SAMA5D41 / SAMA5D42
SAMA5D43 / SAMA5D44

Atmel | SMART ARM-based Embedded MPU

DATASHEET

Description

The Atmel® | SMART SAMA5D4 Series is a high-performance, power-efficient ARM® Cortex®-A5 processor MPU capable of running up to 600 MHz. It integrates the ARM NEON™ SIMD engine for accelerated signal processing, multimedia and graphics as well as a 128 KB L2-Cache for high system performance. The device features the ARM TrustZone® enabling a strong security perimeter for critical software, as well as several hardware security features. The device also features advanced user interface and connectivity peripherals.

The SAMA5D4 devices have three software-selectable low-power modes: Idle, Ultra-low-power, and Backup. In Idle mode, the processor is stopped while all other functions can be kept running at normal operating bus frequency. In Ultra-low-power mode, the processor is stopped while all other functions can be kept running at minimum operating bus frequency. In Backup mode, only the real-time clock, real-time timer, backup SRAM, backup registers, and wakeup logic are running.

The SAMA5D4 features an internal multi-layer bus architecture associated with 32 DMA channels to sustain the high bandwidth required by the processor and the high-speed peripherals. The device supports DDR2/LPDDR/LPDDR2 and SLC/MLC NAND Flash memory with 24-bit ECC.

The comprehensive peripheral set includes a 720p hardware video decoder, an LCD controller with overlays for hardware-accelerated image composition, a resistive touchscreen function, and a CMOS sensor interface. Connectivity peripherals include a dual 10/100 Ethernet MAC with IEEE1588, three HS USB ports, UARTs, SPIs and I2Cs.

Security features includes an “on-the-fly” encryption-decryption process from the external DDR memory, tamper detection pins, secure storage of critical data, an integrity check monitor (ICM) to detect modification of the memory contents and a secure boot. The product also includes a dedicated coprocessor for public key cryptography such as RSA and elliptic curves algorithms (ECC), as well as AES, 3DES, SHA function and TRNG. These features permit to protect the system against counterfeiting, to safeguard sensitive data, authenticate safe program or secure external data transfers.

The SAMA5D4 series is optimized for control panel/HMI applications needing video playback and applications that require high levels of connectivity in the industrial and consumer market. Its security features make the SAMA5D4 well suited for secure gateways or for the IoT.

Features

- ARM Cortex-A5 Core
  - ARMv7-A Thumb2® instruction set
  - ARM TrustZone
  - NEON Media Processing Engine
  - 945 MIPS @ 600 MHz in worst conditions
- Memory Architecture
  - Memory Management Unit
  - 32 Kbyte Data Cache, 32 Kbyte Instruction Cache
  - 128 Kbyte L2 Cache
  - One 128 Kbyte scrambled internal ROM single-cycle access at system speed, embedding Atmel boot loader/Atmel Secure boot loader
  - One 128 Kbyte scrambled internal SRAM, single-cycle access at system speed
  - High-bandwidth scramblable 16-bit or 32-bit Double Data Rate Multi-port Dynamic RAM Controller supporting 512 Mbyte 8-bank DDR2/LPDDR/LPDDR2, including partial areas “on-the-fly” AES encryption/decryption
  - EBI (External Bus interface) supporting:
    - 16-bit NAND Flash controller, including 24-bit error correction code (PMECC) for 8-bit NAND Flash
    - Independent Static Memory Controller (SMC) with datapath scrambling
- System running up to 200 MHz in worst conditions
  - Power-on Reset Cells, Reset Controller, Shutdown Controller, Periodic Interval Timer, Watchdog Timer and secure Real-time Clock
  - Internal regulator
  - One 600–1200 MHz PLL for the System and one PLL at 480 MHz optimized for USB High Speed
  - Internal Low-power 12 MHz RC Oscillator
  - Low-power 32 kHz RC Oscillator
  - Selectable 32.768 KHz Low-power oscillator and 12 MHz Oscillator
  - Two 64-bit, 16-channel DMA Controllers
  - 64-bit Advanced Interrupt Controller
  - 64-bit Secure Advanced Interrupt Controller
  - Three Programmable External Clock Signals
  - Programmable fuse box with 512 fuse bits available for customer, including JTAG protection
- Three Low-power Modes: Idle, Ultra-low-power, and Backup
- Peripherals
  - Video Decoder (VDEC) supporting formats MPEG-4, H.264, H.263, VP8 and JPEG, and image postprocessing
  - LCD TFT Controller with 4 overlays up to 2048x2048 or up to 720p in video format, with rotation and alpha blending
  - ITU-R BT. 601/656 Image Sensor Interface (ISI)
  - One High-Speed USB Device, Three High-Speed USB Host with On-chip Transceiver
  - Two 10/100 Mbps Ethernet MAC Controllers with IEEE 1588 v2 support
  - Software Modem Interface (SMD)
  - Two high-speed memory card hosts (eMMC 4.3 and SD 2.0)
  - Three Master/Slave Serial Peripheral Interfaces (SPI)
  - Five USARTs, two UARTs, one DBGU
  - Two Synchronous Serial Controllers (SSC)
  - Four Two-wire Interfaces up to 400 Kbits/s supporting I2C protocol and SMBUS (TWI)
  - Three 3-channel 32-bit Timer/Counters (TC)
  - One 4-channel 16-bit PWM Controller

  - One 5-channel 10-bit Analog-to-Digital Converter with Resistive Touchscreen function
- Safety
  - Internal and external memory integrity monitoring, with Integrity Check Monitor (ICM) based on SHA256
  - Power-on Reset Cells
  - Main Crystal Clock Failure Detector
  - Independent Watchdog
  - Register Write Protection
  - Memory Management Unit
- Security
  - 512 bits of scrambled and erasable registers (1)
  - 8 Kbytes of internal scrambled RAM with non-imprinting support, 6 Kbytes are erasable (1)
  - 8 PIOBU tamper pins for static or dynamic intrusion detections (1)
  - Atmel secure boot (2)
- Cryptography
  - True Random Number Generator (TRNG), compliant with NIST special publication 800-22 test suite and FIPS PBUs 140-2 and 140-3
  - SHA: Supports Secure Hash Algorithm (SHA1, SHA224, SHA256, SHA384, SHA512) compliant with FIPS publications 180-2
  - AES: 256-bit, 192-bit, 128-bit Key Algorithm, compliant with FIPS PUB 197 specifications
  - Advanced Encryption Standard Bridge (AESB): AES 128 that includes Automatic Bridge Mode for automatic DDR port Encryption/Decryption
  - TDES: Two-key or Three-key Algorithms, compliant with FIPS PUB 46-3 specifications
  - Public Key Coprocessor (CPKCC) and associated Classical Public Key Cryptography Library (CPKCL) for RSA, DSA, ECC GF(2^n), ECC GF(p) (3)
- Up to 152 I/Os
  - Five Parallel Input/Output Controllers with slew rate control on high-speed I/Os
  - Input Change Interrupt capability on each I/O Line, selectable Schmitt Trigger input
  - Individually Programmable Open-drain, Pull-up and Pull-down Resistor, Synchronous Output, Filtering
- Packages
  - 361-ball stubless TFBGA, 16x16 mm body, pitch 0.8 mm
  - 289-ball stubless LFBGA, 14x14 mm body, pitch 0.8 mm

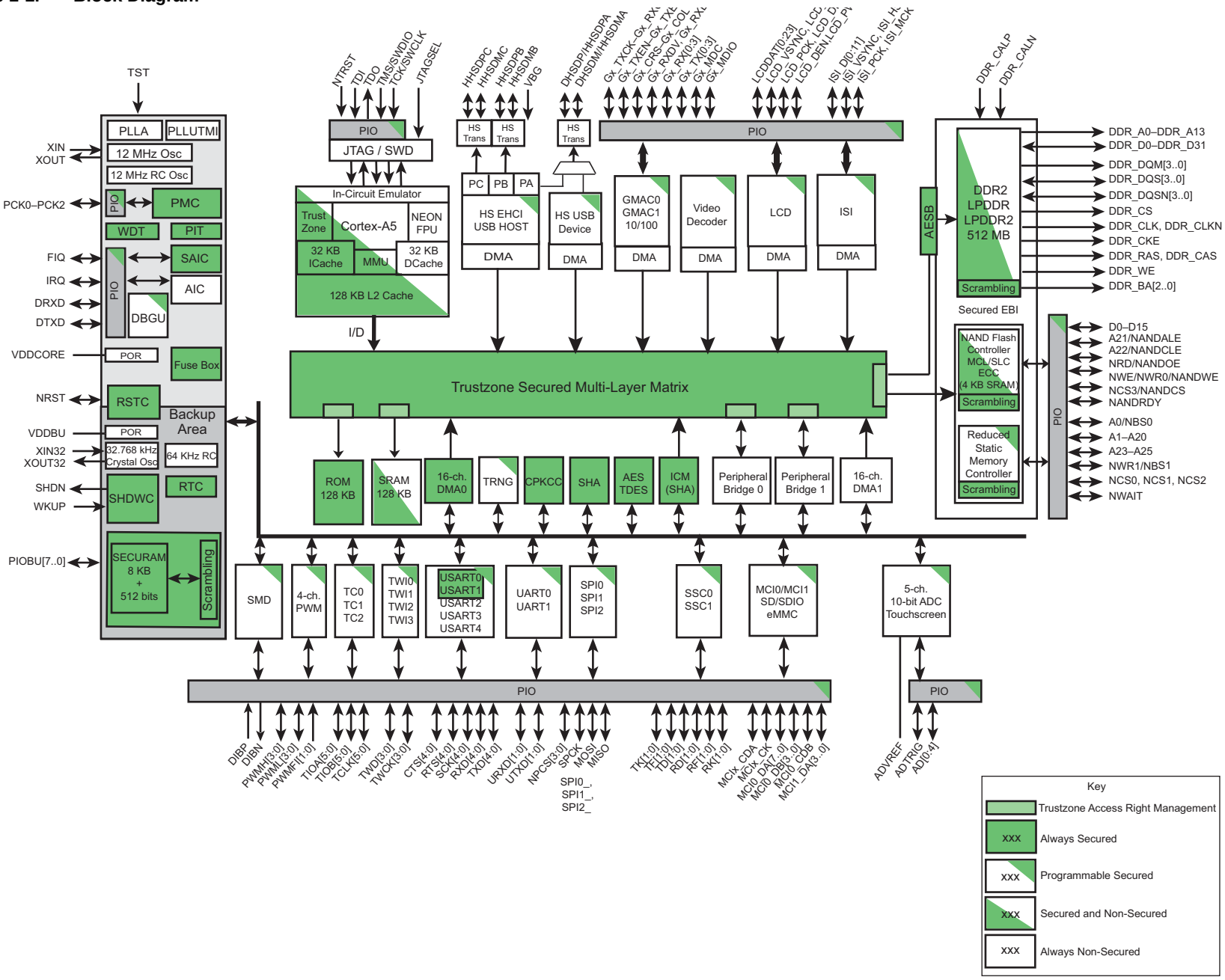
SAMA5D4 Series Devices

Device	Package	Video Decoder	DDR Datapath
SAMA5D44	TFBGA361	X	32 bits
SAMA5D43	LFBGA289	X	16 bits
SAMA5D42	TFBGA361	–	32 bits
SAMA5D41	LFBGA289	–	16 bits

1. This feature is described in the document “Secure Box Module (SBM)”, Atmel literature No. 11254. This document is available under Non-Disclosure Agreement (NDA). Contact an Atmel Sales Representative for further details.
2. For secure boot strategies, refer to the application note “SAMA5D4x Secure Boot Strategy”, Atmel literature No. 11295. This document is available under Non-Disclosure Agreement (NDA). Contact an Atmel Sales Representative for further details.
3. CPKCC and CPKCL are described in the application note “Using CPKCL Version 02.05.01.xx on SAMA5D4”, Atmel literature No. 11214. This document is available under Non-Disclosure Agreement (NDA). Contact an Atmel Sales Representative for further details.

1. Block Diagram

Figure 1-1. Block Diagram



2. Signal Description

Table 2-1 gives details on signal names classified by peripheral.

Table 2-1. Signal Description List

Signal Name	Function	Type	Active Level
Clocks, Oscillators and PLLs			
XIN	Main Oscillator Input	Input	–
XOUT	Main Oscillator Output	Output	–
XIN32	Slow Clock Oscillator Input	Input	–
XOUT32	Slow Clock Oscillator Output	Output	–
VBG	Bias Voltage Reference for USB	Analog	–
PCK0–PCK2	Programmable Clock Output	Output	–
Shutdown, Wakeup Logic			
SHDN	Shutdown Control	Output	–
WKUP	Wakeup Input	Input	–
ICE and JTAG			
TCK/SWCLK	Test Clock/Serial Wire Clock	Input	–
TDI	Test Data In	Input	–
TDO	Test Data Out	Output	–
TMS/SWDIO	Test Mode Select/Serial Wire Input/Output	I/O	–
JTAGSEL	JTAG Selection	Input	–
Reset/Test			
NRST	Microprocessor Reset	Input	Low
TST	Test Mode Select	Input	–
NTRST	Test Reset Signal	Input	–
Debug Unit - DBGU			
DRXD	Debug Receive Data	Input	–
DTXD	Debug Transmit Data	Output	–
Advanced Interrupt Controller - AIC			
IRQ	External Interrupt Input	Input	–
Secured Advanced Interrupt Controller - SAIC			
FIQ	Fast Interrupt Input	Input	–
PIO Controller - PIOA - PIOB - PIOC - PIOD - PIOE			
PA0–PAxx	Parallel IO Controller A	I/O	–
PB0–PBxx	Parallel IO Controller B	I/O	–
PC0–PCxx	Parallel IO Controller C	I/O	–
PD8–PDxx	Parallel IO Controller D	I/O	–
PE0–PExx	Parallel IO Controller E	I/O	–

Table 2-1. Signal Description List (Continued)

Signal Name	Function	Type	Active Level
External Bus Interface - EBI			
D0–D15	Data Bus	I/O	–
A0–A25	Address Bus	Output	–
NWAIT	External Wait Signal	Input	Low
Static Memory Controller - SMC			
NCS0–NCS3	Chip Select Lines	Output	Low
NWR0–NWR1	Write Signal	Output	Low
NRD	Read Signal	Output	Low
NWE	Write Enable	Output	Low
NBS0–NBS1	Byte Mask Signal	Output	Low
NANDOE	NAND Flash Output Enable	Output	Low
NANDWE	NAND Flash Write Enable	Output	Low
DDR2/LPDDR2 Controller			
DDR_CK,DDR_CKN	DDR2 Differential Clock	Output	–
DDR_CKE	DDR2 Clock Enable	Output	High
DDR_CS	DDR2 Controller Chip Select	Output	Low
DDR_BA[2..0]	Bank Select	Output	Low
DDR_WE	DDR2 Write Enable	Output	Low
DDR_RAS - DDR_CAS	Row and Column Signal	Output	Low
DDR_A[13..0]	DDR2 Address Bus	Output	–
DDR_D[31..0]	DDR2 Data Bus	I/O/-PD	–
DDR_DQS[3..0], DDR_DQSN[3..0]	Differential Data Strobe	I/O-PD	–
DDR_DQM[3..0]	Write Data Mask	Output	–
DDR_CALP, DDR_CALN	DDR2/LPDDR2 Calibration	Input	–
DDR_VREF	DDR2/LPDDR2 Reference	Input	–
High Speed Multimedia Card Interface - HSMCIx [1..0]			
MCI0_CK, MCI1_CK	Multimedia Card Clock	I/O	–
MCI0_CDA, MCI0_CDB, MCI1_CDA	Multimedia Card Command	I/O	–
MCI0_DA[7..0]	Multimedia Card 0 Data slot A	I/O	–
MCI0_DB[3..0]	Multimedia Card 0 Data slot B	I/O	–
MCI1_DA[3..0]	Multimedia Card 1 Data	I/O	–
Universal Synchronous Asynchronous Receiver Transmitter - USARTx [4..0]			
SCKx	USARTx Serial Clock	I/O	–
TXDx	USARTx Transmit Data	Output	–
RXDx	USARTx Receive Data	Input	–
RTSx	USARTx Request To Send	Output	–
CTSx	USARTx Clear To Send	Input	–

Table 2-1. Signal Description List (Continued)

Signal Name	Function	Type	Active Level
Universal Asynchronous Receiver Transmitter - UARTx [1..0]			
UTXDx	UARTx Transmit Data	Output	–
URXDx	UARTx Receive Data	Input	–
Synchronous Serial Controller - SSCx [1..0]			
TDx	SSC Transmit Data	Output	–
RDx	SSC Receive Data	Input	–
TKx	SSC Transmit Clock	I/O	–
RKx	SSC Receive Clock	I/O	–
TFx	SSC Transmit Frame Sync	I/O	–
RFx	SSC Receive Frame Sync	I/O	–
Timer/Counter - TCx [8..0]			
TCLKx	TC Channel x External Clock Input	Input	–
TIOAx	TC Channel x I/O Line A	I/O	–
TIOBx	TC Channel x I/O Line B	I/O	–
Serial Peripheral Interface - SPIx [2..0]			
SPIx_MISO	Master In Slave Out	I/O	–
SPIx_MOSI	Master Out Slave In	I/O	–
SPIx_SPCK	SPI Serial Clock	I/O	–
SPIx_NPCS0	SPI Peripheral Chip Select 0	I/O	Low
SPIx_NPCS[3..1]	SPI Peripheral Chip Select	Output	Low
Two-wire Interface - TWIx [3..0]			
TWDx	Two-wire Serial Data	I/O	–
TWCKx	Two-wire Serial Clock	I/O	–
Pulse Width Modulation Controller - PWM			
PWMH0–3	PWM Waveform Output High	–	Output
PWML0–3	PWM Waveform Output Low	–	Output
PWMFI0–1	PWM Fault Inputs	–	Input
USB Host High Speed Port - UHPHS			
HHSDPA	USB Host Port A High Speed Data +	Analog	–
HHSDMA	USB Host Port A High Speed Data -	Analog	–
HHSDPB	USB Host Port B High Speed Data +	Analog	–
HHSDMB	USB Host Port B High Speed Data -	Analog	–
HHSDPC	USB Host Port C High Speed Data +	Analog	–
HHSDMC	USB Host Port C High Speed Data -	Analog	–
USB Device High Speed Port - UDPHS			
DHSDP	USB Device High Speed Data +	Analog	–
DHSDM	USB Device High Speed Data -	Analog	–

Table 2-1. Signal Description List (Continued)

Signal Name	Function	Type	Active Level
Ethernet 10/100 - GMACx [1..0]			
GxTXCK	Transmit Clock or Reference Clock	Input	–
GxRXCK	Receive Clock	Input	–
GxTXEN	Transmit Enable	Output	–
GxTX0–3	Transmit Data	Output	–
GxTXER	Transmit Coding Error	Output	–
GxRXDV	Receive Data Valid	Input	–
GxRX0–3	Receive Data	Input	–
GxRXER	Receive Error	Input	–
GxCRS	Carrier Sense and Data Valid	Input	–
GxCOL	Collision Detect	Input	–
GxMDC	Management Data Clock	Output	–
GxMDIO	Management Data Input/Output	I/O	–
LCD Controller - LCDC			
LCDDAT0–23	LCD Data Bus	Output	–
LCDVSYNC	LCD Vertical Synchronization	Output	–
LCDHSYNC	LCD Horizontal Synchronization	Output	–
LCDPCK	LCD Pixel Clock	Output	–
LCDDEN	LCD Data Enable	Output	–
LCDPWM	LCDPWM for Contrast Control	Output	–
LCDDISP	LCD Display ON/OFF	Output	–
Touchscreen Analog-to-Digital Converter - ADC			
AD0–4	4 Analog Inputs	Analog	–
ADTRG	ADC Trigger	Input	–
ADVREF	ADC Reference	Analog	–
Secure Box Module - SBM			
PIOBU0–7	Secured I/Os	I/O	–
Image Sensor Interface - ISI			
ISI_D0–ISI_D11	Image Sensor Data	Input	–
ISI_HSYNC	Image Sensor Horizontal Synchro	Input	–
ISI_VSYNC	Image Sensor Vertical Synchro	Input	–
ISI_PCK	Image Sensor Data clock	Input	–
Software Modem Device - SMD			
DIBN	Software Modem Signal	I/O	–
DIBP	Software Modem Signal	I/O	–

3. Package and Pinout

The SAMA5D4 product is available in two packages:

- 361-ball TFBGA
- 289-ball LFBGA

The pinouts are provided in the following Section 3.1 “361-ball TFBGA Package Pinout” and Section 3.2 “289-ball LFBGA Package Pinout”.

The package mechanical characteristics are described in Section 56. “Mechanical Characteristics”.

3.1 361-ball TFBGA Package Pinout

Table 3-1. TFBGA361 Pin Description

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
A7	VDDIOP	GPIO	PA0	I/O	–	–	LCDDAT0	O	–	–	TMS	I	TMS, PU, ST
F6	VDDIOP	GPIO	PA1	I/O	–	–	LCDDAT1	O	–	–	–	–	PIO, I, PU, ST
E6	VDDIOP	GPIO_CLK	PA2	I/O	–	–	LCDDAT2	O	G1_TXCK	I	–	–	PIO, I, PU, ST
C6	VDDIOP	GPIO_CLK	PA3	I/O	–	–	LCDDAT3	O	G1_RXCK	I	–	–	PIO, I, PU, ST
D6	VDDIOP	GPIO	PA4	I/O	–	–	LCDDAT4	O	G1_TXEN	O	–	–	PIO, I, PU, ST
B6	VDDIOP	GPIO	PA5	I/O	–	–	LCDDAT5	O	G1_TXER	O	–	–	PIO, I, PU, ST
A6	VDDIOP	GPIO	PA6	I/O	–	–	LCDDAT6	O	G1_CRS	I	–	–	PIO, I, PU, ST
E5	VDDIOP	GPIO	PA7	I/O	–	–	LCDDAT7	O	–	–	–	–	PIO, I, PU, ST
A5	VDDIOP	GPIO	PA8	I/O	–	–	LCDDAT8	O	–	–	TCK	I	TCK, PU
F4	VDDIOP	GPIO	PA9	I/O	–	–	LCDDAT9	O	G1_COL	I	–	–	PIO, I, PU, ST
F5	VDDIOP	GPIO	PA10	I/O	–	–	LCDDAT10	O	G1_RXDV	I	–	–	PIO, I, PU, ST
D5	VDDIOP	GPIO	PA11	I/O	–	–	LCDDAT11	O	G1_RXER	I	–	–	PIO, I, PU, ST
G5	VDDIOP	GPIO	PA12	I/O	–	–	LCDDAT12	O	G1_RX0	I	–	–	PIO, I, PU, ST
C5	VDDIOP	GPIO	PA13	I/O	–	–	LCDDAT13	O	G1_RX1	I	–	–	PIO, I, PU, ST
E4	VDDIOP	GPIO	PA14	I/O	–	–	LCDDAT14	O	G1_TX0	O	–	–	PIO, I, PU, ST
B5	VDDIOP	GPIO	PA15	I/O	–	–	LCDDAT15	O	G1_TX1	O	–	–	PIO, I, PU, ST
H6	VDDIOP	GPIO	PA16	I/O	–	–	LCDDAT16	O	–	–	NTRST	I	NTRST, PU, ST
D4	VDDIOP	GPIO	PA17	I/O	–	–	LCDDAT17	O	–	–	–	–	PIO, O, LOW
G4	VDDIOP	GPIO	PA18	I/O	–	–	LCDDAT18	O	G1_RX2	I	–	–	PIO, O, LOW
C4	VDDIOP	GPIO	PA19	I/O	–	–	LCDDAT19	O	G1_RX3	I	–	–	PIO, O, LOW
A3	VDDIOP	GPIO	PA20	I/O	–	–	LCDDAT20	O	G1_TX2	O	–	–	PIO, I, PU, ST
B4	VDDIOP	GPIO	PA21	I/O	–	–	LCDDAT21	O	G1_TX3	O	–	–	PIO, I, PU, ST
B3	VDDIOP	GPIO	PA22	I/O	–	–	LCDDAT22	O	G1_MDC	O	–	–	PIO, I, PU, ST
A4	VDDIOP	GPIO	PA23	I/O	–	–	LCDDAT23	O	G1_MDIO	I/O	–	–	PIO, I, PU, ST
H5	VDDIOP	GPIO_CLK	PA24	I/O	–	–	LCDPWM	O	PCK0	O	–	–	PIO, I, PU, ST
F3	VDDIOP	GPIO	PA25	I/O	–	–	LCDDISP	O	TD0	O	–	–	PIO, I, PU, ST
E3	VDDIOP	GPIO	PA26	I/O	–	–	LCDVSYNC	O	PWMH0	O	SPI1_NPCS1	O	PIO, I, PU, ST
H4	VDDIOP	GPIO	PA27	I/O	–	–	LCDHSYNC	O	PWML0	O	SPI1_NPCS2	O	PIO, I, PU, ST
G3	VDDIOP	GPIO_CLK2	PA28	I/O	–	–	LCDPCK	O	PWMH1	O	SPI1_NPCS3	O	PIO, I, PU, ST
J5	VDDIOP	GPIO	PA29	I/O	–	–	LCDDEN	O	PWML1	O	–	–	PIO, I, PU, ST
D3	VDDIOP	GPIO	PA30	I/O	–	–	TWD0	I/O	–	–	–	–	PIO, I, PU, ST
J4	VDDIOP	GPIO	PA31	I/O	–	–	TWCK0	O	–	–	–	–	PIO, I, PU, ST
C3	VDDIOP	GPIO_CLK	PB0	I/O	–	–	G0_TXCK	I	–	–	–	–	PIO, I, PU, ST
A2	VDDIOP	GPIO_CLK	PB1	I/O	–	–	G0_RXCK	I	SCK2	I/O	ISI_PCK	I	PIO, I, PU, ST
B2	VDDIOP	GPIO	PB2	I/O	–	–	G0_TXEN	O	–	–	–	–	PIO, I, PU, ST
C2	VDDIOP	GPIO	PB3	I/O	–	–	G0_TXER	O	CTS2	I	ISI_VSYNC	I	PIO, I, PU, ST

Table 3-1. TFBGA361 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
J3	VDDIOP	GPIO	PB4	I/O	–	–	G0_CRS	I	RXD2	I	ISI_HSYNC	I	PIO, I, PU, ST
H2	VDDIOP	GPIO	PB5	I/O	–	–	G0_COL	I	TXD2	O	PCK2	O	PIO, I, PU, ST
G2	VDDIOP	GPIO	PB6	I/O	–	–	G0_RXDV	I	–	–	–	–	PIO, I, PU, ST
H3	VDDIOP	GPIO	PB7	I/O	–	–	G0_RXER	I	–	–	–	–	PIO, I, PU, ST
F2	VDDIOP	GPIO	PB8	I/O	–	–	G0_RX0	I	–	–	–	–	PIO, I, PU, ST
J2	VDDIOP	GPIO	PB9	I/O	–	–	G0_RX1	I	–	–	–	–	PIO, I, PU, ST
F1	VDDIOP	GPIO_CLK	PB10	I/O	–	–	G0_RX2	I	PCK2	O	PWML1	O	PIO, I, PU, ST
K4	VDDIOP	GPIO	PB11	I/O	–	–	G0_RX3	I	RTS2	O	PWMH1	O	PIO, I, PU, ST
D2	VDDIOP	GPIO	PB12	I/O	–	–	G0_TX0	O	–	–	–	–	PIO, I, PU, ST
K3	VDDIOP	GPIO	PB13	I/O	–	–	G0_TX1	O	–	–	–	–	PIO, I, PU, ST
A1	VDDIOP	GPIO	PB14	I/O	–	–	G0_TX2	O	SPI2_NPCS1	O	PWMH0	O	PIO, I, PU, ST
E2	VDDIOP	GPIO	PB15	I/O	–	–	G0_TX3	O	SPI2_NPCS2	O	PWML0	O	PIO, I, PU, ST
B1	VDDIOP	GPIO	PB16	I/O	–	–	G0_MDC	O	–	–	–	–	PIO, I, PU, ST
K5	VDDIOP	GPIO	PB17	I/O	–	–	G0_MDIO	I/O	–	–	–	–	PIO, I, PU, ST
K2	VDDIOP	GPIO	PB18	I/O	–	–	SPI1_MISO	I/O	D8	I/O	–	–	PIO, I, PU, ST
C1	VDDIOP	GPIO	PB19	I/O	–	–	SPI1_MOSI	I/O	D9	I/O	–	–	PIO, I, PU, ST
D1	VDDIOP	GPIO_CLK	PB20	I/O	–	–	SPI1_SPCK	I/O	D10	I/O	–	–	PIO, I, PU, ST
L3	VDDIOP	GPIO	PB21	I/O	–	–	SPI1_NPCS0	I/O	D11	I/O	–	–	PIO, I, PU, ST
G1	VDDIOP	GPIO	PB22	I/O	–	–	SPI1_NPCS1	O	D12	I/O	–	–	PIO, I, PU, ST
H1	VDDIOP	GPIO	PB23	I/O	–	–	SPI1_NPCS2	O	D13	I/O	–	–	PIO, I, PU, ST
E1	VDDIOP	GPIO	PB24	I/O	–	–	DRXD	I	D14	I/O	TDI	I	TDI, PU, ST
J1	VDDIOP	GPIO	PB25	I/O	–	–	DTXD	O	D15	I/O	TDO	O	TDO, ST
M5	VDDIOP	GPIO_CLK	PB26	I/O	–	–	PCK0	O	RK0	I/O	PWMH0	O	PIO, I, PU, ST
L2	VDDIOP	GPIO	PB27	I/O	–	–	SPI1_NPCS3	O	TK0	I/O	PWML0	O	PIO, I, PU, ST
K1	VDDIOP	GPIO	PB28	I/O	–	–	SPI2_NPCS3	O	TD0	O	PWMH1	O	PIO, I, PU, ST
M3	VDDIOP	GPIO	PB29	I/O	–	–	TWD2	I/O	RD0	I	PWML1	O	PIO, O, LOW
M4	VDDIOP	GPIO	PB30	I/O	–	–	TWCK2	O	RF0	I/O	–	–	PIO, O, LOW
L1	VDDIOP	GPIO	PB31	I/O	–	–	–	–	TF0	I/O	–	–	PIO, I, PU, ST
V4	VDDIOM	GPIO	PC0	I/O	–	–	SPI0_MISO	I/O	PWMH2	O	ISI_D8	I	PIO, I, PU, ST
P8	VDDIOM	GPIO	PC1	I/O	–	–	SPI0_MOSI	I/O	PWML2	O	ISI_D9	I	PIO, I, PU, ST
V5	VDDIOM	GPIO_CLK	PC2	I/O	–	–	SPI0_SPCK	I/O	PWMH3	O	ISI_D10	I	PIO, I, PU, ST
R8	VDDIOM	GPIO	PC3	I/O	–	–	SPI0_NPCS0	I/O	PWML3	O	ISI_D11	I	PIO, I, PU, ST
W5	VDDIOM	MCI_CLK	PC4	I/O	–	–	SPI0_NPCS1	O	MCI0_CK	I/O	PCK1	O	PIO, I, PU, ST
T8	VDDIOM	GPIO	PC5	I/O	–	–	D0	I/O	MCI0_CDA	I/O	–	–	PIO, I, PU, ST
W6	VDDIOM	GPIO	PC6	I/O	–	–	D1	I/O	MCI0_DA0	I/O	–	–	PIO, I, PU, ST
R19	VDDIOM	GPIO	PC7	I/O	–	–	D2	I/O	MCI0_DA1	I/O	–	–	PIO, I, PU, ST
N15	VDDIOM	GPIO	PC8	I/O	–	–	D3	I/O	MCI0_DA2	I/O	–	–	PIO, I, PU, ST
U8	VDDIOM	GPIO	PC9	I/O	–	–	D4	I/O	MCI0_DA3	I/O	–	–	PIO, I, PU, ST

Table 3-1. TFBGA361 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
V6	VDDIOM	GPIO	PC10	I/O	–	–	D5	I/O	MCI0_DA4	I/O	–	–	PIO, I, PU, ST
V7	VDDIOM	GPIO	PC11	I/O	–	–	D6	I/O	MCI0_DA5	I/O	–	–	PIO, I, PU, ST
W7	VDDIOM	GPIO	PC12	I/O	–	–	D7	I/O	MCI0_DA6	I/O	–	–	PIO, I, PU, ST
V8	VDDIOM	GPIO	PC13	I/O	–	–	NRD/NANDOE	O	MCI0_DA7	I/O	–	–	PIO, I, PU, ST
U9	VDDIOM	GPIO	PC14	I/O	–	–	NWE/NANDWE	O	–	–	–	–	PIO, I, PU, ST
W8	VDDIOM	GPIO	PC15	I/O	–	–	NCS3	O	–	–	–	–	PIO, I, PU, ST
V9	VDDIOM	GPIO	PC16	I/O	–	–	NANDRDY	I	–	–	–	–	PIO, I, PU, ST
W9	VDDIOM	GPIO	PC17	I/O	–	–	A21/NANDALE	O	–	–	–	–	A21
V10	VDDIOM	GPIO	PC18	I/O	–	–	A22/NANDCLE	O	–	–	–	–	A22
U14	VDDIOM	GPIO	PC19	I/O	–	–	ISI_D0	I	TK1	I/O	–	–	PIO, I, PU, ST
V11	VDDIOM	GPIO	PC20	I/O	–	–	ISI_D1	I	TF1	I/O	–	–	PIO, I, PU, ST
U15	VDDIOM	GPIO	PC21	I/O	–	–	ISI_D2	I	TD1	O	–	–	PIO, I, PU, ST
T15	VDDIOM	GPIO	PC22	I/O	–	–	ISI_D3	I	RF1	I/O	–	–	PIO, I, PU, ST
U16	VDDIOM	GPIO	PC23	I/O	–	–	ISI_D4	I	RD1	I	–	–	PIO, I, PU, ST
T16	VDDIOM	GPIO	PC24	I/O	–	–	ISI_D5	I	RK1	I	PCK1	O	PIO, I, PU, ST
V17	VDDIOM	GPIO	PC25	I/O	–	–	ISI_D6	I	TWD3	I/O	URXD1	I	PIO, I, PU, ST
R16	VDDIOM	GPIO	PC26	I/O	–	–	ISI_D7	I	TWCK3	O	UTXD1	O	PIO, I, PU, ST
U12	VDDANA	GPIO_ANA	PC27	I/O	AD0	–	–	I	SPI0_NPCS1	O	PWML0	O	PIO, I, PU, ST
T11	VDDANA	GPIO_ANA	PC28	I/O	AD1	–	–	I	SPI0_NPCS2	O	PWML1	O	PIO, I, PU, ST
R13	VDDANA	GPIO_ANA	PC29	I/O	AD2	–	–	I	SPI0_NPCS3	O	PWMFI0	O	PIO, I, PU, ST
T12	VDDANA	GPIO_ANA	PC30	I/O	AD3	–	–	I	–	–	PWMH0	O	PIO, I, PU, ST
T13	VDDANA	GPIO_ANA	PC31	I/O	AD4	–	–	I	–	–	PWMH1	I	PIO, I, PU, ST
M1	VDDIOP	GPIO_CLK	PD8	I/O	–	–	PCK0	O	–	–	–	–	PIO, I, PU, ST
M2	VDDIOP	GPIO	PD9	I/O	–	–	FIQ	I	–	–	–	–	PIO, I, PU, ST
N2	VDDIOP	GPIO	PD10	I/O	–	–	CTS0	I	–	–	–	–	PIO, I, PU, ST
N3	VDDIOP	GPIO	PD11	I/O	–	–	RTS0	O	SPI2_MISO	I/O	–	–	PIO, I, PU, ST
N1	VDDIOP	GPIO	PD12	I/O	–	–	RXD0	I	–	–	–	–	PIO, O, PD
P3	VDDIOP	GPIO	PD13	I/O	–	–	TXD0	O	SPI2_MOSI	I/O	–	–	PIO, I, PU, ST
P2	VDDIOP	GPIO	PD14	I/O	–	–	CTS1	I	–	–	–	–	PIO, I, PU, ST
N4	VDDIOP	GPIO	PD15	I/O	–	–	RTS1	O	SPI2_SPCK	I/O	–	–	PIO, I, PU, ST
R2	VDDIOP	GPIO	PD16	I/O	–	–	RXD1	I	–	–	–	–	PIO, O, PD
R3	VDDIOP	GPIO	PD17	I/O	–	–	TXD1	O	SPI2_NPCS0	I/O	–	–	PIO, I, PU, ST
T9	VDDANA	GPIO	PD18	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
P11	VDDANA	GPIO	PD19	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
T10	VDDANA	GPIO	PD20	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
P10	VDDANA	GPIO	PD21	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
U11	VDDANA	GPIO	PD22	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
R10	VDDANA	GPIO	PD23	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST

Table 3-1. TFBGA361 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
U10	VDDANA	GPIO	PD24	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
R11	VDDANA	GPIO	PD25	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
U13	VDDANA	GPIO	PD26	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
T14	VDDANA	GPIO	PD27	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
R1	VDDIOP	GPIO_CLK	PD28	I/O	–	–	SCK0	I/O	–	–	–	–	PIO, I, PU, ST
P1	VDDIOP	GPIO_CLK	PD29	I/O	–	–	SCK1	I/O	–	–	–	–	PIO, I, PU, ST
N5	VDDIOP	GPIO	PD30	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
P5	VDDIOP	GPIO_CLK	PD31	I/O	–	–	SPI0_NPCS2	O	PCK1	O	–	–	PIO, I, PU, ST
W19	VDDIOM	MCI_CLK	PE0	I/O	–	–	A0/NBS0	O	MCI0_CDB	I/O	CTS4	I	O, High
U17	VDDIOM	EBI	PE1	I/O	–	–	A1	O	MCI0_DB0	I/O	–	–	O, High
T17	VDDIOM	EBI	PE2	I/O	–	–	A2	O	MCI0_DB1	I/O	–	–	A2, LOW
P16	VDDIOM	EBI	PE3	I/O	–	–	A3	O	MCI0_DB2	I/O	–	–	A3, LOW
U18	VDDIOM	EBI	PE4	I/O	–	–	A4	O	MCI0_DB3	I/O	–	–	A4, LOW
R17	VDDIOM	EBI	PE5	I/O	–	–	A5	O	CTS3	I	–	–	A5, LOW
V19	VDDIOM	EBI	PE6	I/O	–	–	A6	O	TIOA3	I/O	–	–	PIO, O, LOW
U19	VDDIOM	EBI	PE7	I/O	–	–	A7	O	TIOB3	I/O	PWMFI1	I	A7, LOW
T19	VDDIOM	EBI	PE8	I/O	–	–	A8	O	TCLK3	I	PWML3	O	A8, LOW
T18	VDDIOM	EBI	PE9	I/O	–	–	A9	O	TIOA2	I/O	–	–	A9, LOW
N14	VDDIOM	EBI	PE10	I/O	–	–	A10	O	TIOB2	I/O	–	–	A10, LOW
R18	VDDIOM	EBI	PE11	I/O	–	–	A11	O	TCLK2	I	–	–	A11, LOW
P17	VDDIOM	EBI	PE12	I/O	–	–	A12	O	TIOA1	I/O	PWMH2	O	A12, LOW
P18	VDDIOM	EBI	PE13	I/O	–	–	A13	O	TIOB1	I/O	PWML2	O	A13, LOW
N17	VDDIOM	EBI	PE14	I/O	–	–	A14	O	TCLK1	I	PWMH3	O	A14, LOW
N18	VDDIOM	EBI	PE15	I/O	–	–	A15	O	SCK3	I/O	TIOA0	I/O	A15, LOW
M15	VDDIOM	EBI	PE16	I/O	–	–	A16	O	RXD3	I	TIOB0	I/O	A16, LOW
N19	VDDIOM	EBI	PE17	I/O	–	–	A17	O	TXD3	O	TCLK0	I	A17, LOW
P19	VDDIOM	EBI	PE18	I/O	–	–	A18	O	TIOA5	I/O	MCI1_CK	I/O	A18, LOW
N16	VDDIOM	EBI	PE19	I/O	–	–	A19	O	TIOB5	I/O	MCI1_CDA	I/O	A19, LOW
M14	VDDIOM	EBI	PE20	I/O	–	–	A20	O	TCLK5	I	MCI1_DA0	I/O	A20, LOW
M18	VDDIOM	EBI	PE21	I/O	–	–	A23	O	TIOA4	I/O	MCI1_DA1	I/O	A23, LOW
M19	VDDIOM	EBI	PE22	I/O	–	–	A24	O	TIOB4	I/O	MCI1_DA2	I/O	A24, LOW
L18	VDDIOM	EBI	PE23	I/O	–	–	A25	O	TCLK4	I	MCI1_DA3	I/O	A25, LOW
L19	VDDIOM	EBI	PE24	I/O	–	–	NCS0	O	RTS3	O	–	–	NCS0, HIGH
M17	VDDIOM	EBI	PE25	I/O	–	–	NCS1	O	SCK4	I/O	IRQ	I	NCS1, HIGH
L15	VDDIOM	EBI	PE26	I/O	–	–	NCS2	O	RXD4	I	A18	O	NCS2, HIGH
M16	VDDIOM	EBI	PE27	I/O	–	–	NWR1/NBS1	O	TXD4	O	–	–	PIO, I, PD
L17	VDDIOM	EBI	PE28	I/O	–	–	NWAIT	I	RTS4	O	A19	O	PIO, I, PD
V1	VDDIOP	DIB	PE29	I/O	–	–	DIBP	O	URXD0	I	TWD1	I/O	PIO, O, LOW

Table 3-1. TFBGA361 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
U2	VDDIOP	DIB	PE30	I/O	–	–	DIBN	O	UTXD0	O	TWCK1	O	PIO, O, LOW
L4	VDDIOP	GPIO	PE31	I/O	–	–	ADTRG	I	–	–	–	–	PIO, O, LOW
H10 G10 K10 J10 K9 J9 H9 G9 E7 A8 D7 B7 C7 E9 A10 B9 D8 A9	–	Not connected	–	–	–	–	–	–	–	–	–	–	–
P4	VDDBU	SYSC	TST	I	–	–	–	–	–	–	–	–	I, PD, ST
W12	VDDIOP	CLOCK	XIN	I	–	–	–	–	–	–	–	–	I
V12	VDDIOP	CLOCK	XOUT	O	–	–	–	–	–	–	–	–	O
W2	VDDBU	CLOCK	XIN32	I	–	–	–	–	–	–	–	–	I
W3	VDDBU	CLOCK	XOUT32	O	–	–	–	–	–	–	–	–	O
T2	VDDBU	SYSC	SHDN	O	–	–	–	–	–	–	–	–	O, PU
V3	VDDBU	SYSC	WKUP	I	–	–	–	–	–	–	–	–	I, ST
U3	VDDBU	PIOBU	PIOBU0	I	–	–	–	–	–	–	–	–	I, PU
T3	VDDBU	PIOBU	PIOBU1	I	–	–	–	–	–	–	–	–	I, PU
T4	VDDBU	PIOBU	PIOBU2	I	–	–	–	–	–	–	–	–	I, PU
U4	VDDBU	PIOBU	PIOBU3	I	–	–	–	–	–	–	–	–	I, PU
P6	VDDBU	PIOBU	PIOBU4	I	–	–	–	–	–	–	–	–	I, PU
T5	VDDBU	PIOBU	PIOBU5	I	–	–	–	–	–	–	–	–	I, PU
R4	VDDBU	PIOBU	PIOBU6	I	–	–	–	–	–	–	–	–	I, PU
U5	VDDBU	PIOBU	PIOBU7	I	–	–	–	–	–	–	–	–	I, PU
R5 U6 R6 T6 R7 U7 P7 T7	–	Not connected	–	–	–	–	–	–	–	–	–	–	–
T1	VDDBU	RST	NRST	I	–	–	–	–	–	–	–	–	I
V2	VDDBU	SYSC	JTAGSEL	I	–	–	–	–	–	–	–	–	I, PD
F15	VDDIODDR	DDR_IO	DDR_A0	O	–	–	–	–	–	–	–	–	O, LOW
F16	VDDIODDR	DDR_IO	DDR_A1	O	–	–	–	–	–	–	–	–	O, LOW

Table 3-1. TFBGA361 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
E17	VDDIODDR	DDR_IO	DDR_A2	O	–	–	–	–	–	–	–	–	O, LOW
G15	VDDIODDR	DDR_IO	DDR_A3	O	–	–	–	–	–	–	–	–	O, LOW
B18	VDDIODDR	DDR_IO	DDR_A4	O	–	–	–	–	–	–	–	–	O, LOW
C16	VDDIODDR	DDR_IO	DDR_A5	O	–	–	–	–	–	–	–	–	O, LOW
E15	VDDIODDR	DDR_IO	DDR_A6	O	–	–	–	–	–	–	–	–	O, LOW
F17	VDDIODDR	DDR_IO	DDR_A7	O	–	–	–	–	–	–	–	–	O, LOW
F18	VDDIODDR	DDR_IO	DDR_A8	O	–	–	–	–	–	–	–	–	O, LOW
D19	VDDIODDR	DDR_IO	DDR_A9	O	–	–	–	–	–	–	–	–	O, LOW
E18	VDDIODDR	DDR_IO	DDR_A10	O	–	–	–	–	–	–	–	–	O, LOW
D18	VDDIODDR	DDR_IO	DDR_A11	O	–	–	–	–	–	–	–	–	O, LOW
C18	VDDIODDR	DDR_IO	DDR_A12	O	–	–	–	–	–	–	–	–	O, LOW
D16	VDDIODDR	DDR_IO	DDR_A13	O	–	–	–	–	–	–	–	–	O, LOW
L14	VDDIODDR	DDR_IO	DDR_D0	I/O	–	–	–	–	–	–	–	–	I, HiZ
K16	VDDIODDR	DDR_IO	DDR_D1	I/O	–	–	–	–	–	–	–	–	I, HiZ
K15	VDDIODDR	DDR_IO	DDR_D2	I/O	–	–	–	–	–	–	–	–	I, HiZ
K14	VDDIODDR	DDR_IO	DDR_D3	I/O	–	–	–	–	–	–	–	–	I, HiZ
J18	VDDIODDR	DDR_IO	DDR_D4	I/O	–	–	–	–	–	–	–	–	I, HiZ
J17	VDDIODDR	DDR_IO	DDR_D5	I/O	–	–	–	–	–	–	–	–	I, HiZ
J15	VDDIODDR	DDR_IO	DDR_D6	I/O	–	–	–	–	–	–	–	–	I, HiZ
H19	VDDIODDR	DDR_IO	DDR_D7	I/O	–	–	–	–	–	–	–	–	I, HiZ
H18	VDDIODDR	DDR_IO	DDR_D8	I/O	–	–	–	–	–	–	–	–	I, HiZ
J14	VDDIODDR	DDR_IO	DDR_D9	I/O	–	–	–	–	–	–	–	–	I, HiZ
G18	VDDIODDR	DDR_IO	DDR_D10	I/O	–	–	–	–	–	–	–	–	I, HiZ
H17	VDDIODDR	DDR_IO	DDR_D11	I/O	–	–	–	–	–	–	–	–	I, HiZ
H15	VDDIODDR	DDR_IO	DDR_D12	I/O	–	–	–	–	–	–	–	–	I, HiZ
H14	VDDIODDR	DDR_IO	DDR_D13	I/O	–	–	–	–	–	–	–	–	I, HiZ
G16	VDDIODDR	DDR_IO	DDR_D14	I/O	–	–	–	–	–	–	–	–	I, HiZ
E19	VDDIODDR	DDR_IO	DDR_D15	I/O	–	–	–	–	–	–	–	–	I, HiZ
E14	VDDIODDR	DDR_IO	DDR_D16	I/O	–	–	–	–	–	–	–	–	I, HiZ
E13	VDDIODDR	DDR_IO	DDR_D17	I/O	–	–	–	–	–	–	–	–	I, HiZ
H13	VDDIODDR	DDR_IO	DDR_D18	I/O	–	–	–	–	–	–	–	–	I, HiZ
F13	VDDIODDR	DDR_IO	DDR_D19	I/O	–	–	–	–	–	–	–	–	I, HiZ
B15	VDDIODDR	DDR_IO	DDR_D20	I/O	–	–	–	–	–	–	–	–	I, HiZ
A14	VDDIODDR	DDR_IO	DDR_D21	I/O	–	–	–	–	–	–	–	–	I, HiZ
D12	VDDIODDR	DDR_IO	DDR_D22	I/O	–	–	–	–	–	–	–	–	I, HiZ
B14	VDDIODDR	DDR_IO	DDR_D23	I/O	–	–	–	–	–	–	–	–	I, HiZ
B13	VDDIODDR	DDR_IO	DDR_D24	I/O	–	–	–	–	–	–	–	–	I, HiZ
G12	VDDIODDR	DDR_IO	DDR_D25	I/O	–	–	–	–	–	–	–	–	I, HiZ

Table 3-1. TFBGA361 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
B12	VDDIODDR	DDR_IO	DDR_D26	I/O	–	–	–	–	–	–	–	–	I, HiZ
C12	VDDIODDR	DDR_IO	DDR_D27	I/O	–	–	–	–	–	–	–	–	I, HiZ
F11	VDDIODDR	DDR_IO	DDR_D28	I/O	–	–	–	–	–	–	–	–	I, HiZ
C11	VDDIODDR	DDR_IO	DDR_D29	I/O	–	–	–	–	–	–	–	–	I, HiZ
D11	VDDIODDR	DDR_IO	DDR_D30	I/O	–	–	–	–	–	–	–	–	I, HiZ
B11	VDDIODDR	DDR_IO	DDR_D31	I/O	–	–	–	–	–	–	–	–	I, HiZ
L16	VDDIODDR	DDR_IO	DDR_DQM0	O	–	–	–	–	–	–	–	–	O, LOW
J16	VDDIODDR	DDR_IO	DDR_DQM1	O	–	–	–	–	–	–	–	–	O, LOW
D13	VDDIODDR	DDR_IO	DDR_DQM2	O	–	–	–	–	–	–	–	–	O, LOW
F12	VDDIODDR	DDR_IO	DDR_DQM3	O	–	–	–	–	–	–	–	–	O, LOW
J19	VDDIODDR	DDR_IO	DDR_DQS0	I/O	–	–	–	–	–	–	–	–	O, LOW
F19	VDDIODDR	DDR_IO	DDR_DQS1	I/O	–	–	–	–	–	–	–	–	O, LOW
A15	VDDIODDR	DDR_IO	DDR_DQS2	I/O	–	–	–	–	–	–	–	–	O, LOW
A12	VDDIODDR	DDR_IO	DDR_DQS3	I/O	–	–	–	–	–	–	–	–	O, LOW
K19	VDDIODDR	DDR_IO	DDR_DQSN0	I/O	–	–	–	–	–	–	–	–	O, HIGH
G19	VDDIODDR	DDR_IO	DDR_DQSN1	I/O	–	–	–	–	–	–	–	–	O, HIGH
A16	VDDIODDR	DDR_IO	DDR_DQSN2	I/O	–	–	–	–	–	–	–	–	O, HIGH
A13	VDDIODDR	DDR_IO	DDR_DQSN3	I/O	–	–	–	–	–	–	–	–	O, HIGH
B16	VDDIODDR	DDR_IO	DDR_CS	O	–	–	–	–	–	–	–	–	O, LOW
A18	VDDIODDR	DDR_IO	DDR_CLK	O	–	–	–	–	–	–	–	–	O
A19	VDDIODDR	DDR_IO	DDR_CLKN	O	–	–	–	–	–	–	–	–	O
D15	VDDIODDR	DDR_IO	DDR_CKE	O	–	–	–	–	–	–	–	–	O, LOW
B17	VDDIODDR	DDR_IO	DDR_RAS	O	–	–	–	–	–	–	–	–	O, LOW
A17	VDDIODDR	DDR_IO	DDR_CAS	O	–	–	–	–	–	–	–	–	O, LOW
E16	VDDIODDR	DDR_IO	DDR_WE	O	–	–	–	–	–	–	–	–	O, LOW
C15	VDDIODDR	DDR_IO	DDR_BA0	O	–	–	–	–	–	–	–	–	O, LOW
D14	VDDIODDR	DDR_IO	DDR_BA1	O	–	–	–	–	–	–	–	–	O, LOW
G13	VDDIODDR	DDR_IO	DDR_BA2	O	–	–	–	–	–	–	–	–	O, LOW
C19	VDDIODDR	Reference	DDR_CALN	I	–	–	–	–	–	–	–	–	I
B19	GNDIODDR	Reference	DDR_CALP	I	–	–	–	–	–	–	–	–	I
K12	VDDIODDR/2	Reference	DDR_VREF	I	–	–	–	–	–	–	–	–	I
W11	VBG	VBG	VBG	I	–	–	–	–	–	–	–	–	I
R12	VDDANA	Reference	ADCVREF	I	–	–	–	–	–	–	–	–	I
W16	VDDUTMII	USBHS	HHSDPC	I/O	–	–	–	–	–	–	–	–	O, PD
V16	VDDUTMII	USBHS	HHSDMC	I/O	–	–	–	–	–	–	–	–	O, PD
W15	VDDUTMII	USBHS	HHSDPB	I/O	–	–	–	–	–	–	–	–	O, PD
V15	VDDUTMII	USBHS	HHSDMB	I/O	–	–	–	–	–	–	–	–	O, PD
W14	VDDUTMII	USBHS	HHSDPA	I/O	DHSDP	I/O	–	–	–	–	–	–	O, PD

Table 3-1. TFBGA361 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
V14	VDDUTMII	USBHS	HHSDMA	I/O	DHSDM	I/O	–	–	–	–	–	–	O, PD
W4	VDDBU	Power supply	VDDBU	I	–	–	–	–	–	–	–	–	I
W1	GNDBU	Ground	GNDBU	I	–	–	–	–	–	–	–	–	I
G8 N9	VDDCORE	Power supply	VDDCORE	I	–	–	–	–	–	–	–	–	I
B10 D9 D10 F9 H8 J8 J12 K11 L8 L10 L12 M9 M10 M11 V18 W18	GNDCORE	Ground	GNDCORE	I	–	–	–	–	–	–	–	–	I
J7 J11 K7 K8 L9 L11 N10	VCCCORE	Power supply	VCCCORE	I	–	–	–	–	–	–	–	–	I
C14 D17 E10 E12 F14 H12 H16 K13 K17 M13	VDDIODDR	Power supply	VDDIODDR	I	–	–	–	–	–	–	–	–	I
C13 C17 E11 F10 G11 G14 G17 J13 K18 L13	GNDIODDR	Ground	GNDIODDR	I	–	–	–	–	–	–	–	–	I
M8 N7 P15 R9	VDDIOM	Power supply	VDDIOM	I	–	–	–	–	–	–	–	–	I

Table 3-1. TFBGA361 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
M7 M12 N8 P9	GNDIOM	Ground	GNDIOM	I	–	–	–	–	–	–	–	–	I
B8 C8 E8 F8	GNDIOP	Ground	GNDIOP	I	–	–	–	–	–	–	–	–	I
H7 K6 L5 M6	VDDIOP	Power supply	VDDIOP	I	–	–	–	–	–	–	–	–	I
F7 G6 G7 J6 L6 N6	GNDIOP	Ground	GNDIOP	I	–	–	–	–	–	–	–	–	I
V13	VDDUTMIC	Power supply	VDDUTMIC	I	–	–	–	–	–	–	–	–	I
W13 W17	VDDUTMII	Power supply	VDDUTMII	I	–	–	–	–	–	–	–	–	I
P13	GNDUTMI	Ground	GNDUTMI	I	–	–	–	–	–	–	–	–	I
W10	VDDPLLA	Power supply	VDDPLLA	I	–	–	–	–	–	–	–	–	I
L7	GNDPLL	Ground	GNDPLL	I	–	–	–	–	–	–	–	–	I
P14	VDDOSC	Power supply	VDDOSC	I	–	–	–	–	–	–	–	–	I
N13	GNDOSC	Ground	GNDOSC	I	–	–	–	–	–	–	–	–	I
A11	GNDIOP	Ground	GNDIOP	I	–	–	–	–	–	–	–	–	I
C9 N11 P12	VDDANA	Power supply	VDDANA	I	–	–	–	–	–	–	–	–	I
C10 H11 N12	GNDANA	Ground	GNDANA	I	–	–	–	–	–	–	–	–	I
R14	VDDFUSE	Power supply	VDDFUSE	I	–	–	–	–	–	–	–	–	I
R15	GNDFUSE	Ground	GNDFUSE	I	–	–	–	–	–	–	–	–	I
U1	–	Not connected	–	–	–	–	–	–	–	–	–	–	–

3.2 289-ball LFBGA Package Pinout

In this package, the DDRC datapath is reduced to 16 bits.

Table 3-2. LFBGA289 Pin Description

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
C5	VDDIOP	GPIO	PA0	I/O	–	–	LCDDAT0	O	–	–	TMS	I	TMS, PU, ST
F6	VDDIOP	GPIO	PA1	I/O	–	–	LCDDAT1	O	–	–	–	–	PIO, I, PU, ST
F5	VDDIOP	GPIO_CLK	PA2	I/O	–	–	LCDDAT2	O	G1_TXCK	I	–	–	PIO, I, PU, ST
B5	VDDIOP	GPIO_CLK	PA3	I/O	–	–	LCDDAT3	O	G1_RXCK	I	–	–	PIO, I, PU, ST
E5	VDDIOP	GPIO	PA4	I/O	–	–	LCDDAT4	O	G1_TXEN	O	–	–	PIO, I, PU, ST
A5	VDDIOP	GPIO	PA5	I/O	–	–	LCDDAT5	O	G1_TXER	O	–	–	PIO, I, PU, ST
A4	VDDIOP	GPIO	PA6	I/O	–	–	LCDDAT6	O	G1_CRS	I	–	–	PIO, I, PU, ST
E4	VDDIOP	GPIO	PA7	I/O	–	–	LCDDAT7	O	–	–	–	–	PIO, I, PU, ST
B4	VDDIOP	GPIO	PA8	I/O	–	–	LCDDAT8	O	–	–	TCK	I	TCK, PU
D4	VDDIOP	GPIO	PA9	I/O	–	–	LCDDAT9	O	G1_COL	I	–	–	PIO, I, PU, ST
C4	VDDIOP	GPIO	PA10	I/O	–	–	LCDDAT10	O	G1_RXDV	I	–	–	PIO, I, PU, ST
A3	VDDIOP	GPIO	PA11	I/O	–	–	LCDDAT11	O	G1_RXER	I	–	–	PIO, I, PU, ST
F4	VDDIOP	GPIO	PA12	I/O	–	–	LCDDAT12	O	G1_RX0	I	–	–	PIO, I, PU, ST
F3	VDDIOP	GPIO	PA13	I/O	–	–	LCDDAT13	O	G1_RX1	I	–	–	PIO, I, PU, ST
D3	VDDIOP	GPIO	PA14	I/O	–	–	LCDDAT14	O	G1_TX0	O	–	–	PIO, I, PU, ST
B3	VDDIOP	GPIO	PA15	I/O	–	–	LCDDAT15	O	G1_TX1	O	–	–	PIO, I, PU, ST
G3	VDDIOP	GPIO	PA16	I/O	–	–	LCDDAT16	O	–	–	NTRST	I	NTRST, PU, ST
E3	VDDIOP	GPIO	PA17	I/O	–	–	LCDDAT17	O	–	–	–	–	PIO, O, LOW
C3	VDDIOP	GPIO	PA18	I/O	–	–	LCDDAT18	O	G1_RX2	I	–	–	PIO, O, LOW
A2	VDDIOP	GPIO	PA19	I/O	–	–	LCDDAT19	O	G1_RX3	I	–	–	PIO, O, LOW
G5	VDDIOP	GPIO	PA20	I/O	–	–	LCDDAT20	O	G1_TX2	O	–	–	PIO, I, PU, ST
A1	VDDIOP	GPIO	PA21	I/O	–	–	LCDDAT21	O	G1_TX3	O	–	–	PIO, I, PU, ST
D2	VDDIOP	GPIO	PA22	I/O	–	–	LCDDAT22	O	G1_MDC	O	–	–	PIO, I, PU, ST
E2	VDDIOP	GPIO	PA23	I/O	–	–	LCDDAT23	O	G1_MDIO	I/O	–	–	PIO, I, PU, ST
G4	VDDIOP	GPIO_CLK	PA24	I/O	–	–	LCDPWM	O	PCK0	O	–	–	PIO, I, PU, ST
C2	VDDIOP	GPIO	PA25	I/O	–	–	LCDDISP	O	TD0	O	–	–	PIO, I, PU, ST
B2	VDDIOP	GPIO	PA26	I/O	–	–	LCDVSYNC	O	PWMH0	O	SPI1_NPCS1	O	PIO, I, PU, ST
H3	VDDIOP	GPIO	PA27	I/O	–	–	LCDHSYNC	O	PWML0	O	SPI1_NPCS2	O	PIO, I, PU, ST
F2	VDDIOP	GPIO_CLK2	PA28	I/O	–	–	LCDPCK	O	PWMH1	O	SPI1_NPCS3	O	PIO, I, PU, ST
B1	VDDIOP	GPIO	PA29	I/O	–	–	LCDDEN	O	PWML1	O	–	–	PIO, I, PU, ST
C1	VDDIOP	GPIO	PA30	I/O	–	–	TWD0	I/O	–	–	–	–	PIO, I, PU, ST
H5	VDDIOP	GPIO	PA31	I/O	–	–	TWCK0	O	–	–	–	–	PIO, I, PU, ST
D1	VDDIOP	GPIO_CLK	PB0	I/O	–	–	G0_TXCK	I	–	–	–	–	PIO, I, PU, ST
H4	VDDIOP	GPIO_CLK	PB1	I/O	–	–	G0_RXCK	I	SCK2	I/O	ISI_PCK	I	PIO, I, PU, ST
G2	VDDIOP	GPIO	PB2	I/O	–	–	G0_TXEN	O	–	–	–	–	PIO, I, PU, ST

Table 3-2. LFBGA289 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
E1	VDDIOP	GPIO	PB3	I/O	–	–	G0_TXER	O	CTS2	I	ISI_VSYNC	I	PIO, I, PU, ST
F1	VDDIOP	GPIO	PB4	I/O	–	–	G0_CRS	I	RXD2	I	ISI_HSYNC	I	PIO, I, PU, ST
J3	VDDIOP	GPIO	PB5	I/O	–	–	G0_COL	I	TXD2	O	PCK2	O	PIO, I, PU, ST
H2	VDDIOP	GPIO	PB6	I/O	–	–	G0_RXDV	I	–	–	–	–	PIO, I, PU, ST
J5	VDDIOP	GPIO	PB7	I/O	–	–	G0_RXER	I	–	–	–	–	PIO, I, PU, ST
J2	VDDIOP	GPIO	PB8	I/O	–	–	G0_RX0	I	–	–	–	–	PIO, I, PU, ST
G1	VDDIOP	GPIO	PB9	I/O	–	–	G0_RX1	I	–	–	–	–	PIO, I, PU, ST
H1	VDDIOP	GPIO_CLK	PB10	I/O	–	–	G0_RX2	I	PCK2	O	PWML1	O	PIO, I, PU, ST
J4	VDDIOP	GPIO	PB11	I/O	–	–	G0_RX3	I	RTS2	O	PWMH1	O	PIO, I, PU, ST
J1	VDDIOP	GPIO	PB12	I/O	–	–	G0_TX0	O	–	–	–	–	PIO, I, PU, ST
K6	VDDIOP	GPIO	PB13	I/O	–	–	G0_TX1	O	–	–	–	–	PIO, I, PU, ST
K1	VDDIOP	GPIO	PB14	I/O	–	–	G0_TX2	O	SPI2_NPCS1	O	PWMH0	O	PIO, I, PU, ST
K2	VDDIOP	GPIO	PB15	I/O	–	–	G0_TX3	O	SPI2_NPCS2	O	PWML0	O	PIO, I, PU, ST
L1	VDDIOP	GPIO	PB16	I/O	–	–	G0_MDC	O	–	–	–	–	PIO, I, PU, ST
K3	VDDIOP	GPIO	PB17	I/O	–	–	G0_MDIO	I/O	–	–	–	–	PIO, I, PU, ST
L2	VDDIOP	GPIO	PB18	I/O	–	–	SPI1_MISO	I/O	D8	I/O	–	–	PIO, I, PU, ST
M1	VDDIOP	GPIO	PB19	I/O	–	–	SPI1_MOSI	I/O	D9	I/O	–	–	PIO, I, PU, ST
N1	VDDIOP	GPIO_CLK	PB20	I/O	–	–	SPI1_SPCK	I/O	D10	I/O	–	–	PIO, I, PU, ST
K4	VDDIOP	GPIO	PB21	I/O	–	–	SPI1_NPCS0	I/O	D11	I/O	–	–	PIO, I, PU, ST
P1	VDDIOP	GPIO	PB22	I/O	–	–	SPI1_NPCS1	O	D12	I/O	–	–	PIO, I, PU, ST
M2	VDDIOP	GPIO	PB23	I/O	–	–	SPI1_NPCS2	O	D13	I/O	–	–	PIO, I, PU, ST
R1	VDDIOP	GPIO	PB24	I/O	–	–	DRXD	I	D14	I/O	TDI	I	TDI, PU, ST
T1	VDDIOP	GPIO	PB25	I/O	–	–	DTXD	O	D15	I/O	TDO	O	TDO, ST
K5	VDDIOP	GPIO_CLK	PB26	I/O	–	–	PCK0	O	RK0	I/O	PWMH0	O	PIO, I, PU, ST
U1	VDDIOP	GPIO	PB27	I/O	–	–	SPI1_NPCS3	O	TK0	I/O	PWML0	O	PIO, I, PU, ST
K7	VDDIOP	GPIO	PB28	I/O	–	–	SPI2_NPCS3	O	TD0	O	PWMH1	O	PIO, I, PU, ST
L3	VDDIOP	GPIO	PB29	I/O	–	–	TWD2	I/O	RD0	I	PWML1	O	PIO, O, LOW
L4	VDDIOP	GPIO	PB30	I/O	–	–	TWCK2	O	RF0	I/O	–	–	PIO, O, LOW
U2	VDDIOP	GPIO	PB31	I/O	–	–	–	–	TF0	I/O	–	–	PIO, I, PU, ST
U7	VDDIOM	GPIO	PC0	I/O	–	–	SPI0_MISO	I/O	PWMH2	O	ISI_D8	I	PIO, I, PU, ST
U9	VDDIOM	GPIO	PC1	I/O	–	–	SPI0_MOSI	I/O	PWML2	O	ISI_D9	I	PIO, I, PU, ST
U8	VDDIOM	GPIO_CLK	PC2	I/O	–	–	SPI0_SPCK	I/O	PWMH3	O	ISI_D10	I	PIO, I, PU, ST
M8	VDDIOM	GPIO	PC3	I/O	–	–	SPI0_NPCS0	I/O	PWML3	O	ISI_D11	I	PIO, I, PU, ST
U10	VDDIOM	MCI_CLK	PC4	I/O	–	–	SPI0_NPCS1	O	MCI0_CK	I/O	PCK1	O	PIO, I, PU, ST
N7	VDDIOM	GPIO	PC5	I/O	–	–	D0	I/O	MCI0_CDA	I/O	–	–	PIO, I, PU, ST
T7	VDDIOM	GPIO	PC6	I/O	–	–	D1	I/O	MCI0_DA0	I/O	–	–	PIO, I, PU, ST
G17	VDDIOM	GPIO	PC7	I/O	–	–	D2	I/O	MCI0_DA1	I/O	–	–	PIO, I, PU, ST
J13	VDDIOM	GPIO	PC8	I/O	–	–	D3	I/O	MCI0_DA2	I/O	–	–	PIO, I, PU, ST

Table 3-2. LFBGA289 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
P7	VDDIOM	GPIO	PC9	I/O	–	–	D4	I/O	MCI0_DA3	I/O	–	–	PIO, I, PU, ST
R7	VDDIOM	GPIO	PC10	I/O	–	–	D5	I/O	MCI0_DA4	I/O	–	–	PIO, I, PU, ST
U11	VDDIOM	GPIO	PC11	I/O	–	–	D6	I/O	MCI0_DA5	I/O	–	–	PIO, I, PU, ST
T8	VDDIOM	GPIO	PC12	I/O	–	–	D7	I/O	MCI0_DA6	I/O	–	–	PIO, I, PU, ST
U12	VDDIOM	GPIO	PC13	I/O	–	–	NRD/NANDOE	O	MCI0_DA7	I/O	–	–	PIO, I, PU, ST
R8	VDDIOM	GPIO	PC14	I/O	–	–	NWE/NANDWE	O	–	–	–	–	PIO, I, PU, ST
U13	VDDIOM	GPIO	PC15	I/O	–	–	NCS3	O	–	–	–	–	PIO, I, PU, ST
P8	VDDIOM	GPIO	PC16	I/O	–	–	NANDRDY	I	–	–	–	–	PIO, I, PU, ST
T9	VDDIOM	GPIO	PC17	I/O	–	–	A21/NANDALE	O	–	–	–	–	A21
T11	VDDIOM	GPIO	PC18	I/O	–	–	A22/NANDCLE	O	–	–	–	–	A22
T10	VDDIOM	GPIO	PC19	I/O	–	–	ISI_D0	I	TK1	I/O	–	–	PIO, I, PU, ST
N8	VDDIOM	GPIO	PC20	I/O	–	–	ISI_D1	I	TF1	I/O	–	–	PIO, I, PU, ST
P15	VDDIOM	GPIO	PC21	I/O	–	–	ISI_D2	I	TD1	O	–	–	PIO, I, PU, ST
N16	VDDIOM	GPIO	PC22	I/O	–	–	ISI_D3	I	RF1	I/O	–	–	PIO, I, PU, ST
P16	VDDIOM	GPIO	PC23	I/O	–	–	ISI_D4	I	RD1	I	–	–	PIO, I, PU, ST
N17	VDDIOM	GPIO	PC24	I/O	–	–	ISI_D5	I	RK1	I	PCK1	O	PIO, I, PU, ST
P17	VDDIOM	GPIO	PC25	I/O	–	–	ISI_D6	I	TWD3	I/O	URXD1	I	PIO, I, PU, ST
M17	VDDIOM	GPIO	PC26	I/O	–	–	ISI_D7	I	TWCK3	O	UTXD1	O	PIO, I, PU, ST
T12	VDDANA	GPIO_ANA	PC27	I/O	AD0	–	–	I	SPI0_NPCS1	O	PWML0	O	PIO, I, PU, ST
R13	VDDANA	GPIO_ANA	PC28	I/O	AD1	–	–	I	SPI0_NPCS2	O	PWML1	O	PIO, I, PU, ST
T13	VDDANA	GPIO_ANA	PC29	I/O	AD2	–	–	I	SPI0_NPCS3	O	PWMFI0	O	PIO, I, PU, ST
R14	VDDANA	GPIO_ANA	PC30	I/O	AD3	–	–	I	–	–	PWMH0	O	PIO, I, PU, ST
R15	VDDANA	GPIO_ANA	PC31	I/O	AD4	–	–	I	–	–	PWMH1	I	PIO, I, PU, ST
L7	VDDIOP	GPIO_CLK	PD8	I/O	–	–	PCK0	O	–	–	–	–	PIO, I, PU, ST
P2	VDDIOP	GPIO	PD9	I/O	–	–	FIQ	I	–	–	–	–	PIO, I, PU, ST
T2	VDDIOP	GPIO	PD10	I/O	–	–	CTS0	I	–	–	–	–	PIO, I, PU, ST
M3	VDDIOP	GPIO	PD11	I/O	–	–	RTS0	O	SPI2_MISO	I/O	–	–	PIO, I, PU, ST
N2	VDDIOP	GPIO	PD12	I/O	–	–	RXD0	I	–	–	–	–	PIO, O, PD
M4	VDDIOP	GPIO	PD13	I/O	–	–	TXD0	O	SPI2_MOSI	I/O	–	–	PIO, I, PU, ST
K8	VDDIOP	GPIO	PD14	I/O	–	–	CTS1	I	–	–	–	–	PIO, I, PU, ST
N3	VDDIOP	GPIO	PD15	I/O	–	–	RTS1	O	SPI2_SPCK	I/O	–	–	PIO, I, PU, ST
L8	VDDIOP	GPIO	PD16	I/O	–	–	RXD1	I	–	–	–	–	PIO, I, PU, ST
P3	VDDIOP	GPIO	PD17	I/O	–	–	TXD1	O	SPI2_NPCS0	I/O	–	–	PIO, I, PU, ST
P9	VDDANA	GPIO	PD18	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
M10	VDDANA	GPIO	PD19	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
R9	VDDANA	GPIO	PD20	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
R10	VDDANA	GPIO	PD21	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
P10	VDDANA	GPIO	PD22	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST

Table 3-2. LFBGA289 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
L11	VDDANA	GPIO	PD23	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
R11	VDDANA	GPIO	PD24	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
M11	VDDANA	GPIO	PD25	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
P11	VDDANA	GPIO	PD26	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
L12	VDDANA	GPIO	PD27	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
L9	VDDIOP	GPIO_CLK	PD28	I/O	–	–	SCK0	I/O	–	–	–	–	PIO, I, PU, ST
R2	VDDIOP	GPIO_CLK	PD29	I/O	–	–	SCK1	I/O	–	–	–	–	PIO, I, PU, ST
L5	VDDIOP	GPIO	PD30	I/O	–	–	–	–	–	–	–	–	PIO, I, PU, ST
L6	VDDIOP	GPIO_CLK	PD31	I/O	–	–	SPI0_NPCS2	O	PCK1	O	–	–	PIO, I, PU, ST
N14	VDDIOM	MCI_CLK	PE0	I/O	–	–	A0/NBS0	O	MCI0_CDB	I/O	CTS4	I	O, HIGH
N13	VDDIOM	EBI	PE1	I/O	–	–	A1	O	MCI0_DB0	I/O	–	–	O, HIGH
M16	VDDIOM	EBI	PE2	I/O	–	–	A2	O	MCI0_DB1	I/O	–	–	A2, LOW
M15	VDDIOM	EBI	PE3	I/O	–	–	A3	O	MCI0_DB2	I/O	–	–	A3, LOW
J16	VDDIOM	EBI	PE4	I/O	–	–	A4	O	MCI0_DB3	I/O	–	–	A4, LOW
L17	VDDIOM	EBI	PE5	I/O	–	–	A5	O	CTS3	I	–	–	A5, LOW
J17	VDDIOM	EBI	PE6	I/O	–	–	A6	O	TIOA3	I/O	–	–	PIO, O, LOW
K17	VDDIOM	EBI	PE7	I/O	–	–	A7	O	TIOB3	I/O	PWMFI1	I	A7, LOW
H16	VDDIOM	EBI	PE8	I/O	–	–	A8	O	TCLK3	I	PWML3	O	A8, LOW
L16	VDDIOM	EBI	PE9	I/O	–	–	A9	O	TIOA2	I/O	–	–	A9, LOW
L14	VDDIOM	EBI	PE10	I/O	–	–	A10	O	TIOB2	I/O	–	–	A10, LOW
H17	VDDIOM	EBI	PE11	I/O	–	–	A11	O	TCLK2	I	–	–	A11, LOW
L15	VDDIOM	EBI	PE12	I/O	–	–	A12	O	TIOA1	I/O	PWMH2	O	A12, LOW
G16	VDDIOM	EBI	PE13	I/O	–	–	A13	O	TIOB1	I/O	PWML2	O	A13, LOW
K12	VDDIOM	EBI	PE14	I/O	–	–	A14	O	TCLK1	I	PWMH3	O	A14, LOW
F16	VDDIOM	EBI	PE15	I/O	–	–	A15	O	SCK3	I/O	TIOA0	I/O	A15, LOW
K16	VDDIOM	EBI	PE16	I/O	–	–	A16	O	RXD3	I	TIOB0	I/O	A16, LOW
F17	VDDIOM	EBI	PE17	I/O	–	–	A17	O	TXD3	O	TCLK0	I	A17, LOW
E16	VDDIOM	EBI	PE18	I/O	–	–	A18	O	TIOA5	I/O	MCI1_CK	I/O	A18, LOW
D16	VDDIOM	EBI	PE19	I/O	–	–	A19	O	TIOB5	I/O	MCI1_CDA	I/O	A19, LOW
E17	VDDIOM	EBI	PE20	I/O	–	–	A20	O	TCLK5	I	MCI1_DA0	I/O	A20, LOW
D17	VDDIOM	EBI	PE21	I/O	–	–	A23	O	TIOA4	I/O	MCI1_DA1	I/O	A23, LOW
C16	VDDIOM	EBI	PE22	I/O	–	–	A24	O	TIOB4	I/O	MCI1_DA2	I/O	A24, LOW
C17	VDDIOM	EBI	PE23	I/O	–	–	A25	O	TCLK4	I	MCI1_DA3	I/O	A25, LOW
K13	VDDIOM	EBI	PE24	I/O	–	–	NCS0	O	RTS3	O	–	–	NCS0, HIGH
B17	VDDIOM	EBI	PE25	I/O	–	–	NCS1	O	SCK4	I/O	IRQ	I	NCS1, HIGH
K14	VDDIOM	EBI	PE26	I/O	–	–	NCS2	O	RXD4	I	A18	O	NCS2, HIGH
K15	VDDIOM	EBI	PE27	I/O	–	–	NWR1/NBS1	O	TXD4	O	–	–	PIO, I, PU, ST
J10	VDDIOM	EBI	PE28	I/O	–	–	NWAIT	I	RTS4	O	A19	O	PIO, I, PU, ST

Table 3-2. LFBGA289 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
P6	VDDIOP	DIB	PE29	I/O	–	–	DIBP	O	URXD0	I	TWD1	I/O	PIO, O, LOW
N6	VDDIOP	DIB	PE30	I/O	–	–	DIBN	O	UTXD0	O	TWCK1	O	PIO, O, LOW
K9	VDDIOP	GPIO	PE31	I/O	–	–	ADTRG	I	–	–	–	–	PIO, O, LOW
R3	VDDBU	SYSC	TST	I	–	–	–	–	–	–	–	–	I, PD, ST
T15	VDDIOP	CLOCK	XIN	I	–	–	–	–	–	–	–	–	I
U15	VDDIOP	CLOCK	XOUT	O	–	–	–	–	–	–	–	–	O
U5	VDDBU	CLOCK	XIN32	I	–	–	–	–	–	–	–	–	I
T5	VDDBU	CLOCK	XOUT32	O	–	–	–	–	–	–	–	–	O
U4	VDDBU	SYSC	SHDN	O	–	–	–	–	–	–	–	–	O, PU
T4	VDDBU	SYSC	WKUP	I	–	–	–	–	–	–	–	–	I, ST
M5	VDDBU	PIOBU	PIOBU0	I	–	–	–	–	–	–	–	–	I, PU
R4	VDDBU	PIOBU	PIOBU1	I	–	–	–	–	–	–	–	–	I, PU
P4	VDDBU	PIOBU	PIOBU2	I	–	–	–	–	–	–	–	–	I, PU
R5	VDDBU	PIOBU	PIOBU3	I	–	–	–	–	–	–	–	–	I, PU
N5	VDDBU	PIOBU	PIOBU4	I	–	–	–	–	–	–	–	–	I, PU
P5	VDDBU	PIOBU	PIOBU5	I	–	–	–	–	–	–	–	–	I, PU
N4	VDDBU	PIOBU	PIOBU6	I	–	–	–	–	–	–	–	–	I, PU
R6	VDDBU	PIOBU	PIOBU7	I	–	–	–	–	–	–	–	–	I, PU
U3	VDDBU	PIOBU	NRST	I	–	–	–	–	–	–	–	–	I
T3	VDDBU	SYSC	JTAGSEL	I	–	–	–	–	–	–	–	–	I, PD
B12	VDDIODDR	DDR_IO	DDR_A0	O	–	–	–	–	–	–	–	–	O, LOW
A12	VDDIODDR	DDR_IO	DDR_A1	O	–	–	–	–	–	–	–	–	O, LOW
E15	VDDIODDR	DDR_IO	DDR_A2	O	–	–	–	–	–	–	–	–	O, LOW
G11	VDDIODDR	DDR_IO	DDR_A3	O	–	–	–	–	–	–	–	–	O, LOW
C13	VDDIODDR	DDR_IO	DDR_A4	O	–	–	–	–	–	–	–	–	O, LOW
D12	VDDIODDR	DDR_IO	DDR_A5	O	–	–	–	–	–	–	–	–	O, LOW
C11	VDDIODDR	DDR_IO	DDR_A6	O	–	–	–	–	–	–	–	–	O, LOW
A14	VDDIODDR	DDR_IO	DDR_A7	O	–	–	–	–	–	–	–	–	O, LOW
F12	VDDIODDR	DDR_IO	DDR_A8	O	–	–	–	–	–	–	–	–	O, LOW
C14	VDDIODDR	DDR_IO	DDR_A9	O	–	–	–	–	–	–	–	–	O, LOW
G12	VDDIODDR	DDR_IO	DDR_A10	O	–	–	–	–	–	–	–	–	O, LOW
A13	VDDIODDR	DDR_IO	DDR_A11	O	–	–	–	–	–	–	–	–	O, LOW
B13	VDDIODDR	DDR_IO	DDR_A12	O	–	–	–	–	–	–	–	–	O, LOW
C10	VDDIODDR	DDR_IO	DDR_A13	O	–	–	–	–	–	–	–	–	O, LOW
J14	VDDIODDR	DDR_IO	DDR_D0	I/O	–	–	–	–	–	–	–	–	I, HiZ
B16	VDDIODDR	DDR_IO	DDR_D1	I/O	–	–	–	–	–	–	–	–	I, HiZ
J9	VDDIODDR	DDR_IO	DDR_D2	I/O	–	–	–	–	–	–	–	–	I, HiZ
J12	VDDIODDR	DDR_IO	DDR_D3	I/O	–	–	–	–	–	–	–	–	I, HiZ

Table 3-2. LFBGA289 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
A16	VDDIODDR	DDR_IO	DDR_D4	I/O	–	–	–	–	–	–	–	–	I, HiZ
A15	VDDIODDR	DDR_IO	DDR_D5	I/O	–	–	–	–	–	–	–	–	I, HiZ
H10	VDDIODDR	DDR_IO	DDR_D6	I/O	–	–	–	–	–	–	–	–	I, HiZ
B15	VDDIODDR	DDR_IO	DDR_D7	I/O	–	–	–	–	–	–	–	–	I, HiZ
G15	VDDIODDR	DDR_IO	DDR_D8	I/O	–	–	–	–	–	–	–	–	I, HiZ
H13	VDDIODDR	DDR_IO	DDR_D9	I/O	–	–	–	–	–	–	–	–	I, HiZ
C15	VDDIODDR	DDR_IO	DDR_D10	I/O	–	–	–	–	–	–	–	–	I, HiZ
D15	VDDIODDR	DDR_IO	DDR_D11	I/O	–	–	–	–	–	–	–	–	I, HiZ
H12	VDDIODDR	DDR_IO	DDR_D12	I/O	–	–	–	–	–	–	–	–	I, HiZ
H11	VDDIODDR	DDR_IO	DDR_D13	I/O	–	–	–	–	–	–	–	–	I, HiZ
B14	VDDIODDR	DDR_IO	DDR_D14	I/O	–	–	–	–	–	–	–	–	I, HiZ
H9	VDDIODDR	DDR_IO	DDR_D15	I/O	–	–	–	–	–	–	–	–	I, HiZ
A17	VDDIODDR	DDR_IO	DDR_DQM0	O	–	–	–	–	–	–	–	–	O, LOW
H14	VDDIODDR	DDR_IO	DDR_DQM1	O	–	–	–	–	–	–	–	–	O, LOW
H15	VDDIODDR	DDR_IO	DDR_DQS0	I/O	–	–	–	–	–	–	–	–	O, LOW
F15	VDDIODDR	DDR_IO	DDR_DQS1	I/O	–	–	–	–	–	–	–	–	O, LOW
J15	VDDIODDR	DDR_IO	DDR_DQSN0	I/O	–	–	–	–	–	–	–	–	O, HIGH
F14	VDDIODDR	DDR_IO	DDR_DQSN1	I/O	–	–	–	–	–	–	–	–	O, HIGH
C9	VDDIODDR	DDR_IO	DDR_CS	O	–	–	–	–	–	–	–	–	O, LOW
B10	VDDIODDR	DDR_IO	DDR_CLK	O	–	–	–	–	–	–	–	–	O
B11	VDDIODDR	DDR_IO	DDR_CLKN	O	–	–	–	–	–	–	–	–	O
D9	VDDIODDR	DDR_IO	DDR_CKE	O	–	–	–	–	–	–	–	–	O, LOW
A10	VDDIODDR	DDR_IO	DDR_RAS	O	–	–	–	–	–	–	–	–	O, LOW
A11	VDDIODDR	DDR_IO	DDR_CAS	O	–	–	–	–	–	–	–	–	O, LOW
C12	VDDIODDR	DDR_IO	DDR_WE	O	–	–	–	–	–	–	–	–	O, LOW
D11	VDDIODDR	DDR_IO	DDR_BA0	O	–	–	–	–	–	–	–	–	O, LOW
D10	VDDIODDR	DDR_IO	DDR_BA1	O	–	–	–	–	–	–	–	–	O, LOW
E10	VDDIODDR	DDR_IO	DDR_BA2	O	–	–	–	–	–	–	–	–	O, LOW
G10	VDDIODDR	Reference	DDR_CALN	I	–	–	–	–	–	–	–	–	I
E14	GNDIODDR	Reference	DDR_CALP	I	–	–	–	–	–	–	–	–	I
G14	VDDIODDR/2	Reference	DDR_VREF	I	–	–	–	–	–	–	–	–	I
P14	VBG	VBG	VBG	I	–	–	–	–	–	–	–	–	I
R12	VDDANA	Reference	ADCVREF	I	–	–	–	–	–	–	–	–	I
R16	VDDUTMII	USBHS	HHSDPC	I/O	–	–	–	–	–	–	–	–	O, PD
R17	VDDUTMII	USBHS	HHSDMC	I/O	–	–	–	–	–	–	–	–	O, PD
U17	VDDUTMII	USBHS	HHSDPB	I/O	–	–	–	–	–	–	–	–	O, PD
T17	VDDUTMII	USBHS	HHSDMB	I/O	–	–	–	–	–	–	–	–	O, PD
U16	VDDUTMII	USBHS	HHSDPA	I/O	DHSDP	–	–	–	–	–	–	–	O, PD

Table 3-2. LFBGA289 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
T16	VDDUTMII	USBHS	HHSDMA	I/O	DHSDM	–	–	–	–	–	–	–	O, PD
T6	VDDBU	Power Supply	VDDBU	I	–	–	–	–	–	–	–	–	I
U6	GNDBU	Ground	GNDBU	I	–	–	–	–	–	–	–	–	I
J6	VDDCORE	Power Supply	VDDCORE	I	–	–	–	–	–	–	–	–	I
E9 F9 F10 J7 K11	GNDCORE	Ground	GNDCORE	I	–	–	–	–	–	–	–	–	I
H6 H7 J11 N9	VCCCORE	Power Supply	VCCCORE	I	–	–	–	–	–	–	–	–	I
D13 E11 F11 G13	VDDIODDR	Power Supply	VDDIODDR	I	–	–	–	–	–	–	–	–	I
D14 E12 E13 F13	GNDIODDR	Ground	GNDIODDR	I	–	–	–	–	–	–	–	–	I
M6 M7	VDDIOM	Power Supply	VDDIOM	I	–	–	–	–	–	–	–	–	I
M9 N11	GNDIOM	Ground	GNDIOM	I	–	–	–	–	–	–	–	–	I
B9 D6 D7 E6 E8	GNDIOP	Ground	GNDIOP	I	–	–	–	–	–	–	–	–	I
G8 H8 J8	VDDIOP	Power Supply	VDDIOP	I	–	–	–	–	–	–	–	–	I
A6 A7 A8 A9 B6 B7 B8 C6 C7 C8 D5 D8 E7 F7 F8 G6 G7	GNDIOP	Ground	GNDIOP	I	–	–	–	–	–	–	–	–	I
P13	VDDUTMIC	Power Supply	VDDUTMIC	I	–	–	–	–	–	–	–	–	I

Table 3-2. LFBGA289 Pin Description (Continued)

Pin	Power Rail	I/O Type	Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
			Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
L13 M13	VDDUTMII	Power Supply	VDDUTMII	I	–	–	–	–	–	–	–	–	I
N12	GNDUTMI	Ground	GNDUTMI	I	–	–	–	–	–	–	–	–	I
U14	VDDPLLA	Power Supply	VDDPLLA	I	–	–	–	–	–	–	–	–	I
T14	GNDPLL	Ground	GNDPLL	I	–	–	–	–	–	–	–	–	I
P12	VDDOSC	Power Supply	VDDOSC	I	–	–	–	–	–	–	–	–	I
M12	GNDOSC	Ground	GNDOSC	I	–	–	–	–	–	–	–	–	I
G9 L10	VDDANA	Power Supply	VDDANA	I	–	–	–	–	–	–	–	–	I
N10	GNDANA	Ground	GNDANA	I	–	–	–	–	–	–	–	–	I
N15	VDDFUSE	Power Supply	VDDFUSE	I	–	–	–	–	–	–	–	–	I
M14	GNDFUSE	Ground	GNDFUSE	I	–	–	–	–	–	–	–	–	I
K10	–	Not connected	–	–	–	–	–	–	–	–	–	–	–

3.3 Input/Output Description

Table 3-3. SAMA5D4 I/O Type Description

I/O Type	Voltage Range	Analog	Pull-up		Pull-down		Schmitt Trigger (2)
			Type (2)	Typ Value (Ω)	Type (2)	Typ Value (Ω)	
GPIO	3.0–3.6V	—	Switchable	(1)	Switchable	(1)	Switchable
GPIO_CLK	3.0–3.6V	—	Switchable	(1)	Switchable	(1)	Switchable
GPIO_CLK2	3.0–3.6V	—	Switchable	(1)	Switchable	(1)	Switchable
GPIO_ANA	3.0–3.6V	I	Switchable	(1)	—	(1)	Switchable
EBI	1.65–1.95V, 3.0–3.6V	—	Switchable	(1)	Switchable	(1)	—
RST	3.0–3.6V	—	Reset State	100K	Reset State	100K	Reset State
SYSC	1.65–3.6V	—	Reset State	100K	Reset State	15K	Reset State
USBHS	3.0–3.6V	I/O	—	—	—	—	—
CLOCK	1.65–3.6V	I/O	—	—	—	—	—
PIOBU	1.88–2.12V	—	Switchable	150K	Switchable	150K	Switchable
DIB	3.0–3.6V	I/O	—	(1)	—	(1)	—

Notes: 1. Refer to Section 55.2 “DC Characteristics”.

2. When “Reset State” is indicated, the configuration is defined by the “Reset State” column of the pin description tables (refer to Table 3-1 and Table 3-2).

