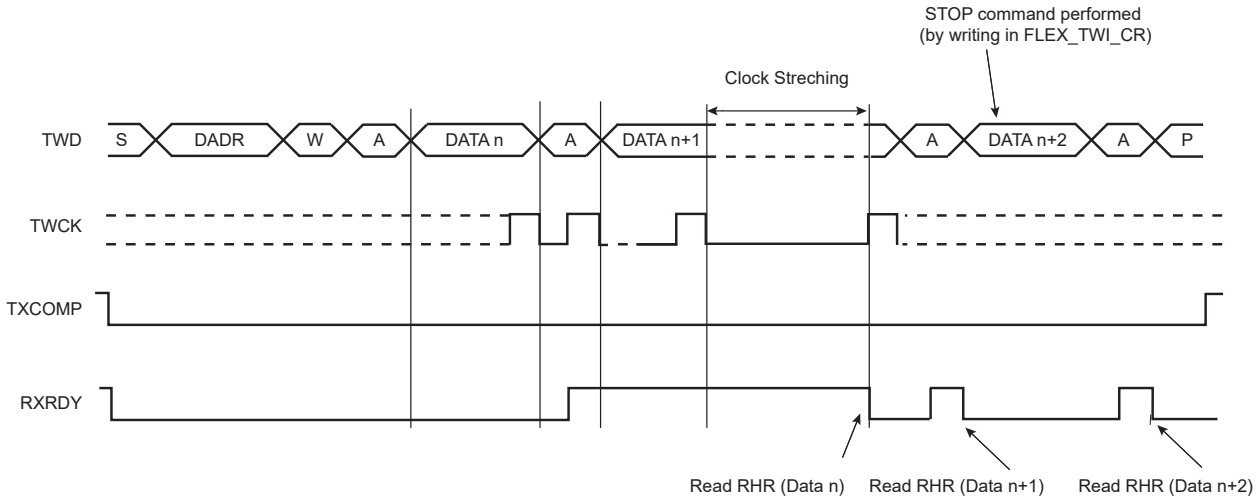


Figure 38-98. Master Read Clock Stretching with Multiple Data Bytes



RXRDY is used as receive ready trigger event for the DMA receive channel.

38.9.3.6 Internal Address

The TWI interface can perform transfers with 7-bit slave address devices and with 10-bit slave address devices.

38.9.3.6.1 7-bit Slave Addressing

When addressing 7-bit slave devices, the internal address bytes are used to perform random address (read or write) accesses to reach one or more data bytes, e.g., within a memory page location in a serial memory. When performing read operations with an internal address, the TWI performs a write operation to set the internal address into the slave device, and then switch to Master Receiver mode. Note that the second start condition (after sending the IADR) is sometimes called “repeated start” (Sr) in I^2C fully-compatible devices. See figure Master Read with One, Two or Three Bytes Internal Address and One Data Byte.

See figures Master Write with One, Two or Three Bytes Internal Address and One Data Byte and Internal Address Usage for the master write operation with internal address.

The three internal address bytes are configurable through the Master Mode Register (FLEX_TWI_MMR).

If the slave device supports only a 7-bit address, i.e., no internal address, IADRSZ must be configured to 0.

The abbreviations listed below are used in the following figures:

S	Start
Sr	Repeated Start
P	Stop
W	Write
R	Read
A	Acknowledge
N	Not Acknowledge
DADR	Device Address
IADR	Internal Address

Figure 38-99. Master Write with One, Two or Three Bytes Internal Address and One Data Byte

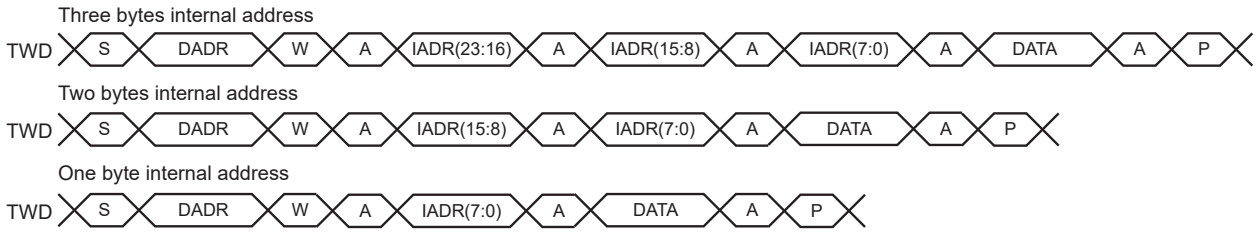
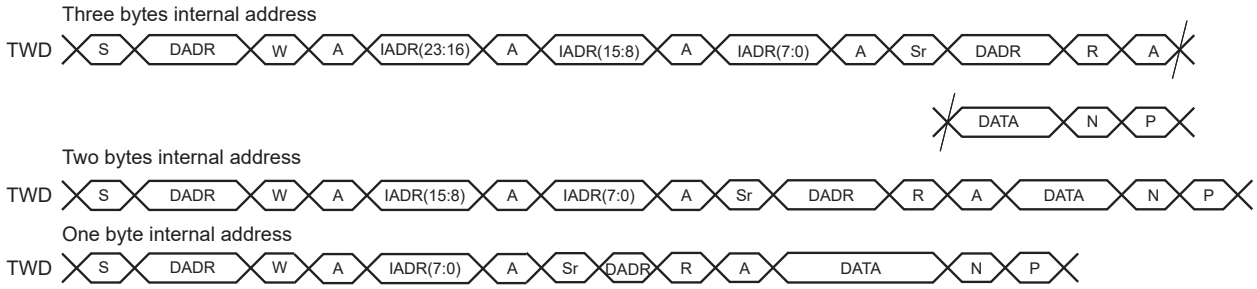


Figure 38-100. Master Read with One, Two or Three Bytes Internal Address and One Data Byte



38.9.3.6.2 10-bit Slave Addressing

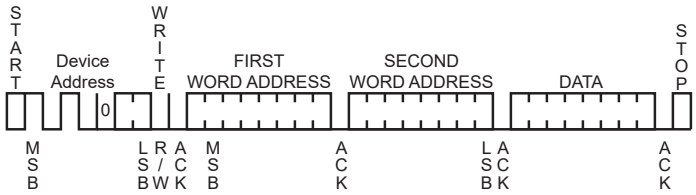
For a slave address higher than seven bits, the user must configure the address size (IADRSZ) and set the other slave address bits in the Internal Address Register (FLEX_TWI_IADR). The two remaining internal address bytes, IADR[15:8] and IADR[23:16], can be used the same way as in 7-bit slave addressing.

Example: Address a 10-bit device (10-bit device address is b1 b2 b3 b4 b5 b6 b7 b8 b9 b10)

1. Program IADRSZ = 1
2. Program DADR with 1 1 1 1 0 b1 b2 (b1 is the MSB of the 10-bit address, b2, etc.)
3. Program FLEX_TWI_IADR with b3 b4 b5 b6 b7 b8 b9 b10 (b10 is the LSB of the 10-bit address)

The following figure shows a byte write to an Atmel AT24LC512 EEPROM. This demonstrates the use of internal addresses to access the device.

Figure 38-101. Internal Address Usage



38.9.3.7 Repeated Start

In addition to Internal Address mode, repeated start (Sr) can be generated manually by writing the START bit at the end of a transfer instead of the STOP bit. In such case the parameters of the next transfer (direction, SADR, etc.) will need to be set before writing the START bit at the end of the previous transfer.

See section Read/Write Flowcharts.

38.9.3.8 Bus Clear Command

The TWI interface can perform a Bus Clear Command:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Start the transfer by setting the FLEX_TWI_CR.CLEAR bit.

Note: If an alternative command is used (ACMEN bit = 1), the DATAL field must be cleared.

38.9.3.9 SMBus Mode

SMBus mode is enabled when the FLEX_TWI_CR.SMBEN bit is written to one. SMBus mode operation is similar to I²C operation with the following exceptions:

1. Only 7-bit addressing can be used.
2. The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into FLEX_TWI_SMBTR.
3. Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
4. A set of addresses has been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring FLEX_TWI_CR appropriately.

38.9.3.9.1 Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing the FLEX_TWI_CR.PECEN bit to one enables automatic PEC handling in the current transfer. Transfers with and without PEC can freely be intermixed in the same system, since some slaves may not support PEC. The PEC LFSR is always updated on every bit transmitted or received, so that PEC handling on combined transfers will be correct.

In Master Transmitter mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave will compare it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave will return an ACK to the master. If the PEC values differ, data was corrupted, and the slave will return a NACK value. Some slaves may not be able to check the received PEC in time to return a NACK if an error occurred. In this case, the slave should always return an ACK after the PEC byte, and some other mechanism must be implemented to verify that the transmission was received correctly.

In Master Receiver mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master will compare it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the FLEX_TWI_SR.PECERR bit is set. In Master Receiver mode, the PEC byte is always followed by a NACK transmitted by the master, since it is the last byte in the transfer.

In combined transfers, the PECRQ bit should only be set in the last of the combined transfers. If Alternative Command mode is enabled, only the NPEC bit should be set.

Consider the following transfer:

S, ADR+W, COMMAND_BYTE, ACK, SR, ADR+R, DATA_BYTE, ACK, PEC_BYTE, NACK, P

See section Read/Write Flowcharts for detailed flowcharts.

38.9.3.9.2 Timeouts

The FLEX_TWI_SMBTR.TLOWS/TLOWM fields configure the SMBus timeout values. If a timeout occurs, the master transmits a STOP condition and leaves the bus. Furthermore, the FLEX_TWI_SR.TOUT bit is set.

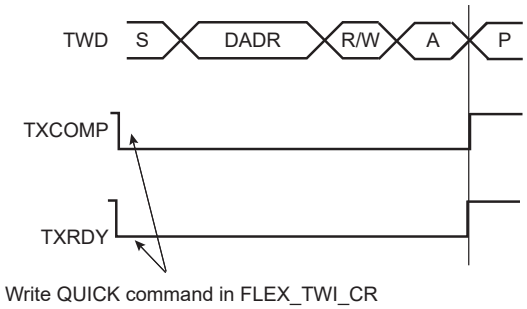
38.9.3.10 SMBus Quick Command (Master Mode Only)

The TWI interface can perform a quick command:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Write the FLEX_TWI_MMR.MREAD bit at the value of the one-bit command to be sent.
3. Start the transfer by setting the FLEX_TWI_CR.QUICK bit.

Note: If an alternative command is used (ACMEN bit = 1), the DATAL field must be cleared.

Figure 38-102. SMBus Quick Command



38.9.3.11 Alternative Command

Another way to configure the transfer is to enable the Alternative Command mode with the ACMEN bit of the TWI Control Register.

In this mode, the transfer is configured through the TWI Alternative Command Register. It is possible to define a simple read or write transfer or a combined transfer with a repeated start.

In order to set a simple transfer, the DATAL field and the DIR field of the TWI Alternative Command Register must be filled accordingly and the NDATAL field must be cleared. To begin the transfer, either set the START bit in the TWI Control Register in case of a read transfer, or write the TWI Transmit Holding Register in case of a write transfer.

For a combined transfer linked by a repeated start, the NDATAL field must be filled with the length of the second transfer and NDIR with the corresponding direction.

The PEC and NPEC bits are used to set a PEC field. In the case of a single transfer with PEC, the PEC bit must be set. In the case of a combined transfer, the NPEC bit must be set.

Note: If the Alternative Command mode is used, the TWIHS_MMR.IADRSZ field must be set to 0.

See Read/Write Flowcharts for detailed flowcharts.

38.9.3.12 Handling Errors in Alternative Command

In case of NACK generated by a slave device or SMBus timeout error, the TWI stops immediately the frame, but the DMA transfer may still be active. To prevent a new frame to be restarted with the remaining DMA data (transmit), the TWI prevents any start of frame until the FLEX_TWI_SR.LOCK flag is cleared.

The FLEX_TWI_SR.LOCK bit indicates the state of the TWI (locked or not locked).

When the TWI is locked, no transfer can begin until the LOCK is cleared using the FLEX_TWI_CR.LOCKCLR bit and until the error flags are cleared reading FLEX_TWI_SR.

In case of error, FLEX_TWI_THR may have been loaded with a new data. The FLEX_TWI_CR.THRCLR bit can be used to flush FLEX_TWI_THR. If the THRCLR bit is set, the TXRDY and TXCOMP flags are set.

38.9.3.13 Read/Write Flowcharts

The flowcharts shown in this section provide examples for read and write operations. A polling or interrupt method can be used to check the status bits. The interrupt method requires that the Interrupt Enable Register (FLEX_TWI_IER) be configured first.

Figure 38-103. TWI Write Operation with Single Data Byte without Internal Address

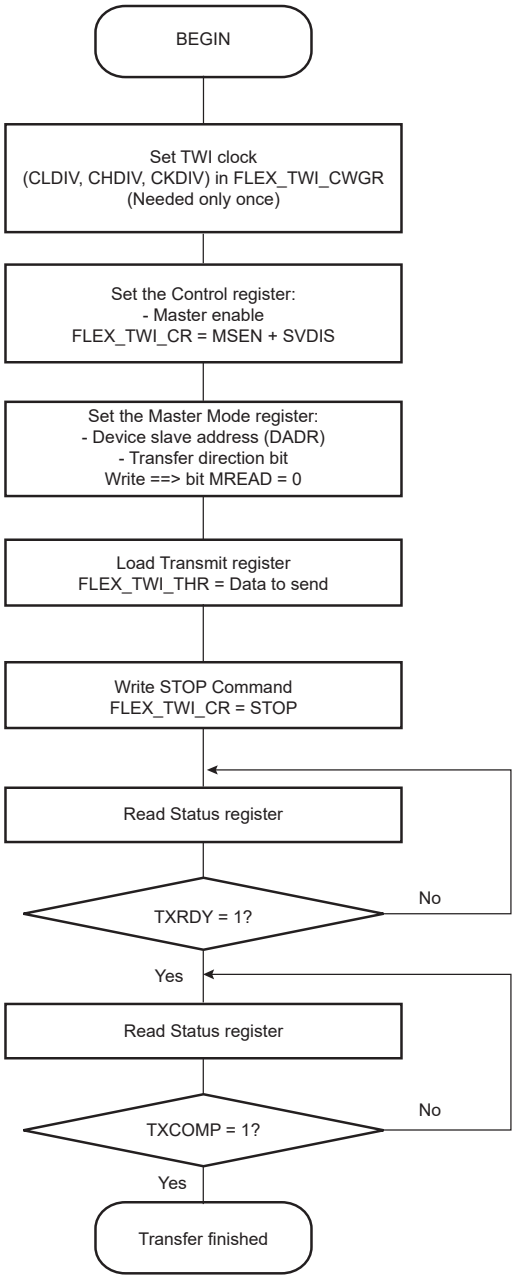


Figure 38-104. TWI Write Operation with Single Data Byte and Internal Address

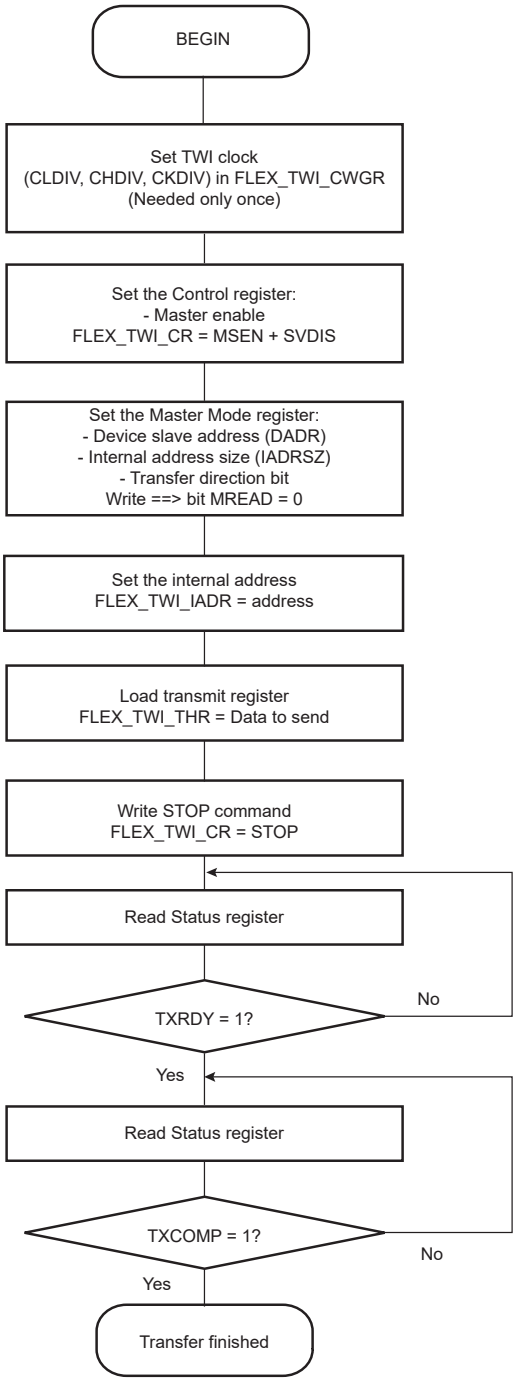


Figure 38-105. TWI Write Operation with Multiple Data Bytes with or without Internal Address

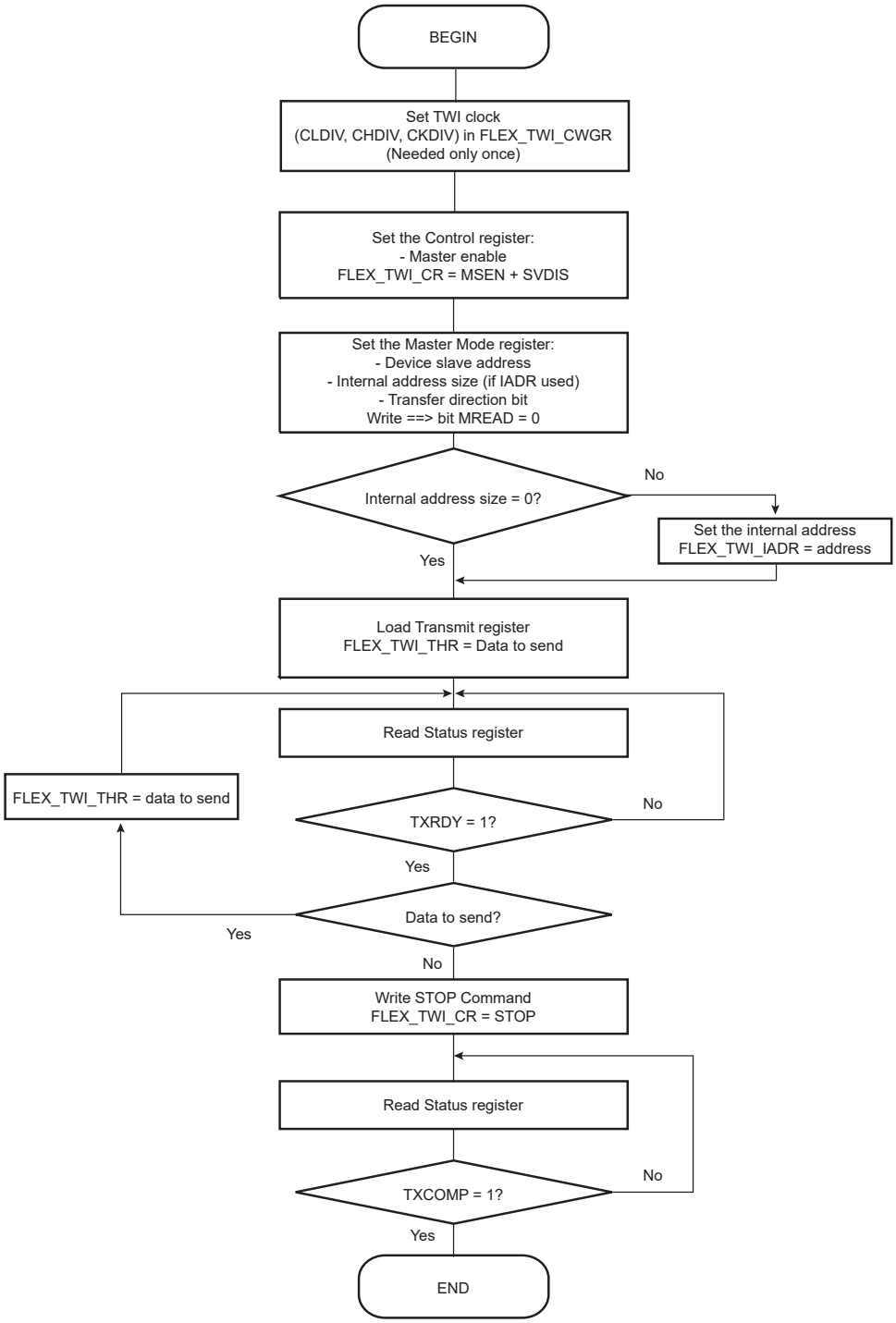


Figure 38-106. SMBus Write Operation with Multiple Data Bytes with or without Internal Address and PEC Sending

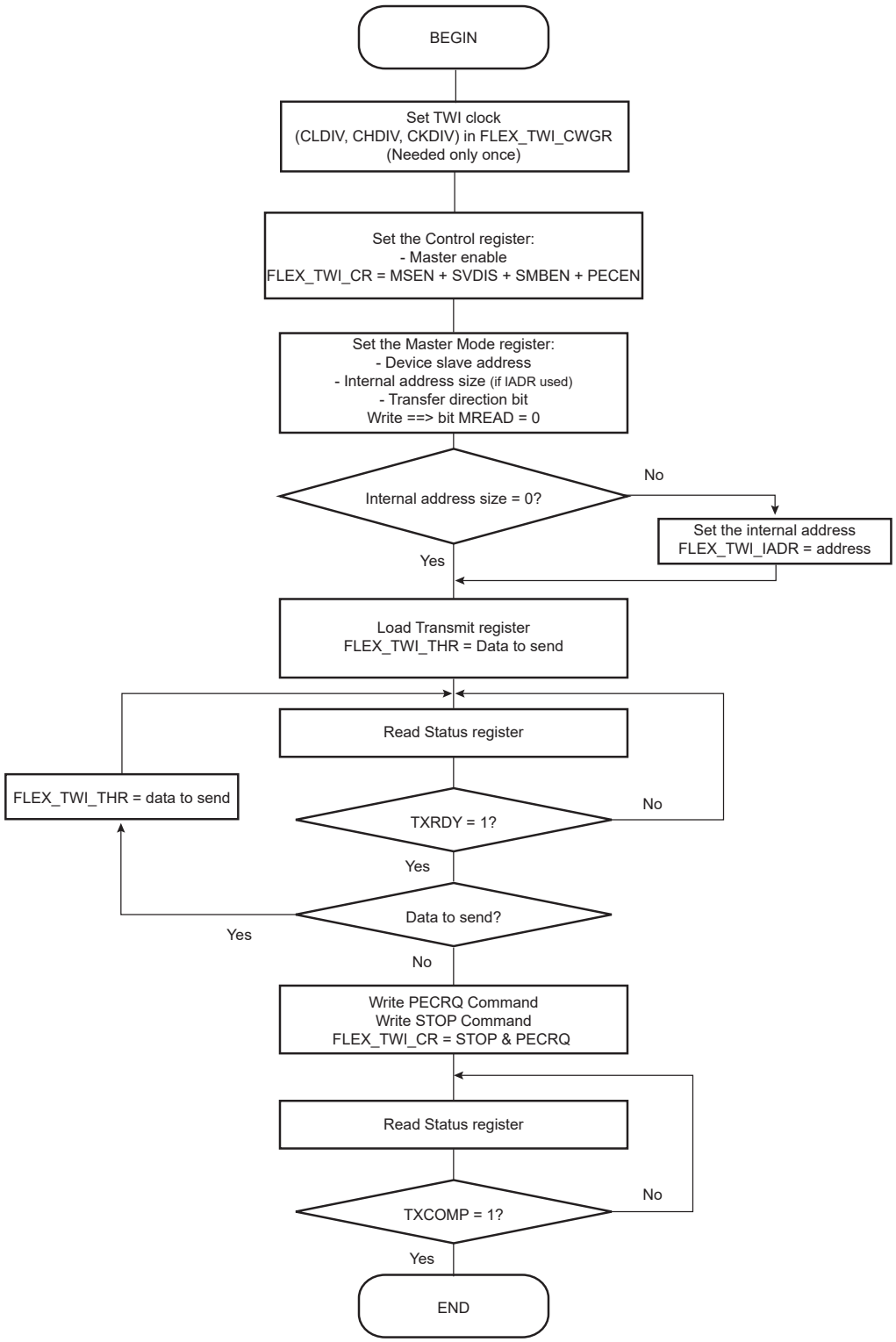


Figure 38-107. SMBus Write Operation with Multiple Data Bytes with PEC and Alternative Command Mode

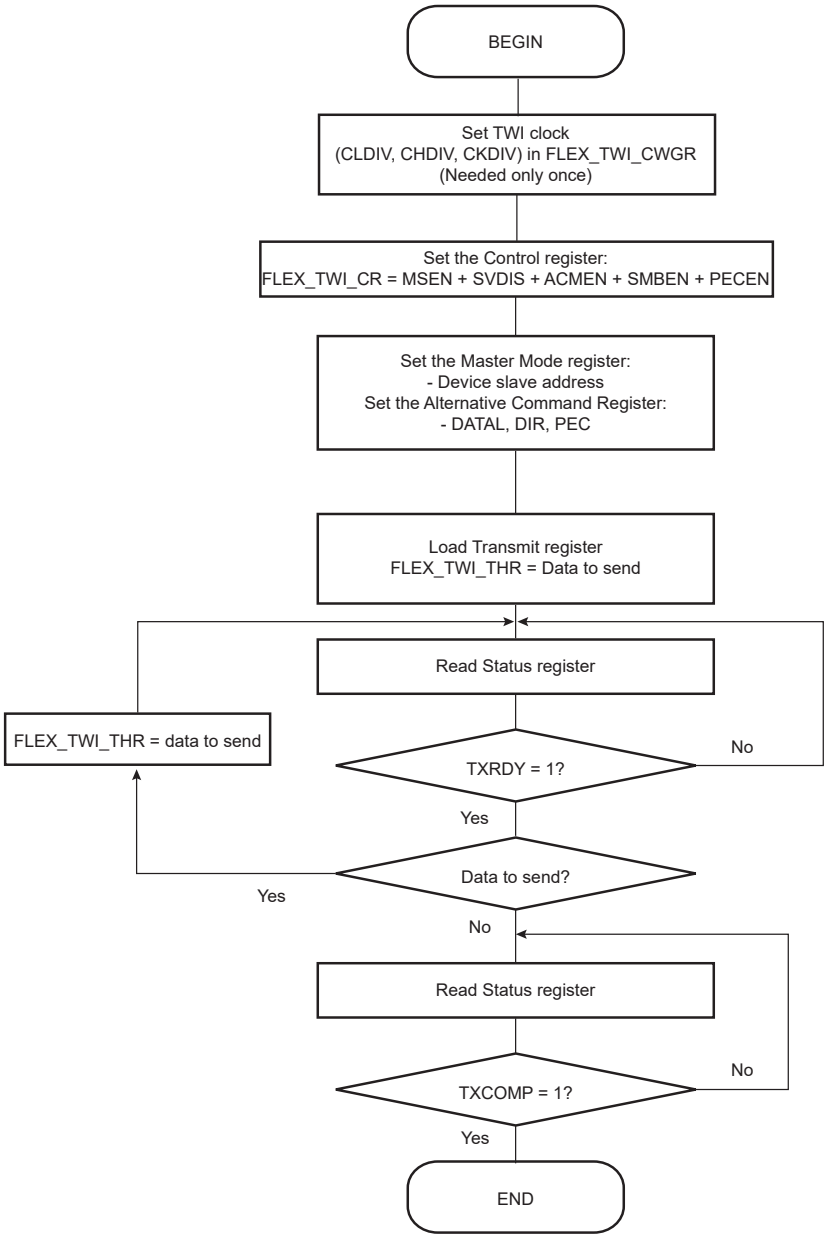


Figure 38-108. TWI Write Operation with Multiple Data Bytes and Read Operation with Multiple Data Bytes (Sr)

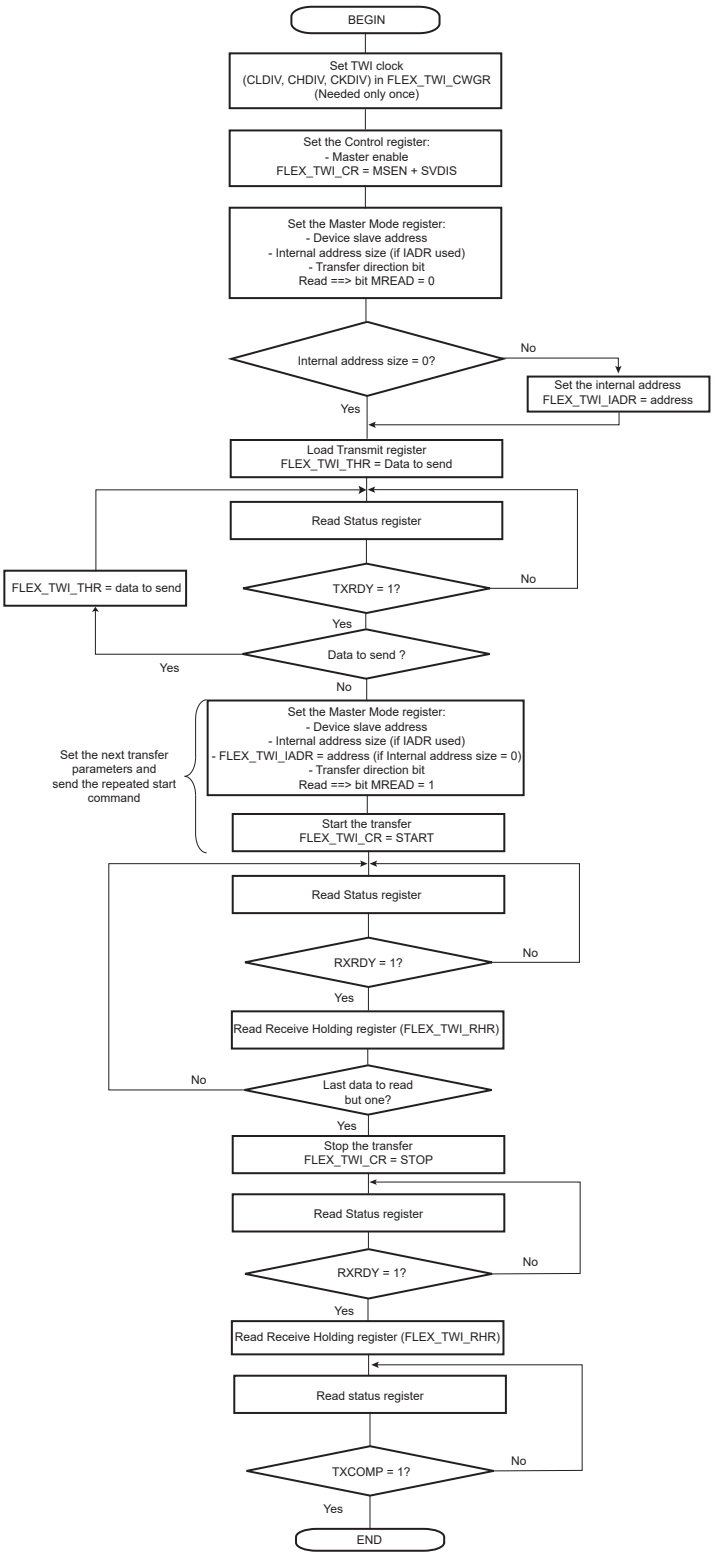


Figure 38-109. TWI Write Operation with Multiple Data Bytes + Read Operation and Alternative Command Mode + PEC

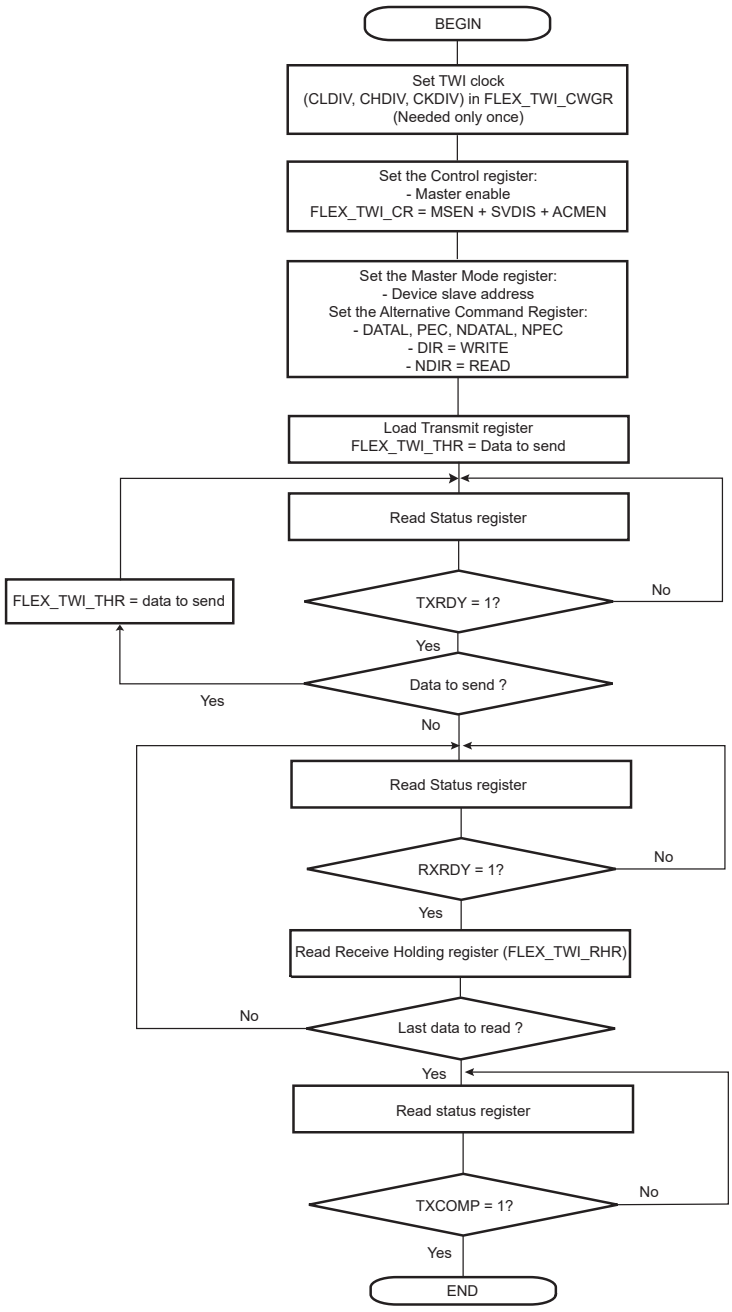


Figure 38-110. TWI Read Operation with Single Data Byte without Internal Address

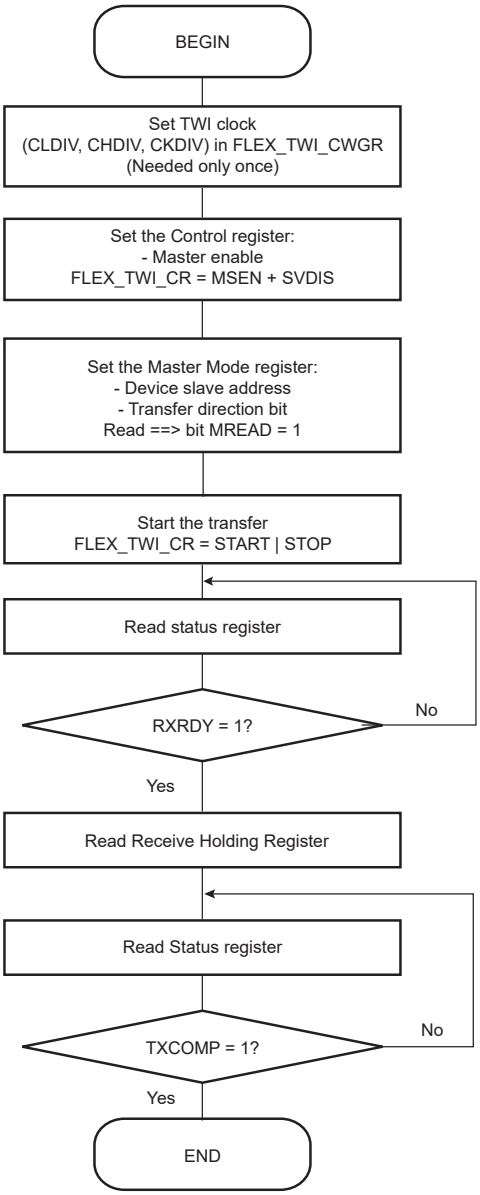


Figure 38-111. TWI Read Operation with Single Data Byte and Internal Address

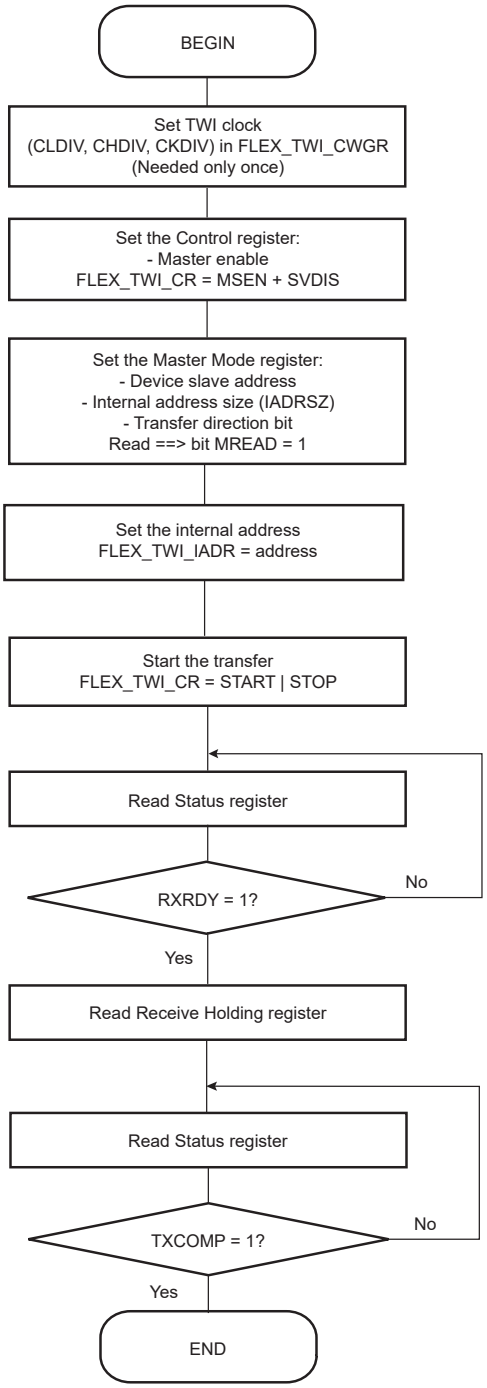


Figure 38-112. TWI Read Operation with Multiple Data Bytes with or without Internal Address

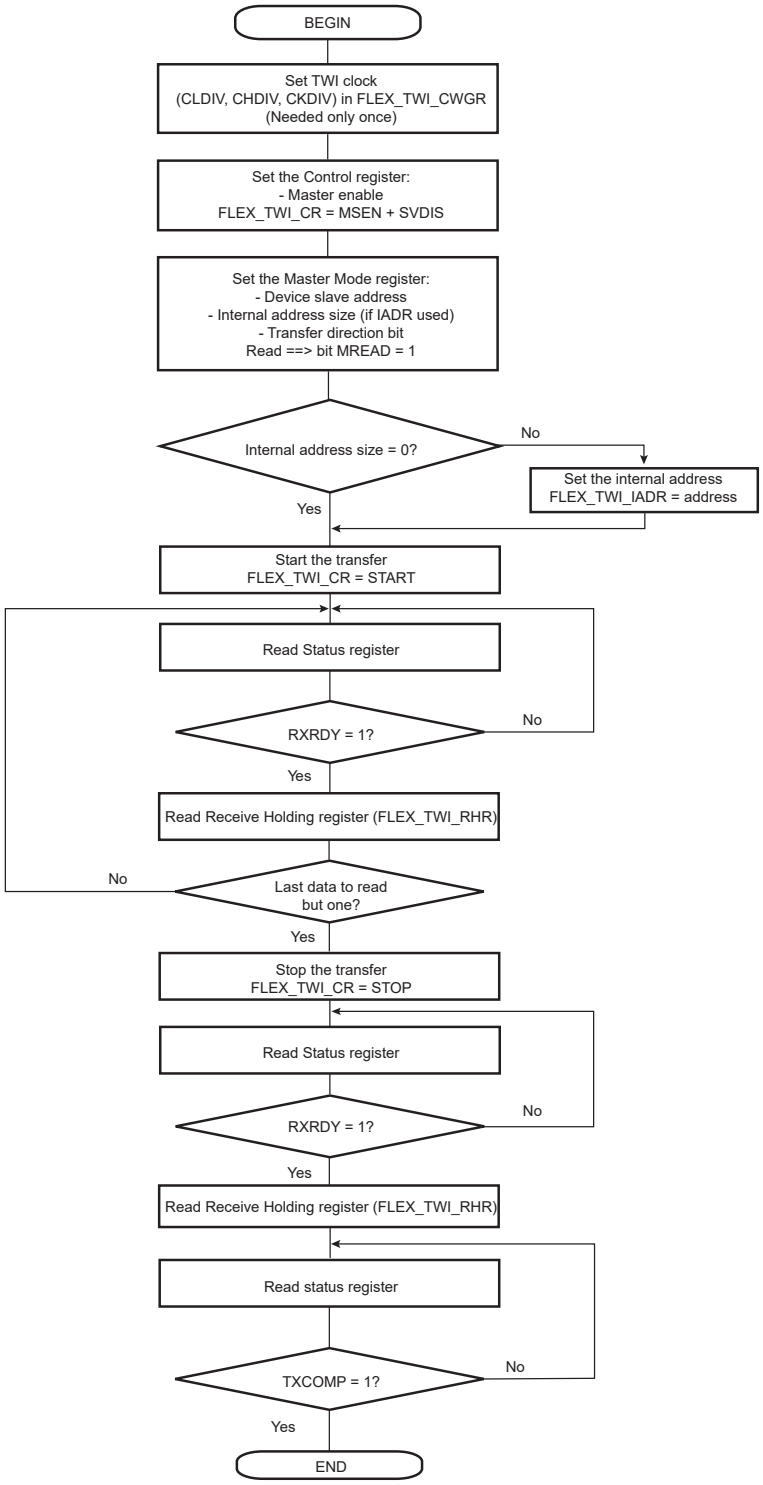


Figure 38-113. TWI Read Operation with Multiple Data Bytes with or without Internal Address with PEC

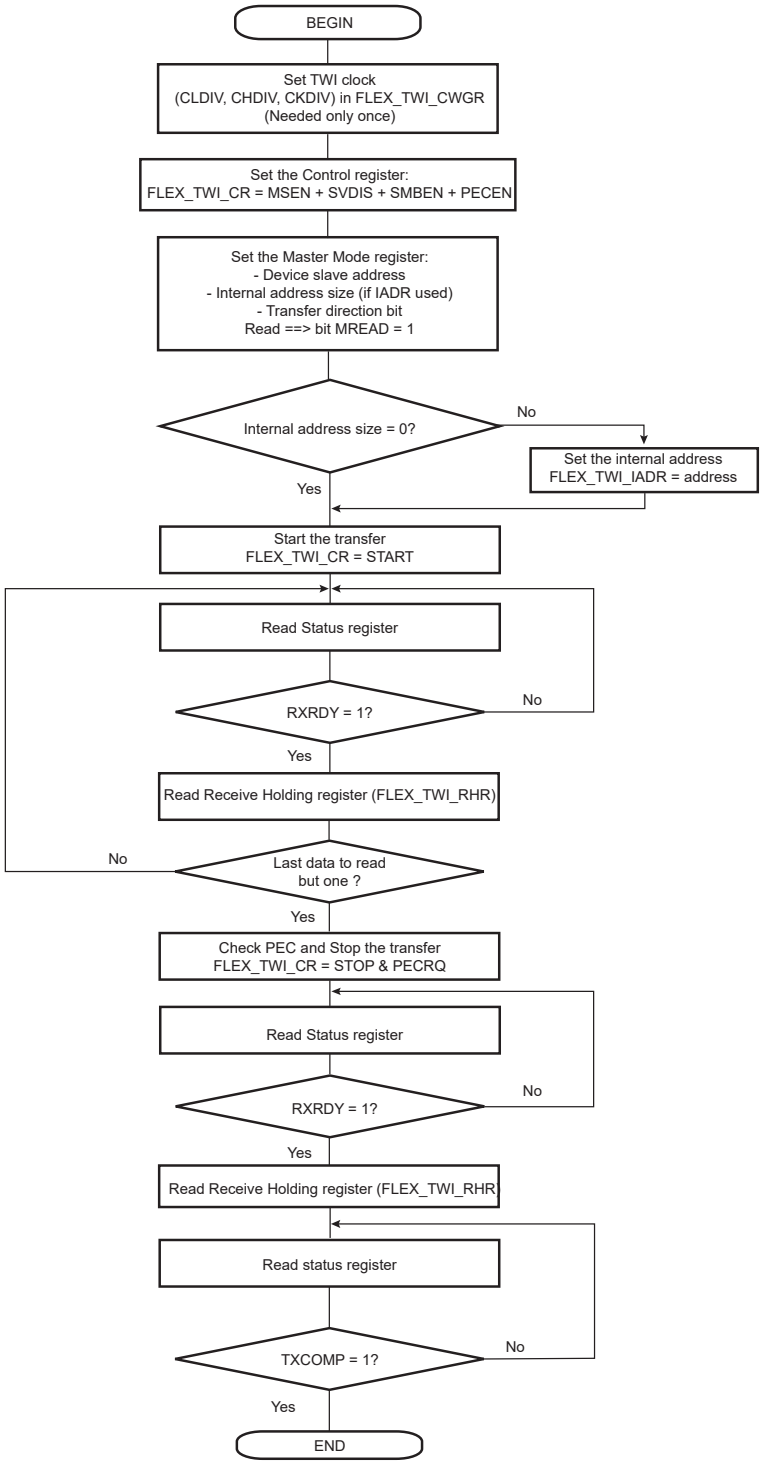


Figure 38-114. TWI Read Operation with Multiple Data Bytes with Alternative Command Mode with PEC

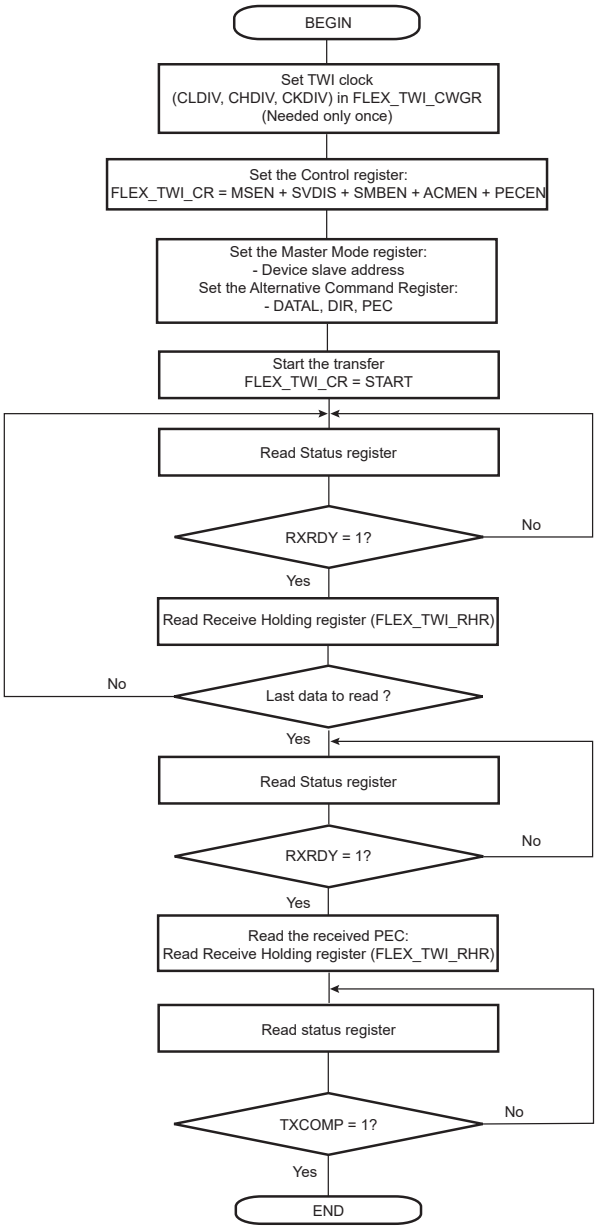


Figure 38-115. TWI Read Operation with Multiple Data Bytes + Write Operation with Multiple Data Bytes (Sr)

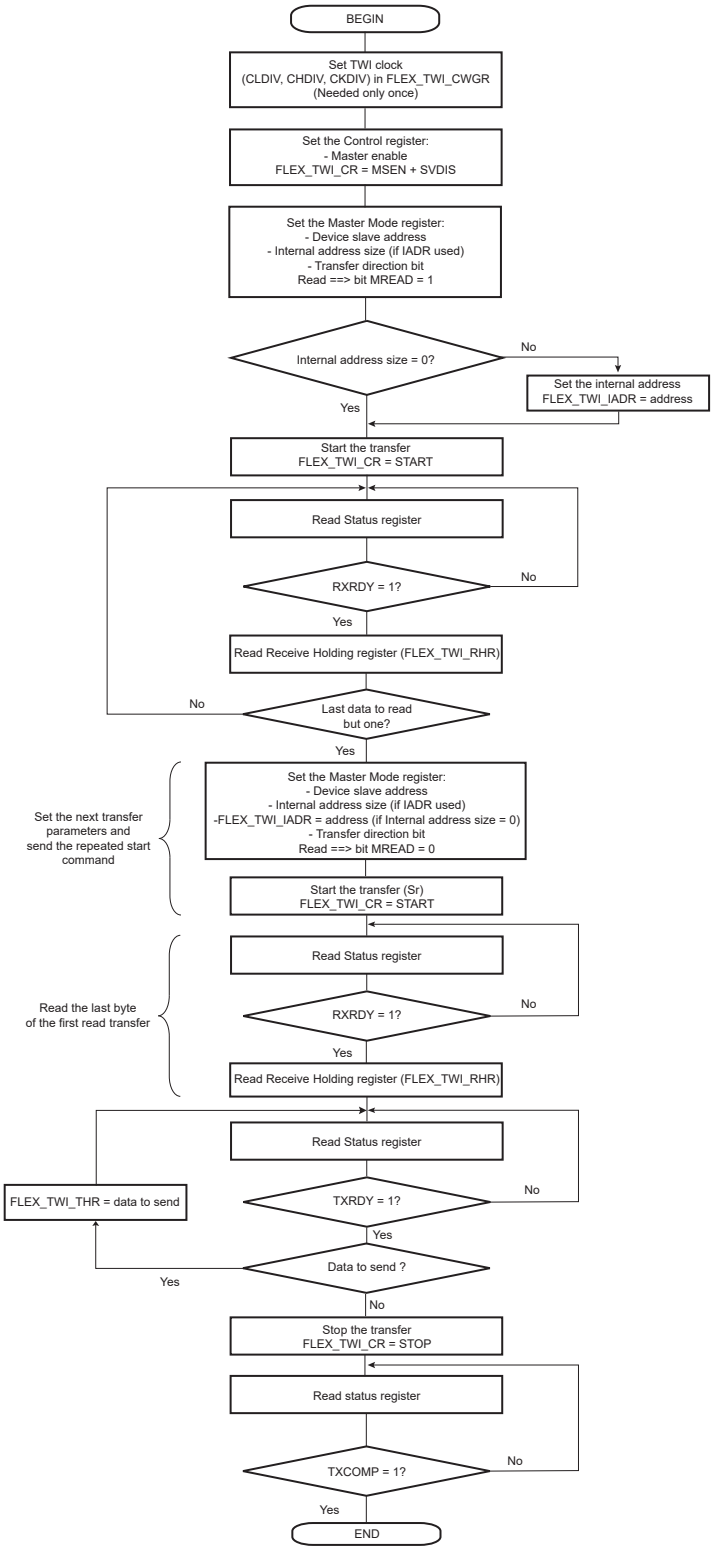
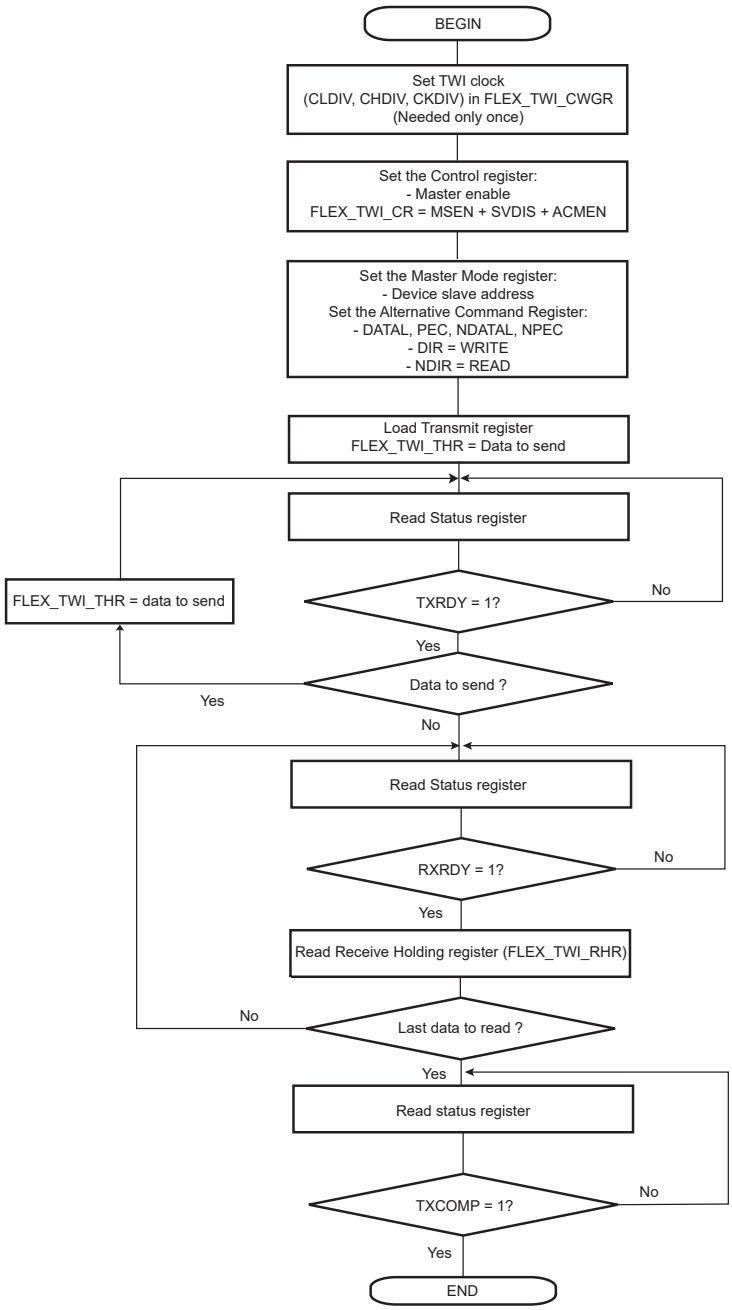


Figure 38-116. TWI Read Operation with Multiple Data Bytes + Write with Alternative Command Mode with PEC



38.9.4 Multi-Master Mode

38.9.4.1 Definition

In Multi-Master mode, more than one master may handle the bus at the same time without data corruption by using arbitration.

Arbitration starts as soon as two or more masters place information on the bus at the same time, and stops (arbitration is lost) for the master that intends to send a logical one while the other master sends a logical zero.

As soon as arbitration is lost by a master, it stops sending data and listens to the bus in order to detect a STOP. When the STOP is detected, the master that has lost arbitration may put its data on the bus by respecting arbitration.

Arbitration is illustrated in figure "Arbitration Cases" below.

38.9.4.2 Different Multi-Master Modes

Two Multi-Master modes may be distinguished:

- TWI as Master Only—TWI is considered as a master only and will never be addressed.
- TWI as Master or Slave—TWI may be either a master or a slave and may be addressed.

Note: Arbitration in supported in both Multi-Master modes.

38.9.4.2.1 TWI as Master Only

In this mode, the TWI is considered as a master only (MSEN is always at one) and must be driven like a master with the ARBLST (ARBitration Lost) flag in addition.

If arbitration is lost (ARBLST = 1), the user must reinitiate the data transfer.

If the user starts a transfer (ex.: DADR + START + W + Write in THR) and if the bus is busy, the TWI automatically waits for a STOP condition on the bus to initiate the transfer (see figure "User Sends Data While the Bus is Busy" below).

Note: The state of the bus (busy or free) is not indicated in the user interface.

38.9.4.2.2 TWI as Master or Slave

The automatic reversal from master to slave is not supported in case of a lost arbitration.

Then, in the case where TWI may be either a master or a slave, the user must manage the pseudo Multi-Master mode described in the steps below:

1. Program the TWI in Slave mode (SADR + MSDIS + SVEN) and perform a slave access (if TWI is addressed).
2. If the TWI has to be set in Master mode, wait until TXCOMP flag is at 1.
3. Program the Master mode (DADR + SVDIS + MSEN) and start the transfer (ex: START + Write in THR).
4. As soon as the Master mode is enabled, the TWI scans the bus in order to detect if it is busy or free. When the bus is considered as free, the TWI initiates the transfer.
5. As soon as the transfer is initiated and until a STOP condition is sent, the arbitration becomes relevant and the user must monitor the ARBLST flag.
6. If the arbitration is lost (ARBLST is = 1), the user must program the TWI in Slave mode in case the master that won the arbitration needs to access the TWI.
7. If the TWI has to be set in Slave mode, wait until TXCOMP flag is at 1 and then program the Slave mode.
   Note: In case the arbitration is lost and the TWI is addressed, the TWI will not acknowledge even if it is programmed in Slave mode as soon as ARBLST = 1. Then the master must repeat SADR.

Figure 38-117. User Sends Data While the Bus is Busy

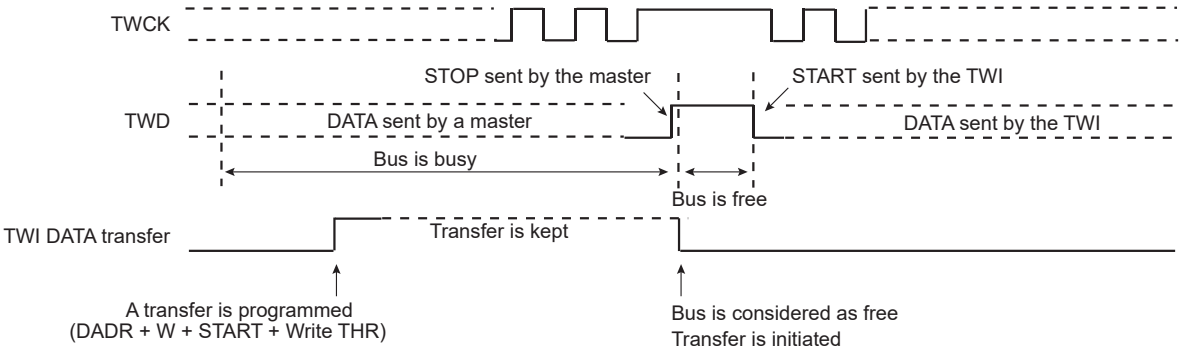
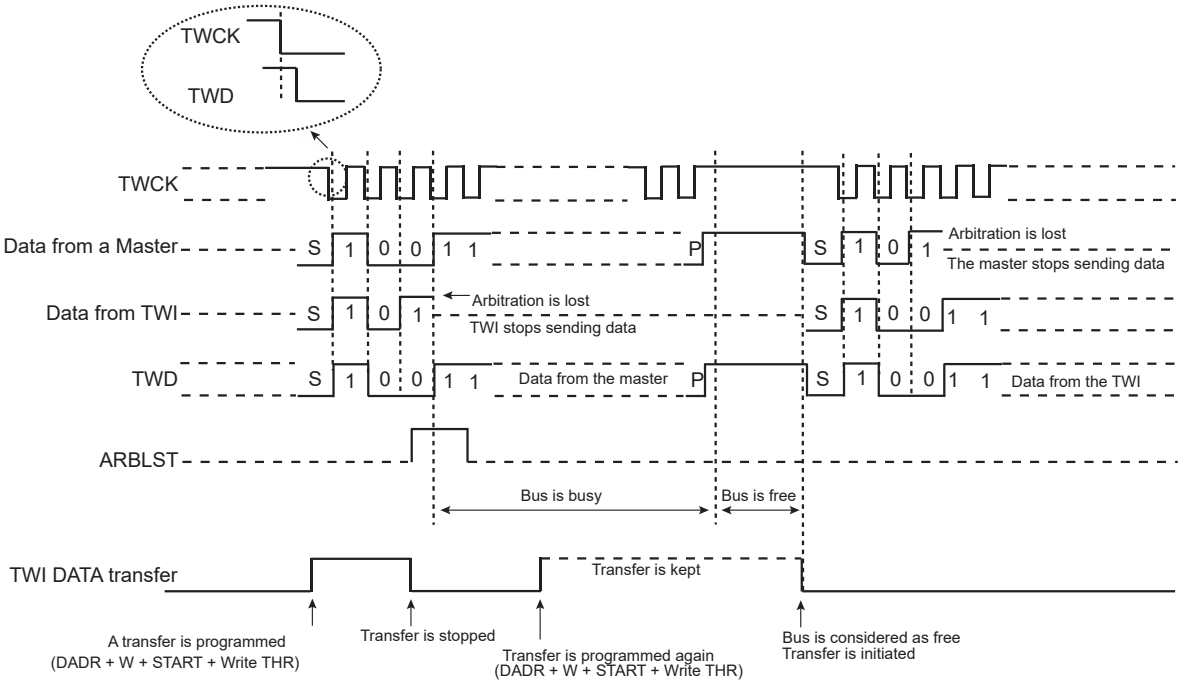
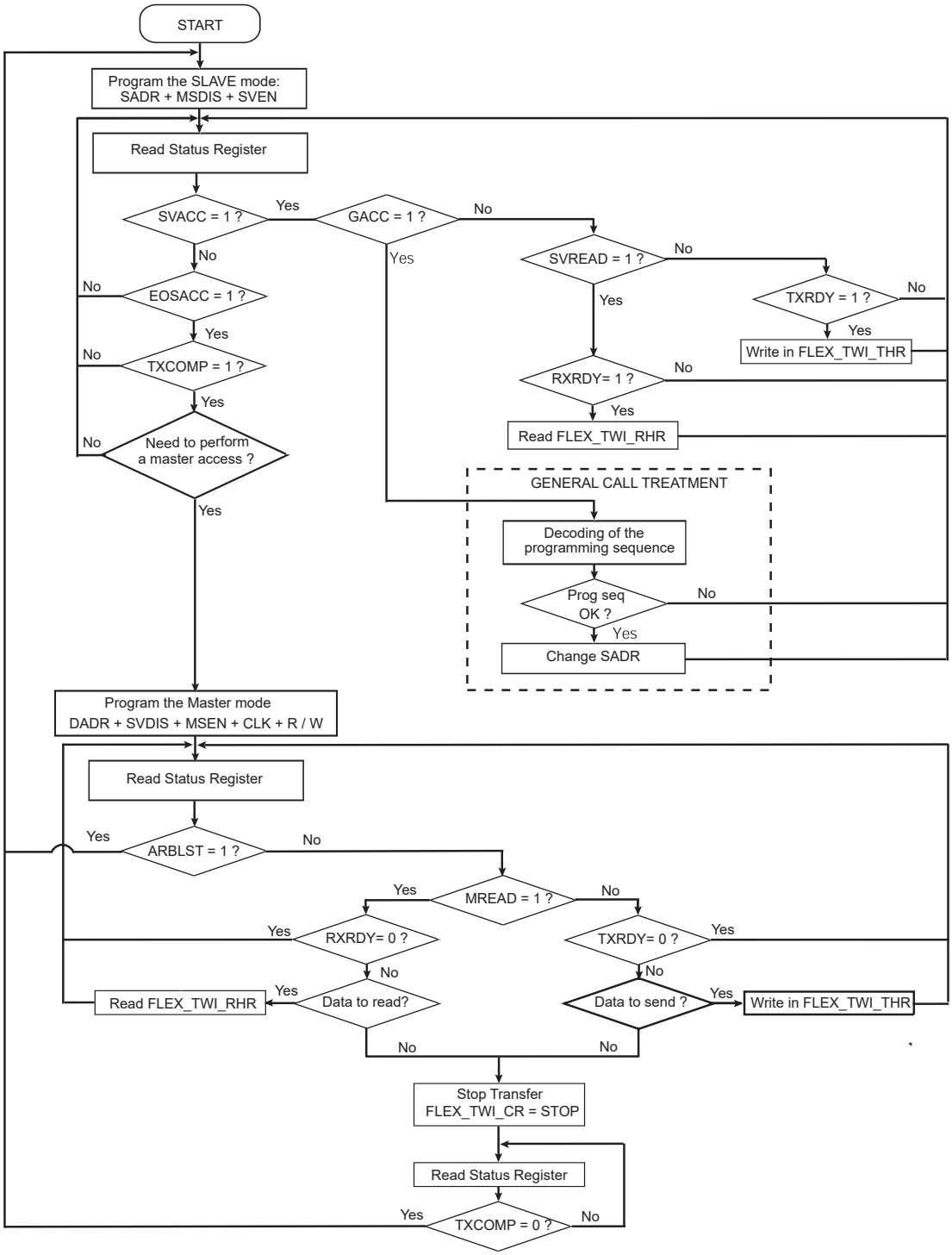


Figure 38-118. Arbitration Cases



The flowchart shown in the following figure gives an example of read and write operations in Multi-Master mode.

Figure 38-119. Multi-Master Mode



38.9.5 Slave Mode

38.9.5.1 Definition

Slave mode is defined as a mode where the device receives the clock and the address from another device called the master.

In this mode, the device never initiates and never completes the transmission (START, REPEATED_START and STOP conditions are always provided by the master).

38.9.5.2 Programming Slave Mode

The following fields must be programmed before entering Slave mode:

1. FLEX_TWI_SMR.SADR: The slave device address is used in order to be accessed by master devices in Read or Write mode.
2. (Optional) FLEX_TWI_SMR.MASK can be set to mask some SADR address bits and thus allow multiple address matching.
3. FLEX_TWI_CR.MSDIS: Disables the Master mode.
4. FLEX_TWI_CR.SVEN: Enables the Slave mode.

As the device receives the clock, values written in FLEX_TWI_CWGR are not processed.

38.9.5.3 Receiving Data

After a START or repeated START condition is detected, and if the address sent by the master matches the slave address programmed in the SADR (Slave Address) field, the SVACC (Slave Access) flag is set and SVREAD (Slave Read) indicates the direction of the transfer.

SVACC remains high until a STOP condition or a repeated START is detected. When such a condition is detected, EOSACC (End Of Slave Access) flag is set.

38.9.5.3.1 Read Sequence

In the case of a read sequence (SVREAD is high), the TWI transfers data written in FLEX_TWI_THR (TWI Transmit Holding Register) until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the read sequence TXCOMP (Transmission Complete) flag is set and SVACC is reset.

As soon as data is written in FLEX_TWI_THR, the TXRDY (Transmit Holding Register Ready) flag is reset, and it is set when the internal shifter is empty and the sent data acknowledged or not. If the data is not acknowledged, the NACK flag is set.

Note that a STOP or a repeated START always follows a NACK.

See figure "Read Access Ordered by a Master" below.

Note: To clear the TXRDY flag in Slave mode, write the FLEX_TWI_CR.SVDIS bit to 1, then write the FLEX_TWI_CR.SVEN bit to 1.

38.9.5.3.2 Write Sequence

In the case of a write sequence (SVREAD is low), the RXRDY (Receive Holding Register Ready) flag is set as soon as a character has been received in FLEX_TWI_RHR (TWI Receive Holding Register). RXRDY is reset when reading FLEX_TWI_RHR.

TWI continues receiving data until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the write sequence TXCOMP flag is set and SVACC reset.

See figure "Write Access Ordered by a Master" below.

38.9.5.3.3 Clock Stretching Sequence

If FLEX_TWI_THR or FLEX_TWI_RHR is not written/read in time, the TWI performs a clock stretching.

Clock stretching information is given by the SCLWS (Clock Wait State) bit.

See figures "Clock Stretching in Read Mode" and "Clock Stretching in Write Mode" below.

Note: Clock stretching can be disabled by configuring the FLEX_TWI_SMR.SCLWSDIS bit. In that case, UNRE and OVRE flags will indicate underrun (when FLEX_TWI_THR is not filled on time) or overrun (when FLEX_TWI_RHR is not read on time).

38.9.5.3.4 General Call

In the case where a GENERAL CALL is performed, the GACC (General Call Access) flag is set.

After GACC is set, it is up to the user to interpret the meaning of the GENERAL CALL and to decode the new address programming sequence.

See figure "Master Performs a General Call" below.

38.9.5.4 Data Transfer

38.9.5.4.1 Read Operation

The Read mode is defined as a data requirement from the master.

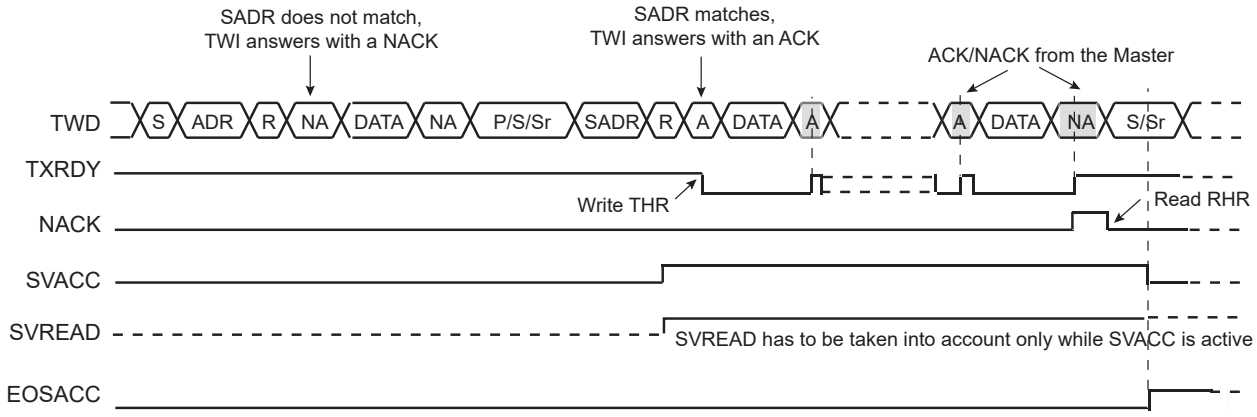
After a START or a REPEATED START condition is detected, the decoding of the address starts. If the slave address (SADR) is decoded, SVACC is set and SVREAD indicates the direction of the transfer.

Until a STOP or REPEATED START condition is detected, TWI continues sending data loaded in FLEX_TWI_THR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

The following figure describes the read operation.

Figure 38-120. Read Access Ordered by a Master



Note:

1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. TXRDY is reset when data has been transmitted from FLEX_TWI_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.

38.9.5.4.2 Write Operation

The Write mode is defined as a data transmission from the master.

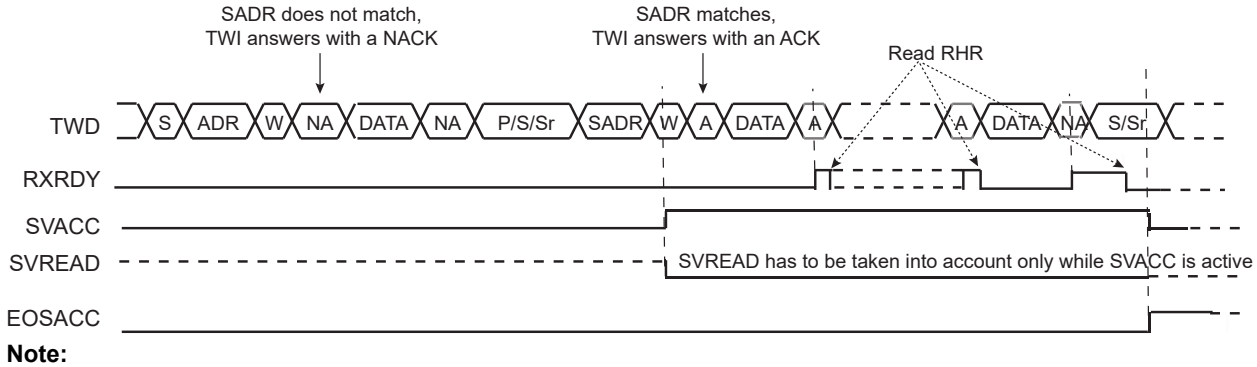
After a START or a REPEATED START, the decoding of the address starts. If the slave address is decoded, SVACC is set and SVREAD indicates the direction of the transfer (SVREAD is low in this case).

Until a STOP or REPEATED START condition is detected, TWI stores the received data in FLEX_TWI_RHR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

The following figure describes the write operation.

Figure 38-121. Write Access Ordered by a Master



Note:

1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. RXRDY is set when data has been transmitted from the internal shifter to FLEX_TWI_RHR, and reset when this data is read.

38.9.5.4.3 General Call

The general call is performed in order to change the address of the slave.

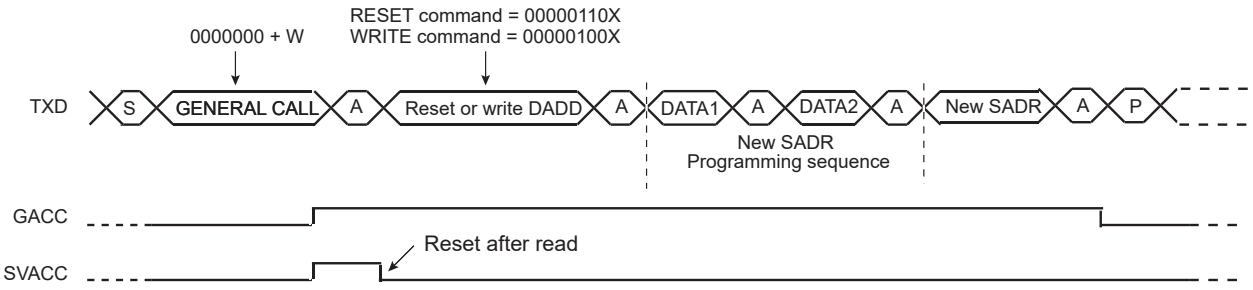
If a GENERAL CALL is detected, GACC is set.

After the detection of general call, it is up to the user to decode the commands which follow.

In case of a WRITE command, the user has to decode the programming sequence and program a new SADR if the programming sequence matches.

The following figure describes the general call access.

Figure 38-122. Master Performs a General Call



Note: This method enables to create a user-specific programming sequence by choosing the number of programming bytes. The programming sequence has to be provided to the master.

38.9.5.4.4 Clock Stretching

In both Read and Write modes, it may happen that the FLEX_TWI_THR/FLEX_TWI_RHR buffer is not filled/emptied before the transmission/reception of a new character. In this case, to avoid sending/receiving undesired data, a clock stretching mechanism is implemented.

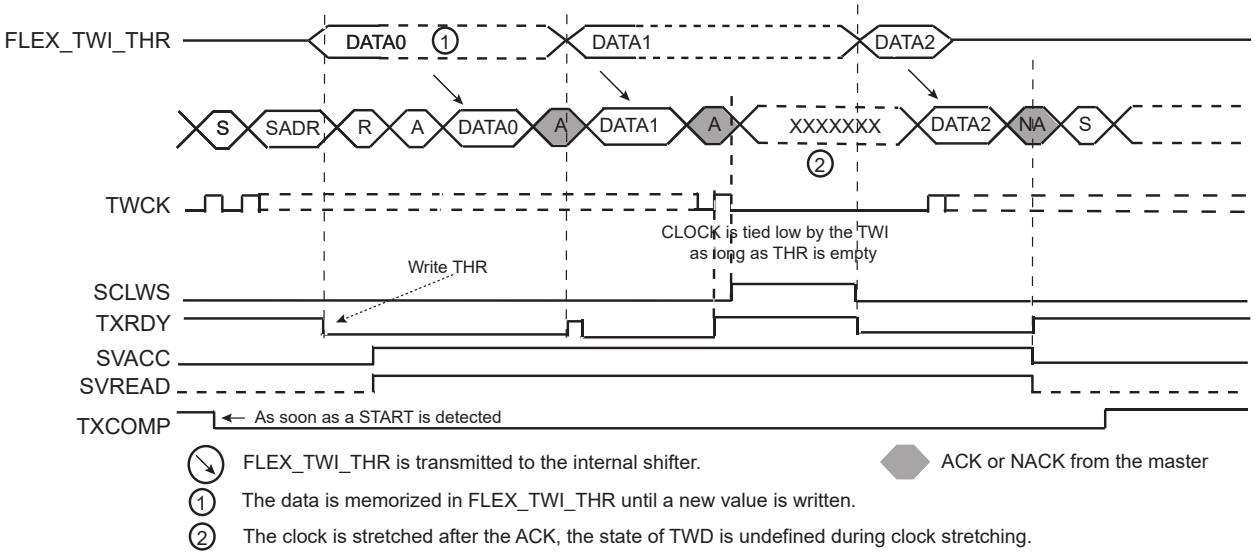
Note: Clock stretching can be disabled by setting the FLEX_TWI_SMR.SCLWSDIS bit. In that case, the UNRE and OVRE flags indicate an underrun (when FLEX_TWI_THR is not filled on time) or an overrun (when FLEX_TWI_RHR is not read on time).

— Clock Stretching in Read Mode

The clock is tied low if the internal shifter is empty and if a STOP or REPEATED START condition was not detected. It is tied low until the internal shifter is loaded.

The following figure describes clock stretching in Read mode.

Figure 38-123. Clock Stretching in Read Mode



Note:

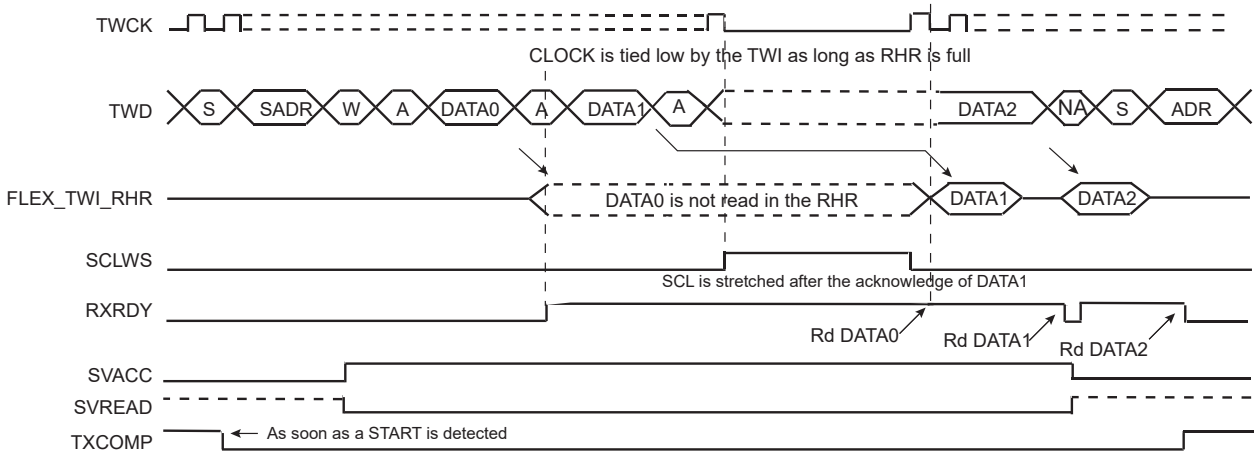
1. TXRDY is reset when data has been written in FLEX_TWI_THR to the internal shifter, and set when this data has been acknowledged or non acknowledged.
2. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
3. SCLWS is automatically set when the clock stretching mechanism is started.

— Clock Stretching in Write Mode

The clock is tied low if the internal shifter and FLEX_TWI_RHR are full. If a STOP or REPEATED_START condition was not detected, it is tied low until FLEX_TWI_RHR is read.

The following figure describes the clock stretching in Write mode.

Figure 38-124. Clock Stretching in Write Mode



Note:

1. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
2. SCLWS is automatically set when the clock stretching mechanism is started and automatically reset when the mechanism is finished.

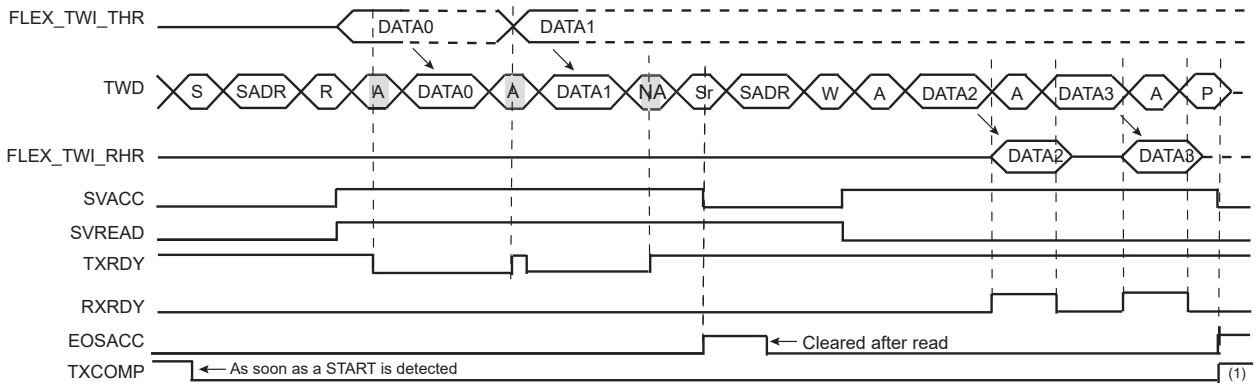
38.9.5.4.5 Reversal after a Repeated Start

— Reversal of Read to Write

The master initiates the communication by a read command and finishes it by a write command.

The following figure describes the repeated start and the reversal from Read mode to Write mode.

Figure 38-125. Repeated Start and Reversal from Read Mode to Write Mode



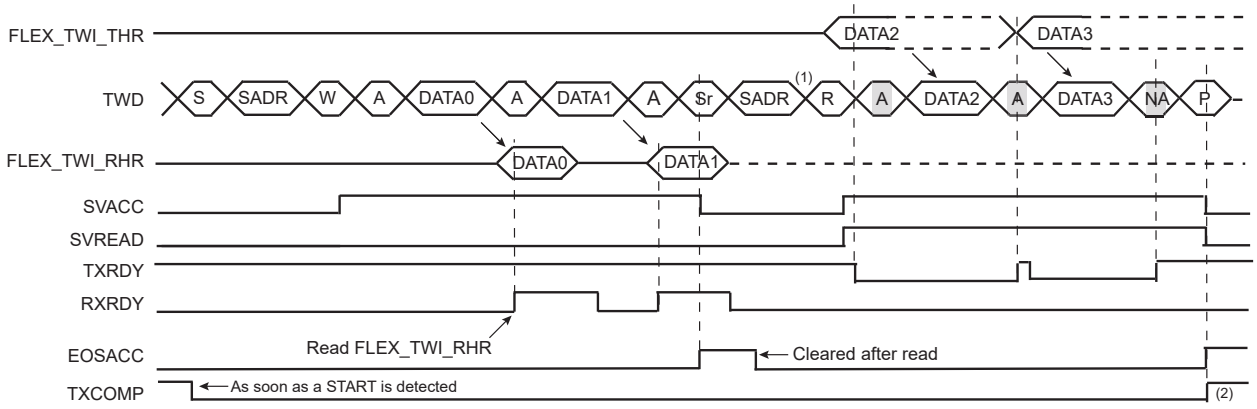
Note:
1. TXCOMP is only set at the end of the transmission because after the repeated start, SADR is detected again.

— Reversal of Write to Read

The master initiates the communication by a write command and finishes it by a read command.

The following figure describes the repeated start and the reversal from Write mode to Read mode.

Figure 38-126. Repeated Start and Reversal from Write Mode to Read Mode



Notes:
1. In this case, if FLEX_TWI_THR has not been written at the end of the read command, the clock is automatically stretched before the ACK.
2. TXCOMP is only set at the end of the transmission because after the repeated start, SADR is detected again.

38.9.5.4.6 SMBus Mode

SMBus mode is enabled when the FLEX_TWI_CR.SMEN bit is written to one. SMBus mode operation is similar to I²C operation with the following exceptions:

1. Only 7-bit addressing can be used.
2. The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into FLEX_TWI_SMBTR.
3. Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
4. A set of addresses have been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring FLEX_TWI_CR appropriately.

— Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing the FLEX_TWI_CR.PECEN bit to one will send/check the FLEX_TWI_ACR.PEC field in the current transfer. The PEC generator is always updated on every bit transmitted or received, so that PEC handling on following linked transfers will be correct.

In Slave Receiver mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave will compare it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave will return an ACK to the master. If the PEC values differ, data was corrupted, and the slave will return a NACK value. The FLEX_TWI_SR.PECERR bit is set automatically if a PEC error occurred.

In Slave Transmitter mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master will compare it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the master must take appropriate action.

See section Slave Read/Write Flowcharts for detailed flowcharts.

— Timeouts

The TWI SMBus Timing Register (FLEX_TWI_SMBTR) configures the SMBus timeout values. If a timeout occurs, the slave will leave the bus. Furthermore, the FLEX_TWI_SR.TOUT bit is set.

38.9.5.5 High-Speed Slave Mode

High-speed mode is enabled when the FLEX_TWI_CR.HSEN bit is written to one. Furthermore, the analog pad filter must be enabled, the FLEX_TWI_FILTR.PADFEN bit must be written to one and the FLEX_TWI_FILTR.FILT bit must be cleared. TWI High-speed mode operation is similar to TWI operation with the following exceptions:

1. A master code is received first at normal speed before entering High-speed mode period.
2. When TWI High-speed mode is active, clock stretching is only allowed after acknowledge (ACK), not-acknowledge (NACK), START (S) or repeated START (Sr) (asa consequence, OVF may happen).

TWI High-speed mode allows transfers of up to 3.4 Mbit/s.

The TWI slave in High-speed mode requires that the peripheral clock runs at a minimum of 14 MHz if slave clock stretching is enabled (SCLWSDIS bit at ‘0’). If slave clock stretching is disabled (SCLWSDIS bit at ‘1’), the peripheral clock must run at a minimum of 11 MHz (assuming the system has no latency).

Note:

1. When slave clock stretching is disabled, FLEX_TWI_RHR must always be read before receiving the next data (MASTER write frame). It is strongly recommended to use either the polling method on the FLEX_TWI_SR.RXRDY flag, or the DMA. If the receive is managed by an interrupt, the TWI interrupt priority must be set to the right level and its latency minimized to avoid receive overrun.
2. When slave clock stretching is disabled, FLEX_TWI_THR must be filled with the first data to send before the beginning of the frame (MASTER read frame). It is strongly recommended to use either the polling method on the FLEX_TWI_SR.TXRDY flag, or the DMA. If the transmit is managed by an interrupt, the TWI interrupt priority must be set to the right level and its latency minimized to avoid transmit underrun.

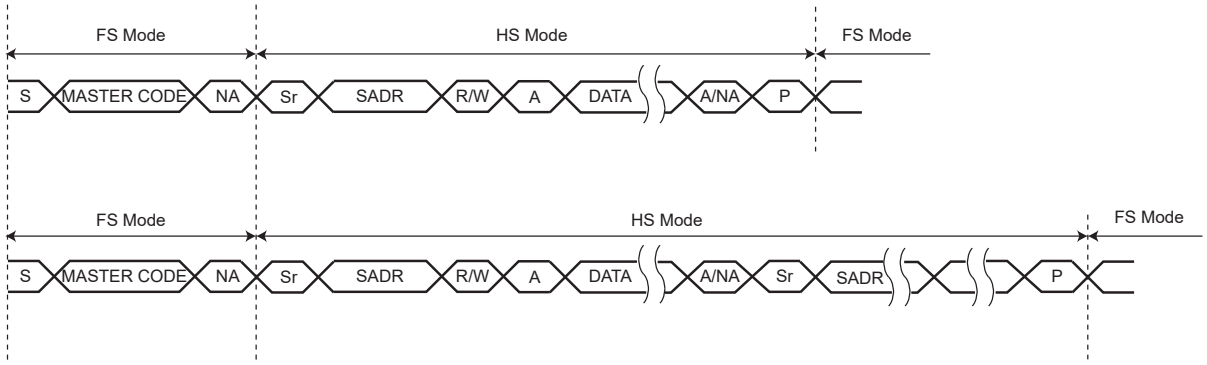
38.9.5.5.1 Read/Write Operation

A TWI high-speed frame always begins with the following sequence:

1. START condition (S)
2. Master Code (0000 1XXX)
3. Not-acknowledge (NACK)

When the TWI is programmed in Slave mode and TWI High-speed mode is activated, master code matching is activated and internal timings are set to match the TWI High-speed mode requirements.

Figure 38-127. High-Speed Mode Read/Write



38.9.5.5.2 Usage

TWI High-speed mode usage is the same as the standard TWI (see section Read/Write Flowcharts).

38.9.5.6 Alternative Command

In Slave mode, the Alternative Command mode is used when the SMBus mode is enabled to send or check the PEC byte.

The Alternative Command mode is enabled by setting the ACMEN bit of the TWIHS Control Register, and the transfer is configured in TWIHS_ACR.

For a combined transfer with PEC, only the NPEC bit in TWIHS_ACR must be set as the PEC byte is sent once at the end of the frame.

See section Slave Read/Write Flowcharts for detailed flowcharts.

38.9.5.7 Slave Read/Write Flowcharts

The flowchart shown in the following figure gives an example of read and write operations in Slave mode. A polling or interrupt method can be used to check the status bits. The interrupt method requires that the Interrupt Enable Register (FLEX_TWI_IER) be configured first.

Figure 38-128. Read/Write in Slave Mode

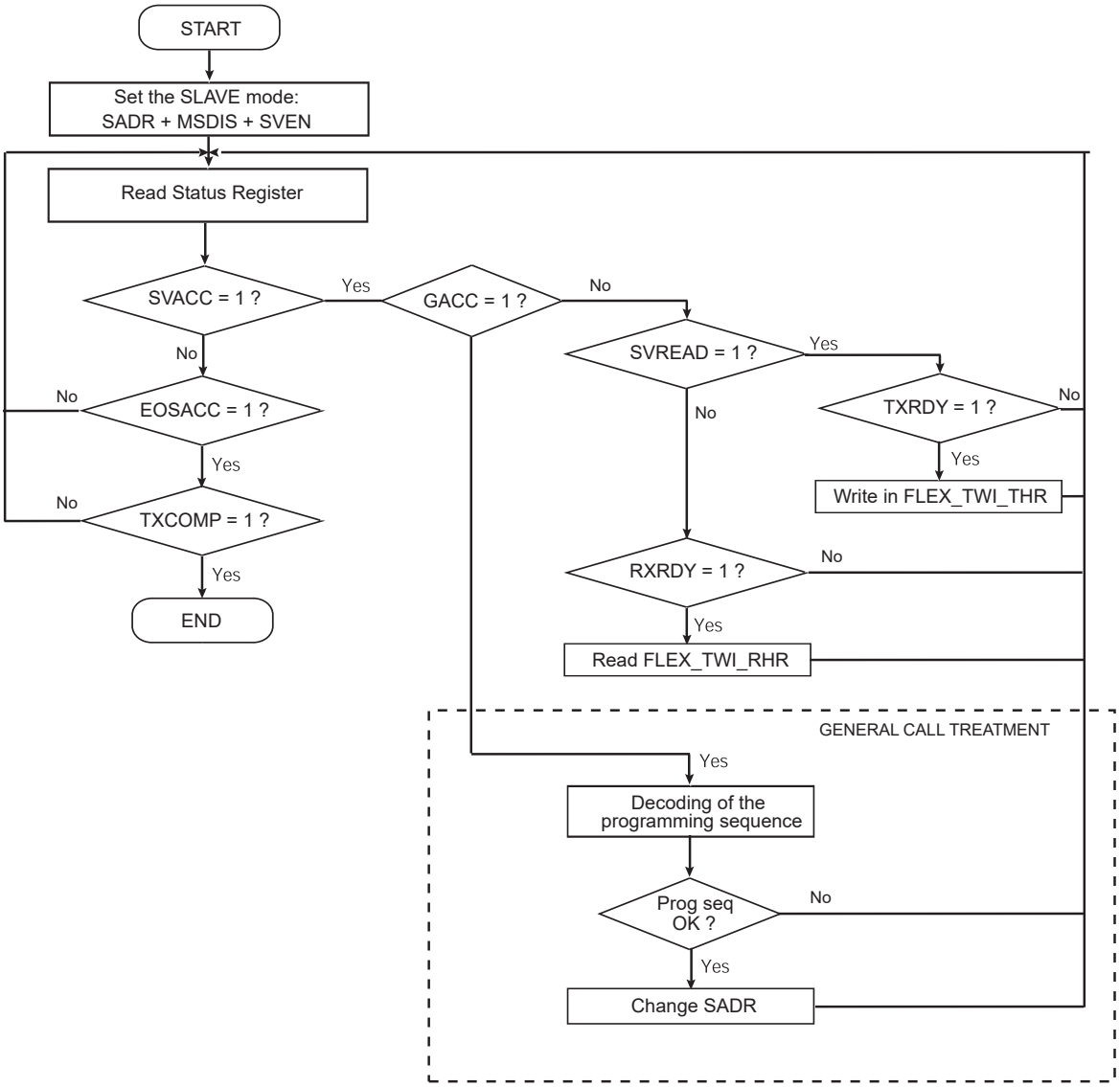


Figure 38-129. Read/Write in Slave Mode with SMBus PEC

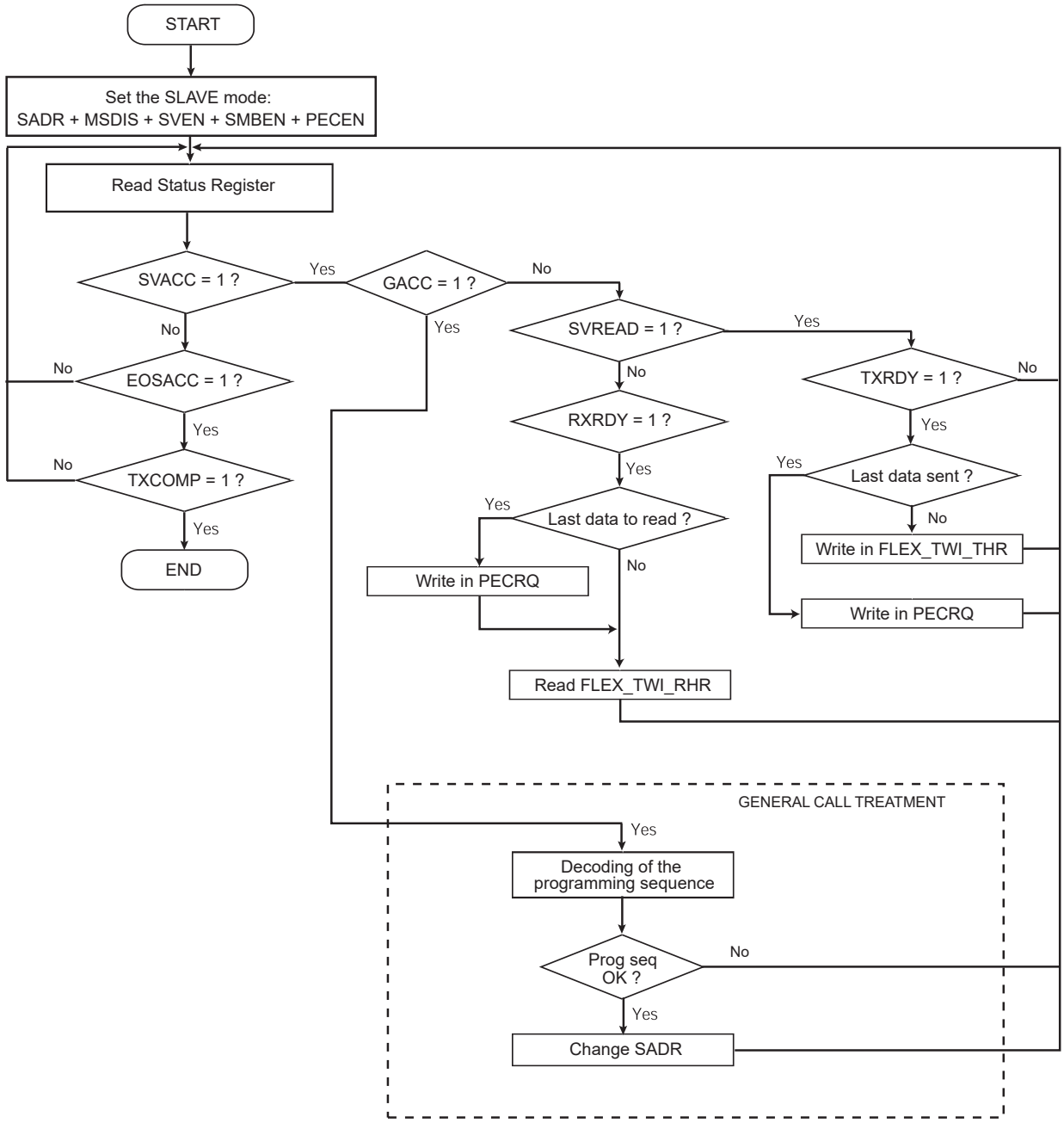
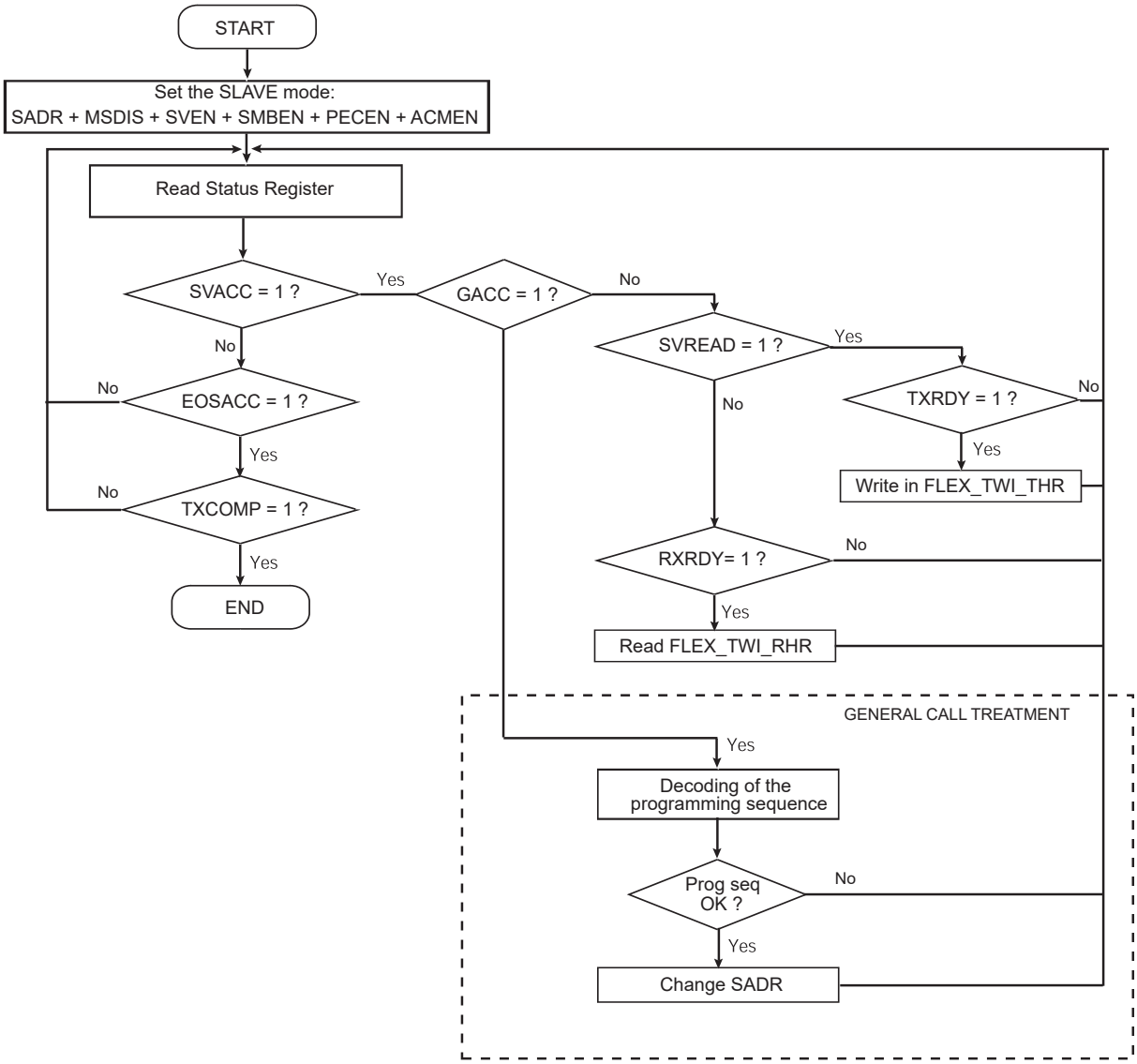


Figure 38-130. Read/Write in Slave Mode with SMBus PEC and Alternative Command Mode



38.9.6 TWI FIFOs

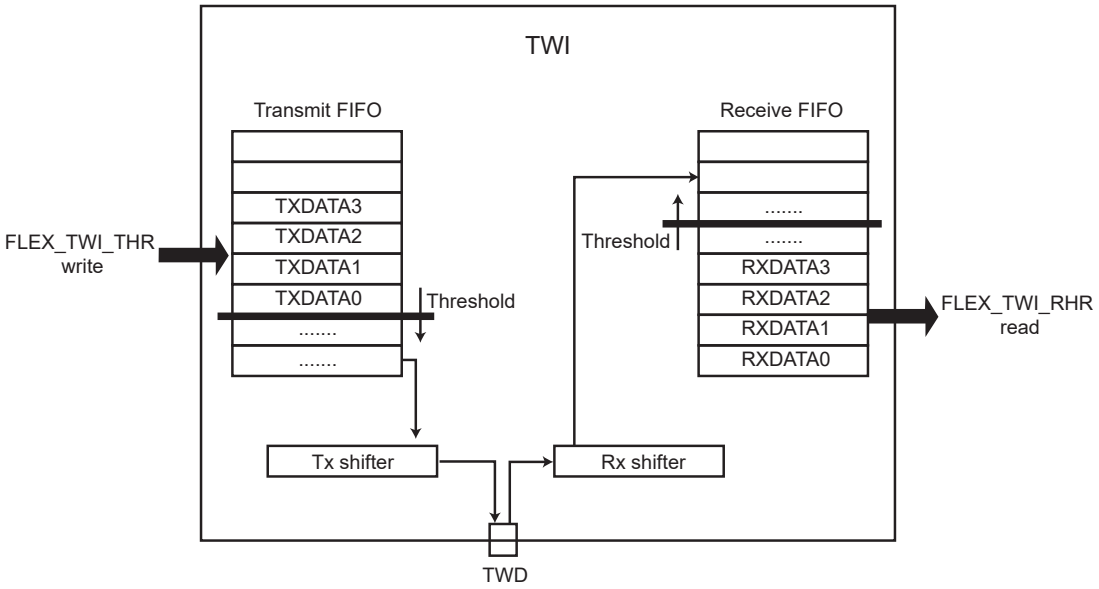
38.9.6.1 Overview

The TWI includes two FIFOs which can be enabled/disabled using FLEX_TWI_CR.FIFOEN/FIFODIS. It is recommended to disable both Master and Slave modes before enabling or disabling the FIFOs (FLEX_TWI_CR.MSDIS/SVDIS).

Writing FLEX_TWI_CR.FIFOEN to ‘1’ enables a 8-data Transmit FIFO and a 8-data Receive FIFO.

It is possible to write or to read single or multiple data in the same access to FLEX_TWI_THR/RHR, depending on FLEX_TWI_FMR.TXRDYM/RXRDYM settings.

Figure 38-131. FIFOs Block Diagram



38.9.6.2 Sending Data with FIFO Enabled

When the Transmit FIFO is enabled, write access to FLEX_TWI_THR loads the Transmit FIFO.

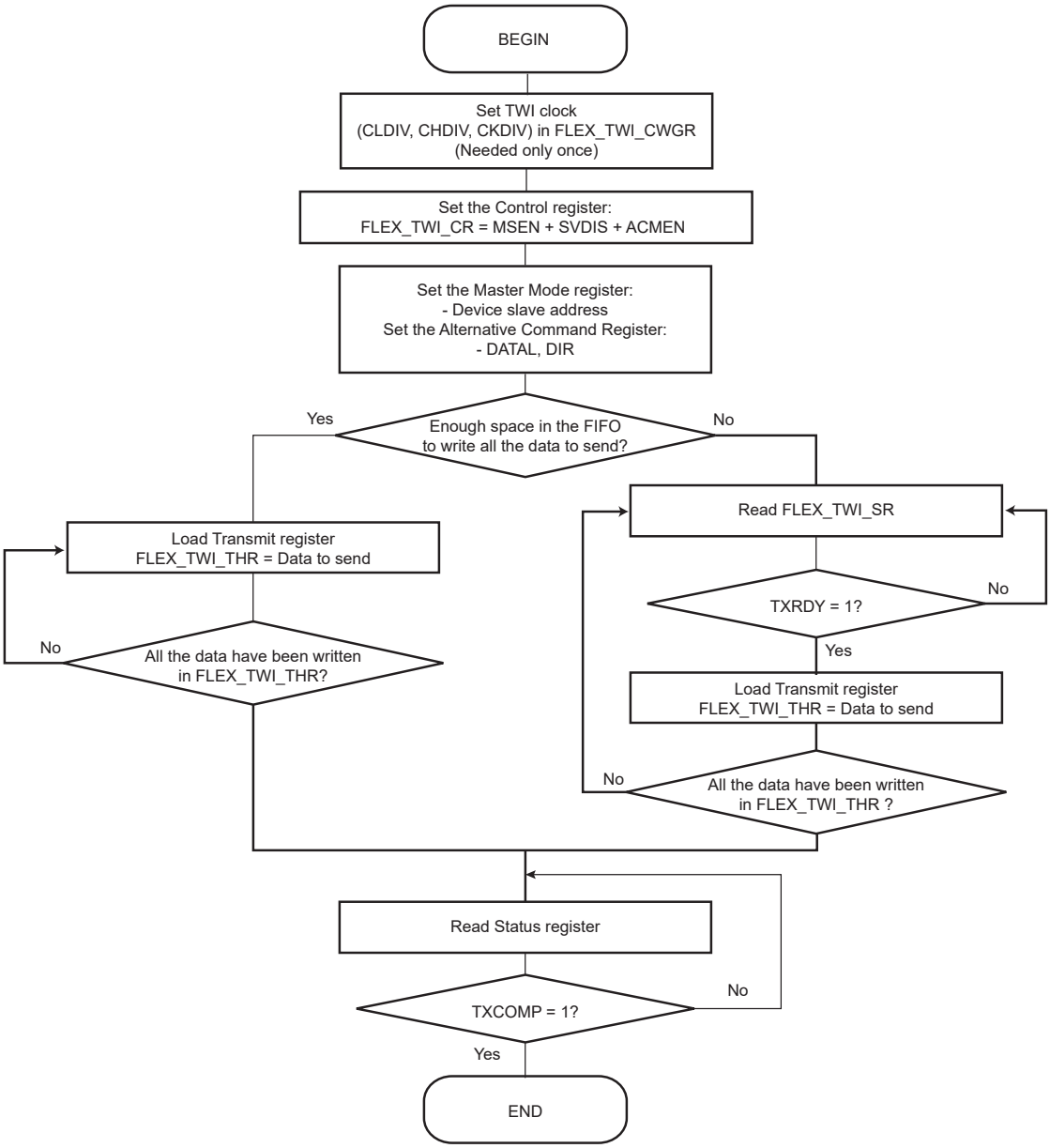
The Transmit FIFO level is provided in FLEX_TWI_FLR.TXFL. If the FIFO can accept the number of data to be transmitted, there is no need to monitor FLEX_TWI_SR.TXRDY and the data can be successively written in FLEX_TWI_THR.

If the FIFO cannot accept the data due to insufficient space, wait for the TXRDY flag to be set before writing the data in FLEX_TWI_THR.

When the space in the FIFO allows only a portion of the data to be written, the TXRDY flag must be monitored before writing the remaining data.

See figures Sending Data with FIFO Enabled in Master Mode and Sending/Receiving Data with FIFO Enabled in Slave Mode.

Figure 38-132. Sending Data with FIFO Enabled in Master Mode



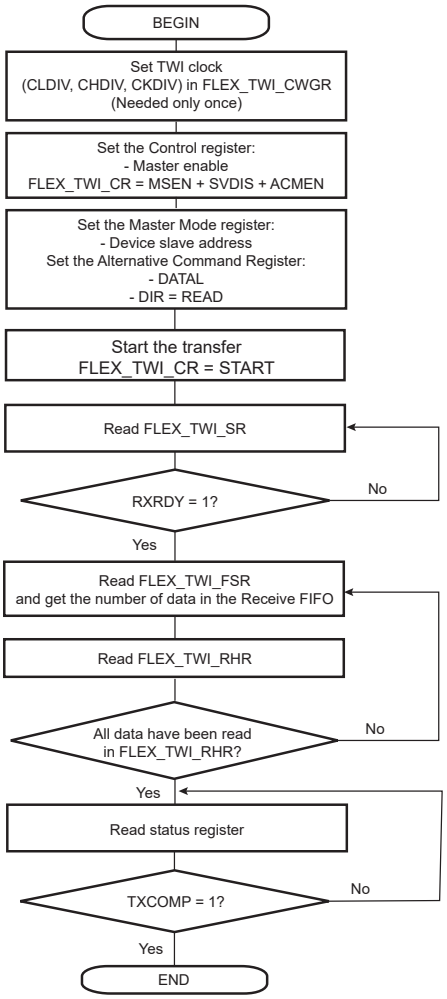
38.9.6.3 Receiving Data with FIFO Enabled

When the Receive FIFO is enabled, FLEX_TWI_RHR access reads the FIFO.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with FLEX_TWI_FLR.RXFL. All the data can be read successively in FLEX_TWI_RHR without checking the FLEX_TWI_SR.RXRDY flag between each access.

See figures Receiving Data with FIFO Enabled in Master Mode and Sending/Receiving Data with FIFO Enabled in Slave Mode.

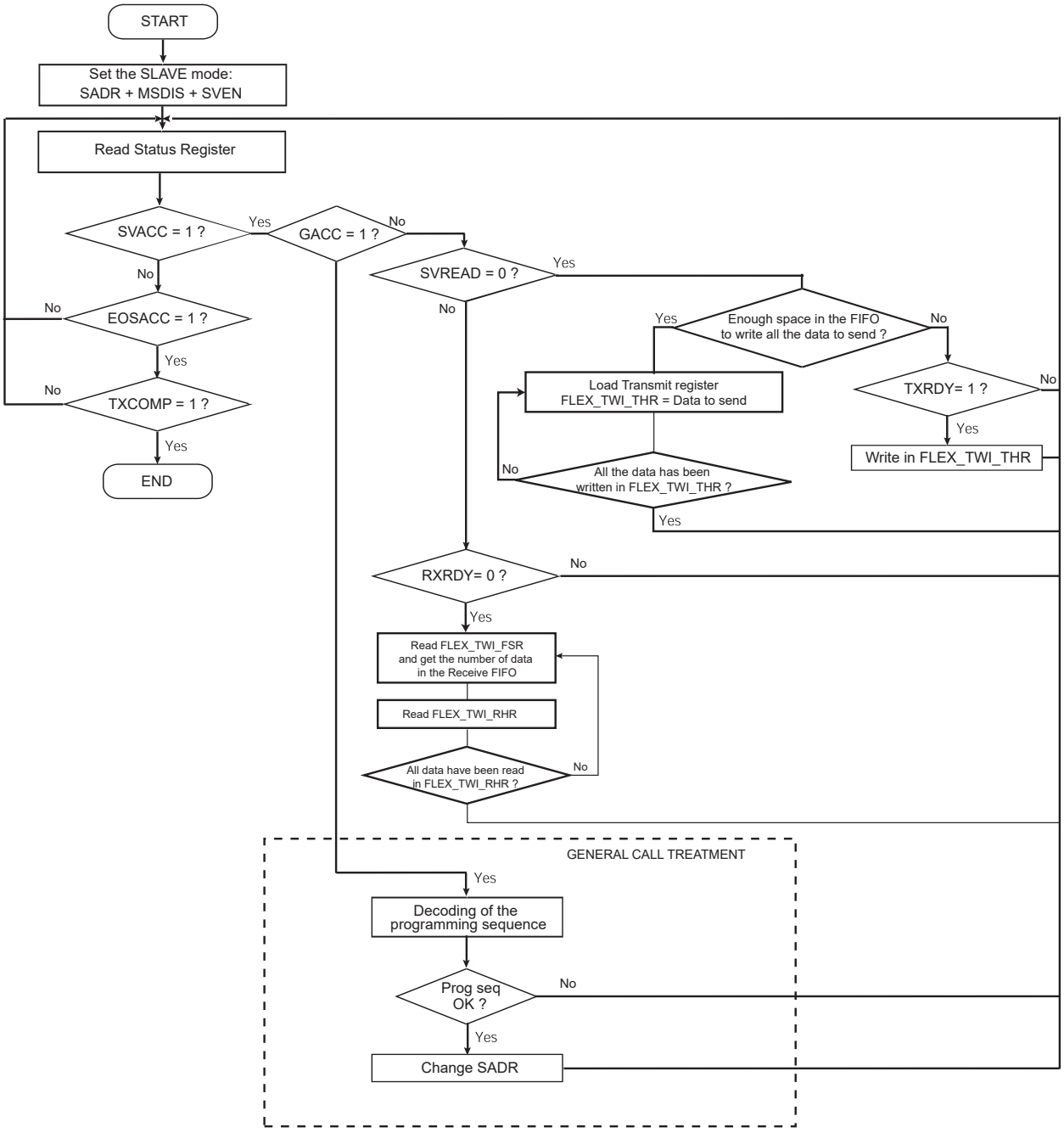
Figure 38-133. Receiving Data with FIFO Enabled in Master Mode



38.9.6.4 Sending/Receiving with FIFO Enabled in Slave Mode

See sections Sending Data with FIFO Enabled and Receiving Data with FIFO Enabled for details.