Table 3-4. SAMA5D4 I/O Type Assignment and Frequency

I/O Type	I/O Frequency (MHz)	Load (pF)	Fan-out	Drive Control	Signal Name
GPIO	—	—	—	High/Medium/Low	All PIO lines except the lines indicated further on in this table
MCI_CLK	—	—	—	High/Medium/Low	MCI0CK, MCI1CK
GPIO_CLK	—	—	—	High/Medium/Low	SPI0CK, SPI1CK, ETXCLK, ERXCLK
GPIO_CLK2	—	—	—	High/Medium/Low	LCDPCK
GPIO_ANA	—	—	—	Fixed to Medium	ADx
EBI	—	—	—	High/Medium/Low 1.8V/3.3V	All EBI signals
DDR_IO	—	—	—	High/Medium/Low	All DDR signals
RST	—	—	—	Fixed to Low	NRST, NTRST, RST
JTAG	—	—	—	Fixed to Medium	TCK, TDI, TMS, TDO
SYSC	—	—	—	No	WKUP, SHDN, JTAGSEL, TST
VBG	—	—	—	No	VBG
USBHS	480	20	—	No	HHSDPC, HHSDPB, HHSDPA/DHSDP, HHSDMC, HHSDMB, HHSDMA/DHSDM
CLOCK	50	50	—	No	XIN, XOUT, XIN32, XOUT32
PIOBU	—	—	—	No	PIOBUx

4. Power Considerations

4.1 Power Supplies

Table 4-1 defines the different power supplies rails and the estimated power consumption at typical voltage.

All 3.3V power rails are to be established prior to VDDCORE and must always be present. Specific power sequences ensure reliable operation of the device and avoid unwanted security events.

Table 4-1. Power Supplies

Name	Voltage Range, Nominal	Associated Ground	Powers
VDDCORE	1.62–1.98V, 1.8V	GNDCORE	Regulator that generates core power supply on VCCCORE 10 µF decoupling capacitor is to be connected to VCCCORE MUST BE ESTABLISHED AFTER VDDIOP OR AT THE SAME TIME
VCCCORE	1.1–1.32V, 1.2V	GNDCORE	Core
VDDIODDR	1.70–1.90V, 1.8V	GNDIODDR	DDR2 Interface I/O lines
	1.14–1.30V, 1.2V		LP-DDR2 Interface I/O lines
VDDIOM	1.65–1.95V, 1.8V 3.0–3.6V, 3.3V	GNDIOM	NAND and HSMC Interface I/O lines
VDDIOP (1)	3.0–3.6V, 3.3V	GNDIOP	Peripherals I/O lines MUST BE ESTABLISHED PRIOR TO VDDCORE
VDDBU	1.88V–2.6V, 2V	GNDBU	Slow Clock oscillator, the internal 64 kHz RC and a part of the System Controller MUST BE ESTABLISHED FIRST
VDDUTMIC	1.1–1.32V, 1.2V	GNDUTMI	USB device and host UTMI+ core and the UTMI PLL MUST be connected to VCCCORE
VDDUTMII	3.0–3.6V, 3.3V	GNDUTMI	USB device and host UTMI+ interface
VDDPLLA	1.1–1.32V, 1.2V	GNDPLL	PLLA cell MUST be connected to VCCCORE
VDDOSC	3.0V–3.6V, 3.3V	GNDOSC	Main Oscillator cell
VDDANA (1)	3.0–3.6V, 3.3V	GNDANA	Analog parts MUST be connected to VDDIOP with filtering
VDDFUSE	2.25–2.75V, 2.5V	GNDFUSE	Fuse box for programming VDDFUSE must be 2.5V or 0V and must not be left floating

Notes: 1. VDDIOP and VDDANA must rise at the same time.

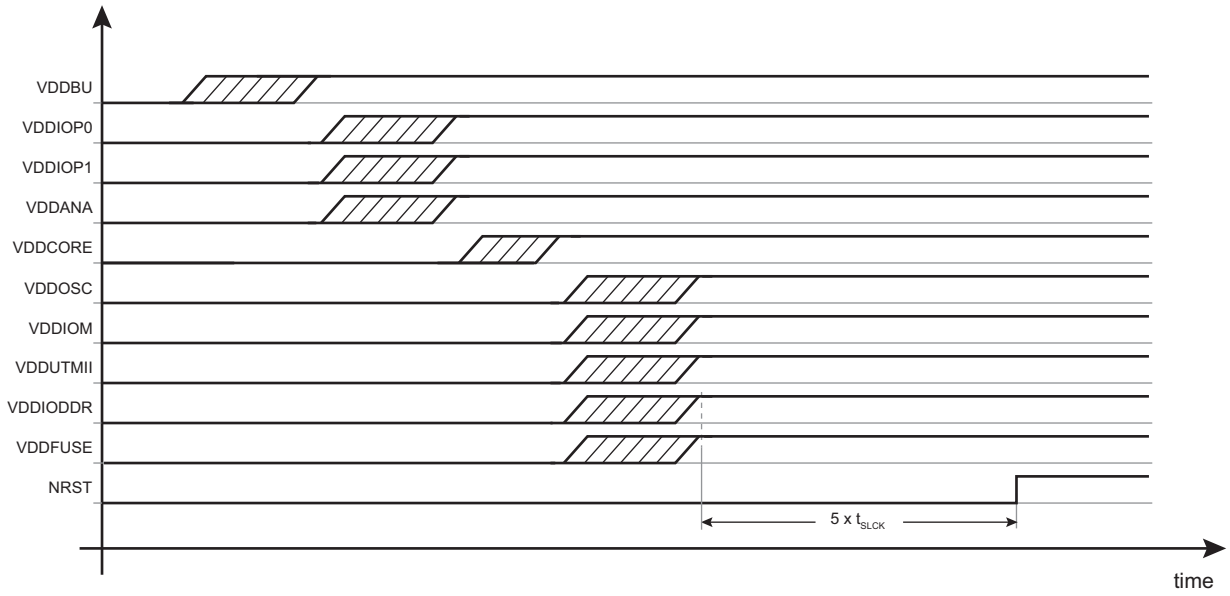
4.2 Powerup Considerations

VDDBU must be set first and for a permanent duration.

The user must maintain NRST at ‘L’ prior to switching on the power supplies. Then VDDIOP and VDDANA are to be switched on, followed by VDDCORE. Afterward, other power supplies can be switched on. After a delay of five SLCK periods, the user can assert NRST to ‘H’ and make the system start.

Figure 4-1 illustrates the SAMA5D4 powerup sequence.

Figure 4-1. Recommended Powerup Sequence



4.3 Shut-down Considerations

When the SHDN pin is asserted, NRST must be maintained at ‘L’ prior to switching off the power supplies. After a delay of five SLCK periods, VDDPLL, then VDDCORE, then VDDIOP and VDDANA can be switched off. Afterward, other power supplies can be switched off.

VDDBU must never be switched off when other supplies are on.

4.4 Wakeup Considerations

When SHDN is rising, NRST is to be maintained at ‘L’ prior to switching on the power supplies. Then VDDIOP and VDDANA are to be switched on, followed by VDDCORE and VDDPLL. Afterward, other power supplies can be switched on. After a delay of five SLCK periods, the user can assert NRST to ‘H’ and make the system wakeup.

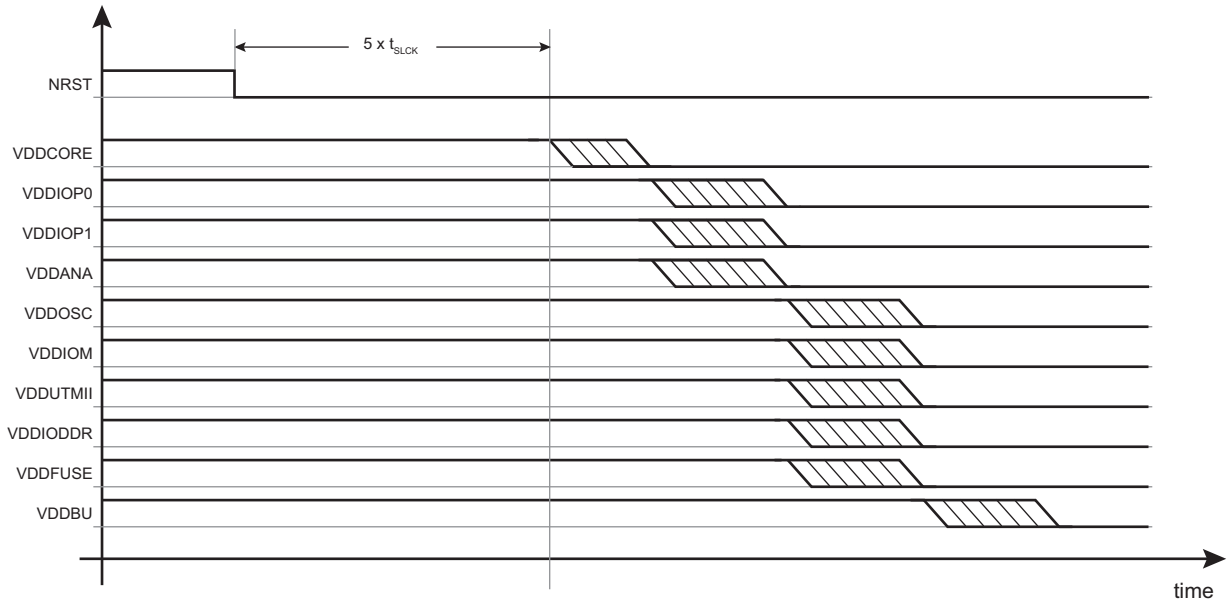
4.5 Powerdown Considerations

The user must maintain NRST at ‘L’ prior to switching off the power supplies. After a delay of five SLCK periods, the user can switch off VDDCORE, then VDDIOP and VDDANA. Afterward, other power supplies can be switched off.

VDDBU must never be switched when other supplies are on.

Figure 4-2 illustrates the SAMA5D4 powerdown sequence.

Figure 4-2. Recommended Powerdown Sequence



4.6 Power-on Reset

The SAMA5D4 embeds several Power-On Resets (POR) to ensure that the power supply is switched on when the reset is released. These PORs are dedicated to VDDBU, VDDIOP and VDDCORE respectively.

4.7 Programmable I/O Lines and Current Drive

4.7.1 DDR2 Bus interface

16-bit or 32-bit wide interface, supporting:

- 16-bit or 32-bit DDR2/LPDDR/LPDDR2

The DDR2/LPDDR/LPDDR2 I/Os embeds an automatic impedance matching control to avoid overshoots and to reach the best performances according to the bus load and external memories.

Two specific analog inputs, DDR_CALP and DDR_CALN are used to calibrate all the DDR I/Os.

4.7.2 LP-DDR2 Power Fail Management

The DDR controller (MPDDRC) allows to manage the LPDDR memory when an uncontrolled power off occurs.

The DDR power rail must be monitored externally and generate an interrupt when a power fail condition is triggered. The interrupt handler must apply the sequence defined in the MPDDRC Low-power Register by setting the bit LPDDR2_PWOFF (LPDDR2 Power Off Bit).

4.7.3 External Bus Interface

16-bit wide interface, working at MCK/2, supporting:

- Static Memories
- NAND Flash with Multi-bit ECC

The EBI I/Os accept three drive level (LOW, MEDIUM, HIGH) allowing to avoid overshoots and give the best performances according to the bus load and external memories voltage.

The drive levels are configured line by line with the LINEx field in the PIO I/O Drive Register x (PIO_DRIVER1 and PIODRIVER2).

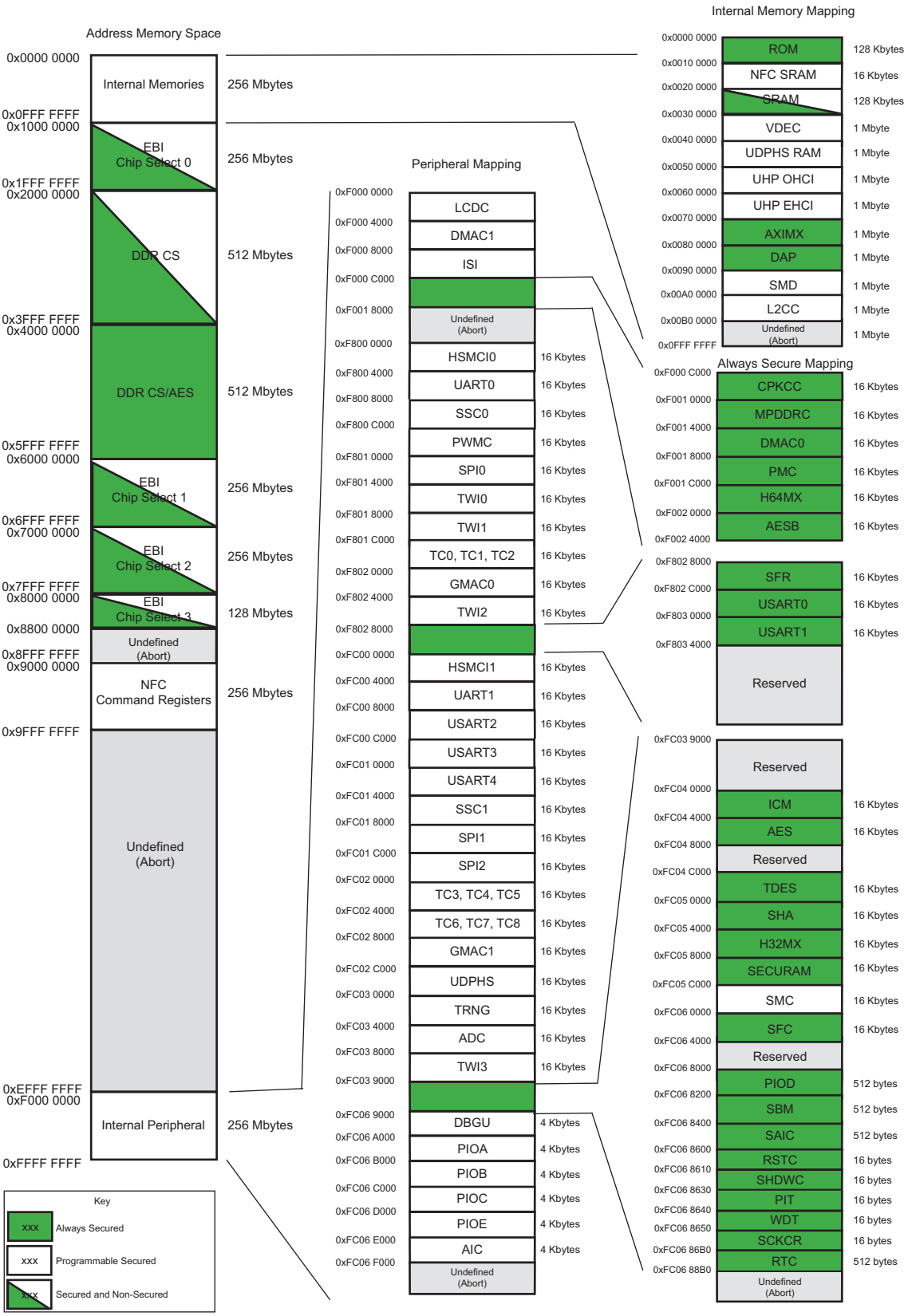
At reset, the selected drive is low. The user must make sure to program the correct drive according to the device load.

4.8 I/O Drive Selection

The aim of this control is to adapt the signal drive to the frequency. The general purpose I/O lines can drive high speed or low speed signals depending on the PIO multiplexing. To reduce the overshoots and improve the EMI behavior, the I/Os feature a drive control which can be enabled in the PIO user interface. The PIO controller embeds drive control registers. Two bits per I/O allow to select one drive from [High, Medium, Low] list.

5. Memories

Figure 5-1. Memory Mapping



5.1 Embedded Memory

5.1.1 Scrambled Internal SRAM

The SAMA5D4 product embeds a total of 128 Kbytes of scrambled high-speed SRAM. After reset and until the Remap command is performed, SRAM is accessible at the address: 0x0020 0000. After remap of AXI Bus Matrix, SRAM is also available at the address 0x0.

5.1.2 Secured Backup SRAM

The device embeds secure memories (8 Kbytes of SRAM) which are dedicated to the storage of sensitive data. The secure backup SRAM is described in the document “Secure Box Module (SBM)”, Atmel literature No. 11254. This document is available under Non-Disclosure Agreement (NDA). Contact an Atmel Sales Representative for further details.

5.1.3 Scrambled Internal ROM

The product embeds one 128-Kbyte secured scrambled internal ROM mapped at address 0 after reset. The ROM contains a standard and a secure bootloader as well as the BCH (Bose, Chaudhuri and Hocquenghem) code tables for NAND Flash ECC correction.

The standard bootloader supports booting from:

- 8-bit NAND Flash with ECC management
- SPI Serial Flash
- SDCARD
- EMMC
- TWI EEPROM

The boot sequence can be selected using the boot order facility (Boot Select Control Register). The internal ROM embeds Galois field tables that are used to compute NAND Flash ECC. Refer to Figure 12-9 “Galois Field Table Mapping” in Section 12. “Standard Boot Strategies”.

5.1.4 Boot Strategies

For standard boot strategies, refer to Section 12. “Standard Boot Strategies”.

For secure boot strategies, refer to the application note “SAMA5D4x Secure Boot Strategy”, Atmel literature No. 11295 (NDA required).

5.2 External Memory

The SAMA5D4 offers connection to a wide range of external memories or to parallel peripherals.

5.2.1 Supported Memories on DDR2/LPDDR/LPDDR2 Interface

- 16-bit or 32-bit external interface
- 512 Mbytes of address space on DDR CS and DDR/AES CS in 32-bit mode
- 256 Mbytes of address space on DDR CS and DDR/AES CS in 16-bit mode
- Supports 16-bit or 32-bit 8-banks DDR2, LPDDR and LPDDR2 memories
- Automatic drive level control
- Multi-port
- Dynamic scrambling
- The port 0 of this interface has an embedded automatic AES encryption and decryption mechanism (refer to Section 52. “Advanced Encryption Standard Bridge (AESB)”). Writing to or reading from the address 0x40000000 may trigger the encryption or decryption mechanism depending on the AESB on External Memories configuration.
- TrustZone: The multi-port feature of this interface implies TrustZone configuration constraints. Refer to Section 15.12 “TrustZone Extension to AHB and APB” for more details.

5.2.2 Supported Memories on Static Memories and NAND Flash Interfaces

The Static Memory Controller is dedicated to interfacing external memory devices:

- Asynchronous SRAM-like memories and parallel peripherals
- NAND Flash (MLC and SLC) 8-bit data path

The Static Memory Controller is able to drive up to four chip selects. NCS3 is dedicated to the NAND Flash control.

The HSMC embeds the NAND Flash Controller (NFC). The NFC can handle automatic transfers, sending the commands and address cycles to the NAND Flash and transferring the content of the page (for read and write) to the NFC SRAM. It minimizes the CPU overhead.

In order to improve overall system performance, the DATA phase of the transfer can be DMA assisted. The static memory embeds the NAND Flash Error Correcting Code Controller with the following features:

- Algorithm based on BCH codes
- Supports also SLC 1-bit (BCH 2-bit), SLC 4-bit (BCH 4-bit)
- Programmable Error Correcting Capability
  - 2-bit, 4-bit, 8-bit and 16-bit errors for 512 bytes/sector (4 Kbyte page)
  - 24-bit error for 1024 bytes/sector (8 Kbyte page)
- Programmable sector size: 512 bytes or 1024 bytes
- Programmable number of sector per page: 1, 2, 4 or 8 blocks of data per page
- Programmable spare area size
- Supports spare area ECC protection
- Supports 8-Kbyte page size using 1024 bytes/sector and 4-Kbyte page size using 512 bytes/sector
- Error detection is interrupt driven
- Provides hardware acceleration for error location
- Finds roots of error-locator polynomial
- Programmable number of roots
- Dynamic scrambling

6. Real-time Event Management

The events generated by 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.

6.1 Embedded Characteristics

- Timers, PWM, IO peripherals generate event triggers which are directly routed to event managers such as ADC, for example, to start measurement/conversion without processor intervention.
- UART, USART, SPI, TWI, PWM, HSMCI, AES, ADC, PIO, Timer (capture mode) also generate event triggers directly connected to DMA Controller (XDMAC0 or XDMAC1) for data transfer without processor intervention.
- PWM safety events (faults) are in combinational form and directly routed from event generators (ADC, PMC, Timer) to PWM module.
- PMC safety event (clock failure detection) can be programmed to switch the MCK on reliable main RC internal clock without processor intervention.

Table 6-1. Real-time Event Mapping List

Function	Application(s)	Description	Event Source	Event Destination
Safety	General-purpose	Automatic switch to reliable main RC oscillator in case of main crystal clock failure(1)	Power Management Controller (PMC)	PMC
	General-purpose, motor control	Puts the PWM outputs in Safe mode (main crystal clock failure detection)(1)(2)	Power Management Controller (PMC)	Pulse Width Modulation (PWM)
	Motor control	Puts the PWM outputs in Safe mode (Overspeed, Overcurrent detection, etc.)(2)(3)	Analog-to-Digital-Converter (ADC)	
	Motor control	Puts the PWM Outputs in Safe mode (Overspeed, Overcurrent detection, etc.)(2)(4)	Timer Counter Block 0 (channels TC0,TC1,TC2)	
	General-purpose, motor control	Puts the PWM outputs in Safe mode (general purpose fault inputs)(2)	Two IOs (PWM_FI0 and PWM_FI1)	
Measurement trigger	General-purpose	Trigger source selection in ADC(5)	Timer Counter Block 0 (TIOA0,TIOA1,TIOA2)	ADC
	Motor control	ADC-PWM synchronization(6)(7) Trigger source selection in ADC(5)	PWM Event Line 0 and 1	
	General-purpose	Trigger source selection in ADC(5)	ADTRG	

Notes:
1. Refer to “Main Crystal Oscillator Failure Detection” in Section 26. “Power Management Controller (PMC)”.
2. Refer to “Fault Inputs” and “Fault Protection” in Section 46. “Pulse Width Modulation Controller (PWM)”.
3. Refer to “Fault Output” in Section 47. “Analog-to-Digital Converter (ADC)”.
4. Refer to “Fault Mode” in Section 45. “Timer Counter (TC)”.
5. Refer to “Conversion Triggers” and the “ADC Mode Register” (ADC_MR) in Section 47. “Analog-to-Digital Converter (ADC)”.
6. Refer to “PWM Comparison x Value Register” (PWM_CMPVx) in Section 46. “Pulse Width Modulation Controller (PWM)”.
7. Refer to “PWM Comparison Units” and “PWM Event Lines” in Section 46. “Pulse Width Modulation Controller (PWM)”.

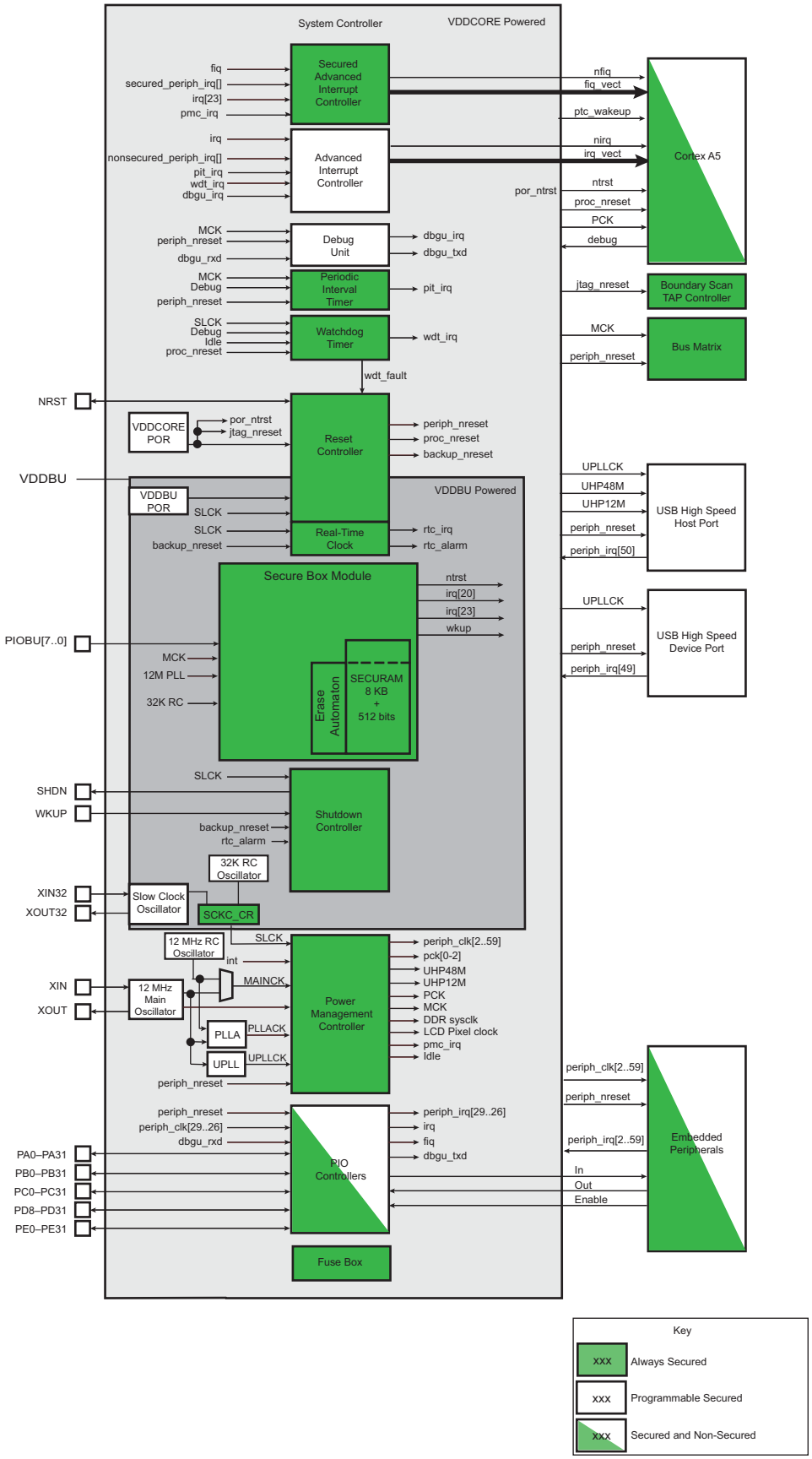
7. System Controller

The System Controller is a set of peripherals handling key elements of the system, such as power, resets, clocks, time, interrupts, watchdog, etc.

The System Controller's peripherals are all mapped between addresses 0xFC06 0000 and 0xFC06 F000.

Figure 7-1 shows the System Controller block diagram.

Figure 7-1. System Controller Block Diagram



7.1 Chip Identification

- Chip ID: 0x8A5C07Cx
- SAMA5D41 Ext ID: 0x1
- SAMA5D42 Ext ID: 0x2
- SAMA5D43 Ext ID: 0x3
- SAMA5D44 Ext ID: 0x4
- Boundary JTAG ID: 0x05B3903F
- Debug Port JTAG IDCODE: 0x4BA00477
- Debug Port Serial Wire IDCODE: 0x2BA01477

8. Peripherals

8.1 Peripheral Mapping

As shown in Figure 5-1 “Memory Mapping”, the peripherals are mapped in the upper 256 Mbytes of the address space between the addresses 0xF000 0000 and 0xFFFF FFFF.

Each user peripheral is allocated 16 Kbytes of the address space.

8.2 Peripheral Identifiers

In the following table, AS stands for “Always Secured” and PS stands for “Programmable Secured”.

Table 8-1. Peripheral Identifiers

Instance ID	Instance Name	Instance Description	External Interrupt	Wired-OR Interrupt	Clock Type	Security Type	In Matrix
0	SAIC	FIQ Interrupt ID	FIQ	–	MCK2	AS	H32MX
1	SYS	System Controller	–	PMC, RSTC, RTC, SHDWC	MCK2	AS	H32MX
2	ARM	Performance Monitor Unit (PMU)	–	–	PCK	AS	H64MX
3	PIT	Periodic Interval Timer	–	–	MCK2	AS	H32MX
4	WDT	Watchdog Timer	–	–	MCK2	AS	H32MX
5	PIOD	Parallel I/O Controller D	–	–	PCLOCK_LS	AS	H32MX
6	USART0	Universal Synchronous Asynchronous Receiver Transceiver 0	–	–	PCLOCK_LS	AS	H32MX
7	USART1	Universal Synchronous Asynchronous Receiver Transceiver 1	–	–	PCLOCK_LS	AS	H32MX
8	XDMAC0	DMA Controller 0	–	–	HCLOCK_HS + PCLOCK_HS	AS	H64MX
9	ICM	Integrity Check Monitor	–	–	PCLOCK_LS	AS	H32MX
10	CPKCC	Classic Public Key Crypto Controller	–	–	PCLOCK_HS	AS	H64MX
12	AES	Advanced Encryption Standard	–	–	PCLOCK_LS	AS	H32MX
13	AESB	AES Bridge	–	–	PCLOCK_HS	AS	H64MX
14	TDES	Triple Data Encryption Standard	–	–	PCLOCK_LS	AS	H32MX
15	SHA	SHA Signature	–	–	PCLOCK_LS	AS	H32MX
16	MPDDRC	MPDDR Controller	–	–	HCLOCK_HS	AS	H64MX
17	MATRIX1	H32MXMX, 32-bit AHB Matrix	–	–	PCLOCK_LS	AS	H32MX
18	MATRIX0	H64MX, 64-bit AHB Matrix	–	–	PCLOCK_HS	AS	H64MX
19	VDEC	Video Decoder	–	–	PCLOCK_HS	PS	H64MX
20	SBM	Secure Box Module	–	–	MCK2	AS	H32MX
22	HSMC	Multi-bit ECC Interrupt	–	–	PCLOCK_LS	PS	H32MX
23	PIOA	Parallel I/O Controller A	–	–	PCLOCK_LS	PS	H32MX
24	PIOB	Parallel I/O Controller B	–	–	PCLOCK_LS	PS	H32MX

Table 8-1. Peripheral Identifiers (Continued)

Instance ID	Instance Name	Instance Description	External Interrupt	Wired-OR Interrupt	Clock Type	Security Type	In Matrix
25	PIOC	Parallel I/O Controller C	–	–	PCLOCK_LS	PS	H32MX
26	PIOE	Parallel I/O Controller E	–	–	PCLOCK_LS	PS	H32MX
27	UART0	Universal Asynchronous Receiver Transmitter 0	–	–	PCLOCK_LS	PS	H32MX
28	UART1	Universal Asynchronous Receiver Transmitter 1	–	–	PCLOCK_LS	PS	H32MX
29	USART2	Universal Synchronous Asynchronous Receiver Transceiver 2	–	–	PCLOCK_LS	PS	H32MX
30	USART3	Universal Synchronous Asynchronous Receiver Transceiver3	–	–	PCLOCK_LS	PS	H32MX
31	USART4	Universal Synchronous Asynchronous Receiver Transceiver 4	–	–	PCLOCK_LS	PS	H32MX
32	TWI0	Two-wire Interface 0	–	–	PCLOCK_LS	PS	H32MX
33	TWI1	Two-wire Interface 1	–	–	PCLOCK_LS	PS	H32MX
34	TWI2	Two-wire Interface 2	–	–	PCLOCK_LS	PS	H32MX
35	HSMCI0	High Speed Multimedia Card Interface 0	–	–	PCLOCK_LS	PS	H32MX
36	HSMCI1	High Speed Multimedia Card Interface 1	–	–	PCLOCK_LS	PS	H32MX
37	SPI0	Serial Peripheral Interface 0	–	–	PCLOCK_LS	PS	H32MX
38	SPI1	Serial Peripheral Interface 1	–	–	PCLOCK_LS	PS	H32MX
39	SPI2	Serial Peripheral Interface 2	–	–	PCLOCK_LS	PS	H32MX
40	TC0	Timer Counter 0 (ch. 0, 1, 2)	–	–	PCLOCK_LS	PS	H32MX
41	TC1	Timer Counter 1 (ch. 3, 4, 5)	–	–	PCLOCK_LS	PS	H32MX
42	TC2	Timer Counter 2 (ch. 6, 7, 8)	–	–	PCLOCK_LS	PS	H32MX
43	PWM	Pulse Width Modulation Controller	–	–	PCLOCK_LS	PS	H32MX
44	ADC	Touchscreen ADC Controller	–	–	PCLOCK_LS	PS	H32MX
45	DBGU	Debug Unit	–	–	PCLOCK_LS	PS	H32MX
46	UHPHS	USB Host High Speed	–	–	HCLOCK_LS	PS	H32MX
47	UDPHS	USB Device High Speed	–	–	HCLOCK_LS + PCLOCK_LS	PS	H32MX
48	SSC0	Synchronous Serial Controller 0	–	–	PCLOCK_LS	PS	H32MX
49	SSC1	Synchronous Serial Controller 1	–	–	PCLOCK_LS	PS	H32MX
50	XDMAC1	DMA Controller 1	–	–	HCLOCK_HS + PCLOCK_HS	Non-Secured	H64MX
51	LCDC	LCD Controller	–	–	HCLOCK_HS	PS	H64MX
52	ISI	Camera Interface	–	–	PCLOCK_HS	PS	H64MX

Table 8-1. Peripheral Identifiers (Continued)

Instance ID	Instance Name	Instance Description	External Interrupt	Wired-OR Interrupt	Clock Type	Security Type	In Matrix
53	TRNG	True Random Number Generator	–	–	PCLOCK_LS	PS	H32MX
54	GMAC0	Ethernet MAC 0	–	–	HCLOCK_LS + PCLOCK_LS	PS	H32MX
55	GMAC1	Ethernet MAC 1	–	–	HCLOCK_LS + PCLOCK_LS	PS	H32MX
56	AIC	IRQ Interrupt ID	IRQ	–	MCK2	Non-Secured	H32MX
57	SFC	Fuse Controller	–	–	PCLOCK_LS	AS	H32MX
58	–	Reserved	–	–	–	–	–
59	SECURAM	Secured RAM	–	–	PCLOCK_LS	AS	H32MX
61	SMD	SMD Soft Modem	–	–	SMDCK	PS	H32MX
62	TWI3	Two-Wire Interface 3	–	–	PCLOCK_LS	PS	H32MX
63	–	Reserved	–	–	–		
64	SFR	Special Function Register (1)	–	–	–	AS	H32MX
65	AIC	Advanced Interrupt Controller (1)	–	–	–	Non-Secured	H32MX
66	SAIC	Secured Advanced Interrupt Controller (1)	–	–	–	AS	H32MX
67	L2CC	L2 Cache Controller (1)	–	–	–	PS	H64MX

Notes: 1. For security purposes, there is no matching clock but a peripheral ID only.

8.3 Peripheral Signal Multiplexing on I/O Lines

The SAMA5D4 product features five PIO controllers: PIOA, PIOB, PIOC, PIOD, and PIOE, that multiplex the I/O lines of the peripheral set.

Each line can be assigned to one of three peripheral functions: A, B, or C. The multiplexing tables in the pin description paragraphs define how the I/O lines of the peripherals A, B and C are multiplexed on the PIO Controllers.

Note that some peripheral functions which are output only, might be duplicated within the both tables.

The column “Reset State” indicates whether the PIO Line resets in I/O mode or in peripheral mode. If I/O is mentioned, the PIO line resets in input with the pull-up enabled, so that the device is maintained in a static state as soon as the reset is released. As a result, the bit corresponding to the PIO line in 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 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. Note that the pull-up resistor is also enabled in this case.

8.4 Peripheral Clock Type

The SAMA5D4 series embeds peripherals with the following clock types:

- HCLOCK_HS, HCLOCK_LS: AHB Clocks, managed with the PMC_SCER, PMC_SCDR and PMC_SCSR registers of PMC System Clock
- PCLOCK_HS, PCLOCK_LS: APB Clocks, managed with the PMC_PCER, PMC_PCDR, PMC_PCSR and PMC_PCR registers of Peripheral Clock
- MCK2: This clock cannot be disabled.
- PCK: The Processor Clock is managed with the PMC_SCDR and PMC_SCSR registers of PMC System Clock.

Refer to Table 8-1 “Peripheral Identifiers” for details. In the table, clock type suffixes _HS and _LS refer to H64MX and H32MX, respectively.

9. ARM Cortex-A5

9.1 Description

The ARM Cortex-A5 processor is a high-performance, low-power, ARM macrocell with an L1 cache subsystem that provides full virtual memory capabilities. The Cortex-A5 processor implements the ARMv7 architecture and runs 32-bit ARM instructions, 16-bit and 32-bit Thumb instructions, and 8-bit Java® byte codes in Jazelle® state.

The Cortex-A5 NEON Media Processing Engine (MPE) extends the Cortex-A5 functionality to provide support for the ARM v7 Advanced SIMD v2 and *Vector Floating-Point v4* (VFPv4) instruction sets. The Cortex-A5 NEON MPE provides flexible and powerful acceleration for signal processing algorithms including multimedia such as image processing, video decode/encode, 2D/3D graphics, and audio. Refer to the *Cortex-A5 NEON Media Processing Engine Technical Reference Manual*.

The Cortex-A5 processor includes TrustZone technology to enhance security by partitioning the SoC's hardware and software resources in a Secure world for the security subsystem and a Normal world for the rest, enabling a strong security perimeter to be built between the two. Refer to *Security Extensions overview* in the *Cortex-A5 Technical Reference Manual*. Refer to the *ARM Architecture Reference Manual* for details on how TrustZone works in the architecture.

Note: All ARM publications referenced in this datasheet can be found at www.arm.com.

9.1.1 Power Management

The Cortex-A5 design supports the following main levels of power management:

- Run Mode
- Standby Mode

9.1.1.1 Run Mode

Run mode is the normal mode of operation where all of the processor functionality is available. Everything, including core logic and embedded RAM arrays, is clocked and powered up.

9.1.1.2 Standby Mode

Standby mode disables most of the clocks of the processor, while keeping it powered up. This reduces the power drawn to the static leakage current, plus a small clock power overhead required to enable the processor to wake up from Standby mode. The transition from Standby mode to Run mode is caused by one of the following:

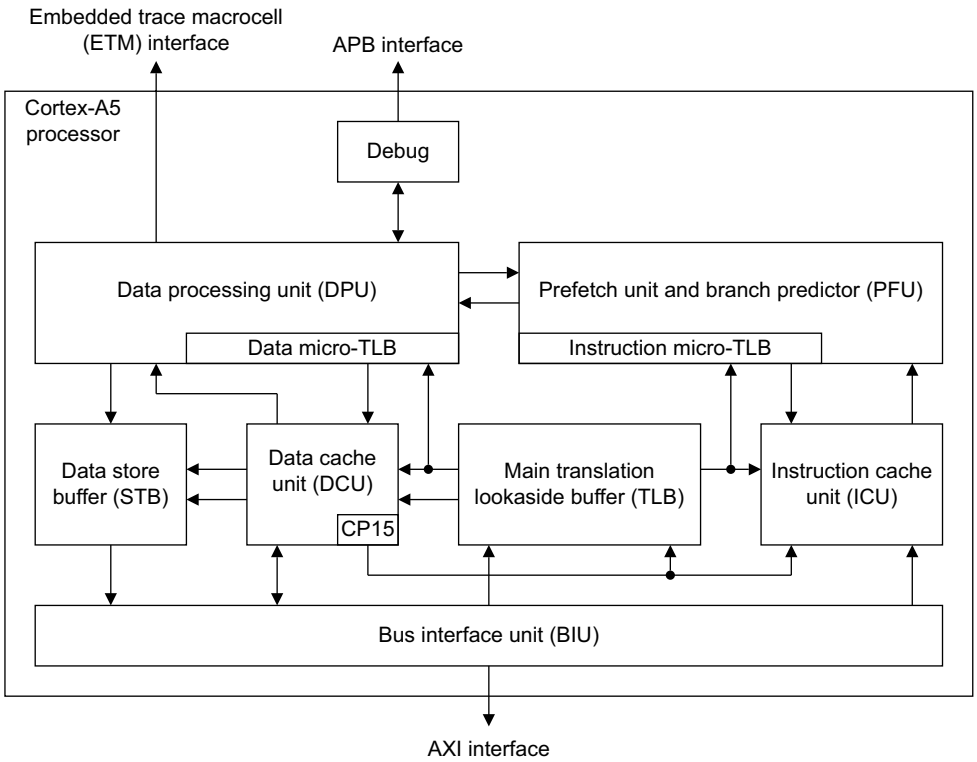
- the arrival of an interrupt, either masked or unmasked
- the arrival of an event, if standby mode was initiated by a Wait for Event (WFE) instruction
- a debug request, when either debug is enabled or disabled
- a reset.

9.2 Embedded Characteristics

- In-order pipeline with dynamic branch prediction
- ARM, Thumb, and ThumbEE instruction set support
- TrustZone security extensions
- Harvard level 1 memory system with a Memory Management Unit (MMU)
- 32 Kbytes Data Cache
- 32 Kbytes Instruction Cache
- 64-bit AXI master interface
- ARM v7 debug architecture
- Media Processing Engine (MPE) with NEON technology
- Jazelle hardware acceleration

9.3 Block Diagram

Figure 9-1. Cortex-A5 Processor Top-level Diagram



9.4 Programmer Model

9.4.1 Processor Operating Modes

The following operating modes are present in all states:

- User mode (USR) is the usual ARM program execution state. It is used for executing most application programs.
- Fast Interrupt (FIQ) mode is used for handling fast interrupts. It is suitable for high-speed data transfer or channel process.
- Interrupt (IRQ) mode is used for general-purpose interrupt handling.
- Supervisor mode (SVC) is a protected mode for the operating system.
- Abort mode (ABT) is entered after a data or instruction prefetch abort.
- System mode (SYS) is a privileged user mode for the operating system.
- Undefined mode (UND) is entered when an undefined instruction exception occurs.
- Monitor mode (MON) is secure mode that enables change between Secure and Non-secure states, and can also be used to handle any of FIQs, IRQs and external aborts. Entered on execution of a Secure Monitor Call (SMC) instruction.

Mode changes may be made under software control, or may be brought about by external interrupts or exception processing. Most application programs execute in User Mode. The non-user modes, known as privileged modes, are entered in order to service interrupts or exceptions or to access protected resources.

9.4.2 Processor Operating States

The processor has the following instruction set states controlled by the T bit and J bit in the CPSR.

- ARM state:

  The processor executes 32-bit, word-aligned ARM instructions.

- Thumb state:

  The processor executes 16-bit and 32-bit, halfword-aligned Thumb instructions.

- ThumbEE state:

  The processor executes a variant of the Thumb instruction set designed as a target for dynamically generated code. This is code compiled on the device either shortly before or during execution from a portable bytecode or other intermediate or native representation.

- Jazelle state:

  The processor executes variable length, byte-aligned Java bytecodes.

The J bit and the T bit determine the instruction set used by the processor. Table 9-1 shows the encoding of these bits.

Table 9-1. CPSR J and T Bit Encoding

J	T	Instruction Set State
0	0	ARM
0	1	Thumb
1	0	Jazelle
1	1	ThumbEE

Changing between ARM and Thumb states does not affect the processor mode or the register contents. Refer to the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for information on entering and exiting ThumbEE state.

9.4.2.1 Switching State

It is possible to change the instruction set state of the processor between:

- ARM state and Thumb state using the BX and BLX instructions.
- Thumb state and ThumbEE state using the ENTERX and LEAVEX instructions.
- ARM and Jazelle state using the BXJ instruction.
- Thumb and Jazelle state using the BXJ instruction.

Refer to the *ARM Architecture Reference Manual* for more information about changing instruction set state.

9.4.3 Cortex-A5 Registers

This view provides 16 ARM core registers, R0 to R15, that include the Stack Pointer (SP), Link Register (LR), and Program Counter (PC). These registers are selected from a total set of either 31 or 33 registers, depending on whether or not the Security Extensions are implemented. The current execution mode determines the selected set of registers, as shown in Table 9-2. This shows that the arrangement of the registers provides duplicate copies of some registers, with the current register selected by the execution mode. This arrangement is described as banking of the registers, and the duplicated copies of registers are referred to as banked registers.

Table 9-2. Cortex-A5 Modes and Registers Layout

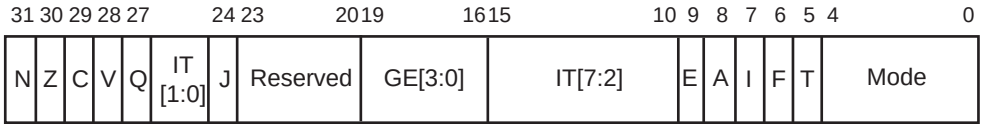
User and System	Monitor	Supervisor	Abort	Undefined	Interrupt	Fast Interrupt
R0	R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7	R7
R8	R8	R8	R8	R8	R8	R8_FIQ
R9	R9	R9	R9	R9	R9	R9_FIQ
R10	R10	R10	R10	R10	R10	R10_FIQ
R11	R11	R11	R11	R11	R11	R11_FIQ
R12	R12	R12	R12	R12	R12	R12_FIQ
R13	R13_MON	R13_SVC	R13_ABT	R13_UND	R13_IRQ	R13_FIQ
R14	R14_MON	R14_SVC	R14_ABT	R14_UND	R14_IRQ	R14_FIQ
PC	PC	PC	PC	PC	PC	PC
CPSR	CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_MON	SPSR_SVC	SPSR_ABT	SPSR_UND	SPSR_IRQ	SPSR_FIQ

	Mode-specific banked registers

The core contains one CPSR, and six SPSRs for exception handlers to use. The program status registers:

- hold information about the most recently performed ALU operation
- control the enabling and disabling of interrupts
- set the processor operating mode

Figure 9-2. Status Register Format



- N: Negative, Z: Zero, C: Carry, and V: Overflow are the four ALU flags
- Q: cumulative saturation flag
- IT: If-Then execution state bits for the Thumb IT (If-Then) instruction
- J: Jazelle bit, refer to the description of the T bit
- GE: Greater than or Equal flags, for SIMD instructions

- E: Endianness execution state bit. Controls the load and store endianness for data accesses. This bit is ignored by instruction fetches.
  - E = 0: Little endian operation
  - E = 1: Big endian operation
- A: Asynchronous abort disable bit. Used to mask asynchronous aborts.
- I: Interrupt disable bit. Used to mask IRQ interrupts.
- F: Fast interrupt disable bit. Used to mask FIQ interrupts.
- T: Thumb execution state bit. This bit and the J execution state bit, bit [24], determine the instruction set state of the processor, ARM, Thumb, Jazelle, or ThumbEE.
- Mode: five bits to encode the current processor mode. The effect of setting M[4:0] to a reserved value is UNPREDICTABLE.

Table 9-3. Processor Mode vs. Mode Field

Mode	M[4:0]
USR	10000
FIQ	10001
IRQ	10010
SVC	10011
MON	10110
ABT	10111
UND	11011
SYS	11111
Reserved	Other

9.4.3.1 CP15 Coprocessor

Coprocessor 15, or System Control Coprocessor CP15, is used to configure and control all the items in the list below:

- Cortex A5
- Caches (ICache, DCache and write buffer)
- MMU
- Security
- Other system options

To control these features, CP15 provides 16 additional registers. Refer to Table 9-4.

Table 9-4. CP15 Registers

Register	Name	Read/Write
0	ID Code(1)	Read/Unpredictable
0	Cache type(1)	Read/Unpredictable
1	Control(1)	Read/Write
1	Security(1)	Read/Write
2	Translation Table Base	Read/Write
3	Domain Access Control	Read/Write
4	Reserved	None

Table 9-4. CP15 Registers (Continued)

Register	Name	Read/Write
5	Data fault Status[1]	Read/Write
5	Instruction fault status	Read/Write
6	Fault Address	Read/Write
7	Cache and MMU Operations[1]	Read/Write
8	TLB operations	Unpredictable/Write
9	Cache lockdown[1]	Read/Write
10	TLB lockdown	Read/Write
11	Reserved	None
12	Interrupts management	Read/Write
12	Monitor vectors	Read-only
13	FCSE PID[1]	Read/Write
13	Context ID[1]	Read/Write
14	Reserved	None
15	Test configuration	Read/Write

Note: 1. This register provides access to more than one register. The register accessed depends on the value of the CRm field or Opcode_2 field.

9.4.4 CP 15 Register Access

CP15 registers can only be accessed in privileged mode by:

- MCR (Move to Coprocessor from ARM Register) instruction is used to write an ARM register to CP15.
- MRC (Move to ARM Register from Coprocessor) instruction is used to read the value of CP15 to an ARM register.

Other instructions such as CDP, LDC, STC can cause an undefined instruction exception.

The assembler code for these instructions is:

```
MCR/MRC{cond} p15, opcode_1, Rd, CRn, CRm, opcode_2.
```

The MCR/MRC instructions bit pattern is shown below:

31	30	29	28	27	26	25	24
cond				1	1	1	0

23	22	21	20	19	18	17	16
opcode_1			L	CRn			

15	14	13	12	11	10	9	8
Rd				1	1	1	1

7	6	5	4	3	2	1	0
opcode_2			1	CRm			

- **CRm[3:0]: Specified Coprocessor Action**

Determines specific coprocessor action. Its value is dependent on the CP15 register used. For details, refer to CP15 specific register behavior.

- **opcode_2[7:5]**

Determines specific coprocessor operation code. By default, set to 0.

- **Rd[15:12]: ARM Register**

Defines the ARM register whose value is transferred to the coprocessor. If R15 is chosen, the result is unpredictable.

- **CRn[19:16]: Coprocessor Register**

Determines the destination coprocessor register.

- **L: Instruction Bit**

0: MCR instruction

1: MRC instruction

- **opcode_1[23:20]: Coprocessor Code**

Defines the coprocessor specific code. Value is c15 for CP15.

- **cond [31:28]: Condition**

9.4.5 Addresses in the Cortex-A5 processor

The Cortex-A5 processor operates using *virtual addresses* (VAs). The *Memory Management Unit* (MMU) translates these VAs into the *physical addresses* (PAs) used to access the memory system. Translation tables hold the mappings between VAs and PAs.

Refer to the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for more information.

When the Cortex-A5 processor is executing in Non-secure state, the processor performs translation table look-ups using the Non-secure versions of the Translation Table Base Registers. In this situation, any VA can only translate into a Non-secure PA. When it is in Secure state, the Cortex-A5 processor performs translation table look-ups using the Secure versions of the Translation Table Base Registers. In this situation, the security state of any VA is determined by the NS bit of the translation table descriptors for that address.

Following is an example of the address manipulation that occurs when the Cortex-A5 processor requests an instruction:

1. The Cortex-A5 processor issues the VA of the instruction as Secure or Non-secure VA accesses according to the state the processor is in.
2. The instruction cache is indexed by the bits of the VA. The MMU performs the translation table look-up in parallel with the cache access. If the processor is in the Secure state it uses the Secure translation tables, otherwise it uses the Non-secure translation tables.
3. If the protection check carried out by the MMU on the VA does not abort and the PA tag is in the instruction cache, the instruction data is returned to the processor.
4. If there is a cache miss, the MMU passes the PA to the AXI bus interface to perform an external access. The external access is always Non-secure when the core is in the Non-secure state. In the Secure state, the external access is Secure or Non-secure according to the NS attribute value in the selected translation table entry. In Secure state, both L1 and L2 translation table walk accesses are marked as Secure, even if the first level descriptor is marked as NS.

9.4.6 Security Extensions Overview

The purpose of the Security Extensions is to enable the construction of a secure software environment. Refer to the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for details of the Security Extensions.

9.4.6.1 System Boot Sequence

CAUTION: The Security Extensions enable the construction of an isolated software environment for more secure execution, depending on a suitable system design around the processor. The technology does not protect the processor from hardware attacks, and care must be taken to be sure that the hardware containing the reset handling code is appropriately secure.

The processor always boots in the privileged Supervisor mode in the Secure state, with the NS bit set to 0. This means that code that does not attempt to use the Security Extensions always runs in the Secure state. If the software uses both Secure and Non-secure states, the less trusted software, such as a complex operating system and application code running under that operating system, executes in Non-secure state, and the most trusted software executes in the Secure state.

The following sequence is expected to be typical use of the Security Extensions:

1. Exit from reset in Secure state.
2. Configure the security state of memory and peripherals. Some memory and peripherals are accessible only to the software running in Secure state.
3. Initialize the secure operating system. The required operations depend on the operating system, and include initialization of caches, MMU, exception vectors, and stacks.

4. Initialize Secure Monitor software to handle exceptions that switch execution between the Secure and Non-secure operating systems.
5. Optionally lock aspects of the secure state environment against further configuration.
6. Pass control through the Secure Monitor software to the non-secure OS with an SMC instruction.
7. Enable the Non-secure operating system to initialize. The required operations depend on the operating system, and typically include initialization of caches, MMU, exception vectors, and stacks.

The overall security of the secure software depends on the system design, and on the secure software itself.

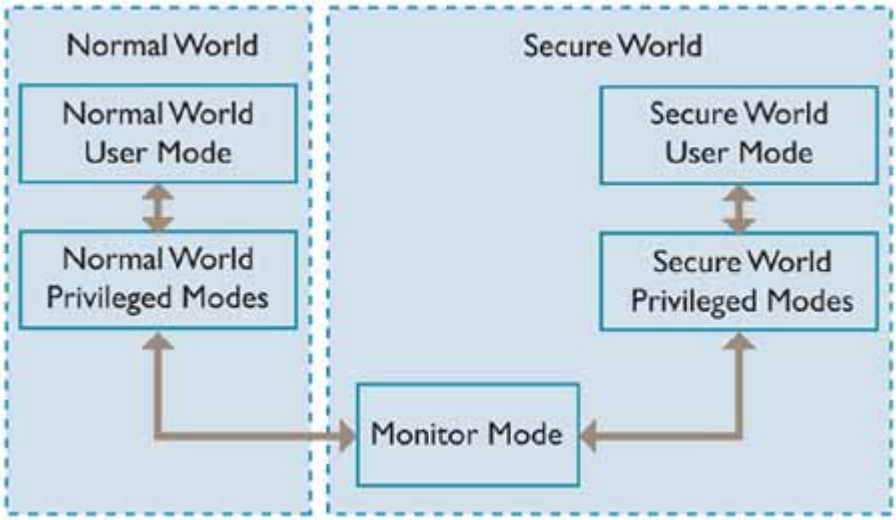
9.4.7 TrustZone

9.4.7.1 Hardware

TrustZone enables a single physical processor core to execute code safely and efficiently from both the Normal world and the Secure world. This removes the need for a dedicated security processor core, saving silicon area and power, and allowing high performance security software to run alongside the Normal world operating environment.

The two virtual processors context switch via a new processor mode called monitor mode when changing the currently running virtual processor.

Figure 9-3. TrustZone Hardware Implementation

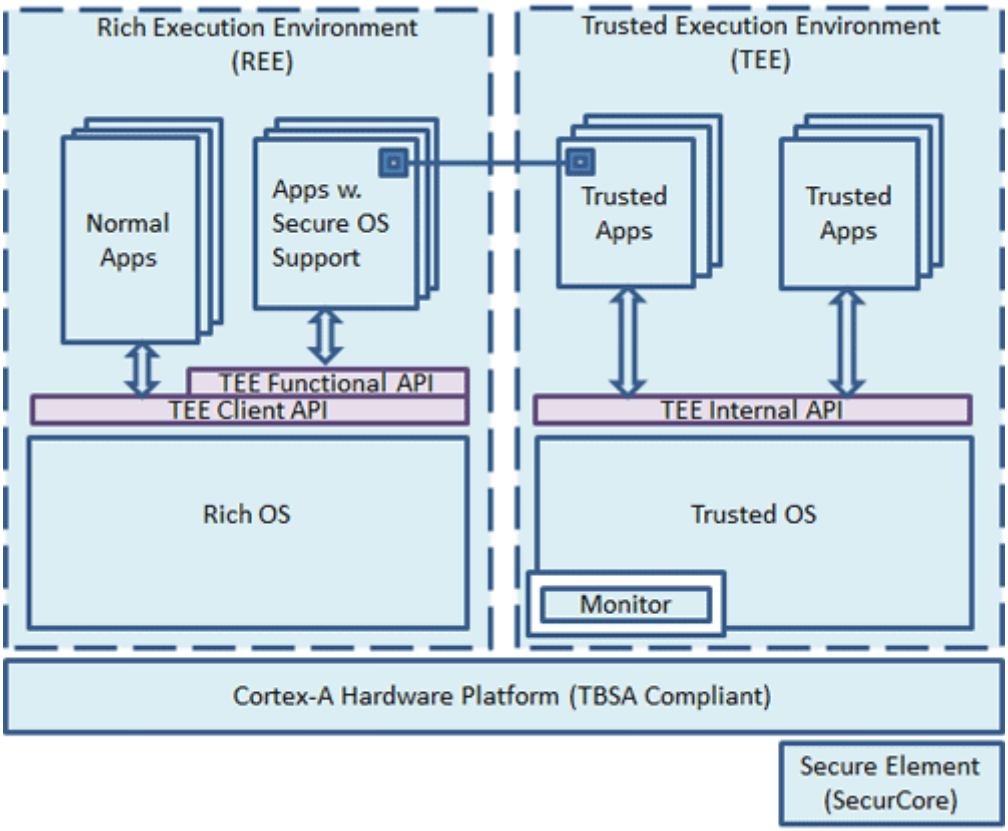


9.4.7.2 Software

The mechanisms by which the physical processor can enter monitor mode from the Normal world are tightly controlled, and are all viewed as exceptions to the monitor mode software. Software executing a dedicated instruction can trigger entry to monitor, the Secure Monitor Call (SMC) instruction, or by a subset of the hardware exception mechanisms. Configuration of the IRQ, FIQ, external Data Abort, and external Prefetch Abort exceptions can cause the processor to switch into monitor mode.

The software that executes within monitor mode is implementation defined, but it generally saves the state of the current world and restores the state of the world at the location to which it switches. It then performs a return-from-exception to restart processing in the restored world.

Figure 9-4. TrustZone Software implementation in a Trusted Execution Environment (TEE)



9.4.7.3 Debug

TrustZone hardware architecture is a security-aware debug infrastructure that can enable control over access to secure world debug, without impairing debug visibility of the Normal world. This is controlled with bits in the Secure Fuse Controller.

Note: Secure debug modes are described in the document “Secure Box Module (SBM)”. This document is available under Non-Disclosure Agreement (NDA). Contact an Atmel Sales Representative for further details.

9.5 Memory Management Unit

9.5.1 About the MMU

The MMU works with the L1 and L2 memory system to translate virtual addresses to physical addresses. It also controls accesses to and from external memory.

The ARM v7 Virtual Memory System Architecture (VMSA) features include the following:

- Page table entries that support:
  - 16 Mbyte supersections. The processor supports supersections that consist of 16 Mbyte blocks of memory.
  - 1 Mbyte sections
  - 64 Kbyte large pages
  - 4 Kbyte small pages
- 16 access domains
- Global and application-specific identifiers to remove the requirement for context switch TLB flushes.
- Extended permissions checking capability.

TLB maintenance and configuration operations are controlled through a dedicated coprocessor, CP15, integrated with the core. This coprocessor provides a standard mechanism for configuring the L1 memory system.

Refer to the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for a full architectural description of the ARMv7 VMSA.

9.5.2 Memory Management System

The Cortex-A5 processor supports the ARM v7 VMSA including the TrustZone security extension. The translation of a Virtual Address (VA) used by the instruction set architecture to a Physical Address (PA) used in the memory system and the management of the associated attributes and permissions is carried out using a two-level MMU.

The first level MMU uses a Harvard design with separate micro TLB structures in the PFU for instruction fetches (IuTLB) and in the DPU for data read and write requests (DuTLB).

A miss in the micro TLB results in a request to the main unified TLB shared between the data and instruction sides of the memory system. The TLB consists of a 128-entry two-way set-associative RAM based structure. The TLB page-walk mechanism supports page descriptors held in the L1 data cache. The caching of page descriptors is configured globally for each translation table base register, TTBRx, in the system coprocessor, CP15.

The TLB contains a hitmap cache of the page types which have already been stored in the TLB.

9.5.2.1 Memory types

Although various different memory types can be specified in the page tables, the Cortex-A5 processor does not implement all possible combinations:

- Write-through caches are not supported. Any memory marked as write-through is treated as Non-cacheable.
- The outer shareable attribute is not supported. Anything marked as outer shareable is treated in the same way as inner shareable.
- Write-back no write-allocate is not supported. It is treated as write-back write-allocate.

Table 9-5 shows the treatment of each different memory type in the Cortex-A5 processor in addition to the architectural requirements.

Table 9-5. Treatment of Memory Attributes

Memory Type Attribute	Shareability	Other Attributes	Notes
Strongly Ordered	—	—	—
Device	Non-shareable	—	—
	Shareable	—	—
Normal	Non-shareable	Non-cacheable	Does not access L1 caches
		Write-through cacheable	Treated as non-cacheable
		Write-back cacheable, write allocate	Can dynamically switch to no write allocate, if more than three full cache lines are written in succession
		Write-back cacheable, no write allocate	Treated as non-shareable write-back cacheable, write allocate
	Inner shareable	Non-cacheable	—
		Write-through cacheable	Treated as inner shareable non-cacheable
		Write-back cacheable, write allocate	Treated as inner shareable non-cacheable unless the SMP bit in the Auxiliary Control Register is set (ACTLR[6] = b1). If this bit is set the area is treated as write-back cacheable write allocate.
		Write-back cacheable, no write allocate	
	Outer shareable	Non-cacheable	Treated as inner shareable non-cacheable
		Write-through cacheable	
		Write-back cacheable, write allocate	Treated as inner shareable non-cacheable unless the SMP bit in the Auxiliary Control Register is set (ACTLR[6] = b1). If this bit is set the area is treated as write-back cacheable write allocate.
		Write-back cacheable, no write allocate	

9.5.3 TLB Organization

TLB Organization is described in the sections that follow:

- Micro TLB
- Main TLB

9.5.3.1 Micro TLB

The first level of caching for the page table information is a micro TLB of 10 entries that is implemented on each of the instruction and data sides. These blocks provide a lookup of the virtual addresses in a single cycle.

The micro TLB returns the physical address to the cache for the address comparison, and also checks the access permissions to signal either a Prefetch Abort or a Data Abort.

All main TLB related maintenance operations affect both the instruction and data micro TLBs, causing them to be flushed. In the same way, any change of the following registers causes the micro TLBs to be flushed:

- Context ID Register (CONTEXTIDR)
- Domain Access Control Register (DACR)
- Primary Region Remap Register (PRRR)
- Normal Memory Remap Register (NMRR)
- Translation Table Base Registers (TTBR0 and TTBR1)

9.5.3.2 Main TLB

Misses from the instruction and data micro TLBs are handled by a unified main TLB. Accesses to the main TLB take a variable number of cycles, according to competing requests from each of the micro TLBs and other implementation-dependent factors.

The main TLB is 128-entry two-way set-associative.

TLB match process

Each TLB entry contains a virtual address, a page size, a physical address, and a set of memory properties. Each is marked as being associated with a particular application space (ASID), or as global for all application spaces. The CONTEXTIDR determines the currently selected application space.

A TLB entry matches when these conditions are true:

- Its virtual address matches that of the requested address.
- Its Non-secure TLB ID (NSTID) matches the Secure or Non-secure state of the MMU request.
- Its ASID matches the current ASID in the CONTEXTIDR or is global.

The operating system must ensure that, at most, one TLB entry matches at any time. The TLB can store entries based on the following block sizes:

Supersections	Describe 16 Mbyte blocks of memory
Sections	Describe 1 Mbyte blocks of memory
Large pages	Describe 64 Kbyte blocks of memory
Small pages	Describe 4 Kbyte blocks of memory

Supersections, sections and large pages are supported to permit mapping of a large region of memory while using only a single entry in the TLB. If no mapping for an address is found within the TLB, then the translation table is automatically read by hardware and a mapping is placed in the TLB.

9.5.4 Memory Access Sequence

When the processor generates a memory access, the MMU:

1. Performs a lookup for the requested virtual address and current ASID and security state in the relevant instruction or data micro TLB.
2. If there is a miss in the micro TLB, performs a lookup for the requested virtual address and current ASID and security state in the main TLB.
3. If there is a miss in main TLB, performs a hardware translation table walk.

The MMU can be configured to perform hardware translation table walks in cacheable regions by setting the IRGN bits in Translation Table Base Register 0 and Translation Table Base Register 1. If the encoding of the IRGN bits is write-back, an L1 data cache lookup is performed and data is read from the data cache. If the encoding of the IRGN bits is write-through or non-cacheable, an access to external memory is performed. For more information refer to the *Cortex-A5 Technical Reference Manual*.

The MMU might not find a global mapping, or a mapping for the currently selected ASID, with a matching Non-secure TLB ID (NSTID) for the virtual address in the TLB. In this case, the hardware does a translation table walk if the translation table walk is enabled by the PD0 or PD1 bit in the Translation Table Base Control Register. If translation table walks are disabled, the processor returns a Section Translation fault. For more information refer to the *Cortex-A5 Technical Reference Manual*.

If the TLB finds a matching entry, it uses the information in the entry as follows:

1. The access permission bits and the domain determine if the access is enabled. If the matching entry does not pass the permission checks, the MMU signals a memory abort. Refer to the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for a description of access permission bits, abort types and priorities, and for a description of the Instruction Fault Status Register (IFSR) and Data Fault Status Register (DFSR).
2. The memory region attributes specified in both the TLB entry and the CP15 c10 remap registers determine if the access is
   - Secure or Non-secure
   - Shared or not
   - Normal memory, Device, or Strongly-ordered

For more information refer to the *Cortex-A5 Technical Reference Manual*, Memory region remap.

3. The TLB translates the virtual address to a physical address for the memory access.

9.5.5 Interaction with Memory System

The MMU can be enabled or disabled as described in the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition*.

9.5.6 External Aborts

External memory errors are defined as those that occur in the memory system rather than those that are detected by the MMU. External memory errors are expected to be extremely rare. External aborts are caused by errors flagged by the AXI interfaces when the request goes external to the Cortex-A5 processor. External aborts can be configured to trap to Monitor mode by setting the EA bit in the Secure Configuration Register. For more information refer to the *Cortex-A5 Technical Reference Manual*.

9.5.6.1 External Aborts on Data Write

Externally generated errors during a data write can be asynchronous. This means that the r14_abt on entry into the abort handler on such an abort might not hold the address of the instruction that caused the exception.

The DFAR is Unpredictable when an asynchronous abort occurs.

Externally generated errors during data read are always synchronous. The address captured in the DFAR matches the address which generated the external abort.

9.5.6.2 Synchronous and Asynchronous Aborts

Chapter 4, System Control in the *Cortex-A5 Technical Reference Manual* describes synchronous and asynchronous aborts, their priorities, and the IFSR and DFSR. To determine a fault type, read the DFSR for a data abort or the IFSR for an instruction abort.

The processor supports an Auxiliary Fault Status Register for software compatibility reasons only. The processor does not modify this register because of any generated abort.

9.5.7 MMU Software Accessible Registers

The system control coprocessor registers, CP15, in conjunction with page table descriptors stored in memory, control the MMU.

Access all the registers with instructions of the form:

MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>

MCR p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>

CRn is the system control coprocessor register. Unless specified otherwise, CRm and Opcode_2 Should Be Zero.

10. Debug and Test

10.1 Description

The device features a number of complementary debug and test capabilities.

A common JTAG/ICE (In-Circuit Emulator) port is used for standard debugging functions, such as downloading code and single-stepping through programs.

A 2-pin debug port Serial Wire Debug (SWD). SWD replaces the 5-pin JTAG port and provides an easy and risk free alternative to JTAG as the two signals SWDIO and SWCLK are overlaid on the TMS and TCK pins, allowing for bi-modal devices that provide the other JTAG signals. These extra JTAG pins can be switched to other uses when in SWD mode.

The Debug Unit provides a two-pin UART that can be used to upload an application into internal SRAM. It manages the interrupt handling of the internal COMMTX and COMMRX signals that trace the activity of the Debug Communication Channel.

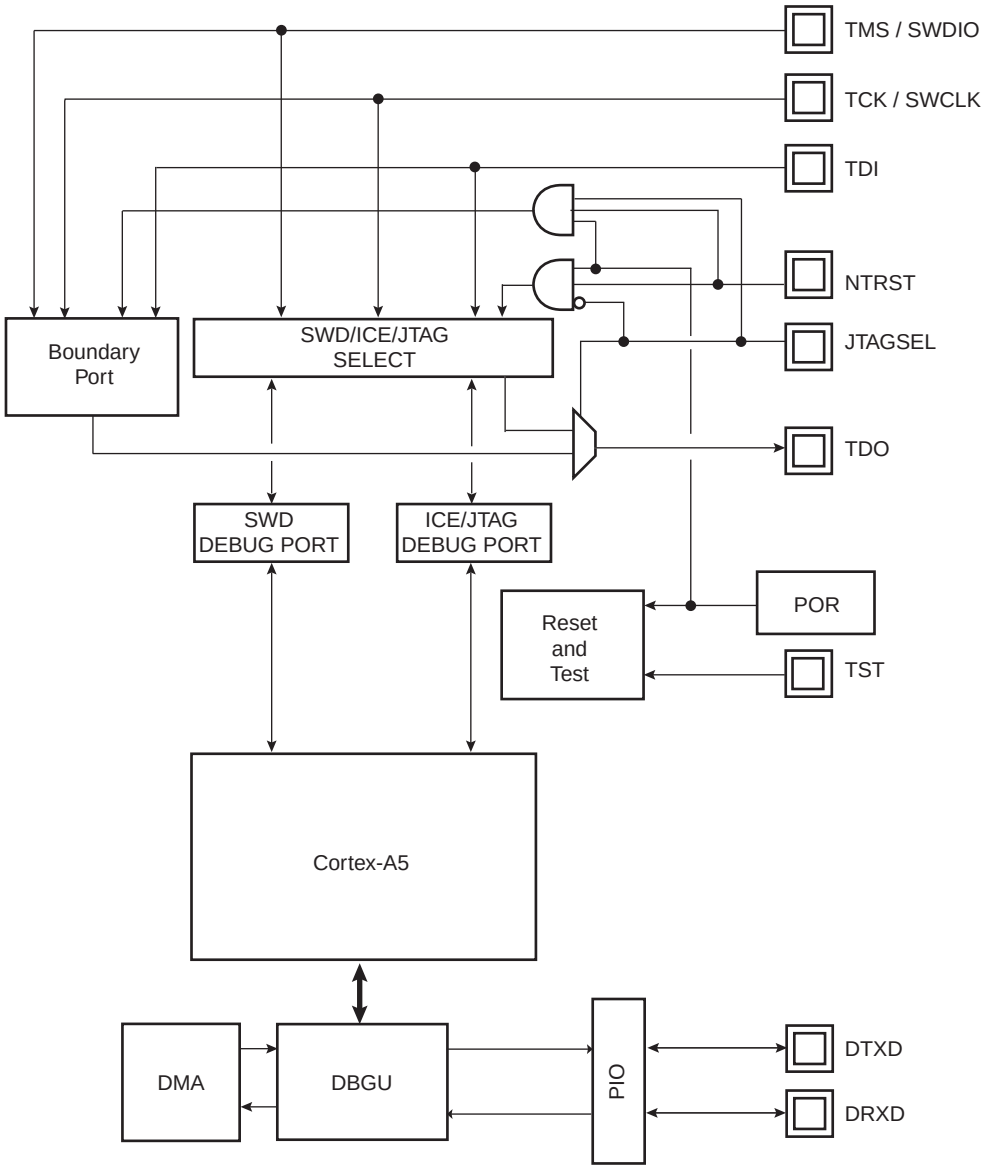
A set of dedicated debug and test input/output pins gives direct access to these capabilities from a PC-based test environment.

10.2 Embedded Characteristics

- Cortex-A5 Real-time In-circuit Emulator
  - Two real-time Watchpoint Units
  - Two Independent Registers: Debug Control Register and Debug Status Register
  - Test Access Port Accessible through JTAG Protocol
  - Debug Communications Channel
  - Serial Wire Debug
- Debug Unit
  - Two-pin UART
  - Debug Communication Channel Interrupt Handling
  - Chip ID Register
- IEEE1149.1 JTAG Boundary-scan on All Digital Pins

10.3 Block Diagram

Figure 10-1. Debug and Test Block Diagram

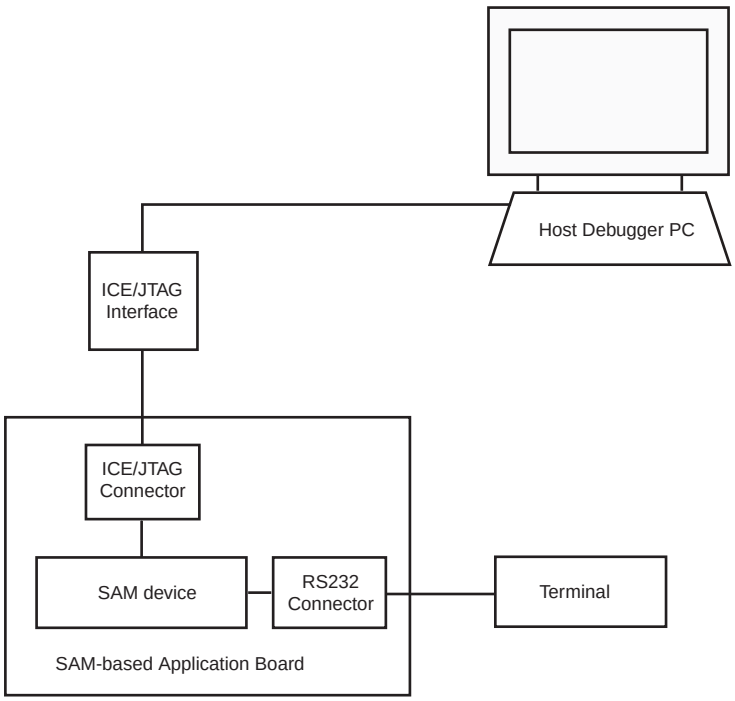


10.4 Application Examples

10.4.1 Debug Environment

Figure 10-2 shows a complete debug environment example. The ICE/JTAG interface is used for standard debugging functions, such as downloading code and single-stepping through the program. A software debugger running on a personal computer provides the user interface for configuring a Trace Port interface utilizing the ICE/JTAG interface.

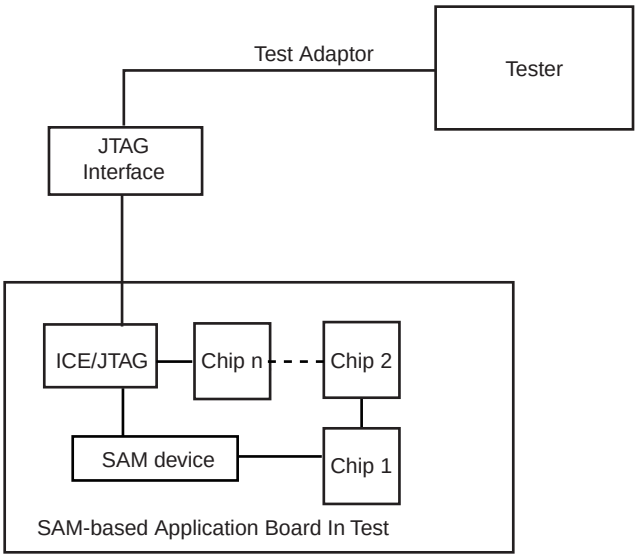
Figure 10-2. Application Debug and Trace Environment Example



10.4.2 Test Environment

Figure 10-3 shows a test environment example. 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 10-3. Application Test Environment Example



10.5 Debug and Test Pin Description

Table 10-1. Debug and Test Pin List

Pin Name	Function	Type	Active Level
Reset/Test			
NRST	Microprocessor Reset	Input	Low
TST	Test Mode Select	Input	High
ICE and JTAG			
NTRST	Test Reset Signal	Input	Low
TCK	Test Clock	Input	–
TDI	Test Data In	Input	–
TDO	Test Data Out	Output	–
TMS	Test Mode Select	Input	–
JTAGSEL	JTAG Selection	Input	–
SWD			
SWCLK	Serial Debug Clock	Input	–
SWDIO	Serial Debug IO	Input/Output	–
Debug Unit			
DRXD	Debug Receive Data	Input	–
DTXD	Debug Transmit Data	Output	–

10.6 Functional Description

10.6.1 Test Pin

One dedicated pin, TST, is used to define the device operating mode. The user must make sure that this pin is tied at low level to ensure normal operating conditions. Other values associated with this pin are reserved for manufacturing test.

10.6.2 EmbeddedICE

The Cortex-A5 EmbeddedICE-RT™ is supported via the ICE/JTAG port. It is connected to a host computer via an ICE interface.The internal state of the Cortex-A5 is examined through an ICE/JTAG port which allows instructions to be serially inserted into the pipeline of the core without using the external data bus. Therefore, when in debug state, a store-multiple (STM) can be inserted into the instruction pipeline. This exports the contents of the Cortex-A5 registers. This data can be serially shifted out without affecting the rest of the system.

There are two scan chains inside the Cortex-A5 processor which support testing, debugging, and programming of the EmbeddedICE-RT. The scan chains are controlled by the ICE/JTAG port.

EmbeddedICE mode is selected when JTAGSEL is low. It is not possible to switch directly between ICE and JTAG operations. A chip reset must be performed after JTAGSEL is changed.

Refer to the ARM document *ARM IHI 0031A_ARM_debug_interface_v5.pdf* for further details on the EmbeddedICE-RT.

10.6.3 JTAG Signal Description

TMS is the Test Mode Select input which controls the transitions of the test interface state machine.

TDI is the Test Data Input line which supplies the data to the JTAG registers (Boundary Scan Register, Instruction Register, or other data registers).

TDO is the Test Data Output line which is used to serially output the data from the JTAG registers to the equipment controlling the test. It carries the sampled values from the boundary scan chain (or other JTAG registers) and propagates them to the next chip in the serial test circuit.

NTRST (optional in IEEE Standard 1149.1) is a Test-ReSeT input which is mandatory in ARM cores and used to reset the debug logic. On Atmel Cortex-A5-based cores, NTRST is a Power On Reset output. It is asserted on power on. If necessary, the user can also reset the debug logic with the NTRST pin assertion during 2.5 MCK periods.

TCK is the Test ClocK input which enables the test interface. TCK is pulsed by the equipment controlling the test and not by the tested device. It can be pulsed at any frequency.

10.6.4 Chip Access Using JTAG Connection

The JTAG connection is not enabled by default on this chip at delivery due to the secure ROM code implementation.

By default, the SAMA5D4 devices boot in Standard mode and not in Secure mode. When the secure ROM code starts, it disables the JTAG access for the entire boot sequence.

If the secure ROM code does not find any program in the external memory, it enables the USB connection and the serial port and waits for a dedicated command to switch the chip into Secure mode.

If any other character is received, the secure ROM code starts the standard SAM-BA® monitor, locks access to the ROM memory, and enables the JTAG.

The chip can then be accessed using the JTAG connection.

If the secure ROM code finds a bootable program, it automatically disables ROM access and enables the JTAG connection just before launching the program.

The procedure to enable JTAG access is as follows:

- Connect your computer to the board with JTAG and USB (J20 USB-A)
- Power on the chip
- Open a terminal console (TeraTerm or HyperTerminal, etc.) on your computer and connect to the USB CDC Serial COM port related to the J20 connector on the board
- Send the '#' character. You will see then the prompt '>' character sent by the device (indicating that the Standard SAM-BA Monitor is running)
- Use the Standard SAM-BA Monitor to connect to the chip with JTAG

Note that you don't need to follow this sequence in order to connect the Standard SAM-BA Monitor with USB.

10.6.5 Debug Unit

The Debug Unit provides a two-pin (DXRD and TXRD) USART that can be used for several debug and trace purposes and offers an ideal means for in-situ programming solutions and debug monitor communication. Moreover, the association with two peripheral data controller channels permits packet handling of these tasks with processor time reduced to a minimum.

The Debug Unit also manages the interrupt handling of the COMMTX and COMMRX signals that come from the ICE and that trace the activity of the Debug Communication Channel.The Debug Unit allows blockage of access to the system through the ICE interface.

A specific register, the Debug Unit Chip ID Register, gives information about the product version and its internal configuration.

For further details on the Debug Unit, refer to Section 41. “Debug Unit (DBGU)”.

10.6.6 IEEE 1149.1 JTAG Boundary Scan

IEEE 1149.1 JTAG Boundary Scan allows pin-level access independent of the device packaging technology.

IEEE 1149.1 JTAG Boundary Scan is enabled when JTAGSEL is high. The SAMPLE, EXTEST and BYPASS functions are implemented. In ICE debug mode, the ARM processor responds with a non-JTAG chip ID that identifies the processor to the ICE system. This is not IEEE 1149.1 JTAG-compliant.

It is not possible to switch directly between JTAG and ICE operations. A chip reset must be performed after JTAGSEL is changed.

A Boundary-scan Descriptor Language (BSDL) file is provided to set up test.

10.7 Boundary JTAG ID Register

Access: Read-only

31	30	29	28	27	26	25	24
VERSION				PART NUMBER			

23	22	21	20	19	18	17	16
PART NUMBER							

15	14	13	12	11	10	9	8
PART NUMBER				MANUFACTURER IDENTITY			

7	6	5	4	3	2	1	0
MANUFACTURER IDENTITY							1

- **VERSION[31:28]: Product Version Number**

Set to 0x0.

- **PART NUMBER[27:12]: Product Part Number**

Product part number is 0x5B39.

- **MANUFACTURER IDENTITY[11:1]**

Set to 0x01F.

Bit[0] required by IEEE Std. 1149.1.

Set to 0x1.

JTAG ID Code value is 0x05B3903F.

10.8 Cortex-A5 DP Identification Code Register IDCODE

The Identification Code Register is always present on all DP implementations. It provides identification information about the ARM Debug Interface.

10.8.1 JTAG Debug Port (JTAG-DP)

It is accessed using its own scan chain, the JTAG-DP Device ID Code Register.

Access: Read-only

31	30	29	28	27	26	25	24
VERSION				PART NUMBER			

23	22	21	20	19	18	17	16
PART NUMBER							

15	14	13	12	11	10	9	8
PART NUMBER				DESIGNER			

7	6	5	4	3	2	1	0
DESIGNER							1

- **VERSION[31:28]: Product Version Number**

Set to 0x0.

- **PART NUMBER[27:12]: Product Part Number**

Product part Number is 0xBA00

- **DESIGNER[11:1]**

Set to 0x23B.

Bit[0] required by IEEE Std. 1149.1.

Set to 0x1.

Debug Port JTAG IDCODE value is 0x4BA00477.

10.8.2 Serial Wire Debug Port (SW-DP)

It is at address 0x0 on read operations when the APnDP bit = 0. Access to the Identification Code Register is not affected by the value of the CTRLSEL bit in the Select Register

Access: Read-only

31	30	29	28	27	26	25	24
VERSION				PART NUMBER			

23	22	21	20	19	18	17	16
PART NUMBER							

15	14	13	12	11	10	9	8
PART NUMBER				DESIGNER			

7	6	5	4	3	2	1	0
DESIGNER							1

- **VERSION[31:28]: Product Version Number**

Set to 0x0.

- **PART NUMBER[27:12]: Product Part Number**

Product part Number is 0xBA01

- **DESIGNER[11:1]**

Set to 0x23B.

Bit[0] required by IEEE Std. 1149.1.

Set to 0x1.

Debug Port Serial Wire IDCODE is 0x2BA01477.

11. Boot Sequence Controller (BSC)

11.1 Description

The System Controller embeds a Boot Sequence Controller (BSC). The boot sequence is programmable through the Boot Sequence Controller Configuration Register (BSC_CR) to save timeout delays on boot.

The BSC_CR is powered by VDDBU. Any modification of the register value is stored and applied after the next reset. The register defaults to the factory value in case of battery removal.

The BSC_CR is programmable with user programs or SAM-BA and is key-protected.

11.2 Embedded Characteristics

- VDDBU powered register

11.3 Product Dependencies

- Product-dependent order

11.4 Boot Sequence Controller (BSC) Registers User Interface

Table 11-1. Register Mapping

Offset	Register	Name	Access	Reset
0x0	Boot Sequence Controller Configuration Register	BSC_CR	Read/Write	–

11.4.1 Boot Sequence Controller Configuration Register

Name: BSC_CR

Access: Read/Write

Factory Value: 0x0000_0000

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
BOOT							

- **BOOT: Boot Media Sequence**

This value is defined in section “Standard Boot Strategies”. It is only written if WPKEY carries the valid value.

- **WPKEY: Write Protection Key** (Write-only)

Value	Name	Description
0x6683	PASSWD	Writing any other value in this field aborts the write operation of the BOOT field. Always reads as 0.

12. Standard Boot Strategies

12.1 Description

The system always boots from the ROM memory at address 0x0.

The ROM code is a boot program contained in the embedded ROM. It is also called “First level bootloader”.

This microprocessor can be configured to run a Standard Boot mode or a Secure Boot mode. More information on how the Secure Boot mode can be enabled, and how the chip operates in this mode, is provided in the application note “SAMA5D4x Secure Boot Strategy”, Atmel literature No. 11295. To obtain this application note and additional information about the secure boot and related tools, contact an Atmel Sales Representative for further details.

By default, the chip starts in Standard Boot Mode.

Note: JTAG access is disabled during the execution of ROM Code Sequence. It is re-enabled when jumping into SRAM when a valid code has been found on an external NVM, in the same time the ROM memory is hidden. If no valid boot has been found on an external NVM, the ROM Code enables the USB connection and the DBGU serial port, and then waits for a special command to set the chip in Secure mode. If any other character is received on any of the two connection link above, the ROM code starts the standard SAM-BA Monitor, locks access to the ROM memory and re-enables the JTAG connection.

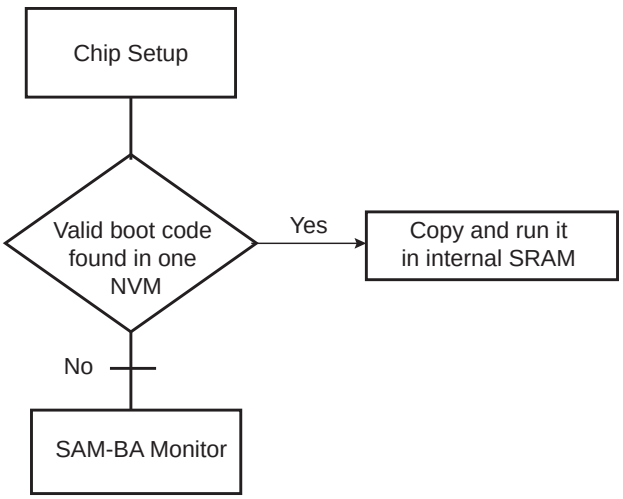
The ROM code standard boot sequence is executed as follows:

- Basic chip initialization: runs on internal 12 MHz RC oscillator and PLL
- Attempt to retrieve a valid code from external non-volatile memories (NVM)
- Crystal or external clock frequency detection to get the clock accuracy required for USB connection
- Execution of a monitor called SAM-BA Monitor, in case no valid application has been found on any NVM

12.2 Flow Diagram

The ROM Code implements the algorithm shown in Figure 12-1.

Figure 12-1. ROM Code Algorithm Flow Diagram



12.3 Chip Setup

At boot start-up, the processor clock (PCK) and the master clock (MCK) source is the 12 MHz fast RC oscillator.

Initialization follows the steps described below:

1. **Stack Setup** for ARM Supervisor mode
2. **Main Clock Selection**: The Master Clock source is switched from the Slow Clock to the Main Oscillator without prescaler. The PMC Status Register is polled to wait for MCK Ready. PCK and MCK are now the Main Clock.
3. **C Variable Initialization:** Non zero-initialized data is initialized in the RAM (copy from ROM to RAM). Zero-initialized data is set to 0 in the RAM.
4. **PLLA Initialization**: PLLA is configured to get a PCK at 96 MHz and an MCK at 48 MHz.

12.4 NVM Boot

12.4.1 NVM Boot Sequence

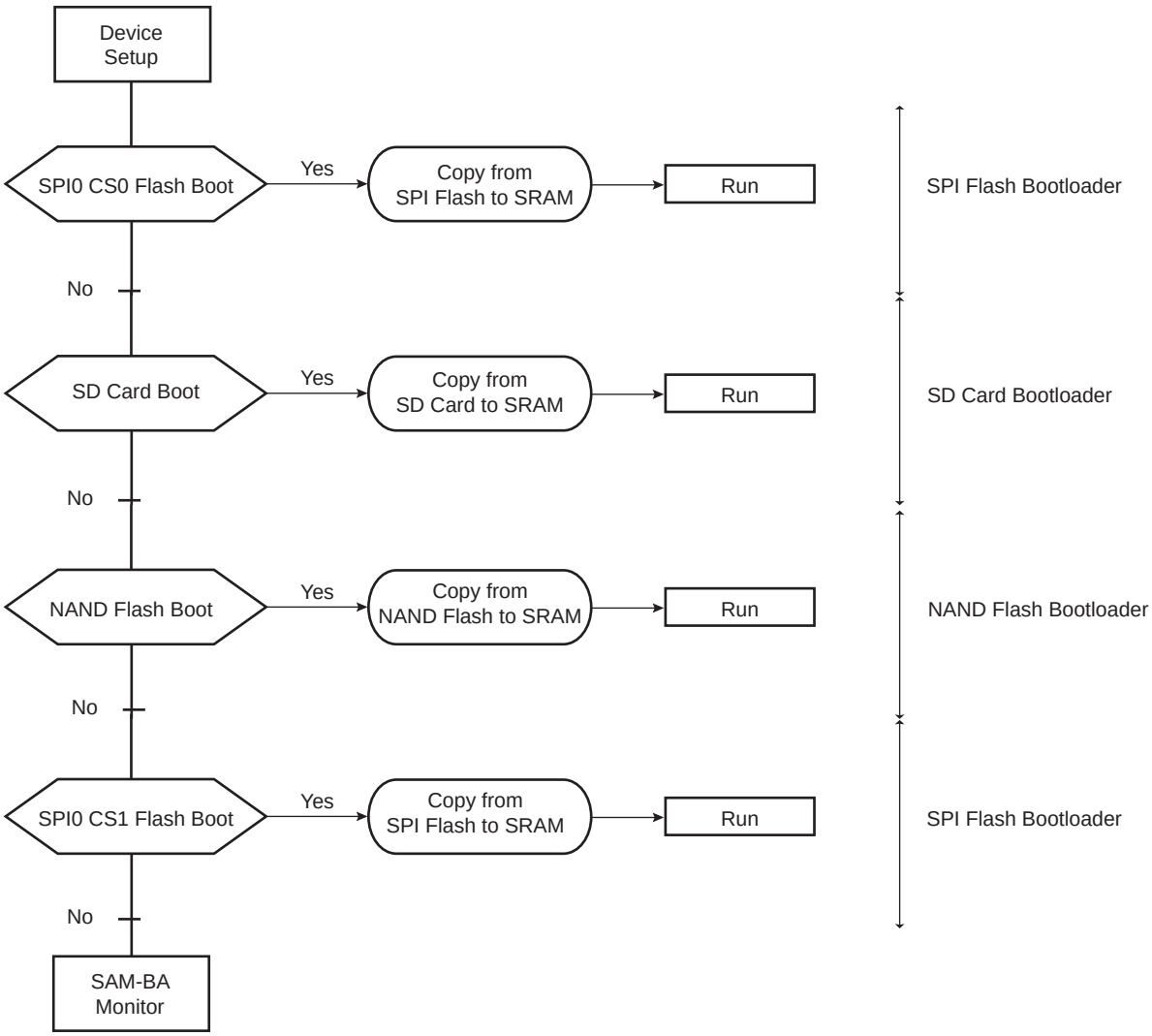
The boot sequence on external memory devices can be controlled using the Boot Sequence Controller Configuration Register (BSC_CR).