Figure 38-134. Sending/Receiving Data with FIFO Enabled in Slave Mode



38.9.6.5 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using FLEX_TWI_CR.TXFCLR/RXFCLR.

38.9.6.6 TXRDY and RXRDY Behavior

FLEX_TWI_SR.TXRDY/RXRDY flags display a specific behavior when FIFOs are enabled.

TXRDY indicates if a data can be written in the Transmit FIFO. Thus the TXRDY flag is set as long as the Transmit FIFO can accept new data. See figure TXRDY Behavior when TXRDYM = 0 in Master mode.

RXRDY indicates if an unread data is present in the Receive FIFO. Thus the RXRDY flag is set as soon as one unread data is in the Receive FIFO. Refer to figure RXRDY Behavior when RXRDYM = 0 in Master and Slave modes.

TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in the TWI FIFO Mode Register (FLEX_TWI_FMR) to reduce the number of accesses to FLEX_TWI_THR/RHR.

As an example, in Master mode, the Transmit FIFO can be loaded with multiple data in the same access by configuring TXRDYM>0.

See TWI FIFO Mode Register for the FIFO configuration.

Figure 38-135. TXRDY Behavior when TXRDYM = 0 in Master Mode

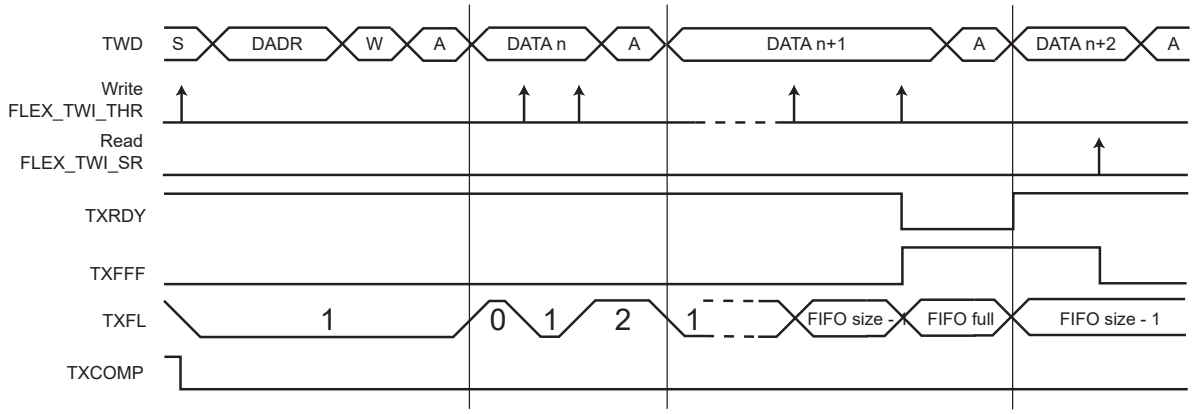


Figure 38-136. RXRDY Behavior when RXRDYM = 0 in Master and Slave Modes

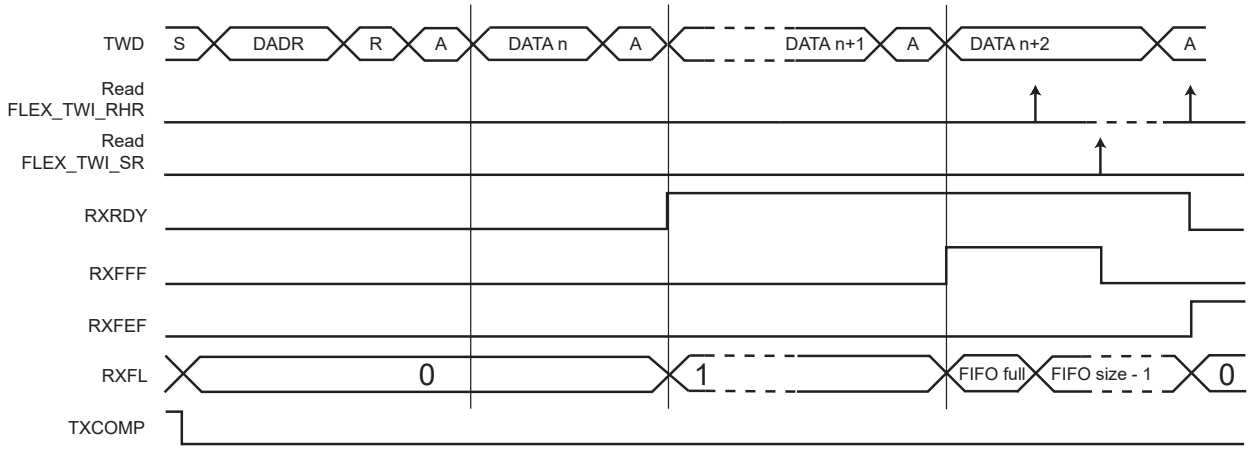
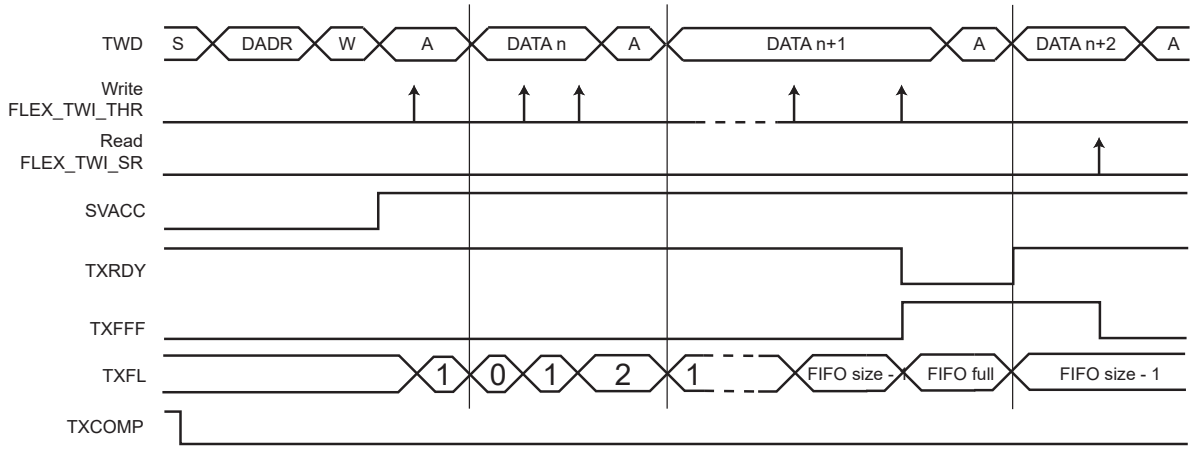


Figure 38-137. TXRDY Behavior when TXRDYM = 0 in Slave Mode



38.9.6.7 TWI Single Data Mode

In Single Data mode, only one data is written every time FLEX_TWI_THR is accessed, and only one data is read every time FLEX_TWI_RHR is accessed.

When FLEX_TWI_FMR.TXRDYM = 0, the Transmit FIFO operates in Single Data mode.

When FLEX_TWI_FMR.RXRDYM = 0, the Receive FIFO operates in Single Data mode.

See sections TWI Transmit Holding Register and TWI Receive Holding Register.

38.9.6.8 TWI Multiple Data Mode

Multiple Data mode minimizes the number of accesses by concatenating the data to send/read in one access.

When FLEX_TWI_FMR.TXRDYM > 0, the Transmit FIFO operates in Multiple Data mode.

When FLEX_TWI_FMR.RXRDYM > 0, the Receive FIFO operates in Multiple Data mode.

In Multiple Data mode, it is possible to write/read up to four data in one FLEX_TWI_THR/FLEX_TWI_RHR register access.

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read. If the access is a halfword size register access, then up to two data are read and only one data is written. Lastly, if the access is a wordsize register access, then up to four data are read and up to two data are written.

Written/Read data are always right-aligned, as described in sections TWI Receive Holding Register (FIFO Enabled) and TWI Transmit Holding Register (FIFO Enabled).

As an example, if the Transmit FIFO is empty and there are six data to send, either of the following write accesses may be performed:

- six FLEX_TWI_THR-byte write accesses
- three FLEX_TWI_THR-halfword write accesses
- one FLEX_TWI_THR-word write access and one FLEX_TWI_THR halfword write access

With a Receive FIFO containing six data, any of the following read accesses may be performed:

- six FLEX_TWI_RHR-byte read accesses
- three FLEX_TWI_RHR-halfword read accesses
- one FLEX_TWI_RHR-word read access and one FLEX_TWI_RHR-halfword read access

38.9.6.8.1 TXRDY and RXRDY Configuration

In Multiple Data mode, it is possible to write one or more data in the same FLEX_TWI_THR/FLEX_TWI_RHR access. The TXRDY flag indicates if one or more data can be written in the FIFO depending on the configuration of FLEX_TWI_FMR.TXRDYM/RXRDYM.

As an example, if two data are written each time in FLEX_TWI_THR, it is useful to configure the TXRDYM field to the value ‘1’ so that the TXRDY flag is at ‘1’ only when at least two data can be written in the Transmit FIFO.

In the same way, if four data are read each time in FLEX_TWI_RHR, it is useful to configure the RXRDYM field to the value ‘2’ so that the RXRDY flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

38.9.6.8.2 DMAC

When FIFOs operate in Multiple Data mode, the DMAC transfer type must be configured in byte, halfword or word depending on the FLEX_TWI_FMR.TXRDYM/RXRDYM settings.

38.9.6.9 Transmit FIFO Lock

If a frame is terminated early due to a not-acknowledge error (NACK flag), SMBus timeout error (TOUT flag) or master code acknowledge error (MACK flag), a lock is set on the Transmit FIFO preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, resetting DMAC channels, etc., without any risk.

FLEX_TWI_SR.LOCK is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared by setting FLEX_TWI_CR.TXFLCLR to ‘1’.

38.9.6.10 FIFO Pointer Error

A FIFO overflow is reported in FLEX_TWI_FSR.

If the Transmit FIFO is full and a write access is performed on FLEX_TWI_THR, it generates a Transmit FIFO pointer error and sets FLEX_TWI_FSR.TXFPTEF.

In Multiple Data mode, if the number of data written in FLEX_TWI_THR (according to the register access size) is greater than the free space in the Transmit FIFO, a Transmit FIFO pointer error is generated and FLEX_TWI_FSR.TXFPTEF is set.

A FIFO underflow is reported in FLEX_TWI_FSR.

In Multiple Data mode, if the number of data read in FLEX_TWI_RHR (according to the register access size) is greater than the number of unread data in the Receive FIFO, a Receive FIFO pointer error is generated and FLEX_TWI_FSR.RXFPTEF is set.

No pointer error occurs if the FIFO state/level is checked before writing/reading in FLEX_TWI_THR/FLEX_TWI_RHR. The FIFO state/level can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags may not behave as expected; their states should be ignored.

If a Transmit or Receive pointer error occurs, a software reset must be performed using FLEX_TWI_CR.SWRST. Note that issuing a software while transmitting might leave a slave in an unknown state holding the TWD line. In such case, a Bus Clear Command will allow to make the slave release the TWD line (the first frame sent afterwards might not be received properly by the slave).

38.9.6.11 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

The Transmit FIFO threshold can be set using the field FLEX_TWI_FMR.TXFTHRES. Each time the Transmit FIFO level goes from ‘above threshold’ to ‘equal to or below threshold’, the flag FLEX_TWI_FESR.TXFTHF is set. The application is warned that the Transmit FIFO has reached the defined threshold and that it can be reloaded.

The Receive FIFO threshold can be set using the field FLEX_TWI_FMR.RXFTHRES. Each time the Receive FIFO level goes from ‘below threshold’ to ‘equal to or above threshold’, the flag FLEX_TWI_FESR.RXFTHF is set. The application is warned that the Receive FIFO has reached the defined threshold and that it can be read to prevent an underflow.

The TXFTHF and RXFTHF flags can be configured to generate an interrupt using FLEX_TWI_FIER and FLEX_TWI_FIDR.

38.9.6.12 FIFO Flags

FIFOs come with a set of flags which can be configured to generate interrupts through FLEX_TWI_FIER and FLEX_TWI_FIDR.

FIFO flags state can be read in FLEX_TWI_FSR. They are cleared when FLEX_TWI_FSR is read.

38.9.7 Sniffer Mode

The Slave Sniffer mode can be used to put the TWI slave node in Sniffer mode. In this mode, the TWI reports all (or part of) the TWI bus activity without impacting the TWI bus. This mode is ideal for debugging purposes, or to log TWI network activity. Depending on the MASK field value, only some specific transfers can be logged instead of the whole activity.

The following fields must be programmed before entering Slave mode:

1. FLEX_TWI_SMR.SADR: Use the slave device address to indicate which frame(s) to log.
2. FLEX_TWI_SMR.MASK: Indicate which SADR bits should be masked and thus which transfers should be logged (set to 0x7F to log the whole TWI bus activity; all SADR bits are masked in this case). General Call accesses will always match.
3. FLEX_TWI_SMR.SNIFF: Set to ‘1’ to enable Slave Sniffer mode.
4. FLEX_TWI_CR.MSDIS: Disable Master mode.
5. FLEX_TWI_CR.SVEN: Enable Slave mode.

As the device receives the clock, values written in FLEX_TWI_CWGR are not taken into account.

Once configured in Slave Sniffer mode, the FLEX_TWI_SR.RXRDY bit indicates when a transfer has been logged in FLEX_TWI_RHR. An interrupt can be generated if configured. The FLEX_TWI_SR.OVRE flag indicates if an overrun error occurred if the application is not fast enough to read FLEX_TWI_RHR.

In Slave Sniffer mode, FLEX_TWI_RHR logs data as follows:

- The RXDATA field reports the sniffed 8-bit data field.
- The SSTATE field indicates if a START condition has been detected before the 8-bit data field.
- The PSTATE field indicates if a STOP condition has been detected after the previously sniffed 8-bit data field.
- The ASTATE field indicates which acknowledge condition has been detected after the previously sniffed 8-bit data field.

Figure 38-138. Sniffer Mode Application Overview

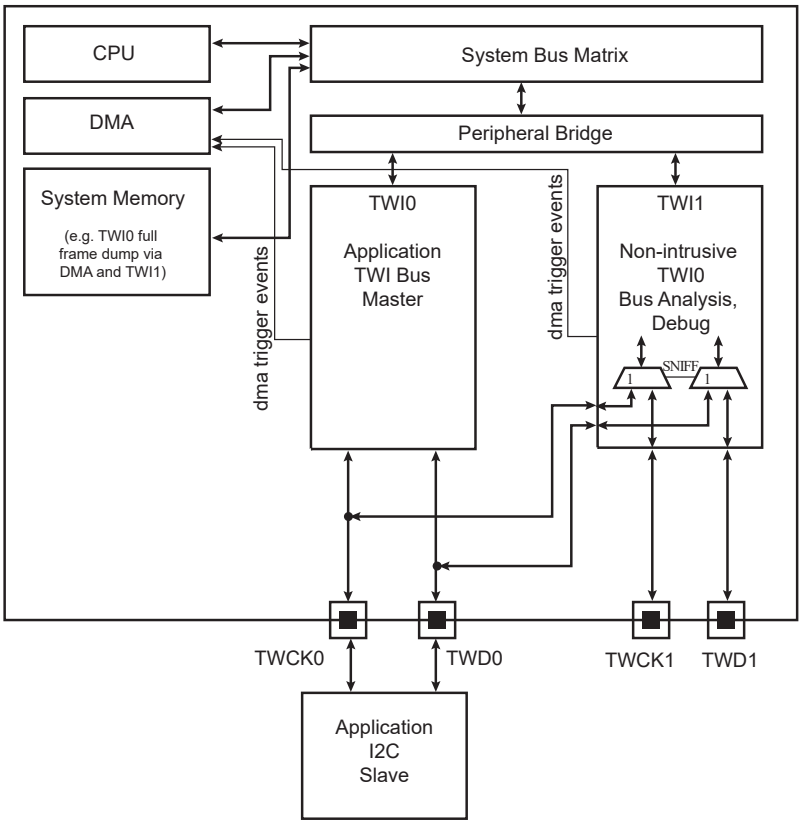
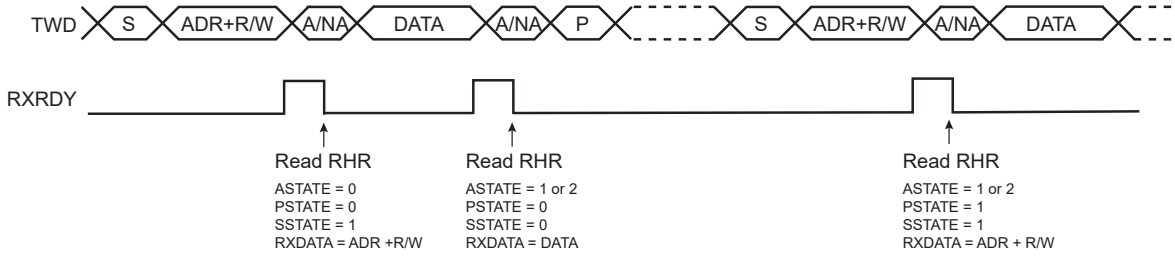


Figure 38-139. Slave Sniffer Mode Log



38.9.8 TWI Register Write Protection

The FLEXCOM operating mode (FLEX_MR.OPMODE) must be set to FLEX_MR_OPMODE_TWI to enable access to the write protection registers.

To prevent any single software error from corrupting TWI behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable), WPITEN (Write Protection Interrupt Enable), and/or WPCREN (Write Protection Control Enable) bits in the TWI Write Protection Mode Register (FLEX_TWI_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the TWI Write Protection Status Register (FLEX_TWI_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_TWI_WPSR.

The following register(s) can be write-protected when WPEN is set:

- TWI Slave Mode Register
- TWI Clock Waveform Generator Register
- TWI SMBus Timing Register
- TWI FIFO Mode Register

The following register(s) can be write-protected when WPITEN is set:

- TWI Interrupt Enable Register
- TWI Interrupt Disable Register

The following register(s) can be write-protected when WPCREN is set:

- TWI Control Register

38.10 Register Summary

Some registers have alternative views. Only the main views are listed in the Register Summary. The unlisted views are:

- FLEX_US_IER (LIN MODE)
- FLEX_US_IER (LON MODE)
- FLEX_US_IDR (LIN MODE)
- FLEX_US_IDR (LON MODE)
- FLEX_US_IMR (LIN MODE)
- FLEX_US_IMR (LON MODE)
- FLEX_US_CSR (LIN MODE)
- FLEX_US_CSR (LON MODE)
- FLEX_US_RHR (FIFO MULTI DATA)
- FLEX_US_THR (FIFO MULTI DATA)
- FLEX_US_TTGR (LON_MODE)
- FLEX_US_FIDI (LON_MODE)
- FLEX_SPI_RDR (FIFO MULTI DATA 8)
- FLEX_SPI_RDR (FIFO MULTI DATA 16)
- FLEX_SPI_TDR (FIFO MULTI DATA)
- FLEX_TWI_CR (FIFO ENABLED)
- FLEX_TWI_SR (FIFO ENABLED)
- FLEX_TWI_RHR (FIFO ENABLED)
- FLEX_TWI_THR (FIFO ENABLED)

Offset	Name	Bit Pos.								
0x00	FLEX_MR	7:0							OPMODE[1:0]	
		15:8								
		23:16								
		31:24								
0x04 ... 0x0F	Reserved									
0x10	FLEX_RHR	7:0	RXDATA[7:0]							
		15:8	RXDATA[15:8]							
		23:16								
		31:24								
0x14 ... 0x1F	Reserved									
0x20	FLEX_THR	7:0	TXDATA[7:0]							
		15:8	TXDATA[15:8]							
		23:16								
		31:24								
0x24 ... 0x01FF	Reserved									
0x0200	FLEX_US_CR	7:0	TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX		
		15:8	RETTO	RSTNACK	RSTIT	SENDA	STTTO	STPBRK	STTBRK	RSTSTA
		23:16			LINWKUP	LINABT	RTSDIS	RTSEN		
		31:24	FIFODIS	FIFOEN		REQCLR		TXFLCLR	RXFCLR	TXFCLR
0x0204	FLEX_US_MR	7:0	CHRL[1:0]		USCLKS[1:0]		USART_MODE[3:0]			
		15:8	CHMODE[1:0]		NBSTOP[1:0]		PAR[2:0]			SYNC
		23:16	INVDATA	VAR_SYNC	DSNACK	INACK	OVER	CLKO	MODE9	MSBF
		31:24	ONEBIT	MODSYNC	MAN	FILTER		MAX_ITERATION[2:0]		

.....continued

Offset	Name	Bit Pos.								
0x0208	FLEX_US_IER	7:0	PARE	FRAME	OVRE			RXBRK	TXRDY	RXRDY
		15:8			NACK			ITER	TXEMPTY	TIMEOUT
		23:16		CMP			CTSIC			
		31:24								MANE
0x0208	FLEX_US_IER (LIN_MODE)	7:0	PARE	FRAME	OVRE				TXRDY	RXRDY
		15:8	LINTC	LINID	LINBK				TXEMPTY	TIMEOUT
		23:16								
		31:24	LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	
0x0208	FLEX_US_IER (LON_MODE)	7:0	LCRCE	LSFE	OVRE				TXRDY	RXRDY
		15:8						UNRE	TXEMPTY	
		23:16								
		31:24				LBLOVFE	LRXD	LFET	LCOL	LTXD
0x020C	FLEX_US_IDR	7:0	PARE	FRAME	OVRE			RXBRK	TXRDY	RXRDY
		15:8			NACK			ITER	TXEMPTY	TIMEOUT
		23:16		CMP			CTSIC			
		31:24								MANE
0x020C	FLEX_US_IDR (LIN_MODE)	7:0	PARE	FRAME	OVRE				TXRDY	RXRDY
		15:8	LINTC	LINID	LINBK				TXEMPTY	TIMEOUT
		23:16								
		31:24	LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	
0x020C	FLEX_US_IDR (LON_MODE)	7:0	LCRCE	LSFE	OVRE				TXRDY	RXRDY
		15:8						UNRE	TXEMPTY	
		23:16								
		31:24				LBLOVFE	LRXD	LFET	LCOL	LTXD
0x0210	FLEX_US_IMR	7:0	PARE	FRAME	OVRE			RXBRK	TXRDY	RXRDY
		15:8			NACK			ITER	TXEMPTY	TIMEOUT
		23:16		CMP			CTSIC			
		31:24								MANE
0x0210	FLEX_US_IMR (LIN_MODE)	7:0	PARE	FRAME	OVRE				TXRDY	RXRDY
		15:8	LINTC	LINID	LINBK				TXEMPTY	TIMEOUT
		23:16								
		31:24	LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	
0x0210	FLEX_US_IMR (LON_MODE)	7:0	LCRCE	LSFE	OVRE				TXRDY	RXRDY
		15:8						UNRE	TXEMPTY	
		23:16								
		31:24				LBLOVFE	LRXD	LFET	LCOL	LTXD
0x0214	FLEX_US_CSR	7:0	PARE	FRAME	OVRE			RXBRK	TXRDY	RXRDY
		15:8			NACK			ITER	TXEMPTY	TIMEOUT
		23:16	CTS	CMP			CTSIC			
		31:24								MANE
0x0214	FLEX_US_CSR (LIN_MODE)	7:0	PARE	FRAME	OVRE				TXRDY	RXRDY
		15:8	LINTC	LINID	LINBK				TXEMPTY	TIMEOUT
		23:16	LINBLS							
		31:24	LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	
0x0214	FLEX_US_CSR (LON_MODE)	7:0	LCRCE	LSFE	OVRE				TXRDY	RXRDY
		15:8						UNRE	TXEMPTY	
		23:16								
		31:24				LBLOVFE	LRXD	LFET	LCOL	LTXD
0x0218	FLEX_US_RHR	7:0	RXCHR[7:0]							
		15:8	RXSYNH							RXCHR[8]
		23:16								
		31:24								
0x0218	FLEX_US_RHR (FIFO_MULTI_DATA)	7:0	RXCHR0[7:0]							
		15:8	RXCHR1[7:0]							
		23:16	RXCHR2[7:0]							
		31:24	RXCHR3[7:0]							

.....continued

Offset	Name	Bit Pos.								
0x021C	FLEX_US_THR	7:0	TXCHR[7:0]							
		15:8	TXSYNH							TXCHR[8]
		23:16								
		31:24								
0x021C	FLEX_US_THR (FIFO_MULTI_DATA)	7:0	TXCHR0[7:0]							
		15:8	TXCHR1[7:0]							
		23:16	TXCHR2[7:0]							
		31:24	TXCHR3[7:0]							
0x0220	FLEX_US_BRGR	7:0	CD[7:0]							
		15:8	CD[15:8]							
		23:16						FP[2:0]		
		31:24								
0x0224	FLEX_US_RTOR	7:0	TO[7:0]							
		15:8	TO[15:8]							
		23:16								TO[16]
		31:24								
0x0228	FLEX_US_TTGR	7:0	TG[7:0]							
		15:8								
		23:16								
		31:24								
0x0228	FLEX_US_TTGR (LON_MODE)	7:0	PCYCLE[7:0]							
		15:8	PCYCLE[15:8]							
		23:16	PCYCLE[23:16]							
		31:24								
0x022C ... 0x023F	Reserved									
0x0240	FLEX_US_FIDI	7:0	FI_DI_RATIO[7:0]							
		15:8	FI_DI_RATIO[15:8]							
		23:16								
		31:24								
0x0240	FLEX_US_FIDI (LON_MODE)	7:0	BETA2[7:0]							
		15:8	BETA2[15:8]							
		23:16	BETA2[23:16]							
		31:24								
0x0244	FLEX_US_NER	7:0	NB_ERRORS[7:0]							
		15:8								
		23:16								
		31:24								
0x0248 ... 0x024B	Reserved									
0x024C	FLEX_US_IF	7:0	IRDA_FILTER[7:0]							
		15:8								
		23:16								
		31:24								
0x0250	FLEX_US_MAN	7:0					TX_PL[3:0]			
		15:8				TX_MPOL			TX_PP[1:0]	
		23:16					RX_PL[3:0]			
		31:24	RXIDLEV	DRIFT	ONE	RX_MPOL			RX_PP[1:0]	
0x0254	FLEX_US_LINMR	7:0	WKUPTYP	FSDIS	DLM	CHKTYP	CHKDIS	PARDIS	NACT[1:0]	
		15:8	DLC[7:0]							
		23:16							SYNCDIS	PDCM
		31:24								
0x0258	FLEX_US_LINIR	7:0	IDCHR[7:0]							
		15:8								
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.									
0x025C	FLEX_US_LINBRR	7:0	LINCD[7:0]								
		15:8	LINCD[15:8]								
		23:16						LINFP[2:0]			
		31:24									
0x0260	FLEX_US_LONMR	7:0			LCDS	DMAM	CDTAIL	TCOL	COLDET	COMMT	
		15:8									
		23:16	EOFS[7:0]								
		31:24									
0x0264	FLEX_US_LONPR	7:0	LONPL[7:0]								
		15:8			LONPL[13:8]						
		23:16									
		31:24									
0x0268	FLEX_US_LONDL	7:0	LONDL[7:0]								
		15:8									
		23:16									
		31:24									
0x026C	FLEX_US_LONL2HDR	7:0	PB	ALTP	BLI[5:0]						
		15:8									
		23:16									
		31:24									
0x0270	FLEX_US_LONBL	7:0			LONBL[5:0]						
		15:8									
		23:16									
		31:24									
0x0274	FLEX_US_LONB1TX	7:0	BETA1TX[7:0]								
		15:8	BETA1TX[15:8]								
		23:16	BETA1TX[23:16]								
		31:24									
0x0278	FLEX_US_LONB1RX	7:0	BETA1RX[7:0]								
		15:8	BETA1RX[15:8]								
		23:16	BETA1RX[23:16]								
		31:24									
0x027C	FLEX_US_LONPRIO	7:0		PSNB[6:0]							
		15:8		NPS[6:0]							
		23:16									
		31:24									
0x0280	FLEX_US_IDTTX	7:0	IDTTX[7:0]								
		15:8	IDTTX[15:8]								
		23:16	IDTTX[23:16]								
		31:24									
0x0284	FLEX_US_IDTRX	7:0	IDTRX[7:0]								
		15:8	IDTRX[15:8]								
		23:16	IDTRX[23:16]								
		31:24									
0x0288	FLEX_US_ICDIFF	7:0					ICDIFF[3:0]				
		15:8									
		23:16									
		31:24									
0x028C ... 0x028F	Reserved										
0x0290	FLEX_US_CMPR	7:0	VAL1[7:0]								
		15:8		CMPPAR		CMPMODE				VAL1[8]	
		23:16	VAL2[7:0]								
		31:24								VAL2[8]	
0x0294 ... 0x029F	Reserved										

.....continued

Offset	Name	Bit Pos.								
0x02A0	FLEX_US_FMR	7:0	FRTSC		RXRDYM[1:0]				TXRDYM[1:0]	
		15:8			TXFTHRES[5:0]					
		23:16			RXFTHRES[5:0]					
		31:24			RXFTHRES2[5:0]					
0x02A4	FLEX_US_FLR	7:0			TXFL[5:0]					
		15:8								
		23:16			RXFL[5:0]					
		31:24								
0x02A8	FLEX_US_FIER	7:0	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
		15:8							RXFTHF2	
		23:16								
		31:24								
0x02AC	FLEX_US_FIDR	7:0	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
		15:8							RXFTHF2	
		23:16								
		31:24								
0x02B0	FLEX_US_FIMR	7:0	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
		15:8							RXFTHF2	
		23:16								
		31:24								
0x02B4	FLEX_US_FESR	7:0	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
		15:8							RXFTHF2	TXFLOCK
		23:16								
		31:24								
0x02B8 ... 0x02E3	Reserved									
0x02E4	FLEX_US_WPMR	7:0								WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0x02E8	FLEX_US_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16	WPVSRC[15:8]							
		31:24								
0x02EC ... 0x03FF	Reserved									
0x0400	FLEX_SPI_CR	7:0	SWRST						SPIDIS	SPIEN
		15:8				REQCLR				
		23:16							RXFCLR	TXFCLR
		31:24	FIFODIS	FIFOEN						LASTXFER
0x0404	FLEX_SPI_MR	7:0	LLB		WDRBT	MODFDIS	BRSRCCLK	PCSDEC	PS	MSTR
		15:8				CMPMODE				
		23:16					PCS[3:0]			
		31:24	DLYBCS[7:0]							
0x0408	FLEX_SPI_RDR	7:0	RD[7:0]							
		15:8	RD[15:8]							
		23:16					PCS[3:0]			
		31:24								
0x0408	FLEX_SPI_RDR (FIFO_MULTI_DATA_8)	7:0	RD0[7:0]							
		15:8	RD1[7:0]							
		23:16	RD2[7:0]							
		31:24	RD3[7:0]							
0x0408	FLEX_SPI_RDR (FIFO_MULTI_DATA_16)	7:0	RD0[7:0]							
		15:8	RD0[15:8]							
		23:16	RD1[7:0]							
		31:24	RD1[15:8]							

.....continued

Offset	Name	Bit Pos.								
0x040C	FLEX_SPI_TDR	7:0	TD[7:0]							
		15:8	TD[15:8]							
		23:16					PCS[3:0]			
		31:24								LASTXFER
0x040C	FLEX_SPI_TDR (FIFO_MULTI_DATA)	7:0	TD0[7:0]							
		15:8	TD0[15:8]							
		23:16	TD1[7:0]							
		31:24	TD1[15:8]							
0x0410	FLEX_SPI_SR	7:0					OVRES	MODF	TDRE	RDRF
		15:8				SFERR	CMP	UNDES	TXEMPTY	NSSR
		23:16								SPIENS
		31:24	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
0x0414	FLEX_SPI_IER	7:0					OVRES	MODF	TDRE	RDRF
		15:8					CMP	UNDES	TXEMPTY	NSSR
		23:16								
		31:24	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
0x0418	FLEX_SPI_IDR	7:0					OVRES	MODF	TDRE	RDRF
		15:8					CMP	UNDES	TXEMPTY	NSSR
		23:16								
		31:24	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
0x041C	FLEX_SPI_IMR	7:0					OVRES	MODF	TDRE	RDRF
		15:8					CMP	UNDES	TXEMPTY	NSSR
		23:16								
		31:24	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
0x0420 ... 0x042F	Reserved									
0x0430	FLEX_SPI_CSR0	7:0	BITS[3:0]				CSAAT	CSNAAT	NCPHA	CPOL
		15:8	SCBR[7:0]							
		23:16	DLYBS[7:0]							
		31:24	DLYBCT[7:0]							
0x0434	FLEX_SPI_CSR1	7:0	BITS[3:0]				CSAAT	CSNAAT	NCPHA	CPOL
		15:8	SCBR[7:0]							
		23:16	DLYBS[7:0]							
		31:24	DLYBCT[7:0]							
0x0438	FLEX_SPI_CSR2	7:0	BITS[3:0]				CSAAT	CSNAAT	NCPHA	CPOL
		15:8	SCBR[7:0]							
		23:16	DLYBS[7:0]							
		31:24	DLYBCT[7:0]							
0x043C	FLEX_SPI_CSR3	7:0	BITS[3:0]				CSAAT	CSNAAT	NCPHA	CPOL
		15:8	SCBR[7:0]							
		23:16	DLYBS[7:0]							
		31:24	DLYBCT[7:0]							
0x0440	FLEX_SPI_FMR	7:0			RXRDYM[1:0]				TXRDYM[1:0]	
		15:8								
		23:16			TXFTHRES[5:0]					
		31:24			RXFTHRES[5:0]					
0x0444	FLEX_SPI_FLR	7:0			TXFL[5:0]					
		15:8								
		23:16			RXFL[5:0]					
		31:24								
0x0448	FLEX_SPI_CMPR	7:0	VAL1[7:0]							
		15:8	VAL1[15:8]							
		23:16	VAL2[7:0]							
		31:24	VAL2[15:8]							
0x044C ... 0x04E3	Reserved									

.....continued

Offset	Name	Bit Pos.								
0x04E4	FLEX_SPI_WPMR	7:0						WPCREN	WPITEN	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0x04E8	FLEX_SPI_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16								
		31:24								
0x04EC ... 0x05FF	Reserved									
0x0600	FLEX_TWI_CR	7:0	SWRST	QUICK	SVDIS	SVEN	MSDIS	MSEN	STOP	START
		15:8	CLEAR	PECRQ	PECDIS	PECEN	SMBDIS	SMBEN	HSDIS	HSEN
		23:16							ACMDIS	ACMEN
		31:24			FIFODIS	FIFOEN		LOCKCLR		THRCLR
0x0600	FLEX_TWI_CR (FIFO_ENABLED)	7:0	SWRST	QUICK	SVDIS	SVEN	MSDIS	MSEN	STOP	START
		15:8	CLEAR	PECRQ	PECDIS	PECEN	SMBDIS	SMBEN	HSDIS	HSEN
		23:16							ACMDIS	ACMEN
		31:24			FIFODIS	FIFOEN		TXFLCLR	RXFCLR	TXFCLR
0x0604	FLEX_TWI_MMR	7:0								
		15:8				MREAD			IADRSZ[1:0]	
		23:16		DADR[6:0]						
		31:24								NOAP
0x0608	FLEX_TWI_SMR	7:0	SNIFF	SCLWSDIS		SADAT	SMHH	SMDA		NACKEN
		15:8		MASK[6:0]						
		23:16		SADR[6:0]						
		31:24								
0x060C	FLEX_TWI_IADR	7:0	IADR[7:0]							
		15:8	IADR[15:8]							
		23:16	IADR[23:16]							
		31:24								
0x0610	FLEX_TWI_CWGR	7:0	CLDIV[7:0]							
		15:8	CHDIV[7:0]							
		23:16				BRSRCCLK		CKDIV[2:0]		
		31:24			HOLD[5:0]					
0x0614 ... 0x061F	Reserved									
0x0620	FLEX_TWI_SR	7:0	UNRE	OVRE	GACC	SVACC	SVREAD	TXRDY	RXRDY	TXCOMP
		15:8					EOSACC	SCLWS	ARBLST	NACK
		23:16	LOCK		SMBHHM	SMBDAM	PECERR	TOUT		MCACK
		31:24						SR	SDA	SCL
0x0620	FLEX_TWI_SR (FIFO_ENABLED)	7:0	UNRE	OVRE	GACC	SVACC	SVREAD	TXRDY	RXRDY	TXCOMP
		15:8					EOSACC	SCLWS	ARBLST	NACK
		23:16	TXFLOCK		SMBHHM	SMBDAM	PECERR	TOUT		MCACK
		31:24						SR	SDA	SCL
0x0624	FLEX_TWI_IER	7:0	UNRE	OVRE	GACC	SVACC		TXRDY	RXRDY	TXCOMP
		15:8	TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK
		23:16			SMBHHM	SMBDAM	PECERR	TOUT		MCACK
		31:24								
0x0628	FLEX_TWI_IDR	7:0	UNRE	OVRE	GACC	SVACC		TXRDY	RXRDY	TXCOMP
		15:8	TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK
		23:16			SMBHHM	SMBDAM	PECERR	TOUT		MCACK
		31:24								
0x062C	FLEX_TWI_IMR	7:0	UNRE	OVRE	GACC	SVACC		TXRDY	RXRDY	TXCOMP
		15:8	TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK
		23:16			SMBHHM	SMBDAM	PECERR	TOUT		MCACK
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0630	FLEX_TWI_RHR	7:0	RXDATA[7:0]							
		15:8				ASTATE[1:0]		PSTATE	SSTATE[1:0]	
		23:16								
		31:24								
0x0630	FLEX_TWI_RHR (FIFO_ENABLED)	7:0	RXDATA0[7:0]							
		15:8	RXDATA1[7:0]							
		23:16	RXDATA2[7:0]							
		31:24	RXDATA3[7:0]							
0x0634	FLEX_TWI_THR	7:0	TXDATA[7:0]							
		15:8								
		23:16								
		31:24								
0x0634	FLEX_TWI_THR (FIFO_ENABLED)	7:0	TXDATA0[7:0]							
		15:8	TXDATA1[7:0]							
		23:16	TXDATA2[7:0]							
		31:24	TXDATA3[7:0]							
0x0638	FLEX_TWI_SMBTR	7:0					PRESC[3:0]			
		15:8	TLOWS[7:0]							
		23:16	TLOWM[7:0]							
		31:24	THMAX[7:0]							
0x063C ... 0x063F	Reserved									
0x0640	FLEX_TWI_ACR	7:0	DATAL[7:0]							
		15:8							PEC	DIR
		23:16	NDATAL[7:0]							
		31:24							NPEC	NDIR
0x0644	FLEX_TWI_FILTR	7:0						PADFCFG	PADFEN	FILT
		15:8						THRES[2:0]		
		23:16								
		31:24								
0x0648 ... 0x064F	Reserved									
0x0650	FLEX_TWI_FMR	7:0			RXRDYM[1:0]				TXRDYM[1:0]	
		15:8								
		23:16			TXFTHRES[5:0]					
		31:24			RXFTHRES[5:0]					
0x0654	FLEX_TWI_FLR	7:0			TXFL[5:0]					
		15:8								
		23:16			RXFL[5:0]					
		31:24								
0x0658 ... 0x065F	Reserved									
0x0660	FLEX_TWI_FSR	7:0	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
		15:8								
		23:16								
		31:24								
0x0664	FLEX_TWI_FIER	7:0	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
		15:8								
		23:16								
		31:24								
0x0668	FLEX_TWI_FIDR	7:0	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
		15:8								
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x066C	FLEX_TWI_FIMR	7:0	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
		15:8								
		23:16								
		31:24								
0x0670 ... 0x06E3	Reserved									
0x06E4	FLEX_TWI_WPMR	7:0						WPCREN	WPITEN	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0x06E8	FLEX_TWI_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16	WPVSRC[15:8]							
		31:24	WPVSRC[23:16]							

38.10.1 FLEXCOM Mode Register

Name: FLEX_MR
Offset: 0x000
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							OPMODE[1:0]	
Access							R/W	R/W
Reset							0	0

Bits 1:0 – OPMODE[1:0] FLEXCOM Operating Mode

Value	Name	Description
0	NO_COM	No communication
1	USART	All UART-related protocols are selected (RS232, RS485, IrDA, ISO7816, LIN, LON) SPI/TWI-related registers are not accessible and have no impact on IOs.
2	SPI	SPI operating mode is selected. USART/TWI related registers are not accessible and have no impact on IOs.
3	TWI	All TWI-related protocols are selected (TWI, SMBus). USART/SPI-related registers are not accessible and have no impact on IOs.

38.10.2 FLEXCOM Receive Holding Register

Name: FLEX_RHR
Offset: 0x010
Reset: 0x00000000
Property: Read-only

If enabled in the SFR module (default is disabled), a debugger read access does not clear the clear-on-read flags in FLEX_US_CSR, FLEX_SPI_SR, FLEX_TWI_SR.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RXDATA[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RXDATA[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – RXDATA[15:0] Receive Data
This register is a mirror of:
- USART Receive Holding Register (FLEX_US_RHR) if FLEX_MR.OPMODE field equals 1
- SPI Receive Data Register (FLEX_SPI_RDR) if FLEX_MR.OPMODE field equals 2
- TWI Transmit Holding Register (FLEX_TWI_RHR) if FLEX_MR.OPMODE field equals 3

38.10.3 FLEXCOM Transmit Holding Register

Name: FLEX_THR
Offset: 0x020
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TXDATA[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TXDATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – TXDATA[15:0] Transmit Data
This register is a mirror of:
- USART Transmit Holding Register (FLEX_US_THR) if FLEX_MR.OPMODE field equals 1
- SPI Transmit Data Register (FLEX_SPI_TDR) if FLEX_MR.OPMODE field equals 2
- TWI Transmit Holding Register (FLEX_TWI_THR) if FLEX_MR.OPMODE field equals 3

38.10.4 USART Control Register

Name: FLEX_US_CR
Offset: 0x200
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	FIFODIS	FIFOEN		REQCLR		TXFLCLR	RXFCLR	TXFCLR
Access	W	W		W		W	W	W
Reset	–	–		–		–	–	–

Bit	23	22	21	20	19	18	17	16
			LINWKUP	LINABT	RTSDIS	RTSEN		
Access			W	W	W	W		
Reset			–	–	–	–		

Bit	15	14	13	12	11	10	9	8
	RETTO	RSTNACK	RSTIT	SENDA	STTTO	STPBRK	STTBRK	RSTSTA
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX		
Access	W	W	W	W	W	W		
Reset	–	–	–	–	–	–		

Bit 31 – FIFODIS FIFO Disable

Value	Description
0	No effect.
1	Disables the Transmit and Receive FIFOs.

Bit 30 – FIFOEN FIFO Enable

Value	Description
0	No effect.
1	Enables the Transmit and Receive FIFOs.

Bit 28 – REQCLR Request to Clear the Comparison Trigger
0: No effect.
1: Restarts the comparison trigger to enable FLEX_US_RHR loading.

Bit 26 – TXFLCLR Transmit FIFO Lock CLEAR

Value	Description
0	No effect.
1	Clears the Transmit FIFO Lock.

Bit 25 – RXFCLR Receive FIFO Clear

Value	Description
0	No effect.
1	Empties the Receive FIFO.

Bit 24 – TXFCLR Transmit FIFO Clear

Value	Description
0	No effect.
1	Empties the Transmit FIFO.

Bit 21 – LINWKUP Send LIN Wakeup Signal

Value	Description
0	No effect:
1	Sends a wakeup signal on the LIN bus.

Bit 20 – LINABT Abort LIN Transmission

Value	Description
0	No effect.
1	Aborts the current LIN transmission.

Bit 19 – RTSDIS Request to Send Disable

Value	Description
0	No effect.
1	Drives the RTS pin to 0 if FLEX_US_MR.USART_MODE field = 2, else drives the RTS pin to 1 if FLEX_US_MR.USART_MODE field = 0.

Bit 18 – RTSEN Request to Send Enable

Value	Description
0	No effect.
1	Drives the RTS pin to 1 if FLEX_US_MR.USART_MODE field = 2, else drives the RTS pin to 0 if FLEX_US_MR.USART_MODE field = 0.

Bit 15 – RETTO Start Timeout Immediately

Value	Description
0	No effect
1	Immediately restarts timeout period.

Bit 14 – RSTNACK Reset Non Acknowledge

Value	Description
0	No effect
1	Resets FLEX_US_CSR.NACK.

Bit 13 – RSTIT Reset Iterations

Value	Description
0	No effect.
1	Resets FLEX_US_CSR.ITER. No effect if the ISO7816 is not enabled.

Bit 12 – SENDA Send Address

Value	Description
0	No effect.
1	In Multidrop mode only, the next character written to FLEX_US_THR is sent with the address bit set.

Bit 11 – STTTO Clear TIMEOUT Flag and Start Timeout After Next Character Received

Value	Description
0	No effect.
1	Starts waiting for a character before clocking the timeout counter. Immediately disables a timeout period in progress. Resets the FLEX_US_CSR.TIMEOUT status bit.

Bit 10 – STPBRK Stop Break

Value	Description
0	No effect.
1	Stops transmission of the break after a minimum of one character length and transmits a high level during 12-bit periods. No effect if no break is being transmitted.

Bit 9 – STTBRK Start Break

Value	Description
0	No effect.

Value	Description
1	Starts transmission of a break after the characters present in FLEX_US_THR and the Transmit Shift Register have been transmitted. No effect if a break is already being transmitted.

Bit 8 – RSTSTA Reset Status Bits

Value	Description
0	No effect.
1	Resets the PARE, FRAME, OVRE, MANE, LINBE, LINISFE, LINIPE, LINCE, LINSNRE, LINSTE, LINHTE, LINID, LINTC, LINBK, CMP and RXBRK in FLEX_US_CSR status bits, as well as the TXFEF, TXFFF, TXFTHF, RXFEF, RXFFF, RXFTHF, TXFPTEF, RXFPTEF in FLEX_US_FESR status bits.

Bit 7 – TXDIS Transmitter Disable

Value	Description
0	No effect.
1	Disables the transmitter.

Bit 6 – TXEN Transmitter Enable

Value	Description
0	No effect.
1	Enables the transmitter if TXDIS is 0.

Bit 5 – RXDIS Receiver Disable

Value	Description
0	No effect.
1	Disables the receiver.

Bit 4 – RXEN Receiver Enable

Value	Description
0	No effect.
1	Enables the receiver, if RXDIS is 0.

Bit 3 – RSTTX Reset Transmitter

Value	Description
0	No effect.
1	Resets the transmitter.

Bit 2 – RSTRX Reset Receiver

Value	Description
0	No effect.
1	Resets the receiver.

38.10.5 USART Mode Register

Name: FLEX_US_MR
Offset: 0x204
Reset: –
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	ONEBIT	MODSYNC	MAN	FILTER		MAX_ITERATION[2:0]		
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset	–	–	–	–		–	–	–

Bit	23	22	21	20	19	18	17	16
	INVDATA	VAR_SYNC	DSNACK	INACK	OVER	CLKO	MODE9	MSBF
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	CHMODE[1:0]		NBSTOP[1:0]		PAR[2:0]			SYNC
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	CHRL[1:0]		USCLKS[1:0]		USART_MODE[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	–	–	–	–	–	–	–	–

Bit 31 – ONEBIT Start Frame Delimiter Selector

Value	Description
0	Start frame delimiter is COMMAND or DATA SYNC.
1	Start frame delimiter is one bit.

Bit 30 – MODSYNC Manchester Synchronization Mode

Value	Description
0	The Manchester start bit is a 0 to 1 transition
1	The Manchester start bit is a 1 to 0 transition.

Bit 29 – MAN Manchester Encoder/Decoder Enable

Value	Description
0	Manchester encoder/decoder are disabled.
1	Manchester encoder/decoder are enabled.

Bit 28 – FILTER Receive Line Filter

Value	Description
0	The USART does not filter the receive line.
1	The USART filters the receive line using a three-sample filter (1/16-bit clock) (2 over 3 majority).

Bits 26:24 – MAX_ITERATION[2:0] Maximum Number of Automatic Iterations

Value	Description
0–7	Defines the maximum number of iterations in mode ISO7816, protocol T = 0.

Bit 23 – INVDATA Inverted Data

Value	Description
0	The data field transmitted on TXD line is the same as the one written in FLEX_US_THR or the content read in FLEX_US_RHR is the same as RXD line. Normal mode of operation.

Value	Description
1	The data field transmitted on TXD line is inverted (voltage polarity only) compared to the value written in FLEX_US_THR or the content read in FLEX_US_RHR is inverted compared to what is received on RXD line (or ISO7816 IO line). Inverted mode of operation, useful for contactless card application. To be used with configuration bit MSBF.

Bit 22 – VAR_SYNC Variable Synchronization of Command/Data Sync Start Frame Delimiter

Value	Description
0	User defined configuration of command or data sync field depending on MODSYNC value.
1	The sync field is updated when a character is written into FLEX_US_THR.

Bit 21 – DSNACK Disable Successive NACK

The MAX_ITERATION field must be cleared if DSNACK is cleared.

Value	Description
0	NACK is sent on the ISO line as soon as a parity error occurs in the received character (unless INACK is set).
1	Successive parity errors are counted up to the value specified in the MAX_ITERATION field. These parity errors generate a NACK on the ISO line. As soon as this value is reached, no additional NACK is sent on the ISO line. The flag ITER is asserted.

Bit 20 – INACK Inhibit Non Acknowledge

Value	Description
0	The NACK is generated.
1	The NACK is not generated.

Bit 19 – OVER Oversampling Mode

Value	Description
0	16x Oversampling.
1	8x Oversampling.

Bit 18 – CLKO Clock Output Select

Value	Description
0	The USART does not drive the SCK pin (Synchronous Slave mode or Asynchronous mode with external baud rate clock source).
1	The USART drives the SCK pin if USCLKS does not select the external clock SCK (USART Synchronous Master mode).

Bit 17 – MODE9 9-bit Character Length

Value	Description
0	CHRL defines character length.
1	9-bit character length.

Bit 16 – MSBF Bit Order

Value	Description
0	Least significant bit is sent/received first.
1	Most significant bit is sent/received first.

Bits 15:14 – CHMODE[1:0] Channel Mode

Value	Name	Description
0	NORMAL	Normal mode
1	AUTOMATIC	Automatic Echo. Receiver input is connected to the TXD pin.
2	LOCAL_LOOPBACK	Local Loopback. Transmitter output is connected to the Receiver Input.
3	REMOTE_LOOPBACK	Remote Loopback. RXD pin is internally connected to the TXD pin.

Bits 13:12 – NBSTOP[1:0] Number of Stop Bits

Value	Name	Description
0	1_BIT	1 stop bit

Value	Name	Description
1	1_5_BIT	1.5 stop bit (SYNC = 0) or reserved (SYNC = 1)
2	2_BIT	2 stop bits

Bits 11:9 – PAR[2:0] Parity Type

Value	Name	Description
0	EVEN	Even parity
1	ODD	Odd parity
2	SPACE	Parity forced to 0 (Space)
3	MARK	Parity forced to 1 (Mark)
4	NO	No parity
6	MULTIDROP	Multidrop mode

Bit 8 – SYNC Synchronous Mode Select

Value	Description
0	USART operates in Asynchronous mode (UART).
1	USART operates in Synchronous mode.

Bits 7:6 – CHRL[1:0] Character Length

Value	Name	Description
0	5_BIT	Character length is 5 bits
1	6_BIT	Character length is 6 bits
2	7_BIT	Character length is 7 bits
3	8_BIT	Character length is 8 bits

Bits 5:4 – USCLKS[1:0] Clock Selection

Value	Name	Description
0	MCK	Peripheral clock is selected
1	DIV	Peripheral clock divided (DIV = 8) is selected
2	GCLK	PMC generic clock is selected. If the SCK pin is driven (CLKO = 1), the CD field must be greater than 1.
3	SCK	External pin SCK is selected

Bits 3:0 – USART_MODE[3:0] USART Mode of Operation

Value	Name	Description
0x0	NORMAL	Normal mode
0x1	RS485	RS485
0x2	HW_HANDSHAKING	Hardware handshaking
0x3	MODEM	Modem
0x4	IS07816_T_0	IS07816 Protocol: T = 0
0x6	IS07816_T_1	IS07816 Protocol: T = 1
0x8	IRDA	IrDA
0x9	LON	LON
0xA	LIN_MASTER	LIN Master mode
0xB	LIN_SLAVE	LIN Slave mode
0xC	DATA16BIT_MASTER	16-bit data master
0xD	DATA16BIT_SLAVE	16-bit data slave
0xE	SPI_MASTER	SPI Master mode (CLKO must be written to 1 and USCLKS = 0, 1 or 2)
0xF	SPI_SLAVE	SPI Slave mode

38.10.6 USART Interrupt Enable Register

Name: FLEX_US_IER
Offset: 0x208
Reset: –
Property: Write-only

For LIN-specific configurations, see USART Interrupt Enable Register (LIN_MODE).

For LON-specific configurations, see USART Interrupt Enable Register (LON_MODE).

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
								MANE
Access								W
Reset								–

Bit	23	22	21	20	19	18	17	16
		CMP			CTSIC			
Access		W			W			
Reset		–			–			

Bit	15	14	13	12	11	10	9	8
			NACK			ITER	TXEMPTY	TIMEOUT
Access			W			W	W	W
Reset			–			–	–	–

Bit	7	6	5	4	3	2	1	0
	PARE	FRAME	OVRE			RXBRK	TXRDY	RXRDY
Access	W	W	W			W	W	W
Reset	–	–	–			–	–	–

Bit 24 – MANE Manchester Error Interrupt Enable

Bit 22 – CMP Comparison Interrupt Enable

Bit 19 – CTSIC Clear to Send Input Change Interrupt Enable

Bit 13 – NACK Non Acknowledge Interrupt Enable

Bit 10 – ITER Max number of Repetitions Reached Interrupt Enable

Bit 9 – TXEMPTY TXEMPTY Interrupt Enable

Bit 8 – TIMEOUT Timeout Interrupt Enable

Bit 7 – PARE Parity Error Interrupt Enable

Bit 6 – FRAME Framing Error Interrupt Enable

Bit 5 – OVRE Overrun Error Interrupt Enable

Bit 2 – RXBRK Receiver Break Interrupt Enable

Bit 1 – TXRDY TXRDY Interrupt Enable

Bit 0 – RXRDY RXRDY Interrupt Enable

38.10.7 USART Interrupt Enable Register (LIN_MODE)

Name: FLEX_US_IER (LIN_MODE)
Offset: 0x208
Reset: –
Property: Write-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

This register can only be written if the WPITEN bit is cleared in the USART Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
	LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	
Access	W	W	W	W	W	W	W	
Reset	–	–	–	–	–	–	–	

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	LINTC	LINID	LINBK				TXEMPTY	TIMEOUT
Access	W	W	W				W	W
Reset	–	–	–				–	–

Bit	7	6	5	4	3	2	1	0
	PARE	FRAME	OVRE				TXRDY	RXRDY
Access	W	W	W				W	W
Reset	–	–	–				–	–

Bit 31 – LINHTE LIN Header Timeout Error Interrupt Enable

Bit 30 – LINSTE LIN Synch Tolerance Error Interrupt Enable

Bit 29 – LINSNRE LIN Slave Not Responding Error Interrupt Enable

Bit 28 – LINCE LIN Checksum Error Interrupt Enable

Bit 27 – LINIPE LIN Identifier Parity Interrupt Enable

Bit 26 – LINISFE LIN Inconsistent Synch Field Error Interrupt Enable

Bit 25 – LINBE LIN Bus Error Interrupt Enable

Bit 15 – LINTC LIN Transfer Completed Interrupt Enable

Bit 14 – LINID LIN Identifier Sent or LIN Identifier Received Interrupt Enable

Bit 13 – LINBK LIN Break Sent or LIN Break Received Interrupt Enable

Bit 9 – TXEMPTY TXEMPTY Interrupt Enable

Bit 8 – TIMEOUT Timeout Interrupt Enable

Bit 7 – PARE Parity Error Interrupt Enable

Bit 6 – FRAME Framing Error Interrupt Enable

Bit 5 – OVRE Overrun Error Interrupt Enable

Bit 1 – TXRDY TXRDY Interrupt Enable

Bit 0 – RXRDY RXRDY Interrupt Enable

38.10.8 USART Interrupt Enable Register (LON_MODE)

Name: FLEX_US_IER (LON_MODE)
Offset: 0x208
Reset: –
Property: Write-only

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPITEN bit is cleared in the USART Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
				LBLOVFE	LRXD	LFET	LCOL	LTXD
Access				W	W	W	W	W
Reset				–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						UNRE	TXEMPTY	
Access						W	W	
Reset						–	–	

Bit	7	6	5	4	3	2	1	0
	LCRCE	LSFE	OVRE				TXRDY	RXRDY
Access	W	W	W				W	W
Reset	–	–	–				–	–

Bit 28 – LBLOVFE LON Backlog Overflow Error Interrupt Enable

Bit 27 – LRXD LON Reception Done Interrupt Enable

Bit 26 – LFET LON Frame Early Termination Interrupt Enable

Bit 25 – LCOL LON Collision Interrupt Enable

Bit 24 – LTXD LON Transmission Done Interrupt Enable

Bit 10 – UNRE Underrun Error Interrupt Enable

Bit 9 – TXEMPTY TXEMPTY Interrupt Enable

Bit 7 – LCRCE LON CRC Error Interrupt Enable

Bit 6 – LSFE LON Short Frame Error Interrupt Enable

Bit 5 – OVRE Overrun Error Interrupt Enable

Bit 1 – TXRDY TXRDY Interrupt Enable

Bit 0 – RXRDY RXRDY Interrupt Enable

38.10.9 USART Interrupt Disable Register

Name: FLEX_US_IDR
Offset: 0x20C
Reset: –
Property: Write-only

For LIN-specific configurations, see USART Interrupt Disable Register (LIN_MODE).

For LON-specific configurations, see USART Interrupt Disable Register (LON_MODE).

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
								MANE
Access								W
Reset								–

Bit	23	22	21	20	19	18	17	16
		CMP			CTSIC			
Access		W			W			
Reset		–			–			

Bit	15	14	13	12	11	10	9	8
			NACK			ITER	TXEMPTY	TIMEOUT
Access			W			W	W	W
Reset			–			–	–	–

Bit	7	6	5	4	3	2	1	0
	PARE	FRAME	OVRE			RXBRK	TXRDY	RXRDY
Access	W	W	W			W	W	W
Reset	–	–	–			–	–	–

Bit 24 – MANE Manchester Error Interrupt Disable

Bit 22 – CMP Comparison Interrupt Disable

Bit 19 – CTSIC Clear to Send Input Change Interrupt Disable

Bit 13 – NACK Non Acknowledge Interrupt Disable

Bit 10 – ITER Max Number of Repetitions Reached Interrupt Disable

Bit 9 – TXEMPTY TXEMPTY Interrupt Disable

Bit 8 – TIMEOUT Timeout Interrupt Disable

Bit 7 – PARE Parity Error Interrupt Disable

Bit 6 – FRAME Framing Error Interrupt Disable

Bit 5 – OVRE Overrun Error Interrupt Enable

Bit 2 – RXBRK Receiver Break Interrupt Disable

Bit 1 – TXRDY TXRDY Interrupt Disable

Bit 0 – RXRDY RXRDY Interrupt Disable

38.10.10 USART Interrupt Disable Register (LIN_MODE)

Name: FLEX_US_IDR (LIN_MODE)
Offset: 0x20C
Reset: –
Property: Write-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

This register can only be written if the WPITEN bit is cleared in the USART Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
	LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	
Access	W	W	W	W	W	W	W	
Reset	–	–	–	–	–	–	–	

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	LINTC	LINID	LINBK				TXEMPTY	TIMEOUT
Access	W	W	W				W	W
Reset	–	–	–				–	–

Bit	7	6	5	4	3	2	1	0
	PARE	FRAME	OVRE				TXRDY	RXRDY
Access	W	W	W				W	W
Reset	–	–	–				–	–

Bit 31 – LINHTE LIN Header Timeout Error Interrupt Disable

Bit 30 – LINSTE LIN Synch Tolerance Error Interrupt Disable

Bit 29 – LINSNRE LIN Slave Not Responding Error Interrupt Disable

Bit 28 – LINCE LIN Checksum Error Interrupt Disable

Bit 27 – LINIPE LIN Identifier Parity Interrupt Disable

Bit 26 – LINISFE LIN Inconsistent Synch Field Error Interrupt Disable

Bit 25 – LINBE LIN Bus Error Interrupt Disable

Bit 15 – LINTC LIN Transfer Completed Interrupt Disable

Bit 14 – LINID LIN Identifier Sent or LIN Identifier Received Interrupt Disable

Bit 13 – LINBK LIN Break Sent or LIN Break Received Interrupt Disable

Bit 9 – TXEMPTY TXEMPTY Interrupt Disable

Bit 8 – TIMEOUT Timeout Interrupt Disable

Bit 7 – PARE Parity Error Interrupt Disable

Bit 6 – FRAME Framing Error Interrupt Disable

Bit 5 – OVRE Overrun Error Interrupt Disable

Bit 1 – TXRDY TXRDY Interrupt Disable

Bit 0 – RXRDY RXRDY Interrupt Disable

38.10.11 USART Interrupt Disable Register (LON_MODE)

Name: FLEX_US_IDR (LON_MODE)
Offset: 0x20C
Reset: –
Property: Write-only

This configuration is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPITEN bit is cleared in the USART Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
				LBLOVFE	LRXD	LFET	LCOL	LTXD
Access				W	W	W	W	W
Reset				–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						UNRE	TXEMPTY	
Access						W	W	
Reset						–	–	

Bit	7	6	5	4	3	2	1	0
	LCRCE	LSFE	OVRE				TXRDY	RXRDY
Access	W	W	W				W	W
Reset	–	–	–				–	–

Bit 28 – LBLOVFE LON Backlog Overflow Error Interrupt Disable

Bit 27 – LRXD LON Reception Done Interrupt Disable

Bit 26 – LFET LON Frame Early Termination Interrupt Disable

Bit 25 – LCOL LON Collision Interrupt Disable

Bit 24 – LTXD LON Transmission Done Interrupt Disable

Bit 10 – UNRE Underrun Error Interrupt Disable

Bit 9 – TXEMPTY TXEMPTY Interrupt Disable

Bit 7 – LCRCE LON CRC Error Interrupt Disable

Bit 6 – LSFE LON Short Frame Error Interrupt Disable

Bit 5 – OVRE Overrun Error Interrupt Disable

Bit 1 – TXRDY TXRDY Interrupt Disable

Bit 0 – RXRDY RXRDY Interrupt Disable

38.10.12 USART Interrupt Mask Register

Name: FLEX_US_IMR
Offset: 0x210
Reset: 0x00000000
Property: Read-only

For LIN-specific configurations, see USART Interrupt Mask Register (LIN_MODE).

For LON-specific configurations, see USART Interrupt Mask Register (LON_MODE).

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
								MANE
Access								R
Reset								0

Bit	23	22	21	20	19	18	17	16
		CMP			CTSIC			
Access		R			R			
Reset		0			0			

Bit	15	14	13	12	11	10	9	8
			NACK			ITER	TXEMPTY	TIMEOUT
Access			R			R	R	R
Reset			0			0	0	0

Bit	7	6	5	4	3	2	1	0
	PARE	FRAME	OVRE			RXBRK	TXRDY	RXRDY
Access	R	R	R			R	R	R
Reset	0	0	0			0	0	0

Bit 24 – MANE Manchester Error Interrupt Mask

Bit 22 – CMP Comparison Interrupt Mask

Bit 19 – CTSIC Clear to Send Input Change Interrupt Mask

Bit 13 – NACK Non Acknowledge Interrupt Mask

Bit 10 – ITER Max Number of Repetitions Reached Interrupt Mask

Bit 9 – TXEMPTY TXEMPTY Interrupt Mask

Bit 8 – TIMEOUT Timeout Interrupt Mask

Bit 7 – PARE Parity Error Interrupt Mask

Bit 6 – FRAME Framing Error Interrupt Mask

Bit 5 – OVRE Overrun Error Interrupt Mask

Bit 2 – RXBRK Receiver Break Interrupt Mask

Bit 1 – TXRDY TXRDY Interrupt Mask

Bit 0 – RXRDY RXRDY Interrupt Mask

38.10.13 USART Interrupt Mask Register (LIN_MODE)

Name: FLEX_US_IMR (LIN_MODE)
Offset: 0x210
Reset: 0x00000000
Property: Read-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
	LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	
Access	R	R	R	R	R	R	R	
Reset	0	0	0	0	0	0	0	

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	LINTC	LINID	LINBK				TXEMPTY	TIMEOUT
Access	R	R	R				R	R
Reset	0	0	0				0	0

Bit	7	6	5	4	3	2	1	0
	PARE	FRAME	OVRE				TXRDY	RXRDY
Access	R	R	R				R	R
Reset	0	0	0				0	0

Bit 31 – LINHTE LIN Header Timeout Error Interrupt Mask

Bit 30 – LINSTE LIN Synch Tolerance Error Interrupt Mask

Bit 29 – LINSNRE LIN Slave Not Responding Error Interrupt Mask

Bit 28 – LINCE LIN Checksum Error Interrupt Mask

Bit 27 – LINIPE LIN Identifier Parity Interrupt Mask

Bit 26 – LINISFE LIN Inconsistent Synch Field Error Interrupt Mask

Bit 25 – LINBE LIN Bus Error Interrupt Mask

Bit 15 – LINTC LIN Transfer Completed Interrupt Mask

Bit 14 – LINID LIN Identifier Sent or LIN Identifier Received Interrupt Mask

Bit 13 – LINBK LIN Break Sent or LIN Break Received Interrupt Mask

Bit 9 – TXEMPTY TXEMPTY Interrupt Mask

Bit 8 – TIMEOUT Timeout Interrupt Mask

Bit 7 – PARE Parity Error Interrupt Mask

Bit 6 – FRAME Framing Error Interrupt Mask

Bit 5 – OVRE Overrun Error Interrupt Mask

Bit 1 – TXRDY TXRDY Interrupt Mask

Bit 0 – RXRDY RXRDY Interrupt Mask

38.10.14 USART Interrupt Mask Register (LON_MODE)

Name: FLEX_US_IMR (LON_MODE)
Offset: 0x210
Reset: 0x00000000
Property: Read-only

This configuration is relevant only if USART_MODE = 0x9 in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
				LBLOVFE	LRXD	LFET	LCOL	LTXD
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						UNRE	TXEMPTY	
Access						R	R	
Reset						0	0	

Bit	7	6	5	4	3	2	1	0
	LCRCE	LSFE	OVRE				TXRDY	RXRDY
Access	R	R	R				R	R
Reset	0	0	0				0	0

Bit 28 – LBLOVFE LON Backlog Overflow Error Interrupt Mask

Bit 27 – LRXD LON Reception Done Interrupt Mask

Bit 26 – LFET LON Frame Early Termination Interrupt Mask

Bit 25 – LCOL LON Collision Interrupt Mask

Bit 24 – LTXD LON Transmission Done Interrupt Mask

Bit 10 – UNRE Underrun Error Interrupt Mask

Bit 9 – TXEMPTY TXEMPTY Interrupt Mask

Bit 7 – LCRCE LON CRC Error Interrupt Mask

Bit 6 – LSFE LON Short Frame Error Interrupt Mask

Bit 5 – OVRE Overrun Error Interrupt Mask

Bit 1 – TXRDY TXRDY Interrupt Mask

Bit 0 – RXRDY RXRDY Interrupt Mask

38.10.15 USART Channel Status Register

Name: FLEX_US_CSR
Offset: 0x214
Reset: 0x00000000
Property: Read-only

If enabled in the SFR module (default is disabled), a debugger read access does not clear the flag CTSIC.

For LIN-specific configurations, see USART Channel Status Register (LIN_MODE).

For LON-specific configurations, see USART Channel Status Register (LON_MODE).

Bit	31	30	29	28	27	26	25	24
								MANE
Access								R
Reset								–

Bit	23	22	21	20	19	18	17	16
	CTS	CMP			CTSIC			
Access	R	R			R			
Reset	–	–			–			

Bit	15	14	13	12	11	10	9	8
			NACK			ITER	TXEMPTY	TIMEOUT
Access			R			R	R	R
Reset			–			–	–	–

Bit	7	6	5	4	3	2	1	0
	PARE	FRAME	OVRE			RXBRK	TXRDY	RXRDY
Access	R	R	R			R	R	R
Reset	–	–	–			–	–	–

Bit 24 – MANE Manchester Error

Value	Description
0	No Manchester error has been detected since the last RSTSTA command was issued.
1	At least one Manchester error has been detected since the last RSTSTA command was issued.

Bit 23 – CTS Image of CTS Input

Value	Description
0	CTS input is driven low.
1	CTS input is driven high.

Bit 22 – CMP Comparison Status

Value	Description
0	No received character matched the comparison criteria programmed in VAL1, VAL2 fields and CMPPAR bit in since the last RSTSTA command was issued.
1	A received character matched the comparison criteria since the last RSTSTA command was issued.

Bit 19 – CTSIC Clear to Send Input Change Flag

Value	Description
0	No input change has been detected on the CTS pin since the last read of FLEX_US_CSR.
1	At least one input change has been detected on the CTS pin since the last read of FLEX_US_CSR.

Bit 13 – NACK Non Acknowledge Interrupt

Value	Description
0	Non acknowledge has not been detected since the last RSTNACK.
1	At least one non acknowledge has been detected since the last RSTNACK.

Bit 10 – ITER Max Number of Repetitions Reached

Value	Description
0	Maximum number of repetitions has not been reached since the last RSTIT command was issued.
1	Maximum number of repetitions has been reached since the last RSTIT command was issued.

Bit 9 – TXEMPTY Transmitter Empty (cleared by writing FLEX_US_THR)

Value	Description
0	There are characters in either FLEX_US_THR or the Transmit Shift Register, or the transmitter is disabled.
1	There are no characters in FLEX_US_THR, nor in the Transmit Shift Register.

Bit 8 – TIMEOUT Receiver Timeout

Value	Description
0	There has not been a timeout since the last Start Timeout command (FLEX_US_CR.STTTO) or the Timeout Register is 0.
1	There has been a timeout since the last Start Timeout command (FLEX_US_CR.STTTO).

Bit 7 – PARE Parity Error

Value	Description
0	No parity error has been detected since the last RSTSTA command was issued.
1	At least one parity error has been detected since the last RSTSTA command was issued.

Bit 6 – FRAME Framing Error

Value	Description
0	No stop bit has been detected low since the last RSTSTA command was issued.
1	At least one stop bit has been detected low since the last RSTSTA command was issued.

Bit 5 – OVRE Overrun Error

Value	Description
0	No overrun error has occurred since the last RSTSTA command was issued.
1	At least one overrun error has occurred since the last RSTSTA command was issued.

Bit 2 – RXBRK Break Received/End of Break

Value	Description
0	No break received or end of break detected since the last RSTSTA command was issued.
1	Break received or end of break detected since the last RSTSTA command was issued.

Bit 1 – TXRDY Transmitter Ready (cleared by writing FLEX_US_THR)
When FIFOs are disabled:
0: A character in FLEX_US_THR is waiting to be transferred to the Transmit Shift Register, or an STTBRK command has been requested, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1.
1: There is no character in FLEX_US_THR.
When FIFOs are enabled:
0: Transmit FIFO is full and cannot accept more data.
1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration.
TXRDY behavior with FIFO enabled is illustrated in 38.7.11.5 TXEMPTY, TXRDY and RXRDY Behavior.

Bit 0 – RXRDY Receiver Ready (cleared by reading FLEX_US_RHR)
When FIFOs are disabled:
0: No complete character has been received since the last read of FLEX_US_RHR or the receiver is disabled. If characters were received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled.
1: At least one complete character has been received and FLEX_US_RHR has not yet been read.
When FIFOs are enabled:
0: Receive FIFO is empty; no data to read
1: At least one unread data is in the Receive FIFO
RXRDY behavior with FIFO enabled is illustrated in 38.7.11.5 TXEMPTY, TXRDY and RXRDY Behavior.

38.10.16 USART Channel Status Register (LIN_MODE)

Name: FLEX_US_CSR (LIN_MODE)
Offset: 0x214
Reset: –
Property: Read-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

Bit	31	30	29	28	27	26	25	24
	LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	
Access	R	R	R	R	R	R	R	
Reset	–	–	–	–	–	–	–	

Bit	23	22	21	20	19	18	17	16
	LINBLS							
Access	R							
Reset	–							

Bit	15	14	13	12	11	10	9	8
	LINTC	LINID	LINBK				TXEMPTY	TIMEOUT
Access	R	R	R				R	R
Reset	–	–	–				–	–

Bit	7	6	5	4	3	2	1	0
	PARE	FRAME	OVRE				TXRDY	RXRDY
Access	R	R	R				R	R
Reset	–	–	–				–	–

Bit 31 – LINHTE LIN Header Timeout Error

Value	Description
0	No LIN header timeout error has been detected since the last RSTSTA command was issued.
1	A LIN header timeout error has been detected since the last RSTSTA command was issued.

Bit 30 – LINSTE LIN Synch Tolerance Error

Value	Description
0	No LIN synch tolerance error has been detected since the last RSTSTA command was issued.
1	A LIN synch tolerance error has been detected since the last RSTSTA command was issued.

Bit 29 – LINSNRE LIN Slave Not Responding Error

Value	Description
0	No LIN slave not responding error has been detected since the last RSTSTA command was issued.
1	A LIN slave not responding error has been detected since the last RSTSTA command was issued.

Bit 28 – LINCE LIN Checksum Error

Value	Description
0	No LIN checksum error has been detected since the last RSTSTA command was issued.
1	A LIN checksum error has been detected since the last RSTSTA command was issued.

Bit 27 – LINIPE LIN Identifier Parity Error

Value	Description
0	No LIN identifier parity error has been detected since the last RSTSTA command was issued.
1	A LIN identifier parity error has been detected since the last RSTSTA command was issued.

Bit 26 – LINISFE LIN Inconsistent Synch Field Error

Value	Description
0	No LIN inconsistent synch field error has been detected since the last RSTSTA
1	The USART is configured as a slave node and a LIN Inconsistent synch field error has been detected since the last RSTSTA command was issued.

Bit 25 – LINBE LIN Bit Error

Value	Description
0	No bit error has been detected since the last RSTSTA command was issued.
1	A bit error has been detected since the last RSTSTA command was issued.

Bit 23 – LINBLS LIN Bus Line Status

Value	Description
0	LIN bus line is set to 0.
1	LIN bus line is set to 1.

Bit 15 – LINTC LIN Transfer Completed

Value	Description
0	The USART is idle or a LIN transfer is ongoing.
1	A LIN transfer has been completed since the last RSTSTA command was issued.

Bit 14 – LINID LIN Identifier Sent or LIN Identifier Received
If USART operates in LIN Master mode (USART_MODE = 0xA):
0: No LIN identifier has been sent since the last RSTSTA command was issued.
1: At least one LIN identifier has been sent since the last RSTSTA command was issued.
If USART operates in LIN Slave mode (USART_MODE = 0xB):
0: No LIN identifier has been received since the last RSTSTA command was issued.
1: At least one LIN identifier has been received since the last RSTSTA.

Bit 13 – LINBK LIN Break Sent or LIN Break Received
Applicable if USART operates in LIN Master mode (USART_MODE = 0xA):
0: No LIN break has been sent since the last RSTSTA command was issued.
1: At least one LIN break has been sent since the last RSTSTA.
If USART operates in LIN Slave mode (USART_MODE = 0xB):
0: No LIN break has received sent since the last RSTSTA command was issued.
1: At least one LIN break has been received since the last RSTSTA command was issued.

Bit 9 – TXEMPTY Transmitter Empty (cleared by writing FLEX_US_THR)

Value	Description
0	There are characters in either FLEX_US_THR or the Transmit Shift Register, or the transmitter is disabled.
1	There are no characters in FLEX_US_THR, nor in the Transmit Shift Register.

Bit 8 – TIMEOUT Receiver Timeout

Value	Description
0	There has not been a timeout since the last start timeout command (FLEX_US_CR.STTTO) or the Timeout Register is 0.
1	There has been a timeout since the last start timeout command (FLEX_US_CR.STTTO).

Bit 7 – PARE Parity Error

Value	Description
0	No parity error has been detected since the last RSTSTA command was issued.
1	At least one parity error has been detected since the last RSTSTA command was issued.

Bit 6 – FRAME Framing Error

Value	Description
0	No stop bit has been detected low since the last RSTSTA command was issued.
1	At least one stop bit has been detected low since the last RSTSTA command was issued.

Bit 5 – OVRE Overrun Error

Value	Description
0	No overrun error has occurred since the last RSTSTA command was issued.
1	At least one overrun error has occurred since the last RSTSTA command was issued.

Bit 1 – TXRDY Transmitter Ready (cleared by writing FLEX_US_THR)

Value	Description
0	A character in FLEX_US_THR is waiting to be transferred to the Transmit Shift Register, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1.
1	There is no character in FLEX_US_THR.

Bit 0 – RXRDY Receiver Ready (cleared by reading FLEX_US_RHR)

Value	Description
0	No complete character has been received since the last read of FLEX_US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled.
1	At least one complete character has been received and FLEX_US_RHR has not yet been read.

38.10.17 USART Channel Status Register (LON_MODE)

Name: FLEX_US_CSR (LON_MODE)
Offset: 0x214
Reset: –
Property: Read-only

This configuration is relevant only if USART_MODE = 0x9 in the USART Mode Register.

Bit	31	30	29	28	27	26	25	24
				LBLOVFE	LRXD	LFET	LCOL	LTXD
Access				R	R	R	R	R
Reset				–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						UNRE	TXEMPTY	
Access						R	R	
Reset						–	–	

Bit	7	6	5	4	3	2	1	0
	LCRCE	LSFE	OVRE				TXRDY	RXRDY
Access	R	R	R				R	R
Reset	–	–	–				–	–

Bit 28 – LBLOVFE LON Backlog Overflow Error

Value	Description
0	No backlog overflow error occurred since the last RSTSTA command was issued.
1	At least one backlog error overflow occurred since the last RSTSTA command was issued.

Bit 27 – LRXD LON Reception End Flag

Value	Description
0	Reception on going or no reception occurred since the last RSTSTA command was issued.
1	At least one reception has been performed since the last RSTSTA command was issued.

Bit 26 – LFET LON Frame Early Termination

Value	Description
0	No frame has been terminated early due to collision detection since the last RSTSTA command was issued.
1	At least one transmission has been terminated due to collision detection since the last RSTSTA command was issued. (This stops the DMA until reset with RSTSTA bit).

Bit 25 – LCOL LON Collision Detected Flag

Value	Description
0	No collision occurred while transmitting since the last RSTSTA command was issued.
1	At least one collision occurred while transmitting since the last RSTSTA command was issued.

Bit 24 – LTXD LON Transmission End Flag

Value	Description
0	Transmission on going or no transmission occurred since the last RSTSTA command was issued.
1	At least one transmission has been performed since the last RSTSTA command was issued.

Bit 10 – UNRE Underrun Error

Value	Description
0	No LON underrun error has occurred since the last RSTSTA command was issued.
1	At least one LON underrun error has occurred since the last RSTSTA command was issued.

Bit 9 – TXEMPTY Transmitter Empty (cleared by writing FLEX_US_THR)

Value	Description
0	There are characters in either FLEX_US_THR or the Transmit Shift Register, or the transmitter is disabled.
1	There are no characters in FLEX_US_THR, nor in the Transmit Shift Register.

Bit 7 – LCRCE LON CRC Error

Value	Description
0	No CRC error has been detected since the last RSTSTA command was issued.
1	At least one CRC error has been detected since the last RSTSTA command was issued.

Bit 6 – LSFE LON Short Frame Error

Value	Description
0	No short frame received since the last RSTSTA command was issued.
1	At least one short frame received since the last RSTSTA command was issued.

Bit 5 – OVRE Overrun Error

Value	Description
0	No overrun error has occurred since the last RSTSTA command was issued.
1	At least one overrun error has occurred since the last RSTSTA command was issued.

Bit 1 – TXRDY Transmitter Ready (cleared by writing FLEX_US_THR)

Value	Description
0	A character in FLEX_US_THR is waiting to be transferred to the Transmit Shift Register, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1.
1	There is no character in FLEX_US_THR.

Bit 0 – RXRDY Receiver Ready (cleared by reading FLEX_US_RHR)

Value	Description
0	No complete character has been received since the last read of FLEX_US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled.
1	At least one complete character has been received and FLEX_US_RHR has not yet been read.

38.10.18 USART Receive Holding Register

Name: FLEX_US_RHR
Offset: 0x218
Reset: 0x00000000
Property: Read-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_US_FMR.RXRDYM = 0, see 38.7.11.6 USART Single Data Mode for details.

If enabled in the SFR module (default is disabled), a debugger read access on FLEX_US_RHR does not clear the FLEX_US_CSR.RXRDY flag.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RXSYNH							RXCHR[8]
Access	R							R
Reset	0							0

Bit	7	6	5	4	3	2	1	0
	RXCHR[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 15 – RXSYNH Received Sync

Value	Description
0	Last character received is a data.
1	Last character received is a command.

Bits 8:0 – RXCHR[8:0] Received Character
Last character received if RXRDY is set.

38.10.19 USART Receive Holding Register (FIFO Multi Data)

Name: FLEX_US_RHR (FIFO_MULTI_DATA)
Offset: 0x218
Reset: 0x00000000
Property: Read-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_US_FMR.RXRDYM > 0, see 38.7.11.7 USART Multiple Data Mode for details.

If enabled in the SFR module (default is disabled), a debugger read access on FLEX_US_RHR does not modify the Receive FIFO status.

Bit	31	30	29	28	27	26	25	24
	RXCHR3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RXCHR2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RXCHR1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RXCHR0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 0:7, 8:15, 16:23, 24:31 – RXCHRx Received Character
First unread character in the Receive FIFO if RXRDY is set.

38.10.20 USART Transmit Holding Register

Name: FLEX_US_THR
Offset: 0x21C
Reset: –
Property: Write-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_US_FMR.TXRDY = 0, see 38.7.11.6 USART Single Data Mode for details.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TXSYNH							TXCHR[8]
Access	W							W
Reset	–							–

Bit	7	6	5	4	3	2	1	0
	TXCHR[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 15 – TXSYNH Sync Field to be Transmitted

Value	Description
0	The next character sent is encoded as a data. Start frame delimiter is DATA SYNC.
1	The next character sent is encoded as a command. Start frame delimiter is COMMAND SYNC.

Bits 8:0 – TXCHR[8:0] Character to be Transmitted
The next character to be transmitted after the current character if TXRDY is not set.

38.10.21 USART Transmit Holding Register (FIFO Multi Data)

Name: FLEX_US_THR (FIFO_MULTI_DATA)
Offset: 0x21C
Reset: –
Property: Write-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_US_FMR.TXRDY > 0, see 38.7.11.7 USART Multiple Data Mode for details.

Bit	31	30	29	28	27	26	25	24
	TXCHR3[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	TXCHR2[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	TXCHR1[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	TXCHR0[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 0:7, 8:15, 16:23, 24:31 – TXCHRx Character to be Transmitted
Next character to be transmitted.

38.10.22 USART Baud Rate Generator Register

Name: FLEX_US_BRGR
Offset: 0x220
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
						FP[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bit	15	14	13	12	11	10	9	8
	CD[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 18:16 – FP[2:0] Fractional Part

WARNING When the value of field FP is greater than 0, the SCK (oversampling clock) generates nonconstant duty cycles. The SCK high duration is increased by “selected clock” period from time to time. The duty cycle depends on the value of the CD field.

Value	Description
0	Fractional divider is disabled.
1–7	Baud rate resolution, defined by FP × 1/8.

Bits 15:0 – CD[15:0] Clock Divider

CD	USART_MODE ≠ ISO7816			USART_MODE = ISO7816
	SYNC = 0		SYNC = 1	
	OVER = 0	OVER = 1		
0	Baud Rate Clock Disabled			
1 to 65535	CD = Selected Clock / (16 × Baud Rate)	CD = Selected Clock / (8 × Baud Rate)	CD = Selected Clock / Baud Rate	CD = Selected Clock / (FI_DI_RATIO × Baud Rate)

38.10.23 USART Receiver Timeout Register

Name: FLEX_US_RTOR
Offset: 0x224
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
								TO[16]
Access								R/W
Reset								0

Bit	15	14	13	12	11	10	9	8
	TO[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TO[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 16:0 – TO[16:0] Timeout Value

Value	Description
0	The receiver timeout is disabled.
1–131071	The receiver timeout is enabled and the timeout delay is TO × bit period.

38.10.24 USART Transmitter Timeguard Register

Name: FLEX_US_TTGR
Offset: 0x228
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

For LON-specific configurations, see USART Transmitter Timeguard Register (LON_MODE).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	TG[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – TG[7:0] Timeguard Value

Value	Description
0	The transmitter timeguard is disabled.
1–255	The transmitter timeguard is enabled and TG is timeguard delay / bit period.

38.10.25 USART Transmitter Timeguard Register (LON_MODE)

Name: FLEX_US_TTGR (LON_MODE)
Offset: 0x228
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	PCYCLE[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	PCYCLE[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	PCYCLE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – PCYCLE[23:0] LON PCYCLE Length

Value	Description
1-16777215	LON PCYCLE length in t_{bit}.

38.10.26 USART FI DI RATIO Register

Name: FLEX_US_FIDI
Offset: 0x240
Reset: 0x174
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

For LON-specific configurations, see USART Transmitter Timeguard Register (LON_MODE).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	FI_DI_RATIO[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit	7	6	5	4	3	2	1	0
	FI_DI_RATIO[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	1	0	1	0	0

Bits 15:0 – FI_DI_RATIO[15:0] FI Over DI Ratio Value

Value	Description
0	If ISO7816 mode is selected, the baud rate generator generates no signal.
1-2	Do not use.
3-2047	If ISO7816 mode is selected, the baud rate is the clock provided on SCK divided by FI_DI_RATIO.

38.10.27 USART FI DI RATIO Register (LON_MODE)

Name: FLEX_US_FIDI (LON_MODE)
Offset: 0x240
Reset: 0x174
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	BETA2[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	BETA2[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit	7	6	5	4	3	2	1	0
	BETA2[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	1	1	0	1	0	0

Bits 23:0 – BETA2[23:0] LON BETA2 Length

Value	Description
1-16777215	LON BETA2 length in t_{bit}.

38.10.28 USART Number of Errors Register

Name: FLEX_US_NER
Offset: 0x244
Reset: –
Property: Read-only

This register is relevant only if USART_MODE = 0x4 or 0x6 in the USART Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	NB_ERRORS[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	–	–	–	–	–	–	–	–

Bits 7:0 – NB_ERRORS[7:0] Number of Errors
Total number of errors that occurred during an ISO7816 transfer. This register automatically clears when read.

38.10.29 USART IrDA FILTER Register

Name: FLEX_US_IF
Offset: 0x24C
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x8 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	IRDA_FILTER[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – IRDA_FILTER[7:0] IrDA Filter
The IRDA_FILTER value must be defined to meet the following criteria:
$t_{peripheral\ clock} \times (IRDA\_FILTER + 3) < 1.41\ \mu s$

38.10.30 USART Manchester Configuration Register

Name: FLEX_US_MAN
Offset: 0x250
Reset: 0xB0011004
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	RXIDLEV	DRIFT	ONE	RX_MPOL			RX_PP[1:0]	
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	1	0	1	1			0	0

Bit	23	22	21	20	19	18	17	16
					RX_PL[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	1

Bit	15	14	13	12	11	10	9	8
				TX_MPOL			TX_PP[1:0]	
Access				R/W			R/W	R/W
Reset				1			0	0

Bit	7	6	5	4	3	2	1	0
					TX_PL[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	1	0	0

Bit 31 – RXIDLEV Receiver Idle Value

Value	Description
0	Receiver line idle value is 0.
1	Receiver line idle value is 1.

Bit 30 – DRIFT Drift Compensation

Value	Description
0	The USART cannot recover from an important clock drift.
1	The USART can recover from clock drift. The 16X Clock mode must be enabled.

Bit 29 – ONE Must Be Set to 1
Bit 29 must always be set to 1 when programming the FLEX_US_MAN register.

Bit 28 – RX_MPOL Receiver Manchester Polarity

Value	Description
0	Logic zero is coded as a zero-to-one transition, Logic one is coded as a one-to-zero transition.
1	Logic zero is coded as a one-to-zero transition, Logic one is coded as a zero-to-one transition.

Bits 25:24 – RX_PP[1:0] Receiver Preamble Pattern detected
The following values assume that RX_MPOL field is not set:

Value	Name	Description
0	ALL_ONE	The preamble is composed of ‘1’s.
1	ALL_ZERO	The preamble is composed of ‘0’s.
2	ZERO_ONE	The preamble is composed of ‘01’s.
3	ONE_ZERO	The preamble is composed of ‘10’s.

Bits 19:16 – RX_PL[3:0] Receiver Preamble Length

Value	Description
0	The receiver preamble pattern detection is disabled.
1–15	The detected preamble length is RX_PL × Bit Period.

Bit 12 – TX_MPOL Transmitter Manchester Polarity

Value	Description
0	Logic zero is coded as a zero-to-one transition, Logic one is coded as a one-to-zero transition.
1	Logic zero is coded as a one-to-zero transition, Logic one is coded as a zero-to-one transition.

Bits 9:8 – TX_PP[1:0] Transmitter Preamble Pattern
The following values assume that TX_MPOL field is not set:

Value	Name	Description
0	ALL_ONE	The preamble is composed of ‘1’s.
1	ALL_ZERO	The preamble is composed of ‘0’s.
2	ZERO_ONE	The preamble is composed of ‘01’s.
3	ONE_ZERO	The preamble is composed of ‘10’s.

Bits 3:0 – TX_PL[3:0] Transmitter Preamble Length

Value	Description
0	The transmitter preamble pattern generation is disabled.
1–15	The preamble length is TX_PL × Bit Period.

38.10.31 USART LIN Mode Register

Name: FLEX_US_LINMR
Offset: 0x254
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access							R/W	R/W
Reset							0	0

Bit	15	14	13	12	11	10	9	8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 17 – SYNCDIS Synchronization Disable

Value	Description
0	The synchronization procedure is performed in LIN slave node configuration.
1	The synchronization procedure is not performed in LIN slave node configuration.

Bit 16 – PDCM DMAC Mode

Value	Description
0	The LIN mode register FLEX_US_LINMR is not written by the DMAC.
1	The LIN mode register FLEX_US_LINMR (excepting that flag) is written by the DMAC.

Bits 15:8 – DLC[7:0] Data Length Control

Value	Description
0-255	Defines the response data length if DLM = 0, in that case the response data length is equal to DLC+1 bytes.

Bit 7 – WKUPTYP Wakeup Signal Type

Value	Description
0	Setting the LINWKUP bit in the control register sends a LIN 2.0 wakeup signal.
1	Setting the LINWKUP bit in the control register sends a LIN 1.3 wakeup signal.

Bit 6 – FSDIS Frame Slot Mode Disable

Value	Description
0	The Frame Slot mode is enabled.
1	The Frame Slot mode is disabled.

Bit 5 – DLM Data Length Mode

Value	Description
0	The response data length is defined by the DLC field of this register.
1	The response data length is defined by the bits 5 and 6 of the identifier (FLEX_US_LINIR.IDCHR).

Bit 4 – CHKTYP Checksum Type

Value	Description
0	LIN 2.0 “enhanced” checksum
1	LIN 1.3 “classic” checksum

Bit 3 – CHKDIS Checksum Disable

Value	Description
0	In master node configuration, the checksum is computed and sent automatically. In slave node configuration, the checksum is checked automatically.
1	Whatever the node configuration is, the checksum is not computed/sent and it is not checked.

Bit 2 – PARDIS Parity Disable

Value	Description
0	In master node configuration, the identifier parity is computed and sent automatically. In master node and slave node configuration, the parity is checked automatically.
1	Whatever the node configuration is, the Identifier parity is not computed/sent and it is not checked.

Bits 1:0 – NACT[1:0] LIN Node Action
Values which are not listed in the table must be considered as “reserved”.

Value	Name	Description
0	PUBLISH	The USART transmits the response.
1	SUBSCRIBE	The USART receives the response.
2	IGNORE	The USART does not transmit and does not receive the response.

38.10.32 USART LIN Identifier Register

Name: FLEX_US_LINIR
Offset: 0x258
Reset: 0x00000000
Property: Read/Write

Write is possible only in LIN master node configuration.

This register is relevant only if USART_MODE = 0xA or 0xB in USART Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	IDCHR[7:0]							
Access	R/W-R	R/W-R	R/W-R	R/W-R	R/W-R	R/W-R	R/W-R	R/W-R
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – IDCHR[7:0] Identifier Character
If USART_MODE = 0xA (master node configuration):

- IDCHR is Read/Write and its value is the identifier character to be transmitted.

If USART_MODE = 0xB (slave node configuration):

- IDCHR is Read-only and its value is the last identifier character that has been received.

38.10.33 USART LIN Baud Rate Register

Name: FLEX_US_LINBRR
Offset: 0x25C
Reset: 0x00000000
Property: Read-only

This register is relevant only if USART_MODE = 0xA or 0xB in USART Mode Register.

Returns the baud rate value after the synchronization process completion.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
						LINFP[2:0]		
Access						R	R	R
Reset						0	0	0

Bit	15	14	13	12	11	10	9	8
	LINCD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LINCD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 18:16 – LINFP[2:0] Fractional Part after Synchronization

Bits 15:0 – LINCD[15:0] Clock Divider after Synchronization

38.10.34 USART LON Mode Register

Name: FLEX_US_LONMR
Offset: 0x260
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	EOFS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			LCDS	DMAM	CDTAIL	TCOL	COLDET	COMMT
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bits 23:16 – EOFS[7:0] End of Frame Condition Size

Value	Description
0-255	Defines the minimum transitionless time for the IP to detect a LON end of frame condition. t_{eof} = (EOFS+1) × t_{clock} × 8 × (2- OVER)

Bit 5 – LCDS LON Collision Detection Source

Value	Description
0	LON collision detection source is external.
1	LON collision detection source is internal.

Bit 4 – DMAM LON DMA Mode

Value	Description
0	The LON data length register FLEX_US_LONDL is not written by the DMA.
1	The LON data length register FLEX_US_LONDL is written by the DMA.

Bit 3 – CDTAIL LON Collision Detection on Frame Tail

Value	Description
0	Detect collisions after CRC has been sent but prior end of transmission in LON comm_type = 1 mode.
1	Ignore collisions after CRC has been sent but prior end of transmission in LON comm_type = 1 mode.

Bit 2 – TCOL Terminate Frame upon Collision Notification

Value	Description
0	Do not terminate the frame in LON comm_type = 1 mode upon collision detection.
1	Terminate the frame in LON comm_type = 1 mode upon collision detection if possible.

Bit 1 – COLDET LON Collision Detection Feature

Value	Description
0	LON collision detection feature disabled.
1	LON collision detection feature enabled.

Bit 0 – COMMT LON comm_type Parameter Value

Value	Description
0	LON comm_type = 1 mode.
1	LON comm_type = 2 mode.

38.10.35 USART LON Preamble Register

Name: FLEX_US_LONPR
Offset: 0x264
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
			LONPL[13:8]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	LONPL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 13:0 – LONPL[13:0] LON Preamble Length

Value	Description
1–16383	LON preamble length in t_{bit} (without byte-sync).

38.10.36 USART LON Data Length Register

Name: FLEX_US_LONDL
Offset: 0x268
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	LONDL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – LONDL[7:0] LON Data Length

Value	Description
0–255	LON data length is LONDL + 1 byte.

38.10.37 USART LON L2HDR Register

Name: FLEX_US_LONL2HDR
Offset: 0x26C
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	PB	ALTP	BLI[5:0]					
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 7 – PB LON Priority Bit

Value	Description
0	LON priority bit reset.
1	LON priority bit set.

Bit 6 – ALTP LON Alternate Path Bit

Value	Description
0	LON alternate path bit reset.
1	LON alternate path bit set.

Bits 5:0 – BLI[5:0] LON Backlog Increment

Value	Description
0-63	LON backlog increment to be generated as a result of delivering the LON frame.

38.10.38 USART LON Backlog Register

Name: FLEX_US_LONBL
Offset: 0x270
Reset: 0x00000000
Property: Read-only

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			LONBL[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bits 5:0 – LONBL[5:0] LON Node Backlog Value

Value	Description
1-63	LON node backlog value.

38.10.39 USART LON Beta1 Tx Register

Name: FLEX_US_LONB1TX
Offset: 0x274
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	BETA1TX[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	BETA1TX[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BETA1TX[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – BETA1TX[23:0] LON Beta1 Length after Transmission

Value	Description
1–16777215	LON beta1 length after transmission in t_{bit}.

38.10.40 USART LON Beta1 Rx Register

Name: FLEX_US_LONB1RX
Offset: 0x278
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	BETA1RX[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	BETA1RX[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BETA1RX[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – BETA1RX[23:0] LON Beta1 Length after Reception

Value	Description
1–16777215	LON beta1 length after reception in t_{bit}.

38.10.41 USART LON Priority Register

Name: FLEX_US_LONPRIO
Offset: 0x27C
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
		NPS[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		PSNB[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bits 14:8 – NPS[6:0] LON Node Priority Slot

Value	Description
0-127	Node priority slot.

Bits 6:0 – PSNB[6:0] LON Priority Slot Number

Value	Description
0-127	Number of priority slots in the LON network configuration.

38.10.42 USART LON IDT Tx Register

Name: FLEX_US_IDTTX
Offset: 0x280
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	IDTTX[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IDTTX[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IDTTX[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – IDTTX[23:0] LON Indeterminate Time after Transmission (comm_type = 1 mode only)

Value	Description
0–16777215	LON indeterminate time after transmission in t_{bit}.

38.10.43 USART LON IDT Rx Register

Name: FLEX_US_IDTRX
Offset: 0x284
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	IDTRX[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IDTRX[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IDTRX[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – IDTRX[23:0] LON Indeterminate Time after Reception (comm_type = 1 mode only)

Value	Description
0–16777215	LON indeterminate time after reception in t_{bit}.

38.10.44 USART IC DIFF Register

Name: FLEX_US_ICDIFF
Offset: 0x288
Reset: 0x00000000
Property: Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					ICDIFF[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 3:0 – ICDIFF[3:0] IC Differentiator Number

38.10.45 USART Comparison Register

Name: FLEX_US_CMPR
Offset: 0x290
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
								VAL2[8]
Access								R/W
Reset								0

Bit	23	22	21	20	19	18	17	16
	VAL2[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
		CMPPAR		CMPMODE				VAL1[8]
Access		R/W		R/W				R/W
Reset		0		0				0

Bit	7	6	5	4	3	2	1	0
	VAL1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 24:16 – VAL2[8:0] Second Comparison Value for Received Character

Value	Description
0-511	The received character must be lower than or equal to the value of VAL2 and higher than or equal to VAL1 to set the FLEX_US_CSR.CMP flag.

Bit 14 – CMPPAR Compare Parity

Value	Description
0	The parity is not checked and a bad parity cannot prevent from waking up the system.
1	The parity is checked and a matching condition on data can be cancelled by an error on parity bit, so no wakeup is performed.

Bit 12 – CMPMODE Comparison Mode

Value	Name	Description
0	FLAG_ONLY	Any character is received and comparison function drives CMP flag.
1	START_CONDITION	Comparison condition must be met to start reception.
2	FILTER	Comparison must be met to receive the current data only

Bits 8:0 – VAL1[8:0] First Comparison Value for Received Character

Value	Description
0-511	The received character must be higher than or equal to the value of VAL1 and lower than or equal to VAL2 to set the FLEX_US_CSR.CMP flag.

38.10.46 USART FIFO Mode Register

Name: FLEX_US_FMR
Offset: 0x2A0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
			RXFTHRES2[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			RXFTHRES[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
			TXFTHRES[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FRTSC		RXRDYM[1:0]				TXRDYM[1:0]	
Access	R/W		R/W	R/W			R/W	R/W
Reset	0		0	0			0	0

Bits 29:24 – RXFTHRES2[5:0] Receive FIFO Threshold 2

Value	Description
0-8	Defines the Receive FIFO threshold 2 value (number of data). The FLEX_US_FESR.RXFTHF2 flag will be set when Receive FIFO goes from “above” threshold state to “equal to or below” threshold state.

Bits 21:16 – RXFTHRES[5:0] Receive FIFO Threshold

Value	Description
0-8	Defines the Receive FIFO threshold value (number of data). The FLEX_US_FESR.RXFTHF flag will be set when Receive FIFO goes from “below” threshold state to “equal to or above” threshold state.

Bits 13:8 – TXFTHRES[5:0] Transmit FIFO Threshold

Value	Description
0-8	Defines the Transmit FIFO threshold value (number of data). The FLEX_US_FESR.TXFTHF flag will be set when Transmit FIFO goes from “above” threshold state to “equal to or below” threshold state.

Bit 7 – FRTSC FIFO RTS Pin Control enable (Hardware Handshaking mode only)
See Hardware Handshaking for details.

Value	Description
0	RTS pin is not controlled by Receive FIFO thresholds.
1	RTS pin is controlled by Receive FIFO thresholds.

Bits 5:4 – RXRDYM[1:0] Receiver Ready Mode
If FIFOs are enabled, the FLEX_US_CSR.RXRDY flag behaves as follows.

Value	Name	Description
0	ONE_DATA	RXRDY will be at level ‘1’ when at least one unread data is in the Receive FIFO.
1	TWO_DATA	RXRDY will be at level ‘1’ when at least two unread data are in the Receive FIFO.
2	FOUR_DATA	RXRDY will be at level ‘1’ when at least four unread data are in the Receive FIFO.

Bits 1:0 – TXRDYM[1:0] Transmitter Ready Mode
If FIFOs are enabled, the FLEX_US_CSR.TXRDY flag behaves as follows.

Value	Name	Description
0	ONE_DATA	TXRDY will be at level ‘1’ when at least one data can be written in the Transmit FIFO
1	TWO_DATA	TXRDY will be at level ‘1’ when at least two data can be written in the Transmit FIFO
2	FOUR_DATA	TXRDY will be at level ‘1’ when at least four data can be written in the Transmit FIFO

38.10.47 USART FIFO Level Register

Name: FLEX_US_FLR
Offset: 0x2A4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			RXFL[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TXFL[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bits 21:16 – RXFL[5:0] Receive FIFO Level

Value	Description
0	There is no unread data in the Receive FIFO.
1-8	Indicates the number of unread data in the Receive FIFO.

Bits 5:0 – TXFL[5:0] Transmit FIFO Level

Value	Description
0	There is no data in the Transmit FIFO.
1-8	Indicates the number of data in the Transmit FIFO.

38.10.48 USART FIFO Interrupt Enable Register

Name: FLEX_US_FIER
Offset: 0x2A8
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							RXFTHF2	
Access							W	
Reset							–	

Bit	7	6	5	4	3	2	1	0
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 9 – RXFTHF2 RXFTHF2 Interrupt Enable

Bit 7 – RXFPTEF RXFPTEF Interrupt Enable

Bit 6 – TXFPTEF TXFPTEF Interrupt Enable

Bit 5 – RXFTHF RXFTHF Interrupt Enable

Bit 4 – RXFFF RXFFF Interrupt Enable

Bit 3 – RXFEF RXFEF Interrupt Enable

Bit 2 – TXFTHF TXFTHF Interrupt Enable

Bit 1 – TXFFF TXFFF Interrupt Enable

Bit 0 – TXFEF TXFEF Interrupt Enable

38.10.49 USART FIFO Interrupt Disable Register

Name: FLEX_US_FIDR
Offset: 0x2AC
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							RXFTHF2	
Access							W	
Reset							–	

Bit	7	6	5	4	3	2	1	0
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 9 – RXFTHF2 RXFTHF2 Interrupt Disable

Bit 7 – RXFPTEF RXFPTEF Interrupt Disable

Bit 6 – TXFPTEF TXFPTEF Interrupt Disable

Bit 5 – RXFTHF RXFTHF Interrupt Disable

Bit 4 – RXFFF RXFFF Interrupt Disable

Bit 3 – RXFEF RXFEF Interrupt Disable

Bit 2 – TXFTHF TXFTHF Interrupt Disable

Bit 1 – TXFFF TXFFF Interrupt Disable

Bit 0 – TXFEF TXFEF Interrupt Disable

38.10.50 USART FIFO Interrupt Mask Register

Name: FLEX_US_FIMR
Offset: 0x2B0
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							RXFTHF2	
Access							R	
Reset							0	

Bit	7	6	5	4	3	2	1	0
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 9 – RXFTHF2 RXFTHF2 Interrupt Mask

Bit 7 – RXFPTEF RXFPTEF Interrupt Mask

Bit 6 – TXFPTEF TXFPTEF Interrupt Mask

Bit 5 – RXFTHF RXFTHF Interrupt Mask

Bit 4 – RXFFF RXFFF Interrupt Mask

Bit 3 – RXFEF RXFEF Interrupt Mask

Bit 2 – TXFTHF TXFTHF Interrupt Mask

Bit 1 – TXFFF TXFFF Interrupt Mask

Bit 0 – TXFEF TXFEF Interrupt Mask

38.10.51 USART FIFO Event Status Register

Name: FLEX_US_FESR
Offset: 0x2B4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
							RXFTHF2	TXFLOCK
Access							R	R
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 9 – RXFTHF2 Receive FIFO Threshold Flag 2 (cleared by writing the FLEX_US_CR.RSTSTA bit)

Value	Description
0	Number of unread data in Receive FIFO is above RXFTHRES threshold.
1	Number of unread data in Receive FIFO has reached RXFTHRES2 threshold since the last RSTSTA command was issued.

Bit 8 – TXFLOCK Transmit FIFO Lock

Value	Description
0	The Transmit FIFO is not locked.
1	The Transmit FIFO is locked.

Bit 7 – RXFPTEF Receive FIFO Pointer Error Flag
See 38.7.11.9 FIFO Pointer Error for details.

Value	Description
0	No Receive FIFO pointer occurred.
1	Receive FIFO pointer error occurred. Receiver must be reset.

Bit 6 – TXFPTEF Transmit FIFO Pointer Error Flag
See 38.7.11.9 FIFO Pointer Error for details.

Value	Description
0	No Transmit FIFO pointer occurred.
1	Transmit FIFO pointer error occurred. Transceiver must be reset.

Bit 5 – RXFTHF Receive FIFO Threshold Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

Value	Description
0	Number of unread data in Receive FIFO is below RXFTHRES threshold.
1	Number of unread data in Receive FIFO has reached RXFTHRES threshold since the last RSTSTA command was issued.

Bit 4 – RXFFF Receive FIFO Full Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

Value	Description
0	Receive FIFO is not empty.
1	Receive FIFO has been filled since the last RSTSTA command was issued.

Bit 3 – RXFEF Receive FIFO Empty Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

Value	Description
0	Receive FIFO is not empty.
1	Receive FIFO has been emptied since the last RSTSTA command was issued.

Bit 2 – TXFTHF Transmit FIFO Threshold Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

Value	Description
0	Number of data in Transmit FIFO is above TXFTHRES threshold.
1	Number of data in Transmit FIFO has reached TXFTHRES threshold since the last RSTSTA command was issued.

Bit 1 – TXFFF Transmit FIFO Full Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

Value	Description
0	Transmit FIFO is not full.
1	Transmit FIFO has been filled since the last RSTSTA command was issued.

Bit 0 – TXFEF Transmit FIFO Empty Flag (cleared by writing the FLEX_US_CR.RSTSTA bit)

Value	Description
0	Transmit FIFO is not empty.
1	Transmit FIFO has been emptied since the last RSTSTA command was issued.

38.10.52 USART Write Protection Mode Register

Name: FLEX_US_WPMR
Offset: 0x2E4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPEN
Access								R/W
Reset								0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x555341	PASSWD	Writing any other value in this field aborts the write operation of bit WPEN. Always reads as 0.

Bit 0 – WPEN Write Protection Enable

See USART Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection on configuration registers if WPKEY corresponds to 0x555341 (“USA” in ASCII).
1	Enables the write protection on configuration registers if WPKEY corresponds to 0x555341 (“USA” in ASCII).

38.10.53 USART Write Protection Status Register

Name: FLEX_US_WPSR
Offset: 0x2E8
Reset: 0x00000000
Property: Read-only

If enabled in the SFR module (default is disabled), a debugger read access on FLEX_US_WPSR does not clear the WPVS flag.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	WPVSRC[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 23:8 – WPVSRC[15:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protection violation has occurred since the last read of FLEX_US_WPSR.
1	A write protection violation has occurred since the last read of FLEX_US_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

38.10.54 SPI Control Register

Name: FLEX_SPI_CR
Offset: 0x400
Reset: –
Property: Write-only

This register can only be written if the WPCREN bit is cleared in the SPI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	FIFODIS	FIFOEN						LASTXFER
Access	W	W						W
Reset	–	–						–

Bit	23	22	21	20	19	18	17	16
							RXFCLR	TXFCLR
Access							W	W
Reset							–	–

Bit	15	14	13	12	11	10	9	8
				REQCLR				
Access				W				
Reset				–				

Bit	7	6	5	4	3	2	1	0
	SWRST						SPIDIS	SPIEN
Access	W						W	W
Reset	–						–	–

Bit 31 – FIFODIS FIFO Disable

Value	Description
0	No effect.
1	Disables the Transmit and Receive FIFOs

Bit 30 – FIFOEN FIFO Enable

Value	Description
0	No effect.
1	Enables the Transmit and Receive FIFOs

Bit 24 – LASTXFER Last Transfer
See Peripheral Selection for more details.

Value	Description
0	No effect.
1	The current NPCS will be de-asserted after the character written in TD has been transferred. When CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed.

Bit 17 – RXFCLR Receive FIFO Clear

Value	Description
0	No effect.
1	Empties the Receive FIFO.

Bit 16 – TXFCLR Transmit FIFO Clear

Value	Description
0	No effect.
1	Empties the Transmit FIFO.

Bit 12 – REQCLR Request to Clear the Comparison Trigger
0: No effect.
1: Restarts the comparison trigger to enable FLEX_SPI_RDR loading.

Bit 7 – SWRST SPI Software Reset
The SPI is in Slave mode after software reset.

Value	Description
0	No effect.
1	Resets the SPI. A software-triggered hardware reset of the SPI interface is performed.

Bit 1 – SPIDIS SPI Disable
If a transfer is in progress when SPIDIS is set, the SPI completes the transmission of the shifter register and does not start any new transfer, even if the FLEX_US_THR is loaded.
All pins are set in Input mode after completion of the transmission in progress, if any.

Value	Description
0	No effect.
1	Disables the SPI.

Bit 0 – SPIEN SPI Enable

Value	Description
0	No effect.
1	Enables the SPI to transfer and receive data.

38.10.55 SPI Mode Register

Name: FLEX_SPI_MR
Offset: 0x404
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	DLYBCS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
					PCS[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
				CMPMODE				
Access				R/W				
Reset				0				

Bit	7	6	5	4	3	2	1	0
	LLB		WDRBT	MODFDIS	BRSRCCLK	PCSDEC	PS	MSTR
Access	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Reset	0		0	0	0	0	0	0

Bits 31:24 – DLYBCS[7:0] Delay Between Chip Selects
This field defines the delay between the inactivation and the activation of NPCS. The DLYBCS time guarantees nonoverlapping chip selects and solves bus contentions in case of peripherals having long data float times.
If DLYBCS is ≤ 6, six peripheral clock periods are inserted by default.
Otherwise, the following equations determine the delay:
If FLEX_SPI_MR.BRSRCCLK = 0: DLYBCS = Delay Between Chip Selects × $f_{peripheral\ clock}$
If FLEX_SPI_MR.BRSRCCLK = 1: DLYBCS = Delay Between Chip Selects × f_{GCLK}

Bits 19:16 – PCS[3:0] Peripheral Chip Select
This field is only used if fixed peripheral select is active (PS = 0).
If PCSDEC = 0:
PCS = xxx0 NPCS[3:0] = 1110
PCS = xx01 NPCS[3:0] = 1101
PCS = x011 NPCS[3:0] = 1011
PCS = 0111 NPCS[3:0] = 0111
PCS = 1111 forbidden (no peripheral is selected)
(x = don't care)
If PCSDEC = 1:
NPCS[3:0] output signals = PCS

Bit 12 – CMPMODE Comparison Mode

Value	Name	Description
0	FLAG_ONLY	Any character is received and comparison function drives CMP flag.
1	START_CONDITION	Comparison condition must be met to start reception of all incoming characters until REQCLR is set.

Bit 7 – LLB Local Loopback Enable
LLB controls the local loopback on the data shift register for testing in Master mode only (MISO is internally connected on MOSI).

Value	Description
0	Local loopback path disabled.
1	Local loopback path enabled.

Bit 5 – WDRBT Wait Data Read Before Transfer

Value	Description
0	No Effect. In Master mode, a transfer can be initiated regardless of the FLEX_SPI_RDR state.
1	In Master mode, a transfer can start only if FLEX_SPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception.

Bit 4 – MODFDIS Mode Fault Detection

Value	Description
0	Mode fault detection is enabled.
1	Mode fault detection is disabled.

Bit 3 – BRSRCCLK Bit Rate Source Clock
If the bit BRSRCCLK = 1, the FLEX_US_CSRx.SCBR field must be programmed with a value greater than 1.

Value	Name	Description
0	PERIPH_CLK	The peripheral clock is the source clock for the bit rate generation.
1	GCLK	GCLK is the source clock for the bit rate generation, thus the bit rate can be independent of the core/peripheral clock.

Bit 2 – PCSDEC Chip Select Decode
When PCSDEC equals one, up to 15 Chip Select signals can be generated with the four NPCS lines using an external 4- to 16-bit decoder. The Chip Select registers define the characteristics of the 15 chip selects, with the following rules:
FLEX_SPI_CSR0 defines peripheral chip select signals 0 to 3.
FLEX_SPI_CSR1 defines peripheral chip select signals 4 to 7.
FLEX_SPI_CSR2 defines peripheral chip select signals 8 to 11.
FLEX_SPI_CSR3 defines peripheral chip select signals 12 to 14.

Value	Description
0	The chip selects are directly connected to a peripheral device.
1	The four NPCS chip select lines are connected to a 4- to 16-bit decoder.

Bit 1 – PS Peripheral Select

Value	Description
0	Fixed Peripheral Select
1	Variable Peripheral Select

Bit 0 – MSTR Master/Slave Mode

Value	Description
0	SPI is in Slave mode.
1	SPI is in Master mode.

38.10.56 SPI Receive Data Register

Name: FLEX_SPI_RDR
Offset: 0x408
Reset: 0x00000000
Property: Read-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.RXRDYM = 0, see SPI Single Data Mode for details.

By default, any debugger or software read access to FLEX_SPI_RDR clears the FLEX_SPI_SR.RDRF flag. To prevent this flag from being cleared on debugger read access, a specific bit must be enabled in the SFR module.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					PCS[3:0]			
Access					R	R	R	R
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RD[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RD[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 19:16 – PCS[3:0] Peripheral Chip Select
In Master mode only, these bits indicate the value on the NPCS pins at the end of a transfer. Otherwise, these bits are read as zero.
Note: When using Variable Peripheral Select mode (FLEX_SPI_MR.PS = 1), it is mandatory to set the FLEX_SPI_MR.WDRBT bit to 1 if the PCS field must be processed in FLEX_SPI_RDR.

Bits 15:0 – RD[15:0] Receive Data
Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

38.10.57 SPI Receive Data Register (FIFO Multiple Data, 8-bit)

Name: FLEX_SPI_RDR (FIFO_MULTI_DATA_8)
Offset: 0x408
Reset: 0x00000000
Property: Read-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.RXRDYM > 0, see SPI Multiple Data Mode for details.

If enabled in the SFR module (default is disabled), a debugger read access does not modify the Receive FIFO status.

Bit	31	30	29	28	27	26	25	24
	RD3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RD2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RD1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RD0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 0:7, 8:15, 16:23, 24:31 – RDx Receive Data
First unread data in the Receive FIFO. Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

38.10.58 SPI Receive Data Register (FIFO Multiple Data, 16-bit)

Name: FLEX_SPI_RDR (FIFO_MULTI_DATA_16)
Offset: 0x408
Reset: 0x00000000
Property: Read-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.RXRDYM > 0, see SPI Multiple Data Mode for details.

If enabled in the SFR module (default is disabled), a debugger read access does not modify the Receive FIFO status.

Bit	31	30	29	28	27	26	25	24
	RD1[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RD1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RD0[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RD0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 0:15, 16:31 – RDx Receive Data
First unread data in the Receive FIFO. Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

38.10.59 SPI Transmit Data Register

Name: FLEX_SPI_TDR
Offset: 0x40C
Reset: –
Property: Write-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.TXRDYM = 0, see 38.8.6.6 SPI Single Data Mode for details.

Bit	31	30	29	28	27	26	25	24
								LASTXFER
Access								W
Reset								–

Bit	23	22	21	20	19	18	17	16
					PCS[3:0]			
Access					W	W	W	W
Reset					–	–	–	–

Bit	15	14	13	12	11	10	9	8
	TD[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	TD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 24 – LASTXFER Last Transfer
This field is only used if variable peripheral select is active (FLEX_SPI_MR.PS = 1).

Value	Description
0	No effect.
1	The current NPCS is de-asserted after the transfer of the character written in TD. When FLEX_SPI_CSRx.CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed.

Bits 19:16 – PCS[3:0] Peripheral Chip Select
This field is only used if variable peripheral select is active (FLEX_SPI_MR.PS = 1).
If FLEX_SPI_MR.PCSDEC = 0:
PCS = xxx0 NPCS[3:0] = 1110
PCS = xx01 NPCS[3:0] = 1101
PCS = x011 NPCS[3:0] = 1011
PCS = 0111 NPCS[3:0] = 0111
PCS = 1111 forbidden (no peripheral is selected)
(x = don't care)
If FLEX_SPI_MR.PCSDEC = 1:
NPCS[3:0] output signals = PCS

Bits 15:0 – TD[15:0] Transmit Data
Data to be transmitted by the SPI Interface is stored in this register. Information to be transmitted must be written to the transmit data register in a right-justified format.

38.10.60 SPI Transmit Data Register (FIFO Multiple Data, 8- to 16-bit)

Name: FLEX_SPI_TDR (FIFO_MULTI_DATA)
Offset: 0x40C
Reset: –
Property: Write-only

If FIFO is enabled (FLEX_SPI_CR.FIFOEN) and FLEX_SPI_FMR.TXRDYM > 0, see Section 1.8.7.7 “SPI Multiple Data Mode” for details.

Bit	31	30	29	28	27	26	25	24
	TD1[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	TD1[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	TD0[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	TD0[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 0:15, 16:31 – TDx Transmit Data
Next data to write in the Transmit FIFO. Information to be transmitted must be written to this register in a right-justified format.

38.10.61 SPI Status Register

Name: FLEX_SPI_SR
Offset: 0x410
Reset: 0x00000000
Property: Read-only

By default, any debugger or software read access to FLEX_SPI_SR clears the MODF, OVRES, NSSR, UNDES, and CMP flags. To prevent these flags from being cleared on debugger read access, a specific bit must be enabled in the SFR module.

Bit	31	30	29	28	27	26	25	24
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
								SPIENS
Access								R
Reset								0

Bit	15	14	13	12	11	10	9	8
				SFERR	CMP	UNDES	TXEMPTY	NSSR
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					OVRES	MODF	TDRE	RDRF
Access					R	R	R	R
Reset					0	0	0	0

Bit 31 – RXFPTEF Receive FIFO Pointer Error Flag
See 38.8.6.8 FIFO Pointer Error for details.

Value	Description
0	No Receive FIFO pointer occurred
1	Receive FIFO pointer error occurred. Receiver must be reset

Bit 30 – TXFPTEF Transmit FIFO Pointer Error Flag
See 38.8.6.8 FIFO Pointer Error for details.

Value	Description
0	No Transmit FIFO pointer occurred
1	Transmit FIFO pointer error occurred. Transceiver must be reset

Bit 29 – RXFTHF Receive FIFO Threshold Flag

Value	Description
0	Number of unread data in Receive FIFO is below RXFTHRES threshold or RXFTH flag has been cleared.
1	Number of unread data in Receive FIFO has reached RXFTHRES threshold (changing states from “below threshold” to “equal to or above threshold”).

Bit 28 – RXFFF Receive FIFO Full Flag

Value	Description
0	Receive FIFO is not empty or RXFE flag has been cleared.
1	Receive FIFO has been filled (changing states from “not full” to “full”).

Bit 27 – RXFEF Receive FIFO Empty Flag

Value	Description
0	Receive FIFO is not empty or RXFE flag has been cleared.
1	Receive FIFO has been emptied (changing states from “not empty” to “empty”).

Bit 26 – TXFTHF Transmit FIFO Threshold Flag (cleared on read)

Value	Description
0	Number of data in Transmit FIFO is above TXFTHRES threshold.
1	Number of data in Transmit FIFO has reached TXFTHRES threshold since the last read of FLEX_SPI_SR.

Bit 25 – TXFFF Transmit FIFO Full Flag (cleared on read)

Value	Description
0	Transmit FIFO is not full or TXFF flag has been cleared.
1	Transmit FIFO has been filled since the last read of FLEX_SPI_SR.

Bit 24 – TXFEF Transmit FIFO Empty Flag (cleared on read)

Value	Description
0	Transmit FIFO is not empty.
1	Transmit FIFO has been emptied since the last read of FLEX_SPI_SR.

Bit 16 – SPIENS SPI Enable Status

Value	Description
0	SPI is disabled.
1	SPI is enabled.

Bit 12 – SFERR Slave Frame Error (cleared on read)

Value	Description
0	There is no frame error detected for a slave access since the last read of FLEX_SPI_SR.
1	In Slave mode, the chip select raised while the character defined in FLEX_SPI_CSR0.BITS was not complete.

Bit 11 – CMP Comparison Status (cleared on read)

Value	Description
0	No received character matched the comparison criteria programmed in VAL1 and VAL2 fields in FLEX_SPI_CMPR since the last read of FLEX_SPI_SR.
1	A received character matched the comparison criteria since the last read of FLEX_SPI_SR.

Bit 10 – UNDES Underrun Error Status (Slave mode only) (cleared on read)

Value	Description
0	No underrun has been detected since the last read of FLEX_SPI_SR.
1	A transfer starts whereas no data has been loaded in FLEX_SPI_TDR, cleared when FLEX_SPI_SR is read.

Bit 9 – TXEMPTY Transmission Registers Empty (cleared by writing FLEX_SPI_TDR)

Value	Description
0	As soon as data is written in FLEX_SPI_TDR.
1	FLEX_SPI_TDR and internal shift register are empty. If a transfer delay has been defined, TXEMPTY is set after the end of this delay.

Bit 8 – NSSR NSS Rising (cleared on read)

Value	Description
0	No rising edge detected on NSS pin since the last read of FLEX_SPI_SR.
1	A rising edge occurred on NSS pin since the last read of FLEX_SPI_SR.

Bit 3 – OVRES Overrun Error Status (cleared on read)
An overrun occurs when FLEX_SPI_RDR is loaded at least twice from the shift register since the last read of FLEX_SPI_RDR.

Value	Description
0	No overrun has been detected since the last read of FLEX_SPI_SR.
1	An overrun has occurred since the last read of FLEX_SPI_SR.

Bit 2 – MODF Mode Fault Error (cleared on read)

Value	Description
0	No mode fault has been detected since the last read of FLEX_SPI_SR.
1	A mode fault occurred since the last read of FLEX_SPI_SR.

Bit 1 – TDRE Transmit Data Register Empty (cleared by writing FLEX_SPI_TDR)
When FIFOs are disabled:
0: Data has been written to FLEX_SPI_TDR and not yet transferred to the internal shift register.
1: The last data written to FLEX_SPI_TDR has been transferred to the internal shift register.
TDRE is cleared when the SPI is disabled or at reset. Enabling the SPI sets the TDRE flag.
When FIFOs are enabled:
0: Transmit FIFO cannot accept more data.
1: Transmit FIFO can accept data; one or more data can be written according to TXRDYM field configuration.
TDRE behavior with FIFOs enabled is illustrated in 38.8.6.5 TXEMPTY, TDRE and RDRF Behavior.

Bit 0 – RDRF Receive Data Register Full (cleared by reading FLEX_SPI_RDR)
When FIFOs are disabled:
0: No data has been received since the last read of FLEX_SPI_RDR.
1: Data has been received and the received data has been transferred from the internal shift register to FLEX_SPI_RDR since the last read of FLEX_SPI_RDR.
When FIFOs are enabled:
0: Receive FIFO is empty; no data to read.
1: At least one unread data is in the Receive FIFO.
RDRF behavior with FIFOs enabled is illustrated in 38.8.6.5 TXEMPTY, TDRE and RDRF Behavior.

38.10.62 SPI Interrupt Enable Register

Name: FLEX_SPI_IER
Offset: 0x414
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the SPI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					CMP	UNDES	TXEMPTY	NSSR
Access					W	W	W	W
Reset					–	–	–	–

Bit	7	6	5	4	3	2	1	0
					OVRES	MODF	TDRE	RDRF
Access					W	W	W	W
Reset					–	–	–	–

Bit 31 – RXFPTEF RXFPTEF Interrupt Enable

Bit 30 – TXFPTEF TXFPTEF Interrupt Enable

Bit 29 – RXFTHF RXFTHF Interrupt Enable

Bit 28 – RXFFF RXFFF Interrupt Enable

Bit 27 – RXFEF RXFEF Interrupt Enable

Bit 26 – TXFTHF TXFTHF Interrupt Enable

Bit 25 – TXFFF TXFFF Interrupt Enable

Bit 24 – TXFEF TXFEF Interrupt Enable

Bit 11 – CMP Comparison Interrupt Enable

Bit 10 – UNDES Underrun Error Interrupt Enable

Bit 9 – TXEMPTY Transmission Registers Empty Enable

Bit 8 – NSSR NSS Rising Interrupt Enable

Bit 3 – OVRES Overrun Error Interrupt Enable

Bit 2 – MODF Mode Fault Error Interrupt Enable

Bit 1 – TDRE SPI Transmit Data Register Empty Interrupt Enable

Bit 0 – RDRF Receive Data Register Full Interrupt Enable

38.10.63 SPI Interrupt Disable Register

Name: FLEX_SPI_IDR
Offset: 0x418
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the SPI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					CMP	UNDES	TXEMPTY	NSSR
Access					W	W	W	W
Reset					–	–	–	–

Bit	7	6	5	4	3	2	1	0
					OVRES	MODF	TDRE	RDRF
Access					W	W	W	W
Reset					–	–	–	–

Bit 31 – RXFPTEF RXFPTEF Interrupt Disable

Bit 30 – TXFPTEF TXFPTEF Interrupt Disable

Bit 29 – RXFTHF RXFTHF Interrupt Disable

Bit 28 – RXFFF RXFFF Interrupt Disable

Bit 27 – RXFEF RXFEF Interrupt Disable

Bit 26 – TXFTHF TXFTHF Interrupt Disable

Bit 25 – TXFFF TXFFF Interrupt Disable

Bit 24 – TXFEF TXFEF Interrupt Disable

Bit 11 – CMP Comparison Interrupt Disable

Bit 10 – UNDES Underrun Error Interrupt Disable

Bit 9 – TXEMPTY Transmission Registers Empty Disable

Bit 8 – NSSR NSS Rising Interrupt Disable

Bit 3 – OVRES Overrun Error Interrupt Disable

Bit 2 – MODF Mode Fault Error Interrupt Disable

Bit 1 – TDRE SPI Transmit Data Register Empty Interrupt Disable

Bit 0 – RDRF Receive Data Register Full Interrupt Disable

38.10.64 SPI Interrupt Mask Register

Name: FLEX_SPI_IMR
Offset: 0x41C
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
					CMP	UNDES	TXEMPTY	NSSR
Access					R	R	R	R
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
					OVRES	MODF	TDRE	RDRF
Access					R	R	R	R
Reset					0	0	0	0

Bit 31 – RXFPTEF RXFPTEF Interrupt Mask

Bit 30 – TXFPTEF TXFPTEF Interrupt Mask

Bit 29 – RXFTHF RXFTHF Interrupt Mask

Bit 28 – RXFFF RXFFF Interrupt Mask

Bit 27 – RXFEF RXFEF Interrupt Mask

Bit 26 – TXFTHF TXFTHF Interrupt Mask

Bit 25 – TXFFF TXFFF Interrupt Mask

Bit 24 – TXFEF TXFEF Interrupt Mask

Bit 11 – CMP Comparison Interrupt Mask

Bit 10 – UNDES Underrun Error Interrupt Mask

Bit 9 – TXEMPTY Transmission Registers Empty Mask

Bit 8 – NSSR NSS Rising Interrupt Mask

Bit 3 – OVRES Overrun Error Interrupt Mask

Bit 2 – MODF Mode Fault Error Interrupt Mask

Bit 1 – TDRE SPI Transmit Data Register Empty Interrupt Mask

Bit 0 – RDRF Receive Data Register Full Interrupt Mask

38.10.65 SPI Chip Select Register

Name: FLEX_SPI_CSRx
Offset: 0x0430 + x*0x04 [x=0..3]
Reset: 0x00000000
Property: R/W

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

FLEX_SPI_CSRx must be written even if the user wants to use the default reset values. The BITS field is not updated with the translated value unless the register is written.

Bit	31	30	29	28	27	26	25	24
	DLYBCT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DLYBS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SCBR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BITS[3:0]				CSAAT	CSNAAT	NCPHA	CPOL
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:24 – DLYBCT[7:0] Delay Between Consecutive Transfers
This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.
When DLYBCT = 0, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.
Otherwise, the following equations determine the delay:
If FLEX_SPI_MR.BRSRCCLK = 0: DLYBCT = Delay Between Consecutive Transfers × $f_{peripheral\ clock}$ / 32
If FLEX_SPI_MR.BRSRCCLK = 1: DLYBCT = Delay Between Consecutive Transfers × f_{GCLK} / 32

Bits 23:16 – DLYBS[7:0] Delay Before SPCK
This field defines the delay from NPCS falling edge (activation) to the first valid SPCK transition.
When DLYBS = 0, the delay is half the SPCK clock period.
Otherwise, the following equations determine the delay:
If FLEX_SPI_MR.BRSRCCLK = 0: DLYBS = Delay Before SPCK × $f_{peripheral\ clock}$
If FLEX_SPI_MR.BRSRCCLK = 1: DLYBS = Delay Before SPCK × f_{GCLK}

Bits 15:8 – SCBR[7:0] Serial Clock Bit Rate
In Master mode, the SPI Interface uses a modulus counter to derive the SPCK bit rate from the clock defined by the bit BRSRCCLK. The bit rate is selected by writing a value from 1 to 255 in the SCBR field. The following equations determine the SPCK bit rate:
If FLEX_SPI_MR.BRSRCCLK = 0: SCBR = $f_{peripheral\ clock}$ / SPCK Bit Rate
If FLEX_SPI_MR.BRSRCCLK = 1: SCBR = f_{GCLK} / SPCK Bit Rate
Programming the SCBR field to 0 is forbidden. Triggering a transfer while SCBR is at 0 can lead to unpredictable results.
If BRSRCCLK = 1 in FLEX_SPI_MR, SCBR must be programmed with a value greater than 1.
At reset, SCBR is 0 and the user has to program it at a valid value before performing the first transfer.

Note: If one of the FLEX_SPI_CSRx.SCBR fields is set to 1, the other FLEX_SPI_CSRx.SCBR fields must be set to 1 as well, if they are used to process transfers. If they are not used to transfer data, they can be set at any value.

Bits 7:4 – BITS[3:0] Bits Per Transfer
See Note.
The BITS field determines the number of data bits transferred. Reserved values should not be used.

Value	Name	Description
0	8_BIT	8 bits for transfer
1	9_BIT	9 bits for transfer
2	10_BIT	10 bits for transfer
3	11_BIT	11 bits for transfer
4	12_BIT	12 bits for transfer
5	13_BIT	13 bits for transfer
6	14_BIT	14 bits for transfer
7	15_BIT	15 bits for transfer
8	16_BIT	16 bits for transfer
9–15	Reserved	

Bit 3 – CSAAT Chip Select Active After Transfer

Value	Description
0	The Peripheral Chip Select Line rises as soon as the last transfer is achieved.
1	The Peripheral Chip Select does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select.

Bit 2 – CSNAAT Chip Select Not Active After Transfer (Ignored if CSAAT = 1)

If FLEX_SPI_MR.BRSRCCLK = 0: $\frac{\text{DLYBCS}}{\text{f}_{\text{peripheral clock}}}$(if DLYBCS ≠ 0)

If FLEX_SPI_MR.BRSRCCLK = 1: $\frac{\text{DLYBCS}}{\text{f}_{GCLK}}$

If DLYBCS < 6, a minimum of six periods is introduced.

Value	Description
0	The Peripheral Chip Select does not rise between two transfers if the FLEX_SPI_TDR is reloaded before the end of the first transfer and if the two transfers occur on the same Chip Select.
1	The Peripheral Chip Select rises systematically after each transfer performed on the same slave. It remains inactive after the end of transfer for a minimal duration of:

Bit 1 – NCPHA Clock Phase
NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

Value	Description
0	Data are changed on the leading edge of SPCK and captured on the following edge of SPCK.
1	Data are captured on the leading edge of SPCK and changed on the following edge of SPCK.

Bit 0 – CPOL Clock Polarity
CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

Value	Description
0	The inactive state value of SPCK is logic level zero.
1	The inactive state value of SPCK is logic level one.

38.10.66 SPI FIFO Mode Register

Name: FLEX_SPI_FMR
Offset: 0x440
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
			RXFTHRES[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			TXFTHRES[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			RXRDYM[1:0]				TXRDYM[1:0]	
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 29:24 – RXFTHRES[5:0] Receive FIFO Threshold

Value	Description
0-16	Defines the Receive FIFO threshold value (number of data). The FLEX_SPI_SR.RXFTH flag will be set when Receive FIFO goes from “below” threshold state to “equal to or above” threshold state.

Bits 21:16 – TXFTHRES[5:0] Transmit FIFO Threshold

Value	Description
0-16	Defines the Transmit FIFO threshold value (number of data). The FLEX_SPI_SR.TXFTH flag will be set when Transmit FIFO goes from “above” threshold state to “equal to or below” threshold state.

Bits 5:4 – RXRDYM[1:0] Receive Data Register Full Mode
If FIFOs are enabled, the FLEX_SPI_SR.RDRF flag behaves as follows.

Value	Name	Description
0	ONE_DATA	RDRF will be at level ‘1’ when at least one unread data is in the Receive FIFO.
1	TWO_DATA	RDRF will be at level ‘1’ when at least two unread data are in the Receive FIFO. Cannot be used if FLEX_SPI_MR.PS =1.
2	FOUR_DATA	RDRF will be at level ‘1’ when at least four unread data are in the Receive FIFO. Cannot be used when FLEX_SPI_CSRx.BITS is greater than 0, or if FLEX_SPI_MR.MSTR =1, or if FLEX_SPI_MR.PS =1.

Bits 1:0 – TXRDYM[1:0] Transmit Data Register Empty Mode
If FIFOs are enabled, the FLEX_SPI_SR.TDRE flag behaves as follows.

Value	Name	Description
0	ONE_DATA	TDRE will be at level ‘1’ when at least one data can be written in the Transmit FIFO.
1	TWO_DATA	TDRE will be at level ‘1’ when at least two data can be written in the Transmit FIFO. Cannot be used if FLEX_SPI_MR.PS =1.

38.10.67 SPI FIFO Level Register

Name: FLEX_SPI_FLR
Offset: 0x444
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			RXFL[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TXFL[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bits 21:16 – RXFL[5:0] Receive FIFO Level

Value	Description
0	There is no unread data in the Receive FIFO.
1-16	Indicates the number of unread data in the Receive FIFO.

Bits 5:0 – TXFL[5:0] Transmit FIFO Level

Value	Description
0	There is no data in the Transmit FIFO.
1-16	Indicates the number of data in the Transmit FIFO.

38.10.68 SPI Comparison Register

Name: FLEX_SPI_CMPR
Offset: 0x448
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	VAL2[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	VAL2[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	VAL1[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	VAL1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:16 – VAL2[15:0] Second Comparison Value for Received Character

Value	Description
0-65535	The received character must be lower than or equal to the value of VAL2 and higher than or equal to VAL1 to set the FLEX_SPI_CSR.CMP flag.

Bits 15:0 – VAL1[15:0] First Comparison Value for Received Character

Value	Description
0-65535	The received character must be higher than or equal to the value of VAL1 and lower than or equal to VAL2 to set the FLEX_SPI_SR.CMP flag.

38.10.69 SPI Write Protection Mode Register

Name: FLEX_SPI_WPMR
Offset: 0x4E4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
						WPCREN	WPITEN	WPEN
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x535049	PASSWD	Writing any other value in this field aborts the write operation of bits WPEN, WPITEN and WPCREN. Always reads as 0.

Bit 2 – WPCREN Write Protection Control Enable

Value	Description
0	Disables the write protection on control register if WPKEY corresponds to 0x535049 (“SPI” in ASCII).
1	Enables the write protection on control register if WPKEY corresponds to 0x535049 (“SPI” in ASCII).

Bit 1 – WPITEN Write Protection Interrupt Enable

Value	Description
0	Disables the write protection on interrupt registers if WPKEY corresponds to 0x535049 (“SPI” in ASCII).
1	Enables the write protection on interrupt registers if WPKEY corresponds to 0x535049 (“SPI” in ASCII).

Bit 0 – WPEN Write Protection Enable

See 38.8.7 SPI Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII).

38.10.70 SPI Write Protection Status Register

Name: FLEX_SPI_WPSR
Offset: 0x4E8
Reset: 0x00000000
Property: Read-only

By default, any debugger or software read access to FLEX_US_WPSR clears the WPVS flag. To prevent this flag from being cleared on debugger read access, a specific bit must be enabled in the SFR module.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 15:8 – WPVSRC[7:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status

Value	Description
0	No write protect violation has occurred since the last read of FLEX_SPI_WPSR.
1	A write protect violation has occurred since the last read of FLEX_SPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

38.10.71 TWI Control Register

Name: FLEX_TWI_CR
Offset: 0x600
Reset: –
Property: Write-only

This register can only be written if the WPCREN bit is cleared in the TWI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
			FIFODIS	FIFOEN		LOCKCLR		THRCLR
Access			W	W		W		W
Reset			–	–		–		–

Bit	23	22	21	20	19	18	17	16
							ACMDIS	ACMEN
Access							W	W
Reset							–	–

Bit	15	14	13	12	11	10	9	8
	CLEAR	PECRQ	PECDIS	PECEN	SMBDIS	SMBEN	HSDIS	HSEN
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	SWRST	QUICK	SVDIS	SVEN	MSDIS	MSEN	STOP	START
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 29 – FIFODIS FIFO Disable

Value	Description
0	No effect.
1	Disable the Transmit and Receive FIFOs

Bit 28 – FIFOEN FIFO Enable

Value	Description
0	No effect.
1	Enable the Transmit and Receive FIFOs

Bit 26 – LOCKCLR Lock Clear

Value	Description
0	No effect.
1	Clear the TWI FSM lock.

Bit 24 – THRCLR Transmit Holding Register Clear

Value	Description
0	No effect.
1	Clear the Transmit Holding Register and set TXRDY, TXCOMP flags.

Bit 17 – ACMDIS Alternative Command Mode Disable

Value	Description
0	No effect.
1	Alternative Command mode disabled.

Bit 16 – ACMEN Alternative Command Mode Enable

Value	Description
0	No effect.
1	Alternative Command mode enabled.

Bit 15 – CLEAR Bus CLEAR Command

Value	Description
0	No effect.
1	If Master mode is enabled, send a bus clear command.

Bit 14 – PECRQ PEC Request

Value	Description
0	No effect.
1	A PEC check or transmission is requested.

Bit 13 – PECDIS Packet Error Checking Disable

Value	Description
0	No effect.
1	SMBus PEC (CRC) generation and check disabled.

Bit 12 – PECEN Packet Error Checking Enable

Value	Description
0	No effect.
1	SMBus PEC (CRC) generation and check enabled.

Bit 11 – SMBDIS SMBus Mode Disabled

Value	Description
0	No effect.
1	SMBus mode disabled.

Bit 10 – SMBEN SMBus Mode Enabled

Value	Description
0	No effect.
1	If SMBDIS = 0, SMBus mode enabled.

Bit 9 – HSDIS TWI High-Speed Mode Disabled

Value	Description
0	No effect.
1	High-speed mode disabled.

Bit 8 – HSEN TWI High-Speed Mode Enabled

Value	Description
0	No effect.
1	High-speed mode enabled.

Bit 7 – SWRST Software Reset

Value	Description
0	No effect.
1	Equivalent to a system reset.

Bit 6 – QUICK SMBus Quick Command

Value	Description
0	No effect.
1	If Master mode is enabled, an SMBus Quick Command is sent.

Bit 5 – SVDIS TWI Slave Mode Disabled

Value	Description
0	No effect.
1	Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in the case of a read operation. In a write operation, the character being transferred must be completely received before disabling.

Bit 4 – SVEN TWI Slave Mode Enabled
Switching from Master to Slave mode is only permitted when TXCOMP = 1.

Value	Description
0	No effect.
1	Enables the Slave mode (SVDIS must be written to 0).

Bit 3 – MSDIS TWI Master Mode Disabled

Value	Description
0	No effect.
1	The Master mode is disabled, all pending data is transmitted. The shifter and holding characters (if it contains data) are transmitted in case of write operation. In read operation, the character being transferred must be completely received before disabling.

Bit 2 – MSEN TWI Master Mode Enabled
Switching from Slave to Master mode is only permitted when TXCOMP = 1.

Value	Description
0	No effect.
1	Enables the Master mode (MSDIS must be written to 0).

Bit 1 – STOP Send a STOP Condition

Value	Description
0	No effect.
1	STOP condition is sent just after completing the current byte transmission in Master Read mode. – In single data byte master read, both START and STOP must be set. – In multiple data bytes master read, the STOP must be set after the last data received but one. – In Master Read mode, if a NACK bit is received, the STOP is automatically performed. – In master data write operation, a STOP condition will be sent after the transmission of the current data is finished.

Bit 0 – START Send a START Condition
This action is necessary when the TWI peripheral needs to read data from a slave. When configured in Master mode with a write operation, a frame is sent as soon as the user writes a character in the Transmit Holding Register (FLEX_TWI_THR).

Value	Description
0	No effect.
1	A frame beginning with a START bit is transmitted according to the features defined in the TWI Master Mode Register (FLEX_TWI_MMR).

38.10.72 TWI Control Register (FIFO_ENABLED)

Name: FLEX_TWI_CR (FIFO_ENABLED)
Offset: 0x600
Reset: –
Property: Write-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit), see 38.9.6.8 TWI Multiple Data Mode for details.

This register can only be written if the WPCREN bit is cleared in the TWI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
			FIFODIS	FIFOEN		TXFLCLR	RXFCLR	TXFCLR
Access			W	W		W	W	W
Reset			–	–		–	–	–

Bit	23	22	21	20	19	18	17	16
							ACMDIS	ACMEN
Access							W	W
Reset							–	–

Bit	15	14	13	12	11	10	9	8
	CLEAR	PECRQ	PECDIS	PECEN	SMBDIS	SMBEN	HSDIS	HSEN
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	SWRST	QUICK	SVDIS	SVEN	MSDIS	MSEN	STOP	START
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 29 – FIFODIS FIFO Disable

Value	Description
0	No effect.
1	Disable the Transmit and Receive FIFOs.

Bit 28 – FIFOEN FIFO Enable

Value	Description
0	No effect.
1	Enable the Transmit and Receive FIFOs.

Bit 26 – TXFLCLR Transmit FIFO Lock CLEAR

Value	Description
0	No effect.
1	Clears the Transmit FIFO Lock.

Bit 25 – RXFCLR Receive FIFO Clear

Value	Description
0	No effect.
1	Empties the Receive FIFO.

Bit 24 – TXFCLR Transmit FIFO Clear

Value	Description
0	No effect.
1	Empties the Transmit FIFO.

Bit 17 – ACMDIS Alternative Command Mode Disable

Value	Description
0	No effect.
1	Alternative Command mode disabled.

Bit 16 – ACMEN Alternative Command Mode Enable

Value	Description
0	No effect.
1	Alternative Command mode enabled.

Bit 15 – CLEAR Bus CLEAR Command

Value	Description
0	No effect.
1	If Master mode is enabled, send a bus clear command.

Bit 14 – PECRQ PEC Request

Value	Description
0	No effect.
1	A PEC check or transmission is requested.

Bit 13 – PECDIS Packet Error Checking Disable

Value	Description
0	No effect.
1	SMBus PEC (CRC) generation and check disabled.

Bit 12 – PECEN Packet Error Checking Enable

Value	Description
0	No effect.
1	SMBus PEC (CRC) generation and check enabled.

Bit 11 – SMBDIS SMBus Mode Disabled

Value	Description
0	No effect.
1	SMBus mode disabled.

Bit 10 – SMBEN SMBus Mode Enabled

Value	Description
0	No effect.
1	If SMBDIS = 0, SMBus mode enabled.

Bit 9 – HSDIS TWI High-Speed Mode Disabled

Value	Description
0	No effect.
1	High-speed mode disabled.

Bit 8 – HSEN TWI High-Speed Mode Enabled

Value	Description
0	No effect.
1	High-speed mode enabled.

Bit 7 – SWRST Software Reset

Value	Description
0	No effect.
1	Equivalent to a system reset.

Bit 6 – QUICK SMBus Quick Command

Value	Description
0	No effect.
1	If Master mode is enabled, a SMBus Quick Command is sent.

Bit 5 – SVDIS TWI Slave Mode Disabled

Value	Description
0	No effect.
1	Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in the case of a read operation. In a write operation, the character being transferred must be completely received before disabling.

Bit 4 – SVEN TWI Slave Mode Enabled
Switching from Master to Slave mode is only permitted when TXCOMP = 1.

Value	Description
0	No effect.
1	Enables the Slave mode (SVDIS must be written to 0).

Bit 3 – MSDIS TWI Master Mode Disabled

Value	Description
0	No effect.
1	The Master mode is disabled, all pending data is transmitted. The shifter and holding characters (if it contains data) are transmitted in the case of a write operation. In a read operation, the character being transferred must be completely received before disabling.

Bit 2 – MSEN TWI Master Mode Enabled
Switching from Slave to Master mode is only permitted when TXCOMP = 1.

Value	Description
0	No effect.
1	Enables the Master mode (MSDIS must be written to 0).

Bit 1 – STOP Send a STOP Condition

Value	Description
0	No effect.
1	STOP condition is sent just after completing the current byte transmission in Master Read mode. – In single data byte master read, both START and STOP must be set. – In multiple data bytes master read, the STOP must be set after the last data received but one. – In Master Read mode, if a NACK bit is received, the STOP is automatically performed. – In master data write operation, a STOP condition will be sent after the transmission of the current data is finished.

Bit 0 – START Send a START Condition
This action is necessary when the TWI peripheral needs to read data from a slave. When configured in Master mode with a write operation, a frame is sent as soon as the user writes a character in the Transmit Holding Register (FLEX_TWI_THR).

Value	Description
0	No effect.
1	A frame beginning with a START bit is transmitted according to the features defined in the TWI Master Mode Register (FLEX_TWI_MMR).

38.10.73 TWI Master Mode Register

Name: FLEX_TWI_MMR
Offset: 0x604
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
								NOAP
Access								R/W
Reset								0

Bit	23	22	21	20	19	18	17	16
		DADR[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				MREAD			IADRSZ[1:0]	
Access				R/W			R/W	R/W
Reset				0			0	0

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 24 – NOAP No Auto-Stop On NACK Error

Value	Description
0	A stop condition is sent automatically upon Not-Acknowledge error detection.
1	No automatic action is performed upon Not-Acknowledge error detection.

Bits 22:16 – DADR[6:0] Device Address
The device address is used to access slave devices in Read or Write mode. Those bits are only used in Master mode.

Bit 12 – MREAD Master Read Direction

Value	Description
0	Master write direction.
1	Master read direction.

Bits 9:8 – IADRSZ[1:0] Internal Device Address Size

Value	Name	Description
0	NONE	No internal device address
1	1_BYTE	One-byte internal device address
2	2_BYTE	Two-byte internal device address
3	3_BYTE	Three-byte internal device address

38.10.74 TWI Slave Mode Register

Name: FLEX_TWI_SMR
Offset: 0x608
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the TWI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
		SADR[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
		MASK[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SNIFF	SCLWSDIS		SADAT	SMHH	SMDA		NACKEN
Access	R/W	R/W		R/W	R/W	R/W		R/W
Reset	0	0		0	0	0		0

Bits 22:16 – SADR[6:0] Slave Address
The slave device address is used in Slave mode in order to be accessed by master devices in Read or Write mode. SADR must be programmed before enabling the Slave mode or after a general call. Writes at other times have no effect.

Bits 14:8 – MASK[6:0] Slave Address Mask
A mask can be applied on the slave device address in Slave mode in order to allow multiple address answer. For each bit of the MASK field set to one, the corresponding SADR bit will be masked.
If the MASK field is set to 0, no mask is applied to the SADR field.

Bit 7 – SNIFF Slave Sniffer Mode

Value	Description
0	Slave Sniffer mode is disabled.
1	Slave Sniffer mode is enabled.

Bit 6 – SCLWSDIS Clock Wait State Disable

Value	Description
0	No effect.
1	Clock stretching disabled in Slave mode, OVRE and UNRE will indicate overrun and underrun.

Bit 4 – SADAT Slave Address Treated as Data

Value	Description
0	Slave address is handled normally (will not trig RXRDY flag and will not fill FLEX_TWI_RHR upon reception).
1	Slave address is handled as data field, RXRDY will be set and FLEX_TWI_RHR filled upon slave address reception.

Bit 3 – SMHH SMBus Host Header

Value	Description
0	Acknowledge of the SMBus Host Header disabled.
1	Acknowledge of the SMBus Host Header enabled.

Bit 2 – SMDA SMBus Default Address

Value	Description
0	Acknowledge of the SMBus Default Address disabled.
1	Acknowledge of the SMBus Default Address enabled.

Bit 0 – NACKEN Slave Receiver Data Phase NACK Enable

Value	Description
0	Normal value to be returned in the ACK cycle of the data phase in Slave Receiver mode.
1	NACK value to be returned in the ACK cycle of the data phase in Slave Receiver mode.

38.10.75 TWI Internal Address Register

Name: FLEX_TWI_IADR
Offset: 0x60C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	IADR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IADR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IADR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – IADR[23:0] Internal Address
0, 1, 2 or 3 bytes depending on IADRSZ.

38.10.76 TWI Clock Waveform Generator Register

Name: FLEX_TWI_CWGR
Offset: 0x610
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the TWI Write Protection Mode Register.

FLEX_TWI_CWGR is only used in Master mode.

Bit	31	30	29	28	27	26	25	24
			HOLD[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
				BRSRCCLK		CKDIV[2:0]		
Access				R/W		R/W	R/W	R/W
Reset				0		0	0	0

Bit	15	14	13	12	11	10	9	8
	CHDIV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CLDIV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 29:24 – HOLD[5:0] TWD Hold Time Versus TWCK Falling
If High-speed mode is selected TWD is internally modified on the TWCK falling edge to meet the I^2C specified maximum hold time, else if High-speed mode is not configured TWD is kept unchanged after TWCK falling edge for a period of (HOLD + 3) × $t_{peripheral\ clock}$.

Bit 20 – BRSRCCLK Bit Rate Source Clock

Value	Name	Description
0	PERIPH_CLK	The peripheral clock is the source clock for the bit rate generation.
1	GCLK	GCLK is the source clock for the bit rate generation, thus the bit rate can be independent of the core/peripheral clock.

Bits 18:16 – CKDIV[2:0] Clock Divider
The CKDIV is used to increase both SCL high and low periods.

Bits 15:8 – CHDIV[7:0] Clock High Divider
The SCL high period is defined as follows:
If BRSRCCLK = 0: CHDIV = ((t_{high} / $t_{peripheral\ clock}$) - 3) / 2^{CKDIV}
If BRSRCCLK = 1: CHDIV = (t_{high} / t_{ext_ck}) / 2^{CKDIV}

Bits 7:0 – CLDIV[7:0] Clock Low Divider
The SCL low period is defined as follows:
If BRSRCCLK = 0: CLDIV = ((t_{low} / $t_{peripheral\ clock}$) - 3) / 2^{CKDIV}
If BRSRCCLK = 1: CLDIV = (t_{low} / t_{ext_ck}) / 2^{CKDIV}

Value	Name	Description
0	PERIPH_CLK	The peripheral clock is used to generate the TWI bit rate.
1	GCLK	GCLK is used to generate the TWI bit rate.

38.10.77 TWI Status Register

Name: FLEX_TWI_SR
Offset: 0x620
Reset: 0x0300F009
Property: Read-only

If enabled in the SFR module (default is disabled), a debugger read access does not clear the clear-on-read flags.

Bit	31	30	29	28	27	26	25	24
						SR	SDA	SCL
Access						R	R	R
Reset						0	1	1

Bit	23	22	21	20	19	18	17	16
	LOCK		SMBHHM	SMBDAM	PECERR	TOUT		MCACK
Access	R		R	R	R	R		R
Reset	0		0	0	0	0		0

Bit	15	14	13	12	11	10	9	8
					EOSACC	SCLWS	ARBLST	NACK
Access					R	R	R	R
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	UNRE	OVRE	GACC	SVACC	SVREAD	TXRDY	RXRDY	TXCOMP
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	0	0	1

Bit 26 – SR Start Repeated

Value	Description
0	No repeated start has been detected since last FLEX_TWI_SR read.
1	At least one repeated start has been detected since last FLEX_TWI_SR read.

Bit 25 – SDA SDA Line Value

Value	Description
0	SDA line sampled value is ‘0’.
1	SDA line sampled value is ‘1’.

Bit 24 – SCL SCL Line Value

Value	Description
0	SCL line sampled value is ‘0’.
1	SCL line sampled value is ‘1.’

Bit 23 – LOCK TWI Lock Due to Frame Errors

Value	Description
0	The TWI is not locked.
1	The TWI is locked due to frame errors (see 38.9.3.12 Handling Errors in Alternative Command and 38.9.6 TWI FIFOs).

Bit 21 – SMBHHM SMBus Host Header Address Match (cleared on read)

Value	Description
0	No SMBus Host Header Address received.
1	A SMBus Host Header Address was received.

Bit 20 – SMBDAM SMBus Default Address Match (cleared on read)

Value	Description
0	No SMBus Default Address received.
1	A SMBus Default Address was received.

Bit 19 – PECERR PEC Error (cleared on read)

Value	Description
0	No SMBus PEC error occurred.
1	A SMBus PEC error occurred.

Bit 18 – TOUT Timeout Error (cleared on read)

Value	Description
0	No SMBus timeout occurred.
1	SMBus timeout occurred.

Bit 16 – MCACK Master Code Acknowledge (cleared on read)
MACK used in Slave mode:

Value	Description
0	No master code has been received.
1	A master code has been received.

Bit 11 – EOSACC End Of Slave Access (cleared on read)
This bit is only used in Slave mode.
EOSACC behavior can be seen in figures Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

Value	Description
0	A slave access is being performing.
1	The Slave Access is finished. End Of Slave Access is automatically set as soon as SVACC is reset.

Bit 10 – SCLWS Clock Wait State
This bit is only used in Slave mode.
SCLWS behavior can be seen in figures Clock Stretching in Read Mode and Clock Stretching in Write Mode.

Value	Description
0	The clock is not stretched.
1	The clock is stretched. FLEX_TWI_THR / FLEX_TWI_RHR buffer is not filled / emptied before the transmission / reception of a new character.

Bit 9 – ARBLST Arbitration Lost (cleared on read)
This bit is only used in Master mode.

Value	Description
0	Arbitration won.
1	Arbitration lost. Another master of the TWI bus has won the multi-master arbitration. TXCOMP is set at the same time.

Bit 8 – NACK Not Acknowledged (cleared on read)
NACK used in Master mode:
0: Each data byte has been correctly received by the far-end side TWI slave component.
1: A data or address byte has not been acknowledged by the slave component. Set at the same time as TXCOMP.
NACK used in Slave Read mode:
0: Each data byte has been correctly received by the master.
1: In Read mode, a data byte has not been acknowledged by the master. When NACK is set, the user must not fill FLEX_TWI_THR even if TXRDY is set, because it means that the master will stop the data transfer or reinitiate it..
Note that in Slave Write mode, all data are acknowledged by the TWI.

Bit 7 – UNRE Underrun Error (cleared on read)
This bit is only used in Slave mode if clock stretching is disabled.

Value	Description
0	FLEX_TWI_THR has been filled on time.

Value	Description
1	FLEX_TWI_THR has not been filled on time.

Bit 6 – OVRE Overrun Error (cleared on read)
This bit is only used in Slave mode if clock stretching is disabled.

Value	Description
0	FLEX_TWI_RHR has not been loaded while RXRDY was set.
1	FLEX_TWI_RHR has been loaded while RXRDY was set. Reset by read in FLEX_TWI_SR when TXCOMP is set.

Bit 5 – GACC General Call Access (cleared on read)
This bit is only used in Slave mode.
GACC behavior can be seen in figure Master Performs a General Call.

Value	Description
0	No general call has been detected.
1	A general call has been detected. After the detection of general call, if need be, the user may acknowledge this access and decode the following bytes and respond according to the value of the bytes.

Bit 4 – SVACC Slave Access
This bit is only used in Slave mode.
SVACC behavior can be seen in figures Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

Value	Description
0	TWI is not addressed. SVACC is automatically cleared after a NACK or a STOP condition is detected.
1	Indicates that the address decoding sequence has matched (a master has sent SADR). SVACC remains high until a NACK or a STOP condition is detected.

Bit 3 – SVREAD Slave Read
This bit is only used in Slave mode. When SVACC is low (no slave access has been detected) SVREAD is irrelevant.
SVREAD behavior can be seen in figures Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

Value	Description
0	Indicates that a write access is performed by a master.
1	Indicates that a read access is performed by a master.

Bit 2 – TXRDY Transmit Holding Register Ready (cleared by writing FLEX_TWI_THR)
TXRDY used in Master mode:
0: The transmit holding register has not been transferred into the internal shifter. Set to 0 when writing into FLEX_TWI_THR.
1: As soon as a data byte is transferred from FLEX_TWI_THR to internal shifter or if a NACK error is detected, TXRDY is set at the same time as TXCOMP and NACK. TXRDY is also set when MSEN is set (enables TWI).
TXRDY behavior in Master mode can be seen in figures Master Write with One Data Byte,Master Write with Multiple Data Bytes and Master Write with One Byte Internal Address and Multiple Data Bytes.
TXRDY used in Slave mode:
0: As soon as data is written in FLEX_TWI_THR, until this data has been transmitted and acknowledged (ACK or NACK).
1: Indicates that FLEX_TWI_THR is empty and that data has been transmitted and acknowledged.
If TXRDY is high and if a NACK has been detected, the transmission will be stopped. Thus when TRDY = NACK = 1, the user must not fill FLEX_TWI_THR to avoid losing it.
TXRDY behavior in Slave mode can be seen in figures Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.
When FIFOs are enabled:
0: Transmit FIFO is full and cannot accept more data.
1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration.

TXRDY behavior with FIFOs enabled is illustrated in TXRDY and RXRDY Behavior.

Bit 1 – RXRDY Receive Holding Register Ready (cleared when reading FLEX_TWI_RHR)
When FIFOs are disabled:
0: No character has been received since the last FLEX_TWI_RHR read operation.
1: A byte has been received in FLEX_TWI_RHR since the last read.
RXRDY behavior in Master mode can be seen in figure Master Read with Multiple Data Bytes.
RXRDY behavior in Slave mode can be seen in figures Write Access Ordered by a Master, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.
When FIFOs are enabled:
0: Receive FIFO is empty; no data to read.
1: At least one unread data is in the Receive FIFO.
RXRDY behavior with FIFO enabled is illustrated in TXRDY and RXRDY Behavior.

Bit 0 – TXCOMP Transmission Completed (cleared by writing FLEX_TWI_THR)
TXCOMP used in Master mode:
0: During the length of the current frame.
1: When both the holding register and the internal shifter are empty and STOP condition has been sent.
TXCOMP behavior in Master mode can be seen in figures Master Write with One Byte Internal Address and Multiple Data Bytes and Master Read with Multiple Data Bytes.
TXCOMP used in Slave mode:
0: As soon as a Start is detected.
1: After a Stop or a Repeated Start + an address different from SADR is detected.
TXCOMP behavior in Slave mode can be seen in figures Clock Stretching in Read Mode, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

38.10.78 TWI Status Register (FIFO ENABLED)

Name: FLEX_TWI_SR (FIFO_ENABLED)
Offset: 0x620
Reset: 0x0300F009
Property: Read-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit), see TWI Multiple Data Mode for details.

If enabled in the SFR module (default is disabled), a debugger read access does not clear the clear-on-read flags.

Bit	31	30	29	28	27	26	25	24
						SR	SDA	SCL
Access						R	R	R
Reset						0	1	1

Bit	23	22	21	20	19	18	17	16
	TXFLOCK		SMBHHM	SMBDAM	PECERR	TOUT		MCACK
Access	R		R	R	R	R		R
Reset	0		0	0	0	0		0

Bit	15	14	13	12	11	10	9	8
					EOSACC	SCLWS	ARBLST	NACK
Access					R	R	R	R
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	UNRE	OVRE	GACC	SVACC	SVREAD	TXRDY	RXRDY	TXCOMP
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	1	0	0	1

Bit 26 – SR Start Repeated

Value	Description
0	No repeated start has been detected since last FLEX_TWI_SR read.
1	At least one repeated start has been detected since last FLEX_TWI_SR read.

Bit 25 – SDA SDA Line Value

Value	Description
0	SDA line sampled value is ‘0’.
1	SDA line sampled value is ‘1’.

Bit 24 – SCL SCL Line Value

Value	Description
0	SCL line sampled value is ‘0’.
1	SCL line sampled value is ‘1.’

Bit 23 – TXFLOCK Transmit FIFO Lock

Value	Description
0	The Transmit FIFO is not locked.
1	The Transmit FIFO is locked.

Bit 21 – SMBHHM SMBus Host Header Address Match (cleared on read)

Value	Description
0	No SMBus Host Header Address received.
1	A SMBus Host Header Address was received.

Bit 20 – SMBDAM SMBus Default Address Match (cleared on read)

Value	Description
0	No SMBus Default Address received.
1	A SMBus Default Address was received.

Bit 19 – PECERR PEC Error (cleared on read)

Value	Description
0	No SMBus PEC error occurred.
1	A SMBus PEC error occurred.

Bit 18 – TOUT Timeout Error (cleared on read)

Value	Description
0	No SMBus timeout occurred.
1	SMBus timeout occurred.

Bit 16 – MCACK Master Code Acknowledge (cleared on read)
MACK used in Slave mode:

Value	Description
0	No master code has been received.
1	A master code has been received.

Bit 11 – EOSACC End Of Slave Access (cleared on read)
This bit is only used in Slave mode.
EOSACC behavior can be seen in figures Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

Value	Description
0	A Slave Access is being performed.
1	The Slave Access is finished. End Of Slave Access is automatically set as soon as SVACC is reset.

Bit 10 – SCLWS Clock Wait State
This bit is only used in Slave mode.
SCLWS behavior can be seen in figures Clock Stretching in Read Mode and Clock Stretching in Write Mode.

Value	Description
0	The clock is not stretched.
1	The clock is stretched. FLEX_TWI_THR / FLEX_TWI_RHR buffer is not filled / emptied before the transmission / reception of a new character.

Bit 9 – ARBLST Arbitration Lost (cleared on read)
This bit is only used in Master mode.

Value	Description
0	Arbitration won.
1	Arbitration lost. Another master of the TWI bus has won the multi-master arbitration. TXCOMP is set at the same time.

Bit 8 – NACK Not Acknowledged (cleared on read)
NACK used in Master mode:
0: Each data byte has been correctly received by the far-end side TWI slave component.
1: A data or address byte has not been acknowledged by the slave component. Set at the same time as TXCOMP.
NACK used in Slave Read mode:
0: Each data byte has been correctly received by the master.
1: In Read mode, a data byte has not been acknowledged by the master. When NACK is set the user must not fill FLEX_TWI_THR even if TXRDY is set, because it means that the master will stop the data transfer or re initiate it. Note that in Slave Write mode all data are acknowledged by the TWI.

Bit 7 – UNRE Underrun Error (cleared on read)
This bit is only used in Slave mode if clock stretching is disabled.

Value	Description
0	FLEX_TWI_THR has been filled on time.

Value	Description
1	FLEX_TWI_THR has not been filled on time.

Bit 6 – OVRE Overrun Error (cleared on read)
This bit is only used in Slave mode if clock stretching is disabled.

Value	Description
0	FLEX_TWI_RHR has not been loaded while RXRDY was set.
1	FLEX_TWI_RHR has been loaded while RXRDY was set. Reset by read in FLEX_TWI_SR when TXCOMP is set.

Bit 5 – GACC General Call Access (cleared on read)
This bit is only used in Slave mode.
GACC behavior can be seen in figure Master Performs a General Call.

Value	Description
0	No general call has been detected.
1	A general call has been detected. After the detection of general call, if need be, the user may acknowledge this access and decode the following bytes and respond according to the value of the bytes.

Bit 4 – SVACC Slave Access
This bit is only used in Slave mode.
SVACC behavior can be seen in figures Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

Value	Description
0	TWI is not addressed. SVACC is automatically cleared after a NACK or a STOP condition is detected.
1	Indicates that the address decoding sequence has matched (a master has sent SADR). SVACC remains high until a NACK or a STOP condition is detected.

Bit 3 – SVREAD Slave Read
This bit is only used in Slave mode. When SVACC is low (no slave access has been detected) SVREAD is irrelevant.
SVREAD behavior can be seen in figures Read Access Ordered by a Master, Write Access Ordered by a Master, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

Value	Description
0	Indicates that a write access is performed by a master.
1	Indicates that a read access is performed by a master.

Bit 2 – TXRDY Transmit Holding Register Ready (cleared by writing FLEX_TWI_THR)
TXRDY used in Master mode:
0: The transmit holding register has not been transferred into the internal shifter. Set to 0 when writing into FLEX_TWI_THR.
1: As soon as a data byte is transferred from FLEX_TWI_THR to internal shifter or if a NACK error is detected, TXRDY is set at the same time as TXCOMP and NACK. TXRDY is also set when MSEN is set (enables TWI).
TXRDY behavior in Master mode can be seen in figures Master Write with One Data Byte,Master Write with Multiple Data Bytes and Master Write with One Byte Internal Address and Multiple Data Bytes.
TXRDY used in Slave mode:
0: As soon as data is written in FLEX_TWI_THR, until this data has been transmitted and acknowledged (ACK or NACK).
1: Indicates that FLEX_TWI_THR is empty and that data has been transmitted and acknowledged.
If TXRDY is high and if a NACK has been detected, the transmission will be stopped. Thus when TRDY = NACK = 1, the user must not fill FLEX_TWI_THR to avoid losing it.
TXRDY behavior in Slave mode can be seen in figures Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.
When FIFOs are enabled:
0: Transmit FIFO is full and cannot accept more data.
1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration.

TXRDY behavior with FIFOs enabled is illustrated in TXRDY and RXRDY Behavior.

Bit 1 – RXRDY Receive Holding Register Ready (cleared when reading FLEX_TWI_RHR)
When FIFOs are disabled:
0: No character has been received since the last FLEX_TWI_RHR read operation.
1: A byte has been received in FLEX_TWI_RHR since the last read.
RXRDY behavior in Master mode can be seen in figure Master Read with Multiple Data Bytes.
RXRDY behavior in Slave mode can be seen in figures Write Access Ordered by a Master, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.
When FIFOs are enabled:
0: Receive FIFO is empty; no data to read.
1: At least one unread data is in the Receive FIFO.
RXRDY behavior with FIFO enabled is illustrated in TXRDY and RXRDY Behavior.

Bit 0 – TXCOMP Transmission Completed (cleared by writing FLEX_TWI_THR)
TXCOMP used in Master mode:
0: During the length of the current frame.
1: When both holding register and internal shifter are empty and STOP condition has been sent.
TXCOMP behavior in Master mode can be seen in figures Master Write with One Byte Internal Address and Multiple Data Bytes and Master Read with Multiple Data Bytes.
TXCOMP used in Slave mode:
0: As soon as a Start is detected.
1: After a Stop or a Repeated Start + an address different from SADR is detected.
TXCOMP behavior in Slave mode can be seen in figures Clock Stretching in Read Mode, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

38.10.79 TWI Interrupt Enable Register

Name: FLEX_TWI_IER
Offset: 0x624
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the TWI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			SMBHHM	SMBDAM	PECERR	TOUT		MCACK
Access			W	W	W	W		W
Reset			–	–	–	–		–

Bit	15	14	13	12	11	10	9	8
	TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	UNRE	OVRE	GACC	SVACC		TXRDY	RXRDY	TXCOMP
Access	W	W	W	W		W	W	W
Reset	–	–	–	–		–	–	–

Bit 21 – SMBHHM SMBus Host Header Address Match Interrupt Enable

Bit 20 – SMBDAM SMBus Default Address Match Interrupt Enable

Bit 19 – PECERR PEC Error Interrupt Enable

Bit 18 – TOUT Timeout Error Interrupt Enable

Bit 16 – MCACK Master Code Acknowledge Interrupt Enable

Bit 15 – TXBUFE Transmit Buffer Empty Interrupt Enable

Bit 14 – RXBUFF Receive Buffer Full Interrupt Enable

Bit 13 – ENDTX End of Transmit Buffer Interrupt Enable

Bit 12 – ENDRX End of Receive Buffer Interrupt Enable

Bit 11 – EOSACC End Of Slave Access Interrupt Enable

Bit 10 – SCL_WS Clock Wait State Interrupt Enable

Bit 9 – ARBLST Arbitration Lost Interrupt Enable

Bit 8 – NACK Not Acknowledge Interrupt Enable

Bit 7 – UNRE Underrun Error Interrupt Enable

Bit 6 – OVRE Overrun Error Interrupt Enable

Bit 5 – GACC General Call Access Interrupt Enable

Bit 4 – SVACC Slave Access Interrupt Enable

Bit 2 – TXRDY Transmit Holding Register Ready Interrupt Enable

Bit 1 – RXRDY Receive Holding Register Ready Interrupt Enable

Bit 0 – TXCOMP Transmission Completed Interrupt Enable

38.10.80 TWI Interrupt Disable Register

Name: FLEX_TWI_IDR
Offset: 0x628
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the TWI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			SMBHHM	SMBDAM	PECERR	TOUT		MCACK
Access			W	W	W	W		W
Reset			–	–	–	–		–

Bit	15	14	13	12	11	10	9	8
	TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	UNRE	OVRE	GACC	SVACC		TXRDY	RXRDY	TXCOMP
Access	W	W	W	W		W	W	W
Reset	–	–	–	–		–	–	–

Bit 21 – SMBHHM SMBus Host Header Address Match Interrupt Disable

Bit 20 – SMBDAM SMBus Default Address Match Interrupt Disable

Bit 19 – PECERR PEC Error Interrupt Disable

Bit 18 – TOUT Timeout Error Interrupt Disable

Bit 16 – MCACK Master Code Acknowledge Interrupt Disable

Bit 15 – TXBUFE Transmit Buffer Empty Interrupt Disable

Bit 14 – RXBUFF Receive Buffer Full Interrupt Disable

Bit 13 – ENDTX End of Transmit Buffer Interrupt Disable

Bit 12 – ENDRX End of Receive Buffer Interrupt Disable

Bit 11 – EOSACC End Of Slave Access Interrupt Disable

Bit 10 – SCL_WS Clock Wait State Interrupt Disable

Bit 9 – ARBLST Arbitration Lost Interrupt Disable

Bit 8 – NACK Not Acknowledge Interrupt Disable

Bit 7 – UNRE Underrun Error Interrupt Disable

Bit 6 – OVRE Overrun Error Interrupt Disable

Bit 5 – GACC General Call Access Interrupt Disable

Bit 4 – SVACC Slave Access Interrupt Disable

Bit 2 – TXRDY Transmit Holding Register Ready Interrupt Disable

Bit 1 – RXRDY Receive Holding Register Ready Interrupt Disable

Bit 0 – TXCOMP Transmission Completed Interrupt Disable

38.10.81 TWI Interrupt Mask Register

Name: FLEX_TWI_IMR
Offset: 0x62C
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			SMBHHM	SMBDAM	PECERR	TOUT		MCACK
Access			R	R	R	R		R
Reset			0	0	0	0		0

Bit	15	14	13	12	11	10	9	8
	TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	UNRE	OVRE	GACC	SVACC		TXRDY	RXRDY	TXCOMP
Access	R	R	R	R		R	R	R
Reset	0	0	0	0		0	0	0

Bit 21 – SMBHHM SMBus Host Header Address Match Interrupt Mask

Bit 20 – SMBDAM SMBus Default Address Match Interrupt Mask

Bit 19 – PECERR PEC Error Interrupt Mask

Bit 18 – TOUT Timeout Error Interrupt Mask

Bit 16 – MCACK Master Code Acknowledge Interrupt Mask

Bit 15 – TXBUFE Transmit Buffer Empty Interrupt Mask

Bit 14 – RXBUFF Receive Buffer Full Interrupt Mask

Bit 13 – ENDTX End of Transmit Buffer Interrupt Mask

Bit 12 – ENDRX End of Receive Buffer Interrupt Mask

Bit 11 – EOSACC End Of Slave Access Interrupt Mask

Bit 10 – SCL_WS Clock Wait State Interrupt Mask

Bit 9 – ARBLST Arbitration Lost Interrupt Mask

Bit 8 – NACK Not Acknowledge Interrupt Mask

Bit 7 – UNRE Underrun Error Interrupt Mask

Bit 6 – OVRE Overrun Error Interrupt Mask

Bit 5 – GACC General Call Access Interrupt Mask

Bit 4 – SVACC Slave Access Interrupt Mask

Bit 2 – TXRDY Transmit Holding Register Ready Interrupt Mask

Bit 1 – RXRDY Receive Holding Register Ready Interrupt Mask

Bit 0 – TXCOMP Transmission Completed Interrupt Mask

38.10.82 TWI Receive Holding Register

Name: FLEX_TWI_RHR
Offset: 0x630
Reset: 0x00000000
Property: Read-only

If FIFO is enabled (FLEX_TWI_CR.FIFOEN bit) and FLEX_TWI_FMR.RXRDYM = 0, see TWI Single Data Mode for details.

If enabled in the SFR module (default is disabled), a debugger read access does not clear the FLEX_TWI_SR.RXRDY flag.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
				ASTATE[1:0]		PSTATE	SSTATE[1:0]	
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RXDATA[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 12:11 – ASTATE[1:0] Acknowledge State (Slave Sniffer Mode only)

Value	Name	Description
0	NONE	No Acknowledge or Nackowledge detected after previously logged data
1	ACK	Acknowledge (A) detected after previously logged data
2	NACK	Nackowledge (NA) detected after previously logged data
3	UNDEF	Not defined

Bit 10 – PSTATE Stop State (Slave Sniffer Mode only)

Value	Description
0	No STOP (P) detected after previous logged data.
1	Stop detected (P) after previous logged data.

Bits 9:8 – SSTATE[1:0] Start State (Slave Sniffer Mode only)

Value	Name	Description
0	NOSTART	No START detected with the logged data
1	START	START (S) detected with the logged data
2	RSTART	Repeated START (Sr) detected with the logged data
3	UNDEF	Not defined

Bits 7:0 – RXDATA[7:0] Master or Slave Receive Holding Data

38.10.83 TWI Receive Holding Register (FIFO Enabled)

Name: FLEX_TWI_RHR (FIFO_ENABLED)
Offset: 0x630
Reset: 0x00000000
Property: Read-only

If FIFO is enabled (FLEX_TWI_CR.FIFOEN bit) and FLEX_TWI_FMR.RXRDYM > 0, see TWI Multiple Data Mode for details.

If enabled in the SFR module (default is disabled), a debugger read access does not modify the Receive FIFO.

Bit	31	30	29	28	27	26	25	24
	RXDATA3[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RXDATA2[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RXDATA1[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RXDATA0[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:24 – RXDATA3[7:0] Master or Slave Receive Holding Data 3

Bits 23:16 – RXDATA2[7:0] Master or Slave Receive Holding Data 2

Bits 15:8 – RXDATA1[7:0] Master or Slave Receive Holding Data 1

Bits 7:0 – RXDATA0[7:0] Master or Slave Receive Holding Data 0

38.10.84 TWI Transmit Holding Register

Name: FLEX_TWI_THR
Offset: 0x634
Reset: –
Property: Write-only

If FIFO is enabled (FIFOEN bit in FLEX_TWI_CR) and FLEX_TWI_FMR.TXRDYM = 0, see TWI Single Data Mode for details.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	TXDATA[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 7:0 – TXDATA[7:0] Master or Slave Transmit Holding Data

38.10.85 TWI Transmit Holding Register (FIFO Enabled)

Name: FLEX_TWI_THR (FIFO_ENABLED)
Offset: 0x634
Reset: –
Property: Write-only

If FIFO is enabled (FLEX_US_CR.FIFOEN bit) and FLEX_TWI_FMR.TXRDYM > 0, see TWI Multiple Data Mode for details.

Bit	31	30	29	28	27	26	25	24
	TXDATA3[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	23	22	21	20	19	18	17	16
	TXDATA2[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	15	14	13	12	11	10	9	8
	TXDATA1[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit	7	6	5	4	3	2	1	0
	TXDATA0[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 31:24 – TXDATA3[7:0] Master or Slave Transmit Holding Data 3

Bits 23:16 – TXDATA2[7:0] Master or Slave Transmit Holding Data 2

Bits 15:8 – TXDATA1[7:0] Master or Slave Transmit Holding Data 1

Bits 7:0 – TXDATA0[7:0] Master or Slave Transmit Holding Data 0

38.10.86 TWI SMBus Timing Register

Name: FLEX_TWI_SMBTR
Offset: 0x638
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	THMAX[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TLOWM[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TLOWS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					PRESC[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 31:24 – THMAX[7:0] Clock High Maximum Cycles
Clock cycles in clock high maximum count. Prescaled by PRESC. Used for bus free detection. Used to time THIGH:MAX.

Bits 23:16 – TLOWM[7:0] Master Clock Stretch Maximum Cycles

Value	Description
0	TLOW:MEXT timeout check disabled.
1-255	Clock cycles in master maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:MEXT.

Bits 15:8 – TLOWS[7:0] Slave Clock Stretch Maximum Cycles

Value	Description
0	TLOW:SEXT timeout check disabled.
1-255	Clock cycles in slave maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:SEXT.

Bits 3:0 – PRESC[3:0] SMBus Clock Prescaler
Used to specify how to prescale the TLOWS, TLOWM and THMAX counters in SMBTR. Counters are prescaled according to the following formula: PRESC = Log(fMCK / $f_{Prescaled}$) / Log(2) - 1

38.10.87 TWI Alternative Command Register

Name: FLEX_TWI_ACR
Offset: 0x640
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
							NPEC	NDIR
Access							R/W	R/W
Reset							0	0

Bit	23	22	21	20	19	18	17	16
	NDATAL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
							PEC	DIR
Access							R/W	R/W
Reset							0	0

Bit	7	6	5	4	3	2	1	0
	DATAL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 25 – NPEC Next PEC Request (SMBus Mode only)

Value	Description
0	The next transfer does not use a PEC byte.
1	The next transfer uses a PEC byte.

Bit 24 – NDIR Next Transfer Direction

Value	Description
0	Write direction.
1	Read direction.

Bits 23:16 – NDATAL[7:0] Next Data Length

Value	Description
0	No data to send (see 38.9.3.11 Alternative Command).
1-255	Number of bytes to send for the next transfer.

Bit 9 – PEC PEC Request (SMBus Mode only)

Value	Description
0	The transfer does not use a PEC byte.
1	The transfer uses a PEC byte.

Bit 8 – DIR Transfer Direction

Value	Description
0	Write direction.
1	Read direction.

Bits 7:0 – DATAL[7:0] Data Length

Value	Description
0	No data to send (see 38.9.3.11 Alternative Command).

Value	Description
1-255	Number of bytes to send during the transfer.

38.10.88 TWI Filter Register

Name: FLEX_TWI_FILTR
Offset: 0x644
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						THRES[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
						PADFCFG	PADFEN	FILT
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 10:8 – THRES[2:0] Digital Filter Threshold

Value	Description
0	No filtering applied on TWI inputs.
1-7	Maximum pulse width of spikes which will be suppressed by the input filter, defined in peripheral clock cycles.

Bit 2 – PADFCFG PAD Filter Config
Refer to the section “Electrical Characteristics” for filter configuration details.

Bit 1 – PADFEN PAD Filter Enable

Value	Description
0	PAD analog filter is disabled.
1	PAD analog filter is enabled. (The analog filter must be enabled if High-speed mode is enabled.)

Bit 0 – FILT RX Digital Filter
TWI digital input filtering follows a majority decision based on three samples from SDA/SCL lines at peripheral clock frequency.

Value	Description
0	No filtering applied on TWI inputs.
1	TWI input filtering is active. (Only in Standard and Fast modes)

38.10.89 TWI FIFO Mode Register

Name: FLEX_TWI_FMR
Offset: 0x650
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the TWI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
			RXFTHRES[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			TXFTHRES[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			RXRDYM[1:0]				TXRDYM[1:0]	
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bits 29:24 – RXFTHRES[5:0] Receive FIFO Threshold

Value	Description
0-8	Defines the Receive FIFO threshold value (number of data). The FLEX_TWI_FSR.RXFTH flag will be set when Receive FIFO goes from “below” threshold state to “equal to or above” threshold state.

Bits 21:16 – TXFTHRES[5:0] Transmit FIFO Threshold

Value	Description
0-8	Defines the Transmit FIFO threshold value (number of data). The FLEX_TWI_FSR.TXFTH flag will be set when Transmit FIFO goes from “above” threshold state to “equal to or below” threshold state.

Bits 5:4 – RXRDYM[1:0] Receiver Ready Mode
If FIFOs are enabled, the FLEX_TWI_SR.RXRDY flag behaves as follows.

Value	Name	Description
0	ONE_DATA	RXRDY will be at level ‘1’ when at least one unread data is in the Receive FIFO
1	TWO_DATA	RXRDY will be at level ‘1’ when at least two unread data are in the Receive FIFO
2	FOUR_DATA	RXRDY will be at level ‘1’ when at least four unread data are in the Receive FIFO

Bits 1:0 – TXRDYM[1:0] Transmitter Ready Mode
If FIFOs are enabled, the FLEX_TWI_SR.TXRDY flag behaves as follows.

Value	Name	Description
0	ONE_DATA	TXRDY will be at level ‘1’ when at least one data can be written in the Transmit FIFO
1	TWO_DATA	TXRDY will be at level ‘1’ when at least two data can be written in the Transmit FIFO
2	FOUR_DATA	TXRDY will be at level ‘1’ when at least four data can be written in the Transmit FIFO

38.10.90 TWI FIFO Level Register

Name: FLEX_TWI_FLR
Offset: 0x654
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			RXFL[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			TXFL[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bits 21:16 – RXFL[5:0] Receive FIFO Level

Value	Description
0	There is no unread data in the Receive FIFO.
1-8	Indicates the number of unread data in the Receive FIFO.

Bits 5:0 – TXFL[5:0] Transmit FIFO Level

Value	Description
0	There is no data in the Transmit FIFO.
1-8	Indicates the number of data in the Transmit FIFO.

38.10.91 TWI FIFO Status Register

Name: FLEX_TWI_FSR
Offset: 0x660
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 7 – RXFPTEF Receive FIFO Pointer Error Flag
See 38.9.6.10 FIFO Pointer Error for details.

Value	Description
0	No Receive FIFO pointer occurred.
1	Receive FIFO pointer error occurred. Receiver must be reset.

Bit 6 – TXFPTEF Transmit FIFO Pointer Error Flag
See 38.9.6.10 FIFO Pointer Error for details.

Value	Description
0	No Transmit FIFO pointer occurred.
1	Transmit FIFO pointer error occurred. Transceiver must be reset.

Bit 5 – RXFTHF Receive FIFO Threshold Flag

Value	Description
0	Number of unread data in Receive FIFO is below RXFTHRES threshold.
1	Number of unread data in Receive FIFO has reached RXFTHRES threshold since the last read of FLEX_TWI_FSR.

Bit 4 – RXFFF Receive FIFO Full Flag

Value	Description
0	Receive FIFO is not empty.
1	Receive FIFO has been filled since the last read of FLEX_TWI_FSR.

Bit 3 – RXFEF Receive FIFO Empty Flag

Value	Description
0	Receive FIFO is not empty.
1	Receive FIFO has been emptied since the last read of FLEX_TWI_FSR.

Bit 2 – TXFTHF Transmit FIFO Threshold Flag (cleared on read)

Value	Description
0	Number of data in Transmit FIFO is above TXFTHRES threshold.
1	Number of data in Transmit FIFO has reached TXFTHRES threshold since the last read of FLEX_TWI_FSR.

Bit 1 – TXFFF Transmit FIFO Full Flag (cleared on read)

Value	Description
0	Transmit FIFO is not full.
1	Transmit FIFO has been filled since the last read of FLEX_TWI_FSR.

Bit 0 – TXFEF Transmit FIFO Empty Flag (cleared on read)

Value	Description
0	Transmit FIFO is not empty.
1	Transmit FIFO has been emptied since the last read of FLEX_TWI_FSR.

38.10.92 TWI FIFO Interrupt Enable Register

Name: FLEX_TWI_FIER
Offset: 0x664
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the TWI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 7 – RXFPTEF RXFPTEF Interrupt Enable

Bit 6 – TXFPTEF TXFPTEF Interrupt Enable

Bit 5 – RXFTHF RXFTHF Interrupt Enable

Bit 4 – RXFFF RXFFF Interrupt Enable

Bit 3 – RXFEF RXFEF Interrupt Enable

Bit 2 – TXFTHF TXFTHF Interrupt Enable

Bit 1 – TXFFF TXFFF Interrupt Enable

Bit 0 – TXFEF TXFEF Interrupt Enable

38.10.93 TWI FIFO Interrupt Disable Register

Name: FLEX_TWI_FIDR
Offset: 0x668
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the TWI Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 7 – RXFPTEF RXFPTEF Interrupt Disable

Bit 6 – TXFPTEF TXFPTEF Interrupt Disable

Bit 5 – RXFTHF RXFTHF Interrupt Disable

Bit 4 – RXFFF RXFFF Interrupt Disable

Bit 3 – RXFEF RXFEF Interrupt Disable

Bit 2 – TXFTHF TXFTHF Interrupt Disable

Bit 1 – TXFFF TXFFF Interrupt Disable

Bit 0 – TXFEF TXFEF Interrupt Disable

38.10.94 TWI FIFO Interrupt Mask Register

Name: FLEX_TWI_FIMR
Offset: 0x66C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 7 – RXFPTEF RXFPTEF Interrupt Mask

Bit 6 – TXFPTEF TXFPTEF Interrupt Mask

Bit 5 – RXFTHF RXFTHF Interrupt Mask

Bit 4 – RXFFF RXFFF Interrupt Mask

Bit 3 – RXFEF RXFEF Interrupt Mask

Bit 2 – TXFTHF TXFTHF Interrupt Mask

Bit 1 – TXFFF TXFFF Interrupt Mask

Bit 0 – TXFEF TXFEF Interrupt Mask

38.10.95 TWI Write Protection Mode Register

Name: FLEX_TWI_WPMR
Offset: 0x6E4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
						WPCREN	WPITEN	WPEN
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x545749	PASSWD	Writing any other value in this field aborts the write operation of bits WPEN, WPITEN and WPCREN. Always reads as 0.

Bit 2 – WPCREN Write Protection Control Enable

Value	Description
0	Disables the write protection on control register if WPKEY corresponds to 0x545749 (“TWI” in ASCII).
1	Enables the write protection on control register if WPKEY corresponds to 0x545749 (“TWI” in ASCII).

Bit 1 – WPITEN Write Protection Interrupt Enable

Value	Description
0	Disables the write protection on interrupt registers if WPKEY corresponds to 0x545749 (“TWI” in ASCII).
1	Enables the write protection on interrupt registers if WPKEY corresponds to 0x545749 (“TWI” in ASCII).

Bit 0 – WPEN Write Protection Enable
See TWI Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII).

38.10.96 TWI Write Protection Status Register

Name: FLEX_TWI_WPSR
Offset: 0x6E8
Reset: 0x00000000
Property: Read-only

If enabled in the SFR module (default is disabled), a debugger read access does not clear the WPVS flag.

Bit	31	30	29	28	27	26	25	24
	WPVSRC[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPVSRC[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 31:8 – WPVSRC[23:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protect Violation Status

Value	Description
0	No Write Protection Violation has occurred since the last read of FLEX_TWI_WPSR.
1	A Write Protection Violation has occurred since the last read of FLEX_TWI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

39. Controller Area Network (MCAN)

39.1 Description

The Controller Area Network (MCAN) performs communication according to ISO 11898-1:2015 and to Bosch CAN-FD specification. Additional transceiver hardware is required for connection to the physical layer.

All functions concerning the handling of messages are implemented by the Rx Handler and the Tx Handler. The Rx Handler manages message acceptance filtering, the transfer of received messages from the CAN core to the Message RAM, as well as providing receive message status information. The Tx Handler is responsible for the transfer of transmit messages from the Message RAM to the CAN core, as well as providing transmit status information.

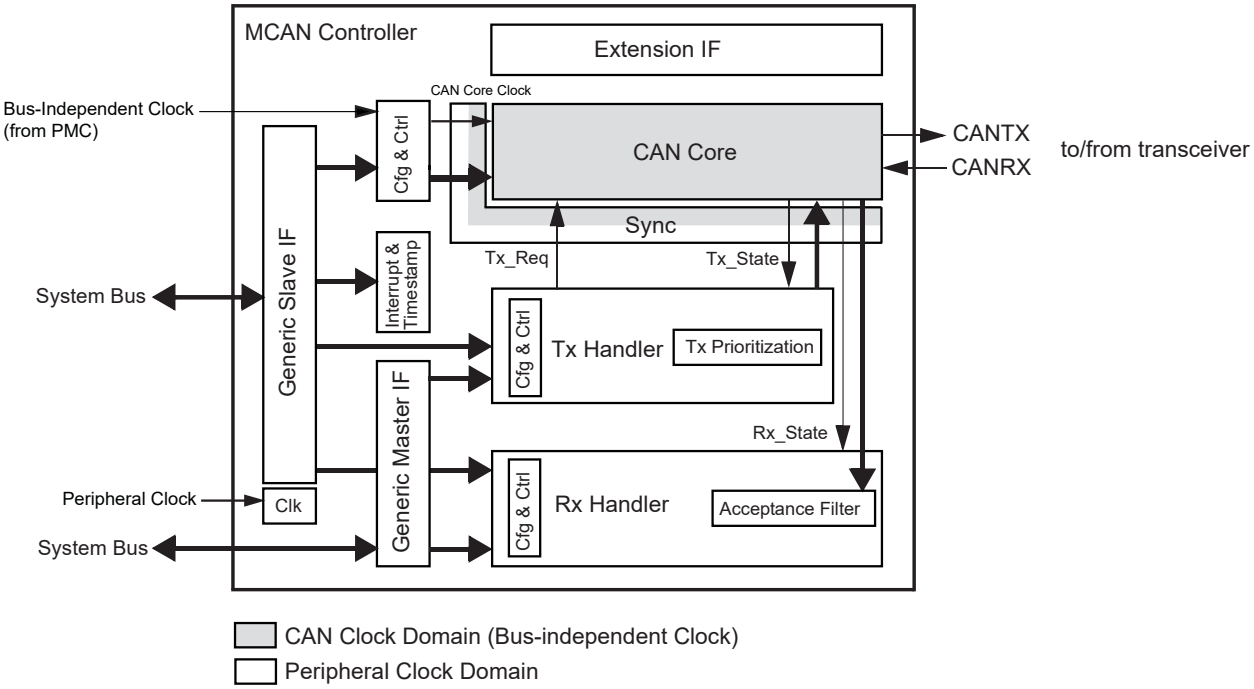
Acceptance filtering is implemented by a combination of up to 128 filter elements, where each element can be configured as a range, as a bit mask, or as a dedicated ID filter.

39.2 Embedded Characteristics

- Compliant with CAN Protocol Version 2.0 Part A, B and ISO 11898-1
- CAN-FD with up to 64 Data Bytes Supported
- CAN Error Logging
- AUTOSAR Optimized
- SAE J1939 Optimized
- Improved Acceptance Filtering
- Two Configurable Receive FIFOs
- Separate Signalling on Reception of High Priority Messages
- Up to 64 Dedicated Receive Buffers
- Up to 32 Dedicated Transmit Buffers
- Configurable Transmit FIFO
- Configurable Transmit Queue
- Configurable Transmit Event FIFO
- Direct Message RAM Access for Processor
- Multiple MCANs May Share the Same Message RAM
- Programmable Loop-back Test Mode
- Maskable Module Interrupts
- Support for Asynchronous CAN and System Bus Clocks
- Power-down Support
- Debug on CAN Support

39.3 Block Diagram

Figure 39-1. MCAN Block Diagram



Note: Refer to section “Power Management Controller (PMC)” for details about the bus-independent clock (GCLK).

39.4 Product Dependencies

39.4.1 I/O Lines

The pins used to interface to the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the CAN pins to their peripheral functions.

39.4.2 Power Management

The MCAN can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the MCAN clock.

In order to achieve a stable function of the MCAN, the system bus clock must always be faster than or equal to the CAN clock.

It is recommended to use the CAN clock at frequencies of 20, 40 or 80 MHz. To achieve these frequencies, PMC GCLK must select PLLA OR PLLB and the appropriate divider. GCLK allows the system bus and processor clock to be modified without affecting the bit rate communication.

39.4.3 Interrupt Sources

The two MCAN interrupt lines (MCAN_INT0, MCAN_INT1) are connected on internal sources of the Interrupt Controller.

Using the MCAN interrupts requires the Interrupt Controller to be programmed first.

Interrupt sources can be routed either to MCAN_INT0 or to MCAN_INT1. By default, all interrupt sources are routed to interrupt line MCAN_INT0/1. By programming MCAN_ILE.EINT0 and MCAN_ILE.EINT1, the interrupt sources can be enabled or disabled separately.

39.4.4 Address Configuration

The LSBs [bits 15:2] for each section of the CAN Message RAM are configured in the respective buffer configuration registers as detailed in Message RAM.

The MSBs [bits 31:16] of the CAN Message RAM for CAN0 and CAN1 are configured in SFR_CANx registers.

39.4.5 Timestamping

Timestamping uses the value of CV in the Timer Counter channel 0 register (TC_CV0). TC0 may use the generic clock (GCLK) from the Power Management Controller. Refer to the section “Timer Counter (TC)” for more details about clock source selection.

39.5 Functional Description

39.5.1 Operating Modes

39.5.1.1 Software Initialization

Software initialization is started by setting bit MCAN_CCCR.INIT, either by software or by a hardware reset, when an uncorrected bit error was detected in the Message RAM, or by going Bus_Off. While MCAN_CCCR.INIT is set, message transfer from and to the CAN bus is stopped and the status of the CAN bus output CANTX is recessive (HIGH). The counters of the Error Management Logic EML are unchanged. Setting MCAN_CCCR.INIT does not change any configuration register. Resetting MCAN_CCCR.INIT finishes the software initialization. Afterwards the Bit Stream Processor BSP synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (≡ Bus_Idle) before it can take part in bus activities and start the message transfer.

Access to the MCAN configuration registers is only enabled when both bits MCAN_CCCR.INIT and MCAN_CCCR.CCE are set (protected write).

MCAN_CCCR.CCE can only be configured when MCAN_CCCR.INIT = ‘1’. MCAN_CCCR.CCE is automatically cleared when MCAN_CCCR.INIT = ‘0’.

The following registers are cleared when MCAN_CCCR.CCE = ‘1’:

- High Priority Message Status (MCAN_HPMS)
- Receive FIFO 0 Status (MCAN_RXF0S)
- Receive FIFO 1 Status (MCAN_RXF1S)
- Transmit FIFO/Queue Status (MCAN_TXFQS)
- Transmit Buffer Request Pending (MCAN_TXBRP)
- Transmit Buffer Transmission Occurred (MCAN_TXBTO)
- Transmit Buffer Cancellation Finished (MCAN_TXBCF)
- Transmit Event FIFO Status (MCAN_TXEFS)

The Timeout Counter value MCAN_TOCV.TOC is loaded with the value configured by MCAN_TOCC.TOP when MCAN_CCCR.CCE = ‘1’.

In addition, the state machines of the Tx Handler and Rx Handler are held in idle state while MCAN_CCCR.CCE = ‘1’.

The following registers are only writeable while MCAN_CCCR.CCE = ‘0’

- Transmit Buffer Add Request (MCAN_TXBAR)
- Transmit Buffer Cancellation Request (MCAN_TXBCR)

MCAN_CCCR.TEST and MCAN_CCCR.MON can only be set when MCAN_CCCR.INIT = ‘1’ and MCAN_CCCR.CCE = ‘1’. Both bits may be cleared at any time. MCAN_CCCR.DAR can only be configured when MCAN_CCCR.INIT = ‘1’ and MCAN_CCCR.CCE = ‘1’.

39.5.1.2 Normal Operation

Once the MCAN is initialized and MCAN_CCCR.INIT is cleared, the MCAN synchronizes itself to the CAN bus and is ready for communication.

After passing the acceptance filtering, received messages including Message ID and DLC are stored into a dedicated Rx Buffer or into Rx FIFO 0 or Rx FIFO 1.

For messages to be transmitted, dedicated Tx Buffers and/or a Tx FIFO or a Tx Queue can be initialized or updated. Automated transmission on reception of remote frames is not implemented.

39.5.1.3 CAN FD Operation

There are two variants in the CAN FD frame format, first the CAN FD frame without bit rate switching where the data field of a CAN frame may be longer than 8 bytes. The second variant is the CAN FD frame where control field, data field, and CRC field of a CAN frame are transmitted with a higher bit rate than the beginning and the end of the frame.

The previously reserved bit in CAN frames with 11-bit identifiers and the first previously reserved bit in CAN frames with 29-bit identifiers will now be decoded as FDF bit. FDF = recessive signifies a CAN FD frame, FDF = dominant signifies a Classic CAN frame. In a CAN FD frame, the two bits following FDF, res and BRS, decide whether the bit rate inside of this CAN FD frame is switched. A CAN FD bit rate switch is signified by res = dominant and BRS = recessive. The coding of res = recessive is reserved for future expansion of the protocol. In case the MCAN receives a frame with FDF = recessive and res = recessive, it will signal a Protocol Exception Event by setting bit MCAN_PSR.PXE. When Protocol Exception Handling is enabled (MCAN_CCCR.PXHD = 0), this causes the operation state to change from Receiver (MCAN_PSR.ACT = 2) to Integrating (MCAN_PSR.ACT = 00) at the next sample point. In case Protocol Exception Handling is disabled (MCAN_CCCR.PXHD = 1), the MCAN will treat a recessive res bit as an form error and will respond with an error frame.

CAN FD operation is enabled by programming CCCR.FDOE. In case CCCR.FDOE = ‘1’, transmission and reception of CAN FD frames is enabled. Transmission and reception of Classic CAN frames is always possible. Whether a CAN FD frame or a Classic CAN frame is transmitted can be configured via bit FDF in the respective Tx Buffer element. With CCCR.FDOE = ‘0’, received frames are interpreted as Classic CAN frames, which leads to the transmission of an error frame when receiving a CAN FD frame. When CAN FD operation is disabled, no CAN FD frames are transmitted even if bit FDF of a Tx Buffer element is set. CCCR.FDOE and CCCR.BRSE can only be changed while CCCR.INIT and CCCR.CCE are both set.

With MCAN_CCCR.FDOE = 0, the setting of bits FDF and BRS is ignored and frames are transmitted in Classic CAN format. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 0, only bit FDF of a Tx Buffer element is evaluated. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 1, transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer elements with bits FDF and BRS set are transmitted in CAN FD format with bit rate switching.

A mode change during CAN operation is only recommended under the following conditions:

- The failure rate in the CAN FD data phase is significant higher than in the CAN FD arbitration phase. In this case disable the CAN FD bit rate switching option for transmissions.
- During system startup all nodes are transmitting according to ISO11898-1 until it is verified that they are able to communicate in CAN FD format. If this is true, all nodes switch to CAN FD operation.
- Wake-up messages in CAN Partial Networking have to be transmitted in Classic CAN format.
- End-of-line programming in case not all nodes are CAN FD-capable. Non-CAN FD nodes are held in Silent mode until programming has completed. Then all nodes revert to Classic CAN communication.

In the CAN FD format, the coding of the DLC differs from the standard CAN format. The DLC codes 0 to 8 have the same coding as in standard CAN, the codes 9 to 15, which in standard CAN all code a data field of 8 bytes, are coded according to the table below.

Table 39-1. Coding of DLC in CAN FD

DLC	9	10	11	12	13	14	15
Number of Data Bytes	12	16	20	24	32	48	64

In CAN FD frames, the bit timing will be switched inside the frame, after the BRS (Bit Rate Switch) bit, if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by the Nominal Bit Timing and Prescaler register (MCAN_NBTP). In the following CAN FD data phase, the data phase CAN bit timing is used as defined by the Data Bit Timing and Prescaler register (MCAN_DBTP). The bit timing reverts back from the data phase timing at the CRC delimiter or when an error is detected, whichever occurs first.

The maximum configurable bit rate in the CAN FD data phase depends on the CAN core clock frequency. Example: with a CAN clock frequency of 20 MHz and the shortest configurable bit time of 4 t_q, the bit rate in the data phase is 5 Mbit/s.

In both data frame formats, CAN FD and CAN FD with bit rate switching, the value of the bit ESI (Error Status Indicator) is determined by the transmitter's error state at the start of the transmission. If the transmitter is error passive, ESI is transmitted recessive, else it is transmitted dominant.

39.5.1.4 Transmitter Delay Compensation

During the data phase of a CAN FD transmission only one node is transmitting, all others are receivers. The length of the bus line has no impact. When transmitting via pin CANTX the protocol controller receives the transmitted data from its local CAN transceiver via pin CANRX. The received data is delayed by the transmitter delay. In case this delay is greater than TSEG1 (time segment before sample point), a bit error is detected. In order to enable a data phase bit time that is even shorter than the transmitter delay, the delay compensation is introduced. Without delay compensation, the bit rate in the data phase of a CAN FD frame is limited by the delay.

39.5.1.4.1 Description

The MCAN protocol unit has implemented a delay compensation mechanism to compensate the delay, thereby enabling transmission with higher bit rates during the CAN FD data phase independent of the delay of a specific CAN transceiver.

To check for bit errors during the data phase, the delayed transmit data is compared against the received data at the secondary sample point. If a bit error is detected, the transmitter will react to this bit error at the next following regular sample point. During arbitration phase the delay compensation is always disabled.

The transmitter delay compensation enables configurations where the data bit time is shorter than the transmitter delay, it is described in detail in the new ISO11898-1. It is enabled by setting bit MCAN_DBTP.TDC.

The received bit is compared against the transmitted bit at the SSP. The SSP position is defined as the sum of the measured delay from the MCAN's transmit output CANTX through the transceiver to the receive input CANRX plus the transmitter delay compensation offset as configured by MCAN_TDCR.TDCO. The transmitter delay compensation offset is used to adjust the position of the SSP inside the received bit (e.g. half of the bit time in the data phase). The position of the secondary sample point is rounded down to the next integer number of CAN core clock periods.

MCAN_PSR.TDCV shows the actual transmitter delay compensation value. MCAN_PSR.TDCV is cleared when MCAN_CCCR.INIT is set and is updated at each transmission of an FD frame while MCAN_DBTP.TDC is set.

The following boundary conditions have to be considered for the delay compensation implemented in the MCAN:

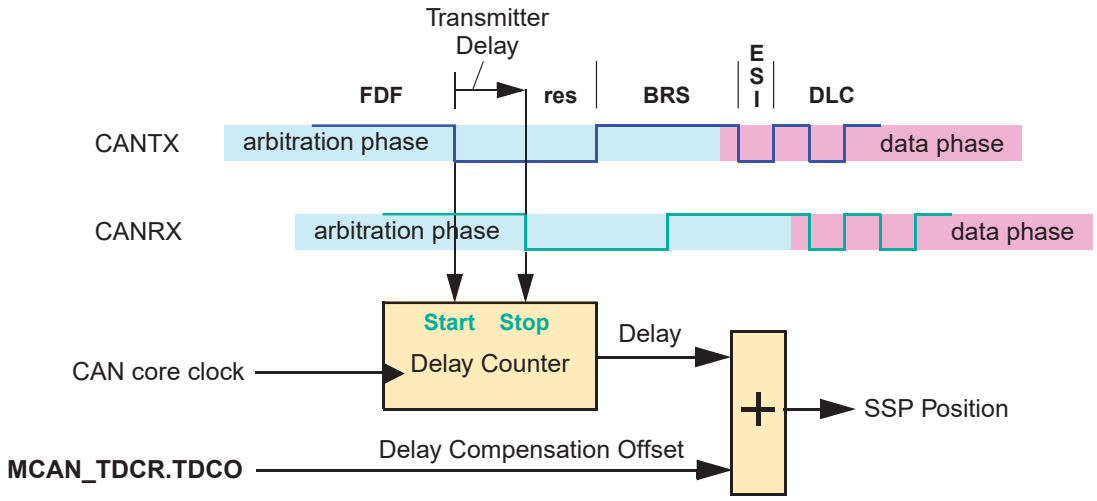
- The sum of the measured delay from CANTX to CANRX and the configured delay compensation offset MCAN_TDCR.TDCO has to be less than 6 bit times in the data phase.
- The sum of the measured delay from CANTX to CANRX and the configured delay compensation offset MCAN_TDCR.TDCO has to be less or equal 127 CAN core clock periods. In case this sum exceeds 127 CAN core clock periods, the maximum value of 127 CAN core clock periods is used for delay compensation.
- The data phase ends at the sample point of the CRC delimiter, that stops checking of receive bits at the SSPs.

39.5.1.4.2 Transmitter Delay Measurement

If transmitter delay compensation is enabled by programming MCAN_DBTP.TDC = '1', the measurement is started within each transmitted CAN FD frame at the falling edge of bit FDF to bit res. The measurement is stopped when this edge is seen at the receive input CANRX of the transmitter.

The resolution of this measurement is one mtq.

Figure 39-2. Transmitter Delay Measurement



To avoid that a dominant glitch inside the received FDF bit ends the delay compensation measurement before the falling edge of the received res bit, resulting in a to early SSP position, the use of a transmitter delay compensation filter window can be enabled by programming MCAN_TDCR.TDCF.

This defines a minimum value for the SSP position. Dominant edges on CANRX, that would result in an earlier SSP position are ignored for transmitter delay measurement. The measurement is stopped when the SSP position is at least MCAN_TDCR.TDCF AND CANRX is low.

39.5.1.5 Restricted Operation Mode

In Restricted Operation mode, the node is able to receive data and remote frames and to give acknowledge to valid frames, but it does not send data frames, remote frames, active error frames, or overload frames. In case of an error condition or overload condition, it does not send dominant bits, instead it waits for the occurrence of bus idle condition to resynchronize itself to the CAN communication. The error counters are not incremented. The processor can set the MCAN into Restricted Operation mode by setting bit MCAN_CCCR.ASM. The bit can only be set by the processor when both MCAN_CCCR.CCE and MCAN_CCCR.INIT are set to '1'. The bit can be reset by the processor at any time.

Restricted Operation mode is automatically entered when the Tx Handler was not able to read data from the Message RAM in time. To leave Restricted Operation mode, the processor has to reset MCAN_CCCR.ASM.

The Restricted Operation mode can be used in applications that adapt themselves to different CAN bit rates. In this case the application tests different bit rates and leaves the Restricted Operation mode after it has received a valid frame.

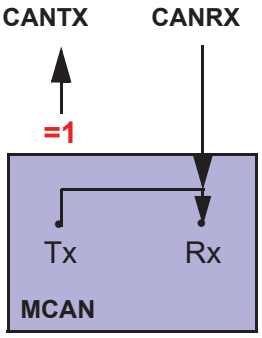
Note: The Restricted Operation Mode must not be combined with the Loop Back mode (internal or external).

39.5.1.6 Bus Monitoring Mode

The MCAN is set in Bus Monitoring mode by setting MCAN_CCCR.MON. In Bus Monitoring mode (see ISO11898-1, 10.12 Bus monitoring), the MCAN is able to receive valid data frames and valid remote frames, but cannot start a transmission. In this mode, it sends only recessive bits on the CAN bus. If the MCAN is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the MCAN monitors this dominant bit, although the CAN bus may remain in recessive state. In Bus Monitoring mode, the Tx Buffer Request Pending register (MCAN_TXBRP) is held in reset state.

The Bus Monitoring mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. The figure below shows the connection of signals CANTX and CANRX to the MCAN in Bus Monitoring mode.

Figure 39-3. Pin Control in Bus Monitoring Mode



Bus Monitoring Mode

39.5.1.7 Disabled Automatic Retransmission

According to the CAN Specification (see ISO11898-1, 6.3.3 Recovery Management), the MCAN provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. By default automatic retransmission is enabled. To support time-triggered communication as described in ISO 11898-1, chapter 9.2, the automatic retransmission may be disabled via MCAN_CCCR.DAR.

39.5.1.7.1 Frame Transmission in DAR Mode

In DAR mode, all transmissions are automatically cancelled after they start on the CAN bus. A Tx Buffer's Tx Request Pending bit TXBRP.TRPx is reset after successful transmission, when a transmission has not yet been started at the point of cancellation, has been aborted due to lost arbitration, or when an error occurred during frame transmission.

- Successful transmission:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx set

  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx not set
- Successful transmission in spite of cancellation:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx set

  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx set
- Arbitration lost or frame transmission disturbed:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx not set

  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx set

In case of a successful frame transmission, and if storage of Tx events is enabled, a Tx Event FIFO element is written with Event Type ET = “10” (transmission in spite of cancellation).

39.5.1.8 Power-down (Sleep Mode)

The MCAN can be set into Power-down mode via bit MCAN_CCCR.CSR.

When all pending transmission requests have completed, the MCAN waits until bus idle state is detected. Then the MCAN sets MCAN_CCCR.INIT to prevent any further CAN transfers. Now the MCAN acknowledges that it is ready for power down by setting to one the bit MCAN_CCCR.CSA. In this state, before the clocks are switched off, further register accesses can be made. A write access to MCAN_CCCR.INIT will have no effect. Now the bus clock (peripheral clock) and the CAN core clock may be switched off.

To leave Power-down mode, the application has to turn on the MCAN clocks before clearing CC Control Register flag MCAN_CCCR.CSR. The MCAN will acknowledge this by clearing MCAN_CCCR.CSA. The application can then restart CAN communication by clearing the bit CCCR.INIT.

39.5.1.9 Test Modes

To enable write access to the MCAN Test register (MCAN_TEST) (see Section 7.6), bit MCAN_CCCR.TEST must be set. This allows the configuration of the test modes and test functions.

Four output functions are available for the CAN transmit pin CANTX by programming MCAN_TEST.TX. Additionally to its default function – the serial data output – it can drive the CAN Sample Point signal to monitor the MCAN's bit

timing and it can drive constant dominant or recessive values. The actual value at pin CANRX can be read from MCAN_TEST.RX. Both functions can be used to check the CAN bus' physical layer.

Due to the synchronization mechanism between CAN clock and system bus clock domain, there may be a delay of several system bus clock periods between writing to MCAN_TEST.TX until the new configuration is visible at output pin CANTX. This applies also when reading input pin CANRX via MCAN_TEST.RX.

Note: Test modes should be used for production tests or self-test only. The software control for pin CANTX interferes with all CAN protocol functions. It is not recommended to use test modes for application.

39.5.1.9.1 External Loop Back Mode

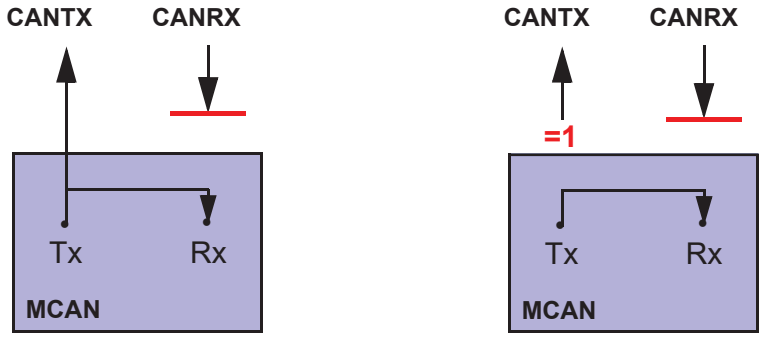
The MCAN can be set in External Loop Back mode by setting the bit MCAN_TEST.LBCK. In Loop Back mode, the MCAN treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into an Rx Buffer or an Rx FIFO. The figure below shows the connection of signals CANTX and CANRX to the MCAN in External Loop Back mode.

This mode is provided for hardware self-test. To be independent from external stimulation, the MCAN ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back mode. In this mode, the MCAN performs an internal feedback from its Tx output to its Rx input. The actual value of the CANRX input pin is disregarded by the MCAN. The transmitted messages can be monitored at the CANTX pin.

39.5.1.9.2 Internal Loop Back Mode

Internal Loop Back mode is entered by setting bits MCAN_TEST.LBCK and MCAN_CCCR.MON. This mode can be used for a "Hot Selftest", meaning the MCAN can be tested without affecting a running CAN system connected to the pins CANTX and CANRX. In this mode, pin CANRX is disconnected from the MCAN, and pin CANTX is held recessive. The figure below shows the connection of CANTX and CANRX to the MCAN when Internal Loop Back mode is enabled.

Figure 39-4. Pin Control in Loop Back Modes



39.5.2 Timestamp Generation

For timestamp generation the MCAN supplies a 16-bit wrap-around counter. A prescaler TSCC.TCP can be configured to clock the counter in multiples of CAN bit times (1…16). The counter is readable via MCAN_TSCV.TSC. A write access to the Timestamp Counter Value register (MCAN_TSCV) resets the counter to zero. When the timestamp counter wraps around, interrupt flag MCAN_IR.TSW is set.

On start of frame reception / transmission the counter value is captured and stored into the timestamp section of an Rx Buffer / Rx FIFO (RXTS[15:0]) or Tx Event FIFO (TXTS[15:0]) element.

By programming bit MCAN_TSCC.TSS an external 16-bit timestamp can be used. See Timestamping for more details.

39.5.3 Timeout Counter

To signal timeout conditions for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO, the MCAN supplies a 16-bit Timeout Counter. It operates as down-counter and uses the same prescaler controlled by TSCC.TCP as the Timestamp Counter. The Timeout Counter is configured via the Timeout Counter Configuration register (MCAN_TOCC). The actual counter value can be read from MCAN_TOCV.TOC. The Timeout Counter can only be started while MCAN_CCCR.INIT = '0'. It is stopped when MCAN_CCCR.INIT = '1', e.g. when the MCAN enters Bus_Off state.

The operating mode is selected by MCAN_TOCC.TOS. When operating in Continuous mode, the counter starts when MCAN_CCCR.INIT is reset. A write to MCAN_TOCV presets the counter to the value configured by MCAN_TOCC.TOP and continues down-counting.

When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by MCAN_TOCC.TOP. Down-counting is started when the first FIFO element is stored. Writing to MCAN_TOCV has no effect.

When the counter reaches zero, interrupt flag MCAN_IR.TOO is set. In Continuous mode, the counter is immediately restarted at MCAN_TOCC.TOP.

Note: The clock signal for the Timeout Counter is derived from the CAN Core's sample point signal. Therefore the point in time where the Timeout Counter is decremented may vary due to the synchronization / re-synchronization mechanism of the CAN Core. If the bit rate switch feature in CAN FD is used, the timeout counter is clocked differently in arbitration and data field.

39.5.4 Rx Handling

The Rx Handler controls the acceptance filtering, the transfer of received messages to the Rx Buffers or to one of the two Rx FIFOs, as well as the Rx FIFO's Put and Get Indices.

39.5.4.1 Acceptance Filtering

The MCAN offers the possibility to configure two sets of acceptance filters, one for standard identifiers and one for extended identifiers. These filters can be assigned to an Rx Buffer or to Rx FIFO 0,1. For acceptance filtering each list of filters is executed from element #0 until the first matching element. Acceptance filtering stops at the first matching element. The following filter elements are not evaluated for this message.

The main features are:

- Each filter element can be configured as
  - range filter (from - to)
  - filter for one or two dedicated IDs
  - classic bit mask filter
- Each filter element is configurable for acceptance or rejection filtering
- Each filter element can be enabled / disabled individually
- Filters are checked sequentially, execution stops with the first matching filter element

Related configuration registers are:

- Global Filter Configuration (MCAN_GFC)
- Standard ID Filter Configuration (MCAN_SIDFC)
- Extended ID Filter Configuration (MCAN_XIDFC)
- Extended ID and Mask (MCAN_XIDAM)

Depending on the configuration of the filter element (SFEC/EFEC) a match triggers one of the following actions:

- Store received frame in FIFO 0 or FIFO 1
- Store received frame in Rx Buffer
- Store received frame in Rx Buffer and generate pulse at filter event pin
- Reject received frame
- Set High Priority Message interrupt flag (MCAN_IR.HPM)
- Set High Priority Message interrupt flag (MCAN_IR.HPM) and store received frame in FIFO 0 or FIFO 1

Acceptance filtering is started after the complete identifier has been received. After acceptance filtering has completed, and if a matching Rx Buffer or Rx FIFO has been found, the Message Handler starts writing the received message data in portions of 32 bit to the matching Rx Buffer or Rx FIFO. If the CAN protocol controller has detected an error condition (e.g. CRC error), this message is discarded with the following impact on the effected Rx Buffer or Rx FIFO:

- Rx Buffer
  New Data flag of matching Rx Buffer is not set, but Rx Buffer (partly) overwritten with received data. For error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC.

- Rx FIFO
  Put index of matching Rx FIFO is not updated, but related Rx FIFO element (partly) overwritten with received data. For error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC. In case the matching Rx FIFO is operated in Overwrite mode, the boundary conditions described in Rx FIFO Overwrite Mode have to be considered.
  Note: When an accepted message is written to one of the two Rx FIFOs, or into an Rx Buffer, the unmodified received identifier is stored independent of the filter(s) used. The result of the acceptance filter process is strongly depending on the sequence of configured filter elements.

39.5.4.1.1 Range Filter

The filter matches for all received frames with Message IDs in the range defined by SF1ID/SF2ID resp. EF1ID/EF2ID.

There are two possibilities when range filtering is used together with extended frames:

- EFT = “00”: The Message ID of received frames is ANDed with MCAN_XIDAM before the range filter is applied.
- EFT = “11”: MCAN_XIDAM is not used for range filtering.

39.5.4.1.2 Filter for Specific IDs

A filter element can be configured to filter for one or two specific Message IDs. To filter for one specific Message ID, the filter element has to be configured with SF1ID = SF2ID resp. EF1ID = EF2ID.

39.5.4.1.3 Classic Bit Mask Filter

Classic bit mask filtering is intended to filter groups of Message IDs by masking single bits of a received Message ID. With classic bit mask filtering SF1ID/EF1ID is used as Message ID filter, while SF2ID/EF2ID is used as filter mask.

A zero bit at the filter mask will mask out the corresponding bit position of the configured ID filter, e.g. the value of the received Message ID at that bit position is not relevant for acceptance filtering. Only those bits of the received Message ID where the corresponding mask bits are one are relevant for acceptance filtering.

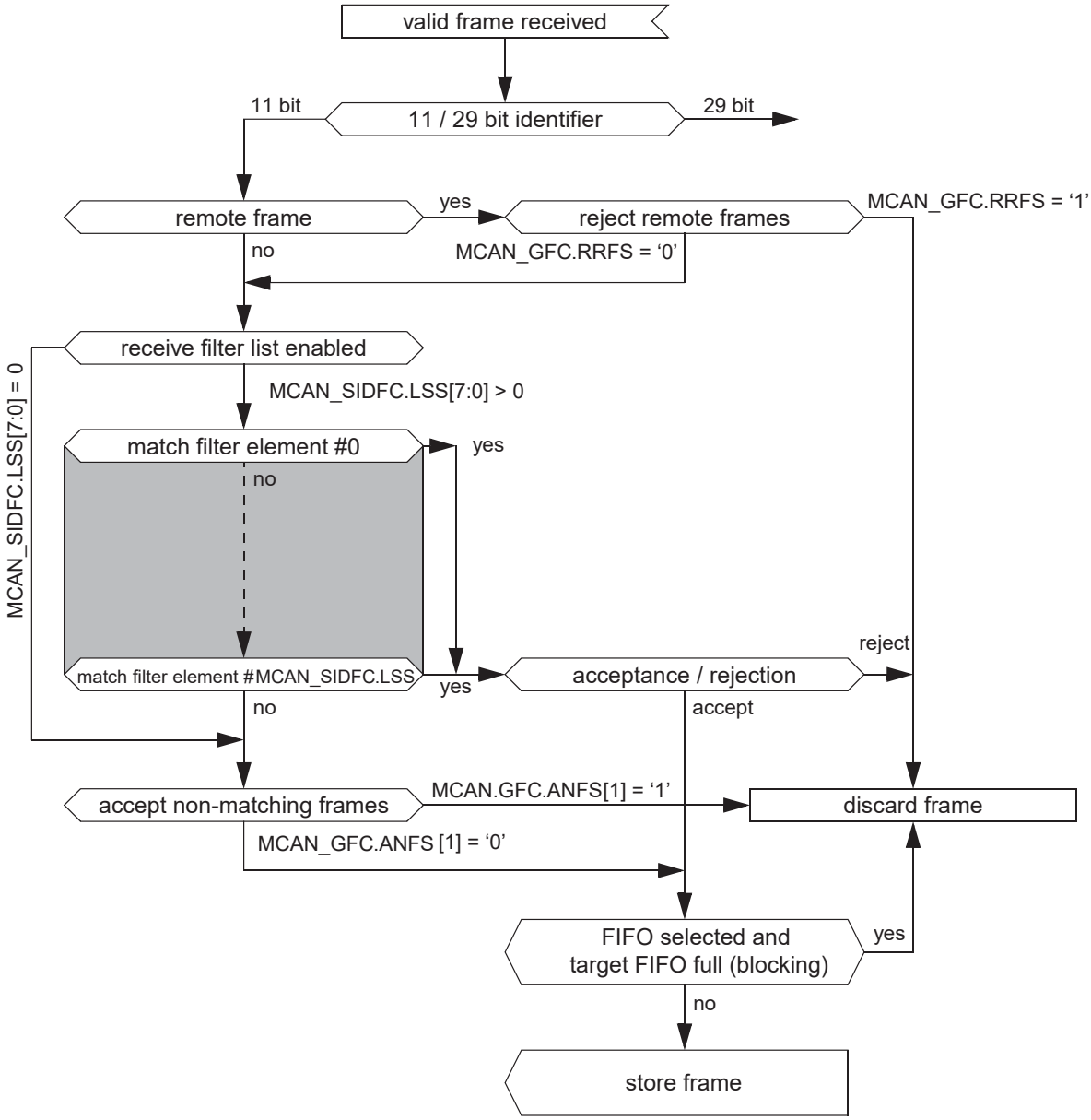
In case all mask bits are one, a match occurs only when the received Message ID and the Message ID filter are identical. If all mask bits are zero, all Message IDs match.

39.5.4.1.4 Standard Message ID Filtering

The figure below shows the flow for standard Message ID (11-bit Identifier) filtering. The Standard Message ID Filter element is described in 39.5.7.5 Standard Message ID Filter Element.