The user can then choose to bypass some steps shown in Figure 12-2 “NVM Bootloader Sequence Diagram” according to the BOOT value in the BSC_CR.

Table 12-1. Values of the Boot Sequence Controller Configuration Register

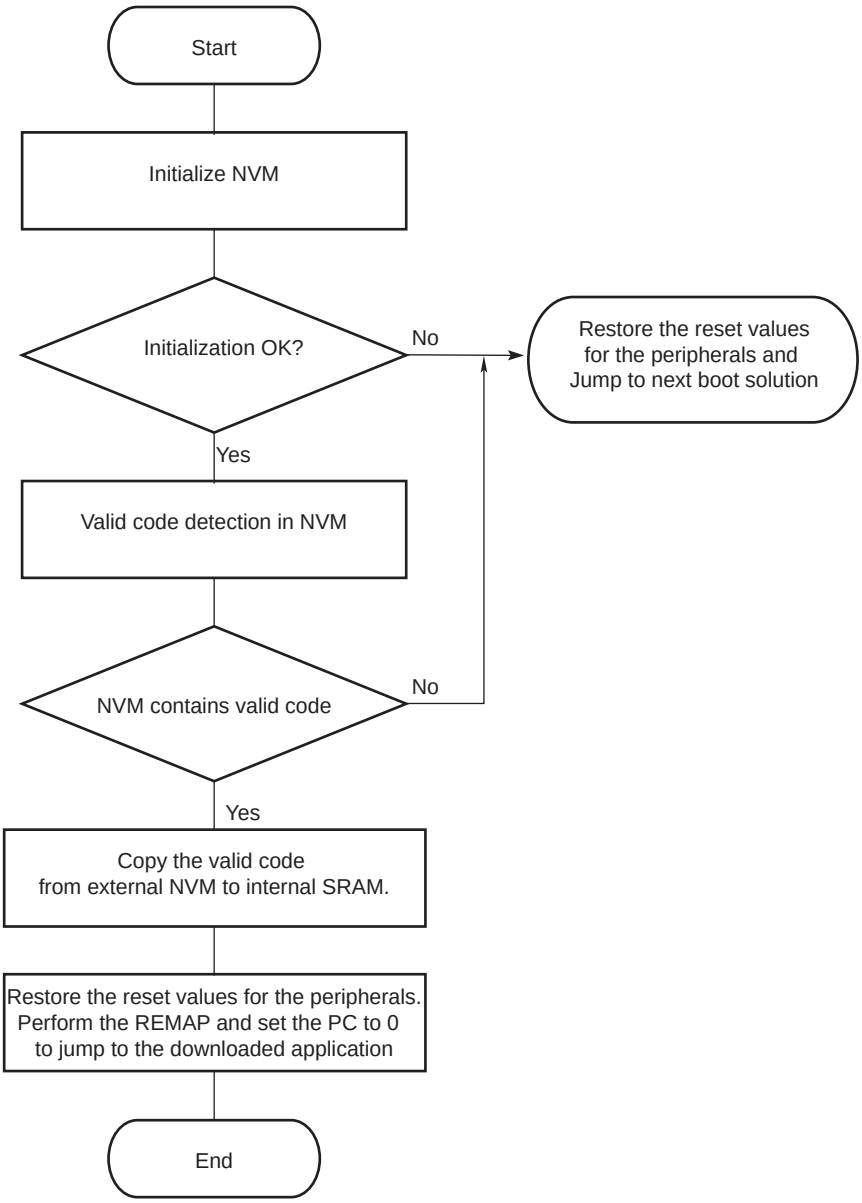
BOOT Value	SPI0 NPCS0	SD Card/eMMC (MCI0)	SD Card/eMMC (MCI1)	8-bit NAND Flash	SPI0 NPCS1	SAM-BA Monitor
0	Y	Y	Y	Y	Y	Y
1	Y	—	Y	Y	Y	Y
2	Y	—	—	Y	Y	Y
3	Y	—	—	—	Y	Y
4	Y	—	—	—	Y	Y
5	—	—	—	—	—	Y
6	—	—	—	—	—	Y
7	—	—	—	—	—	Y

Figure 12-2. NVM Bootloader Sequence Diagram



12.4.2 NVM Bootloader Program Description

Figure 12-3. NVM Bootloader Program Diagram



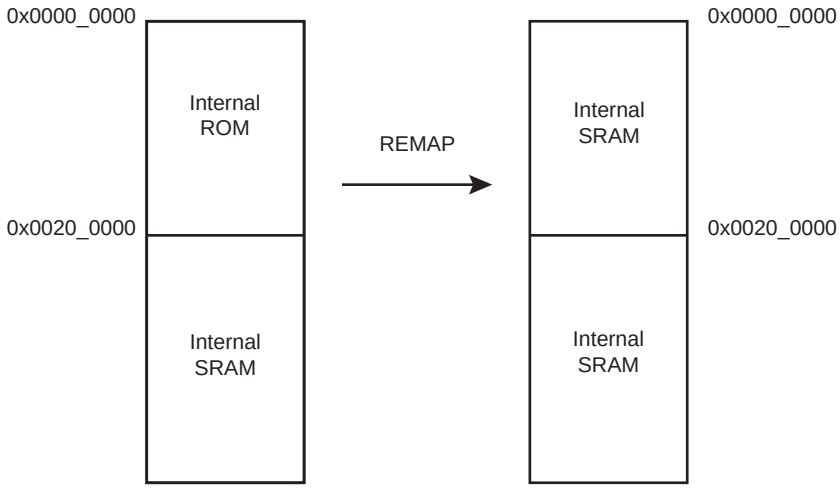
The NVM bootloader program first initializes the PIOs related to the NVM device. Then it configures the right peripheral depending on the NVM and tries to access this memory. If the initialization fails, it restores the reset values for the PIO and the peripheral, and then tries to fulfill the same operations on the next NVM of the sequence.

If the initialization is successful, the NVM bootloader program reads the beginning of the NVM and determines if the NVM contains a valid code.

If the NVM does not contain a valid code, the NVM bootloader program restores the reset value for the peripherals and then tries to fulfill the same operations on the next NVM of the sequence.

If a valid code is found, this code is loaded from the NVM into the internal SRAM and executed by branching at address 0x0000_0000 after remap. This code may be the application code or a second-level bootloader. All the calls to functions are PC relative and do not use absolute addresses.

Figure 12-4. Remap Action after Download Completion



12.4.3 Valid Code Detection

There are two kinds of valid code detection.

12.4.3.1 ARM Exception Vectors Check

The NVM bootloader program reads and analyzes the first 28 bytes corresponding to the first seven ARM exception vectors. Except for the sixth vector, these bytes must implement the ARM instructions for either branch or load PC with PC relative addressing.

Figure 12-5. LDR Opcode

31 28	27 24	23 20	19 16	15 12	11 0
1 1 1 0	0 1 I P	U 1 W 0	Rn	Rd	Offset

Figure 12-6. B Opcode

31 28	27 24	23 0
1 1 1 0	1 0 1 0	Offset (24 bits)

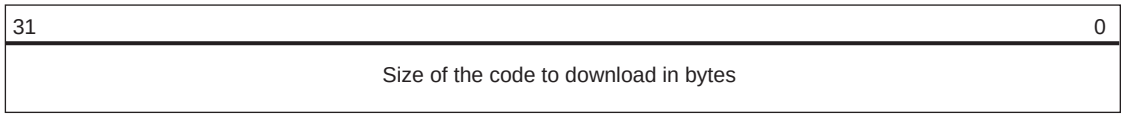
Unconditional instruction: 0xE for bits 31 to 28.

Load PC with the PC relative addressing instruction:

- Rn = Rd = PC = 0xF
- I==0 (12-bit immediate value)
- P==1 (pre-indexed)
- U offset added (U==1) or subtracted (U==0)
- W==1

The sixth vector, at the offset 0x14, contains the size of the image to download. The user must replace this vector with the user's own vector. This procedure is described below.

Figure 12-7. Structure of the ARM Vector 6



The value has to be smaller than 64 Kbytes.

Example

An example of valid vectors:

```
00          ea000006          B 0x20
04          eafffffe          B 0x04
08          ea00002f          B _main
0c          eafffffe          B 0x0c
10          eafffffe          B 0x10
14          00001234          B 0x14 ← Code size = 4660 bytes
18          eafffffe          B 0x18
```

12.4.3.2 boot.bin File Check

This method is the one used on FAT formatted SD Card and eMMC. The boot program must be a file named "`boot.bin`" written in the root directory of the file system. Its size must not exceed the maximum size allowed: 64 Kbytes (0x10000).

12.4.4 Detailed Memory Boot Procedures

12.4.4.1 NAND Flash Boot: NAND Flash Detection

After the NAND Flash interface configuration, a reset command is sent to the memory.

Hardware ECC detection and correction are provided by the PMECC peripheral. Refer to "Section 29.18 "PMECC Controller Functional Description" for more details.

The Boot Program is able to retrieve NAND Flash parameters and ECC requirements using two methods as follows:

- The detection of a specific header written at the beginning of the first page of the NAND Flash,

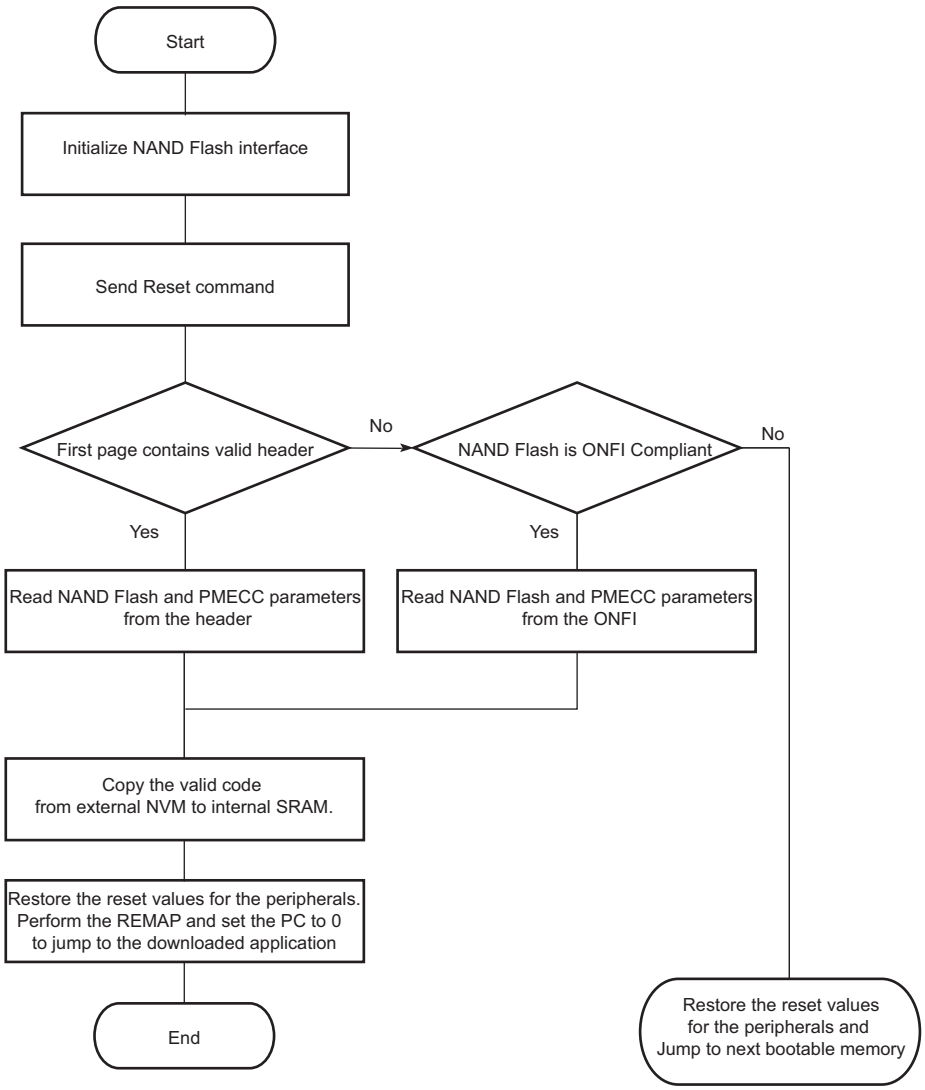
or

- Through the ONFI parameters for the ONFI compliant memories

However, it is highly recommended to use the NAND Flash Header method (first bullet above) since it indicates exactly how the PMECC has been configured to write the bootable program in the NAND Flash, and not to rely only on the NAND Flash capabilities.

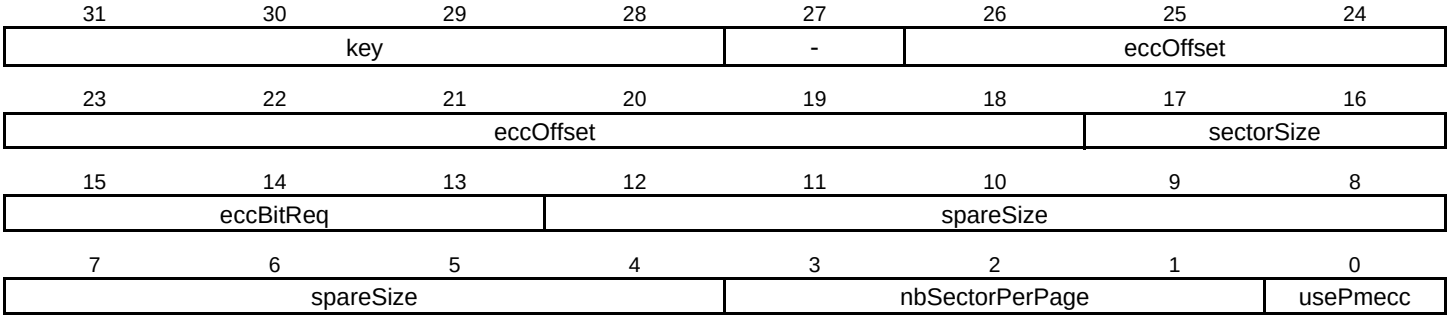
Note: Booting on 16-bit NAND Flash is not possible; only 8-bit NAND Flash memories are supported.

Figure 12-8. Boot NAND Flash Download



NAND Flash Specific Header Detection (recommended solution)

This is the first method used to determine NAND Flash parameters. After Initialization and Reset command, the Boot Program reads the first page without an ECC check, to determine if the NAND parameter header is present. The header is made of 52 times the same 32-bit word (for redundancy reasons) which must contain NAND and PMECC parameters used to correctly perform the read of the rest of the data in the NAND. This 32-bit word is described below:



Note: Booting on 16-bit NAND Flash is not possible; only 8-bit NAND Flash memories are supported.

- **usePmecc: Use PMECC**

0: Do not use PMECC to detect and correct the data

1: Use PMECC to detect and correct the data

- **nbSectorPerPage: Number of Sectors per Page**

- **spareSize: Size of the Spare Zone in Bytes**

- **eccBitReq: Number of ECC Bits Required**

0: 2-bit ECC

1: 4-bit ECC

2: 8-bit ECC

3: 12-bit ECC

4: 24-bit ECC

- **sectorSize: Size of the ECC Sector**

0: For 512 bytes

1: For 1024 bytes per sector

Other value for future use.

- **eccOffset: Offset of the First ECC Byte in the Spare Zone**

A value below 2 is not allowed and will be considered as 2.

- **key: Value 0xC Must be Written here to Validate the Content of the Whole Word.**

If the header is valid, the Boot Program continues with the detection of a valid code.

ONFI 2.2 Parameters (not recommended)

In case no valid header is found, the Boot Program checks if the NAND Flash is ONFI compliant, sending a Read Id command (0x90) with 0x20 as parameter for the address. If the NAND Flash is ONFI compliant, the Boot Program retrieves the following parameters with the help of the Get Parameter Page command:

- Number of bytes per page (byte 80)
- Number of bytes in spare zone (byte 84)
- Number of ECC bit correction required (byte 112)
- ECC sector size: by default, set to 512 bytes; or to 1024 bytes if the ECC bit capability above is 0xFF

By default, the ONFI NAND Flash detection will turn ON the usePmecc parameter, and the ECC correction algorithm is automatically activated.

Once the Boot Program retrieves the parameter, using one of the two methods described above, it reads the first page again, with or without ECC, depending on the usePmecc parameter. Then it looks for a valid code programmed just after the header offset 0xD0. If the code is valid, the program is copied at the beginning of the internal SRAM.

Note: Booting on 16-bit NAND Flash is not possible; only 8-bit NAND Flash memories are supported.

12.4.4.2 NAND Flash Boot: PMECC Error Detection and Correction

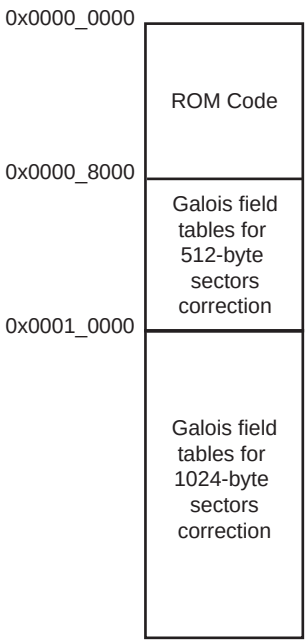
NAND Flash boot procedure uses PMECC to detect and correct errors during NAND Flash read operations in two cases:

- When the usePmecc flag is set in a specific NAND header.
  If the flag is not set, no ECC correction is performed during the NAND Flash page read.
- When the NAND Flash has been detected using ONFI parameters.

The ROM memory embeds the Galois field tables. The user does not need to embed them in his own software.

The Galois field tables are mapped in the ROM just after the ROM code, as described in Figure 12-9.

Figure 12-9. Galois Field Table Mapping



For a full description and an example of how to use the PMECC detection and correction feature, refer to the software package dedicated to this device on Atmel's web site.

12.4.4.3 SD Card / eMMC Boot

The SD Card / eMMC bootloader looks for a “`boot.bin`” file in the root directory of a FAT12/16/32 file system.

Supported SD Card Devices

SD Card Boot supports all SD Card memories compliant with the SD Memory Card Specification V2.0. This includes SDHC cards.

12.4.4.4 SPI Flash Boot

Two types of SPI Flash are supported: SPI Serial Flash and SPI DataFlash.

The SPI Flash bootloader tries to boot on SPI0, first looking for SPI Serial Flash, and then for SPI DataFlash.

It uses only one valid code detection: analysis of ARM exception vectors.

The SPI Flash read is done by means of a Continuous Read command from the address 0x0. This command is 0xE8 for DataFlash and 0x0B for Serial Flash devices.

Supported DataFlash Devices

The SPI Flash Boot program supports the DataFlash devices listed in Table 12-2.

Table 12-2. DataFlash Devices

Device	Density	Page Size (bytes)	Number of Pages
AT45DB011	1 Mbit	264	512
AT45DB021	2 Mbits	264	1024
AT45DB041	4 Mbits	264	2048
AT45DB081	8 Mbits	264	4096
AT45DB161	16 Mbits	528	4096
AT45DB321	32 Mbits	528	8192
AT45DB642	64 Mbits	1056	8192
AT45DB641	64 Mbits	264	37768

Supported Serial Flash Devices

The SPI Flash Boot program supports all SPI Serial Flash devices responding correctly at both Get Status and Continuous Read commands.

12.4.5 Hardware and Software Constraints

The NVM drivers use several PIOs in Peripheral mode to communicate with external memory devices. Care must be taken when these PIOs are used by the application. The connected devices could be unintentionally driven at boot time, and thus electrical conflicts between output pins used by the NVM drivers and the connected devices could occur.

To ensure correct functionality, it is recommended to plug in critical devices to other pins, not used by the NVM.

Table 12-3 contains a list of pins that are driven during the boot program execution. These pins are driven during the boot sequence for a period of less than 1 second if no correct boot program is found.

Before performing the jump to the application in the internal SRAM, all the PIOs and peripherals used in the boot program are set to their reset state.

Table 12-3. PIO Driven during Boot Program Execution

NVM Bootloader	Peripheral	Pin	PIO Line
NAND	EBI CS3 SMC	NANDOE	PIOC13
	EBI CS3 SMC	NANDWE	PIOC14
	EBI CS3 SMC	NANDCS	PIOC15
	EBI CS3 SMC	NAND ALE	PIOC17
	EBI CS3 SMC	NAND CLE	PIOC18
	EBI CS3 SMC	Cmd/Addr/Data	PIOC5–PIOC12
SD Card / eMMC	MCI0	MCI0_CK	PIOC4
	MCI0	MCI0_CDA	PIOC5
	MCI0	MCI0_D0	PIOC6
	MCI0	MCI0_D1	PIOC7
	MCI0	MCI0_D2	PIOC8
	MCI0	MCI0_D3	PIOC9
	MCI1	MCI1_CK	PIOE18
	MCI1	MCI1_CDA	PIOE19
	MCI1	MCI1_D0	PIOE20
	MCI1	MCI1_D1	PIOE21
	MCI1	MCI1_D2	PIOE22
	MCI1	MCI1_D3	PIOE23
SPI Flash	SPI0	MOSI	PIOC1
	SPI0	MISO	PIOC0
	SPI0	SPCK	PIOC2
	SPI0	NPCS0	PIOC3
	SPI0	NPCS1	PIOC4
SAM-BA Monitor	DBGU	DRXD	PIOE16
	DBGU	DTXD	PIOE17

12.5 SAM-BA Monitor

If no valid code is found in the NVM during the NVM bootloader sequence.

The Main Clock is switched to the 32 kHz RC oscillator to allow external clock frequency to be measured.

The Main Oscillator is enabled and set in Bypass mode. If the MOSCSELS bit rises, an external clock is connected**.** If not, the Bypass mode is cleared to attempt external quartz detection. This detection is successful when the MOSCXTS and MOSCSELS bits rise, else the internal 12 MHz fast RC oscillator is used as the Main Clock.

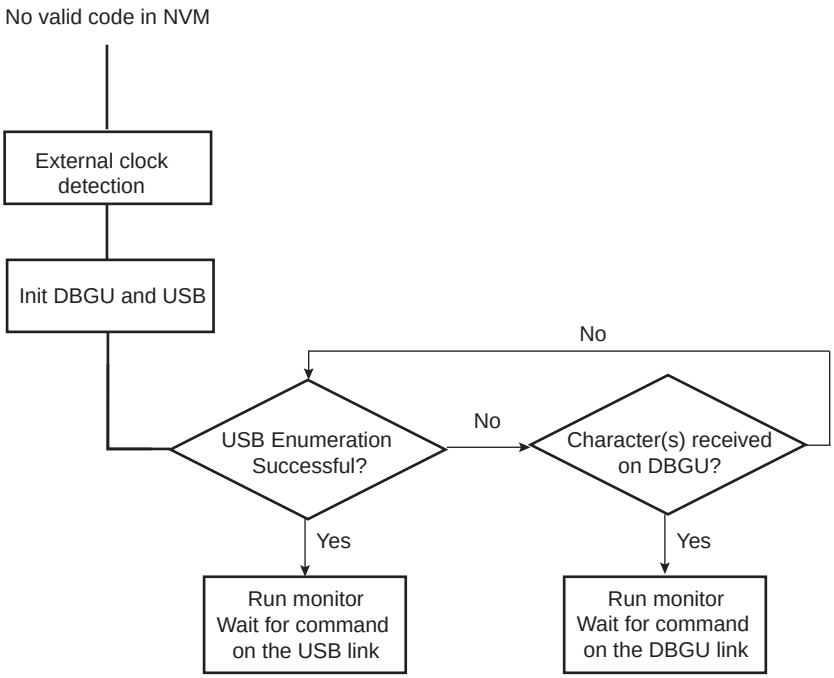
If an external clock or crystal frequency running at 12 MHz is found, then the PLLA is configured to allow communication on the USB link for the SAM-BA Monitor else the Main Clock is switched to the internal 12 MHz fast RC oscillator, but USB will not be activated.

The SAM-BA Monitor steps are:

- Initialize DBGU and USB
- Check if USB Device enumeration occurred
- Check if characters are received on the DBGU

Once the communication interface is identified, the application runs in an infinite loop waiting for different commands as listed in Table 12-4.

Figure 12-10. SAM-BA Monitor Diagram



12.5.1 Command List

Table 12-4. Commands Available through the SAM-BA Monitor

Command	Action	Argument(s)	Example
N	Set Normal Mode	No argument	**N**#
T	Set Terminal Mode	No argument	**T**#
O	Write a byte	Address, Value#	**O**200001,CA#
o	Read a byte	Address,#	**o**200001,#
H	Write a half word	Address, Value#	**H**200002,CAFE#
h	Read a half word	Address,#	**h**200002,#
W	Write a word	Address, Value#	**W**200000,CAFEDECA#
w	Read a word	Address,#	**w**200000,#
S	Send a file	Address,#	**S**200000,#
R	Receive a file	Address, NbOfBytes#	**R**200000,1234#
G	Go	Address#	**G**200200#
V	Display version	No argument	**V**#

- Mode commands:
  - Normal mode configures SAM-BA Monitor to send / receive data in binary format,
  - Terminal mode configures SAM-BA Monitor to send / receive data in ASCII format.
- Write commands: Write a byte (**O**), a halfword (**H**) or a word (**W**) to the target
  - *Address*: Address in hexadecimal
  - *Value*: Byte, halfword or word to write in hexadecimal
  - *Output*: '>'
- Read commands: Read a byte (**o**), a halfword (**h**) or a word (**w**) from the target
  - *Address*: Address in hexadecimal
  - *Output*: The byte, halfword or word read in hexadecimal followed by '>'
- Send a file (**S**): Send a file to a specified address
  - *Address*: Address in hexadecimal
  - *Output*: '>'

Note: There is a timeout on this command which is reached when the prompt '>' appears before the end of the command execution.

- Receive a file (**R**): Receive data into a file from a specified address
  - *Address*: Address in hexadecimal
  - *NbOfBytes*: Number of bytes in hexadecimal to receive
  - *Output*: '>'
- Go (**G**): Jump to a specified address and execute the code
  - *Address*: Address to jump in hexadecimal
  - *Output*: '>' once returned from the program execution. If the executed program does not handle the link register at its entry and does not return, the prompt will not be displayed
- Get Version (**V**): Return the Boot Program version
  - *Output*: version, date and time of ROM code followed by '>'

12.5.2 DBGU Serial Port

Communication is performed through the DBGU serial port initialized to 115,200 Baud, 8 bits of data, no parity, 1 stop bit.

12.5.2.1 Xmodem Protocol

The Send and Receive File commands use the Xmodem protocol to communicate. Any terminal performing this protocol can be used to send the application file to the target. The size of the binary file to send depends on the SRAM size embedded in the product. In all cases, the size of the binary file must be lower than the SRAM size because the Xmodem protocol requires some SRAM memory in order to work.

The Xmodem protocol supported is the 128-byte length block. This protocol uses a two-character CRC16 to guarantee detection of maximum bit errors.

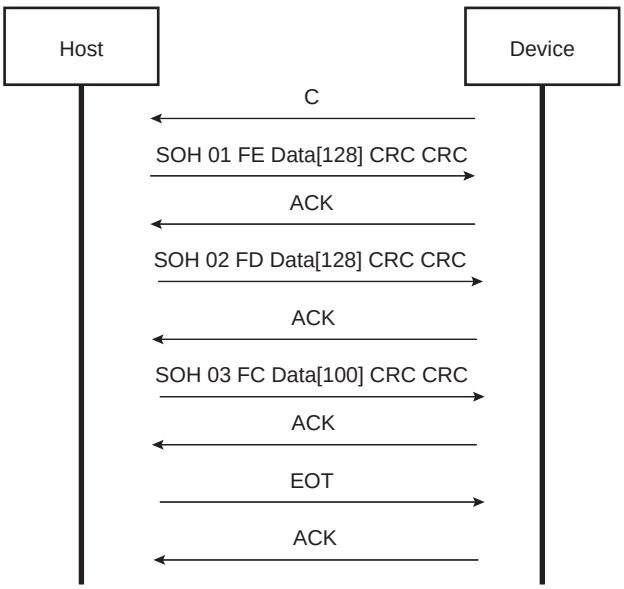
Xmodem protocol with CRC is supported by successful transmission reports provided both by a sender and by a receiver. Each transfer block is as follows:

<SOH><blk #><255-blk #><--128 data bytes--><checksum> in which:

- <SOH> = 01 hex
- <blk #> = binary number, starts at 01, increments by 1, and wraps 0FFH to 00H (not to 01)
- <255-blk #> = 1's complement of the blk#.
- <checksum> = 2 bytes CRC16

Figure 12-11 shows a transmission using this protocol.

Figure 12-11. Xmodem Transfer Example



12.5.3 USB Device Port

12.5.3.1 Supported External Crystal / External Clocks

The SAM-BA Monitor only supports a frequency of 12 MHz to allow USB communication for both external crystal and external clock.

12.5.3.2 USB Class

The device uses the USB Communication Device Class (CDC) drivers to take advantage of the installed PC Serial Communication software to talk over the USB. The CDC is implemented in all releases of Windows®, beginning

with Windows 98 SE. The CDC document, available at www.usb.org, describes how to implement devices such as ISDN modems and virtual COM ports.

The Vendor ID is the Atmel's vendor ID 0x03EB. The product ID is 0x6124. These references are used by the host operating system to mount the correct driver. On Windows systems, INF files contain the correspondence between vendor ID and product ID.

12.5.3.3 Enumeration Process

The USB protocol is a master/slave protocol. The host starts the enumeration, sending requests to the device through the control endpoint. The device handles standard requests as defined in the USB Specification.

Table 12-5. Handled Standard Requests

Request	Definition
GET_DESCRIPTOR	Returns the current device configuration value
SET_ADDRESS	Sets the device address for all future device access
SET_CONFIGURATION	Sets the device configuration
GET_CONFIGURATION	Returns the current device configuration value
GET_STATUS	Returns status for the specified recipient
SET_FEATURE	Used to set or enable a specific feature
CLEAR_FEATURE	Used to clear or disable a specific feature

The device also handles some class requests defined in the CDC class.

Table 12-6. Handled Class Requests

Request	Definition
SET_LINE_CODING	Configures DTE rate, stop bits, parity and number of character bits
GET_LINE_CODING	Requests current DTE rate, stop bits, parity and number of character bits
SET_CONTROL_LINE_STATE	RS-232 signal used to indicate to the DCE device that the DTE device is now present

Unhandled requests are stalled.

12.5.3.4 Communication Endpoints

Endpoint 0 is used for the enumeration process.

Endpoint 1 (64-byte Bulk OUT) and endpoint 2 (64-byte Bulk IN) are used as communication endpoints.

SAM-BA Boot commands are sent by the host through Endpoint 1. If required, the message is split into several data payloads by the host driver.

If the command requires a response, the host sends IN transactions to pick up the response.

12.6 Fuse Box Controller

SAMA5D4 embeds 512 fuse bits reserved for customer needs.

Setting the write-once FUSE bit in the SFR_SECURE register disables access to the Secure Fuse Controller (SFC).

12.6.1 Fuse Bit Mapping

The read/write access to the fuse bits requires that the internal 12 MHz RC oscillator is enabled.

To avoid any malfunctioning, the user must not write the "DO NOT USE (DNU)" Fuse bits.

Table 12-7. Customer Fuse Matrix

SFC_DR	Bits	Use					
15	[511:480]	JTAG_DIS [511]	SEC_DEBUG_DIS [510]	DNU [509:483]	MD [482]	DNU [481]	S [480]
14	[479:448]	USER_DATA[479:0]					
13	[447:416]						
12	[415:384]						
11	[383:352]						
10	[351:320]						
9	[319:288]						
8	[287:256]						
7	[255:224]						
6	[223:192]						
5	[191:160]						
4	[159:128]						
3	[127:96]						
2	[95:64]						
1	[63:32]						
0	[31:0]						

Table 12-8 provides details on the four special function fuse bits JTAG_DIS, SEC_DEBUG_DIS, MD, and S.

Table 12-8. Special Function Fuse Bits

JTAG_DIS [511] JTAG Disable	SEC_DEBUG_DIS [510] Secure Debug Disable	MD [482] Monitor Disable	S [480] Secure Boot Mode	Description
0	0	–	–	Full JTAG debug allowed in Secure and Normal modes
0	1	–	–	JTAG debug only allowed in Normal mode (non-secure)
1	X	–	–	JTAG debug disabled
–	–	0	–	(Default case) SAM-BA Monitor is launched in the case where no bootable program is found in an external memory.
–	–	1	–	No SAM-BA Monitor is launched in the case where no bootable program is found in an external memory. No communication link is set up, JTAG remains disabled and the ROM Code is in an infinite loop.
–	–	–	0	Standard boot sequence
–	–	–	1	Secure boot sequence

13. L2 Cache Controller (L2CC)

13.1 Description

The L2 Cache Controller (L2CC) is based on the L2CC-PL310 ARM multi-way cache macrocell, version r3p2. The addition of an on-chip secondary cache, also referred to as a Level 2 or L2 cache, is a method of improving the system performance when significant memory traffic is generated by the processor.

13.2 Embedded Characteristics

- 8-way set associative cache architecture
- Data banking is not supported
- No parity bit embedded
- Lockdown by master is not supported
- Lockdown by line is not supported
- TrustZone architecture for enhanced OS security

13.3 Product Dependencies

13.3.1 Power Management

The L2 Cache Controller is continuously clocked by the Processor Clock. The Power Management Controller has no effect on the behavior of the L2 Cache Controller.

13.4 Functional Description

The addition of an on-chip secondary cache, also referred to as a Level 2 or L2 cache, is a recognized method of improving the performance of ARM-based systems when significant memory traffic is generated by the processor. By definition a secondary cache assumes the presence of a Level 1 or primary cache, closely coupled or internal to the processor. Memory access is fastest to L1 cache, followed closely by L2 cache. Memory access is typically significantly slower with L3 main memory.

The cache controller is a unified, physically addressed, physically tagged cache with up to 8 ways. The user can lock the replacement algorithm on a way basis, enabling the associativity to be reduced from 8-way down to 1-way (directly mapped).

The cache controller does not have snooping hardware to maintain coherency between caches, so the user has to maintain coherency by software.

13.4.1 Double Linefill Issuing

The L2CC cache line length is 32-byte. Therefore, by default, on each L2 cache miss,

L2CC issues 32-byte linefills, 4 x 64-bit read bursts, to the L3 memory system. L2CC can issue 64-byte linefills, 8 x 64-bit read bursts, on an L2 cache miss. When the L2CC is waiting for the data from L3, it performs a lookup on the second cache line targeted by the 64-byte linefill. If it misses, data corresponding to the second cache line are allocated to the L2 cache. If it hits, data corresponding to the second cache line are discarded.

The user can control this feature using the DLEN, DLFWRDIS and DLEN bits of the L2CC Prefetch Control Register. The IDLEN and DLFWRDIS bits are only used if you set the DLEN bit HIGH. Table 13-1 shows the behavior of the L2CC master ports, depending on the configuration chosen by the user.

Table 13-1. L2CC Master Port Behavior

Bit 30 DLEN	Bit 27 DLFWRDIS	Bit 23 IDLEN	Original Read Address from L1	Read Address to L3	AXI Burst Type	AXI Burst Length	Targeted Cache Lines
0	0 or 1	0 or 1	0x00	0x00	WRAP	0x3, 4x64-bit	0x00
0	0 or 1	0 or 1	0x20	0x20	WRAP	0x3, 4x64-bit	0x20
1	0 or 1	0	0x00	0x00	WRAP	0x7, 8x64-bit	0x00 and 0x20
1	1	0	0x08 or 0x10 or 0x18	0x08	WRAP	0x3, 4x64-bit	0x00
1	0	0	0x08 or 0x10 or 0x18	0x00	WRAP	0x7, 8x64-bit	0x00 and 0x20
1	0 or 1	0	0x20	0x20	WRAP	0x7, 8x64-bit	0x00 and 0x20
1	1	0	0x28 or 0x30 or 0x38	0x28	WRAP	0x3, 4x64-bit	0x20
1	0	0	0x28 or 0x30 or 0x38	0x20	WRAP	0x7, 8x64-bit	0x00 and 0x20
1	0 or 1	1	0x00	0x00	INCR or WRAP	0x7, 8x64-bit	0x00 and 0x20
1	1	1	0x08 or 0x10 or 0x18	0x08	WRAP	0x3, 4x64-bit	0x00
1	0	1	0x08 or 0x10 or 0x18	0x00	INCR or WRAP	0x7, 8x64-bit	0x00 and 0x20
1	0 or 1	1	0x20	0x20	INCR	0x7, 8x64-bit	0x20 and 0x40
1	1	1	0x28 or 0x30 or 0x38	0x28	WRAP	0x3, 4x64-bit	0x20
1	0	1	0x28 or 0x30 or 0x38	0x20	INCR	0x7, 8x64-bit	0x20 and 0x40

Notes:
1. Double linefills are not issued for prefetch reads if you enable exclusive cache configuration.
2. Double linefills are not launched when crossing a 4-Kbyte boundary.
3. Double linefills only occur if a WRAP4 or an INCR4 64-bit transaction is received on the slave ports. This transaction is most commonly seen as a result of a cache linefill in a master, but can be produced by a master when accessing memory marked as inner non-cacheable.

13.5 L2 Cache Controller (L2CC) User Interface

Table 13-2. Register Mapping

Offset	Register	Name	Access	Reset
0x000	Cache ID Register	L2CC_IDR	Read-only	0x4100_00C9
0x004	Cache Type Register	L2CC_TYPR	Read-only	0x0010_0100
0x100	Control Register	L2CC_CR	Read/Write, Read-only[1]	0x0000_0000
0x104	Auxiliary Control Register	L2CC_ACR	Read/Write, Read-only[1]	0x0202_0000
0x108	Tag RAM Control Register	L2CC_TRCR	Read/Write, Read-only[1]	0x0000_0111
0x10C	Data RAM Control Register	L2CC_DRCR	Read/Write, Read-only[1]	0x0000_0111
0x110–0x1FC	Reserved	–	–	–
0x200	Event Counter Control Register	L2CC_ECR	Read/Write	0x0000_0000
0x204	Event Counter 1 Configuration Register	L2CC_ECFGR1	Read/Write	0x0202_0000
0x208	Event Counter 0 Configuration Register	L2CC_ECFGR0	Read/Write	0x0000_0000
0x20C	Event Counter 1 Value Register	L2CC_EVR1	Read/Write	0x0000_0000
0x210	Event Counter 0 Value Register	L2CC_EVR0	Read/Write	0x0000_0000
0x214	Interrupt Mask Register	L2CC_IMR	Programmable[2]	0x0000_0000
0x218	Masked Interrupt Status Register	L2CC_MISR	Read-only	0x0000_0000
0x21C	Raw Interrupt Status Register	L2CC_RISR	Read-only	0x0000_0000
0x220	Interrupt Clear Register	L2CC_ICR	Programmable[2]	0x0000_0000
0x224–0x72C	Reserved	–	–	–
0x730	Cache Synchronization Register	L2CC_CSR	Read/Write	0x0000_0000
0x734–0x76C	Reserved	–	–	–
0x770	Invalidate Physical Address Line Register	L2CC_IPALR	Read/Write	0x0000_0000
0x774–0x778	Reserved	–	–	–
0x77C	Invalidate Way Register	L2CC_IWR	Read/Write	0x0000_0000
0x780–0x7AF	Reserved	–	–	–
0x7B0	Clean Physical Address Line Register	L2CC_CPALR	Read/Write	0x0000_0000
0x7B4	Reserved	–	–	–
0x7B8	Clean Index Register	L2CC_CIR	Read/Write	0x0000_0000
0x7BC	Clean Way Register	L2CC_CWR	Read/Write	0x0000_0000
0x7C0–0x7EC	Reserved	–	–	–
0x7F0	Clean Invalidate Physical Address Line Register	L2CC_CIPALR	Read/Write	0x0000_0000
0x7F4	Reserved	–	–	–
0x7F8	Clean Invalidate Index Register	L2CC_CIIR	Read/Write	0x0000_0000
0x7FC	Clean Invalidate Way Register	L2CC_CIWR	Read/Write	0x0000_0000
0x800–0x8FC	Reserved	–	–	–
0x900	Data Lockdown Register	L2CC_DLKR	Programmable[2]	0x0000_0000
0x904	Instruction Lockdown Register	L2CC_ILKR	Programmable[2]	0x0000_0000

Table 13-2. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x908–0xF3C	Reserved	–	–	–
0xF40	Debug Control Register	L2CC_DCR	Read/Write, Read-only(1)	0x0000_0000
0xF44–0xF5C	Reserved	–	–	–
0xF60	Prefetch Control Register	L2CC_PCR	Read/Write, Read-only(1)	0x0000_0000
0xF64–0xF7C	Reserved	–	–	–
0xF80	Power Control Register	L2CC_POWCR	Read/Write, Read-only(1)	0x0000_0000

Notes: 1. Read/Write in Secure mode, Read-only in Non-secure mode.
2. Programmable in Auxiliary Control Register.

13.5.1 L2CC Cache ID Register

Name: L2CC_IDR

Address: 0x00A00000

Access: Read-only

31	30	29	28	27	26	25	24
ID							

23	22	21	20	19	18	17	16
ID							

15	14	13	12	11	10	9	8
ID							

7	6	5	4	3	2	1	0
ID							

- **ID: Cache Controller ID**

The cache ID is 0x410000C9.

13.5.2 L2CC Type Register

Name: L2CC_TYPR

Address: 0x00A00004

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	DL2WSIZE			–	DL2ASS	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	IL2WSIZE		

7	6	5	4	3	2	1	0
–	IL2ASS	–	–	–	–	–	–

- **IL2ASS: Instruction L2 Cache Associativity**

The value is read from the field ASS in Auxiliary Control Register, should be 0.

- **IL2WSIZE: Instruction L2 Cache Way Size**

The value is read from the field WAYSIZE in Auxiliary Control Register, should be 0x1.

- **DL2ASS: Data L2 Cache Associativity**

The value is read from the field ASS in Auxiliary Control Register, should be 0.

- **DL2WSIZE: Data L2 Cache Way Size**

The value is read from the field WAYSIZE in Auxiliary Control Register, should be 0x1.

13.5.3 L2CC Control Register

Name: L2CC_CR

Address: 0x00A00100

Access: Read/Write in Secure mode

Read-only in Non-secure mode

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
–	–	–	–	–	–	–	L2CEN

- **L2CEN: L2 Cache Enable**

0: L2 Cache is disabled. This is the default value.

1: L2 Cache is enabled.

13.5.4 L2CC Auxiliary Control Register

Name: L2CC_ACR

Address: 0x00A00104

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	–	IPEN	DPEN	NSIAC	NSLEN	CRPOL	FWA
23	22	21	20	19	18	17	16
FWA	SAOEN	PEN	EMBEN	WAYSIZE			ASS
15	14	13	12	11	10	9	8
–	–	SAIE	EXCC	SBDLE	HPSO	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

Note: The L2 Cache Controller (L2CC) must be disabled in the L2CC Control Register prior to any write access to this register.

- **HPSO: High Priority for SO and Dev Reads Enable**

0: Strongly Ordered and Device reads have lower priority than cacheable accesses when arbitrated in the L2CC master ports. This is the default value.

1: Strongly Ordered and Device reads get the highest priority when arbitrated in the L2CC master ports.

- **SBDLE: Store Buffer Device Limitation Enable**

0: Store buffer device limitation is disabled. Device writes can take all slots in the store buffer. This is the default value.

1: Store buffer device limitation is enabled.

- **EXCC: Exclusive Cache Configuration**

0: Disabled. This is the default value.

1: Enabled.

- **SAIE: Shared Attribute Invalidate Enable**

0: Shared invalidate behavior is disabled. This is the default value.

1: Shared invalidate behavior is enabled if the Shared Attribute Override Enable bit is not set.

Shared invalidate behavior is enabled if both:

- Shareable Attribute Invalidate Enable bit is set in the Auxiliary Control Register, bit[13]
- Shared Attribute Override Enable bit is not set in the Auxiliary Control Register, bit[22]

- **ASS: Associativity**

0: 8-way.This is the default value.

1: Reserved.

- **WAYSIZE: Way Size**

Value	Name	Description
0x0	RESERVED	Reserved
0x1	16KB_WAY	16-Kbyte way set associative
0x2	RESERVED	Reserved
0x3	RESERVED	Reserved
0x4	RESERVED	Reserved
0x5	RESERVED	Reserved
0x6	RESERVED	Reserved
0x7	RESERVED	Reserved

- **EMBEN: Event Monitor Bus Enable**

0: Disabled. This is the default value.

1: Enabled.

- **PEN: Parity Enable**

0: Disabled. This is the default value.

1: Enabled.

- **SAOEN: Shared Attribute Override Enable**

0: Treats shared accesses. This is the default value.

1: Shared attribute is internally ignored.

- **FWA: Force Write Allocate**

0: The L2 Cache controller uses AWCACHE attributes for WA. This is the default value.

1: User forces no allocate, WA bit must be set to 0.

2: User overrides AWCACHE attributes, WA bit must be set to 1. All cacheable write misses become write allocated.

3: The write allocation is internally mapped to 00.

- **CRPOL: Cache Replacement Policy**

0: Pseudo-random replacement using the LFSR algorithm.

1: Round-robin replacement. This is always the default value.

- **NSLEN: Non-Secure Lockdown Enable**

0: Lockdown registers cannot be modified using non-secure accesses. This is the default value.

1: Non-secure accesses can write to the lockdown registers.

- **NSIAC: Non-Secure Interrupt Access Control**

0: Interrupt Clear Register and Interrupt Mask Register can only be modified or read with secure accesses. This is the default value.

1: Interrupt Clear Register and Interrupt Mask Register can be modified or read with secure or non-secure accesses.

- **DPEN: Data Prefetch Enable**

0: Data prefetching is disabled. This is the default value.

1: Data prefetching is enabled.

- **IPEN: Instruction Prefetch Enable**

0: Instruction prefetching is disabled. This is the default value.

1: Instruction prefetching is enabled.

13.5.5 L2CC Tag RAM Latency Control Register

Name: L2CC_TRCR

Address: 0x00A00108

Access: Read/Write in Secure mode

Read-only in Non-secure mode

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

Note: The L2 Cache Controller (L2CC) must be disabled in the L2CC Control Register prior to any write access to this register.

- **TSETLAT: Setup Latency**
- **TRDLAT: Read Access Latency**
- **TWRLAT: Write Access Latency**

Latency to Tag RAM is the programmed value + 1.

Default value is 0.

13.5.6 L2CC Data RAM Latency Control Register

Name: L2CC_DRCR

Address: 0x00A0010C

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	DWRLAT		

7	6	5	4	3	2	1	0
–	DRDLAT			–	DSETLAT		

Note: The L2 Cache Controller (L2CC) must be disabled in the L2CC Control Register prior to any write access to this register.

- **DSETLAT: Setup Latency**
- **DRDLAT: Read Access Latency**
- **DWRLAT: Write Access Latency**

Latency to Data RAM is the programmed value + 1.

Default value is 0.

13.5.7 L2CC Event Counter Control Register

Name: L2CC_ECR

Address: 0x00A00200

Access: Read/Write

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
–	–	–	–	–	EVC1RST	EVC0RST	EVCEN

- **EVCEN: Event Counter Enable**

0: Disables Event Counter. This is the default value.

1: Enables Event Counter.

- **EVC0RST: Event Counter 0 Reset**

0: No effect, always read as zero.

1: Resets Counter 0.

- **EVC1RST: Event Counter 1 Reset**

0: No effect, always read as zero.

1: Resets Counter 1.

13.5.8 L2CC Event Counter 1 Configuration Register

Name: L2CC_ECFGR1

Address: 0x00A00204

Access: Read/Write

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
–	–	ESRC				EIGEN	

- **EIGEN: Event Counter Interrupt Generation**

Value	Name	Description
0x0	INT_DIS	Disables (default)
0x1	INT_EN_INCR	Enables with Increment condition
0x2	INT_EN_OVER	Enables with Overflow condition
0x3	INT_GEN_DIS	Disables Interrupt generation

- **ESRC: Event Counter Source**

Value	Name	Description
0x0	CNT_DIS	Counter Disabled
0x1	SRC_CO	Source is CO
0x2	SRC_DRHIT	Source is DRHIT
0x3	SRC_DRREQ	Source is DRREQ
0x4	SRC_DWHIT	Source is DWHIT
0x5	SRC_DWREQ	Source is DWREQ
0x6	SRC_DWTREQ	Source is DWTREQ
0x7	SRC_IRHIT	Source is IRHIT
0x8	SRC_IRREQ	Source is IRREQ
0x9	SRC_WA	Source is WA
0xa	SRC_IPFALLOC	Source is IPFALLOC
0xb	SRC_EPFHIT	Source is EPFHIT
0xc	SRC_EPFALLOC	Source is EPFALLOC
0xd	SRC_SRRCVD	Source is SRRCVD
0xe	SRC_SRCONF	Source is SRCONF
0xf	SRC_EPFRCVD	Source is EPFRCVD

13.5.9 L2CC Event Counter 0 Configuration Register

Name: L2CC_ECFGR0

Address: 0x00A00208

Access: Read/Write

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
–	–	ESRC				EIGEN	

- **EIGEN: Event Counter Interrupt Generation**

Value	Name	Description
0x0	INT_DIS	Disables (default)
0x1	INT_EN_INCR	Enables with Increment condition
0x2	INT_EN_OVER	Enables with Overflow condition
0x3	INT_GEN_DIS	Disables Interrupt generation

- **ESRC: Event Counter Source**

Value	Name	Description
0x0	CNT_DIS	Counter Disabled
0x1	SRC_CO	Source is CO
0x2	SRC_DRHIT	Source is DRHIT
0x3	SRC_DRREQ	Source is DRREQ
0x4	SRC_DWHIT	Source is DWHIT
0x5	SRC_DWREQ	Source is DWREQ
0x6	SRC_DWTREQ	Source is DWTREQ
0x7	SRC_IRHIT	Source is IRHIT
0x8	SRC_IRREQ	Source is IRREQ
0x9	SRC_WA	Source is WA
0xa	SRC_IPFALLOC	Source is IPFALLOC
0xb	SRC_EPFHIT	Source is EPFHIT
0xc	SRC_EPFALLOC	Source is EPFALLOC
0xd	SRC_SRRCVD	Source is SRRCVD
0xe	SRC_SRCONF	Source is SRCONF
0xf	SRC_EPFRCVD	Source is EPFRCVD

13.5.10 L2CC Event Counter 1 Value Register

Name: L2CC_EVR1

Address: 0x00A0020C

Access: Read/Write

31	30	29	28	27	26	25	24
VALUE							

23	22	21	20	19	18	17	16
VALUE							

15	14	13	12	11	10	9	8
VALUE							

7	6	5	4	3	2	1	0
VALUE							

Note: Counter 1 must be disabled in the L2CC Event Counter 1 Configuration Register prior to any write access to this register.

- **VALUE: Event Counter Value**

Value returns the number of instance of the selected event.

If a counter reaches its maximum value, it remains saturated at that value until it is reset.

13.5.11 L2CC Event Counter 0 Value Register

Name: L2CC_EVR0

Address: 0x00A00210

Access: Read/Write

31	30	29	28	27	26	25	24
VALUE							

23	22	21	20	19	18	17	16
VALUE							

15	14	13	12	11	10	9	8
VALUE							

7	6	5	4	3	2	1	0
VALUE							

Note: Counter 0 must be disabled in the L2CC Event Counter 0 Configuration Register prior to any write access to this register.

- **VALUE: Event Counter Value**

Value returns the number of instance of the selected event.

If a counter reaches its maximum value, it remains saturated at that value until it is reset.

13.5.12 L2CC Interrupt Mask Register

Name: L2CC_IMR

Address: 0x00A00214

Access: Programmable in Auxiliary Control Register

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DECERR

7	6	5	4	3	2	1	0
SLVERR	ERRRD	ERRRT	ERRWD	ERRWT	PARRD	PARRT	ECNTR

- **ECNTR: Event Counter 1/0 Overflow Increment**
- **PARRT: Parity Error on L2 Tag RAM, Read**
- **PARRD: Parity Error on L2 Data RAM, Read**
- **ERRWT: Error on L2 Tag RAM, Write**
- **ERRWD: Error on L2 Data RAM, Write**
- **ERRRT: Error on L2 Tag RAM, Read**
- **ERRRD: Error on L2 Data RAM, Read**
- **SLVERR: SLVERR from L3 Memory**
- **DECERR: DECERR from L3 Memory**

0: Masked. This is the default value.

1: Enabled.

13.5.13 L2CC Masked Interrupt Status Register

Name: L2CC_MISR

Address: 0x00A00218

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DECERR

7	6	5	4	3	2	1	0
SLVERR	ERRRD	ERRRT	ERRWD	ERRWT	PARRD	PARRT	ECNTR

- **ECNTR: Event Counter 1/0 Overflow Increment**
- **PARRT: Parity Error on L2 Tag RAM, Read**
- **PARRD: Parity Error on L2 Data RAM, Read**
- **ERRWT: Error on L2 Tag RAM, Write**
- **ERRWD: Error on L2 Data RAM, Write**
- **ERRRT: Error on L2 Tag RAM, Read**
- **ERRRD: Error on L2 Data RAM, Read**
- **SLVERR: SLVERR from L3 memory**
- **DECERR: DECERR from L3 memory**

0: No interrupt has been generated or the interrupt is masked.

1: The input lines have triggered an interrupt.

13.5.14 L2CC Raw Interrupt Status Register

Name: L2CC_RISR

Address: 0x00A0021C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DECERR

7	6	5	4	3	2	1	0
SLVERR	ERRRD	ERRRT	ERRWD	ERRWT	PARRD	PARRT	ECNTR

- **ECNTR: Event Counter 1/0 Overflow Increment**
- **PARRT: Parity Error on L2 Tag RAM, Read**
- **PARRD: Parity Error on L2 Data RAM, Read**
- **ERRWT: Error on L2 Tag RAM, Write**
- **ERRWD: Error on L2 Data RAM, Write**
- **ERRRT: Error on L2 Tag RAM, Read**
- **ERRRD: Error on L2 Data RAM, Read**
- **SLVERR: SLVERR from L3 memory**
- **DECERR: DECERR from L3 memory**

0: No interrupt has been generated.

1: The input lines have triggered an interrupt.

13.5.15 L2CC Interrupt Clear Register

Name: L2CC_ICR

Address: 0x00A00220

Access: Programmable in Auxiliary Control Register

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DECERR

7	6	5	4	3	2	1	0
SLVERR	ERRRD	ERRRT	ERRWD	ERRWT	PARRD	PARRT	ECNTR

- **ECNTR: Event Counter 1/0 Overflow Increment**
- **PARRT: Parity Error on L2 Tag RAM, Read**
- **PARRD: Parity Error on L2 Data RAM, Read**
- **ERRWT: Error on L2 Tag RAM, Write**
- **ERRWD: Error on L2 Data RAM, Write**
- **ERRRT: Error on L2 Tag RAM, Read**
- **ERRRD: Error on L2 Data RAM, Read**
- **SLVERR: SLVERR from L3 memory**
- **DECERR: DECERR from L3 memory**

0: No effect. Read returns zero.

1: Clears the corresponding bit in the Raw Interrupt Status Register.

13.5.16 L2CC Cache Synchronization Register

Name: L2CC_CSR

Address: 0x00A00730

Access: Read/Write

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
–	–	–	–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

13.5.17 L2CC Invalidate Physical Address Line Register

Name: L2CC_IPALR

Address: 0x00A00770

Access: Read/Write

31	30	29	28	27	26	25	24
TAG							

23	22	21	20	19	18	17	16
TAG							

15	14	13	12	11	10	9	8
TAG		IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index Number**
- **TAG: Tag Number**

13.5.18 L2CC Invalidate Way Register

Name: L2CC_IWR

Address: 0x00A0077C

Access: Read/Write

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
WAY7	WAY6	WAY5	WAY4	WAY3	WAY2	WAY1	WAY0

- **WAYx: Invalidate Way Number x**

0: The corresponding way is totally invalidated.

1: Invalidates the way. This bit is read as ‘1’ as long as invalidation of the way is in progress.

13.5.19 L2CC Clean Physical Address Line Register

Name: L2CC_CPALR

Address: 0x00A007B0

Access: Read/Write

31	30	29	28	27	26	25	24
TAG							

23	22	21	20	19	18	17	16
TAG							

15	14	13	12	11	10	9	8
TAG		IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index number**
- **TAG: Tag number**

13.5.20 L2CC Clean Index Register

Name: L2CC_CIR

Address: 0x00A007B8

Access: Read/Write

31	30	29	28	27	26	25	24
–	WAY			–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index number**
- **WAY: Way number**

13.5.21 L2CC Clean Way Register

Name: L2CC_CWR

Address: 0x00A007BC

Access: Read/Write

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
WAY7	WAY6	WAY5	WAY4	WAY3	WAY2	WAY1	WAY0

- **WAYx: Clean Way Number x**

0: The corresponding way is totally cleaned.

1: Cleans the way. This bit is read as ‘1’ as long as cleaning of the way is in progress.

13.5.22 L2CC Clean Invalidate Physical Address Line Register

Name: L2CC_CIPALR

Address: 0x00A007F0

Access: Read/Write

31	30	29	28	27	26	25	24
TAG							

23	22	21	20	19	18	17	16
TAG							

15	14	13	12	11	10	9	8
TAG		IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index Number**
- **TAG: Tag Number**

13.5.23 L2CC Clean Invalidate Index Register

Name: L2CC_CIIR

Address: 0x00A007F8

Access: Read/Write

31	30	29	28	27	26	25	24
–	WAY			–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index Number**
- **WAY: Way Number**

13.5.24 L2CC Clean Invalidate Way Register

Name: L2CC_CIWR

Address: 0x00A007FC

Access: Read/Write

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
WAY7	WAY6	WAY5	WAY4	WAY3	WAY2	WAY1	WAY0

- **WAYx: Clean Invalidate Way Number x**

0: The corresponding way is totally invalidated and cleaned.

1: Invalidates and cleans the way. This bit is read as ‘1’ as long as invalidation and cleaning of the way is in progress.

13.5.25 L2CC Data Lockdown Register

Name: L2CC_DLKR

Address: 0x00A00900

Access: Programmable in Auxiliary Control Register

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
DLK7	DLK6	DLK5	DLK4	DLK3	DLK2	DLK1	DLK0

- **DLKx: Data Lockdown in Way Number x**

0: Allocation can occur in the corresponding way.

1: There is no allocation in the corresponding way.

13.5.26 L2CC Instruction Lockdown Register

Name: L2CC_ILKR

Address: 0x00A00904

Access: Programmable in Auxiliary Control Register

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
ILK7	ILK6	ILK5	ILK4	ILK3	ILK2	ILK1	ILK0

- **ILKx: Instruction Lockdown in Way Number x**

0: Allocation can occur in the corresponding way.

1: There is no allocation in the corresponding way.

13.5.27 L2CC Debug Control Register

Name: L2CC_DCR

Address: 0x00A00F40

Access: Read/Write in Secure mode

Read-only in Non-secure mode

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
–	–	–	–	–	SPNIDEN	DWB	DCL

- **DCL: Disable Cache Linefill**

0: Enables cache linefills. This is the default value.

1: Disables cache linefills.

- **DWB: Disable Write-back, Force Write-through**

0: Enables write-back behavior. This is the default value.

1: Forces write-through behavior.

- **SPNIDEN: SPNIDEN Value**

Reads value of the SPNIDEN input.

13.5.28 L2CC Prefetch Control Register

Name: L2CC_PCR

Address: 0x00A00F60

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	DLEN	INSPEN	DATPEN	DLFWRDIS	–	–	PDEN
23	22	21	20	19	18	17	16
IDLEN	–	NSIDEN	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	OFFSET				

- **OFFSET: Prefetch Offset**

You must only use the Prefetch offset values of 0-7, 15, 23, and 31 for these bits. The L2CC does not support the other values.

- **NSIDEN: Not Same ID on Exclusive Sequence Enable**

0: Read and write portions of a non-cacheable exclusive sequence have the same AXI ID when issued to L3. This is the default value.

1: Read and write portions of a non-cacheable exclusive sequence do not have the same AXI ID when issued to L3.

- **IDLEN: INCR Double Linefill Enable**

0: The L2CC does not issue INCR 8x64-bit read bursts to L3 on reads that miss in the L2 cache. This is the default value.

1: The L2CC can issue INCR 8x64-bit read bursts to L3 on reads that miss in the L2 cache.

Note: This bit can only be used if the DLEN bit is set HIGH. Refer to Section 13.4.1 “Double Linefill Issuing” for details on double linefill functionality.

- **PDEN: Prefetch Drop Enable**

0: The L2CC does not discard prefetch reads issued to L3. This is the default value.

1: The L2CC discards prefetch reads issued to L3 when there is a resource conflict with explicit reads.

- **DLFWRDIS: Double Linefill on WRAP Read Disable**

0: Double linefill on WRAP read is enabled. This is the default value.

1: Double linefill on WRAP read is disabled.

Note: This bit can only be used if the DLEN bit is set HIGH. Refer to Section 13.4.1 “Double Linefill Issuing” for details on double linefill functionality.

- **DATPEN: Data Prefetch Enable**

0: Data prefetching is disabled. This is the default value.

1: Data prefetching is enabled.

- **INSPEN: Instruction Prefetch Enable**

0: Instruction prefetching is disabled. This is the default value.

1: Instruction prefetching is enabled.

- **DLEN: Double Linefill Enable**

0: The L2CC always issues 4x64-bit read bursts to L3 on reads that miss in the L2 cache. This is the default value.

1: The L2CC issues 8x64-bit read bursts to L3 on reads that miss in the L2 cache.

Refer to Section 13.4.1 “Double Linefill Issuing” for details on double linefill functionality.

13.5.29 L2CC Power Control Register

Name: L2CC_POWCR

Address: 0x00A00F80

Access: Read/Write in Secure mode

Read-only in Non-secure mode

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
–	–	–	–	–	–	DCKGATEN	STBYEN

- **STBYEN: Standby Mode Enable**

0: Disabled. This is the default value.

1: Enabled.

- **DCKGATEN: Dynamic Clock Gating Enable**

0: Disabled. This is the default value.

1: Enabled.

14. AXI Matrix (AXIMX)

14.1 Description

The AXI Matrix comprises the embedded Advanced Extensible Interface (AXI) bus protocol which supports separate address/control and data phases, unaligned data transfers using byte strobes, burst-based transactions with only start address issued, separate read and write data channels to enable low-cost DMA, ability to issue multiple outstanding addresses, out-of-order transaction completion, and easy addition of register stages to provide timing closure.

14.2 Embedded Characteristics

- High performance AXI network interconnect
- 1 Master:
  - Cortex A5 Core
- 4 Slaves:
  - ROM
  - PKCC RAM
  - PKCC ROM
  - AXI/AHB bridge to AHB Matrix
- Single-cycle arbitration
- Full pipelining to prevent master stalls
- 1 remap state

14.3 Operation

14.3.1 Remap

Remap states are managed in the AXI Matrix Remap Register (AXIMX_REMAP): AXIMX_REMAP.REMAP0 (register bit 0) is used to remap RAM @ addr 0x00000000.

Refer to Section 14.4 “AXI Matrix (AXIMX) User Interface”.

The number of remap states can be defined using eight bits of the AXIMX_REMAP register, and a bit in AXIMX_REMAP controls each remap state.

Each remap state can be used to control the address decoding for one or more slave interfaces. If a slave interface is affected by two remap states that are both asserted, the remap state with the lowest remap bit number takes precedence.

Each slave interface can be configured independently so that a remap state can perform different functions for different masters.

A remap state can:

- Alias a memory region into two different address ranges
- Move an address region
- Remove an address region

Because of the nature of the distributed register subsystem, the masters receive the updated remap bit states in sequence, and not simultaneously.

A slave interface does not update to the latest remap bit setting until:

- The address completion handshake accepts any transaction that is pending
- Any current lock sequence completes

At powerup, ROM is seen at address 0. After powerup, the internal SRAM can be moved down to address 0 by means of the remap bits.

14.4 AXI Matrix (AXIMX) User Interface

Table 14-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	AXI Matrix Remap Register	AXIMX_REMAP	Write-only	–
0x04–0x43108	Reserved	–	–	–

14.4.1 AXI Matrix Remap Register

Name: AXIMX_REMAP

Address: 0x00700000

Access: Read/Write

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
–	–	–	–	–	–	–	REMAP0

- **REMAP0: Remap State 0**

SRAM is seen at address 0x00000000 (through AHB slave interface) instead of ROM.

15. Matrix (H64MX/H32MX)

15.1 Description

In order to reduce power consumption without loss in performance, the system embeds three matrixes: one based on AXI protocol (AXIMX) and two based on AHB protocol (H64MX and H32MX). The description of the 64-bit AHB Matrix (H64MX) and the 32-bit AHB Matrix (H32MX) implementation follows.

Refer to description of product AXIMX for complete information on the AXI Matrix.

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

Note: When a master and a slave are on different bus matrixes (AXIMX, H64MX, or H32MX), both matrixes (H64MX and H32MX) and the bridge between the bus matrixes must be configured accordingly.

15.2 Embedded Characteristics

- 32-bit or 64-bit data bus
- MATRIX0—a 64-bit AHB matrix (H64MX) providing 10 masters for 13 slaves
- MATRIX1—a 32-bit AHB matrix (H32MX) providing 7 masters for 7 slaves
- One address decoder 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
- ARM TrustZone technology

15.3 MATRIX0 (H64MX)

15.3.1 Matrix Masters

The H64MX manages 10 masters, which means that each master can perform an access concurrently with others, to an available slave.

This matrix operates at MCK.

Each master has its own decoder, which is defined specifically for each master. In order to simplify the addressing, all the masters have the same decodings.

Table 15-1. List of H64MX Masters

Master No.	Name
0	Bridge from AXI matrix (Core)
1, 2	DMA Controller 0
3, 4	DMA Controller 1
5, 6	LCDC DMA
7	Video Decoder DMA
8	ISI DMA
9	Bridge from H32MX to H64MX

15.3.2 Matrix Slaves

The H64MX manages 13 slaves. Each slave has its own arbitrator providing a dedicated arbitration per slave.

Table 15-2. List of H64MX Slaves

Slave No.	Description	TZ Access Management
0	Bridge from H64MX to AXIMX (Internal ROM, Crypto Library, PKCC RAM)	Always Secured
1	H64MX Peripheral Bridge	–
2	Video Decoder	Programmable Secure(1)
3	DDR2 Port0 - AESB	Scalable Secure
4	DDR2 Port1	Scalable Secure
5	DDR2 Port2	Scalable Secure
6	DDR2 Port3	Scalable Secure
7	DDR2 Port4	Scalable Secure
8	DDR2 Port5	Scalable Secure
9	DDR2 Port6	Scalable Secure
10	DDR2 Port7	Scalable Secure
11	Internal SRAM 128K	Internal Secure
12	Bridge from H64MX to H32MX	–

Notes: 1. This AHB slave is programmed like APB slave in the MATRIX_SPSELRx.

15.3.3 Master to Slave Access

Table 15-3 gives the interconnect between all the masters and slaves. Writing in a register or field not dedicated to a master or a slave will have no effect.

Table 15-3. Master to Slave Access on H64MX

		Master									
		0	1	2	3	4	5	6	7	8	9
Slave		Bridge from AXIMX (Core)	XDMAC0		XDMAC1		LCDC DMA		VDEC DMA	ISI DMA	Bridge from H32MX
			IF0	IF1	IF0	IF1					
0	Bridge from H64MX to AXIMX		X	X							
1	H64MX Peripheral Bridge	X		X		X					
2	VDEC	X									
3	DDR2 port0 - AESB	X									
4	DDR2 port1	X									
5	DDR2 port2						X				
6	DDR2 port3							X			
7	DDR2 port4								X	X	
8	DDR2 port5		X		X						
9	DDR2 port6			X		X					
10	DDR2 port7										X
11	Internal SRAM	X	X				X				X
12	Bridge from H64MX to H32MX	X	X	X	X	X					

15.4 MATRIX1 (H32MX)

15.4.1 Matrix Masters

The H32MX manages seven masters, which means that each master can perform an access concurrently with others, to an available slave.

This matrix can operate at MCK if MCK is lower than 90 MHz, or at MCK/2 if MCK is higher than 90 MHz. Refer to Section 26. “Power Management Controller (PMC)” for more details.

Each master has its own decoder, which is defined specifically for each master. In order to simplify the addressing, all the masters have the same decodings.

Table 15-4. List of H32MX Masters

Master No.	Name
0	Bridge from H64MX to H32MX
1	Integrity Check Monitor (ICM)
2	UHP EHCI DMA
3	UHP OHCI DMA

Table 15-4. List of H32MX Masters (Continued)

4	UDPHS DMA
5	GMAC0 DMA
6	GMAC1 DMA

15.4.2 Matrix Slaves

The H32MX manages seven slaves. Each slave has its own arbitrator providing a dedicated arbitration per slave.

Table 15-5. List of H32MX Slaves

Slave No.	Description	TZ Access Management
0	Bridge from H32MX to H64MX	–
1	H32MX Peripheral Bridge 0	–
2	H32MX Peripheral Bridge 1	–
3	External Bus Interface	External Secure
3	NFC command register	Programmable Secure(1)
4	NFC SRAM	Programmable Secure(1)
5	USB Device High Speed Dual Port RAM (DPR)	Programmable Secure(1)
	USB Host OHCI registers	
	USB Host EHCI registers	
6	Soft Modem (SMD)	Programmable Secure(1)

Notes: 1. These AHB slaves are programmed like APB slaves in the MATRIX_SPSELRx.

15.4.3 Master to Slave Access

Table 15-6 gives the interconnect between all the masters and slaves. Writing in a register or field not dedicated to a master or a slave will have no effect.

Table 15-6. Master to Slave Access on H32MX

		Master										
		0 (through Bridge from H64MX)					1	2	3	4	5	6
Slave		Core	XDMAC0 IF0	XDMAC0 IF1	XDMAC1 IF0	XDMAC1 IF1	ICM	UHPHS EHCI DMA	UHPHS OHCI DMA	UDPHS DMA	GMAC 0 DMA	GMAC 1 DMA
0	Bridge from H32MX to H64MX						X	X	X	X	X	X
1	H32MX APB0 - user interfaces	X		X		X	X					
2	H32MX APB1 - user interfaces	X		X		X	X					
3	EBI CS0..CS3	X	X		X		X					
	NFC Command Register	X	X		X							
4	NFC SRAM	X	X		X							

Table 15-6. Master to Slave Access on H32MX (Continued)

		Master										
		0 (through Bridge from H64MX)					1	2	3	4	5	6
Slave		Core	XDMAC0		XDMAC1		ICM	UHPHS EHCI DMA	UHPHS OHCI DMA	UDPHS DMA	GMAC 0 DMA	GMAC 1 DMA
			IF0	IF1	IF0	IF1						
5	UDPHS RAM	X				X						
	UHP OHCI Register	X				X						
	UHP EHCI Register	X				X						
6	SMD	X			X							

15.5 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 ROM or internal Flash, etc.) becomes possible.

15.6 Special Bus Granting Mechanism

The Bus Matrix provides some speculative bus granting techniques in order to anticipate access requests from masters. This mechanism reduces latency 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 mechanism 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 Bus Matrix user interface provides Slave Configuration Registers, one for every slave, which set a default master for each slave. The Slave Configuration Register contains 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 15.13.2 “Bus Matrix Slave Configuration Registers”.

15.7 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 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 regardless of the number of requesting masters.

15.8 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 allows 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 used 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.

15.9 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 allows 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 will get one latency cycle. This technique is used 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.

15.10 Arbitration

The Bus Matrix provides an arbitration mechanism that reduces latency when conflicts 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 Bus Matrix provides the user with the possibility of choosing between two arbitration types or mixing them for each slave:

- Round-robin Arbitration (default)
- 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. Refer to Section 15.10.1 “Arbitration Scheduling”.

15.10.1 Arbitration Scheduling

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

- Idle Cycles: when a slave is not connected to any master or is connected to a master which is not currently accessing it.
- Single Cycles: when a slave is currently performing a single access.
- End of Burst Cycles: when the current cycle is the last cycle of a burst transfer. For defined burst length, predicted end of burst matches the size of the transfer but is managed differently for undefined burst length. Refer to Section 15.10.1.1 “Undefined Length Burst Arbitration”.
- 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. Refer to Section 15.10.1.2 “Slot Cycle Limit Arbitration”.

15.10.1.1 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).

15.10.1.2 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 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.

15.10.2 Arbitration Priority Scheme

The bus 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 Table 15-7.

Table 15-7. 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. Refer to Section 15.10.2.2 “Round-robin Arbitration”.

For each slave, each master is assigned to one of the slave priority pools through 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, they will be granted bus access in a biased round-robin manner which allows tight and deterministic maximum access latency from AHB requests. In the worst case, any currently occurring high-priority master request will be granted after the current bus master access has ended and other high priority pool master requests, if any, have been granted once each.

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

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

For good CPU performance, it is recommended configure CPU priority 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 fix priority levels.

15.10.2.1 Fixed Priority Arbitration

Fixed priority arbitration algorithm is the first and only arbitration algorithm applied between masters from distinct priority pools. It is also used in priority pools other than the highest and lowest priority pools (intermediate priority pools).

Fixed priority arbitration allows the Bus Matrix arbiters to dispatch the requests from different masters to the same slave by using the fixed priority defined by the user in the MxPR field for each master in the Priority Registers, MATRIX_PRAS and MATRIX_PRBS. If two or more master requests are active at the same time, the master with the highest priority MxPR number is serviced first.

In intermediate priority pools, if two or more master requests with the same priority are active at the same time, the master with the highest number is serviced first.

15.10.2.2 Round-robin Arbitration

This algorithm is only used in the highest and lowest priority pools. It allows the Bus Matrix arbiters to properly dispatch requests from different masters to the same slave. If two or more master requests are active at the same time in the priority pool, they are serviced in a round-robin increasing master number order.

15.11 Register Write Protection

To prevent any single software error from corrupting Bus 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 bit 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 Bus Matrix Write Protect Mode Register (MATRIX_WPMR) with the appropriate access key WPKEY.

The following registers can be write-protected:

- Bus Matrix Master Configuration Registers
- Bus Matrix Slave Configuration Registers
- Bus Matrix Priority Registers A For Slaves
- Bus Matrix Priority Registers B For Slaves
- Master Error Interrupt Enable Register
- Master Error Interrupt Disable Register
- Security Slave Registers
- Security Areas Split Slave Registers
- Security Region Top Slave Registers
- Security Peripheral Select x Registers

15.12 TrustZone Extension to AHB and APB

TrustZone secure software is supported through the filtering of each slave access with master security bit AMBA hprot[6] extension signals.

The TrustZone extension adds the ability to manage the access rights for Secure and Non-Secure accesses. The access rights are defined through the hardware and software configuration of the device. The operating mode is the following:

- With the TrustZone extension, the Bus Masters transmit requests with the Secure or Non-Secure Security option.
- The AHB Matrix, according to its configuration and the request, grants or denies the access.

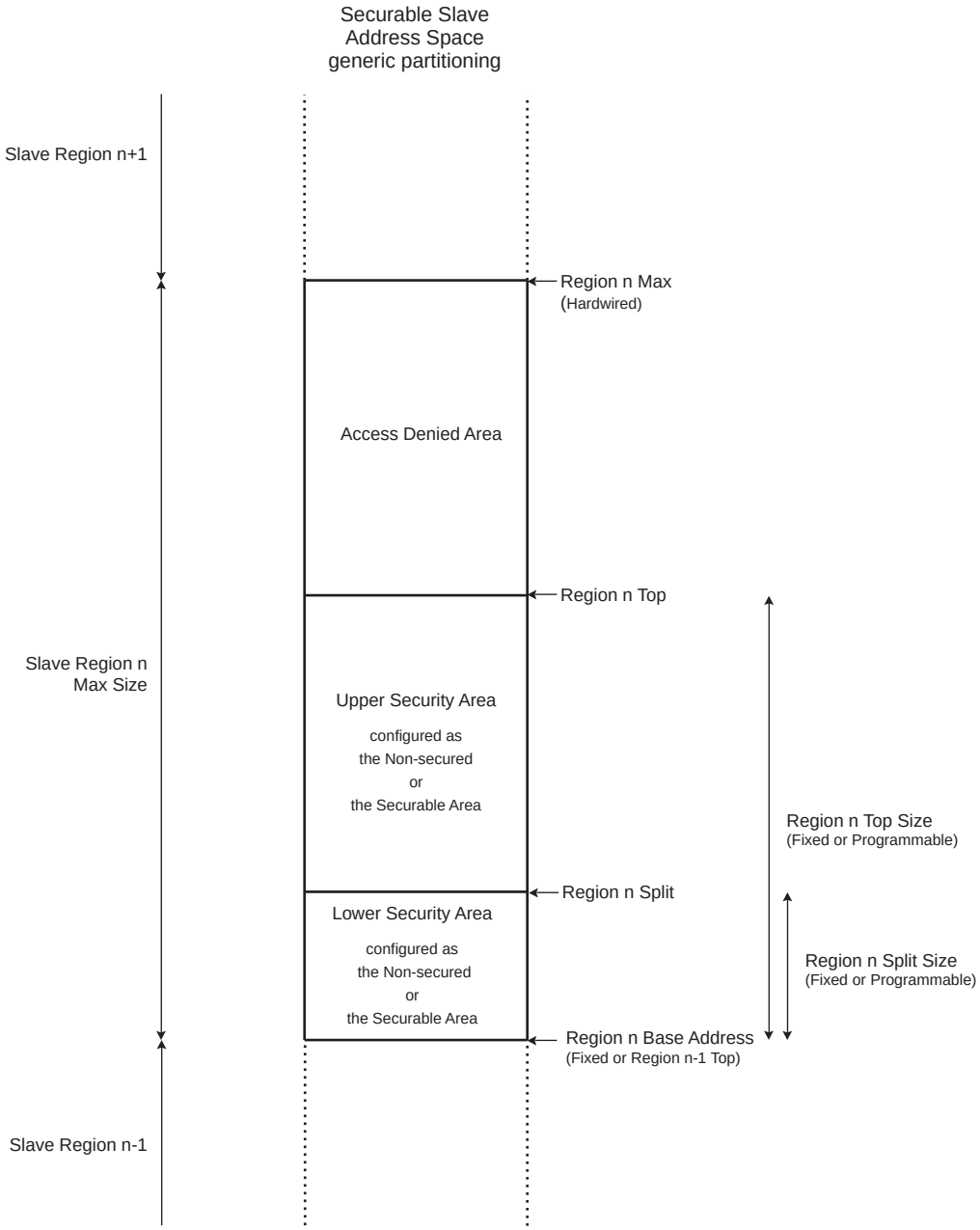
The slave address space is divided into one or more slave regions. The slave regions are generally contiguous parts of the slave address space. The slave region is potentially split into an access denied area (upper part) and a security region which can be split (lower part), unless the slave security region occupies the whole slave region. The security region itself may or may not be split into one Securable area and one Non-secured area. The Securable area may be independently secured 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
- Max Size of the slave region: the maximum size for the region’s physical content
- Top Size of the slave security region: the programmed or fixed size for the region’s physical content
- Split Size of the slave security region: the size of the lower security area of the region.

Figure 15-1 shows how the terms defined here are implemented in an AHB slave address space.

Figure 15-1. Generic Partitioning of the AHB Slave Address Space



A set of Bus Matrix security registers allows to specify, for each AHB slave, slave security region or slave security area, the security mode required for accessing this slave, slave security region or slave security area.

Additional Bus Matrix security registers allow to specify, for each APB slave, the security mode required for accessing this slave (refer to Section 15.13.15 “Security Peripheral Select x Registers”).

Refer to Section 15.13.12 “Security Slave Registers”.

The Bus Matrix registers can only be accessed in Secure Mode.

The Bus Matrix propagates the AHB security bit down to the AHB slaves to let them perform additional security checks, and the Bus Matrix itself allows, or not, the access to the slaves by means of its TrustZone embedded controller.

Access violations may be reported either by an AHB slave through the bus error response (example from the AHB/APB Bridge), or by the Bus Matrix embedded TrustZone controller. 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 to the Secure world, 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 security region are also known.

Depending on the hardware parameters and software configuration, the address space of each AHB slave security region may or may not be split into two parts, one belonging to the Secure world and the other one to the Normal world.

Five different security types of AHB slaves are supported. The number of security regions is fixed by design for each slave, independently, from 1 to 8, totalling from 1 up to 16 security areas for security configurable slaves.

15.12.1 Security Types of AHB Slaves

15.12.1.1Principles

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

- Always Non-secured
- Always Secured
- Internal Securable
- External Securable
- Scalable Securable

The security type is fixed at hardware design on a per-master and a per-slave basis. **Always Non-secured** and **Always Secured** security types are not software configurable.

The different security types have the following characteristics:

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

avoids mirroring Secured areas into Non-secured areas, and further restricts the overall accessible address range. Any access with security or configured address range violation is denied and reported.

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

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

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

15.12.1.2Examples

Table 15-8 shows an example of Security Type settings.

Table 15-8. Security Type Setting Example

Slave	Master0	Master1	Master2
Slave0 Internal Memory	Always Non-secured	Internal Securable 1 Region	Internal Securable 1 Region
Slave1 EBI	External Securable 2 Regions	Always Secured	External Securable 2 Regions

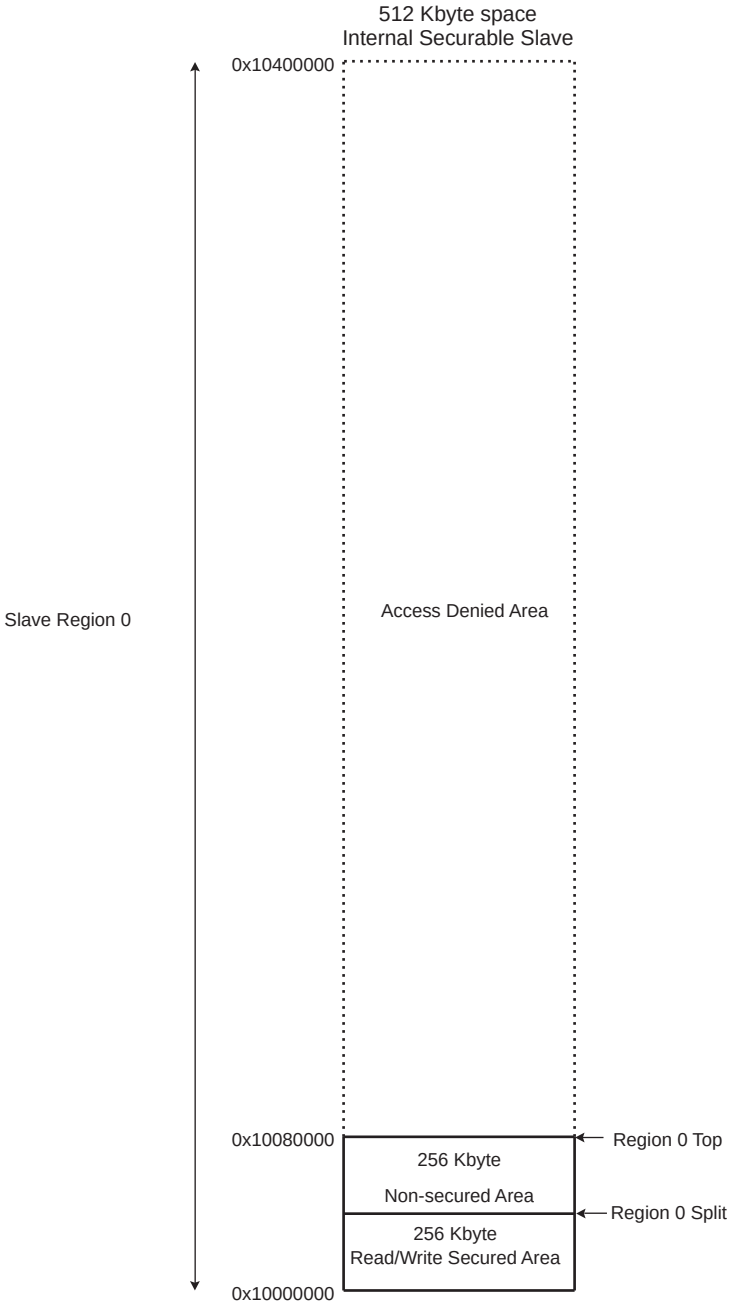
This example is constructed with the following characteristics:

- Slave0 is an Internal Memory containing one region:
  - The Access from Master0 to Slave0 is Always Non-secured
  - The Access from Master1 and Master2 to Slave0 is Internal Securable with one region and with the same Software Configuration (Choice of SPLIT0 and the Security Configuration bits LANSECH, RDNSECH, WRNSECH).
- Slave1 is an EBI containing two regions:
  - The Access from Master1 to Slave1 is Always Secured
  - The Access from Master0 and Master2 to Slave1 is External Securable with two regions and with the same Software Configuration (Choice of TOP0, TOP1, SPLIT0, SPLIT1 and the Security Configuration bits LANSECH, RDNSECH, WRNSECH).

Figure 15-2 shows an Internal Securable slave example. This example is constructed with the following hypothesis:

- The slave is an Internal Memory containing one region. The Slave region Max Size is 4 Mbytes.
- The slave region 0 base address equals 0x10000000. Its Top Size is 512 Kbytes (hardware configuration).
- The slave software configuration is:
  - SPLIT0 is set to 256 Kbytes
  - LANSECH0 is set to 0, the low area of region 0 is the Securable one
  - RDNSECH0 is set to 0, region 0 Securable area is secured for reads
  - WRNSECH0 is set to 0, region 0 Securable area is secured for writes

Figure 15-2. Partitioning Example of an Internal Securable Slave Featuring 1 Security Region of 512 KB Split into 1 or 2 Security Areas of 4 KB to 512 KB



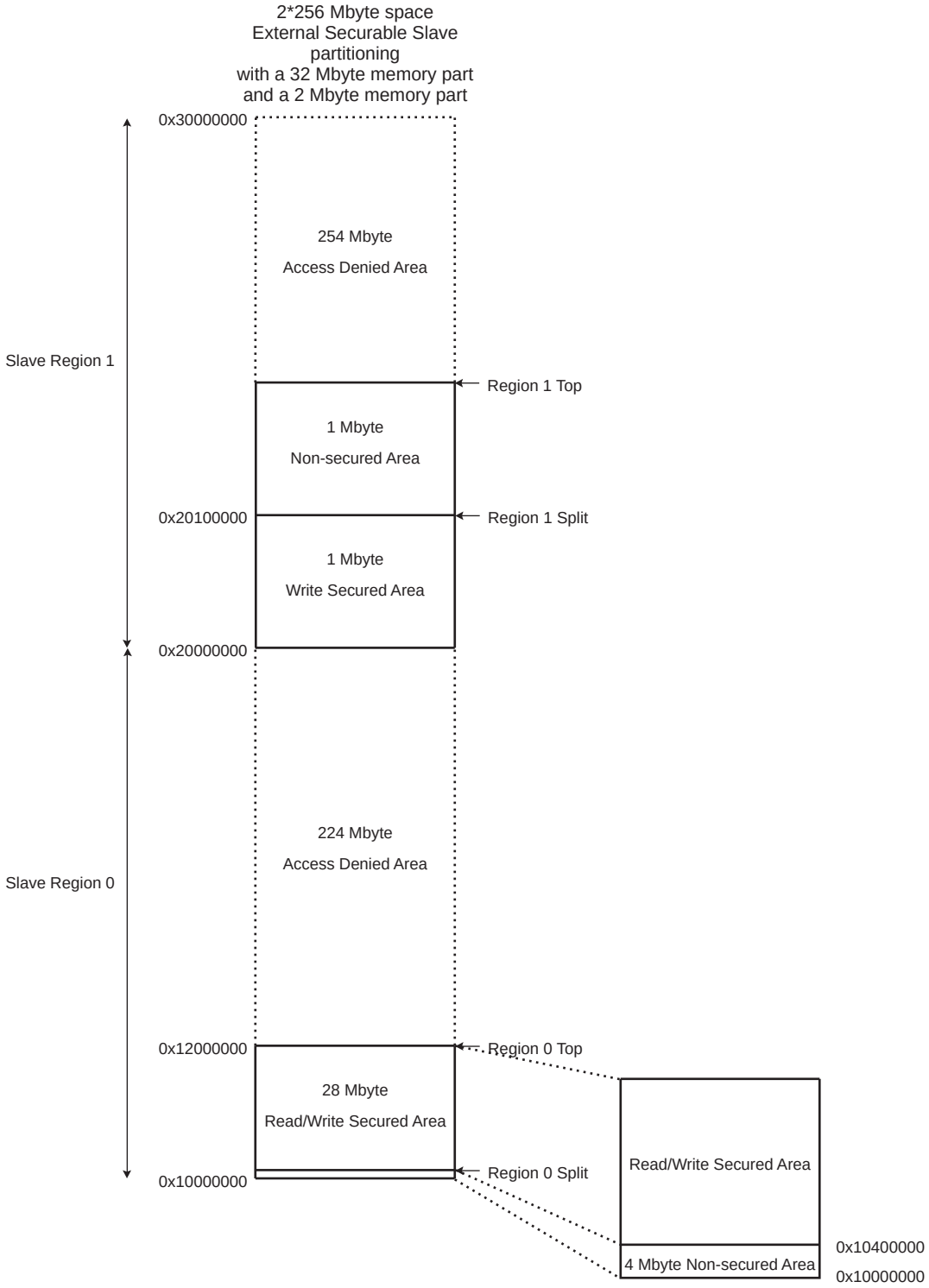
Note: The slave security areas split inside the security region are configured by writing into the Security Areas Split Slave Registers.

Figure 15-3 shows an External Securable slave example. This example is constructed with the following hypothesis:

- The slave is an interface with the external bus (EBI) containing two regions. The slave size is 2 × 256 Mbytes. Each slave region Max Size is 256 Mbytes.
- The slave region 0 base address equals 0x10000000. It is connected to a 32 Mbyte 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 0x20000000. It is connected to a 2 Mbyte memory, for example an external NAND Flash. The slave region 1 Top Size must be set to 2 Mbytes.
- The slave software configuration is:
  - TOP0 is set to 32 Mbytes
  - TOP1 is set to 2 Mbytes
  - SPLIT0 is set to 4 Mbytes
  - SPLIT1 is set to 1 Mbyte
  - LANSECH0 is set to 1, the low area of region 0 is the non-Securable one
  - RDNSECH0 is set to 0, region 0 Securable area is secured for reads
  - WRNSECH0 is set to 0, region 0 Securable area is secured for writes
  - LANSECH1 is set to 0, the low area of region 1 is the Securable one
  - RDNSECH1 is set to 1, region 1 Securable area is Non-secured for reads
  - WRNSECH1 is set to 0, region 1 Securable area is secured for writes

Figure 15-3. Partitioning Example of an External Securable Slave Featuring 2 Security Regions of 4 KB to 128 MB each and up to 4 Security Areas of 4 KB to 128 MB

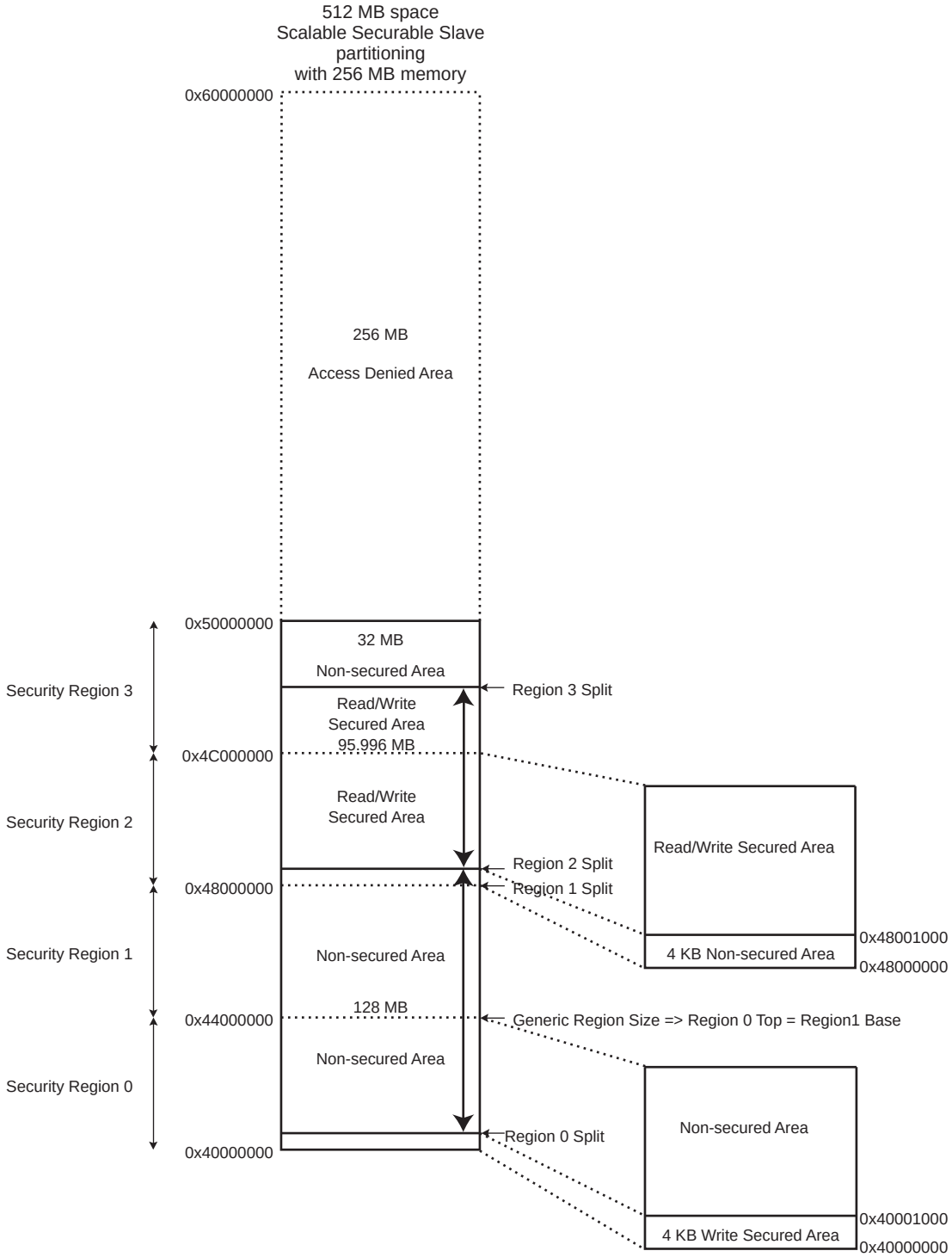


Note: The slave region sizes are configured by writing into the Security Region Top Slave Registers.
The slave security area split inside each region is configured by writing into the Security Areas Split Slave Registers.

Figure 15-4 shows a Scalable Securable slave example. This example is constructed with the following hypothesis:

- The slave is an external memory with dedicated slave containing four regions, for example an external DDR.
- The slave size is 512 Mbytes.
- The slave base address equals 0x40000000. It is connected to a 256 Mbyte external memory.
- As the connected memory size is 256 Mbytes and there are four 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:
  - 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
  - LANSECH0 is set to 0, the low area of region 0 is the Securable one
  - RDNSECH0 is set to 1, region 0 Securable area is Non-secured for reads
  - WRNSECH0 is set to 0, region 0 Securable area is secured for writes
  - LANSECH1 is set to 1, the low area of region 1 is the Non-securable one
  - RDNSECH1 is ‘don’t care’ since the low area occupies the whole region 1
  - WRNSECH1 is ‘don’t care’ since the low area occupies the whole region 1
  - LANSECH2 is set to 1, the low area of region 2 is the Non-securable one
  - RDNSECH2 is set to 0, region 2 Securable area is secured for reads
  - WRNSECH2 is set to 0, region 2 Securable area is secured for writes
  - LANSECH3 is set to 0, the low area of region 3 is the Securable one
  - RDNSECH3 is set to 0, region 3 Securable area is secured for reads
  - WRNSECH3 is set to 0, region 3 Securable area is secured for writes

Figure 15-4. Partitioning Example of a Scalable Securable Slave Featuring 4 Equally-sized Security Regions of 1 MB to 128 MB each and up to 8 Security Areas of 4 KB to 128 MB



Note: The slaves' generic security regions sizes are configured by writing into field SRTOP0 of the Security Region Top Slave Registers and the custom slave security areas splits inside each region is configured by writing into the Security Areas Split Slave Registers.

15.12.2 Security of APB Slaves

To be able to configure the security mode required for accessing a particular APB slave connected to the AHB/APB Bridge, the Bus Matrix features three 32-bit Security Peripheral Select x Registers. Some of these bits may have been fixed to a Secured or a Non-secured value by design, whereas others are programmed by software (refer to Section 15.13.15 “Security Peripheral Select x Registers”).

Peripheral security state, “Secure” or “Non-Secure” is an AND operation between H32MX MATRIX_SPSELRx and H64MX MATRIX_SPSELRx for the bit corresponding to the peripheral.

MATRIX_SPSELRx bits in the H32MX or H64MX user interface are respectively read/write or read-only to ‘1’ depending on whether the peripheral is connected or not, on the Matrix.

All bit values in Table 15-9 except those marked ‘UD’ (User Defined) are read-only and cannot be changed. Values marked ‘UD’ can be changed. Refer to the following examples.

- Example for PIOC, Peripheral ID 25, which is connected to the H32MX Matrix
  - H64MX MATRIX_SPSELR1[25] = 1 (read-only); no influence on the security configuration
  - H32MX MATRIX_SPSELR1[25] can be written by user to program the security.
- Example for LCDC, Peripheral ID 51, which is connected to the H64MX Matrix
  - H64MX MATRIX_SPSELR2[19] can be written by user to program the security.
  - H32MX MATRIX_SPSELR2[19] = 1 (read-only); no influence on the security configuration
- Example for XDMAC1, Peripheral ID 50, which is connected to the H64MX Matrix
  - H64MX MATRIX_SPSELR2[18] = 1 (read-only); sets the peripheral as Non-Secure by hardware, also called “Always Non-Secure”
  - H32MX MATRIX_SPSELR2[18] = 1 (read-only); no influence on the security configuration
- Example for SAIC, Peripheral ID 0, which is connected to the H32MX Matrix
  - H64MX MATRIX_SPSELR1[0] = 1 (read-only); no influence on the security configuration
  - H32MX MATRIX_SPSELR1[0] = 0 (read-only); sets the peripheral as Secure by hardware, also called “Always Secure”

Table 15-9. Peripheral Identifiers

ID	Peripheral	Security Type	Matrix	MATRIX_SPSELRx Bit	Bit Value in H32MX	Bit Value in H64MX
0	SAIC	Always Secure	H32MX	MATRIX_SPSELR1[0]	0	1
1	SYS	Always Secure	H32MX	MATRIX_SPSELR1[1]	0	1
2	ARM	Always Secure	H64MX	MATRIX_SPSELR1[2]	1	0
3	PIT	Always Secure	H32MX	MATRIX_SPSELR1[3]	0	1
4	WDT	Always Secure	H32MX	MATRIX_SPSELR1[4]	0	1
5	PIOD	Always Secure	H32MX	MATRIX_SPSELR1[5]	0	1
6	USART0	Always Secure	H32MX	MATRIX_SPSELR1[6]	0	1
7	USART1	Always Secure	H32MX	MATRIX_SPSELR1[7]	0	1
8	XDMAC0	Always Secure	H64MX	MATRIX_SPSELR1[8]	1	0
9	ICM	Always Secure	H32MX	MATRIX_SPSELR1[9]	0	1
10	PKCC	Always Secure	H64MX	MATRIX_SPSELR1[10]	1	0
11	SCI	Always Secure	H32MX	MATRIX_SPSELR1[11]	0	1
12	AES	Always Secure	H32MX	MATRIX_SPSELR1[12]	0	1
13	AESB	Always Secure	H64MX	MATRIX_SPSELR1[13]	1	0

Table 15-9. Peripheral Identifiers (Continued)

ID	Peripheral	Security Type	Matrix	MATRIX_SPSELRx Bit	Bit Value in H32MX	Bit Value in H64MX
14	TDES	Always Secure	H32MX	MATRIX_SPSELR1[14]	0	1
15	SHA	Always Secure	H32MX	MATRIX_SPSELR1[15]	0	1
16	MPDDRC	Always Secure	H64MX	MATRIX_SPSELR1[16]	1	0
17	MATRIX1	Always Secure	H32MX	MATRIX_SPSELR1[17]	0	1
18	MATRIX0	Always Secure	H64MX	MATRIX_SPSELR1[18]	1	0
19	VDEC	Programmable Secure	H64MX	MATRIX_SPSELR1[19]	1	UD
20	SECUMOD	Always Secure	H32MX	MATRIX_SPSELR1[20]	0	1
21	MSADCC	Always Secure	H32MX	MATRIX_SPSELR1[21]	0	1
22	HSMC	Programmable Secure	H32MX	MATRIX_SPSELR1[22]	UD	1
23	PIOA	Programmable Secure	H32MX	MATRIX_SPSELR1[23]	UD	1
24	PIOB	Programmable Secure	H32MX	MATRIX_SPSELR1[24]	UD	1
25	PIOC	Programmable Secure	H32MX	MATRIX_SPSELR1[25]	UD	1
26	PIOE	Programmable Secure	H32MX	MATRIX_SPSELR1[26]	UD	1
27	UART0	Programmable Secure	H32MX	MATRIX_SPSELR1[27]	UD	1
28	UART1	Programmable Secure	H32MX	MATRIX_SPSELR1[28]	UD	1
29	USART2	Programmable Secure	H32MX	MATRIX_SPSELR1[29]	UD	1
30	USART3	Programmable Secure	H32MX	MATRIX_SPSELR1[30]	UD	1
31	USART4	Programmable Secure	H32MX	MATRIX_SPSELR1[31]	UD	1
32	TWI0	Programmable Secure	H32MX	MATRIX_SPSELR2[0]	UD	1
33	TWI1	Programmable Secure	H32MX	MATRIX_SPSELR2[1]	UD	1
34	TWI2	Programmable Secure	H32MX	MATRIX_SPSELR2[2]	UD	1
35	HSMCI0	Programmable Secure	H32MX	MATRIX_SPSELR2[3]	UD	1
36	HSMCI1	Programmable Secure	H32MX	MATRIX_SPSELR2[4]	UD	1
37	SPI0	Programmable Secure	H32MX	MATRIX_SPSELR2[5]	UD	1
38	SPI1	Programmable Secure	H32MX	MATRIX_SPSELR2[6]	UD	1
39	SPI2	Programmable Secure	H32MX	MATRIX_SPSELR2[7]	UD	1
40	TC0	Programmable Secure	H32MX	MATRIX_SPSELR2[8]	UD	1
41	TC1	Programmable Secure	H32MX	MATRIX_SPSELR2[9]	UD	1
42	TC2	Programmable Secure	H32MX	MATRIX_SPSELR2[10]	UD	1
43	PWM	Programmable Secure	H32MX	MATRIX_SPSELR2[11]	UD	1
44	ADC	Programmable Secure	H32MX	MATRIX_SPSELR2[12]	UD	1
45	DBGU	Programmable Secure	H32MX	MATRIX_SPSELR2[13]	UD	1
46	UHPHS	Programmable Secure	H32MX	MATRIX_SPSELR2[14]	UD	1
47	UDPHS	Programmable Secure	H32MX	MATRIX_SPSELR2[15]	UD	1
48	SSC0	Programmable Secure	H32MX	MATRIX_SPSELR2[16]	UD	1
49	SSC1	Programmable Secure	H32MX	MATRIX_SPSELR2[17]	UD	1
50	XDMAC1	Non-Secure	H64MX	MATRIX_SPSELR2[18]	1	1

Table 15-9. Peripheral Identifiers (Continued)

ID	Peripheral	Security Type	Matrix	MATRIX_SPSELRx Bit	Bit Value in H32MX	Bit Value in H64MX
51	LCDC	Programmable Secure	H64MX	MATRIX_SPSELR2[19]	1	UD
52	ISI	Programmable Secure	H64MX	MATRIX_SPSELR2[20]	1	UD
53	TRNG	Programmable Secure	H32MX	MATRIX_SPSELR2[21]	UD	1
54	GMAC0	Programmable Secure	H32MX	MATRIX_SPSELR2[22]	UD	1
55	GMAC1	Programmable Secure	H32MX	MATRIX_SPSELR2[23]	UD	1
56	AIC	Non-Secure	H32MX	MATRIX_SPSELR2[24]	1	1
57	SFC	Always Secure	H32MX	MATRIX_SPSELR2[25]	0	1
58	Reserved	–	–	–	–	–
59	SECURAM	Always Secure	H32MX	MATRIX_SPSELR2[27]	0	1
60	Reserved	–	–	–	–	–
61	SMD	PS	H32MX	MATRIX_SPSELR2[29]	UD	1
62	TWI3	PS	H32MX	MATRIX_SPSELR2[30]	UD	1
63	Reserved	–	–	–	–	–
64	SFR	Always Secure	H32MX	MATRIX_SPSELR3[0]	0	1
65	AIC	Non-Secure	H32MX	MATRIX_SPSELR3[1]	1	1
66	SAIC	Always Secure	H32MX	MATRIX_SPSELR3[2]	0	1
67	L2CC	Always Secure	H64MX	MATRIX_SPSELR3[3]	1	0
68	PMC	Always Secure	H64MX	MATRIX_SPSELR3[4]	1	0

The AHB/APB Bridge compares the incoming master request security bit with the required security mode for the selected peripheral, and accepts or denies access. In the last case, its bus error response is internally flagged in the Bus Matrix Master Error Status Register; the offending address is registered in the Master Error Address Registers so that the slave and the targeted protected region are also known.

15.13 AHB Matrix (MATRIX) User Interface

Table 15-10. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	Master Configuration Register 0	MATRIX_MCFG0	Read/Write	0x00000004
0x0004	Master Configuration Register 1	MATRIX_MCFG1	Read/Write	0x00000004
0x0008	Master Configuration Register 2	MATRIX_MCFG2	Read/Write	0x00000004
0x000C	Master Configuration Register 3	MATRIX_MCFG3	Read/Write	0x00000004
0x0010	Master Configuration Register 4	MATRIX_MCFG4	Read/Write	0x00000004
0x0014	Master Configuration Register 5	MATRIX_MCFG5	Read/Write	0x00000004
0x0018	Master Configuration Register 6	MATRIX_MCFG6	Read/Write	0x00000004
0x001C	Master Configuration Register 7	MATRIX_MCFG7	Read/Write	0x00000004
0x0020	Master Configuration Register 8	MATRIX_MCFG8	Read/Write	0x00000004
0x0024	Master Configuration Register 9	MATRIX_MCFG9	Read/Write	0x00000004
0x0028–0x003C	Reserved	–	–	–
0x0040	Slave Configuration Register 0	MATRIX_SCFG0	Read/Write	0x000001FF
0x0044	Slave Configuration Register 1	MATRIX_SCFG1	Read/Write	0x000001FF
0x0048	Slave Configuration Register 2	MATRIX_SCFG2	Read/Write	0x000001FF
0x004C	Slave Configuration Register 3	MATRIX_SCFG3	Read/Write	0x000001FF
0x0050	Slave Configuration Register 4	MATRIX_SCFG4	Read/Write	0x000001FF
0x0054	Slave Configuration Register 5	MATRIX_SCFG5	Read/Write	0x000001FF
0x0058	Slave Configuration Register 6	MATRIX_SCFG6	Read/Write	0x000001FF
0x005C	Slave Configuration Register 7	MATRIX_SCFG7	Read/Write	0x000001FF
0x0060	Slave Configuration Register 8	MATRIX_SCFG8	Read/Write	0x000001FF
0x0064	Slave Configuration Register 9	MATRIX_SCFG9	Read/Write	0x000001FF
0x0068	Slave Configuration Register 10	MATRIX_SCFG10	Read/Write	0x000001FF
0x006C	Slave Configuration Register 11	MATRIX_SCFG11	Read/Write	0x000001FF
0x0070	Slave Configuration Register 12	MATRIX_SCFG12	Read/Write	0x000001FF
0x0074–0x007C	Reserved	–	–	–
0x0080	Priority Register A for Slave 0	MATRIX_PRAS0	Read/Write	0x00000000
0x0084	Priority Register B for Slave 0	MATRIX_PRBS0	Read/Write	0x00000000
0x0088	Priority Register A for Slave 1	MATRIX_PRAS1	Read/Write	0x00000000
0x008C	Priority Register B for Slave 1	MATRIX_PRBS1	Read/Write	0x00000000
0x0090	Priority Register A for Slave 2	MATRIX_PRAS2	Read/Write	0x00000000
0x0094	Priority Register B for Slave 2	MATRIX_PRBS2	Read/Write	0x00000000
0x0098	Priority Register A for Slave 3	MATRIX_PRAS3	Read/Write	0x00000000
0x009C	Priority Register B for Slave 3	MATRIX_PRBS3	Read/Write	0x00000000
0x00A0	Priority Register A for Slave 4	MATRIX_PRAS4	Read/Write	0x00000000
0x00A4	Priority Register B for Slave 4	MATRIX_PRBS4	Read/Write	0x00000000

Table 15-10. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x00A8	Priority Register A for Slave 5	MATRIX_PRAS5	Read/Write	0x00000000
0x00AC	Priority Register B for Slave 5	MATRIX_PRBS5	Read/Write	0x00000000
0x00B0	Priority Register A for Slave 6	MATRIX_PRAS6	Read/Write	0x00000000
0x00B4	Priority Register B for Slave 6	MATRIX_PRBS6	Read/Write	0x00000000
0x00B8	Priority Register A for Slave 7	MATRIX_PRAS7	Read/Write	0x00000000
0x00BC	Priority Register B for Slave 7	MATRIX_PRBS7	Read/Write	0x00000000
0x00C0	Priority Register A for Slave 8	MATRIX_PRAS8	Read/Write	0x00000000
0x00C4	Priority Register B for Slave 8	MATRIX_PRBS8	Read/Write	0x00000000
0x00C8	Priority Register A for Slave 9	MATRIX_PRAS9	Read/Write	0x00000000
0x00CC	Priority Register B for Slave 9	MATRIX_PRBS9	Read/Write	0x00000000
0x00D0	Priority Register A for Slave 10	MATRIX_PRAS10	Read/Write	0x00000000
0x00D4	Priority Register B for Slave 10	MATRIX_PRBS10	Read/Write	0x00000000
0x00D8	Priority Register A for Slave 11	MATRIX_PRAS11	Read/Write	0x00000000
0x00DC	Priority Register B for Slave 11	MATRIX_PRBS11	Read/Write	0x00000000
0x00E0	Priority Register A for Slave 12	MATRIX_PRAS12	Read/Write	0x00000000
0x00E4	Priority Register B for Slave 12	MATRIX_PRBS12	Read/Write	0x00000000
0x00E8–0x014C	Reserved	–	–	–
0x0150	Master Error Interrupt Enable Register	MATRIX_MEIER	Write-only	–
0x0154	Master Error Interrupt Disable Register	MATRIX_MEIDR	Write-only	–
0x0158	Master Error Interrupt Mask Register	MATRIX_MEIMR	Read-only	0x00000000
0x015C	Master Error Status Register	MATRIX_MESR	Read-only	0x00000000
0x0160	Master 0 Error Address Register	MATRIX_MEAR0	Read-only	0x00000000
0x0164	Master 1 Error Address Register	MATRIX_MEAR1	Read-only	0x00000000
0x0168	Master 2 Error Address Register	MATRIX_MEAR2	Read-only	0x00000000
0x016C	Master 3 Error Address Register	MATRIX_MEAR3	Read-only	0x00000000
0x0170	Master 4 Error Address Register	MATRIX_MEAR4	Read-only	0x00000000
0x0174	Master 5 Error Address Register	MATRIX_MEAR5	Read-only	0x00000000
0x0178	Master 6 Error Address Register	MATRIX_MEAR6	Read-only	0x00000000
0x017C	Master 7 Error Address Register	MATRIX_MEAR7	Read-only	0x00000000
0x0180	Master 8 Error Address Register	MATRIX_MEAR8	Read-only	0x00000000
0x0184	Master 9 Error Address Register	MATRIX_MEAR9	Read-only	0x00000000
0x0188–0x01E0	Reserved	–	–	–
0x01E4	Write Protection Mode Register	MATRIX_WPMR	Read/Write	0x00000000
0x01E8	Write Protection Status Register	MATRIX_WPSR	Read-only	0x00000000
0x01EC–0x01FC	Reserved	–	–	–
0x0200	Security Slave 0 Register	MATRIX_SSR0	Read/Write	0x00000000
0x0204	Security Slave 1 Register	MATRIX_SSR1	Read/Write	0x00000000

Table 15-10. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x0208	Security Slave 2 Register	MATRIX_SSR2	Read/Write	0x00000000
0x020C	Security Slave 3 Register	MATRIX_SSR3	Read/Write	0x00000000
0x0210	Security Slave 4 Register	MATRIX_SSR4	Read/Write	0x00000000
0x0214	Security Slave 5 Register	MATRIX_SSR5	Read/Write	0x00000000
0x0218	Security Slave 6 Register	MATRIX_SSR6	Read/Write	0x00000000
0x021C	Security Slave 7 Register	MATRIX_SSR7	Read/Write	0x00000000
0x0220	Security Slave 8 Register	MATRIX_SSR8	Read/Write	0x00000000
0x0224	Security Slave 9 Register	MATRIX_SSR9	Read/Write	0x00000000
0x0228	Security Slave 10 Register	MATRIX_SSR10	Read/Write	0x00000000
0x022C	Security Slave 11 Register	MATRIX_SSR11	Read/Write	0x00000000
0x0230	Security Slave 12 Register	MATRIX_SSR12	Read/Write	0x00000000
0x0234–0x023C	Reserved	–	–	–
0x0240	Security Areas Split Slave 0 Register	MATRIX_SASSR0	Read/Write	0x00000000
0x0244	Security Areas Split Slave 1 Register	MATRIX_SASSR1	Read/Write	0x00000000
0x0248	Security Areas Split Slave 2 Register	MATRIX_SASSR2	Read/Write	0x00000000
0x024C	Security Areas Split Slave 3 Register	MATRIX_SASSR3	Read/Write	0x00000000
0x0250	Security Areas Split Slave 4 Register	MATRIX_SASSR4	Read/Write	0x00000000
0x0254	Security Areas Split Slave 5 Register	MATRIX_SASSR5	Read/Write	0x00000000
0x0258	Security Areas Split Slave 6 Register	MATRIX_SASSR6	Read/Write	0x00000000
0x025C	Security Areas Split Slave 7 Register	MATRIX_SASSR7	Read/Write	0x00000000
0x0260	Security Areas Split Slave 8 Register	MATRIX_SASSR8	Read/Write	0x00000000
0x0264	Security Areas Split Slave 9 Register	MATRIX_SASSR9	Read/Write	0x00000000
0x0268	Security Areas Split Slave 10 Register	MATRIX_SASSR10	Read/Write	0x00000000
0x026C	Security Areas Split Slave 11 Register	MATRIX_SASSR11	Read/Write	0x00000000
0x0270	Security Areas Split Slave 12 Register	MATRIX_SASSR12	Read/Write	0x00000000
0x0274–0x0280	Reserved	–	–	–
0x0284	Security Region Top Slave 1 Register	MATRIX_SRTSR1	Read/Write	0x00000000
0x0288	Security Region Top Slave 2 Register	MATRIX_SRTSR2	Read/Write	0x00000000
0x028C	Security Region Top Slave 3 Register	MATRIX_SRTSR3	Read/Write	0x00000000
0x0290	Security Region Top Slave 4 Register	MATRIX_SRTSR4	Read/Write	0x00000000
0x0294	Security Region Top Slave 5 Register	MATRIX_SRTSR5	Read/Write	0x00000000
0x0298	Security Region Top Slave 6 Register	MATRIX_SRTSR6	Read/Write	0x00000000
0x029C	Security Region Top Slave 7 Register	MATRIX_SRTSR7	Read/Write	0x00000000
0x02A0	Security Region Top Slave 8 Register	MATRIX_SRTSR8	Read/Write	0x00000000
0x02A4	Security Region Top Slave 9 Register	MATRIX_SRTSR9	Read/Write	0x00000000
0x02A8	Security Region Top Slave 10 Register	MATRIX_SRTSR10	Read/Write	0x00000000
0x02AC	Security Region Top Slave 11 Register	MATRIX_SRTSR11	Read/Write	0x00000000

Table 15-10. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x02B0	Security Region Top Slave 12 Register	MATRIX_SRTSR12	Read/Write	0x00000000
0x02B4–0x02BC	Reserved	–	–	–
0x02C0	Security Peripheral Select 1 Register	MATRIX_SPSELR1	Read/Write	0x00000000(1)
0x02C4	Security Peripheral Select 2 Register	MATRIX_SPSELR2	Read/Write	0x00000000(2)
0x02C8	Security Peripheral Select 3 Register	MATRIX_SPSELR3	Read/Write	0x00000000(3)

Notes:
1. This value is 0x000D2504 for H32MX and 0xFFF2DAFB for H64MX.
2. This value is 0x011C0000 for H32MX and 0xFFE7FFFF for H64MX.
3. This value is 0xFFFFFFFA for H32MX and 0xFFFFFFE7 for H64MX.

15.13.1 Bus Matrix Master Configuration Registers

Name: MATRIX_MCFGx [x=0..9]

Address: 0xF001C000 (0), 0xFC054000 (1)

Access: Read/Write

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
–	–	–	–	–	ULBT		

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

- **ULBT: 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.

15.13.2 Bus Matrix Slave Configuration Registers

Name: MATRIX_SCFGx [x=0..12]

Address: 0xF001C040 (0), 0xFC054040 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	FIXED_DEFMSTR				DEFMSTR_TYPE	

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	SLOT_CYCLE

7	6	5	4	3	2	1	0
SLOT_CYCLE							

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

- **SLOT_CYCLE: 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.

Refer to Section 15.10.1.2 “Slot Cycle Limit Arbitration” for more details.

- **DEFMSTR_TYPE: 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.

- **FIXED_DEFMSTR: Fixed Default Master**

This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE value = 2. Specifying the number of a master which is not connected to the selected slave is equivalent to clearing DEFMSTR_TYPE.

15.13.3 Bus Matrix Priority Registers A For Slaves

Name: MATRIX_PRASx [x=0..12]

Address: 0xF001C080 (0)[0], 0xF001C088 (0)[1], 0xF001C090 (0)[2], 0xF001C098 (0)[3], 0xF001C0A0 (0)[4], 0xF001C0A8 (0)[5], 0xF001C0B0 (0)[6], 0xF001C0B8 (0)[7], 0xF001C0C0 (0)[8], 0xF001C0C8 (0)[9], 0xF001C0D0 (0)[10], 0xF001C0D8 (0)[11], 0xF001C0E0 (0)[12], 0xFC054080 (1)[0], 0xFC054088 (1)[1], 0xFC054090 (1)[2], 0xFC054098 (1)[3], 0xFC0540A0 (1)[4], 0xFC0540A8 (1)[5], 0xFC0540B0 (1)[6], 0xFC0540B8 (1)[7], 0xFC0540C0 (1)[8], 0xFC0540C8 (1)[9], 0xFC0540D0 (1)[10], 0xFC0540D8 (1)[11], 0xFC0540E0 (1)[12]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	M7PR		–	–	M6PR	

23	22	21	20	19	18	17	16
–	–	M5PR		–	–	M4PR	

15	14	13	12	11	10	9	8
–	–	M3PR		–	–	M2PR	

7	6	5	4	3	2	1	0
–	–	M1PR		–	–	M0PR	

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

- **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).

Refer to Section 15.10.2 “Arbitration Priority Scheme” for more details.

15.13.4 Bus Matrix Priority Registers B For Slaves

Name: MATRIX_PRBSx [x=0..12]

Address: 0xF001C084 (0)[0], 0xF001C08C (0)[1], 0xF001C094 (0)[2], 0xF001C09C (0)[3], 0xF001C0A4 (0)[4], 0xF001C0AC (0)[5], 0xF001C0B4 (0)[6], 0xF001C0BC (0)[7], 0xF001C0C4 (0)[8], 0xF001C0CC (0)[9], 0xF001C0D4 (0)[10], 0xF001C0DC (0)[11], 0xF001C0E4 (0)[12], 0xFC054084 (1)[0], 0xFC05408C (1)[1], 0xFC054094 (1)[2], 0xFC05409C (1)[3], 0xFC0540A4 (1)[4], 0xFC0540AC (1)[5], 0xFC0540B4 (1)[6], 0xFC0540BC (1)[7], 0xFC0540C4 (1)[8], 0xFC0540CC (1)[9], 0xFC0540D4 (1)[10], 0xFC0540DC (1)[11], 0xFC0540E4 (1)[12]

Access: Read/Write

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
–	–	M9PR		–	–	M8PR	

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

- **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).

Refer to Section 15.10.2 “Arbitration Priority Scheme” for more details.

15.13.5 Master Error Interrupt Enable Register

Name: MATRIX_MEIER

Address: 0xF001C150 (0), 0xFC054150 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	MERR9	MERR8
7	6	5	4	3	2	1	0
MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0

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

- **MERRx: Master x Access Error**

0: No effect

1: Enables Master x Access Error interrupt source

15.13.6 Master Error Interrupt Disable Register

Name: MATRIX_MEIDR

Address: 0xF001C154 (0), 0xFC054154 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	MERR9	MERR8
7	6	5	4	3	2	1	0
MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0

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

- **MERRx: Master x Access Error**

0: No effect

1: Disables Master x Access Error interrupt source

15.13.7 Master Error Interrupt Mask Register

Name: MATRIX_MEIMR

Address: 0xF001C158 (0), 0xFC054158 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	MERR9	MERR8

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

- **MERRx: Master x Access Error**

0: Master x Access Error does not trigger any interrupt.

1: Master x Access Error triggers the Bus Matrix interrupt line.

15.13.8 Master Error Status Register

Name: MATRIX_MESR

Address: 0xF001C15C (0), 0xFC05415C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	MERR9	MERR8
7	6	5	4	3	2	1	0
MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0

- **MERRx: Master x Access Error**

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.

15.13.9 Master Error Address Registers

Name: MATRIX_MEARx [x=0..9]

Address: 0xF001C160 (0), 0xFC054160 (1)

Access: Read-only

31	30	29	28	27	26	25	24
ERRADD							

23	22	21	20	19	18	17	16
ERRADD							

15	14	13	12	11	10	9	8
ERRADD							

7	6	5	4	3	2	1	0
ERRADD							

- **ERRADD: Master Error Address**

Master Last Access Error Address

15.13.10Write Protection Mode Register

Name: MATRIX_WPMR

Address: 0xF001C1E4 (0), 0xFC0541E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

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).

Refer to Section 15.11 “Register Write Protection” for list of registers that can be write-protected.

- **WPKEY: Write Protection Key** (Write-only)

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

15.13.11 Write Protection Status Register

Name: MATRIX_WPSR

Address: 0xF001C1E8 (0), 0xFC0541E8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of MATRIX_WPSR.

1: A write protection violation has occurred since the last write of MATRIX_WPMR.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

15.13.12Security Slave Registers

Name: MATRIX_SSRx [x=0..12]

Address: 0xF001C200 (0), 0xFC054200 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WRNSECH7	WRNSECH6	WRNSECH5	WRNSECH4	WRNSECH3	WRNSECH2	WRNSECH1	WRNSECH0

15	14	13	12	11	10	9	8
RDNSECH7	RDNSECH6	RDNSECH5	RDNSECH4	RDNSECH3	RDNSECH2	RDNSECH1	RDNSECH0

7	6	5	4	3	2	1	0
LANSECH7	LANSECH6	LANSECH5	LANSECH4	LANSECH3	LANSECH2	LANSECH1	LANSECH0

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

- **LANSECHx: Low Area Non-secured in HSELx Security Region**

0: The security of the HSELx AHB slave area laying below the corresponding MATRIX_SASSR / SASPLITx boundary is configured according to RDNSECHx and WRNSECHx. The whole remaining HSELx upper address space is configured as Non-secured access.

1: The HSELx AHB slave address area laying below the corresponding MATRIX_SASSR / SASPLITx boundary is configured as Non-secured access, and the whole remaining upper address space according to RDNSECHx and WRNSECHx.

- **RDNSECHx: Read Non-secured for HSELx Security Region**

0: The HSELx AHB slave security region is split into one Read Secured and one Read Non-secured area, according to LANSECHx and MATRIX_SASSR / SASPLITx. That is, the so defined Securable high or low area is Secured for Read access.

1: The HSELx AHB slave security region is Non-secured for Read access.

- **WRNSECHx: Write Non-secured for HSELx Security Region**

0: The HSELx AHB slave security region is split into one Write Secured and one Write Non-secured area, according to LANSECHx and MATRIX_SASSR / SASPLITx. That is, the so defined Securable high or low area is Secured for Write access.

1: The HSELx AHB slave security region is Non-secured for Write access.

Securable Area access rights:

WRNSECHx / RDNSECHx	Non-secure Access	Secure Access
00	Denied	Write - Read
01	Read	Write - Read
10	Write	Write - Read
11	Write - Read	Write - Read

15.13.13Security Areas Split Slave Registers

Name: MATRIX_SASSRx [x=0..12]

Address: 0xF001C240 (0), 0xFC054240 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
SASPLIT7				SASPLIT6			

23	22	21	20	19	18	17	16
SASPLIT5				SASPLIT4			

15	14	13	12	11	10	9	8
SASPLIT3				SASPLIT2			

7	6	5	4	3	2	1	0
SASPLIT1				SASPLIT0			

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

- **SASPLITx: Security Areas Split for HSELx Security Region**

This field defines the boundary address offset where the HSELx AHB slave security region splits into two Security Areas whose access is controlled according to the corresponding MATRIX_SSR. So it also defines the Security Low Area size inside the HSELx region.

If this Low Area size is set at or above the HSELx Region Size, then there is no more Security High Area and the MATRIX_SSR settings for the Low area apply to the whole HSELx Security Region.

SASPLITx	Split Offset	Security 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

15.13.14Security Region Top Slave Registers

Name: MATRIX_SRTSRx [x=0..12]

Address: 0xF001C284 (0), 0xFC054284 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
SRTOP7				SRTOP6			

23	22	21	20	19	18	17	16
SRTOP5				SRTOP4			

15	14	13	12	11	10	9	8
SRTOP3				SRTOP2			

7	6	5	4	3	2	1	0
SRTOP1				SRTOP0			

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

- **SRTOPx: HSELx Security Region Top**

This field defines the size of the HSELx security region address space. Invalid sizes for the slave region must never be programmed. Valid sizes and number of security regions are product, slave and slave configuration dependent.

Note: The slaves featuring multiple scalable contiguous security regions have a single SRTOP0 field for all the security regions.

If this HSELx security region size is set at or below the HSELx low area size, then there is no more security high area and the MATRIX_SSR settings for the low area apply to the whole HSELx security region.

SRTOPx	Top Offset	Security 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
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes
1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

15.13.15Security Peripheral Select x Registers

Name: MATRIX_SPSELRx [x=1..3]

Address: 0xF001C2C0 (0), 0xFC0542C0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
NSECP31	NSECP30	NSECP29	NSECP28	NSECP27	NSECP26	NSECP25	NSECP24
23	22	21	20	19	18	17	16
NSECP23	NSECP22	NSECP21	NSECP20	NSECP19	NSECP18	NSECP17	NSECP16
15	14	13	12	11	10	9	8
NSECP15	NSECP14	NSECP13	NSECP12	NSECP11	NSECP10	NSECP9	NSECP8
7	6	5	4	3	2	1	0
NSECP7	NSECP6	NSECP5	NSECP4	NSECP3	NSECP2	NSECP1	NSECP0

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

Each MATRIX_SPSELR can configure the access security type for up to 32 peripherals:

- MATRIX_SPSELR1 configures the access security type for peripheral identifiers 0–31 (bits NSECP0–NSECP31).
- MATRIX_SPSELR2 configures the access security type for peripheral identifiers 32–63 (bits NSECP0–NSECP31).
- MATRIX_SPSELR3 configures the access security type for peripheral identifiers 64–95 (bits NSECP0–NSECP31).

Note: The actual number of peripherals implemented is device-specific; refer to section “Peripheral Identifiers” for more details.

- **NSECPy: Non-secured Peripheral**

0: The selected peripheral address space is configured as “Secured” access (value of ‘0’ has no effect if the peripheral security type is “Non-secured”).

1: The selected peripheral address space is configured as “Non-secured” access (value of ‘1’ has no effect if the peripheral security type is “Always Secured”).

16. Special Function Registers (SFR)

16.1 Description

Special Function Registers (SFR) manage specific aspects of the integrated memory, bridge implementations, processor and other functionality not controlled elsewhere.

16.2 Embedded Characteristics

- 32-bit Special Function Registers control specific behavior of the product

16.3 Special Function Registers (SFR) User Interface

Table 16-1. Register Mapping

Offset[1]	Register	Name	Access	Reset
0x04	DDR Configuration Register	SFR_DDRCFG	Read/Write	0x01
0x08–0x0C	Reserved	–	–	–
0x10	OHCI Interrupt Configuration Register	SFR_OHCIICR	Read/Write	0x0
0x14	OHCI Interrupt Status Register	SFR_OHCIISR	Read-only	–
0x18	Reserved	–	–	–
0x28	Security Configuration Register	SFR_SECURE	Read/Write	0x0
0x2C–0x3C	Reserved	–	–	–
0x40	Reserved	–	–	–
0x44	Analog Configuration Register	SFR_ANACFG	Read/Write	0x0
0x48	Reserved	–	–	–
0x4C	Serial Number 0 Register	SFR_SN0	Read-only	–
0x50	Serial Number 1 Register	SFR_SN1	Read-only	–
0x54	AIC Interrupt Redirection Register	SFR_AICREDIR	Read/Write	0x0
0x58–0x3FFC	Reserved	–	–	–

Note: 1. If an offset is not listed in the table it must be considered as reserved.

16.3.1 DDR Configuration Register

Name: SFR_DDRCFG

Address: 0xF8028004

Access: Read/Write

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

- **FDQIEN: Force DDR_DQ Input Buffer Always On**

0: DDR_DQ input buffer controlled by DDR controller.

1: DDR_DQ input buffer always on.

- **FDQSIEN: Force DDR_DQS Input Buffer Always On**

0: DDR_DQS input buffer controlled by DDR controller.

1: DDR_DQS input buffer always on.

Note: FDQIEN and FDQSIEN = 1 are used to force the selection of the analog comparator inside the IO. If those bits are cleared the DDR controller automatically manages the selection of the analog comparator. Forcing the bits to 0 reduces power consumption.

16.3.2 OHCI Interrupt Configuration Register

Name: SFR_OHCIICR

Address: 0xF8028010

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
UDPPUDIS	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	APPSTART	ARIE	–	RES2	RES1	RES0

- **RESx: USB PORTx RESET**

0: Resets USB PORT.

1: Usable USB PORT.

- **ARIE: OHCI Asynchronous Resume Interrupt Enable**

0: Interrupt disabled.

1: Interrupt enabled.

- **APPSTART: Reserved**

0: Must write 0.

- **UDPPUDIS: USB DEVICE PULL-UP DISABLE**

0: USB device pull-up connection is enabled.

1: USB device pull-up connection is disabled.

16.3.3 OHCI Interrupt Status Register

Name: SFR_OHCIISR

Address: 0xF8028014

Access: Read/Write

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
–	–	–	–	–	RIS2	RIS1	RIS0

- **RISx: OHCI Resume Interrupt Status Port x**

0: OHCI port resume not detected.

1: OHCI port resume detected.

16.3.4 Security Configuration Register

Name: SFR_SECURE

Address: 0xF8028028

Access: Read/Write

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

- **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.

0: ROM is enabled.

1: ROM is disabled.

- **FUSE: Disable Access to Fuse Controller**

This bit is writable once only. When the Fuse Controller is secured, only a reset signal can clear this bit.

0: Fuse Controller is enabled.

1: Fuse Controller is disabled.

16.3.5 Analog Configuration Register

Name: SFR_ANACFG

Address: 0xF8028044

Access: Read/Write

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
–	–	–	–	–	–	–	SM_DDR_EN

This register controls special analog cell configuration signals.

- **SM_DDR_EN: DDR Supply Monitor Enable**

0: DDR Supply Monitor is disabled.

1: DDR Supply Monitor is enabled.

16.3.6 Serial Number 0 Register

Name: SFR_SN0

Address: 0xF802804C

Access: Read-only

31	30	29	28	27	26	25	24
SN0							

23	22	21	20	19	18	17	16
SN0							

15	14	13	12	11	10	9	8
SN0							

7	6	5	4	3	2	1	0
SN0							

This register is used to read the first 32 bits of the 64-bit Serial Number (unique ID).

- **SN0: Serial Number 0**

16.3.7 Serial Number 1 Register

Name: SFR_SN1

Address: 0xF8028050

Access: Read-only

31	30	29	28	27	26	25	24
SN1							

23	22	21	20	19	18	17	16
SN1							

15	14	13	12	11	10	9	8
SN1							

7	6	5	4	3	2	1	0
SN1							

This register is used to read the last 32 bits of the 64-bit Serial Number (unique ID).

- **SN1: Serial Number 1**

16.3.8 AIC Interrupt Redirection Register

Name: SFR_AICREDIR

Address: 0xF8028054

Access: Read/Write

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

- **NSAIC: Interrupt Redirection to Non-Secure AIC**

0: Interrupts are managed by the AIC corresponding to the Secure State of the peripheral (secure AIC or non-secure AIC).

1: All interrupts are managed by the non-secure AIC.

- **AICREDIRKEY: Unlock Key**

Value is a XOR between 0xb6d81c4d and SN1[31:0] but only field [31:1] of the result must be written in this field. In case of set in Secure mode by fuse configuration, this register is read_only 0 (it is not possible to redirect secure interrupts on non-secure AIC for products set in secure mode for security reasons).

Note: After three tries, entering a wrong key results in locking the NSAIC bit. A reset is needed.

17. Advanced Interrupt Controller (AIC)

17.1 Description

The Advanced Interrupt Controller (AIC) is an 8-level priority, individually maskable, vectored interrupt controller providing handling of up to one hundred and twenty-eight interrupt sources. It is designed to substantially reduce the software and real-time overhead in handling internal and external interrupts.

The AIC drives the nFIQ (fast interrupt request) and the nIRQ (standard interrupt request) inputs of an ARM processor. Inputs of the AIC are either internal peripheral interrupts or external interrupts coming from the product's pins.

The 8-level Priority Controller allows the user to define the priority for each interrupt source, thus permitting higher priority interrupts to be serviced even if a lower priority interrupt is being processed.

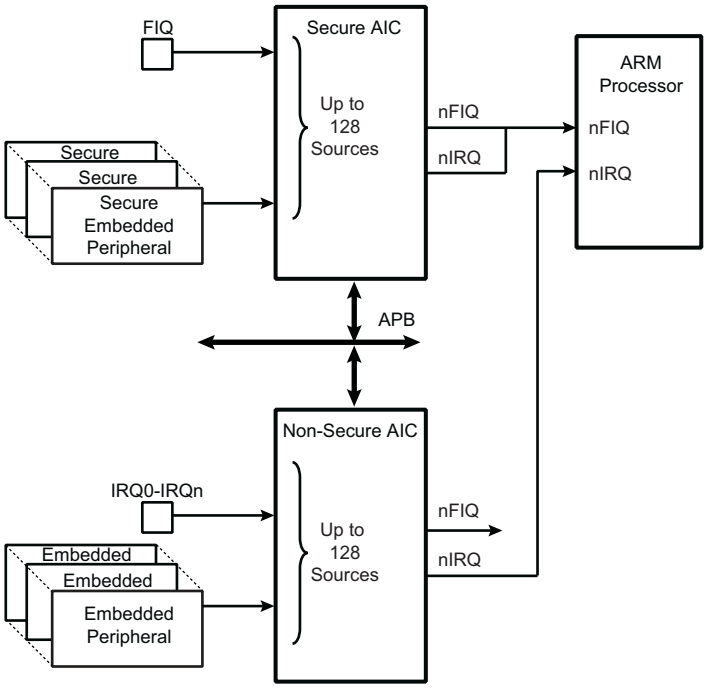
Internal interrupt sources can be programmed to be level-sensitive or edge-triggered. External interrupt sources can be programmed to be positive-edge or negative-edge triggered or high-level or low-level sensitive.

17.2 Embedded Characteristics

- Controls the Interrupt Lines (nIRQ and nFIQ) of an ARM® Processor
- 128 Individually Maskable and Vectored Interrupt Sources
  - Source 0 is Reserved for the Fast Interrupt Input (FIQ)
  - Source 1 is Reserved for System Peripheral Interrupts
  - Source 2 to Source 127 Control up to 126 Embedded Peripheral Interrupts or External Interrupts
  - Programmable Edge-triggered or Level-sensitive Internal Sources
  - Programmable Positive/Negative Edge-triggered or High/Low Level-sensitive External Sources
- 8-level Priority Controller
  - Drives the Normal Interrupt of the Processor
  - Handles Priority of the Interrupt Sources 1 to 127
  - Higher Priority Interrupts Can Be Served During Service of Lower Priority Interrupt
- Vectoring
  - Optimizes Interrupt Service Routine Branch and Execution
  - One 32-bit Vector Register for all Interrupt Sources
  - Interrupt Vector Register Reads the Corresponding Current Interrupt Vector
- Protect Mode
  - Easy Debugging by Preventing Automatic Operations when Protect Models are Enabled
- General Interrupt Mask
  - Provides Processor Synchronization on Events Without Triggering an Interrupt
- Register Write Protection
- AIC0 is Non-Secure AIC, AIC1 is Secure AIC
- AIC0 manages nIRQ line, AIC1 manages nFIQ line

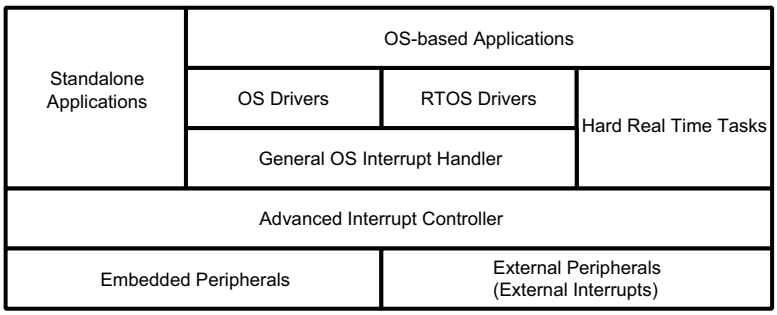
17.3 Block Diagram

Figure 17-1. Block Diagram



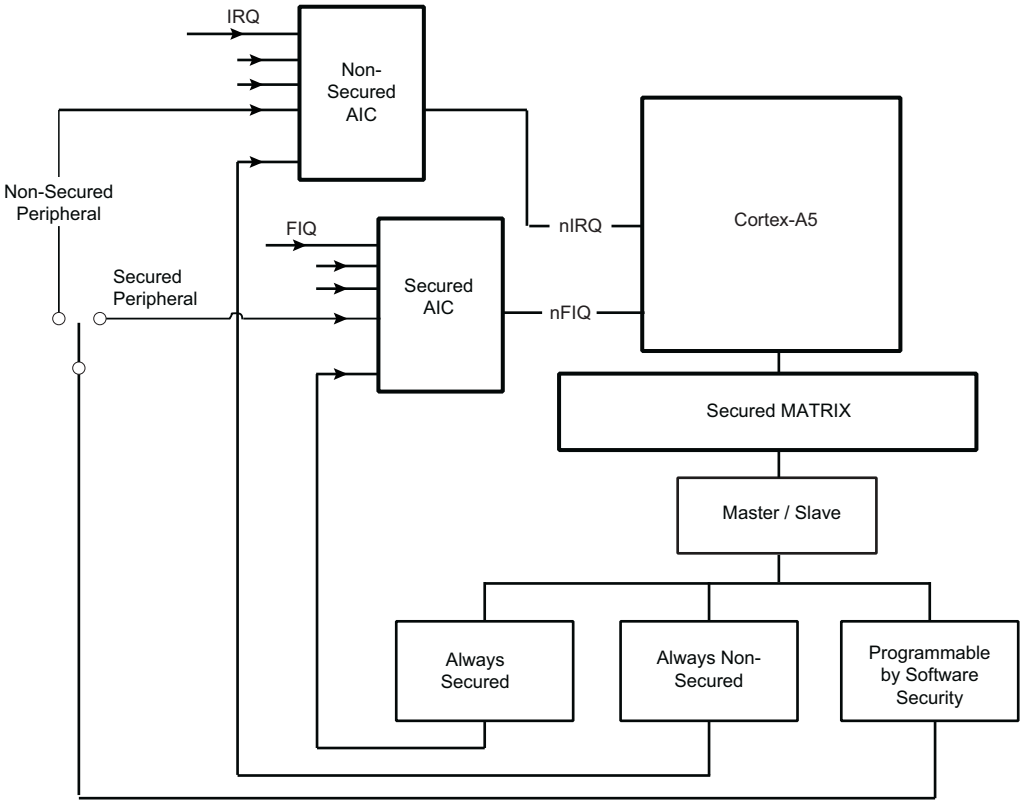
17.4 Application Block Diagram

Figure 17-2. Description of the Application Block



17.5 AIC Detailed Block Diagram

Figure 17-3. AIC Detailed Block Diagram



17.6 I/O Line Description

Table 17-1. I/O Line Description

Pin Name	Pin Description	Type
FIQ	Fast Interrupt	Input
IRQ0–IRQn	Interrupt 0–Interrupt n	Input

17.7 Product Dependencies

17.7.1 I/O Lines

The interrupt signals FIQ and IRQ0 to IRQn are normally multiplexed through the PIO controllers. Depending on the features of the PIO controller used in the product, the pins must be programmed in accordance with their assigned interrupt functions. This is not applicable when the PIO controller used in the product is transparent on the input path.

Table 17-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
AIC	FIQ	PD9	A
AIC	IRQ	PE25	C

17.7.2 Power Management

The Advanced Interrupt Controller is continuously clocked. The Power Management Controller has no effect on the Advanced Interrupt Controller behavior.

The assertion of the Advanced Interrupt Controller outputs, either nIRQ or nFIQ, wakes up the ARM processor while it is in Idle mode. The General Interrupt Mask feature enables the AIC to wake up the processor without asserting the interrupt line of the processor, thus providing synchronization of the processor on an event.

17.7.3 Interrupt Sources

Interrupt Source 0 is always located at FIQ. If the product does not feature any FIQ pin, Interrupt Source 0 cannot be used.

Interrupt Source 1 is always located at System Interrupt. This is the result of the OR-wiring of the system peripheral interrupt lines. When a system interrupt occurs, the service routine must first distinguish the cause of the interrupt. This is performed by reading successively the status registers of the above-mentioned system peripherals.

Interrupt sources 2 to 127 can either be connected to the interrupt outputs of an embedded user peripheral, or to external interrupt lines. The external interrupt lines can be connected either directly or through the PIO Controller.

PIO controllers are considered as user peripherals in the scope of interrupt handling. Accordingly, the PIO controller interrupt lines are connected to interrupt sources 2 to 127.

The peripheral identification defined at the product level corresponds to the interrupt source number (as well as the bit number controlling the clock of the peripheral). Consequently, to simplify the description of the functional operations and the user interface, the interrupt sources are named FIQ, SYS, and PID2 to PID127.

AIC0 manages all Non-Secure Interrupts including IRQn; AIC1 manages all Secure Interrupts including FIQ.

Each AIC has its own User Interface. The user should pay attention to use the relevant user interface for each source.

17.8 Functional Description

17.8.1 Interrupt Source Control

17.8.1.1 Interrupt Source Mode

The Advanced Interrupt Controller independently programs each interrupt source. The SRCTYPE field of the Source Mode Register (AIC_SMR) selects the interrupt condition of the interrupt source selected by the INTSEL field of the Source Select Register (AIC_SSR).

Note: Configuration registers such as AIC_SMR and AIC_SSR return the values corresponding to the interrupt source selected by INTSEL.

The internal interrupt sources wired on the interrupt outputs of the embedded peripherals can be programmed either in Level-Sensitive mode or in Edge-Triggered mode. The active level of the internal interrupts is not important for the user.

The external interrupt sources can be programmed either in High Level-Sensitive or Low Level-Sensitive modes, or in Positive Edge-Triggered or Negative Edge-Triggered modes.

17.8.1.2 Interrupt Source Enabling

Each interrupt source, including the FIQ in source 0, can be enabled or disabled by using the command registers Interrupt Enable Command Register (AIC_IECR) and Interrupt Disable Command Register (AIC_IDCR). The interrupt mask of the selected interrupt source can be read in the Interrupt Mask Register (AIC_IMR). A disabled interrupt does not affect servicing of other interrupts.

17.8.1.3 Interrupt Clearing and Setting

All interrupt sources programmed to be edge-triggered (including the FIQ in source 0) can be individually set or cleared by writing respectively the Interrupt Set Command Register (AIC_ISCR) and Interrupt Clear Command Register (AIC_ICCR). Clearing or setting interrupt sources programmed in Level-Sensitive mode has no effect.

The clear operation is perfunctory, as the software must perform an action to reset the “memorization” circuitry activated when the source is programmed in Edge-Triggered mode. However, the set operation is available for auto-test or software debug purposes. It can also be used to execute an AIC-implementation of a software interrupt.

The AIC features an automatic clear of the current interrupt when AIC_IVR (Interrupt Vector Register) is read. Only the interrupt source being detected by the AIC as the current interrupt is affected by this operation. (Refer to Section 17.8.3.1 “Priority Controller”.) The automatic clear reduces the operations required by the interrupt service routine entry code to read AIC_IVR.

The automatic clear of interrupt source 0 is performed when AIC_FVR is read.

17.8.1.4 Interrupt Status

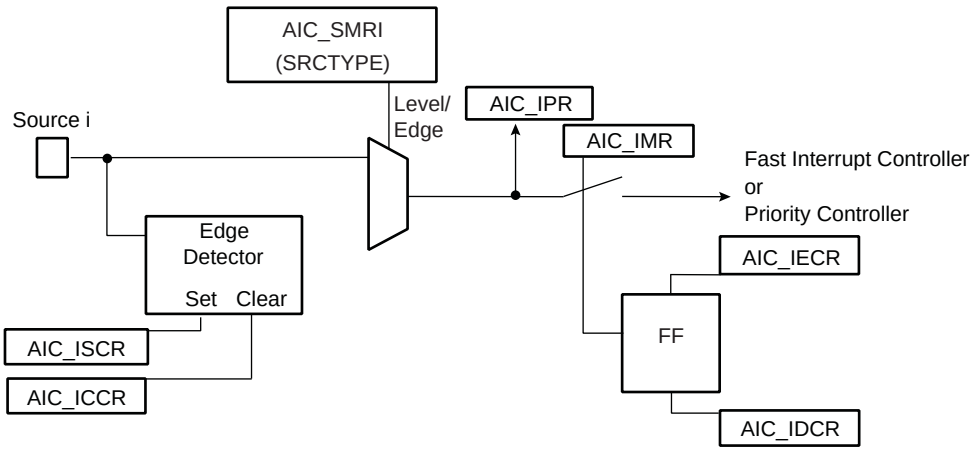
Interrupt Pending Registers (AIC_IPR) represent the state of the interrupt lines, whether they are masked or not. AIC_IMR can be used to define the mask of the interrupt lines.

The Interrupt Status Register (AIC_ISR) reads the number of the current interrupt (refer to Section 17.8.3.1 “Priority Controller”) and the Core Interrupt Status Register (AIC_CISR) gives an image of the nIRQ and nFIQ signals driven on the processor.

Each status referred to above can be used to optimize the interrupt handling of the systems.

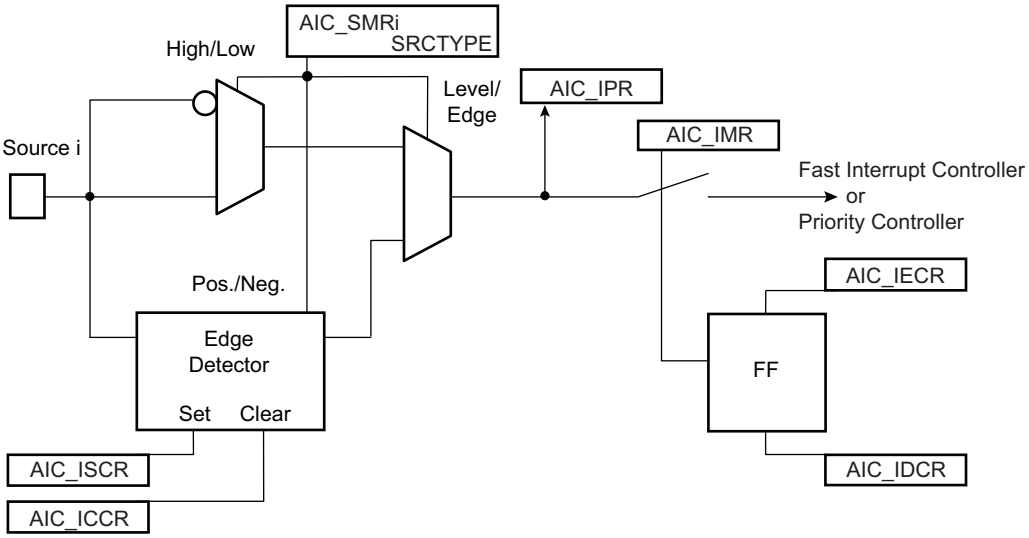
17.8.1.5 Internal Interrupt Source Input Stage

Figure 17-4. Internal Interrupt Source Input Stage



17.8.1.6 External Interrupt Source Input Stage

Figure 17-5. External Interrupt Source Input Stage



17.8.2 Interrupt Latencies

Global interrupt latencies depend on several parameters, including:

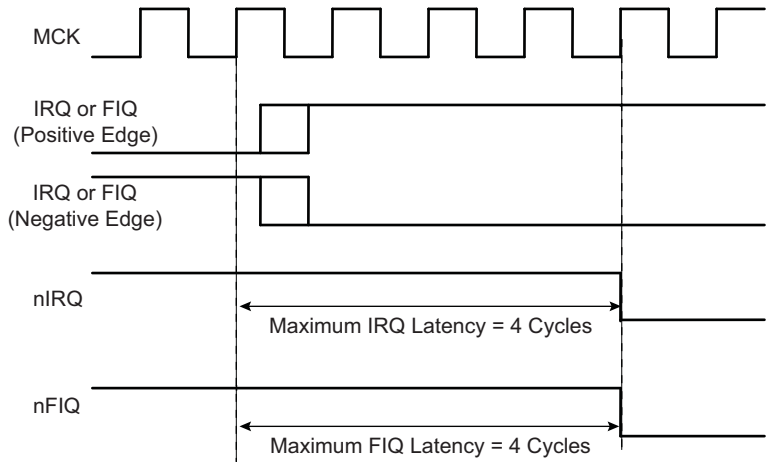
- The time the software masks the interrupts
- Occurrence, either at the processor level or at the AIC level
- The execution time of the instruction in progress when the interrupt occurs
- The treatment of higher priority interrupts and the resynchronization of the hardware signals

This section addresses hardware resynchronizations only. It gives details about the latency times between the events on an external interrupt leading to a valid interrupt (edge or level) or the assertion of an internal interrupt source and the assertion of the nIRQ or nFIQ line on the processor. The resynchronization time depends on the programming of the interrupt source and on its type (internal or external). For the standard interrupt, resynchronization times are given assuming there is no higher priority in progress.

The PIO Controller multiplexing has no effect on the interrupt latencies of the external interrupt sources.

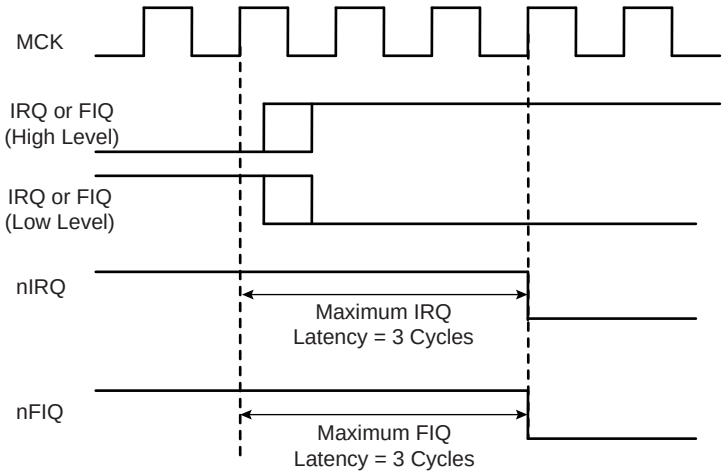
17.8.2.1 External Interrupt Edge Triggered Source

Figure 17-6. External Interrupt Edge Triggered Source



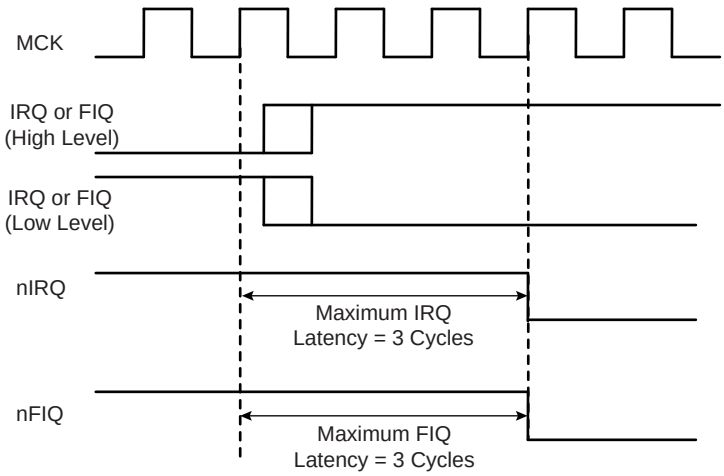
17.8.2.2 External Interrupt Level Sensitive Source

Figure 17-7. External Interrupt Level Sensitive Source



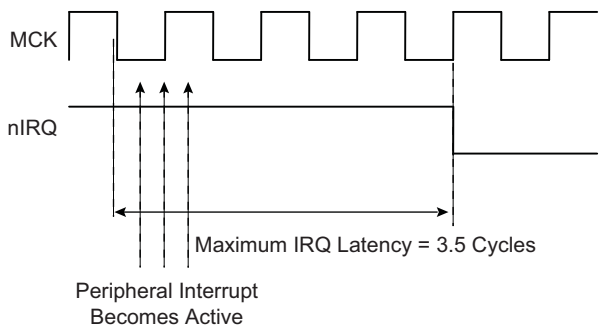
17.8.2.3 Internal Interrupt Edge Triggered Source

Figure 17-8. Internal Interrupt Edge Triggered Source



17.8.2.4 Internal Interrupt Level Sensitive Source

Figure 17-9. Internal Interrupt Level Sensitive Source



17.8.3 Normal Interrupt

17.8.3.1 Priority Controller

An 8-level priority controller drives the nIRQ line of the processor, depending on the interrupt conditions occurring on the interrupt sources 1 to 127.

Each interrupt source has a programmable priority level of 7 to 0, which is user-definable by writing the PRIOR field of AIC_SMR (Source Mode Register). Level 7 is the highest priority and level 0 the lowest.

As soon as an interrupt condition occurs, as defined by the SRCTYPE field in AIC_SMR (Source Mode Register), the nIRQ line is asserted. As a new interrupt condition might have happened on other interrupt sources since the nIRQ has been asserted, the priority controller determines the current interrupt at the time AIC_IVR (Interrupt Vector Register) is read. The read of AIC_IVR is the entry point of the interrupt handling which allows the AIC to consider that the interrupt has been taken into account by the software.

The current priority level is defined as the priority level of the current interrupt.

If several interrupt sources of equal priority are pending and enabled when AIC_IVR is read, the interrupt with the lowest interrupt source number is serviced first.

The nIRQ line can be asserted only if an interrupt condition occurs on an interrupt source with a higher priority. If an interrupt condition happens (or is pending) during the interrupt treatment in progress, it is delayed until the software indicates to the AIC the end of the current service by writing AIC_EOICR (End of Interrupt Command Register). The write of AIC_EOICR is the exit point of the interrupt handling.

17.8.3.2 Interrupt Nesting

The priority controller utilizes interrupt nesting in order for the high priority interrupt to be handled during the service of lower priority interrupts. This requires the interrupt service routines of the lower interrupts to re-enable the interrupt at the processor level.

When an interrupt of a higher priority happens during an already occurring interrupt service routine, the nIRQ line is re-asserted. If the interrupt is enabled at the core level, the current execution is interrupted and the new interrupt service routine should read AIC_IVR. At this time, the current interrupt number and its priority level are pushed into an embedded hardware stack, so that they are saved and restored when the higher priority interrupt servicing is finished and AIC_EOICR is written.

The AIC is equipped with an 8-level wide hardware stack in order to support up to eight interrupt nestings to match the eight priority levels.

17.8.3.3 Interrupt Handlers

This section gives an overview of the fast interrupt handling sequence when using the AIC. It is assumed that the programmer understands the architecture of the ARM processor, and especially the Processor Interrupt modes and the associated status bits.

It is assumed that:

1. The Advanced Interrupt Controller has been programmed, AIC_SVR registers are loaded with corresponding interrupt service routine addresses and interrupts are enabled.
2. The instruction at the ARM interrupt exception vector address is required to work with the vectoring. Load the PC with the absolute address of the interrupt handler.

When nIRQ is asserted, if the bit “I” of CPSR is 0, the sequence is as follows:

1. The CPSR is stored in SPSR_irq, the current value of the Program Counter is loaded in the Interrupt link register (R14_irq) and the Program Counter (R15) is loaded with 0x18. In the following cycle during fetch at address 0x1C, the ARM core adjusts R14_irq, decrementing it by four.
2. The ARM core enters Interrupt mode, if it has not already done so.
3. When the instruction loaded at address 0x18 is executed, the program counter is loaded with the value read in AIC_IVR. Reading AIC_IVR has the following effects:

- Sets the current interrupt to be the pending and enabled interrupt with the highest priority. The current level is the priority level of the current interrupt.
- De-asserts the nIRQ line on the processor. Even if vectoring is not used, AIC_IVR must be read in order to de-assert nIRQ.
- Automatically clears the interrupt, if it has been programmed to be edge-triggered.
- Pushes the current level and the current interrupt number on to the stack.
- Returns the value written in AIC_SVR corresponding to the current interrupt.

4. The previous step has the effect of branching to the corresponding interrupt service routine. This should start by saving the link register (R14_irq) and SPSR_IRQ. The link register must be decremented by four when it is saved if it is to be restored directly into the program counter at the end of the interrupt. For example, the instruction `SUB PC, LR, #4` may be used.
5. Further interrupts can then be unmasked by clearing the “I” bit in CPSR, allowing re-assertion of the nIRQ to be taken into account by the core. This can happen if an interrupt with a higher priority than the current interrupt occurs.
6. The interrupt handler can then proceed as required, saving the registers that will be used and restoring them at the end. During this phase, an interrupt of higher priority than the current level will restart the sequence from step 1.

Note: If the interrupt is programmed to be level-sensitive, the source of the interrupt must be cleared during this phase.

7. The “I” bit in CPSR must be set in order to mask interrupts before exiting to ensure that the interrupt is completed in an orderly manner.
8. The End of Interrupt Command Register (AIC_EOICR) must be written in order to indicate to the AIC that the current interrupt is finished. This causes the current level to be popped from the stack, restoring the previous current level if one exists on the stack. If another interrupt is pending, with lower or equal priority than the old current level but with higher priority than the new current level, the nIRQ line is re-asserted, but the interrupt sequence does not immediately start because the “I” bit is set in the core. SPSR_irq is restored. Finally, the saved value of the link register is restored directly into the PC. This has the effect of returning from the interrupt to whatever was being executed before, and of loading the CPSR with the stored SPSR, masking or unmasking the interrupts depending on the state saved in SPSR_irq.

Note: The “I” bit in SPSR is significant. If it is set, it indicates that the ARM core was on the verge of masking an interrupt when the mask instruction was interrupted. Hence, when SPSR is restored, the mask instruction is completed (interrupt is masked).

17.8.4 Fast Interrupt

17.8.4.1 Fast Interrupt Source

Interrupt source 0 is the only source which can raise a fast interrupt request to the processor. Interrupt source 0 is generally connected to a FIQ pin of the product, either directly or through a PIO Controller.

17.8.4.2 Fast Interrupt Control

The fast interrupt logic of the AIC has no priority controller. The mode of interrupt source 0 is programmed with AIC_SMR and INTSEL = 0; the PRIOR field of this register is not used even if it reads what has been written. The SRCTYPE field of AIC_SMR enables programming the fast interrupt source to be positive-edge triggered or negative-edge triggered or high-level sensitive or low-level sensitive.

Writing 0x1 in AIC_IECR (Interrupt Enable Command Register) and AIC_IDCR (Interrupt Disable Command Register) respectively enables and disables the fast interrupt when INTSEL = 0. Bit 0 of AIC_IMR indicates whether the fast interrupt is enabled or disabled.

17.8.4.3 Fast Interrupt Handlers

This section gives an overview of the fast interrupt handling sequence when using the AIC. It is assumed that the programmer understands the architecture of the ARM processor, and especially the Processor Interrupt modes and associated status bits.

Assuming that:

1. The Advanced Interrupt Controller has been programmed, AIC_SVR is loaded with the fast interrupt service routine address, and interrupt source 0 is enabled.
2. The Instruction at address 0x1C (FIQ exception vector address) is required to vector the fast interrupt. Load the PC with the absolute address of the interrupt handler.
3. The user does not need nested fast interrupts.

When nFIQ is asserted, if bit “F” of CPSR is 0, the sequence is:

1. The CPSR is stored in SPSR_fiq, the current value of the program counter is loaded in the FIQ link register (R14_FIQ) and the program counter (R15) is loaded with 0x1C. In the following cycle, during fetch at address 0x20, the ARM core adjusts R14_fiq, decrementing it by four.
2. The ARM core enters FIQ mode.
3. The routine must read AIC1_CISR to know if the interrupt is the FIQ or a Secure Internal interrupt.

```
        ldr         r1, =REG_SAIC_CISR
        ldr         r1, [r1]
        cmp         r1, #AIC_CISR_NFIQ
        beq         get_fiqvec_addr
```

   If FIQ is active, it is processed in priority, even if another interrupt is active.

```
get_irqvec_addr
        ldr         r14, =REG_SAIC_IVR
        b           read_vec
get_fiqvec_addr
        ldr         r14, =REG_SAIC_FVR
 read_vec
        ldr         r0, [r14]
```

   Now r0 contains the correct vector address, IVR for a Secure Internal interrupt or FVR for FIQ.

   The system can branch to the routine pointed to by r0.

```
FIQ_Handler_Branch
        mov     r14, pc
        bx      r0
```

4. The previous step enables branching to the corresponding interrupt service routine. It is not necessary to save the link register R14_fiq and SPSR_fiq if nested fast interrupts are not needed.
5. The Interrupt Handler can then proceed as required. It is not necessary to save registers R8 to R13 because the FIQ mode has its own dedicated registers and registers R8 to R13 are banked. The other registers, R0 to R7, must be saved before being used, and restored at the end (before the next step).

Note: If the fast interrupt is programmed to be level-sensitive, the source of the interrupt must be cleared during this phase in order to de-assert interrupt source 0.

6. Finally, Link Register R14_fiq is restored into the PC after decrementing it by four (with instruction `SUB PC, LR, #4` for example). This has the effect of returning from the interrupt to whatever was being executed before, loading the CPSR with the SPSR and masking or unmasking the fast interrupt depending on the state saved in the SPSR.

Note: The “F” bit in SPSR is significant. If it is set, it indicates that the ARM core was just about to mask FIQ interrupts when the mask instruction was interrupted. Hence, when the SPSR is restored, the interrupted instruction is completed (FIQ is masked).

Another way to handle the fast interrupt is to map the interrupt service routine at the address of the ARM vector 0x1C. This method does not use vectoring, so that reading AIC_FVR must be performed at the very beginning of the handler operation. However, this method saves the execution of a branch instruction.

17.8.5 Protect Mode

The Protect mode is used to read the Interrupt Vector Register without performing the associated automatic operations. This is necessary when working with a debug system. When a debugger, working either with a Debug Monitor or the ARM processor's ICE, stops the applications and updates the opened windows, it might read the AIC User Interface and thus the IVR. This has adverse consequences:

- If an enabled interrupt with a higher priority than the current one is pending, it is stacked.
- If there is no enabled pending interrupt, the spurious vector is returned.

In either case, an End of Interrupt command is necessary to acknowledge and restore the context of the AIC. This operation is generally not performed by the debug system, as the debug system would become strongly intrusive and cause the application to enter an undesired state.

This is avoided by using the Protect mode. Writing PROT in AIC_DCR (Debug Control Register) at 0x1 enables the Protect mode.

When the Protect mode is enabled, the AIC performs interrupt stacking only when a write access is performed on AIC_IVR. Therefore, the Interrupt Service Routines must write (arbitrary data) to AIC_IVR just after reading it. The new context of the AIC, including the value of AIC_ISR, is updated with the current interrupt only when AIC_IVR is written.

An AIC_IVR read on its own (e.g., by a debugger) modifies neither the AIC context nor AIC_ISR. Extra AIC_IVR reads perform the same operations. However, it is recommended to not stop the processor between the read and the write of AIC_IVR of the interrupt service routine to make sure the debugger does not modify the AIC context.

To summarize, in normal operating mode, the read of AIC_IVR performs the following operations within the AIC:

1. Calculates active interrupt (higher than current or spurious).
2. Determines and returns the vector of the active interrupt.
3. Memorizes the interrupt.
4. Pushes the current priority level onto the internal stack.
5. Acknowledges the interrupt.

However, while the Protect mode is activated, only operations 1 to 3 are performed when AIC_IVR is read. Operations 4 and 5 are only performed by the AIC when AIC_IVR is written.

Software that has been written and debugged using the Protect mode runs correctly in normal mode without modification. However, in normal mode, the AIC_IVR write has no effect and can be removed to optimize the code.

17.8.6 Spurious Interrupt

The Advanced Interrupt Controller features a protection against spurious interrupts. A spurious interrupt is defined as being the assertion of an interrupt source long enough for the AIC to assert the nIRQ, but no longer present when AIC_IVR is read. This is most prone to occur when:

- An external interrupt source is programmed in Level-Sensitive mode and an active level occurs for only a short time.
- An internal interrupt source is programmed in level-sensitive and the output signal of the corresponding embedded peripheral is activated for a short time (as is the case for the watchdog).
- An interrupt occurs just a few cycles before the software begins to mask it, thus resulting in a pulse on the interrupt source.

The AIC detects a spurious interrupt at the time AIC_IVR is read while no enabled interrupt source is pending. When this happens, the AIC returns the value stored by the programmer in the Spurious Vector Register

(AIC_SPU). The programmer must store the address of a spurious interrupt handler in AIC_SPU as part of the application, to enable an as fast as possible return to the normal execution flow. This handler writes in AIC_EOICR and performs a return from interrupt.

17.8.7 General Interrupt Mask

The AIC features a General Interrupt Mask bit (GMSK in AIC_DCR) to prevent interrupts from reaching the processor. Both the nIRQ and the nFIQ lines are driven to their inactive state if the GMSK is set. However, this mask does not prevent waking up the processor if it has entered Idle mode. This function facilitates synchronizing the processor on a next event and, as soon as the event occurs, performs subsequent operations without having to handle an interrupt. It is strongly recommended to use this mask with caution.

17.8.8 Register Write Protection

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

If a write access to a write-protected register is detected, the WPVS flag in the AIC Write Protection Status Register (AIC_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 AIC_WPSR.

The following registers can be write-protected:

- AIC Source Mode Register
- AIC Source Vector Register
- AIC Spurious Interrupt Vector Register
- AIC Debug Control Register

17.9 Advanced Interrupt Controller (AIC) User Interface

Table 17-3. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Source Select Register	AIC_SSR	Read/Write	0x0
0x04	Source Mode Register	AIC_SMR	Read/Write	0x0
0x08	Source Vector Register	AIC_SVR	Read/Write	0x0
0x0C	Reserved	–	–	–
0x10	Interrupt Vector Register	AIC_IVR	Read-only	0x0
0x14	FIQ Vector Register	AIC_FVR	Read-only	0x0
0x18	Interrupt Status Register	AIC_ISR	Read-only	0x0
0x1C	Reserved	–	–	–
0x20	Interrupt Pending Register 0(2)	AIC_IPR0	Read-only	0x0(1)
0x24	Interrupt Pending Register 1(2)	AIC_IPR1	Read-only	0x0(1)
0x28	Interrupt Pending Register 2(2)	AIC_IPR2	Read-only	0x0(1)
0x2C	Interrupt Pending Register 3(2)	AIC_IPR3	Read-only	0x0(1)
0x30	Interrupt Mask Register	AIC_IMR	Read-only	0x0
0x34	Core Interrupt Status Register	AIC_CISR	Read-only	0x0
0x38	End of Interrupt Command Register	AIC_EOICR	Write-only	–
0x3C	Spurious Interrupt Vector Register	AIC_SPU	Read/Write	0x0
0x40	Interrupt Enable Command Register	AIC_IECR	Write-only	–
0x44	Interrupt Disable Command Register	AIC_IDCR	Write-only	–
0x48	Interrupt Clear Command Register	AIC_ICCR	Write-only	–
0x4C	Interrupt Set Command Register	AIC_ISCR	Write-only	–
0x50–0x5C	Reserved	–	–	–
0x60–0x68	Reserved	–	–	–
0x6C	Debug Control Register	AIC_DCR	Read/Write	0x0
0x70–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	AIC_WPMR	Read/Write	0x0
0xE8	Write Protection Status Register	AIC_WPSR	Read-only	0x0
0xEC–0xFC	Reserved	–	–	–

Notes: 1. The reset value of this register depends on the level of the external interrupt source. All other sources are cleared at reset, thus not pending.

2. PID2...PID127 bit fields refer to the identifiers as defined in Section 8.2 “Peripheral Identifiers”.

17.9.1 AIC Source Select Register

Name: AIC_SSR

Address: 0xFC06E000 (AIC), 0xFC068400 (SAIC)

Access: Read/Write

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
–	INTSEL						

- **INTSEL: Interrupt Line Selection**

0–127 = Selects the interrupt line to handle.

Refer to Section 17.8.1.1 “Interrupt Source Mode”.

17.9.2 AIC Source Mode Register

Name: AIC_SMR

Address: 0xFC06E004 (AIC), 0xFC068404 (SAIC)

Access: Read/Write

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
–	SRCTYPE		–	–	PRIOR		

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

- **PRIOR: Priority Level**

Programs the priority level of the source selected by INTSEL except FIQ source (source 0).

The priority level can be between 0 (lowest) and 7 (highest).

The priority level is not used for the FIQ.

- **SRCTYPE: Interrupt Source Type**

The active level or edge is not programmable for the internal interrupt source selected by INTSEL.

Value	Name	Description
0	INT_LEVEL_SENSITIVE	High level Sensitive for internal source Low level Sensitive for external source
1	INT_EDGE_TRIGGERED	Positive edge triggered for internal source Negative edge triggered for external source
2	EXT_HIGH_LEVEL	High level Sensitive for internal source High level Sensitive for external source
3	EXT_POSITIVE_EDGE	Positive edge triggered for internal source Positive edge triggered for external source

17.9.3 AIC Source Vector Register

Name: AIC_SVR

Address: 0xFC06E008 (AIC), 0xFC068408 (SAIC)

Access: Read/Write

31	30	29	28	27	26	25	24
VECTOR							

23	22	21	20	19	18	17	16
VECTOR							

15	14	13	12	11	10	9	8
VECTOR							

7	6	5	4	3	2	1	0
VECTOR							

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

- **VECTOR: Source Vector**

The user may store in this register the address of the corresponding handler for the interrupt source selected by INTSEL.

17.9.4 AIC Interrupt Vector Register

Name: AIC_IVR

Address: 0xFC06E010 (AIC), 0xFC068410 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
IRQV							

23	22	21	20	19	18	17	16
IRQV							

15	14	13	12	11	10	9	8
IRQV							

7	6	5	4	3	2	1	0
IRQV							

- **IRQV: Interrupt Vector Register**

The Interrupt Vector Register contains the vector programmed by the user in the Source Vector Register corresponding to the current interrupt.

The Source Vector Register is indexed using the current interrupt number when the Interrupt Vector Register is read.

When there is no current interrupt, the Interrupt Vector Register reads the value stored in AIC_SPU.

17.9.5 AIC FIQ Vector Register

Name: AIC_FVR

Address: 0xFC06E014 (AIC), 0xFC068414 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
FIQV							

23	22	21	20	19	18	17	16
FIQV							

15	14	13	12	11	10	9	8
FIQV							

7	6	5	4	3	2	1	0
FIQV							

- **FIQV: FIQ Vector Register**

The FIQ Vector Register contains the vector programmed by the user in the Source Vector Register when INTSEL = 0. When there is no fast interrupt, the FIQ Vector Register reads the value stored in AIC_SPU.

17.9.6 AIC Interrupt Status Register

Name: AIC_ISR

Address: 0xFC06E018 (AIC), 0xFC068418 (SAIC)

Access: Read-only

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
–	IRQID						

- **IRQID: Current Interrupt Identifier**

The Interrupt Status Register returns the current interrupt source number.

17.9.7 AIC Interrupt Pending Register 0

Name: AIC_IPR0

Address: 0xFC06E020 (AIC), 0xFC068420 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	SYS	FIQ

- **FIQ: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

- **SYS: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

- **PIDx: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

17.9.8 AIC Interrupt Pending Register 1

Name: AIC_IPR1

Address: 0xFC06E024 (AIC), 0xFC068424 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
23	22	21	20	19	18	17	16
PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
15	14	13	12	11	10	9	8
PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
7	6	5	4	3	2	1	0
PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32

- **PIDx: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

17.9.9 AIC Interrupt Pending Register 2

Name: AIC_IPR2

Address: 0xFC06E028 (AIC), 0xFC068428 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
PID95	PID94	PID93	PID92	PID91	PID90	PID89	PID88
23	22	21	20	19	18	17	16
PID87	PID86	PID85	PID84	PID83	PID82	PID81	PID80
15	14	13	12	11	10	9	8
PID79	PID78	PID77	PID76	PID75	PID74	PID73	PID72
7	6	5	4	3	2	1	0
PID71	PID70	PID69	PID68	PID67	PID66	PID65	PID64

- **PIDx: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

17.9.10 AIC Interrupt Pending Register 3

Name: AIC_IPR3

Address: 0xFC06E02C (AIC), 0xFC06842C (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
PID127	PID126	PID125	PID124	PID123	PID122	PID121	PID120
23	22	21	20	19	18	17	16
PID119	PID118	PID117	PID116	PID115	PID114	PID113	PID112
15	14	13	12	11	10	9	8
PID111	PID110	PID109	PID108	PID107	PID106	PID105	PID104
7	6	5	4	3	2	1	0
PID103	PID102	PID101	PID100	PID99	PID98	PID97	PID96

- **PIDx: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

17.9.11 AIC Interrupt Mask Register

Name: AIC_IMR

Address: 0xFC06E030 (AIC), 0xFC068430 (SAIC)

Access: Read-only

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
–	–	–	–	–	–	–	INTM

- **INTM: Interrupt Mask**

0: The interrupt source selected by INTSEL is disabled.

1: The interrupt source selected by INTSEL is enabled.

17.9.12 AIC Core Interrupt Status Register

Name: AIC_CISR

Address: 0xFC06E034 (AIC), 0xFC068434 (SAIC)

Access: Read-only

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
–	–	–	–	–	–	NIRQ	NFIQ

- **NFIQ: NFIQ Status**

0: nFIQ line is deactivated.

1: nFIQ line is active.

- **NIRQ: NIRQ Status**

0: nIRQ line is deactivated.

1: nIRQ line is active.

17.9.13 AIC End of Interrupt Command Register

Name: AIC_EOICR

Address: 0xFC06E038 (AIC), 0xFC068438 (SAIC)

Access: Write-only

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
–	–	–	–	–	–	–	ENDIT

- **ENDIT: Interrupt Processing Complete Command**

The End of Interrupt Command Register is used by the interrupt routine to indicate that the interrupt treatment is complete. Any value can be written because it is only necessary to make a write to this register location to signal the end of interrupt treatment.

17.9.14 AIC Spurious Interrupt Vector Register

Name: AIC_SPU

Address: 0xFC06E03C (AIC), 0xFC06843C (SAIC)

Access: Read/Write

31	30	29	28	27	26	25	24
SIVR							

23	22	21	20	19	18	17	16
SIVR							

15	14	13	12	11	10	9	8
SIVR							

7	6	5	4	3	2	1	0
SIVR							

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

- **SIVR: Spurious Interrupt Vector Register**

The user may store the address of a spurious interrupt handler in this register. The written value is returned in AIC_IVR in case of a spurious interrupt, or in AIC_FVR in case of a spurious fast interrupt.

17.9.15 AIC Interrupt Enable Command Register

Name: AIC_IECR

Address: 0xFC06E040 (AIC), 0xFC068440 (SAIC)

Access: Write-only

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
–	–	–	–	–	–	–	INTEN

- **INTEN: Interrupt Enable**

0: No effect.

1: Enables the interrupt source selected by INTSEL.

17.9.16 AIC Interrupt Disable Command Register

Name: AIC_IDCR

Address: 0xFC06E044 (AIC), 0xFC068444 (SAIC)

Access: Write-only

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
–	–	–	–	–	–	–	INTD

- **INTD: Interrupt Disable**

0: No effect.

1: Disables the interrupt source selected by INTSEL.

17.9.17 AIC Interrupt Clear Command Register

Name: AIC_ICCR

Address: 0xFC06E048 (AIC), 0xFC068448 (SAIC)

Access: Write-only

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
–	–	–	–	–	–	–	INTCLR

- **INTCLR: Interrupt Clear**

Clears one the following depending on the setting of the INTSEL bit FIQ, SYS, PID2-PID127

0: No effect.

1: Clears the interrupt source selected by INTSEL.

17.9.18 AIC Interrupt Set Command Register

Name: AIC_ISCR

Address: 0xFC06E04C (AIC), 0xFC06844C (SAIC)

Access: Write-only

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
–	–	–	–	–	–	–	INTSET

- **INTSET: Interrupt Set**

0: No effect.

1: Sets the interrupt source selected by INTSEL.

17.9.19 AIC Debug Control Register

Name: AIC_DCR

Address: 0xFC06E06C (AIC), 0xFC06846C (SAIC)

Access: Read/Write

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
–	–	–	–	–	–	GMSK	PROT

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

- **PROT: Protection Mode**

0: The Protection mode is disabled.

1: The Protection mode is enabled.

- **GMSK: General Interrupt Mask**

0: The nIRQ and nFIQ lines are normally controlled by the AIC.

1: The nIRQ and nFIQ lines are tied to their inactive state.

17.9.20 AIC Write Protection Mode Register

Name: AIC_WPMR

Address: 0xFC06E0E4 (AIC), 0xFC0684E4 (SAIC)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x414943 (“AIC” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x414943 (“AIC” in ASCII).

Refer to Section 17.8.8 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x414943	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

17.9.21 AIC Write Protection Status Register

Name: AIC_WPSR

Address: 0xFC06E0E8 (AIC), 0xFC0684E8 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of AIC_WPSR.