Controlled by MCAN_GFC and MCAN_SIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

Figure 39-5. Standard Message ID Filter Path



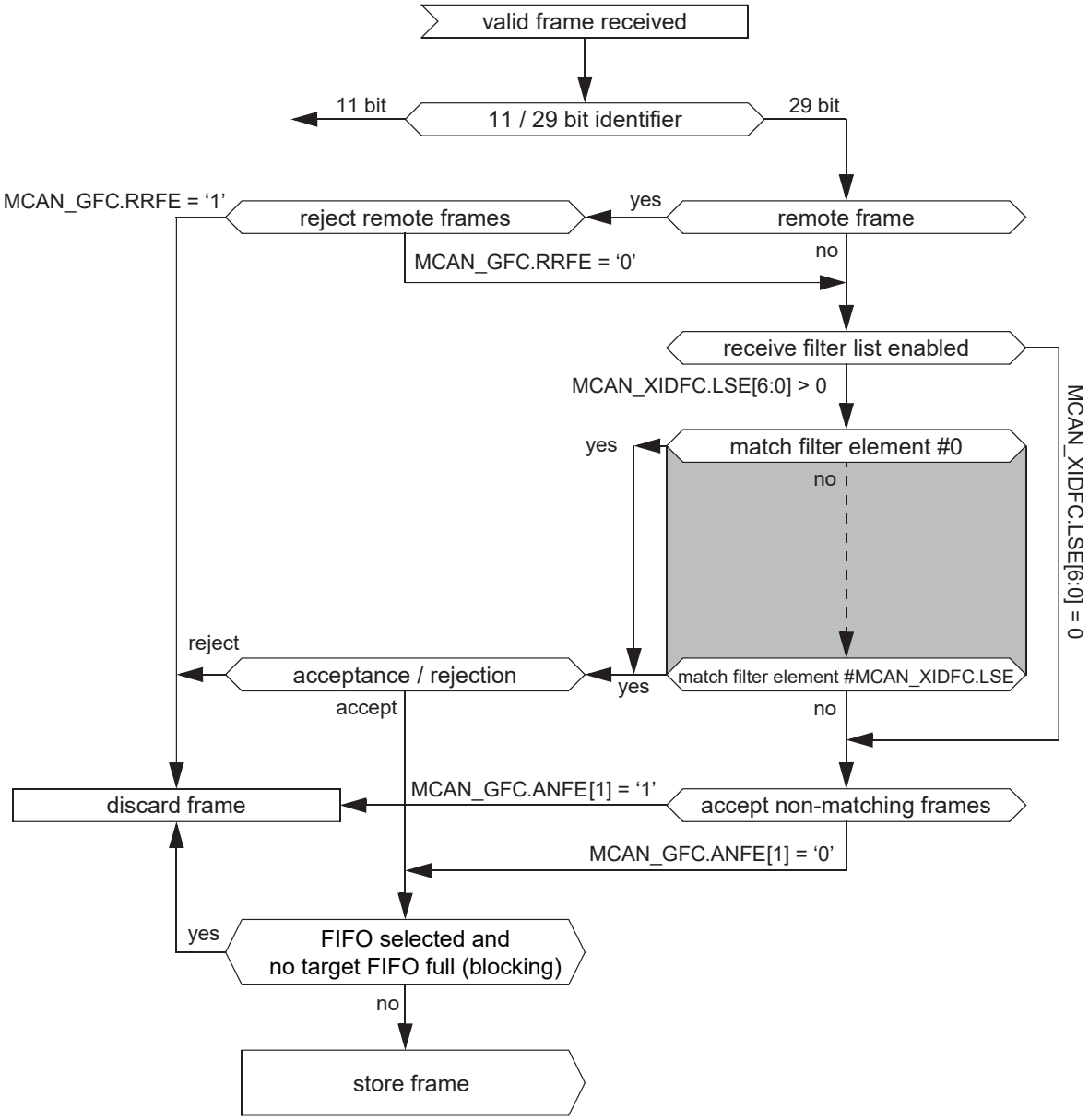
Extended Message ID Filtering

The figure below shows the flow for extended Message ID (29-bit Identifier) filtering. The Extended Message ID Filter element is described in 39.5.7.6 Extended Message ID Filter Element.

Controlled by MCAN_GFC and MCAN_XIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

MCAN_XIDAM is ANDed with the received identifier before the filter list is executed.

Figure 39-6. Extended Message ID Filter Path



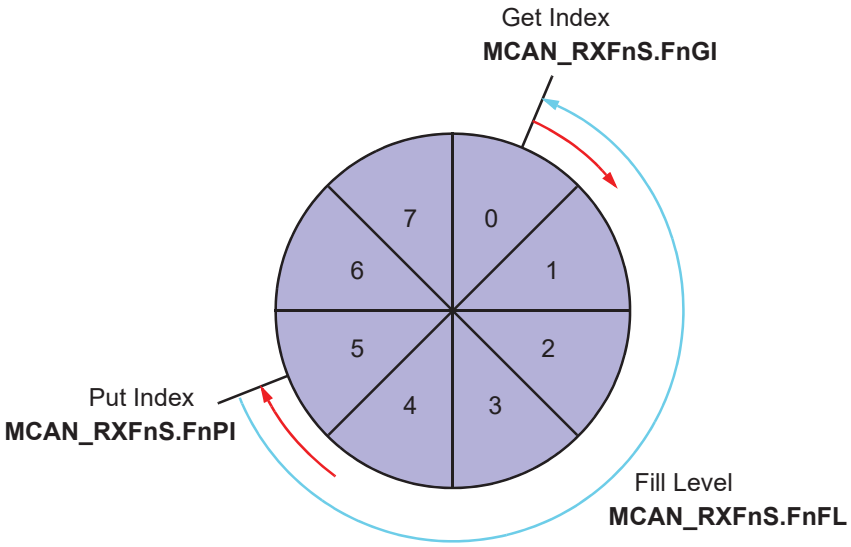
39.5.4.2 Rx FIFOs

Rx FIFO 0 and Rx FIFO 1 can be configured to hold up to 64 elements each. Configuration of the two Rx FIFOs is done via the Rx FIFO 0 Configuration register (MCAN_RXF0C) and the Rx FIFO 1 Configuration register (MCAN_RXF1C).

Received messages that passed acceptance filtering are transferred to the Rx FIFO as configured by the matching filter element. For a description of the filter mechanisms available for Rx FIFO 0 and Rx FIFO 1, see Acceptance Filtering. The Rx FIFO element is described in Rx Buffer and FIFO Element.

To avoid an Rx FIFO overflow, the Rx FIFO watermark can be used. When the Rx FIFO fill level reaches the Rx FIFO watermark configured by MCAN_RXFnC.FnWM, interrupt flag MCAN_IR.RFnW is set. When the Rx FIFO Put Index reaches the Rx FIFO Get Index, an Rx FIFO Full condition is signalled by MCAN_RXFnS.FnF. In addition, the interrupt flag MCAN_IR.RFnF is set.

Figure 39-7. Rx FIFO Status



When reading from an Rx FIFO, Rx FIFO Get Index MCAN_RXFnS.FnGI × FIFO Element Size has to be added to the corresponding Rx FIFO start address MCAN_RXFnC.FnSA.

Table 39-2. Rx Buffer / FIFO Element Size

MCAN_RXESC.RBDS[2:0] MCAN_RXESC.FnDS[2:0]	Data Field [bytes]	FIFO Element Size [RAM words]
0	8	4
1	12	5
2	16	6
3	20	7
4	24	8
5	32	10
6	48	14
7	64	18

39.5.4.2.1 Rx FIFO Blocking Mode

The Rx FIFO Blocking mode is configured by MCAN_RXFnC.FnOM = '0'. This is the default operating mode for the Rx FIFOs.

When an Rx FIFO full condition is reached (MCAN_RXFnS.FnPI = MCAN_RXFnS.FnGI), no further messages are written to the corresponding Rx FIFO until at least one message has been read out and the Rx FIFO Get Index has been incremented. An Rx FIFO full condition is signalled by MCAN_RXFnS.FnF = '1'. In addition, the interrupt flag MCAN_IR.RFnF is set.

In case a message is received while the corresponding Rx FIFO is full, this message is discarded and the message lost condition is signalled by MCAN_RXFnS.RFnL = '1'. In addition, the interrupt flag MCAN_IR.RFnL is set.

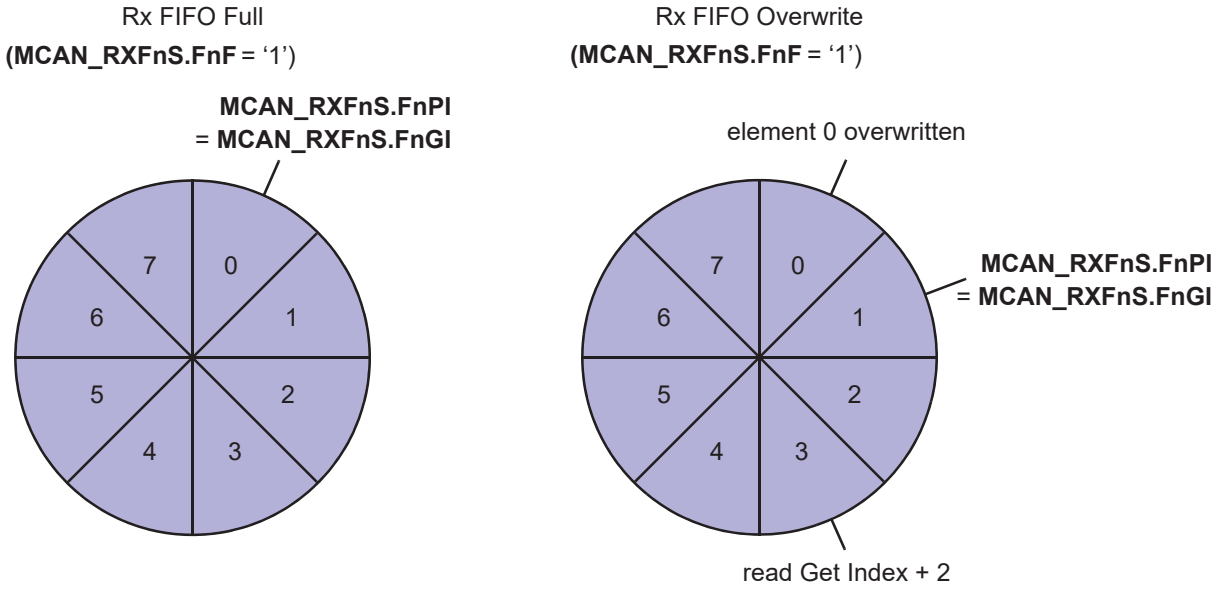
39.5.4.2.2 Rx FIFO Overwrite Mode

The Rx FIFO Overwrite mode is configured by MCAN_RXFnC.FnOM = '1'.

When an Rx FIFO full condition (MCAN_RXFnS.FnPI = MCAN_RXFnS.FnGI) is signalled by MCAN_RXFnS.FnF = '1', the next message accepted for the FIFO will overwrite the oldest FIFO message. Put and get index are both incremented by one.

When an Rx FIFO is operated in Overwrite mode and an Rx FIFO full condition is signalled, reading of the Rx FIFO elements should start at least at get index + 1. The reason for that is, that it might happen, that a received message is written to the Message RAM (put index) while the processor is reading from the Message RAM (get index). In this case inconsistent data may be read from the respective Rx FIFO element. Adding an offset to the get index when reading from the Rx FIFO avoids this problem. The offset depends on how fast the processor accesses the Rx FIFO. The figure below shows an offset of two with respect to the get index when reading the Rx FIFO. In this case the two messages stored in element 1 and 2 are lost.

Figure 39-8. Rx FIFO Overflow Handling



After reading from the Rx FIFO, the number of the last element read has to be written to the Rx FIFO Acknowledge Index MCAN_RXFnA.FnA. This increments the get index to that element number. In case the put index has not been incremented to this Rx FIFO element, the Rx FIFO full condition is reset (MCAN_RXFnS.FnF = '0').

39.5.4.3 Dedicated Rx Buffers

The MCAN supports up to 64 dedicated Rx Buffers. The start address of the dedicated Rx Buffer section is configured via MCAN_RXBC.RBSA.

For each Rx Buffer, a Standard or Extended Message ID Filter Element with SFEC / EFEC = 7 and SFID2 / EFID2[10:9] = 0 has to be configured.

After a received message has been accepted by a filter element, the message is stored into the Rx Buffer in the Message RAM referenced by the filter element. The format is the same as for an Rx FIFO element. In addition, the flag MCAN_IR.DRX (Message stored in dedicated Rx Buffer) in MCAN_IR is set.

Table 39-3. Example Filter Configuration for Rx Buffers

Filter Element	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID message 1	0	0
1	ID message 2	0	1
2	ID message 3	0	2

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in the New Data 1 register (MCAN_NDAT1) and New Data 2 register (MCAN_NDAT2) is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the processor by writing a '1' to the respective bit position.

While an Rx Buffer's New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. Following Message ID Filter Elements may cause the received

message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

39.5.4.3.1 Rx Buffer Handling

- Reset interrupt flag IR.DRX
- Read New Data registers
- Read messages from Message RAM
- Reset New Data flags of processed messages

39.5.4.4 Debug on CAN Support

Debug messages are stored into Rx Buffers. For debug handling three consecutive Rx buffers (e.g. #61, #62, #63) have to be used for storage of debug messages A, B, and C. The format is the same as for an Rx Buffer or an Rx FIFO element (see Rx Buffer and FIFO Element).

Advantage: Fixed start address for the DMA transfers (relative to MCAN_RXBC.RBSA), no additional configuration required.

For filtering of debug messages Standard / Extended Filter Elements with SFEC / EFEC = '111' have to be set up. Messages matching these filter elements are stored into the Rx Buffers addressed by SFID2 / EFID2[5:0].

After message C has been stored, the DMA request output m_can_dma_req is activated and the three messages can be read from the Message RAM under DMA control. The RAM words holding the debug messages will not be changed by the MCAN while m_can_dma_req is activated. The behavior is similar to that of an Rx Buffer with its New Data flag set.

After the DMA has completed, the MCAN is prepared to receive the next set of debug messages.

39.5.4.4.1 Filtering for Debug Messages

Filtering for debug messages is done by configuring one Standard / Extended Message ID Filter Element for each of the three debug messages. To enable a filter element to filter for debug messages SFEC / EFEC has to be programmed to "111". In this case fields SFID1 / SFID2 and EFID1 / EFID2 have a different meaning. While SFID2 / EFID2[10:9] controls the debug message handling state machine, SFID2 / EFID2[5:0] controls the location for storage of a received debug message.

When a debug message is stored, neither the respective New Data flag nor MCAN_IR.DRX are set. The reception of debug messages can be monitored via RXF1S.DMS.

Table 39-4. Example Filter Configuration for Debug Messages

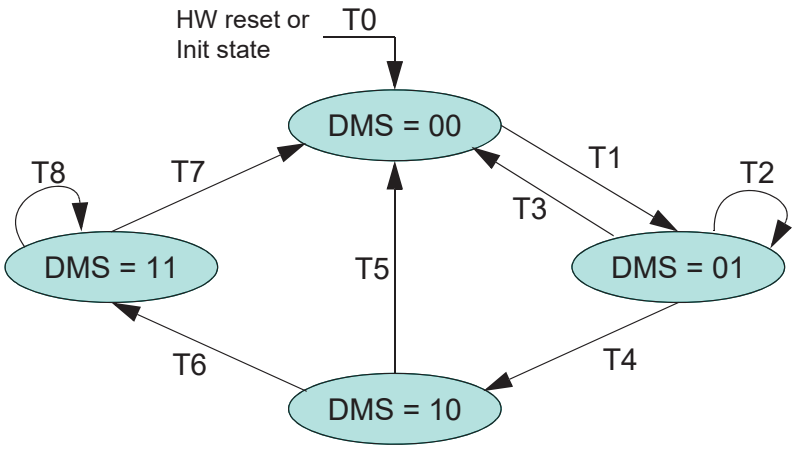
Filter Element	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID debug message A	1	11 1101
1	ID debug message B	2	11 1110
2	ID debug message C	3	11 1111

39.5.4.4.2 Debug Message Handling

The debug message handling state machine ensures that debug messages are stored to three consecutive Rx Buffers in the correct order. If some messages are missing, the process is restarted. The DMA request is activated only when all three debug messages A, B, C have been received in the correct order.

The status of the debug message handling state machine is signalled via MCAN_RXF1S.DMS.

Figure 39-9. Debug Message Handling State Machine



T0: reset m_can_dma_req output, enable reception of debug messages A, B, and C

T1: reception of debug message A

T2: reception of debug message A

T3: reception of debug message C

T4: reception of debug message B

T5: reception of debug messages A, B

T6: reception of debug message C

T7: DMA transfer completed

T8: reception of debug message A,B,C (message rejected)

39.5.5 Tx Handling

The Tx Handler handles transmission requests for the dedicated Tx Buffers, the Tx FIFO, and the Tx Queue. It controls the transfer of transmit messages to the CAN Core, the Put and Get Indices, and the Tx Event FIFO. Up to 32 Tx Buffers can be set up for message transmission. The CAN mode for transmission (Classic CAN or CAN FD) can be configured separately for each Tx Buffer element. The Tx Buffer element is described in Tx Buffer Element. The table below describes the possible configurations for frame transmission.

Table 39-5. Possible Configurations for Frame Transmission

MCAN_CCCR		Tx Buffer Element		Frame Transmission
BRSE	FDOE	FDF	BRS	
ignored	0	ignored	ignored	Classic CAN
0	1	0	ignored	Classic CAN
0	1	1	ignored	FD without bit rate switching
1	1	0	ignored	Classic CAN
1	1	1	0	FD without bit rate switching
1	1	1	1	FD with bit rate switching

Note: AUTOSAR requires at least three Tx Queue Buffers and support of transmit cancellation.

The Tx Handler starts a Tx scan to check for the highest priority pending Tx request (Tx Buffer with lowest Message ID) when MCAN_TXBRP is updated, or when a transmission has been started.

39.5.5.1 Transmit Pause

The transmit pause feature is intended for use in CAN systems where the CAN message identifiers are (permanently) specified to specific values and cannot easily be changed. These message identifiers may have a higher CAN arbitration priority than other defined messages, while in a specific application their relative arbitration priority should be inverse. This may lead to a case where one ECU sends a burst of CAN messages that cause another ECU's CAN messages to be delayed because that other messages have a lower CAN arbitration priority.

If e.g. CAN ECU-1 has the transmit pause feature enabled and is requested by its application software to transmit four messages, it will, after the first successful message transmission, wait for two CAN bit times of bus idle before it is allowed to start the next requested message. If there are other ECUs with pending messages, those messages are started in the idle time, they would not need to arbitrate with the next message of ECU-1. After having received a message, ECU-1 is allowed to start its next transmission as soon as the received message releases the CAN bus.

The transmit pause feature is controlled by bit MCAN_CCCR.TXP. If the bit is set, the MCAN will, each time it has successfully transmitted a message, pause for two CAN bit times before starting the next transmission. This enables other CAN nodes in the network to transmit messages even if their messages have lower prior identifiers. Default is transmit pause disabled (MCAN_CCCR.TXP = '0').

This feature looses up burst transmissions coming from a single node and it protects against "babbling idiot" scenarios where the application program erroneously requests too many transmissions.

39.5.5.2 Dedicated Tx Buffers

Dedicated Tx Buffers are intended for message transmission under complete control of the processor. Each dedicated Tx Buffer is configured with a specific Message ID. In case that multiple Tx Buffers are configured with the same Message ID, the Tx Buffer with the lowest buffer number is transmitted first.

If the data section has been updated, a transmission is requested by an Add Request via MCAN_TXBAR.ARn. The requested messages arbitrate internally with messages from an optional Tx FIFO or Tx Queue and externally with messages on the CAN bus, and are sent out according to their Message ID.

A dedicated Tx Buffer allocates Element Size 32-bit words in the Message RAM (see the table below). Therefore the start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index (0…31) × Element Size to the Tx Buffer Start Address TXBC.TBSA.

Table 39-6. Tx Buffer / FIFO / Queue Element Size

TXESC.TBDS[2:0]	Data Field [bytes]	Element Size [RAM words]
0	8	4
1	12	5
2	16	6
3	20	7
4	24	8
5	32	10
6	48	14
7	64	18

39.5.5.3 Tx FIFO

Tx FIFO operation is configured by programming MCAN_TXBC.TFQM to '0'. Messages stored in the Tx FIFO are transmitted starting with the message referenced by the Get Index MCAN_TXFQS.TFGI. After each transmission the Get Index is incremented cyclically until the Tx FIFO is empty. The Tx FIFO enables transmission of messages with the same Message ID from different Tx Buffers in the order these messages have been written to the Tx FIFO. The MCAN calculates the Tx FIFO Free Level MCAN_TXFQS.TFFL as difference between Get and Put Index. It indicates the number of available (free) Tx FIFO elements.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQPI. An Add Request increments the Put Index to the next free Tx FIFO element. When the Put

Index reaches the Get Index, Tx FIFO Full (MCAN_TXFQS.TFQF = ‘1’) is signalled. In this case no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

When a single message is added to the Tx FIFO, the transmission is requested by writing a ‘1’ to the TXBAR bit related to the Tx Buffer referenced by the Tx FIFO’s Put Index.

When multiple (n) messages are added to the Tx FIFO, they are written to n consecutive Tx Buffers starting with the Put Index. The transmissions are then requested via MCAN_TXBAR. The Put Index is then cyclically incremented by n. The number of requested Tx buffers should not exceed the number of free Tx Buffers as indicated by the Tx FIFO Free Level.

When a transmission request for the Tx Buffer referenced by the Get Index is cancelled, the Get Index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO Free Level is recalculated. When transmission cancellation is applied to any other Tx Buffer, the Get Index and the FIFO Free Level remain unchanged.

A Tx FIFO element allocates Element Size 32-bit words in the Message RAM (see the table Table 39-6). Therefore the start address of the next available (free) Tx FIFO Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQPI (0…31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA.

39.5.5.4 Tx Queue

Tx Queue operation is configured by programming MCAN_TXBC.TFQM to ‘1’. Messages stored in the Tx Queue are transmitted starting with the message with the lowest Message ID (highest priority). In case that multiple Queue Buffers are configured with the same Message ID, the Queue Buffer with the lowest buffer number is transmitted first.

New messages have to be written to the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQPI. An Add Request cyclically increments the Put Index to the next free Tx Buffer. In case that the Tx Queue is full (MCAN_TXFQS.TFQF = ‘1’), the Put Index is not valid and no further message should be written to the Tx Queue until at least one of the requested messages has been sent out or a pending transmission request has been cancelled.

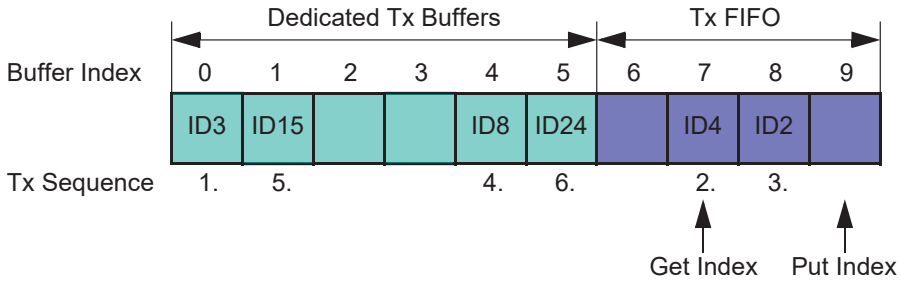
The application may use register MCAN_TXBRP instead of the Put Index and may place messages to any Tx Buffer without pending transmission request.

A Tx Queue Buffer allocates Element Size 32-bit words in the Message RAM (see the table Tx Buffer / FIFO / Queue Element Size). Therefore the start address of the next available (free) Tx Queue Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQPI (0…31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA.

39.5.5.5 Mixed Dedicated Tx Buffers / Tx FIFO

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx FIFO. The number of dedicated Tx Buffers is configured by MCAN_TXBC.NDTB. The number of Tx Buffers assigned to the Tx FIFO is configured by MCAN_TXBC.TFQS. In case MCAN_TXBC.TFQS is programmed to zero, only dedicated Tx Buffers are used.

Figure 39-10. Example of Mixed Configuration Dedicated Tx Buffers / Tx FIFO



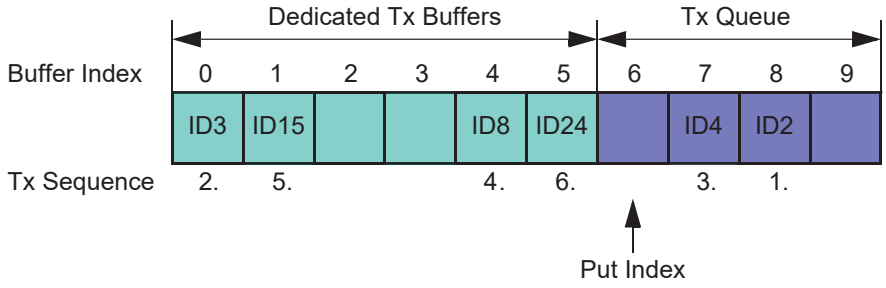
Tx prioritization:

- Scan dedicated Tx Buffers and oldest pending Tx FIFO Buffer (referenced by MCAN_TXFS.TFGI)
- Buffer with lowest Message ID gets highest priority and is transmitted next

39.5.5.6 Mixed Dedicated Tx Buffers / Tx Queue

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx Queue. The number of dedicated Tx Buffers is configured by MCAN_TXBC.NDTB. The number of Tx Queue Buffers is configured by MCAN_TXBC.TFQS. In case MCAN_TXBC.TFQS is programmed to zero, only dedicated Tx Buffers are used.

Figure 39-11. Example of Mixed Configuration Dedicated Tx Buffers / Tx Queue



Tx prioritization:

- Scan all Tx Buffers with activated transmission request
- Tx Buffer with lowest Message ID gets highest priority and is transmitted next

39.5.5.7 Transmit Cancellation

The MCAN supports transmit cancellation. This feature is especially intended for gateway applications and AUTOSAR-based applications. To cancel a requested transmission from a dedicated Tx Buffer or a Tx Queue Buffer, the processor has to write a '1' to the corresponding bit position (=number of Tx Buffer) of register MCAN_TXBCR. Transmit cancellation is not intended for Tx FIFO operation.

Successful cancellation is signalled by setting the corresponding bit of register MCAN_TXBCF to '1'.

In case a transmit cancellation is requested while a transmission from a Tx Buffer is already ongoing, the corresponding TXBRP bit remains set as long as the transmission is in progress. If the transmission was successful, the corresponding MCAN_TXBTO and MCAN_TXBCF bits are set. If the transmission was not successful, it is not repeated and only the corresponding MCAN_TXBCF bit is set.

Note: In case a pending transmission is cancelled immediately before this transmission could have been started, there follows a short time window where no transmission is started even if another message is also pending in this node. This may enable another node to transmit a message which may have a lower priority than the second message in this node.

39.5.5.8 Tx Event Handling

To support Tx event handling the MCAN has implemented a Tx Event FIFO. After the MCAN has transmitted a message on the CAN bus, Message ID and timestamp are stored in a Tx Event FIFO element. To link a Tx event to a Tx Event FIFO element, the Message Marker from the transmitted Tx Buffer is copied into the Tx Event FIFO element.

The Tx Event FIFO can be configured to a maximum of 32 elements. The Tx Event FIFO element is described in Debug on CAN Support.

When a Tx Event FIFO full condition is signalled by IR.TEFF, no further elements are written to the Tx Event FIFO until at least one element has been read out and the Tx Event FIFO Get Index has been incremented. In case a Tx event occurs while the Tx Event FIFO is full, this event is discarded and interrupt flag MCAN_IR.TEFL is set.

To avoid a Tx Event FIFO overflow, the Tx Event FIFO watermark can be used. When the Tx Event FIFO fill level reaches the Tx Event FIFO watermark configured by MCAN_TXEFC.EFWM, interrupt flag MCAN_IR.TEFW is set.

When reading from the Tx Event FIFO, two times the Tx Event FIFO Get Index MCAN_TXEFS.EFGI has to be added to the Tx Event FIFO start address MCAN_TXEFC.EFSA.

39.5.6 FIFO Acknowledge Handling

The Get Indices of Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index in the registers MCAN_RXF0A, MCAN_RXF1A and MCAN_TXEFA. Writing to the FIFO

Acknowledge Index will set the FIFO Get Index to the FIFO Acknowledge Index plus one and thereby updates the FIFO Fill Level. There are two use cases:

When only a single element has been read from the FIFO (the one being pointed to by the Get Index), this Get Index value is written to the FIFO Acknowledge Index.

When a sequence of elements has been read from the FIFO, it is sufficient to write the FIFO Acknowledge Index only once at the end of that read sequence (value: Index of the last element read), to update the FIFO's Get Index.

Due to the fact that the processor has free access to the MCAN's Message RAM, special care has to be taken when reading FIFO elements in an arbitrary order (Get Index not considered). This might be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case the FIFO's Acknowledge Index should not be written because this would set the Get Index to a wrong position and also alters the FIFO's Fill Level. In this case some of the older FIFO elements would be lost.

Note: The application has to ensure that a valid value is written to the FIFO Acknowledge Index. The MCAN does not check for erroneous values.

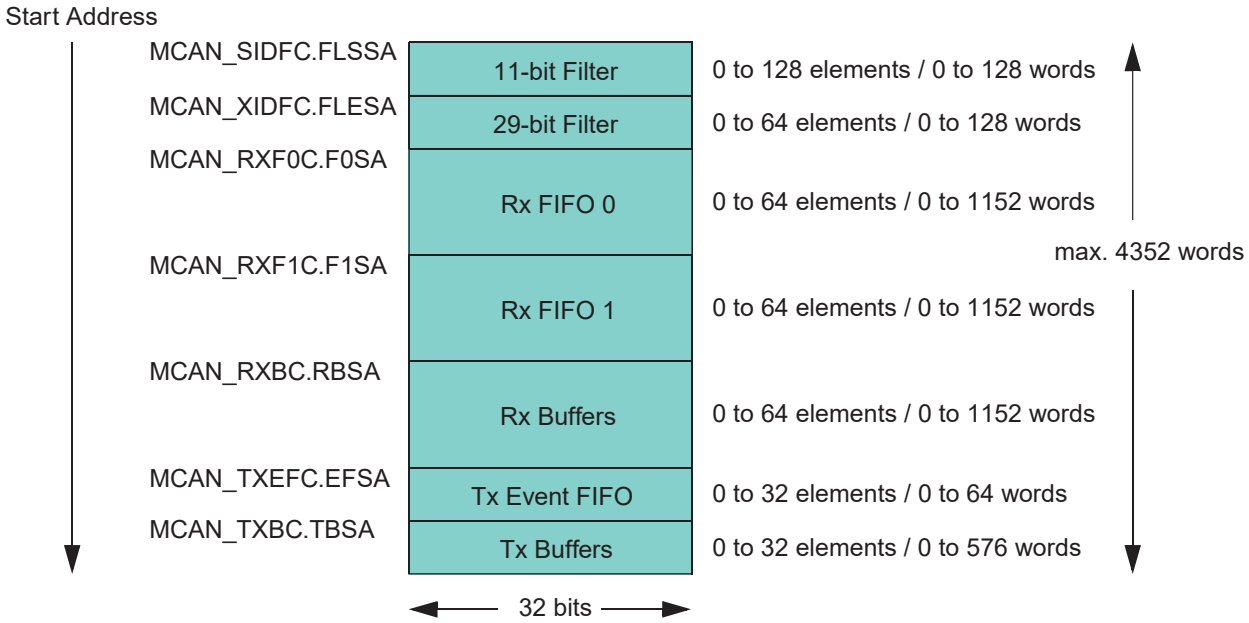
39.5.7 Message RAM

39.5.7.1 Message RAM Configuration

The Message RAM has a width of 32 bits. The MCAN module can be configured to allocate up to 4352 words in the Message RAM. It is not necessary to configure each of the sections listed in the figure below, nor is there any restriction with respect to the sequence of the sections.

When operated in CAN FD mode, the required Message RAM size depends on the element size configured for Rx FIFO0, Rx FIFO1, Rx Buffers, and Tx Buffers via MCAN_RXESC.F0DS, MCAN_RXESC.F1DS, MCAN_RXESC.RBDS, and MCAN_TXESC.TBDS.

Figure 39-12. Message RAM Configuration



When the MCAN addresses the Message RAM, it addresses 32-bit words, not single bytes. The configurable start addresses are 32-bit word addresses; i.e., only bits 15 to 2 are evaluated, the two least significant bits are ignored.

Note: The MCAN does not check for erroneous configuration of the Message RAM. The configuration of the start addresses of the different sections and the number of elements of each section must be checked carefully to avoid falsification or loss of data.

39.5.7.2 Rx Buffer and FIFO Element

Up to 64 Rx Buffers and two Rx FIFOs can be configured in the Message RAM. Each Rx FIFO section can be configured to store up to 64 received messages. The structure of a Rx Buffer / FIFO element is shown in the table

below. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register MCAN_RXESC.

Table 39-7. Rx Buffer and FIFO Element

	31			24	23				16	15		8	7		0
R0	ESI	XTD	RTR	ID[28:0]											
R1	ANMF	FIDX[6:0]			–	FDF	BRS	DLC[3:0]		RXTS[15:0]					
R2	DB3[7:0]				DB2[7:0]					DB1[7:0]			DB0[7:0]		
R3	DB7[7:0]				DB6[7:0]					DB5[7:0]			DB4[7:0]		
...	...				...					...			...		
Rn	DBm[7:0]				DBm-1[7:0]					DBm-2[7:0]			DBm-3[7:0]		

• R0 Bit 31 ESI: Error State Indicator

0: Transmitting node is error active.

1: Transmitting node is error passive.

• R0 Bit 30 XTD: Extended Identifier

Signals to the processor whether the received frame has a standard or extended identifier.

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• R0 Bit 29 RTR: Remote Transmission Request

Signals to the processor whether the received frame is a data frame or a remote frame.

0: Received frame is a data frame.

1: Received frame is a remote frame.

Note: There are no remote frames in CAN FD format. In case a CAN FD frame was received (FDF = 1), bit RTR reflects the state of the reserved bit r1.

• R0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].

• R1 Bit 31 ANMF: Accepted Non-matching Frame

Acceptance of non-matching frames may be enabled via MCAN_GFC.ANFS and MCAN_GFC.ANFE.

0: Received frame matching filter index FIDX.

1: Received frame did not match any Rx filter element.

• R1 Bits 30:24 FIDX[6:0]: Filter Index

0-127: Index of matching Rx acceptance filter element (invalid if ANMF = ‘1’).
Range is 0 to MCAN_SIDFC.LSS - 1 resp. MCAN_XIDFC.LSE - 1.

• R1 Bit 21 FDF: FD Format

0: Standard frame format.

1: CAN FD frame format (new DLC-coding and CRC).

• R1 Bit 20 BRS: Bit Rate Switch

0: Frame received without bit rate switching.

1: Frame received with bit rate switching.

Note:
Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled (MCAN_CCCR.FDOE = 1). Bit BRS is only evaluated when in addition MCAN_CCCR.BRSE = 1.

• R1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: received frame has 0-8 data bytes.

9-15: CAN: received frame has 8 data bytes.

9-15: CAN FD: received frame has 12/16/20/24/32/48/64 data bytes.

• R1 Bits 15:0 RXTS[15:0]: Rx Timestamp

Timestamp Counter value captured on start of frame reception. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.

• R2 Bits 31:24 DB3[7:0]: Data Byte 3

• R2 Bits 23:16 DB2[7:0]: Data Byte 2

• R2 Bits 15:8 DB1[7:0]: Data Byte 1

• R2 Bits 7:0 DB0[7:0]: Data Byte 0

• R3 Bits 31:24 DB7[7:0]: Data Byte 7

• R3 Bits 23:16 DB6[7:0]: Data Byte 6

• R3 Bits 15:8 DB5[7:0]: Data Byte 5

• R3 Bits 7:0 DB4[7:0]: Data Byte 4

... ...

• Rn Bits 31:24 DBm[7:0]: Data Byte m

• Rn Bits 23:16 DBm-1[7:0]: Data Byte m-1

• Rn Bits 15:8 DBm-2[7:0]: Data Byte m-2

• Rn Bits 7:0 DBm-3[7:0]: Data Byte m-3

Note: Depending on the configuration of the element size (MCAN_RXESC), between two and sixteen 32-bit words (Rn = 3 ..17) are used for storage of a CAN message's data field.

39.5.7.3 Tx Buffer Element

The Tx Buffers section can be configured to hold dedicated Tx Buffers as well as a Tx FIFO / Tx Queue. In case that the Tx Buffers section is shared by dedicated Tx buffers and a Tx FIFO / Tx Queue, the dedicated Tx Buffers start at the beginning of the Tx Buffers section followed by the buffers assigned to the Tx FIFO or Tx Queue. The Tx Handler distinguishes between dedicated Tx Buffers and Tx FIFO / Tx Queue by evaluating the Tx Buffer configuration TXBC.TFQS and TXBC.NDTB. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register TXESC.

Table 39-8. Tx Buffer Element

	31			24	23				16	15	8	7	0
T0	ESI	XTD	RTR	ID[28:0]									
T1	MM[7:0]				EFC	reserved	FDF	BRS	DLC[3:0]	reserved			
T2	DB3[7:0]				DB2[7:0]					DB1[7:0]		DB0[7:0]	
T3	DB7[7:0]				DB6[7:0]					DB5[7:0]		DB4[7:0]	
...	...				...					...		...	
Tn	DBm[7:0]				DBm-1[7:0]					DBm-2[7:0]		DBm-3[7:0]	

• T0 Bit 30 ESI: Error State Indicator

T0 Bit 31 ESI: Error State Indicator

0: ESI bit in CAN FD format depends only on error passive flag

1: ESI bit in CAN FD format transmitted recessive

Note: The ESI bit of the transmit buffer is or'ed with the error passive flag to decide the value of the ESI bit in the transmitted FD frame. As required by the CAN FD protocol specification, an error active node may optionally transmit the ESI bit recessive, but an error passive node will always transmit the ESI bit recessive. This feature can be used in gateway applications when a message from an error passive node is routed to another CAN network.

• T0 Bit 30 XTD: Extended Identifier

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• T0 Bit 29 RTR: Remote Transmission Request

0: Transmit data frame.

1: Transmit remote frame.

Note: When RTR = 1, the MCAN transmits a remote frame according to ISO11898-1, even if MCAN_CCCR.FDOE enables the transmission in CAN FD format.

• T0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier has to be written to ID[28:18].

• T1 Bits 31:24 MM[7:0]: Message Marker

Written by processor during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.

• T1 Bit 23 EFC: Event FIFO Control

0: Do not store Tx events.

1: Store Tx events.

• T1 Bit 21 FDF: FD Format

0: Frame transmitted in Classic CAN format

1: Frame transmitted in CAN FD format

• T1 Bit 20 BRS: Bit Rate Switching

0: CAN FD frames transmitted without bit rate switching

1: CAN FD frames transmitted with bit rate switching

Note:
Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled (MCAN_CCCR.FDOE = 1). Bit BRS is only evaluated when in addition MCAN_CCCR.BRSE = 1.

• T1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: transmit frame has 0-8 data bytes.

9-15: CAN: transmit frame has 8 data bytes.

9-15: CAN FD: transmit frame has 12/16/20/24/32/48/64 data bytes.

• T2 Bits 31:24 DB3[7:0]: Data Byte 3

• T2 Bits 23:16 DB2[7:0]: Data Byte 2

• T2 Bits 15:8 DB1[7:0]: Data Byte 1

• T2 Bits 7:0 DB0[7:0]: Data Byte 0

• T3 Bits 31:24 DB7[7:0]: Data Byte 7

• T3 Bits 23:16 DB6[7:0]: Data Byte 6

• T3 Bits 15:8 DB5[7:0]: Data Byte 5

• T3 Bits 7:0 DB4[7:0]: Data Byte 4

... ...

• Tn Bits 31:24 DBm[7:0]: Data Byte m

• Tn Bits 23:16 DBm-1[7:0]: Data Byte m-1

• Tn Bits 15:8 DBm-2[7:0]: Data Byte m-2

• Tn Bits 7:0 DBm-3[7:0]: Data Byte m-3

Note: Depending on the configuration of the element size (MCAN_TXESC), between two and sixteen 32-bit words (Tn = 3 ..17) are used for storage of a CAN message's data field.

39.5.7.4 Tx Event FIFO Element

Each element stores information about transmitted messages. By reading the Tx Event FIFO the processor gets this information in the order the messages were transmitted. Status information about the Tx Event FIFO can be obtained from register TXEFS.

Table 39-9. Tx Event FIFO Element

	31			24	23						16	15	8	7	0
E0	ESI	XTD	RTR	ID[28:0]											
E1	MM[7:0]				ET [1:0]	FDF	BRS	DLC[3:0]				TXTS[15:0]			

• E0 Bit 31 ESI: Error State Indicator

0: Transmitting node is error active.

1: Transmitting node is error passive.

• E0 Bit 30 XTD: Extended Identifier

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• E0 Bit 29 RTR: Remote Transmission Request

0: Data frame transmitted.

1: Remote frame transmitted.

• E0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].

• E1 Bits 31:24 MM[7:0]: Message Marker

Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status.

• E1 Bit 23:22 ET[1:0]: Event Type

0: Reserved

1: Tx event

2: Transmission in spite of cancellation (always set for transmissions in DAR mode)

3: Reserved

• E1 Bit 21 FDF: FD Format

0: Standard frame format.

1: CAN FD frame format (new DLC-coding and CRC).

• E1 Bit 20 BRS: Bit Rate Switch

0: Frame transmitted without bit rate switching.

1: Frame transmitted with bit rate switching.

• E1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: frame with 0-8 data bytes transmitted.

9-15: CAN: frame with 8 data bytes transmitted.

9-15: CAN FD: frame with 12/16/20/24/32/48/64 data bytes transmitted

• E1 Bits 15:0 TXTS[15:0]: Tx Timestamp

Timestamp Counter value captured on start of frame transmission. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.

39.5.7.5 Standard Message ID Filter Element

Up to 128 filter elements can be configured for 11-bit standard IDs. When accessing a Standard Message ID Filter element, its address is the Filter List Standard Start Address MCAN_SIDFC.FLSSA plus the index of the filter element (0…127).

Table 39-10. Standard Message ID Filter Element

	31					24	23	16	15	8	7	0
S0	SFT[1:0]		SFEC [2:0]			SFID1[10:0]			–	SFID2[10:0]		

• Bits 31:30 SFT[1:0]: Standard Filter Type

0: Range filter from SF1ID to SF2ID (SF2ID ≥ SF1ID)

1: Dual ID filter for SF1ID or SF2ID

2: Classic filter: SF1ID = filter, SF2ID = mask

3: Reserved

• Bit 29:27 SFEC[2:0]: Standard Filter Element Configuration

All enabled filter elements are used for acceptance filtering of standard frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If SFEC = “100”, “101”, or “110” a match sets interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case register HPMS is updated with the status of the priority match.

0: Disable filter element

1: Store in Rx FIFO 0 if filter matches

2: Store in Rx FIFO 1 if filter matches

3: Reject ID if filter matches

4: Set priority if filter matches

5: Set priority and store in FIFO 0 if filter matches

6: Set priority and store in FIFO 1 if filter matches

7: Store into Rx Buffer or as debug message, configuration of SFT[1:0] ignored

• Bits 26:16 SFID1[10:0]: Standard Filter ID 1

First ID of standard ID filter element.

When filtering for Rx Buffers or for debug messages this field defines the ID of a standard message to be stored. The received identifiers must match exactly, no masking mechanism is used.

• Bits 10:0 SFID2[10:0]: Standard Filter ID 2

This field has a different meaning depending on the configuration of SFEC:

- SFEC = “001”...“110”–Second ID of standard ID filter element
- SFEC = “111”–Filter for Rx Buffers or for debug messages

SFID2[10:9] decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence.

0: Store message in a Rx buffer

1: Debug Message A

2: Debug Message B

3: Debug Message C

SFID2[5:0] defines the index of the dedicated Rx Buffer element to which a matching message is stored.

39.5.7.6 Extended Message ID Filter Element

Up to 64 filter elements can be configured for 29-bit extended IDs. When accessing an Extended Message ID Filter element, its address is the Filter List Extended Start Address MCAN_XIDFC.FLESA plus two times the index of the filter element (0…63).

Table 39-11. Extended Message ID Filter Element

	31			24	23	16	15	8	7	0
F0	EFEC [2:0]			EFID1[28:0]						
F1	EFT[1:0]		–	EFID2[28:0]						

• F0 Bit 31:29 EFEC[2:0]: Extended Filter Element Configuration

All enabled filter elements are used for acceptance filtering of extended frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If EFEC = “100”, “101”, or “110”, a match sets the interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case, register MCAN_HPMS is updated with the status of the priority match.

0: Disable filter element

1: Store in Rx FIFO 0 if filter matches

2: Store in Rx FIFO 1 if filter matches

3: Reject ID if filter matches

4: Set priority if filter matches

5: Set priority and store in FIFO 0 if filter matches

6: Set priority and store in FIFO 1 if filter matches

7: Store into Rx Buffer or as debug message, configuration of EFT[1:0] ignored

• F0 Bits 28:0 EFID1[28:0]: Extended Filter ID 1

First ID of extended ID filter element.

When filtering for Rx Buffers or for debug messages this field defines the ID of an extended message to be stored. The received identifiers must match exactly, only MCAN_XIDAM masking mechanism (see Extended Message ID Filtering) is used.

• F1 Bits 31:30 EFT[1:0]: Extended Filter Type

0: Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID)

1: Dual ID filter for EF1ID or EF2ID

2: Classic filter: EF1ID = filter, EF2ID = mask

3: Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID), MCAN_XIDAM mask not applied

• F1 Bits 28:0 EFID2[28:0]: Extended Filter ID 2

This field has a different meaning depending on the configuration of EFEC:

• EFEC = “001”...“110”–Second ID of extended ID filter element

• EFEC = “111”–Filter for Rx Buffers or for debug messages

EFID2[10:9] decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence.

0: Store message in an Rx buffer

1: Debug Message A

2: Debug Message B

3: Debug Message C

EFID2[5:0] defines the index of the dedicated Rx Buffer element to which a matching message is stored.

39.5.8 Hardware Reset Description

After hardware reset, the registers of the MCAN hold the reset values listed in the register descriptions. Additionally the Bus_Off state is reset and the output CANTX is set to recessive (HIGH). The value 0x0001 (MCAN_CCCR.INIT = ‘1’) in the CC Control register enables software initialization. The MCAN does not influence the CAN bus until the processor resets MCAN_CCCR.INIT to ‘0’.

39.5.9 Access to Reserved Register Addresses

In case the application software accesses one of the reserved addresses in the MCAN register map (read or write access), interrupt flag MCAN_IR.ARA is set and, if enabled, the selected interrupt line is risen.

39.6 Register Summary

Offset	Name	Bit Pos.								
0x00 ... 0x03	Reserved									
0x04	MCAN_ENDN	7:0	ETV[7:0]							
		15:8	ETV[15:8]							
		23:16	ETV[23:16]							
		31:24	ETV[31:24]							
0x08	MCAN_CUST	7:0	CSV[7:0]							
		15:8	CSV[15:8]							
		23:16	CSV[23:16]							
		31:24	CSV[31:24]							
0x0C	MCAN_DBTP	7:0	DTSEG2[3:0]					DSJW[2:0]		
		15:8				DTSEG1[4:0]				
		23:16	TDC			DBRP[4:0]				
		31:24								
0x10	MCAN_TEST	7:0	RX	TX[1:0]		LBCK				
		15:8								
		23:16								
		31:24								
0x14	MCAN_RWD	7:0	WDC[7:0]							
		15:8	WDV[7:0]							
		23:16								
		31:24								
0x18	MCAN_CCCR	7:0	TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
		15:8	NISO	TXP	EFBI	PXHD			BRSE	FDOE
		23:16								
		31:24								
0x1C	MCAN_NBTP	7:0		NTSEG2[6:0]						
		15:8	NTSEG1[7:0]							
		23:16	NBRP[7:0]							
		31:24	NSJW[6:0]							NBRP[8]
0x20	MCAN_TSCC	7:0							TSS[1:0]	
		15:8								
		23:16					TCP[3:0]			
		31:24								
0x24	MCAN_TSCV	7:0	TSC[7:0]							
		15:8	TSC[15:8]							
		23:16								
		31:24								
0x28	MCAN_TOCC	7:0						TOS[1:0]		ETOC
		15:8								
		23:16	TOP[7:0]							
		31:24	TOP[15:8]							
0x2C	MCAN_TOCV	7:0	TOC[7:0]							
		15:8	TOC[15:8]							
		23:16								
		31:24								
0x30 ... 0x3F	Reserved									
0x40	MCAN_ECR	7:0	TEC[7:0]							
		15:8	RP	REC[6:0]						
		23:16	CEL[7:0]							
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x44	MCAN_PSR	7:0	BO	EW	EP	ACT[1:0]		LEC[2:0]		
		15:8		PXE	RFDF	RBRS	RESI	DLEC[2:0]		
		23:16		TDCV[6:0]						
		31:24								
0x48	MCAN_TDCR	7:0		TDCF[6:0]						
		15:8		TDCO[6:0]						
		23:16								
		31:24								
0x4C ... 0x4F	Reserved									
0x50	MCAN_IR	7:0	RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N
		15:8	TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM
		23:16	EP	ELO			DRX	TOO	MRAF	TSW
		31:24			ARA	PED	PEA	WDI	BO	EW
0x54	MCAN_IE	7:0	RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE
		15:8	TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME
		23:16	EPE	ELOE			DRXE	TOOE	MRAFE	TSWE
		31:24			ARAE	PEDE	PEAE	WDIE	BOE	EWE
0x58	MCAN_ILS	7:0	RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL
		15:8	TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML
		23:16	EPL	ELOL			DRXL	TOOL	MRAFL	TSWL
		31:24			ARAL	PEDL	PEAL	WDIL	BOL	EWL
0x5C	MCAN_ILE	7:0							EINT1	EINT0
		15:8								
		23:16								
		31:24								
0x60 ... 0x7F	Reserved									
0x80	MCAN_GFC	7:0			ANFS[1:0]		ANFE[1:0]		RRFS	RRFE
		15:8								
		23:16								
		31:24								
0x84	MCAN_SIDFC	7:0	FLSSA[5:0]							
		15:8	FLSSA[13:6]							
		23:16	LSS[7:0]							
		31:24								
0x88	MCAN_XIDFC	7:0	FLESA[5:0]							
		15:8	FLESA[13:6]							
		23:16		LSE[6:0]						
		31:24								
0x8C ... 0x8F	Reserved									
0x90	MCAN_XIDAM	7:0	EIDM[7:0]							
		15:8	EIDM[15:8]							
		23:16	EIDM[23:16]							
		31:24				EIDM[28:24]				
0x94	MCAN_HPMS	7:0	MSI[1:0]		BIDX[5:0]					
		15:8	FLST	FIDX[6:0]						
		23:16								
		31:24								
0x98	MCAN_NDAT1	7:0	ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
		15:8	ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8
		23:16	ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16
		31:24	ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24

.....continued

Offset	Name	Bit Pos.								
0x9C	MCAN_NDAT2	7:0	ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32
		15:8	ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40
		23:16	ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48
		31:24	ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56
0xA0	MCAN_RXF0C	7:0	F0SA[5:0]							
		15:8	F0SA[13:6]							
		23:16		F0S[6:0]						
		31:24	F0OM	F0WM[6:0]						
0xA4	MCAN_RXF0S	7:0		F0FL[6:0]						
		15:8			F0GI[5:0]					
		23:16			F0PI[5:0]					
		31:24							RF0L	F0F
0xA8	MCAN_RXF0A	7:0			F0AI[5:0]					
		15:8								
		23:16								
		31:24								
0xAC	MCAN_RXBC	7:0	RBSA[5:0]							
		15:8	RBSA[13:6]							
		23:16								
		31:24								
0xB0	MCAN_RXF1C	7:0	F1SA[5:0]							
		15:8	F1SA[13:6]							
		23:16		F1S[6:0]						
		31:24	F1OM	F1WM[6:0]						
0xB4	MCAN_RXF1S	7:0		F1FL[6:0]						
		15:8			F1GI[5:0]					
		23:16			F1PI[5:0]					
		31:24	DMS[1:0]						RF1L	F1F
0xB8	MCAN_RXF1A	7:0			F1AI[5:0]					
		15:8								
		23:16								
		31:24								
0xBC	MCAN_RXESC	7:0		F1DS[2:0]				F0DS[2:0]		
		15:8						RBDS[2:0]		
		23:16								
		31:24								
0xC0	MCAN_TXBC	7:0	TBSA[5:0]							
		15:8	TBSA[13:6]							
		23:16			NDTB[5:0]					
		31:24		TFQM	TFQS[5:0]					
0xC4	MCAN_TXFQS	7:0			TFFL[5:0]					
		15:8				TFGI[4:0]				
		23:16			TFQF	TFQPI[4:0]				
		31:24								
0xC8	MCAN_TXESC	7:0						TBDS[2:0]		
		15:8								
		23:16								
		31:24								
0xCC	MCAN_TXBRP	7:0	TRP7	TRP6	TRP5	TRP4	TRP3	TRP2	TRP1	TRP0
		15:8	TRP15	TRP14	TRP13	TRP12	TRP11	TRP10	TRP9	TRP8
		23:16	TRP23	TRP22	TRP21	TRP20	TRP19	TRP18	TRP17	TRP16
		31:24	TRP31	TRP30	TRP29	TRP28	TRP27	TRP26	TRP25	TRP24
0xD0	MCAN_TXBAR	7:0	AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
		15:8	AR15	AR14	AR13	AR12	AR11	AR10	AR9	AR8
		23:16	AR23	AR22	AR21	AR20	AR19	AR18	AR17	AR16
		31:24	AR31	AR30	AR29	AR28	AR27	AR26	AR25	AR24

.....continued

Offset	Name	Bit Pos.								
0xD4	MCAN_TXBCR	7:0	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
		15:8	CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8
		23:16	CR23	CR22	CR21	CR20	CR19	CR18	CR17	CR16
		31:24	CR31	CR30	CR29	CR28	CR27	CR26	CR25	CR24
0xD8	MCAN_TXBTO	7:0	TO7	TO6	TO5	TO4	TO3	TO2	TO1	TO0
		15:8	TO15	TO14	TO13	TO12	TO11	TO10	TO9	TO8
		23:16	TO23	TO22	TO21	TO20	TO19	TO18	TO17	TO16
		31:24	TO31	TO30	TO29	TO28	TO27	TO26	TO25	TO24
0xDC	MCAN_TXBCF	7:0	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
		15:8	CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8
		23:16	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
		31:24	CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24
0xE0	MCAN_TXBTIE	7:0	TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0
		15:8	TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8
		23:16	TIE23	TIE22	TIE21	TIE20	TIE19	TIE18	TIE17	TIE16
		31:24	TIE31	TIE30	TIE29	TIE28	TIE27	TIE26	TIE25	TIE24
0xE4	MCAN_TXBCIE	7:0	CFIE7	CFIE6	CFIE5	CFIE4	CFIE3	CFIE2	CFIE1	CFIE0
		15:8	CFIE15	CFIE14	CFIE13	CFIE12	CFIE11	CFIE10	CFIE9	CFIE8
		23:16	CFIE23	CFIE22	CFIE21	CFIE20	CFIE19	CFIE18	CFIE17	CFIE16
		31:24	CFIE31	CFIE30	CFIE29	CFIE28	CFIE27	CFIE26	CFIE25	CFIE24
0xE8 ... 0xEF	Reserved									
0xF0	MCAN_TXEFC	7:0	EFSA[5:0]							
		15:8	EFSA[13:6]							
		23:16			EFS[5:0]					
		31:24			EFWM[5:0]					
0xF4	MCAN_TXEFS	7:0			EFFL[5:0]					
		15:8				EFGI[4:0]				
		23:16				EFPI[4:0]				
		31:24							TEFL	EFF
0xF8	MCAN_TXEFA	7:0				EFAI[4:0]				
		15:8								
		23:16								
		31:24								

39.6.1 MCAN Endian Register

Name: MCAN_ENDN
Offset: 0x04
Reset: 0x87654321
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ETV[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	1	0	0	0	0	1	1	1

Bit	23	22	21	20	19	18	17	16
	ETV[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	1	1	0	0	1	0	1

Bit	15	14	13	12	11	10	9	8
	ETV[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	1	0	0	0	0	1	1

Bit	7	6	5	4	3	2	1	0
	ETV[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	1	0	0	0	0	1

Bits 31:0 – ETV[31:0] Endianness Test Value
The endianness test value is 0x87654321.

39.6.2 MCAN Customer Register

Name: MCAN_CUST
Offset: 0x08
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	CSV[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CSV[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CSV[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CSV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – CSV[31:0] Customer-specific Value
Customer-specific value.

39.6.3 MCAN Data Bit Timing and Prescaler Register

Name: MCAN_DBTP
Offset: 0x0C
Reset: 0x00000A33
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

The CAN bit time may be programmed in the range of 4 to 25 time quanta. The CAN time quantum may be programmed in the range of 1 to 32 CAN core clock periods. t_q = (DBRP + 1) CAN core clock periods.

DTSEG1 is the sum of Prop_Seg and Phase_Seg1. DTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [DTSEG1 + DTSEG2 + 3] t_q
or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

With a CAN core clock frequency of 8 MHz, the reset value of 0x00000A33 configures the MCAN for a fast bit rate of 500 kbit/s.

The bit rate configured for the CAN FD data phase via MCAN_DBTP must be higher than or equal to the bit rate configured for the arbitration phase via MCAN_NBTP.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	TDC			DBRP[4:0]				
Access	R/W			R/W	R/W	R/W	R/W	R/W
Reset	0			0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				DTSEG1[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	1	0	1	0

Bit	7	6	5	4	3	2	1	0
	DTSEG2[3:0]					DSJW[2:0]		
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset	0	0	1	1		0	1	1

Bit 23 – TDC Transmitter Delay Compensation
0 (DISABLED): Transmitter Delay Compensation disabled.
1 (ENABLED): Transmitter Delay Compensation enabled.

Bits 20:16 – DBRP[4:0] Data Bit Rate Prescaler
The value by which the peripheral clock is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

Bits 12:8 – DTSEG1[4:0] Data Time Segment Before Sample Point
0: Forbidden.
1 to 31: The duration of time segment is t_q x (DTSEG1 + 1).

Bits 7:4 – DTSEG2[3:0] Data Time Segment After Sample Point
The duration of time segment is t_q x (DTSEG2 + 1).

Bits 2:0 – DSJW[2:0] Data (Re) Synchronization Jump Width
The duration of a synchronization jump is t_q x (DSJW + 1).

39.6.4 MCAN Test Register

Name: MCAN_TEST
Offset: 0x10
Reset: 0x00000000
Property: Read/Write

Write access to the Test Register has to be enabled by setting bit MCAN_CCCR.TEST to '1'.

All MCAN Test Register functions are set to their reset values when bit MCAN_CCCR.TEST is cleared.

Loop Back mode and software control of pin CANTX are hardware test modes. Programming of TX ≠ 0 disturbs the message transfer on the CAN bus.

The reset value for MCAN_TEST.RX is undefined.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	RX	TX[1:0]		LBCK				
Access	R	R/W	R/W	R/W				
Reset	x	0	0	0				

Bit 7 – RX Receive Pin (read-only)
Monitors the actual value of pin CANRX.
The reset value for this bit is undefined.

Value	Description
0	The CAN bus is dominant (CANRX = '0').
1	The CAN bus is recessive (CANRX = '1').

Bits 6:5 – TX[1:0] Control of Transmit Pin (read/write)

Value	Name	Description
0	RESET	Reset value, CANTX controlled by the CAN Core, updated at the end of the CAN bit time.
1	SAMPLE_POINT_MONITORING	Sample Point can be monitored at pin CANTX.
2	DOMINANT	Dominant ('0') level at pin CANTX.
3	RECESSIVE	Recessive ('1') at pin CANTX.

Bit 4 – LBCK Loop Back Mode (read/write)
0 (DISABLED): Reset value. Loop Back mode is disabled.
1 (ENABLED): Loop Back mode is enabled (see Test Modes).

39.6.5 MCAN RAM Watchdog Register

Name: MCAN_RWD
Offset: 0x14
Reset: 0x00000000
Property: Read/Write

The RAM Watchdog monitors the Message RAM response time. A Message RAM access via the MCAN's Generic Master Interface starts the Message RAM Watchdog Counter with the value configured by MCAN_RWD.WDC. The counter is reloaded with MCAN_RWD.WDC when the Message RAM signals successful completion by activating its READY output. In case there is no response from the Message RAM until the counter has counted down to zero, the counter stops and interrupt flag MCAN_IR.WDI is set. The RAM Watchdog Counter is clocked by the system bus clock (peripheral clock).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	WDV[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	WDC[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:8 – WDV[7:0] Watchdog Value (read-only)
Watchdog Counter Value for the current message located in RAM.

Bits 7:0 – WDC[7:0] Watchdog Configuration (read/write)
Start value of the Message RAM Watchdog Counter. The counter is disabled when WDC is cleared.

39.6.6 MCAN CC Control Register

Name: MCAN_CCCR
Offset: 0x18
Reset: 0x00000001
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	NISO	TXP	EFBI	PXHD			BRSE	FDOE
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	0	0	0	0			0	0

Bit	7	6	5	4	3	2	1	0
	TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
Access	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	1

Bit 15 – NISO Non-ISO Operation
If this bit is set, the MCAN uses the CAN FD frame format as specified by the Bosch CAN FD Specification V1.0.

Value	Description
0	CAN FD frame format according to ISO11898-1 (default).
1	CAN FD frame format according to Bosch CAN FD Specification V1.0.

Bit 14 – TXP Transmit Pause (read/write, write protection)
If this bit is set, the MCAN pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame (see Tx Handling).

Value	Description
0	Transmit pause disabled.
1	Transmit pause enabled.

Bit 13 – EFBI Edge Filtering during Bus Integration (read/write, write protection)

Value	Description
0	Edge filtering is disabled.
1	Edge filtering is enabled. Two consecutive dominant tq required to detect an edge for hard synchronization.

Bit 12 – PXHD Protocol Exception Event Handling (read/write, write protection)

Value	Description
0	Protocol exception handling enabled.
1	Protocol exception handling disabled.

Bit 9 – BRSE Bit Rate Switching Enable (read/write, write protection)
0 (DISABLED): Bit rate switching for transmissions disabled.
1 (ENABLED): Bit rate switching for transmissions enabled.

Bit 8 – FDOE CAN FD Operation Enable (read/write, write protection)
0 (DISABLED): FD operation disabled.
1 (ENABLED): FD operation enabled.

Bit 7 – TEST Test Mode Enable (read/write, write protection against '1')
0 (DISABLED): Normal operation, MCAN_TEST register holds reset values.
1 (ENABLED): Test mode, write access to MCAN_TEST register enabled.

Bit 6 – DAR Disable Automatic Retransmission (read/write, write protection)
0 (AUTO_RETX): Automatic retransmission of messages not transmitted successfully enabled.
1 (NO_AUTO_RETX): Automatic retransmission disabled.

Bit 5 – MON Bus Monitoring Mode (read/write, write protection against '1')
0 (DISABLED): Bus Monitoring mode is disabled.
1 (ENABLED): Bus Monitoring mode is enabled.

Bit 4 – CSR Clock Stop Request (read/write)
0 (NO_CLOCK_STOP): No clock stop is requested.
1 (CLOCK_STOP): Clock stop requested. When clock stop is requested, first INIT and then CSA will be set after all pending transfer requests have been completed and the CAN bus reached idle.

Bit 3 – CSA Clock Stop Acknowledge (read-only)

Value	Description
0	No clock stop acknowledged.
1	MCAN may be set in power down by stopping the peripheral clock and the CAN core clock.

Bit 2 – ASM Restricted Operation Mode (read/write, write protection against '1')
For a description of the Restricted Operation mode see Restricted Operation Mode.
0 (NORMAL): Normal CAN operation.
1 (RESTRICTED): Restricted Operation mode active.

Bit 1 – CCE Configuration Change Enable (read/write, write protection)
0 (PROTECTED): The processor has no write access to the protected configuration registers.
1 (CONFIGURABLE): The processor has write access to the protected configuration registers (while MCAN_CCCR.INIT = '1').

Bit 0 – INIT Initialization (read/write)
Due to the synchronization mechanism between the two clock domains, there may be a delay until the value written to INIT can be read back. Therefore the programmer has to ensure that the previous value written to INIT has been accepted by reading INIT before setting INIT to a new value.
0 (DISABLED): Normal operation.
1 (ENABLED): Initialization is started.

39.6.7 MCAN Nominal Bit Timing and Prescaler Register

Name: MCAN_NBTP
Offset: 0x1C
Reset: 0x06000A03
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN_CCCR.

The CAN bit time may be programmed in the range of 4 to 385 time quanta. The CAN time quantum may be programmed in the range of 1 to 512 CAN core clock periods. $t_q = t_{core\ clock}$ x (NBRP + 1).

NTSEG1 is the sum of Prop_Seg and Phase_Seg1. NTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [NTSEG1 + NTSEG2 + 3] t_q
or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

With a CAN core clock frequency of 8 MHz, the reset value of 0x06000A03 configures the MCAN for a bit rate of 500 kbit/s.

Bit	31	30	29	28	27	26	25	24
	NSJW[6:0]							NBRP[8]
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	0

Bit	23	22	21	20	19	18	17	16
	NBRP[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	NTSEG1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	1	0	1	0

Bit	7	6	5	4	3	2	1	0
		NTSEG2[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	1	1

Bits 31:25 – NSJW[6:0] Nominal (Re) Synchronization Jump Width
0 to 127: The duration of a synchronization jump is t_q x (NSJW + 1).

Bits 24:16 – NBRP[8:0] Nominal Bit Rate Prescaler
0 to 511: The value by which the oscillator frequency is divided for generating the CAN time quanta. The CAN time is built up from a multiple of this quanta. CAN time quantum (tq) = $t_{core\ clock}$ x (NBRP + 1)

Bits 15:8 – NTSEG1[7:0] Nominal Time Segment Before Sample Point

Value	Description
0	Reserved; do not use.
1 to 255	The duration of time segment is t_q x (NTSEG1 + 1).

Bits 6:0 – NTSEG2[6:0] Nominal Time Segment After Sample Point

Value	Description
0	Reserved; do not use.
1 to 127	The duration of time segment is t_q x (NTSEG2 + 1).

39.6.8 MCAN Timestamp Counter Configuration Register

Name: MCAN_TSCC
Offset: 0x20
Reset: 0x00000000
Property: Read/Write

For a description of the Timestamp Counter see Timestamp Generation.

With CAN FD, an external counter is required for timestamp generation (TSS = 2).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					TCP[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							TSS[1:0]	
Access							R/W	R/W
Reset							0	0

Bits 19:16 – TCP[3:0] Timestamp Counter Prescaler
Configures the timestamp and timeout counters time unit in multiples of CAN bit times [1…16]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

Bits 1:0 – TSS[1:0] Timestamp Select

Value	Name	Description
0	ALWAYS_0	Timestamp counter value always 0x0000
1	TCP_INC	Timestamp counter value incremented according to TCP
2	EXT_TIMESTAMP	External timestamp counter value used
3	ALWAYS_0	Timestamp counter value always 0x0000

39.6.9 MCAN Timestamp Counter Value Register

Name: MCAN_TSCV
Offset: 0x24
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TSC[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TSC[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 15:0 – TSC[15:0] Timestamp Counter (cleared on write)
The internal/external Timestamp Counter value is captured on start of frame (both Receive and Transmit). When MCAN_TSCC.TSS = 1, the Timestamp Counter is incremented in multiples of CAN bit times [1…16] depending on the configuration of MCAN_TSCC.TCP. A wrap around sets interrupt flag MCAN_IR.TSW. Write access resets the counter to zero.
When MCAN_TSCC.TSS = 2, TSC reflects the external Timestamp Counter value. Thus a write access has no impact.
Note: A “wrap around” is a change of the Timestamp Counter value from non-zero to zero not caused by write access to MCAN_TSCV.

39.6.10 MCAN Timeout Counter Configuration Register

Name: MCAN_TOCC
Offset: 0x28
Reset: 0xFFFF0000
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

For a description of the Timeout Counter, see Timeout Counter.

Bit	31	30	29	28	27	26	25	24
	TOP[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	TOP[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						TOS[1:0]		ETOC
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 31:16 – TOP[15:0] Timeout Period
Start value of the Timeout Counter (down-counter). Configures the Timeout Period.

Bits 2:1 – TOS[1:0] Timeout Select
When operating in Continuous mode, a write to MCAN_TOCV presets the counter to the value configured by MCAN_TOCC.TOP and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by MCAN_TOCC.TOP. Down-counting is started when the first FIFO element is stored.

Value	Name	Description
0	CONTINUOUS	Continuous operation.
1	TX_EV_TIMEOUT	Timeout controlled by Tx Event FIFO.
2	RX0_EV_TIMEOUT	Timeout controlled by Receive FIFO 0.
3	RX1_EV_TIMEOUT	Timeout controlled by Receive FIFO 1.

Bit 0 – ETOC Enable Timeout Counter
0 (NO_TIMEOUT): Timeout Counter disabled.
1 (TOS_CONTROLLED): Timeout Counter enabled.
For use of timeout function with CAN FD, see Timeout Counter.

39.6.11 MCAN Timeout Counter Value Register

Name: MCAN_TOCV
Offset: 0x2C
Reset: 0x0000FFFF
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TOC[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	TOC[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 15:0 – TOC[15:0] Timeout Counter (cleared on write)
The Timeout Counter is decremented in multiples of CAN bit times [1…16] depending on the configuration of MCAN_TSCC.TCP. When decremented to zero, interrupt flag MCAN_IR.TOO is set and the Timeout Counter is stopped. Start and reset/restart conditions are configured via MCAN_TOCC.TOS.

39.6.12 MCAN Error Counter Register

Name: MCAN_ECR
Offset: 0x40
Reset: 0x00000000
Property: Read-only

When MCAN_CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CEL[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RP	REC[6:0]						
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TEC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 23:16 – CEL[7:0] CAN Error Logging (cleared on read)
The counter is incremented each time when a CAN protocol error causes the Transmit Error Counter or the Receive Error Counter to be incremented. It is reset by read access to CEL. The counter stops at 0xFF; the next increment of TEC or REC sets interrupt flag IR.ELO.

Bit 15 – RP Receive Error Passive

Value	Description
0	The Receive Error Counter is below the error passive level of 128.
1	The Receive Error Counter has reached the error passive level of 128.

Bits 14:8 – REC[6:0] Receive Error Counter
Actual state of the Receive Error Counter, values between 0 and 127.

Bits 7:0 – TEC[7:0] Transmit Error Counter
Actual state of the Transmit Error Counter, values between 0 and 255.

39.6.13 MCAN Protocol Status Register

Name: MCAN_PSR
Offset: 0x44
Reset: 0x00000707
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
		TDCV[6:0]						
Access		R	R	R	R	R	R	R
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
		PXE	RFDF	RBRS	RESI	DLEC[2:0]		
Access								
Reset		0	0	0	0	1	1	1

Bit	7	6	5	4	3	2	1	0
	BO	EW	EP	ACT[1:0]		LEC[2:0]		
Access	R	R	R	R	R			
Reset	0	0	0	0	0	1	1	1

Bits 22:16 – TDCV[6:0] Transmitter Delay Compensation Value
0 to 127: Position of the secondary sample point, in CAN core clock periods, defined by the sum of the measured delay from CANTX to CANRX and MCAN_TDCR.TDCO.

Bit 14 – PXE Protocol Exception Event (cleared on read)

Value	Description
0	No protocol exception event occurred since last read access
1	Protocol exception event occurred

Bit 13 – RFDF Received a CAN FD Message (cleared on read)
This bit is set independently from acceptance filtering.

Value	Description
0	Since this bit was reset by the CPU, no CAN FD message has been received
1	Message in CAN FD format with FDF flag set has been received

Bit 12 – RBRS BRS Flag of Last Received CAN FD Message (cleared on read)
This bit is set together with RFDF, independently from acceptance filtering.

Value	Description
0	Last received CAN FD message did not have its BRS flag set.
1	Last received CAN FD message had its BRS flag set.

Bit 11 – RESI ESI Flag of Last Received CAN FD Message (cleared on read)
This bit is set together with RFDF, independently from acceptance filtering.

Value	Description
0	Last received CAN FD message did not have its ESI flag set.
1	Last received CAN FD message had its ESI flag set.

Bits 10:8 – DLEC[2:0] Data Phase Last Error Code (set to 111 on read)
Type of last error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. This field will be cleared to zero when a CAN FD format frame with its BRS flag set has been transferred (reception or transmission) without error.

Bit 7 – BO Bus_Off Status

Value	Description
0	The MCAN is not Bus_Off.
1	The MCAN is in Bus_Off state.

Bit 6 – EW Warning Status

Value	Description
0	Both error counters are below the Error_Warning limit of 96.
1	At least one of error counter has reached the Error_Warning limit of 96.

Bit 5 – EP Error Passive

Value	Description
0	The MCAN is in the Error_Active state. It normally takes part in bus communication and sends an active error flag when an error has been detected.
1	The MCAN is in the Error_Passive state.

Bits 4:3 – ACT[1:0] Activity
Monitors the CAN communication state of the CAN module.

Value	Name	Description
0	SYNCHRONIZING	Node is synchronizing on CAN communication
1	IDLE	Node is neither receiver nor transmitter
2	RECEIVER	Node is operating as receiver
3	TRANSMITTER	Node is operating as transmitter

Bits 2:0 – LEC[2:0] Last Error Code (set to 111 on read)
The LEC indicates the type of the last error to occur on the CAN bus. This field is cleared when a message has been transferred (reception or transmission) without error.

Value	Name	Description
0	NO_ERROR	No error occurred since LEC has been reset by successful reception or transmission.
1	STUFF_ERROR	More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
2	FORM_ERROR	A fixed format part of a received frame has the wrong format.
3	ACK_ERROR	The message transmitted by the MCAN was not acknowledged by another node.
4	BIT1_ERROR	During transmission of a message (with the exception of the arbitration field), the device tried to send a recessive level (bit of logical value ‘1’), but the monitored bus value was dominant.
5	BIT0_ERROR	During transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device tried to send a dominant level (data or identifier bit logical value ‘0’), but the monitored bus value was recessive. During Bus_Off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the processor to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed).
6	CRC_ERROR	The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match the CRC calculated from the received data.
7	NO_CHANGE	Any read access to the Protocol Status Register re-initializes the LEC to ‘7’. When the LEC shows value ‘7’, no CAN bus event was detected since the last processor read access to the Protocol Status Register.

39.6.14 MCAN Transmitter Delay Compensation Register

Name: MCAN_TDCR
Offset: 0x48
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
		TDCO[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		TDCF[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bits 14:8 – TDCO[6:0] Transmitter Delay Compensation Offset
0 to 127: Offset value, in CAN core clock periods, defining the distance between the measured delay from CANTX to CANRX and the secondary sample point.

Bits 6:0 – TDCF[6:0] Transmitter Delay Compensation Filter
0 to 127: defines the minimum value for the SSP position, in CAN core clock periods. Dominant edges on CANRX that would result in an earlier SSP position are ignored for transmitter delay measurement. The feature is enabled when TDCF is configured to a value greater than TDCO.

39.6.15 MCAN Interrupt Register

Name: MCAN_IR
Offset: 0x50
Reset: 0x00000000
Property: Read/Write

The flags are set when one of the listed conditions is detected (edge-sensitive). The flags remain set until the processor clears them. A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. A hard reset will clear the register. The configuration of IE controls whether an interrupt is generated. The configuration of ILS controls on which interrupt line an interrupt is signalled.

Bit	31	30	29	28	27	26	25	24
			ARA	PED	PEA	WDI	BO	EW
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EP	ELO			DRX	TOO	MRAF	TSW
Access	R/W	R/W			R/W	R/W	R/W	R/W
Reset	0	0			0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 29 – ARA Access to Reserved Address

Value	Description
0	No access to reserved address occurred
1	Access to reserved address occurred

Bit 28 – PED Protocol Error in Data Phase

Value	Description
0	No protocol error in data phase
1	Protocol error in data phase detected (MCAN_PSR.DLEC differs from 0 or 7)

Bit 27 – PEA Protocol Error in Arbitration Phase

Value	Description
0	No protocol error in arbitration phase
1	Protocol error in arbitration phase detected (MCAN_PSR.LEC differs from 0 or 7)

Bit 26 – WDI Watchdog Interrupt

Value	Description
0	No Message RAM Watchdog event occurred.
1	Message RAM Watchdog event due to missing READY.

Bit 25 – BO Bus_Off Status

Value	Description
0	Bus_Off status unchanged.
1	Bus_Off status changed.

Bit 24 – EW Warning Status

Value	Description
0	Error_Warning status unchanged.
1	Error_Warning status changed.

Bit 23 – EP Error Passive

Value	Description
0	Error_Passive status unchanged.
1	Error_Passive status changed.

Bit 22 – ELO Error Logging Overflow

Value	Description
0	CAN Error Logging Counter did not overflow.
1	Overflow of CAN Error Logging Counter occurred.

Bit 19 – DRX Message stored to Dedicated Receive Buffer

The flag is set whenever a received message has been stored into a dedicated Receive Buffer.

Value	Description
0	No Receive Buffer updated.
1	At least one received message stored into a Receive Buffer.

Bit 18 – TOO Timeout Occurred

Value	Description
0	No timeout.
1	Timeout reached.

Bit 17 – MRAF Message RAM Access Failure

The flag is set, when the Rx Handler

• has not completed acceptance filtering or storage of an accepted message until the arbitration field of the following message has been received. In this case acceptance filtering or message storage is aborted and the Rx Handler starts processing of the following message.

• was not able to write a message to the Message RAM. In this case message storage is aborted.

In both cases the FIFO put index is not updated resp. the New Data flag for a dedicated Receive Buffer is not set, a partly stored message is overwritten when the next message is stored to this location.

The flag is also set when the Tx Handler was not able to read a message from the Message RAM in time. In this case message transmission is aborted. In case of a Tx Handler access failure the MCAN is switched into Restricted Operation mode (see Restricted Operation Mode). To leave Restricted Operation mode, the processor has to reset MCAN_CCCR.ASM.

Value	Description
0	No Message RAM access failure occurred.
1	Message RAM access failure occurred.

Bit 16 – TSW Timestamp Wraparound

Value	Description
0	No timestamp counter wrap-around.
1	Timestamp counter wrapped around.

Bit 15 – TEFL Tx Event FIFO Element Lost

Value	Description
0	No Tx Event FIFO element lost.
1	Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero.

Bit 14 – TEFF Tx Event FIFO Full

Value	Description
0	Tx Event FIFO not full.
1	Tx Event FIFO full.

Bit 13 – TEFW Tx Event FIFO Watermark Reached

Value	Description
0	Tx Event FIFO fill level below watermark.
1	Tx Event FIFO fill level reached watermark.

Bit 12 – TEFN Tx Event FIFO New Entry

Value	Description
0	Tx Event FIFO unchanged.
1	Tx Handler wrote Tx Event FIFO element.

Bit 11 – TFE Tx FIFO Empty

Value	Description
0	Tx FIFO non-empty.
1	Tx FIFO empty.

Bit 10 – TCF Transmission Cancellation Finished

Value	Description
0	No transmission cancellation finished.
1	Transmission cancellation finished.

Bit 9 – TC Transmission Completed

Value	Description
0	No transmission completed.
1	Transmission completed.

Bit 8 – HPM High Priority Message

Value	Description
0	No high priority message received.
1	High priority message received.

Bit 7 – RF1L Receive FIFO 1 Message Lost

Value	Description
0	No Receive FIFO 1 message lost.
1	Receive FIFO 1 message lost, also set after write attempt to Receive FIFO 1 of size zero.

Bit 6 – RF1F Receive FIFO 1 Full

Value	Description
0	Receive FIFO 1 not full.
1	Receive FIFO 1 full.

Bit 5 – RF1W Receive FIFO 1 Watermark Reached

Value	Description
0	Receive FIFO 1 fill level below watermark.
1	Receive FIFO 1 fill level reached watermark.

Bit 4 – RF1N Receive FIFO 1 New Message

Value	Description
0	No new message written to Receive FIFO 1.
1	New message written to Receive FIFO 1.

Bit 3 – RF0L Receive FIFO 0 Message Lost

Value	Description
0	No Receive FIFO 0 message lost.
1	Receive FIFO 0 message lost, also set after write attempt to Receive FIFO 0 of size zero.

Bit 2 – RF0F Receive FIFO 0 Full

Value	Description
0	Receive FIFO 0 not full.
1	Receive FIFO 0 full.

Bit 1 – RF0W Receive FIFO 0 Watermark Reached

Value	Description
0	Receive FIFO 0 fill level below watermark.
1	Receive FIFO 0 fill level reached watermark.

Bit 0 – RF0N Receive FIFO 0 New Message

Value	Description
0	No new message written to Receive FIFO 0.
1	New message written to Receive FIFO 0.

39.6.16 MCAN Interrupt Enable Register

Name: MCAN_IE
Offset: 0x54
Reset: 0x00000000
Property: Read/Write

The following configuration values are valid for all listed bit names of this register:

0: Disables the corresponding interrupt.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
			ARAE	PEDE	PEAE	WDIE	BOE	EWE
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EPE	ELOE			DRXE	TOOE	MRAFE	TSWE
Access	R/W	R/W			R/W	R/W	R/W	R/W
Reset	0	0			0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 29 – ARAE Access to Reserved Address Enable

Bit 28 – PEDE Protocol Error in Data Phase Enable

Bit 27 – PEAE Protocol Error in Arbitration Phase Enable

Bit 26 – WDIE Watchdog Interrupt Enable

Bit 25 – BOE Bus_Off Status Interrupt Enable

Bit 24 – EWE Warning Status Interrupt Enable

Bit 23 – EPE Error Passive Interrupt Enable

Bit 22 – ELOE Error Logging Overflow Interrupt Enable

Bit 19 – DRXE Message stored to Dedicated Receive Buffer Interrupt Enable

Bit 18 – TOOE Timeout Occurred Interrupt Enable

Bit 17 – MRAFE Message RAM Access Failure Interrupt Enable

Bit 16 – TSWE Timestamp Wraparound Interrupt Enable

Bit 15 – TEFLE Tx Event FIFO Event Lost Interrupt Enable

Bit 14 – TEFFE Tx Event FIFO Full Interrupt Enable

Bit 13 – TEFWE Tx Event FIFO Watermark Reached Interrupt Enable

Bit 12 – TEFNE Tx Event FIFO New Entry Interrupt Enable

Bit 11 – TFEE Tx FIFO Empty Interrupt Enable

Bit 10 – TCFE Transmission Cancellation Finished Interrupt Enable

Bit 9 – TCE Transmission Completed Interrupt Enable

Bit 8 – HPME High Priority Message Interrupt Enable

Bit 7 – RF1LE Receive FIFO 1 Message Lost Interrupt Enable

Bit 6 – RF1FE Receive FIFO 1 Full Interrupt Enable

Bit 5 – RF1WE Receive FIFO 1 Watermark Reached Interrupt Enable

Bit 4 – RF1NE Receive FIFO 1 New Message Interrupt Enable

Bit 3 – RF0LE Receive FIFO 0 Message Lost Interrupt Enable

Bit 2 – RF0FE Receive FIFO 0 Full Interrupt Enable

Bit 1 – RF0WE Receive FIFO 0 Watermark Reached Interrupt Enable

Bit 0 – RF0NE Receive FIFO 0 New Message Interrupt Enable

39.6.17 MCAN Interrupt Line Select Register

Name: MCAN_ILS
Offset: 0x58
Reset: 0x00000000
Property: Read/Write

The Interrupt Line Select register assigns an interrupt generated by a specific interrupt flag from the Interrupt Register to one of the two module interrupt lines.

0: Interrupt assigned to interrupt line MCAN_INT0.

1: Interrupt assigned to interrupt line MCAN_INT1.

Bit	31	30	29	28	27	26	25	24
			ARAL	PEDL	PEAL	WDIL	BOL	EWL
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	EPL	ELOL			DRXL	TOOL	MRAFL	TSWL
Access	R/W	R/W			R/W	R/W	R/W	R/W
Reset	0	0			0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 29 – ARAL Access to Reserved Address Line

Bit 28 – PEDL Protocol Error in Data Phase Line

Bit 27 – PEAL Protocol Error in Arbitration Phase Line

Bit 26 – WDIL Watchdog Interrupt Line

Bit 25 – BOL Bus_Off Status Interrupt Line

Bit 24 – EWL Warning Status Interrupt Line

Bit 23 – EPL Error Passive Interrupt Line

Bit 22 – ELOL Error Logging Overflow Interrupt Line

Bit 19 – DRXL Message stored to Dedicated Receive Buffer Interrupt Line

Bit 18 – TOOL Timeout Occurred Interrupt Line

Bit 17 – MRAFL Message RAM Access Failure Interrupt Line

Bit 16 – TSWL Timestamp Wraparound Interrupt Line

Bit 15 – TEFLL Tx Event FIFO Event Lost Interrupt Line

Bit 14 – TEFFL Tx Event FIFO Full Interrupt Line

Bit 13 – TEFWL Tx Event FIFO Watermark Reached Interrupt Line

Bit 12 – TEFNL Tx Event FIFO New Entry Interrupt Line

Bit 11 – TFEL Tx FIFO Empty Interrupt Line

Bit 10 – TCFL Transmission Cancellation Finished Interrupt Line

Bit 9 – TCL Transmission Completed Interrupt Line

Bit 8 – HPML High Priority Message Interrupt Line

Bit 7 – RF1LL Receive FIFO 1 Message Lost Interrupt Line

Bit 6 – RF1FL Receive FIFO 1 Full Interrupt Line

Bit 5 – RF1WL Receive FIFO 1 Watermark Reached Interrupt Line

Bit 4 – RF1NL Receive FIFO 1 New Message Interrupt Line

Bit 3 – RF0LL Receive FIFO 0 Message Lost Interrupt Line

Bit 2 – RF0FL Receive FIFO 0 Full Interrupt Line

Bit 1 – RF0WL Receive FIFO 0 Watermark Reached Interrupt Line

Bit 0 – RF0NL Receive FIFO 0 New Message Interrupt Line

39.6.18 MCAN Interrupt Line Enable

Name: MCAN_ILE
Offset: 0x5C
Reset: 0x00000000
Property: Read/Write

Each of the two interrupt lines to the processor can be enabled/disabled separately by programming bits EINT0 and EINT1.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							EINT1	EINT0
Access							R/W	R/W
Reset							0	0

Bit 1 – EINT1 Enable Interrupt Line 1

Value	Description
0	Interrupt line MCAN_INT1 disabled.
1	Interrupt line MCAN_INT1 enabled.

Bit 0 – EINT0 Enable Interrupt Line 0

Value	Description
0	Interrupt line MCAN_INT0 disabled.
1	Interrupt line MCAN_INT0 enabled.

39.6.19 MCAN Global Filter Configuration

Name: MCAN_GFC
Offset: 0x80
Reset: 0x00000000
Property: Read/Write

Global settings for Message ID filtering. The Global Filter Configuration controls the filter path for standard and extended messages as illustrated in Standard Message ID Filter Path and Extended Message ID Filter Path.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			ANFS[1:0]		ANFE[1:0]		RRFS	RRFE
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bits 5:4 – ANFS[1:0] Accept Non-matching Frames Standard
Defines how received messages with 11-bit IDs that do not match any element of the filter list are treated.

Value	Name	Description
0	RX_FIFO_0	Accept in Rx FIFO 0
1	RX_FIFO_1	Accept in Rx FIFO 1
2-3	REJECTED	Message rejected

Bits 3:2 – ANFE[1:0] Accept Non-matching Frames Extended
Defines how received messages with 29-bit IDs that do not match any element of the filter list are treated.

Value	Name	Description
0	RX_FIFO_0	Accept in Rx FIFO 0
1	RX_FIFO_1	Accept in Rx FIFO 1
2-3	REJECTED	Message rejected

Bit 1 – RRFS Reject Remote Frames Standard
0 (FILTER): Filter remote frames with 11-bit standard IDs.
1 (REJECT): Reject all remote frames with 11-bit standard IDs.

Bit 0 – RRFE Reject Remote Frames Extended
0 (FILTER): Filter remote frames with 29-bit extended IDs.
1 (REJECT): Reject all remote frames with 29-bit extended IDs.

39.6.20 MCAN Standard ID Filter Configuration

Name: MCAN_SIDFC
Offset: 0x84
Reset: 0x00000000
Property: Read/Write

Settings for 11-bit standard Message ID filtering. The Standard ID Filter Configuration controls the filter path for standard messages as illustrated in Standard Message ID Filter Path.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	LSS[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FLSSA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FLSSA[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bits 23:16 – LSS[7:0] List Size Standard
>128: Values greater than 128 are interpreted as 128.

Value	Description
0	No standard Message ID filter.
1-128	Number of standard Message ID filter elements.

Bits 15:2 – FLSSA[13:0] Filter List Standard Start Address
Start address of standard Message ID filter list (32-bit word address, see Message RAM Configuration).
Write FLSSA with the bits [15:2] of the 32-bit address.

39.6.21 MCAN Extended ID Filter Configuration

Name: MCAN_XIDFC
Offset: 0x88
Reset: 0x00000000
Property: Read/Write

Settings for 29-bit extended Message ID filtering. The Extended ID Filter Configuration controls the filter path for standard messages as described in Extended Message ID Filter Path.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
		LSE[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FLESA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FLESA[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bits 22:16 – LSE[6:0] List Size Extended

Value	Description
0	No extended Message ID filter.
1-64	Number of extended Message ID filter elements.
>64	Values greater than 64 are interpreted as 64.

Bits 15:2 – FLESA[13:0] Filter List Extended Start Address
Start address of extended Message ID filter list (32-bit word address, see Message RAM Configuration).
Write FLESA with the bits [15:2] of the 32-bit address.

39.6.22 MCAN Extended ID AND Mask

Name: MCAN_XIDAM
Offset: 0x90
Reset: 0x1FFFFFFF
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Bit	31	30	29	28	27	26	25	24
				EIDM[28:24]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				1	1	1	1	1

Bit	23	22	21	20	19	18	17	16
	EIDM[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8
	EIDM[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bit	7	6	5	4	3	2	1	0
	EIDM[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 28:0 – EIDM[28:0] Extended ID Mask
For acceptance filtering of extended frames the Extended ID AND Mask is ANDed with the Message ID of a received frame. Intended for masking of 29-bit IDs in SAE J1939. With the reset value of all bits set to one the mask is not active.

39.6.23 MCAN High Priority Message Status

Name: MCAN_HPMS
Offset: 0x94
Reset: 0x00000000
Property: Read-only

This register is updated every time a Message ID filter element configured to generate a priority event matches. This can be used to monitor the status of incoming high priority messages and to enable fast access to these messages.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	FLST	FIDX[6:0]						
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MSI[1:0]		BIDX[5:0]					
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 15 – FLST Filter List
Indicates the filter list of the matching filter element.

Value	Description
0	Standard filter list
1	Extended filter list

Bits 14:8 – FIDX[6:0] Filter Index
Index of matching filter element. Range is 0 to MCAN_SIDFC.LSS - 1 resp. MCAN_XIDFC.LSE - 1.

Bits 7:6 – MSI[1:0] Message Storage Indicator

Value	Name	Description
0	NO_FIFO_SEL	No FIFO selected.
1	LOST	FIFO message lost.
2	FIFO_0	Message stored in FIFO 0.
3	FIFO_1	Message stored in FIFO 1.

Bits 5:0 – BIDX[5:0] Buffer Index
Index of Receive FIFO element to which the message was stored. Only valid when MSI[1] = ‘1’.

39.6.24 MCAN New Data 1

Name: MCAN_NDAT1
Offset: 0x98
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – NDx New Data
The register holds the New Data flags of Receive Buffers 0 to 31. The flags are set when the respective Receive Buffer has been updated from a received frame. The flags remain set until the processor clears them. A flag is cleared by writing a ‘1’ to the corresponding bit position. Writing a ‘0’ has no effect. A hard reset will clear the register.

Value	Description
0	Receive Buffer not updated
1	Receive Buffer updated from new message

39.6.25 MCAN New Data 2

Name: MCAN_NDAT2
Offset: 0x9C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – NDx New Data
The register holds the New Data flags of Receive Buffers 32 to 63. The flags are set when the respective Receive Buffer has been updated from a received frame. The flags remain set until the processor clears them. A flag is cleared by writing a ‘1’ to the corresponding bit position. Writing a ‘0’ has no effect. A hard reset will clear the register.

Value	Description
0	Receive Buffer not updated.
1	Receive Buffer updated from new message.

39.6.26 MCAN Receive FIFO 0 Configuration

Name: MCAN_RXF0C
Offset: 0xA0
Reset: 0x00000000
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Bit	31	30	29	28	27	26	25	24
	F0OM	F0WM[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
		F0S[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	F0SA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	F0SA[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bit 31 – F0OM FIFO 0 Operation Mode
FIFO 0 can be operated in Blocking or in Overwrite mode (see Rx FIFOs).

Value	Description
0	FIFO 0 Blocking mode.
1	FIFO 0 Overwrite mode.

Bits 30:24 – F0WM[6:0] Receive FIFO 0 Watermark

Value	Description
0	Watermark interrupt disabled.
1-64	Level for Receive FIFO 0 watermark interrupt (MCAN_IR.RF0W).
>64	Watermark interrupt disabled.

Bits 22:16 – F0S[6:0] Receive FIFO 0 Size
The Receive FIFO 0 elements are indexed from 0 to F0S-1.

Value	Description
0	No Receive FIFO 0
1-64	Number of Receive FIFO 0 elements.
>64	Values greater than 64 are interpreted as 64.

Bits 15:2 – F0SA[13:0] Receive FIFO 0 Start Address
Start address of Receive FIFO 0 in Message RAM (32-bit word address, see Message RAM Configuration).
Write F0SA with the bits [15:2] of the 32-bit address.

39.6.27 MCAN Receive FIFO 0 Status

Name: MCAN_RXF0S
Offset: 0xA4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
							RF0L	F0F
Access							R	R
Reset							0	0

Bit	23	22	21	20	19	18	17	16
			F0PI[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
			F0GI[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		F0FL[6:0]						
Access		R	R	R	R	R	R	R
Reset		0	0	0	0	0	0	0

Bit 25 – RF0L Receive FIFO 0 Message Lost
This bit is a copy of interrupt flag MCAN_IR.RF0L. When MCAN_IR.RF0L is reset, this bit is also reset.
Overwriting the oldest message when MCAN_RXF0C.F0OM = '1' will not set this flag.

Value	Description
0	No Receive FIFO 0 message lost
1	Receive FIFO 0 message lost, also set after write attempt to Receive FIFO 0 of size zero

Bit 24 – F0F Receive FIFO 0 Full

Value	Description
0	Receive FIFO 0 not full.
1	Receive FIFO 0 full.

Bits 21:16 – F0PI[5:0] Receive FIFO 0 Put Index
Receive FIFO 0 write index pointer, range 0 to 63.

Bits 13:8 – F0GI[5:0] Receive FIFO 0 Get Index
Receive FIFO 0 read index pointer, range 0 to 63.

Bits 6:0 – F0FL[6:0] Receive FIFO 0 Fill Level
Number of elements stored in Receive FIFO 0, range 0 to 64.

39.6.28 MCAN Receive FIFO 0 Acknowledge

Name: MCAN_RXF0A
Offset: 0xA8
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			F0AI[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bits 5:0 – F0AI[5:0] Receive FIFO 0 Acknowledge Index
After the processor has read a message or a sequence of messages from Receive FIFO 0 it has to write the buffer index of the last element read from Receive FIFO 0 to F0AI. This will set the Receive FIFO 0 Get Index MCAN_RXF0S.F0GI to F0AI + 1 and update the FIFO 0 Fill Level MCAN_RXF0S.F0FL.

39.6.29 MCAN Receive Buffer Configuration

Name: MCAN_RXBC
Offset: 0xAC
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RBSA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RBSA[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bits 15:2 – RBSA[13:0] Receive Buffer Start Address
Configures the start address of the Receive Buffers section in the Message RAM (32-bit word address, see Message RAM Configuration). Also used to reference debug messages A,B,C.
Write RBSA with the bits [15:2] of the 32-bit address.

39.6.30 MCAN Receive FIFO 1 Configuration

Name: MCAN_RXF1C
Offset: 0xB0
Reset: 0x00000000
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Bit	31	30	29	28	27	26	25	24
	F1OM	F1WM[6:0]						
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
		F1S[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	F1SA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	F1SA[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bit 31 – F1OM FIFO 1 Operation Mode
FIFO 1 can be operated in Blocking or in Overwrite mode (see Rx FIFOs).

Value	Description
0	FIFO 1 Blocking mode.
1	FIFO 1 Overwrite mode.

Bits 30:24 – F1WM[6:0] Receive FIFO 1 Watermark

Value	Description
0	Watermark interrupt disabled
1-64	Level for Receive FIFO 1 watermark interrupt (MCAN_IR.RF1W).
>64	Watermark interrupt disabled.

Bits 22:16 – F1S[6:0] Receive FIFO 1 Size
The elements in Receive FIFO 1 are indexed from 0 to F1S - 1.

Value	Description
0	No Receive FIFO 1
1-64	Number of elements in Receive FIFO 1.
>64	Values greater than 64 are interpreted as 64.

Bits 15:2 – F1SA[13:0] Receive FIFO 1 Start Address
Start address of Receive FIFO 1 in Message RAM (32-bit word address, see Message RAM Configuration).
Write F1SA with the bits [15:2] of the 32-bit address.

39.6.31 MCAN Receive FIFO 1 Status

Name: MCAN_RXF1S
Offset: 0xB4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	DMS[1:0]						RF1L	F1F
Access	R	R					R	R
Reset	0	0					0	0

Bit	23	22	21	20	19	18	17	16
			F1PI[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
			F1GI[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
		F1FL[6:0]						
Access		R	R	R	R	R	R	R
Reset		0	0	0	0	0	0	0

Bits 31:30 – DMS[1:0] Debug Message Status

Value	Name	Description
0	IDLE	Idle state, wait for reception of debug messages, DMA request is cleared.
1	MSG_A	Debug message A received.
2	MSG_AB	Debug messages A, B received.
3	MSG_ABC	Debug messages A, B, C received, DMA request is set.

Bit 25 – RF1L Receive FIFO 1 Message Lost
This bit is a copy of interrupt flag IR.RF1L. When IR.RF1L is reset, this bit is also reset.
Overwriting the oldest message when MCAN_RXF1C.F1OM = '1' will not set this flag.

Value	Description
0	No Receive FIFO 1 message lost.
1	Receive FIFO 1 message lost, also set after write attempt to Receive FIFO 1 of size zero.

Bit 24 – F1F Receive FIFO 1 Full

Value	Description
0	Receive FIFO 1 not full.
1	Receive FIFO 1 full.

Bits 21:16 – F1PI[5:0] Receive FIFO 1 Put Index
Receive FIFO 1 write index pointer, range 0 to 63.

Bits 13:8 – F1GI[5:0] Receive FIFO 1 Get Index
Receive FIFO 1 read index pointer, range 0 to 63.

Bits 6:0 – F1FL[6:0] Receive FIFO 1 Fill Level
Number of elements stored in Receive FIFO 1, range 0 to 64.

39.6.32 MCAN Receive FIFO 1 Acknowledge

Name: MCAN_RXF1A
Offset: 0xB8
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
			F1AI[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bits 5:0 – F1AI[5:0] Receive FIFO 1 Acknowledge Index
After the processor has read a message or a sequence of messages from Receive FIFO 1 it has to write the buffer index of the last element read from Receive FIFO 1 to F1AI. This will set the Receive FIFO 1 Get Index MCAN_RXF1S.F1GI to F1AI + 1 and update the FIFO 1 Fill Level MCAN_RXF1S.F1FL.

39.6.33 MCAN Receive Buffer / FIFO Element Size Configuration

Name: MCAN_RXESC
Offset: 0xBC
Reset: 0x00000000
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Configures the number of data bytes belonging to a Receive Buffer / Receive FIFO element. Data field sizes >8 bytes are intended for CAN FD operation only.

In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Receive Buffer or Receive FIFO, only the number of bytes as configured by MCAN_RXESC are stored to the Receive Buffer resp. Receive FIFO element. The rest of the frame's data field is ignored.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
						RBDS[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bit	7	6	5	4	3	2	1	0
		F1DS[2:0]				F0DS[2:0]		
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset		0	0	0		0	0	0

Bits 10:8 – RBDS[2:0] Receive Buffer Data Field Size

Value	Name	Description
0	8_BYTE	8-byte data field
1	12_BYTE	12-byte data field
2	16_BYTE	16-byte data field
3	20_BYTE	20-byte data field
4	24_BYTE	24-byte data field
5	32_BYTE	32-byte data field
6	48_BYTE	48-byte data field
7	64_BYTE	64-byte data field

Bits 6:4 – F1DS[2:0] Receive FIFO 1 Data Field Size

Value	Name	Description
0	8_BYTE	8-byte data field
1	12_BYTE	12-byte data field
2	16_BYTE	16-byte data field
3	20_BYTE	20-byte data field
4	24_BYTE	24-byte data field
5	32_BYTE	32-byte data field
6	48_BYTE	48-byte data field
7	64_BYTE	64-byte data field

Bits 2:0 – F0DS[2:0] Receive FIFO 0 Data Field Size

Value	Name	Description
0	8_BYTE	8-byte data field
1	12_BYTE	12-byte data field
2	16_BYTE	16-byte data field
3	20_BYTE	20-byte data field
4	24_BYTE	24-byte data field
5	32_BYTE	32-byte data field
6	48_BYTE	48-byte data field
7	64_BYTE	64-byte data field

39.6.34 MCAN Tx Buffer Configuration

Name: MCAN_TXBC
Offset: 0xC0
Reset: 0x00000000
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

The sum of TFQS and NDTB may not exceed 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers.

Bit	31	30	29	28	27	26	25	24
		TFQM	TFQS[5:0]					
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			NDTB[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TBSA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TBSA[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bit 30 – TFQM Tx FIFO/Queue Mode

Value	Description
0	Tx FIFO operation.
1	Tx Queue operation.

Bits 29:24 – TFQS[5:0] Transmit FIFO/Queue Size

Value	Description
0	No Tx FIFO/Queue.
1-32	Number of Tx Buffers used for Tx FIFO/Queue.
>32	Values greater than 32 are interpreted as 32.

Bits 21:16 – NDTB[5:0] Number of Dedicated Transmit Buffers

Value	Description
0	No dedicated Tx Buffers.
1-32	Number of dedicated Tx Buffers.
>32	Values greater than 32 are interpreted as 32.

Bits 15:2 – TBSA[13:0] Tx Buffers Start Address
Start address of Tx Buffers section in Message RAM (32-bit word address, see Message RAM Configuration).
Write TBSA with the bits [15:2] of the 32-bit address.

39.6.35 MCAN Tx FIFO/Queue Status

Name: MCAN_TXFQS
Offset: 0xC4
Reset: 0x00000000
Property: Read-only

The Tx FIFO/Queue status is related to the pending Tx requests listed in register MCAN_TXBRP. Therefore the effect of Add/Cancellation requests may be delayed due to a running Tx scan (MCAN_TXBRP not yet updated).

In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers.
Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
			TFQF	TFQPI[4:0]				
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				TFGI[4:0]				
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
			TFFL[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 21 – TFQF Tx FIFO/Queue Full

Value	Description
0	Tx FIFO/Queue not full.
1	Tx FIFO/Queue full.

Bits 20:16 – TFQPI[4:0] Tx FIFO/Queue Put Index
Tx FIFO/Queue write index pointer, range 0 to 31.

Bits 12:8 – TFGI[4:0] Tx FIFO Get Index
Tx FIFO read index pointer, range 0 to 31. Read as zero when Tx Queue operation is configured (MCAN_TXBC.TFQM = ‘1’).

Bits 5:0 – TFFL[5:0] Tx FIFO Free Level
Number of consecutive free Tx FIFO elements starting from TFGI, range 0 to 32. Read as zero when Tx Queue operation is configured (MCAN_TXBC.TFQM = ‘1’).

39.6.36 MCAN Tx Buffer Element Size Configuration

Name: MCAN_TXESC
Offset: 0xC8
Reset: 0x00000000
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Configures the number of data bytes belonging to a Tx Buffer element. Data field sizes > 8 bytes are intended for CAN FD operation only.

In case the data length code DLC of a Tx Buffer element is configured to a value higher than the Tx Buffer data field size MCAN_TXESC.TBDS, the bytes not defined by the Tx Buffer are transmitted as “0xCC” (padding bytes).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						TBDS[2:0]		
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 2:0 – TBDS[2:0] Tx Buffer Data Field Size

Value	Name	Description
0	8_BYTE	8-byte data field
1	12_BYTE	12-byte data field
2	16_BYTE	16-byte data field
3	20_BYTE	20-byte data field
4	24_BYTE	24-byte data field
5	32_BYTE	32-byte data field
6	48_BYTE	48- byte data field
7	64_BYTE	64-byte data field

39.6.37 MCAN Transmit Buffer Request Pending

Name: MCAN_TXBRP
Offset: 0xCC
Reset: 0x00000000
Property: Read-only

MCAN_TXBRP bits which are set while a Tx scan is in progress are not considered during this particular Tx scan. In case a cancellation is requested for such a Tx Buffer, this Add Request is cancelled immediately, the corresponding MCAN_TXBRP bit is reset.

Bit	31	30	29	28	27	26	25	24
	TRP31	TRP30	TRP29	TRP28	TRP27	TRP26	TRP25	TRP24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TRP23	TRP22	TRP21	TRP20	TRP19	TRP18	TRP17	TRP16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TRP15	TRP14	TRP13	TRP12	TRP11	TRP10	TRP9	TRP8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TRP7	TRP6	TRP5	TRP4	TRP3	TRP2	TRP1	TRP0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – TRPx Transmission Request Pending for Buffer x
Each Tx Buffer has its own Transmission Request Pending bit. The bits are set via register MCAN_TXBAR. The bits are reset after a requested transmission has completed or has been cancelled via register MCAN_TXBCR.
TXBRP bits are set only for those Tx Buffers configured via MCAN_TXBC. After a MCAN_TXBRP bit has been set, a Tx scan (see Tx Handling) is started to check for the pending Tx request with the highest priority (Tx Buffer with lowest Message ID).
A cancellation request resets the corresponding transmission request pending bit of register MCAN_TXBRP. In case a transmission has already been started when a cancellation is requested, this is done at the end of the transmission, regardless whether the transmission was successful or not. The cancellation request bits are reset directly after the corresponding TXBRP bit has been reset.
After a cancellation has been requested, a finished cancellation is signalled via MCAN_TXBCF.
• after successful transmission together with the corresponding MCAN_TXBTO bit.
• when the transmission has not yet been started at the point of cancellation.
• when the transmission has been aborted due to lost arbitration.
• when an error occurred during frame transmission.
In DAR mode, all transmissions are automatically cancelled if they are not successful. The corresponding MCAN_TXBCF bit is set for all unsuccessful transmissions.

Value	Description
0	No transmission request pending
1	Transmission request pending

39.6.38 MCAN Transmit Buffer Add Request

Name: MCAN_TXBAR
Offset: 0xD0
Reset: 0x00000000
Property: Read/Write

If an add request is applied for a Transmit Buffer with pending transmission request (corresponding MCAN_TXBRP bit already set), this Add Request is ignored.

Bit	31	30	29	28	27	26	25	24
	AR31	AR30	AR29	AR28	AR27	AR26	AR25	AR24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	AR23	AR22	AR21	AR20	AR19	AR18	AR17	AR16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	AR15	AR14	AR13	AR12	AR11	AR10	AR9	AR8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – ARx Add Request for Transmit Buffer x
Each Transmit Buffer has its own Add Request bit. Writing a '1' will set the corresponding Add Request bit; writing a '0' has no impact. This enables the processor to set transmission requests for multiple Transmit Buffers with one write to MCAN_TXBAR. MCAN_TXBAR bits are set only for those Transmit Buffers configured via TXBC. When no Transmit scan is running, the bits are reset immediately, else the bits remain set until the Transmit scan process has completed.

Value	Description
0	No transmission request added.
1	Transmission requested added.

39.6.39 MCAN Transmit Buffer Cancellation Request

Name: MCAN_TXBCR
Offset: 0xD4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	CR31	CR30	CR29	CR28	CR27	CR26	CR25	CR24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CR23	CR22	CR21	CR20	CR19	CR18	CR17	CR16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – CRx Cancellation Request for Transmit Buffer x
Each Transmit Buffer has its own Cancellation Request bit. Writing a ‘1’ will set the corresponding Cancellation Request bit; writing a ‘0’ has no impact. This enables the processor to set cancellation requests for multiple Transmit Buffers with one write to MCAN_TXBCR. MCAN_TXBCR bits are set only for those Transmit Buffers configured via TXBC. The bits remain set until the corresponding bit of MCAN_TXBRP is reset.

Value	Description
0	No cancellation pending.
1	Cancellation pending.

39.6.40 MCAN Transmit Buffer Transmission Occurred

Name: MCAN_TXBTO
Offset: 0xD8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	TO31	TO30	TO29	TO28	TO27	TO26	TO25	TO24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TO23	TO22	TO21	TO20	TO19	TO18	TO17	TO16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TO15	TO14	TO13	TO12	TO11	TO10	TO9	TO8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TO7	TO6	TO5	TO4	TO3	TO2	TO1	TO0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – TOx Transmission Occurred for Buffer x
Each Transmit Buffer has its own Transmission Occurred bit. The bits are set when the corresponding MCAN_TXBRP bit is cleared after a successful transmission. The bits are reset when a new transmission is requested by writing a ‘1’ to the corresponding bit of register MCAN_TXBAR.

Value	Description
0	No transmission occurred.
1	Transmission occurred.

39.6.41 MCAN Transmit Buffer Cancellation Finished

Name: MCAN_TXBCF
Offset: 0xDC
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 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 – CFx Cancellation Finished for Transmit Buffer x
Each Transmit Buffer has its own Cancellation Finished bit. The bits are set when the corresponding MCAN_TXBRP bit is cleared after a cancellation was requested via MCAN_TXBCR. In case the corresponding MCAN_TXBRP bit was not set at the point of cancellation, CF is set immediately. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register MCAN_TXBAR.

Value	Description
0	No transmit buffer cancellation.
1	Transmit buffer cancellation finished.

39.6.42 MCAN Transmit Buffer Transmission Interrupt Enable

Name: MCAN_TXBTIE
Offset: 0xE0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	TIE31	TIE30	TIE29	TIE28	TIE27	TIE26	TIE25	TIE24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	TIE23	TIE22	TIE21	TIE20	TIE19	TIE18	TIE17	TIE16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – TIEx Transmission Interrupt Enable for Buffer x
Each Transmit Buffer has its own Transmission Interrupt Enable bit.

Value	Description
0	Transmission interrupt disabled
1	Transmission interrupt enable

39.6.43 MCAN Transmit Buffer Cancellation Finished Interrupt Enable

Name: MCAN_TXBCIE
Offset: 0xE4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	CFIE31	CFIE30	CFIE29	CFIE28	CFIE27	CFIE26	CFIE25	CFIE24
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CFIE23	CFIE22	CFIE21	CFIE20	CFIE19	CFIE18	CFIE17	CFIE16
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CFIE15	CFIE14	CFIE13	CFIE12	CFIE11	CFIE10	CFIE9	CFIE8
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CFIE7	CFIE6	CFIE5	CFIE4	CFIE3	CFIE2	CFIE1	CFIE0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 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 – CFIEx Cancellation Finished Interrupt Enable for Transmit Buffer x
Each Transmit Buffer has its own Cancellation Finished Interrupt Enable bit.

Value	Description
0	Cancellation finished interrupt disabled.
1	Cancellation finished interrupt enabled.

39.6.44 MCAN Transmit Event FIFO Configuration

Name: MCAN_TXEFC
Offset: 0xF0
Reset: 0x00000000
Property: Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Bit	31	30	29	28	27	26	25	24
			EFWM[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
			EFS[5:0]					
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	EFSA[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	EFSA[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset	0	0	0	0	0	0		

Bits 29:24 – EFWM[5:0] Event FIFO Watermark

Value	Description
0	Watermark interrupt disabled.
1-32	Level for Tx Event FIFO watermark interrupt (MCAN_IR.TEFW).
>32	Watermark interrupt disabled.

Bits 21:16 – EFS[5:0] Event FIFO Size
The Tx Event FIFO elements are indexed from 0 to EFS - 1.

Value	Description
0	Tx Event FIFO disabled.
1-32	Number of Tx Event FIFO elements.
>32	Values greater than 32 are interpreted as 32.

Bits 15:2 – EFSA[13:0] Event FIFO Start Address
Start address of Tx Event FIFO in Message RAM (32-bit word address, see Message RAM Configuration).
Write EFSA with the bits [15:2] of the 32-bit address.

39.6.45 MCAN Tx Event FIFO Status

Name: MCAN_TXEFS
Offset: 0xF4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
							TEFL	EFF
Access							R	R
Reset							0	0

Bit	23	22	21	20	19	18	17	16
				EFPI[4:0]				
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
				EFGI[4:0]				
Access				R	R	R	R	R
Reset				0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
			EFFL[5:0]					
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit 25 – TEFL Tx Event FIFO Element Lost
This bit is a copy of interrupt flag MCAN_IR.TEFL. When MCAN_IR.TEFL is reset, this bit is also reset.

Value	Description
0	No Tx Event FIFO element lost.
1	Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero.

Bit 24 – EFF Event FIFO Full

Value	Description
0	Tx Event FIFO not full.
1	Tx Event FIFO full.

Bits 20:16 – EFPI[4:0] Event FIFO Put Index
Tx Event FIFO write index pointer, range 0 to 31.

Bits 12:8 – EFGI[4:0] Event FIFO Get Index
Tx Event FIFO read index pointer, range 0 to 31.

Bits 5:0 – EFFL[5:0] Event FIFO Fill Level
Number of elements stored in Tx Event FIFO, range 0 to 32.

39.6.46 MCAN Tx Event FIFO Acknowledge

Name: MCAN_TXEFA
Offset: 0xF8
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
				EFAI[4:0]				
Access				R/W	R/W	R/W	R/W	R/W
Reset				0	0	0	0	0

Bits 4:0 – EFAI[4:0] Event FIFO Acknowledge Index
After the processor has read an element or a sequence of elements from the Tx Event FIFO, it has to write the index of the last element read from Tx Event FIFO to EFAI. This will set the Tx Event FIFO Get Index MCAN_TXEFS.EFGI to EFAI + 1 and update the FIFO 0 Fill Level MCAN_TXEFS.EFFL.

40. Timer Counter (TC)

40.1 Description

A Timer Counter (TC) module includes three identical TC channels. The number of implemented TC modules is device-specific.

Each TC channel can be independently programmed to perform a wide range of functions including frequency measurement, event counting, interval measurement, pulse generation, delay timing and pulse width modulation.

Each channel has three external clock inputs, five internal clock inputs and two multipurpose input/output signals which can be configured by the user. Each channel drives an internal interrupt signal which can be programmed to generate processor interrupts.

The TC embeds a quadrature decoder (QDEC) connected in front of the timers and driven by TIOA0, TIOB0 and TIOB1 inputs. When enabled, the QDEC performs the input lines filtering, decoding of quadrature signals and connects to the timers/counters in order to read the position and speed of the motor through the user interface.