1: A write protection violation has occurred since the last read of AIC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

18. Watchdog Timer (WDT)

18.1 Description

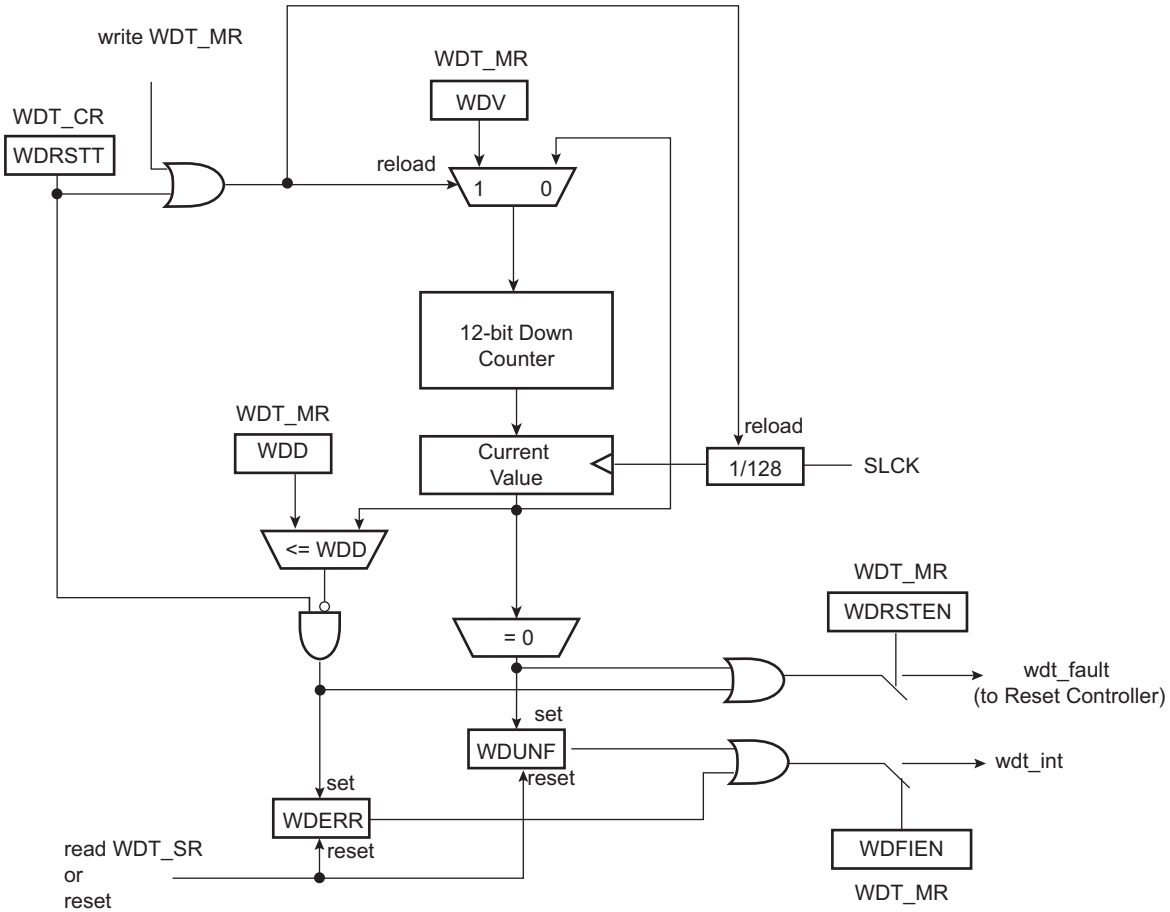
The Watchdog Timer (WDT) is used to prevent system lock-up if the software becomes trapped in a deadlock. It features a 12-bit down counter that allows a watchdog period of up to 16 seconds (slow clock around 32 kHz). It can generate a general reset or a processor reset only. In addition, it can be stopped while the processor is in Debug mode or Sleep mode (Idle mode).

18.2 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

18.3 Block Diagram

Figure 18-1. Watchdog Timer Block Diagram



18.4 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 powerup. 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 until a LOCKMR command is issued in WDT_CR. 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).

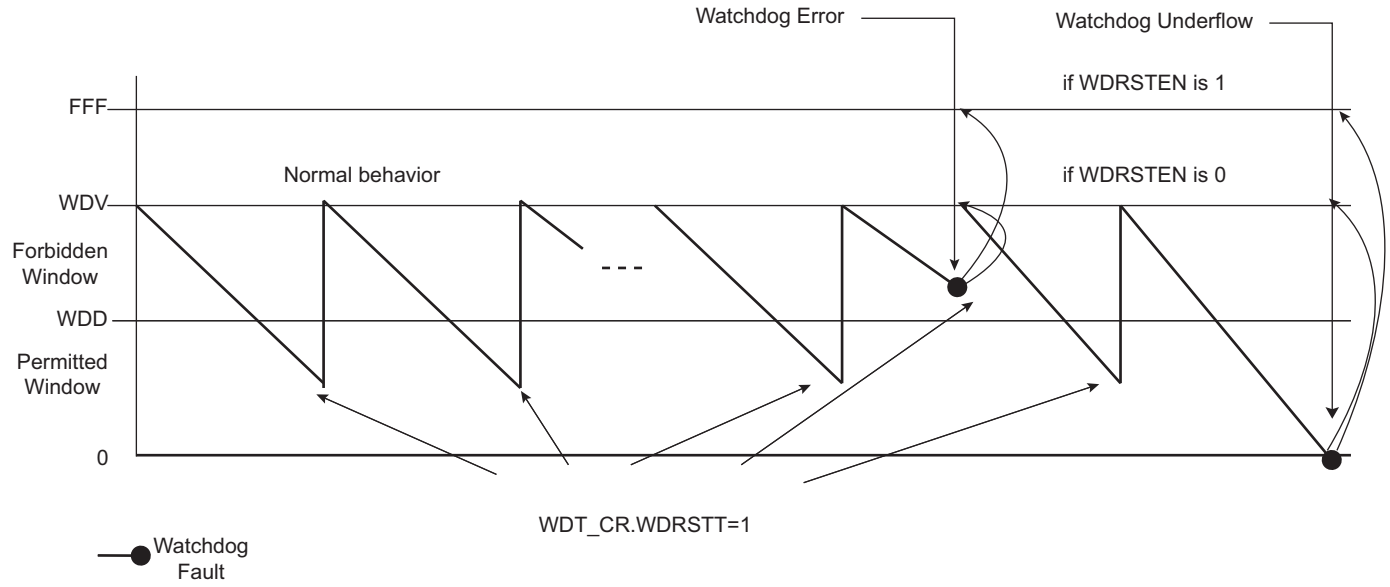
The status bits WDUNF (Watchdog Underflow) and WDERR (Watchdog Error) trigger an interrupt, provided the bit WDT_MR.WDFIEN is set. The signal “wdt_fault” to the Reset Controller causes a watchdog reset if the WDRSTEN bit is set as already explained in the Reset Controller documentation. In this case, 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 or in Sleep mode, the counter may be stopped depending on the value programmed for the bits WDIDLEHLT and WDDBGHLT in WDT_MR.

Figure 18-2. Watchdog Behavior



18.5 Watchdog Timer (WDT) User Interface

Table 18-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	WDT_CR	Write-only	–
0x04	Mode Register	WDT_MR	Read/Write	0x3FFF_2FFF
0x08	Status Register	WDT_SR	Read-only	0x0000_0000

18.5.1 Watchdog Timer Control Register

Name: WDT_CR

Address: 0xFC068640

Access: Write-only

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

Note: 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.

- **WDRSTT: Watchdog Restart**

0: No effect.

1: Restarts the watchdog if KEY is written to 0xA5.

- **LOCKMR: Lock Mode Register Write Access**

0: No effect.

1: Locks the Mode Register (WDT_MR) if KEY is written to 0xA5, write access to WDT_MR has no effect.

- **KEY: Password**

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation.

18.5.2 Watchdog Timer Mode Register

Name: WDT_MR

Address: 0xFC068644

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	WDIDLEHLT	WDDBGHLT	WDD			
23	22	21	20	19	18	17	16
WDD							
15	14	13	12	11	10	9	8
WDDIS	–	WDRSTEN	WDFIEN	WDV			
7	6	5	4	3	2	1	0
WDV							

Notes:
1. Write access to this register has no effect if the LOCKMR command is issued in WDT_CR (unlocked on hardware reset).
2. 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.

- **WDV: Watchdog Counter Value**

Defines the value loaded in the 12-bit watchdog counter.

- **WDFIEN: Watchdog Fault Interrupt Enable**

0: A watchdog fault (underflow or error) has no effect on interrupt.

1: A watchdog fault (underflow or error) asserts interrupt.

- **WDRSTEN: Watchdog Reset Enable**

0: A watchdog fault (underflow or error) has no effect on the resets.

1: A watchdog fault (underflow or error) triggers a watchdog reset.

- **WDDIS: Watchdog Disable**

0: Enables the Watchdog Timer.

1: Disables the Watchdog Timer.

Note: When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.

- **WDD: 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.

- **WDDBGHLT: Watchdog Debug Halt**

0: The watchdog runs when the processor is in debug state.

1: The watchdog stops when the processor is in debug state.

- **WDIDLEHLT: Watchdog Idle Halt**

0: The watchdog runs when the system is in idle state.

1: The watchdog stops when the system is in idle state.

18.5.3 Watchdog Timer Status Register

Name: WDT_SR

Address: 0xFC068648

Access Read-only

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
–	–	–	–	–	–	WDERR	WDUNF

- **WDUNF: Watchdog Underflow (cleared on read)**

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.

- **WDERR: Watchdog Error (cleared on read)**

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.

19. Reset Controller (RSTC)

19.1 Description

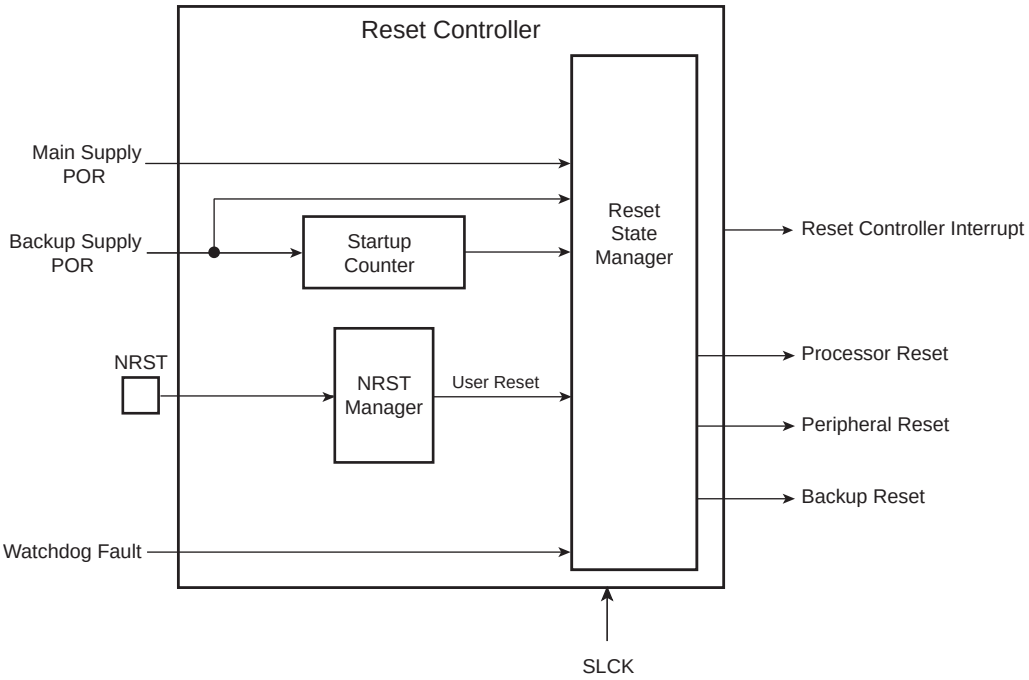
The Reset Controller (RSTC), based on power-on reset cells, handles all the resets of the system without any external components. It reports which reset occurred last.

19.2 Embedded Characteristics

- Manages All Resets of the System, Including
  - Processor Reset
  - Peripheral Reset
  - Backed-up Peripheral Reset
- Based on 2 Embedded Power-on Reset Cells
- Reset Source Status
  - Status of the Last Reset
  - Either General Reset, Wakeup Reset, Software Reset, User Reset, Watchdog Reset

19.3 Block Diagram

Figure 19-1. Reset Controller Block Diagram



19.4 Functional Description

19.4.1 Reset Controller Overview

The Reset Controller is made up of an NRST Manager, a Startup Counter and a Reset State Manager. It runs at Slow Clock and generates the following reset signals:

- Processor Reset: resets the processor and the Watchdog Timer.
- Peripheral Reset: resets the whole set of embedded peripherals.

- Backup Reset: resets all the peripherals powered by VDDBU.

These reset signals are asserted by the Reset Controller, either on external events or on software action. The Reset State Manager controls the generation of reset signals.

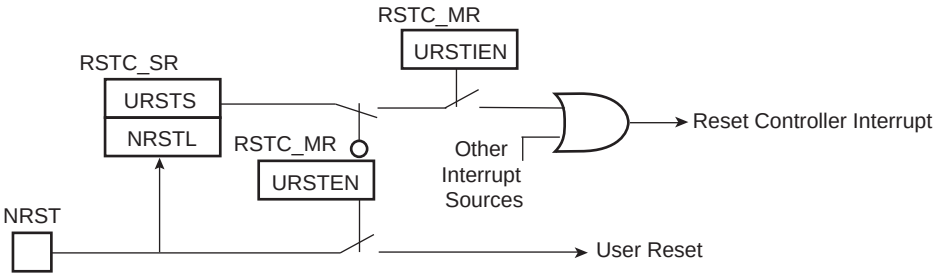
The startup counter waits for the complete crystal oscillator startup. The wait delay is given by the crystal oscillator startup time maximum value that can be found in Section 55.5 “Crystal Oscillator Characteristics”.

The Reset Controller Mode Register (RSTC_MR), used to configure the reset controller, is powered with VDDBU, so that its configuration is saved as long as VDDBU is on.

19.4.2 NRST Manager

The NRST Manager samples the NRST input pin and drives this pin low when required by the Reset State Manager. Figure 19-2 shows the block diagram of the NRST Manager.

Figure 19-2. NRST Manager



19.4.2.1 NRST Signal or Interrupt

The NRST Manager samples the NRST pin at Slow Clock speed. When the line is detected low, a User Reset is reported to the Reset State Manager.

However, the NRST Manager can be programmed to not trigger a reset when an assertion of NRST occurs. Writing a zero to the URSTEN bit in the RSTC_MR disables the User Reset trigger.

The level of the pin NRST can be read at any time in the bit NRSTL (NRST level) in the Reset Controller Status Register (RSTC_SR). As soon as the pin NRST is asserted, the bit URSTS in the RSTC_SR is set. This bit clears only when RSTC_SR is read.

The reset controller can also be programmed to generate an interrupt instead of generating a reset. To do so, the bit URSTIEN in the RSTC_MR must be set.

19.4.3 Reset States

The Reset State Manager handles the different reset sources and generates the internal reset signals. It reports the reset status in the field RSTTYP of the RSTC_SR. The update of the field RSTTYP is performed when the processor reset is released.

19.4.3.1 General Reset

A general reset occurs when VDDBU and VDDCORE are powered on. The backup supply POR cell output rises and is filtered with a Startup Counter, which operates at Slow Clock. The purpose of this counter is to make sure the Slow Clock oscillator is stable before starting up the device. The length of startup time is hardcoded to comply with the Slow Clock Oscillator startup time.

After this time, the processor clock is released at Slow Clock and all the other signals remain valid for 2 cycles for proper processor and logic reset. Then, all the reset signals are released and the field RSTTYP in the RSTC_SR reports a General Reset.

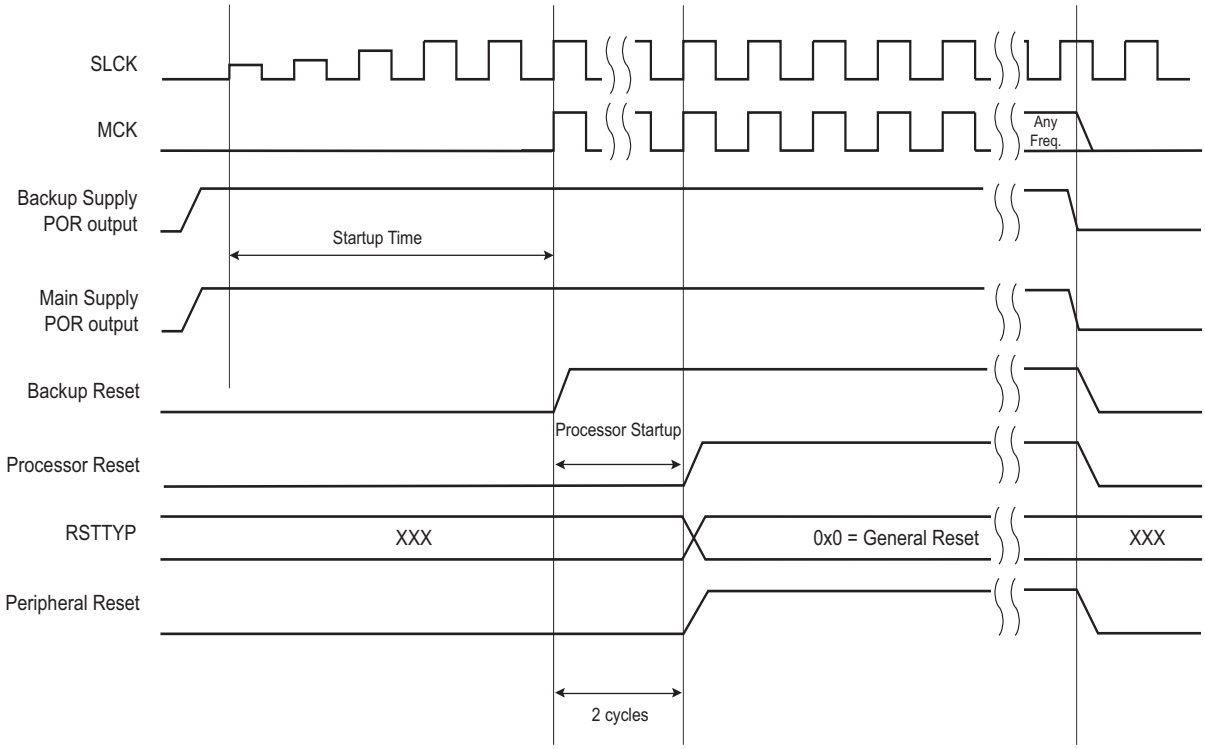
When VDDBU is detected low by the backup supply POR cell, all resets signals are immediately asserted, even if the main supply POR cell does not report a main supply shutdown.

VDDBU only activates the Backup Reset signal.

Backup Reset must be released so that any other reset can be generated by VDDCORE (main supply POR output).

Figure 19-3 shows how the General Reset affects the reset signals.

Figure 19-3. General Reset State



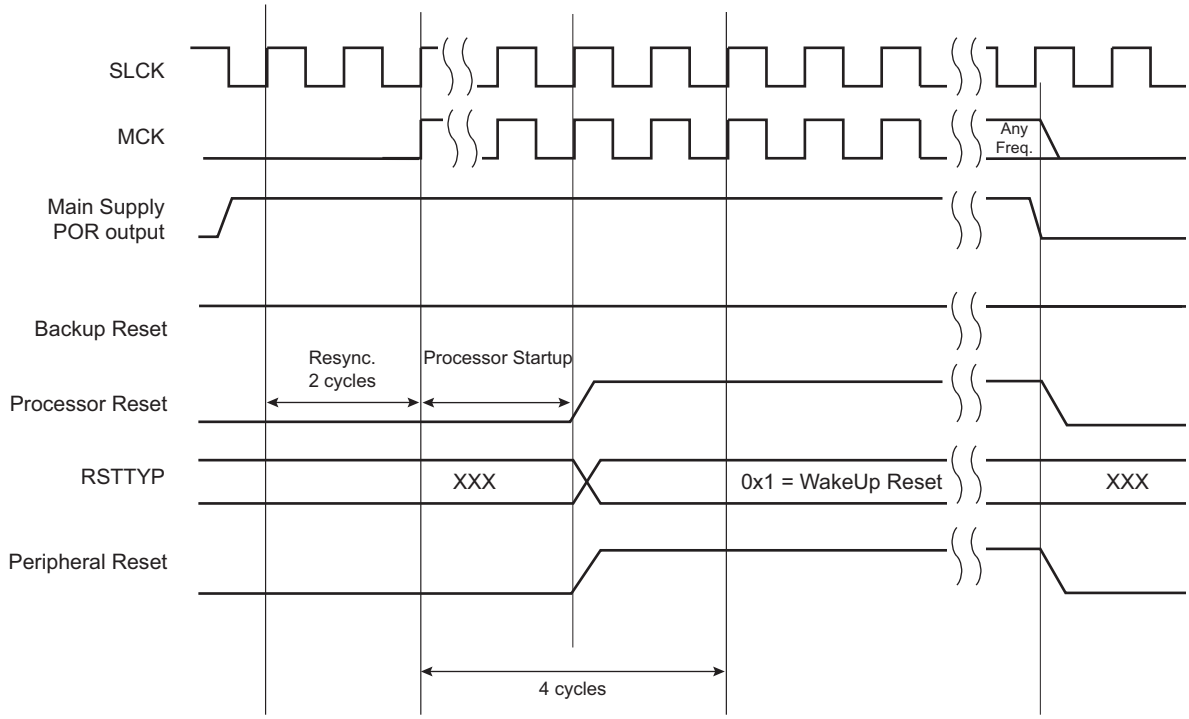
19.4.3.2 Wakeup Reset

The wakeup reset occurs when the main supply is down. When the main supply POR output is active, all the reset signals are asserted except Backup Reset. When the main supply powers up, the POR output is resynchronized on Slow Clock. The processor clock is then re-enabled during 2 Slow Clock cycles, depending on the requirements of the ARM processor.

At the end of this delay, the processor and other reset signals rise. The field RSTTYP in the RSTC_SR is updated to report a wakeup reset.

When the main supply is detected falling, the reset signals are immediately asserted. This transition is synchronous with the output of the main supply POR.

Figure 19-4. Wakeup Reset



19.4.3.3 User Reset

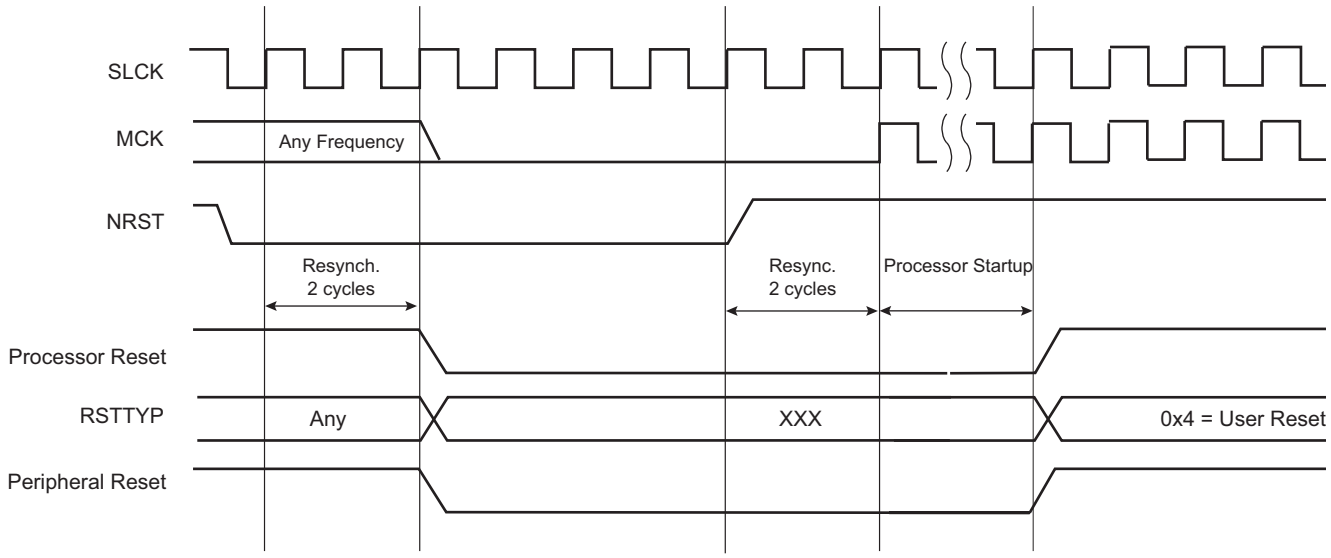
The User Reset is entered when a low level is detected on the NRST pin and the bit URSTEN in RSTC_MR is at 1. The NRST input signal is resynchronized with SLCK to ensure proper behavior of the system.

The Processor Reset and the Peripheral Reset are asserted.

The User Reset is left when NRST rises, after a two-cycle resynchronization time and a 2-cycle processor startup. The processor clock is re-enabled as soon as NRST is confirmed high.

When the processor reset signal is released, the RSTTYP field of the RSTC_SR is loaded with the value 0x4, indicating a User Reset.

Figure 19-5. User Reset State



19.4.3.4 Software Reset

The Reset Controller offers several commands used to assert the different reset signals. These commands are performed by writing the Control Register (RSTC_CR) with the following bits at 1:

- PROCRST: Writing PROCRST at 1 resets the processor and the watchdog timer.
- PERRST: Writing PERRST at 1 resets all the embedded peripherals, including the memory system, and, in particular, the Remap Command. The Peripheral Reset is generally used for debug purposes.
  PERRST must always be used in conjunction with PROCRST (PERRST and PROCRST set both at 1 simultaneously.)

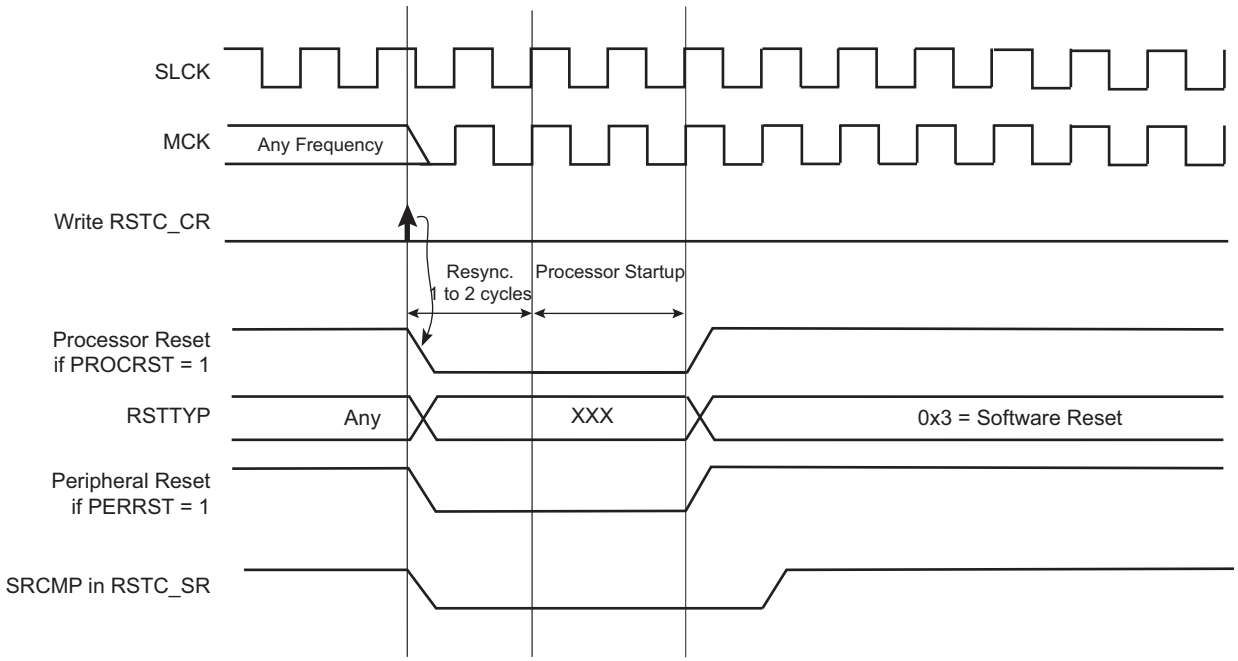
The software reset is entered if at least one of these bits is set by the software. All these commands can be performed independently or simultaneously. The software reset lasts two Slow Clock cycles.

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 is left, i.e., synchronously to SLCK.

If and only if the PROCRST bit is set, the reset controller reports the software status in the field RSTTYP of the RSTC_SR. Other software resets are not reported in RSTTYP.

As soon as a software operation is detected, the bit SRCMP (Software Reset Command in Progress) is set in the RSTC_SR. It is cleared as soon as the software reset is left. No other software reset can be performed while the SRCMP bit is set, and writing any value in RSTC_CR has no effect.

Figure 19-6. Software Reset



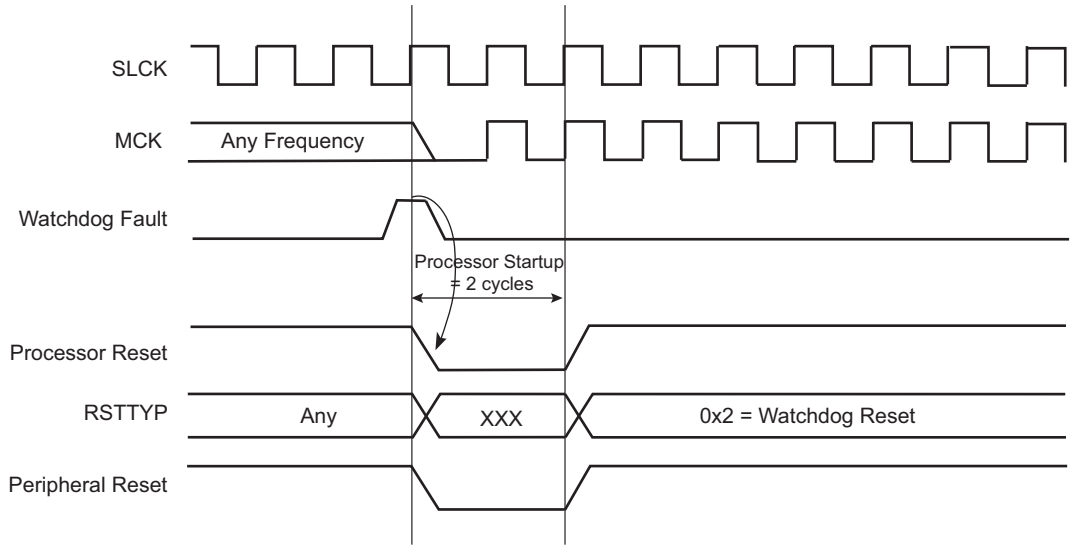
19.4.3.5 Watchdog Reset

The Watchdog Reset is entered when a watchdog fault occurs. This state lasts two Slow Clock cycles.

The Watchdog Timer is reset by the Processor Reset signal. As the watchdog fault always causes a processor reset if WDRSTEN is set, 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 the WDRSTEN in WDT_MR bit is reset, the watchdog fault has no impact on the reset controller.

Figure 19-7. Watchdog Reset



19.4.4 Reset State Priorities

The Reset State Manager manages the following priorities between the different reset sources, given in descending order:

- Backup Reset
- Wakeup Reset
- Watchdog Reset
- Software Reset
- User Reset

Particular cases are listed below:

- When in User Reset:
  - A watchdog event is impossible because the Watchdog Timer is being reset by the Processor Reset 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.

19.5 Reset Controller (RSTC) User Interface

Table 19-1. Register Mapping

Offset	Register	Name	Access	Reset	Back-up Reset
0x00	Control Register	RSTC_CR	Write-only	–	–
0x04	Status Register	RSTC_SR	Read-only	0x0000_0100 (1)	0x0000_0000 (2)
0x08	Mode Register	RSTC_MR	Read/Write	–	0x0000_0000

Notes: 1. Only power supply VDDCORE rising
2. Both power supplies VDDCORE and VDDBU rising

19.5.1 Reset Controller Control Register

Name: RSTC_CR

Access: Write-only

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

- **PROCRST: Processor Reset**

0: No effect

1: If KEY value = 0xA5, resets the processor

- **PERRST: Peripheral Reset**

0: No effect

1: If KEY value = 0xA5, resets the peripherals

- **KEY: Write Access Password**

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

19.5.2 Reset Controller Status Register

Name: RSTC_SR

Access: Read-only

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

- **URSTS: User Reset Status**

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. Reading the RSTC_SR resets the URSTS bit and clears the interrupt.

- **RSTTYP: 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	Both VDDCORE and VDDBU rising
1	WKUP_RST	VDDCORE rising
2	WDT_RST	Watchdog fault occurred
3	SOFT_RST	Processor reset required by the software
4	USER_RST	NRST pin detected low

- **NRSTL: NRST Pin Level**

This bit records the level of the NRST pin sampled on each Master Clock (MCK) rising edge.

- **SRCMP: Software Reset Command in Progress**

0: No software command is being performed by the reset controller. The reset controller is ready for a software command.

1: A software reset command is being performed by the reset controller. The reset controller is busy.

19.5.3 Reset Controller Mode Register

Name: RSTC_MR

Access: Read/Write

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	URSTIEN	–	–	–	URSTEN

- **URSTEN: User Reset Enable**

0: The detection of a low level on the pin NRST does not generate a User Reset.

1: The detection of a low level on the pin NRST triggers a User Reset.

- **URSTIEN: User Reset Interrupt Enable**

0: USRTS bit in RSTC_SR at 1 has no effect on the Reset Controller Interrupt.

1: USRTS bit in RSTC_SR at 1 asserts the Reset Controller Interrupt if URSTEN = 0.

- **KEY: Write Access Password**

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

20. Shutdown Controller (SHDWC)

20.1 Description

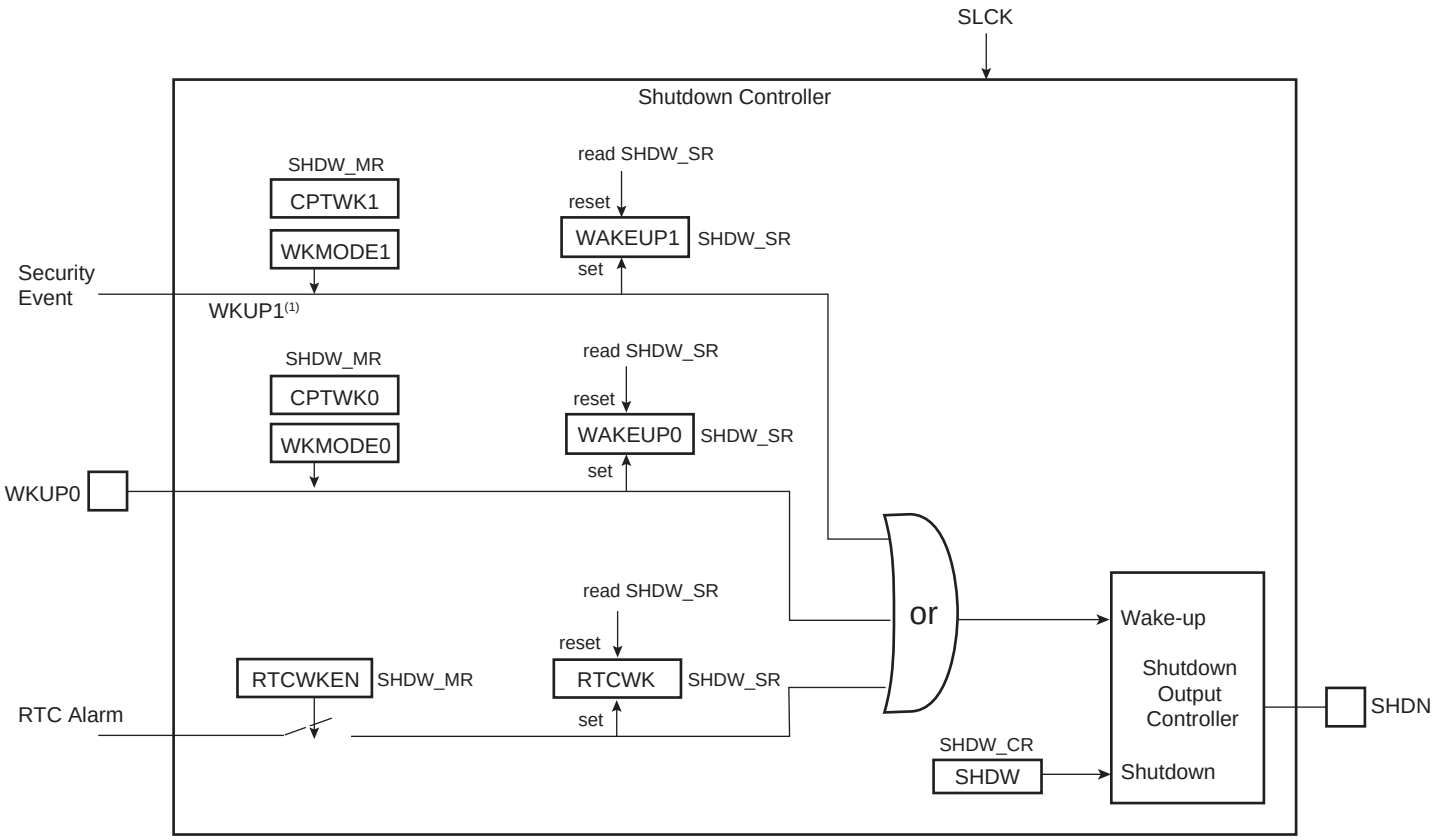
The Shutdown Controller (SHDWC) controls the power supplies VDDIO and VDDCORE and the wakeup detection on debounced input lines.

20.2 Embedded Characteristics

- Shutdown Logic
  - Software Assertion of the Shutdown Output Pin (SHDN)
  - Programmable De-assertion from the WKUP Input Pins
- Wakeup Logic
  - Programmable Assertion from the WKUP Input Pins, and Internal Wakeup Event from RTC

20.3 Block Diagram

Figure 20-1. Shutdown Controller Block Diagram



Note: 1. WKUP1 input connected to Secure Box Module (SBM)

20.4 I/O Lines Description

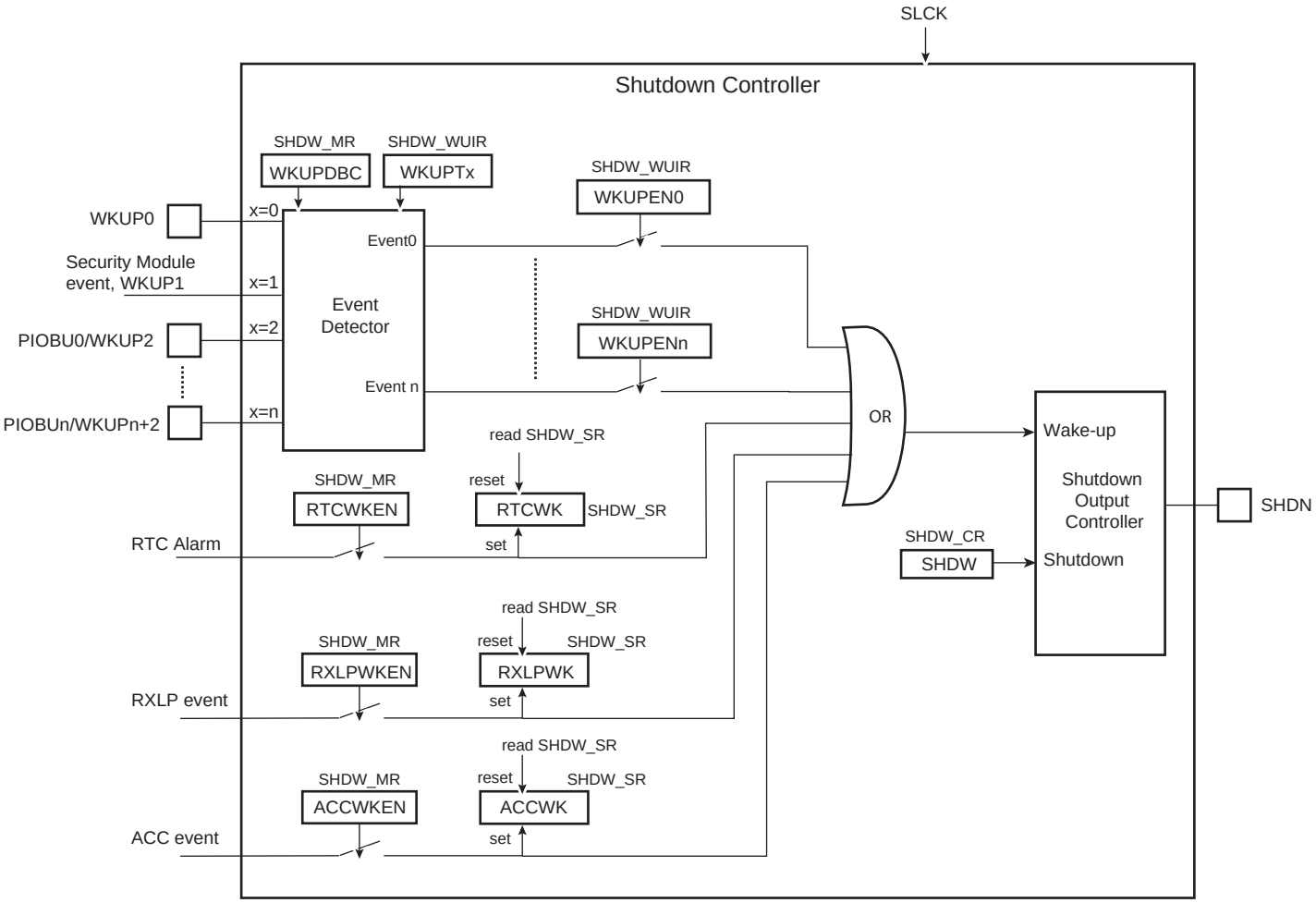


Table 20-1. I/O Lines Description

Name	Description	Type
WKUP0	Wakeup 0 input	Input
WKUP1	Wakeup 1 input	Input
SHDN	Shutdown output	Output

20.5 Product Dependencies

20.5.1 Power Management

The Shutdown Controller is continuously clocked by the Slow Clock (SLCK). The Power Management Controller has no effect on the behavior of the Shutdown Controller.

20.6 Functional Description

The Shutdown Controller manages the main power supply. To do so, it is supplied with VDDBU and manages wakeup input pins and one output pin, SHDN.

A typical application connects the pin SHDN to the shutdown input of the DC/DC Converter providing the main power supplies of the system, and especially VDDCORE and/or VDDIO. The wakeup inputs (WKUP0, WKUP1) connect to any push-buttons or signal that wake up the system.

The software is able to control the pin SHDN by writing the Shutdown Control Register (SHDW_CR) with the bit SHDW at 1. The shutdown is taken into account only two slow clock cycles after the write of SHDW_CR. This register is password-protected and so the value written should contain the correct key for the command to be taken into account. As a result, the system should be powered down.

20.6.1 Wakeup Inputs

A level change on WKUP0 or WKUP1 can trigger a wakeup. Wakeup is configured in the Shutdown Mode Register (SHDW_MR). The transition detector can be programmed to detect either a positive or negative transition or any level change on WKUP0 and WKUP1. The detection can also be disabled. Programming is performed by defining WKMODE0 and WKMODE1.

Moreover, a debouncing circuit can be programmed for WKUP0 or WKUP1. The debouncing circuit filters pulses on WKUP0 or WKUP1 shorter than the programmed number of 16 SLCK cycles in CPTWK0 or CPTWK1 of the SHDW_MR. If the programmed level change is detected on a pin, a counter starts. When the counter reaches the value programmed in the corresponding field, CPTWK0 or CPTWK1, the SHDN pin is released. If a new input change is detected before the counter reaches the corresponding value, the counter is stopped and cleared. WAKEUP0 and/or WAKEUP1 of the Status Register (SHDW_SR) reports the detection of the programmed events on WKUP0 or WKUP1, with a reset after the read of SHDW_SR.

The Shutdown Controller can be programmed so as to activate the wakeup using the RTC alarm (detection of the rising edge event is synchronized with SLCK). This is done by writing the SHDW_MR using the RTCWKEN bit. When enabled, the detection of RTC alarm is reported in the RTCWK bit of SHDW_SR. They are cleared after reading SHDW_SR. When using the RTC alarm to wake up the system, the user must ensure that RTC alarm status flag is cleared before shutting down the system. Otherwise, no rising edge of the status flags may be detected and the wakeup will fail.

20.7 Shutdown Controller (SHDWC) User Interface

Table 20-2. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Shutdown Control Register	SHDW_CR	Write-only	–
0x04	Shutdown Mode Register	SHDW_MR	Read/Write	0x0000_0303
0x08	Shutdown Status Register	SHDW_SR	Read-only	0x0000_0000

20.7.1 Shutdown Control Register

Name: SHDW_CR

Address: 0xFC068610

Access: Write-only

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SHDW

- **SHDW: Shutdown Command**

0: No effect.

1: If KEY value is correct, asserts the SHDN pin.

- **KEY: Password**

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation.

20.7.2 Shutdown Mode Register

Name: SHDW_MR

Address: 0xFC068614

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	RTCWKEN	–

15	14	13	12	11	10	9	8
CPTWK1				–	–	WKMODE1	

7	6	5	4	3	2	1	0
CPTWK0				–	–	WKMODE0	

- **WKMODE0: Wakeup Mode 0**

Value	Name	Description
0	NO_DETECTION	No detection is performed on the wakeup input
1	RISING_EDGE	Low to high transition triggers the detection process
2	FALLING_EDGE	High to low level transition triggers the detection process
3	ANY_EDGE	Any edge on the wakeup input triggers the detection process

- **CPTWK0: Debounce Counter on Wakeup 0**

Defines the minimum duration of the WKUP1 pin after the occurrence of the selected triggering edge (WKMODE0).

The SHDN pin is released if the WKUP0 holds the selected level for (CPTWK $\times$ 16 + 1) consecutive Slow Clock cycles after the occurrence of the selected triggering edge on WKUP0.

- **WKMODE1: Wakeup Mode 1**

Value	Name	Description
0	NO_DETECTION	No detection is performed on the wakeup input
1	RISING_EDGE	Low to high transition triggers the detection process
2	FALLING_EDGE	High to low level transition triggers the detection process
3	ANY_EDGE	Any edge on the wakeup input triggers the detection process

- **CPTWK1: Debounce Counter on Wakeup 1**

Defines the minimum duration of the WKUP1 pin after the occurrence of the selected triggering edge (WKMODE1).

The SHDN pin is released if the WKUP1 holds the selected level for (CPTWK $\times$ 16 + 1) consecutive Slow Clock cycles after the occurrence of the selected triggering edge on WKUP1.

- **RTCWKEN: Real-time Clock Wakeup Enable**

0: The RTC Alarm signal has no effect on the Shutdown Controller.

1: The RTC Alarm signal forces the de-assertion of the SHDN pin.

20.7.3 Shutdown Status Register

Name: SHDW_SR

Address: 0xFC068618

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	RTCWK	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	WAKEUP1	WAKEUP0

- **WAKEUP0: Wakeup 0 Status**

0: No wakeup event occurred on WKUP0 input since the last read of SHDW_SR.

1: At least one wakeup event occurred on WKUP0 input since the last read of SHDW_SR.

- **WAKEUP1: Wakeup 1 Status**

0: No wakeup event occurred on WKUP1 input since the last read of SHDW_SR.

1: At least one wakeup event occurred on WKUP1 input since the last read of SHDW_SR.

- **RTCWK: Real-time Clock Wakeup**

0: No wakeup alarm from the RTC occurred since the last read of SHDW_SR.

1: At least one wakeup alarm from the RTC occurred since the last read of SHDW_SR.

21. Periodic Interval Timer (PIT)

21.1 Description

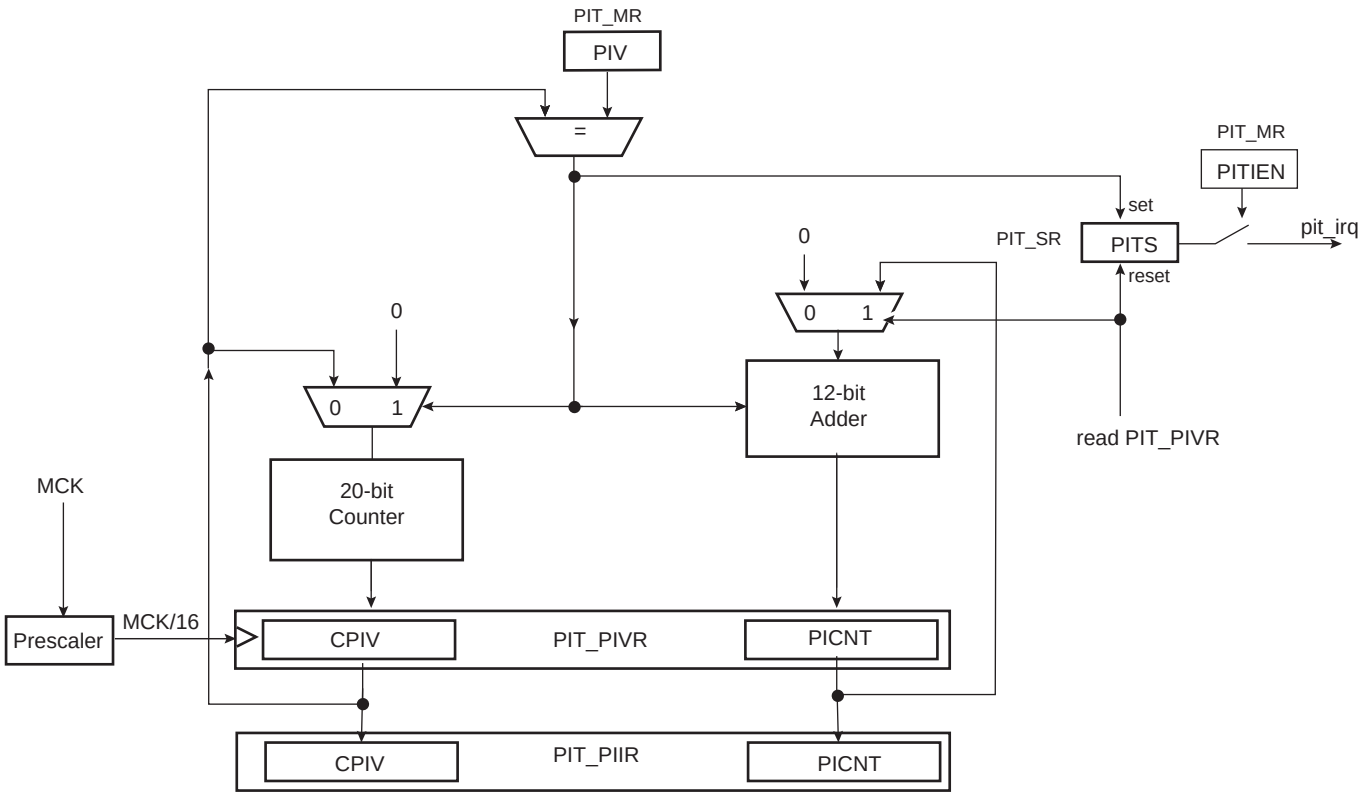
The Periodic Interval Timer (PIT) provides the operating system's scheduler interrupt. It is designed to offer maximum accuracy and efficient management, even for systems with long response time.

21.2 Embedded Characteristics

- 20-bit Programmable Counter plus 12-bit Interval Counter
- Reset-on-read Feature
- Both Counters Work on Master Clock/16

21.3 Block Diagram

Figure 21-1. Periodic Interval Timer



21.4 Functional Description

The Periodic Interval Timer aims at providing periodic interrupts for use by operating systems.

The PIT provides a programmable overflow counter and a reset-on-read feature. It is built around two counters: a 20-bit CPIV counter and a 12-bit PICNT counter. Both counters work at Master Clock /16.

The first 20-bit CPIV counter increments from 0 up to a programmable overflow value set in the field PIV of the Mode Register (PIT_MR). When the counter CPIV reaches this value, it resets to 0 and increments the Periodic Interval Counter, PICNT. The status bit PITS in the Status Register (PIT_SR) rises and triggers an interrupt, provided the interrupt is enabled (PITIEN in PIT_MR).

Writing a new PIV value in PIT_MR does not reset/restart the counters.

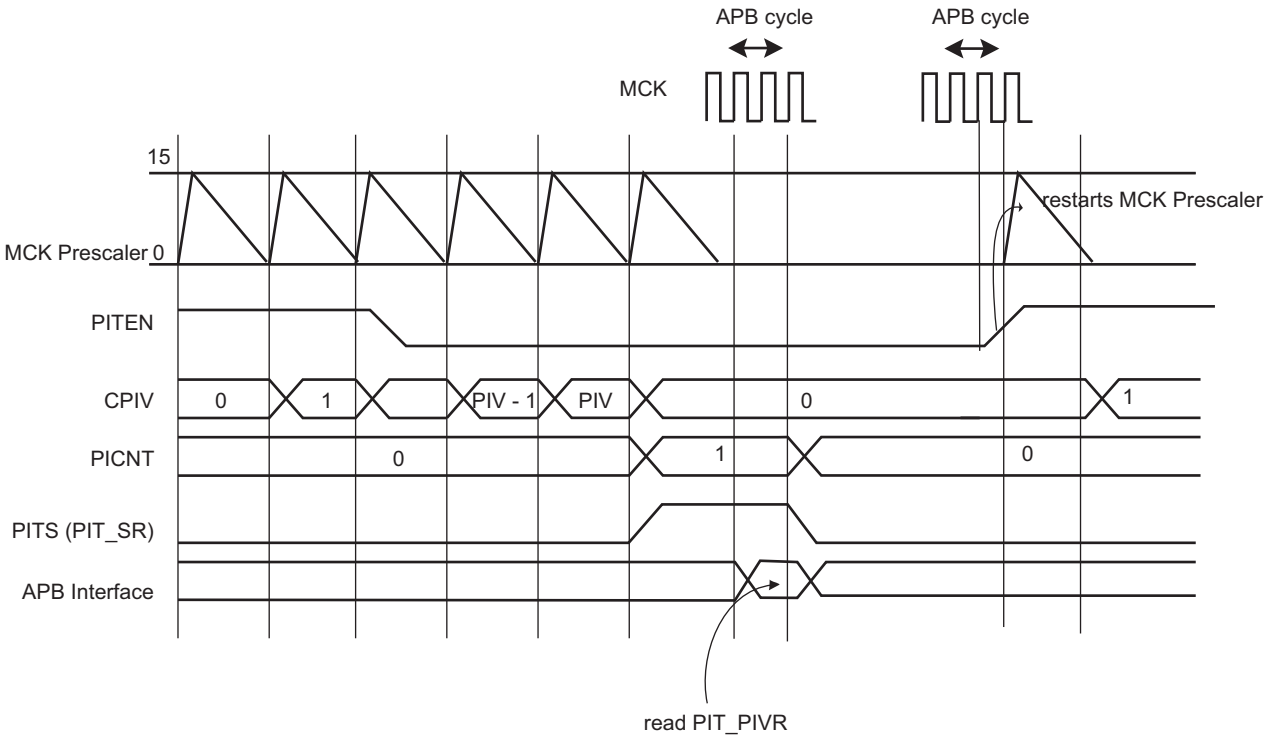
When CPIV and PICNT values are obtained by reading the Periodic Interval Value Register (PIT_PIVR), the overflow counter (PICNT) is reset and the PITS bit is cleared, thus acknowledging the interrupt. The value of PICNT gives the number of periodic intervals elapsed since the last read of PIT_PIVR.

When CPIV and PICNT values are obtained by reading the Periodic Interval Image Register (PIT_PIIR), there is no effect on the counters CPIV and PICNT, nor on the bit PITS. For example, a profiler can read PIT_PIIR without clearing any pending interrupt, whereas a timer interrupt clears the interrupt by reading PIT_PIVR.

The PIT may be enabled/disabled using the PITEN bit in the PIT_MR register (disabled on reset). The PITEN bit only becomes effective when the CPIV value is 0. Figure 21-2 illustrates the PIT counting. After the PIT Enable bit is reset (PITEN = 0), the CPIV goes on counting until the PIV value is reached, and is then reset. PIT restarts counting, only if the PITEN is set again.

The PIT is stopped when the core enters debug state.

Figure 21-2. Enabling/Disabling PIT with PITEN



21.5 Periodic Interval Timer (PIT) User Interface

Table 21-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Mode Register	PIT_MR	Read/Write	0x000F_FFFF
0x04	Status Register	PIT_SR	Read-only	0x0000_0000
0x08	Periodic Interval Value Register	PIT_PIVR	Read-only	0x0000_0000
0x0C	Periodic Interval Image Register	PIT_PIIR	Read-only	0x0000_0000

21.5.1 Periodic Interval Timer Mode Register

Name: PIT_MR

Address: 0xFC068630

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	PITIEN	PITEN

23	22	21	20	19	18	17	16
–	–	–	–	PIV			

15	14	13	12	11	10	9	8
PIV							

7	6	5	4	3	2	1	0
PIV							

- **PIV: Periodic Interval Value**

Defines the value compared with the primary 20-bit counter of the Periodic Interval Timer (CPIV). The period is equal to (PIV + 1).

- **PITEN: Period Interval Timer Enabled**

0: The Periodic Interval Timer is disabled when the PIV value is reached.

1: The Periodic Interval Timer is enabled.

- **PITIEN: Periodic Interval Timer Interrupt Enable**

0: The bit PITS in PIT_SR has no effect on interrupt.

1: The bit PITS in PIT_SR asserts interrupt.

21.5.2 Periodic Interval Timer Status Register

Name: PIT_SR

Address: 0xFC068634

Access: Read-only

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
–	–	–	–	–	–	–	PITS

- **PITS: Periodic Interval Timer Status**

0: The Periodic Interval timer has not reached PIV since the last read of PIT_PIVR.

1: The Periodic Interval timer has reached PIV since the last read of PIT_PIVR.

21.5.3 Periodic Interval Timer Value Register

Name: PIT_PIVR

Address: 0xFC068638

Access: Read-only

31	30	29	28	27	26	25	24
PICNT							

23	22	21	20	19	18	17	16
PICNT				CPIV			

15	14	13	12	11	10	9	8
CPIV							

7	6	5	4	3	2	1	0
CPIV							

Reading this register clears PITS in PIT_SR.

- **CPIV: Current Periodic Interval Value**

Returns the current value of the periodic interval timer.

- **PICNT: Periodic Interval Counter**

Returns the number of occurrences of periodic intervals since the last read of PIT_PIVR.

21.5.4 Periodic Interval Timer Image Register

Name: PIT_PIIR

Address: 0xFC06863C

Access: Read-only

31	30	29	28	27	26	25	24
PICNT							

23	22	21	20	19	18	17	16
PICNT				CPIV			

15	14	13	12	11	10	9	8
CPIV							

7	6	5	4	3	2	1	0
CPIV							

- **CPIV: Current Periodic Interval Value**

Returns the current value of the periodic interval timer.

- **PICNT: Periodic Interval Counter**

Returns the number of occurrences of periodic intervals since the last read of PIT_PIVR.

22. Real-time Clock (RTC)

22.1 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.

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.

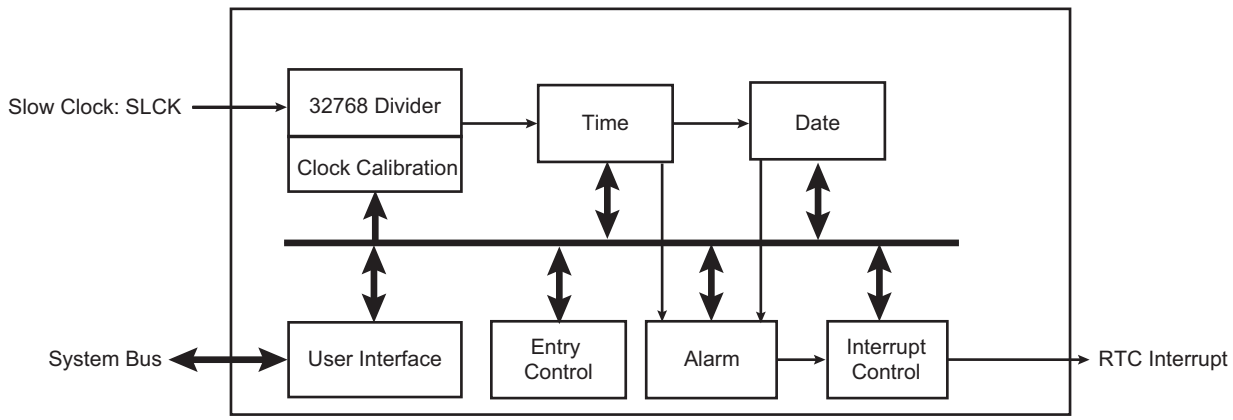
Timestamping capability reports the first and last occurrences of tamper events.

22.2 Embedded Characteristics

- Full Asynchronous Design for Ultra-low-power Consumption
- Gregorian and Persian Modes Supported
- 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
- Tamper Timestamping Registers
- Register Write Protection

22.3 Block Diagram

Figure 22-1. Real-time Clock Block Diagram



22.4 Product Dependencies

22.4.1 Power Management

The Real-time Clock is continuously clocked at 32.768 kHz. The Power Management Controller has no effect on RTC behavior.

22.4.2 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.

Table 22-1. Peripheral IDs

Instance	ID
RTC	1

22.5 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) and RTC Calendar Register (RTC_CALR).

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.

22.5.1 Reference Clock

The reference clock is the Slow Clock (SLCK). It 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 has to take into account the current consumption for power saving and the frequency drift due to temperature effect on the circuit for time accuracy.

22.5.2 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.

22.5.3 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 alarm (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.

22.5.4 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.

22.5.5 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).

Anyway 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.

22.5.6 Updating Time/Calendar

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 Figure 22-2.

When entering the Programming mode of the calendar fields, the time fields remain enabled. When entering the Programming mode of the time fields, 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 22-2. Time/Calendar Update Timing Diagram

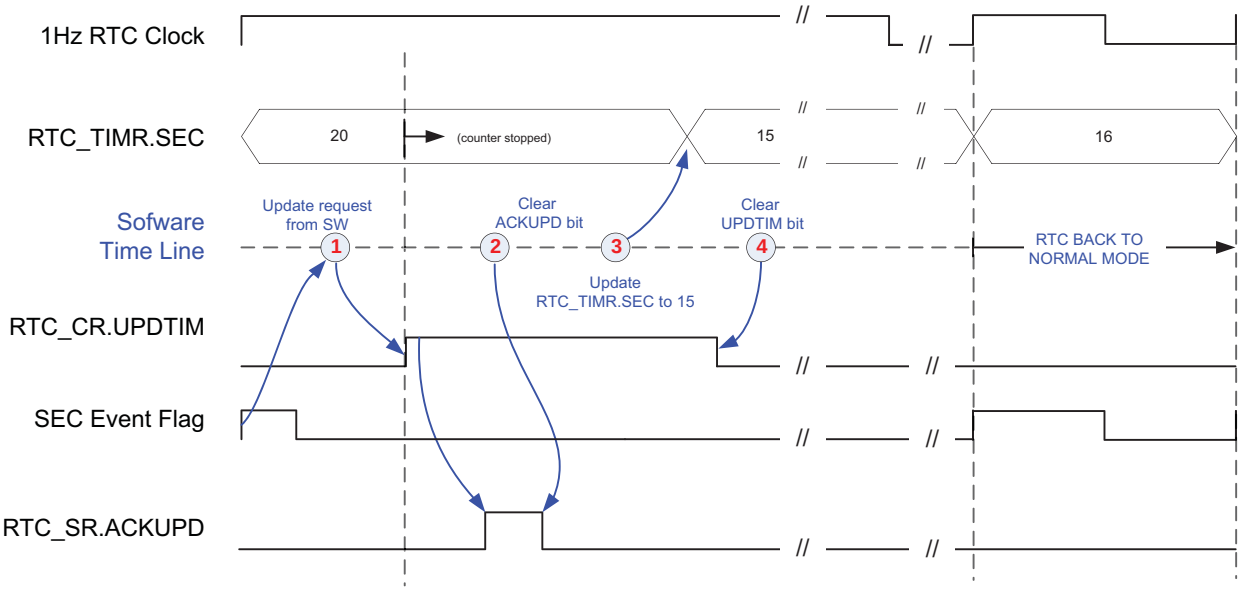
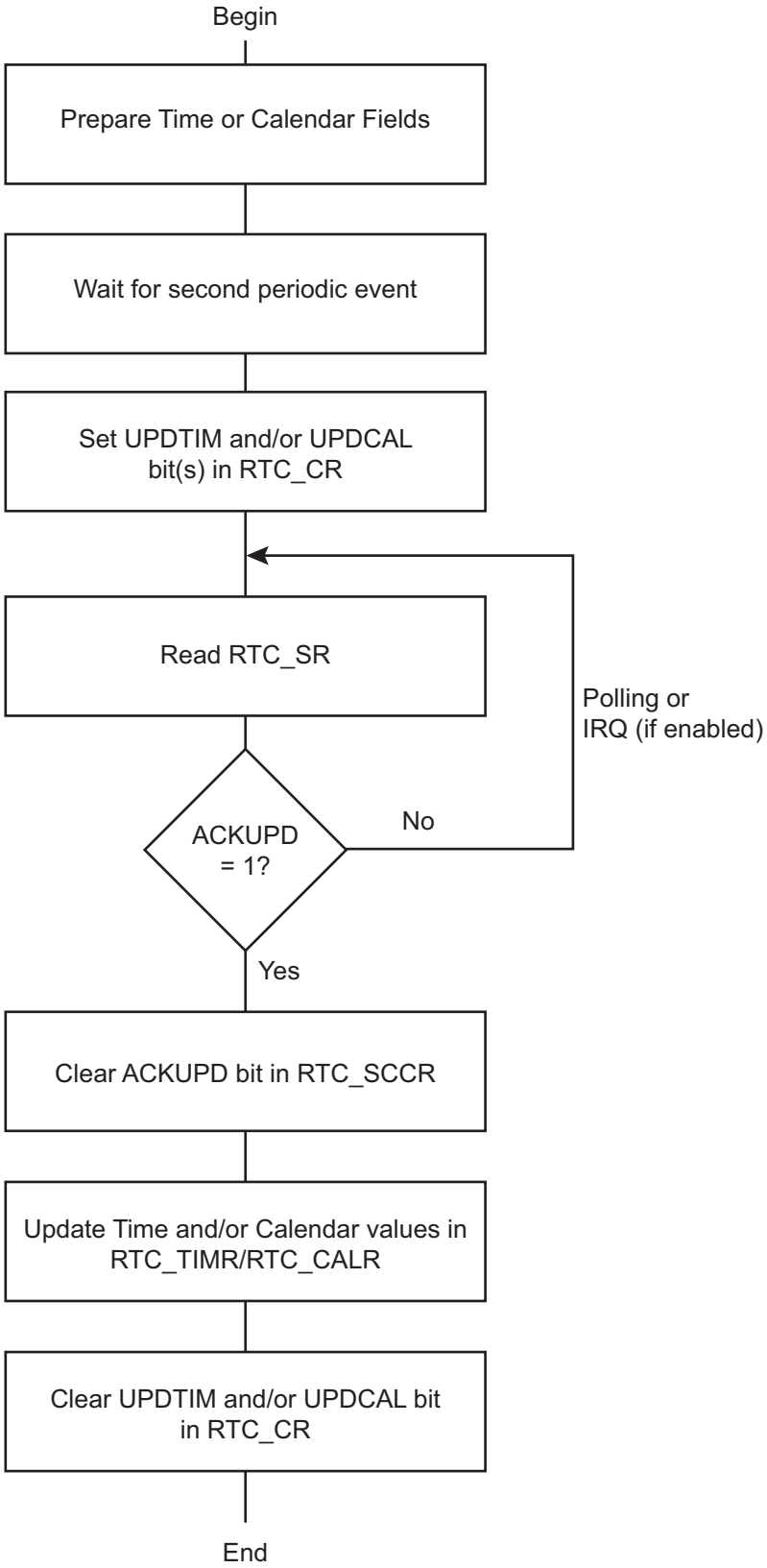


Figure 22-3. Gregorian and Persian Modes Update Sequence



22.5.7 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 22-4. Calibration Circuitry

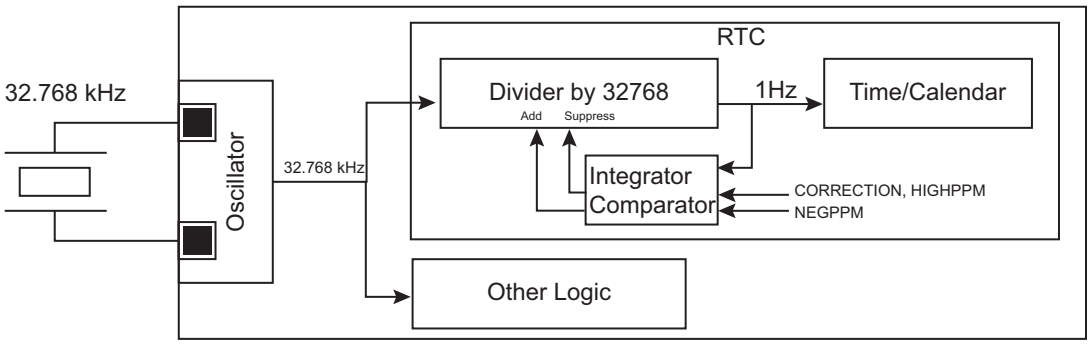
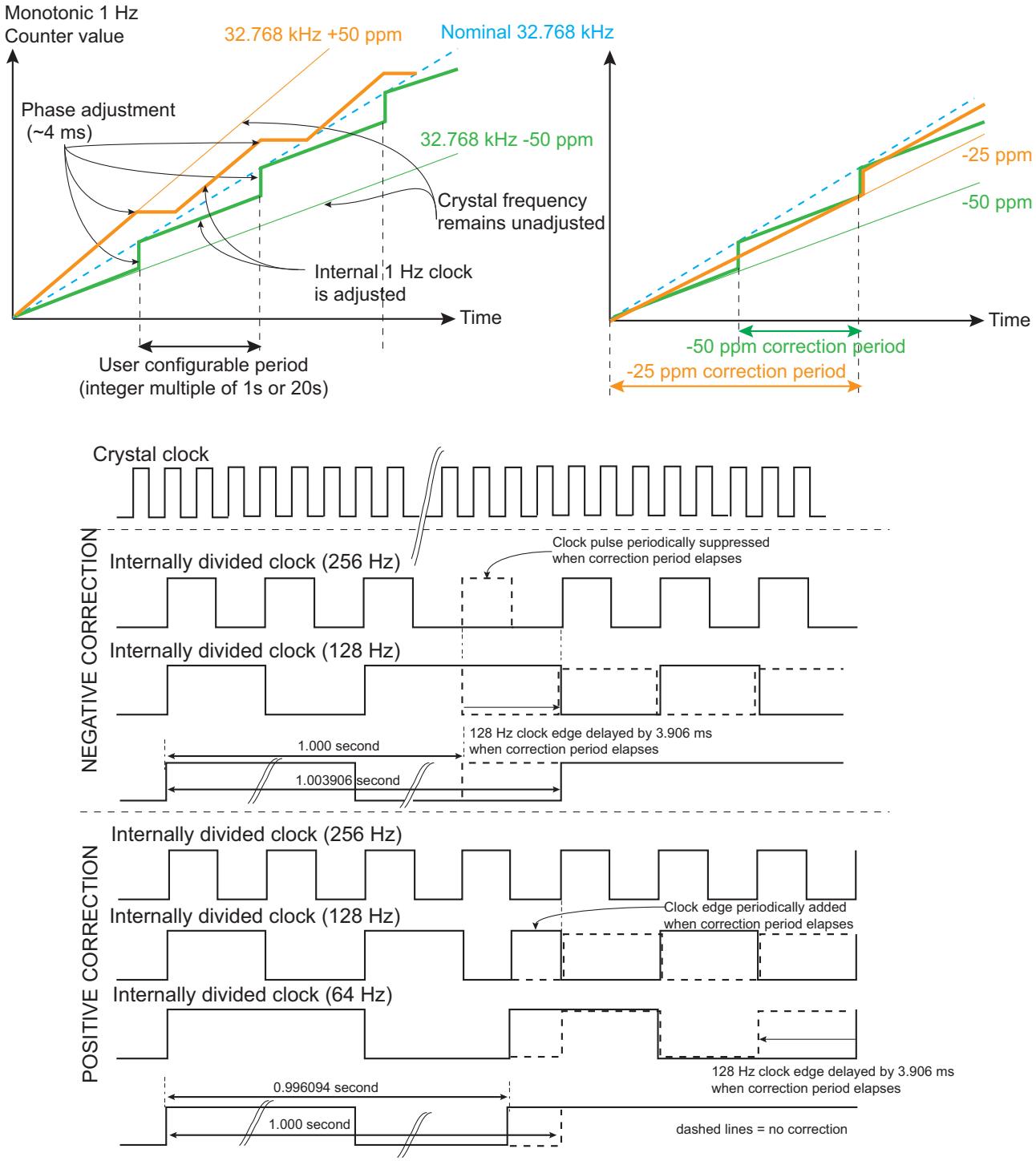


Figure 22-5. Calibration Circuitry Waveforms



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.

In any event, 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.

22.5.8 Tamper Timestamping

As soon as a tamper is detected, the tamper counter is incremented and the RTC stores the time of the day, the date and the source of the tamper event in registers located in the backup area. Up to two tamper events can be stored.

The tamper counter saturates at 15. Once this limit is reached, the exact number of tamper occurrences since the last read of stamping registers cannot be known.

The first set of timestamping registers (RTC_TSTR0, RTC_TSDR0, RTC_TSSR0) cannot be overwritten, so once they have been written all data are stored until the registers are reset. Therefore these registers are storing the first tamper occurrence after a read.

The second set of timestamping registers (RTC_TSTR1, RTC_TSDR1, RTC_TSSR1) are overwritten each time a tamper event is detected. Thus the date and the time data of the first and the second stamping registers may be equal. This occurs when the tamper counter value carried on field TEVCNT in RTC_TSTR0 equals 1. Thus this second set of registers stores the last occurrence of tamper before a read.

Reading a set of timestamping registers requires three accesses, one for the time of the day, one for the date and one for the tamper source.

Reading the third part (RTC_TSSR0/1) of a timestamping register set clears the whole content of the registers (time, date and tamper source) and makes the timestamping registers available to store a new event.

22.6 Real-time Clock (RTC) User Interface

Table 22-2. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	RTC_CR	Read/Write	0x00000000
0x04	Mode Register	RTC_MR	Read/Write	0x00000000
0x08	Time Register	RTC_TIMR	Read/Write	0x00000000
0x0C	Calendar Register	RTC_CALR	Read/Write	0x01E111220
0x10	Time Alarm Register	RTC_TIMALR	Read/Write	0x00000000
0x14	Calendar Alarm Register	RTC_CALALR	Read/Write	0x01010000
0x18	Status Register	RTC_SR	Read-only	0x00000000
0x1C	Status Clear Command Register	RTC_SCCR	Write-only	–
0x20	Interrupt Enable Register	RTC_IER	Write-only	–
0x24	Interrupt Disable Register	RTC_IDR	Write-only	–
0x28	Interrupt Mask Register	RTC_IMR	Read-only	0x00000000
0x2C	Valid Entry Register	RTC_VER	Read-only	0x00000000
0xB0	TimeStamp Time Register 0	RTC_TSTR0	Read-only	0x00000000
0xB4	TimeStamp Date Register 0	RTC_TSDR0	Read-only	0x00000000
0xB8	TimeStamp Source Register 0	RTC_TSSR0	Read-only	0x00000000
0xBC	TimeStamp Time Register 1	RTC_TSTR1	Read-only	0x00000000
0xC0	TimeStamp Date Register 1	RTC_TSDR1	Read-only	0x00000000
0xC4	TimeStamp Source Register 1	RTC_TSSR1	Read-only	0x00000000
0xC8	Reserved	–	–	–
0xCC	Reserved	–	–	–
0xD0	Reserved	–	–	–
0xD4–0xF8	Reserved	–	–	–
0xFC	Reserved	–	–	–

Note: If an offset is not listed in the table it must be considered as reserved.

22.6.1 RTC Control Register

Name: RTC_CR

Address: 0xFC0686B0

Access: Read/Write

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

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

- **UPDTIM: Update Request Time Register**

0: No effect or, if UPDTIM has been previously written to 1, stops the update procedure.

1: Stops the RTC time counting.

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.

- **UPDCAL: Update Request Calendar Register**

0: No effect or, if UPDCAL has been previously written to 1, stops the update procedure.

1: Stops the RTC calendar counting.

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.

- **TIMEVSEL: 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

- **CALEVSEL: 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	–	Reserved

22.6.2 RTC Mode Register

Name: RTC_MR

Address: 0xFC0686B4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
HIGHPPM	CORRECTION						
7	6	5	4	3	2	1	0
–	–	–	NEGPPM	–	–	PERSIAN	HRMOD

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

- **HRMOD: 12-/24-hour Mode**

0: 24-hour mode is selected.

1: 12-hour mode is selected.

- **PERSIAN: PERSIAN Calendar**

0: Gregorian calendar.

1: Persian calendar.

- **NEGPPM: NEGative PPM Correction**

0: Positive correction (the divider will be slightly higher than 32768).

1: Negative correction (the divider will be slightly lower than 32768).

Refer to CORRECTION and HIGHPPM field descriptions.

Note: NEGPPM must be cleared to correct a crystal slower than 32.768 kHz.

- **CORRECTION: Slow Clock Correction**

0: No correction

1–127: The slow clock will be corrected according to the formula given in HIGHPPM description.

- **HIGHPPM: HIGH PPM Correction**

0: Lower range ppm correction with accurate correction.

1: Higher range ppm correction with accurate 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:

$$CORRECTION = \frac{3906}{20 \times ppm} - 1$$

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:

$$CORRECTION = \frac{3906}{ppm} - 1$$

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).

22.6.3 RTC Time Register

Name: RTC_TIMR

Address: 0xFC0686B8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	AMPM	HOUR					

15	14	13	12	11	10	9	8
–	MIN						

7	6	5	4	3	2	1	0
–	SEC						

- **SEC: 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.

- **MIN: 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.

- **HOUR: Current Hour**

The range that can be set is 1–12 (BCD) in 12-hour mode or 0–23 (BCD) in 24-hour mode.

- **AMPM: Ante Meridiem Post Meridiem Indicator**

This bit is the AM/PM indicator in 12-hour mode.

0: AM.

1: PM.

22.6.4 RTC Calendar Register

Name: RTC_CALR

Address: 0xFC0686BC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	DATE					

23	22	21	20	19	18	17	16
DAY			MONTH				

15	14	13	12	11	10	9	8
YEAR							

7	6	5	4	3	2	1	0
–	CENT						

- **CENT: 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.

- **YEAR: 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.

- **MONTH: 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.

- **DAY: 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.

- **DATE: 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.

22.6.5 RTC Time Alarm Register

Name: RTC_TIMALR

Address: 0xFC0686C0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
HOUREN	AMPM	HOUR					

15	14	13	12	11	10	9	8
MINEN	MIN						

7	6	5	4	3	2	1	0
SECEN	SEC						

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

Note: 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.

- **SEC: Second Alarm**

This field is the alarm field corresponding to the BCD-coded second counter.

- **SECEN: Second Alarm Enable**

0: The second-matching alarm is disabled.

1: The second-matching alarm is enabled.

- **MIN: Minute Alarm**

This field is the alarm field corresponding to the BCD-coded minute counter.

- **MINEN: Minute Alarm Enable**

0: The minute-matching alarm is disabled.

1: The minute-matching alarm is enabled.

- **HOUR: Hour Alarm**

This field is the alarm field corresponding to the BCD-coded hour counter.

- **AMPM: AM/PM Indicator**

This field is the alarm field corresponding to the BCD-coded hour counter.

- **HOUREN: Hour Alarm Enable**

0: The hour-matching alarm is disabled.

1: The hour-matching alarm is enabled.

22.6.6 RTC Calendar Alarm Register

Name: RTC_CALALR

Address: 0xFC0686C4

Access: Read/Write

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

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

Note: 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.

- **MONTH: Month Alarm**

This field is the alarm field corresponding to the BCD-coded month counter.

- **MTHEN: Month Alarm Enable**

0: The month-matching alarm is disabled.

1: The month-matching alarm is enabled.

- **DATE: Date Alarm**

This field is the alarm field corresponding to the BCD-coded date counter.

- **DATEEN: Date Alarm Enable**

0: The date-matching alarm is disabled.

1: The date-matching alarm is enabled.

22.6.7 RTC Status Register

Name: RTC_SR

Address: 0xFC0686C8

Access: Read-only

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
–	–	TDERR	CALEV	TIMEV	SEC	ALARM	ACKUPD

- **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.

- **ALARM: Alarm Flag**

Value	Name	Description
0	NO_ALARMEVENT	No alarm matching condition occurred.
1	ALARMEVENT	An alarm matching condition has occurred.

- **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.

- **TIMEV: Time Event**

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.

Note: 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).

- **CALEV: Calendar Event**

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.

Note: 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.

- **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.

22.6.8 RTC Status Clear Command Register

Name: RTC_SCCR

Address: 0xFC0686CC

Access: Write-only

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
–	–	TDERRCLR	CALCLR	TIMCLR	SECCLR	ALRCLR	ACKCLR

- **ACKCLR: Acknowledge Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **ALRCLR: Alarm Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **SECCLR: Second Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **TIMCLR: Time Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **CALCLR: Calendar Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **TDERRCLR: Time and/or Date Free Running Error Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

22.6.9 RTC Interrupt Enable Register

Name: RTC_IER

Address: 0xFC0686D0

Access: Write-only

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
–	–	TDERREN	CALEN	TIMEN	SECEN	ALREN	ACKEN

- **ACKEN: Acknowledge Update Interrupt Enable**

0: No effect.

1: The acknowledge for update interrupt is enabled.

- **ALREN: Alarm Interrupt Enable**

0: No effect.

1: The alarm interrupt is enabled.

- **SECEN: Second Event Interrupt Enable**

0: No effect.

1: The second periodic interrupt is enabled.

- **TIMEN: Time Event Interrupt Enable**

0: No effect.

1: The selected time event interrupt is enabled.

- **CALEN: Calendar Event Interrupt Enable**

0: No effect.

1: The selected calendar event interrupt is enabled.

- **TDERREN: Time and/or Date Error Interrupt Enable**

0: No effect.

1: The time and date error interrupt is enabled.

22.6.10 RTC Interrupt Disable Register

Name: RTC_IDR

Address: 0xFC0686D4

Access: Write-only

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
–	–	TDERRDIS	CALDIS	TIMDIS	SECDIS	ALRDIS	ACKDIS

- **ACKDIS: Acknowledge Update Interrupt Disable**

0: No effect.

1: The acknowledge for update interrupt is disabled.

- **ALRDIS: Alarm Interrupt Disable**

0: No effect.

1: The alarm interrupt is disabled.

- **SECDIS: Second Event Interrupt Disable**

0: No effect.

1: The second periodic interrupt is disabled.

- **TIMDIS: Time Event Interrupt Disable**

0: No effect.

1: The selected time event interrupt is disabled.

- **CALDIS: Calendar Event Interrupt Disable**

0: No effect.

1: The selected calendar event interrupt is disabled.

- **TDERRDIS: Time and/or Date Error Interrupt Disable**

0: No effect.

1: The time and date error interrupt is disabled.

22.6.11 RTC Interrupt Mask Register

Name: RTC_IMR

Address: 0xFC0686D8

Access: Read-only

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
–	–	TDERR	CAL	TIM	SEC	ALR	ACK

- **ACK: Acknowledge Update Interrupt Mask**

0: The acknowledge for update interrupt is disabled.

1: The acknowledge for update interrupt is enabled.

- **ALR: Alarm Interrupt Mask**

0: The alarm interrupt is disabled.

1: The alarm interrupt is enabled.

- **SEC: Second Event Interrupt Mask**

0: The second periodic interrupt is disabled.

1: The second periodic interrupt is enabled.

- **TIM: Time Event Interrupt Mask**

0: The selected time event interrupt is disabled.

1: The selected time event interrupt is enabled.

- **CAL: Calendar Event Interrupt Mask**

0: The selected calendar event interrupt is disabled.

1: The selected calendar event interrupt is enabled.

- **TDERR: Time and/or Date Error Mask**

0: The time and/or date error event is disabled.

1: The time and/or date error event is enabled.

22.6.12 RTC Valid Entry Register

Name: RTC_VER

Address: 0xFC0686DC

Access: Read-only

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
–	–	–	–	NVCALALR	NVTIMALR	NVCAL	NVTIM

- **NVTIM: Non-valid Time**

0: No invalid data has been detected in RTC_TIMR (Time Register).

1: RTC_TIMR has contained invalid data since it was last programmed.

- **NVCAL: Non-valid Calendar**

0: No invalid data has been detected in RTC_CALR (Calendar Register).

1: RTC_CALR has contained invalid data since it was last programmed.

- **NVTIMALR: Non-valid Time Alarm**

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.

- **NVCALALR: Non-valid Calendar Alarm**

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.

22.6.13 RTC TimeStamp Time Register 0

Name: RTC_TSTR0

Address: 0xFC068760

Access: Read-only

31	30	29	28	27	26	25	24
BACKUP	–	–	–	TEVCNT			
23	22	21	20	19	18	17	16
–	AMPM	HOUR					
15	14	13	12	11	10	9	8
–	MIN						
7	6	5	4	3	2	1	0
–	SEC						

RTC_TSTR0 reports the timestamp of the first tamper event after reading RTC_TSSR0.

This register is cleared by reading RTC_TSSR0.

- **SEC: Seconds of the Tamper**
- **MIN: Minutes of the Tamper**
- **HOUR: Hours of the Tamper**
- **AMPM: AM/PM Indicator of the Tamper**
- **TEVCNT: Tamper Events Counter**

Each time a tamper event occurs, this counter is incremented. This counter saturates at 15. Once this value is reached, it is no more possible to know the exact number of tamper events.

If this field is not null, this implies that at least one tamper event occurs since last register reset and that the values stored in timestamping registers are valid.

- **BACKUP: System Mode of the Tamper**

0: The state of the system is different from backup mode when the tamper event occurs.

1: The system is in backup mode when the tamper event occurs.

22.6.14 RTC TimeStamp Time Register 1

Name: RTC_TSTR1

Address: 0xFC06876C

Access: Read-only

31	30	29	28	27	26	25	24
BACKUP	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	AMPM	HOUR					

15	14	13	12	11	10	9	8
–	MIN						

7	6	5	4	3	2	1	0
–	SEC						

RTC_TSTR1 reports the timestamp of the last tamper event.

This register is cleared by reading RTC_TSSR1.

- **SEC: Seconds of the Tamper**
- **MIN: Minutes of the Tamper**
- **HOUR: Hours of the Tamper**
- **AMPM: AM/PM Indicator of the Tamper**
- **BACKUP: System Mode of the Tamper**

0: The state of the system is different from Backup mode when the tamper event occurs.

1: The system is in Backup mode when the tamper event occurs.

22.6.15 RTC TimeStamp Date Register

Name: RTC_TSDRx

Address: 0xFC068764 [0], 0xFC068770 [1]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	DATE					

23	22	21	20	19	18	17	16
DAY			MONTH				

15	14	13	12	11	10	9	8
YEAR							

7	6	5	4	3	2	1	0
–	CENT						

RTC_TSDR0 reports the timestamp of the first tamper event after reading RTC_TSSR0, and RTC_TSDR1 reports the timestamp of the last tamper event.

This register is cleared by reading RTC_TSSR.

- **CENT: Century of the Tamper**
- **YEAR: Year of the Tamper**
- **MONTH: Month of the Tamper**
- **DAY: Day of the Tamper**
- **DATE: Date of the Tamper**

The fields contain the date and the source of a tamper occurrence if the TEVCNT is not null.

22.6.16 RTC TimeStamp Source Register

Name: RTC_TSSRx

Address: 0xFC068768 [0], 0xFC068774 [1]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
DET7	DET6	DET5	DET4	DET3	DET2	DET1	DET0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	JTAG	TST	–	–

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

0: No alarm generated since the last clear.

1: An alarm has been generated by the corresponding monitor since the last clear.

- **TST: Test Pin Monitor**
- **JTAG: JTAG Pins Monitor**
- **DETx: PIOBU Intrusion Detector**

23. Slow Clock Controller (SCKC)

23.1 Description

The System Controller embeds a Slow Clock Controller (SCKC). The SCKC selects the slow clock from one of two sources:

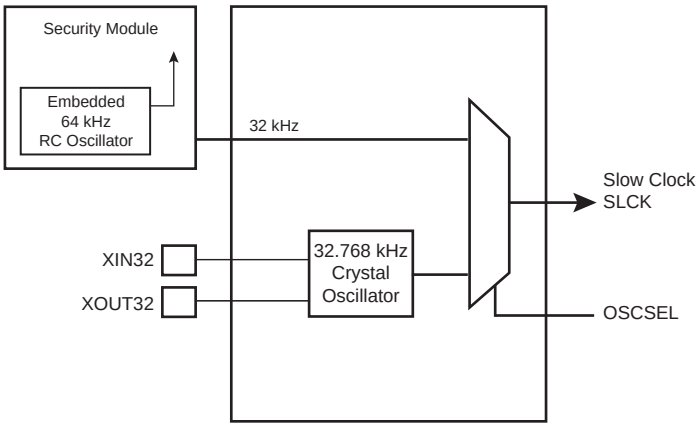
- External 32.768 kHz crystal oscillator
- Embedded 32 kHz (typical) RC oscillator

23.2 Embedded Characteristics

- 32 kHz (typical) RC Oscillator or 32.768 kHz Crystal Oscillator Selector
- VDDBU Powered

23.3 Block Diagram

Figure 23-1. Block Diagram



23.4 Functional Description

The OSCSEL bit is located in the Slow Clock Controller Configuration Register (SCKC_CR) located at the address 0xFC068650 in the backed-up part of the System Controller and, thus, it is preserved while VDDBU is present.

The embedded 32 kHz (typical) RC oscillator and the 32.768 kHz crystal oscillator are always enabled as soon as VDDBU is established. The Slow Clock Selector command (OSCSEL bit) selects the slow clock source.

After the VDDBU power-on reset, the default configuration is OSCSEL = 0, allowing the system to start on the embedded 32 kHz (typical) RC oscillator.

The programmer controls the slow clock switching by software and so must take precautions during the switching phase.

23.4.1 Switching from Embedded 32 kHz RC Oscillator to 32.768 kHz Crystal Oscillator

The sequence to switch from the embedded 32 kHz (typical) RC oscillator to the 32.768 kHz crystal oscillator is the following:

1. Switch the master clock to a source different from slow clock (PLL or Main Oscillator) through the Power Management Controller.
2. Switch from the embedded 32 kHz (typical) RC oscillator to the 32.768 kHz crystal oscillator by writing a 1 to the OSCSEL bit.
3. Wait 5 slow clock cycles for internal resynchronization.

23.4.2 Switching from 32.768 kHz Crystal Oscillator to Embedded 32 kHz RC Oscillator

The sequence to switch from the 32.768 kHz crystal oscillator to the embedded 32 kHz (typical) RC oscillator is the following:

1. Switch the master clock to a source different from slow clock (PLL or Main Oscillator).
2. Switch from the 32.768 kHz crystal oscillator to the embedded RC oscillator by writing a 0 to the OSCSEL bit.
3. Wait 5 slow clock cycles for internal resynchronization.

23.5 Slow Clock Controller (SCKC) User Interface

Table 23-1. Register Mapping

Offset	Register	Name	Access	Reset
0x0	Slow Clock Controller Configuration Register	SCKC_CR	Read/Write	0x0000_0001

23.5.1 Slow Clock Controller Configuration Register

Name: SCKC_CR

Address: 0xFC068650

Access: Read/Write

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
–	–	–	–	OSCSEL	–	–	–

- **OSCSEL: Slow Clock Selector**

0 (RC): Slow clock is the embedded 32 kHz (typical) RC oscillator.

1 (XTAL): Slow clock is the 32.768 kHz crystal oscillator.

24. Secure Fuse Controller (SFC)

24.1 Description

The Secure Fuse Controller (SFC) interfaces the system with electrical fuses in a secure way.

The default value of a fuse is logic ‘0’ (not programmed). A programmed fuse is logic ‘1’.

An electrical fuse matrix is a type of non-volatile memory. Each fuse in the matrix can be programmed only one time. They are typically used to store calibration bits for analog cells such as oscillators, configuration settings, chip identifiers or cryptographic keys.

A specific number of fuse bits are programmed by Atmel during the production tests through the test interface. The remaining 512 fuse bits are programmed by the user and by software through the user interface.

The SFC automatically reads the fuse values on start-up and stores them in 32-bit registers in order to make them accessible by the software. Only fuses set to level ‘1’ are programmed.

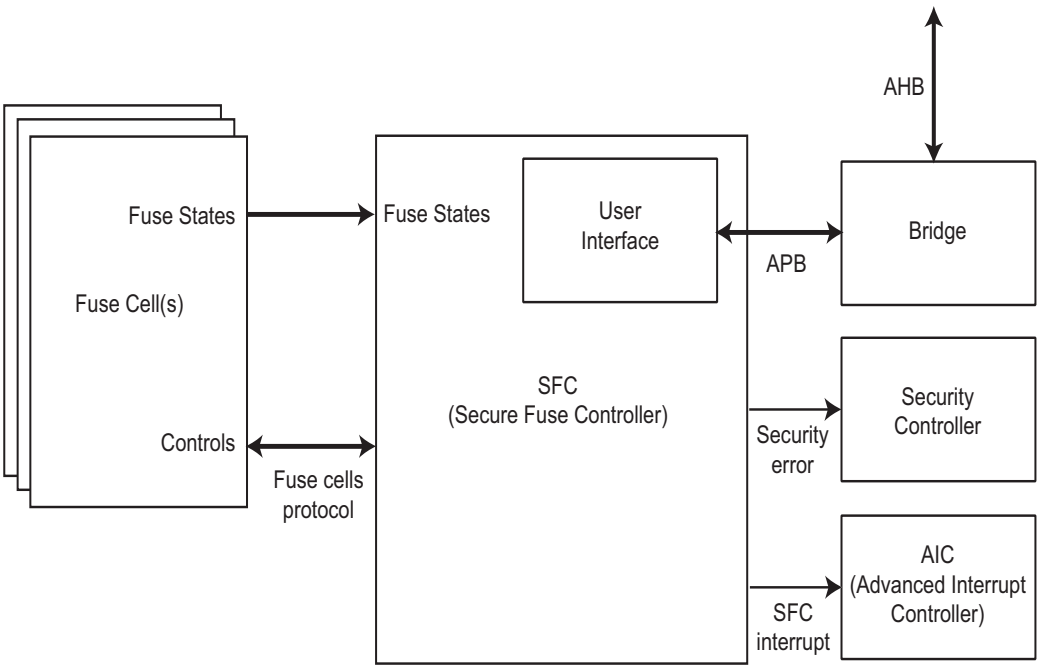
Several security mechanisms make irregular data recovery more complex to achieve.

24.2 Embedded Characteristics

- Fuse bits partitioned into two areas:
  - Atmel reserved area
  - 512-bit user area
- Program and read the fuse states by software
- Automatic check of programmed fuses
- Detection of irregular alteration of the fuse states in Atmel reserved area during start-up and report
- Part of fuse states maskable for reading

24.3 Block Diagram

Figure 24-1. SFC Block Diagram



24.4 Functional Description

24.4.1 Accessing the SFC

Setting the write-once FUSE bit in the SFR_SECURE register disables access to the Secure Fuse Controller (SFC).

24.4.2 Fuse Partitioning

The fuses are split into a user area of 512 bits and an Atmel reserved area.

The Atmel reserved area is typically used to store calibration bits for analog cells such as oscillators, configuration settings, chip identifiers, etc. The user area fuses are programmed later on by the user.

24.4.3 Fuse Integrity Checking

The SFC automatically reads the fuses values at start-up and stores them in 32-bit registers in order to make them accessible by software. At this time, the SFC checks the integrity of the fuse states in the Atmel reserved area.

If an inconsistency is detected, the CHECK error flag (named ACE for the Atmel reserved area) in the Status Register (SFC_SR) is set to ‘1’ and can trigger an interrupt. This flag is automatically cleared at ‘0’, when the Status Register (SFC_SR) is read.

24.4.4 Fuse Access

24.4.4.1 Fuse Reading

The fuse states are automatically latched at core start-up and are available for reading in the Data Registers (SFC_DRx).

The fuse states of bits 0 to 31 are available in the Data Register 0 (SFC_DR0), the fuse states of bits 32 to 63 are available in the Data Register 1 (SFC_DR1) and so on.

When fuse programming is performed, the fuse states are automatically updated in the Data Registers (SFC_DRx).

24.4.4.2 Fuse Programming

All the fuses can be written by software.

The sequence of instructions to program fuses is the following:

1. Write the key code 0xFB in the Key Register (SFC_KR).
2. Write the word to program in the corresponding Data Register (SFC_DRx).
   For example, if fuses 0 to 31 must be programmed, Data Register 0 (SFC_DR0) must be written. If fuses 32 to 61 must be programmed, Data Register 1 (SFC_DR1) must be written. Only the data bits set to level ‘1’ are programmed.
3. Wait for flag PGMC to rise in the Status Register (SFC_SR) by polling or interrupt.
4. Check the value of flag PGMF: if it is set to 1, it means that the programming procedure failed. After programming, the fuses are read back in the corresponding SFC_DRx.

24.4.4.3 Fuse Masking

It is possible to mask a fuse array. Once the fuse masking is enabled, the data registers from SFC_DRx to SFC_DRx are read at a value of ‘0’, regardless of the fuse state (the registers that are masked depend on the SFC hardware customizing).

To activate fuse masking, the MSK bit of the SFC Mode Register (SFC_MR) must be written to level ‘1’. The MSK bit is set-only. Only a hardware reset can disable fuse masking.

The MSK bit has no effect on the programming of masked fuses.

24.4.5 Fuse Functions

The “Fuse Box Controller” section defines the fuse bits that can be used as general purpose bits when standard boot is used.

If secure boot is used, refer to the device “Secure Boot Strategy” application note included in the Secure Package.

24.5 Secure Fuse Controller (SFC) User Interface

Table 24-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	SFC Key Register	SFC_KR	Write-only	–
0x04	SFC Mode Register	SFC_MR	Read/Write	0x0
0x08–0x0C	Reserved	–	–	–
0x10	SFC Interrupt Enable Register	SFC_IER	Write-only	–
0x14	SFC Interrupt Disable Register	SFC_IDR	Write-only	–
0x18	SFC Interrupt Mask Register	SFC_IMR	Read-only	0x0
0x1C	SFC Status Register	SFC_SR	Read-only	0x0
0x20	SFC Data Register 0	SFC_DR0	Read/Write	0x0
0x24	SFC Data Register 1	SFC_DR1	Read/Write	0x0
...	...	...	...	...
0x5C	SFC Data Register 15	SFC_DR15	Read/Write	0x0
0x60–0xFC	Reserved	–	–	–

24.5.1 SFC Key Register

Name: SFC_KR

Address: 0xFC060000

Access: Write-only

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
KEY							

- **KEY: Key Code**

This field must be written with the correct key code (0xFB) prior to any write in a Data Register (SFC_DRx) in order to enable the fuse programming. For each write of SFC_DRx, this field must be written immediately before.

24.5.2 SFC Mode Register

Name: SFC_MR

Address: 0xFC060004

Access: Read/Write

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
–	–	–	–	–	–	–	MSK

- **MSK: Mask Data Registers**

0: No effect

1: The data registers from SFC_DRx to SFC_DRx are always read at 0x00000000.

Note: The MSK bit is set-only. Only a hardware reset can disable fuse masking.

24.5.3 SFC Interrupt Enable Register

Name: SFC_IER

Address: 0xFC060010

Access: Write-only

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

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

0: No effect

1: Enables the corresponding interrupt

- **PGMC: Programming Sequence Completed Interrupt Enable**
- **PGMF: Programming Sequence Failed Interrupt Enable**
- **ACE: Atmel Check Error Interrupt Enable**

24.5.4 SFC Interrupt Disable Register

Name: SFC_IDR

Address: 0xFC060014

Access: Write-only

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

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

0: No effect

1: Disables the corresponding interrupt

- **PGMC: Programming Sequence Completed Interrupt Disable**
- **PGMF: Programming Sequence Failed Interrupt Disable**
- **ACE: Atmel Check Error Interrupt Disable**

24.5.5 SFC Interrupt Mask Register

Name: SFC_IMR

Address: 0xFC060018

Access: Read-only

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

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

0: Corresponding interrupt is not enabled.

1: Corresponding interrupt is enabled.

- **PGMC: Programming Sequence Completed Interrupt Mask**
- **PGMF: Programming Sequence Failed Interrupt Mask**
- **ACE: Atmel Check Error Interrupt Mask**

24.5.6 SFC Status Register

Name: SFC_SR

Address: 0xFC06001C

Access: Read-only

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

- **PGMC: Programming Sequence Completed (cleared on read)**

0: No programming sequence completion since the last read of SFC_SR.

1: At least one programming sequence completion since the last read of SFC_SR.

- **PGMF: Programming Sequence Failed (cleared on read)**

0: No programming failure occurred during last programming sequence since the last read of SFC_SR.

1: A programming failure occurred since the last read of SFC_SR.

- **APLE: Atmel Programming Lock Error (cleared on read)**

0: No programming attempt has been made in the Atmel locked area since the last read of SFC_SR.

1: A programming attempt has been made in the Atmel locked area since the last read of SFC_SR.

- **ACE: Atmel Check Error (cleared on read)**

0: No check error in the Atmel reserved area since the last read of SFC_SR.

1: At least one check error in the Atmel reserved area since the last read of SFC_SR.

24.5.7 SFC Data Register x

Name: SFC_DRx [x=0..15]

Address: 0xFC060020

Access: Read/Write

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

- **DATA: Fuse Data**

READ: Reports the state of the corresponding fuses.

WRITE: The data to be programmed in the corresponding fuses. Only bits with a value of ‘1’ are programmed. Writing this register automatically triggers a programming sequence of the corresponding fuses. Note that a write to the Key Register (SFC_KR) with the correct key code must always precede any write to SFC_DRx.

25. Clock Generator

25.1 Description

The Clock Generator User Interface is embedded within the Power Management Controller and is described in Section 26.19 “Power Management Controller (PMC) User Interface”. 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
- A low-power embedded 64 kHz (typical) RC oscillator generating the 32 kHz source clock
- A 12 MHz crystal oscillator or a 12 to 48 MHz XRCGB crystal resonator with Bypass mode
- A 12 (±1%) MHz RC oscillator
- A 480 MHz UTMI PLL providing a clock for the USB High-speed Device Controller
- A 600 to 1200 MHz programmable PLL (input from 8 to 50 MHz), provides the clock to the processor and to the peripherals

The Clock Generator provides the following clocks:

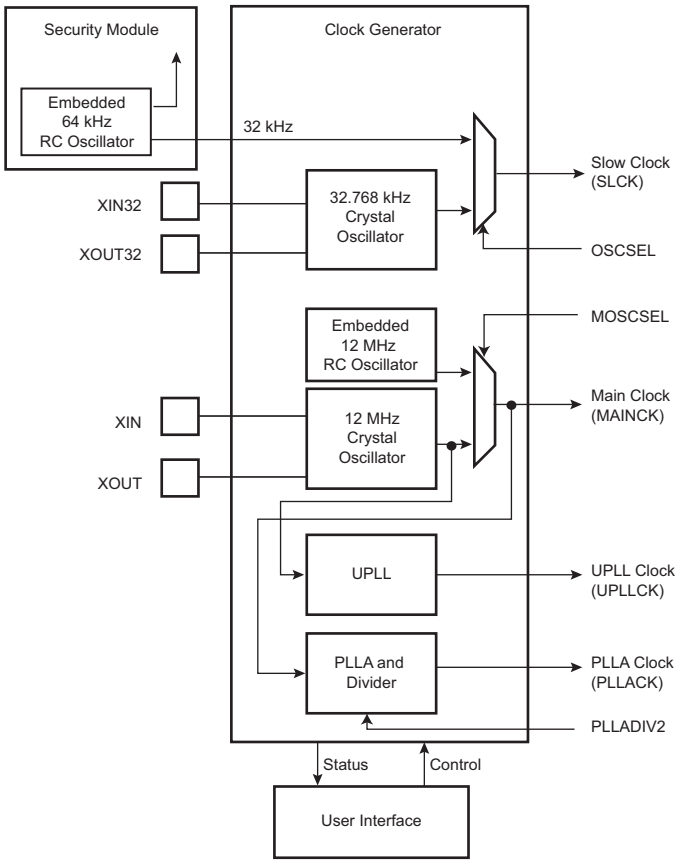
- SLCK—Slow clock. The only permanent clock within the system.
- MAINCK—Output of the Main clock oscillator selection: either 12 MHz crystal oscillator or 12 MHz RC oscillator
- PLLACK—Output of the divider and the 600 to 1200 MHz programmable PLL (PLLA)
- UPLLCK—Output of the 480 MHz UTMI PLL (UPLL)
- SMDCK—Software Modem clock

The Power Management Controller also provides the following operations on clocks:

- 12 MHz crystal oscillator clock failure detector
- 32.768 kHz crystal oscillator frequency monitor
- Frequency counter on Main clock and an on-the-fly adjustable 12 MHz RC oscillator frequency

25.3 Block Diagram

Figure 25-1. Clock Generator Block Diagram



The embedded 64 kHz RC oscillator is always enabled when VDDBU is powered. The 12 MHz RC oscillator is always enabled when VDDCORE is powered.

25.4 Slow Clock

The Slow Clock Controller embeds a Slow clock generator that is supplied with the VDDBU power supply. As soon as VDDBU is supplied, both the 32.768 kHz crystal oscillator and the embedded 64 kHz (typical) RC oscillator are powered, but only the RC oscillator is enabled.

The Slow clock is generated either by the 32.768 kHz crystal oscillator or by the embedded 64 kHz (typical) RC oscillator divided by two.

The selection of the Slow clock source is made via the OSCSEL bit in the Slow Clock Controller Configuration register (SCKC_CR).

SCKC_CR.OSCSEL and PMC_SR.OSCSELS report which oscillator is selected as the Slow clock source. PMC_SR.OSCSELS informs when the switch sequence initiated by a new value written in SCKC_CR.OSCSEL is done.

25.4.1 Embedded 64 kHz (typical) RC Oscillator

By default, the embedded 64 kHz (typical) RC oscillator is enabled and selected as a source of SLCK. The user has to take into account the possible drifts of this oscillator. Refer to Section 55.2 “DC Characteristics”.

25.4.2 32.768 kHz Crystal Oscillator

The Clock Generator integrates a low-power 32.768 kHz crystal oscillator. To use this oscillator, the XIN32 and XOUT32 pins must be connected to a 32.768 kHz crystal. Refer to Section 55. “Electrical Characteristics” for appropriate loading capacitor selection on XIN32 and XOUT32.

Note that the user is not obliged to use the 32.768 kHz crystal oscillator and can use the 64 kHz (typical) RC oscillator instead.

The 32.768 kHz crystal oscillator provides a more accurate frequency than the 64 kHz (typical) RC oscillator.

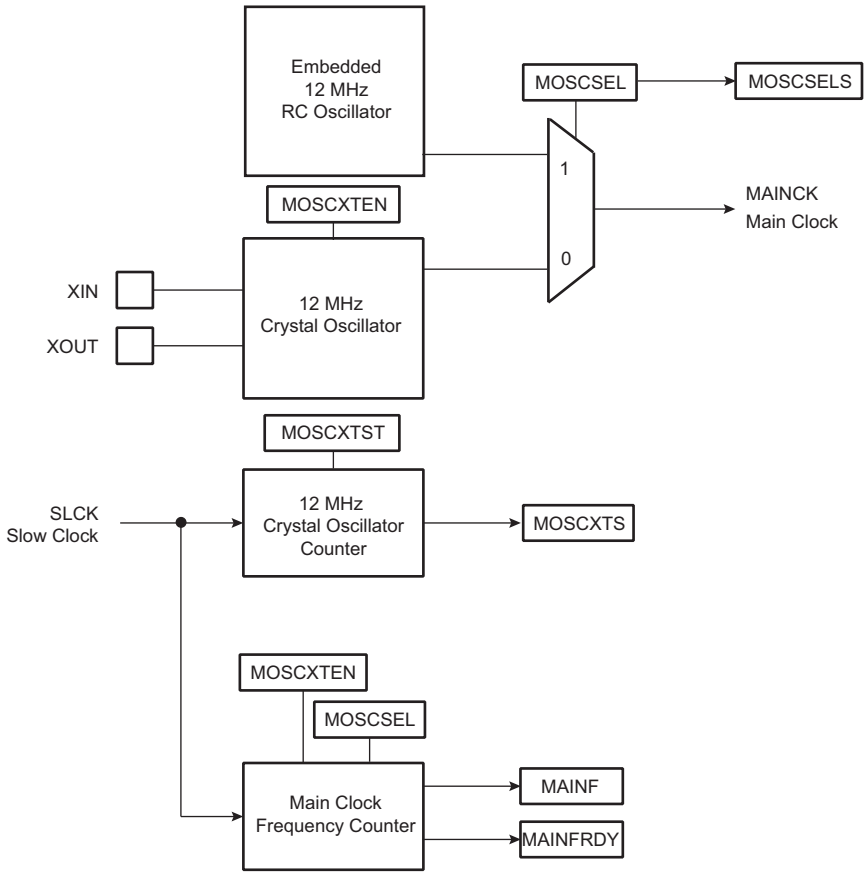
To select the 32.768 kHz crystal oscillator as the source of the Slow clock, the bit SCKC_CR.OSCSEL must be set. This results in a sequence which enables the 32.768 kHz crystal oscillator. The switch of the Slow clock source is glitch-free.

25.5 Main Clock

The Main clock has two sources:

- a 12 MHz RC oscillator which is always enabled
- a 12 MHz crystal oscillator with Bypass mode

Figure 25-2. Main Clock Block Diagram



25.5.1 12 MHz RC Oscillator

After reset, the 12 MHz RC oscillator is enabled and selected as the source of MAINCK and MCK. MCK is the default clock selected to start up the system.

Refer to Section 55.2 “DC Characteristics”.

25.5.2 12 MHz Crystal Oscillator

After reset, the 12 MHz crystal oscillator is disabled and is not selected as the source of MAINCK.

As the source of MAINCK, the 12 MHz crystal oscillator provides an accurate 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 bit, the PMC_SR.MOSCXTS bit is automatically cleared, indicating the 12 MHz crystal oscillator is off.

When enabling this oscillator, the user must initiate the startup time counter. This startup time depends on the characteristics of the external device connected to this oscillator. Refer to Section 55. “Electrical Characteristics” for startup time.

When CKGR_MOR.MOSCXTEN and CKGR_MOR.MOSCXTST are written to enable this oscillator, the PMC_SR.MOSCXTS bit is cleared and the counter starts counting down on the Slow clock divided by 8 from the MOSCXTST value. When the counter reaches 0, the PMC_SR.MOSCXTS is set, indicating that the 12 MHz crystal oscillator is stabilized. Setting MOSCXTS in the PMC Interrupt Mask register (PMC_IMR) triggers an interrupt to the processor.

25.5.3 Main Clock Source Selection

The source of the Main clock can be selected from the following:

- embedded 12 MHz RC oscillator
- 12 MHz crystal oscillator
- an XRCGB crystal resonator

The advantage of the Main RC oscillator is its fast startup 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 is made by writing CKGR_MOR.MOSCSEL. The switch of the Main clock source is glitch-free, so there is no need to run out of SLCK or PLLACK in order to change the selection. PMC_SR.MOSCSELS indicates when the switch sequence is done.

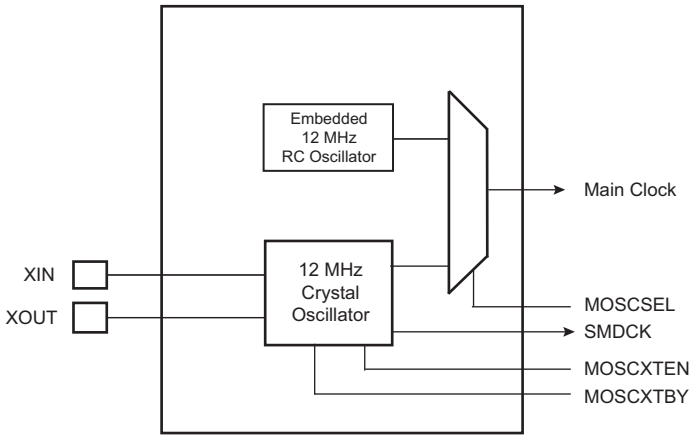
Setting PMC_IMR.MOSCSELS triggers an interrupt to the processor.

The 12 MHz crystal oscillator can be bypassed by setting the CKGR_MOR.MOSCXTBY to accept an external Main clock on XIN (refer to Section 25.5.4 “Bypassing the 12 MHz Crystal Oscillator”).

MOSCSEL, MOSCXTEN and MOSCXTBY bits are located in the PMC Clock Generator Main Oscillator Register (CKGR_MOR).

After a VDDBU power-on reset, the default configuration is MOSCXTEN = 0 and MOSCSEL = 0, allowing the 12 MHz RC oscillator to start as Main clock.

Figure 25-3. Main Clock Source Selection



25.5.4 Bypassing the 12 MHz Crystal Oscillator

Prior to bypassing the 12 MHz crystal oscillator, the external clock frequency provided on the XIN pin must be stable and within the values specified in the XIN clock characteristics. Refer to Section 55. “Electrical Characteristics”.

The sequence to bypass the crystal oscillator is the following:

1. Ensure that an external clock is connected on XIN.
2. Enable the bypass by setting CKGR_MOR.MOSCXTBY.
3. Disable the 12 MHz crystal oscillator by clearing CKGR_MOR.MOSCXTEN.

25.5.5 Main Frequency Counter

The main frequency counter measures the Main RC oscillator against the SLCK and is managed by CKGR_MCFR.

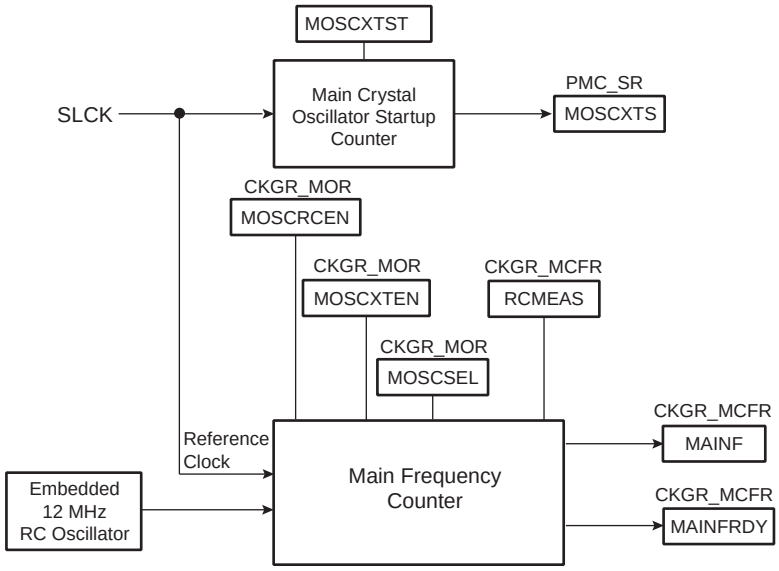
During the measurement period, the main frequency counter increments at the speed of the Main RC oscillator.

A measurement is started in the following cases:

- When CKGR_MCFR.RCMEAS is written to ‘1’.
- When the 12 MHz RC oscillator is selected as the source of the Main clock
- When the 12 MHz crystal oscillator is selected as the source of the Main clock and when this oscillator becomes stable (i.e., when the MOSCXTS bit is set)
- When the Main clock source selection is modified

The measurement period ends at the 16th falling edge of the Slow clock, CKGR_MCFR.MAINFRDY is set and the counter stops counting. Its value can be read in the CKGR_MCFR.MAINF and gives the number of Main clock cycles during 16 periods of Slow clock, so that the frequency of the 12 MHz RC oscillator or the crystal oscillator can be determined.

Figure 25-4. Main Frequency Counter Block Diagram



25.5.6 Switching Main Clock Between the RC Oscillator and the Crystal Oscillator

When switching the source of the Main clock between the RC oscillator and the crystal oscillator, both oscillators must be enabled. After completion of the switch, the unused oscillator can be disabled.

If switching to the crystal oscillator, a check must be carried out to ensure that the oscillator is present and that its frequency is valid. Follow the sequence below:

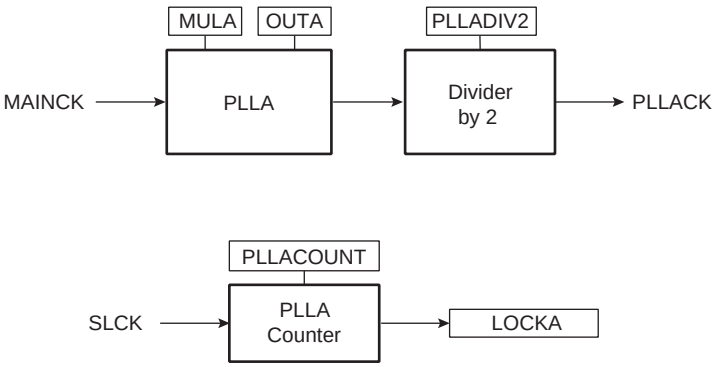
1. Select the Slow clock as MCK by configuring bit CSS = 0 in the Master Clock register (PMC_MCKR)).
2. Wait for PMC_SR.MCKRDY flag to rise.
3. Enable the crystal oscillator by setting CKGR_MOR.MOSCXTEN. Configure the CKGR_MOR. MOSCXTST field with the crystal oscillator startup time as defined in the section “Electrical Characteristics”.
4. Wait for PMC_SR.MOSCXTS flag to rise, indicating the end of a startup period of the crystal oscillator.
5. Select the crystal oscillator as the source of the Main clock by setting CKGR_MOR.MOSCSEL.
6. Read CKGR_MOR.MOSCSEL until its value equals 1.
7. Check the status of PMC_SR.MOSCSELS flag:
   - If MOSCSELS = 1: There is a crystal oscillator connected.
     1. Initiate a new frequency measurement by setting CKGR_MCFR.RCMEAS.
     2. Read CKGR_MCFR.MAINFRDY unit its value equals 1.
     3. Read CKGR_MCFR.MAINF and compute the value of the crystal frequency.
     4. If the MAINF value is valid, the Main clock can be switched to the crystal oscillator.
   - If MOSCSELS = 0: There is no crystal oscillator connected or the crystal oscillator is out of specification.
     5. Select the RC oscillator as the source of the Main clock by clearing CKGR_MOR.MOSCSEL.

25.6 Divider and PLLA Block

The PLLA embeds an input divider to increase the accuracy of the resulting clock signals. However, the user must respect the PLLA minimum input frequency when programming the divider.

Figure 25-5 shows the block diagram of the divider and PLLA block.

Figure 25-5. Divider and PLLA Block Diagram



Whenever the PLLA is re-enabled or one of its parameters is changed, PMC_SR.LOCKA is automatically cleared. The values written in the PLLACOUNT field in the Clock Generator PLLA register (CKGR_PLLAR) are loaded in the PLLA counter. The PLLA counter then decrements at the speed of the Slow clock until it reaches 0. At this time, PMC_SR.LOCKA is set and can trigger an interrupt to the processor. The user has to load the number of Slow clock cycles required to cover the PLLA transient time into CKGR_PLLACOUNT.

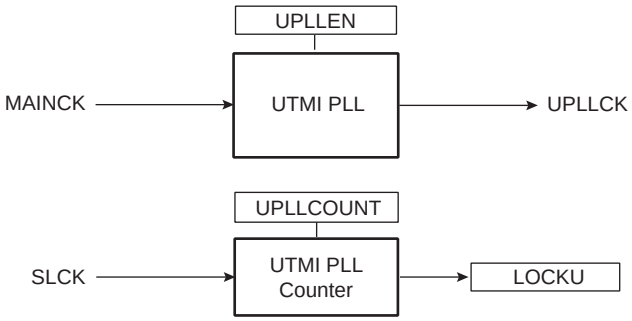
The PLLA clock can be divided by 2 by writing the PMC_MCKR.PLLADIV2.

25.7 UTMI PLL Clock

The source of the UTMI PLL (UPLL) is the Main clock (MAINCK). MAINCK must select the Main crystal oscillator to meet the frequency accuracy required by USB.

A frequency of 12 MHz is required for the built-in UTMI PLL multiplier of x 40 to obtain the USB High-speed 480 MHz.

Figure 25-6. UTMI PLL Block Diagram



Whenever the UTMI PLL is enabled by writing UPLLEN in the UTMI Clock register (CKGR_UCKR), PMC_SR.LOCKU is automatically cleared. The values written in CKGR_UCKR.UPLLCOUNT are loaded in the UTMI PLL counter. The UTMI PLL counter then decrements at the speed of the Slow clock divided by 8 until it reaches 0. At this time, the PMC_SR.LOCKU is set in and can trigger an interrupt to the processor. The user has to load the number of Slow clock cycles required to cover the UTMI PLL transient time into CKGR_UCKR.UPLLCOUNT.

26. Power Management Controller (PMC)

26.1 Description

The Power Management Controller (PMC) optimizes power consumption by controlling all system and user peripheral clocks. The PMC enables/disables the clock inputs to many of the peripherals and the Core.

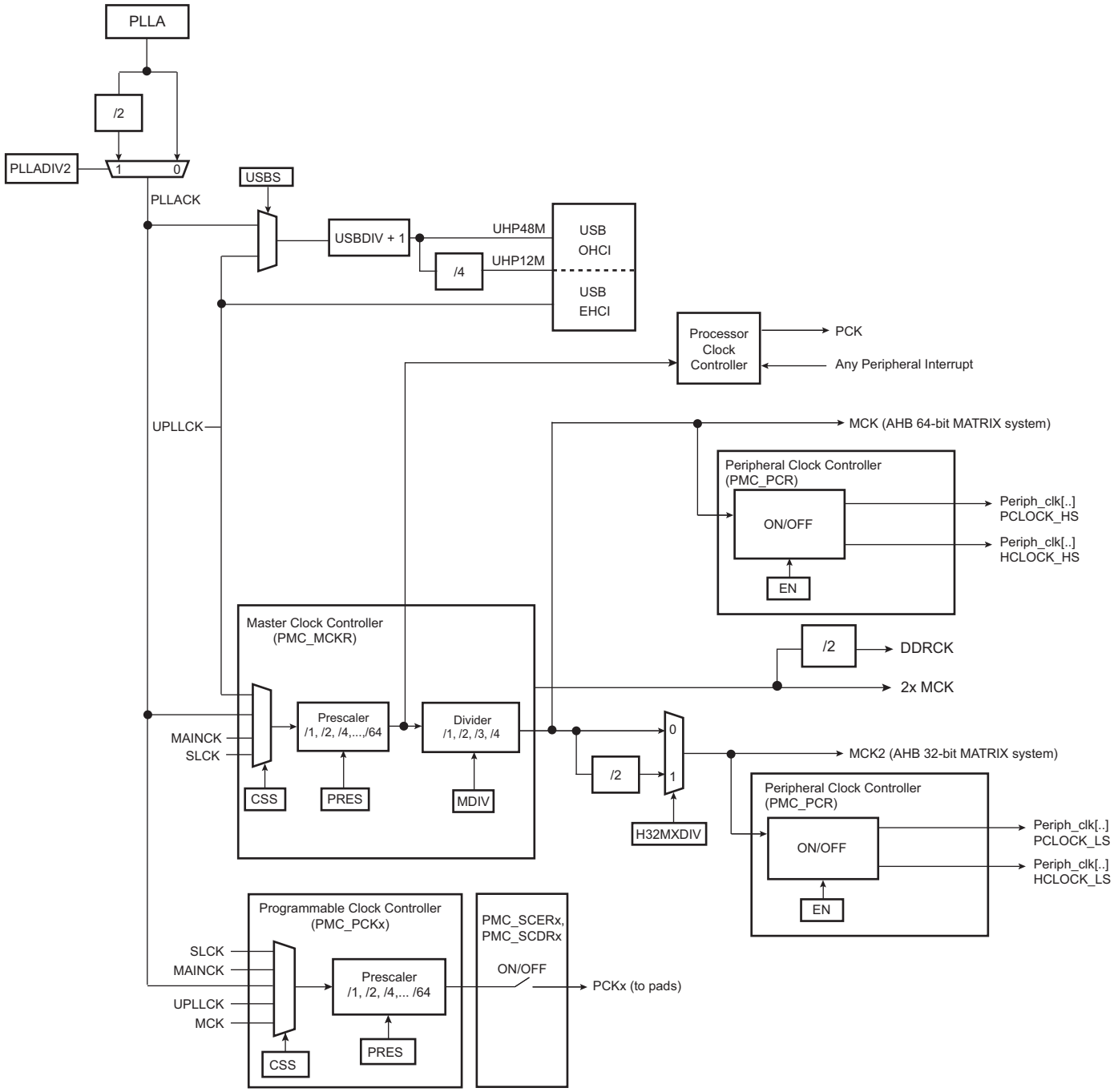
26.2 Embedded Characteristics

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 (PCK)—must be switched off when processor is entering Idle mode
- HS USB Device Clock (UDPCK)
- Software Modem Clock (SMDCK)
- H64MX Matrix Clock (MCK) and H32MX Matrix Clock (MCK or MCK/2)
- Peripheral Clocks—provided to the embedded peripherals and independently controllable
- Programmable Clock outputs can be selected from the clocks provided by the clock generator and driven on the PCKx pins.

26.3 Block Diagram

Figure 26-1. General Clock Block Diagram



26.4 Master Clock Controller

The Master Clock Controller provides selection and division of the Master clock (MCK). MCK is the source clock of the peripheral clocks.

The Master clock is selected from one of the clocks provided by the Clock Generator. Selecting the Slow clock provides a Slow clock signal to the whole device. Selecting the Main clock saves power consumption of the PLLs.

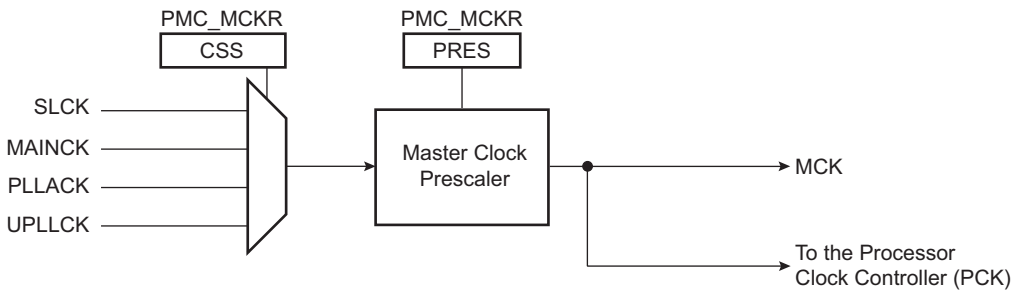
The Master Clock Controller is made up of a clock selector and a prescaler. It also contains a Master clock divider which allows the processor clock to be faster than the Master clock.

The Master clock selection is made by writing the CSS (Clock Source Selection) field in the Master Clock register (PMC_MCKR). The prescaler supports the division by a power of 2 of the selected clock between 1 and 64, and the division by 6. PMC_MCKR.PRES programs the prescaler.

Note: It is forbidden to modify MDIV and CSS at the same access. Each field must be modified separately with a wait for MCKRDY flag between the first field modification and the second field modification.

Each time PMC_MCKR is written to define a new Master clock, PMC_SR.MCKRDY is cleared. It reads 0 until the Master clock 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 actually done.

Figure 26-2. Master Clock Controller



26.5 Processor Clock Controller

The PMC features a Processor Clock (PCK) Controller that implements the processor Idle mode.

The Processor clock can be disabled by executing the WFI (WaitForInterrupt) processor instruction.

The Processor clock can be disabled by writing the PMC System Clock Disable Register (PMC_SCDR). The status of this clock (at least for debug purposes) can be read in the PMC System Clock Status Register (PMC_SCSR).

The Processor clock is enabled after a reset and is automatically re-enabled by any enabled interrupt. The processor Idle mode is entered by disabling the Processor clock, which is automatically re-enabled by any enabled fast or normal interrupt, or by the reset of the product.

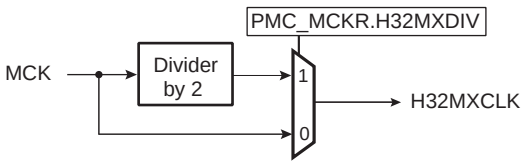
When processor Idle mode is entered, the current instruction is finished before the clock is stopped, but this does not prevent data transfers from other masters of the system bus.

26.6 Matrix Clock Controller

The AXI Matrix and H64MX 64-bit Matrix clocks are MCK.

The H32MX 32-bit matrix clock is to be configured as MCK if MCK does not exceed 100 MHz (refer to “Section 55.4.2 “Master Clock Characteristics”); otherwise, this clock is to be configured as MCK/2. Selection is done with the H32MXDIV bit in PMC Master Clock Register.

Figure 26-3. H32MX 32-bit Matrix Clock Configuration



26.7 Peripheral Clock Controller

The PMC controls the clocks of each embedded peripheral by means of the Peripheral Clock Controller. The user can individually enable and disable the clock on the peripherals and select a division factor from MCK. This is done in the Peripheral Control register (PMC_PCR).

In order to reduce power consumption, the division factor can be 1, 2, 4 or 8.

The divisor is defined in PMC_PCR. To apply a division factor, PID, CMD and DIV must be written in a single operation. The target peripheral clock is defined by the PID field. The divisor value is defined by DIV and the bit CMD must be set. To read the current division factor associated with a peripheral clock, two separate operations must be performed:

1. Write a zero to the CMD bit and configure PID for the target peripheral clock. DIV is not significant for this operation.
2. Read PMC_PCR. The value of DIV is the divisor applied to the peripheral clock defined by PID.

When a peripheral clock is disabled, the clock is immediately stopped. The peripheral clocks are automatically disabled after a reset.

In order to stop a peripheral, 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 value written in the PID field in PMC_PCR is the Peripheral Identifier defined at the product level (refer to Section 8.2 “Peripheral Identifiers”). Generally, the field value corresponds to the interrupt source number assigned to the peripheral.

26.8 Programmable Clock Controller

The PMC controls three signals to be outputs on external pins PCKx. Each signal can be independently programmed via the Programmable Clock registers (PMC_PCKx).

PCKx can be independently selected between the Slow clock (SLCK), the Master clock (MAINCK), the PLLACK, the UTMI PLL output and the Main clock by writing PMC_PCKx.CSS. Each output signal can also be divided by a power of 2 between 1 and 64 by writing PMC_PCKx.PRES.

Each output signal can be enabled and disabled by writing a ‘1’ in the corresponding bits, PMC_SCER.PCKx and PMC_SCDR.PCKx, respectively. The status of each active programmable output clocks is given in PMC_SCSR.PCKx.

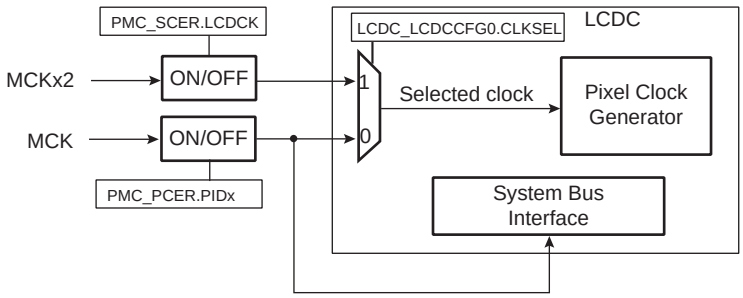
The status flag PMC_SR.PCKRDYx indicates that the Programmable clock programmed in PMC_PCKx is ready.

As the Programmable Clock Controller does not implement glitch prevention when switching clocks, it is strongly recommended to disable the Programmable clock before any configuration change and to re-enable it after the change is performed.

26.9 LCDC Clock Controller

In order to have more flexibility on the pixel clock, the LCDC can use MCK, or MCKx2 if LCDCK is set in the PMC System Clock Enable Register (PMC_SCER).

Figure 26-4. LCDCLK Clock Configuration



26.10 USB Device and Host Clocks

The USB Device and Host High Speed ports (UDPHS and UHPHS) clocks are enabled by the corresponding PIDx bits in the Peripheral Clock Enable register (PMC_PCERx). To save power on this peripheral when they are not used, the user can set these bits in the Peripheral Clock Disable register (PMC_PCDRx). Corresponding PIDx bits in the Peripheral Clock Status register (PMC_PCSRx) give the status of these clocks.

The PMC also provides the clocks UHP48M and UHP12M to the USB Host OHCI. The USB Host OHCI clocks are controlled by PMC_SCER.UHP. To save power on this peripheral when they are not used, the user can set PMC_SCDR.UHP. PMC_SCSR.UHP gives the status of this clock. The USB host OHCI requires both the 12/48 MHz signal and the Master clock. The USBDIV field in the USB Clock register (PMC_USB) is to be programmed to 9 (division by 10) for normal operations.

To further reduce power consumption the user can stop the UTMI PLL. In this case USB high-speed operations are not possible. Nevertheless, as the USB OHCI Input clock can be selected with PMC_USB.USBS (PLLA or UTMI PLL), OHCI full-speed operation remains possible.

The user must program the USB OHCI Input clock and the USBDIV divider in the PMC_USB register to generate a 48 MHz and a 12 MHz signal with an accuracy of ±0.25%.

26.11 DDR2/LPDDR/LPDDR2 Clock Controller

The PMC controls the clocks of the DDR memory.

The DDR clock can be enabled and disabled with the DDRCK bit in PMC_SCER and PMC_SDER, respectively. At reset, the DDR clock is disabled to reduce power consumption.

If PMC_MCKR.MDIV = 0 (PCK = MCK), the DDR clock is not available.

To reduce PLLA power consumption, the user can choose UPLLCK as an input clock for the system. In this case, the DDR Controller can drive LPDDR or LPDDR2 at up to 120 MHz.

26.12 Software Modem Clock Controller

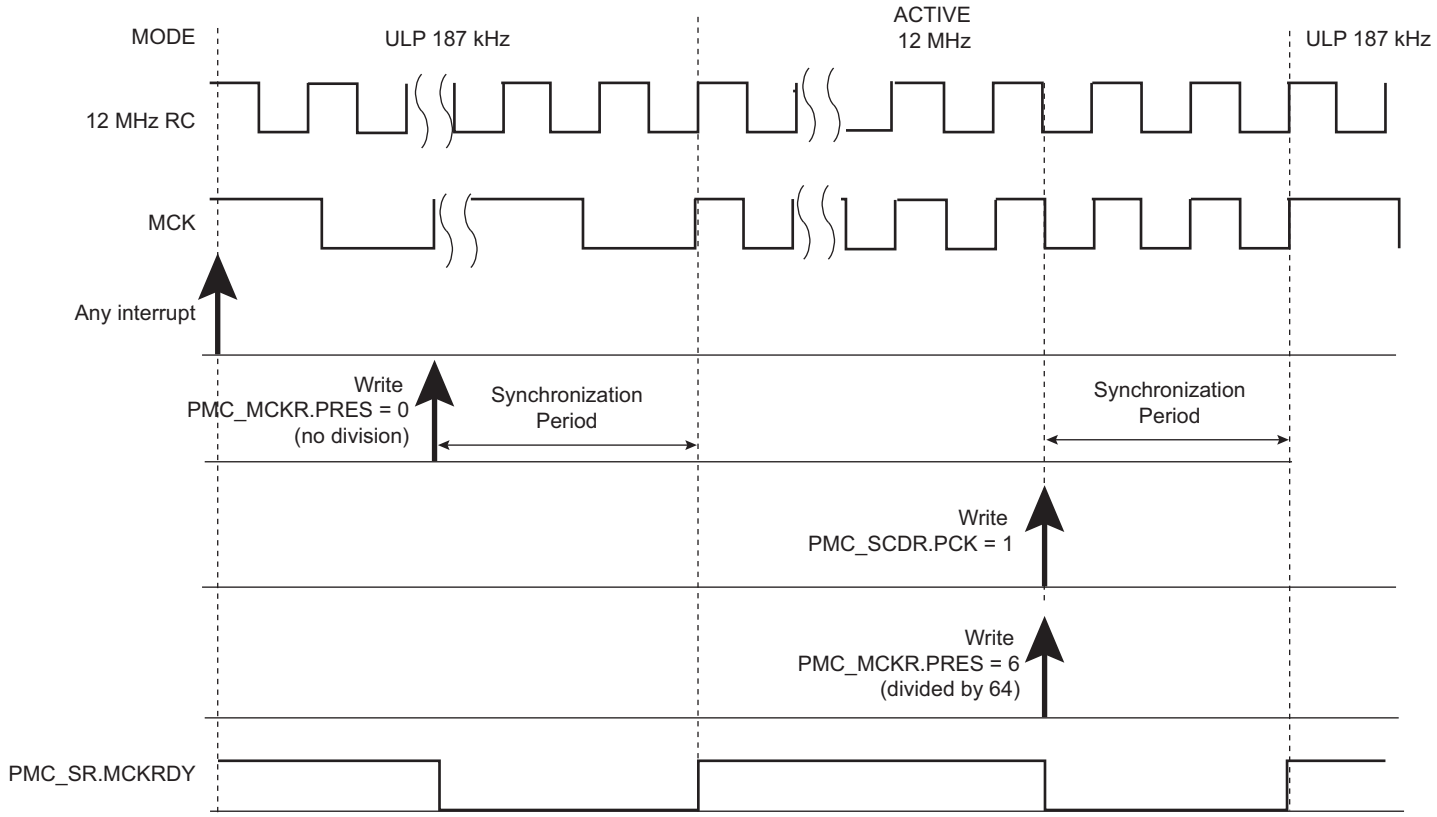
The PMC controls the clocks of the Software Modem.

SMDCK is a division of UPLL or PLLA.

26.13 Fast Startup from Ultra-low-power (ULP) Mode

In Ultra-low-power (ULP) mode, the Main clock (MAINCK) must be running, thus either the 12 MHz crystal oscillator or the Fast RC oscillator must be enabled. The lowest power consumption that can be achieved in ULP Mode, can be obtained when dividing the selected oscillator frequency by 64 by writing PMC_MCKR.PRES to 6. Any interrupt exits the system from ULP Mode. The software must write PMC_MCKR.PRES to 1 to provide MCK with the fastest clock. If the PLL is used, the startup procedure must be done prior to writing PMC_MCKR.PRES to 1. Figure 26-5 illustrates an example of startup phase from ULP Mode without use of PLL.

Figure 26-5. Fast Startup from Ultra-low-power Mode



Warning: The duration of the WKUPx pins active level must be greater than four MAINCK cycles.

26.14 Main Crystal Oscillator Failure Detection

The Main crystal oscillator failure detector monitors the 12 MHz crystal oscillator or ceramic resonator-based oscillator to identify a possible failure of this oscillator.

The clock failure detector can be enabled or disabled by configuring CKGR_MOR.CFDEN. The detector is also disabled in either of the following cases:

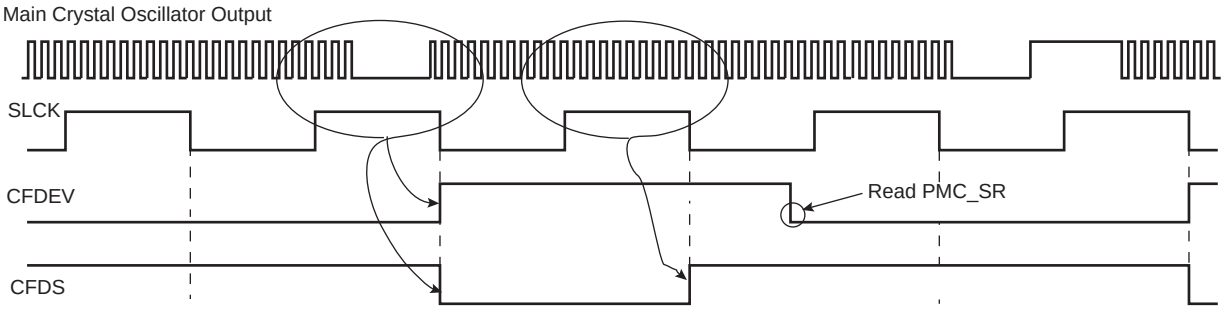
- after a VDDCORE reset
- when the oscillator is disabled (CKGR_MOR.MOSCXTEN = 0)

A failure is detected by means of a counter incrementing on the main oscillator clock edge and detection logic is triggered by the 32 kHz generated by the 64 kHz (typical) RC oscillator. This oscillator is automatically enabled when CKGR_MOR.CFDEN = 1.

The counter is cleared when the 32 kHz generated by the 64 kHz (typical) RC oscillator clock signal is low, and enabled when the signal is high. Thus, the failure detection time is one RC oscillator period. If, during the high level period of the 32 kHz generated by the 64 kHz (typical) RC oscillator clock signal, less than eight 12 MHz crystal oscillator clock periods have been counted, then a failure is reported.

If a failure of the Main clock is detected, bit PMC_SR.CFDEV indicates a failure event and generates an interrupt if the corresponding interrupt source is enabled. The interrupt remains active until a read occurs in PMC_SR. The user can know the status of the clock failure detection at any time by reading bit PMC_SR.CFDS.

Figure 26-6. Clock Failure Detection (Example)



Note: Ratio of clock periods is for illustration purposes only.

If the 12 MHz crystal oscillator or ceramic resonator-based oscillator is selected as the source clock of MAINCK (CKGR_MOR.MOSCSEL = 1), and if MCK source is PLLACK or UPLLCK (PMC_MCKR.CSS = 2 or 3), a clock failure detection automatically forces MAINCK to be the source clock for the Master clock (MCK). Then, regardless of the PMC configuration, a clock failure detection automatically forces the 12 MHz RC oscillator to be the source clock for MAINCK. If this oscillator is disabled when a clock failure detection occurs, it is automatically re-enabled by the clock failure detection mechanism.

It takes two 32 kHz (typical) clock cycles to detect and switch from the 12 MHz crystal oscillator to the 12 MHz RC oscillator if the source Master clock (MCK) is Main clock (MAINCK), or three 32 kHz (typical) cycles if the source of MCK is PLLACK or UPLLCK.

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.

The user can know the status of the clock failure detector at any time by reading bit PMC_SR.FOS.

This fault output remains active until the defect is detected and until it is cleared by the bit FOCLR in the PMC Fault Output Clear register (PMC_FOCR).

26.15 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 12 MHz RC oscillator known as a reliable clock source. This function is enabled by configuring the XT32KFME bit of CKGR_MOR.

The 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 Slow clock frequency monitoring is disabled.

The monitored clock frequency is declared invalid if at least four consecutive clock period measurement results are over the nominal period ±10%.

Due to the possible frequency variation of the embedded 12 MHz RC oscillator acting as reference clock for the monitor logic, any Slow clock 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 a 32.768 kHz frequency ensures that the error flag is not asserted. The permitted drift of the crystal is 10000 ppm (1%), which allows any standard crystal to be used.

The error flag can be defined as an interrupt source of the PMC by setting PMC_IER.XT32KERR.

26.16 Programming Sequence

1. If the 12 MHz crystal oscillator is not required, PLL can be directly configured (begin with Step 6. or Step 7.) else this oscillator must be started (begin with Step 2.).
2. Enable the 12 MHz crystal oscillator by setting CKGR_MOR.MOSCXTEN. The user can define a startup time. This can be achieved by writing a value in 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 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 the MAINCK to the 12 MHz crystal oscillator by setting CKGR_MOR.MOSCSEL.
4. Wait for PMC_SR.MOSCSELS to be set to ensure the switchover is complete.
5. Check the Main clock frequency:

   The Main clock frequency can be measured via CKGR_MCFR.

   Read CKGR_MCFR until the MAINFRDY field is set, after which the user can read the field 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 Slow clock cycles.

   If MAINF = 0, switch the MAINCK to the 12 MHz RC oscillator by clearing CKGR_MOR.MOSCSEL. If MAINF ≠ 0, proceed to Step 6.
6. Set the PLLA and divider (if not required, proceed to Step 7.)

   All parameters needed to configure PLLA and the divider are located in CKGR_PLLAR.

   The MULA field is the PLLA multiplier factor. This parameter can be programmed between 0 and 127. If MULA is cleared, PLLA is turned off, otherwise the PLLA output frequency is PLLA input frequency multiplied by (MULA + 1).

   The PLLACOUNT field specifies the number of Slow clock cycles before LOCKA bit is set in PMC_SR after CKGR_PLLAR has been written.

   Once CKGR_PLLAR has been written, the user must wait for the LOCKA bit to be set in PMC_SR. This can be done either by polling LOCKA in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (LOCKA) has been enabled in PMC_IER. All parameters in CKGR_PLLAR can be programmed in a single write operation. If at some stage parameter MULA is modified, 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. Set Bias and High-speed PLL (UPLL) for UTMI

   The UTMI PLL is enabled by setting CKGR_UCKR.UPLLEN. The UTMI Bias must be enabled by setting CKGR_UCKR.BIASEN at the same time. In some cases, it may be preferable to define a startup time. This can be achieved by writing a value in CKGR_UCKR.PLLCOUNT.

   Once this register has been correctly configured, the user must wait for PMC_SR.LOCKU to be set. This can be done either by polling LOCKU in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (LOCKU) has been enabled in PMC_IER.
8. Select Master Clock and Processor Clock

   The Master clock and the Processor clock are configurable via PMC_MCKR.

   The CSS field is used to select the clock source of the Master clock and Processor clock dividers. By default, the selected clock source is the Main clock.

   The PRES field is used to define the Processor clock and Master clock prescaler. The user can choose between different values 1, 2, 4, 8, 16, 32, 64). Prescaler output is the selected clock source frequency divided by the PRES value.

The MDIV field is used to define the Master clock divider. It is possible to choose between different values (0, 1, 2, 3). The Master clock output is Processor clock frequency divided by 1, 2, 3 or 4, depending on the value programmed in MDIV.

The PMC PLLA clock input can be divided by 2 by writing the PLLADIV2 bit.

By default, MDIV and PLLLADIV2 are cleared, which indicates that Processor clock is equal to the Master clock.

Once PMC_MCKR has been written, the user must wait for the MCKRDY bit to be set in PMC_SR. This can be done either by polling MCKRDY in PMC_SR 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 the following:

If a new value for CSS field corresponds to PLL clock,

1. Program PMC_MCKR.PRES.
2. Wait for PMC_SR.MCKRDY to be set.
3. Program PMC_MCKR.MDIV.
4. Wait for PMC_SR.MCKRDY to be set.
5. Program PMC_MCKR.CSS.
6. Wait for PMC_SR.MCKRDY to be set.

If a new value for CSS field corresponds to Main clock or Slow clock,

1. Program PMC_MCKR.CSS.
2. Wait for PMC_SR.MCKRDY to be set.
3. Program PMC_MCKR.PRES.
4. Wait for PMC_SR.MCKRDY to be set.

If CSS, MDIV or PRES are modified at some stage, the MCKRDY bit goes low to indicate that the Master clock and the Processor clock are not yet ready. The user must wait for the MCKRDY bit to be set again before using the Master and Processor clocks.

Note: If PLLA clock was selected as the Master clock and the user decides to modify it by writing in CKGR_PLLR, the MCKRDY flag goes low while PLL is unlocked. Once PLL is locked again, LOCKA goes high and MCKRDY is set. While PLL is unlocked, the Master clock selection is automatically changed to Slow clock. For further information, see Section 26.17.2 “Clock Switching Waveforms”.

Code Example:

```
write_register(PMC_MCKR,0x00000001)
wait (MCKRDY=1)
write_register(PMC_MCKR,0x00000011)
wait (MCKRDY=1)
```

The Master clock is Main clock divided by 2.

The Processor clock is the Master clock.

9. Select Programmable Clocks

Programmable clocks can be enabled and/or disabled via PMC_SCER and PMC_SCDR. Three programmable clocks can be used. PMC_SCSR indicates which programmable clock is enabled. By default all programmable clocks are disabled.

PMC_PCKx registers are used to configure programmable clocks.

The PMC_PCKx.CSS field selects the programmable clock divider source. Five clock options are available: Main clock, Slow clock, Master clock, PLLACK, UPLLCK. The Slow clock is the default clock source.

The PRES field is used to control the programmable clock prescaler. It is possible to choose among different values (1, 2, 4, 8, 16, 32, 64). Programmable clock output is prescaler input divided by PRES parameter. By default, the PRES value is cleared which means that PCKx is equal to Slow clock.

Once the PMC_PCKx register has been configured, The corresponding programmable clock must be enabled and the user is constrained to wait for the PCKRDYx bit to be set in PMC_SR. This can be done either by polling PCKRDYx in PMC_SR 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 CSS and PRES parameters are to be modified, the corresponding programmable clock must be disabled first. The parameters can then be modified. Once this has been done, the user must re-enable the programmable clock and wait for the PCKRDYx bit to be set.

10. Enable Peripheral Clocks

    Once all of the previous steps have been completed, the peripheral clocks can be enabled and/or disabled via PMC_PCERx and PMC_PCDRx.

26.17 Clock Switching Details

26.17.1 Master Clock Switching Timings

Table 26-1 and Table 26-2 give the worst case timings required for the Master clock 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 new selected clock has to be added.

Table 26-1. Clock Switching Timings (Worst Case)

To	From		
	Main Clock	**SLCK**	**PLL Clock**
Main Clock	–	4 × SLCK + 2.5 × Main Clock	3 × PLL Clock + 4 × SLCK + 1 × Main Clock
SLCK	0.5 × Main Clock + 4.5 × SLCK	–	3 × PLL Clock + 5 × SLCK
PLL Clock	0.5 × Main Clock + 4 × SLCK + PLLCOUNT × SLCK + 2.5 × PLL Clock	2.5 × PLL Clock + 5 × SLCK + PLLCOUNT × SLCK	2.5 × PLL Clock + 4 × SLCK + PLLCOUNT × SLCK

Notes: 1. PLL designates either the PLLA or the UPLL Clock.
2. PLLCOUNT designates either PLLACOUNT or UPLLCOUNT.

Table 26-2. Clock Switching Timings Between Two PLLs (Worst Case)

To	From	
	PLLA Clock	**UPLL Clock**
PLLA Clock	2.5 × PLLA Clock + 4 × SLCK + PLLACOUNT × SLCK	3 × PLLA Clock + 4 × SLCK + 1.5 × PLLA Clock
UPLL Clock	3 × UPLL Clock + 4 × SLCK + 1.5 × UPLL Clock	2.5 × UPLL Clock + 4 × SLCK + UPLLCOUNT × SLCK

26.17.2 Clock Switching Waveforms

Figure 26-7. Switch Master Clock from Slow Clock to PLL Clock

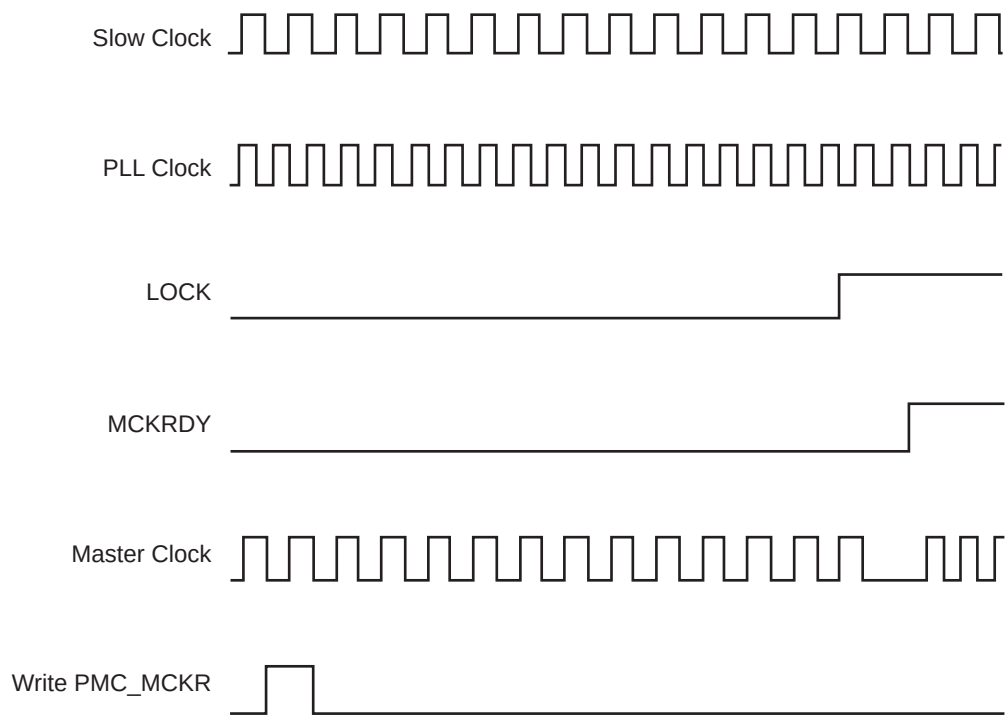


Figure 26-8. Switch Master Clock from Main Clock to Slow Clock

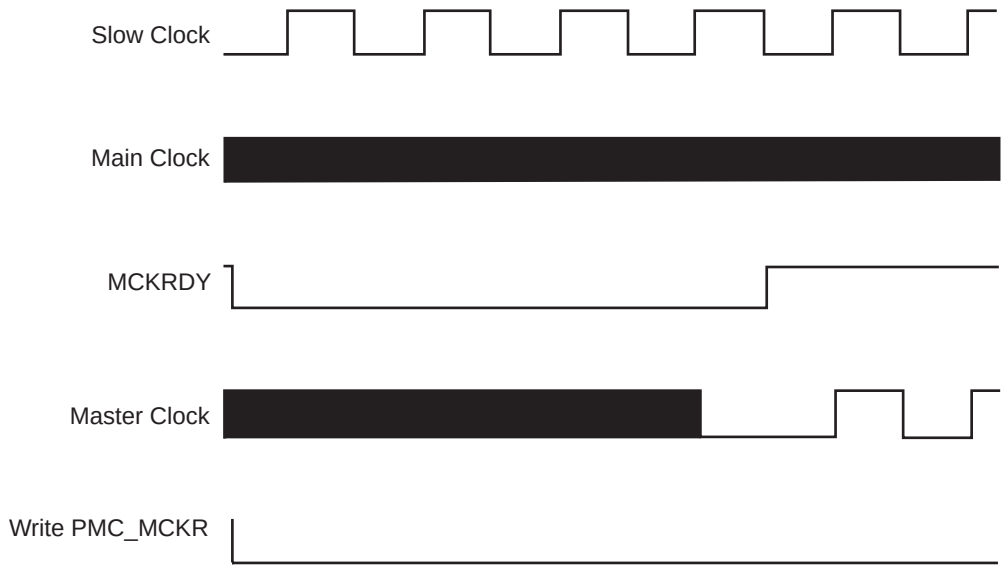


Figure 26-9. Change PLLA Programming

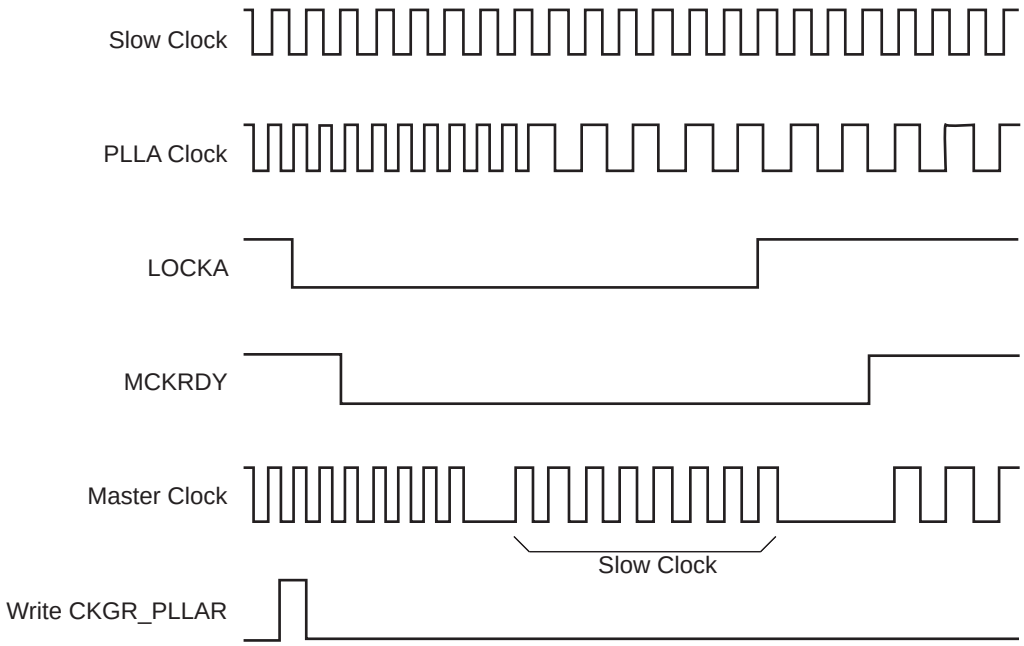
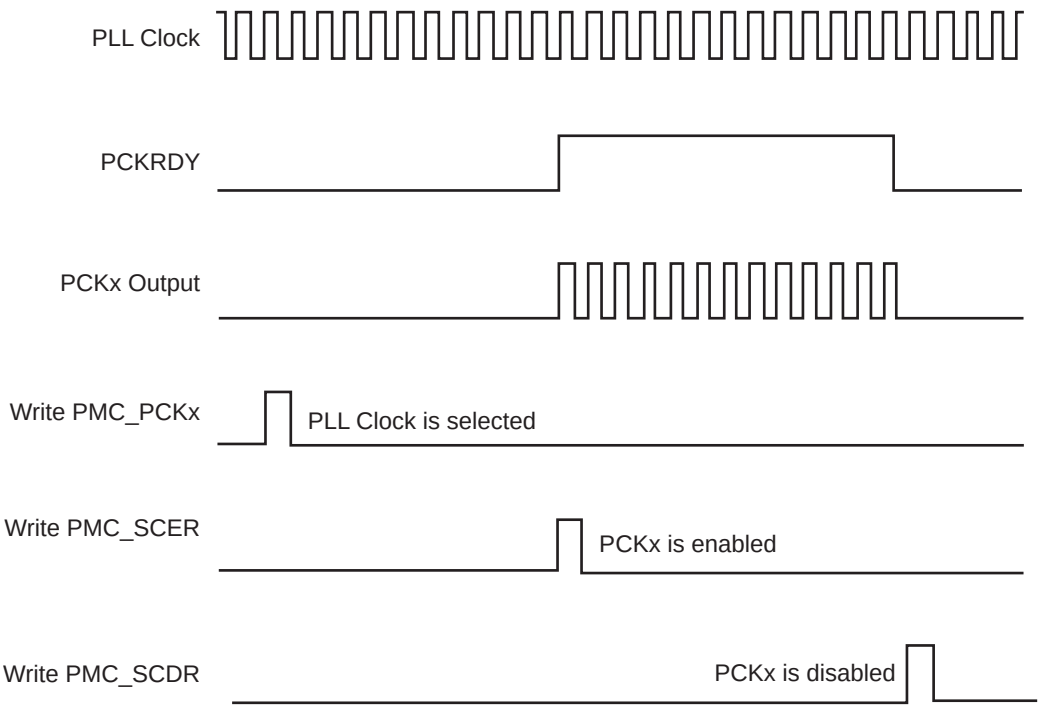


Figure 26-10. Programmable Clock Output Programming



26.18 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 in the PMC Write Protection Mode Register (PMC_WPMR).

If a write access to a write-protected register is detected, the WPVS bit 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 PMC_WPSR.

The following registers can be write-protected:

- PMC System Clock Enable Register
- PMC System Clock Disable Register
- PMC Peripheral Clock Enable Register 0
- PMC Peripheral Clock Disable Register 0
- PMC Clock Generator Main Oscillator Register
- PMC Clock Generator Main Clock Frequency Register
- PMC Clock Generator PLLA Register
- PMC Master Clock Register
- PMC USB Clock Register
- PMC Programmable Clock Register
- PLL Charge Pump Current Register
- PMC Peripheral Clock Enable Register 1
- PMC Peripheral Clock Disable Register 1

26.19 Power Management Controller (PMC) User Interface

Table 26-3. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	System Clock Enable Register	PMC_SCER	Write-only	–
0x0004	System Clock Disable Register	PMC_SCDR	Write-only	–
0x0008	System Clock Status Register	PMC_SCSR	Read-only	0x0000_0005
0x000C	Reserved	–	–	–
0x0010	Peripheral Clock Enable Register 0	PMC_PCER0	Write-only	–
0x0014	Peripheral Clock Disable Register 0	PMC_PCDR0	Write-only	–
0x0018	Peripheral Clock Status Register 0	PMC_PCSR0	Read-only	0x0000_0000
0x001C	UTMI Clock Register	CKGR_UCKR	Read/Write	0x1020_0000
0x0020	Main Oscillator Register	CKGR_MOR	Read/Write	0x0000_0000
0x0024	Main Clock Frequency Register	CKGR_MCFR	Read/Write	0x0000_0000
0x0028	PLLA Register	CKGR_PLLAR	Read/Write	0x0000_3F00
0x002C	Reserved	–	–	–
0x0030	Master Clock Register	PMC_MCKR	Read/Write	0x0000_0001
0x0034	Reserved	–	–	–
0x0038	USB Clock Register	PMC_USB	Read/Write	0x0000_0000
0x003C	Soft Modem Clock Register	PMC_SMD	Read/Write	0x0000_0000
0x0040	Programmable Clock 0 Register	PMC_PCK0	Read/Write	0x0000_0000
0x0044	Programmable Clock 1 Register	PMC_PCK1	Read/Write	0x0000_0000
0x0048	Programmable Clock 2 Register	PMC_PCK2	Read/Write	0x0000_0000
0x004C–0x005C	Reserved	–	–	–
0x0060	Interrupt Enable Register	PMC_IER	Write-only	–
0x0064	Interrupt Disable Register	PMC_IDR	Write-only	–
0x0068	Status Register	PMC_SR	Read-only	0x0001_0008
0x006C	Interrupt Mask Register	PMC_IMR	Read-only	0x0000_0000
0x0070–0x0074	Reserved	–	–	–
0x0078	Fault Output Clear Register	PMC_FOCR	Write-only	–
0x007C	Reserved	–	–	–
0x0080	PLL Charge Pump Current Register	PMC_PLLICPR	Read/Write	0x0000 0101
0x0084–0x00E0	Reserved	–	–	–
0x00E4	Write ProtectIon Mode Register	PMC_WPMR	Read/Write	0x0000_0000
0x00E8	Write Protection Status Register	PMC_WPSR	Read-only	0x0000_0000
0x00EC–0x00FC	Reserved	–	–	–
0x0100	Peripheral Clock Enable Register 1	PMC_PCER1	Write-only	–
0x0104	Peripheral Clock Disable Register 1	PMC_PCDR1	Write-only	–

Table 26-3. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x0108	Peripheral Clock Status Register 1	PMC_PCSR1	Read-only	0x0000_0000
0x010C	Peripheral Control Register	PMC_PCR	Read/Write	0x0000_0000
0x0110	Reserved	–	–	–

26.19.1 PMC System Clock Enable Register

Name: PMC_SCER

Address: 0xF0018000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	PCK2	PCK1	PCK0
7	6	5	4	3	2	1	0
UDP	UHP	–	SMDCK	LCDCK	DDRCK	–	–

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

- **DDRCK: DDR Clock Enable**

0: No effect.

1: Enables the DDR clock.

- **LCDCK: MCK2x Clock Enable**

0: No effect.

1: Enables the MCK2x clock.

Note: MCK2x is selected as LCD Pixel source clock if LCDC_LCDCFG0.CLKSEL = 1.

- **SMDCK: SMD Clock Enable**

0: No effect.

1: Enables the soft modem clock.

- **UHP: USB Host OHCI Clocks Enable**

0: No effect.

1: Enables the UHP48M and UHP12M OHCI clocks.

- **UDP: USB Device Clock Enable**

0: No effect.

1: Enables the USB Device clock.

- **PCKx: Programmable Clock x Output Enable**

0: No effect.

1: Enables the corresponding Programmable Clock output.

26.19.2 PMC System Clock Disable Register

Name: PMC_SCDR

Address: 0xF0018004

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	PCK2	PCK1	PCK0
7	6	5	4	3	2	1	0
UDP	UHP	–	SMDCK	LCDCK	DDRCK	–	PCK

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

- **PCK: Processor Clock Disable**

0: No effect.

1: Disables the Processor clock. This is used to enter the processor in Idle mode.

- **DDRCK: DDR Clock Disable**

0: No effect.

1: Disables the DDR clock.

- **LCDCK: MCK2x Clock Disable**

0: No effect.

1: Disables the MCK2x clock.

- **SMDCK: SMD Clock Disable**

0: No effect.

1: Disables the soft modem clock.

- **UHP: USB Host OHCI Clock Disable**

0: No effect.

1: Disables the UHP48M and UHP12M OHCI clocks.

- **UDP: USB Device Clock Enable**

0: No effect.

1: Disables the USB Device clock.

- **PCKx: Programmable Clock x Output Disable**

0: No effect.

1: Disables the corresponding Programmable Clock output.

26.19.3 PMC System Clock Status Register

Name: PMC_SCSR

Address: 0xF0018008

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	PCK2	PCK1	PCK0
7	6	5	4	3	2	1	0
UDP	UHP	–	SMDCK	LCDCK	DDRCK	–	PCK

- **PCK: Processor Clock Status**

0: The Processor clock is disabled.

1: The Processor clock is enabled.

- **DDRCK: DDR Clock Status**

0: The DDR clock is disabled.

1: The DDR clock is enabled.

- **LCDCK: MCK2x Clock Status**

0: The MCK2x clock is disabled.

1: The MCK2x clock is enabled.

Note: MCK2x is selected as LCD Pixel source clock if LCDC_LCDCFG0.CLKSEL = 1.

- **SMDCK: SMD Clock Status**

0: The soft modem clock is disabled.

1: The soft modem clock is enabled.

- **UHP: USB Host Port Clock Status**

0: The UHP48M and UHP12M OHCI clocks are disabled.

1: The UHP48M and UHP12M OHCI clocks are enabled.

- **UDP: USB Device Port Clock Status**

0: The USB Device clock is disabled.

1: The USB Device clock is enabled.

- **PCKx: Programmable Clock x Output Status**

0: The corresponding Programmable Clock output is disabled.

1: The corresponding Programmable Clock output is enabled.

26.19.4 PMC Peripheral Clock Enable Register 0

Name: PMC_PCER0

Address: 0xF0018010

Access: Write-only

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	–	–

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

- **PIDx: Peripheral Clock x Enable**

0: No effect.

1: Enables the corresponding peripheral clock.

Notes:
1. PID2 to PID31 refer to identifiers as defined in Section 8.2 “Peripheral Identifiers”. Other peripherals can be enabled in PMC_PCER1.
2. Programming the control bits of the Peripheral ID that are not implemented has no effect on the behavior of the PMC.

26.19.5 PMC Peripheral Clock Disable Register 0

Name: PMC_PCDR0

Address: 0xF0018014

Access: Write-only

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	–	–

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

- **PIDx: Peripheral Clock x Disable**

0: No effect.

1: Disables the corresponding peripheral clock.

Note: PID2 to PID31 refer to identifiers as defined in Section 8.2 “Peripheral Identifiers”. Other peripherals can be disabled in PMC_PCDR1.

26.19.6 PMC Peripheral Clock Status Register 0

Name: PMC_PCSR0

Address: 0xF0018018

Access: Read-only

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	–	–

- **PIDx: Peripheral Clock x Status**

0: The corresponding peripheral clock is disabled.

1: The corresponding peripheral clock is enabled.

Note: PID2 to PID31 refer to identifiers as defined in Section 8.2 “Peripheral Identifiers”. Other peripherals status can be read in PMC_PCSR1.

26.19.7 PMC UTMI Clock Configuration Register

Name: CKGR_UCKR

Address: 0xF001801C

Access: Read/Write

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

- **UPLLEN: UTMI PLL Enable**

0: The UTMI PLL is disabled.

1: The UTMI PLL is enabled.

When UPLLEN is set, the LOCKU flag is set once the UTMI PLL startup time is achieved.

- **UPLLCOUNT: UTMI PLL Startup Time**

Specifies the number of Slow clock cycles multiplied by 8 for the UTMI PLL startup time.

- **BIASEN: UTMI BIAS Enable**

0: The UTMI BIAS is disabled.

1: The UTMI BIAS is enabled.

- **BIASCOUNT: UTMI BIAS Startup Time**

Specifies the number of Slow clock cycles for the UTMI BIAS startup time.

26.19.8 PMC Clock Generator Main Oscillator Register

Name: CKGR_MOR

Address: 0xF0018020

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	XT32KFME	CFDEN	MOSCSEL

23	22	21	20	19	18	17	16
KEY							

15	14	13	12	11	10	9	8
MOSCXTST							

7	6	5	4	3	2	1	0
–	0			–	–	MOSCXTBY	MOSCXTEN

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

Warning: bits 6:4 must always be configured to 0 when programming CKGR_MOR.

- **MOSCXTEN: 12 MHz Crystal Oscillator Enable**

A crystal must be connected between XIN and XOUT.

0: The 12 MHz crystal oscillator is disabled.

1: The 12 MHz crystal oscillator is enabled. MOSCXTBY must be cleared.

When MOSCXTEN is set, the MOSCXTS flag is set once the crystal oscillator startup time is achieved.

- **MOSCXTBY: 12 MHz Crystal Oscillator Bypass**

0: No effect.

1: The 12 MHz crystal oscillator is bypassed. MOSCXTEN must be cleared. An external clock must be connected on XIN.

When MOSCXTBY is set, the MOSCXTS flag in PMC_SR is automatically set.

Clearing MOSCXTEN and MOSCXTBY bits allows resetting the MOSCXTS flag.

Note: When Main Oscillator Bypass is disabled (MOSCXTBY = 0), the MOSCXTS flag must be read as 0 in PMC_SR prior to enabling the main crystal oscillator (MOSCXTEN = 1).

- **MOSCXTST: 12 MHz Crystal Oscillator Startup Time**

Specifies the number of Slow clock cycles multiplied by 8 for the crystal oscillator startup time.

- **KEY: Password**

Value	Name	Description
0x37	PASSWD	Writing any other value in this field aborts the write operation.

- **MOSCSEL: Main Clock Oscillator Selection**

0: The 12 MHz oscillator is selected.

1: The 12 MHz crystal oscillator is selected.

- **CFDEN: Clock Failure Detector Enable**

0: The clock failure detector is disabled.

1: The clock failure detector is enabled.

- **XT32KFME: 32.768 kHz Crystal Oscillator Frequency Monitoring Enable**

0: The 32.768 kHz crystal oscillator frequency monitoring is disabled.

1: The 32.768 kHz crystal oscillator frequency monitoring is enabled.

26.19.9 PMC Clock Generator Main Clock Frequency Register

Name: CKGR_MCFR

Address: 0xF0018024

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	RCMEAS	–	–	–	MAINFRDY

15	14	13	12	11	10	9	8
MAINF							

7	6	5	4	3	2	1	0
MAINF							

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

- **MAINF: Main Clock Frequency**

Gives the number of Main clock cycles within 16 Slow clock periods. To calculate the frequency of the measured clock:

$f_{MAINCK} = (MAINF \times f_{SLCK}) / 16$

where frequency is in MHz.

- **MAINFRDY: Main Clock Frequency Measure Ready**

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.

Note: 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.

- **RCMEAS: RC Oscillator Frequency Measure (write-only)**

0: No effect.

1: Restarts measuring of the frequency of the Main clock source. MAINF will carry the new frequency as soon as a low to high transition occurs on the MAINFRDY flag.

The measure is performed on the main frequency (i.e., not limited to RC oscillator only), but if the Main clock frequency source is the 12 MHz crystal oscillator, the restart of measuring is not needed because of the well known stability of crystal oscillators.

26.19.10PMC Clock Generator PLLA Register

Name: CKGR_PLLAR

Address: 0xF0018028

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	ONE	–	–	–	–	MULA

23	22	21	20	19	18	17	16
MULA						OUTA	

15	14	13	12	11	10	9	8
OUTA		PLLACOUNT					

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DIVA

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

Possible limitations on PLL input frequencies and multiplier factors should be checked before using the PMC.

- **DIVA: Divider A**

0: PLLA is disabled.

1: Divider is bypassed and the PLL input entry is Main clock (MAINCK).

- **PLLACOUNT: PLLA Counter**

Specifies the number of Slow clock cycles before the LOCKA bit is set in PMC_SR after CKGR_PLLAR is written.

- **OUTA: PLLA Clock Frequency Range**

To be programmed to 0.

- **MULA: PLLA Multiplier**

0: The PLLA is disabled.

1–127: The PLLA Clock frequency is the PLLA input frequency multiplied by MULA + 1.

- **ONE: Must Be Set to 1**

Bit 29 must always be set to 1 when programming CKGR_PLLAR.

26.19.11PMC Master Clock Register

Name: PMC_MCKR

Address: 0xF0018030

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	H32MXDIV

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	PLLADIV2	–	–	MDIV	

7	6	5	4	3	2	1	0
–	PRES			–	–	CSS	

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

- **CSS: Master/Processor Clock Source Selection**

Value	Name	Description
0	SLOW_CLK	Slow clock is selected
1	MAIN_CLK	Main clock is selected
2	PLLA_CLK	PLLACK is selected
3	UPLL_CLK	UPLL Clock is selected

- **PRES: Master/Processor Clock Prescaler**

Value	Name	Description
0	CLOCK	Selected clock
1	CLOCK_DIV2	Selected clock divided by 2
2	CLOCK_DIV4	Selected clock divided by 4
3	CLOCK_DIV8	Selected clock divided by 8
4	CLOCK_DIV16	Selected clock divided by 16
5	CLOCK_DIV32	Selected clock divided by 32
6	CLOCK_DIV64	Selected clock divided by 64
7	–	Reserved

- **MDIV: Master Clock Division**

Value	Name	Description
0	EQ_PCK	Master Clock is Prescaler Output Clock divided by 1. **Warning**: DDRCK is not available.
1	PCK_DIV2	Master Clock is Prescaler Output Clock divided by 2. DDRCK is equal to MCK.
2	PCK_DIV4	Master Clock is Prescaler Output Clock divided by 4. DDRCK is equal to MCK.
3	PCK_DIV3	Master Clock is Prescaler Output Clock divided by 3. DDRCK is equal to MCK.

- **PLLADIV2: PLLA Divisor by 2**

Bit PLLADIV2 must always be set to 1 when MDIV is set to 3.

- **H32MXDIV: AHB 32-bit Matrix Divisor**

Value	Name	Description
0	H32MXDIV1	The AHB 32-bit Matrix frequency is equal to the AHB 64-bit Matrix frequency. It is possible only if the AHB 64-bit Matrix frequency does not exceed 100 MHz.
1	H32MXDIV2	The AHB 32-bit Matrix frequency is equal to the AHB 64-bit Matrix frequency divided by 2.

26.19.12PMC USB Clock Register

Name: PMC_USB

Address: 0xF0018038

Access: Read/Write

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

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

- **USBS: USB OHCI Input Clock Selection**

0: USB Clock Input is PLLA.

1: USB Clock Input is UPLL.

- **USBDIV: Divider for USB OHCI Clock**

USB Clock is Input clock divided by USBDIV + 1.

26.19.13PMC SMD Clock Register

Name: PMC_SMD

Address: 0xF001803C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	SMDDIV				

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SMDS

- **SMDS: SMD Input Clock Selection**

0: SMD clock input is PLLA.

1: SMD clock input is UPLL.

- **SMDDIV: Divider for SMD Clock**

SMD clock is input clock divided by SMD + 1.

26.19.14PMC Programmable Clock Register

Name: PMC_PCKx[x = 0..2]

Address: 0xF0018040

Access: Read/Write

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
–	PRES			–	CSS		

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

- **CSS: Master Clock Source Selection**

Value	Name	Description
0	SLOW_CLK	Slow clock is selected
1	MAIN_CLK	Main clock is selected
2	PLLA_CLK	PLLACK is selected
3	UPLL_CLK	UPLL Clock is selected
4	MCK_CLK	Master Clock is selected

- **PRES: Programmable Clock Prescaler**

Value	Name	Description
0	CLOCK	Selected clock
1	CLOCK_DIV2	Selected clock divided by 2
2	CLOCK_DIV4	Selected clock divided by 4
3	CLOCK_DIV8	Selected clock divided by 8
4	CLOCK_DIV16	Selected clock divided by 16
5	CLOCK_DIV32	Selected clock divided by 32
6	CLOCK_DIV64	Selected clock divided by 64
7	–	Reserved

26.19.15PMC Interrupt Enable Register

Name: PMC_IER

Address: 0xF0018060

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	XT32KERR	–	–	CFDEV	–	MOSCSELS
15	14	13	12	11	10	9	8
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	LOCKU	–	–	MCKRDY	–	LOCKA	MOSCXTS

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

0: No effect

1: Enables the corresponding interrupt

- **MOSCXTS: 12 MHz Crystal Oscillator Status Interrupt Enable**
- **LOCKA: PLLA Lock Interrupt Enable**
- **MCKRDY: Master Clock Ready Interrupt Enable**
- **LOCKU: UTMI PLL Lock Interrupt Enable**
- **PCKRDYx: Programmable Clock Ready x Interrupt Enable**
- **MOSCSELS: Main Clock Source Oscillator Selection Status Interrupt Enable**
- **CFDEV: Clock Failure Detector Event Interrupt Enable**
- **XT32KERR: 32.768 kHz Crystal Oscillator Error Interrupt Enable**

26.19.16PMC Interrupt Disable Register

Name: PMC_IDR

Address: 0xF0018064

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	XT32KERR	–	–	CFDEV	–	MOSCSELS
15	14	13	12	11	10	9	8
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	LOCKU	–	–	MCKRDY	–	LOCKA	MOSCXTS

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

0: No effect

1: Disables the corresponding interrupt

- **MOSCXTS: 12 MHz Crystal Oscillator Status Interrupt Disable**
- **LOCKA: PLLA Lock Interrupt Disable**
- **MCKRDY: Master Clock Ready Interrupt Disable**
- **LOCKU: UTMI PLL Lock Interrupt Enable**
- **PCKRDYx: Programmable Clock Ready x Interrupt Disable**
- **MOSCSELS: Main Oscillator Clock Source Selection Status Interrupt Disable**
- **CFDEV: Clock Failure Detector Event Interrupt Disable**
- **XT32KERR: 32.768 kHz Crystal Oscillator Error Interrupt Disable**

26.19.17PMC Status Register

Name: PMC_SR

Address: 0xF0018068

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	XT32KERR	FOS	CFDS	CFDEV	–	MOSCSELS
15	14	13	12	11	10	9	8
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
OSCSELS	LOCKU	–	–	MCKRDY	–	LOCKA	MOSCXTS

- **MOSCXTS: 12 MHz Crystal Oscillator Status**

0: 12 MHz crystal oscillator is not stabilized.

1: 12 MHz crystal oscillator is stabilized.

- **LOCKA: PLLA Lock Status**

0: PLLA is not locked.

1: PLLA is locked.

- **MCKRDY: Master Clock Status**

0: Master Clock is not ready.

1: Master Clock is ready.

- **LOCKU: UPLL Clock Status**

0: UPLL Clock is not ready.

1: UPLL Clock is ready.

- **OSCSELS: Slow Clock Oscillator Selection**

0: Embedded 64 kHz RC oscillator is selected.

1: 32.768 kHz crystal oscillator is selected.

- **PCKRDYx: Programmable Clock Ready Status**

0: Programmable Clock x is not ready.

1: Programmable Clock x is ready.

- **MOSCSELS: Main Oscillator Selection Status**

0: Selection is in progress.

1: Selection is done.

- **CFDEV: Clock Failure Detector Event**

0: No clock failure detection of the 12 MHz crystal oscillator has occurred since the last read of PMC_SR.

1: At least one clock failure detection of the 12 MHz crystal oscillator has occurred since the last read of PMC_SR.

- **CFDS: Clock Failure Detector Status**

0: A clock failure of the 12 MHz crystal oscillator is not detected.

1: A clock failure of the 12 MHz crystal oscillator is detected.

- **FOS: Clock Failure Detector Fault Output Status**

0: The fault output of the clock failure detector is inactive.

1: The fault output of the clock failure detector is active.

- **XT32KERR: 32.768 kHz Crystal Oscillator Error**

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.

26.19.18PMC Interrupt Mask Register

Name: PMC_IMR

Address: 0xF001806C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	XT32KERR	–	–	CFDEV	–	MOSCSELS
15	14	13	12	11	10	9	8
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	–	–	–	MCKRDY	–	LOCKA	MOSCXTS

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

0: Corresponding interrupt is not enabled.

1: Corresponding interrupt is enabled.

- **MOSCXTS: 12 MHz Crystal Oscillator Status Interrupt Mask**
- **LOCKA: PLLA Lock Interrupt Mask**
- **MCKRDY: Master Clock Ready Interrupt Mask**
- **PCKRDYx: Programmable Clock Ready x Interrupt Mask**
- **MOSCSELS: Main Oscillator Clock Source Selection Status Interrupt Mask**
- **CFDEV: Clock Failure Detector Event Interrupt Mask**
- **XT32KERR: 32.768 kHz Crystal Oscillator Error Interrupt Mask**

26.19.19PMC Fault Output Clear Register

Name: PMC_FOCR

Address: 0xF0018078

Access: Write-only

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
–	–	–	–	–	–	–	FOCLR

- **FOCLR: Fault Output Clear**

Clears the clock failure detector fault output.

26.19.20PLL Charge Pump Current Register

Name: PMC_PLLICPR

Address: 0xF0018080

Access: Read/Write

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

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

- **ICP_PLLA: Must Be Written to Zero**
- **IPLL_PLLA: Engineering Configuration PLLA**

Should be written to 0.

- **ICP_PLLU: Charge Pump Current PLL UTMI**

Should be written to 0.

- **IVCO_PLLU: Voltage Control Output Current PLL UTMI**

Should be written to 0.

26.19.21PMC Write Protection Mode Register

Name: PMC_WPMR

Address: 0xF00180E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

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).

See Section 26.18 “Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: 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.

26.19.22PMC Write Protection Status Register

Name: PMC_WPSR

Address: 0xF00180E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of PMC_WPSR.

1: A write protection violation has occurred since the last read of PMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

26.19.23PMC Peripheral Clock Enable Register 1

Name: PMC_PCER1

Address: 0xF0018100

Access: Write-only

31	30	29	28	27	26	25	24
PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
23	22	21	20	19	18	17	16
PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
15	14	13	12	11	10	9	8
PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
7	6	5	4	3	2	1	0
PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32

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

- **PIDx: Peripheral Clock x Enable**

0: No effect.