The TC block has the following two global registers which act upon all TC channels:

- Block Control register (TC_BCR) — Allows channels to be started simultaneously with the same instruction
- Block Mode register (TC_BMR) — Defines the external clock inputs for each channel, allowing them to be chained

40.2 Embedded Characteristics

- Total of twelve Channels
- 32-bit Channel Size
- Wide Range of Functions Including:
  - Frequency measurement
  - Event counting
  - Interval measurement
  - Pulse generation
  - Delay timing
  - Pulse Width Modulation
  - Up/down capabilities
  - Quadrature decoder
  - 2-bit Gray up/down count for stepper motor
- Each Channel is User-Configurable and Contains:
  - Three external clock inputs
  - Five Internal clock inputs
  - Two multipurpose input/output signals acting as trigger event
  - Trigger/capture events can be directly synchronized by PWM signals
- Internal Interrupt Signal
- Read of the Capture Registers by the DMAC
- Compare Event Fault Generation for PWM
- Register Write Protection

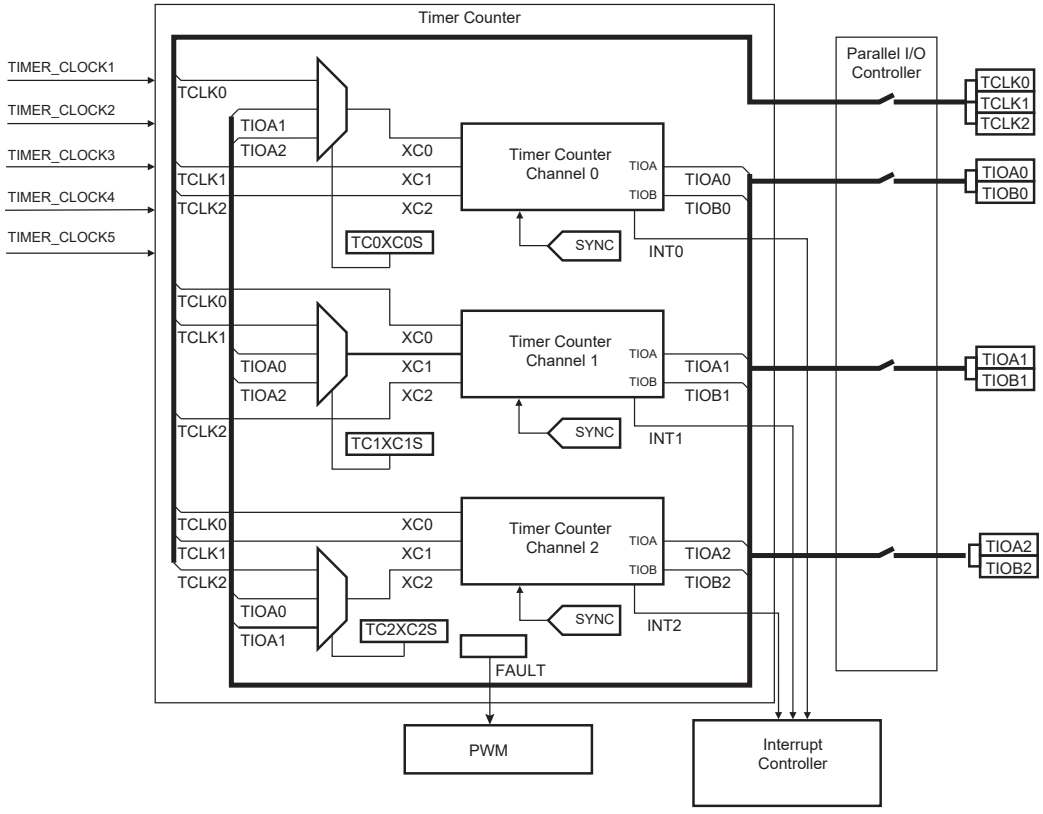
40.3 Block Diagram

Table 40-1. Timer Counter Clock Assignment

Name	Definition
TIMER_CLOCK1	GCLK
TIMER_CLOCK2	MCK/8
TIMER_CLOCK3	MCK/32
TIMER_CLOCK4	MCK/128
TIMER_CLOCK5 (1)	SLCK

1. When SLCK is selected for Peripheral Clock (CSS = 0 in PMC Master Clock register), SLCK input is equivalent to Peripheral Clock.
2. The GCLK frequency must be at least three times lower than peripheral clock frequency.

Figure 40-1. Timer Counter Block Diagram



Note:
The QDEC connections are detailed in Predefined Connection of the Quadrature Decoder with Timer Counters.

Table 40-2. Channel Signal Description

Signal Name	Description
XC0, XC1, XC2	External Clock Inputs
TIOAx	Capture Mode: Timer Counter Input Waveform Mode: Timer Counter Output

.....continued	
Signal Name	**Description**
TIOBx	Capture Mode: Timer Counter Input Waveform Mode: Timer Counter Input/Output
INT	Interrupt Signal Output (internal signal)
SYNC	Synchronization Input Signal (from configuration register)

40.4 Pin List

Table 40-3. Pin List

Pin Name	Description	Type
TCLK0–TCLK2	External Clock Input	Input
TIOA0–TIOA2	I/O Line A	I/O
TIOB0–TIOB2	I/O Line B	I/O

40.5 Product Dependencies

40.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the TC pins to their peripheral functions.

40.5.2 Power Management

The TC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the Timer Counter clock of each channel.

40.5.3 Interrupt Sources

The TC has an interrupt line per channel connected to the interrupt controller. Handling the TC interrupt requires programming the interrupt controller before configuring the TC.

40.5.4 Synchronization Inputs from PWM

The TC has trigger/capture inputs internally connected to the PWM. Refer to "Synchronization with PWM" and to the implementation of the Pulse Width Modulation (PWM) in this product.

40.5.5 Fault Output

The TC has the FAULT output internally connected to the fault input of PWM. Refer to "Fault Mode" and to the implementation of the Pulse Width Modulation (PWM) in this product.

40.6 Functional Description

40.6.1 Description

All channels of the Timer Counter are independent and identical in operation except when the QDEC is enabled. The registers for channel programming are listed in 40.7 Register Summary.

40.6.2 32-bit Counter

Each 32-bit channel is organized around a 32-bit counter. The value of the counter is incremented at each positive edge of the selected clock. When the counter has reached the value 2^{32}-1 and passes to zero, an overflow occurs and the COVFS bit in the Interrupt Status register (TC_SR) is set.

The current value of the counter is accessible in real time by reading the Counter Value register (TC_CV). The counter can be reset by a trigger. In this case, the counter value passes to zero on the next valid edge of the selected clock.

40.6.3 Clock Selection

At block level, input clock signals of each channel can be connected either to the external inputs TCLKx, or to the internal I/O signals TIOAx for chaining[1] by programming the Block Mode register (TC_BMR). See Clock Chaining Selection.

Each channel can independently select an internal or external clock source for its counter[2]:

- External clock signals: XC0, XC1 or XC2
- Internal clock signals: GCLK, MCK/8, MCK/32, MCK/128, SLCK

This selection is made by the TCCLKS bits in the Channel Mode register (TC_CMRx).

The selected clock can be inverted with TC_CMRx.CLKI. This allows counting on the opposite edges of the clock.

The burst function allows the clock to be validated when an external signal is high. The BURST parameter in the TC_CMRx defines this signal (none, XC0, XC1, XC2). See Clock Selection.

Note:

1. In Waveform mode, to chain two timers, it is mandatory to initialize some parameters:
   – Configure TIOx outputs to 1 or 0 by writing the required value to TC_CMRx.ASWTRG.
   – Bit TC_BCR.SYNC must be written to 1 to start the channels at the same time.
2. In all cases, if an external clock or asynchronous internal clock GCLK is used, the duration of each of its levels must be longer than the peripheral clock period, so the clock frequency will be at least 2.5 times lower than the peripheral clock.

Figure 40-2. Clock Chaining Selection

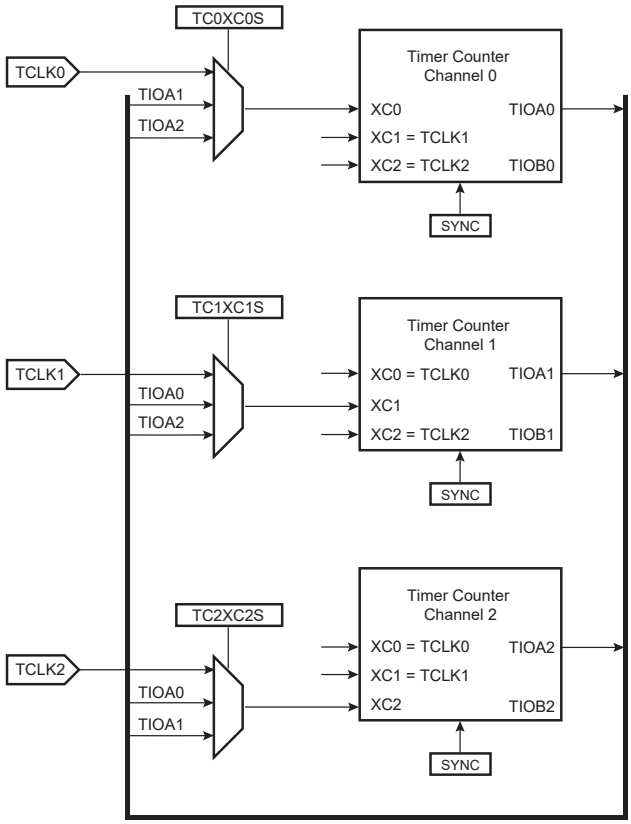
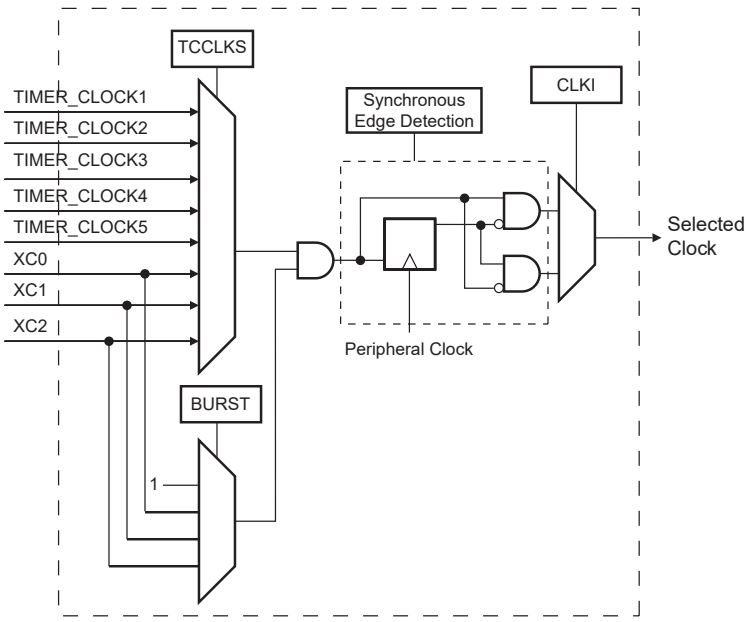


Figure 40-3. Clock Selection

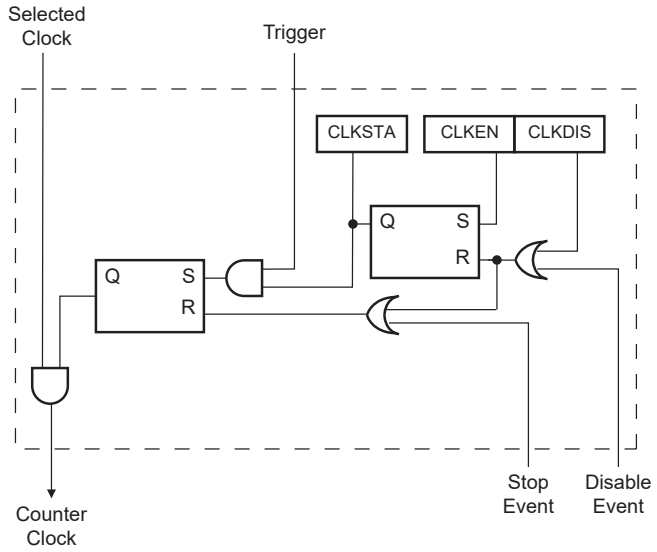


40.6.4 Clock Control

The clock of each counter can be controlled in two different ways: it can be enabled/disabled and started/stopped, as shown in the following figure.

- The clock can be enabled or disabled by the user with the CLKEN and the CLKDIS commands in the Channel Control register (TC_CCR). In Capture mode it can be disabled by an RB load event if TC_CMRx.LDBDIS is set to ‘1’. In Waveform mode, it can be disabled by an RC Compare event if TC_CMRx.CPCDIS is set to ‘1’. When disabled, the start or the stop actions have no effect: only a CLKEN command in the TC_CCR can reenable the clock. When the clock is enabled, TC_SR.CLKSTA is set.
- The clock can also be started or stopped: a trigger (software, synchro, external or compare) always starts the clock. The clock can be stopped by an RB load event in Capture mode (TC_CMRx.LDBSTOP = 1) or an RC compare event in Waveform mode (TC_CMRx.CPCSTOP = 1). The start and the stop commands are effective only if the clock is enabled.

Figure 40-4. Clock Control



40.6.5 Operating Modes

Each channel can operate independently in two different modes:

- Capture mode provides measurement on signals.
- Waveform mode provides wave generation.

The TC operating mode is programmed with TC_CMRx.WAVE.

In Capture mode, TIOAx and TIOBx are configured as inputs.

In Waveform mode, TIOAx is always configured to be an output and TIOBx is an output if it is not selected to be the external trigger.

40.6.6 Trigger

A trigger resets the counter and starts the counter clock. Three types of triggers are common to both modes, and a fourth external trigger is available to each mode.

Regardless of the trigger used, it will be taken into account at the following active edge of the selected clock. This means that the counter value can be read differently from zero just after a trigger, especially when a low frequency signal is selected as the clock.

The following triggers are common to both modes:

- Software Trigger: Each channel has a software trigger, available by setting TC_CCR.SWTRG.
- SYNC: Each channel has a synchronization signal SYNC. When asserted, this signal has the same effect as a software trigger. The SYNC signals of all channels are asserted simultaneously by writing TC_BCR with SYNC set.
- Compare RC Trigger: RC is implemented in each channel and can provide a trigger when the counter value matches the RC value if TC_CMRx.CPCTRG is set .

The channel can also be configured to have an external trigger. In Capture mode, the external trigger signal can be selected between TIOAx and TIOBx. In Waveform mode, an external event can be programmed on one of the following signals: TIOBx, XC0, XC1 or XC2. This external event can then be programmed to perform a trigger by setting TC_CMRx.ENETRG.

If an external trigger is used, the duration of the pulses must be longer than the peripheral clock period in order to be detected.

40.6.7 Capture Mode

Capture mode is entered by clearing TC_CMRx.WAVE.

Capture mode allows the TC channel to perform measurements such as pulse timing, frequency, period, duty cycle and phase on TIOAx and TIOBx signals which are considered as inputs.

The figure Figure 40-6 shows the configuration of the TC channel when programmed in Capture mode.

40.6.8 Capture Registers A and B

Registers A and B (TC_RA and TC_RB) are used as capture registers. They can be loaded with the counter value when a programmable event occurs on the signal TIOAx.

TC_CMRx.LDRA defines the TIOAx selected edge for the loading of TC_RA, and TC_CMRx.LDRB defines the TIOAx selected edge for the loading of TC_RB.

The subsampling ratio defined by TC_CMRx.SBSMPLR is applied to these selected edges, so that the loading of Register A and Register B occurs once every 1, 2, 4, 8 or 16 selected edges.

TC_RA is loaded only if it has not been loaded since the last trigger or if TC_RB has been loaded since the last loading of TC_RA.

TC_RB is loaded only if TC_RA has been loaded since the last trigger or the last loading of TC_RB.

Loading TC_RA or TC_RB before the read of the last value loaded sets TC_SR.LOVRS. In this case, the old value is overwritten.

When DMA is used, the Register AB (TC_RAB) address must be configured as source address of the transfer. TC_RAB provides the next unread value from TC_RA and TC_RB. It may be read by the DMA after a request has been triggered upon loading TC_RA or TC_RB.

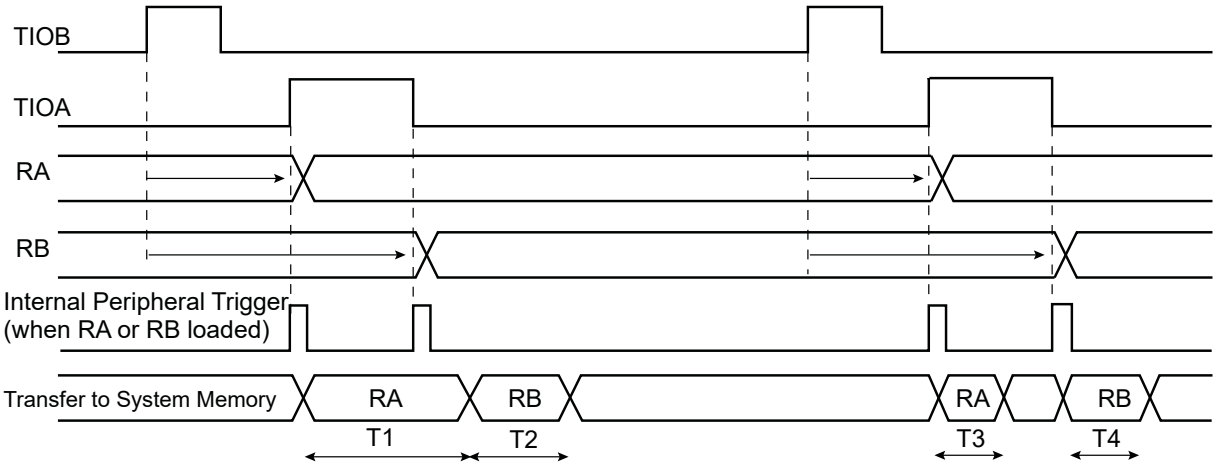
40.6.9 Transfer with DMAC in Capture Mode

The DMAC can perform access from the TC to system memory in Capture mode only.

The following figure illustrates how TC_RA and TC_RB can be loaded in the system memory without processor intervention.

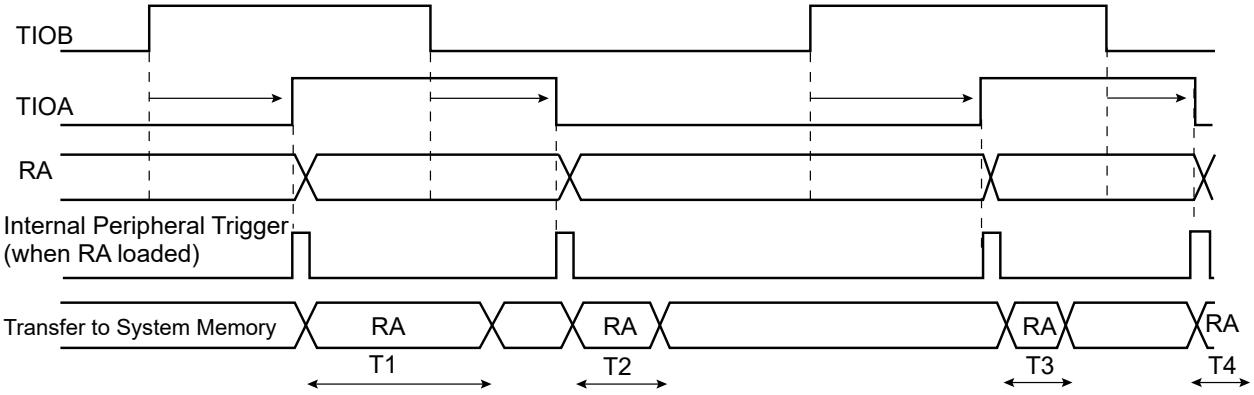
Figure 40-5. Example of Transfer with DMAC in Capture Mode

ETRGEDG = 1, LDRA = 1, LDRB = 2, ABETRG = 0



T1,T2,T3,T4 = System Bus load dependent (t_{min} = 8 Peripheral Clocks)

ETRGEDG = 3, LDRA = 3, LDRB = 0, ABETRG = 0



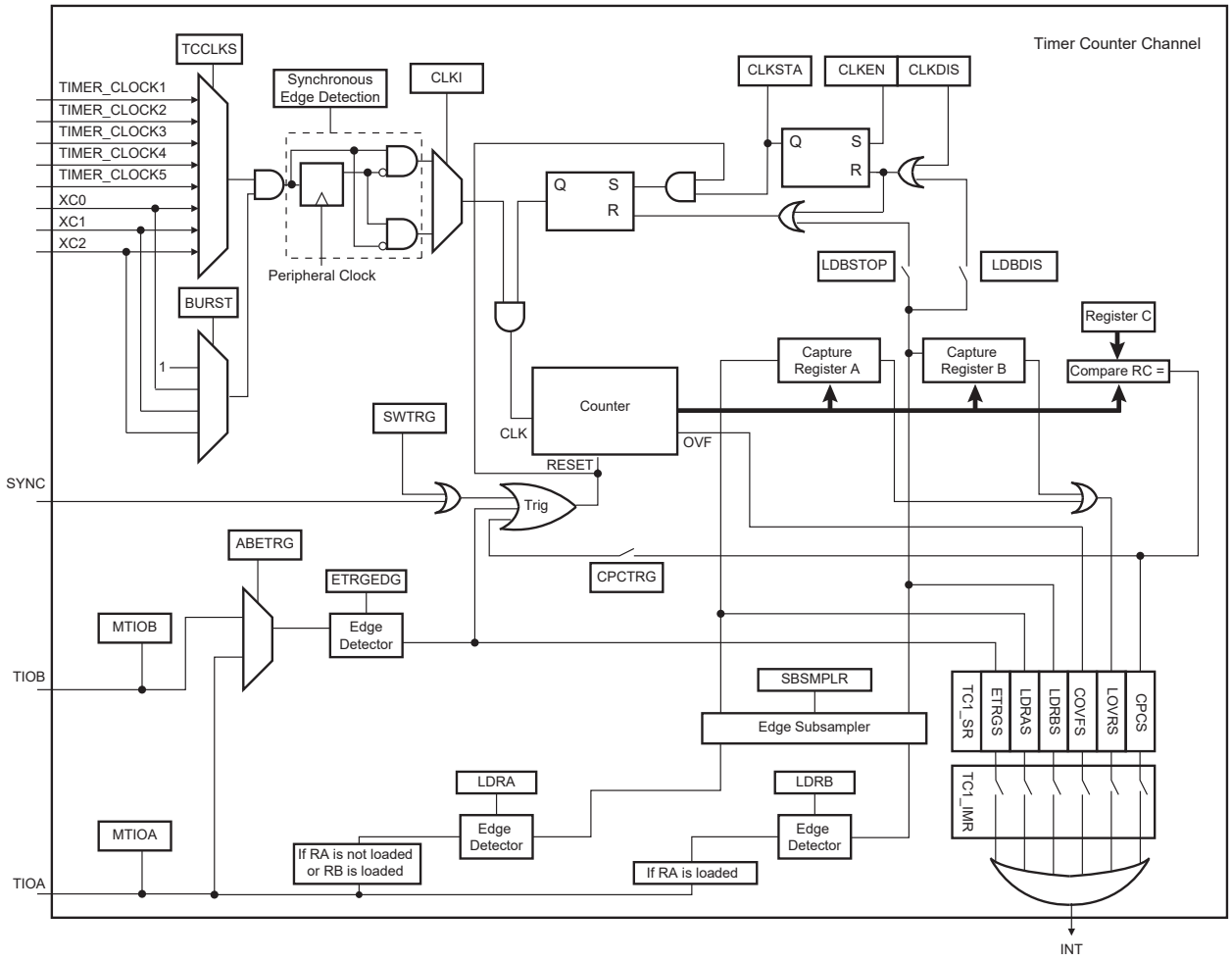
T1,T2,T3,T4 = System Bus load dependent (t_{min} = 8 Peripheral Clocks)

40.6.10 Trigger Conditions

In addition to the SYNC signal, the software trigger and the RC compare trigger, an external trigger can be defined.

The ABETRG bit in the TC_CMR selects TIOAx or TIOBx input signal as an external trigger or the trigger signal from the output comparator of the PWM module. The External Trigger Edge Selection parameter (ETRGEDG field in TC_CMR) defines the edge (rising, falling, or both) detected to generate an external trigger. If ETRGEDG = 0 (none), the external trigger is disabled.

Figure 40-6. Capture Mode



40.6.11 Waveform Mode

Waveform mode is entered by setting the TC_CMRx.WAVE bit.

In Waveform mode, the TC channel generates one or two PWM signals with the same frequency and independently programmable duty cycles, or generates different types of one-shot or repetitive pulses.

In this mode, TIOAx is configured as an output and TIOBx is defined as an output if it is not used as an external event (EEVT parameter in TC_CMR).

Waveform Mode shows the configuration of the TC channel when programmed in Waveform operating mode.

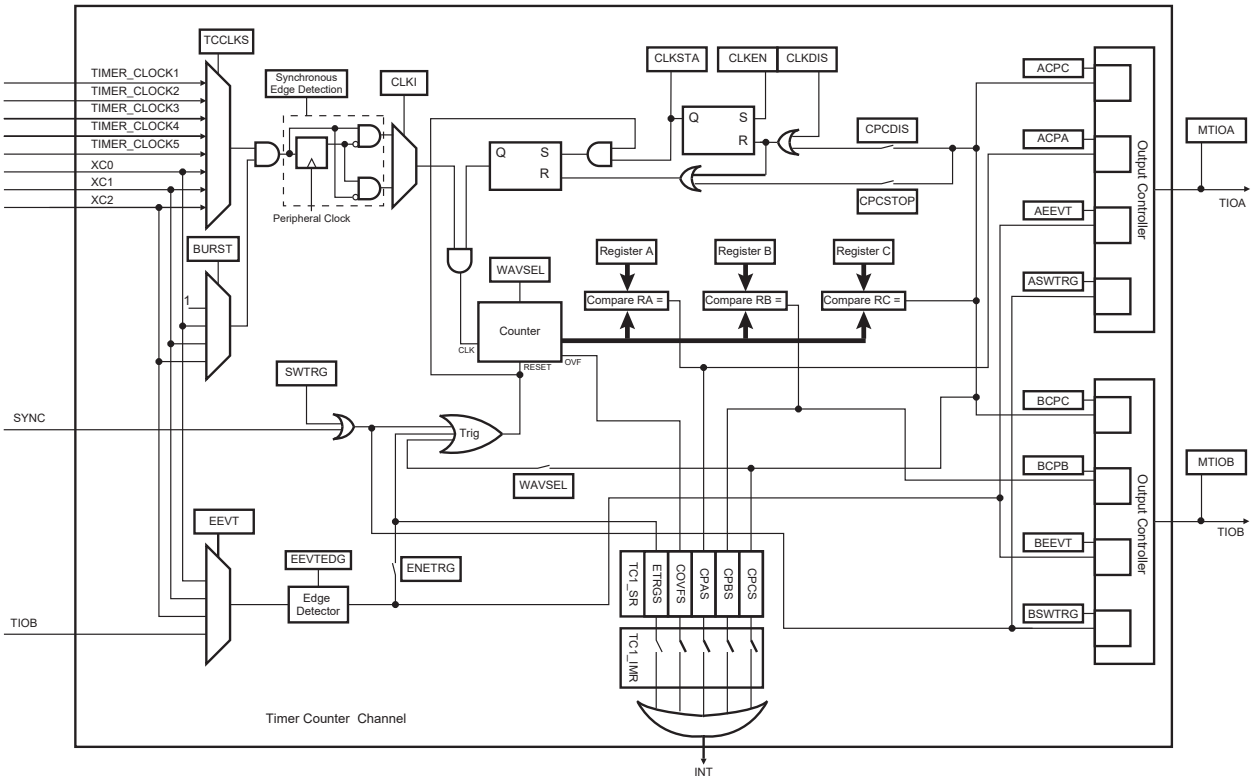
40.6.12 Waveform Selection

Depending on the WAVSEL parameter in TC_CMR, the behavior of TC_CV varies.

With any selection, TC_RA, TC_RB and TC_RC can all be used as compare registers.

RA Compare is used to control the TIOAx output, RB Compare is used to control the TIOBx output (if correctly configured) and RC Compare is used to control TIOAx and/or TIOBx outputs.

Figure 40-7. Waveform Mode



40.6.12.1 WAVSEL = 00

When WAVSEL = 00, the value of TC_CV is incremented from 0 to 2^{32}-1. Once 2^{32}-1 has been reached, the value of TC_CV is reset. Incrementation of TC_CV starts again and the cycle continues.

An external event trigger or a software trigger can reset the value of TC_CV. It is important to note that the trigger may occur at any time.

Refer to the figures below.

RC Compare cannot be programmed to generate a trigger in this configuration. At the same time, RC Compare can stop the counter clock (CPCSTOP = 1 in TC_CMR) and/or disable the counter clock (CPCDIS = 1 in TC_CMR).

Figure 40-8. WAVSEL = 00 without Trigger

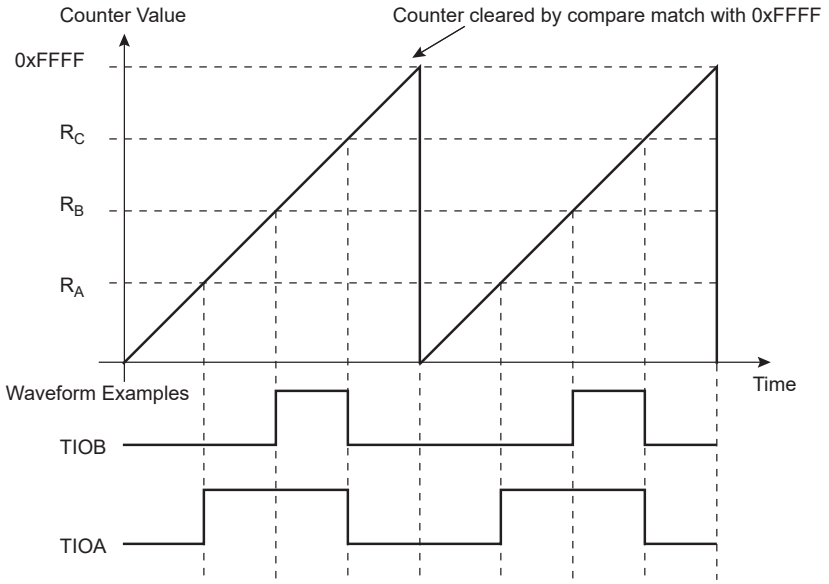
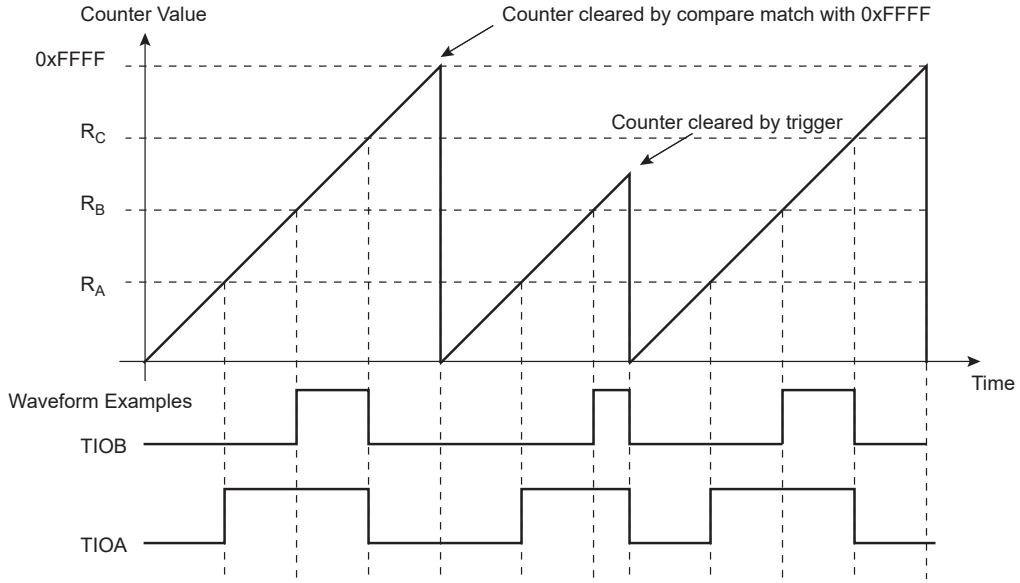


Figure 40-9. WAVSEL = 00 with Trigger



40.6.12.2 WAVSEL = 10

When WAVSEL = 10, the value of TC_CV is incremented from 0 to the value of RC, then automatically reset on a RC Compare. Once the value of TC_CV has been reset, it is then incremented and so on.

It is important to note that TC_CV can be reset at any time by an external event or a software trigger if both are programmed correctly.

Refer to the figures below.

In addition, RC Compare can stop the counter clock (CPCSTOP = 1 in TC_CMR) and/or disable the counter clock (CPCDIS = 1 in TC_CMR).

Figure 40-10. WAVSEL = 10 without Trigger

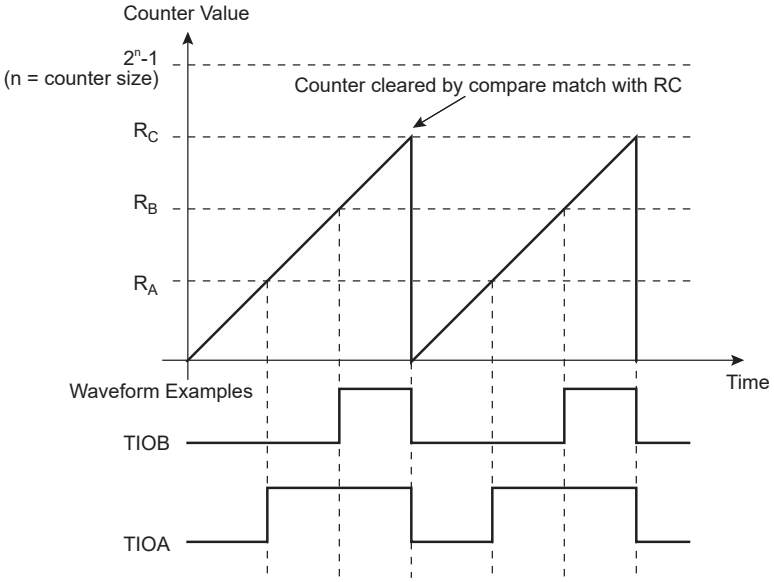
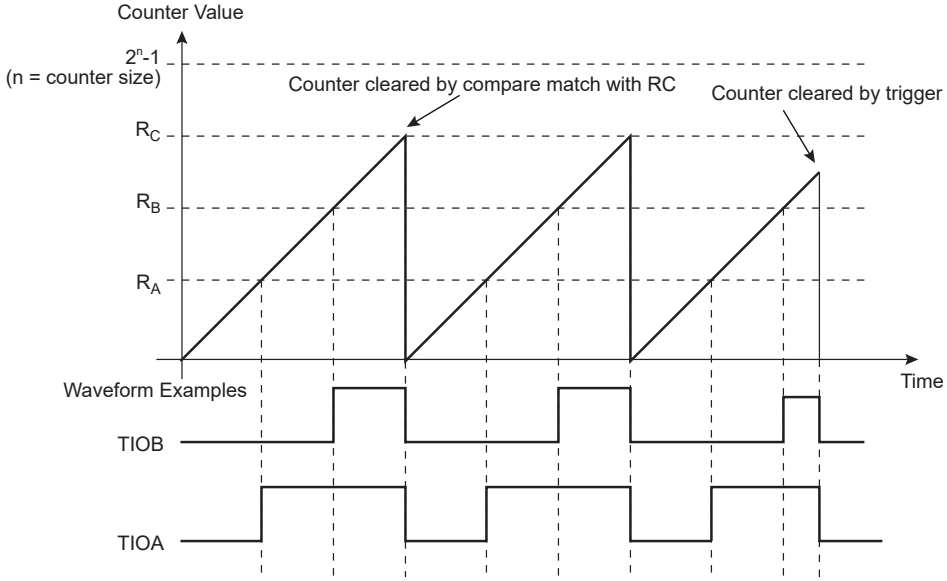


Figure 40-11. WAVSEL = 10 with Trigger



40.6.12.3 WAVSEL = 01

When WAVSEL = 01, the value of TC_CV is incremented from 0 to 2^{32}-1 . Once 2^{32}-1 is reached, the value of TC_CV is decremented to 0, then reincremented to 2^{32}-1 and so on.

A trigger such as an external event or a software trigger can modify TC_CV at any time. If a trigger occurs while TC_CV is incrementing, TC_CV then decrements. If a trigger is received while TC_CV is decrementing, TC_CV then increments.

Refer to the figures below.

RC Compare cannot be programmed to generate a trigger in this configuration.

At the same time, RC Compare can stop the counter clock (CPCSTOP = 1) and/or disable the counter clock (CPCDIS = 1).

Figure 40-12. WAVSEL = 01 without Trigger

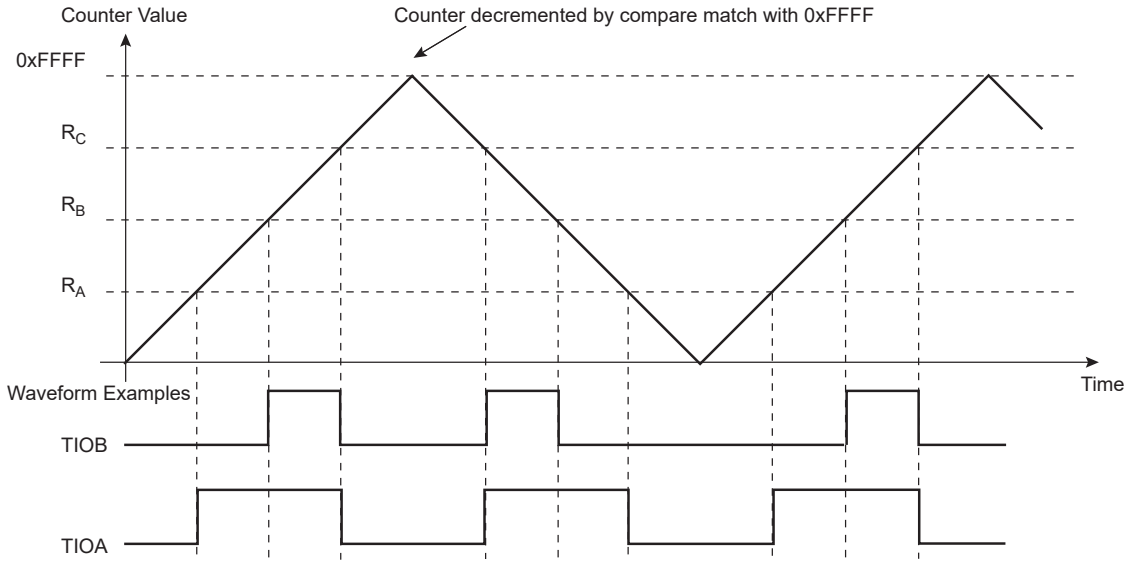
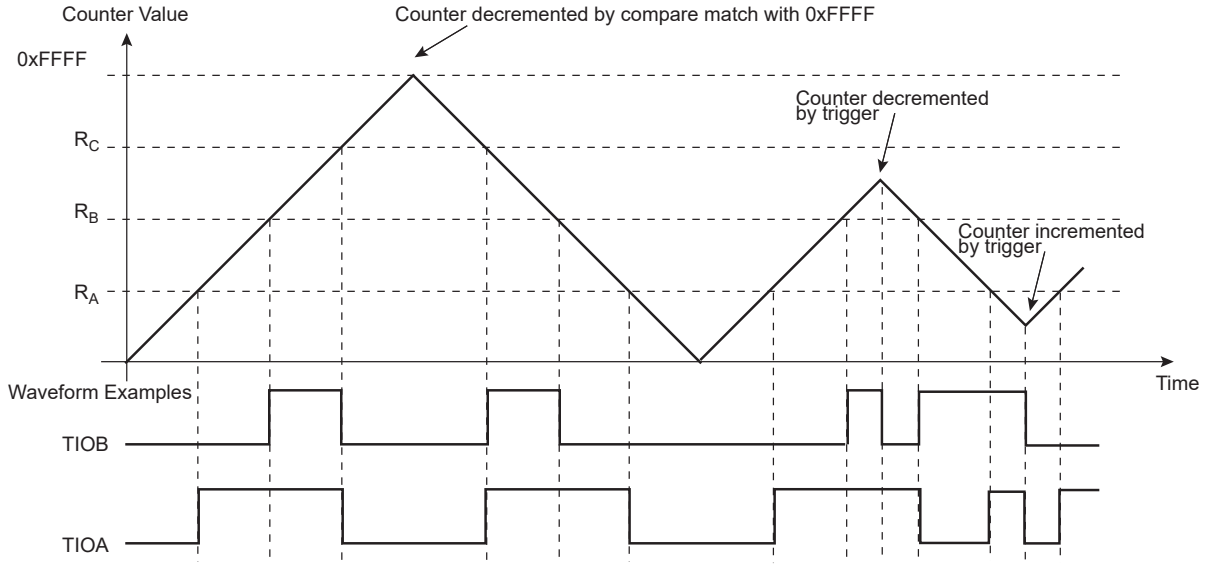


Figure 40-13. WAVSEL = 01 with Trigger



40.6.12.4 WAVSEL = 11

When WAVSEL = 11, the value of TC_CV is incremented from 0 to RC. Once RC is reached, the value of TC_CV is decremented to 0, then reincremented to RC and so on.

A trigger such as an external event or a software trigger can modify TC_CV at any time. If a trigger occurs while TC_CV is incrementing, TC_CV then decrements. If a trigger is received while TC_CV is decrementing, TC_CV then increments.

Refer to the figures below.

RC Compare can stop the counter clock (CPCSTOP = 1) and/or disable the counter clock (CPCDIS = 1).

Figure 40-14. WAVSEL = 11 without Trigger

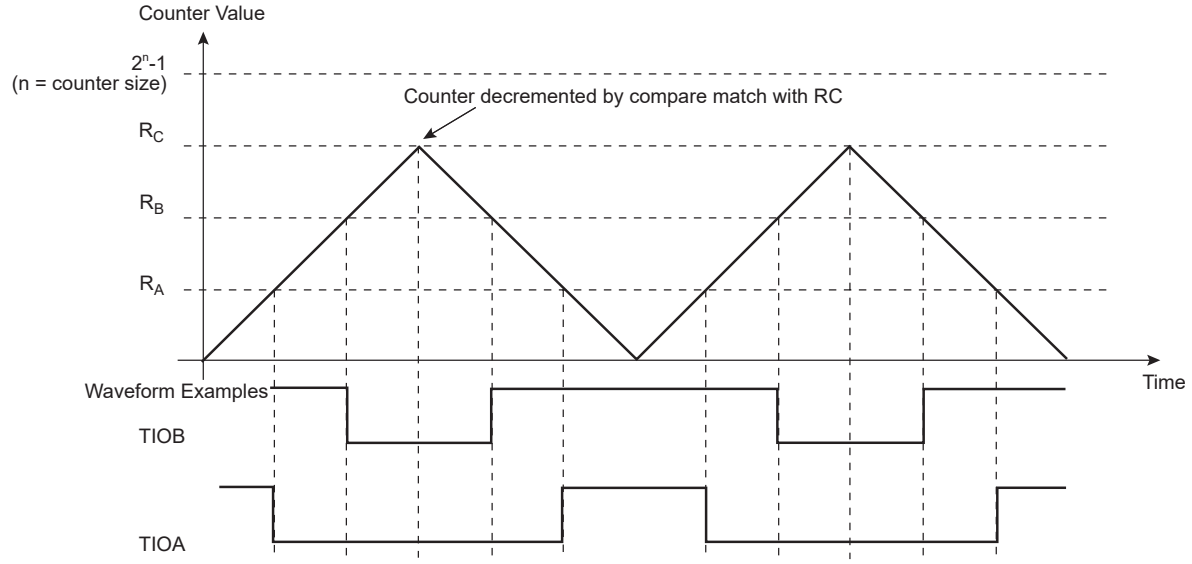
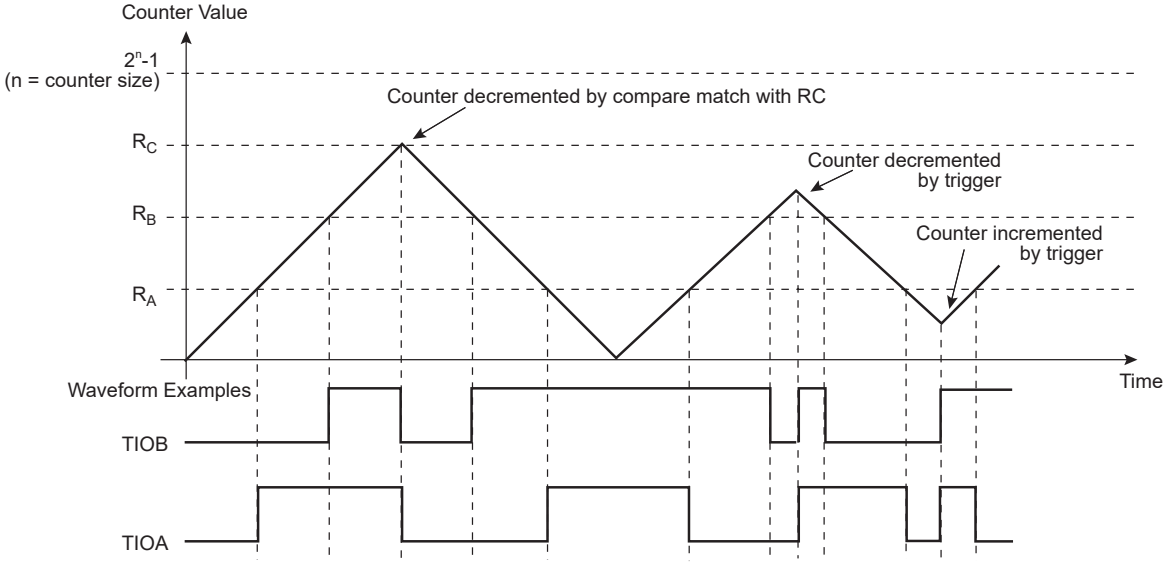


Figure 40-15. WAVSEL = 11 with Trigger



40.6.13 External Event/Trigger Conditions

An external event can be programmed to be detected on one of the clock sources (XC0, XC1, XC2) or TIOBx. The external event selected can then be used as a trigger.

The event trigger is selected using TC_CMR.EEVT. The trigger edge (rising, falling or both) for each of the possible external triggers is defined in TC_CMR.EEVTEDG. If EEVTEDG is cleared (none), no external event is defined.

If TIOBx is defined as an external event signal (EEVT = 0), TIOBx is no longer used as an output and the compare register B is not used to generate waveforms and subsequently no IRQs. In this case, the TC channel can only generate a waveform on TIOAx.

When an external event is defined, it can be used as a trigger by setting TC_CMR.ENETRG.

As in Capture mode, the SYNC signal and the software trigger are also available as triggers. RC Compare can also be used as a trigger depending on the parameter WAVSEL.

40.6.14 Synchronization with PWM

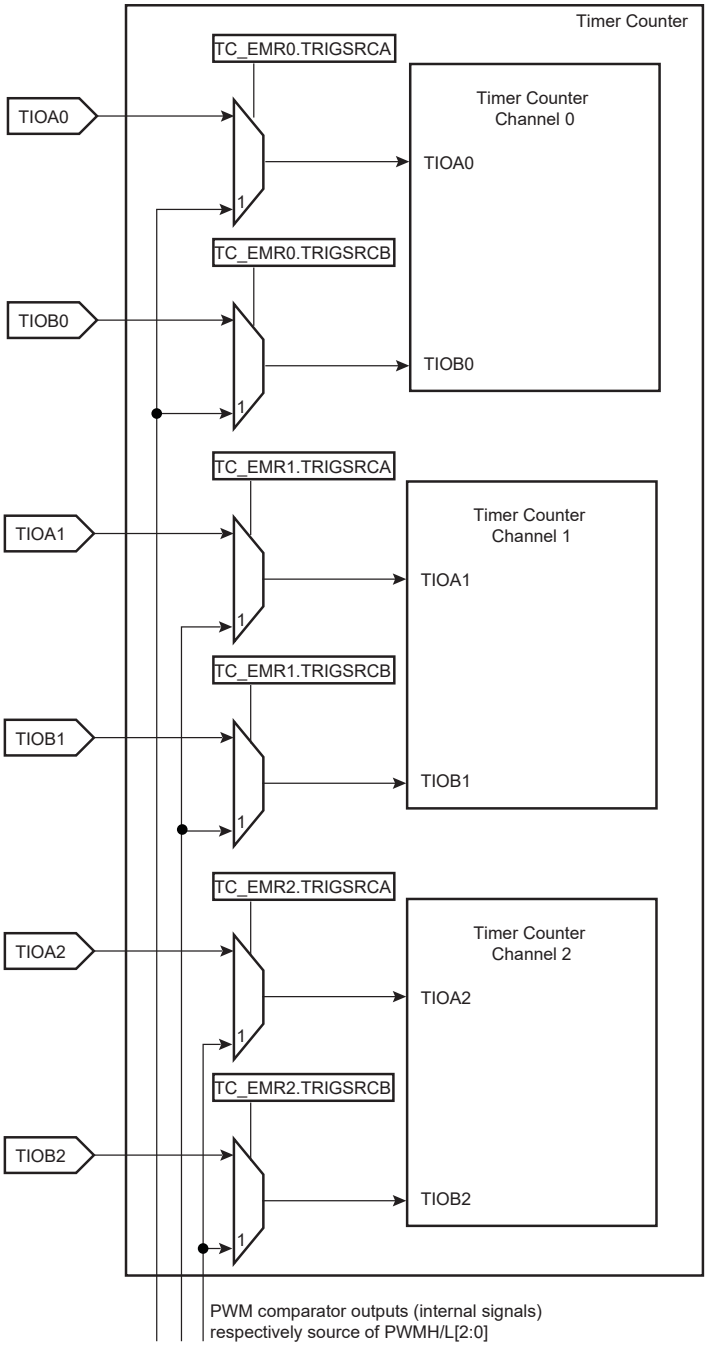
The inputs TIOAx/TIOBx can be bypassed, and thus channel trigger/capture events can be directly driven by the independent PWM module.

PWM comparator outputs (internal signals without dead-time insertion - OCx), respectively source of the PWMH/L[2:0] outputs, are routed to the internal TC inputs. These specific TC inputs are multiplexed with TIOA/B input signal to drive the internal trigger/capture events.

The selection is made in the Extended Mode register (TC_EMR) fields TRIGSRCA and TRIGSRCB (see "TC Extended Mode Register").

Each channel of the TC module can be synchronized by a different PWM channel as described in the following figure.

Figure 40-16. Synchronization with PWM



40.6.15 Output Controller

The output controller defines the output level changes on TIOAx and TIOBx following an event. TIOBx control is used only if TIOBx is defined as output (not as an external event).

The following events control TIOAx and TIOBx:

- Software trigger
- External event
- RC compare

RA Compare controls TIOAx, and RB Compare controls TIOBx. Each of these events can be programmed to set, clear or toggle the output as defined in the corresponding parameter in TC_CMR.

40.6.16 Quadrature Decoder

40.6.16.1 Description

The quadrature decoder (QDEC) is driven by TIOA0, TIOB0 and TIOB1 input pins and drives the timer counter of channel 0 and 1. Channel 2 can be used as a time base in case of speed measurement requirements (refer to Predefined Connection of the Quadrature Decoder with Timer Counters).

When writing a ‘0’ to TC_BMR.QDEN, the QDEC is bypassed and the IO pins are directly routed to the timer counter function.

TIOA0 and TIOB0 are to be driven by the two dedicated quadrature signals from a rotary sensor mounted on the shaft of the off-chip motor.

A third signal from the rotary sensor can be processed through pin TIOB1 and is typically dedicated to be driven by an index signal if it is provided by the sensor. This signal is not required to decode the quadrature signals PHA, PHB.

TC_CMRx.TCCLKS must be configured to select XC0 input (i.e., 0x101). Field TC0XC0S has no effect as soon as the QDEC is enabled.

Either speed or position/revolution can be measured. Position channel 0 accumulates the edges of PHA, PHB input signals giving a high accuracy on motor position whereas channel 1 accumulates the index pulses of the sensor, therefore the number of rotations. Concatenation of both values provides a high level of precision on motion system position.

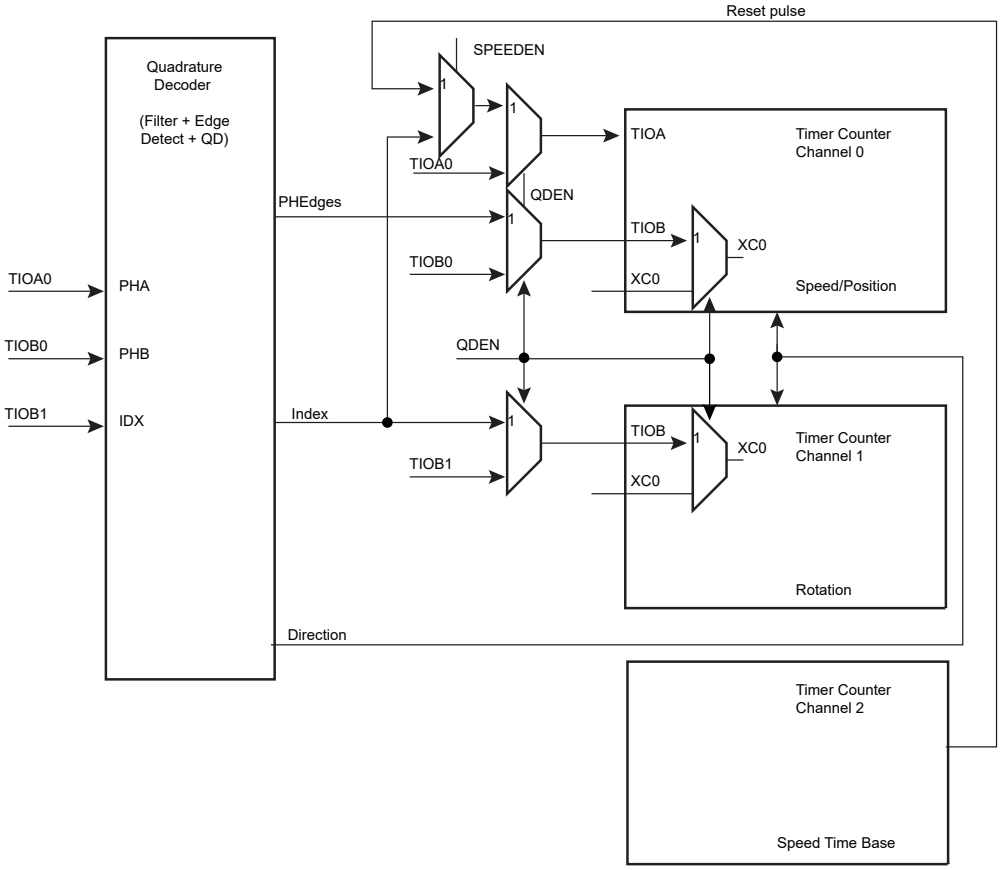
In Speed mode, position cannot be measured but revolution can be measured.

Inputs from the rotary sensor can be filtered prior to downstream processing. Accommodation of input polarity, phase definition and other factors are configurable.

Interruptions can be generated on different events.

A compare function (using TC_RC) is available on channel 0 (speed/position) or channel 1 (rotation) and can generate an interrupt by means of TC_SRx.CPCS.

Figure 40-17. Predefined Connection of the Quadrature Decoder with Timer Counters



40.6.16.2 Input Preprocessing

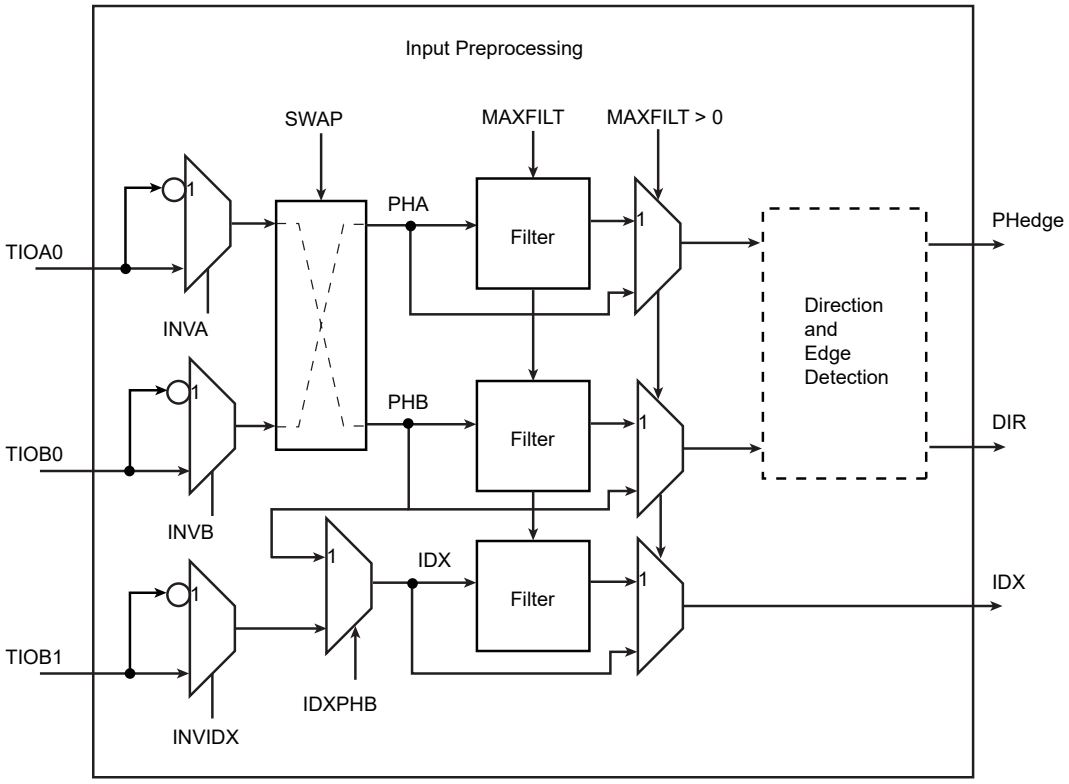
Input preprocessing consists of capabilities to take into account rotary sensor factors such as polarities and phase definition followed by configurable digital filtering.

Each input can be negated and swapping PHA, PHB is also configurable.

TC_BMR. MAXFILT is used to configure a minimum duration for which the pulse is stated as valid. When the filter is active, pulses with a duration lower than (MAXFILT +1) × $t_{peripheral\ clock}$ are not passed to downstream logic.

The value of (MAXFILT +1) × $t_{peripheral\ clock}$ must not be greater than 10% of the minimum pulse on PHA, PHB or index when the rotary encoder speed is at its maximum. This speed depends on the application.

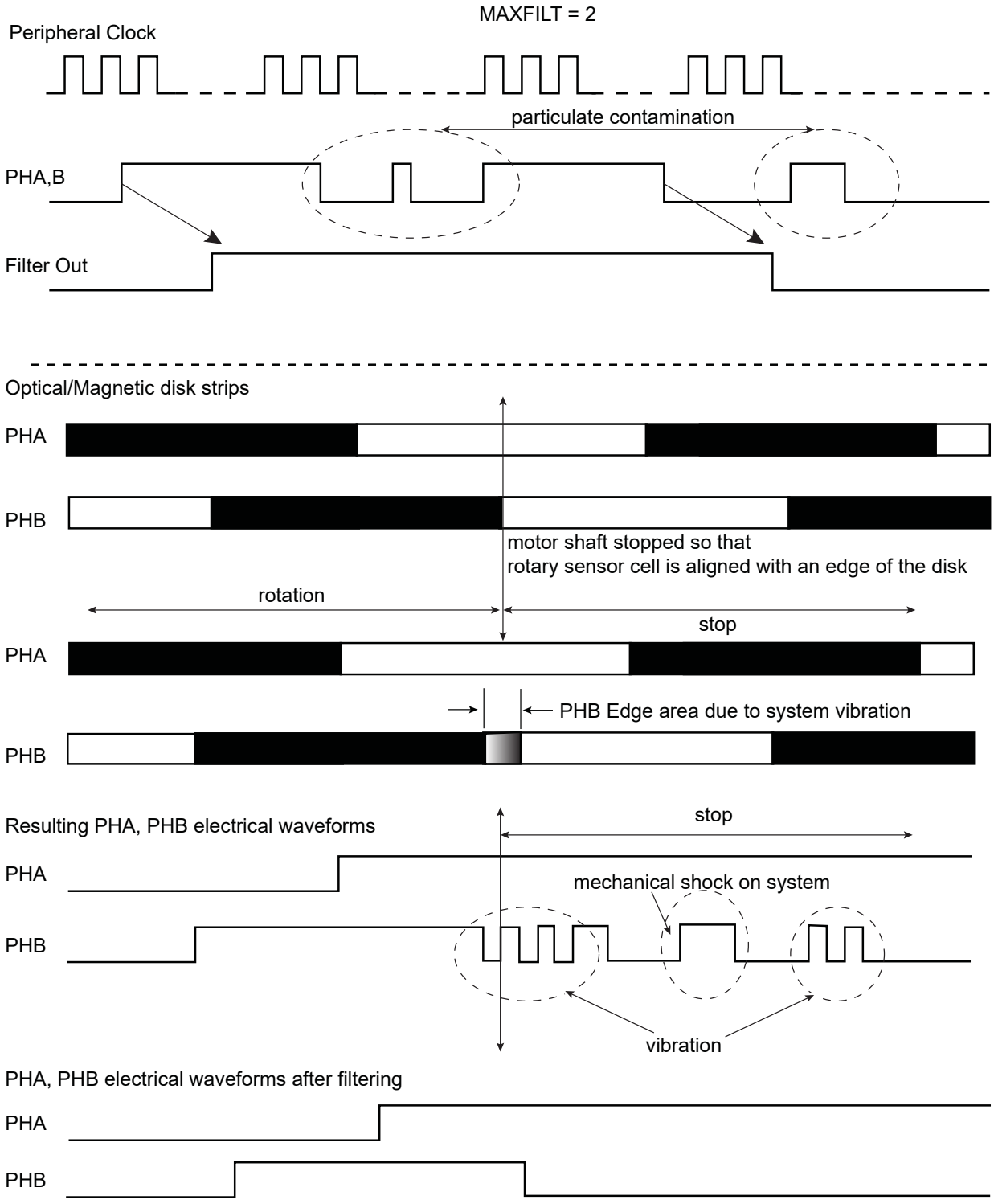
Figure 40-18. Input Stage



Input filtering can efficiently remove spurious pulses that might be generated by the presence of particulate contamination on the optical or magnetic disk of the rotary sensor.

Spurious pulses can also occur in environments with high levels of electromagnetic interference. Or, simply if vibration occurs even when rotation is fully stopped and the shaft of the motor is in such a position that the beginning of one of the reflective or magnetic bars on the rotary sensor disk is aligned with the light or magnetic (Hall) receiver cell of the rotary sensor. Any vibration can make the PHA, PHB signals toggle for a short duration.

Figure 40-19. Filtering Examples



40.6.16.3 Direction Status and Change Detection

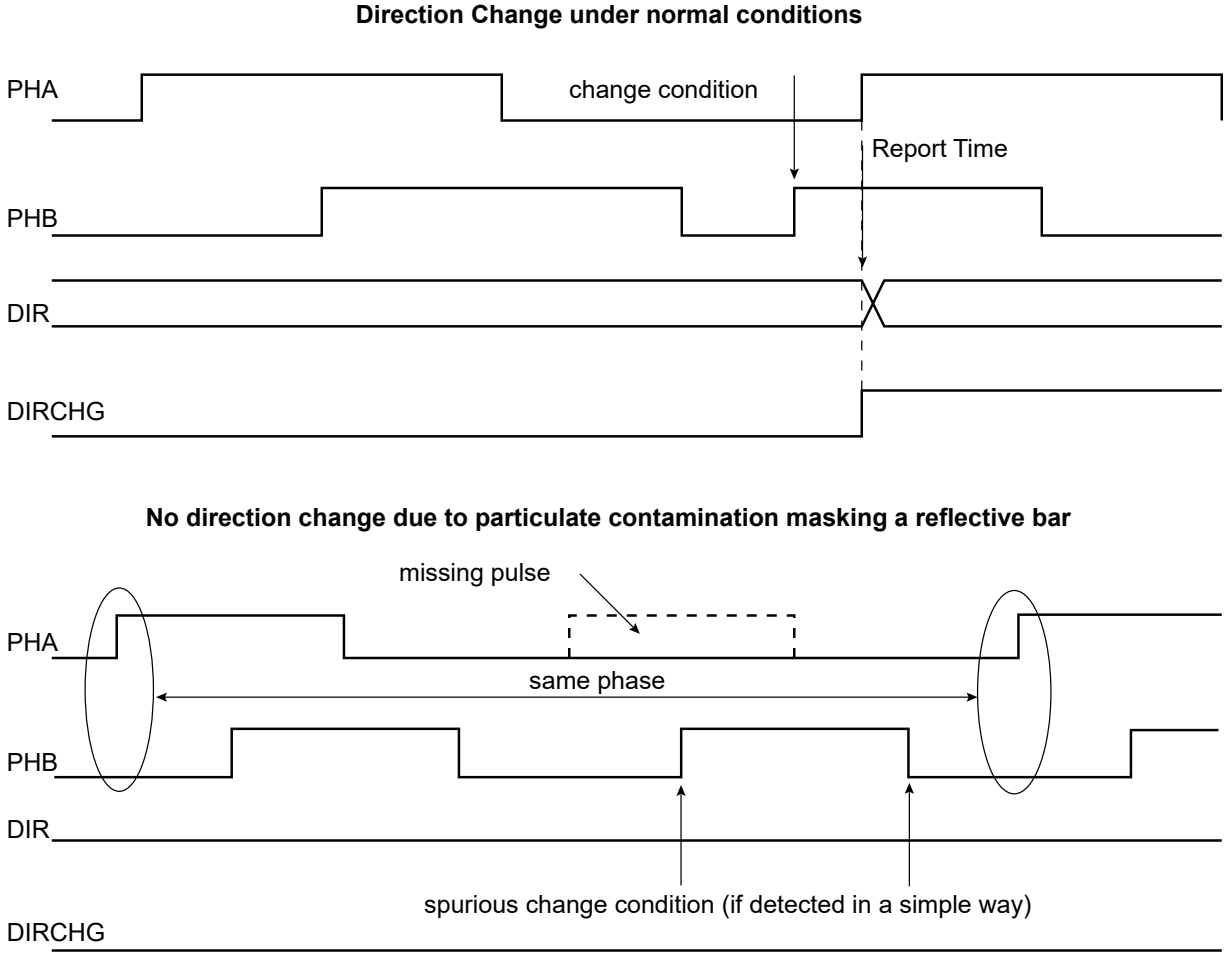
After filtering, the quadrature signals are analyzed to extract the rotation direction and edges of the two quadrature signals detected in order to be counted by TC logic downstream.

The direction status can be directly read at anytime in the TC_QISR. The polarity of the direction flag status depends on the configuration written in TC_BMR. INVA, INVB, INVIDX, SWAP modify the polarity of DIR flag.

Any change in rotation direction is reported in the TC_QISR and can generate an interrupt.

The direction change condition is reported as soon as two consecutive edges on a phase signal have sampled the same value on the other phase signal and there is an edge on the other signal. The two consecutive edges of one phase signal sampling the same value on other phase signal is not sufficient to declare a direction change, as particulate contamination may mask one or more reflective bars on the optical or magnetic disk of the sensor. Refer to the following figure for waveforms.

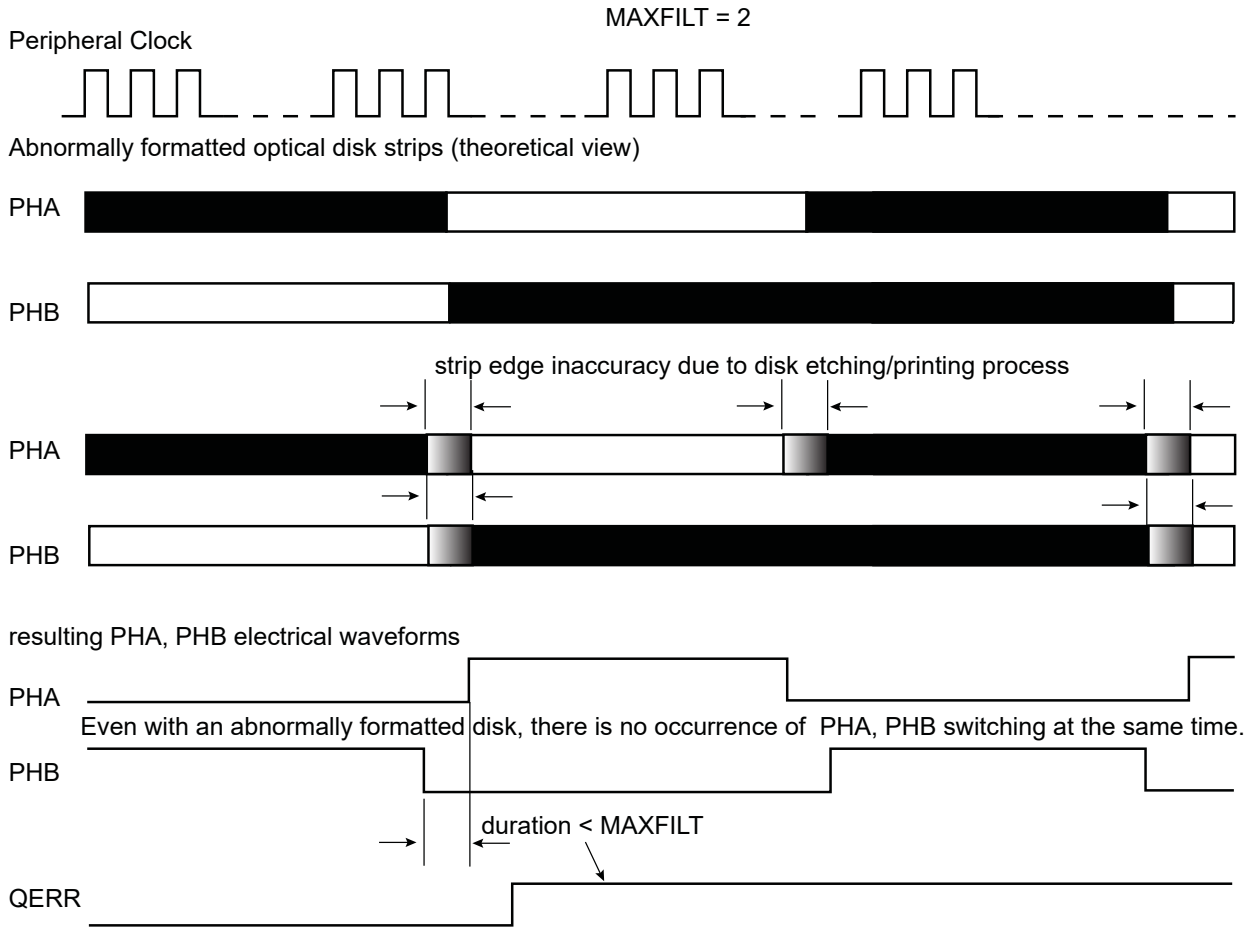
Figure 40-20. Rotation Change Detection



The direction change detection is disabled when TC_BMR.QDTRANS is set. In this case, the DIR flag report must not be used.

A quadrature error is also reported by the QDEC via TC_QISR.QERR. This error is reported if the time difference between two edges on PHA, PHB is lower than a predefined value. This predefined value is configurable and corresponds to (TC_BMR.MAXFILT + 1) × $t_{peripheral\ clock}$ ns. After being filtered, there is no reason to have two edges closer than (TC_BMR.MAXFILT + 1) × $t_{peripheral\ clock}$ ns under normal mode of operation.

Figure 40-21. Quadrature Error Detection



MAXFILT must be tuned according to several factors such as the peripheral clock frequency, type of rotary sensor and rotation speed to be achieved.

40.6.16.4 Position and Rotation Measurement

When TC_BMR.POSEN is set, the motor axis position is processed on channel 0 (by means of the PHA, PHB edge detections) and the number of motor revolutions are recorded on channel 1 if the IDX signal is provided on the TIOB1 input. If no IDX signal is available, the internal counter can be cleared for each revolution if the number of counts per revolution is configured in TC_RC0.RC and the TC_CMR.CPCTRG bit is written to '1'. The position measurement can be read in the TC_CV0 register and the rotation measurement can be read in the TC_CV1 register.

Channel 0 and 1 must be configured in Capture mode (TC_CMR0.WAVE = 0). 'Rising edge' must be selected as the External Trigger Edge (TC_CMR.ETRGEDG = 0x01) and 'TIOAx' must be selected as the External Trigger (TC_CMR.ABETRG = 0x1). The process must be started by configuring TC_CCR.CLKEN and TC_CCR.SWTRG.

In parallel, the number of edges are accumulated on TC channel 0 and can be read on the TC_CV0 register.

Therefore, the accurate position can be read on both TC_CV registers and concatenated to form a 32-bit word.

The TC channel 0 is cleared for each increment of IDX count value.

Depending on the quadrature signals, the direction is decoded and allows to count up or down in TC channels 0 and 1. The direction status is reported on TC_QISR.

40.6.16.5 Speed Measurement

When TC_BMR.SPEEDEN is set, the speed measure is enabled on channel 0.

A time base must be defined on channel 2 by writing the TC_RC2 period register. Channel 2 must be configured in Waveform mode (WAVE bit set) in TC_CMR2. The WAVSEL field must be defined with 0x10 to clear the counter by comparison and matching with TC_RC value. Field ACPC must be defined at 0x11 to toggle TIOAx output.

This time base is automatically fed back to TIOAx of channel 0 when QDEN and SPEEDEN are set.

Channel 0 must be configured in Capture mode (WAVE = 0 in TC_CMR0). TC_CMR0.ABETRG must be configured at 1 to select TIOAx as a trigger for this channel.

EDGTRG must be set to 0x01, to clear the counter on a rising edge of the TIOAx signal and field LDRA must be set accordingly to 0x01, to load TC_RA0 at the same time as the counter is cleared (LDRB must be set to 0x01). As a consequence, at the end of each time base period the differentiation required for the speed calculation is performed.

The process must be started by configuring bits CLKEN and SWTRG in the TC_CCR.

The speed can be read on field RA in TC_RA0.

Channel 1 can still be used to count the number of revolutions of the motor.

40.6.16.6 Detecting a Missing Index Pulse

To detect a missing index pulse due contamination, dust, etc., the TC_SR0.CPCS flag can be used. It is also possible to assert the interrupt line if the TC_SR0.CPCS flag is enabled as a source of the interrupt by writing a ‘1’ to TC_IER0.CPCS.

The TC_RC0.RC field must be written with the nominal number of counts per revolution provided by the rotary encoder, plus a margin to eliminate potential noise (e.g., if nominal count per revolution is 1024, then TC_RC0.RC=1026).

If the index pulse is missing, the timer value is not cleared and the nominal value is exceeded, then the comparator on the RC triggers an event, TC_SR0.CPCS=1, and the interrupt line is asserted if TC_IER0.CPCS=1.

The missing index pulse detection is only valid if the bit TC_QISR.DIRCHG=0.

40.6.16.7 Detecting Contamination/Dust at Rotary Encoder Low Speed

The contamination/dust that can be filtered when the rotary encoder speed is high may not be filtered at low speed, thus creating unsollicited direction change, etc.

At low speed, even a minor contamination may appear as a long pulse, and thus not filtered and processed as a standard quadrature encoder pulse.

This contamination can be detected by using the similar method as the missing index detection.

A contamination exists on a phase line if TC_SR.CPCS = 1 and TC_QISR.DIRCHG = 1 when there is no sollicited change of direction.

40.6.17 2-bit Gray Up/Down Counter for Stepper Motor

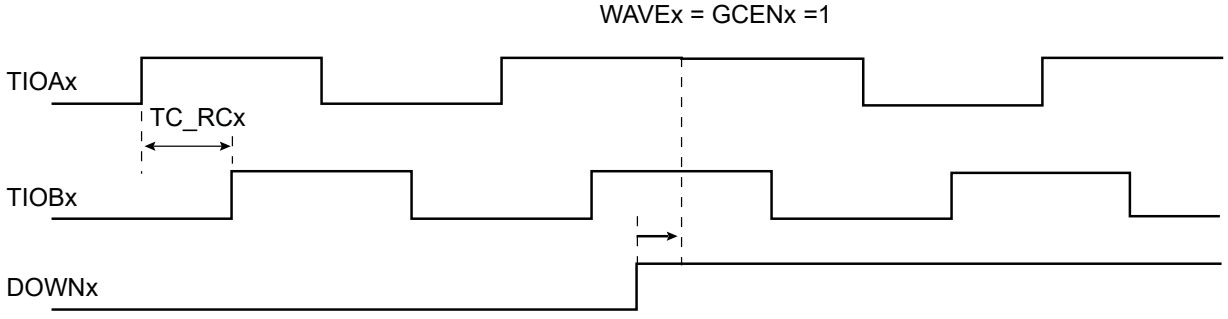
Each channel can be independently configured to generate a 2-bit Gray count waveform on corresponding TIOAx, TIOBx outputs by means of TC_SMMRx.GCEN.

Up or Down count can be defined by writing TC_SMMRx.DOWN.

It is mandatory to configure the channel in Waveform mode in the TC_CMR.

The period of the counters can be programmed in TC_RCx.

Figure 40-22. 2-bit Gray Up/Down Counter



40.6.18 Fault Mode

At any time, the TC_RCx registers can be used to perform a comparison on the respective current channel counter value (TC_CVx) with the value of TC_RCx register.

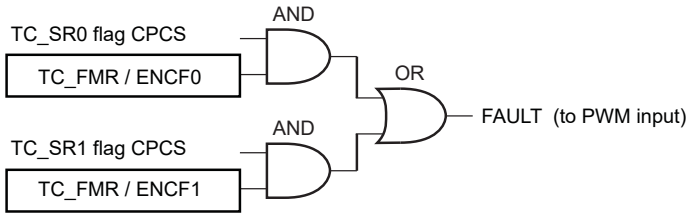
The CPCSx flags can be set accordingly and an interrupt can be generated.

This interrupt is processed but requires an unpredictable amount of time to be achieve the required action.

It is possible to trigger the FAULT output of the TIMER1 with CPCS from TC_SR0 and/or CPCS from TC_SR1. Each source can be independently enabled/disabled in the TC_FMR.

This can be useful to detect an overflow on speed and/or position when QDEC is processed and to act immediately by using the FAULT output.

Figure 40-23. Fault Output Generation



40.6.19 Register Write Protection

To prevent any single software error from corrupting TC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the TC Write Protection Mode Register (TC_WPMR).

The Timer Counter clock of the first channel must be enabled to access TC_WPMR.

The following registers can be write-protected when WPEN is set:

- TC Block Mode Register
- TC Channel Mode Register Capture Mode
- TC Channel Mode Register Waveform Mode
- TC Fault Mode Register
- TC Stepper Motor Mode Register
- TC Register A
- TC Register B
- TC Register C
- TC Extended Mode Register

40.7 Register Summary

Note: The register TC_CMR has two modes, Capture Mode and Waveform Mode. In this register summary, only TC_CMR (CAPTURE_MODE) is displayed. For the configuration of TC_CMR (WAVEFORM_MODE), refer to the section TC Channel Mode Register: Waveform Mode.

Offset	Name	Bit Pos.								
0x00	TC_CCR0	7:0						SWTRG	CLKDIS	CLKEN
		15:8								
		23:16								
		31:24								
0x04	TC_CMR0	7:0	LDBDIS	LDBSTOP	BURST[1:0]		CLKI	TCCLKS[2:0]		
		15:8	WAVE	CPCTRG				ABETRG	ETRGEDG[1:0]	
		23:16		SBSMPLR[2:0]			LDRB[1:0]		LDRA[1:0]	
		31:24								
0x04	TC_CMR0	7:0	CPCDIS	CPCSTOP	BURST[1:0]		CLKI	TCCLKS[2:0]		
		15:8	WAVE	WAVSEL[1:0]		ENETRG	EEVT[1:0]		EEVTEDG[1:0]	
		23:16	ASWTRG[1:0]		AEEVT[1:0]		ACPC[1:0]		ACPA[1:0]	
		31:24	BSWTRG[1:0]		BEEVT[1:0]		BCPC[1:0]		BCPB[1:0]	
0x08	TC_SMMR0	7:0							DOWN	GCEN
		15:8								
		23:16								
		31:24								
0x0C	TC_RAB0	7:0	RAB[7:0]							
		15:8	RAB[15:8]							
		23:16	RAB[23:16]							
		31:24	RAB[31:24]							
0x10	TC_CV0	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24	CV[31:24]							
0x14	TC_RA0	7:0	RA[7:0]							
		15:8	RA[15:8]							
		23:16	RA[23:16]							
		31:24	RA[31:24]							
0x18	TC_RB0	7:0	RB[7:0]							
		15:8	RB[15:8]							
		23:16	RB[23:16]							
		31:24	RB[31:24]							
0x1C	TC_RC0	7:0	RC[7:0]							
		15:8	RC[15:8]							
		23:16	RC[23:16]							
		31:24	RC[31:24]							
0x20	TC_SR0	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16						MTIOB	MTIOA	CLKSTA
		31:24								
0x24	TC_IER0	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16								
		31:24								
0x28	TC_IDR0	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16								
		31:24								
0x2C	TC_IMR0	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x30	TC_EMR0	7:0			TRIGSRCB[1:0]				TRIGSRCA[1:0]	
		15:8								NODIVCLK
		23:16								
		31:24								
0x34 ... 0x3F	Reserved									
0x40	TC_CCR1	7:0						SWTRG	CLKDIS	CLKEN
		15:8								
		23:16								
		31:24								
0x44	TC_CMR1	7:0	LDBDIS	LDBSTOP	BURST[1:0]		CLKI	TCCLKS[2:0]		
		15:8	WAVE	CPCTRG				ABETRG	ETRGEDG[1:0]	
		23:16		SBSMPLR[2:0]			LDRB[1:0]		LDRA[1:0]	
		31:24								
0x44	TC_CMR1	7:0	CPCDIS	CPCSTOP	BURST[1:0]		CLKI	TCCLKS[2:0]		
		15:8	WAVE	WAVSEL[1:0]		ENETRG	EEVT[1:0]		EEVTEDG[1:0]	
		23:16	ASWTRG[1:0]		AEEVT[1:0]		ACPC[1:0]		ACPA[1:0]	
		31:24	BSWTRG[1:0]		BEEVT[1:0]		BCPC[1:0]		BCPB[1:0]	
0x48	TC_SMMR1	7:0							DOWN	GCEN
		15:8								
		23:16								
		31:24								
0x4C	TC_RAB1	7:0	RAB[7:0]							
		15:8	RAB[15:8]							
		23:16	RAB[23:16]							
		31:24	RAB[31:24]							
0x50	TC_CV1	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24	CV[31:24]							
0x54	TC_RA1	7:0	RA[7:0]							
		15:8	RA[15:8]							
		23:16	RA[23:16]							
		31:24	RA[31:24]							
0x58	TC_RB1	7:0	RB[7:0]							
		15:8	RB[15:8]							
		23:16	RB[23:16]							
		31:24	RB[31:24]							
0x5C	TC_RC1	7:0	RC[7:0]							
		15:8	RC[15:8]							
		23:16	RC[23:16]							
		31:24	RC[31:24]							
0x60	TC_SR1	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16						MTIOB	MTIOA	CLKSTA
		31:24								
0x64	TC_IER1	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16								
		31:24								
0x68	TC_IDR1	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16								
		31:24								
0x6C	TC_IMR1	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x70	TC_EMR1	7:0			TRIGSRCB[1:0]				TRIGSRCA[1:0]	
		15:8								NODIVCLK
		23:16								
		31:24								
0x74 ... 0x7F	Reserved									
0x80	TC_CCR2	7:0						SWTRG	CLKDIS	CLKEN
		15:8								
		23:16								
		31:24								
0x84	TC_CMR2	7:0	LDBDIS	LDBSTOP	BURST[1:0]		CLKI	TCCLKS[2:0]		
		15:8	WAVE	CPCTRG				ABETRG	ETRGEDG[1:0]	
		23:16		SBSMPLR[2:0]			LDRB[1:0]		LDRA[1:0]	
		31:24								
0x84	TC_CMR2	7:0	CPCDIS	CPCSTOP	BURST[1:0]		CLKI	TCCLKS[2:0]		
		15:8	WAVE	WAVSEL[1:0]		ENETRG	EEVT[1:0]		EEVTEDG[1:0]	
		23:16	ASWTRG[1:0]		AEEVT[1:0]		ACPC[1:0]		ACPA[1:0]	
		31:24	BSWTRG[1:0]		BEEVT[1:0]		BCPC[1:0]		BCPB[1:0]	
0x88	TC_SMMR2	7:0							DOWN	GCEN
		15:8								
		23:16								
		31:24								
0x8C	TC_RAB2	7:0	RAB[7:0]							
		15:8	RAB[15:8]							
		23:16	RAB[23:16]							
		31:24	RAB[31:24]							
0x90	TC_CV2	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24	CV[31:24]							
0x94	TC_RA2	7:0	RA[7:0]							
		15:8	RA[15:8]							
		23:16	RA[23:16]							
		31:24	RA[31:24]							
0x98	TC_RB2	7:0	RB[7:0]							
		15:8	RB[15:8]							
		23:16	RB[23:16]							
		31:24	RB[31:24]							
0x9C	TC_RC2	7:0	RC[7:0]							
		15:8	RC[15:8]							
		23:16	RC[23:16]							
		31:24	RC[31:24]							
0xA0	TC_SR2	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16						MTIOB	MTIOA	CLKSTA
		31:24								
0xA4	TC_IER2	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16								
		31:24								
0xA8	TC_IDR2	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16								
		31:24								
0xAC	TC_IMR2	7:0	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
		15:8								
		23:16								
		31:24								

.....continued

Offset	Name	Bit Pos.								
0xB0	TC_EMR2	7:0			TRIGSRCB[1:0]				TRIGSRCA[1:0]	
		15:8								NODIVCLK
		23:16								
		31:24								
0xB4 ... 0xBF	Reserved									
0xC0	TC_BCR	7:0								SYNC
		15:8								
		23:16								
		31:24								
0xC4	TC_BMR	7:0			TC2XC2S[1:0]		TC1XC1S[1:0]		TC0XC0S[1:0]	
		15:8	INVIDX	INVB	INVA	EDGPHA	QDTRANS	SPEEDEN	POSEN	QDEN
		23:16	MAXFILT[3:0]						IDXPHB	SWAP
		31:24							MAXFILT[5:4]	
0xC8	TC_QIER	7:0						QERR	DIRCHG	IDX
		15:8								
		23:16								
		31:24								
0xCC	TC_QIDR	7:0						QERR	DIRCHG	IDX
		15:8								
		23:16								
		31:24								
0xD0	TC_QIMR	7:0						QERR	DIRCHG	IDX
		15:8								
		23:16								
		31:24								
0xD4	TC_QISR	7:0						QERR	DIRCHG	IDX
		15:8								DIR
		23:16								
		31:24								
0xD8	TC_FMR	7:0							ENCF1	ENCF0
		15:8								
		23:16								
		31:24								
0xDC ... 0xE3	Reserved									
0xE4	TC_WPMR	7:0								WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							

40.7.1 TC Channel Control Register

Name: TC_CCRx
Offset: 0x00 + x*0x40 [x=0..2]
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						SWTRG	CLKDIS	CLKEN
Access						W	W	W
Reset						–	–	–

Bit 2 – SWTRG Software Trigger Command

Value	Description
0	No effect.
1	A software trigger is performed: the counter is reset and the clock is started.

Bit 1 – CLKDIS Counter Clock Disable Command

Value	Description
0	No effect.
1	Disables the clock.

Bit 0 – CLKEN Counter Clock Enable Command

Value	Description
0	No effect.
1	Enables the clock if CLKDIS is not 1.

40.7.2 TC Channel Mode Register: Capture Mode

Name: TC_CMRx
Offset: 0x04 + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read/Write

This register can be written only if the WPEN bit is cleared in the TC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
		SBSMPLR[2:0]			LDRB[1:0]		LDRA[1:0]	
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WAVE	CPCTRG				ABETRG	ETRGEDG[1:0]	
Access	R/W	R/W				R/W	R/W	R/W
Reset	0	0				0	0	0

Bit	7	6	5	4	3	2	1	0
	LDBDIS	LDBSTOP	BURST[1:0]		CLKI	TCCLKS[2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 22:20 – SBSMPLR[2:0] Loading Edge Subsampling Ratio

Value	Name	Description
0	ONE	Load a Capture register each selected edge.
1	HALF	Load a Capture register every 2 selected edges.
2	FOURTH	Load a Capture register every 4 selected edges.
3	EIGHTH	Load a Capture register every 8 selected edges.
4	SIXTEENTH	Load a Capture register every 16 selected edges.

Bits 19:18 – LDRB[1:0] RB Loading Edge Selection

Value	Name	Description
0	NONE	None
1	RISING	Rising edge of TIOAx
2	FALLING	Falling edge of TIOAx
3	EDGE	Each edge of TIOAx

Bits 17:16 – LDRA[1:0] RA Loading Edge Selection

Value	Name	Description
0	NONE	None
1	RISING	Rising edge of TIOAx
2	FALLING	Falling edge of TIOAx
3	EDGE	Each edge of TIOAx

Bit 15 – WAVE Waveform Mode

Value	Description
0	Capture mode is enabled.
1	Capture mode is disabled (Waveform mode is enabled).

Bit 14 – CPCTRG RC Compare Trigger Enable

Value	Description
0	RC Compare has no effect on the counter and its clock.
1	RC Compare resets the counter and starts the counter clock.

Bit 10 – ABETRG TIOAx or TIOBx External Trigger Selection

Value	Description
0	TIOBx is used as an external trigger.
1	TIOAx is used as an external trigger.

Bits 9:8 – ETRGEDG[1:0] External Trigger Edge Selection

Value	Name	Description
0	NONE	The clock is not gated by an external signal.
1	RISING	Rising edge
2	FALLING	Falling edge
3	EDGE	Each edge

Bit 7 – LDBDIS Counter Clock Disable with RB Loading

Value	Description
0	Counter clock is not disabled when RB loading occurs.
1	Counter clock is disabled when RB loading occurs.

Bit 6 – LDBSTOP Counter Clock Stopped with RB Loading

Value	Description
0	Counter clock is not stopped when RB loading occurs.
1	Counter clock is stopped when RB loading occurs.

Bits 5:4 – BURST[1:0] Burst Signal Selection

Value	Name	Description
0	NONE	The clock is not gated by an external signal.
1	XC0	XC0 is ANDed with the selected clock.
2	XC1	XC1 is ANDed with the selected clock.
3	XC2	XC2 is ANDed with the selected clock.

Bit 3 – CLKI Clock Invert

Value	Description
0	Counter is incremented on rising edge of the clock.
1	Counter is incremented on falling edge of the clock.

Bits 2:0 – TCCLKS[2:0] Clock Selection

To operate at maximum peripheral clock frequency, refer to “TC Extended Mode Register”.

Value	Name	Description
0	TIMER_CLOCK1	Clock selected: internal GCLK clock signal (from PMC)
1	TIMER_CLOCK2	Clock selected: internal MCK/8 clock signal (from PMC)
2	TIMER_CLOCK3	Clock selected: internal MCK/32 clock signal (from PMC)
3	TIMER_CLOCK4	Clock selected: internal MCK/128 clock signal (from PMC)
4	TIMER_CLOCK5	Clock selected: internal SLCK clock signal (from PMC)
5	XC0	Clock selected: XC0
6	XC1	Clock selected: XC1
7	XC2	Clock selected: XC2

40.7.3 TC Channel Mode Register: Waveform Mode

Name: TC_CMRx
Offset: 0x04 + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	BSWTRG[1:0]		BEEVT[1:0]		BCPC[1:0]		BCPB[1:0]	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ASWTRG[1:0]		AEEVT[1:0]		ACPC[1:0]		ACPA[1:0]	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WAVE	WAVSEL[1:0]		ENETRG	EEVT[1:0]		EEVTEDG[1:0]	
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CPCDIS	CPCSTOP	BURST[1:0]		CLKI	TCCLKS[2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:30 – BSWTRG[1:0] Software Trigger Effect on TIOBx

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

Bits 29:28 – BEEVT[1:0] External Event Effect on TIOBx

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

Bits 27:26 – BCPC[1:0] RC Compare Effect on TIOBx

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

Bits 25:24 – BCPB[1:0] RB Compare Effect on TIOBx

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

Bits 23:22 – ASWTRG[1:0] Software Trigger Effect on TIOAx

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

Bits 21:20 – AEEVT[1:0] External Event Effect on TIOAx

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

Bits 19:18 – ACPC[1:0] RC Compare Effect on TIOAx

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

Bits 17:16 – ACPA[1:0] RA Compare Effect on TIOAx

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

Bit 15 – WAVE Waveform Mode

Value	Description
0	Waveform mode is disabled (Capture mode is enabled).
1	Waveform mode is enabled.

Bits 14:13 – WAVSEL[1:0] Waveform Selection

Value	Name	Description
0	UP	UP mode without automatic trigger on RC Compare
1	UPDOWN	UPDOWN mode without automatic trigger on RC Compare
2	UP_RC	UP mode with automatic trigger on RC Compare
3	UPDOWN_RC	UPDOWN mode with automatic trigger on RC Compare

Bit 12 – ENETRG External Event Trigger Enable
Whatever the value programmed in ENETRG, the selected external event only controls the TIOAx output and TIOBx if not used as input (trigger event input or other input used).

Value	Description
0	The external event has no effect on the counter and its clock.
1	The external event resets the counter and starts the counter clock.

Bits 11:10 – EEVT[1:0] External Event Selection
Signal selected as external event.

Value	Name	Description	TIOB Direction
0	TIOB	TIOB	Input
1	XC0	XC0	Output
2	XC1	XC1	Output

.....continued			
Value	**Name**	**Description**	**TIOB Direction**
3	XC2	XC2	Output

Note: If TIOB is chosen as the external event signal, it is configured as an input and no longer generates waveforms and subsequently no IRQs.

Bits 9:8 – EEVTEDG[1:0] External Event Edge Selection

Value	Name	Description
0	NONE	None
1	RISING	Rising edge
2	FALLING	Falling edge
3	EDGE	Each edge

Bit 7 – CPCDIS Counter Clock Disable with RC Compare

Value	Description
0	Counter clock is not disabled when counter reaches RC.
1	Counter clock is disabled when counter reaches RC.

Bit 6 – CPCSTOP Counter Clock Stopped with RC Compare

Value	Description
0	Counter clock is not stopped when counter reaches RC.
1	Counter clock is stopped when counter reaches RC.

Bits 5:4 – BURST[1:0] Burst Signal Selection

Value	Name	Description
0	NONE	The clock is not gated by an external signal.
1	XC0	XC0 is ANDed with the selected clock.
2	XC1	XC1 is ANDed with the selected clock.
3	XC2	XC2 is ANDed with the selected clock.

Bit 3 – CLKI Clock Invert

Value	Description
0	Counter is incremented on rising edge of the clock.
1	Counter is incremented on falling edge of the clock.

Bits 2:0 – TCCLKS[2:0] Clock Selection

To operate at maximum peripheral clock frequency, refer to “TC Extended Mode Register”.

Value	Name	Description
0	TIMER_CLOCK1	Clock selected: internal GCLK clock signal (from PMC)
1	TIMER_CLOCK2	Clock selected: internal MCK/8 clock signal (from PMC)
2	TIMER_CLOCK3	Clock selected: internal MCK/32 clock signal (from PMC)
3	TIMER_CLOCK4	Clock selected: internal MCK/128 clock signal (from PMC)
4	TIMER_CLOCK5	Clock selected: internal SLCK clock signal (from PMC)
5	XC0	Clock selected: XC0
6	XC1	Clock selected: XC1
7	XC2	Clock selected: XC2

40.7.4 TC Stepper Motor Mode Register

Name: TC_SMMRx
Offset: 0x08 + x*0x40 [x=0..2]
Reset: 0x00000000
Property: R/W

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							DOWN	GCEN
Access							R/W	R/W
Reset							0	0

Bit 1 – DOWN Down Count

Value	Description
0	Up counter.
1	Down counter.

Bit 0 – GCEN Gray Count Enable

Value	Description
0	TIOAx [x=0..2] and TIOBx [x=0..2] are driven by internal counter of channel x.
1	TIOAx [x=0..2] and TIOBx [x=0..2] are driven by a 2-bit Gray counter.

40.7.5 TC Register AB

Name: TC_RABx
Offset: 0x0C + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	RAB[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RAB[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RAB[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RAB[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RAB[31:0] Register A or Register B
RAB contains the next unread capture Register A or Register B value in real time. It is usually read by the DMA after a request due to a valid load edge on TIOAx.
When DMA is used, the RAB register address must be configured as source address of the transfer.

40.7.6 TC Counter Value Register

Name: TC_CVx
Offset: 0x10 + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	CV[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	CV[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CV[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CV[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – CV[31:0] Counter Value
CV contains the counter value in real time.

Important:
For 16-bit channels, CV field size is limited to register bits 15:0.

40.7.7 TC Register A

Name: TC_RAx
Offset: 0x14 + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read/Write

This register has access Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1.
This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	RA[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RA[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RA[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RA[31:0] Register A
RA contains the Register A value in real time.

Important:
For 16-bit channels, RA field size is limited to register bits 15:0.

40.7.8 TC Register B

Name: TC_RBx
Offset: 0x18 + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read/Write

This register has access Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1.
This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	RB[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RB[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RB[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RB[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RB[31:0] Register B
RB contains the Register B value in real time.

Important:
For 16-bit channels, RB field size is limited to register bits 15:0.

40.7.9 TC Register C

Name: TC_RCx
Offset: 0x1C + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	RC[31:24]							
Access								
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	RC[23:16]							
Access								
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RC[15:8]							
Access								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RC[7:0]							
Access								
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – RC[31:0] Register C
RC contains the Register C value in real time.

Important:
For 16-bit channels, RC field size is limited to register bits 15:0.

40.7.10 TC Interrupt Status Register

Name: TC_SRx
Offset: 0x20 + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
						MTIOB	MTIOA	CLKSTA
Access						R	R	R
Reset						0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 18 – MTIOB TIOBx Mirror

Value	Description
0	TIOBx is low. If TC_CMRx.WAVE = 0, TIOBx pin is low. If TC_CMRx.WAVE = 1, TIOBx is driven low.
1	TIOBx is high. If TC_CMRx.WAVE = 0, TIOBx pin is high. If TC_CMRx.WAVE = 1, TIOBx is driven high.

Bit 17 – MTIOA TIOAx Mirror

Value	Description
0	TIOAx is low. If TC_CMRx.WAVE = 0, TIOAx pin is low. If TC_CMRx.WAVE = 1, TIOAx is driven low.
1	TIOAx is high. If TC_CMRx.WAVE = 0, TIOAx pin is high. If TC_CMRx.WAVE = 1, TIOAx is driven high.

Bit 16 – CLKSTA Clock Enabling Status

Value	Description
0	Clock is disabled.
1	Clock is enabled.

Bit 7 – ETRGS External Trigger Status (cleared on read)

Value	Description
0	External trigger has not occurred since the last read of the Status Register.
1	External trigger has occurred since the last read of the Status Register.

Bit 6 – LDRBS RB Loading Status (cleared on read)

Value	Description
0	RB Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1.
1	RB Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0.

Bit 5 – LDRAS RA Loading Status (cleared on read)

Value	Description
0	RA Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1.
1	RA Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0.

Bit 4 – CPCS RC Compare Status (cleared on read)

Value	Description
0	RC Compare has not occurred since the last read of the Status Register.
1	RC Compare has occurred since the last read of the Status Register.

Bit 3 – CPBS RB Compare Status (cleared on read)

Value	Description
0	RB Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0.
1	RB Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1.

Bit 2 – CPAS RA Compare Status (cleared on read)

Value	Description
0	RA Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0.
1	RA Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1.

Bit 1 – LOVRS Load Overrun Status (cleared on read)

Value	Description
0	Load overrun has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1.
1	RA or RB have been loaded at least twice without any read of the corresponding register since the last read of the Status Register, if TC_CMRx.WAVE = 0.

Bit 0 – COVFS Counter Overflow Status (cleared on read)

Value	Description
0	No counter overflow has occurred since the last read of the Status Register.
1	A counter overflow has occurred since the last read of the Status Register.

40.7.11 TC Interrupt Enable Register

Name: TC_IERx
Offset: 0x24 + x*0x40 [x=0..2]
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 7 – ETRGS External Trigger

Bit 6 – LDRBS RB Loading

Bit 5 – LDRAS RA Loading

Bit 4 – CPCS RC Compare

Bit 3 – CPBS RB Compare

Bit 2 – CPAS RA Compare

Bit 1 – LOVRS Load Overrun

Bit 0 – COVFS Counter Overflow

40.7.12 TC Interrupt Disable Register

Name: TC_IDRx
Offset: 0x28 + x*0x40 [x=0..2]
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bit 7 – ETRGS External Trigger

Bit 6 – LDRBS RB Loading

Bit 5 – LDRAS RA Loading

Bit 4 – CPCS RC Compare

Bit 3 – CPBS RB Compare

Bit 2 – CPAS RA Compare

Bit 1 – LOVRS Load Overrun

Bit 0 – COVFS Counter Overflow

40.7.13 TC Interrupt Mask Register

Name: TC_IMRx
Offset: 0x2C + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 7 – ETRGS External Trigger

Bit 6 – LDRBS RB Loading

Bit 5 – LDRAS RA Loading

Bit 4 – CPCS RC Compare

Bit 3 – CPBS RB Compare

Bit 2 – CPAS RA Compare

Bit 1 – LOVRS Load Overrun

Bit 0 – COVFS Counter Overflow

40.7.14 TC Extended Mode Register

Name: TC_EMRx
Offset: 0x30 + x*0x40 [x=0..2]
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								NODIVCLK
Access								R/W
Reset								0

Bit	7	6	5	4	3	2	1	0
			TRIGSRCB[1:0]				TRIGSRCA[1:0]	
Access			R/W	R/W			R/W	R/W
Reset			0	0			0	0

Bit 8 – NODIVCLK No Divided Clock

Value	Description
0	The selected clock is defined by field TCCLKS in TC_CMRx.
1	The selected clock is peripheral clock and TCCLKS field (TC_CMRx) has no effect.

Bits 5:4 – TRIGSRCB[1:0] Trigger Source for Input B

Value	Name	Description
0	EXTERNAL_TIOBx	The trigger/capture input B is driven by external pin TIOBx
1	PWMx	For TC0 to TC10: The trigger/capture input B is driven internally by the comparator output (see Synchronization with PWM) of the PWMx. For TC11: The trigger/capture input B is driven internally by the GTSUCOMP signal of the Ethernet MAC (GMAC).

Bits 1:0 – TRIGSRCA[1:0] Trigger Source for Input A

Value	Name	Description
0	EXTERNAL_TIOAx	The trigger/capture input A is driven by external pin TIOAx
1	PWMx	The trigger/capture input A is driven internally by PWMx

40.7.15 TC Block Control Register

Name: TC_BCR
Offset: 0xC0
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								SYNC
Access								W
Reset								–

Bit 0 – SYNC Synchro Command

Value	Description
0	No effect.
1	Asserts the SYNC signal which generates a software trigger simultaneously for each of the channels.

40.7.16 TC Block Mode Register

Name: TC_BMR
Offset: 0xC4
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
							MAXFILT[5:4]	
Access							R/W	R/W
Reset							0	0

Bit	23	22	21	20	19	18	17	16
	MAXFILT[3:0]						IDXPHB	SWAP
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	0	0	0	0			0	0

Bit	15	14	13	12	11	10	9	8
	INVIDX	INVB	INVA	EDGPHA	QDTRANS	SPEEDEN	POSEN	QDEN
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
			TC2XC2S[1:0]		TC1XC1S[1:0]		TC0XC0S[1:0]	
Access			R/W	R/W	R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bits 25:20 – MAXFILT[5:0] Maximum Filter
Pulses with a period shorter than MAXFILT+1 peripheral clock cycles are discarded. For more details on MAXFILT constraints, see "Input Preprocessing"

Value	Description
1-63	Defines the filtering capabilities.

Bit 17 – IDXPHB Index Pin is PHB Pin

Value	Description
0	IDX pin of the rotary sensor must drive TIOA1.
1	IDX pin of the rotary sensor must drive TIOB0.

Bit 16 – SWAP Swap PHA and PHB

Value	Description
0	No swap between PHA and PHB.
1	Swap PHA and PHB internally, prior to driving the QDEC.

Bit 15 – INVIDX Inverted Index

Value	Description
0	IDX (TIOA1) is directly driving the QDEC.
1	IDX is inverted before driving the QDEC.

Bit 14 – INVB Inverted PHB

Value	Description
0	PHB (TIOB0) is directly driving the QDEC.
1	PHB is inverted before driving the QDEC.

Bit 13 – INVA Inverted PHA

Value	Description
0	PHA (TIOA0) is directly driving the QDEC.
1	PHA is inverted before driving the QDEC.

Bit 12 – EDGPHA Edge on PHA Count Mode

Value	Description
0	Edges are detected on PHA only.
1	Edges are detected on both PHA and PHB.

Bit 11 – QDTRANS Quadrature Decoding Transparent

Value	Description
0	Full quadrature decoding logic is active (direction change detected).
1	Quadrature decoding logic is inactive (direction change inactive) but input filtering and edge detection are performed.

Bit 10 – SPEEDEN Speed Enabled

Value	Description
0	Disabled.
1	Enables the speed measure on channel 0, the time base being provided by channel 2.

Bit 9 – POSEN Position Enabled

Value	Description
0	Disable position.
1	Enables the position measure on channel 0 and 1.

Bit 8 – QDEN Quadrature Decoder Enabled
Quadrature decoding (direction change) can be disabled using QDTRANS bit.
One of the POSEN or SPEEDEN bits must be also enabled.

Value	Description
0	Disabled.
1	Enables the QDEC (filter, edge detection and quadrature decoding).

Bits 5:4 – TC2XC2S[1:0] External Clock Signal 2 Selection

Value	Name	Description
0	TCLK2	Signal connected to XC2: TCLK2
1	–	Reserved
2	TIOA0	Signal connected to XC2: TIOA0
3	TIOA1	Signal connected to XC2: TIOA1

Bits 3:2 – TC1XC1S[1:0] External Clock Signal 1 Selection

Value	Name	Description
0	TCLK1	Signal connected to XC1: TCLK1
1	–	Reserved
2	TIOA0	Signal connected to XC1: TIOA0
3	TIOA2	Signal connected to XC1: TIOA2

Bits 1:0 – TC0XC0S[1:0] External Clock Signal 0 Selection

Value	Name	Description
0	TCLK0	Signal connected to XC0: TCLK0
1	–	Reserved
2	TIOA1	Signal connected to XC0: TIOA1
3	TIOA2	Signal connected to XC0: TIOA2

40.7.17 TC QDEC Interrupt Enable Register

Name: TC_QIER
Offset: 0xC8
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						QERR	DIRCHG	IDX
Access						W	W	W
Reset						–	–	–

Bit 2 – QERR Quadrature Error

Value	Description
0	No effect.
1	Enables the interrupt when a quadrature error occurs on PHA, PHB.

Bit 1 – DIRCHG Direction Change

Value	Description
0	No effect.
1	Enables the interrupt when a change on rotation direction is detected.

Bit 0 – IDX Index

Value	Description
0	No effect.
1	Enables the interrupt when a rising edge occurs on IDX input.

40.7.18 TC QDEC Interrupt Disable Register

Name: TC_QIDR
Offset: 0xCC
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						QERR	DIRCHG	IDX
Access						W	W	W
Reset						–	–	–

Bit 2 – QERR Quadrature Error

Value	Description
0	No effect.
1	Disables the interrupt when a quadrature error occurs on PHA, PHB.

Bit 1 – DIRCHG Direction Change

Value	Description
0	No effect.
1	Disables the interrupt when a change on rotation direction is detected.

Bit 0 – IDX Index

Value	Description
0	No effect.
1	Disables the interrupt when a rising edge occurs on IDX input.

40.7.19 TC QDEC Interrupt Mask Register

Name: TC_QIMR
Offset: 0xD0
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						QERR	DIRCHG	IDX
Access						R	R	R
Reset						0	0	0

Bit 2 – QERR Quadrature Error

Value	Description
0	The interrupt on quadrature error is disabled.
1	The interrupt on quadrature error is enabled.

Bit 1 – DIRCHG Direction Change

Value	Description
0	The interrupt on rotation direction change is disabled.
1	The interrupt on rotation direction change is enabled.

Bit 0 – IDX Index

Value	Description
0	The interrupt on IDX input is disabled.
1	The interrupt on IDX input is enabled.

40.7.20 TC QDEC Interrupt Status Register

Name: TC_QISR
Offset: 0xD4
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								DIR
Access								R
Reset								0

Bit	7	6	5	4	3	2	1	0
						QERR	DIRCHG	IDX
Access						R	R	R
Reset						0	0	0

Bit 8 – DIR Direction
Returns an image of the current rotation direction.

Bit 2 – QERR Quadrature Error

Value	Description
0	No quadrature error since the last read of TC_QISR.
1	A quadrature error occurred since the last read of TC_QISR.

Bit 1 – DIRCHG Direction Change

Value	Description
0	No change on rotation direction since the last read of TC_QISR.
1	The rotation direction changed since the last read of TC_QISR.

Bit 0 – IDX Index

Value	Description
0	No Index input change since the last read of TC_QISR.
1	The IDX input has changed since the last read of TC_QISR.

40.7.21 TC Fault Mode Register

Name: TC_FMR
Offset: 0xD8
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							ENCF1	ENCF0
Access							R/W	R/W
Reset							0	0

Bit 1 – ENCF1 Enable Compare Fault Channel 1

Value	Description
0	Disables the FAULT output source (CPCS flag) from channel 1.
1	Enables the FAULT output source (CPCS flag) from channel 1.

Bit 0 – ENCF0 Enable Compare Fault Channel 0

Value	Description
0	Disables the FAULT output source (CPCS flag) from channel 0.
1	Enables the FAULT output source (CPCS flag) from channel 0.

40.7.22 TC Write Protection Mode Register

Name: TC_WPMR
Offset: 0xE4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPEN
Access								R/W
Reset								0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x54494D	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

Bit 0 – WPEN Write Protection Enable
The Timer Counter clock of the first channel must be enabled to access this register.
See "Register Write Protection" for a list of registers that can be write-protected and Timer Counter clock conditions.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x54494D ("TIM" in ASCII).
1	Enables the write protection if WPKEY corresponds to 0x54494D ("TIM" in ASCII).

41. Pulse Width Modulation Controller (PWM)

41.1 Description

The Pulse Width Modulation Controller (PWM) generates output pulses on 4 channels independently according to parameters defined per channel. Each channel controls two complementary square output waveforms. Characteristics of the output waveforms such as period, duty-cycle, polarity and dead-times (also called dead-bands or non-overlapping times) are configured through the user interface. Each channel selects and uses one of the clocks provided by the clock generator. The clock generator provides several clocks resulting from the division of the PWM peripheral clock. External triggers can be managed to allow output pulses to be modified in real time.

All accesses to the PWM are made through registers mapped on the peripheral bus. All channels integrate a double buffering system in order to prevent an unexpected output waveform while modifying the period, the spread spectrum, the duty-cycle or the dead-times.

Channels can be linked together as synchronous channels to be able to update their duty-cycle or dead-times at the same time.

The update of duty-cycles of synchronous channels can be performed by the DMA Controller channel which offers buffer transfer without processor Intervention.

The PWM includes a spread-spectrum counter to allow a constantly varying period (only for Channel 0). This counter may be useful to minimize electromagnetic interference or to reduce the acoustic noise of a PWM driven motor.

The PWM provides 8 independent comparison units capable of comparing a programmed value to the counter of the synchronous channels (counter of channel 0). These comparisons are intended to generate software interrupts and to trigger DMA Controller transfer requests.

PWM outputs can be overridden synchronously or asynchronously to their channel counter.

The PWM provides a fault protection mechanism with 7 fault inputs, capable to detect a fault condition and to override the PWM outputs asynchronously (outputs forced to ‘0’, ‘1’ or Hi-Z).

For safety usage, some configuration registers are write-protected.