1: Enables the corresponding peripheral clock.

Notes:
1. PID32 to PID63 refer to identifiers as defined in Section 8.2 “Peripheral Identifiers”.
2. Programming the control bits of the Peripheral ID that are not implemented has no effect on the behavior of the PMC.

26.19.24PMC Peripheral Clock Disable Register 1

Name: PMC_PCDR1

Address: 0xF0018104

Access: Write-only

31	30	29	28	27	26	25	24
PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
23	22	21	20	19	18	17	16
PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
15	14	13	12	11	10	9	8
PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
7	6	5	4	3	2	1	0
PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32

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

- **PIDx: Peripheral Clock x Disable**

0: No effect.

1: Disables the corresponding peripheral clock.

Note: PID32 to PID63 refer to identifiers as defined in Section 8.2 “Peripheral Identifiers”.

26.19.25PMC Peripheral Clock Status Register 1

Name: PMC_PCSR1

Address: 0xF0018108

Access: Read-only

31	30	29	28	27	26	25	24
PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
23	22	21	20	19	18	17	16
PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
15	14	13	12	11	10	9	8
PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
7	6	5	4	3	2	1	0
PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32

- **PIDx: Peripheral Clock x Status**

0: The corresponding peripheral clock is disabled.

1: The corresponding peripheral clock is enabled.

Note: PID32 to PID63 refer to identifiers as defined in Section 8.2 “Peripheral Identifiers”.

26.19.26PMC Peripheral Control Register

Name: PMC_PCR

Address: 0xF001810C

Access: Read/Write

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

- **PID: Peripheral ID**

Peripheral ID selection from PID2 to the maximum PID number. This refers to identifiers as defined in the section “Peripheral Identifiers”.

- **CMD: Command**

0: Read mode

1: Write mode

- **EN: Enable**

0: The selected peripheral clock is disabled.

1: The selected peripheral clock is enabled.

27. Parallel Input/Output Controller (PIO)

27.1 Description

The Parallel Input/Output Controller (PIO) manages up to 152 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 is associated with a bit number in all of the 32-bit registers of the 32-bit wide user interface.

Each I/O line of the PIO Controller features the following:

- An input change interrupt enabling level change detection on any I/O line
- Additional Interrupt modes enabling rising edge, falling edge, low-level or high-level detection on any I/O line
- A glitch filter providing rejection of glitches lower than one-half of peripheral clock cycle
- A debouncing filter providing rejection of unwanted pulses from key or push button operations
- Multi-drive capability similar to an open drain I/O line
- Control of the I/O line pullup and pulldown
- Input visibility and output control

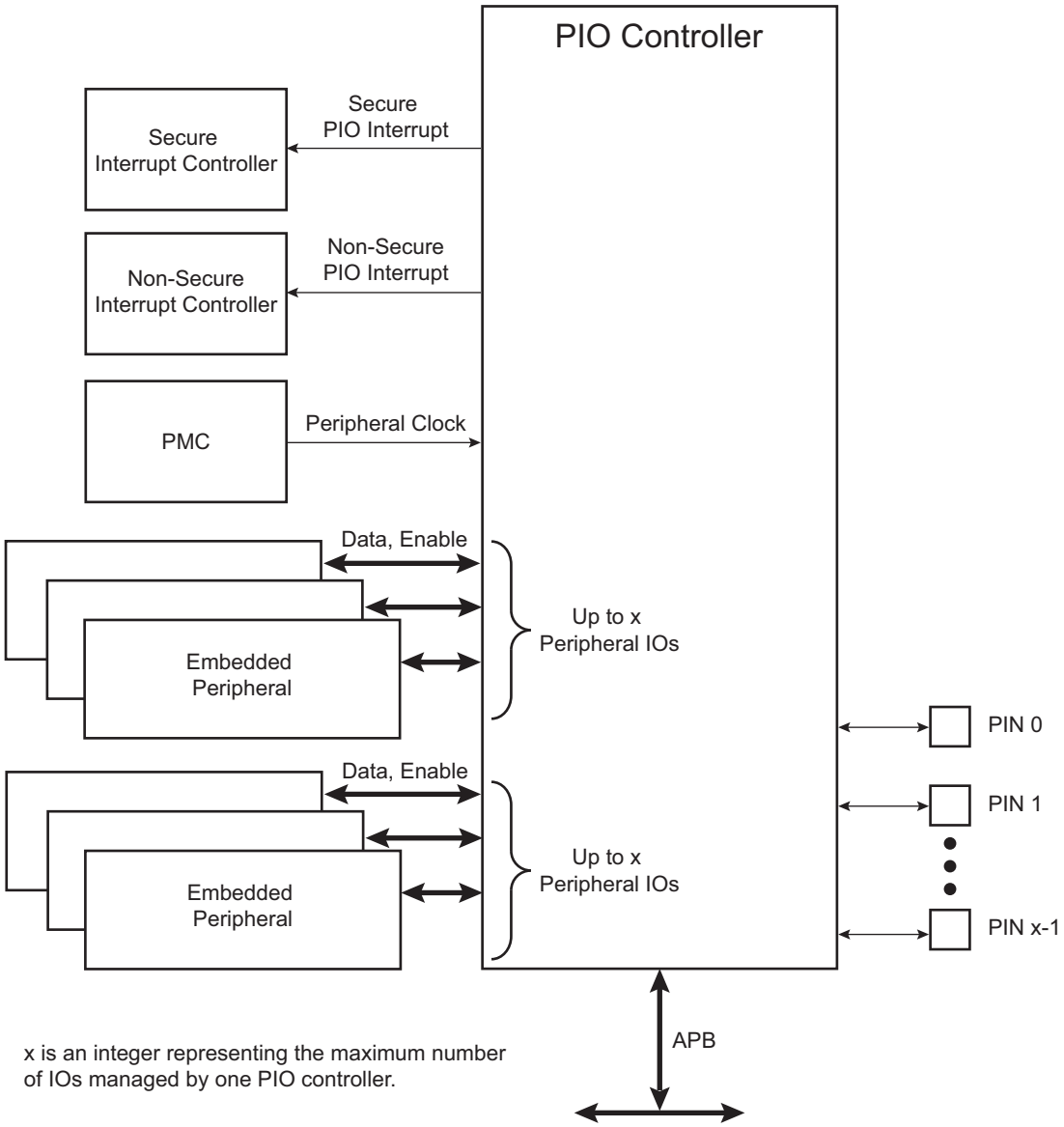
The PIO Controller also features a synchronous output providing up to 152 bits of data output in a single write operation.

27.2 Embedded Characteristics

- Up to 152 Programmable I/O Lines
- Fully Programmable through Set/Clear Registers
- Multiplexing of Four 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
  - Programmable Glitch Filter
  - Programmable Debouncing Filter
  - Multi-drive Option Enables Driving in Open Drain
  - Programmable Pullup on Each I/O Line
  - Pin Data Status Register, Supplies Visibility of the Level on the Pin at Any Time
  - Additional Interrupt Modes on a Programmable Event: Rising Edge, Falling Edge, Low-Level or High-Level
- Synchronous Output, Provides Set and Clear of Several I/O Lines in a Single Write
- Register Write Protection
- Programmable Schmitt Trigger Inputs
- Programmable I/O Drive

27.3 Block Diagram

Figure 27-1. Block Diagram



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 one or two 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 External Interrupt Lines

The interrupt signals FIQ and IRQ0 to IRQn are generally multiplexed through the PIO Controllers. However, it is not necessary to assign the I/O line to the interrupt function as the PIO Controller has no effect on inputs and the external interrupt lines are used only as inputs.

27.4.3 Power Management

The Power Management Controller controls the peripheral clock in order to save power. Writing any of the registers of the user interface does not require the peripheral clock to be enabled. This means that the configuration of the I/O lines does not require the peripheral 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 peripheral clock is disabled by default.

The user must configure the Power Management Controller before any access to the input line information.

27.4.4 Interrupt Sources

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. Refer to the PIO Controller peripheral identifier in Table 8-1 Peripheral Identifiers to identify the interrupt sources dedicated to the PIO Controllers. Using the PIO Controller requires the Interrupt Controller to be programmed first.

The PIO Controller interrupt can be generated only if the peripheral clock is enabled.

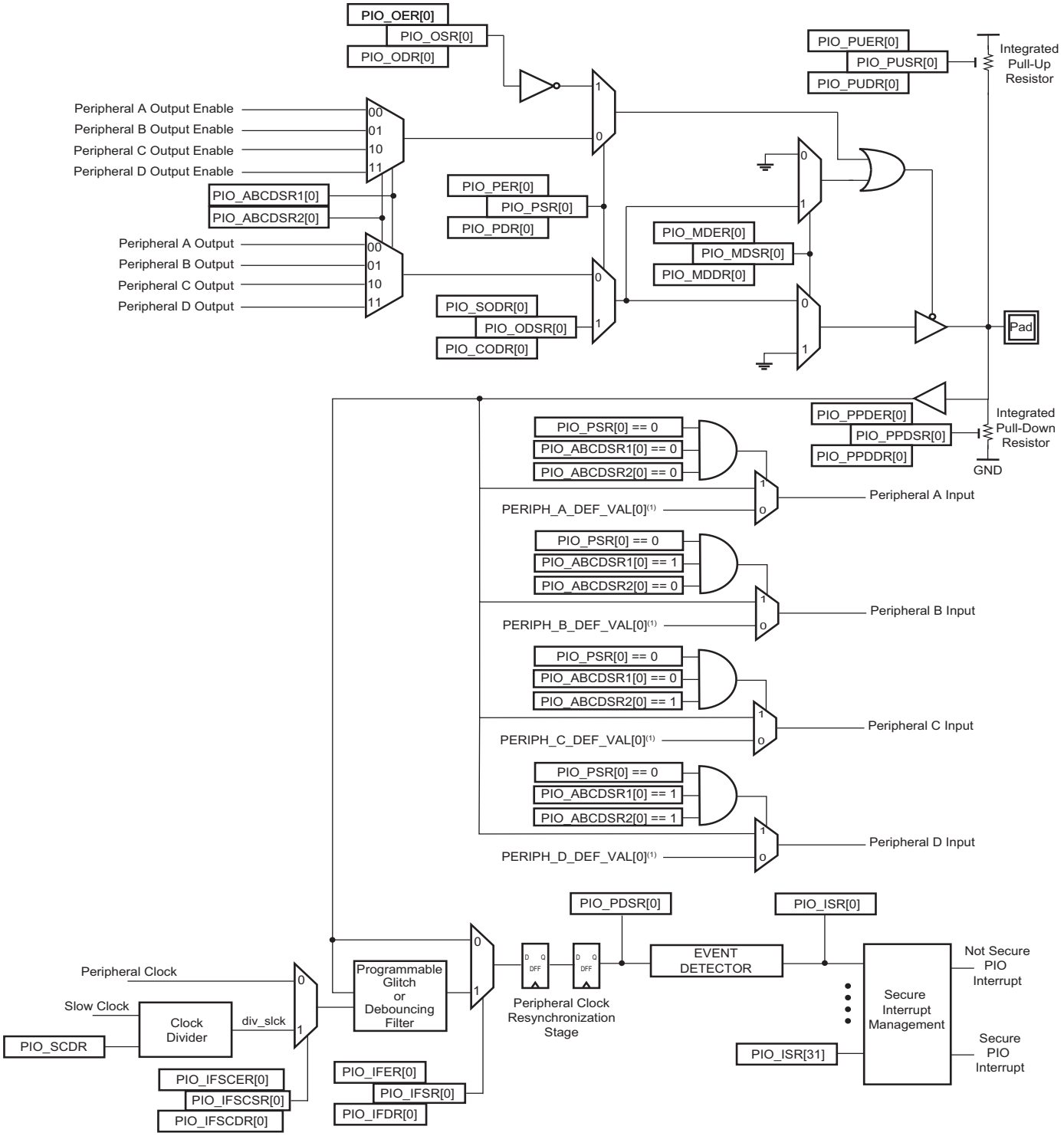
Table 27-1. Peripheral IDs

Instance	ID
PIOA	23
PIOB	24
PIOC	25
PIOD	5
PIOE	26

27.5 Functional Description

The PIO Controller features up to 152 fully-programmable I/O lines. Most of the control logic associated to each I/O is represented in Figure 27-2. In this description each signal shown represents one of up to 152 possible indexes.

Figure 27-2. I/O Line Control Logic



Note: 1. PERIPH_A_DEF_VAL = 0x00000000, PERIPH_B_DEF_VAL = 0x00000000, PERIPH_C_DEF_VAL = 0x00000000, PERIPH_D_DEF_VAL = 0x00000000.

27.5.1 Pullup and Pulldown Resistor Control

Each I/O line is designed with an embedded pullup resistor and an embedded pulldown resistor. The pullup resistor can be enabled or disabled by writing to the Pull-Up Enable Register (PIO_PUER) or Pull-Up Disable Register (PIO_PUDR), respectively. Writing to these registers results in setting or clearing the corresponding bit in the Pull-Up Status Register (PIO_PUSR). Reading a one in PIO_PUSR means the pullup is disabled and reading a zero means the pullup is enabled. The pulldown resistor can be enabled or disabled by writing the Pull-Down Enable Register (PIO_PPDER) or the Pull-Down Disable Register (PIO_PPDDR), respectively. Writing in these registers results in setting or clearing the corresponding bit in the Pull-Down Status Register (PIO_PPDSR). Reading a one in PIO_PPDSR means the pullup is disabled and reading a zero means the pulldown is enabled.

Enabling the pulldown resistor while the pullup resistor is still enabled is not possible. In this case, the write of PIO_PPDER for the relevant I/O line is discarded. Likewise, enabling the pullup resistor while the pulldown resistor is still enabled is not possible. In this case, the write of PIO_PUER for the relevant I/O line is discarded.

Control of the pullup resistor is possible regardless of the configuration of the I/O line.

After reset, depending on the I/O, pullup or pulldown can be set.

27.5.2 I/O Line or Peripheral Function Selection

When a pin is multiplexed with one or two peripheral functions, the selection is controlled with the Enable Register (PIO_PER) and the Disable Register (PIO_PDR). The Status Register (PIO_PSR) is the result of the set and clear registers and indicates whether the pin is controlled by the corresponding peripheral or by the PIO Controller. A value of zero indicates that the pin is controlled by the corresponding on-chip peripheral selected in the Peripheral ABCD Select registers (PIO_ABCDSR1 and PIO_ABCDSR2). A value of one indicates the pin is controlled by the PIO Controller.

If a pin is used as a general-purpose I/O line (not multiplexed with an on-chip peripheral), PIO_PER and PIO_PDR have no effect and PIO_PSR returns a one for the corresponding bit.

After reset, the I/O lines are controlled by the PIO Controller, i.e., PIO_PSR resets at one. However, in some events, it is important that PIO lines are controlled by the peripheral (as in the case of memory chip select lines that must be driven inactive after reset, or for address lines that must be driven low for booting out of an external memory). Thus, the reset value of PIO_PSR is defined at the product level and depends on the multiplexing of the device.

27.5.3 Peripheral A or B or C or D Selection

The PIO Controller provides multiplexing of up to four peripheral functions on a single pin. The selection is performed by writing PIO_ABCDSR1 and PIO_ABCDSR2.

For each pin:

- The corresponding bit at level zero in PIO_ABCDSR1 and the corresponding bit at level zero in PIO_ABCDSR2 means peripheral A is selected.
- The corresponding bit at level one in PIO_ABCDSR1 and the corresponding bit at level zero in PIO_ABCDSR2 means peripheral B is selected.
- The corresponding bit at level zero in PIO_ABCDSR1 and the corresponding bit at level one in PIO_ABCDSR2 means peripheral C is selected.
- The corresponding bit at level one in PIO_ABCDSR1 and the corresponding bit at level one in PIO_ABCDSR2 means peripheral D is selected.

Note that multiplexing of peripheral lines A, B, C and D affects both input and output peripheral lines. When a peripheral is not selected, its inputs are assigned with constant values (refer to Figure 27-2).

Writing in PIO_ABCDSR1 and PIO_ABCDSR2 manages the multiplexing regardless of the configuration of the pin. However, assignment of a pin to a peripheral function requires a write in PIO_ABCDSR1 and PIO_ABCDSR2

in addition to a write in PIO_PDR. The pin input is only routed to the input of the assigned peripheral. If the peripheral is not assigned to the pin, then a constant value is applied to the peripheral input line.

After reset, PIO_ABCDSR1 and PIO_ABCDSR2 are zero, thus indicating that all the PIO lines are configured on peripheral A. However, peripheral A generally does not drive the pin and PERIPH_x_DEF_VAL is applied to peripheral input lines as the PIO Controller resets in I/O Line mode.

If the software selects a peripheral A, B, C or D which does not exist for a pin, no alternate functions are enabled for this pin and the selection is taken into account. The PIO Controller does not carry out checks to prevent selection of a peripheral which does not exist.

27.5.4 Output Control

When the I/O line is assigned to a peripheral function, i.e., the corresponding bit in PIO_PSR is at zero, the drive of the I/O line is controlled by the peripheral. Peripheral A or B or C or D depending on the value in PIO_ABCDSR1 and PIO_ABCDSR2 determines whether the pin is driven or not.

When the I/O line is controlled by the PIO Controller, the pin can be configured to be driven. This is done by writing the Output Enable Register (PIO_OER) and Output Disable Register (PIO_ODR). The results of these write operations are detected in the Output Status Register (PIO_OSR). When a bit in this register is at zero, the corresponding I/O line is used as an input only. When the bit is at one, the corresponding I/O line is 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_SODR) and the Clear Output Data Register (PIO_CODR). These write operations, respectively, set and clear the Output Data Status Register (PIO_ODSR), which represents the data driven on the I/O lines. Writing in PIO_OER and PIO_ODR manages PIO_OSR 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_SODR and PIO_CODR affects PIO_ODSR. 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_SODR and PIO_CODR. 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_ODSR. Only bits unmasked by the Output Write Status Register (PIO_OWSR) are written. The mask bits in PIO_OWSR are set by writing to the Output Write Enable Register (PIO_OWER) and cleared by writing to the Output Write Disable Register (PIO_OWDR).

After reset, the synchronous data output is disabled on all the I/O lines as PIO_OWSR resets at 0x0.

27.5.6 Multi-Drive Control (Open Drain)

Each I/O can be independently programmed in open drain by using the multi-drive feature. This feature permits several drivers to be connected on the I/O line which is driven low only by each device. An external pullup resistor (or enabling of the internal one) is generally required to guarantee a high level on the line.

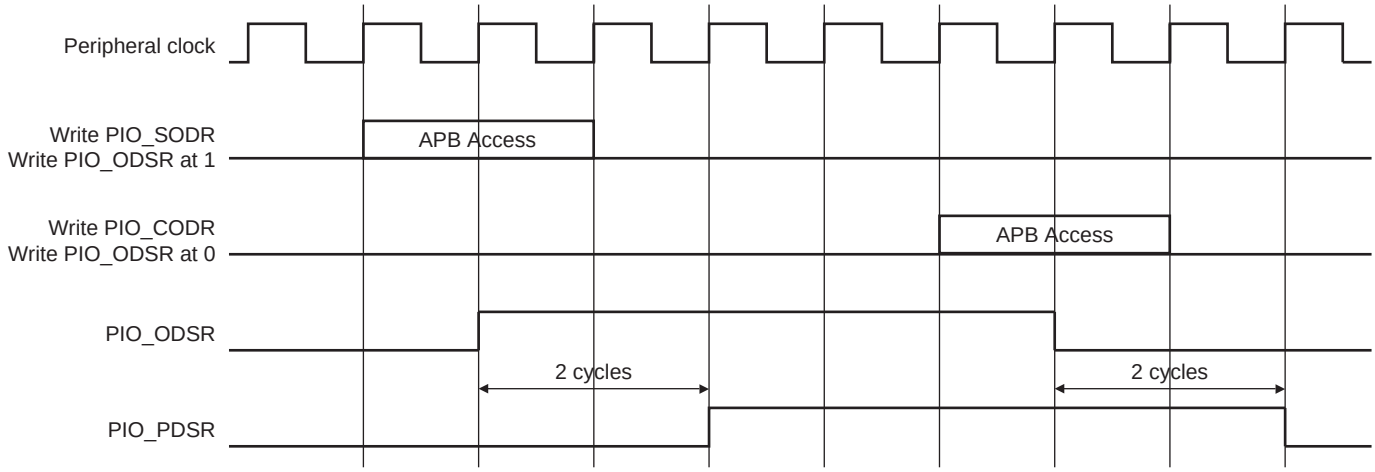
The multi-drive feature is controlled by the Multi-driver Enable Register (PIO_MDER) and the Multi-driver Disable Register (PIO_MDDR). The multi-drive can be selected whether the I/O line is controlled by the PIO Controller or assigned to a peripheral function. The Multi-driver Status Register (PIO_MDSR) indicates the pins that are configured to support external drivers.

After reset, the multi-drive feature is disabled on all pins, i.e., PIO_MDSR resets at value 0x0.

27.5.7 Output Line Timings

Figure 27-3 shows how the outputs are driven either by writing PIO_SODR or PIO_CODR, or by directly writing PIO_ODSR. This last case is valid only if the corresponding bit in PIO_OWSR is set. Figure 27-3 also shows when the feedback in the Pin Data Status Register (PIO_PDSR) is available.

Figure 27-3. Output Line Timings



27.5.8 Inputs

The level on each I/O line can be read through PIO_PDSR. 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_PDSR reads the levels present on the I/O line at the time the clock was disabled.

27.5.9 Input Glitch and Debouncing Filters

Optional input glitch and debouncing filters are independently programmable on each I/O line.

The glitch filter can filter a glitch with a duration of less than 1/2 peripheral clock and the debouncing filter can filter a pulse of less than 1/2 period of a programmable divided slow clock.

The selection between glitch filtering or debounce filtering is done by writing in the PIO Input Filter Slow Clock Disable Register (PIO_IFSCDR) and the PIO Input Filter Slow Clock Enable Register (PIO_IFSCER). Writing PIO_IFSCDR and PIO_IFSCER, respectively, sets and clears bits in the Input Filter Slow Clock Status Register (PIO_IFSCSR).

The current selection status can be checked by reading the PIO_IFSCSR.

- If PIO_IFSCSR[i] = 0: The glitch filter can filter a glitch with a duration of less than 1/2 master clock period.
- If PIO_IFSCSR[i] = 1: The debouncing filter can filter a pulse with a duration of less than 1/2 programmable divided slow clock period.

For the debouncing filter, the period of the divided slow clock is defined by writing in the DIV field of the Slow Clock Divider Debouncing Register (PIO_SCDR):

$t_{div\_slck} = ((DIV + 1) \times 2) \times t_{slck}$

When the glitch or debouncing filter is enabled, a glitch or pulse with a duration of less than 1/2 selected clock cycle (selected clock represents peripheral clock or divided slow clock depending on PIO_IFSCDR and PIO_IFSCER programming) is automatically rejected, while a pulse with a duration of one selected clock (peripheral clock or divided slow clock) cycle or more is accepted. For pulse durations between 1/2 selected clock

cycle and one selected clock cycle, the pulse may or may not be taken into account, depending on the precise timing of its occurrence. Thus for a pulse to be visible, it must exceed one selected clock cycle, whereas for a glitch to be reliably filtered out, its duration must not exceed 1/2 selected clock cycle.

The filters also introduce some latencies, illustrated in Figure 27-4 and Figure 27-5.

The glitch filters are controlled by the Input Filter Enable Register (PIO_IFER), the Input Filter Disable Register (PIO_IFDR) and the Input Filter Status Register (PIO_IFSR). Writing PIO_IFER and PIO_IFDR respectively sets and clears bits in PIO_IFSR. This last register enables the glitch filter on the I/O lines.

When the glitch and/or debouncing filter is enabled, it does not modify the behavior of the inputs on the peripherals. It acts only on the value read in PIO_PDSR and on the input change interrupt detection. The glitch and debouncing filters require that the peripheral clock is enabled.

Figure 27-4. Input Glitch Filter Timing

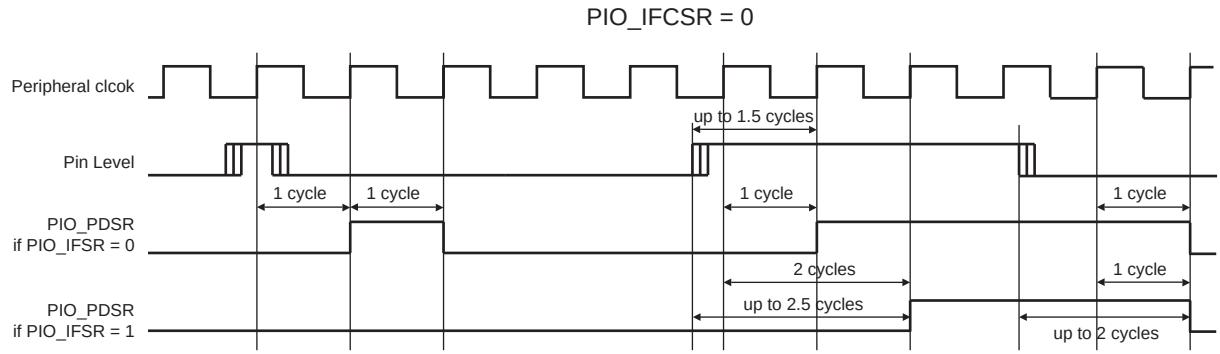
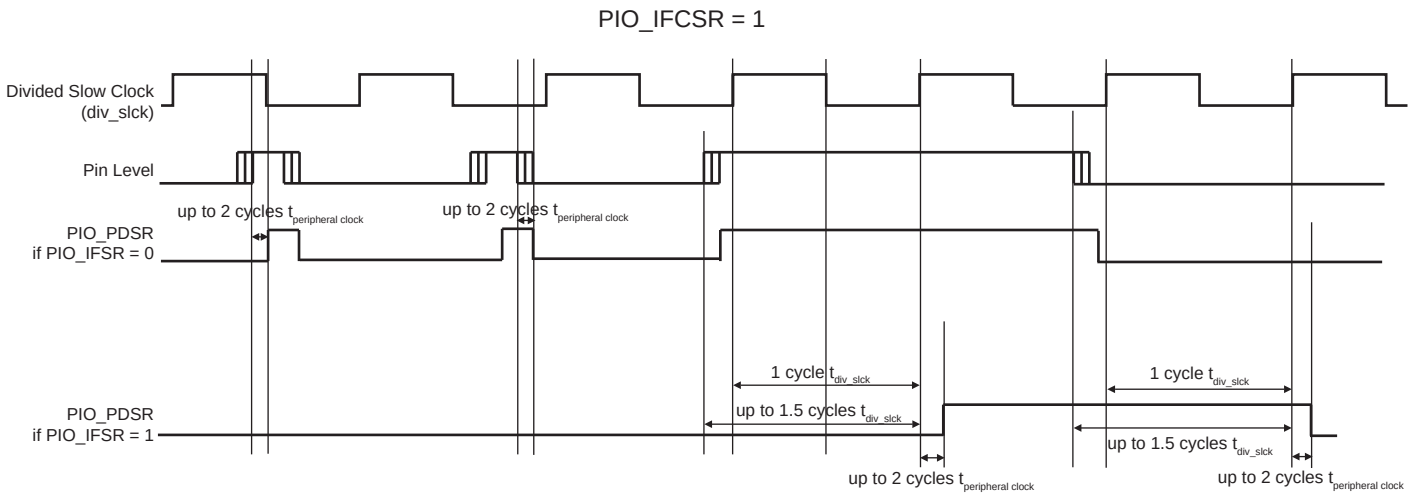


Figure 27-5. Input Debouncing Filter Timing



27.5.10 Input Edge/Level Interrupt

The PIO Controller can be programmed to generate an interrupt when it detects an edge or a level on an I/O line. The Input Edge/Level interrupt is controlled by writing the Interrupt Enable Register (PIO_IER) and the Interrupt Disable Register (PIO_IDR), which enable and disable the input change interrupt respectively by setting and clearing the corresponding bit in the Interrupt Mask Register (PIO_IMR). As input change detection is possible only by comparing two successive samplings of the input of the I/O line, the peripheral 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.

By default, the interrupt can be generated at any time an edge is detected on the input.

Some additional interrupt modes can be enabled/disabled by writing in the Additional Interrupt Modes Enable Register (PIO_AIMER) and Additional Interrupt Modes Disable Register (PIO_AIMDR). The current state of this selection can be read through the Additional Interrupt Modes Mask Register (PIO_AIMMR).

These additional modes are:

- Rising edge detection
- Falling edge detection
- Low-level detection
- High-level detection

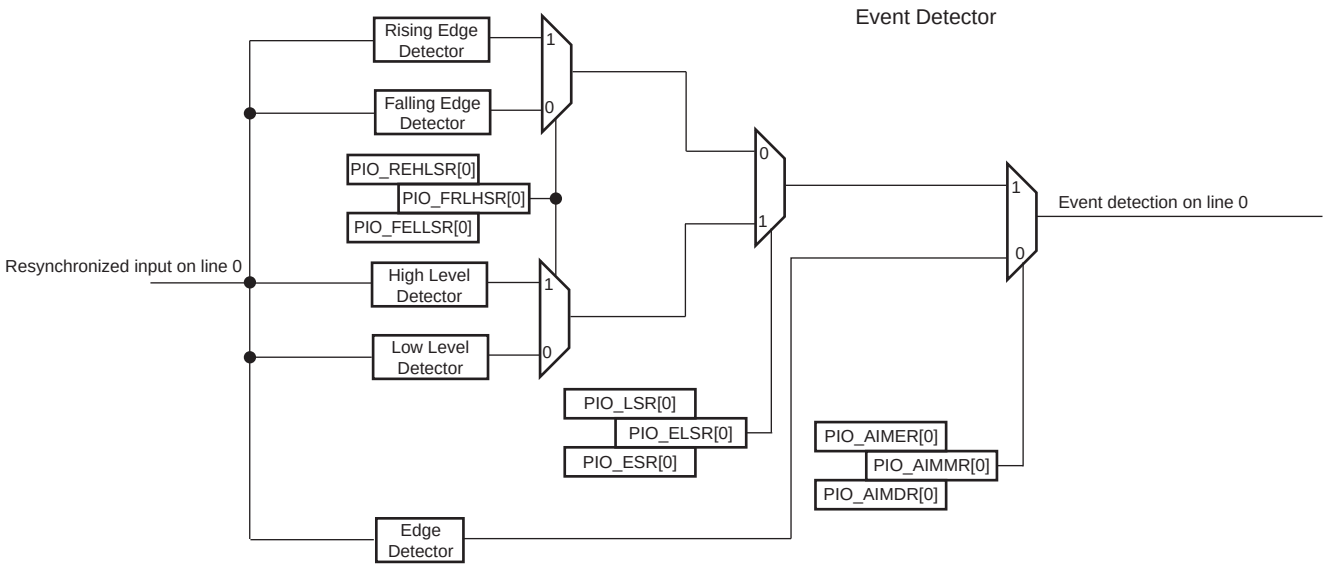
In order to select an additional interrupt mode:

- The type of event detection (edge or level) must be selected by writing in the Edge Select Register (PIO_ESR) and Level Select Register (PIO_LSR) which select, respectively, the edge and level detection. The current status of this selection is accessible through the Edge/Level Status Register (PIO_ELSR).
- The polarity of the event detection (rising/falling edge or high/low-level) must be selected by writing in the Falling Edge/Low-Level Select Register (PIO_FELLSR) and Rising Edge/High-Level Select Register (PIO_REHLSR) which allow to select falling or rising edge (if edge is selected in PIO_ELSR) edge or high- or low-level detection (if level is selected in PIO_ELSR). The current status of this selection is accessible through the Fall/Rise - Low/High Status Register (PIO_FRLHSR).

When an input edge or level is detected on an I/O line, the corresponding bit in the Interrupt Status Register (PIO_ISR) is set. If the corresponding bit in PIO_IMR is set, the PIO Controller interrupt line is asserted. The security level of the interrupt line depends on the corresponding bit value in the Interrupt Security Level Register (PIO_ISLR) and on the MATRIX_SPSELRx[PIO_ID] (Matrix Peripheral register) bit value. The secure interrupt signals of the 152 lines are ORed-wired together to generate a single secure interrupt signal to the secure interrupt controller. The non-secure interrupt signals of the 152 lines are ORed-wired together to generate a single non-secure interrupt signal to the non-secure interrupt controller (refer to Section 27.5.11 “Secure Interrupt Management”).

When the software reads PIO_ISR, all the interrupts are automatically cleared. This signifies that all the interrupts that are pending when PIO_ISR 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_ISR are performed.

Figure 27-6. Event Detector on Input Lines (Figure Represents Line 0)



Example of interrupt generation on following lines:

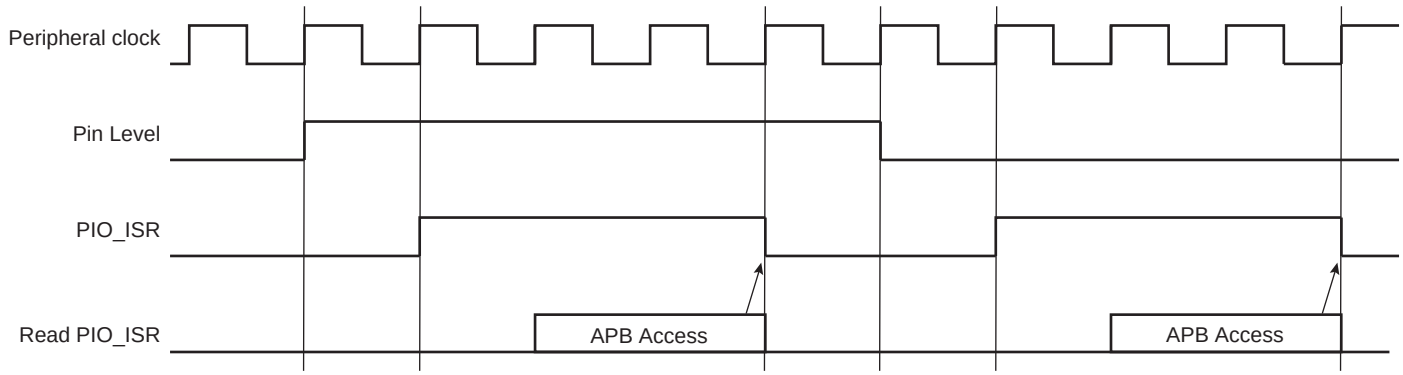
- Rising edge on PIO line 0
- Falling edge on PIO line 1
- Rising edge on PIO line 2
- Low-level on PIO line 3
- High-level on PIO line 4
- High-level on PIO line 5
- Falling edge on PIO line 6
- Rising edge on PIO line 7
- Any edge on the other lines

Table 27-2 provides the required configuration for this example.

Table 27-2. Configuration for Example Interrupt Generation

Configuration	Description
Interrupt Mode	All the interrupt sources are enabled by writing 32'hFFFF_FFFF in PIO_IER. Then the additional Interrupt mode is enabled for lines 0 to 7 by writing 32'h0000_00FF in PIO_AIMER.
Edge or Level Detection	Lines 3, 4 and 5 are configured in level detection by writing 32'h0000_0038 in PIO_LSR. The other lines are configured in edge detection by default, if they have not been previously configured. Otherwise, lines 0, 1, 2, 6 and 7 must be configured in edge detection by writing 32'h0000_00C7 in PIO_ESR.
Falling/Rising Edge or Low/High-Level Detection	Lines 0, 2, 4, 5 and 7 are configured in rising edge or high-level detection by writing 32'h0000_00B5 in PIO_REHLSR. The other lines are configured in falling edge or low-level detection by default if they have not been previously configured. Otherwise, lines 1, 3 and 6 must be configured in falling edge/low-level detection by writing 32'h0000_004A in PIO_FELLSR.

Figure 27-7. Input Change Interrupt Timings When No Additional Interrupt Modes



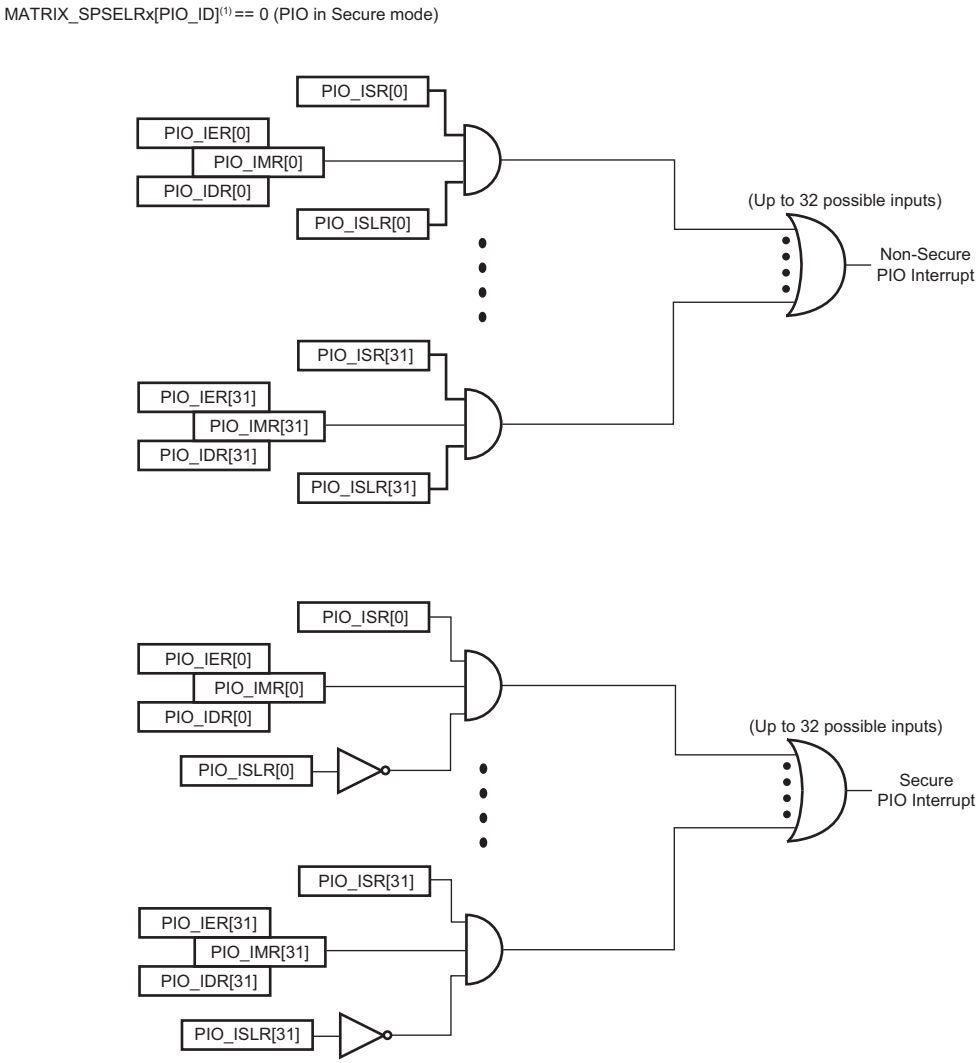
27.5.11 Secure Interrupt Management

The PIO Controller can drive one secure interrupt signal and one non-secure interrupt signal (refer to Figure 27-1). The secure interrupt signal is connected to the secure interrupt controller of the system. The non-secure interrupt signal is connected to the non-secure interrupt controller of the system.

There are two operating cases chosen according to the security level of the PIO Controller that is defined by the MATRIX_SPSELRx[PIO_ID] bit value (refer to the register description in Section 15. “Matrix (H64MX/H32MX)”):

- If MATRIX_SPSELRx[PIO_ID] = 0, the PIO Controller is in Secure mode. The security level of each PIO interrupt line is defined by PIO_ISLR. If the corresponding bit in PIO_ISLR is zero, the secure interrupt line will be asserted. If it is one, the non-secure interrupt line will be asserted. The 152 secure interrupt lines are ORed-wired together to generate the secure interrupt signal and the 152 non-secure interrupt lines are ORed-wired together to generate the non-secure interrupt signal.

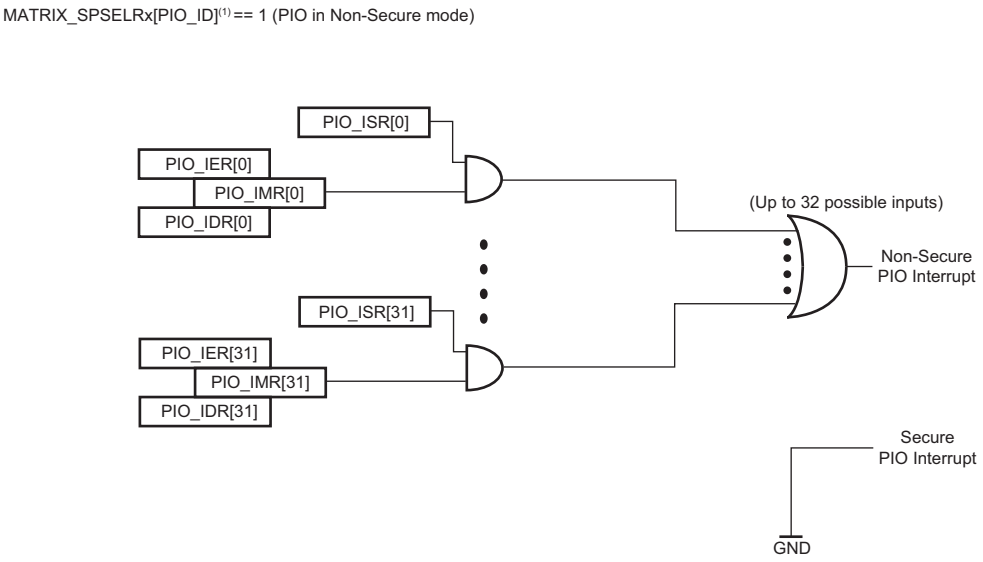
Figure 27-8. Secure PIO Interrupt Management



Note: 1. Refer to Section 15.13 “AHB Matrix (MATRIX) User Interface”.

- If MATRIX_SPSELRx[PIO_ID] = 1, the PIO Controller is in Non-secure mode. The security level of each PIO interrupt line is always non-secure (PIO_ISLR is not taken into account). The 152 non-secure interrupt lines are ORed-wired together to generate the non-secure interrupt signal. The secure interrupt signal is tied to zero.

Figure 27-9. Non-secure PIO Interrupt Management



Note: 1. Refer to Section 15.13 “AHB Matrix (MATRIX) User Interface”.

27.5.12 Programmable I/O Drive

It is possible to configure the I/O drive for pads PA0 to PA31. Refer to Section 55. “Electrical Characteristics”.

27.5.13 Programmable Schmitt Trigger

It is possible to configure each input for the Schmitt trigger. By default the Schmitt trigger is active. Disabling the Schmitt trigger is requested when using the QTouch® Library.

27.5.14 I/O Lines Programming Example

The programming example shown in Table 27-3 is used to obtain the following configuration:

- 4-bit output port on I/O lines 0 to 3 (should be written in a single write operation), open-drain, with pullup resistor
- Four output signals on I/O lines 4 to 7 (to drive LEDs for example), driven high and low, no pullup resistor, no pulldown resistor
- Four input signals on I/O lines 8 to 11 (to read push-button states for example), with pullup resistors, glitch filters and input change interrupts
- Four input signals on I/O line 12 to 15 to read an external device status (polled, thus no input change interrupt), no pullup resistor, no glitch filter
- I/O lines 16 to 19 assigned to peripheral A functions with pullup resistor
- I/O lines 20 to 23 assigned to peripheral B functions with pulldown resistor
- I/O lines 24 to 27 assigned to peripheral C with input change interrupt, no pullup resistor and no pulldown resistor
- I/O lines 28 to 31 assigned to peripheral D, no pullup resistor and no pulldown resistor

Table 27-3. Programming Example

Register	Value to be Written
PIO_PER	0x0000_FFFF
PIO_PDR	0xFFFF_0000

Table 27-3. Programming Example (Continued)

PIO_OER	0x0000_00FF
PIO_ODR	0xFFFF_FF00
PIO_IFER	0x0000_0F00
PIO_IFDR	0xFFFF_F0FF
PIO_SODR	0x0000_0000
PIO_CODR	0x0FFF_FFFF
PIO_IER	0x0F00_0F00
PIO_IDR	0xF0FF_F0FF
PIO_MDER	0x0000_000F
PIO_MDDR	0xFFFF_FFF0
PIO_PUDR	0xFFF0_00F0
PIO_PUER	0x000F_FF0F
PIO_PPDDR	0xFF0F_FFFF
PIO_PPDER	0x00F0_0000
PIO_ABCDSR1	0xF0F0_0000
PIO_ABCDSR2	0xFF00_0000
PIO_OWER	0x0000_000F
PIO_OWDR	0x0FFF_FFF0

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 the WPEN bit 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 can be write-protected:

- PIO Enable Register
- PIO Disable Register
- PIO Output Enable Register
- PIO Interrupt Security Level Register
- PIO Output Disable Register
- PIO Input Filter Enable Register
- PIO Input Filter Disable Register
- PIO Multi-driver Enable Register
- PIO Multi-driver Disable Register
- PIO Pull-Up Disable Register
- PIO Pull-Up Enable Register
- PIO Peripheral ABCD Select Register 1
- PIO Peripheral ABCD Select Register 2
- PIO Output Write Enable Register

- PIO Output Write Disable Register
- PIO Pad Pull-Down Disable Register
- PIO Pad Pull-Down Enable Register

27.6 Parallel Input/Output Controller (PIO) User Interface

Each I/O line controlled by the PIO Controller is associated with a bit in each of the PIO Controller User Interface registers. Each register is 32-bit wide. If a parallel I/O line is not defined, writing to the corresponding bits has no effect. Undefined bits read zero. If the I/O line is not multiplexed with any peripheral, the I/O line is controlled by the PIO Controller and PIO_PSR returns one systematically.

Table 27-4. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	PIO Enable Register	PIO_PER	Write-only	–
0x0004	PIO Disable Register	PIO_PDR	Write-only	–
0x0008	PIO Status Register	PIO_PSR	Read-only	(1)
0x000C	PIO Interrupt Security Level Register	PIO_ISLR	Read/Write	0x00000000
0x0010	Output Enable Register	PIO_OER	Write-only	–
0x0014	Output Disable Register	PIO_ODR	Write-only	–
0x0018	Output Status Register	PIO_OSR	Read-only	0x00000000
0x001C	Reserved	–	–	–
0x0020	Glitch Input Filter Enable Register	PIO_IFER	Write-only	–
0x0024	Glitch Input Filter Disable Register	PIO_IFDR	Write-only	–
0x0028	Glitch Input Filter Status Register	PIO_IFSR	Read-only	0x00000000
0x002C	Reserved	–	–	–
0x0030	Set Output Data Register	PIO_SODR	Write-only	–
0x0034	Clear Output Data Register	PIO_CODR	Write-only	
0x0038	Output Data Status Register	PIO_ODSR	Read-only or(2) Read/Write	–
0x003C	Pin Data Status Register	PIO_PDSR	Read-only	(3)
0x0040	Interrupt Enable Register	PIO_IER	Write-only	–
0x0044	Interrupt Disable Register	PIO_IDR	Write-only	–
0x0048	Interrupt Mask Register	PIO_IMR	Read-only	0x00000000
0x004C	Interrupt Status Register(4)	PIO_ISR	Read-only	0x00000000
0x0050	Multi-driver Enable Register	PIO_MDER	Write-only	–
0x0054	Multi-driver Disable Register	PIO_MDDR	Write-only	–
0x0058	Multi-driver Status Register	PIO_MDSR	Read-only	0x00000000
0x005C	Reserved	–	–	–
0x0060	Pull-Up Disable Register	PIO_PUDR	Write-only	–
0x0064	Pull-Up Enable Register	PIO_PUER	Write-only	–
0x0068	Pad Pull-Up Status Register	PIO_PUSR	Read-only	(1)
0x006C	Reserved	–	–	–

Table 27-4. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x0070	Peripheral ABCD Select Register 1	PIO_ABCDSR1	Read/Write	0x00000000
0x0074	Peripheral ABCD Select Register 2	PIO_ABCDSR2	Read/Write	0x00000000
0x0078–0x007C	Reserved	–	–	–
0x0080	Input Filter Slow Clock Disable Register	PIO_IFSCDR	Write-only	–
0x0084	Input Filter Slow Clock Enable Register	PIO_IFSCER	Write-only	–
0x0088	Input Filter Slow Clock Status Register	PIO_IFSCSR	Read-only	0x00000000
0x008C	Slow Clock Divider Debouncing Register	PIO_SCDR	Read/Write	0x00000000
0x0090	Pad Pull-Down Disable Register	PIO_PPDDR	Write-only	–
0x0094	Pad Pull-Down Enable Register	PIO_PPDER	Write-only	–
0x0098	Pad Pull-Down Status Register	PIO_PPDSR	Read-only	(1)
0x009C	Reserved	–	–	–
0x00A0	Output Write Enable	PIO_OWER	Write-only	–
0x00A4	Output Write Disable	PIO_OWDR	Write-only	–
0x00A8	Output Write Status Register	PIO_OWSR	Read-only	0x00000000
0x00AC	Reserved	–	–	–
0x00B0	Additional Interrupt Modes Enable Register	PIO_AIMER	Write-only	–
0x00B4	Additional Interrupt Modes Disable Register	PIO_AIMDR	Write-only	–
0x00B8	Additional Interrupt Modes Mask Register	PIO_AIMMR	Read-only	0x00000000
0x00BC	Reserved	–	–	–
0x00C0	Edge Select Register	PIO_ESR	Write-only	–
0x00C4	Level Select Register	PIO_LSR	Write-only	–
0x00C8	Edge/Level Status Register	PIO_ELSR	Read-only	0x00000000
0x00CC	Reserved	–	–	–
0x00D0	Falling Edge/Low-Level Select Register	PIO_FELLSR	Write-only	–
0x00D4	Rising Edge/High-Level Select Register	PIO_REHLSR	Write-only	–
0x00D8	Fall/Rise - Low/High Status Register	PIO_FRLHSR	Read-only	0x00000000
0x00DC	Reserved	–	–	–
0x00E0	Reserved	–	–	–
0x00E4	Write Protection Mode Register	PIO_WPMR	Read/Write	0x00000000
0x00E8	Write Protection Status Register	PIO_WPSR	Read-only	0x00000000
0x00EC–0x00FC	Reserved	–	–	–
0x0100	Schmitt Trigger Register	PIO_SCHMITT	Read/Write	0x00000000
0x0104–0x010C	Reserved	–	–	–
0x0110	Reserved	–	–	–
0x0114	Reserved	–	–	–
0x0118	I/O Drive Register 1	PIO_DRIVER1	Read/Write	0xAAAAAAAA
0x011C	I/O Drive Register 2	PIO_DRIVER2	Read/Write	0xAAAAAAAA

Table 27-4. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x0120–0x014C	Reserved	–	–	–

Notes:
1. Reset value depends on the product implementation.
2. PIO_ODSR is Read-only or Read/Write depending on PIO_OWSR I/O lines.
3. 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.
4. PIO_ISR is reset at 0x0. However, the first read of the register may read a different value as input changes may have occurred.
5. If an offset is not listed in the table it must be considered as reserved.

27.6.1 PIO Enable Register

Name: PIO_PER

Address: 0xFC06A000 (PIOA), 0xFC06B000 (PIOB), 0xFC06C000 (PIOC), 0xFC068000 (PIOD), 0xFC06D000 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: PIO Enable**

0: No effect.

1: Enables the PIO to control the corresponding pin (disables peripheral control of the pin).

27.6.2 PIO Disable Register

Name: PIO_PDR

Address: 0xFC06A004 (PIOA), 0xFC06B004 (PIOB), 0xFC06C004 (PIOC), 0xFC068004 (PIOD), 0xFC06D004 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: PIO Disable**

0: No effect.

1: Disables the PIO from controlling the corresponding pin (enables peripheral control of the pin).

27.6.3 PIO Status Register

Name: PIO_PSR

Address: 0xFC06A008 (PIOA), 0xFC06B008 (PIOB), 0xFC06C008 (PIOC), 0xFC068008 (PIOD), 0xFC06D008 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: PIO Status**

0: PIO is inactive on the corresponding I/O line (peripheral is active).

1: PIO is active on the corresponding I/O line (peripheral is inactive).

27.6.4 PIO Interrupt Security Level Register

Name: PIO_ISLR

Address: 0xFC06A00C (PIOA), 0xFC06B00C (PIOB), 0xFC06C00C (PIOC), 0xFC06800C (PIOD), 0xFC06D00C (PIOE)

Access: Read/Write

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: PIO Interrupt Security Level**

0: Interrupt line is defined as secure.

1: Interrupt line is defined as non-secure.

Note: If PIO is set in Non-secure mode, then this register is not taken into account for PIO interrupt generation.

27.6.5 PIO Output Enable Register

Name: PIO_OER

Address: 0xFC06A010 (PIOA), 0xFC06B010 (PIOB), 0xFC06C010 (PIOC), 0xFC068010 (PIOD), 0xFC06D010 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Output Enable**

0: No effect.

1: Enables the output on the I/O line.

27.6.6 PIO Output Disable Register

Name: PIO_ODR

Address: 0xFC06A014 (PIOA), 0xFC06B014 (PIOB), 0xFC06C014 (PIOC), 0xFC068014 (PIOD), 0xFC06D014 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Output Disable**

0: No effect.

1: Disables the output on the I/O line.

27.6.7 PIO Output Status Register

Name: PIO_OSR

Address: 0xFC06A018 (PIOA), 0xFC06B018 (PIOB), 0xFC06C018 (PIOC), 0xFC068018 (PIOD), 0xFC06D018 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Output Status**

0: The I/O line is a pure input.

1: The I/O line is enabled in output.

27.6.8 PIO Input Filter Enable Register

Name: PIO_IFER

Address: 0xFC06A020 (PIOA), 0xFC06B020 (PIOB), 0xFC06C020 (PIOC), 0xFC068020 (PIOD), 0xFC06D020 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Input Filter Enable**

0: No effect.

1: Enables the input glitch filter on the I/O line.

27.6.9 PIO Input Filter Disable Register

Name: PIO_IFDR

Address: 0xFC06A024 (PIOA), 0xFC06B024 (PIOB), 0xFC06C024 (PIOC), 0xFC068024 (PIOD), 0xFC06D024 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Input Filter Disable**

0: No effect.

1: Disables the input glitch filter on the I/O line.

27.6.10 PIO Input Filter Status Register

Name: PIO_IFSR

Address: 0xFC06A028 (PIOA), 0xFC06B028 (PIOB), 0xFC06C028 (PIOC), 0xFC068028 (PIOD), 0xFC06D028 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Filter Status**

0: The input glitch filter is disabled on the I/O line.

1: The input glitch filter is enabled on the I/O line.

27.6.11 PIO Set Output Data Register

Name: PIO_SODR

Address: 0xFC06A030 (PIOA), 0xFC06B030 (PIOB), 0xFC06C030 (PIOC), 0xFC068030 (PIOD), 0xFC06D030 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Set Output Data**

0: No effect.

1: Sets the data to be driven on the I/O line.

27.6.12 PIO Clear Output Data Register

Name: PIO_CODR

Address: 0xFC06A034 (PIOA), 0xFC06B034 (PIOB), 0xFC06C034 (PIOC), 0xFC068034 (PIOD), 0xFC06D034 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Clear Output Data**

0: No effect.

1: Clears the data to be driven on the I/O line.

27.6.13 PIO Output Data Status Register

Name: PIO_ODSR

Address: 0xFC06A038 (PIOA), 0xFC06B038 (PIOB), 0xFC06C038 (PIOC), 0xFC068038 (PIOD), 0xFC06D038 (PIOE)

Access: Read-only or Read/Write

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Output Data Status**

0: The data to be driven on the I/O line is 0.

1: The data to be driven on the I/O line is 1.

27.6.14 PIO Pin Data Status Register

Name: PIO_PDSR

Address: 0xFC06A03C (PIOA), 0xFC06B03C (PIOB), 0xFC06C03C (PIOC), 0xFC06803C (PIOD), 0xFC06D03C (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Output Data Status**

0: The I/O line is at level 0.

1: The I/O line is at level 1.

27.6.15 PIO Interrupt Enable Register

Name: PIO_IER

Address: 0xFC06A040 (PIOA), 0xFC06B040 (PIOB), 0xFC06C040 (PIOC), 0xFC068040 (PIOD), 0xFC06D040 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Enable**

0: No effect.

1: Enables the input change interrupt on the I/O line.

27.6.16 PIO Interrupt Disable Register

Name: PIO_IDR

Address: 0xFC06A044 (PIOA), 0xFC06B044 (PIOB), 0xFC06C044 (PIOC), 0xFC068044 (PIOD), 0xFC06D044 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Disable**

0: No effect.

1: Disables the input change interrupt on the I/O line.

27.6.17 PIO Interrupt Mask Register

Name: PIO_IMR

Address: 0xFC06A048 (PIOA), 0xFC06B048 (PIOB), 0xFC06C048 (PIOC), 0xFC068048 (PIOD), 0xFC06D048 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Mask**

0: Input change interrupt is disabled on the I/O line.

1: Input change interrupt is enabled on the I/O line.

27.6.18 PIO Interrupt Status Register

Name: PIO_ISR

Address: 0xFC06A04C (PIOA), 0xFC06B04C (PIOB), 0xFC06C04C (PIOC), 0xFC06804C (PIOD), 0xFC06D04C (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Status**

0: No input change has been detected on the I/O line since PIO_ISR was last read or since reset.

1: At least one input change has been detected on the I/O line since PIO_ISR was last read or since reset.

27.6.19 PIO Multi-driver Enable Register

Name: PIO_MDER

Address: 0xFC06A050 (PIOA), 0xFC06B050 (PIOB), 0xFC06C050 (PIOC), 0xFC068050 (PIOD), 0xFC06D050 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0-P31: Multi-drive Enable**

0: No effect.

1: Enables multi-drive on the I/O line.

27.6.20 PIO Multi-driver Disable Register

Name: PIO_MDDR

Address: 0xFC06A054 (PIOA), 0xFC06B054 (PIOB), 0xFC06C054 (PIOC), 0xFC068054 (PIOD), 0xFC06D054 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Multi-drive Disable**

0: No effect.

1: Disables multi-drive on the I/O line.

27.6.21 PIO Multi-driver Status Register

Name: PIO_MDSR

Address: 0xFC06A058 (PIOA), 0xFC06B058 (PIOB), 0xFC06C058 (PIOC), 0xFC068058 (PIOD), 0xFC06D058 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Multi-drive Status**

0: The multi-drive is disabled on the I/O line. The pin is driven at high- and low-level.

1: The multi-drive is enabled on the I/O line. The pin is driven at low-level only.

27.6.22 PIO Pull-Up Disable Register

Name: PIO_PUDR

Address: 0xFC06A060 (PIOA), 0xFC06B060 (PIOB), 0xFC06C060 (PIOC), 0xFC068060 (PIOD), 0xFC06D060 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Pull-Up Disable**

0: No effect.

1: Disables the pullup resistor on the I/O line.

27.6.23 PIO Pull-Up Enable Register

Name: PIO_PUER

Address: 0xFC06A064 (PIOA), 0xFC06B064 (PIOB), 0xFC06C064 (PIOC), 0xFC068064 (PIOD), 0xFC06D064 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Pull-Up Enable**

0: No effect.

1: Enables the pullup resistor on the I/O line.

27.6.24 PIO Pull-Up Status Register

Name: PIO_PUSR

Address: 0xFC06A068 (PIOA), 0xFC06B068 (PIOB), 0xFC06C068 (PIOC), 0xFC068068 (PIOD), 0xFC06D068 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Pull-Up Status**

0: Pullup resistor is enabled on the I/O line.

1: Pullup resistor is disabled on the I/O line.

27.6.25 PIO Peripheral ABCD Select Register 1

Name: PIO_ABCDSR1

Address: 0xFC06A070 (PIOA), 0xFC06B070 (PIOB), 0xFC06C070 (PIOC), 0xFC068070 (PIOD), 0xFC06D070 (PIOE)

Access: Read/Write

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Peripheral Select**

If the same bit is set to 0 in PIO_ABCDSR2:

0: Assigns the I/O line to the Peripheral A function.

1: Assigns the I/O line to the Peripheral B function.

If the same bit is set to 1 in PIO_ABCDSR2:

0: Assigns the I/O line to the Peripheral C function.

1: Assigns the I/O line to the Peripheral D function.

27.6.26 PIO Peripheral ABCD Select Register 2

Name: PIO_ABCDSR2

Address: 0xFC06A074 (PIOA), 0xFC06B074 (PIOB), 0xFC06C074 (PIOC), 0xFC068074 (PIOD), 0xFC06D074 (PIOE)

Access: Read/Write

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Peripheral Select**

If the same bit is set to 0 in PIO_ABCDSR1:

0: Assigns the I/O line to the Peripheral A function.

1: Assigns the I/O line to the Peripheral C function.

If the same bit is set to 1 in PIO_ABCDSR1:

0: Assigns the I/O line to the Peripheral B function.

1: Assigns the I/O line to the Peripheral D function.

27.6.27 PIO Input Filter Slow Clock Disable Register

Name: PIO_IFSCDR

Address: 0xFC06A080 (PIOA), 0xFC06B080 (PIOB), 0xFC06C080 (PIOC), 0xFC068080 (PIOD), 0xFC06D080 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Peripheral Clock Glitch Filtering Select**

0: No effect.

1: The glitch filter is able to filter glitches with a duration < $t_{peripheral\ clock}/2$.

27.6.28 PIO Input Filter Slow Clock Enable Register

Name: PIO_IFSCER

Address: 0xFC06A084 (PIOA), 0xFC06B084 (PIOB), 0xFC06C084 (PIOC), 0xFC068084 (PIOD), 0xFC06D084 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Slow Clock Debouncing Filtering Select**

0: No effect.

1: The debouncing filter is able to filter pulses with a duration < $t_{div\_slck}/2$.

27.6.29 PIO Input Filter Slow Clock Status Register

Name: PIO_IFSCSR

Address: 0xFC06A088 (PIOA), 0xFC06B088 (PIOB), 0xFC06C088 (PIOC), 0xFC068088 (PIOD), 0xFC06D088 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Glitch or Debouncing Filter Selection Status**

0: The glitch filter is able to filter glitches with a duration < $t_{peripheral\ clock}/2$.

1: The debouncing filter is able to filter pulses with a duration < $t_{div\_slck}/2$.

27.6.30 PIO Slow Clock Divider Debouncing Register

Name: PIO_SCDR

Address: 0xFC06A08C (PIOA), 0xFC06B08C (PIOB), 0xFC06C08C (PIOC), 0xFC06808C (PIOD), 0xFC06D08C (PIOE)

Access: Read/Write

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

- **DIV: Slow Clock Divider Selection for Debouncing**

$t_{div\_slck} = ((DIV + 1) \times 2) \times t_{slck}$

27.6.31 PIO Pad Pull-Down Disable Register

Name: PIO_PPDDR

Address: 0xFC06A090 (PIOA), 0xFC06B090 (PIOB), 0xFC06C090 (PIOC), 0xFC068090 (PIOD), 0xFC06D090 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Pull-Down Disable**

0: No effect.

1: Disables the pulldown resistor on the I/O line.

27.6.32 PIO Pad Pull-Down Enable Register

Name: PIO_PPDER

Address: 0xFC06A094 (PIOA), 0xFC06B094 (PIOB), 0xFC06C094 (PIOC), 0xFC068094 (PIOD), 0xFC06D094 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Pull-Down Enable**

0: No effect.

1: Enables the pulldown resistor on the I/O line.

27.6.33 PIO Pad Pull-Down Status Register

Name: PIO_PPDSR

Address: 0xFC06A098 (PIOA), 0xFC06B098 (PIOB), 0xFC06C098 (PIOC), 0xFC068098 (PIOD), 0xFC06D098 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Pull-Down Status**

0: Pulldown resistor is enabled on the I/O line.

1: Pulldown resistor is disabled on the I/O line.

27.6.34 PIO Output Write Enable Register

Name: PIO_OWER

Address: 0xFC06A0A0 (PIOA), 0xFC06B0A0 (PIOB), 0xFC06C0A0 (PIOC), 0xFC0680A0 (PIOD), 0xFC06D0A0 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Output Write Enable**

0: No effect.

1: Enables writing PIO_ODSR for the I/O line.

27.6.35 PIO Output Write Disable Register

Name: PIO_OWDR

Address: 0xFC06A0A4 (PIOA), 0xFC06B0A4 (PIOB), 0xFC06C0A4 (PIOC), 0xFC0680A4 (PIOD), 0xFC06D0A4 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

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

- **P0–P31: Output Write Disable**

0: No effect.

1: Disables writing PIO_ODSR for the I/O line.

27.6.36 PIO Output Write Status Register

Name: PIO_OWSR

Address: 0xFC06A0A8 (PIOA), 0xFC06B0A8 (PIOB), 0xFC06C0A8 (PIOC), 0xFC0680A8 (PIOD), 0xFC06D0A8 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Output Write Status**

0: Writing PIO_ODSR does not affect the I/O line.

1: Writing PIO_ODSR affects the I/O line.

27.6.37 PIO Additional Interrupt Modes Enable Register

Name: PIO_AIMER

Address: 0xFC06A0B0 (PIOA), 0xFC06B0B0 (PIOB), 0xFC06C0B0 (PIOC), 0xFC0680B0 (PIOD), 0xFC06D0B0 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Additional Interrupt Modes Enable**

0: No effect.

1: The interrupt source is the event described in PIO_ELSR and PIO_FRLHSR.

27.6.38 PIO Additional Interrupt Modes Disable Register

Name: PIO_AIMDR

Address: 0xFC06A0B4 (PIOA), 0xFC06B0B4 (PIOB), 0xFC06C0B4 (PIOC), 0xFC0680B4 (PIOD), 0xFC06D0B4 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Additional Interrupt Modes Disable**

0: No effect.

1: The Interrupt mode is set to the default Interrupt mode (Both-edge Detection).

27.6.39 PIO Additional Interrupt Modes Mask Register

Name: PIO_AIMMR

Address: 0xFC06A0B8 (PIOA), 0xFC06B0B8 (PIOB), 0xFC06C0B8 (PIOC), 0xFC0680B8 (PIOD), 0xFC06D0B8 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: IO Line Index**

Selects the IO event type triggering an interrupt.

0: The interrupt source is a both-edge detection event.

1: The interrupt source is described by the registers PIO_ELSR and PIO_FRLHSR.

27.6.40 PIO Edge Select Register

Name: PIO_ESR

Address: 0xFC06A0C0 (PIOA), 0xFC06B0C0 (PIOB), 0xFC06C0C0 (PIOC), 0xFC0680C0 (PIOD), 0xFC06D0C0 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Edge Interrupt Selection**

0: No effect.

1: The interrupt source is an edge-detection event.

27.6.41 PIO Level Select Register

Name: PIO_LSR

Address: 0xFC06A0C4 (PIOA), 0xFC06B0C4 (PIOB), 0xFC06C0C4 (PIOC), 0xFC0680C4 (PIOD), 0xFC06D0C4 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Level Interrupt Selection**

0: No effect.

1: The interrupt source is a level-detection event.

27.6.42 PIO Edge/Level Status Register

Name: PIO_ELSR

Address: 0xFC06A0C8 (PIOA), 0xFC06B0C8 (PIOB), 0xFC06C0C8 (PIOC), 0xFC0680C8 (PIOD), 0xFC06D0C8 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Edge/Level Interrupt Source Selection**

0: The interrupt source is an edge-detection event.

1: The interrupt source is a level-detection event.

27.6.43 PIO Falling Edge/Low-Level Select Register

Name: PIO_FELLSR

Address: 0xFC06A0D0 (PIOA), 0xFC06B0D0 (PIOB), 0xFC06C0D0 (PIOC), 0xFC0680D0 (PIOD), 0xFC06D0D0 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Falling Edge/Low-Level Interrupt Selection**

0: No effect.

1: The interrupt source is set to a falling edge detection or low-level detection event, depending on PIO_ELSR.

27.6.44 PIO Rising Edge/High-Level Select Register

Name: PIO_REHLSR

Address: 0xFC06A0D4 (PIOA), 0xFC06B0D4 (PIOB), 0xFC06C0D4 (PIOC), 0xFC0680D4 (PIOD), 0xFC06D0D4 (PIOE)

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Rising Edge/High-Level Interrupt Selection**

0: No effect.

1: The interrupt source is set to a rising edge detection or high-level detection event, depending on PIO_ELSR.

27.6.45 PIO Fall/Rise - Low/High Status Register

Name: PIO_FRLHSR

Address: 0xFC06A0D8 (PIOA), 0xFC06B0D8 (PIOB), 0xFC06C0D8 (PIOC), 0xFC0680D8 (PIOD), 0xFC06D0D8 (PIOE)

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Edge/Level Interrupt Source Selection**

0: The interrupt source is a falling edge detection (if PIO_ELSR = 0) or low-level detection event (if PIO_ELSR = 1).

1: The interrupt source is a rising edge detection (if PIO_ELSR = 0) or high-level detection event (if PIO_ELSR = 1).

27.6.46 PIO Write Protection Mode Register

Name: PIO_WPMR

Address: 0xFC06A0E4 (PIOA), 0xFC06B0E4 (PIOB), 0xFC06C0E4 (PIOC), 0xFC0680E4 (PIOD), 0xFC06D0E4 (PIOE)

Access: Read/Write

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

- **WPEN: Write Protection Enable**

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).

Refer to Section 27.5.15 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: 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.

27.6.47 PIO Write Protection Status Register

Name: PIO_WPSR

Address: 0xFC06A0E8 (PIOA), 0xFC06B0E8 (PIOB), 0xFC06C0E8 (PIOC), 0xFC0680E8 (PIOD), 0xFC06D0E8 (PIOE)

Access: Read-only

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

- **WPVS: Write Protection Violation Status**

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.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

27.6.48 PIO Schmitt Trigger Register

Name: PIO_SCHMITT

Address: 0xFC06A100 (PIOA), 0xFC06B100 (PIOB), 0xFC06C100 (PIOC), 0xFC068100 (PIOD), 0xFC06D100 (PIOE)

Access: Read/Write

31	30	29	28	27	26	25	24
SCHMITT31	SCHMITT30	SCHMITT29	SCHMITT28	SCHMITT27	SCHMITT26	SCHMITT25	SCHMITT24
23	22	21	20	19	18	17	16
SCHMITT23	SCHMITT22	SCHMITT21	SCHMITT20	SCHMITT19	SCHMITT18	SCHMITT17	SCHMITT16
15	14	13	12	11	10	9	8
SCHMITT15	SCHMITT14	SCHMITT13	SCHMITT12	SCHMITT11	SCHMITT10	SCHMITT9	SCHMITT8
7	6	5	4	3	2	1	0
SCHMITT7	SCHMITT6	SCHMITT5	SCHMITT4	SCHMITT3	SCHMITT2	SCHMITT1	SCHMITT0

- **SCHMITTx [x=0..31]: Schmitt Trigger Control**

0: Schmitt trigger is enabled.

1: Schmitt trigger is disabled.

27.6.49 PIO I/O Drive Register 1

Name: PIO_DRIVER1

Address: 0xFC06A118 (PIOA), 0xFC06B118 (PIOB), 0xFC06C118 (PIOC), 0xFC068118 (PIOD), 0xFC06D118 (PIOE)

Access: Read/Write

31	30	29	28	27	26	25	24
LINE15		LINE14		LINE13		LINE12	
23	22	21	20	19	18	17	16
LINE11		LINE10		LINE9		LINE8	
15	14	13	12	11	10	9	8
LINE7		LINE6		LINE5		LINE4	
7	6	5	4	3	2	1	0
LINE3		LINE2		LINE1		LINE0	

- **LINEx [x=0..15]: Drive of PIO Line x**

Value	Name	Description
0	LO_DRIVE	Low drive
1	LO_DRIVE	Low drive
2	ME_DRIVE	Medium drive
3	HI_DRIVE	High drive

27.6.50 PIO I/O Drive Register 2

Name: PIO_DRIVER2

Address: 0xFC06A11C (PIOA), 0xFC06B11C (PIOB), 0xFC06C11C (PIOC), 0xFC06811C (PIOD), 0xFC06D11C (PIOE)

Access: Read/Write

31	30	29	28	27	26	25	24
LINE31		LINE30		LINE29		LINE28	
23	22	21	20	19	18	17	16
LINE27		LINE26		LINE25		LINE24	
15	14	13	12	11	10	9	8
LINE23		LINE22		LINE21		LINE20	
7	6	5	4	3	2	1	0
LINE19		LINE18		LINE17		LINE16	

- **LINEx [x=16..31]: Drive of PIO line x**

Value	Name	Description
0	LO_DRIVE	Low drive
1	LO_DRIVE	Low drive
2	ME_DRIVE	Medium drive
3	HI_DRIVE	High drive

28. Multiport DDR-SDRAM Controller (MPDDRC)

28.1 Description

The Multiport DDR-SDRAM Controller (MPDDRC) is a multiport memory controller. It comprises eight slave AHB interfaces. All simultaneous accesses (eight independent AHB ports) are interleaved to maximize memory bandwidth and minimize transaction latency due to DDR-SDRAM protocol.

The MPDDRC extends the memory capabilities of a chip by providing the interface to the external 16-bit or 32-bit DDR-SDRAM device. The page size supports ranges from 2048 to 16384 rows and from 512 to 4096 columns. It supports dword (64-bit), word (32-bit), half-word (16-bit), and byte (8-bit) accesses.

The MPDDRC supports a read or write burst length of eight locations. This enables the command and address bus to anticipate the next command, thus reducing latency imposed by the DDR-SDRAM protocol and improving the DDR-SDRAM bandwidth. Moreover, MPDDRC keeps track of the active row in each bank, thus maximizing DDR-SDRAM performance, e.g., the application may be placed in one bank and data in other banks. To optimize performance, avoid accessing different rows in the same bank. The MPDDRC supports a CAS latency of 2, 3, 4, 5 or 6 and optimizes the read access depending on the frequency.

Self-refresh, Powerdown and Deep Powerdown modes minimize the consumption of the DDR-SDRAM device.

OCD (Off-chip Driver) and ODT (On-die Termination) modes are not supported.

28.2 Embedded Characteristics

- Eight advanced high performance bus (AHB) interfaces, management of all accesses maximizes memory bandwidth and minimizes transaction latency
- Bus transfer: dword, word, half word, byte access
- Supports low-power DDR2-SDRAM-S4 (LPDDR2), DDR2-SDRAM, low-power DDR1-SDRAM (LPDDR1)
- Numerous configurations supported
  - 2K, 4K, 8K, 16K row address memory parts
  - DDR-SDRAM with two or four internal banks (low-power DDR1-SDRAM)
  - DDR-SDRAM with four or eight internal banks (DDR2-SDRAM/Low-power DDR2-SDRAM-S4)
  - DDR-SDRAM with 16-bit or 32-bit data
  - One chip select for SDRAM device (512-Mbyte address space, 256-Mbyte address space with 16-bit data path)
- Programming Facilities
  - Multibank ping-pong access (up to four or eight banks opened at the same time = reduced average latency of transactions)
  - Timing parameters specified by software
  - Automatic refresh operation, refresh rate is programmable
  - Automatic update of DS, TCR and PASR parameters (low-power DDR-SDRAM devices)
- Energy-saving capabilities
  - Self-refresh, Powerdown, Active Powerdown and Deep Powerdown modes supported
- DDR-SDRAM powerup initialization by software
- CAS latency of 2, 3, 4, 5, 6 supported
- Reset function supported (DDR2-SDRAM)
- Auto-refresh per bank supported (low-power DDR2-SDRAM-S4)
- Automatic adjust refresh rate (low-power DDR2-SDRAM-S4)
- Auto-precharge command not used
- OCD (Off-chip Driver) mode, ODT (On-die Termination) are not supported

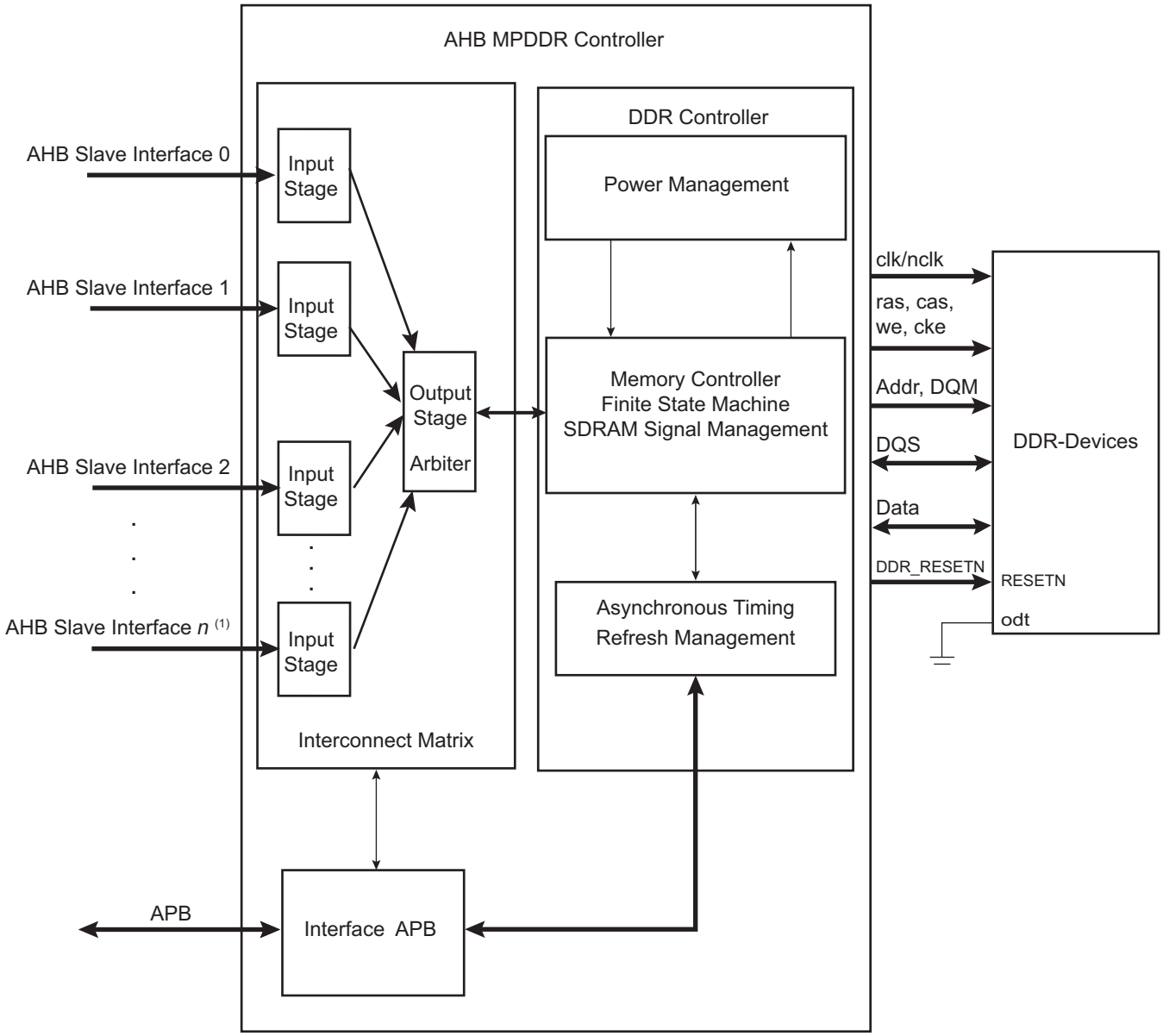
- Dynamic Scrambling with user key (no impact on bandwidth)

28.3 MPDDRC Module Diagram

MPDDRC is partitioned in two blocks (refer to Figure 28-1):

- Interconnect Matrix block that manages concurrent accesses on the AHB bus between four AHB masters and integrates an arbiter
- DDR controller that translates AHB requests (read/write) in the DDR-SDRAM protocol

Figure 28-1. MPDDRC Module Diagram



Note: 1. "*n*" can equal 3 or 7 (value is device-specific).

28.4 Product Dependencies, Initialization Sequence

28.4.1 Low-power DDR1-SDRAM Initialization

The initialization sequence is generated by software.

The low-power DDR1-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the MPDDRC Memory Device Register (MPDDRC_MD).
2. Program the features of the low-power DDR1-SDRAM device in the MPDDRC Configuration Register (number of columns, rows, banks, CAS latency and output drive strength) and in the MPDDRC Timing Parameter 0 Register/MPDDRC Timing Parameter 1 Register (asynchronous timing (TRC, TRAS, etc.)).
3. Program Temperature Compensated Self-refresh (TCR), Partial Array Self-refresh (PASR) and Drive Strength (DS) parameters in the MPDDRC Low-power Register.
4. A NOP command is issued to the low-power DDR1-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM address to acknowledge this command. The clocks which drive the low-power DDR1-SDRAM device are now enabled.
5. A pause of at least 200 µs must be observed before a signal toggle.
6. A NOP command is issued to the low-power DDR1-SDRAM. Program the NOP command in the MPDDRC_MR. The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM address to acknowledge this command. A calibration request is now made to the I/O pad.
7. An All Banks Precharge command is issued to the low-power DDR1-SDRAM. Program All Banks Precharge command in the MPDDRC_MR. The application must write a 2 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM address to acknowledge this command.
8. Two auto-refresh (CBR) cycles are provided. Program the Auto Refresh command (CBR) in the MPDDRC_MR. The application must write a 4 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM location twice to acknowledge these commands.
9. An Extended Mode Register Set (EMRS) cycle is issued to program the low-power DDR1-SDRAM parameters (TCSR, PASR, DS). The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 1 and BA[0] is set to 0. For example: with a 16-bit, 128-Mbit, low-power DDR1-SDRAM (12 rows, 9 columns, 4 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00800000; with a 32-bit, 1-Gbit, low-power DDR1-SDRAM (14 rows, 10 columns, 4 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x08000000. In the case of low-cost and low-density low-power DDR1-SDRAM (2 internal banks), the write address must be chosen so that signal BA[0] is set to 1. BA[1] is not used.

Note: This address is given as an example only. The real address depends on implementation in the product.

10. A Mode Register Set (MRS) cycle is issued to program parameters of the low-power DDR1-SDRAM devices, in particular CAS latency. The application must write a 3 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the low-power DDR1-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[1:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.

11. The application must enter Normal mode, write a zero to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access at any location in the low-power DDR1-SDRAM to acknowledge this command.
12. Write the refresh rate into the COUNT field in the MPDDRC Refresh Timer Register (MPDDRC_RTR): refresh rate = delay between refresh cycles. The low-power DDR1-SDRAM device requires a refresh every 15.625 µs or 7.81 µs. With a 100 MHz frequency, MPDDRC_RTR must be set with (15.625 × 100 MHz) = 1562 i.e., 0x061A or (7.81 × 100 MHz) = 781 i.e., 0x030D.

After initialization, the low-power DDR1-SDRAM device is fully functional.

28.4.2 DDR2-SDRAM Initialization

The initialization sequence is generated by software. The DDR2-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the MPDDRC Memory Device Register (MPDDRC_MD).
2. Program features of the DDR2-SDRAM device in the MPDDRC Configuration Register (number of columns, rows, banks, CAS latency and output driver impedance control) and in the MPDDRC Timing Parameter 0 Register/MPDDRC Timing Parameter 1 Register (asynchronous timing (TRC, TRAS, etc.).
3. A NOP command is issued to the DDR2-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command. The clocks which drive the DDR2-SDRAM device are now enabled.
4. A pause of at least 200 µs must be observed before a signal toggle.
5. A NOP command is issued to the DDR2-SDRAM. Program the NOP command in the MPDDRC_MR. The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command. CKE is now driven high.
6. An All Banks Precharge command is issued to the DDR2-SDRAM. Program All Banks Precharge command in the MPDDRC_MR. The application must write a 2 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command.
7. An Extended Mode Register Set (EMRS2) cycle is issued to choose between commercial or high temperature operations. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 1 and signal BA[0] is set to 0. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00800000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x08000000.

Note: This address is given as an example only. The real address depends on implementation in the product.

8. An Extended Mode Register Set (EMRS3) cycle is issued to set the Extended Mode Register to 0. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 1 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00C00000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x0C000000.
9. An Extended Mode Register Set (EMRS1) cycle is issued to enable DLL and to program D.I.C. (Output Driver Impedance Control). The application must write a 5 to the MODE field in the MPDDRC_MR. Read the

MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 0 and signal BA[0] is set to 1.For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00400000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.

10. An additional 200 cycles of clock are required for locking DLL
11. Write a one to the DLL bit (enable DLL reset) in the MPDDRC Configuration Register (MPDDRC_CR).
12. A Mode Register Set (MRS) cycle is issued to reset DLL. The application must write a 3 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[1:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
13. An All Banks Precharge command is issued to the DDR2-SDRAM. Program the All Banks Precharge command in the MPDDRC_MR. The application must write a 2 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command.
14. Two auto-refresh (CBR) cycles are provided. Program the Auto Refresh command (CBR) in the MPDDRC_MR. The application must write a 4 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM location twice to acknowledge these commands.
15. Write a zero to the DLL bit (disable DLL reset) in the MPDDRC_CR.
16. A Mode Register Set (MRS) cycle is issued to program parameters of the DDR2-SDRAM device, in particular CAS latency and to disable DLL reset. The application must write a 3 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[1:0] are set to 0.For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks) bank address, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
17. Write a seven to the OCD field (default OCD calibration) in the MPDDRC_CR.
18. An Extended Mode Register Set (EMRS1) cycle is issued to the default OCD value. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00400000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.
19. Write a zero to the OCD field (exit OCD calibration mode) in the MPDDRC_CR.
20. An Extended Mode Register Set (EMRS1) cycle is issued to enable OCD exit. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks) bank address, the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00400000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.

21. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command.
22. Write the refresh rate into the COUNT field in the MPDDRC Refresh Timer Register (MPDDRC_RTR): refresh rate = delay between refresh cycles. The DDR2-SDRAM device requires a refresh every 15.625 µs or 7.81 µs. With a 133 MHz frequency, the COUNT field in the MPDDRC_RTR must be set with (15.625 × 133 MHz) = 2079 i.e., 0x081F or (7.81 × 133 MHz) = 1039 i.e., 0x040F.

After initialization, the DDR2-SDRAM devices are fully functional.

28.4.3 Low-power DDR2-SDRAM Initialization

The initialization sequence is generated by software.

The low-power DDR2-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the MPDDRC Memory Device Register (MPDDRC_MD).
2. Program features of the low-power DDR2-SDRAM device into and in the MPDDRC Configuration Register (number of columns, rows, banks, CAS latency and output drive strength) and in the MPDDRC Timing Parameter 0 Register/MPDDRC Timing Parameter 0 Register (asynchronous timing, TRC, TRAS, etc.).
3. A NOP command is issued to the low-power DDR2-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The clocks which drive the Low-power DDR2-SDRAM devices are now enabled.
4. A pause of at least 100 ns must be observed before a signal toggle.
5. A NOP command is issued to the low-power DDR2-SDRAM. Program the NOP command in the MPDDRC_MR. The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. CKE is now driven high.
6. A pause of at least 200 µs must be observed before issuing a Reset command.
7. A Reset command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a 63 to the MRS field. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Reset command is now issued.
8. A pause of at least t_{INIT5} must be observed before issuing any commands.
9. A Calibration command is issued to the low-power DDR2-SDRAM. Program the type of calibration in the MPDDRC Configuration Register (MPDDRC_CR): set the ZQ field to the RESET value. In the MPDDRC_MR, configure the MODE field to the LPDDR2_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a 10 to the MRS field. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The ZQ Calibration command is now issued. Program the type of calibration in the MPDDRC_CR: set the ZQ field to the SHORT value.
10. A Mode Register Write command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a one to the MRS field. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
11. A Mode Register Write command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a two to the MRS field. The Mode Register Write command cycle is issued to program

parameters of the low-power DDR2-SDRAM device, in particular CAS latency. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.

12. A Mode Register Write command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a three to the MRS field. The Mode Register Write command cycle is issued to program parameters of the low-power DDR2-SDRAM device, in particular Drive Strength and Slew Rate. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
13. A Mode Register Write command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR configure the MODE field to the LPDDR2_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a 16 to the MRS field. Mode Register Write command cycle is issued to program parameters of the low-power DDR2-SDRAM device, in particular Partial Array Self Refresh (PASR). Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
14. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command.
15. In the DDR configuration Register (SFR_DDRCCFG), the application must write a 0 to fields 17 and 16 to close the input buffers. The buffers are then driven by the HMPDDRC controller.
16. Write the refresh rate into the COUNT field in the MPDDRC Refresh Timer Register (MPDDRC_RTR): refresh rate = delay between refresh cycles. The low-power DDR2-SDRAM device requires a refresh every 7.81 µs. With a 133 MHz frequency, the COUNT field in the MPDDRC_RTR must be set with (7.81 × 133 MHz) = 1039 i.e., 0x040F.

After initialization, the low-power DDR2-SDRAM devices are fully functional.

28.5 Functional Description

28.5.1 DDR-SDRAM Controller Write Cycle

The MPDDRC provides burst access or single access in Normal mode (MPDDRC_MR.MODE = 0). Whatever the access type, the MPDDRC keeps track of the active row in each bank, thus maximizing performance.

The DDR-SDRAM device is programmed with a burst length (bl) equal to 8. This determines the length of a sequential data input by the write command that is set to 8. The latency from write command to data input depends on the memory type, as shown in Table 28-1.

Table 28-1. CAS Write Latency

Memory Devices	CAS Write Latency (CWL)
Low-power DDR1-SDRAM	1
Low-power DDR2-SDRAM	1
DDR2-SDRAM	2/3/4/5
Low-power DDR3-SDRAM	1/3

To initiate a single access, the MPDDRC checks if the page access is already open. If row/bank addresses match with the previous row/bank addresses, the controller generates a write command. If the bank addresses are not identical or if bank addresses are identical but the row addresses are not identical, the controller generates a precharge command, activates the new row and initiates a write command. To comply with DDR-SDRAM timing

parameters, additional clock cycles are inserted between precharge/active (t_{RP}) commands and active/write (t_{RCD}) command. As the burst length is set to 8, in case of single access, it has to stop the burst, otherwise seven invalid values may be written. In case of the DDR-SDRAM device, the burst stop command is not supported for the burst write operation. Thus, in order to interrupt the write operation, the DM (data mask) input signal must be set to 1 to mask invalid data (refer to Figure 28-2 and Figure 28-4), and DQS must continue to toggle.

To initiate a burst access, the MPDDRC uses the transfer type signal provided by the master requesting the access. If the next access is a sequential write access, writing to the DDR-SDRAM device is carried out. If the next access is a write non-sequential access, then an automatic access break is inserted, the MPDDRC generates a precharge command, activates the new row and initiates a write command. To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active (t_{RP}) commands and active/write (t_{RCD}) commands.