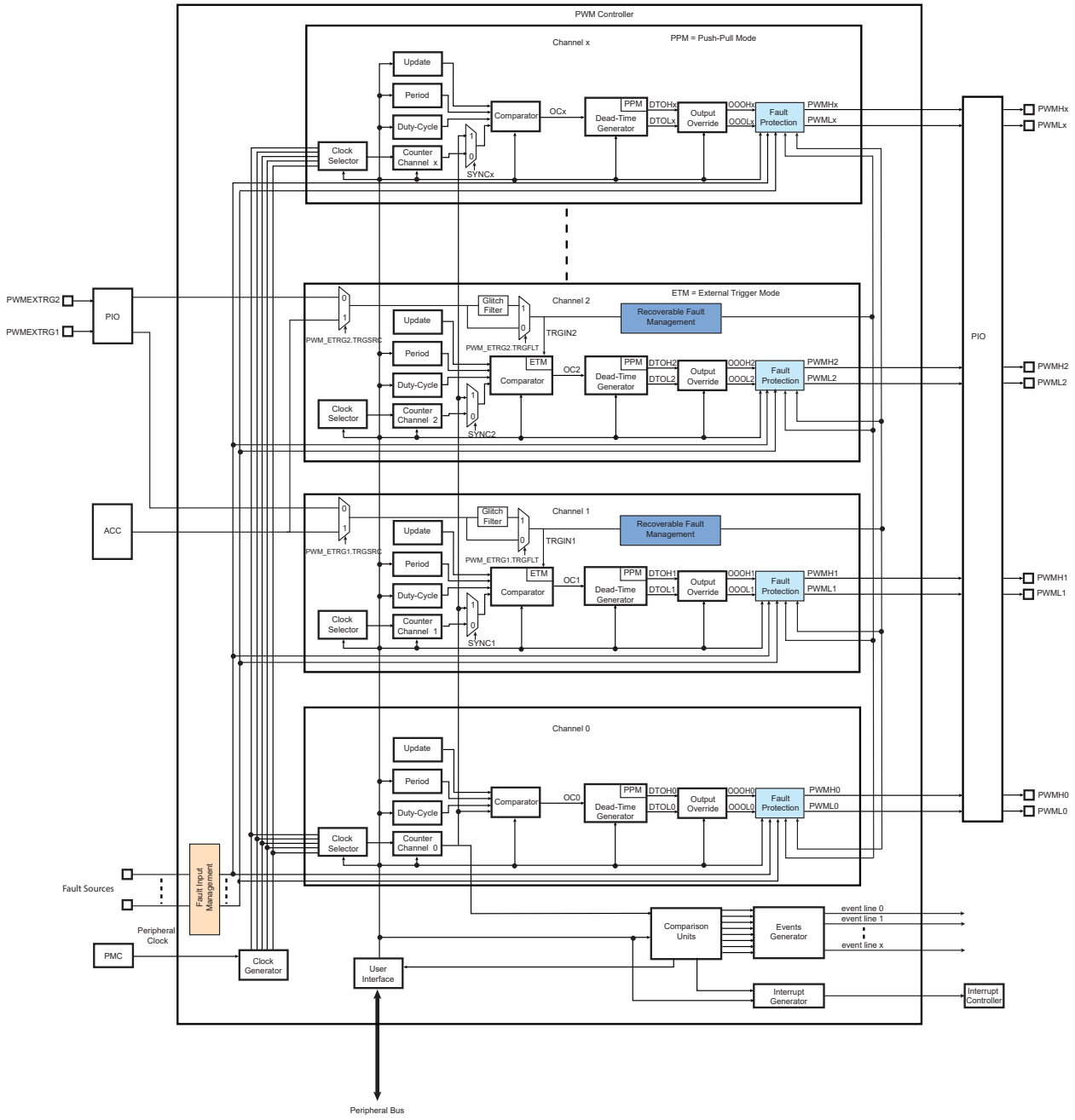
41.2 Embedded Characteristics

- 4 Channels
- Common Clock Generator Providing Thirteen Different Clocks
  - A Modulo n Counter Providing Eleven Clocks
  - Two Independent Linear Dividers Working on Modulo n Counter Outputs
- Independent Channels
  - Independent 16-bit Counter for Each Channel
  - Independent Complementary Outputs with 12-bit Dead-Time Generator (Also Called Dead-Band or Non-Overlapping Time) for Each Channel
  - Independent Push-Pull Mode for Each Channel
  - Independent Enable Disable Command for Each Channel
  - Independent Clock Selection for Each Channel
  - Independent Period, Duty-Cycle and Dead-Time for Each Channel
  - Independent Double Buffering of Period, Duty-Cycle and Dead-Times for Each Channel
  - Independent Programmable Selection of The Output Waveform Polarity for Each Channel, with Double Buffering
  - Independent Programmable Center- or Left-aligned Output Waveform for Each Channel
  - Independent Output Override for Each Channel
  - Independent Interrupt for Each Channel, at Each Period for Left-Aligned or Center-Aligned Configuration

  - Independent Update Time Selection of Double Buffering Registers (Polarity, Duty Cycle) for Each Channel, at Each Period for Left-Aligned or Center-Aligned Configuration
- External Trigger Input Management (e.g., for DC/DC or Lighting Control)
  - External PWM Reset Mode
  - External PWM Start Mode
  - Cycle-By-Cycle Duty Cycle Mode
  - Leading-Edge Blanking
- 2 2-bit Gray Up/Down Channels for Stepper Motor Control
- Spread Spectrum Counter to Allow a Constantly Varying Duty Cycle (only for Channel 0)
- Synchronous Channel Mode
  - Synchronous Channels Share the Same Counter
  - Mode to Update the Synchronous Channels Registers after a Programmable Number of Periods
  - Synchronous Channels Supports Connection of one DMA Controller Channel Which Offers Buffer Transfer Without Processor Intervention To Update Duty-Cycle Registers
- 2 Independent Events Lines
- 8 Comparison Units Intended to Generate Interrupts, Pulses on Event Lines and DMA Controller Transfer Requests
- 7 Programmable Fault Inputs Providing an Asynchronous Protection of PWM Outputs
  - 3 User Driven through PIO Inputs
  - PMC Driven when Crystal Oscillator Clock Fails
  - Timer/Counter Driven through Configurable Comparison Function
- Register Write Protection

41.3 Block Diagram

Figure 41-1. Pulse Width Modulation Controller Block Diagram



Note: For a more detailed illustration of the fault protection circuitry, refer to 41.6.2.7 Fault Protection.

41.4 I/O Lines Description

Each channel outputs two complementary external I/O lines.

Table 41-1. I/O Line Description

Name	Description	Type
PWMHx	PWM Waveform Output High for channel x	Output
PWMLx	PWM Waveform Output Low for channel x	Output
PWMFIx	PWM Fault Input x	Input
PWMEXTRGy	PWM Trigger Input y	Input

41.5 Product Dependencies

41.5.1 I/O Lines

The pins used for interfacing the PWM are multiplexed with PIO lines. The programmer must first program the PIO controller to assign the desired PWM pins to their peripheral function. If I/O lines of the PWM are not used by the application, they can be used for other purposes by the PIO controller.

All of the PWM outputs may or may not be enabled. If an application requires only four channels, then only four PIO lines are assigned to PWM outputs.

41.5.2 Power Management

The PWM is not continuously clocked. The programmer must first enable the PWM clock in the Power Management Controller (PMC) before using the PWM. However, if the application does not require PWM operations, the PWM clock can be stopped when not needed and be restarted later. In this case, the PWM will resume its operations where it left off.

41.5.3 Interrupt Sources

The PWM interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the PWM interrupt requires the Interrupt Controller to be programmed first.

41.5.4 Fault Inputs

The PWM has the fault inputs connected to the different modules. Refer to the implementation of these modules within the product for detailed information about the fault generation procedure. The PWM receives faults from:

- PIO inputs
- PMC
- Timer/Counters

Table 41-2. Fault Inputs

Fault Generator	External PWM Fault Input Number	Polarity Level(1)	Fault Input ID
PWM0			
PIO PA8	PWMC0_PWMFI0	User-defined	0
PIO PA9	PWMC0_PWMFI01	User-defined	1
PIO PA10	PWMC0_PWMFI02	User-defined	2
Main Osc (PMC)	–	To be configured to 1	3
TC0	–	To be configured to 1	6
PWM1			
PIO PB27	PWMC1_PWMFI0	User-defined	0

.....continued

Fault Generator	External PWM Fault Input Number	Polarity Level(1)	Fault Input ID
PIO PA28	PWMC1_PWMFI01	User-defined	1
PIO PA29	PWMC1_PWMFI02	User-defined	2
Main Osc (PMC)	–	To be configured to 1	3
TC0	–	To be configured to 1	6

Note:
1. FPOL field in PWMC_FMR.

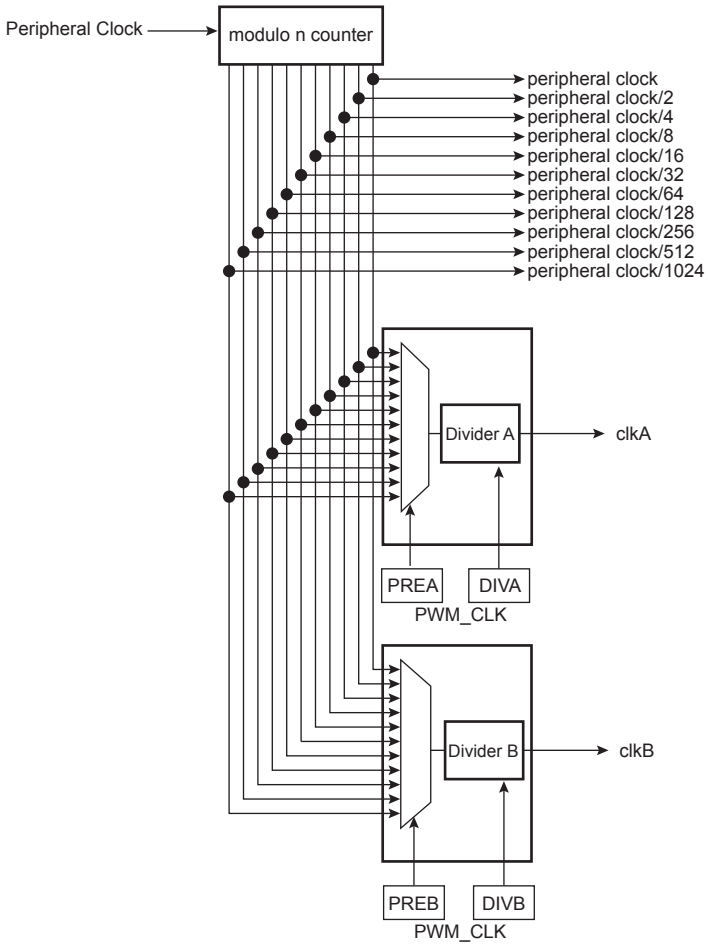
41.6 Functional Description

The PWM controller is primarily composed of a clock generator module and 4 channels.

- Clocked by the peripheral clock, the clock generator module provides 13 clocks.
- Each channel can independently choose one of the clock generator outputs.
- Each channel generates an output waveform with attributes that can be defined independently for each channel through the user interface registers.

41.6.1 PWM Clock Generator

Figure 41-2. Functional View of the Clock Generator Block Diagram



The PWM peripheral clock is divided in the clock generator module to provide different clocks available for all channels. Each channel can independently select one of the divided clocks.

The clock generator is divided into different blocks:

- a modulo n counter which provides 11 clocks:
  - $f_{peripheral\ clock}$
  - $f_{peripheral\ clock}/2$
  - $f_{peripheral\ clock}/4$
  - $f_{peripheral\ clock}/8$
  - $f_{peripheral\ clock}/16$
  - $f_{peripheral\ clock}/32$
  - $f_{peripheral\ clock}/64$
  - $f_{peripheral\ clock}/128$
  - $f_{peripheral\ clock}/256$
  - $f_{peripheral\ clock}/512$
  - $f_{peripheral\ clock}/1024$
- two linear dividers (1, 1/2, 1/3, ... 1/255) that provide two separate clocks: clkA and clkB

Each linear divider can independently divide one of the clocks of the modulo n counter. The selection of the clock to be divided is made according to the PREA (PREB) field of the PWM Clock register (PWM_CLK). The resulting clock clkA (clkB) is the clock selected divided by DIVA (DIVB) field value.

After a reset of the PWM controller, DIVA (DIVB) and PREA (PREB) are set to ‘0’. This implies that after reset clkA (clkB) are turned off.

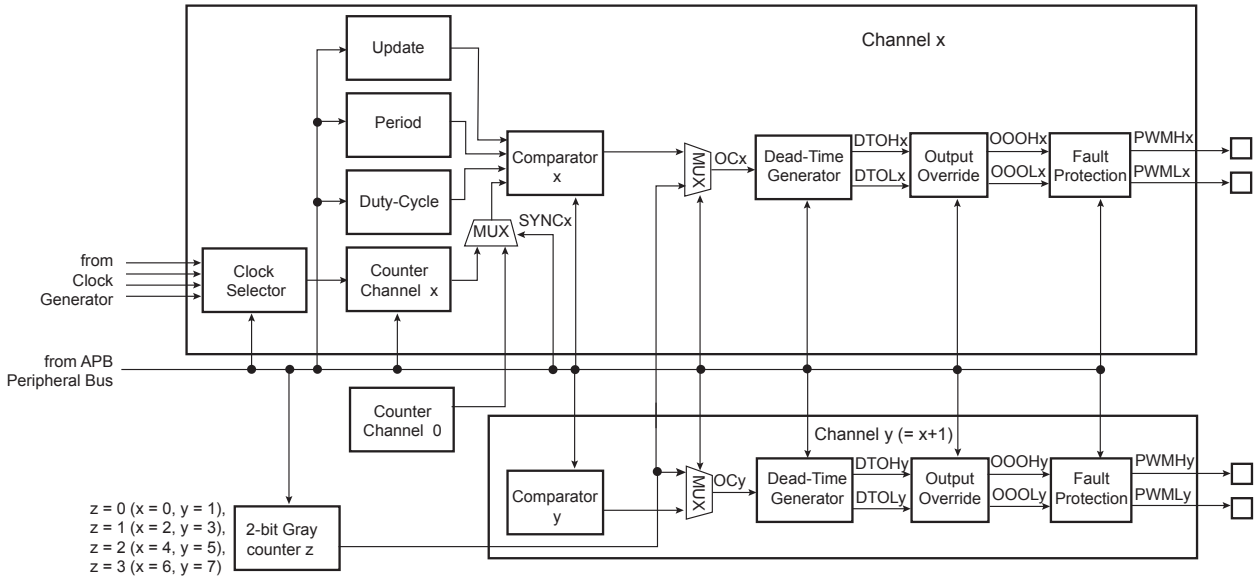
At reset, all clocks provided by the modulo n counter are turned off except the peripheral clock. This situation is also true when the PWM peripheral clock is turned off through the Power Management Controller.

⚠ CAUTION Before using the PWM controller, the programmer must first enable the peripheral clock in the Power Management Controller (PMC).

41.6.2 PWM Channel

41.6.2.1 Channel Block Diagram

Figure 41-3. Functional View of the Channel Block Diagram



Each of the 4 channels is composed of six blocks:

- A clock selector which selects one of the clocks provided by the clock generator (described in PWM Clock Generator).
- A counter clocked by the output of the clock selector. This counter is incremented or decremented according to the channel configuration and comparators matches. The size of the counter is 16 bits.
- A comparator used to compute the OCx output waveform according to the counter value and the configuration. The counter value can be the one of the channel counter or the one of the channel 0 counter according to SYNCx bit in the PWM Sync Channels Mode Register (PWM_SCM).
- A 2-bit configurable Gray counter enables the stepper motor driver. One Gray counter drives 2 channels.
- A dead-time generator providing two complementary outputs (DTOHx/DTOLx) which allows to drive external power control switches safely.
- An output override block that can force the two complementary outputs to a programmed value (OOOHx/OOOLx).
- An asynchronous fault protection mechanism that has the highest priority to override the two complementary outputs (PWMHx/PWMLx) in case of fault detection (outputs forced to ‘0’, ‘1’ or Hi-Z).

41.6.2.2 Comparator

The comparator continuously compares its counter value with the channel period defined by CPRD in the PWM Channel Period Register (PWM_CPRDx) and the duty-cycle defined by CDTY in the PWM Channel Duty Cycle Register (PWM_CDTYx) to generate an output signal OCx accordingly.

The different properties of the waveform of the output OCx are:

- the clock selection. The channel counter is clocked by one of the clocks provided by the clock generator described in the previous section. This channel parameter is defined in the CPRE field of the PWM Channel Mode Register (PWM_CMRx). This field is reset at ‘0’.
- the waveform period. This channel parameter is defined in the CPRD field of the PWM_CPRDx register.
  If the waveform is left-aligned, then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

  $$\frac{(X \times \mathrm{CPRD})}{f_{\mathrm{peripheral\ clock}}}$$

  By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

  $$\frac{(X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

  If the waveform is center-aligned, then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

  $$\frac{(2 \times X \times \mathrm{CPRD})}{f_{\mathrm{peripheral\ clock}}}$$

  By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

  $$\frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or}$$

  $$\frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

- the waveform duty-cycle. This channel parameter is defined in the CDTY field of the PWM_CDTYx register. If the waveform is left-aligned, then:

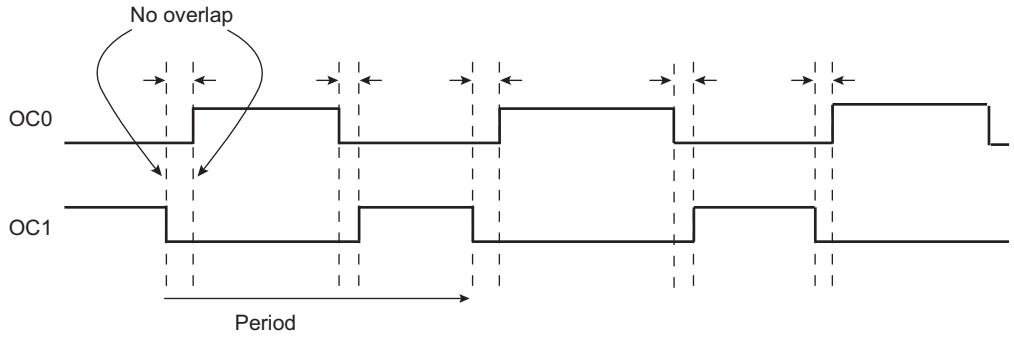
$$\text{duty cycle} = \frac{(\text{period} - 1/\text{fchannel\_x\_clock} \times \text{CDTY})}{\text{period}}$$

If the waveform is center-aligned, then:

$$\text{duty cycle} = \frac{((\text{period}/2) - 1/\text{fchannel\_x\_clock} \times \text{CDTY}))}{(\text{period}/2)}$$

- the waveform polarity. At the beginning of the period, the signal can be at high or low level. This property is defined in the CPOL bit of PWM_CMRx. By default, the signal starts by a low level. The DPOLI bit in PWM_CMRx defines the PWM polarity when the channel is disabled (CHIDx = 0 in PWM_SR). For more details, see the figure *Waveform Properties.*
  - DPOLI = 0: PWM polarity when the channel is disabled is the same as the one defined for the beginning of the PWM period.
  - DPOLI = 1: PWM polarity when the channel is disabled is inverted compared to the one defined for the beginning of the PWM period.
- the waveform alignment. The output waveform can be left- or center-aligned. Center-aligned waveforms can be used to generate non-overlapped waveforms. This property is defined in the CALG bit of PWM_CMRx. The default mode is left-aligned.

Figure 41-4. Non-Overlapped Center-Aligned Waveforms



Note: See the figure Figure 41-5 for a detailed description of center-aligned waveforms.

When center-aligned, the channel counter increases up to CPRD and decreases down to 0. This ends the period.

When left-aligned, the channel counter increases up to CPRD and is reset. This ends the period.

Thus, for the same CPRD value, the period for a center-aligned channel is twice the period for a left-aligned channel.

Waveforms are fixed at 0 when:

- CDTY = CPRD and CPOL = 0 (Note that if TRGMODE = MODE3, the PWM waveform switches to 1 at the external trigger event (see Cycle-By-Cycle Duty Mode)).
- CDTY = 0 and CPOL = 1

Waveforms are fixed at 1 (once the channel is enabled) when:

- CDTY = 0 and CPOL = 0
- CDTY = CPRD and CPOL = 1 (Note that if TRGMODE = MODE3, the PWM waveform switches to 0 at the external trigger event (see Cycle-By-Cycle Duty Mode)).

The waveform polarity must be set before enabling the channel. This immediately affects the channel output level.

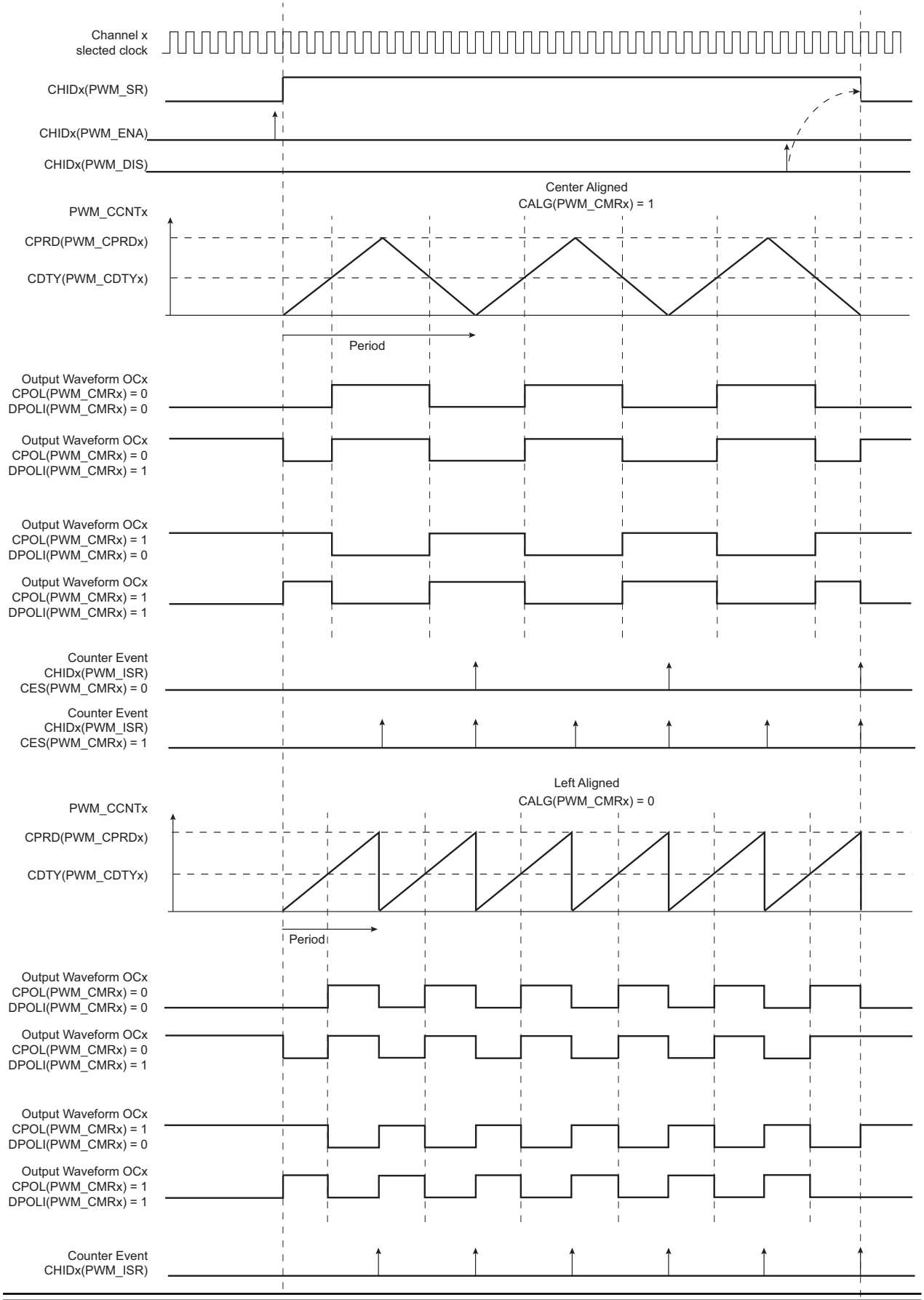
Modifying CPOL in PWM Channel Mode Register while the channel is enabled can lead to an unexpected behavior of the device being driven by PWM.

In addition to generating the output signals OCx, the comparator generates interrupts depending on the counter value. When the output waveform is left-aligned, the interrupt occurs at the end of the counter period. When the output waveform is center-aligned, the bit CES of PWM_CMRx defines when the channel counter interrupt occurs. If

CES is set to ‘0’, the interrupt occurs at the end of the counter period. If CES is set to ‘1’, the interrupt occurs at the end of the counter period and at half of the counter period.

The figure below illustrates the counter interrupts depending on the configuration.

Figure 41-5. Waveform Properties



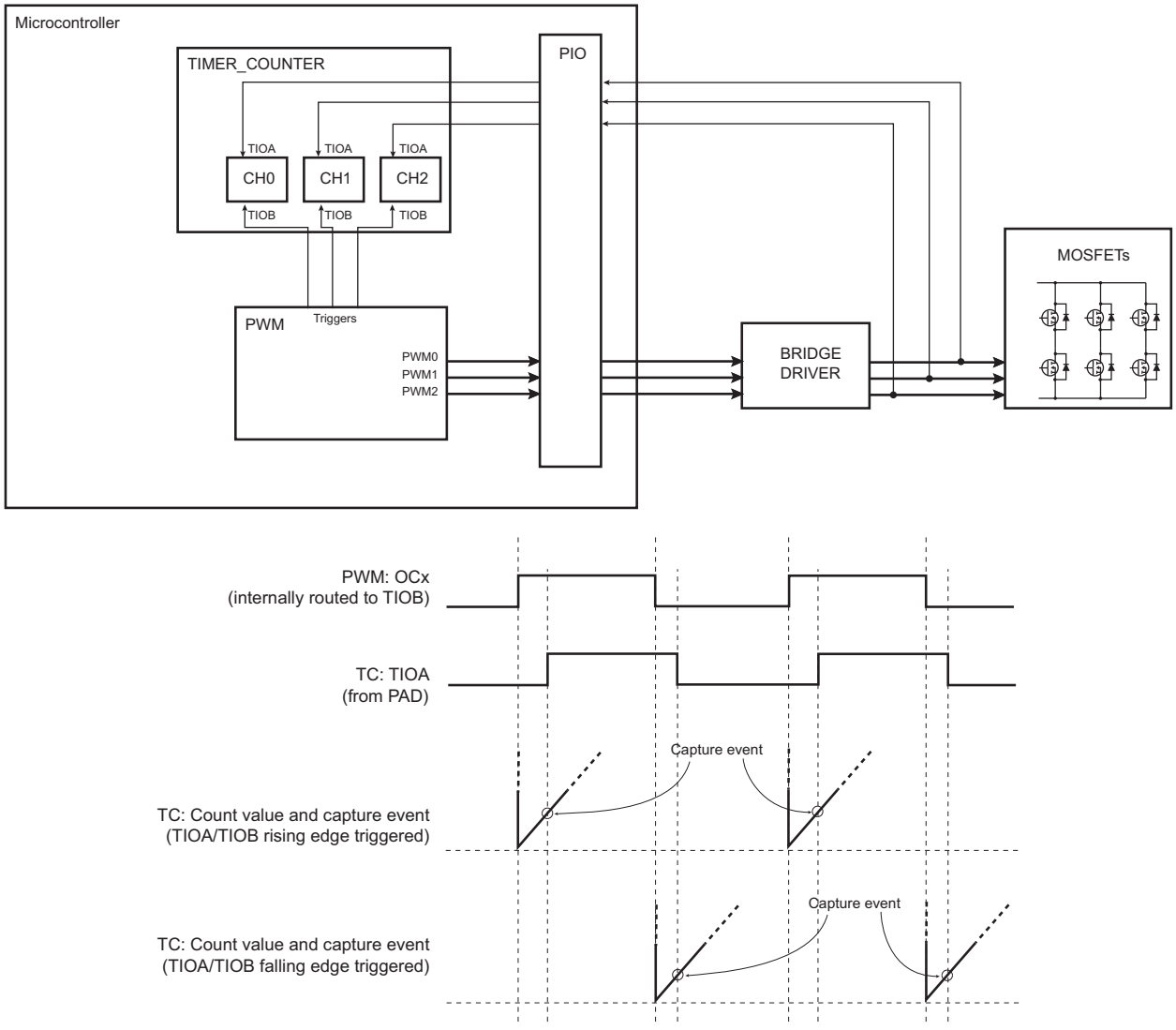
41.6.2.3 Trigger Selection for Timer Counter

The PWM controller can be used as a trigger source for the Timer Counter (TC) to achieve the two application examples described below.

41.6.2.3.1 Delay Measurement

To measure the delay between the channel x comparator output (OCx) and the feedback from the bridge driver of the MOSFETs (see the figure below), the bit TCTS in the PWM Channel Mode Register must be at 0. This defines the comparator output of the channel x as the TC trigger source. The TIOB trigger (TC internal input) is used to start the TC; the TIOA input (from PAD) is used to capture the delay.

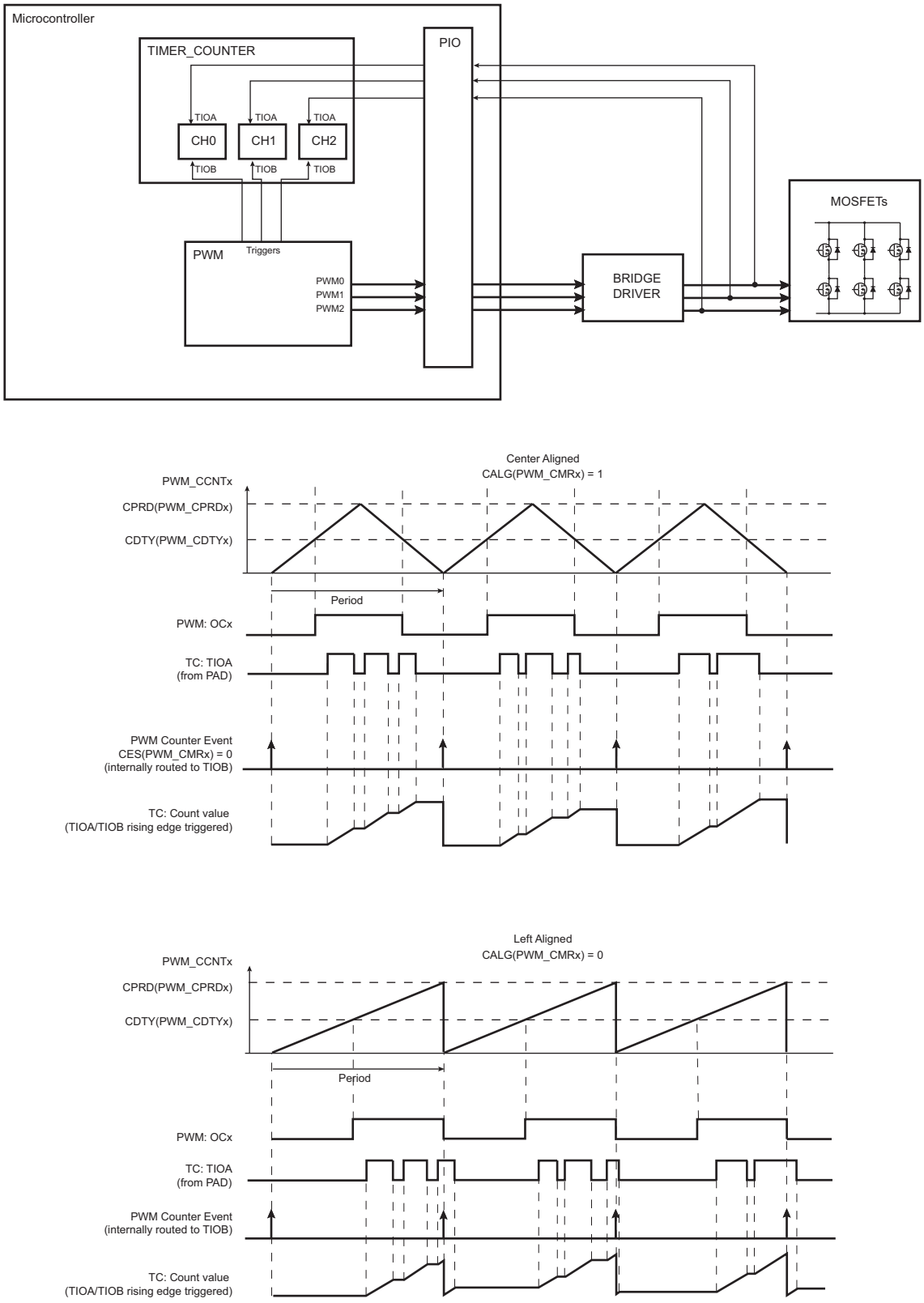
Figure 41-6. Triggering the TC: Delay Measurement



41.6.2.3.2 Cumulated ON Time Measurement

To measure the cumulated “ON” time of MOSFETs (see the figure below), the bit TCTS of the PWM Channel Mode Register must be set to 1 to define the counter event (see the figure *Waveform Properties*) as the Timer Counter trigger source.

Figure 41-7. Triggering the TC: Cumulated "ON" Time Measurement



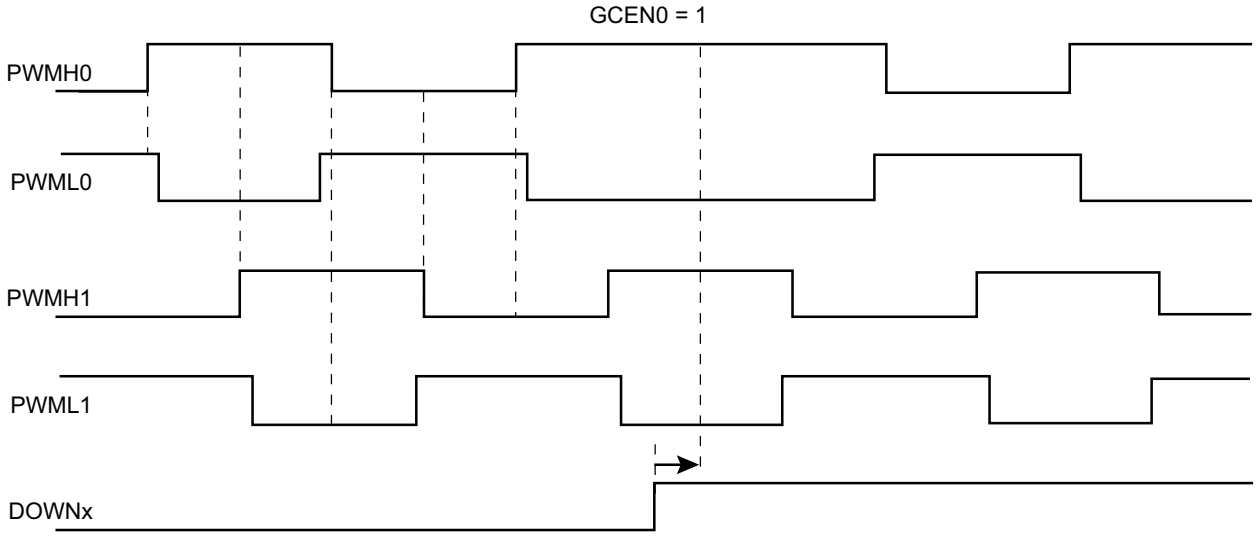
41.6.2.4 2-bit Gray Up/Down Counter for Stepper Motor

A pair of channels may provide a 2-bit Gray count waveform on two outputs. Dead-time generator and other downstream logic can be configured on these channels.

Up or Down Count mode can be configured on-the-fly by means of PWM_SMMR configuration registers.

When GCEN0 is set to ‘1’, channels 0 and 1 outputs are driven with a Gray counter.

Figure 41-8. 2-bit Gray Up/Down Counter



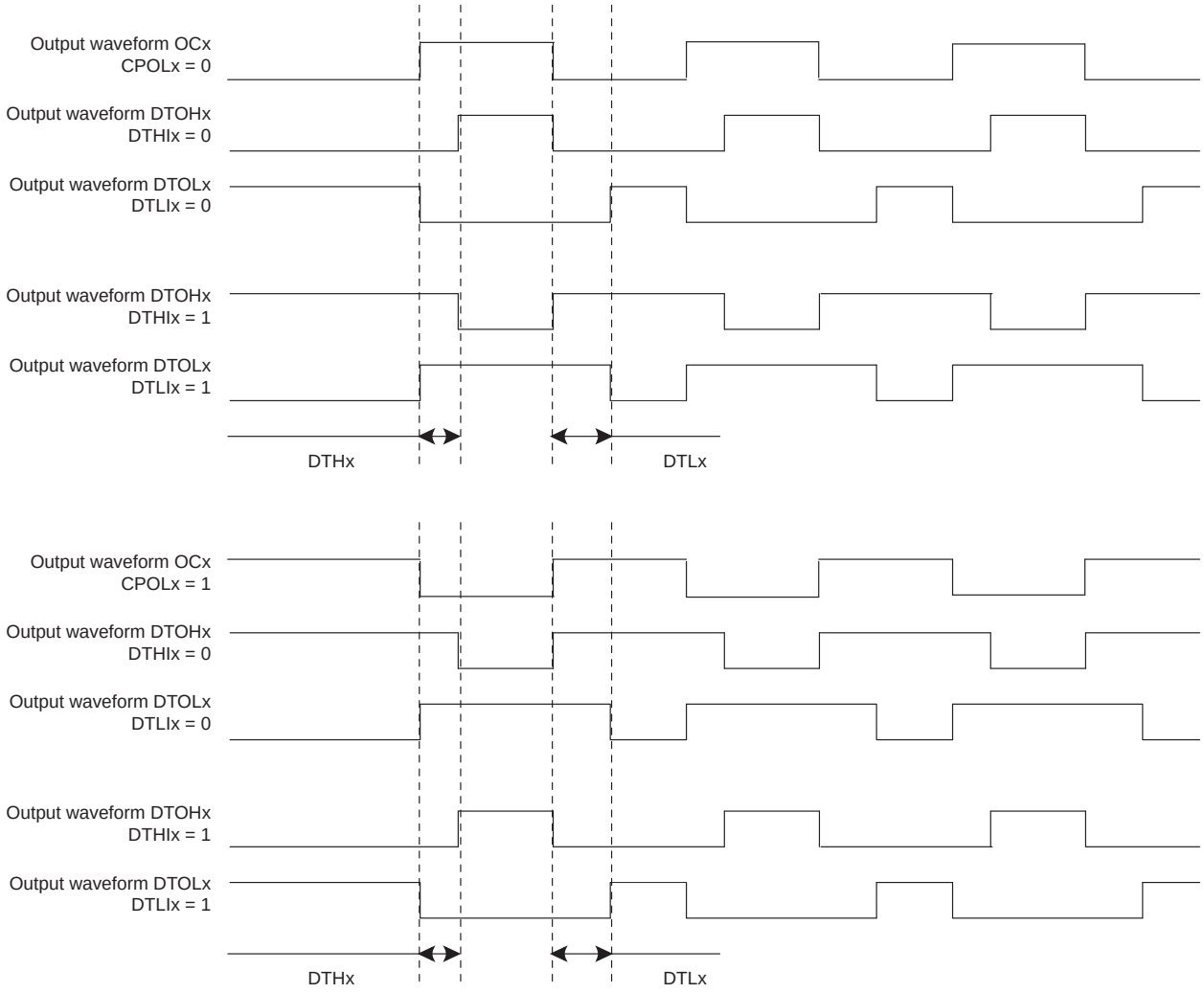
41.6.2.5 Dead-Time Generator

The dead-time generator uses the comparator output OCx to provide the two complementary outputs DTOHx and DTOLx, which allows the PWM macrocell to drive external power control switches safely. When the dead-time generator is enabled by setting the bit DTE to 1 or 0 in the PWM Channel Mode Register (PWM_CMRx), dead-times (also called dead-bands or non-overlapping times) are inserted between the edges of the two complementary outputs DTOHx and DTOLx. Note that enabling or disabling the dead-time generator is allowed only if the channel is disabled.

The dead-time is adjustable by the PWM Channel Dead Time Register (PWM_DTx). Each output of the dead-time generator can be adjusted separately by DTH and DTL. The dead-time values can be updated synchronously to the PWM period by using the PWM Channel Dead Time Update Register (PWM_DTUPDx).

The dead-time is based on a specific counter which uses the same selected clock that feeds the channel counter of the comparator. Depending on the edge and the configuration of the dead-time, DTOHx and DTOLx are delayed until the counter has reached the value defined by DTH or DTL. An inverted configuration bit (DTHI and DTLI bit in PWM_CMRx) is provided for each output to invert the dead-time outputs. The following figure shows the waveform of the dead-time generator.

Figure 41-9. Complementary Output Waveforms



41.6.2.5.1 PWM Push-Pull Mode

When a PWM channel is configured in Push-Pull mode, the dead-time generator output is managed alternately on each PWM cycle. The polarity of the PWM line during the idle state of the Push-Pull mode is defined by the DPOLI bit in the PWM Channel Mode Register (PWM_CMRx). The Push-Pull mode can be enabled separately on each channel by writing a one to bit PPM in the PWM Channel Mode Register.

Figure 41-10. PWM Push-Pull Mode

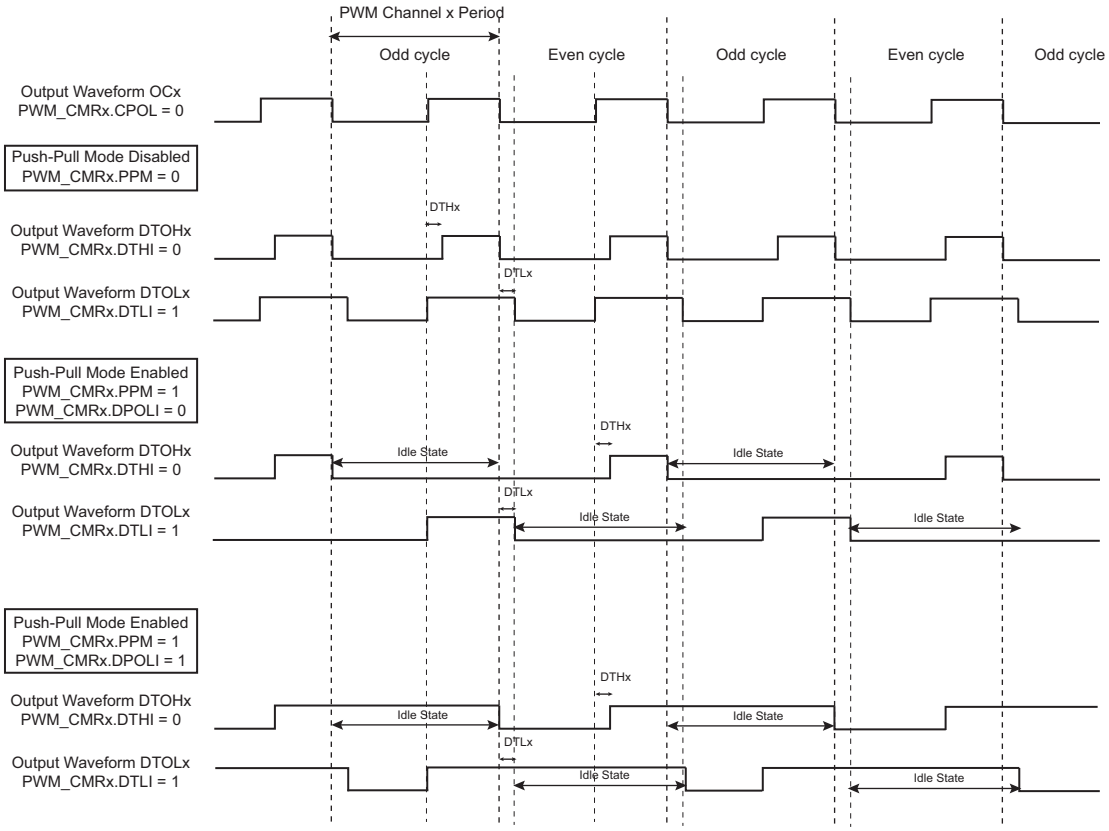


Figure 41-11. PWM Push-Pull Waveforms: Left-Aligned Mode

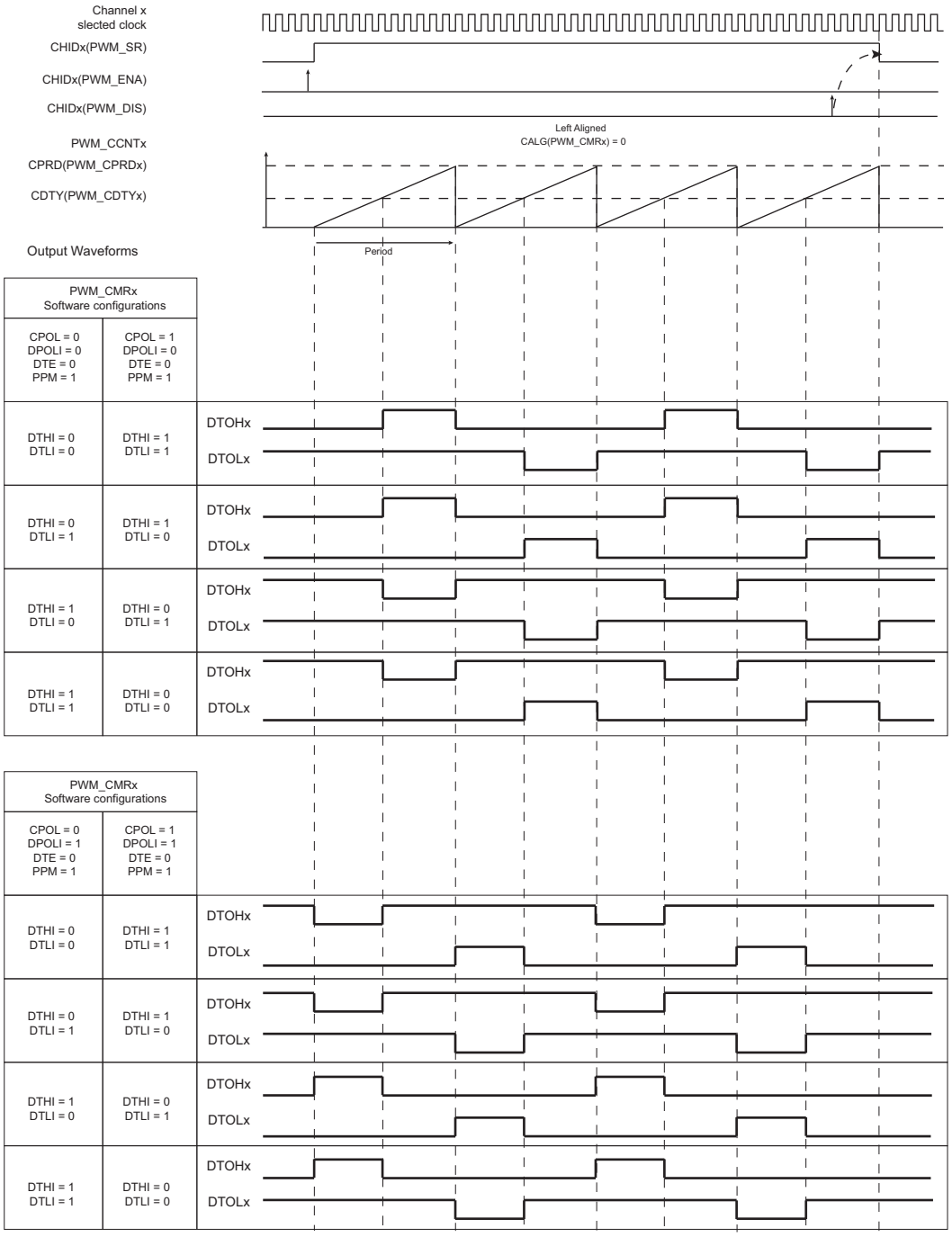
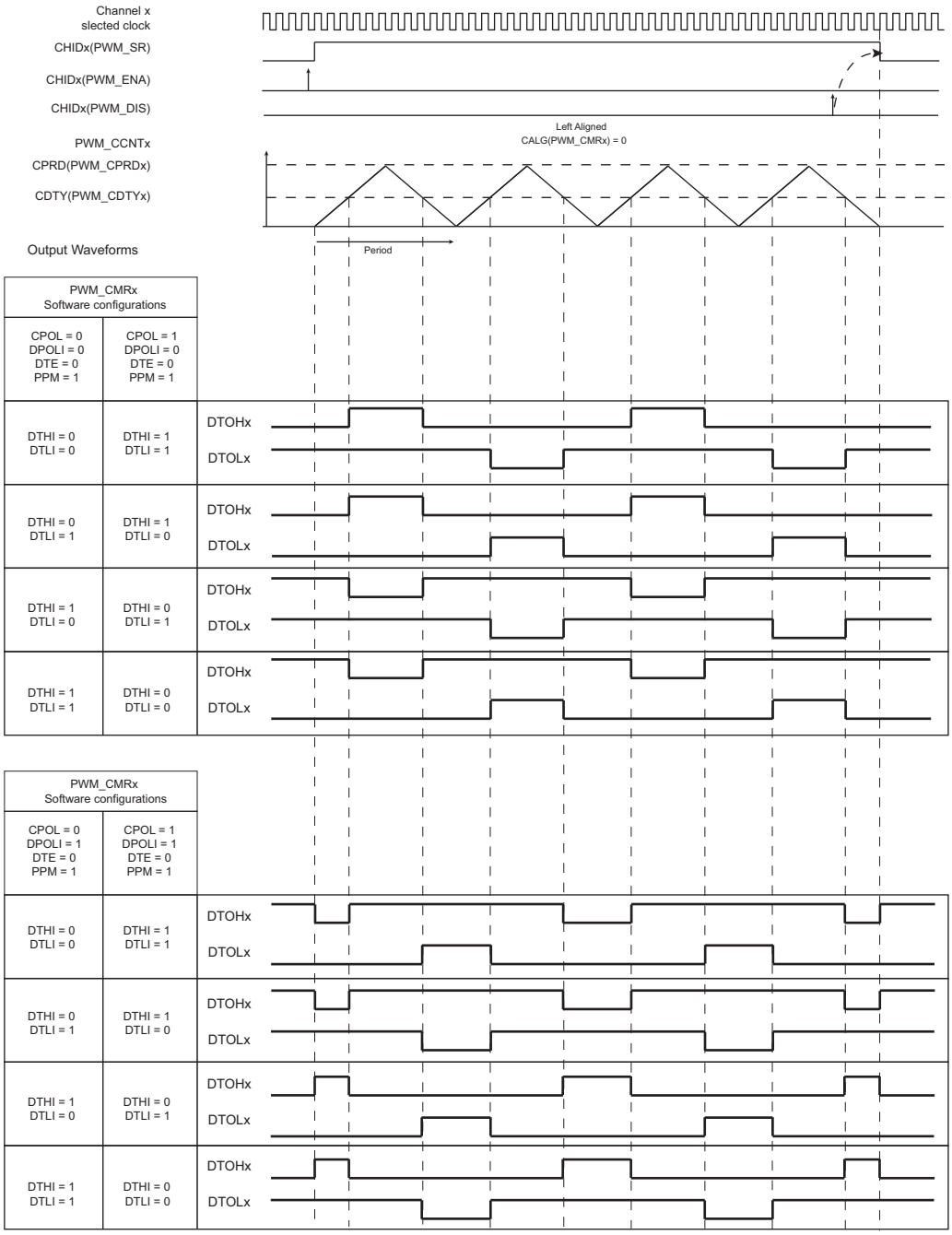
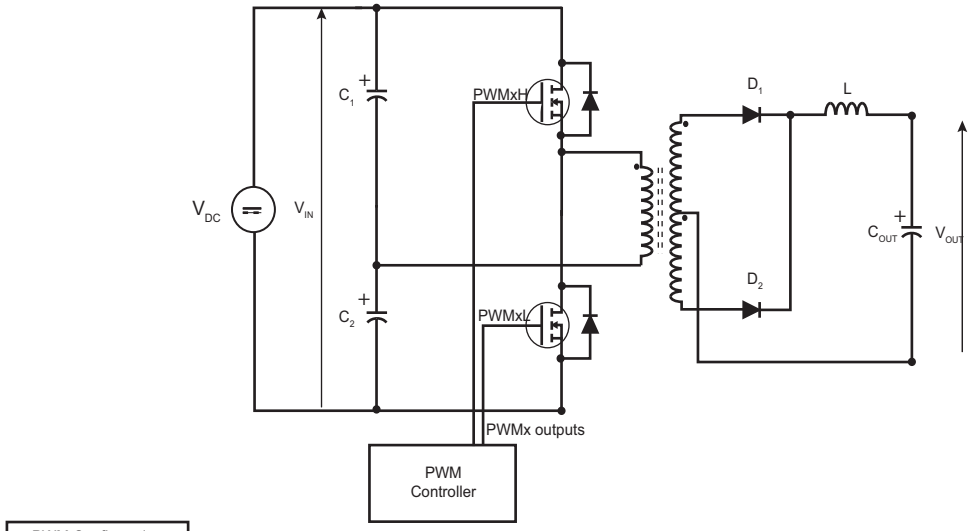


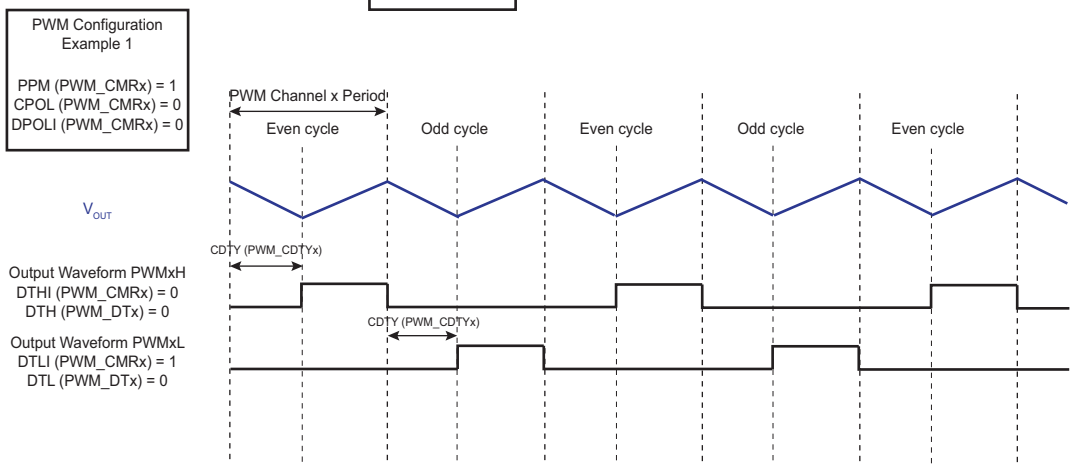
Figure 41-12. PWM Push-Pull Waveforms: Center-Aligned Mode



The PWM Push-Pull mode can be useful in transformer-based power converters, such as a half-bridge converter. The Push-Pull mode prevents the transformer core from being saturated by any direct current.

Figure 41-13. Half-Bridge Converter Application: No Feedback Regulation





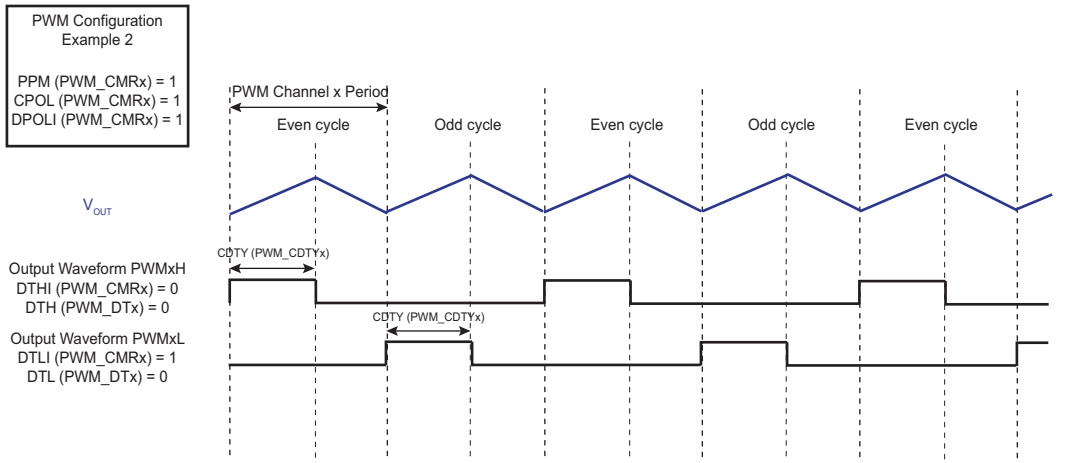
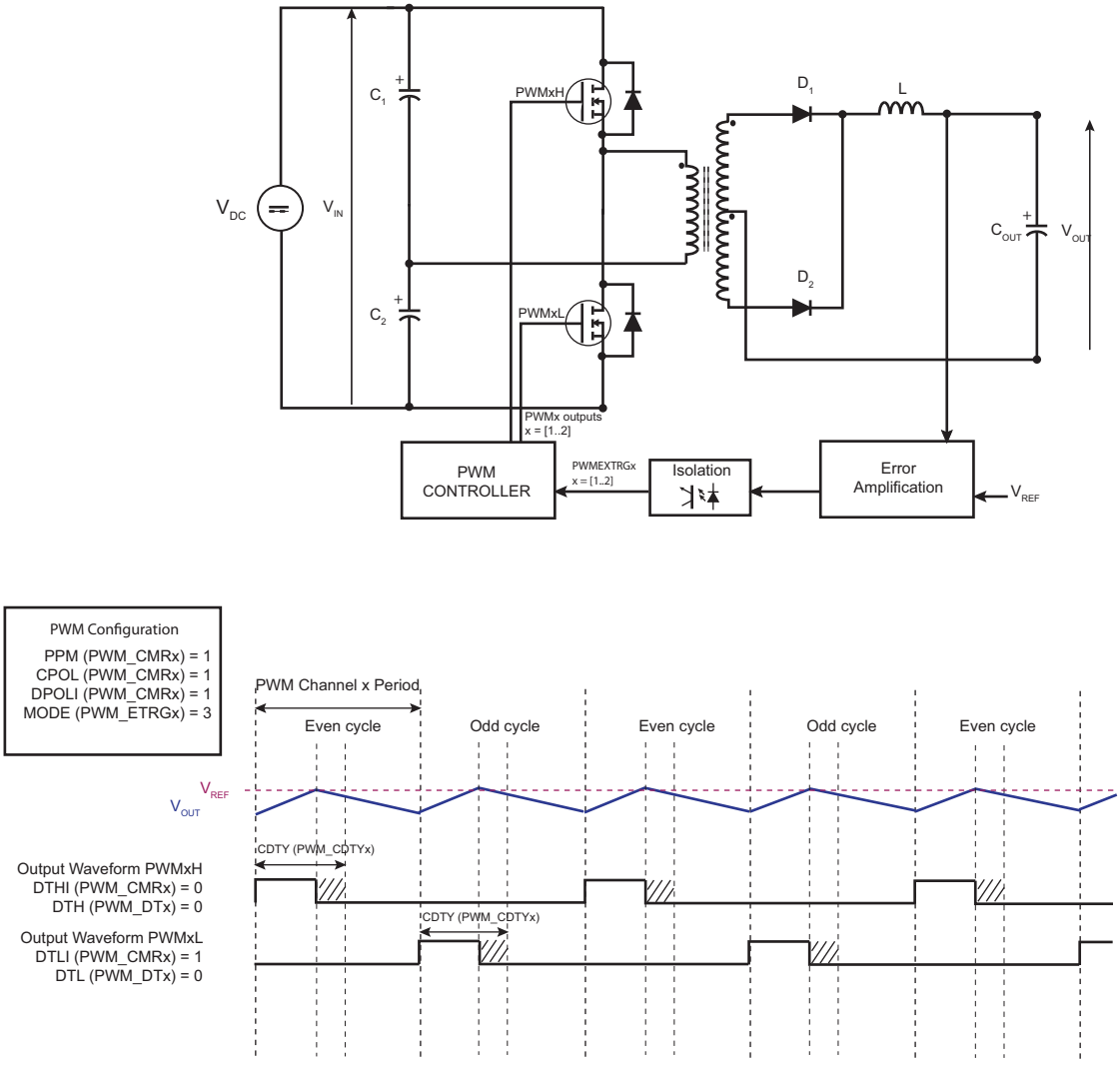


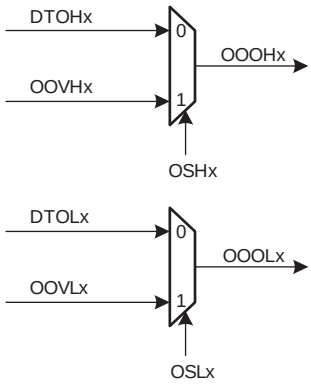
Figure 41-14. Half-Bridge Converter Application: Feedback Regulation



41.6.2.6 Output Override

The two complementary outputs DTOHx and DTOLx of the dead-time generator can be forced to a value defined by the software.

Figure 41-15. Override Output Selection



The fields OSHx and OSLx in the PWM Output Selection Register (PWM_OS) allow the outputs of the dead-time generator DTOHx and DTOLx to be overridden by the value defined in the fields OOVHx and OOVLx in the PWM Output Override Value Register (PWM_OOV).

The set registers PWM Output Selection Set Register (PWM_OSS) and PWM Output Selection Set Update Register (PWM_OSSUPD) enable the override of the outputs of a channel regardless of other channels. In the same way, the clear registers PWM Output Selection Clear Register (PWM_OSC) and PWM Output Selection Clear Update Register (PWM_OSCUPD) disable the override of the outputs of a channel regardless of other channels.

By using buffer registers PWM_OSSUPD and PWM_OSCUPD, the output selection of PWM outputs is done synchronously to the channel counter, at the beginning of the next PWM period.

By using registers PWM_OSS and PWM_OSC, the output selection of PWM outputs is done asynchronously to the channel counter, as soon as the register is written.

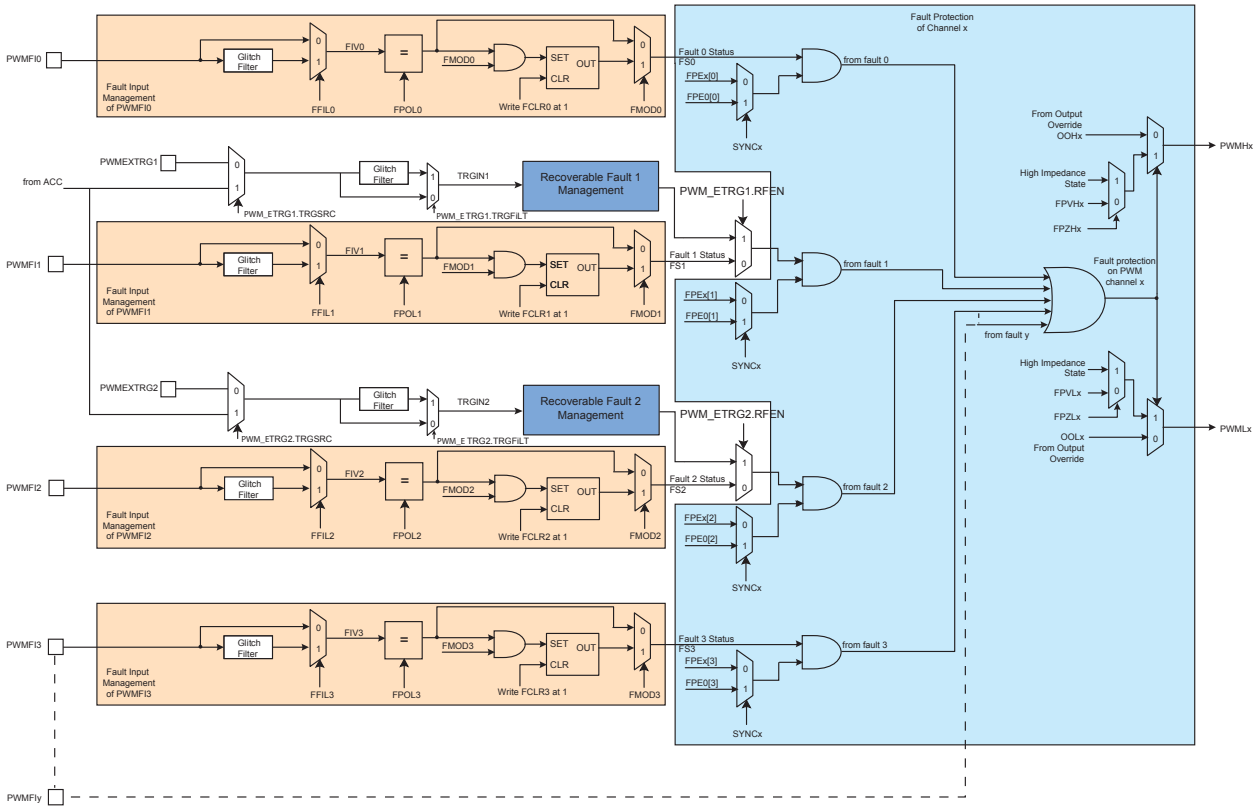
The value of the current output selection can be read in PWM_OS.

While overriding PWM outputs, the channel counters continue to run, only the PWM outputs are forced to user defined values.

41.6.2.7 Fault Protection

7 inputs provide fault protection which can force any of the PWM output pairs to a programmable value. This mechanism has priority over output overriding.

Figure 41-16. Fault Protection



The polarity level of the fault inputs is configured by the FPOL field in the PWM Fault Mode Register (PWM_FMR). For fault inputs coming from internal peripherals such as Timer Counter, the polarity level must be FPOL = 1.

The configuration of the Fault Activation mode (FMOD field in PWMC_FMR) depends on the peripheral generating the fault. If the corresponding peripheral does not have “Fault Clear” management, then the FMOD configuration to use must be FMOD = 1, to avoid spurious fault detection. Refer to the corresponding peripheral documentation for details on handling fault generation.

Fault inputs may or may not be glitch-filtered depending on the FFIL field in PWM_FMR. When the filter is activated, glitches on fault inputs with a width inferior to the PWM peripheral clock period are rejected.

A fault becomes active as soon as its corresponding fault input has a transition to the programmed polarity level. If the corresponding bit FMOD is set to ‘0’ in PWM_FMR, the fault remains active as long as the fault input is at this polarity level. If the corresponding FMOD field is set to ‘1’, the fault remains active until the fault input is no longer at this polarity level and until it is cleared by writing the corresponding bit FCLR in the PWM Fault Clear Register (PWM_FCR). In the PWM Fault Status Register (PWM_FSR), the field FIV indicates the current level of the fault inputs and the field FIS indicates whether a fault is currently active.

Each fault can be taken into account or not by the fault protection mechanism in each channel. To be taken into account in the channel x, the fault y must be enabled by the bit FPEx[y] in the PWM Fault Protection Enable registers (PWM_FPE1). However, synchronous channels (see Synchronous Channels) do not use their own fault enable bits, but those of the channel 0 (bits FPE0[y]).

The fault protection on a channel is triggered when this channel is enabled and when any one of the faults that are enabled for this channel is active. It can be triggered even if the PWM peripheral clock is not running but only by a fault input that is not glitch-filtered.

When the fault protection is triggered on a channel, the fault protection mechanism resets the counter of this channel and forces the channel outputs to the values defined by the fields FPVHx and FPVLx in the PWM Fault Protection Value Register 1 (PWM_FPV) and fields FPZHx/FPZLx in the PWM Fault Protection Value Register 2, as shown in the table below. The output forcing is made asynchronously to the channel counter.

Table 41-3. Forcing Values of PWM Outputs by Fault Protection

FPZH/Lx	FPVH/Lx	Forcing Value of PWMH/Lx
0	0	0
0	1	1
1	–	High impedance state (Hi-Z)

⚠ CAUTION

- To prevent any unexpected activation of the status flag FSy in PWM_FSR, the FMODy bit can be set to ‘1’ only if the FPOLy bit has been previously configured to its final value.
- To prevent any unexpected activation of the Fault Protection on the channel x, the bit FPEx[y] can be set to ‘1’ only if the FPOLy bit has been previously configured to its final value.

If a comparison unit is enabled (see PWM Comparison Units) and if a fault is triggered in the channel 0, then the comparison cannot match.

As soon as the fault protection is triggered on a channel, an interrupt (different from the interrupt generated at the end of the PWM period) can be generated but only if it is enabled and not masked. The interrupt is reset by reading the interrupt status register, even if the fault which has caused the trigger of the fault protection is kept active.

41.6.2.7.1 Recoverable Fault

The PWM provides a Recoverable Fault mode on fault 1 and 2 (see figure *Fault Protection*).

The recoverable fault signal is an internal signal generated as soon as an external trigger event occurs (see PWM External Trigger Mode).

When the fault 1 or 2 is defined as a recoverable fault, the corresponding fault input pin is ignored and bits FFIL1/2, FMOD1/2 and FFIL1/2 are not taken into account.

The fault 1 is managed as a recoverable fault by the PWMEXTRG1 input trigger when PWM_ETRG1.RFEN = 1, PWM_ENA.CHID1 = 1, and PWM_ETRG1.TRGMODE ≠ 0.

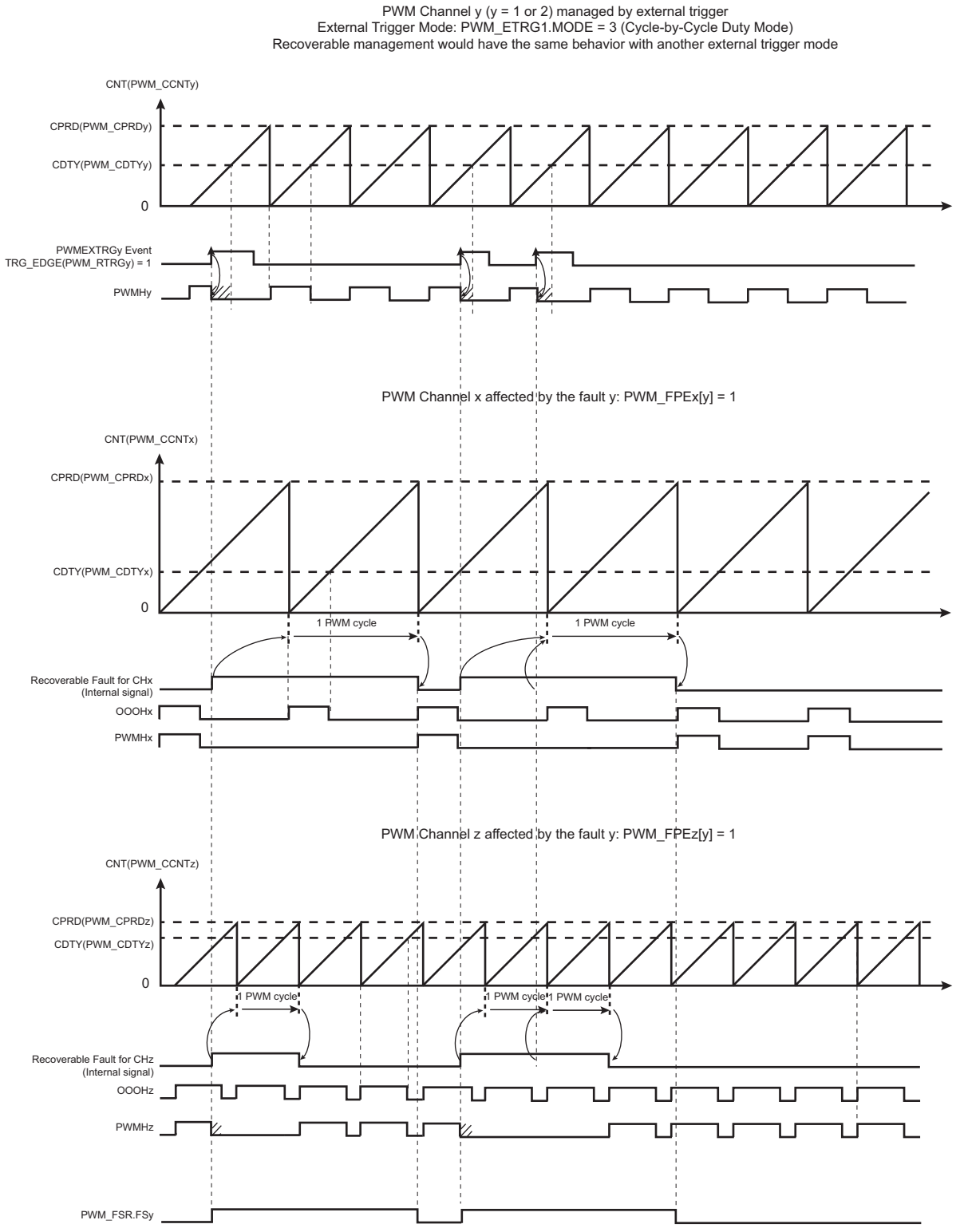
The fault 2 is managed as a recoverable fault by the PWMEXTRG2 input trigger when PWM_ETRG2.RFEN = 1, PWM_ENA.CHID2 = 1, and PWM_ETRG2.TRGMODE ≠ 0.

Recoverable fault 1 and 2 can be taken into account by all channels by enabling the bit FPEx[1/2] in the PWM Fault Protection Enable registers (PWM_FPEx). However the synchronous channels (see Synchronous Channels) do not use their own fault enable bits, but those of the channel 0 (bits FPE0[1/2]).

When a recoverable fault is triggered (according to the PWM_ETRGx.TRGMODE setting), the PWM counter of the affected channels is not cleared (unlike in the classic fault protection mechanism) but the channel outputs are forced to the values defined by the fields FPVHx and FPVLx in the PWM Fault Protection Value Register 1 (PWM_FPV), as per table *Forcing Values of PWM Outputs by Fault Protection*. The output forcing is made asynchronously to the channel counter and lasts from the recoverable fault occurrence to the end of the next PWM cycle (if the recoverable fault is no longer present) (see the figure below).

The recoverable fault does not trigger an interrupt. The Fault Status FSy (with y = 1 or 2) is not reported in the PWM Fault Status Register when the fault y is a recoverable fault.

Figure 41-17. Recoverable Fault Management



41.6.2.8 Spread Spectrum Counter

The PWM macrocell includes a spread spectrum counter allowing the generation of a constantly varying duty cycle on the output PWM waveform (only for the channel 0). This feature may be useful to minimize electromagnetic interference or to reduce the acoustic noise of a PWM driven motor.

This is achieved by varying the effective period in a range defined by a spread spectrum value which is programmed by the field SPRD in the PWM Spread Spectrum Register (PWM_SSPR). The effective period of the output waveform is the value of the spread spectrum counter added to the programmed waveform period CPRD in the PWM Channel Period Register (PWM_CPRD0).

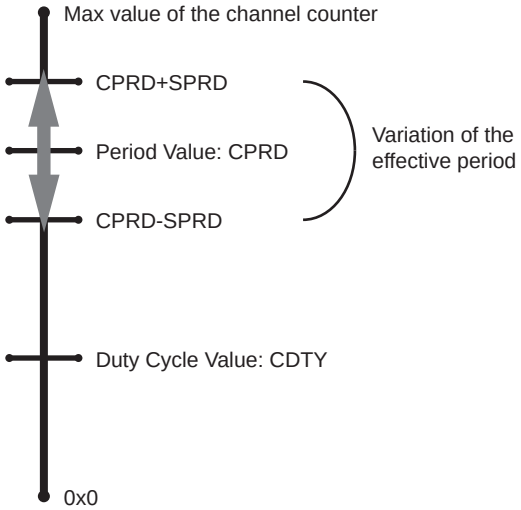
It will cause the effective period to vary from CPRD-SPRD to CPRD+SPRD. This leads to a constantly varying duty cycle on the PWM output waveform because the duty cycle value programmed is unchanged.

The value of the spread spectrum counter can change in two ways depending on the bit SPRDM in PWM_SSPR.

If SPRDM = 0, the Triangular mode is selected. The spread spectrum counter starts to count from -SPRD when the channel 0 is enabled or after reset and counts upwards at each period of the channel counter. When it reaches SPRD, it restarts to count from -SPRD again.

If SPRDM = 1, the Random mode is selected. A new random value is assigned to the spread spectrum counter at each period of the channel counter. This random value is between -SPRD and +SPRD and is uniformly distributed.

Figure 41-18. Spread Spectrum Counter



41.6.2.9 Synchronous Channels

Some channels can be linked together as synchronous channels. They have the same source clock, the same period, the same alignment and are started together. In this way, their counters are synchronized together.

The synchronous channels are defined by the SYNCx bits in the PWM Sync Channels Mode Register (PWM_SCM). Only one group of synchronous channels is allowed.

When a channel is defined as a synchronous channel, the channel 0 is also automatically defined as a synchronous channel. This is because the channel 0 counter configuration is used by all the synchronous channels.

If a channel x is defined as a synchronous channel, the fields/bits for the channel 0 are used instead of those of channel x:

- CPRE in PWM_CMR0 instead of CPRE in PWM_CMRx (same source clock)
- CPRD in PWM_CPRD0 instead of CPRD in PWM_CPRDx (same period)
- CALG in PWM_CMR0 instead of CALG in PWM_CMRx (same alignment)

Modifying the fields CPRE, CPRD and CALG of for channels with index greater than 0 has no effect on output waveforms.

Because counters of synchronous channels must start at the same time, they are all enabled together by enabling the channel 0 (by the CHID0 bit in PWM_ENA register). In the same way, they are all disabled together by disabling channel 0 (by the CHID0 bit in PWM_DIS register). However, a synchronous channel x different from channel 0 can be enabled or disabled independently from others (by the CHIDx bit in PWM_ENA and PWM_DIS registers).

Defining a channel as a synchronous channel while it is an asynchronous channel (by writing the bit SYNCx to ‘1’ while it was at ‘0’) is allowed only if the channel is disabled at this time (CHIDx = 0 in PWM_SR). In the same way,

defining a channel as an asynchronous channel while it is a synchronous channel (by writing the SYNCx bit to ‘0’ while it was ‘1’) is allowed only if the channel is disabled at this time.

The UPDM field (Update Mode) in the PWM_SCM register selects one of the three methods to update the registers of the synchronous channels:

- Method 1 (UPDM = 0): The period value, the duty-cycle values and the dead-time values must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).The update is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM Sync Channels Update Control Register (PWM_SCUC) is set to ‘1’.
- Method 2 (UPDM = 1): The period value, the duty-cycle values, the dead-time values and the update period value must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPD). The update of the period value and of the dead-time values is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM_SCUC register is set to ‘1’. The update of the duty-cycle values and the update period value is triggered automatically after an update period defined by the field UPR in the PWM Sync Channels Update Period Register (PWM_SCUP).
- Method 3 (UPDM = 2): Same as Method 2 apart from the fact that the duty-cycle values of ALL synchronous channels are written by the DMA Controller. The user can choose to synchronize the DMA Controller transfer request with a comparison match (see Section 7.3 “PWM Comparison Units”), by the fields PTRM and PTRCS in the PWM_SCM register. The DMA destination address must be configured to access only the PWM DMA Register (PWM_DMAR). The DMA buffer data structure must consist of sequentially repeated duty cycles. The number of duty cycles in each sequence corresponds to the number of synchronized channels. Duty cycles in each sequence must be ordered from the lowest to the highest channel index. The size of the duty cycle is 16 bits.

Table 41-4. Summary of the Update of Registers of Synchronous Channels

Register	UPDM = 0	UPDM = 1	UPDM = 2
Period Value (PWM_CPRDUPDx)	Write by the processor		
	Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to ‘1’		
Dead-Time Values (PWM_DTUPDx)	Write by the processor		
	Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to ‘1’		
Duty-Cycle Values (PWM_CDTYUPDx)	Write by the processor	Write by the processor	Write by the DMA Controller
	Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to ‘1’	Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR	
Update Period Value (PWM_SCUPUPD)	Not applicable	Write by the processor	
	Not applicable	Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR	

41.6.2.9.1 Method 1: Manual write of duty-cycle values and manual trigger of the update

In this mode, the update of the period value, the duty-cycle values and the dead-time values must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).

To trigger the update, the user must use the bit UPDULOCK in the PWM_SCUC register which allows to update synchronously (at the same PWM period) the synchronous channels:

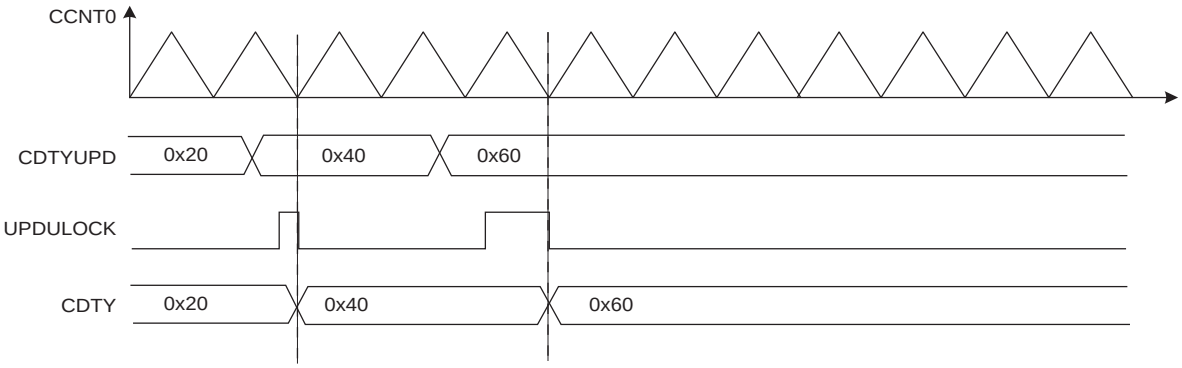
- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

Sequence for Method 1:

1. Select the manual write of duty-cycle values and the manual update by setting the UPDM field to ‘0’ in the PWM_SCM register.
2. Define the synchronous channels by the SYNCx bits in the PWM_SCM register.
3. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
4. If an update of the period value and/or the duty-cycle values and/or the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).
5. Set UPDULOCK to ‘1’ in PWM_SCUC.
6. The update of the registers will occur at the beginning of the next PWM period. When the UPDULOCK bit is reset, go to Step 4. for new values.

Figure 41-19. Method 1 (UPDM = 0)



41.6.2.9.2 Method 2: Manual write of duty-cycle values and automatic trigger of the update

In this mode, the update of the period value, the duty-cycle values, the dead-time values and the update period value must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_CDTYUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK in the PWM_SCUC register, which updates synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the update period by the UPR field in the PWM_SCUP register. The PWM controller waits UPR+1 period of synchronous channels before updating automatically the duty values and the update period value.

The status of the duty-cycle value write is reported in the PWM Interrupt Status Register 2 (PWM_ISR2) by the following flags:

- WRDY: this flag is set to ‘1’ when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to ‘0’ when the PWM_ISR2 register is read.

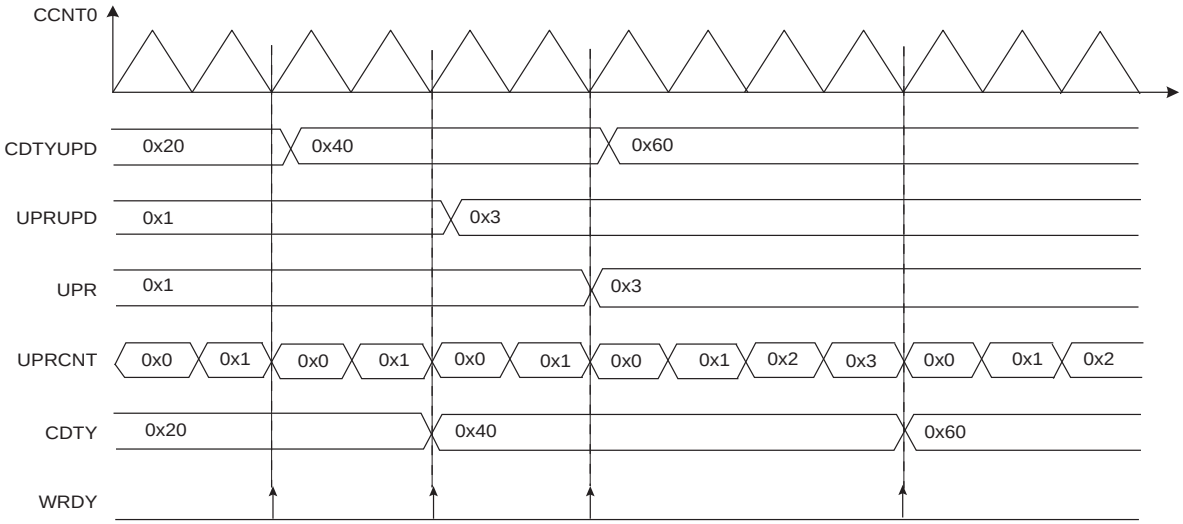
Depending on the interrupt mask in the PWM Interrupt Mask Register 2 (PWM_IMR2), an interrupt can be generated by these flags.

Sequence for Method 2:

1. Select the manual write of duty-cycle values and the automatic update by setting the field UPDM to ‘1’ in the PWM_SCM register
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
5. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 8.
6. Set UPDULOCK to ‘1’ in PWM_SCUC.

7. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 5. for new values.
8. If an update of the duty-cycle values and/or the update period is required, check first that write of new update values is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in PWM_ISR2.
9. Write registers that need to be updated (PWM_CDTYUPDx, PWM_SCUPUPD).
10. The update of these registers will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 8. for new values.

Figure 41-20. Method 2 (UPDM = 1)



41.6.2.9.3 Method 3: Automatic write of duty-cycle values and automatic trigger of the update

In this mode, the update of the duty cycle values is made automatically by the DMA Controller. The update of the period value, the dead-time values and the update period value must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK which allows to update synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period value is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the Update Period by the field UPR in the PWM_SCUP register. The PWM controller waits UPR+1 periods of synchronous channels before updating automatically the duty values and the update period value.

Using the DMA Controller removes processor overhead by reducing its intervention during the transfer. This significantly reduces the number of clock cycles required for a data transfer, which improves microcontroller performance.

The DMA Controller must write the duty-cycle values in the synchronous channels index order. For example if the channels 0, 1 and 3 are synchronous channels, the DMA Controller must write the duty-cycle of the channel 0 first, then the duty-cycle of the channel 1, and finally the duty-cycle of the channel 3.

The status of the DMA Controller transfer is reported in PWM_ISR2 by the following flags:

- WRDY: this flag is set to ‘1’ when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to ‘0’ when PWM_ISR2 is read. The user can choose to synchronize the WRDY flag and the DMA Controller transfer request with a comparison match (see PWM Comparison Units), by the fields PTRM and PTRCS in the PWM_SCM register.
- UNRE: this flag is set to ‘1’ when the update period defined by the UPR field has elapsed while the whole data has not been written by the Peripheral DMA Controller. It is reset to ‘0’ when PWM_ISR2 is read.

Depending on the interrupt mask in PWM_IMR2, an interrupt can be generated by these flags.

Sequence for Method 3:

1. Select the automatic write of duty-cycle values and automatic update by setting the field UPDM to 2 in the PWM_SCM register.
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Define when the WRDY flag and the corresponding DMA Controller transfer request must be set in the update period by the PTRM bit and the PTRCS field in the PWM_SCM register (at the end of the update period or when a comparison matches).
5. Define the DMA Controller transfer settings for the duty-cycle values and enable it in the DMA Controller registers
6. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
7. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 10.
8. Set UPDULOCK to '1' in PWM_SCUC.
9. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 7. for new values.
10. If an update of the update period value is required, check first that write of a new update value is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in PWM_ISR2, else go to Step 14.
11. Write the register that needs to be updated (PWM_SCUPUPD).
12. The update of this register will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 10. for new values.
13. Wait for the DMA status flag indicating that the buffer transfer is complete. If the transfer has ended, define a new DMA transfer for new duty-cycle values. Go to Step 5.
14. **Figure 41-21. Method 3 (UPDM = 2 and PTRM = 0)**

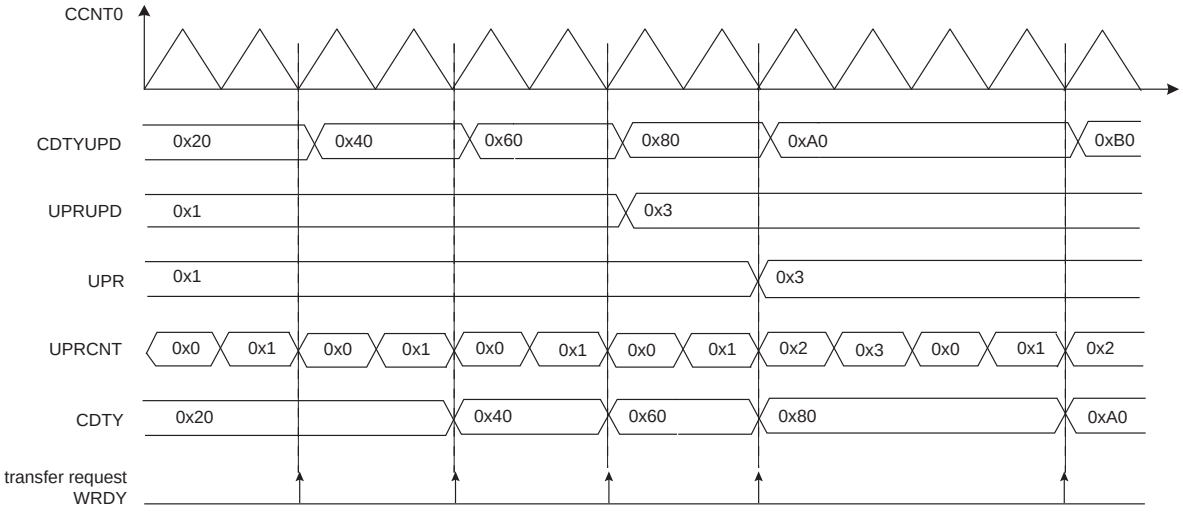
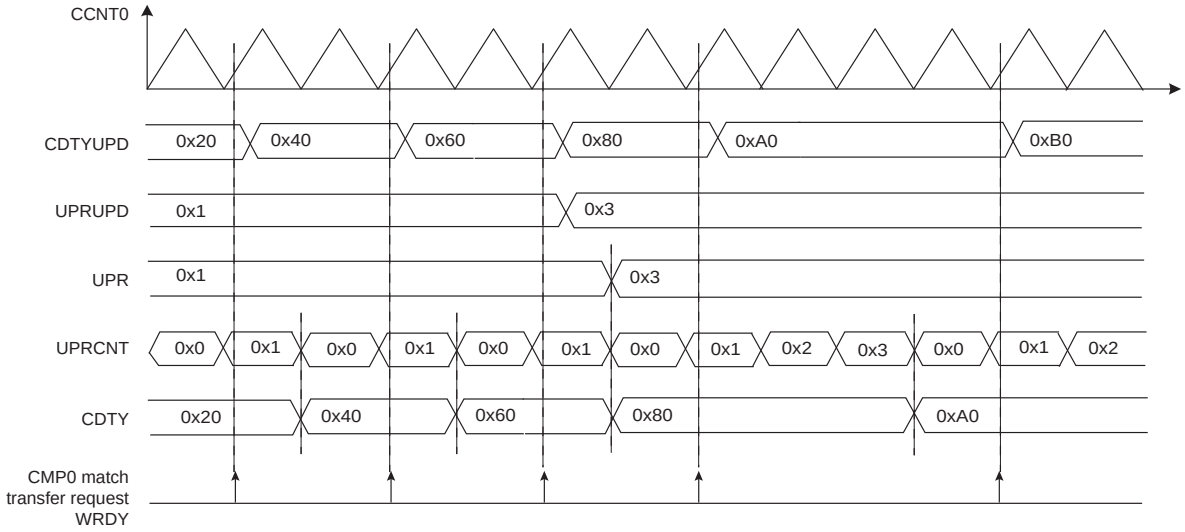


Figure 41-22. Method 3 (UPDM = 2 and PTRM = 1 and PTRCS = 0)



41.6.2.10 Update Time for Double-Buffering Registers

All channels integrate a double-buffering system in order to prevent an unexpected output waveform while modifying the period, the spread spectrum value, the polarity, the duty-cycle, the dead-times, the output override, and the synchronous channels update period.

This double-buffering system comprises the following update registers:

- PWM Sync Channels Update Period Update Register
- PWM Output Selection Set Update Register
- PWM Output Selection Clear Update Register
- PWM Spread Spectrum Update Register
- PWM Channel Duty Cycle Update Register
- PWM Channel Period Update Register
- PWM Channel Dead Time Update Register
- PWM Channel Mode Update Register

When one of these update registers is written to, the write is stored, but the values are updated only at the next PWM period border. In Left-aligned mode (CALG = 0), the update occurs when the channel counter reaches the period value CPRD. In Center-aligned mode, the update occurs when the channel counter value is decremented and reaches the 0 value.

In Center-aligned mode, it is possible to trigger the update of the polarity and the duty-cycle at the next half period border. This mode concerns the following update registers:

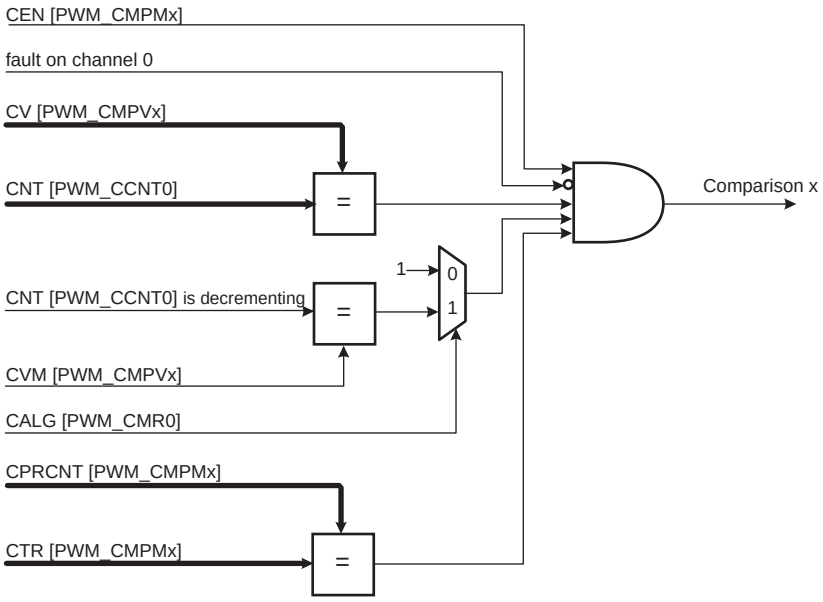
- PWM Channel Duty Cycle Update Register
- PWM Channel Mode Update Register

The update occurs at the first half period following the write of the update register (either when the channel counter value is incrementing and reaches the period value CPRD, or when the channel counter value is decrementing and reaches the 0 value). To activate this mode, the user must write a one to the bit UPDS in the PWM Channel Mode Register.

41.6.3 PWM Comparison Units

The PWM provides 8 independent comparison units able to compare a programmed value with the current value of the channel 0 counter (which is the channel counter of all synchronous channels, “Synchronous Channels”). These comparisons are intended to generate pulses on the event lines, to generate software interrupts and to trigger DMA Controller transfer requests for the synchronous channels (see Method 3: Automatic write of duty-cycle values and automatic trigger of the update).

Figure 41-23. Comparison Unit Block Diagram



The comparison x matches when it is enabled by the bit CEN in the PWM Comparison x Mode Register (PWM_CMPMx for the comparison x) and when the counter of the channel 0 reaches the comparison value defined by the field CV in PWM Comparison x Value Register (PWM_CMPVx for the comparison x). If the counter of the channel 0 is center-aligned (CALG = 1 in PWM Channel Mode Register), the bit CVM in PWM_CMPVx defines if the comparison is made when the counter is counting up or counting down (in Left-alignment mode CALG = 0, this bit is useless).

If a fault is active on the channel 0, the comparison is disabled and cannot match (see Fault Protection).

The user can define the periodicity of the comparison x by the fields CTR and CPR in PWM_CMPMx. The comparison is performed periodically once every CPR+1 periods of the counter of the channel 0, when the value of the comparison period counter CPRCNT in PWM_CMPMx reaches the value defined by CTR. CPR is the maximum value of the comparison period counter CPRCNT. If CPR = CTR = 0, the comparison is performed at each period of the counter of the channel 0.

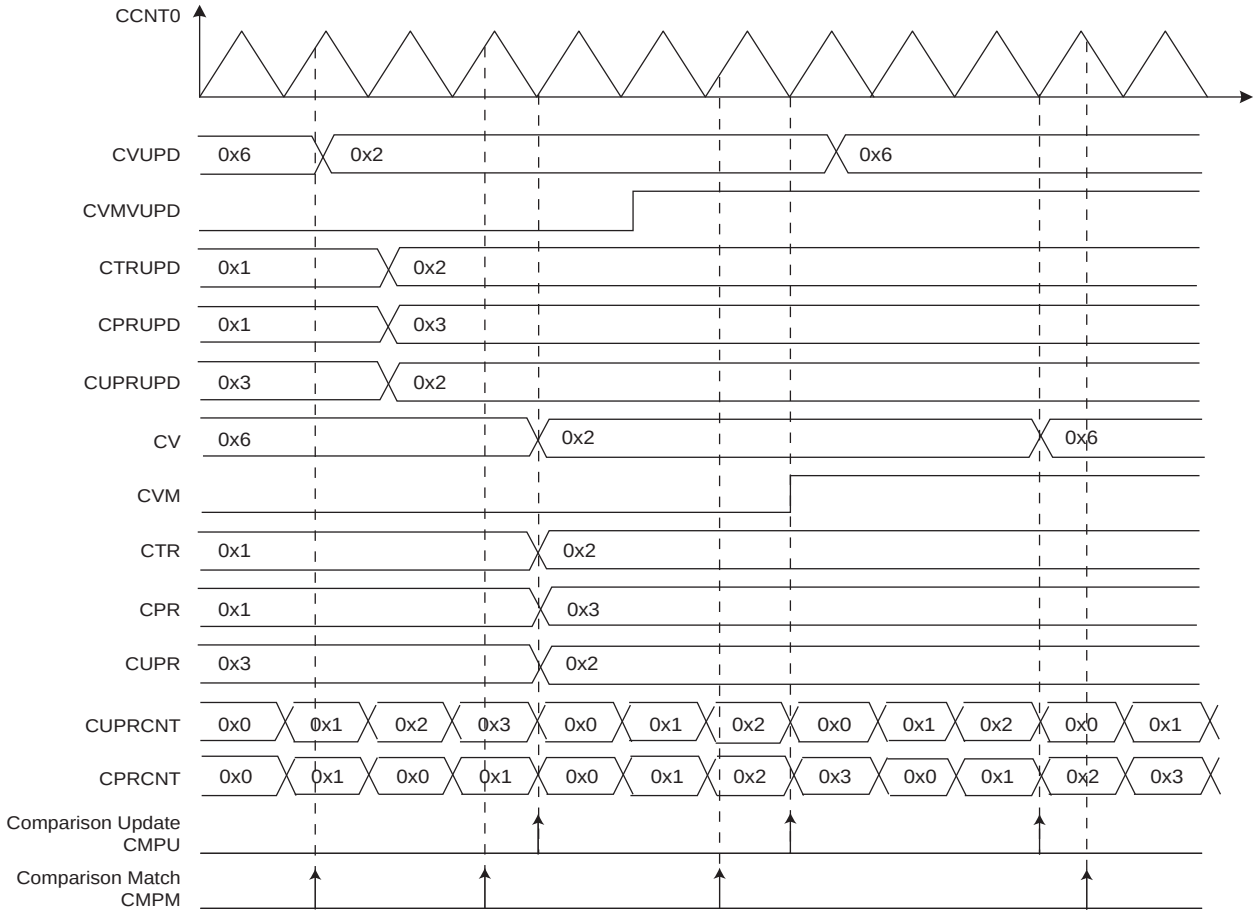
The comparison x configuration can be modified while the channel 0 is enabled by using the PWM Comparison x Mode Update Register (PWM_CMPMUPDx registers for the comparison x). In the same way, the comparison x value can be modified while the channel 0 is enabled by using the PWM Comparison x Value Update Register (PWM_CMPVUPDx registers for the comparison x).

The update of the comparison x configuration and the comparison x value is triggered periodically after the comparison x update period. It is defined by the field CUPR in PWM_CMPMx. The comparison unit has an update period counter independent from the period counter to trigger this update. When the value of the comparison update period counter CUPRCNT (in PWM_CMPMx) reaches the value defined by CUPR, the update is triggered. The comparison x update period CUPR itself can be updated while the channel 0 is enabled by using the PWM_CMPMUPDx register.

⚠ CAUTION The write of PWM_CMPVUPDx must be followed by a write of PWM_CMPMUPDx.

The comparison match and the comparison update can be source of an interrupt, but only if it is enabled and not masked. These interrupts can be enabled by the PWM Interrupt Enable Register 2 and disabled by the PWM Interrupt Disable Register 2. The comparison match interrupt and the comparison update interrupt are reset by reading the PWM Interrupt Status Register 2.

Figure 41-24. Comparison Waveform



41.6.4 PWM Event Lines

The PWM provides 2 independent event lines intended to trigger actions in other peripherals.

A pulse (one cycle of the peripheral clock) is generated on an event line, when at least one of the selected comparisons is matching. The comparisons can be selected or unselected independently by the CSEL bits in the PWM Event Line x Register (PWM_ELMRx for the Event Line x).

An example of event generation is provided in the figure Event Line Generation Waveform (Example).

Figure 41-25. Event Line Block Diagram

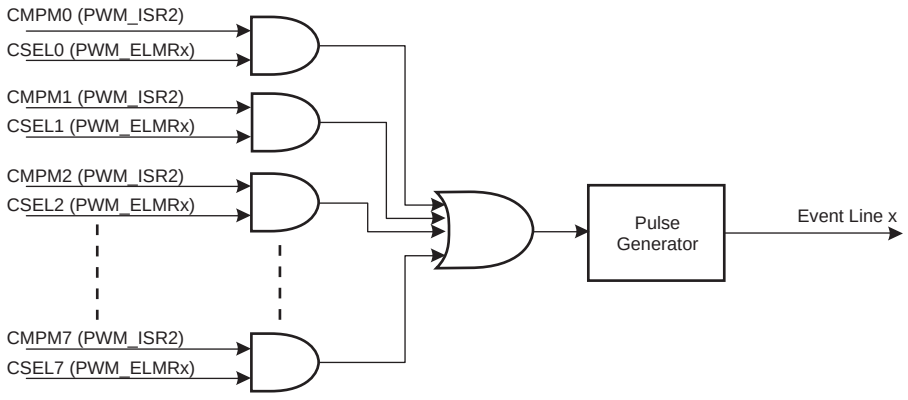
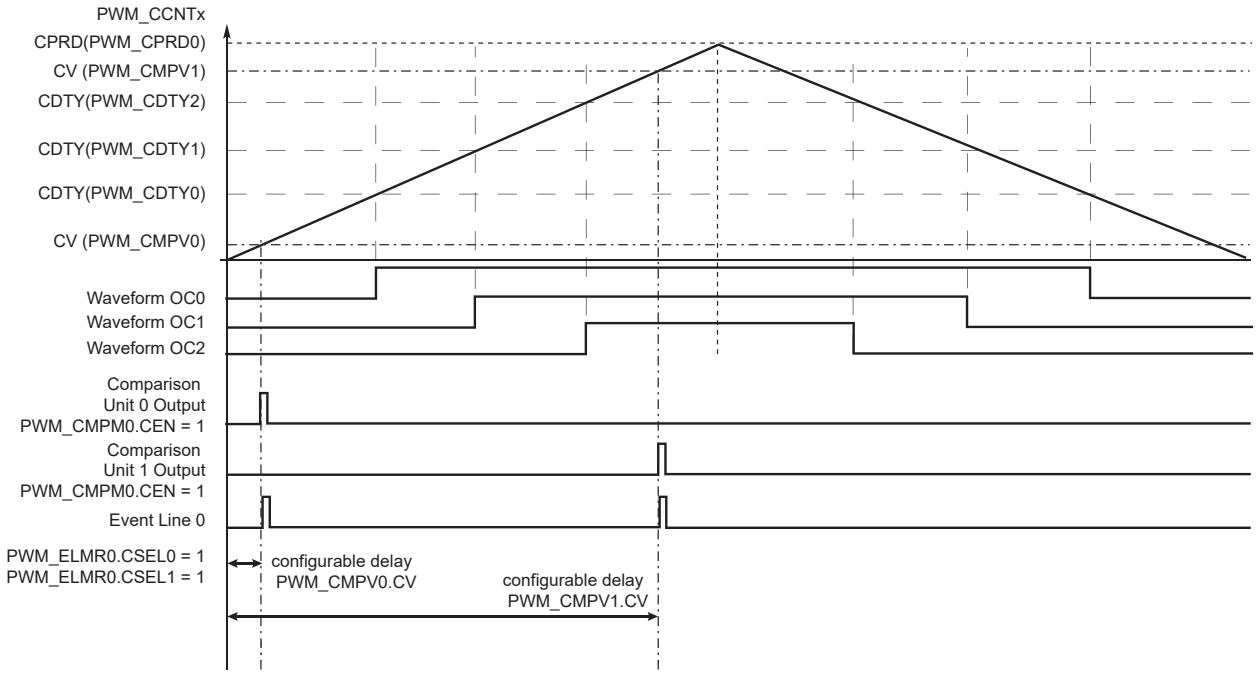


Figure 41-26. Event Line Generation Waveform (Example)



41.6.5 PWM External Trigger Mode

The PWM channels 1 and 2 can be configured to use an external trigger for generating specific PWM signals. The external trigger source can be selected through the bit TRGSRC of the PWM External Trigger Register (see the table below).

Table 41-5. External Event Source Selection

Channel	Trigger Source Selection	Trigger Source
1	PWM_ETRG1.TRGSRC = 0	From PWMEXTRG1 input
	PWM_ETRG1.TRGSRC = 1	From Analog Comparator Controller
2	PWM_ETRG2.TRGSRC = 0	From PWMEXTRG2 input
	PWM_ETRG2.TRGSRC = 1	From Analog Comparator Controller

Each external trigger source can be filtered by writing a one to the TRGFILT bit in the corresponding PWM External Trigger Register (PWM_ETRGx).

Each time an external trigger event is detected, the corresponding PWM channel counter value is stored in the MAXCNT field of the PWM_ETRGx register if it is greater than the previously stored value. Reading the PWM_ETRGx register will clear the MAXCNT value.

Three different modes are available for channels 1 and 2 depending on the value of the TRGMODE field of the PWM_ETRGx register:

- TRGMODE = 1: External PWM Reset Mode
- TRGMODE = 2: External PWM Start Mode
- TRGMODE = 3: Cycle-By-Cycle Duty Mode

See the following sections.

This feature is disabled when TRGMODE = 0.

This feature should only be enabled if the corresponding channel is left-aligned (CALG = 0 in PWM Channel Mode Register of channel 1 or 2) and not managed as a synchronous channel (SYNCx = 0 in PWM Sync Channels Mode

Register where x = 1 or 2). Programming the channel to be center-aligned or synchronous while TRGMODE is not 0 could lead to unexpected behavior.

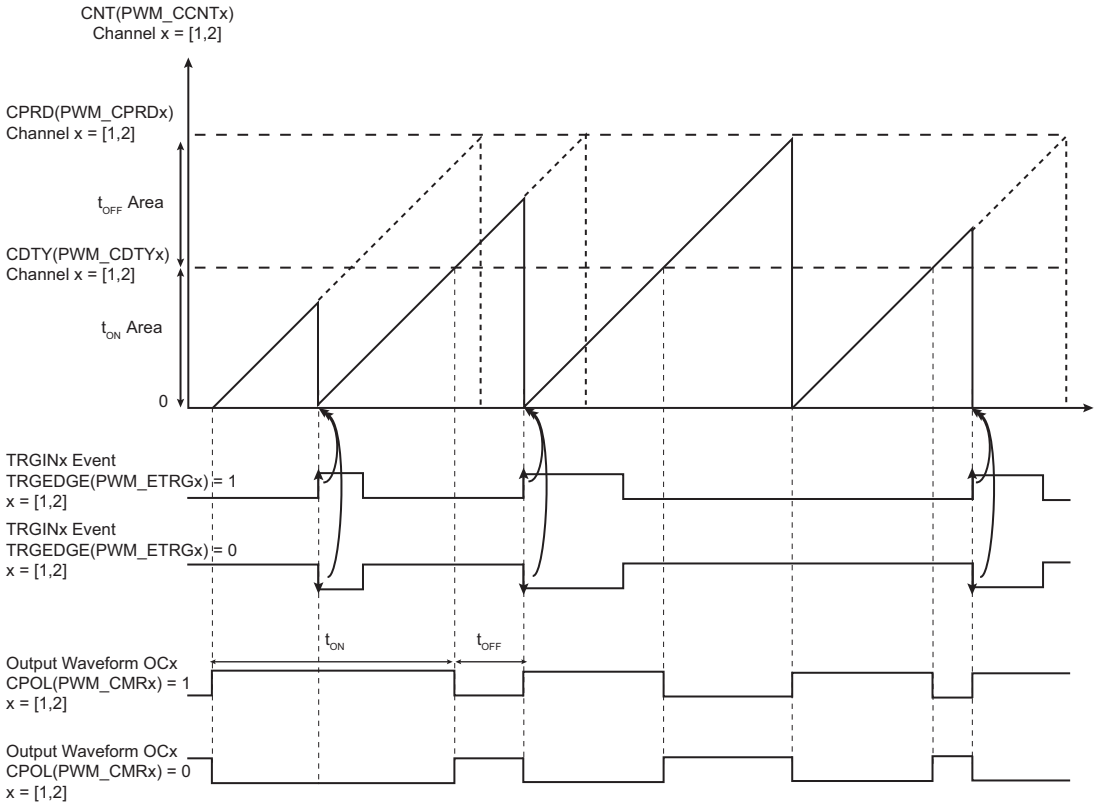
41.6.5.1 External PWM Reset Mode

External PWM Reset mode is selected by programming TRGMODE = 1 in the PWM_ETRGx register.

In this mode, when an edge is detected on the PWMEXTRGx input, the internal PWM counter is cleared and a new PWM cycle is restarted. The edge polarity can be selected by programming the TRGEDGE bit in the PWM_ETRGx register. If no trigger event is detected when the internal channel counter has reached the CPRD value in the PWM Channel Period Register, the internal counter is cleared and a new PWM cycle starts.

Note that this mode does not guarantee a constant t_{ON} or t_{OFF} time.

Figure 41-27. External PWM Reset Mode



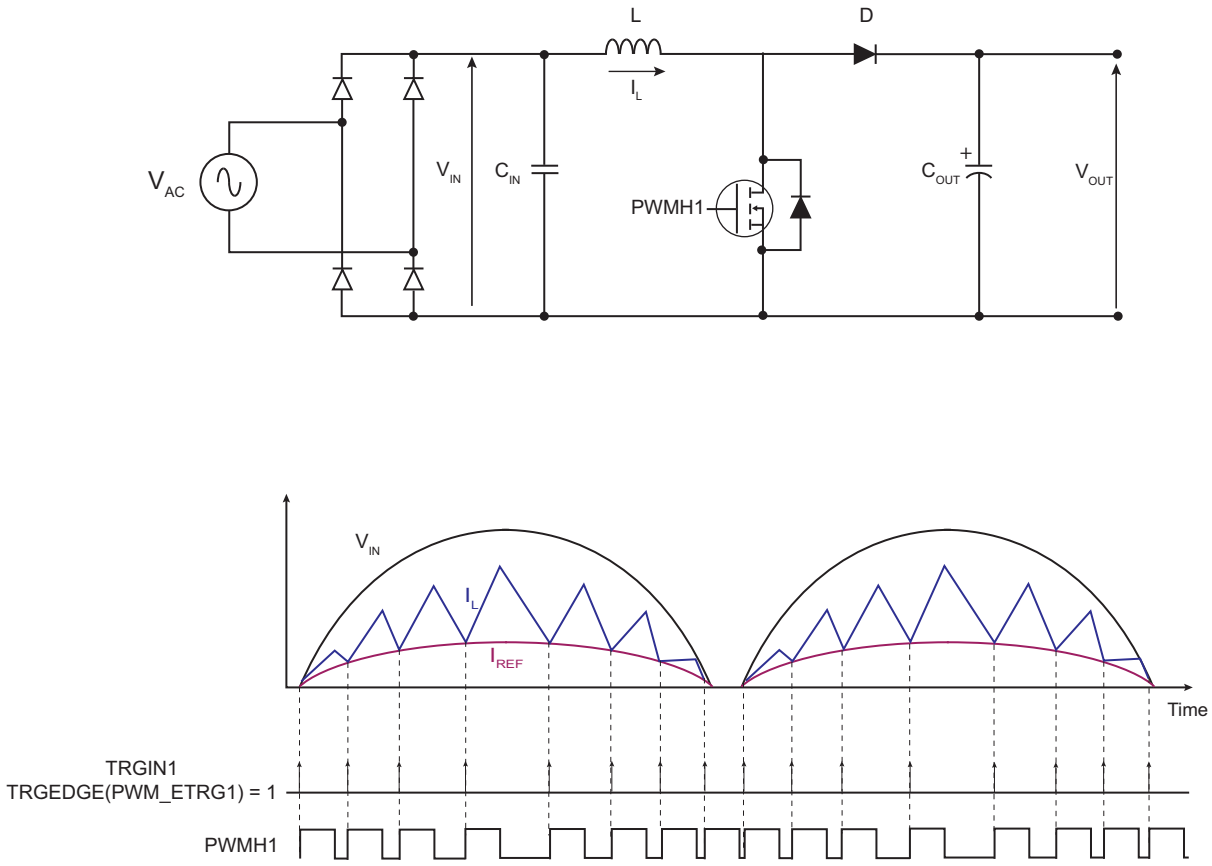
41.6.5.1.1 Application Example

The external PWM Reset mode can be used in power factor correction applications.

In the example below, the external trigger input is the PWMEXTRG1 (therefore the PWM channel used for regulation is the channel 1). The PWM channel 1 period (CPRD in the PWM Channel Period Register of the channel 1) must be programmed so that the TRGIN1 event always triggers before the PWM channel 1 period elapses.

In the figure below, an external circuit (not shown) is required to sense the inductor current I_L. The internal PWM counter of the channel 1 is cleared when the inductor current falls below a specific threshold (I_{REF}). This starts a new PWM period and increases the inductor current.

Figure 41-28. External PWM Reset Mode: Power Factor Correction Application



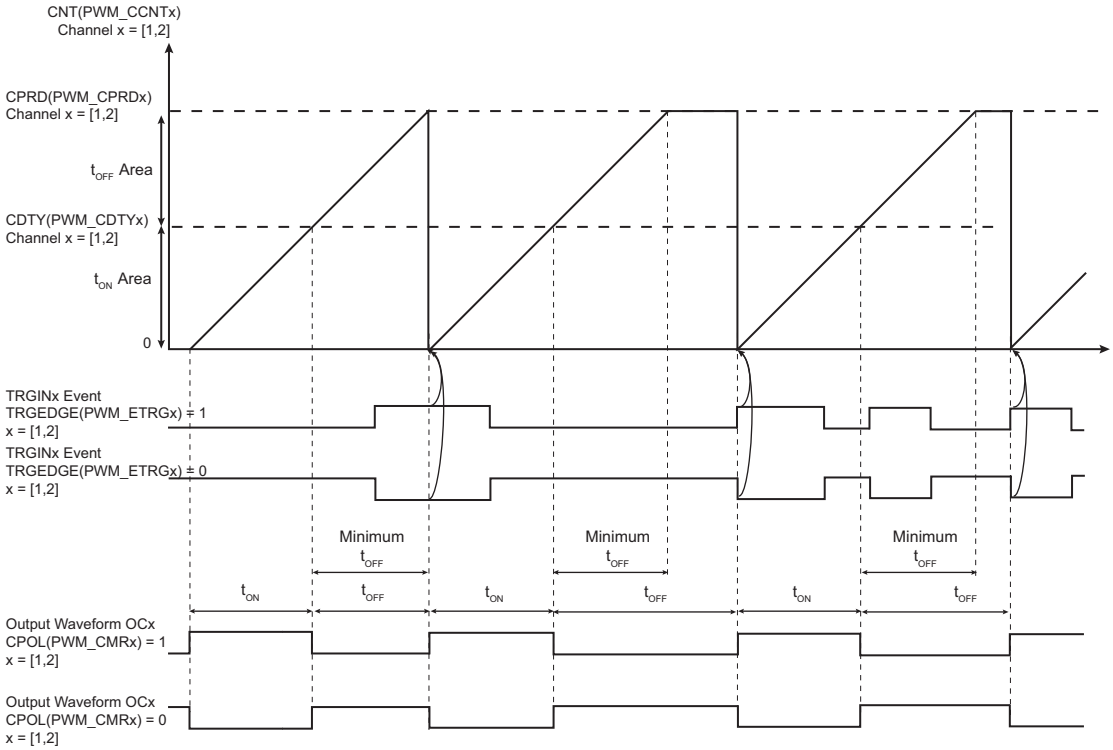
41.6.5.2 External PWM Start Mode

External PWM Start mode is selected by programming TRGMODE = 2 in the PWM_ETRGx register.

In this mode, the internal PWM counter can only be reset once it has reached the CPRD value in the PWM Channel Period Register and when the correct level is detected on the corresponding external trigger input. Both conditions have to be met to start a new PWM period. The active detection level is defined by the bit TRGEDGE of the PWM_ETRGx register.

Note that this mode guarantees a constant t_{ON} time and a minimum t_{OFF} time.

Figure 41-29. External PWM Start Mode



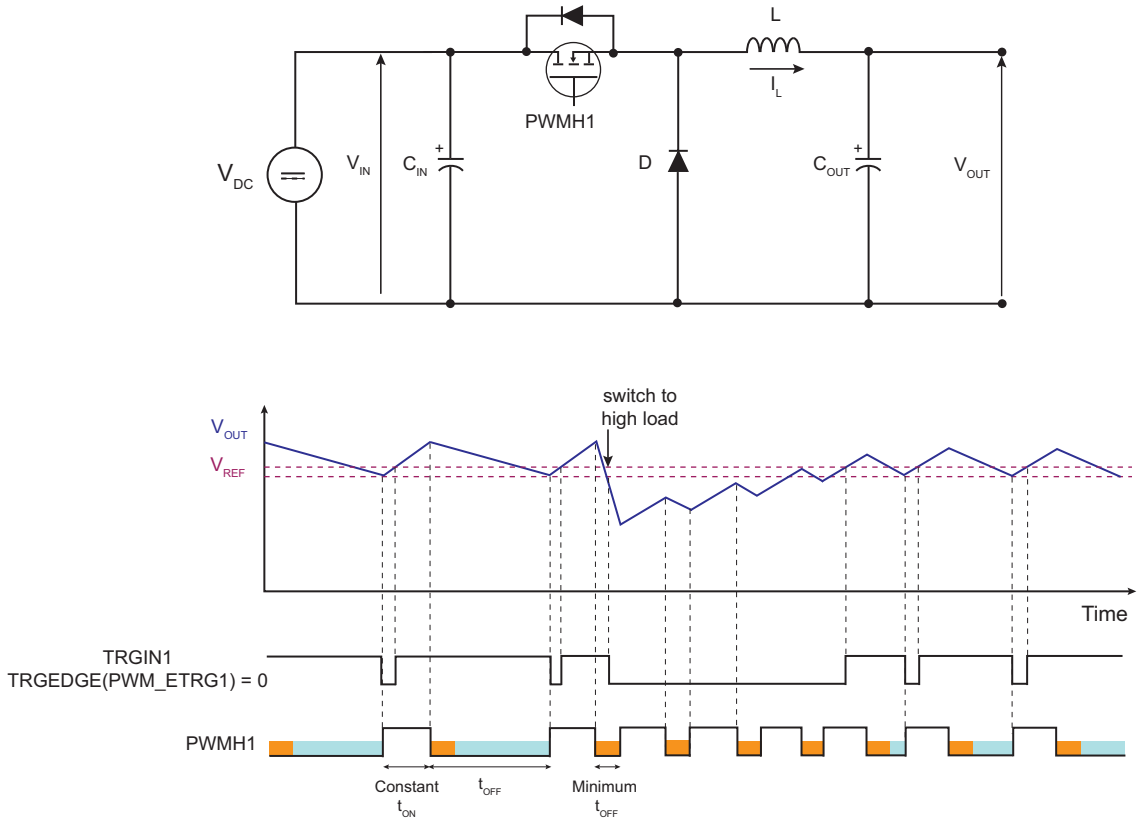
41.6.5.2.1 Application Example

The external PWM Start mode generates a modulated frequency PWM signal with a constant active level duration (t_{ON}) and a minimum inactive level duration (minimum t_{OFF}).

The t_{ON} time is defined by the CDTY value in the PWM Channel Duty Cycle Register. The minimum t_{OFF} time is defined by CDTY - CPRD (PWM Channel Period Register). This mode can be useful in Buck DC/DC Converter applications.

When the output voltage V_{OUT} is above a specific threshold (Vref), the PWM inactive level is maintained as long as V_{OUT} remains above this threshold. If V_{OUT} is below this specific threshold, this mode guarantees a minimum t_{OFF} time required for MOSFET driving (see the figure below).

Figure 41-30. External PWM Start Mode: Buck DC/DC Converter



41.6.5.3 Cycle-By-Cycle Duty Mode

41.6.5.3.1 Description

Cycle-by-cycle duty mode is selected by programming TRGMODE = 3 in PWM_ETRGx.

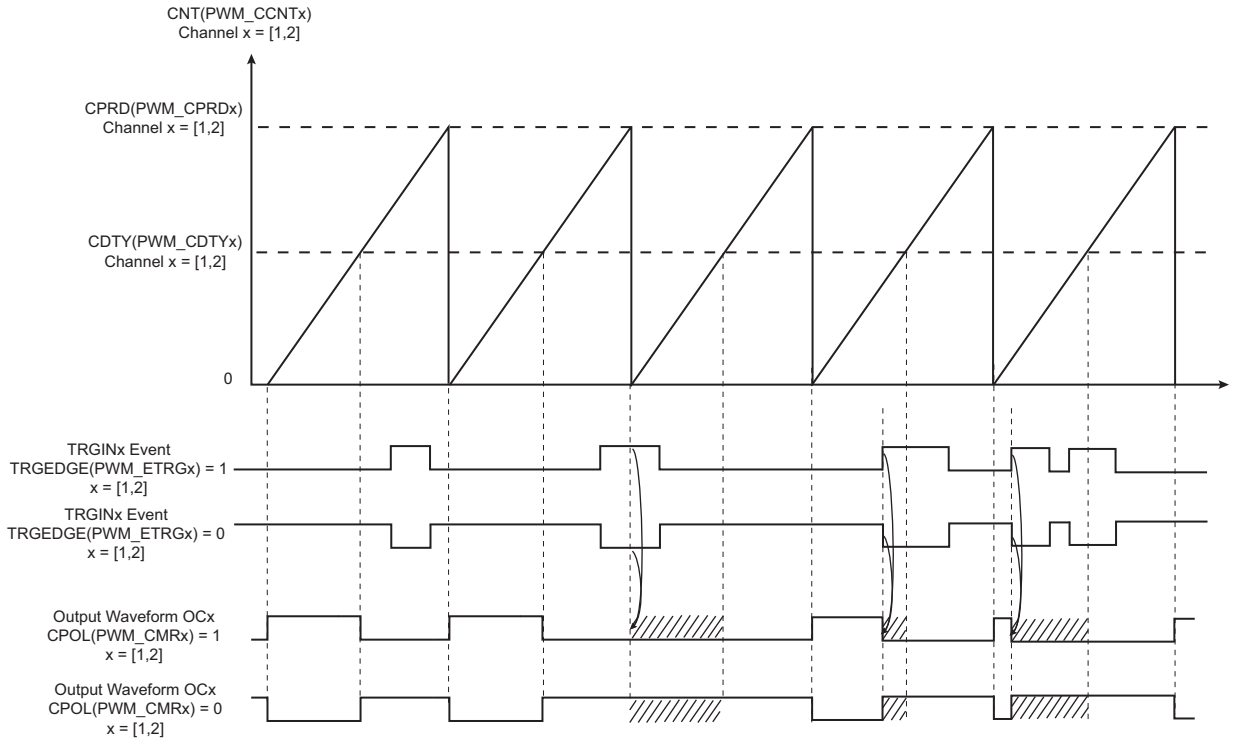
In this mode, the PWM frequency is constant and is defined by the CPRD value in the PWM Channel Period Register.

An external trigger event has no effect on the PWM output if it occurs while the internal PWM counter value is above the CDTY value of the PWM Channel Duty Cycle Register.

If the internal PWM counter value is below the value of CDTY of the PWM Channel Duty Cycle Register, an external trigger event makes the PWM output inactive.

The external trigger event can be detected on rising or falling edge according to the TRGEDGE bit in PWM_ETRGx.

Figure 41-31. Cycle-By-Cycle Duty Mode

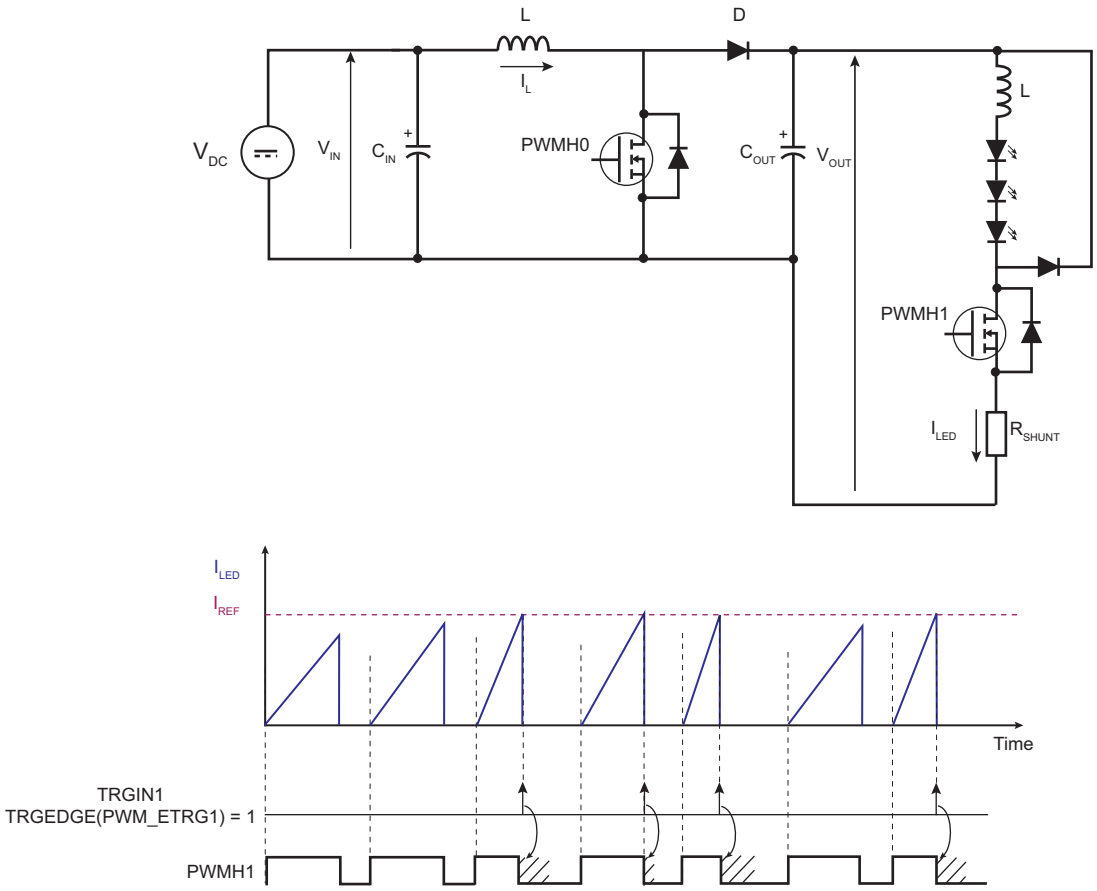


41.6.5.3.2 Application Example

The figure below illustrates an application example of the Cycle-by-cycle Duty mode.

In an LED string control circuit, Cycle-by-cycle Duty mode can be used to automatically limit the current in the LED string.

Figure 41-32. Cycle-By-Cycle Duty Mode: LED String Control



41.6.5.4 Leading-Edge Blanking (LEB)

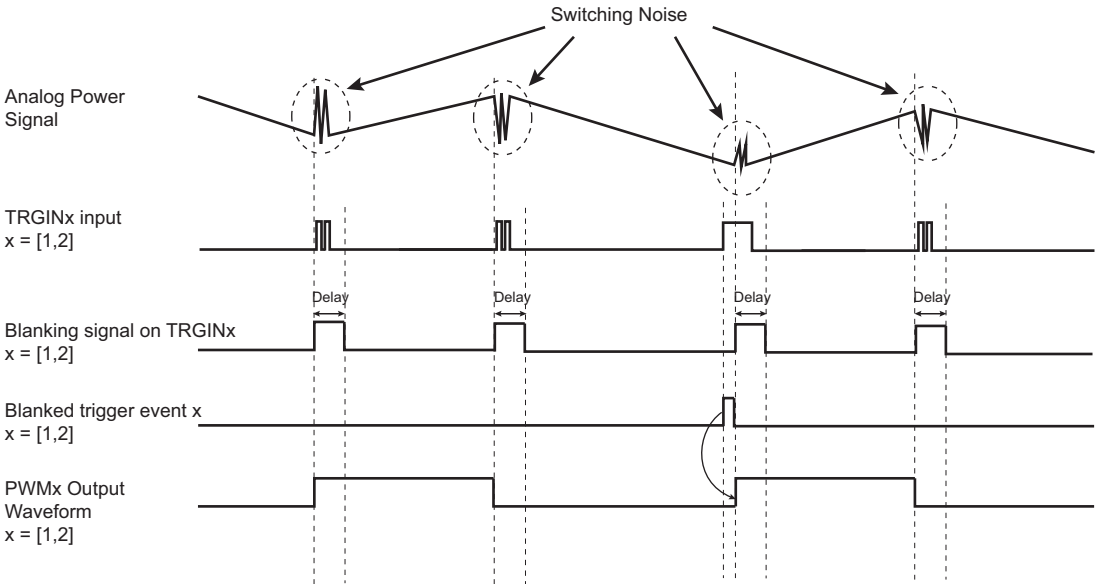
PWM channels 1 and 2 support leading-edge blanking. Leading-edge blanking masks the external trigger input when a transient occurs on the corresponding PWM output. It masks potential spurious external events due to power transistor switching.

The blanking delay on each external trigger input is configured by programming the LEBDELAYx in the PWM Leading-Edge Blanking Register.

The LEB can be enabled on both the rising and the falling edges for the PWMH and PWML outputs through the bits PWMLFEN, PWMLREN, PWMHFEN, PWMHREN.

Any event on the PWMEXTRGx input which occurs during the blanking time is ignored.

Figure 41-33. Leading-Edge Blanking



41.6.6 PWM Controller Operations

41.6.6.1 Initialization

Before enabling the channels, they must be configured by the software application as described below:

- Unlock User Interface by writing the WPCMD field in PWM_WPCR.
- Configuration of the clock generator (DIVA, PREA, DIVB, PREB in the PWM_CLK register if required).
- Selection of the clock for each channel (CPRE field in PWM_CMRx)
- Configuration of the waveform alignment for each channel (CALG field in PWM_CMRx)
- Selection of the counter event selection (if CALG = 1) for each channel (CES field in PWM_CMRx)
- Configuration of the output waveform polarity for each channel (CPOL bit in PWM_CMRx)
- Configuration of the period for each channel (CPRD in the PWM_CPRDx register). Writing in PWM_CPRDx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CPRDUPDx register to update PWM_CPRDx as explained below.
- Configuration of the duty-cycle for each channel (CDTY in the PWM_CDTYx register). Writing in PWM_CDTYx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CDTYUPDx register to update PWM_CDTYx as explained below.
- Configuration of the dead-time generator for each channel (DTH and DTL in PWM_DTx) if enabled (DTE bit in PWM_CMRx). Writing in the PWM_DTx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_DTUPDx register to update PWM_DTx
- Selection of the synchronous channels (SYNCx in the PWM_SCM register)
- Selection of the moment when the WRDY flag and the corresponding DMA Controller transfer request are set (PTRM and PTRCS in the PWM_SCM register)
- Configuration of the Update mode (UPDM in PWM_SCM register)
- Configuration of the update period (UPR in PWM_SCUP register) if needed
- Configuration of the comparisons (PWM_CMPVx and PWM_CMPMx)
- Configuration of the event lines (PWM_ELMRx)
- Configuration of the fault inputs polarity (FPOL in PWM_FMR)
- Configuration of the fault protection (FMOD and FFIL in PWM_FMR, PWM_FPV and PWM_FPE1)
- Enable of the interrupts (writing CHIDx and FCHIDx in PWM_IER1, and writing WRDY, UNRE, CMPMx and CMPUx in PWM_IER2)
- Enable of the PWM channels (writing CHIDx in the PWM_ENA register)

41.6.6.2 Source Clock Selection Criteria

The large number of source clocks can make selection difficult. The relationship between the value in the PWM Channel Period Register (PWM_CPRDx) and the PWM Channel Duty Cycle Register (PWM_CDTYx) helps the user select the appropriate clock. The event number written in the Period Register gives the PWM accuracy. The Duty-Cycle quantum cannot be lower than 1/CPRDx value. The higher the value of PWM_CPRDx, the greater the PWM accuracy.

For example, if the user sets 15 (in decimal) in PWM_CPRDx, the user is able to set a value from between 1 up to 14 in PWM_CDTYx. The resulting duty-cycle quantum cannot be lower than 1/15 of the PWM period.

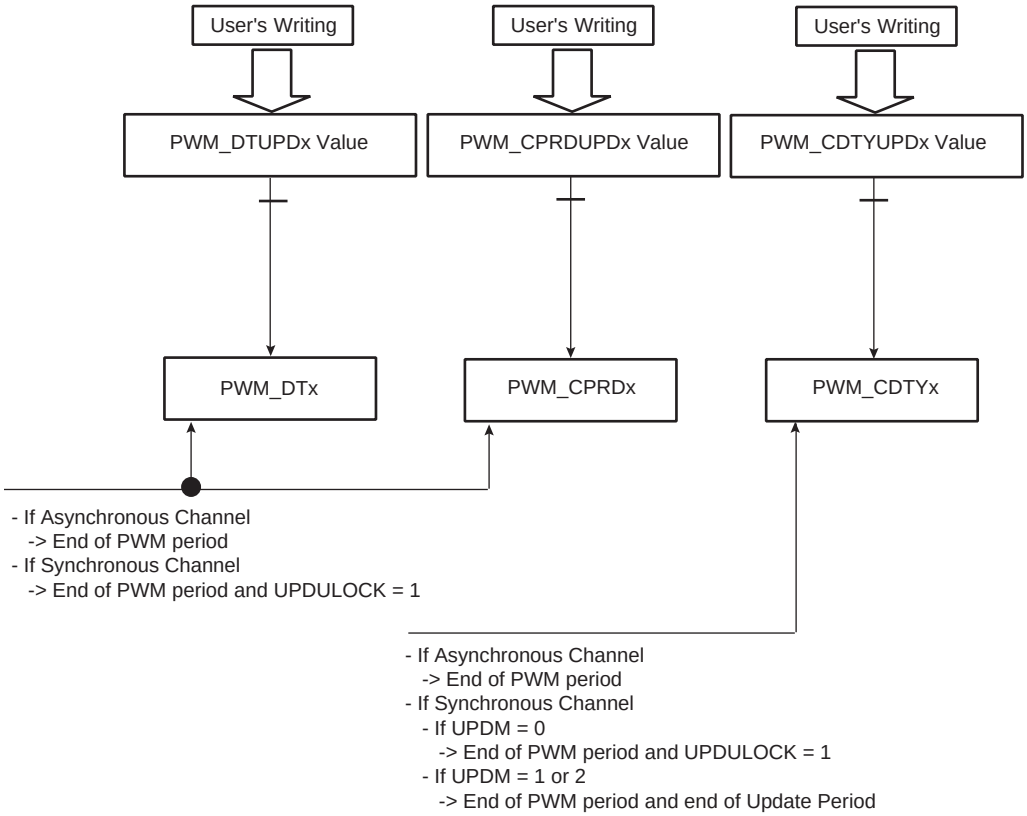
41.6.6.3 Changing the Duty-Cycle, the Period and the Dead-Times

It is possible to modulate the output waveform duty-cycle, period and dead-times.

To prevent unexpected output waveform, the user must use the PWM Channel Duty Cycle Update Register (PWM_CDTYUPDx), the PWM Channel Period Update Register (PWM_CPRDUPDx) and the PWM Channel Dead Time Update Register (PWM_DTUPDx) to change waveform parameters while the channel is still enabled.

- If the channel is an asynchronous channel (SYNCx = 0 in PWM Sync Channels Mode Register (PWM_SCM)), these registers hold the new period, duty-cycle and dead-times values until the end of the current PWM period and update the values for the next period.
- If the channel is a synchronous channel and update method 0 is selected (SYNCx = 1 and UPDM = 0 in PWM_SCM register), these registers hold the new period, duty-cycle and dead-times values until the bit UPDULOCK is written at ‘1’ (in PWM Sync Channels Update Control Register (PWM_SCUC)) and the end of the current PWM period, then update the values for the next period.
- If the channel is a synchronous channel and update method 1 or 2 is selected (SYNCx = 1 and UPDM = 1 or 2 in PWM_SCM register):
  - registers PWM_CPRDUPDx and PWM_DTUPDx hold the new period and dead-times values until the bit UPDULOCK is written at ‘1’ (in PWM_SCUC) and the end of the current PWM period, then update the values for the next period.
  - register PWM_CDTYUPDx holds the new duty-cycle value until the end of the update period of synchronous channels (when UPRCNT is equal to UPR in PWM Sync Channels Update Period Register (PWM_SCUP)) and the end of the current PWM period, then updates the value for the next period.
    Note: If the update registers PWM_CDTYUPDx, PWM_CPRDUPDx and PWM_DTUPDx are written several times between two updates, only the last written value is taken into account.

Figure 41-34. Synchronized Period, Duty-Cycle and Dead-Time Update



41.6.6.4 Changing the Update Period of Synchronous Channels

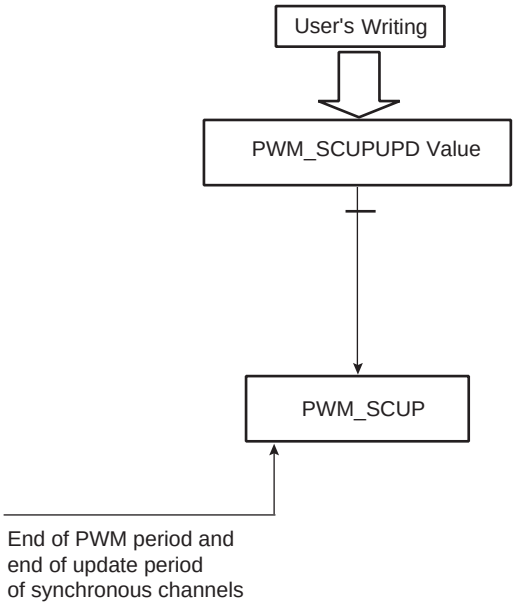
It is possible to change the update period of synchronous channels while they are enabled. See Method 2: Manual write of duty-cycle values and automatic trigger of the update and Method 3: Automatic write of duty-cycle values and automatic trigger of the update .

To prevent an unexpected update of the synchronous channels registers, the user must use the PWM Sync Channels Update Period Update Register (PWM_SCUPUPD) to change the update period of synchronous channels while they are still enabled. This register holds the new value until the end of the update period of synchronous channels (when UPRCNT is equal to UPR in PWM_SCUP) and the end of the current PWM period, then updates the value for the next period.

Note:

1. If the update register PWM_SCUPUPD is written several times between two updates, only the last written value is taken into account.
2. Changing the update period does make sense only if there is one or more synchronous channels and if the update method 1 or 2 is selected (UPDM = 1 or 2 in PWM Sync Channels Mode Register).

Figure 41-35. Synchronized Update of Update Period Value of Synchronous Channels



41.6.6.5 Changing the Comparison Value and the Comparison Configuration

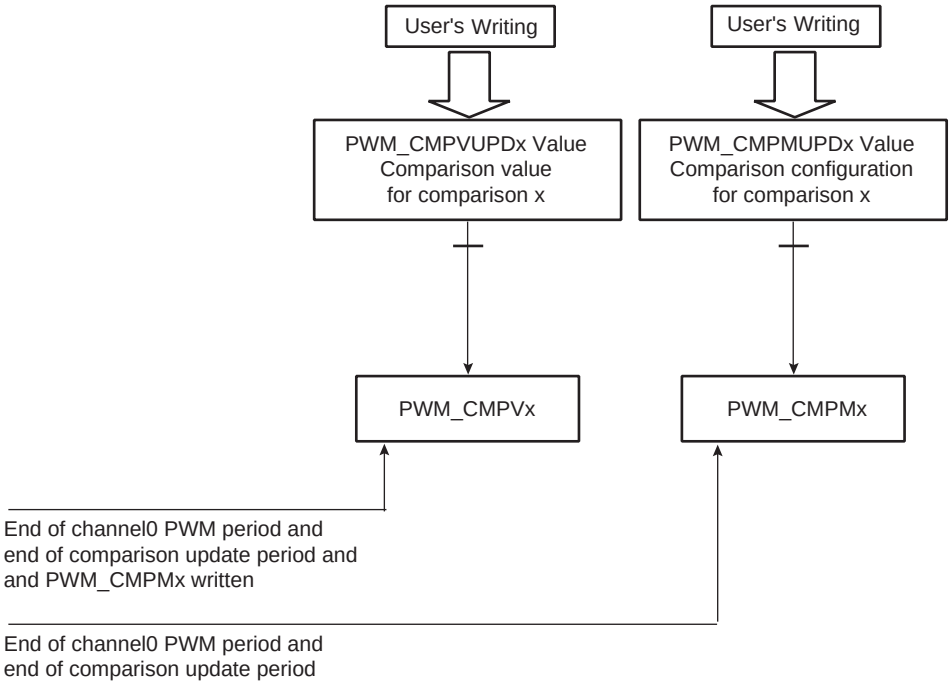
It is possible to change the comparison values and the comparison configurations while the channel 0 is enabled (see PWM Comparison Units).

To prevent unexpected comparison match, the user must use the PWM Comparison x Value Update Register (PWM_CMPVUPDx) and the PWM Comparison x Mode Update Register (PWM_CMPMUPDx) to change, respectively, the comparison values and the comparison configurations while the channel 0 is still enabled. These registers hold the new values until the end of the comparison update period (when CUPRCNT is equal to CUPR in PWM Comparison x Mode Register (PWM_CMPMx) and the end of the current PWM period, then update the values for the next period.

⚠ CAUTION The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

Note: If the update registers PWM_CMPVUPDx and PWM_CMPMUPDx are written several times between two updates, only the last written value are taken into account.

Figure 41-36. Synchronized Update of Comparison Values and Configurations



41.6.6.6 Interrupt Sources

Depending on the interrupt mask in PWM_IMR1 and PWM_IMR2, an interrupt can be generated at the end of the corresponding channel period (CHIDx in the PWM Interrupt Status Register 1 (PWM_ISR1)), after a fault event (FCHIDx in PWM_ISR1), after a comparison match (CMPMx in PWM_ISR2), after a comparison update (CMPUx in PWM_ISR2) or according to the Transfer mode of the synchronous channels (WRDY and UNRE in PWM_ISR2).

If the interrupt is generated by the flags CHIDx or FCHIDx, the interrupt remains active until a read operation in PWM_ISR1 occurs.

If the interrupt is generated by the flags WRDY or UNRE or CMPMx or CMPUx, the interrupt remains active until a read operation in PWM_ISR2 occurs.

A channel interrupt is enabled by setting the corresponding bit in PWM_IER1 and PWM_IER2. A channel interrupt is disabled by setting the corresponding bit in PWM_IDR1 and PWM_IDR2.

41.6.7 Register Write Protection

To prevent any single software error that may corrupt PWM behavior, the registers listed below can be write-protected by writing the field WPCMD in the PWM Write Protection Control Register (PWM_WPCR). They are divided into six groups:

- Register group 0:
  - PWM Clock Register
- Register group 1:
  - PWM Disable Register
  - PWM Interrupt Enable Register 1
  - PWM Interrupt Disable Register 1
  - PWM Interrupt Enable Register 2
  - PWM Interrupt Disable Register 2
- Register group 2:
  - PWM Sync Channels Mode Register
  - PWM Channel Mode Register
  - PWM Stepper Motor Mode Register

  - PWM Fault Protection Value Register 2
  - PWM Leading-Edge Blanking Register
  - PWM Channel Mode Update Register
- Register group 3:
  - PWM Spread Spectrum Register
  - PWM Spread Spectrum Update Register
  - PWM Channel Period Register
  - PWM Channel Period Update Register
- Register group 4:
  - PWM Channel Dead Time Register
  - PWM Channel Dead Time Update Register
- Register group 5:
  - PWM Fault Mode Register
  - PWM Fault Protection Value Register 1

There are two types of write protection:

- SW write protection—can be enabled or disabled by software
- HW write protection—can be enabled by software but only disabled by a hardware reset of the PWM controller

Both types of write protection can be applied independently to a particular register group by means of the WPCMD and WPRGx fields in PWM_WPCR. If at least one type of write protection is active, the register group is write-protected. The value of field WPCMD defines the action to be performed:

- 0: Disables SW write protection of the register groups of which the bit WPRGx is at ‘1’
- 1: Enables SW write protection of the register groups of which the bit WPRGx is at ‘1’
- 2: Enables HW write protection of the register groups of which the bit WPRGx is at ‘1’

At any time, the user can determine whether SW or HW write protection is active in a particular register group by the fields WPSWS and WPHWS in the PWM Write Protection Status Register (PWM_WPSR).

If a write access to a write-protected register is detected, the WPVS flag in PWM_WPSR is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS and WPVSRC fields are automatically cleared after reading PWM_WPSR.

41.7 Register Summary

Offset	Name	Bit Pos.								
0x00	PWM_CLK	7:0	DIVA[7:0]							
		15:8					PREA[3:0]			
		23:16	DIVB[7:0]							
		31:24					PREB[3:0]			
0x04	PWM_ENA	7:0					CHID3	CHID2	CHID1	CHID0
		15:8								
		23:16								
		31:24								
0x08	PWM_DIS	7:0					CHID3	CHID2	CHID1	CHID0
		15:8								
		23:16								
		31:24								
0x0C	PWM_SR	7:0					CHID3	CHID2	CHID1	CHID0
		15:8								
		23:16								
		31:24								
0x10	PWM_IER1	7:0					CHID3	CHID2	CHID1	CHID0
		15:8								
		23:16					FCHID3	FCHID2	FCHID1	FCHID0
		31:24								
0x14	PWM_IDR1	7:0					CHID3	CHID2	CHID1	CHID0
		15:8								
		23:16					FCHID3	FCHID2	FCHID1	FCHID0
		31:24								
0x18	PWM_IMR1	7:0					CHID3	CHID2	CHID1	CHID0
		15:8								
		23:16					FCHID3	FCHID2	FCHID1	FCHID0
		31:24								
0x1C	PWM_ISR1	7:0					CHID3	CHID2	CHID1	CHID0
		15:8								
		23:16					FCHID3	FCHID2	FCHID1	FCHID0
		31:24								
0x20	PWM_SCM	7:0					SYNC3	SYNC2	SYNC1	SYNC0
		15:8								
		23:16	PTRCS[2:0]			PTRM			UPDM[1:0]	
		31:24								
0x24	PWM_DMAR	7:0	DMADUTY[7:0]							
		15:8	DMADUTY[15:8]							
		23:16	DMADUTY[23:16]							
		31:24								
0x28	PWM_SCUC	7:0								UPDULOCK
		15:8								
		23:16								
		31:24								
0x2C	PWM_SCUP	7:0	UPRCNT[3:0]				UPR[3:0]			
		15:8								
		23:16								
		31:24								
0x30	PWM_SCUPUPD	7:0					UPRUPD[3:0]			
		15:8								
		23:16								
		31:24								
0x34	PWM_IER2	7:0					UNRE			WRDY
		15:8	CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
		23:16	CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x38	PWM_IDR2	7:0					UNRE			WRDY
		15:8	CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
		23:16	CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
		31:24								
0x3C	PWM_IMR2	7:0					UNRE			WRDY
		15:8	CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
		23:16	CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
		31:24								
0x40	PWM_ISR2	7:0					UNRE			WRDY
		15:8	CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
		23:16	CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
		31:24								
0x44	PWM_OOV	7:0					OOVH3	OOVH2	OOVH1	OOVH0
		15:8								
		23:16					OOVL3	OOVL2	OOVL1	OOVL0
		31:24								
0x48	PWM_OS	7:0					OSH3	OSH2	OSH1	OSH0
		15:8								
		23:16					OSL3	OSL2	OSL1	OSL0
		31:24								
0x4C	PWM_OSS	7:0					OSSH3	OSSH2	OSSH1	OSSH0
		15:8								
		23:16					OSSL3	OSSL2	OSSL1	OSSL0
		31:24								
0x50	PWM_OSC	7:0					OSCH3	OSCH2	OSCH1	OSCH0
		15:8								
		23:16					OSCL3	OSCL2	OSCL1	OSCL0
		31:24								
0x54	PWM_OSSUPD	7:0					OSSUPH3	OSSUPH2	OSSUPH1	OSSUPH0
		15:8								
		23:16					OSSUPL3	OSSUPL2	OSSUPL1	OSSUPL0
		31:24								
0x58	PWM_OSCUPD	7:0					OSCUPH3	OSCUPH2	OSCUPH1	OSCUPH0
		15:8								
		23:16					OSCUPL3	OSCUPL2	OSCUPL1	OSCUPL0
		31:24								
0x5C	PWM_FMR	7:0	FPOL[7:0]							
		15:8	FMOD[7:0]							
		23:16	FFIL[7:0]							
		31:24								
0x60	PWM_FSR	7:0	FIV[7:0]							
		15:8	FS[7:0]							
		23:16								
		31:24								
0x64	PWM_FCR	7:0	FCLR[7:0]							
		15:8								
		23:16								
		31:24								
0x68	PWM_FPV1	7:0					FPVH3	FPVH2	FPVH1	FPVH0
		15:8								
		23:16					FPVL3	FPVL2	FPVL1	FPVL0
		31:24								
0x6C	PWM_FPE	7:0	FPE0[7:0]							
		15:8	FPE1[7:0]							
		23:16	FPE2[7:0]							
		31:24	FPE3[7:0]							
0x70 ... 0x7B	Reserved									

.....continued

Offset	Name	Bit Pos.								
0x7C	PWM_ELMR0	7:0	CSEL7	CSEL6	CSEL5	CSEL4	CSEL3	CSEL2	CSEL1	CSEL0
		15:8								
		23:16								
		31:24								
0x80	PWM_ELMR1	7:0	CSEL7	CSEL6	CSEL5	CSEL4	CSEL3	CSEL2	CSEL1	CSEL0
		15:8								
		23:16								
		31:24								
0x84 ... 0x9F	Reserved									
0xA0	PWM_SSPR	7:0	SPRD[7:0]							
		15:8	SPRD[15:8]							
		23:16	SPRD[23:16]							
		31:24								SPRDM
0xA4	PWM_SSPUP	7:0	SPRDUP[7:0]							
		15:8	SPRDUP[15:8]							
		23:16	SPRDUP[23:16]							
		31:24								
0xA8 ... 0xAF	Reserved									
0xB0	PWM_SMMR	7:0							GCEN1	GCEN0
		15:8								
		23:16							DOWN1	DOWN0
		31:24								
0xB4 ... 0xBF	Reserved									
0xC0	PWM_FPV2	7:0					FPZH3	FPZH2	FPZH1	FPZH0
		15:8								
		23:16					FPZL3	FPZL2	FPZL1	FPZL0
		31:24								
0xC4 ... 0xE3	Reserved									
0xE4	PWM_WPCR	7:0	WPRG5	WPRG4	WPRG3	WPRG2	WPRG1	WPRG0	WPCMD[1:0]	
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0xE8	PWM_WPSR	7:0	WPVS		WPSWS5	WPSWS4	WPSWS3	WPSWS2	WPSWS1	WPSWS0
		15:8			WPHWS5	WPHWS4	WPHWS3	WPHWS2	WPHWS1	WPHWS0
		23:16	WPVSRC[7:0]							
		31:24	WPVSRC[15:8]							
0xEC ... 0x012F	Reserved									
0x0130	PWM_CMPV0	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM
0x0134	PWM_CMPVUPD0	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0138	PWM_CMPM0	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x013C	PWM_CMPMUPD0	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x0140	PWM_CMPV1	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM
0x0144	PWM_CMPVUPD1	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0148	PWM_CMPM1	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								
0x014C	PWM_CMPMUPD1	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x0150	PWM_CMPV2	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM
0x0154	PWM_CMPVUPD2	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0158	PWM_CMPM2	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								
0x015C	PWM_CMPMUPD2	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x0160	PWM_CMPV3	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM
0x0164	PWM_CMPVUPD3	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0168	PWM_CMPM3	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								
0x016C	PWM_CMPMUPD3	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x0170	PWM_CMPV4	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM

.....continued

Offset	Name	Bit Pos.								
0x0174	PWM_CMPVUPD4	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0178	PWM_CMPM4	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								
0x017C	PWM_CMPMUPD4	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x0180	PWM_CMPV5	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM
0x0184	PWM_CMPVUPD5	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0188	PWM_CMPM5	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								
0x018C	PWM_CMPMUPD5	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x0190	PWM_CMPV6	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM
0x0194	PWM_CMPVUPD6	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x0198	PWM_CMPM6	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								
0x019C	PWM_CMPMUPD6	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x01A0	PWM_CMPV7	7:0	CV[7:0]							
		15:8	CV[15:8]							
		23:16	CV[23:16]							
		31:24								CVM
0x01A4	PWM_CMPVUPD7	7:0	CVUPD[7:0]							
		15:8	CVUPD[15:8]							
		23:16	CVUPD[23:16]							
		31:24								CVMUPD
0x01A8	PWM_CMPM7	7:0	CTR[3:0]							CEN
		15:8	CPRCNT[3:0]				CPR[3:0]			
		23:16	CUPRCNT[3:0]				CUPR[3:0]			
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x01AC	PWM_CMPMUPD7	7:0	CTRUPD[3:0]							CENUPD
		15:8					CPRUPD[3:0]			
		23:16					CUPRUPD[3:0]			
		31:24								
0x01B0 ... 0x01FF	Reserved									
0x0200	PWM_CMR0	7:0					CPRE[3:0]			
		15:8			TCTS	DPOLI	UPDS	CES	CPOL	CALG
		23:16					PPM	DTLI	DTHI	DTE
		31:24								
0x0204	PWM_CDTY0	7:0	CDTY[7:0]							
		15:8	CDTY[15:8]							
		23:16	CDTY[23:16]							
		31:24								
0x0208	PWM_CDTYUPD0	7:0	CDTYUPD[7:0]							
		15:8	CDTYUPD[15:8]							
		23:16	CDTYUPD[23:16]							
		31:24								
0x020C	PWM_CPRD0	7:0	CPRD[7:0]							
		15:8	CPRD[15:8]							
		23:16	CPRD[23:16]							
		31:24								
0x0210	PWM_CPRDUPD0	7:0	CPRDUPD[7:0]							
		15:8	CPRDUPD[15:8]							
		23:16	CPRDUPD[23:16]							
		31:24								
0x0214	PWM_CCNT0	7:0	CNT[7:0]							
		15:8	CNT[15:8]							
		23:16	CNT[23:16]							
		31:24								
0x0218	PWM_DT0	7:0	DTH[7:0]							
		15:8	DTH[15:8]							
		23:16	DTL[7:0]							
		31:24	DTL[15:8]							
0x021C	PWM_DTUPD0	7:0	DTHUPD[7:0]							
		15:8	DTHUPD[15:8]							
		23:16	DTLUPD[7:0]							
		31:24	DTLUPD[15:8]							
0x0220	PWM_CMR1	7:0					CPRE[3:0]			
		15:8			TCTS	DPOLI	UPDS	CES	CPOL	CALG
		23:16					PPM	DTLI	DTHI	DTE
		31:24								
0x0224	PWM_CDTY1	7:0	CDTY[7:0]							
		15:8	CDTY[15:8]							
		23:16	CDTY[23:16]							
		31:24								
0x0228	PWM_CDTYUPD1	7:0	CDTYUPD[7:0]							
		15:8	CDTYUPD[15:8]							
		23:16	CDTYUPD[23:16]							
		31:24								
0x022C	PWM_CPRD1	7:0	CPRD[7:0]							
		15:8	CPRD[15:8]							
		23:16	CPRD[23:16]							
		31:24								
0x0230	PWM_CPRDUPD1	7:0	CPRDUPD[7:0]							
		15:8	CPRDUPD[15:8]							
		23:16	CPRDUPD[23:16]							
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x0234	PWM_CCNT1	7:0	CNT[7:0]							
		15:8	CNT[15:8]							
		23:16	CNT[23:16]							
		31:24								
0x0238	PWM_DT1	7:0	DTH[7:0]							
		15:8	DTH[15:8]							
		23:16	DTL[7:0]							
		31:24	DTL[15:8]							
0x023C	PWM_DTUPD1	7:0	DTHUPD[7:0]							
		15:8	DTHUPD[15:8]							
		23:16	DTLUPD[7:0]							
		31:24	DTLUPD[15:8]							
0x0240	PWM_CMR2	7:0					CPRE[3:0]			
		15:8			TCTS	DPOLI	UPDS	CES	CPOL	CALG
		23:16					PPM	DTLI	DTHI	DTE
		31:24								
0x0244	PWM_CDTY2	7:0	CDTY[7:0]							
		15:8	CDTY[15:8]							
		23:16	CDTY[23:16]							
		31:24								
0x0248	PWM_CDTYUPD2	7:0	CDTYUPD[7:0]							
		15:8	CDTYUPD[15:8]							
		23:16	CDTYUPD[23:16]							
		31:24								
0x024C	PWM_CPRD2	7:0	CPRD[7:0]							
		15:8	CPRD[15:8]							
		23:16	CPRD[23:16]							
		31:24								
0x0250	PWM_CPRDUPD2	7:0	CPRDUPD[7:0]							
		15:8	CPRDUPD[15:8]							
		23:16	CPRDUPD[23:16]							
		31:24								
0x0254	PWM_CCNT2	7:0	CNT[7:0]							
		15:8	CNT[15:8]							
		23:16	CNT[23:16]							
		31:24								
0x0258	PWM_DT2	7:0	DTH[7:0]							
		15:8	DTH[15:8]							
		23:16	DTL[7:0]							
		31:24	DTL[15:8]							
0x025C	PWM_DTUPD2	7:0	DTHUPD[7:0]							
		15:8	DTHUPD[15:8]							
		23:16	DTLUPD[7:0]							
		31:24	DTLUPD[15:8]							
0x0260	PWM_CMR3	7:0					CPRE[3:0]			
		15:8			TCTS	DPOLI	UPDS	CES	CPOL	CALG
		23:16					PPM	DTLI	DTHI	DTE
		31:24								
0x0264	PWM_CDTY3	7:0	CDTY[7:0]							
		15:8	CDTY[15:8]							
		23:16	CDTY[23:16]							
		31:24								
0x0268	PWM_CDTYUPD3	7:0	CDTYUPD[7:0]							
		15:8	CDTYUPD[15:8]							
		23:16	CDTYUPD[23:16]							
		31:24								

.....continued

Offset	Name	Bit Pos.								
0x026C	PWM_CPRD3	7:0	CPRD[7:0]							
		15:8	CPRD[15:8]							
		23:16	CPRD[23:16]							
		31:24								
0x0270	PWM_CPRDUPD3	7:0	CPRDUPD[7:0]							
		15:8	CPRDUPD[15:8]							
		23:16	CPRDUPD[23:16]							
		31:24								
0x0274	PWM_CCNT3	7:0	CNT[7:0]							
		15:8	CNT[15:8]							
		23:16	CNT[23:16]							
		31:24								
0x0278	PWM_DT3	7:0	DTH[7:0]							
		15:8	DTH[15:8]							
		23:16	DTL[7:0]							
		31:24	DTL[15:8]							
0x027C	PWM_DTUPD3	7:0	DTHUPD[7:0]							
		15:8	DTHUPD[15:8]							
		23:16	DTLUPD[7:0]							
		31:24	DTLUPD[15:8]							
0x0280 ... 0x03FF	Reserved									
0x0400	PWM_CMUPD0	7:0								
		15:8			CPOLINVUP				CPOLUP	
		23:16								
		31:24								
0x0404 ... 0x041F	Reserved									
0x0420	PWM_CMUPD1	7:0								
		15:8			CPOLINVUP				CPOLUP	
		23:16								
		31:24								
0x0424 ... 0x042B	Reserved									
0x042C	PWM_ETRG1	7:0	MAXCNT[7:0]							
		15:8	MAXCNT[15:8]							
		23:16	MAXCNT[23:16]							
		31:24	RFEN	TRGSRC	TRGFILT	TRGEDGE			TRGMODE[1:0]	
0x0430	PWM_LEBR1	7:0		LEBDELAY[6:0]						
		15:8								
		23:16					PWMHREN	PWMHFEN	PWMLREN	PWMLFEN
		31:24								
0x0434 ... 0x043F	Reserved									
0x0440	PWM_CMUPD2	7:0								
		15:8			CPOLINVUP				CPOLUP	
		23:16								
		31:24								
0x0444 ... 0x044B	Reserved									
0x044C	PWM_ETRG2	7:0	MAXCNT[7:0]							
		15:8	MAXCNT[15:8]							
		23:16	MAXCNT[23:16]							
		31:24	RFEN	TRGSRC	TRGFILT	TRGEDGE			TRGMODE[1:0]	

.....continued

Offset	Name	Bit Pos.								
0x0450	PWM_LEBR2	7:0		LEBDELAY[6:0]						
		15:8								
		23:16					PWMHREN	PWMHFEN	PWMLREN	PWMLFEN
		31:24								
0x0454 ... 0x045F	Reserved									
0x0460	PWM_CMUPD3	7:0								
		15:8			CPOLINVUP				CPOLUP	
		23:16								
		31:24								

41.7.1 PWM Clock Register

Name: PWM_CLK
Offset: 0x00
Reset: 0x00000000
Property: Read/Write

This register can only be written if bits WPSWS0 and WPHWS0 are cleared in the PWM Write Protection Status Register.

Bit	31	30	29	28	27	26	25	24
					PREB[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DIVB[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
					PREA[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DIVA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 27:24 – PREB[3:0] CLKB Source Clock Selection

Value	Name	Description
0	CLK	Peripheral clock
1	CLK_DIV2	Peripheral clock/2
2	CLK_DIV4	Peripheral clock/4
3	CLK_DIV8	Peripheral clock/8
4	CLK_DIV16	Peripheral clock/16
5	CLK_DIV32	Peripheral clock/32
6	CLK_DIV64	Peripheral clock/64
7	CLK_DIV128	Peripheral clock/128
8	CLK_DIV256	Peripheral clock/256
9	CLK_DIV512	Peripheral clock/512
10	CLK_DIV1024	Peripheral clock/1024
Other	–	Reserved

Bits 23:16 – DIVB[7:0] CLKB Divide Factor

Value	Name	Description
0	CLKB_POFF	CLKB clock is turned off
1	PREB	CLKB clock is clock selected by PREB
2-255	PREB_DIV	CLKB clock is clock selected by PREB divided by DIVB factor

Bits 11:8 – PREA[3:0] CLKA Source Clock Selection

Value	Name	Description
0	CLK	Peripheral clock
1	CLK_DIV2	Peripheral clock/2
2	CLK_DIV4	Peripheral clock/4

Value	Name	Description
3	CLK_DIV8	Peripheral clock/8
4	CLK_DIV16	Peripheral clock/16
5	CLK_DIV32	Peripheral clock/32
6	CLK_DIV64	Peripheral clock/64
7	CLK_DIV128	Peripheral clock/128
8	CLK_DIV256	Peripheral clock/256
9	CLK_DIV512	Peripheral clock/512
10	CLK_DIV1024	Peripheral clock/1024
Other	–	Reserved

Bits 7:0 – DIVA[7:0] CLKA Divide Factor

Value	Name	Description
0	CLKA_POFF	CLKA clock is turned off
1	PREA	CLKA clock is clock selected by PREA
2-255	PREA_DIV	CLKA clock is clock selected by PREA divided by DIVA factor

41.7.2 PWM Enable Register

Name: PWM_ENA
Offset: 0x04
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CHID3	CHID2	CHID1	CHID0
Access					W	W	W	W
Reset					0	0	0	–

Bits 0, 1, 2, 3 – CHIDx Channel ID

Value	Description
0	No effect.
1	Enable PWM output for channel x.

41.7.3 PWM Disable Register

Name: PWM_DIS
Offset: 0x08
Reset: –
Property: Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CHID3	CHID2	CHID1	CHID0
Access					W	W	W	W
Reset					0	0	0	–

Bits 0, 1, 2, 3 – CHIDx Channel ID

Value	Description
0	No effect.
1	Disable PWM output for channel x.

41.7.4 PWM Status Register

Name: PWM_SR
Offset: 0x0C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CHID3	CHID2	CHID1	CHID0
Access					R	R	R	R
Reset					0	0	0	0

Bits 0, 1, 2, 3 – CHIDx Channel ID

Value	Description
0	PWM output for channel x is disabled.
1	PWM output for channel x is enabled.

41.7.5 PWM Interrupt Enable Register 1

Name: PWM_IER1
Offset: 0x10
Reset: –
Property: Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					FCHID3	FCHID2	FCHID1	FCHID0
Access					W	W	W	W
Reset					0	0	0	–

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CHID3	CHID2	CHID1	CHID0
Access					W	W	W	W
Reset					0	0	0	–

Bits 16, 17, 18, 19 – FCHIDx Fault Protection Trigger on Channel x Interrupt Enable

Bits 0, 1, 2, 3 – CHIDx Counter Event on Channel x Interrupt Enable

41.7.6 PWM Interrupt Disable Register 1

Name: PWM_IDR1
Offset: 0x14
Reset: –
Property: Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					FCHID3	FCHID2	FCHID1	FCHID0
Access					W	W	W	W
Reset					0	0	0	–

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CHID3	CHID2	CHID1	CHID0
Access					W	W	W	W
Reset					0	0	0	–

Bits 16, 17, 18, 19 – FCHIDx Fault Protection Trigger on Channel x Interrupt Disable

Bits 0, 1, 2, 3 – CHIDx Counter Event on Channel x Interrupt Disable

41.7.7 PWM Interrupt Mask Register 1

Name: PWM_IMR1
Offset: 0x18
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					FCHID3	FCHID2	FCHID1	FCHID0
Access					R	R	R	R
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CHID3	CHID2	CHID1	CHID0
Access					R	R	R	R
Reset					0	0	0	0

Bits 16, 17, 18, 19 – FCHIDx Fault Protection Trigger on Channel x Interrupt Mask

Bits 0, 1, 2, 3 – CHIDx Counter Event on Channel x Interrupt Mask

41.7.8 PWM Interrupt Status Register 1

Name: PWM_ISR1
Offset: 0x1C
Reset: 0x00000000
Property: Read-only

Note: Reading PWM_ISR1 automatically clears CHIDx and FCHIDx flags.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					FCHID3	FCHID2	FCHID1	FCHID0
Access					R	R	R	R
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					CHID3	CHID2	CHID1	CHID0
Access					R	R	R	R
Reset					0	0	0	0

Bits 16, 17, 18, 19 – FCHIDx Fault Protection Trigger on Channel x

Value	Description
0	No new trigger of the fault protection since the last read of PWM_ISR1.
1	At least one trigger of the fault protection since the last read of PWM_ISR1.

Bits 0, 1, 2, 3 – CHIDx Counter Event on Channel x

Value	Description
0	No new counter event has occurred since the last read of PWM_ISR1.
1	At least one counter event has occurred since the last read of PWM_ISR1.

41.7.9 PWM Sync Channels Mode Register

Name: PWM_SCM
Offset: 0x20
Reset: 0x00000000
Property: Read/Write

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	PTRCS[2:0]			PTRM			UPDM[1:0]	
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	0	0	0	0			0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					SYNC3	SYNC2	SYNC1	SYNC0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 23:21 – PTRCS[2:0] DMA Controller Transfer Request Comparison Selection
Selection of the comparison used to set the flag WRDY and the corresponding DMA Controller transfer request.

Bit 20 – PTRM DMA Controller Transfer Request Mode

UPDM	PTRM	WRDY Flag and DMA Controller Transfer Request
0	x	The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are never set to '1'.
1	x	The WRDY flag in PWM Interrupt Status Register 2 is set to '1' as soon as the update period is elapsed, the DMA Controller transfer request is never set to '1'.
2	0	The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to '1' as soon as the update period is elapsed.
	1	The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to '1' as soon as the selected comparison matches.

Bits 17:16 – UPDM[1:0] Synchronous Channels Update Mode

Value	Name	Description
0	MODE0	Manual write of double buffer registers and manual update of synchronous channels 1[(1)].
1	MODE1	Manual write of double buffer registers and automatic update of synchronous channels 2[(2)].
2	MODE2	The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to '1' as soon as the update period is elapsed.

Note:

1. The update occurs at the beginning of the next PWM period, when the UPDULOCK bit in PWM Sync Channels Update Control Register is set.
2. The update occurs when the Update Period is elapsed.

Bits 0, 1, 2, 3 – SYNCx Synchronous Channel x

Value	Description
0	Channel x is not a synchronous channel.
1	Channel x is a synchronous channel.

41.7.10 PWM DMA Register

Name: PWM_DMAR
Offset: 0x24
Reset: –
Property: Write-only

Only the first 16 bits (channel counter size) are significant.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	DMADUTY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DMADUTY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DMADUTY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 23:0 – DMADUTY[23:0] Duty-Cycle Holding Register for DMA Access
Each write access to PWM_DMAR sequentially updates PWM_CDTYUPDx.CDTYUPD with DMADUTY (only for channel configured as synchronous). See “Method 3: Automatic write of duty-cycle values and automatic trigger of the update” .

41.7.11 PWM Sync Channels Update Control Register

Name: PWM_SCUC
Offset: 0x28
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
Access								R/W
Reset								0

Bit 0 – UPDULOCK Synchronous Channels Update Unlock
This bit is automatically reset when the update is done.

Value	Description
0	No effect
1	If the UPDM field is set to ‘0’ in PWM Sync Channels Mode Register, writing the UPDULOCK bit to ‘1’ triggers the update of the period value, the duty-cycle and the dead-time values of synchronous channels at the beginning of the next PWM period. If the field UPDM is set to ‘1’ or ‘2’, writing the UPDULOCK bit to ‘1’ triggers only the update of the period value and of the dead-time values of synchronous channels.

41.7.12 PWM Sync Channels Update Period Register

Name: PWM_SCUP
Offset: 0x2C
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	UPRCNT[3:0]				UPR[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:4 – UPRCNT[3:0] Update Period Counter
Reports the value of the update period counter.

Bits 3:0 – UPR[3:0] Update Period
Defines the time between each update of the synchronous channels if automatic trigger of the update is activated (UPDM = 1 or UPDM = 2 in PWM Sync Channels Mode Register). This time is equal to UPR+1 periods of the synchronous channels.

41.7.13 PWM Sync Channels Update Period Update Register

Name: PWM_SCUPUPD
Offset: 0x30
Reset: –
Property: Write-only

This register acts as a double buffer for the UPR value. This prevents an unexpected automatic trigger of the update of synchronous channels.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					UPRUPD[3:0]			
Access					W	W	W	W
Reset					0	0	0	–

Bits 3:0 – UPRUPD[3:0] Update Period Update
Defines the wanted time between each update of the synchronous channels if automatic trigger of the update is activated (UPDM = 1 or UPDM = 2 in PWM Sync Channels Mode Register). This time is equal to UPR+1 periods of the synchronous channels.

41.7.14 PWM Interrupt Enable Register 2

Name: PWM_IER2
Offset: 0x34
Reset: –
Property: Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bit	15	14	13	12	11	10	9	8
	CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bit	7	6	5	4	3	2	1	0
					UNRE			WRDY
Access					W			W
Reset					–			–

Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx Comparison x Update Interrupt Enable

Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx Comparison x Match Interrupt Enable

Bit 3 – UNRE Synchronous Channels Update Underrun Error Interrupt Enable

Bit 0 – WRDY Write Ready for Synchronous Channels Update Interrupt Enable

41.7.15 PWM Interrupt Disable Register 2

Name: PWM_IDR2
Offset: 0x38
Reset: –
Property: Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bit	15	14	13	12	11	10	9	8
	CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bit	7	6	5	4	3	2	1	0
					UNRE			WRDY
Access					W			W
Reset					–			–

Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx Comparison x Update Interrupt Disable

Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx Comparison x Match Interrupt Disable

Bit 3 – UNRE Synchronous Channels Update Underrun Error Interrupt Disable

Bit 0 – WRDY Write Ready for Synchronous Channels Update Interrupt Disable

41.7.16 PWM Interrupt Mask Register 2

Name: PWM_IMR2
Offset: 0x3C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					UNRE			WRDY
Access					R			R
Reset					0			0

Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx Comparison x Update Interrupt Mask

Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx Comparison x Match Interrupt Mask

Bit 3 – UNRE Synchronous Channels Update Underrun Error Interrupt Mask

Bit 0 – WRDY Write Ready for Synchronous Channels Update Interrupt Mask

41.7.17 PWM Interrupt Status Register 2

Name: PWM_ISR2
Offset: 0x40
Reset: 0x00000000
Property: Read-only

Reading PWM_ISR2 automatically clears flags WRDY, UNRE and CMPSx.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					UNRE			WRDY
Access					R			R
Reset					0			0

Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx Comparison x Update

Value	Description
0	The comparison x has not been updated since the last read of the PWM_ISR2 register.
1	The comparison x has been updated at least one time since the last read of the PWM_ISR2 register.

Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx Comparison x Match

Value	Description
0	The comparison x has not matched since the last read of the PWM_ISR2 register.
1	The comparison x has matched at least one time since the last read of the PWM_ISR2 register.

Bit 3 – UNRE Synchronous Channels Update Underrun Error

Value	Description
0	No Synchronous Channels Update Underrun has occurred since the last read of the PWM_ISR2 register.
1	At least one Synchronous Channels Update Underrun has occurred since the last read of the PWM_ISR2 register.

Bit 0 – WRDY Write Ready for Synchronous Channels Update

Value	Description
0	New duty-cycle and dead-time values for the synchronous channels cannot be written.
1	New duty-cycle and dead-time values for the synchronous channels can be written.

41.7.18 PWM Output Override Value Register

Name: PWM_OOV
Offset: 0x44
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					OOVL3	OOVL2	OOVL1	OOVL0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OOVH3	OOVH2	OOVH1	OOVH0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 16, 17, 18, 19 – OOVLx Output Override Value for PWML output of the channel x

Value	Description
0	Override value is 0 for PWML output of channel x.
1	Override value is 1 for PWML output of channel x.

Bits 0, 1, 2, 3 – OOVHx Output Override Value for PWMH output of the channel x

Value	Description
0	Override value is 0 for PWMH output of channel x.
1	Override value is 1 for PWMH output of channel x.

41.7.19 PWM Output Selection Register

Name: PWM_OS
Offset: 0x48
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					OSL3	OSL2	OSL1	OSL0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OSH3	OSH2	OSH1	OSH0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 16, 17, 18, 19 – OSLx Output Selection for PWML output of the channel x

Value	Description
0	Dead-time generator output DTOLx selected as PWML output of channel x.
1	Output override value OOVLx selected as PWML output of channel x.

Bits 0, 1, 2, 3 – OSHx Output Selection for PWMH output of the channel x

Value	Description
0	Dead-time generator output DTOHx selected as PWMH output of channel x.
1	Output override value OOVHx selected as PWMH output of channel x.

41.7.20 PWM Output Selection Set Register

Name: PWM_OSS
Offset: 0x4C
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					OSSL3	OSSL2	OSSL1	OSSL0
Access					W	W	W	W
Reset					0	0	0	–

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OSSH3	OSSH2	OSSH1	OSSH0
Access					W	W	W	W
Reset					0	0	0	–

Bits 16, 17, 18, 19 – OSSLx Output Selection Set for PWML output of the channel x

Value	Description
0	No effect.
1	Output override value OOVLx selected as PWML output of channel x.

Bits 0, 1, 2, 3 – OSSHx Output Selection Set for PWMH output of the channel x

Value	Description
0	No effect.
1	Output override value OOVHx selected as PWMH output of channel x.

41.7.21 PWM Output Selection Clear Register

Name: PWM_OSC
Offset: 0x50
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					OSCL3	OSCL2	OSCL1	OSCL0
Access					W	W	W	W
Reset					0	0	0	–

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OSCH3	OSCH2	OSCH1	OSCH0
Access					W	W	W	W
Reset					0	0	0	–

Bits 16, 17, 18, 19 – OSCLx Output Selection Clear for PWML output of the channel x

Value	Description
0	No effect.
1	Dead-time generator output DTOLx selected as PWML output of channel x.

Bits 0, 1, 2, 3 – OSCHx Output Selection Clear for PWMH output of the channel x

Value	Description
0	No effect.
1	Dead-time generator output DTOHx selected as PWMH output of channel x.

41.7.22 PWM Output Selection Set Update Register

Name: PWM_OSSUPD
Offset: 0x54
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					OSSUPL3	OSSUPL2	OSSUPL1	OSSUPL0
Access					W	W	W	W
Reset					0	0	0	–

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OSSUPH3	OSSUPH2	OSSUPH1	OSSUPH0
Access					W	W	W	W
Reset					0	0	0	–

Bits 16, 17, 18, 19 – OSSUPLx Output Selection Set for PWML output of the channel x

Value	Description
0	No effect.
1	Output override value OOVLx selected as PWML output of channel x at the beginning of the next channel x PWM period.

Bits 0, 1, 2, 3 – OSSUPHx Output Selection Set for PWMH output of the channel x

Value	Description
0	No effect.
1	Output override value OOVHx selected as PWMH output of channel x at the beginning of the next channel x PWM period.

41.7.23 PWM Output Selection Clear Update Register

Name: PWM_OSCUPD
Offset: 0x58
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					OSCUPL3	OSCUPL2	OSCUPL1	OSCUPL0
Access					W	W	W	W
Reset					0	0	0	–

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					OSCUPH3	OSCUPH2	OSCUPH1	OSCUPH0
Access					W	W	W	W
Reset					0	0	0	–

Bits 16, 17, 18, 19 – OSCUPLx Output Selection Clear for PWML output of the channel x

Value	Description
0	No effect.
1	Dead-time generator output DTOLx selected as PWML output of channel x at the beginning of the next channel x PWM period.

Bits 0, 1, 2, 3 – OSCUPHx Output Selection Clear for PWMH output of the channel x

Value	Description
0	No effect.
1	Dead-time generator output DTOHx selected as PWMH output of channel x at the beginning of the next channel x PWM period.

41.7.24 PWM Fault Mode Register

Name: PWM_FMR
Offset: 0x5C
Reset: 0x00000000
Property: Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

See Fault Inputs for details on fault generation.

⚠ CAUTION To prevent an unexpected activation of the status flag FSy in the PWM Fault Status Register, the bit FMODy can be set to '1' only if the FPOLy bit has been previously configured to its final value.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	FFIL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FMOD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FPOL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:16 – FFIL[7:0] Fault Filtering
For each bit y of FFIL, where y is the fault input number:
0: The fault input y is not filtered.
1: The fault input y is filtered.

Bits 15:8 – FMOD[7:0] Fault Activation Mode
For each bit y of FMOD, where y is the fault input number:
0: The fault y is active until the fault condition is removed at the peripheral(1) level.
1: The fault y stays active until the fault condition is removed at the peripheral level(1)AND until it is cleared in the PWM Fault Clear Register.
Note:
1. The peripheral generating the fault.

Bits 7:0 – FPOL[7:0] Fault Polarity
For each bit y of FPOL, where y is the fault input number:
0: The fault y becomes active when the fault input y is at 0.
1: The fault y becomes active when the fault input y is at 1.

41.7.25 PWM Fault Status Register

Name: PWM_FSR
Offset: 0x60
Reset: 0x00000000
Property: Read-only

Refer to Fault Inputs for details on fault generation.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	FS[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FIV[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 15:8 – FS[7:0] Fault Status
For each bit y of FS, where y is the fault input number:
0: The fault y is not currently active.
1: The fault y is currently active.

Bits 7:0 – FIV[7:0] Fault Input Value
For each bit y of FIV, where y is the fault input number:
0: The current sampled value of the fault input y is 0 (after filtering if enabled).
1: The current sampled value of the fault input y is 1 (after filtering if enabled).

41.7.26 PWM Fault Clear Register

Name: PWM_FCR
Offset: 0x64
Reset: –
Property: Write-only

See Fault Inputs for details on fault generation.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	FCLR[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 7:0 – FCLR[7:0] Fault Clear
For each bit y of FCLR, where y is the fault input number:
0: No effect.
1: If bit y of FMOD field is set to ‘1’ and if the fault input y is not at the level defined by the bit y of FPOL field, the fault y is cleared and becomes inactive (FMOD and FPOL fields belong to PWM Fault Mode Register), else writing this bit to ‘1’ has no effect.

41.7.27 PWM Fault Protection Value Register 1

Name: PWM_FPV1
Offset: 0x68
Reset: 0x00000000
Property: Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

See Fault Inputs for details on fault generation.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					FPVL3	FPVL2	FPVL1	FPVL0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					FPVH3	FPVH2	FPVH1	FPVH0
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bits 16, 17, 18, 19 – FPVLx Fault Protection Value for PWML output on channel x
This bit is taken into account only if the bit FPZLx is set to ‘0’ in PWM Fault Protection Value Register 2.

Value	Description
0	PWML output of channel x is forced to ‘0’ when fault occurs.
1	PWML output of channel x is forced to ‘1’ when fault occurs.

Bits 0, 1, 2, 3 – FPVHx Fault Protection Value for PWMH output on channel x
This bit is taken into account only if the bit FPZHx is set to ‘0’ in PWM Fault Protection Value Register 2.

Value	Description
0	PWMH output of channel x is forced to ‘0’ when fault occurs.
1	PWMH output of channel x is forced to ‘1’ when fault occurs.

41.7.28 PWM Fault Protection Enable Register

Name: PWM_FPE
Offset: 0x6C
Reset: 0x00000000
Property: Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Only the first 7 bits (number of fault input pins) of the register fields are significant.

Refer to Section 6.4 “Fault Inputs” for details on fault generation.

Bit	31	30	29	28	27	26	25	24
	FPE3[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	FPE2[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	FPE1[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	FPE0[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 0:7, 8:15, 16:23, 24:31 – FPEx Fault Protection Enable for channel x
For each bit y of FPEx, where y is the fault input number:
0: Fault y is not used for the fault protection of channel x.
1: Fault y is used for the fault protection of channel x.

⚠ CAUTION To prevent an unexpected activation of the fault protection, the bit y of FPEx field can be set to ‘1’ only if the corresponding FPOL field has been previously configured to its final value in PWM Fault Mode Register.

41.7.29 PWM Event Line x Mode Register

Name: PWM_ELMRx
Offset: 0x7C + x*0x04 [x=0..1]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
	CSEL7	CSEL6	CSEL5	CSEL4	CSEL3	CSEL2	CSEL1	CSEL0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 0, 1, 2, 3, 4, 5, 6, 7 – CSELy Comparison y Selection

Value	Description
0	A pulse is not generated on the event line x when the comparison y matches.
1	A pulse is generated on the event line x when the comparison y match.

41.7.30 PWM Spread Spectrum Register

Name: PWM_SSPR
Offset: 0xA0
Reset: 0x00000000
Property: Read/Write

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (channel counter size) are significant.

Bit	31	30	29	28	27	26	25	24
								SPRDM
Access								R/W
Reset								0

Bit	23	22	21	20	19	18	17	16
	SPRD[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SPRD[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SPRD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 24 – SPRDM Spread Spectrum Counter Mode

Value	Description
0	Triangular mode. The spread spectrum counter starts to count from -SPRD when the channel 0 is enabled and counts upwards at each PWM period. When it reaches +SPRD, it restarts to count from -SPRD again.
1	Random mode. The spread spectrum counter is loaded with a new random value at each PWM period. This random value is uniformly distributed and is between -SPRD and +SPRD.

Bits 23:0 – SPRD[23:0] Spread Spectrum Limit Value
The spread spectrum limit value defines the range for the spread spectrum counter. It is introduced in order to achieve constant varying PWM period for the output waveform.

41.7.31 PWM Spread Spectrum Update Register

Name: PWM_SSPUP
Offset: 0xA4
Reset: –
Property: Write-only

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the SPRD value. This prevents an unexpected waveform when modifying the spread spectrum limit value.

Only the first 16 bits (channel counter size) are significant.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	SPRDUP[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	SPRDUP[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	SPRDUP[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 23:0 – SPRDUP[23:0] Spread Spectrum Limit Value Update
The spread spectrum limit value defines the range for the spread spectrum counter. It is introduced in order to achieve constant varying period for the output waveform.

41.7.32 PWM Stepper Motor Mode Register

Name: PWM_SMMR
Offset: 0xB0
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
							DOWN1	DOWN0
Access								
Reset							0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
							GCEN1	GCEN0
Access							R/W	R/W
Reset							0	0

Bits 16, 17 – DOWNx Down Count

Value	Description
0	Up counter.
1	Down counter.

Bits 0, 1 – GCENx Gray Count Enable

Value	Description
0	Disable Gray count generation on PWML[2*x], PWMH[2*x], PWML[2*x +1], PWMH[2*x +1]
1	Enable Gray count generation on PWML[2*x], PWMH[2*x], PWML[2*x +1], PWMH[2*x +1].

41.7.33 PWM Fault Protection Value Register 2

Name: PWM_FPV2
Offset: 0xC0
Reset: 0x000F000F
Property: Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					FPZL3	FPZL2	FPZL1	FPZL0
Access					R/W	R/W	R/W	R/W
Reset					1	1	1	1

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
					FPZH3	FPZH2	FPZH1	FPZH0
Access					R/W	R/W	R/W	R/W
Reset					1	1	1	1

Bits 16, 17, 18, 19 – FPZLx Fault Protection to Hi-Z for PWML output on channel x

Value	Description
0	When fault occurs, PWML output of channel x is forced to value defined by the bit FPVLx in PWM Fault Protection Value Register 1.
1	When fault occurs, PWML output of channel x is forced to high-impedance state.

Bits 0, 1, 2, 3 – FPZHx Fault Protection to Hi-Z for PWMH output on channel x

Value	Description
0	When fault occurs, PWMH output of channel x is forced to value defined by the bit FPVHx in PWM Fault Protection Value Register 1.
1	When fault occurs, PWMH output of channel x is forced to high-impedance state.

41.7.34 PWM Write Protection Control Register

Name: PWM_WPCR
Offset: 0xE4
Reset: –
Property: Write-only

See Register Write Protection for the list of registers that can be write-protected.

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bit	7	6	5	4	3	2	1	0
	WPRG5	WPRG4	WPRG3	WPRG2	WPRG1	WPRG0	WPCMD[1:0]	
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	–	0	–

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x50574D	PASSWD	Writing any other value in this field aborts the write operation of the WPCMD field. Always reads as 0

Bits 2, 3, 4, 5, 6, 7 – WPRGx Write Protection Register Group x

Value	Description
0	The WPCMD command has no effect on the register group x.
1	The WPCMD command is applied to the register group x.

Bits 1:0 – WPCMD[1:0] Write Protection Command
This command is performed only if the WPKEY corresponds to 0x50574D (“PWM” in ASCII).

Value	Name	Description
0	DISABLE_SW_PROT	Disables the software write protection of the register groups of which the bit WPRGx is at ‘1’.
1	ENABLE_SW_PROT	Enables the software write protection of the register groups of which the bit WPRGx is at ‘1’.
2	ENABLE_HW_PROT	Enables the hardware write protection of the register groups of which the bit WPRGx is at ‘1’. Only a hardware reset of the PWM controller can disable the hardware write protection. Moreover, to meet security requirements, the PIO lines associated with the PWM can not be configured through the PIO interface.

41.7.35 PWM Write Protection Status Register

Name: PWM_WPSR
Offset: 0xE8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	WPVSRC[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
			WPHWS5	WPHWS4	WPHWS3	WPHWS2	WPHWS1	WPHWS0
Access			R	R	R	R	R	R
Reset			0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	WPVS		WPSWS5	WPSWS4	WPSWS3	WPSWS2	WPSWS1	WPSWS0
Access	R		R	R	R	R	R	R
Reset	0		0	0	0	0	0	0

Bits 31:16 – WPVSRC[15:0] Write Protect Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bits 8, 9, 10, 11, 12, 13 – WPHWSx Write Protect HW Status

Value	Description
0	The HW write protection x of the register group x is disabled.
1	The HW write protection x of the register group x is enabled.

Bit 7 – WPVS Write Protect Violation Status

Value	Description
0	No write protection violation has occurred since the last read of PWM_WPSR.
1	At least one write protection violation has occurred since the last read of PWM_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

Bits 0, 1, 2, 3, 4, 5 – WPSWSx Write Protect SW Status

Value	Description
0	The SW write protection x of the register group x is disabled.
1	The SW write protection x of the register group x is enabled.

41.7.36 PWM Comparison x Value Register

Name: PWM_CMPVx
Offset: 0x0130 + x*0x10 [x=0..7]
Reset: 0x00000000
Property: Read/Write

Only the first 16 bits (channel counter size) of field CV are significant.

Bit	31	30	29	28	27	26	25	24
								CVM
Access								R/W
Reset								0

Bit	23	22	21	20	19	18	17	16
	CV[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CV[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CV[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 24 – CVM Comparison x Value Mode

Value	Description
0	The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is incrementing.
1	The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is decrementing. Note: This bit is not relevant if the counter of the channel 0 is left-aligned (CALG = 0 in PWM Channel Mode Register)

Bits 23:0 – CV[23:0] Comparison x Value
Define the comparison x value to be compared with the counter of the channel 0.

41.7.37 PWM Comparison x Value Update Register

Name: PWM_CMPVUPDx
Offset: 0x0134 + x*0x10 [x=0..7]
Reset: –
Property: Write-only

This register acts as a double buffer for the CV and CVM values. This prevents an unexpected comparison x match.

Only the first 16 bits (channel counter size) of field CVUPD are significant.

⚠ CAUTION The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

Bit	31	30	29	28	27	26	25	24
								CVMUPD
Access								W
Reset								–

Bit	23	22	21	20	19	18	17	16
	CVUPD[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CVUPD[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CVUPD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bit 24 – CVMUPD Comparison x Value Mode Update
Note: This bit is not relevant if the counter of the channel 0 is left-aligned (CALG = 0 in PWM Channel Mode Register)

Value	Description
0	The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is incrementing.
1	The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is decrementing.

Bits 23:0 – CVUPD[23:0] Comparison x Value Update
Define the comparison x value to be compared with the counter of the channel 0.

41.7.38 PWM Comparison x Mode Register

Name: PWM_CMPMx
Offset: 0x0138 + x*0x10 [x=0..7]
Reset: 0x00000000
Property: R/W

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CUPRCNT[3:0]				CUPR[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CPRCNT[3:0]				CPR[3:0]			
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CTR[3:0]							CEN
Access	R/W	R/W	R/W	R/W				R/W
Reset	0	0	0	0				0

Bits 23:20 – CUPRCNT[3:0] Comparison x Update Period Counter
Reports the value of the comparison x update period counter.
Note: The field CUPRCNT is read-only

Bits 19:16 – CUPR[3:0] Comparison x Update Period
Defines the time between each update of the comparison x mode and the comparison x value. This time is equal to CUPR+1 periods of the channel 0 counter.

Bits 15:12 – CPRCNT[3:0] Comparison x Period Counter
Reports the value of the comparison x period counter.
Note: The field CPRCNT is read-only

Bits 11:8 – CPR[3:0] Comparison x Period
CPR defines the maximum value of the comparison x period counter (CPRCNT). The comparison x value is performed periodically once every CPR+1 periods of the channel 0 counter.

Bits 7:4 – CTR[3:0] Comparison x Trigger
The comparison x is performed when the value of the comparison x period counter (CPRCNT) reaches the value defined by CTR.

Bit 0 – CEN Comparison x Enable

Value	Description
0	The comparison x is disabled and can not match.
1	The comparison x is enabled and can match.

41.7.39 PWM Comparison x Mode Update Register

Name: PWM_CMPMUPDx
Offset: 0x013C + x*0x10 [x=0..7]
Reset: –
Property: W

This register acts as a double buffer for the CEN, CTR, CPR and CUPR values. This prevents an unexpected comparison x match.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					CUPRUPD[3:0]			
Access					W	W	W	W
Reset					0	0	0	–

Bit	15	14	13	12	11	10	9	8
					CPRUPD[3:0]			
Access					W	W	W	W
Reset					0	0	0	–

Bit	7	6	5	4	3	2	1	0
	CTRUPD[3:0]							CENUPD
Access	W	W	W	W				W
Reset	0	0	0	–				–

Bits 19:16 – CUPRUPD[3:0] Comparison x Update Period Update
Defines the time between each update of the comparison x mode and the comparison x value. This time is equal to CUPR+1 periods of the channel 0 counter.

Bits 11:8 – CPRUPD[3:0] Comparison x Period Update
CPR defines the maximum value of the comparison x period counter (CPRCNT). The comparison x value is performed periodically once every CPR+1 periods of the channel 0 counter.

Bits 7:4 – CTRUPD[3:0] Comparison x Trigger Update
The comparison x is performed when the value of the comparison x period counter (CPRCNT) reaches the value defined by CTR.

Bit 0 – CENUPD Comparison x Enable Update

Value	Description
0	The comparison x is disabled and can not match.
1	The comparison x is enabled and can match.

41.7.40 PWM Channel Mode Register

Name: PWM_CMRx
Offset: 0x0200 + x*0x20 [x=0..3]
Reset: 0x00000000
Property: Read/Write

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					PPM	DTLI	DTHI	DTE
Access					R/W			
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
			TCTS	DPOLI	UPDS	CES	CPOL	CALG
Access					R/W	R/W	R/W	R/W
Reset			0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
					CPRE[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit 19 – PPM Push-Pull Mode
The Push-Pull mode is enabled for channel x.

Value	Description
0	The Push-Pull mode is disabled for channel x.
1	The Push-Pull mode is enabled for channel x.

Bit 18 – DTLI Dead-Time PWMLx Output Inverted

Value	Description
0	The dead-time PWMLx output is not inverted.
1	The dead-time PWMLx output is inverted.

Bit 17 – DTHI Dead-Time PWMHx Output Inverted

Value	Description
0	The dead-time PWMHx output is not inverted.
1	The dead-time PWMHx output is inverted.

Bit 16 – DTE Dead-Time Generator Enable

Value	Description
0	The dead-time generator is disabled.
1	The dead-time generator is enabled.

Bit 13 – TCTS Timer Counter Trigger Selection

Value	Description
0	The comparator of the channel x (OCx) is used as the trigger source for the Timer Counter (TC).
1	The counter events of the channel x is used as the trigger source for the Timer Counter (TC).

Bit 12 – DPOLI Disabled Polarity Inverted

Value	Description
0	When the PWM channel x is disabled (CHIDx(PWM_SR) = 0), the OCx output waveform is the same as the one defined by the CPOL bit.
1	When the PWM channel x is disabled (CHIDx(PWM_SR) = 0), the OCx output waveform is inverted compared to the one defined by the CPOL bit.

Bit 11 – UPDS Update Selection
If the PWM period is center-aligned (CALG=1):
0: The update occurs at the next end of the PWM period after writing the update register(s).
1: The update occurs at the next end of the PWM half period after writing the update register(s).
If the PWM period is left-aligned (CALG=0), the update always occurs at the end of the PWM period after writing the update register(s).

Bit 10 – CES Counter Event Selection
If the PWM period is center-aligned (CALG=1):
0: The channel counter event occurs at the end of the PWM period.
1: The channel counter event occurs at the end of the PWM period and at half the PWM period.
If the PWM period is left-aligned (CALG=0), the channel counter event occurs at the end of the period and the CES bit has no effect.

Bit 9 – CPOL Channel Polarity

Value	Description
0	The OCx output waveform (output from the comparator) starts at a low level.
1	The OCx output waveform (output from the comparator) starts at a high level.

Bit 8 – CALG Channel Alignment

Value	Description
0	The period is left-aligned.
1	The period is center-aligned.

Bits 3:0 – CPRE[3:0] Channel Prescaler

Value	Name	Description
	MCK	Peripheral clock
1	MCK_DIV_2	Peripheral clock/2
2	MCK_DIV_4	Peripheral clock/4
3	MCK_DIV_8	Peripheral clock/8
4	MCK_DIV_16	Peripheral clock/16
5	MCK_DIV_32	Peripheral clock/32
6	MCK_DIV_64	Peripheral clock/64
7	MCK_DIV_128	Peripheral clock/128
8	MCK_DIV_256	Peripheral clock/256
9	MCK_DIV_512	Peripheral clock/512
10	MCK_DIV_1024	Peripheral clock/1024
11	CLKA	Clock A
12	CLKB	Clock B

41.7.41 PWM Channel Duty Cycle Register

Name: PWM_CDTYx
Offset: 0x0204 + x*0x20 [x=0..3]
Reset: 0x00000000
Property: Read/Write

Only the first 16 bits (channel counter size) are significant.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CDTY[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CDTY[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CDTY[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – CDTY[23:0] Channel Duty-Cycle
Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

41.7.42 PWM Channel Duty Cycle Update Register

Name: PWM_CDTYUPDx
Offset: 0x0208 + x*0x20 [x=0..3]
Reset: –
Property: Write-only

This register acts as a double buffer for the CDTY value. This prevents an unexpected waveform when modifying the waveform duty-cycle.

Only the first 16 bits (channel counter size) are significant.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CDTYUPD[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CDTYUPD[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CDTYUPD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 23:0 – CDTYUPD[23:0] Channel Duty-Cycle Update
Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

41.7.43 PWM Channel Period Register

Name: PWM_CPRDx
Offset: 0x020C + x*0x20 [x=0..3]
Reset: 0x00000000
Property: Read/Write

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (channel counter size) are significant.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CPRD[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CPRD[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CPRD[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – CPRD[23:0] Channel Period
If the waveform is left-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:
– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(X \times \mathrm{CPRD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

If the waveform is center-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:
– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(2 \times X \times \mathrm{CPRD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

41.7.44 PWM Channel Period Update Register

Name: PWM_CPRDUPDx
Offset: 0x0210 + x*0x20 [x=0..3]
Reset: –
Property: Write-only

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the CPRD value. This prevents an unexpected waveform when modifying the waveform period.

Only the first 16 bits (channel counter size) are significant.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CPRDUPD[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CPRDUPD[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CPRDUPD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 23:0 – CPRDUPD[23:0] Channel Period Update
If the waveform is left-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:
– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(X \times \mathrm{CPRDUPD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(X \times \mathrm{CPRDUPD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(X \times \mathrm{CPRDUPD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

If the waveform is center-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:
– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(2 \times X \times \mathrm{CPRDUPD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(2 \times X \times \text{CPRDUPD} \times \text{DIVA})}{f_{\text{peripheral clock}}} \text{ or } \frac{(2 \times X \times \text{CPRDUPD} \times \text{DIVB})}{f_{\text{peripheral clock}}}$$

41.7.45 PWM Channel Counter Register

Name: PWM_CCNTx
Offset: 0x0214 + x*0x20 [x=0..3]
Reset: 0x00000000
Property: Read-only

Only the first 16 bits (channel counter size) are significant.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
	CNT[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	CNT[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	CNT[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 23:0 – CNT[23:0] Channel Counter Register
Channel counter value. This register is reset when:
• the channel is enabled (writing CHIDx in the PWM_ENA register).
• the channel counter reaches CPRD value defined in the PWM_CPRDx register if the waveform is left-aligned.

41.7.46 PWM Channel Dead Time Register

Name: PWM_DTx
Offset: 0x0218 + x*0x20 [x=0..3]
Reset: 0x00000000
Property: Read/Write

This register can only be written if bits WPSWS4 and WPHWS4 are cleared in the PWM Write Protection Status Register.

Only the first 12 bits (dead-time counter size) of fields DTH and DTL are significant.

Bit	31	30	29	28	27	26	25	24
	DTL[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DTL[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DTH[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DTH[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:16 – DTL[15:0] Dead-Time Value for PWMLx Output
Defines the dead-time value for PWMLx output. This value must be defined between 0 and CDTY (PWM_CDTYx).

Bits 15:0 – DTH[15:0] Dead-Time Value for PWMHx Output
Defines the dead-time value for PWMHx output. This value must be defined between 0 and the value (CPRD – CDTY) (PWM_CPRDx and PWM_CDTYx).

41.7.47 PWM Channel Dead Time Update Register

Name: PWM_DTUPDx
Offset: 0x021C + x*0x20 [x=0..3]
Reset: 0x00000000
Property: Write-only

This register can only be written if bits WPSWS4 and WPHWS4 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the DTH and DTL values. This prevents an unexpected waveform when modifying the dead-time values.

Only the first 12 bits (dead-time counter size) of fields DTHUPD and DTLUPD are significant.

Bit	31	30	29	28	27	26	25	24
	DTLUPD[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DTLUPD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bit	15	14	13	12	11	10	9	8
	DTHUPD[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DTHUPD[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 31:16 – DTLUPD[15:0] Dead-Time Value Update for PWMLx Output
Defines the dead-time value for PWMLx output. This value must be defined between 0 and CDTY (PWM_CDTYx). This value is applied only at the beginning of the next channel x PWM period.

Bits 15:0 – DTHUPD[15:0] Dead-Time Value Update for PWMHx Output
Defines the dead-time value for PWMHx output. This value must be defined between 0 and the value (CPRD – CDTY) (PWM_CPRDx and PWM_CDTYx). This value is applied only at the beginning of the next channel x PWM period.

41.7.48 PWM Channel Mode Update Register

Name: PWM_CMUPDx
Offset: 0x0400 + x*0x20 [x=0..3]
Reset: –
Property: Write-only

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the CPOL value. This prevents an unexpected waveform when modifying the polarity value.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
			CPOLINVUP				CPOLUP	
Access			W				W	
Reset			–				–	

Bit	7	6	5	4	3	2	1	0
Access								
Reset								

Bit 13 – CPOLINVUP Channel Polarity Inversion Update
If this bit is written at ‘1’, the write of the bit CPOLUP is not taken into account.

Value	Description
0	No effect.
1	The OCx output waveform (output from the comparator) is inverted.

Bit 9 – CPOLUP Channel Polarity Update
The write of this bit is taken into account only if the bit CPOLINVUP is written at ‘0’ at the same time.

Value	Description
0	The OCx output waveform (output from the comparator) starts at a low level.
1	The OCx output waveform (output from the comparator) starts at a high level.

41.7.49 PWM External Trigger Register

Name: PWM_ETRGx
Offset: 0x042C + (x-1)*0x20 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	RFEN	TRGSRC	TRGFILT	TRGEDGE			TRGMODE[1:0]	
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	0	0	0	0			0	0

Bit	23	22	21	20	19	18	17	16
	MAXCNT[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MAXCNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MAXCNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit 31 – RFEN Recoverable Fault Enable

Value	Description
0	The TRGINx signal does not generate a recoverable fault.
1	The TRGINx signal generate a recoverable fault in place of the fault x input.

Bit 30 – TRGSRC Trigger Source

Value	Description
0	The TRGINx signal is driven by the PWMEXTRGx input.
1	The TRGINx signal is driven by the Analog Comparator Controller.

Bit 29 – TRGFILT Filtered input

Value	Description
0	The external trigger input x is not filtered.
1	The external trigger input x is filtered.

Bit 28 – TRGEDGE Edge Selection

Value	Name	Description
0	FALLING_ZERO	TRGMODE = 1: TRGINx event detection on falling edge. TRGMODE = 2, 3: TRGINx active level is 0
1	RISING_ONE	TRGMODE = 1: TRGINx event detection on rising edge. TRGMODE = 2, 3: TRGINx active level is 1

Bits 25:24 – TRGMODE[1:0] External Trigger Mode

Value	Name	Description
0	OFF	External trigger is not enabled.
1	MODE1	External PWM Reset Mode
2	MODE2	External PWM Start Mode
3	MODE3	Cycle-by-cycle Duty Mode

Bits 23:0 – MAXCNT[23:0] Maximum Counter value
Maximum channel x counter value measured at the TRGINx event since the last read of the register.
At the TRGINx event, if the channel x counter value is greater than the stored MAXCNT value, then MAXCNT is updated by the channel x counter value.

41.7.50 PWM Leading-Edge Blanking Register

Name: PWM_LEBRx
Offset: 0x0430 + (x-1)*0x20 [x=1..2]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
					PWMHREN	PWMHFEN	PWMLREN	PWMLFEN
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
		LEBDELAY[6:0]						
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

Bit 19 – PWMHREN PWMH Rising Edge Enable
Leading-edge blanking is enabled on PWMHx output rising edge.

Value	Description
0	Leading-edge blanking is disabled on PWMHx output rising edge.
1	Leading-edge blanking is enabled on PWMHx output rising edge.

Bit 18 – PWMHFEN PWMH Falling Edge Enable

Value	Description
0	Leading-edge blanking is disabled on PWMHx output falling edge.
1	Leading-edge blanking is enabled on PWMHx output falling edge.

Bit 17 – PWMLREN PWML Rising Edge Enable

Value	Description
0	Leading-edge blanking is disabled on PWMLx output rising edge.
1	Leading-edge blanking is enabled on PWMLx output rising edge.

Bit 16 – PWMLFEN PWML Falling Edge Enable

Value	Description
0	Leading-edge blanking is disabled on PWMLx output falling edge.
1	Leading-edge blanking is enabled on PWMLx output falling edge.

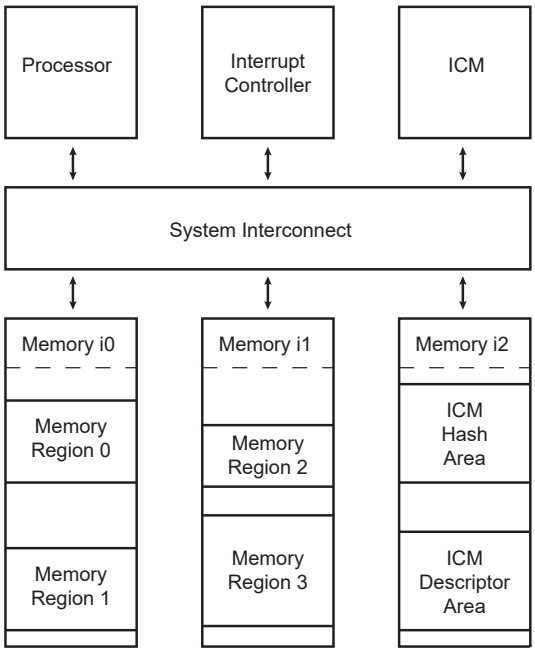
Bits 6:0 – LEBDELAY[6:0] Leading-Edge Blanking Delay for TRGINx
Leading-edge blanking duration for external trigger x input. The delay is calculated according to the following formula:
LEBDELAY = ($f_{peripheral\ clock}$ × Delay) + 1

42. Integrity Check Monitor (ICM)

42.1 Description

The Integrity Check Monitor (ICM) is a DMA controller that performs hash calculation over multiple memory regions through the use of transfer descriptors located in memory (ICM Descriptor Area). The Hash function is based on the Secure Hash Algorithm (SHA). The ICM integrates two modes of operation. The first one is used to hash a list of memory regions and save the digests to memory (ICM Hash Area). The second mode is an active monitoring of the memory. In that mode, the hash function is evaluated and compared to the digest located at a predefined memory address (ICM Hash Area). If a mismatch occurs, an interrupt is raised. See the figure below for an example of four-region monitoring. Hash and Descriptor areas are located in Memory instance i2, and the four regions are split in memory instances i0 and i1.

Figure 42-1. Four-region Monitoring Example



The ICM SHA engine is compliant with the American FIPS (Federal Information Processing Standard) Publication 180-2 specification.

The following terms are concise definitions of the ICM concepts used throughout this document:

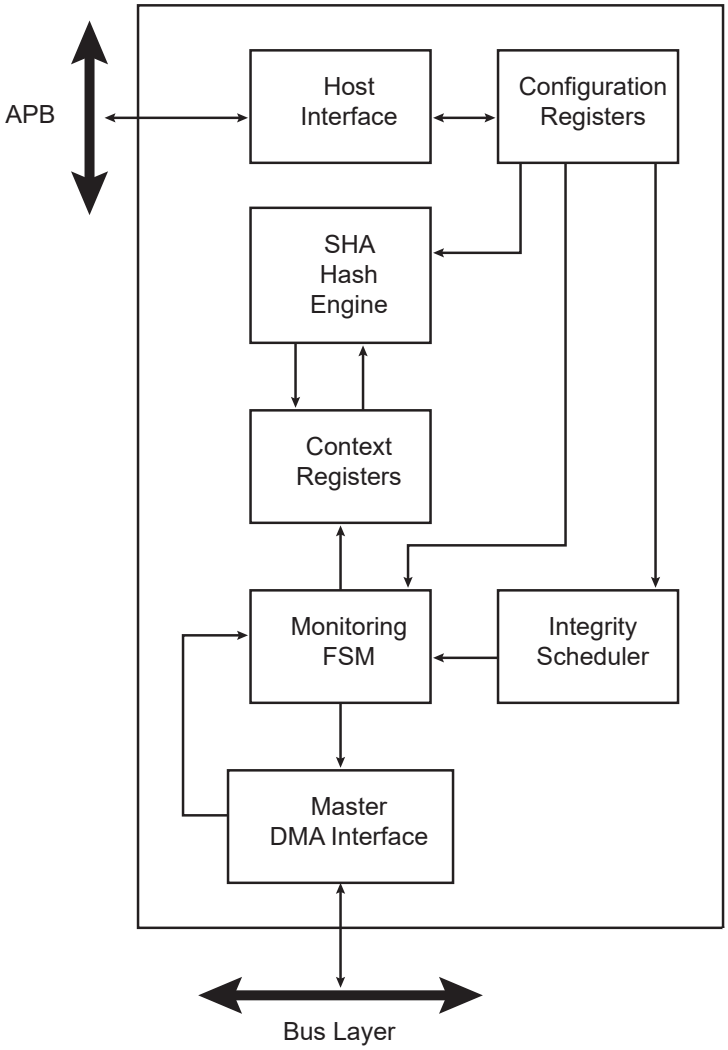
- Region — A partition of instruction or data memory space.
- Region Descriptor — A data structure stored in memory, defining region attributes.
- Region Attributes — Region start address, region size, region SHA engine processing mode, Write Back or Compare function mode.
- Context Registers — A set of ICM non-memory-mapped, internal registers which are automatically loaded, containing the attributes of the region being processed.
- Main List — A list of region descriptors. Each element associates the start address of a region with a set of attributes.
- Secondary List — A linked list defined on a per region basis that describes the memory layout of the region (when the region is non-contiguous).
- Hash Area — predefined memory space where the region hash results (digest) are stored.

42.2 Embedded Characteristics

- DMA AHB Master Interface
- Supports Monitoring of up to 4 Non-Contiguous Memory Regions
- Supports Block Gathering Using Linked Lists
- Supports Secure Hash Algorithm (SHA1, SHA224, SHA256)
- Compliant with FIPS Publication 180-2
- Configurable Processing Period:
  - When SHA1 algorithm is processed, the runtime period is either 85 or 209 clock cycles.
  - When SHA256 or SHA224 algorithm is processed, the runtime period is either 72 or 194 clock cycles.
- Programmable Bus Burden
- Register Write Protection

42.3 Block Diagram

Figure 42-2. Integrity Check Monitor Block Diagram



42.4 Product Dependencies

42.4.1 Power Management

The peripheral clock is not continuously provided to the ICM. The programmer must first enable the ICM clock in the Power Management Controller (PMC) before using the ICM.

42.4.2 Interrupt Sources

The ICM interface has an interrupt line connected to the Interrupt Controller.

Handling the ICM interrupt requires programming the interrupt controller before configuring the ICM.

42.5 Functional Description

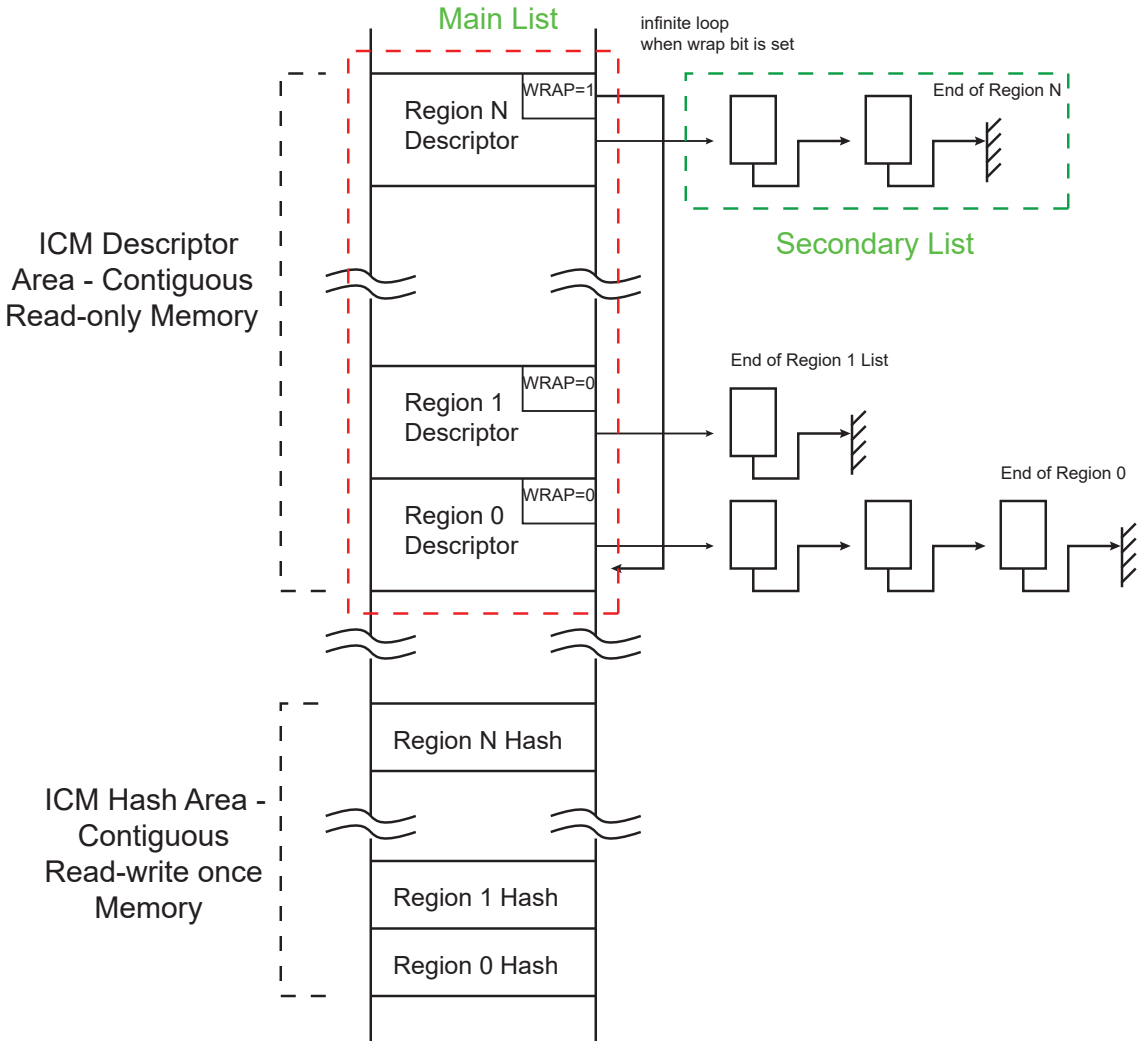
42.5.1 Overview

The Integrity Check Monitor (ICM) is a DMA controller that performs SHA-based memory hashing over memory regions. As shown in figure Integrity Check Monitor Block Diagram, it integrates a DMA interface, a Monitoring Finite State Machine (FSM), an integrity scheduler, a set of context registers, a SHA engine, an interface for configuration and status registers.

The ICM integrates a Secure Hash Algorithm engine (SHA). This engine requires a message padded according to FIPS180-2 specification when used as a SHA calculation unit only. Otherwise, if the ICM is used as integrated check for memory content, the padding is not mandatory. The SHA module produces an N-bit message digest each time a block is read and a processing period ends. N is 160 for SHA1, 224 for SHA224, 256 for SHA256.

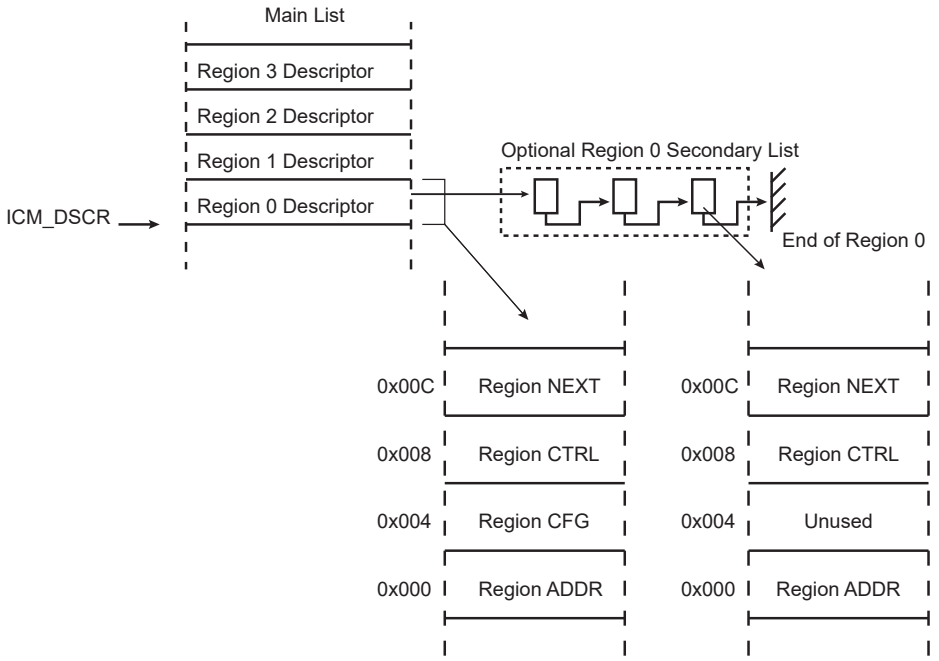
When the ICM module is enabled, it sequentially retrieves a circular list of region descriptors from the memory (Main List described in figure ICM Region Descriptor and Hash Areas). Up to four regions may be monitored. Each region descriptor is composed of four words indicating the layout of the memory region (see figure Region Descriptor). It also contains the hashing engine configuration on a per-region basis. As soon as the descriptor is loaded from the memory and context registers are updated with the data structure, the hashing operation starts. A programmable number of blocks (see TRSIZE field of the ICM_RCTRL structure member) is transferred from the memory to the SHA engine. When the desired number of blocks have been transferred, the digest is either moved to memory (Write Back function) or compared with a digest reference located in the system memory (Compare function). If a digest mismatch occurs, an interrupt is triggered if unmasked. The ICM module passes through the region descriptor list until the end of the list marked by an end of list marker (WRAP or EOM bit in ICM_RCFG structure member set to one). To continuously monitor the list of regions, the WRAP bit must be set to one in the last data structure and EOM must be cleared.

Figure 42-3. ICM Region Descriptor and Hash Areas



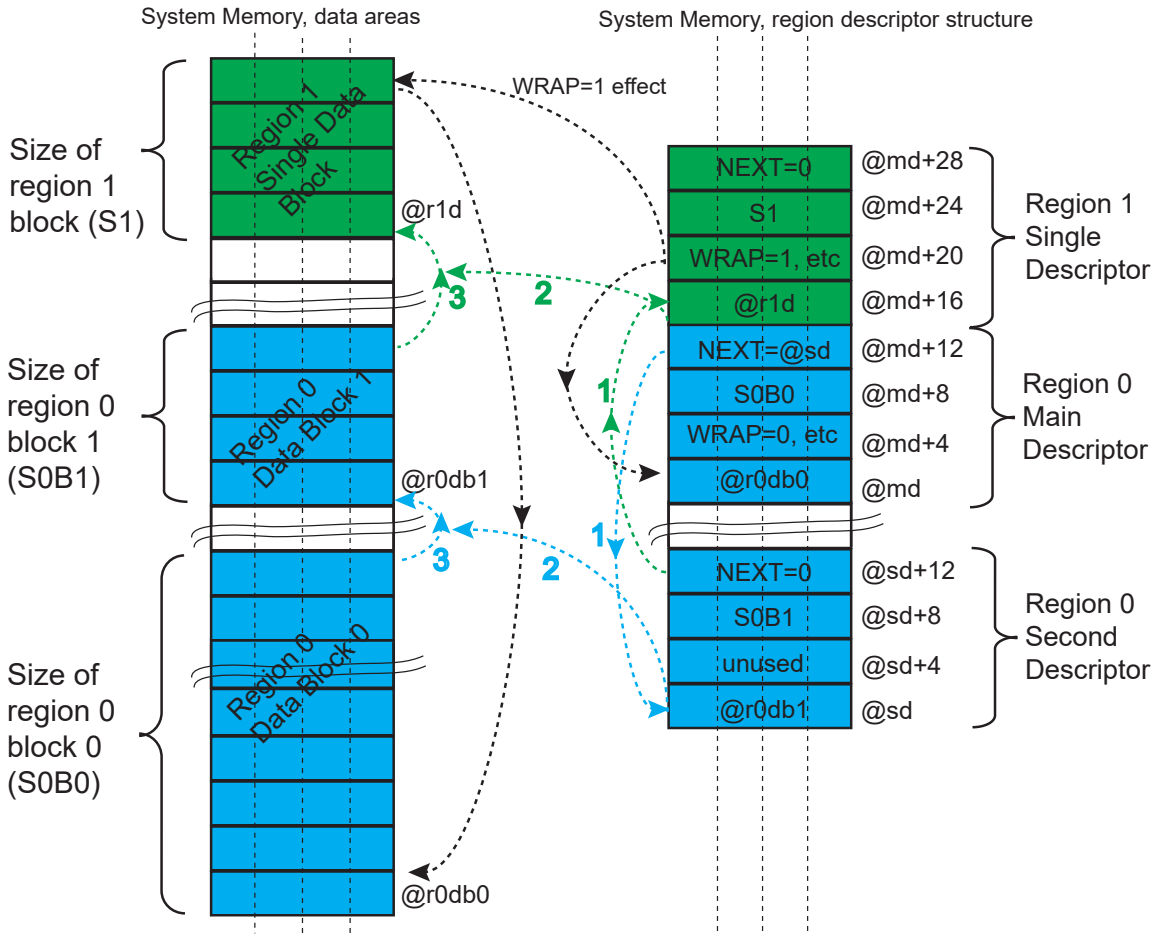
Each region descriptor supports gathering of data through the use of the Secondary List. Unlike the Main List, the Secondary List cannot modify the configuration attributes of the region. When the end of the Secondary List has been encountered, the ICM returns to the Main List. Memory integrity monitoring can be considered as a background service and the mandatory bandwidth shall be very limited. In order to limit the ICM memory bandwidth, use ICM_CFG.BBC to control the ICM memory load.

Figure 42-4. Region Descriptor



The figure below shows an example of the mandatory ICM settings required to monitor three memory data blocks of the system memory (defined as two regions) with one region being not contiguous (two separate areas) and one contiguous memory area. For each region, the SHA algorithm may be independently selected (different for each region). The wrap allows continuous monitoring.

Figure 42-5. Example: Monitoring of 3 Memory Data Blocks (Defined as 2 Regions)



42.5.2 ICM Region Descriptor Structure

The ICM Region Descriptor Area is a contiguous area of system memory that the controller and the processor can access. When the ICM is activated, the controller performs a descriptor fetch operation at *(ICM_DSCR) address. If the Main List contains more than one descriptor (i.e., more than one region is to be monitored), the fetch address is *(ICM_DSCR) + (RID<<4) where RID is the region identifier.

Table 42-1. Region Descriptor Structure (Main List)

Offset	Structure Member	Name
ICM_DSCR+0x000+RID*(0x10)	ICM Region Start Address	ICM_RADDR
ICM_DSCR+0x004+RID*(0x10)	ICM Region Configuration	ICM_RCFG
ICM_DSCR+0x008+RID*(0x10)	ICM Region Control	ICM_RCTRL
ICM_DSCR+0x00C+RID*(0x10)	ICM Region Next Address	ICM_RNEXT

42.5.2.1 ICM Region Start Address Structure Member

Name: ICM_RADDR
Property: Read/Write

Register offset is calculated as ICM_DSCR+0x000+RID*(0x10).

Bit	31	30	29	28	27	26	25	24
	RADDR[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bit	23	22	21	20	19	18	17	16
	RADDR[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bit	15	14	13	12	11	10	9	8
	RADDR[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bit	7	6	5	4	3	2	1	0
	RADDR[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bits 31:0 – RADDR[31:0] Region Start Address
This field indicates the first byte address of the region.

42.5.2.2 ICM Region Configuration Structure Member

Name: ICM_RCFG
Property: Read/Write

Register offset is calculated as ICM_DSCR+0x004+RID*(0x10).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
		ALGO[2:0]				PROCDLY	SUIEN	ECIEN
Access		R/W	R/W	R/W		R/W	R/W	R/W
Reset								

Bit	7	6	5	4	3	2	1	0
	WCIEN	BEIEN	DMIEN	RHIEN		EOM	WRAP	CDWBN
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset								

Bits 14:12 – ALGO[2:0] SHA Algorithm
Values which are not listed in the table must be considered as “reserved”.

Value	Name	Description
0	SHA1	SHA1 algorithm processed
1	SHA256	SHA256 algorithm processed
4	SHA224	SHA224 algorithm processed

Bit 10 – PROCDLY Processing Delay
When SHA1 algorithm is processed, the runtime period is either 85 or 209 clock cycles.
When SHA256 or SHA224 algorithm is processed, the runtime period is either 72 or 194 clock cycles.

Value	Name	Description
0	SHORTEST	SHA processing runtime is the shortest one.
1	LONGEST	SHA processing runtime is the longest one.

Bit 9 – SUIEN Monitoring Status Updated Condition Interrupt (Default Enabled)

Value	Name	Description
0		The ICM_ISR.RSU[i] flag is set when the corresponding descriptor is loaded from memory to ICM.
1		The ICM_ISR.RSU[i] flag remains cleared even if the setting condition is met.

Bit 8 – ECIEN End Bit Condition Interrupt (Default Enabled)

Value	Name	Description
0		The ICM_ISR.REC[i] flag is set when the descriptor with the EOM bit set is processed.
1		The ICM_ISR.REC[i] flag remains cleared even if the setting condition is met.

Bit 7 – WCIEN Wrap Condition Interrupt Disable (Default Enabled)

Value	Name	Description
0		The ICM_ISR.RWC[i] flag is set when the WRAP bit is set in a descriptor of the main list.
1		ICM_ISR.RWC[i] flag remains cleared even if the setting condition is met.

Bit 6 – BEIEN Bus Error Interrupt Disable (Default Enabled)

Value	Name	Description
0		The flag is set when an error is reported on the system bus by the bus matrix.
1		The flag remains cleared even if the setting condition is met.

Bit 5 – DMIEN Digest Mismatch Interrupt Disable (Default Enabled)

Value	Name	Description
0		The ICM_ISR.RBE[i] flag is set when the hash value just calculated from the processed region differs from expected hash value.
1		The ICM_ISR.RBE[i] flag remains cleared even if the setting condition is met.

Bit 4 – RHIEN Region Hash Completed Interrupt Disable (Default Enabled)

Value	Name	Description
0		The ICM_ISR.RHC[i] flag is set when the field NEXT = 0 in a descriptor of the main or second list.
1		The ICM_ISR.RHC[i] flag remains cleared even if the setting condition is met.

Bit 2 – EOM End Of Monitoring

Value	Name	Description
0		The current descriptor does not terminate the monitoring.
1		The current descriptor terminates the Main List. WRAP value has no effect.

Bit 1 – WRAP Wrap Command

Value	Name	Description
0		The next region descriptor address loaded is the current region identifier descriptor address incremented by 0x10.
1		The next region descriptor address loaded is ICM_DSCR.

Bit 0 – CDWBN Compare Digest or Write Back Digest

Value	Name	Description
0		The digest is written to the Hash area.
1		The digest value is compared to the digest stored in the Hash area.

42.5.2.3 ICM Region Control Structure Member

Name: ICM_RCTRL
Property: Read/Write

Register offset is calculated as ICM_DSCR+0x008+RID*(0x10).

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	TRSIZE[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bit	7	6	5	4	3	2	1	0
	TRSIZE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bits 15:0 – TRSIZE[15:0] Transfer Size for the Current Chunk of Data
ICM performs a transfer of (TRSIZE + 1) blocks of 512 bits.

42.5.2.4 ICM Region Next Address Structure Member

Name: ICM_RNEXT
Property: Read/Write

Register offset is calculated as ICM_DSCR+0x00C+RID*(0x10).

Bit	31	30	29	28	27	26	25	24
	NEXT[29:22]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bit	23	22	21	20	19	18	17	16
	NEXT[21:14]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bit	15	14	13	12	11	10	9	8
	NEXT[13:6]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset								

Bit	7	6	5	4	3	2	1	0
	NEXT[5:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W		
Reset								

Bits 31:2 – NEXT[29:0] Region Transfer Descriptor Next Address
When configured to 0, this field indicates that the current descriptor is the last descriptor of the Secondary List, otherwise it points at a new descriptor of the Secondary List.

42.5.3 ICM Hash Area

The ICM Hash Area is a contiguous area of system memory that the controller and the processor can access. The physical location is configured in the ICM hash area start address register. This address is a multiple of 128 bytes. If the CDWBN bit of the context register is cleared (i.e., Write Back activated), the ICM performs a digest write operation at the following starting location: *(ICM_HASH) + (RID<<5), where RID is the current region context identifier. If the CDWBN bit of the context register is set (i.e., Digest Comparison activated), the ICM performs a digest read operation at the same address.

42.5.3.1 Message Digest Example

Considering the following 512-bit message (example given in FIPS 180-2):

“616263800018”

The message is written to memory in a Little Endian (LE) system architecture.

Table 42-2. 512 bits Message Memory Mapping

Memory Address	Address Offset / Byte Lane			
	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000	80	63	62	61
0x004–0x038	00	00	00	00
0x03C	18	00	00	00

The digest is stored at the memory location pointed at by the ICM_HASH pointer with a Region Offset.

Table 42-3. LE Resulting SHA-160 Message Digest Memory Mapping

Memory Address	Address Offset / Byte Lane			
	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000	36	3e	99	a9
0x004	6a	81	06	47
0x008	71	25	3e	ba
0x00C	6c	c2	50	78
0x010	9d	d8	d0	9c

Table 42-4. Resulting SHA-224 Message Digest Memory Mapping

Memory Address	Address Offset / Byte Lane			
	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000	22	7d	09	23
0x004	22	d8	05	34
0x008	77	a4	42	86
0x00C	b3	55	a2	bd
0x010	e4	bc	ad	2a
0x014	f7	b3	a0	bd
0x018	a7	9d	6c	e3

Table 42-5. Resulting SHA-256 Message Digest Memory Mapping

Memory Address	Address Offset / Byte Lane			
	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000	bf	16	78	ba
0x004	ea	cf	01	8f
0x008	de	40	41	41
0x00C	23	22	ae	5d
0x010	a3	61	03	b0
0x014	9c	7a	17	96
0x018	61	ff	10	b4
0x01C	ad	15	00	f2

Considering the following 1024-bit message (example given in FIPS 180-2):

“61626380000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000018”

The message is written to memory in a Little Endian (LE) system architecture.

Table 42-6. 1024 bits Message Memory Mapping

Memory Address	Address Offset / Byte Lane			
	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000	80	63	62	61
0x004–0x078	00	00	00	00
0x07C	18	00	00	00

42.5.4 Using ICM as SHA Engine

The ICM can be configured to only calculate a SHA1, SHA224, SHA256 digest value.

42.5.4.1 Settings for Simple SHA Calculation

The start address of the system memory containing the data to hash must be configured in the transfer descriptor of the DMA embedded in the ICM.

The transfer descriptor is a system memory area integer multiple of 4 x 32-bit words and the start address of the descriptor must be configured in ICM_DSCR (the start address must be aligned on 64-bytes; six LSB must be cleared). If the data to hash is already padded according to SHA standards, only a single descriptor is required, and ICM_RCFG.EOM must be written to 1. If the data to hash does not contain a padding area, it is possible to define the padding area in another system memory location, the ICM can be configured to automatically jump from a memory area to another one by configuring the descriptor register ICM_RNEXT with a value that differs from 0. Configuring ICM_RNEXT.NEXT with the start address of the padding area forces the ICM to concatenate both areas, thus providing the SHA result from the start address of the hash area configured in ICM_HASH.

Whether the system memory is configured as a single or multiple data block area, ICM_RCFG.CDWBN and ICM_RCFG.WRAP must be cleared. The bits WBDIS, EOMDIS, SLBDIS must be cleared in ICM_CFG.

ICM_RCTRL.RHIEN and ICM_RCTRL.ECIEN must be written to 1. The flag RHC[i], i being the region index, is set (if RHIEN is set) when the hash result is available at address defined in ICM_HASH. The flag REC[i], i being the region index, is set (if ECIEN is set) when the hash result is available at the address defined in ICM_HASH.

An interrupt is generated if the bit RHC[i] is written to 1 in the ICM_IER (if RHC[i] is set in ICM_RCTRL of region i) or if the bit REC[i] is written to 1 in the ICM_IER (if REC[i] is set in ICM_RCTRL of region i).

42.5.4.2 Processing Period

The SHA engine processing period can be configured.

The short processing period allows to allocate bandwidth to the SHA module whereas the long processing period allocates more bandwidth on the system bus to other applications.

In SHA mode, the shortest processing period is 85 clock cycles + 2 clock cycles for start command synchronization. The longest period is 209 clock cycles + 2 clock cycles.

In SHA256 and SHA224 modes, the shortest processing period is 72 clock cycles + 2 clock cycles for start command synchronization. The longest period is 194 clock cycles + 2 clock cycles.

42.5.5 ICM Automatic Monitoring Mode

ICM_CFG.ASCD is used to activate the ICM Automatic Monitoring mode. When ICM_CFG.ASCD is set and bits CDWBN and EOM in ICM.RCFG equal 0, the ICM performs the following actions:

1. The ICM passes through the Main List once to calculate the message digest of the monitored area.
2. When WRAP = 1 in ICM_RCFG, the ICM begins monitoring. CDWBN in ICM_RCFG is now automatically set and EOM is cleared. These bits have no effect during the monitoring period that ends when EOM is set.

42.5.6 Programming the ICM

Table 42-7. Region Attributes

Transfer Type		Main List	ICM_RCFG			ICM_RNEXT	Comments
			CDWBN	WRAP	EOM	NEXT	
Single Region	Contiguous list of blocks Digest written to memory Monitoring disabled	1 item	0	0	1	0	The Main List contains only one descriptor. The Secondary List is empty for that descriptor. The digest is computed and saved to memory.
	Non-contiguous list of blocks Digest written to memory Monitoring disabled	1 item	0	0	1	Secondary List address of the current region identifier	The Main List contains only one descriptor. The Secondary List describes the layout of the non-contiguous region.
	Contiguous list of blocks Digest comparison enabled Monitoring enabled	1 item	1	1	0	0	When the hash computation is terminated, the digest is compared with the one saved in memory.
Multiple Regions	Contiguous list of blocks Digest written to memory Monitoring disabled	More than one item	0	0	1 for the last, 0 otherwise	0	ICM passes through the list once.
	Contiguous list of blocks Digest comparison is enabled Monitoring is enabled	More than one item	1	1 for the last, 0 otherwise	0	0	ICM performs active monitoring of the regions. If a mismatch occurs, an interrupt is raised.
	Non-contiguous list of blocks Digest is written to memory Monitoring is disabled	More than one item	0	0	1	Secondary List address	ICM performs hashing and saves digests to the Hash area.
	Non-contiguous list of blocks Digest comparison is enabled Monitoring is enabled	More than one item	1	1	0	Secondary List address	ICM performs data gathering on a per region basis.

42.5.7 Security Features

When an undefined register access occurs, the URAD bit in the Interrupt Status Register (ICM_ISR) is set if unmasked. Its source is then reported in the Undefined Access Status Register (ICM_UASR). Only the first undefined register access is available through the ICM_UASR.URAT field.

Several kinds of unspecified register accesses can occur:

- Unspecified structure member set to one detected when the descriptor is loaded
- Configuration register (ICM_CFG) modified during active monitoring
- Descriptor register (ICM_DSCR) modified during active monitoring
- Hash register (ICM_HASH) modified during active monitoring
- Write-only register read access

The URAD bit and the URAT field can only be reset by writing a 1 to the ICM_CTRL.SWRST bit.

42.5.8 ICM Register Write Protection

To prevent any single software error from corrupting ICM behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable), WPITEN (Write Protection Interrupt Enable), and/or WPCREN (Write Protection Control Enable) bits in the ICM Write Protection Mode Register (ICM_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the ICM Write Protection Status Register (ICM_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading ICM_WPSR.

The following register(s) can be write-protected when WPEN is set:

- ICM Configuration Register
- ICM Descriptor Area Start Address Register
- ICM Hash Area Start Address Register
- ICM User Initial Hash Value Register

The following registers can be write-protected when WPITEN is set:

- ICM Interrupt Enable Register
- ICM Interrupt Disable Register

The following register can be write-protected when WPCREN is set:

- ICM Control Register

42.6 Register Summary

Offset	Name	Bit Pos.								
0x00	ICM_CFG	7:0	BBC[3:0]					SLBDIS	EOMDIS	WBDIS
		15:8	UALGO[2:0]			UIHASH			DUALBUFF	ASCD
		23:16								
		31:24								
0x04	ICM_CTRL	7:0	REHASH[3:0]					SWRST	DISABLE	ENABLE
		15:8	RMEN[3:0]				RMDIS[3:0]			
		23:16								
		31:24								
0x08	ICM_SR	7:0								ENABLE
		15:8	RMDIS[3:0]				RAWRMDIS[3:0]			
		23:16								
		31:24								
0x0C ... 0x0F	Reserved									
0x10	ICM_IER	7:0	RDM[3:0]				RHC[3:0]			
		15:8	RWC[3:0]				RBE[3:0]			
		23:16	RSU[3:0]				REC[3:0]			
		31:24								URAD
0x14	ICM_IDR	7:0	RDM[3:0]				RHC[3:0]			
		15:8	RWC[3:0]				RBE[3:0]			
		23:16	RSU[3:0]				REC[3:0]			
		31:24								URAD
0x18	ICM_IMR	7:0	RDM[3:0]				RHC[3:0]			
		15:8	RWC[3:0]				RBE[3:0]			
		23:16	RSU[3:0]				REC[3:0]			
		31:24								URAD
0x1C	ICM_ISR	7:0	RDM[3:0]				RHC[3:0]			
		15:8	RWC[3:0]				RBE[3:0]			
		23:16	RSU[3:0]				REC[3:0]			
		31:24								URAD
0x20	ICM_UASR	7:0						URAT[2:0]		
		15:8								
		23:16								
		31:24								
0x24 ... 0x2F	Reserved									
0x30	ICM_DSCR	7:0	DASA[1:0]							
		15:8	DASA[9:2]							
		23:16	DASA[17:10]							
		31:24	DASA[25:18]							
0x34	ICM_HASH	7:0	HASA[0]							
		15:8	HASA[8:1]							
		23:16	HASA[16:9]							
		31:24	HASA[24:17]							
0x38	ICM_UIHVAL0	7:0	VAL[7:0]							
		15:8	VAL[15:8]							
		23:16	VAL[23:16]							
		31:24	VAL[31:24]							
0x3C	ICM_UIHVAL1	7:0	VAL[7:0]							
		15:8	VAL[15:8]							
		23:16	VAL[23:16]							
		31:24	VAL[31:24]							

.....continued

Offset	Name	Bit Pos.								
0x40	ICM_UIHVAL2	7:0	VAL[7:0]							
		15:8	VAL[15:8]							
		23:16	VAL[23:16]							
		31:24	VAL[31:24]							
0x44	ICM_UIHVAL3	7:0	VAL[7:0]							
		15:8	VAL[15:8]							
		23:16	VAL[23:16]							
		31:24	VAL[31:24]							
0x48	ICM_UIHVAL4	7:0	VAL[7:0]							
		15:8	VAL[15:8]							
		23:16	VAL[23:16]							
		31:24	VAL[31:24]							
0x4C	ICM_UIHVAL5	7:0	VAL[7:0]							
		15:8	VAL[15:8]							
		23:16	VAL[23:16]							
		31:24	VAL[31:24]							
0x50	ICM_UIHVAL6	7:0	VAL[7:0]							
		15:8	VAL[15:8]							
		23:16	VAL[23:16]							
		31:24	VAL[31:24]							
0x54	ICM_UIHVAL7	7:0	VAL[7:0]							
		15:8	VAL[15:8]							
		23:16	VAL[23:16]							
		31:24	VAL[31:24]							
0x58 ... 0xE3	Reserved									
0xE4	ICM_WPMR	7:0						WPCREN	WPITEN	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0xE8	ICM_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16								
		31:24								

42.6.1 ICM Configuration Register

Name: ICM_CFG
Offset: 0x00
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the ICM Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	UALGO[2:0]			UIHASH			DUALBUFF	ASCD
Access	R/W	R/W	R/W	R/W			R/W	R/W
Reset	0	0	0	0			0	0

Bit	7	6	5	4	3	2	1	0
	BBC[3:0]					SLBDIS	EOMDIS	WBDIS
Access	R/W	R/W	R/W	R/W		R/W	R/W	R/W
Reset	0	0	0	0		0	0	0

Bits 15:13 – UALGO[2:0] User SHA Algorithm

Value	Name	Description
0	SHA1	SHA1 algorithm processed
1	SHA256	SHA256 algorithm processed
4	SHA224	SHA224 algorithm processed

Bit 12 – UIHASH User Initial Hash Value

Value	Description
0	The secure hash standard provides the initial hash value.
1	The initial hash value is programmable. Field UALGO provides the SHA algorithm. The ALGO field of the ICM_RCFG structure member has no effect.

Bit 9 – DUALBUFF Dual Input Buffer

Value	Description
0	Dual Input Buffer mode is disabled.
1	Dual Input Buffer mode is enabled (better performances, higher bandwidth required on system bus).

Bit 8 – ASCD Automatic Switch To Compare Digest

Value	Description
0	Automatic monitoring mode is disabled.
1	The ICM passes through the Main List once to calculate the message digest of the monitored area. When WRAP = 1 in ICM_RCFG, the ICM begins monitoring.

Bits 7:4 – BBC[3:0] Bus Burden Control
This field is used to control the burden of the ICM system bus. The number of system clock cycles between the end of the current processing and the next block transfer is set to 2^{BBC}. Up to 32,768 cycles can be inserted.

Bit 2 – SLBDIS Secondary List Branching Disable

Value	Description
0	Branching to the Secondary List is permitted.
1	Branching to the Secondary List is forbidden. The NEXT field of the ICM_RNEXT structure member has no effect and is always considered as zero.

Bit 1 – EOMDIS End of Monitoring Disable

Value	Description
0	End of Monitoring is permitted.
1	End of Monitoring is forbidden. The EOM bit of the ICM_RCFG structure member has no effect.

Bit 0 – WBDIS Write Back Disable
When ASCD is set, WBDIS has no effect.

Value	Description
0	Write Back operations are permitted.
1	Write Back operations are forbidden. Context register CDWBN bit is internally set to one and cannot be modified by a linked list element. ICM_RCFG.CDWBN has no effect.

42.6.2 ICM Control Register

Name: ICM_CTRL
Offset: 0x04
Reset: –
Property: Write-only

This register can only be written if the WPCREN bit is cleared in the ICM Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RMEN[3:0]				RMDIS[3:0]			
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	–	0	0	0	–

Bit	7	6	5	4	3	2	1	0
	REHASH[3:0]					SWRST	DISABLE	ENABLE
Access	W	W	W	W		W	W	W
Reset	0	0	0	–		–	–	–

Bits 15:12 – RMEN[3:0] Region Monitoring Enable
Monitoring is activated by default.

Value	Description
0	No effect
1	When bit RMEN[i] is set to one, the monitoring of region with identifier i is activated.

Bits 11:8 – RMDIS[3:0] Region Monitoring Disable

Value	Description
0	No effect
1	When bit RMDIS[i] is set to one, the monitoring of region with identifier i is disabled.

Bits 7:4 – REHASH[3:0] Recompute Internal Hash

Value	Description
0	No effect
1	When REHASH[i] is set to one, Region i digest is re-computed. This bit is only available when region monitoring is disabled.

Bit 2 – SWRST Software Reset

Value	Description
0	No effect
1	Resets the ICM.

Bit 1 – DISABLE ICM Disable Register

Value	Description
0	No effect
1	The ICM is disabled. If a region is active, this region is terminated.

Bit 0 – ENABLE ICM Enable

Value	Description
0	No effect
1	When set to one, the ICM is activated.

42.6.3 ICM Status Register

Name: ICM_SR
Offset: 0x08
Reset: –
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	RMDIS[3:0]				RAWRMDIS[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	–	0	0	0	–

Bit	7	6	5	4	3	2	1	0
								ENABLE
Access								R
Reset								–

Bits 15:12 – RMDIS[3:0] Region Monitoring Disabled Status

Value	Description
0	Region i is being monitored (occurs after integrity check value has been calculated and written to Hash area).
1	Region i monitoring is not being monitored.

Bits 11:8 – RAWRMDIS[3:0] Region Monitoring Disabled Raw Status

Value	Description
0	Region i monitoring has been activated by writing a 1 in RMEN[i] of ICM_CTRL.
1	Region i monitoring has been deactivated by writing a 1 in RMDIS[i] of ICM_CTRL.

Bit 0 – ENABLE ICM Enable Register

Value	Description
0	ICM is disabled.
1	ICM is activated.

42.6.4 ICM Interrupt Enable Register

Name: ICM_IER
Offset: 0x10
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the ICM Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
								URAD
Access								W
Reset								–

Bit	23	22	21	20	19	18	17	16
	RSU[3:0]				REC[3:0]			
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	–	0	0	0	–

Bit	15	14	13	12	11	10	9	8
	RWC[3:0]				RBE[3:0]			
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	–	0	0	0	–

Bit	7	6	5	4	3	2	1	0
	RDM[3:0]				RHC[3:0]			
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	–	0	0	0	–

Bit 24 – URAD Undefined Register Access Detection Interrupt Enable

Value	Description
0	No effect.
1	The Undefined Register Access interrupt is enabled.

Bits 23:20 – RSU[3:0] Region Status Updated Interrupt Disable

Value	Description
0	No effect.
1	When RSU[i] is set to one, the region i Status Updated interrupt is enabled.

Bits 19:16 – REC[3:0] Region End bit Condition Detected Interrupt Enable

Value	Description
0	No effect.
1	When REC[i] is set to one, the region i End bit Condition interrupt is enabled.

Bits 15:12 – RWC[3:0] Region Wrap Condition detected Interrupt Enable

Value	Description
0	No effect.
1	When RWC[i] is set to one, the Region i Wrap Condition interrupt is enabled.

Bits 11:8 – RBE[3:0] Region Bus Error Interrupt Enable

Value	Description
0	No effect.
1	When RBE[i] is set to one, the Region i Bus Error interrupt is enabled.

Bits 7:4 – RDM[3:0] Region Digest Mismatch Interrupt Enable

Value	Description
0	No effect.
1	When RDM[i] is set to one, the Region i Digest Mismatch interrupt is enabled.

Bits 3:0 – RHC[3:0] Region Hash Completed Interrupt Enable

Value	Description
0	No effect.
1	When RHC[i] is set to one, the Region i Hash Completed interrupt is enabled.

42.6.5 ICM Interrupt Disable Register

Name: ICM_IDR
Offset: 0x14
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the ICM Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
								URAD
Access								W
Reset								–

Bit	23	22	21	20	19	18	17	16
	RSU[3:0]				REC[3:0]			
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	–	0	0	0	–

Bit	15	14	13	12	11	10	9	8
	RWC[3:0]				RBE[3:0]			
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	–	0	0	0	–

Bit	7	6	5	4	3	2	1	0
	RDM[3:0]				RHC[3:0]			
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	–	0	0	0	–

Bit 24 – URAD Undefined Register Access Detection Interrupt Disable

Value	Description
0	No effect.
1	Undefined Register Access Detection interrupt is disabled.

Bits 23:20 – RSU[3:0] Region Status Updated Interrupt Disable

Value	Description
0	No effect.
1	When RSU[i] is set to one, the region i Status Updated interrupt is disabled.

Bits 19:16 – REC[3:0] Region End bit Condition detected Interrupt Disable

Value	Description
0	No effect.
1	When REC[i] is set to one, the region i End bit Condition interrupt is disabled.

Bits 15:12 – RWC[3:0] Region Wrap Condition Detected Interrupt Disable

Value	Description
0	No effect.
1	When RWC[i] is set to one, the Region i Wrap Condition interrupt is disabled.

Bits 11:8 – RBE[3:0] Region Bus Error Interrupt Disable

Value	Description
0	No effect.
1	When RBE[i] is set to one, the Region i Bus Error interrupt is disabled.

Bits 7:4 – RDM[3:0] Region Digest Mismatch Interrupt Disable

Value	Description
0	No effect.
1	When RDM[i] is set to one, the Region i Digest Mismatch interrupt is disabled.

Bits 3:0 – RHC[3:0] Region Hash Completed Interrupt Disable

Value	Description
0	No effect.
1	When RHC[i] is set to one, the Region i Hash Completed interrupt is disabled.

42.6.6 ICM Interrupt Mask Register

Name: ICM_IMR
Offset: 0x18
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
								URAD
Access								R
Reset								0

Bit	23	22	21	20	19	18	17	16
	RSU[3:0]				REC[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RWC[3:0]				RBE[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RDM[3:0]				RHC[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 24 – URAD Undefined Register Access Detection Interrupt Mask

Value	Description
0	Interrupt is disabled
1	Interrupt is enabled.

Bits 23:20 – RSU[3:0] Region Status Updated Interrupt Mask

Value	Description
0	When RSU[i] is set to zero, the interrupt is disabled for region i.
1	When RSU[i] is set to one, the interrupt is enabled for region i.

Bits 19:16 – REC[3:0] Region End Bit Condition Detected Interrupt Mask

Value	Description
0	When REC[i] is set to zero, the interrupt is disabled for region i.
1	When REC[i] is set to one, the interrupt is enabled for region i.

Bits 15:12 – RWC[3:0] Region Wrap Condition Detected Interrupt Mask

Value	Description
0	When RWC[i] is set to zero, the interrupt is disabled for region i.
1	When RWC[i] is set to one, the interrupt is enabled for region i.

Bits 11:8 – RBE[3:0] Region Bus Error Interrupt Mask

Value	Description
0	When RBE[i] is set to zero, the interrupt is disabled for region i.
1	When RBE[i] is set to one, the interrupt is enabled for region i.

Bits 7:4 – RDM[3:0] Region Digest Mismatch Interrupt Mask

Value	Description
0	When RDM[i] is set to zero, the interrupt is disabled for region i.

Value	Description
1	When RDM[i] is set to one, the interrupt is enabled for region i.

Bits 3:0 – RHC[3:0] Region Hash Completed Interrupt Mask

Value	Description
0	When RHC[i] is set to zero, the interrupt is disabled for region i.
1	When RHC[i] is set to one, the interrupt is enabled for region i.

42.6.7 ICM Interrupt Status Register

Name: ICM_ISR
Offset: 0x1C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
								URAD
Access								R
Reset								0

Bit	23	22	21	20	19	18	17	16
	RSU[3:0]				REC[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	RWC[3:0]				RBE[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	RDM[3:0]				RHC[3:0]			
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit 24 – URAD Undefined Register Access Detection Status
The URAD bit is only reset by the SWRST bit in ICM_CTRL.
The URAT field in ICM_UASR indicates the unspecified access type.

Value	Description
0	No undefined register access has been detected since the last SWRST.
1	At least one undefined register access has been detected since the last SWRST.

Bits 23:20 – RSU[3:0] Region Status Updated Detected
When RSU[i] is set, it indicates that a region status updated condition has been detected.

Bits 19:16 – REC[3:0] Region End Bit Condition Detected
When REC[i] is set, it indicates that an end bit condition has been detected.

Bits 15:12 – RWC[3:0] Region Wrap Condition Detected
When RWC[i] is set, it indicates that a wrap condition has been detected.

Bits 11:8 – RBE[3:0] Region Bus Error
When RBE[i] is set, it indicates that a bus error has been detected while hashing memory region i.

Bits 7:4 – RDM[3:0] Region Digest Mismatch
When RDM[i] is set, it indicates that there is a digest comparison mismatch between the hash value of the region with identifier i and the reference value located in the Hash Area.

Bits 3:0 – RHC[3:0] Region Hash Completed
When RHC[i] is set, it indicates that the ICM has completed the region with identifier i.

42.6.8 ICM Undefined Access Status Register

Name: ICM_UASR
Offset: 0x20
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
						URAT[2:0]		
Access						R	R	R
Reset						0	0	0

Bits 2:0 – URAT[2:0] Undefined Register Access Trace
Only the first Undefined Register Access Trace is available through the URAT field.
The URAT field is only reset by the SWRST bit in the ICM_CTRL register.

Value	Name	Description
0	UNSPEC_STRUCT_MEMBER	Unspecified structure member set to one detected when the descriptor is loaded.
1	ICM_CFG_MODIFIED	ICM_CFG modified during active monitoring.
2	ICM_DSCR_MODIFIED	ICM_DSCR modified during active monitoring.
3	ICM_HASH_MODIFIED	ICM_HASH modified during active monitoring
4	READ_ACCESS	Write-only register read access

42.6.9 ICM Descriptor Area Start Address Register

Name: ICM_DSCR
Offset: 0x30
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the ICM Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	DASA[25:18]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	DASA[17:10]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	DASA[9:2]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	DASA[1:0]							
Access	R/W	R/W						
Reset	0	0						

Bits 31:6 – DASA[25:0] Descriptor Area Start Address
The start address is a multiple of the total size of the data structure (64 bytes).

42.6.10 ICM Hash Area Start Address Register

Name: ICM_HASH
Offset: 0x34
Reset: 0x00000000
Property: Read/Write

This register can only be written if the WPEN bit is cleared in the ICM Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	HASA[24:17]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	HASA[16:9]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	HASA[8:1]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	HASA[0]							
Access	R/W							
Reset	0							

Bits 31:7 – HASA[24:0] Hash Area Start Address
This field points at the Hash memory location. The address must be a multiple of 128 bytes.

42.6.11 ICM User Initial Hash Value Register

Name: ICM_UIHVALx
Offset: 0x38 + x*0x04 [x=0..7]
Reset: –
Property: Write-only

This register can only be written if the WPEN bit is cleared in the ICM Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	VAL[31:24]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	VAL[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	VAL[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	VAL[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 31:0 – VAL[31:0] Initial Hash Value
When ICM_CFG.UIHASH is set, the Initial Hash Value is user-programmable.
To meet the desired standard, use the following example values.
For ICM_UIHVAL0 field:

Example	Comment
0x67452301	SHA1 algorithm
0xC1059ED8	SHA224 algorithm
0x6A09E667	SHA256 algorithm

For ICM_UIHVAL1 field:

Example	Comment
0xEFCDAB89	SHA1 algorithm
0x367CD507	SHA224 algorithm
0xBB67AE85	SHA256 algorithm

For ICM_UIHVAL2 field:

Example	Comment
0x98BADCFE	SHA1 algorithm
0x3070DD17	SHA224 algorithm
0x3C6EF372	SHA256 algorithm

For ICM_UIHVAL3 field:

Example	Comment
0x10325476	SHA1 algorithm
0xF70E5939	SHA224 algorithm
0xA54FF53A	SHA256 algorithm

For ICM_UIHVAL4 field:

Example	Comment
0xC3D2E1F0	SHA1 algorithm
0xFFC00B31	SHA224 algorithm
0x510E527F	SHA256 algorithm

For ICM_UIHVAL5 field:

Example	Comment
0x68581511	SHA224 algorithm
0x9B05688C	SHA256 algorithm

For ICM_UIHVAL6 field:

Example	Comment
0x64F98FA7	SHA224 algorithm
0x1F83D9AB	SHA256 algorithm

For ICM_UIHVAL7 field:

Example	Comment
0xBEFA4FA4	SHA224 algorithm
0x5BE0CD19	SHA256 algorithm

Example of Initial Value for SHA-1 Algorithm

Register Address	Address Offset / Byte Lane			
	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000 ICM_UIHVAL0	01	23	45	67
0x004 ICM_UIHVAL1	89	ab	cd	ef
0x008 ICM_UIHVAL2	fe	dc	ba	98
0x00C ICM_UIHVAL3	76	54	32	10
0x010 ICM_UIHVAL4	f0	e1	d2	c3

42.6.12 ICM Write Protection Mode Register

Name: ICM_WPMR
Offset: 0xE4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
						WPCREN	WPITEN	WPEN
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x49434D	PASSWD	Writing any other value in this field aborts the write operation of the WPEN, WPITEN and WPCREN bits. Always reads as 0

Bit 2 – WPCREN Write Protection Control Enable

Value	Description
0	Disables the write protection on control register if WPKEY corresponds to 0x49434D (“ICM” in ASCII).
1	Enables the write protection on control register if WPKEY corresponds to 0x49434D (“ICM” in ASCII).

Bit 1 – WPITEN Write Protection Interrupt Enable

Value	Description
0	Disables the write protection on interrupt registers if WPKEY corresponds to 0x49434D (“ICM” in ASCII).
1	Enables the write protection on interrupt registers if WPKEY corresponds to 0x49434D (“ICM” in ASCII).

Bit 0 – WPEN Write Protection Enable
See ICM Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection if WPKEY corresponds to 0x49434D (“ICM” in ASCII)
1	Enables the write protection if WPKEY corresponds to 0x49434D (“ICM” in ASCII)

42.6.13 ICM Write Protection Status Register

Name: ICM_WPSR
Offset: 0xE8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 15:8 – WPVSRC[7:0] Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status (Cleared on read)

Value	Description
0	No write protect violation has occurred since the last read of ICM_WPSR.
1	A write protect violation has occurred since the last read of ICM_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

43. True Random Number Generator (TRNG)

43.1 Description

The True Random Number Generator (TRNG) passes the American *NIST Special Publication 800-22 (A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications) and the Diehard Suite of Tests.*

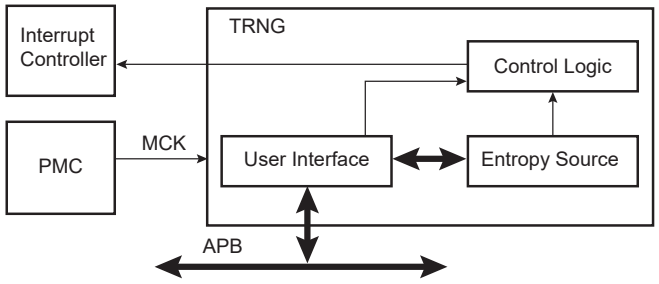
The TRNG may be used as an entropy source for seeding an NIST approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3.

43.2 Embedded Characteristics

- Passes *NIST Special Publication 800-22 Test Suite*
- Passes *Diehard Suite of Tests*
- May be Used as Entropy Source for seeding a NIST-approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3
- Provides a 32-bit Random Number Every 84 Clock Cycles

43.3 Block Diagram

Figure 43-1. TRNG Block Diagram



43.4 Product Dependencies

43.4.1 Power Management

The TRNG interface may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the TRNG user interface clock. The user interface clock is independent from any clock that may be used in the entropy source logic circuitry. The source of entropy can be enabled before enabling the user interface clock.

43.4.2 Interrupt Sources

The TRNG interface has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TRNG.

43.5 Functional Description

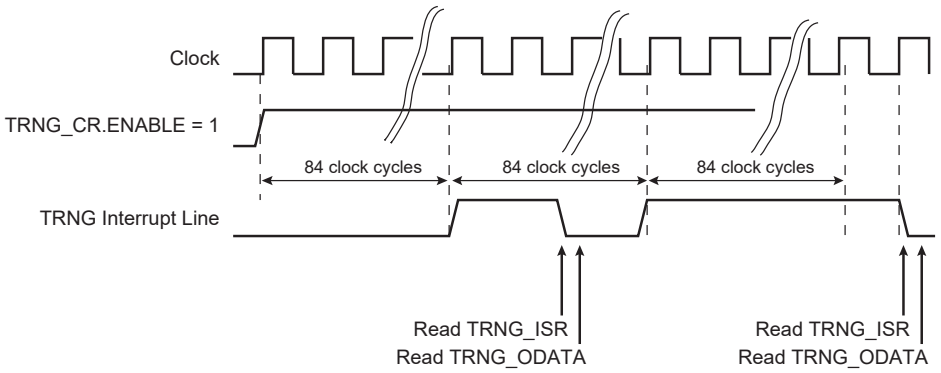
As soon as the TRNG is enabled in the Control register (TRNG_CR), the generator provides one 32-bit random value every 84 clock cycles.

The TRNG interrupt line can be enabled in the Interrupt Enable register (TRNG_IER), and disabled in the Interrupt Disable register (TRNG_IDR). This interrupt is set when a new random value is available and is cleared when the

Status register (TRNG_ISR) is read. The flag TRNG_ISR.DATRDY is set when the random data is ready to be read out on the 32-bit Output Data register (TRNG_ODATA).

The normal mode of operation checks that the flag in TRNG_ISR equals ‘1’ before reading TRNG_ODATA when a 32-bit random value is required by the software application.

Figure 43-2. TRNG Data Generation Sequence



43.6 Register Summary

Offset	Name	Bit Pos.								
0x00	TRNG_CR	7:0								ENABLE
		15:8	WAKEY[7:0]							
		23:16	WAKEY[15:8]							
		31:24	WAKEY[23:16]							
0x04 ... 0x0F	Reserved									
0x10	TRNG_IER	7:0								DATRDY
		15:8								
		23:16								
		31:24								
0x14	TRNG_IDR	7:0								DATRDY
		15:8								
		23:16								
		31:24								
0x18	TRNG_IMR	7:0								DATRDY
		15:8								
		23:16								
		31:24								
0x1C	TRNG_ISR	7:0								DATRDY
		15:8								
		23:16								
		31:24								
0x20 ... 0x4F	Reserved									
0x50	TRNG_ODATA	7:0	ODATA[7:0]							
		15:8	ODATA[15:8]							
		23:16	ODATA[23:16]							
		31:24	ODATA[31:24]							

43.6.1 TRNG Control Register

Name: TRNG_CR
Offset: 0x00
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	WAKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WAKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WAKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bit	7	6	5	4	3	2	1	0
								ENABLE
Access								W
Reset								–

Bits 31:8 – WAKEY[23:0] Register Write Access Key

Value	Name	Description
0x524E47	PASSWD	Writing any other value in this field aborts the write operation.

Bit 0 – ENABLE Enables the TRNG to Provide Random Values

Value	Description
0	Disables the TRNG.
1	Enables the TRNG if 0x524E47 (“RNG” in ASCII) is written in KEY field at the same time.

43.6.2 TRNG Interrupt Enable Register

Name: TRNG_IER
Offset: 0x10
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								DATRDY
Access								W
Reset								–

Bit 0 – DATRDY Data Ready Interrupt Enable

Value	Description
0	No effect.
1	Enables the corresponding interrupt.

43.6.3 TRNG Interrupt Disable Register

Name: TRNG_IDR
Offset: 0x14
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								DATRDY
Access								W
Reset								–

Bit 0 – DATRDY Data Ready Interrupt Disable

Value	Description
0	No effect.
1	Disables the corresponding interrupt.

43.6.4 TRNG Interrupt Mask Register

Name: TRNG_IMR
Offset: 0x18
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								DATRDY
Access								R
Reset								0

Bit 0 – DATRDY Data Ready Interrupt Mask

Value	Description
0	The corresponding interrupt is not enabled.
1	The corresponding interrupt is enabled.

43.6.5 TRNG Interrupt Status Register

Name: TRNG_ISR
Offset: 0x1C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								
Reset								

Bit	7	6	5	4	3	2	1	0
								DATRDY
Access								R
Reset								0

Bit 0 – DATRDY Data Ready (cleared on read)

Value	Description
0	Output data is not valid or TRNG is disabled.
1	New random value is completed since the last read of TRNG_ODATA.

43.6.6 TRNG Output Data Register

Name: TRNG_ODATA
Offset: 0x50
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
	ODATA[31:24]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	ODATA[23:16]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	ODATA[15:8]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	ODATA[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – ODATA[31:0] Output Data
The 32-bit Output Data register contains the 32-bit random data.

44. Secure Hash Algorithm (SHA)

44.1 Description

The Secure Hash Algorithm (SHA) is compliant with the American FIPS (Federal Information Processing Standard) Publication 180-2 specification.

The 512/1024-bit block of message is respectively stored in 16/32 x 32-bit registers, (SHA_IDATARx/ SHA_IODATARx) which are write-only.

As soon as the input data is written, the hash processing may be started. The registers comprising the block of a padded message must be entered consecutively. Then the message digest is ready to be read out on the 5 up to 8/16 x 32-bit output data registers (SHA_IODATARx) or through the DMA channels.

44.2 Embedded Characteristics

- Supports Secure Hash Algorithm (SHA1, SHA224, SHA256, SHA384, SHA512)
- Supports Hash-based Message Authentication Code (HMAC) Algorithm (HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512)
- Compliant with FIPS Publication 180-2
- Supports Automatic Padding of Messages
- Supports Up to 2 Sets of Initial Hash Values Registers (HMAC Acceleration or other)
- Supports Automatic Check of the Hash (HMAC Acceleration or other)
- Tightly Coupled to AES for Protocol Layers Improved Performances
- Configurable Processing Period:
  - 85 Clock Cycles to obtain a fast SHA1 runtime, 88 clock cycles for SHA384, SHA512 or 209 Clock Cycles for Maximizing Bandwidth of Other Applications
  - 72 Clock Cycles to obtain a fast SHA224, SHA256 runtime or 194 Clock Cycles for Maximizing Bandwidth of Other Applications
- Connection to DMA Channel Capabilities Optimizes Data Transfers
- Double Input Buffer Optimizes Runtime
- Register Write Protection

44.3 Product Dependencies

44.3.1 Power Management

The SHA may be clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the SHA clock.

44.3.2 Interrupt Sources

The SHA interface has an interrupt line connected to the Interrupt Controller.

Handling the SHA interrupt requires programming the Interrupt Controller before configuring the SHA.

44.4 Functional Description

The Secure Hash Algorithm (SHA) module requires a padded message according to FIPS180-2 specification. This message can be provided with the padding to the SHA module, or the padding can be automatically computed by the SHA module if the size of the message is provided. The first block of the message must be indicated to the module by a specific command. The SHA module produces an N-bit message digest each time a block is written and processing period ends, where N is 160 for SHA1, 224 for SHA224, 256 for SHA256, 384 for SHA384, 512 for

SHA512. The SHA module is also capable of computing Hash-based Message Authentication Code (HMAC) algorithm.

44.4.1 SHA Algorithm

The SHA can process SHA1, SHA224, SHA256, SHA384, SHA512 by configuring the ALGO field in the SHA Mode register (SHA_MR).

44.4.2 HMAC Algorithm

The HMAC algorithm is as follows:

$HMAC_K(m) = h((K_0 \oplus opad) || h((K_0 \oplus ipad) || m))$

where:

- h = SHA function
- K_0 = The key K after any necessary pre-processing to form a block size key
- m = Message to authenticate
- || = Concatenation operator
- $\oplus$ = XOR operator
- ipad = Predefined constant (0x3636...3636)
- opad = Predefined constant (0x5C5C...5C5C)

The SHA provides a fully optimized processing of the HMAC algorithm by executing the following operations:

- Sarting the SHA algorithm from any user predefined hash value, thus ‘$h(K_0 \oplus ipad)$’ for first HMAC hash and ‘$h(K_0 \oplus opad)$’ for second HMAC hash.
- Performing automatic padding.
- Routing automatically the first hash result ‘$h((K_0 \oplus ipad) || m)$’ to the source of the second hash processing ‘$h((K_0 \oplus opad) || (\text{first hash result}))$’ including the concatenation of the first hash result to ‘$K_0 \oplus opad$’.

To perform the HMAC operation, the ALGO field value must be greater than 7, the automatic padding feature must be enabled (MSGSIZE and BYTCNT fields differ from 0) and the SHA internal initial hash value registers 0 and 1 must be configured, respectively, with the hash results of input blocks “$K_0 \oplus ipad$” and “$K_0 \oplus opad$” (see Internal Registers for Initial Hash Value or Expected Hash Result).

The size of the message (‘m’) must be written in the MSGSIZE and BYTCNT fields.

The FIRST bit in the SHA Control register (SHA_CR) should be set before writing the first block of the message.

The SHA can process HMAC-SHA1, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512 by configuring the ALGO field in the SHA_MR.

44.4.3 Processing Period

The processing period can be configured.

The short processing period allocates bandwidth to the SHA module, whereas the long processing period allocates more bandwidth on the system bus to other applications. An example is DMA channels not associated with SHA.

In SHA1 mode, the shortest processing period is 85 clock cycles + 2 clock cycles for start command synchronization. The longest period is 209 clock cycles + 2 clock cycles.

In SHA384, SHA512 mode, the shortest processing period is 88 clock cycles + 2 clock cycles for start command synchronization. The longest period is 209 clock cycles + 2 clock cycles.

In SHA256 and SHA224 mode, the shortest processing period is 72 clock cycles + 2 clock cycles for start command synchronization. The longest period is 194 clock cycles + 2 clock cycles.

44.4.4 Double Input Buffer

The SHA Input Data registers (SHA_IDATARx) can be double-buffered to reduce the runtime of large files.

Double-buffering allows a new message block to be written while the previous message block is being processed. This is only possible when DMA accesses are performed (SMOD = 2).

The DUALBUFF bit in the SHA_MR must be set to have double input buffer access.

44.4.5 Internal Registers for Initial Hash Value or Expected Hash Result

The SHA module embeds two sets of internal registers (IR0, IR1) to store different data used by the SHA or HMAC algorithms (See the figure below). These internal registers are accessed through SHA Input Data registers (SHA_IDATARx).

When the ALGO field selects SHA algorithms, IR0 can be configured with a user initial hash value. This initial hash value can be used to compute a custom hash algorithm with two sets of different initial constants, or to continue a hash computation by providing the intermediate hash value previously returned by the SHA module.

When the ALGO field selects SHA algorithms, IR1 can be configured with either a user initial hash value or an expected hash result. The expected hash result must be configured in the IR1 if the field CHECK = 1 (refer to Automatic Check). If the field CHECK = 0 or 2, IR1 can be configured with a user initial hash value that differs from IR0 value.

When the ALGO field selects HMAC algorithms, IR0 must be configured with the hash result of $K_0 \oplus$ ipad and IR1 must be configured with the hash result of $K_0 \oplus$ opad. These pre-computed first blocks speed up the HMAC computation by saving the time to compute the intermediate hash values of the first block which is constant while the secret key is constant (See HMAC Algorithm).

Table 44-1. Configuration Values of Internal Registers

Register	SHA Modes (ALGO < 8)			HMAC Modes (ALGO > 7)
	CHECK = 0	CHECK = 1	CHECK = 2	
IR0	User Initial Hash	User Initial Hash	User Initial Hash	hash($K_0 \oplus$ ipad)
IR1	User Initial Hash	Expected Hash Result	User Initial Hash	hash($K_0 \oplus$ opad)

To calculate the initial HMAC values, follow this sequence:

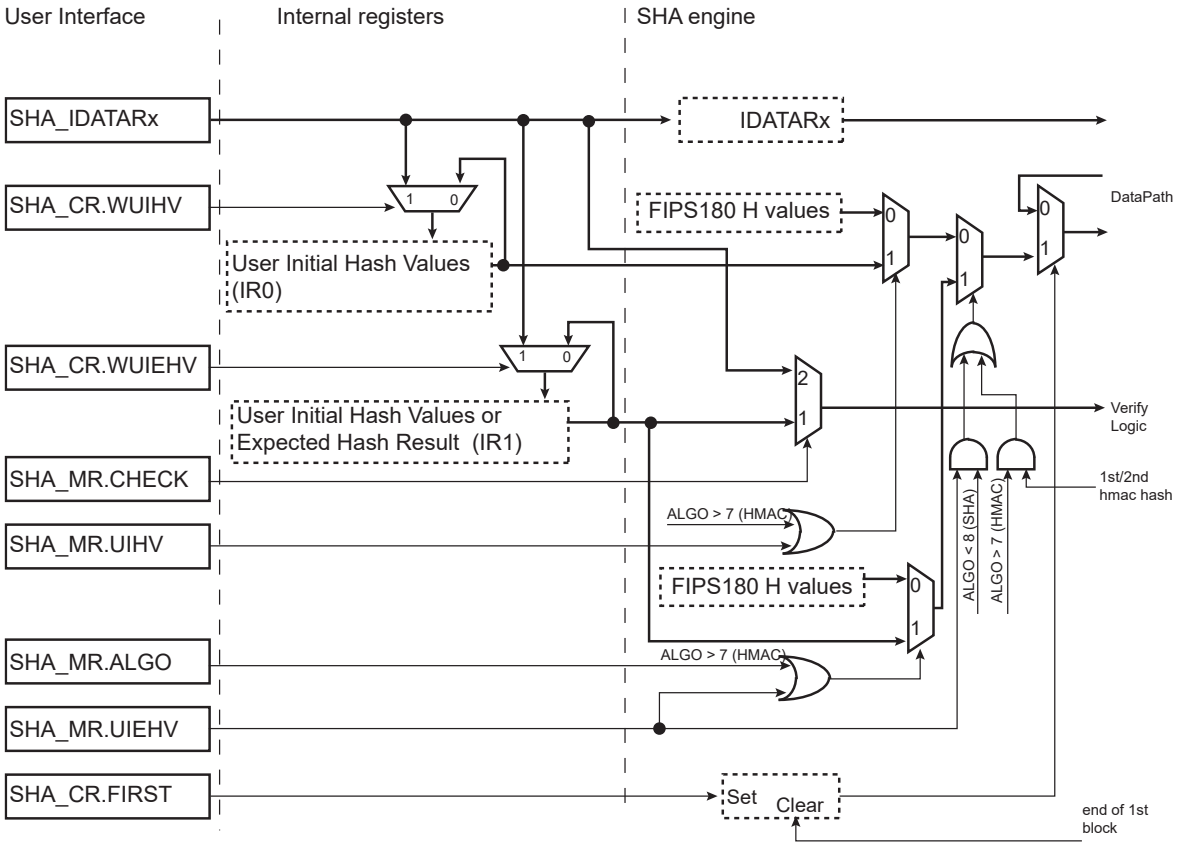
1. Calculate K_0.
2. Calculate $K_0 \oplus$ ipad and $K_0 \oplus$ opad.
3. Perform a hash of the result of $K_0 \oplus$ ipad and $K_0 \oplus$ opad (auto-padding must be disabled for that type of hash).
4. Write h($K_0 \oplus$ ipad) and h($K_0 \oplus$ opad) in IR0 and IR1 respectively.

To write IR0 or IR1, follow this sequence:

1. Set SHA_CR. WUIHV (IR0) or SHA_CR.WUIEHV (IR1).
2. Write the data in SHA_IDATARx. The number of registers to write depends on the type of data (user initial hash values or expected hash result) and on the type of algorithm selected:
   - SHA_IDATAR0 to SHA_IDATAR4 for data used in algorithms based on SHA1
   - SHA_IDATAR0 to SHA_IDATAR7 for data used in algorithms based on SHA256
   - SHA_IDATAR0 to SHA_IDATAR15 for data used in algorithms based on SHA512
   - SHA_IDATAR0 to SHA_IDATAR6 for expected hash result of algorithms based on SHA224
   - SHA_IDATAR0 to SHA_IDATAR11 for expected hash result of algorithms based on SHA384
3. Clear SHA_CR.WUIHV or SHA_CR.WUIEHV.

IR0 and IR1 are automatically selected for HMAC processing if the field ALGO selects HMAC algorithms. If SHA algorithms are selected, the internal registers are selected if the corresponding UIHV or UIEHV bits are set.

Figure 44-1. User Initial Hash Value and Expected Hash Internal Register Access



44.4.6 Automatic Padding

The SHA module features an automatic padding computation to speed up the execution of the algorithm.

The automatic padding function requires the following information:

- Complete message size in bytes to be written in the MSGSIZE field of the SHA Message Size register (SHA_MSR).
  The size of the message is written at the end of the last block, as required by the FIPS180-2 specification (the size is automatically converted into a bit-size).
- Number of remaining bytes (to write in the SHA_IDATARx) to be written in the BYTCNT field of the SHA Bytes Count register (SHA_BCR).
  Automatic padding occurs when the BYTCNT field reaches 0. At each write in the SHA Input registers, the BYTCNT field value is decreased by the number of bytes written.

The BYTCNT field value must be written with the same value as the MSGSIZE field value if the full message is processed. If the message is partially preprocessed and an initial hash value is used, BYTCNT must be written with the remaining bytes to hash while MSGSIZE holds the message size.

To disable the automatic padding feature, the MSGSIZE and BYTCNT fields must be configured with 0.

44.4.7 Automatic Check

The SHA module features an automatic check of the hash result with the expected hash. A check failure can generate an interrupt if configured in the SHA Interrupt Enable register (SHA_IER).

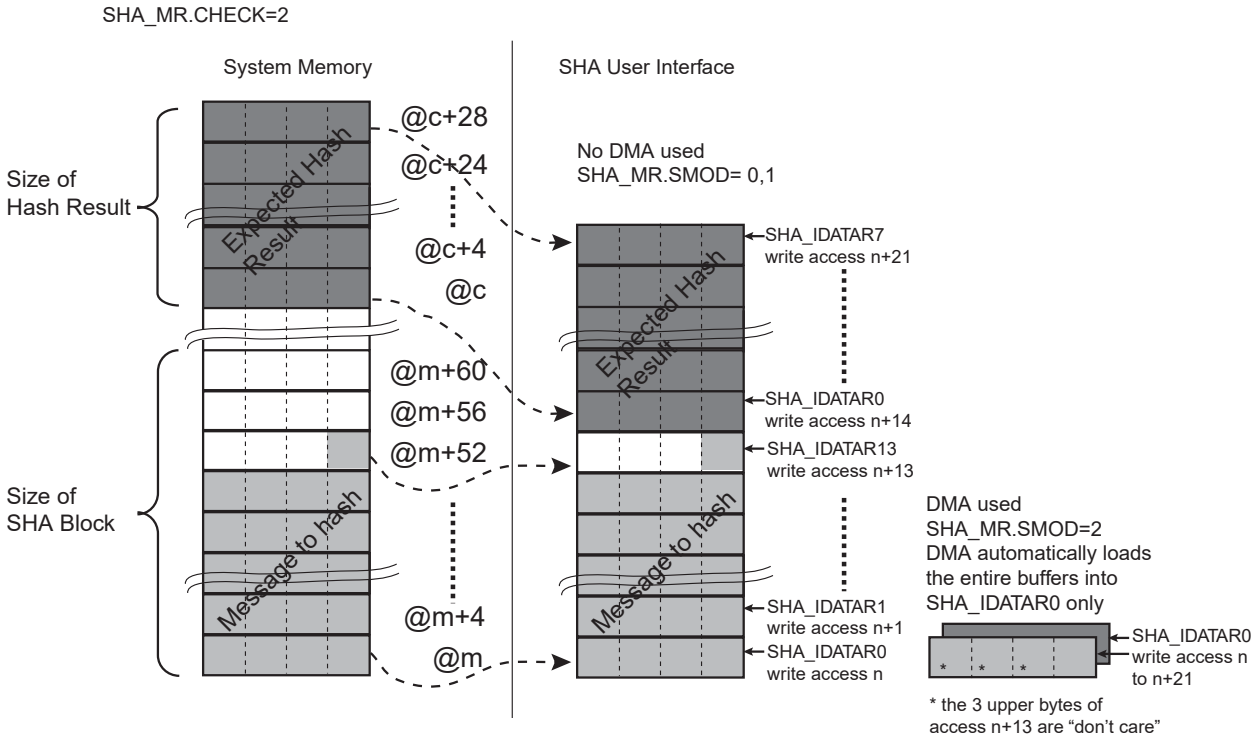
Automatic check requires the automatic padding feature to be enabled (MSGSIZE and BYTCNT fields must be greater than 0).

There are two methods to configure the expected hash result:

- if SHA_MR.CHECK = 1, the expected hash result is read from the internal register (IR1). This method cannot be used when HMAC algorithms is selected because this register is already used to store user initial hash values for the second hash processing. IR1 cannot be read by software.
- If SHA_MR.CHECK = 2, the expected hash result is written in the SHA_IDATARx after the message.

When SHA_MR.CHECK = 2, the method can provide more flexibility of use if a message is stored in system memory together with its expected hash result. A DMA with linked list can be used to ease the transfer of the message and its expected hash result.

Figure 44-2. Message and Expected Hash Result Memory Mapping



The number of 32-bit words of the hash result to check with the expected hash can be selected with SHA_MR.CHKCNT. The status of the check is available in the CHKST field in the SHA Interrupt Status register (SHA_ISR).

An interrupt can be generated (if enabled) when the check is completed. The check occurs several clock cycles after the computation of the requested hash, so the interrupt and the CHECKF bit are set several clock cycles after the DATRDY flag of the SHA_ISR.

44.4.8 Protocol Layers Improved Performances

The SHA can be tightly coupled to the AES module to improve performances when processing protocol layers such as IPsec or OpenSSL.

When the AES is configured to be tightly coupled to SHA (AES_MR), SHA must be always configured in Double Buffer mode (SHA_MR.DUALBUFF = 1).

Refer to the section “Advanced Encryption Standard (AES)” for details.

44.4.9 Start Modes

SHA_MR.SMOD is used to select the Hash Processing Start mode.

44.4.9.1 Manual Mode

In Manual mode, the sequence is as follows:

1. Set SHA_IER.DATRDY (Data Ready) , depending on whether an interrupt is required at the end of processing.
2. If the initial hash values differ from the FIPS standard, set SHA_MR.UIHV and/or SHA_MR.UIEHV. If the initial hash values comply with the FIPS180-2 specification, clear SHA_MR.UIHV and/or SHA_MR.UIEHV.

3. If automatic padding is required, configure SHA_MSR.MSGSIZE with the number of bytes of the message, and configure SHA_BCR.BYTCNT with the remaining number of bytes to write. The BYTCNT field must be written with a value different from MSGSIZE field value if the message is preprocessed and completed by using user initial hash values.
   If automatic padding is not required, configure SHA_MSR.MSGSIZE and SHA_BCR.BYTCNT to 0.
4. For the first block of a message, the FIRST command must be set by writing a 1 into the corresponding bit of the Control register (SHA_CR). For the other blocks, there is nothing to write.
5. Write the block to be processed in the SHA_IDATARx.
6. To begin processing, set SHA_CR.START.
7. When processing is completed, the bit DATRDY in the Interrupt Status register (SHA_ISR) rises. If an interrupt has been enabled by setting SHA_IER.DATRDY, the interrupt line of the SHA is activated.
8. Repeat the write procedure for each block, start procedure and wait for the interrupt procedure up to the last block of the entire message. Each time the start procedure is complete, the DATRDY flag is cleared.
9. After the last block is processed (DATRDY flag is set, if an interrupt has been enabled by setting SHA_IER.DATRDY, the interrupt line of the SHA is activated), read the message digest in the Output Data registers. The DATRDY flag is automatically cleared when reading the SHA_IODATARx registers.

44.4.9.2 Auto Mode

In Auto mode, processing starts as soon as the correct number of SHA_IDATARx is written. No action in the SHA_CR is necessary.

44.4.9.3 DMA Mode

The DMA can be used in association with the SHA to perform the algorithm on a complete message without any action by the software during processing.

SHA_MR.SMOD must be configured to 2.

The DMA must be configured with non-incremental addresses.

The start address of any transfer descriptor must be set to point to the SHA_IDATAR0.

The DMA chunk size must be set to transfer, for each trigger request, 16 words of 32 bits.

The FIRST bit of the SHA_CR must be set before starting the DMA when the first block is transferred.

The DMA generates an interrupt when the end of buffer transfer is completed but the SHA processing is still in progress. The end of SHA processing is indicated by the flag DATRDY in the SHA_ISR.

If automatic padding is disabled, the end of SHA processing requires two interrupts to be verified. The DMA end of transfer interrupt must be verified first, then the SHA DATRDY interrupt must be enabled and verified.

If automatic padding is enabled, the end of SHA processing requires only one interrupt to be verified. The DMA end of transfer is not required, so the SHA DATRDY interrupt must be enabled prior to start the DMA and DATRDY interrupt is the only one to be verified. Refer to the figures below.

Figure 44-3. Interrupts Processing with DMA

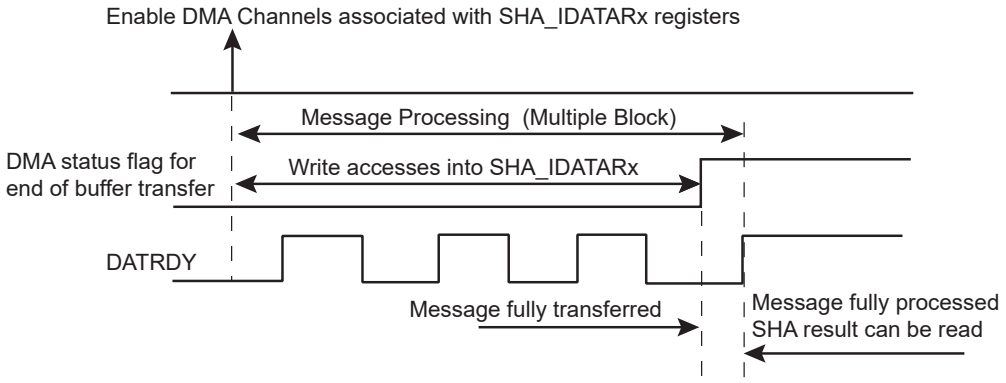
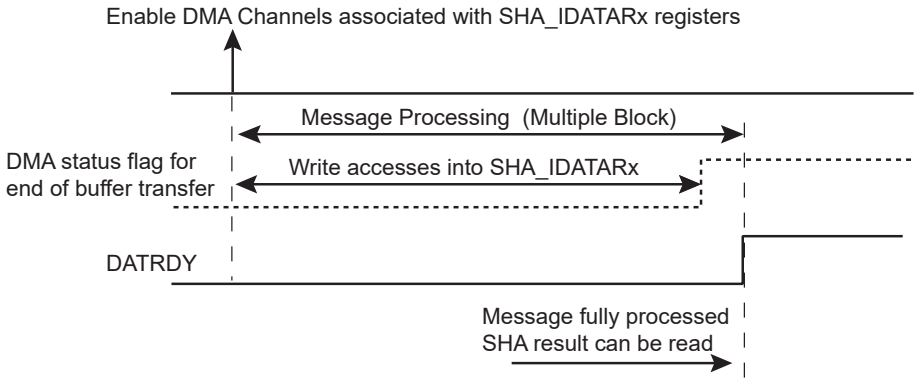


Figure 44-4. Interrupts Processing with DMA and Automatic Padding



44.4.9.4 SHA Register Endianness

In ARM processor-based products, the system bus and processors manipulate data in little-endian form. The SHA interface requires little-endian format words. However, in accordance with the protocol of FIPS 180-2 specification, data is collected, processed and stored by the SHA algorithm in big-endian form.

The following example illustrates how to configure the SHA:

If the first 64 bits of a message (according to FIPS 180-2, i.e., big-endian format) to be processed is 0xcafedeca_01234567, then the SHA_IDATAR0 and SHA_IDATAR1 registers must be written with the following pattern:

- SHA_IDATAR0 = 0xcadefeca
- SHA_IDATAR1 = 0x67452301

In a little-endian system, the message (according to FIPS 180-2) starting with pattern 0xcafedeca_01234567 is stored into memory as follows:

- 0xca stored at initial offset (for example 0x00),
- then 0xfe stored at initial offset + 1 (i.e., 0x01),
- 0xde stored at initial offset + 2 (i.e., 0x02),
- 0xca stored at initial offset + 3 (i.e., 0x03).

If the message is received through a serial-to-parallel communication channel, the first received character is 0xca and it is stored at the first memory location (initial offset). The second byte, 0xfe, is stored at initial offset + 1.

When reading on a 32-bit little-endian system bus, the first word read back from system memory is 0xcadefeca.

When the SHA_IODATARx registers are read, the hash result is organized in little-endian format, allowing system memory storage in the same format as the message.

Taking an example from the FIPS 180-2 specification Appendix B.1, the endian conversion can be observed.

For this example, the 512-bit message is:

0x6162638000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000018

and the expected SHA-256 result is:

0xba7816bf_8f01cfea_414140de_5dae2223_b00361a3_96177a9c_b410ff61_f20015ad

If the message has not already been stored in the system memory, the first step is to convert the input message to little-endian before writing to the SHA_IDATARx registers. This would result in a write of:

SHA_IDATAR0 = 0x80636261..... SHA_IDATAR15 = 0x18000000

The data in the output message digest registers, SHA_IODATARx, contain SHA_IODATAR0 = 0xbf1678ba... SHA_IODATAR7 = 0xad1500f2 which is the little-endian format of 0xba7816bf,..., 0xf20015ad.

Reading SHA_IODATAR0 to SHA_IODATAR1 and storing into a little-endian memory system forces hash results to be stored in the same format as the message.

When the output message is read, the user can convert back to big-endian for a resulting message value of:

0xba7816bf_8f01cfea_414140de_5dae2223_b00361a3_96177a9c_b410ff61_f20015ad

44.4.10 Security Features

44.4.10.1 Unspecified Register Access Detection

When an unspecified register access occurs, the URAD bit in the SHA_ISR is set. Its source is then reported in the Unspecified Register Access Type field (URAT). Only the last unspecified register access is available through the URAT field.

Several kinds of unspecified register accesses can occur:

- SHA_IDATARx written during data processing in DMA mode
- SHA_IODATARx read during data processing
- SHA_MR written during data processing
- Write-only register read access

The URAD bit and the URAT field can only be reset by the SWRST bit in the SHA_CR.

44.4.10.2 Register Write Protection

To prevent any single software error from corrupting SHA behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable), WPITEN (Write Protection Interrupt Enable), and/or WPCREN (Write Protection Control Enable) bits in the SHA Write Protection Mode register (SHA_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the SHA Write Protection Status register (SHA_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading SHA_WPSR.

The following register(s) can be write-protected when SHA_WPMR.WPEN is set:

- SHA Mode Register
- SHA Message Size Register
- SHA Bytes Count Register

The following register(s) can be write-protected when WPITEN is set:

- SHA Interrupt Enable Register
- SHA Interrupt Disable Register

The following register(s) can be write-protected when WPCREN is set:

- SHA Control Register

44.5 Register Summary

Offset	Name	Bit Pos.								
0x00	SHA_CR	7:0				FIRST				START
		15:8			WUIEHV	WUIHV				SWRST
		23:16								
		31:24								
0x04	SHA_MR	7:0		UIEHV	UIHV				SMOD[1:0]	
		15:8					ALGO[3:0]			
		23:16								DUALBUFF
		31:24	CHKCNT[3:0]						CHECK[1:0]	
0x08 ... 0x0F	Reserved									
0x10	SHA_IER	7:0								DATRDY
		15:8								URAD
		23:16								CHECKF
		31:24								SECE
0x14	SHA_IDR	7:0								DATRDY
		15:8								URAD
		23:16								CHECKF
		31:24								SECE
0x18	SHA_IMR	7:0								DATRDY
		15:8								URAD
		23:16								CHECKF
		31:24								SECE
0x1C	SHA_ISR	7:0				WRDY				DATRDY
		15:8		URAT[2:0]						URAD
		23:16	CHKST[3:0]							CHECKF
		31:24								SECE
0x20	SHA_MSR	7:0	MSGSIZE[7:0]							
		15:8	MSGSIZE[15:8]							
		23:16	MSGSIZE[23:16]							
		31:24	MSGSIZE[31:24]							
0x24 ... 0x2F	Reserved									
0x30	SHA_BCR	7:0	BYTCNT[7:0]							
		15:8	BYTCNT[15:8]							
		23:16	BYTCNT[23:16]							
		31:24	BYTCNT[31:24]							
0x34 ... 0x3F	Reserved									
0x40	SHA_IDATAR0	7:0	IDATA[7:0]							
		15:8	IDATA[15:8]							
		23:16	IDATA[23:16]							
		31:24	IDATA[31:24]							
...										
0x7C	SHA_IDATAR15	7:0	IDATA[7:0]							
		15:8	IDATA[15:8]							
		23:16	IDATA[23:16]							
		31:24	IDATA[31:24]							
0x80	SHA_IODATAR0	7:0	IODATA[7:0]							
		15:8	IODATA[15:8]							
		23:16	IODATA[23:16]							
		31:24	IODATA[31:24]							
...										

.....continued

Offset	Name	Bit Pos.								
0xBC	SHA_IODATAR15	7:0	IODATA[7:0]							
		15:8	IODATA[15:8]							
		23:16	IODATA[23:16]							
		31:24	IODATA[31:24]							
0xC0 ... 0xE3	Reserved									
0xE4	SHA_WPMR	7:0						WPCREN	WPITEN	WPEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0xE8	SHA_WPSR	7:0								WPVS
		15:8	WPVSRC[7:0]							
		23:16								
		31:24								

44.5.1 SHA Control Register

Name: SHA_CR
Offset: 0x00
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
			WUIEHV	WUIHV				SWRST
Access			W	W				W
Reset			–	–				–

Bit	7	6	5	4	3	2	1	0
				FIRST				START
Access				W				W
Reset				–				–

Bit 13 – WUIEHV Write User Initial or Expected Hash Values

Value	Description
0	SHA_IDATARx accesses are routed to the data registers.
1	SHA_IDATARx accesses are routed to the internal registers (IR1).

Bit 12 – WUIHV Write User Initial Hash Values

Value	Description
0	SHA_IDATARx accesses are routed to the data registers.
1	SHA_IDATARx accesses are routed to the internal registers (IR0).

Bit 8 – SWRST Software Reset

Value	Description
0	No effect.
1	Resets the SHA. A software-triggered hardware reset of the SHA interface is performed.

Bit 4 – FIRST First Block of a Message

Value	Description
0	No effect.
1	Indicates that the next block to process is the first one of a message.

Bit 0 – START Start Processing

Value	Description
0	No effect.
1	Starts manual hash algorithm process.

44.5.2 SHA Mode Register

Name: SHA_MR
Offset: 0x04
Reset: 0x0000100
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	CHKCNT[3:0]						CHECK[1:0]	
Access								
Reset	0	0	0	0			0	0

Bit	23	22	21	20	19	18	17	16
								DUALBUFF
Access								
Reset								0

Bit	15	14	13	12	11	10	9	8
					ALGO[3:0]			
Access								
Reset					0	0	0	1

Bit	7	6	5	4	3	2	1	0
		UIEHV	UIHV				SMOD[1:0]	
Access								
Reset		0	0				0	0

Bits 31:28 – CHKCNT[3:0] Check Counter
Number of 32-bit words to check. The value 0 indicates that the number of words to compare will be based on the algorithm selected (5 words for SHA1, 7 words for SHA224, 8 words for SHA256, 12 words for SHA384, 16 words for SHA512).

Bits 25:24 – CHECK[1:0] Hash Check
Values not listed in table must be considered as “reserved”.

Value	Name	Description
0	NO_CHECK	No check is performed
1	CHECK_EHV	Check is performed with expected hash stored in internal expected hash value registers.
2	CHECK_MESSAGE	Check is performed with expected hash provided after the message.

Bit 16 – DUALBUFF Dual Input Buffer

Value	Name	Description
0	INACTIVE	SHA_IDATARx and SHA_IODATARx cannot be written during processing of previous block.
1	ACTIVE	SHA_IDATARx and SHA_IODATARx can be written during processing of previous block when SMOD value = 2. It speeds up the overall runtime of large files.

Bits 11:8 – ALGO[3:0] SHA Algorithm
Values not listed in the table must be considered as “reserved”.

Value	Name	Description
0	SHA1	SHA1 algorithm processed
1	SHA256	SHA256 algorithm processed
2	SHA384	SHA384 algorithm processed
3	SHA512	SHA512 algorithm processed
4	SHA224	SHA224 algorithm processed
8	HMAC_SHA1	HMAC algorithm with SHA1 Hash processed

Value	Name	Description
9	HMAC_SHA256	HMAC algorithm with SHA256 Hash processed
10	HMAC_SHA384	HMAC algorithm with SHA384 Hash processed
11	HMAC_SHA512	HMAC algorithm with SHA512 Hash processed
12	HMAC_SHA224	HMAC algorithm with SHA224 Hash processed

Bit 6 – UIEHV User Initial or Expected Hash Value Registers

Value	Description
0	The SHA algorithm is started with the standard initial values as defined in the FIPS180-2 specification.
1	The SHA algorithm is started with the user initial hash values stored in the internal register 1 (IR1). If HMAC is configured, UIEHV has no effect (i.e. IR1 is always selected).

Bit 5 – UIHV User Initial Hash Values

Value	Description
0	The SHA algorithm is started with the standard initial values as defined in the FIPS180-2 specification.
1	The SHA algorithm is started with the user initial hash values stored in the internal register 0 (IR0). If HMAC is configured, UIHV has no effect (i.e. IR0 is selected).

Bits 1:0 – SMOD[1:0] Start Mode
Values not listed in the table must be considered as “reserved”.
If a DMA transfer is used, configure the SMOD value to 2. See DMA Mode for details.

Value	Name	Description
0	MANUAL_START	Manual mode
1	AUTO_START	Auto mode
2	IDATAR0_START	SHA_IDATAR0 access only mode (mandatory when DMA is used)

44.5.3 SHA Interrupt Enable Register

Name: SHA_IER
Offset: 0x10
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
								SECE
Access								W
Reset								–

Bit	23	22	21	20	19	18	17	16
								CHECKF
Access								W
Reset								–

Bit	15	14	13	12	11	10	9	8
								URAD
Access								W
Reset								–

Bit	7	6	5	4	3	2	1	0
								DATRDY
Access								W
Reset								–

Bit 24 – SECE Safety Event Interrupt Enable

Bit 16 – CHECKF Check Done Interrupt Enable

Bit 8 – URAD Unspecified Register Access Detection Interrupt Enable

Bit 0 – DATRDY Data Ready Interrupt Enable

44.5.4 SHA Interrupt Disable Register

Name: SHA_IDR
Offset: 0x14
Reset: –
Property: Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

Bit	31	30	29	28	27	26	25	24
								SECE
Access								W
Reset								–

Bit	23	22	21	20	19	18	17	16
								CHECKF
Access								W
Reset								–

Bit	15	14	13	12	11	10	9	8
								URAD
Access								W
Reset								–

Bit	7	6	5	4	3	2	1	0
								DATRDY
Access								W
Reset								–

Bit 24 – SECE Safety Event Interrupt Disable

Bit 16 – CHECKF Check Done Interrupt Disable

Bit 8 – URAD Unspecified Register Access Detection Interrupt Disable

Bit 0 – DATRDY Data Ready Interrupt Disable

44.5.5 SHA Interrupt Mask Register

Name: SHA_IMR
Offset: 0x18
Reset: 0x00000000
Property: Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

Bit	31	30	29	28	27	26	25	24
								SECE
Access								R
Reset								0

Bit	23	22	21	20	19	18	17	16
								CHECKF
Access								R
Reset								0

Bit	15	14	13	12	11	10	9	8
								URAD
Access								R
Reset								0

Bit	7	6	5	4	3	2	1	0
								DATRDY
Access								R
Reset								0

Bit 24 – SECE Safety Event Interrupt Mask

Bit 16 – CHECKF Check Done Interrupt Mask

Bit 8 – URAD Unspecified Register Access Detection Interrupt Mask

Bit 0 – DATRDY Data Ready Interrupt Mask

44.5.6 SHA Interrupt Status Register

Name: SHA_ISR
Offset: 0x1C
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
								SECE
Access								R
Reset								0

Bit	23	22	21	20	19	18	17	16
	CHKST[3:0]							CHECKF
Access	R	R	R	R				R
Reset	0	0	0	0				0

Bit	15	14	13	12	11	10	9	8
		URAT[2:0]						URAD
Access		R	R	R				R
Reset		0	0	0				0

Bit	7	6	5	4	3	2	1	0
				WRDY				DATRDY
Access				R				R
Reset				0				0

Bit 24 – SECE Security and/or Safety Event

Value	Description
0	There is no report in SHA_WPSR.
1	There is a Security and/or Safety Event reported in SHA_WPSR.

Bits 23:20 – CHKST[3:0] Check Status (cleared by writing START or SWRST bits in SHA_CR or by reading SHA_IODATARx)
Value 5 indicates identical hash values (expected hash = hash result). Any other value indicates different hash values.

Bit 16 – CHECKF Check Done Status (cleared by writing START or SWRST bits in SHA_CR or by reading SHA_IODATARx)

Value	Description
0	Hash check has not been computed.
1	Hash check has been computed, status is available in the CHKST bits.

Bits 14:12 – URAT[2:0] Unspecified Register Access Type (cleared by writing a 1 to SWRST bit in SHA_CR)
Only the last Unspecified Register Access Type is available through the URAT field.

Value	Name
0	SHA_IDATAR0 to SHA_IDATAR15 written during the data processing in DMA mode (URAD = 1 and URAT = 0 can occur only if DUALBUFF is cleared in SHA_MR).
1	Output Data Register read during the data processing.
2	SHA_MR written during the data processing.
3	Write-only register read access.

Bit 8 – URAD Unspecified Register Access Detection Status (cleared by writing a 1 to SWRST bit in SHA_CR)

Value	Description
0	No unspecified register access has been detected since the last SWRST.
1	At least one unspecified register access has been detected since the last SWRST.

Bit 4 – WRDY Input Data Register Write Ready

Value	Description
0	SHA_IDATAR0 cannot be written
1	SHA_IDATAR0 can be written

Bit 0 – DATRDY Data Ready (cleared by writing a 1 to bit SWRST or START in SHA_CR, or by reading SHA_IODATARx)

Value	Description
0	Output data is not valid.
1	512/1024-bit block process is completed. DATRDY is cleared when one of the following conditions is met: • Bit START in SHA_CR is set. • Bit SWRST in SHA_CR is set. • The hash result is read.

44.5.7 SHA Message Size Register

Name: SHA_MSR
Offset: 0x20
Reset: 0x0
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	MSGSIZE[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	MSGSIZE[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	MSGSIZE[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	MSGSIZE[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – MSGSIZE[31:0] Message Size
The size in bytes of the message. When MSGSIZE differs from 0, the SHA appends the corresponding value converted in bits after the padding section, as described in the FIPS180-2 specification.
To disable automatic padding, MSGSIZE field must be written to 0.

44.5.8 SHA Bytes Count Register

Name: SHA_BCR
Offset: 0x30
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	BYTCNT[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	BYTCNT[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	BYTCNT[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	BYTCNT[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – BYTCNT[31:0] Remaining Byte Count Before Auto Padding
When the hash processing starts from the beginning of a message (without preprocessed hash part), BYTCNT must be written with the same value as the MSGSIZE. If a part of the message has been already hashed and the hash does not start from the beginning, BYTCNT must be configured with the number of bytes remaining to process before padding section.
When read, provides the size in bytes of message remaining to be written before the automatic padding starts.
BYTCNT field is automatically updated each time a write occurs in the SHA_IDATARx and SHA_IODATARx.
When BYTCNT reaches 0, the MSGSIZE is converted into bit count and appended at the end of the message after the padding as described in the FIPS180-2 specification.
To disable automatic padding, MSGSIZE and BYTCNT fields must be written to 0.

44.5.9 SHA Input Data x Register

Name: SHA_IDATARx
Offset: 0x40 + x*0x04 [x=0..15]
Reset: –
Property: Write-only

Bit	31	30	29	28	27	26	25	24
	IDATA[31:24]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	IDATA[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IDATA[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IDATA[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	–

Bits 31:0 – IDATA[31:0] Input Data
The 32-bit Input Data registers allow to load the data block used for hash processing.
These registers are write-only to prevent the input data from being read by another application.
SHA_IDATAR0 corresponds to the first word of the block, SHA_IDATAR15 to the last word of the last block in case SHA algorithm is set to SHA1, SHA224, SHA256 or SHA_IODATA15R to the last word of the block if SHA algorithm is SHA384 or SHA512 (see SHA Input/Output Data Register x).

44.5.10 SHA Input/Output Data Register x

Name: SHA_IODATARx
Offset: 0x80 + x*0x04 [x=0..15]
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	IODATA[31:24]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	IODATA[23:16]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	IODATA[15:8]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
	IODATA[7:0]							
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 31:0 – IODATA[31:0] Input/Output Data
These registers can be used to read the resulting message digest and to write the second part of the message block when the SHA algorithm is SHA-384 or SHA-512.
SHA_IODATA0R to SHA_IODATA15R can be written or read but reading these offsets does not return the content of corresponding parts (words) of the message block. Only results from SHA calculation can be read through these registers.
When SHA processing is in progress, these registers return 0x0000.
SHA_IODATAR0 corresponds to the first word of the message digest; SHA_IODATAR4 to the last one in SHA1 mode, SHA_ODATAR6 in SHA224, SHA_IODATAR7 in SHA256, SHA_IODATAR11 in SHA384 or SHA_IODATAR15 in SHA512.
When SHA224 is selected, the content of SHA_ODATAR7 must be ignored.
When SHA384 is selected, the content of SHA_IODATAR12 to SHA_IODATAR15 must be ignored.

44.5.11 SHA Write Protection Mode Register

Name: SHA_WPMR
Offset: 0xE4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
						WPCREN	WPITEN	WPEN
Access						R/W	R/W	R/W
Reset						0	0	0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x534841	PASSWD	Writing any other value in this field aborts the write operation of the WPEN,WPITEN,WPCREN bits. Always reads as 0.

Bit 2 – WPCREN Write Protection Control Enable

Value	Description
0	Disables the write protection on control register if WPKEY corresponds to 0x534841 (“SHA” in ASCII).
1	Enables the write protection on control register if WPKEY corresponds to 0x534841 (“SHA” in ASCII).

Bit 1 – WPITEN Write Protection Interruption Enable

Value	Description
0	Disables the write protection on interrupt registers if WPKEY corresponds to 0x534841 (“SHA” in ASCII).
1	Enables the write protection on interrupt registers if WPKEY corresponds to 0x534841 (“SHA” in ASCII).

Bit 0 – WPEN Write Protection Configuration Enable
See Register Write Protection for the list of registers that can be write-protected.

Value	Description
0	Disables the write protection on configuration registers if WPKEY corresponds to 0x534841 (“SHA” in ASCII).
1	Enables the write protection on configuration registers if WPKEY corresponds to 0x534841 (“SHA” in ASCII).

44.5.12 SHA Write Protection Status Register

Name: SHA_WPSR
Offset: 0xE8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	WPVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
								WPVS
Access								R
Reset								0

Bits 15:8 – WPVSRC[7:0] Write Protection Violation Source
When WPVS=1, WPVSRC indicates the register address offset at which a write access has been attempted.

Bit 0 – WPVS Write Protection Violation Status (cleared on read)

Value	Description
0	No write protect violation has occurred since the last read of SHA_WPSR.
1	A write protect violation has occurred since the last read of SHA_WPSR. The address offset of the violated register is reported into field WPVSRC.

45. Non-Maskable Interrupt Controller (NMIC)

45.1 Description

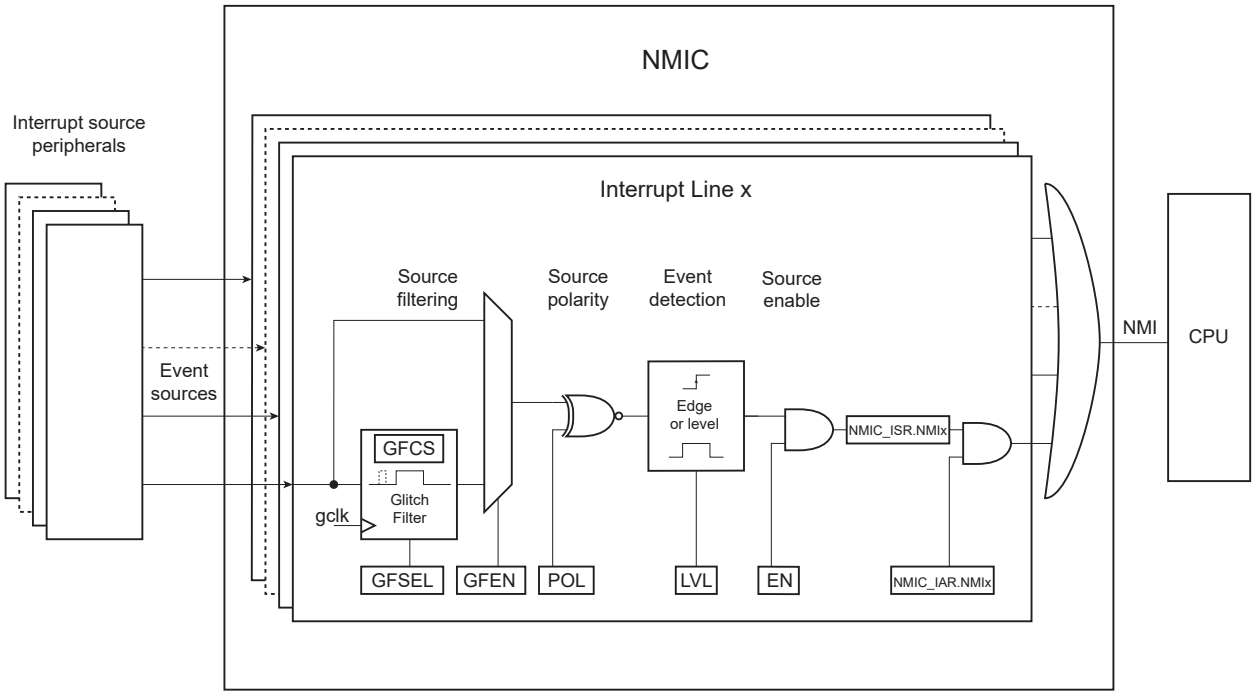
The Non-Maskable Interrupt Controller (NMIC) forwards critical interrupts from the Supply Controller (SUPC) and Power Management Controller (PMC) internal interrupt sources, or from the PIO Controller external interrupt sources to the NMI input of the processor in a programmable way.

45.2 Embedded Characteristics

- Glitch Filter
- Programmable Source Polarity
- Edge or Level Detection
- Freeze Capability for Each Interrupt Line Configuration
- Global Configuration Write Protection Register with Key

45.3 Block Diagram

Figure 45-1. Block Diagram



45.4 I/O Line Description

Table 45-1. I/O Line Description

Pin Name	Pin Description	Type
NMIC_NMI	Non-maskable Interrupt	Input

45.5 Product Dependencies

45.5.1 I/O Lines

The external interrupt signal NMIC_NMI is multiplexed through the PIO controller(s).

Depending on the features of the PIO controller used in the product, the pin must be programmed in accordance with its assigned interrupt function. This is not applicable when the PIO controller used in the product is transparent on the input path.

45.5.2 Power Management

The NMIC is not continuously clocked. It must be activated on the Power Management Controller before using it.

Prior to using the Glitch Filter, the NMIC Generic Clock (gclk), must be enabled in the Power Management Controller.

45.5.3 Interrupt Sources

The following table describes the interrupt sources available in the product.

Table 45-2. Interrupt Sources

Source No.	Source Name	Source Description
NMI0	NMI	NMI
NMI1	CPU_FAIL	CPU_FAIL
NMI2	XTAL_12M_FAIL	XTAL_12M_FAIL
NMI3	XTAL_32K FAIL	XTAL_32K FAIL
NMI4	VDDCORE_FAIL	VDDCORE_FAIL
NMI5	NOFIX_TCM	NOFIX_TCM
NMI6	NOFIX_HEMC	NOFIX_HEMC
NMI7	NOFIX_HEFC	NOFIX_HEFC
NMI8	NOFIX_FlexRAM	NOFIX_FlexRAM
NMI9	–	–
NMI10	–	–
NMI11	–	–
NMI12	–	–
NMI13	–	–
NMI14	–	–
NMI15	–	–

45.6 Functional Description

45.6.1 Interrupt Line Characteristics

For each event source, the NMIC features a dedicated interrupt line as shown in the block diagram above.

If the Glitch Filter Enable (GFEN) bit is cleared in the Source Configuration Register (NMIC_SCFGxR), pulses of any width can be forwarded and detected as valid source events on the interrupt line x regardless of the respective frequencies of the source clock, the generic clock, the NMIC peripheral clock and the CPU clock.

If the NMIC_SCFGxR.GFEN bit is set, the source events are filtered to remove unwanted glitches. The glitch filter is guaranteed to forward its input level when it is maintained for more than 2^{GFSEL} gclk cycles and to reject pulses narrower than 2^{GFSEL}-1 gclk cycles.

The source event polarity can be changed by the NMIC_SCFGxR.POL bit.

Either edge or level event detection can be performed by the NMIC_SCFGxR.LVL bit.

If the NMIC_SCFGxR.EN bit is set, the detection result is flagged into the interrupt pending bit of the Interrupt Status Register (NMIC_ISR).

If the Active Interrupt x (NMIx) bit for the interrupt line is set in the Interrupt Active Register (NMIC_IAR), the pending status is forwarded to the CPU non-maskable interrupt input.

When set, the NMIC_SCFGxR.FRZ bit freezes all interrupt line settings until hardware reset. This includes the settings in the NMIC_SCFGxR and in the interrupt active x bit of the NMIC_IAR. This feature can be used to prevent any corruption of the critical interrupt lines configuration.

The propogation delay of the NMIC affects the activation of the interrupt.

When the NMIC is configured in level-triggered mode, the propogation delay of the NMIC also affects the de-activation of the interrupt.

If the Glitch Filter is enabled in the NMIC_SCFGxR.GFEN bit, the maximum propogation delay through the NMIC (shown in the figure below) is calculated as:

nmic_delay_max = (2 + 2^{GFSEL}) * gclk_period + 2.5 * clock_period

and the minimum propogation delay through the NMIC is:

nmic_delay_min = (1 + 2^{GFSEL}) * gclk_period + 1 * clock_period

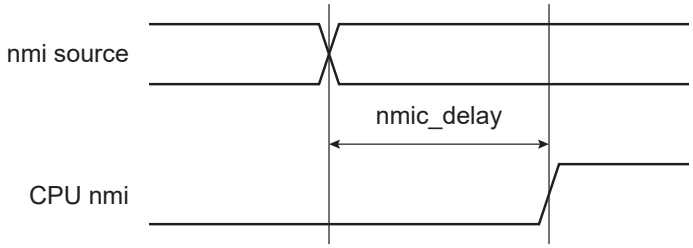
If the Glitch Filter is disabled, the maximum propogation delay through the NMIC is calculated as:

nmic_delay_max = 2.5 * clock_period

and the minimum propogation delay through the NMIC is:

nmic_delay_min = 1 * clock_period

Figure 45-2. NMIC Propagation Delay



The gclk_period is the NMIC generic clock period as programmed in the Power Management Controller.

The clock_period is the peripheral clock period of the NMIC as programmed in the Power Management Controller.

45.6.2 Interrupt Line Programming

The steps to set up or change the interrupt line x configuration are the following:

1. If the source might glitch due to setup changes required at the source peripheral, clear the SCFGxR.EN bit to disable the NMIC interrupt source.
2. Perform any external setup that might be required at the source peripheral and wait until the peripheral source output is stabilized.
3. If the interrupt line is featured with a glitch filter, wait until the Ready flag RDYx is set in the Glitch Filter Configuration Status Register (NMIC_GFCS).
4. Write the required interrupt line configuration fields among GFSEL, GFEN, POL, LVL into the NMIC_SCFGxR.
5. Set the NMI bit in the Interrupt Enable Register (NMIC_IER) to activate the interrupt line if required and if it was disabled.
6. If the interrupt line is featured with a glitch filter, wait until the RDYx flag is set in the NMIC_GFCS register.

7. Set the NMIC_SCFGxR.EN bit if the source is not yet enabled and, if required for safety, freeze the interrupt line x configuration by setting the NMIC_SCFGxR.FRZ bit.

Note that if the source peripheral is glitch free or stable enough compared to the current glitch filter settings, the interrupt source does not need to be disabled with the NMIC_SCFGxR.EN bit prior to its reprogramming.

As an example, if the user desires to stop or minimize the circuit activity after a voltage or frequency failure detection on a rising edge of an NMIC interrupt source, the polarity can be changed with a single write to the NMIC_SCFGxR.POL bit to detect a falling edge of the same source. Detection of the falling edge source indicates the recovery of a normal condition. In this case, the sequence steps described previously can be ignored.

45.6.3 Interrupt Handling

When the interrupt source x is triggered, the corresponding interrupt pending flag NMIx is set by hardware in the NMIC_ISR.

If the interrupt source (NMIx) is enabled in the NMIC_IAR, the CPU non-maskable interrupt input is set and the interrupt handler routine is entered.

The interrupt handler software reads the NMIC_ISR to identify the source of the NMI. This read normally has the effect of clearing the NMIC_ISR so that the software can immediately proceed to the critical task and also immediately detect a new incoming source assertion.

However, if the interrupt was configured as level-triggered in the NMIC_SCFGxR and the source is still keeping the active level at the time of reading the NMIC_ISR, the NMIx bit stays active so that the interrupt can be immediately re-entered upon return from the interrupt handler. To avoid a re-entry to the interrupt handler, the NMIC_ISR can be polled until the NMIx is read at 0 prior ending the interrupt handler.

Alternatively, the interrupt might have been configured as edge-triggered in the NMIC_SCFGxR. In this case, the NMIC_ISR is cleared when read by the software handler routine. The NMIC_ISR.NMIx bit remains at value 0 until the interrupt source is reasserted from the inactive level.

45.6.4 Register Write Protection

To prevent any single software error from corrupting NMIC behavior, certain registers in the address space can be write-protected.

The following registers can be write-protected by setting the WPCFEN bit in the NMIC Write Protection Mode Register (NMIC_WPMR):

- NMIC Source Configuration Register x (NMIC_SCFGxR)

The following registers can be write-protected by setting the WPITEN bit in the NMIC Write Protection Mode Register (NMIC_WPMR):

- NMIC Interrupt Enable Register (NMIC_IER)
- NMIC Interrupt Disable Register (NMIC_IDR)

If a write access to a write-protected register is detected, the WPVS flag in the NMIC Write Protection Status Register (NMIC_WPSR) is set and the field WVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the NMIC_WPSR.

If a write attempts to modify the settings of an interrupt line x that has been previously frozen by setting the NMIC_SCFGxR.FRZ bit, the FZWVS flag in the NMIC_WPSR is set and the field WVSRC indicates the register in which the write access has been attempted.

The FZWVS bit is automatically cleared after reading the NMIC_WPSR.

If a write attempts to modify the settings of the glitch filter of interrupt line x but the NMIC Source Configuration Register x (NMIC_SCFGxR) access is discarded due to an already ongoing glitch filter configuration, the FZWVS flag in the NMIC_WPSR is set and the field WVSRC indicates the register in which the write access has been attempted.

The FZWVS bit is automatically cleared after reading the NMIC_WPSR.

45.7 Register Summary

Offset	Name	Bit Pos.								
0x00	NMIC_IER	7:0	NMI7	NMI6	NMI5	NMI4	NMI3	NMI2	NMI1	NMI0
		15:8								NMI8
		23:16								
		31:24								
0x04	NMIC_IDR	7:0	NMI7	NMI6	NMI5	NMI4	NMI3	NMI2	NMI1	NMI0
		15:8								NMI8
		23:16								
		31:24								
0x08	NMIC_IAR	7:0	NMI7	NMI6	NMI5	NMI4	NMI3	NMI2	NMI1	NMI0
		15:8								NMI8
		23:16								
		31:24								
0x0C	NMIC_ISR	7:0	NMI7	NMI6	NMI5	NMI4	NMI3	NMI2	NMI1	NMI0
		15:8								NMI8
		23:16								
		31:24								
0x10	NMIC_GFCS	7:0	RDY7	RDY6	RDY5	RDY4	RDY3	RDY2	RDY1	RDY0
		15:8								RDY8
		23:16								
		31:24								
0x14	NMIC_SCFG0R	7:0				GFEN			GFSEL[1:0]	
		15:8							LVL	POL
		23:16								EN
		31:24	FRZ							
0x18	NMIC_SCFG1R	7:0				GFEN			GFSEL[1:0]	
		15:8							LVL	POL
		23:16								EN
		31:24	FRZ							
0x1C	NMIC_SCFG2R	7:0				GFEN			GFSEL[1:0]	
		15:8							LVL	POL
		23:16								EN
		31:24	FRZ							
0x20	NMIC_SCFG3R	7:0				GFEN			GFSEL[1:0]	
		15:8							LVL	POL
		23:16								EN
		31:24	FRZ							
0x24	NMIC_SCFG4R	7:0				GFEN			GFSEL[1:0]	
		15:8							LVL	POL
		23:16								EN
		31:24	FRZ							
0x28	NMIC_SCFG5R	7:0				GFEN			GFSEL[1:0]	
		15:8							LVL	POL
		23:16								EN
		31:24	FRZ							
0x2C	NMIC_SCFG6R	7:0				GFEN			GFSEL[1:0]	
		15:8							LVL	POL
		23:16								EN
		31:24	FRZ							
0x30	NMIC_SCFG7R	7:0				GFEN			GFSEL[1:0]	
		15:8							LVL	POL
		23:16								EN
		31:24	FRZ							
0x34 ... 0xE3	Reserved									

.....continued

Offset	Name	Bit Pos.								
0xE4	NMIC_WPMR	7:0							WPITEN	WPCFEN
		15:8	WPKEY[7:0]							
		23:16	WPKEY[15:8]							
		31:24	WPKEY[23:16]							
0xE8	NMIC_WPSR	7:0						BSWVS	FZWVS	WPVS
		15:8	WVSRC[7:0]							
		23:16								
		31:24								

45.7.1 NMIC Interrupt Enable Register

Name: NMIC_IER
Offset: 0x00
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the NMIC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								W
Reset								–

Bit	7	6	5	4	3	2	1	0
	NMI7	NMI6	NMI5	NMI4	NMI3	NMI2	NMI1	NMI0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8 – NMIx Non-maskable Interrupt x Enable

Value	Description
0	No effect
1	Enables the non-maskable interrupt x

45.7.2 NMIC Interrupt Disable Register

Name: NMIC_IDR
Offset: 0x04
Reset: –
Property: Write-only

This register can only be written if the WPITEN bit is cleared in the NMIC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
Access								W
Reset								–

Bit	7	6	5	4	3	2	1	0
	NMI7	NMI6	NMI5	NMI4	NMI3	NMI2	NMI1	NMI0
Access	W	W	W	W	W	W	W	W
Reset	–	–	–	–	–	–	–	–

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8 – NMIx Non-maskable Interrupt x Disable

Value	Description
0	No effect
1	Disables the non-maskable interrupt x

45.7.3 NMIC Interrupt Active Register

Name: NMIC_IAR
Offset: 0x08
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								NMI8
Access								R
Reset								0

Bit	7	6	5	4	3	2	1	0
	NMI7	NMI6	NMI5	NMI4	NMI3	NMI2	NMI1	NMI0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8 – NMIx Active Interrupt x

Value	Description
0	The interrupt source x is not enabled.
1	The interrupt source x is enabled.

45.7.4 NMIC Interrupt Status Register

Name: NMIC_ISR
Offset: 0x0C
Reset: 0x00000000
Property: Read-only

Note: The reset value is product-dependent.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								NMI8
Access								R
Reset								0

Bit	7	6	5	4	3	2	1	0
	NMI7	NMI6	NMI5	NMI4	NMI3	NMI2	NMI1	NMI0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8 – NMIx Non-maskable Interrupt Source x Pending (cleared on read)

Value	Description
0	No interrupt has been detected for source x since the last read of the NMIC_ISR.
1	An interrupt has been detected for source x since the last read of the NMIC_ISR.

45.7.5 NMIC Glitch Filter Configuration Status Register

Name: NMIC_GFCS
Offset: 0x10
Reset: 0x00000001
Property: Read-only

Note: The reset value is product-dependent.

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
								RDY8
Access								R
Reset								0

Bit	7	6	5	4	3	2	1	0
	RDY7	RDY6	RDY5	RDY4	RDY3	RDY2	RDY1	RDY0
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	1

Bits 0, 1, 2, 3, 4, 5, 6, 7, 8 – RDYx Filter x Configuration Ready

Value	Description
0	The interrupt line x Glitch Filter is not yet ready for use due to a previous write to the NMIC_SCFGxR, or the glitch filter is not implemented for this interrupt line. The Glitch Filter must not be reprogrammed in the NMIC_SCFGxR.
1	The interrupt line x Glitch Filter configuration is done and the Glitch Filter is active. The Glitch Filter can be programmed in the NMIC_SCFGxR.

45.7.6 NMIC Source Configuration Register 0

Name: NMIC_SCFG0R
Offset: 0x14
Reset: 0x00000313
Property: Read/Write

Note: The reset value is product-dependent.

This register can only be written if the WPCFEN bit is cleared in the NMIC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	FRZ							
Access								
Reset	0							

Bit	23	22	21	20	19	18	17	16
								EN
Access								
Reset								0

Bit	15	14	13	12	11	10	9	8
							LVL	POL
Access								
Reset							1	1

Bit	7	6	5	4	3	2	1	0
				GFEN			GFSEL[1:0]	
Access							R/W	R/W
Reset				1			1	1

Bit 31 – FRZ Interrupt Line Freeze

Value	Description
0	No effect.
1	The NMIC_SCFGxR value and the NMIC_IAR.NMIx bits are frozen until hardware reset.

Bit 16 – EN Source Enable

Value	Description
0	The NMI source x is disabled. Any source edge or level detection is discarded.
1	The NMI source x is enabled. Any valid source edge or level of polarity (POL) sets the NMIC_ISR.NMIx pending status flag.

Bit 9 – LVL Level Detection (LVL bit is read-only in NMIC_SCFG0 to 8)

Value	Description
0	The NMI source x interrupt status is set on a valid source edge.
1	The NMI source x interrupt status is set on a valid source level.

Bit 8 – POL Polarity (POL bit is read-only in NMIC_SCFG0 to 8)

Value	Description
0	The NMI source x is active low if LVL is set, or active on falling edge if LVL is cleared.
1	The NMI source x is active high if LVL is set, or active on rising edge if LVL is cleared.

Bit 4 – GFEN Glitch Filter Enable (GFEN bit is read-only in NMIC_SCFG0 to 8)

Value	Description
0	The Glitch Filter is disabled or not implemented for NMI source x. Any source x change is forwarded as is to the source detection logic.

Value	Description
1	The Glitch Filter is enabled for NMI source x. The NMI source x glitches are filtered according to the section Interrupt Line Characteristics.

Bits 1:0 – GFSEL[1:0] Glitch Filter Selector (GFSEL field is read-only in NMIC_SCFG0 to 8)
If GFEN is set, NMI source x glitches are filtered according to the section Interrupt Line Characteristics.

45.7.7 NMIC Source Configuration Register x

Name: NMIC_SCFGxR
Offset: 0x18 + (x-1)*0x04 [x=1..7]
Reset: 0x00000300
Property: Read/Write

Note: The reset value is product-dependent.

This register can only be written if the WPCFEN bit is cleared in the NMIC Write Protection Mode Register.

Bit	31	30	29	28	27	26	25	24
	FRZ							
Access								
Reset	0							

Bit	23	22	21	20	19	18	17	16
								EN
Access								
Reset								0

Bit	15	14	13	12	11	10	9	8
							LVL	POL
Access								
Reset							1	1

Bit	7	6	5	4	3	2	1	0
				GFEN			GFSEL[1:0]	
Access							R/W	R/W
Reset				0			0	0

Bit 31 – FRZ Interrupt Line Freeze

Value	Description
0	No effect.
1	The NMIC_SCFGxR value and the NMIC_IAR.NMIx bits are frozen until hardware reset.

Bit 16 – EN Source Enable

Value	Description
0	The NMI source x is disabled. Any source edge or level detection is discarded.
1	The NMI source x is enabled. Any valid source edge or level of polarity (POL) sets the NMIC_ISR.NMIx pending status flag.

Bit 9 – LVL Level Detection (LVL bit is read-only in NMIC_SCFG0 to 8)

Value	Description
0	The NMI source x interrupt status is set on a valid source edge.
1	The NMI source x interrupt status is set on a valid source level.

Bit 8 – POL Polarity (POL bit is read-only in NMIC_SCFG0 to 8)

Value	Description
0	The NMI source x is active low if LVL is set, or active on falling edge if LVL is cleared.
1	The NMI source x is active high if LVL is set, or active on rising edge if LVL is cleared.

Bit 4 – GFEN Glitch Filter Enable (GFEN bit is read-only in NMIC_SCFG0 to 8)

Value	Description
0	The Glitch Filter is disabled or not implemented for NMI source x. Any source x change is forwarded as is to the source detection logic.

Value	Description
1	The Glitch Filter is enabled for NMI source x. The NMI source x glitches are filtered according to the section Interrupt Line Characteristics.

Bits 1:0 – GFSEL[1:0] Glitch Filter Selector (GFSEL field is read-only in NMIC_SCFG0 to 8)
If GFEN is set, NMI source x glitches are filtered according to the section Interrupt Line Characteristics.

45.7.8 NMIC Write Protection Mode Register

Name: NMIC_WPMR
Offset: 0xE4
Reset: 0x00000000
Property: Read/Write

Bit	31	30	29	28	27	26	25	24
	WPKEY[23:16]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	23	22	21	20	19	18	17	16
	WPKEY[15:8]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8
	WPKEY[7:0]							
Access	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
							WPITEN	WPCFEN
Access							R/W	R/W
Reset							0	0

Bits 31:8 – WPKEY[23:0] Write Protection Key

Value	Name	Description
0x4E4D49	PASSWD	Writing any other value in this field aborts the write operation of the WPCFEN and WPITEN bit. Always reads as 0.

Bit 1 – WPITEN Write Protection Interrupt Enable

See Register Write Protection for the list of registers that can be protected.

Value	Description
0	Disables the write protection of NMIC_IER and NMIC_IDR if WPKEY corresponds to 0x4E4D49 (“NMI” in ASCII).
1	Enables the write protection of NMIC_IER and NMIC_IDR if WPKEY corresponds to 0x4E4D49 (“NMI” in ASCII).

Bit 0 – WPCFEN Write Protection Configuration Enable

Value	Description
0	Disables the write protection of the configuration registers if WPKEY corresponds to 0x4E4D49 (“NMI” in ASCII).
1	Enables the write protection of the configuration registers if WPKEY corresponds to 0x4E4D49 (“NMI” in ASCII).

45.7.9 NMIC Write Protection Status Register

Name: NMIC_WPSR
Offset: 0xE8
Reset: 0x00000000
Property: Read-only

Bit	31	30	29	28	27	26	25	24
Access								
Reset								

Bit	23	22	21	20	19	18	17	16
Access								
Reset								

Bit	15	14	13	12	11	10	9	8
	WVSRC[7:0]							
Access	R	R	R	R	R	R	R	R
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
						BSWVS	FZWVS	WPVS
Access						R	R	R
Reset						0	0	0

Bits 15:8 – WVSRC[7:0] Write Violation Source
When one of the WPVS, FZWVS, BSWVS is equal to 1, WVSRC indicates the register address offset at which a write access has been attempted.

Bit 2 – BSWVS Busy Register Write Violation Status

Value	Description
0	No write access violation of a busy register has occurred since the last read of the NMIC_WPSR.
1	A write access violation of a busy register has occurred since the last read of the NMIC_WPSR. The associated violation is reported into field WVSRC.

Bit 1 – FZWVS Frozen Register Write Violation Status

Value	Description
0	No write access violation of a frozen register has occurred since the last read of the NMIC_WPSR.
1	A write access violation of a frozen register has occurred since the last read of the NMIC_WPSR. The associated violation is reported into field WVSRC.

Bit 0 – WPVS Write Protection Register Violation Status

Value	Description
0	No write protection violation has occurred since the last read of the NMIC_WPSR.
1	A write protection violation has occurred since the last read of the NMIC_WPSR. The associated violation is reported into field WVSRC.

46. Mechanical Characteristics

For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging.

Figure 46-1. 256-pin CQFP Package

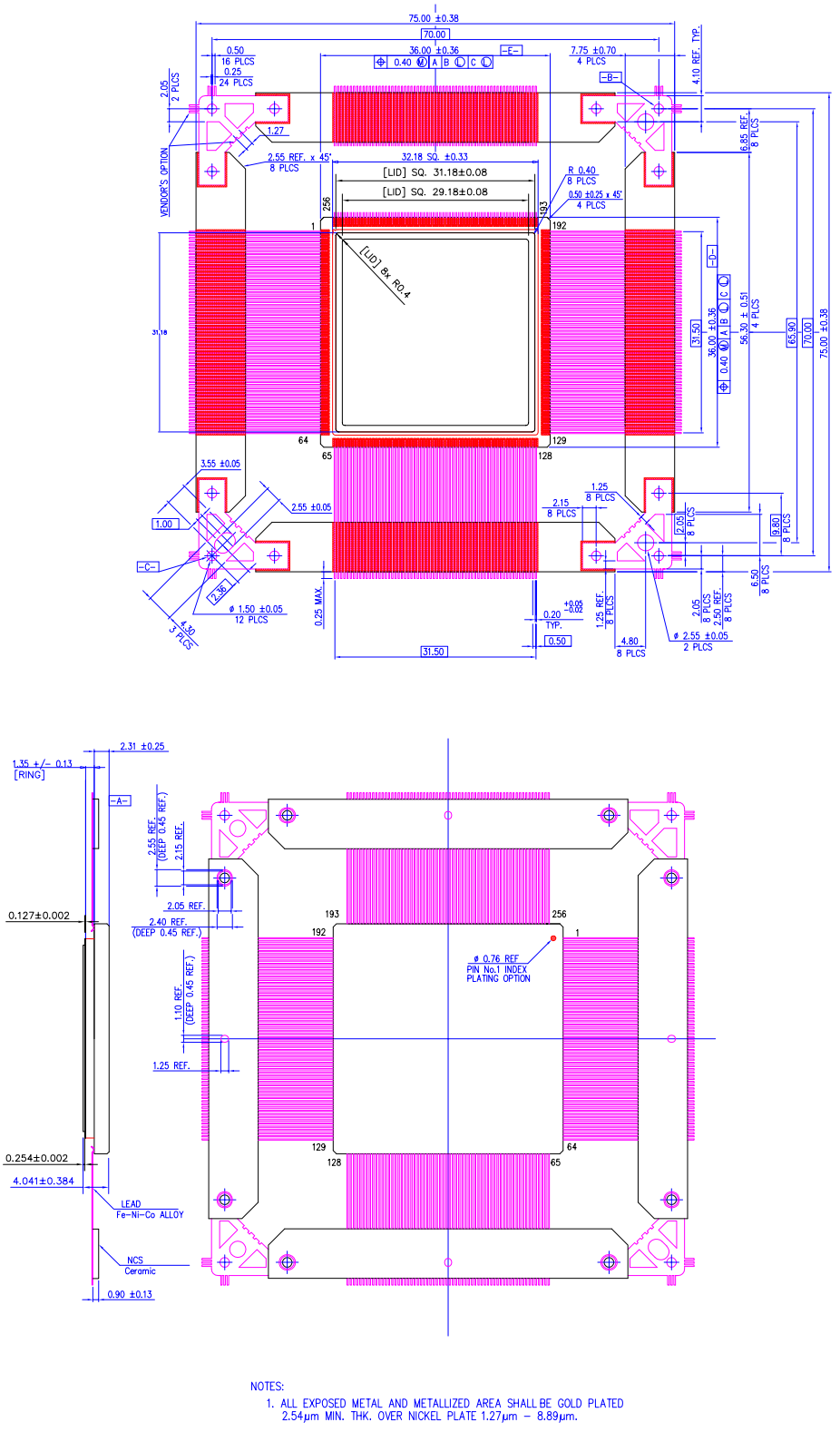


Table 46-1. Device and CQFP Package Maximum Weight

18	g

Table 46-2. CQFP Package Reference

JEDEC Drawing Reference	MO-134
J-STD-609 Classification	e4

47. Package Marking Information

All devices are marked with a company logo and the ordering code.

Additional marking is as follows:

where,

- “YY”: Manufactory year
- “WW”: Manufactory week
- “V”: Revision
- “C”: Assembly country code (optional)
- “XXXXXXX”: Lot number

48. Ordering Information

Table 48-1. SAMRH71 Ordering Information

Ordering Code	Package	Carrier Type	Operating Temperature Range
SAMRH71F20C-7GB-E	CQFP256	--	+25°C
SAMRH71F20C-7GB-MQ	CQFP256	--	-55°C to +125°C
SAMRH71F20C-7GB-SV	CQFP256	--	-55°C to +125°C

49. Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative, or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the last page of this document.

Technical support is available through the Microchip website at: aerospace@nto.atmel.com.

50. Revision History

50.1 Revision A - October 2019

This is the initial released version of this document.

The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Customer Change Notification Service

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at http://www.microchip.com/. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-5170-9

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit http://www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: www.microchip.com	**Australia - Sydney** Tel: 61-2-9868-6733	**India - Bangalore** Tel: 91-80-3090-4444	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393
Atlanta Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455	**China - Beijing** Tel: 86-10-8569-7000	**India - New Delhi** Tel: 91-11-4160-8631	**Denmark - Copenhagen** Tel: 45-4450-2828 Fax: 45-4485-2829
Austin, TX Tel: 512-257-3370	**China - Chengdu** Tel: 86-28-8665-5511	**India - Pune** Tel: 91-20-4121-0141	**Finland - Espoo** Tel: 358-9-4520-820
Boston Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088	**China - Chongqing** Tel: 86-23-8980-9588	**Japan - Osaka** Tel: 81-6-6152-7160	**France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79
Chicago Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075	**China - Dongguan** Tel: 86-769-8702-9880	**Japan - Tokyo** Tel: 81-3-6880- 3770	**Germany - Garching** Tel: 49-8931-9700
Dallas Addison, TX Tel: 972-818-7423 Fax: 972-818-2924	**China - Guangzhou** Tel: 86-20-8755-8029	**Korea - Daegu** Tel: 82-53-744-4301	**Germany - Haan** Tel: 49-2129-3766400
Detroit Novi, MI Tel: 248-848-4000	**China - Hangzhou** Tel: 86-571-8792-8115	**Korea - Seoul** Tel: 82-2-554-7200	**Germany - Heilbronn** Tel: 49-7131-67-3636
Houston, TX Tel: 281-894-5983	**China - Hong Kong SAR** Tel: 852-2943-5100	**Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906	**Germany - Karlsruhe** Tel: 49-721-625370
Indianapolis Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380	**China - Nanjing** Tel: 86-25-8473-2460	**Malaysia - Penang** Tel: 60-4-227-8870	**Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44
Los Angeles Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800	**China - Qingdao** Tel: 86-532-8502-7355	**Philippines - Manila** Tel: 63-2-634-9065	**Germany - Rosenheim** Tel: 49-8031-354-560
Raleigh, NC Tel: 919-844-7510	**China - Shanghai** Tel: 86-21-3326-8000	**Singapore** Tel: 65-6334-8870	**Israel - Ra'anana** Tel: 972-9-744-7705
New York, NY Tel: 631-435-6000	**China - Shenyang** Tel: 86-24-2334-2829	**Taiwan - Hsin Chu** Tel: 886-3-577-8366	**Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781
San Jose, CA Tel: 408-735-9110 Tel: 408-436-4270	**China - Shenzhen** Tel: 86-755-8864-2200	**Taiwan - Kaohsiung** Tel: 886-7-213-7830	**Italy - Padova** Tel: 39-049-7625286
Canada - Toronto Tel: 905-695-1980 Fax: 905-695-2078	**China - Suzhou** Tel: 86-186-6233-1526	**Taiwan - Taipei** Tel: 886-2-2508-8600	**Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340
	China - Wuhan Tel: 86-27-5980-5300	**Thailand - Bangkok** Tel: 66-2-694-1351	**Norway - Trondheim** Tel: 47-7289-7561
	China - Xian Tel: 86-29-8833-7252	**Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Poland - Warsaw** Tel: 48-22-3325737
	China - Xiamen Tel: 86-592-2388138		**Romania - Bucharest** Tel: 40-21-407-87-50
	China - Zhuhai Tel: 86-756-3210040		**Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91
			Sweden - Gothenberg Tel: 46-31-704-60-40
			Sweden - Stockholm Tel: 46-8-5090-4654
			UK - Wokingham Tel: 44-118-921-5800 Fax: 44-118-921-5820