For the definition of timing parameters, refer to Section 28.7.4 “MPDDRC Timing Parameter 0 Register”.

Write accesses to the DDR-SDRAM device are burst oriented and the burst length is programmed to 8. It determines the maximum number of column locations that can be accessed for a given write command. When the write command is issued, eight columns are selected. All accesses for that burst take place within these eight columns, thus the burst wraps within these eight columns if a boundary is reached. These eight columns are selected by addr[13:3]. addr[2:0] is used to select the starting location within the block.

In case of incrementing burst (INCR/INCR4/INCR8/INCR16), the addresses can cross the 16-byte boundary of the DDR-SDRAM device. For example, when a transfer (INCR4) starts at address 0x0C, the next access is 0x10, but since the burst length is programmed to 8, the next access is at 0x00. Since the boundary is reached, the burst is wrapped. The MPDDRC takes this feature of the DDR-SDRAM device into account. In case of a transfer starting at address 0x04/0x08/0x0C or starting at address 0x10/0x14/0x18/0x1C, two write commands are issued to avoid wrapping when the boundary is reached. The last write command is subject to DM input logic level. If DM is registered high, the corresponding data input is ignored and the write access is not done. This avoids additional writing.

Figure 28-2. Single Write Access, Row Closed, DDR-SDRAM Devices

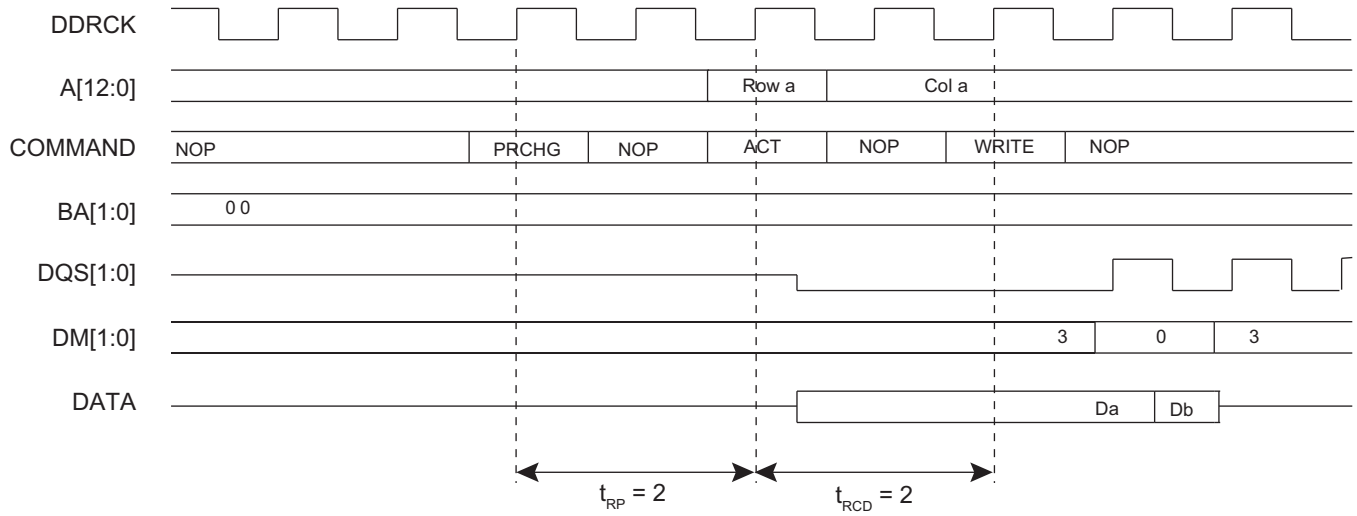


Figure 28-3. Single Write Access, Row Closed, DDR2-SDRAM Devices

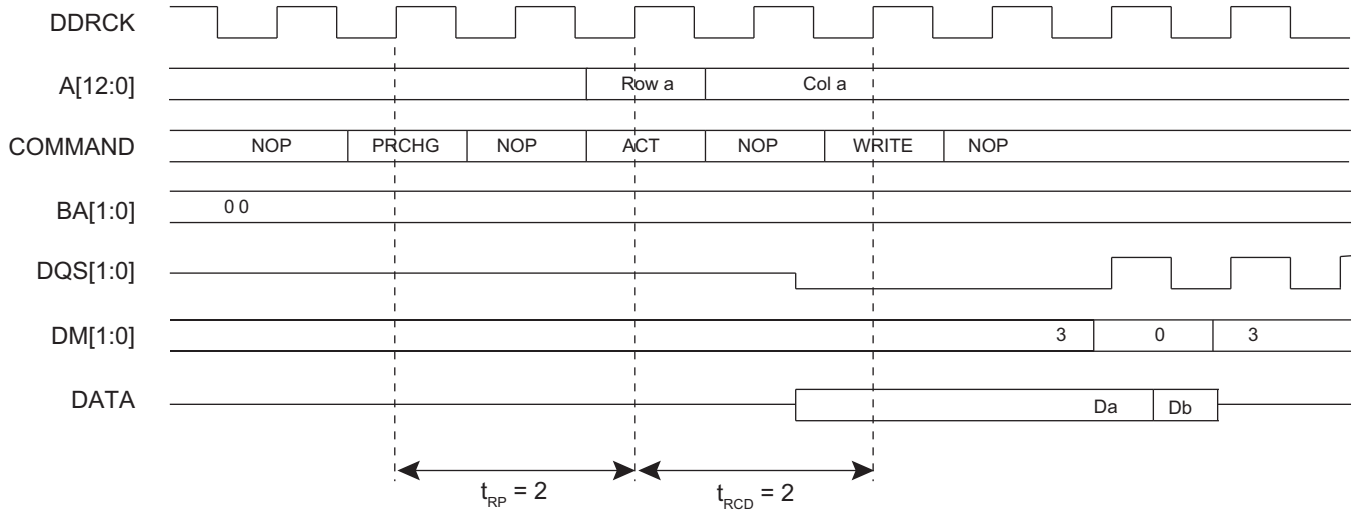


Figure 28-4. Burst Write Access, Row Closed, DDR-SDRAM Devices

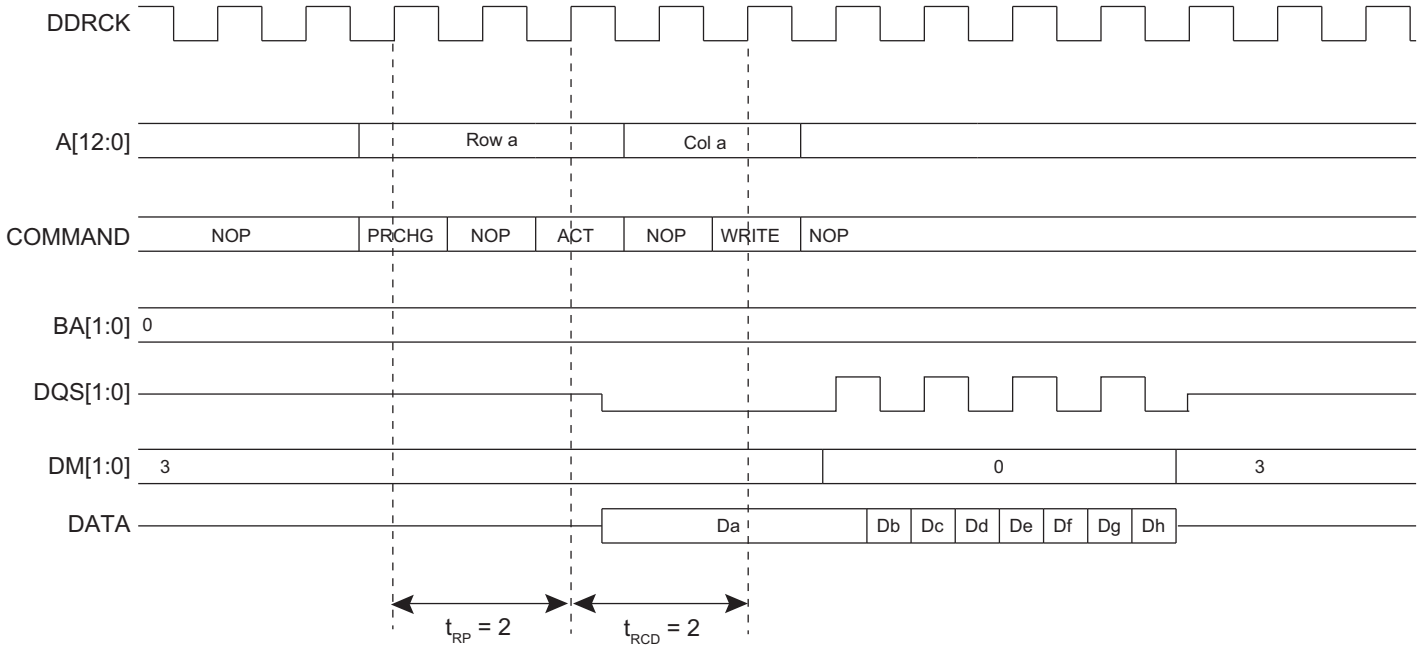
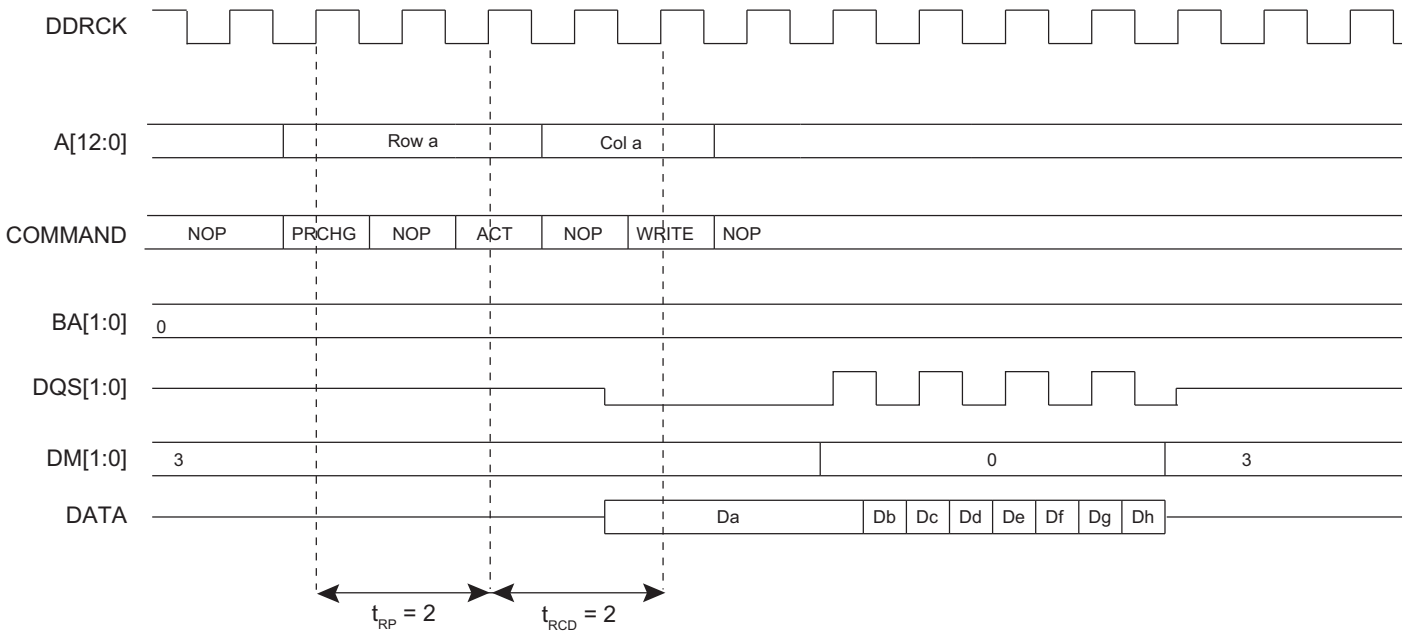
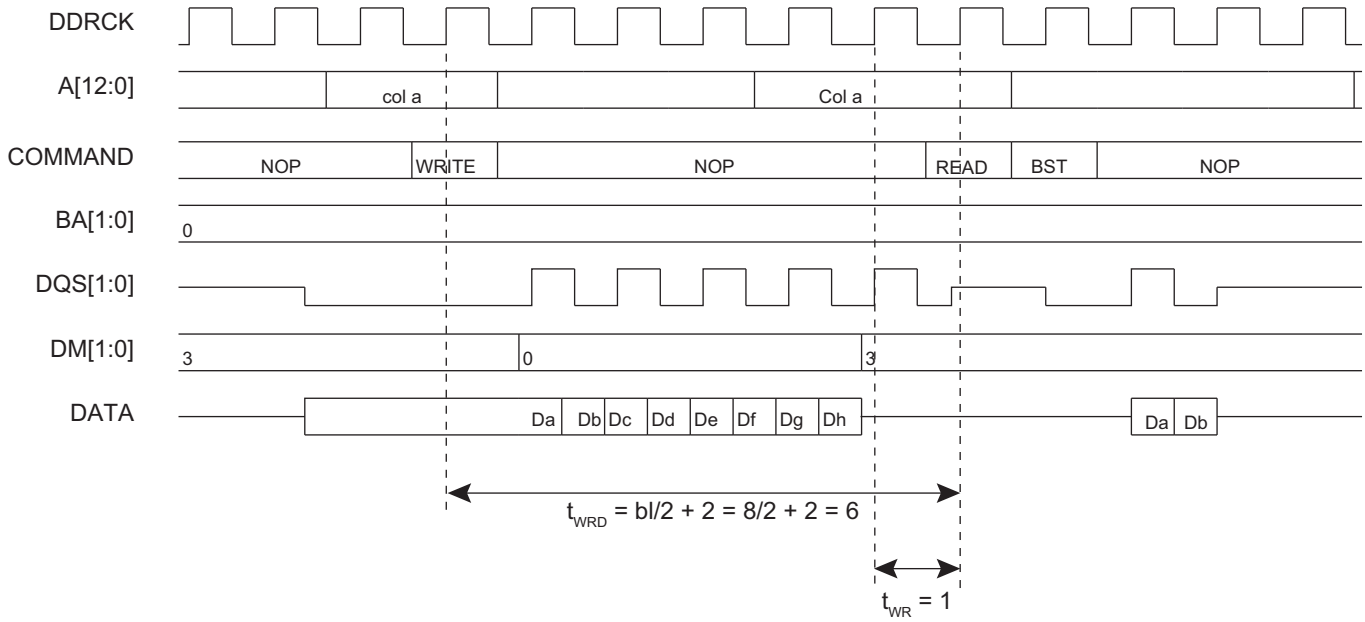


Figure 28-5. Burst Write Access, Row Closed, DDR2-SDRAM Devices



A write command can be followed by a read command. To avoid breaking the current write burst, t_{WTR}/t_{WRD} (bl/2 + 2 = 6 cycles) should be met. Refer to Figure 28-6.

Figure 28-6. Write Command Followed by a Read Command without Burst Write Interrupt, DDR-SDRAM Devices



In case of a single write access, write operation should be interrupted by a read access but DM must be input 1 cycle prior to the read command to avoid writing invalid data. Refer to Figure 28-7.

Figure 28-7. SINGLE Write Access Followed by a Read Access, DDR-SDRAM Devices

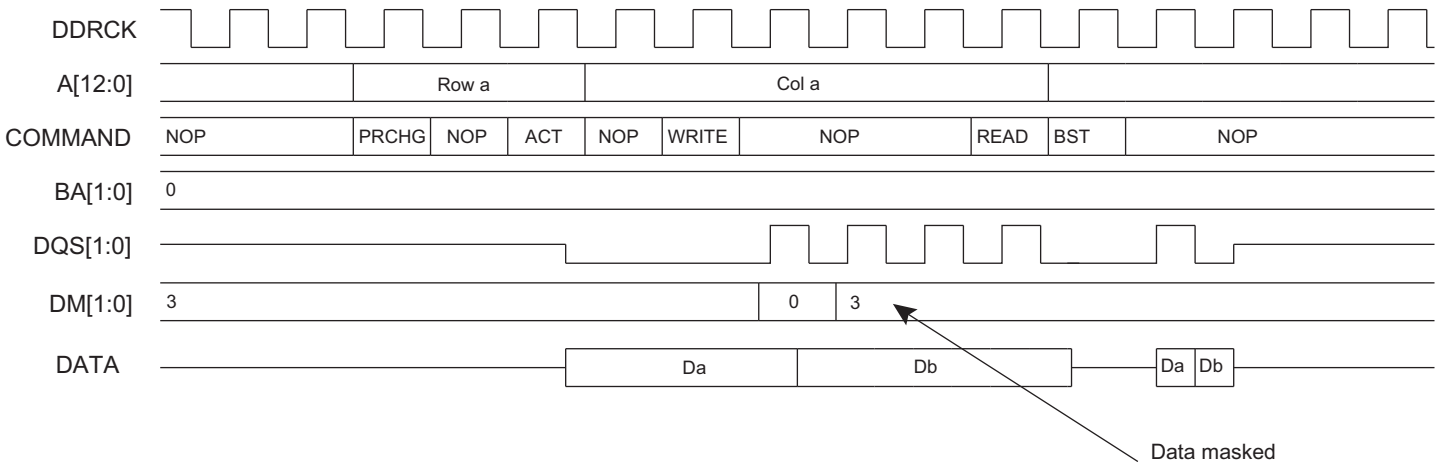
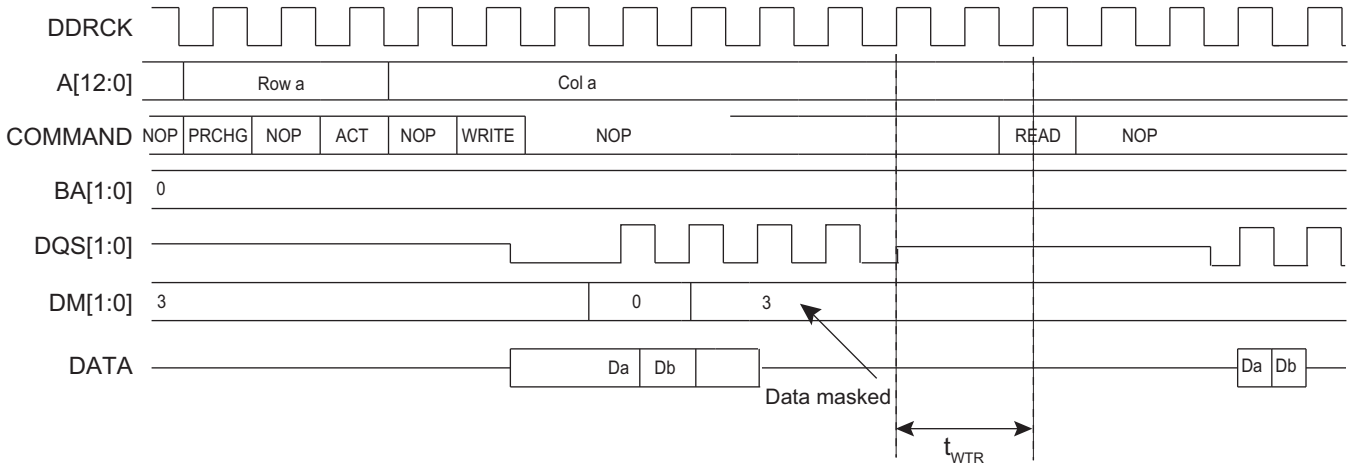


Figure 28-8. SINGLE Write Access Followed by a Read Access, DDR2-SDRAM Devices



28.5.2 DDR-SDRAM Controller Read Cycle

The MPDDRC provides burst access or single access in Normal mode (MPDDRC_MR.MODE = 0). Whatever the access type, the MPDDRC keeps track of the active row in each bank, thus maximizing performance of the MPDDRC.

The DDR-SDRAM devices are programmed with a burst length equal to 8 which determines the length of a sequential data output by the read command that is set to 8. The latency from read command to data output depends on the memory type, as shown in Table 28-2. This value is programmed during the initialization phase (refer to Section 28.4 “Product Dependencies, Initialization Sequence”).

Table 28-2. CAS Read Latency

Memory Devices	CAS Read Latency
Low-power DDR1-SDRAM	2/3
Low-power DDR2-SDRAM	3
DDR2-SDRAM	3/4/5/6
Low-power DDR3-SDRAM	3/6

To initiate a single access, the MPDDRC checks if the page access is already open. If row/bank addresses match with the previous row/bank addresses, the controller generates a read command. If the bank addresses are not identical or if bank addresses are identical but the row addresses are not identical, the controller generates a precharge command, activates the new row and initiates a read command. To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active (t_{RP}) commands and active/read (t_{RCD}) command. After a read command, additional wait states are generated to comply with CAS latency. The MPDDRC supports a CAS latency of two to six (2 to 6 clock cycle delay). As the burst length is set to 8, in case of a single access or a burst access inferior to 8 data requests, it has to stop the burst, otherwise an additional seven or X values could be read. The Burst Stop command (BST) is used to stop output during a burst read. If the DDR2-SDRAM Burst Stop command is not supported by the JEDEC standard, in a single read access, an additional seven unwanted data will be read.

To initiate a burst access, the MPDDRC checks the transfer type signal. If the next accesses are sequential read accesses, reading to the SDRAM device is carried out. If the next access is a read non-sequential access, then an automatic page break can be inserted. If the bank addresses are not identical or if bank addresses are identical but the row addresses are not identical, the controller generates a precharge command, activates the new row and initiates a read command. If page access is already open, a read command is generated.

To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active (t_{RP}) commands and active/read (t_{RCD}) commands. The MPDDRC supports a CAS latency of two to six (2 to 6 clocks delay). During this delay, the controller uses internal signals to anticipate the next access and improve the performance of the controller. Depending on the latency, the MPDDRC anticipates two to six read accesses. In case of burst of specified length, accesses are not anticipated, but if the burst is broken (border, Busy mode, etc.), the next access is treated as an incrementing burst of unspecified length, and depending on the latency, the MPDDRC anticipates two to six read accesses.

For the definition of timing parameters, refer to Section 28.7.3 “MPDDRC Configuration Register”.

Read accesses to the DDR-SDRAM are burst oriented and the burst length is programmed to 8. The burst length determines the maximum number of column locations that can be accessed for a given read command. When the read command is issued, eight columns are selected. All accesses for that burst take place within these eight columns, meaning that the burst wraps within these eight columns if the boundary is reached. These eight columns are selected by addr[13:3]; addr[2:0] is used to select the starting location within the block.

In case of incrementing burst (INCR/INCR4/INCR8/INCR16), the addresses can cross the 16-byte boundary of the DDR-SDRAM device. For example, when a transfer (INCR4) starts at address 0x0C, the next access is 0x10, but since the burst length is programmed to 8, the next access is 0x00. Since the boundary is reached, the burst wraps. The MPDDRC takes into account this feature of the SDRAM device. In case of the DDR-SDRAM device, transfers start at address 0x04/0x08/0x0C. Two read commands are issued to avoid wrapping when the boundary is reached. The last read command may generate additional reading (1 read cmd = 4 DDR words).

To avoid additional reading, it is possible to use the burst stop command to truncate the read burst and to decrease power consumption. The DDR2-SDRAM devices do not support the burst stop command.

Figure 28-9. Single Read Access, Row Closed, Latency = 2, DDR-SDRAM Devices

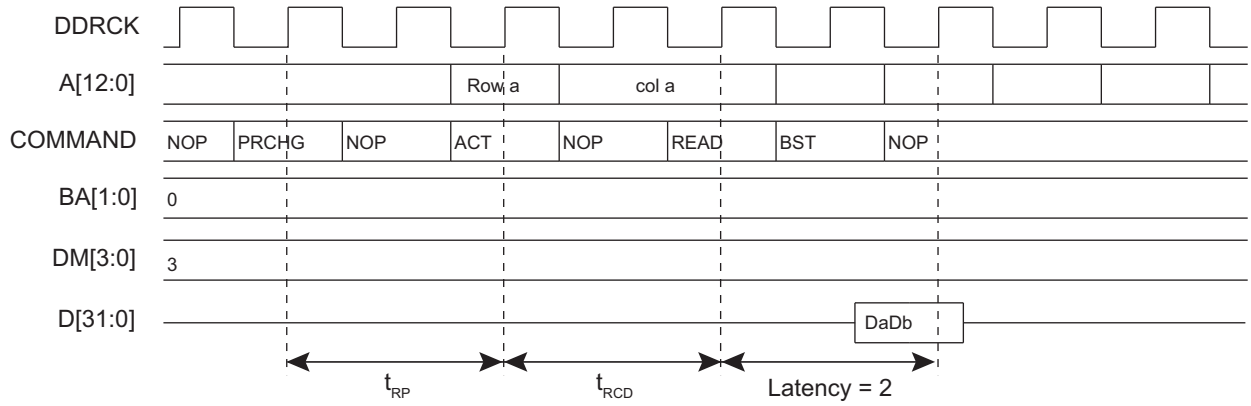


Figure 28-10. Single Read Access, Row Closed, Latency = 3, DDR2-SDRAM Devices

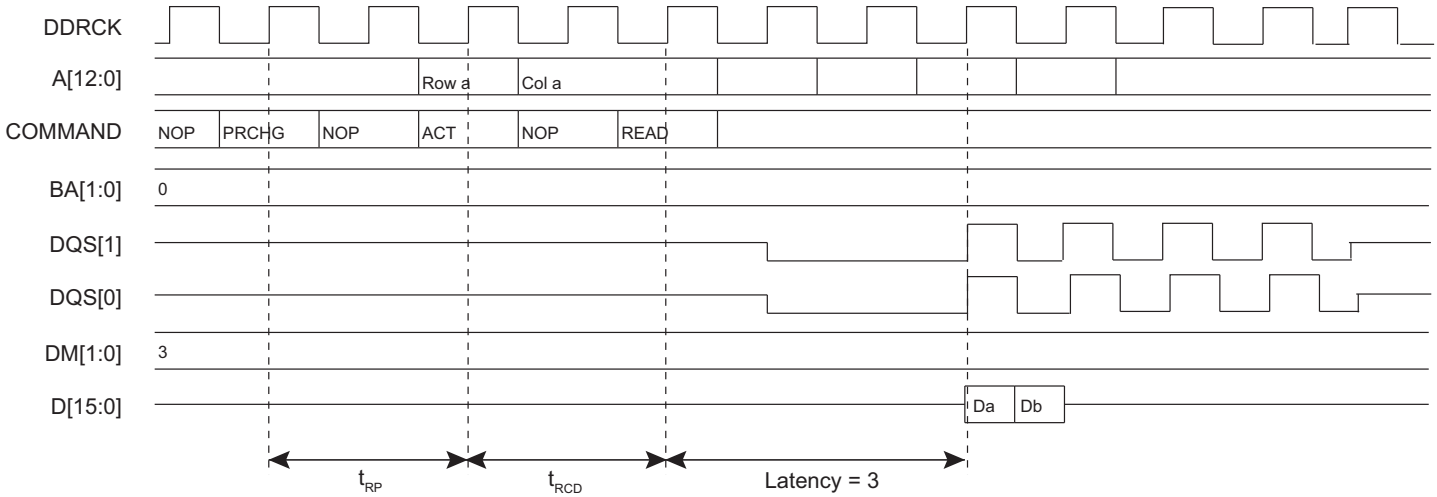


Figure 28-11. Burst Read Access, Latency = 2, DDR-SDRAM Devices

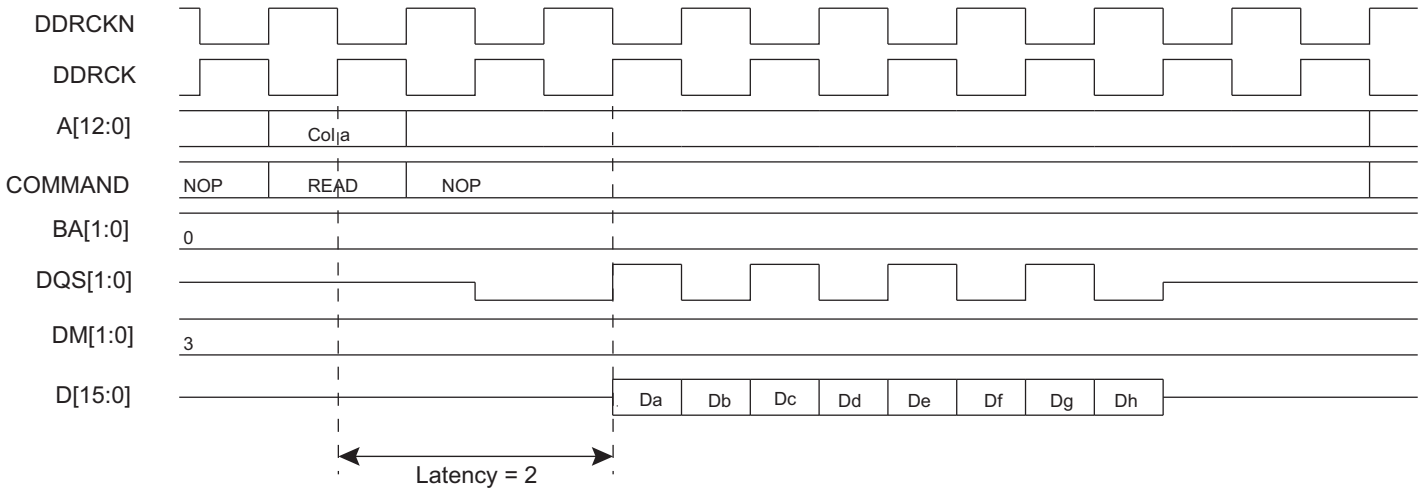
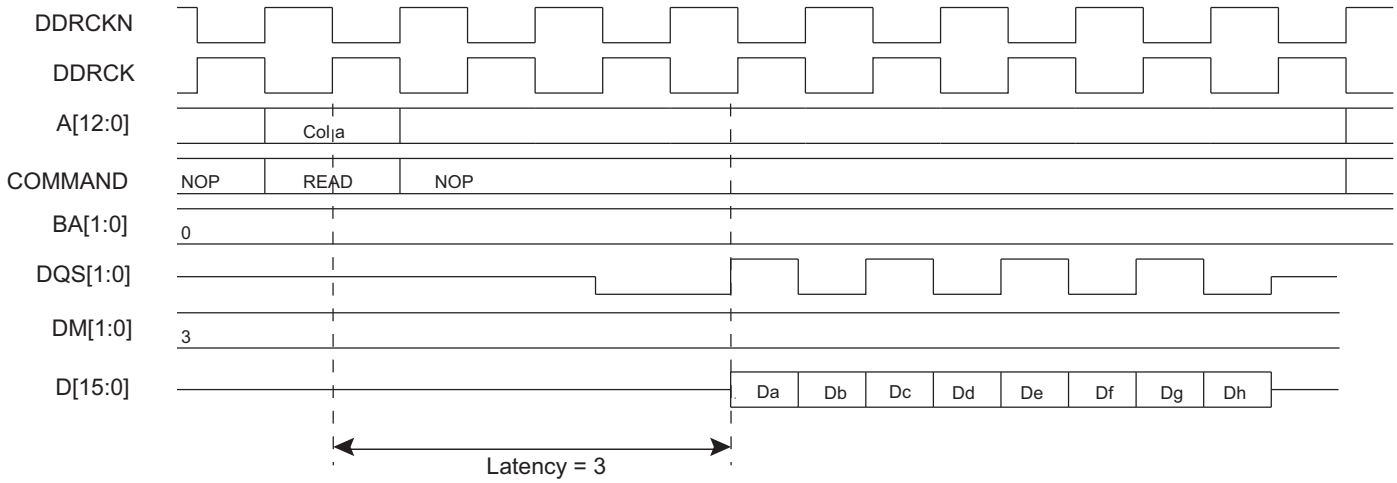


Figure 28-12. Burst Read Access, Latency = 3, DDR2-SDRAM Devices



28.5.2.1 All Banks Auto Refresh

The All Banks Auto Refresh command performs a refresh operation on all banks. An auto refresh command is used to refresh the external device. Refresh addresses are generated internally by the DDR-SDRAM device and incremented after each auto-refresh automatically. The MPDDRC generates these auto-refresh commands periodically. A timer is loaded in the MPDDRC_RTR with the value that indicates the number of clock cycles between refresh cycles (refer to Section 28.7.2 “MPDDRC Refresh Timer Register”). When the MPDDRC initiates a refresh of the DDR-SDRAM device, internal memory accesses are not delayed. However, if the CPU tries to access the DDR-SDRAM device, the slave indicates that the device is busy. A refresh request does not interrupt a burst transfer in progress. This feature is activated by setting Per-bank Refresh bit (REF_PB) to 0 in the MPDDRC_RTR (refer to Section 28.7.2 “MPDDRC Refresh Timer Register”).

28.5.2.2 Per-bank Auto Refresh

The low-power DDR2-SDRAM embeds a new Per-bank Refresh command which performs a refresh operation on the bank scheduled by the bank counter in the memory device. The Per-bank Refresh command is executed in a fixed sequence order of round-robin type: “0-1-2-3-4-5-6-7-0-1-...”. The bank counter is automatically cleared upon issuing a RESET command or when exiting from Self-refresh mode, in order to ensure the synchronism between SDRAM memory device and the MPDDRC. The bank addressing for the Per-bank Refresh count is the same as established in the Single-bank Precharge command. This feature is activated by setting the Per-bank Refresh bit (REF_PB) to 1 in the MPDDRC_RTR (refer to Section 28.7.2 “MPDDRC Refresh Timer Register”). This feature masks the latency due to the refresh procedure. The target bank is inaccessible during the Per-bank Refresh cycle period (t_{RFCpb}), however other banks within the device are accessible and may be addressed during the “Per-bank Refresh” cycle. During the REFpb operation, any bank other than the one being refreshed can be maintained in active state or accessed by a read or a write command. When the “Per-bank Refresh” cycle is completed, the affected bank will be in idle state.

28.5.2.3 Adjust Auto Refresh Rate

The low-power DDR2-SDRAM embeds an internal register, Mode Register 19 (Refresh mode). The content of this register allows to adjust the interval of auto-refresh operations according to temperature variation. This feature is activated by setting the Adjust Refresh bit [ADJ_REF] to 1 in the MPDDRC_RTR (refer to Section 28.7.2 “MPDDRC Refresh Timer Register”). When this feature is enabled, a Mode Register Read (MRR) command is performed every $16 \times t_{REFI}$ (average time between REFRESH commands). Depending on the read value, the auto refresh interval will be modified. In case of high temperature, the interval is reduced and in case of low temperature, the interval is increased.

28.5.3 Power Management

28.5.3.1 Self-refresh Mode

This mode is activated by writing a 1 to the Low-power Command bit (LPCB) in the MPDDRC Low-power Register (MPDDRC_LPR).

Self-refresh mode is used in Powerdown mode, i.e., when no access to the DDR-SDRAM device is possible. In this case, power consumption is very low. In Self-refresh mode, the DDR-SDRAM device retains data without external clocking and provides its own internal clocking, thus performing its own auto refresh cycles. During the self-refresh period, CKE is driven low. As soon as the DDR-SDRAM device is selected, the MPDDRC provides a sequence of commands and exits Self-refresh mode.

The MPDDRC re-enables Self-refresh mode as soon as the DDR-SDRAM device is not selected. It is possible to define when Self-refresh mode is to be enabled by configuring the TIMEOUT field in the MPDDRC_LPR:

0: Self-refresh mode is enabled as soon as the DDR-SDRAM device is not selected.

1: Self-refresh mode is enabled 64 clock cycles after completion of the last access.

2: Self-refresh mode is enabled 128 clock cycles after completion of the last access.

This controller also interfaces the low-power DDR-SDRAM. To optimize power consumption, the Low Power DDR SDRAM provides programmable self-refresh options comprised of Partial Array Self Refresh (full, half, quarter and 1/8 and 1/16 array).

Disabled banks are not refreshed in Self-refresh mode. This feature permits to reduce the self-refresh current. In case of low-power DDR1-SDRAM, the Extended Mode register controls this feature. It includes Temperature Compensated Self-refresh (TSCR) and Partial Array Self-refresh (PASR) parameters and the drive strength (DS) (refer to Section 28.7.7 “MPDDRC Low-power Register”). In case of low-power DDR2-SDRAM, the Mode Registers 16 and 17 control this feature, including PASR Bank Mask (BK_MASK) and PASR Segment Mask (SEG_MASK) parameters and drives strength (DS) (refer to Section 28.7.9 “MPDDRC Low-power DDR2 Low-power Register”). These parameters are set during the initialization phase. After initialization, as soon as the PASR/DS/TCSR fields or BK_MASK/SEG_MASK/DS are modified, the memory device Extended Mode Register or Mode Registers 3/16/17 are automatically accessed. Thus if MPDDRC does not share an external bus with another controller, PASR/DS/TCSR and BK_MASK/SEG_MASK/DS bits are updated before entering Self-refresh mode or during a refresh command. If MPDDRC does share an external bus with another controller, PASR/DS/TCSR and BK_MASK/SEG_MASK/DS bits are also updated during a pending read or write access. This type of update depends on the UPD_MR bit (refer to Section 28.7.7 “MPDDRC Low-power Register”).

The low-power DDR1-SDRAM must remain in Self-refresh mode during the minimum of TRFC periods (refer to Section 28.7.5 “MPDDRC Timing Parameter 1 Register”), and may remain in Self-refresh mode for an indefinite period.

The DDR2-SDRAM must remain in Self-refresh mode during the minimum of t_{CKE} periods (refer to the memory device datasheet), and may remain in Self-refresh mode for an indefinite period.

The low-power DDR2-SDRAM must remain in Self-refresh mode for the minimum of t_{CKESR} periods (refer to the memory device datasheet) and may remain in Self-refresh mode for an indefinite period.

Figure 28-13. Self-refresh Mode Entry, TIMEOUT = 0

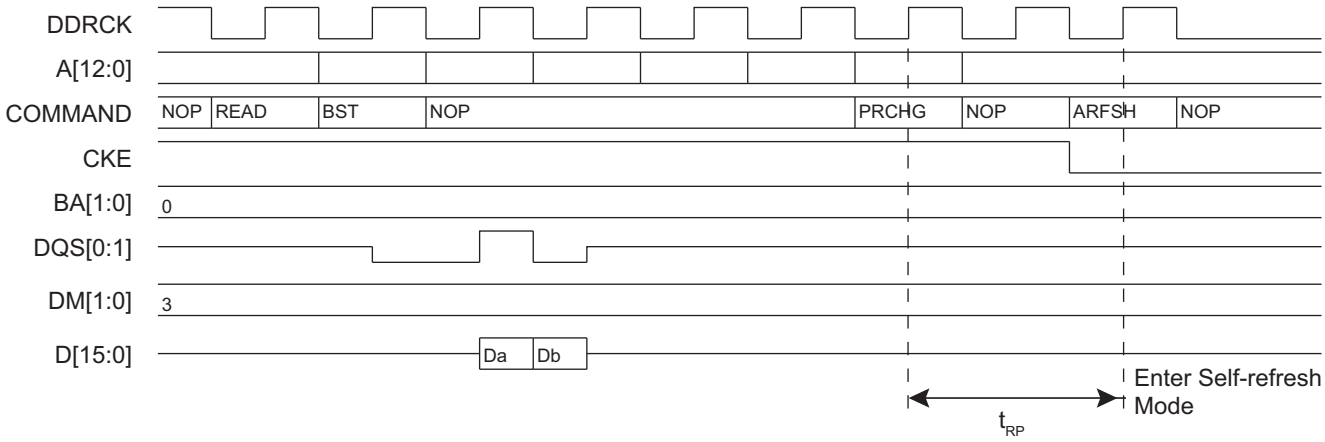


Figure 28-14. Self-refresh Mode Entry, TIMEOUT = 1 or 2

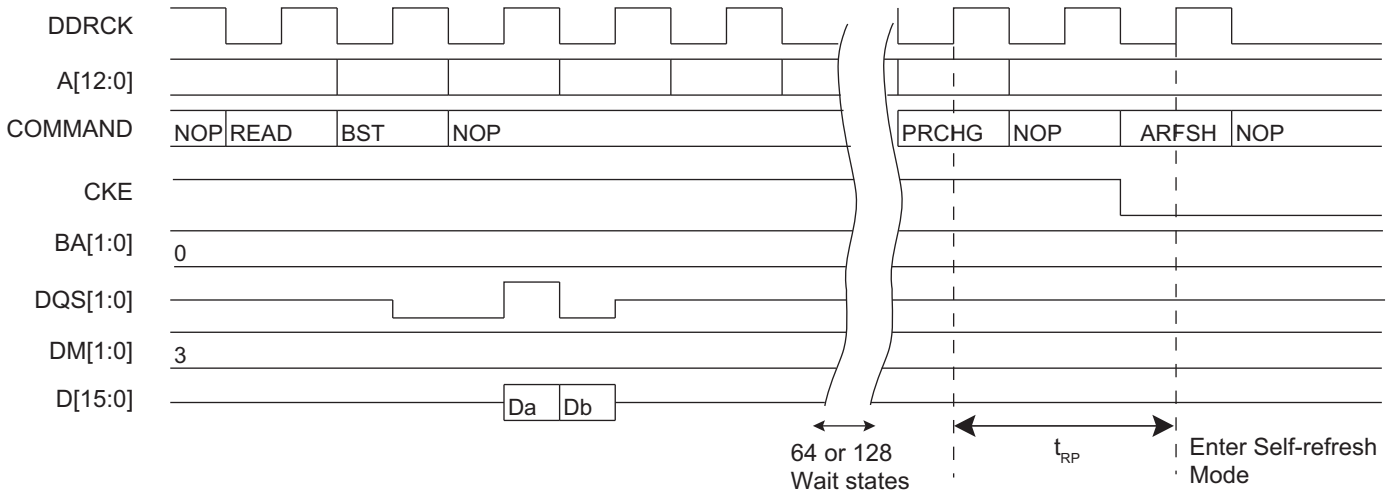
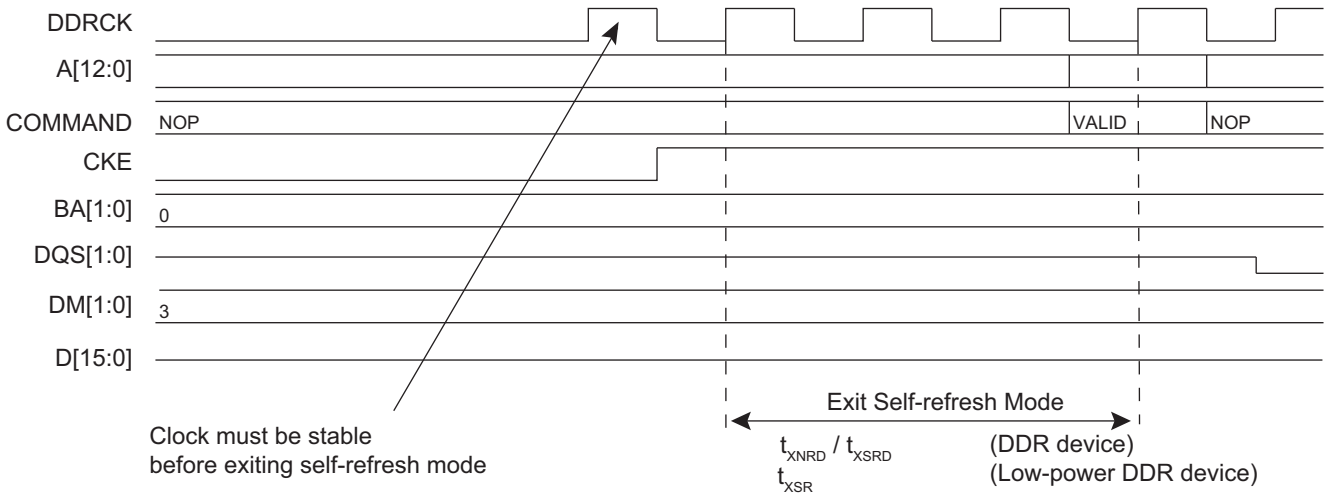


Figure 28-15. Self-refresh Mode Exit



28.5.3.2 Powerdown Mode

This mode is activated by writing a 10 to the Low-power Command bit (LPCB).

Powerdown mode is used when no access to the DDR-SDRAM device is possible. In this mode, power consumption is greater than in Self-refresh mode. This state is similar to Normal mode (no Low-power mode/no Self-refresh mode), but the CKE pin is low and the input and output buffers are deactivated as soon the DDR-SDRAM device is no longer accessible. In contrast to Self-refresh mode, the DDR-SDRAM device cannot remain in Low-power mode longer than one refresh period (64 ms/32 ms). As no auto-refresh operations are performed in this mode, the MPDDRC carries out the refresh operation. For the low-power DDR-SDRAM devices, a NOP command must be generated for a minimum period defined in the TXP field of the MPDDRC Timing Parameter 1 Register (MPDDRC_TPR1). For DDR-SDRAM devices, a NOP command must be generated for a minimum period defined in the TXP field of MPDDRC_TPR1 (refer to Section 28.7.5 “MPDDRC Timing Parameter 1 Register”) and in the TXARD and TXARDS fields of MPDDRC_TPR2 (refer to Section 28.7.6 “MPDDRC Timing Parameter 2 Register”) for DDR2_SDRAM devices. In addition, low-power DDR-SDRAM and DDR-SDRAM must remain in Powerdown mode for a minimum period corresponding to t_{CKE}, t_{PD}, etc. (refer to the memory device datasheet).

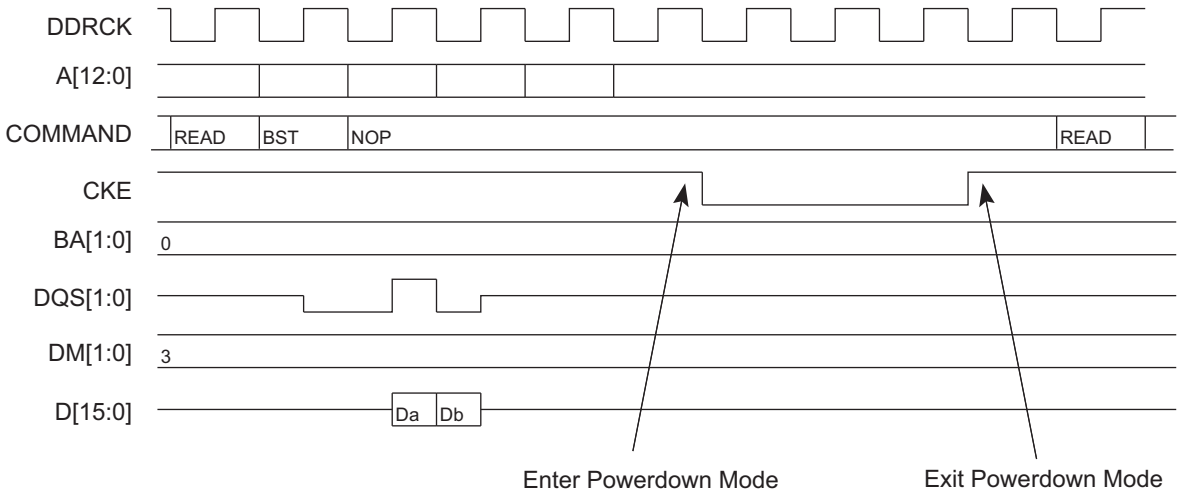
The exit procedure is faster than in Self-refresh mode. Refer to Figure 28-16. The MPDDRC returns to Powerdown mode as soon as the DDR-SDRAM device is not selected. It is possible to define when Powerdown mode is enabled by configuring the TIMEOUT field in the MPDDRC_LPR:

0: Powerdown mode is enabled as soon as the DDR-SDRAM device is not selected.

1: Powerdown mode is enabled 64 clock cycles after completion of the last access.

2: Powerdown mode is enabled 128 clock cycles after completion of the last access.

Figure 28-16. Powerdown Entry/Exit, TIMEOUT = 0



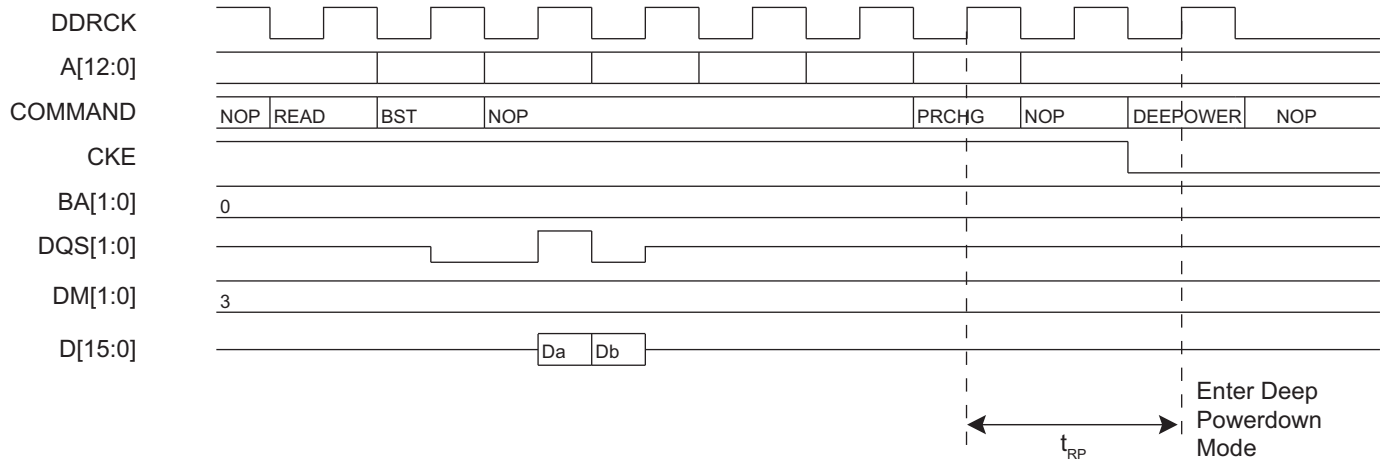
28.5.3.3 Deep Powerdown Mode

The Deep Powerdown mode is a feature of low-power DDR-SDRAM. When this mode is activated, all internal voltage generators inside the device are stopped and all data is lost.

Deep Powerdown mode is activated by writing a 3 to the Low-power Command bit (LPCB). When this mode is enabled, the MPDDRC leaves Normal mode (MPDDRC_MR.MODE = 0) and the controller is frozen. The clock can be stopped during Deep Powerdown mode by setting the CLK_FR field to 1.

Before enabling this mode, the user must make sure there is no access in progress. To exit Deep Powerdown mode, the Low-power Command bit (LPCB) and clock frozen bit (CLK_FR) must be written to zero and the initialization sequence must be generated by software. Refer to Section 28.4.1 “Low-power DDR1-SDRAM Initialization” or Section 28.4.3 “Low-power DDR2-SDRAM Initialization”.

Figure 28-17. Deep Powerdown Mode Entry



28.5.3.4 Reset Mode

The Reset mode is a feature of DDR2-SDRAM. This mode is activated by writing a 3 to the Low-power Command bit (LPCB) and a one to the Clock Frozen Command bit (CLK_FR) in the MPDDRC Low-power Register.

When this mode is enabled, the MPDDRC leaves Normal mode (MPDDRC_MR.MODE = 0) and the controller is frozen. Before enabling this mode, the user must make sure there is no access in progress.

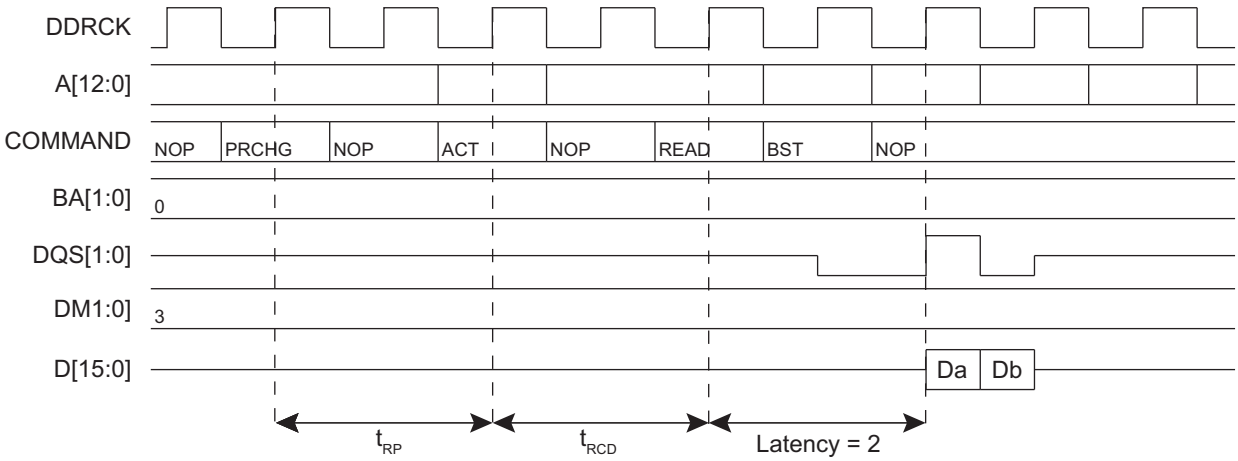
To exit Reset mode, the Low-power Command bit (LPCB) must be written to zero, the Clock Frozen Command bit (CLK_FR) must be written to zero and the initialization sequence must be generated by software (refer to Section 28.4.2 “DDR2-SDRAM Initialization”).

28.5.4 Multiport Functionality

The DDR-SDRAM protocol imposes a check of timings prior to performing a read or a write access, thus decreasing system performance. An access to DDR-SDRAM is performed if banks and rows are open (or active). To activate a row in a particular bank, the last open row must be deactivated and a new row must be open. Two DDR-SDRAM commands must be performed to open a bank: Precharge command and Activate command with respect to t_{RP} timing. Before performing a read or write command, t_{RCD} timing must be checked.

This operation generates a significant bandwidth loss (refer to Figure 28-18).

Figure 28-18. t_{RP} and t_{RCD} Timings

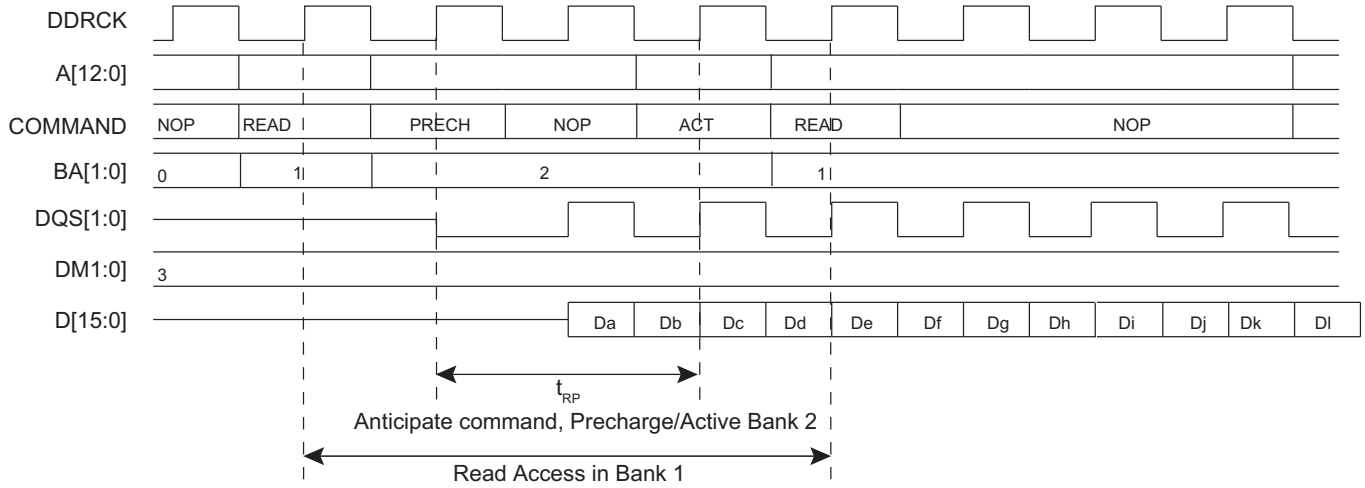


4 cycles before performing a read command

The multiport controller is designed to mask these timings and thus improve the bandwidth of the system.

The MPDDRC is a multiport controller whereby eight masters can simultaneously reach the controller. This feature improves the bandwidth of the system because it can detect eight requests on the AHB slave inputs and thus anticipate the commands that follow, Precharge and Activate command in bank X during the current access in bank Y. This masks t_{RP} and t_{RCD} timings (refer to Figure 28-19). In the best case, all accesses are done as if the banks and rows were already open. The best condition is met when the eight masters work in different banks. In case of eight simultaneous read accesses, when the four or eight banks and associated rows are open, the controller reads with a continuous flow and masks the CAS latency for each access. To allow a continuous flow, the read command must be set at 2 or 6 cycles (CAS latency) before the end of the current access. This requires that the scheme of arbitration changes since arbitration cannot be respected. If the controller anticipates a read access, and thus a master with a high priority arises before the end of the current access, then this master will not be serviced.

Figure 28-19. Anticipate Precharge/Activate Command in Bank 2 during Read Access in Bank 1



MPDDRC is a multiport controller that embeds three arbitration mechanisms based on round -robin arbitration which allows to share the external device between different masters when two or more masters try to access the DDR-SDRAM device at the same time.

The three arbitration types are round-robin arbitration and two weighted round-robin arbitrations. For weighted round-robin arbitrations, the priority can be given either depending on the number of requests or words per port, or depending on the required bandwidth per port. The type of arbitration can be chosen by setting the ARB field in the MPDDRC Configuration Arbiter Register (MPDDRC_CONF_ARBITER) (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

28.5.4.1 Round-robin Arbitration

Round-robin arbitration is used when the ARB field is set to 0 (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”). This algorithm dispatches the requests from different masters to the DDR-SDRAM device in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is serviced first, then the others are serviced in a round-robin manner.

To avoid burst breaking and to provide the maximum throughput for the DDR-SDRAM device, arbitration must only take place during the following cycles:

1. Idle cycles: when no master is connected to the DDR-SDRAM device.
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 bursts of defined length, predicted end of burst matches the size of the transfer.

- For bursts of undefined length, predicted end of burst is generated at the end of each four-beat boundary inside the INCR transfer.

4. Anticipated Access: when an anticipated read access is done while the current access is not complete, the arbitration scheme can be changed if the anticipated access is not the next access serviced by the arbitration scheme.

28.5.4.2 Request-word Weighted Round-robin Arbitration

In request-word weighted round-robin arbitration, the weight is the number of requests or the number of words per port.

This arbitration scheme is enabled by writing a 1 to the ARB field (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”). This algorithm grants a port for X(1) consecutive first transfer (htrans = NON SEQUENTIAL) of a burst or X single transfer, or for X word transfers. It is possible to chose between an arbitration scheme by request or by word per port by setting the RQ_WD_Px field (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

Note: 1. X is an integer value provided by some master modules to the arbiter.

It is also possible for the user to provide the number of requests or words (by overwriting the information provided by a master) on master basis by configuring the MA_PR_Px field. Depending on the application, it is possible to reduce or increase the number of these requests or words by configuring the NRD_NWD_BDW_Px fields (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

The TIMEOUT_Px field defines the delay between two accesses on the same port in number of cycles before to arbitrate and to give the hand to another port. This field allows to avoid a timeout on the system because some masters have the particularity to add idle cycles between two consecutive accesses (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

This algorithm dispatches the requests from different masters to the DDR-SDRAM device in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is serviced first, then the others are serviced in a round-robin manner when the number of requests or words is reached or when the timeout value is reached.

To avoid burst breaking and to provide the maximum throughput for the DDR-SDRAM device, arbitration must only take place during the following cycles:

1. Timeout is reached: the delay between two accesses is equal to TIMEOUT_Px.
2. Number of requests or words is reached: when the current cycle is the last cycle of a transfer.

28.5.4.3 Bandwidth Weighted Round-robin Arbitration

In bandwidth weighted round-robin arbitration, a minimum bandwidth is guaranteed per port.

This arbitration scheme is enabled when the ARB field is set to 2 (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

This algorithm grants to each port a percentage of the bandwidth. The NRD_NWD_BDW_Px field defines the percentage allocated to each port.

The percentage of the bandwidth is programmed with the NRD_NWD_BDW_Px fields (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

The TIMEOUT_Px field defines the delay between two accesses on the same port in number of cycles before to arbitrate and to give the hand to another port. This field allows to avoid a timeout on the system because some masters have the particularity to add idle cycles between two consecutive accesses (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

This algorithm dispatches the requests from different masters to the DDR-SDRAM device in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is serviced first, then the others are serviced in a round-robin manner when the allocated bandwidth is reached or when the timeout value is reached.

The BDW_BURST field allows to arbitrate either when the current master reaches exactly the programmed bandwidth, or when the current master reaches exactly the programmed bandwidth and the current access is ended (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

To provide the maximum throughput for the DDR-SDRAM device, arbitration must only take place during the following cycles:

1. Timeout is reached: the delay between two accesses is equal to TIMEOUT_Px.
2. Allocated Bandwidth is reach although the current cycle is not ended.
3. Allocated Bandwidth is reach and the current cycle is the last cycle of a transfer

28.5.5 Scrambling/Unscrambling Function

The external data bus 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 memory device.

The scrambling and unscrambling are performed on-the-fly without additional wait states.

The scrambling method depends on two user-configurable key registers, MPDDRC_KEY1 in the “MPDDRC OCMS KEY1 Register” and MPDDRC_KEY2 in the “MPDDRC OCMS KEY2 Register”. These key registers are only accessible in Write mode.

The key must be securely stored in a reliable non-volatile 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.

The scrambling/unscrambling function can be enabled or disabled by programming the “MPDDRC OCMS Register”.

28.5.6 Register Write Protection

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

If a write access to a write-protected register is detected, the WPVS flag in the MPDDRC Write Protection Status Register (MPDDRC_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 MPDDRC_WPSR.

The following registers can be write-protected:

- MPDDRC Mode Register
- MPDDRC Refresh Timer Register
- MPDDRC Configuration Register
- MPDDRC Timing Parameter 0 Register
- MPDDRC Timing Parameter 1 Register
- MPDDRC Memory Device Register
- MPDDRC Low-power DDR2 Calibration and MR4 Register
- MPDDRC OCMS Register
- MPDDRC OCMS KEY1 Register
- MPDDRC OCMS KEY2 Register

28.6 Software Interface/SDRAM Organization, Address Mapping

The DDR-SDRAM address space is organized into banks, rows and columns. The MPDDRC maps different memory types depending on values set in the MPDDRC Configuration Register (refer to Section 28.7.3 “MPDDRC Configuration Register”). The tables that follow illustrate the relation between CPU addresses and columns, rows and banks addresses for 32-bit memory data bus widths.

The MPDDRC supports address mapping in Linear mode.

Sequential mode is a method for address mapping where banks alternate at each last DDR-SDRAM page of the current bank.

Interleaved mode is a method for address mapping where banks alternate at each DDR-SDRAM end of page of the current bank.

The MPDDRC makes the DDR-SDRAM device access protocol transparent to the user. The tables that follow illustrate the DDR-SDRAM device memory mapping seen by the user in correlation with the device structure. Various configurations are illustrated.

28.6.1 DDR-SDRAM Address Mapping for 16-bit Memory Data Bus Width

Table 28-3. Sequential Mapping for DDR-SDRAM Configuration, 2K Rows, 512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
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[8:0]									M0
				Bk[1:0]			Row[10:0]											Column[9:0]									M0
			Bk[1:0]				Row[10:0]											Column[10:0]									M0
		Bk[1:0]					Row[10:0]											Column[11:0]									M0

Table 28-4. Interleaved Mapping for DDR-SDRAM Configuration, 2K Rows, 512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
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**
						Row[10:0]										Bk[1:0]		Column[8:0]									M0
					Row[10:0]										Bk[1:0]			Column[9:0]									M0
				Row[10:0]										Bk[1:0]				Column[10:0]									M0
			Row[10:0]										Bk[1:0]					Column[11:0]									M0

Table 28-5. Sequential Mapping for DDR-SDRAM Configuration: 4K Rows, 512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
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[8:0]									M0
			Bk[1:0]				Row[11:0]											Column[9:0]									M0
		Bk[1:0]					Row[11:0]											Column[10:0]									M0
	Bk[1:0]						Row[11:0]											Column[11:0]									M0

Table 28-6. Interleaved Mapping for DDR-SDRAM Configuration: 4K Rows, 512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
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**
						Row[11:0]										Bk[1:0]		Column[8:0]									M0
					Row[11:0]										Bk[1:0]			Column[9:0]									M0
				Row[11:0]										Bk[1:0]				Column[10:0]									M0
			Row[11:0]										Bk[1:0]					Column[11:0]									M0

Table 28-7. Sequential Mapping for DDR-SDRAM Configuration: 8K Rows, 512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
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[8:0]									M0
		Bk[1:0]					Row[12:0]											Column[9:0]									M0
	Bk[1:0]						Row[12:0]											Column[10:0]									M0
Bk[1:0]							Row[12:0]											Column[11:0]									M0

Table 28-8. Interleaved Mapping for DDR-SDRAM Configuration: 8K Rows, 512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
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**
			Row[12:0]													Bk[1:0]		Column[8:0]									M0
		Row[12:0]													Bk[1:0]		Column[9:0]										M0
	Row[12:0]													Bk[1:0]		Column[10:0]											M0
Row[12:0]													Bk[1:0]		Column[11:0]												M0

Table 28-9. Sequential Mapping for DDR-SDRAM Configuration: 16K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																										
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[8:0]									M0
	Bk[1:0]		Row[13:0]															Column[9:0]									M0
Bk[1:0]		Row[13:0]														Column[10:0]											M0

Table 28-10. Interleaved Mapping for DDR-SDRAM Configuration: 16K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																										
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**
		Row[13:0]														Bk[1:0]		Column[8:0]									M0
	Row[13:0]														Bk[1:0]			Column[9:0]									M0
Row[13:0]														Bk[1:0]			Column[10:0]										M0

Table 28-11. Sequential Mapping for DDR-SDRAM Configuration: 8K Rows,1024/ Columns, 8 banks

CPU Address Line																										
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[2:0]			Row[12:0]													Column[9:0]										M0

Table 28-12. Interleaved Mapping for DDR-SDRAM Configuration: 8K Rows,1024/ Columns, 8 banks

CPU Address Line																										
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**
	Row[12:0]													Bk[2:0]			Column[9:0]										M0

Table 28-13. Sequential Mapping for DDR-SDRAM Configuration: 16K Rows,1024/ Columns, 8 banks

CPU Address Line																										
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[2:0]			Row[13:0]														Column[9:0]										M0

Table 28-14. Interleaved Mapping for DDR-SDRAM Configuration: 16K Rows,1024/ Columns, 8 banks

CPU Address Line																										
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**
Row[13:0]														Bk[2:0]			Column[9:0]										M0

28.6.2 DDR-SDRAM Address Mapping for 32-bit Memory Data Bus Width

Table 28-15. Sequential Mapping DDR-SDRAM Configuration Mapping: 2K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																											
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[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]

Table 28-16. Interleaved Mapping DDR-SDRAM Configuration Mapping: 2K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																											
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**
					Row[10:0]											Bk[1:0]		Column[8:0]										M[1:0]
				Row[10:0]											Bk[1:0]			Column[9:0]										M[1:0]
			Row[10:0]											Bk[1:0]				Column[10:0]										M[1:0]

Table 28-17. Sequential Mapping DDR-SDRAM Configuration Mapping: 4K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																											
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[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]

Table 28-18. Interleaved Mapping DDR-SDRAM Configuration Mapping: 4K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																											
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**
				Row[11:0]												Bk[1:0]		Column[8:0]										M[1:0]
			Row[11:0]												Bk[1:0]			Column[9:0]										M[1:0]
		Row[11:0]												Bk[1:0]				Column[10:0]										M[1:0]

Table 28-19. Sequential Mapping DDR-SDRAM Configuration Mapping: 8K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																											
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[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]

Table 28-20. Interleaved Mapping DDR-SDRAM Configuration Mapping: 8K Rows /512/1024/2048 Columns, 4 banks

CPU Address Line																											
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**
					Row[13:0]											Bk[1:0]		Column[8:0]										M[1:0]
			Row[13:0]												Bk[1:0]			Column[9:0]										M[1:0]
		Row[13:0]												Bk[1:0]				Column[10:0]										M[1:0]

Table 28-21. Sequential Mapping DDR-SDRAM Configuration Mapping: 8K Rows /1024 Columns, 8 banks

CPU Address Line																											
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[2:0]			Row[12:0]													Column[9:0]										M[1:0]	

Table 28-22. Interleaved Mapping DDR-SDRAM Configuration Mapping: 8K Rows /1024 Columns, 8 banks

CPU Address Line																											
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**
	Row[12:0]													Bk[2:0]			Column[9:0]										M[1:0]	

Table 28-23. Sequential Mapping DDR-SDRAM Configuration Mapping: 16K Rows /1024 Columns, 4 banks

CPU Address Line																											
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[9:0]										M[1:0]	

Table 28-24. Interleaved Mapping DDR-SDRAM Configuration Mapping: 16K Rows /1024 Columns, 4 banks

CPU Address Line																											
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**
	Row[13:0]														Bk[1:0]		Column[9:0]										M[1:0]	

Table 28-25. Sequential Mapping DDR-SDRAM Configuration Mapping: 16K Rows /1024 Columns, 8 banks

CPU Address Line																											
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[2:0]				Row[13:0]													Column[9:0]										M[1:0]	

Table 28-26. Interleaved Mapping DDR-SDRAM Configuration Mapping: 16K Rows /1024 Columns, 8 banks

CPU Address Line																											
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**
Row[13:0]														Bk[2:0]			Column[9:0]										M[1:0]	

28.6.3 DDR-SDRAM Address Mapping for Low-cost Memories

Table 28-27. Sequential Mapping for DDR-SDRAM Configuration, 2K Rows, 512 Columns, 2 banks, 16 bits

CPU Address Line																									
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	Row[10:0]											Column[8:0]									M0

Table 28-28. Interleaved Mapping for DDR-SDRAM Configuration, 2K Rows, 512 Columns, 2 banks, 16 bits

CPU Address Line																									
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**
						Row[10:0]											Bk	Column[8:0]									M0

Table 28-29.

CPU Address Line																									
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	Row[11:0]											Column[7:0]								M[1:0]	

Table 28-30.

CPU Address Line																									
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**
						Row[11:0]											Bk	Column[7:0]								M[1:0]	

Notes: 1. M[1:0] is the byte address inside a 32-bit word.
2. Bk[2] = BA2, Bk[1] = BA1, Bk[0] = BA0

28.7 AHB Multiport DDR-SDRAM Controller (MPDDRC) User Interface

The User Interface is connected to the APB bus. The MPDDRC is programmed using the registers listed in Table 28-31.

Table 28-31. Register Mapping

Offset	Register	Name	Access	Reset
0x00	MPDDRC Mode Register	MPDDRC_MR	Read/Write	0x00000000
0x04	MPDDRC Refresh Timer Register	MPDDRC_RTR	Read/Write	0x03000000
0x08	MPDDRC Configuration Register	MPDDRC_CR	Read/Write	0x00207024
0x0C	MPDDRC Timing Parameter 0 Register	MPDDRC_TPR0	Read/Write	0x20227225
0x10	MPDDRC Timing Parameter 1 Register	MPDDRC_TPR1	Read/Write	0x3C80808
0x14	MPDDRC Timing Parameter 2 Register	MPDDRC_TPR2	Read/Write	0x00042062
0x18	Reserved	–	–	–
0x1C	MPDDRC Low-power Register	MPDDRC_LPR	Read/Write	0x00010000
0x20	MPDDRC Memory Device Register	MPDDRC_MD	Read/Write	0x13
0x24	Reserved	–	–	–
0x28	MPDDRC Low-power DDR2 Low-power Register	MPDDRC_LPDDR2_LPR	Read/Write	0x00000000
0x2C	MPDDRC Low-power DDR2 Calibration and MR4 Register	MPDDRC_LPDDR2_CAL_MR4	Read/Write	0x00000000
0x30	MPDDRC Low-power DDR2 Timing Calibration Register	MPDDRC_LPDDR2_TIM_CAL	Read/Write	0x06
0x34	MPDDRC I/O Calibration Register	MPDDRC_IO_CALIBR	Read/Write	0x00870000
0x38	MPDDRC OCMS Register	MPDDRC_OCMS	Read/Write	0x00000000
0x3C	MPDDRC OCMS KEY1 Register	MPDDRC_OCMS_KEY1	Write-only	–
0x40	MPDDRC OCMS KEY2 Register	MPDDRC_OCMS_KEY2	Write-only	–
0x44	MPDDRC Configuration Arbiter Register	MPDDRC_CONF_ARBITER	Read/Write	0x00000000
0x48	MPDDRC Timeout Register	MPDDRC_TIMEOUT	Read/Write	0x00000000
0x4C	MPDDRC Request Port 0-1-2-3 Register	MPDDRC_REQ_PORT_0123	Read/Write	0x00000000
0x50	MPDDRC Request Port 4-5-6-7 Register	MPDDRC_REQ_PORT_4567	Read/Write	0x00000000
0x54	MPDDRC Current/Maximum Bandwidth Port 0-1-2-3 Register	MPDDRC_BDW_PORT_0123	Read-only	0x00000000
0x58	MPDDRC Current/Maximum Bandwidth Port 4-5-6-7 Register	MPDDRC_BDW_PORT_4567	Read-only	0x00000000
0x5C	MPDDRC Read Data Path Register	MPDDRC_RD_DATA_PATH	Read/Write	0x00000000
0x60–0xE0	Reserved	–	–	–
0xE4	MPDDRC Write Protection Mode Register	MPDDRC_WPMR	Read/Write	0x00000000
0xE8	MPDDRC Write Protection Status Register	MPDDRC_WPSR	Read-only	0x00000000
0xEC–0xFC	Reserved	–	–	–
0x100	MPDDRC DLL Offset Selection Register	MPDDRC_DLL_OS	Read/Write	0x0
0x104	MPDDRC DLL Master Offset Register	MPDDRC_DLL_MAO	Read/Write	0x0 (1)
0x108	MPDDRC DLL Slave Offset 0 Register	MPDDRC_DLL_SO0	Read/Write	0x0(1)

Table 28-31. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x10C	MPDDRC DLL Slave Offset 1 Register	MPDDRC_DLL_SO1	Read/Write	0x0[1]
0x110	MPDDRC DLL CLKWR Offset Register	MPDDRC_DLL_WRO	Read/Write	0x0[1]
0x114	MPDDRC DLL CLKAD Offset Register	MPDDRC_DLL_ADO	Read/Write	0x0[1]
0x118	MPDDRC DLL Status Master 0 Register	MPDDRC_DLL_SM0	Read-only	0x0
0x11C	MPDDRC DLL Status Master 1 Register	MPDDRC_DLL_SM1	Read-only	0x0
0x120	MPDDRC DLL Status Master 2 Register	MPDDRC_DLL_SM2	Read-only	0x0
0x124	MPDDRC DLL Status Master 3 Register	MPDDRC_DLL_SM3	Read-only	0x0
0x128	MPDDRC DLL Status Slave 0 Register	MPDDRC_DLL_SSL0	Read-only	0x0
0x12C	MPDDRC DLL Status Slave 1 Register	MPDDRC_DLL_SSL1	Read-only	0x0
0x130	MPDDRC DLL Status Slave 2 Register	MPDDRC_DLL_SSL2	Read-only	0x0
0x134	MPDDRC DLL Status Slave 3 Register	MPDDRC_DLL_SSL3	Read-only	0x0
0x138	MPDDRC DLL Status Slave 4 Register	MPDDRC_DLL_SSL4	Read-only	0x0
0x13C	MPDDRC DLL Status Slave 5 Register	MPDDRC_DLL_SSL5	Read-only	0x0
0x140	MPDDRC DLL Status Slave 6 Register	MPDDRC_DLL_SSL6	Read-only	0x0
0x144	MPDDRC DLL Status Slave 7 Register	MPDDRC_DLL_SSL7	Read-only	0x0
0x148	MPDDRC DLL Status CLKWR 0 Register	MPDDRC_DLL_SWR0	Read-only	0x0
0x14C	MPDDRC DLL Status CLKWR 1 Register	MPDDRC_DLL_SWR1	Read-only	0x0
0x150	MPDDRC DLL Status CLKWR 2 Register	MPDDRC_DLL_SWR2	Read-only	0x0
0x154	MPDDRC DLL Status CLKWR 3 Register	MPDDRC_DLL_SWR3	Read-only	0x0
0x158	MPDDRC DLL Status CLKAD Register	MPDDRC_DLL_SAD	Read-only	0x0
0x15C–0x1FC	Reserved	–	–	–

1. Values vary with the product implementation.

28.7.1 MPDDRC Mode Register

Name: MPDDRC_MR

Address: 0xF0010000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
MRS							

7	6	5	4	3	2	1	0
–	–	–	–	–	MODE		

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

- **MODE: MPDDRC Command Mode**

This field defines the command issued by the MPDDRC when the SDRAM device is accessed. This register is used to initialize the SDRAM device and to activate Deep Powerdown mode.

Value	Name	Description
0	NORMAL_CMD	Normal Mode. Any access to the MPDDRC is decoded normally. To activate this mode, the command must be followed by a write to the DDR-SDRAM.
1	NOP_CMD	The MPDDRC issues a NOP command when the DDR-SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the DDR-SDRAM.
2	PRCGALL_CMD	The MPDDRC issues the All Banks Precharge command when the DDR-SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the SDRAM.
3	LMR_CMD	The MPDDRC issues a Load Mode Register command when the DDR-SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the DDR-SDRAM.
4	RFSH_CMD	The MPDDRC issues an Auto-Refresh command when the DDR-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 DDR-SDRAM.
5	EXT_LMR_CMD	The MPDDRC issues an Extended 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 DDR-SDRAM. The write in the DDR-SDRAM must be done in the appropriate bank.
6	DEEP_MD	Deep Power mode: Access to Deep Powerdown mode
7	LPDDR2_CMD	The MPDDRC issues an LPDDR2 Mode Register command when the device is accessed regardless of the cycle. To activate this mode, the Mode Register command must be followed by a write to the low-power DDR2-SDRAM.

- **MRS: Mode Register Select LPDDR2**

Configure this 8-bit field to program all mode registers included in the low-power DDR2-SDRAM device. This field is unique to the low-power DDR2-SDRAM devices.

28.7.2 MPDDRC Refresh Timer Register

Name: MPDDRC_RTR

Address: 0xF0010004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	MR4_VALUE			–	–	REF_PB	ADJ_REF

15	14	13	12	11	10	9	8
–	–	–	–	COUNT			

7	6	5	4	3	2	1	0
COUNT							

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

- **COUNT: MPDDRC Refresh Timer Count**

This 12-bit field is loaded into a timer which generates the refresh pulse. Each time the refresh pulse is generated, a refresh sequence is initiated.

The SDRAM devices require a refresh of all rows every 64 ms. The value to be loaded depends on the MPDDRC clock frequency (MCK: Master Clock) and the number of rows in the device.

For example, for an SDRAM with 8192 rows and a 100 MHz Master clock, the value of the COUNT field is configured: $((64 \times 10^{-3}) / 8192) \times 100 \times 10^{6} = 781$ or 0x030D.

Low-power DDR2-SDRAM devices support Per Bank Refresh operation. In this configuration, average time between refresh command is 0.975 µs. The value of the COUNT field is configured depending on this value. For example, the value of a 100 MHz Master clock refresh timer is 98 or 0x0062.

- **ADJ_REF: Adjust Refresh Rate**

Reset value is 0.

0: Adjust refresh rate is not enabled.

1: Adjust refresh rate is enabled.

This mode is unique to the low-power DDR2-SDRAM devices.

- **REF_PB: Refresh Per Bank**

Reset value is 0.

0: Refresh all banks during auto-refresh operation.

1: Refresh the scheduled bank by the bank counter in the memory interface.

This mode is unique to the low-power DDR2-SDRAM devices.

- **MR4_VALUE: Content of MR4 Register (read-only)**

Reset value is 3.

This field gives the content of MR4 register. This field is updated when MRR command is generated and Adjust Refresh Rate bit is enabled. Update is done when read value is different from MR4_VALUE.

LPDDR2 JEDEC memory standards impose derating LPDDR2 AC timings (t_{RCD}, t_{RC}, t_{RAS}, t_{RP} and t_{RRD}) when the value of MR4 is equal to 6. If the application needs to work in extreme conditions, the derating value must be added to AC timings before the powerup sequence.

This mode is unique to the low-power DDR2-SDRAM devices.

28.7.3 MPDDRC Configuration Register

Name: MPDDRC_CR

Address: 0xF0010008

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
UNAL	DECOD	NDQS	NB	LC_LPDDR1	–	ENRDM	DQMS

15	14	13	12	11	10	9	8
–	OCD			ZQ		DIS_DLL	DIC_DS

7	6	5	4	3	2	1	0
DLL	CAS			NR		NC	

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

- **NC: Number of Column Bits**

Reset value is 0 (9 column bits).

Value	Name	Description
0	9_COL_BITS	9 bits to define the column number, up to 512 columns
1	10_COL_BITS	10 bits to define the column number, up to 1024 columns
2	11_COL_BITS	11 bits to define the column number, up to 2048 columns
3	12_COL_BITS	12 bits to define the column number, up to 4096 columns

- **NR: Number of Row Bits**

Reset value is 1 (12 row bits).

Value	Name	Description
0	11_ROW_BITS	11 bits to define the row number, up to 2048 rows
1	12_ROW_BITS	12 bits to define the row number, up to 4096 rows
2	13_ROW_BITS	13 bits to define the row number, up to 8192 rows
3	14_ROW_BITS	14 bits to define the row number, up to 16384 rows

- **CAS: CAS Latency**

Reset value is 2 (2 cycles).

Value	Name	Description
2	DDR_CAS2	LPDDR1 CAS Latency 2
3	DDR_CAS3	DDR2/LPDDR2/LPDDR1 CAS Latency 3
4	DDR_CAS4	DDR2/LPDDR2 CAS Latency 4
5	DDR_CAS5	DDR2/LPDDR2 CAS Latency 5
6	DDR_CAS6	DDR2 CAS Latency 6

- **DLL: Reset DLL**

Reset value is 0.

This bit defines the value of Reset DLL.

Value	Name	Description
0	RESET_DISABLED	Disable DLL reset
1	RESET_ENABLED	Enable DLL reset

This value is used during the powerup sequence.

This bit is found only in the DDR2-SDRAM devices.

- **DIC_DS: Output Driver Impedance Control (Drive Strength)**

Reset value is 0.

This bit name is described as “DS” in some memory datasheets. It defines the output drive strength. This value is used during the powerup sequence.

Value	Name	Description
0	DDR2_NORMALSTRENGTH	Normal drive strength (DDR2)
1	DDR2_WEAKSTRENGTH	Weak drive strength (DDR2)

This bit is found only in the DDR2-SDRAM devices.

- **DIS_DLL: DISABLE DLL**

Reset value is 0.

0: Enable DLL.

1: Disable DLL.

This value is used during the powerup sequence. It is only found in the DDR2-SDRAM devices.

- **ZQ: ZQ Calibration**

Reset value is 0.

Value	Name	Description
0	INIT	Calibration command after initialization
1	LONG	Long calibration
2	SHORT	Short calibration
3	RESET	ZQ Reset

This parameter is used to calibrate DRAM On resistance (Ron) values over PVT.

This field is found only in the low-power DDR2-SDRAM devices.

- **OCD: Off-chip Driver**

Reset value is 7.

Note: SDRAM Controller supports only two values for OCD (default calibration and exit from calibration). These values MUST always be programmed during the initialization sequence. The default calibration must be programmed first, after which the exit calibration and maintain settings must be programmed.

This field is found only in the DDR2-SDRAM devices.

Value	Name	Description
0	DDR2_EXITCALIB	Exit from OCD Calibration mode and maintain settings
7	DDR2_DEFAULT_CALIB	OCD calibration default

- **DQMS: Mask Data is Shared**

Reset value is 0.

Value	Name	Description
0	NOT_SHARED	DQM is not shared with another controller
1	SHARED	DQM is shared with another controller

- **ENRDM: Enable Read Measure**

Reset value is 0.

Value	Name	Description
0	OFF	DQS/DDR_DATA phase error correction is disabled
1	ON	DQS/DDR_DATA phase error correction is enabled

- **LC_LPDDR1: Low-cost Low-power DDR1**

Reset value is 0.

Value	Name	Description
0	NOT_2_BANKS	Any type of memory devices except of low cost, low density Low Power DDR1.
1	2_BANKS_LPDDR1	Low-cost and low-density low-power DDR1. These devices have a density of 32 Mbits and are organized as two internal banks. To use this feature, the user has to define the type of memory and the data bus width (refer to Section 28.7.8 “MPDDRC Memory Device Register”). The 16-bit memory device is organized as 2 banks, 9 columns and 11 rows.

- **NB: Number of Banks**

Reset value is 0 (4 banks). If LC_LPDDR1 is set to 1, NB is not relevant.

Value	Name	Description
0	4_BANKS	4-bank memory devices
1	8_BANKS	8 banks. Only possible when using the DDR2-SDRAM and low-power DDR2-SDRAM devices.

- **NDQS: Not DQS**

Reset value is 1 (Not DQS is disabled).

Value	Name	Description
0	ENABLED	Not DQS is enabled
1	DISABLED	Not DQS is disabled

This bit is found only in the DDR2-SDRAM devices.

- **DECOD: Type of Decoding**

Reset value is 0.

Value	Name	Description
0	SEQUENTIAL	Method for address mapping where banks alternate at each last DDR-SDRAM page of the current bank.
1	INTERLEAVED	Method for address mapping where banks alternate at each DDR-SDRAM end of page of the current bank.

- **UNAL: Support Unaligned Access**

Reset value is 0 (unaligned access is not supported).

Value	Name	Description
0	UNSUPPORTED	Unaligned access is not supported.
1	SUPPORTED	Unaligned access is supported.

This mode is enabled with masters which have an AXI interface.

28.7.4 MPDDRC Timing Parameter 0 Register

Name: MPDDRC_TPR0

Address: 0xF001000C

Access: Read/Write

31	30	29	28	27	26	25	24
TMRD				TWTR			
23	22	21	20	19	18	17	16
TRRD				TRP			
15	14	13	12	11	10	9	8
TRC				TWR			
7	6	5	4	3	2	1	0
TRCD				TRAS			

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

- **TRAS: Active to Precharge Delay**

Reset value is 5 DDRCK(1) clock cycles.

This field defines the delay between an Activate command and a Precharge command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

- **TRCD: Row to Column Delay**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between an Activate command and a Read/Write command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

- **TWR: Write Recovery Delay**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the Write Recovery Time in number of DDRCK(1) clock cycles. The number of cycles is between 1 and 15.

- **TRC: Row Cycle Delay**

Reset value is 7 DDRCK(1) clock cycles.

This field defines the delay between an Activate command and a Refresh command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

- **TRP: Row Precharge Delay**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between a Precharge command and another command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

- **TRRD: Active BankA to Active BankB**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between an Activate command in BankA and an Activate command in BankB in number of DDRCK(1) clock cycles. The number of cycles is between 1 and 15.

- **TWTR: Internal Write to Read Delay**

Reset value is 0.

This field defines the internal Write to Read command time in number of DDRCK(1) clock cycles. The number of cycles is between 1 and 7.

- **TMRD: Load Mode Register Command to Activate or Refresh Command**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between a Load mode register command and an Activate or Refresh command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15. For low-power DDR2-SDRAM, this field is equivalent to TMRW timing.

Note: 1. DDRCK is the clock that drives the SDRAM device.

28.7.5 MPDDRC Timing Parameter 1 Register

Name: MPDDRC_TPR1

Address: 0xF0010010

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	TXP			

23	22	21	20	19	18	17	16
TXSRD							

15	14	13	12	11	10	9	8
TXSNR							

7	6	5	4	3	2	1	0
–	TRFC						

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

- **TRFC: Row Cycle Delay**

Reset value is 8 DDRCK(1) clock cycles.

This field defines the delay between a Refresh command or a Refresh and Activate command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 127.

In case of low-power DDR2-SDRAM, this field is equivalent to t_{RFCab} timing. If the user enables the function “Refresh Per Bank” (refer to “REF_PB: Refresh Per Bank”), this field is equivalent to t_{RFCpb}.

- **TXSNR: Exit Self-refresh Delay to Non-Read Command**

Reset value is 8 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a Non Read command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 255. This field is used by the DDR-SDRAM devices. In case of low-power DDR-SDRAM, this field is equivalent to t_{XSR} timing.

- **TXSRD: Exit Self-refresh Delay to Read Command**

Reset value is 200 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a Read command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 255.

This field is found only in DDR2-SDRAM devices.

- **TXP: Exit Powerdown Delay to First Command**

Reset value is 3 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a valid command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

Note: 1. DDRCK is the clock that drives the SDRAM device.

28.7.6 MPDDRC Timing Parameter 2 Register

Name: MPDDRC_TPR2

Address: 0xF0010014

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	TFAW			

15	14	13	12	11	10	9	8
–	TRTP			TRPA			

7	6	5	4	3	2	1	0
TXARDS				TXARD			

- **TXARD: Exit Active Powerdown Delay to Read Command in Mode “Fast Exit”**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a Read command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

This field is found only in the DDR2-SDRAM devices.

- **TXARDS: Exit Active Powerdown Delay to Read Command in Mode “Slow Exit”**

Reset value is 6 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a Read command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

This field is found only in the DDR2-SDRAM devices.

- **TRPA: Row Precharge All Delay**

Reset value is 0 DDRCK(1) clock cycles.

This field defines the delay between a Precharge All Banks command and another command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

This field is found only in the DDR2-SDRAM devices.

- **TRTP: Read to Precharge**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between a Read command and a Precharge command in number of DDRCK(1) clock cycles.

The number of cycles is between 0 and 7.

- **TFAW: Four Active Windows**

Reset value is 4 DDRCK[1] clock cycles.

DDR2 devices with eight banks (1 Gbit or larger) have an additional requirement concerning t_{FAW} timing. This requires that no more than four Activate commands may be issued in any given t_{FAW} (MIN) period.

The number of cycles is between 0 and 15.

This field is found only in DDR2-SDRAM and LPDDR2-SDRAM devices.

Note: 1. DDRCK is the clock that drives the SDRAM device.

28.7.7 MPDDRC Low-power Register

Name: MPDDRC_LPR

Address: 0xF001001C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–		–

23	22	21	20	19	18	17	16
–	–	UPD_MR		–	–	–	APDE

15	14	13	12	11	10	9	8
–	–	TIMEOUT		–	DS		

7	6	5	4	3	2	1	0
–	PASR			LPDDR2_PWOFF	CLK_FR	LPCB	

- **LPCB: Low-power Command Bit**

Reset value is 0.

Value	Name	Description
0	NOLOWPOWER	Low-power feature is inhibited. No Powerdown, Self-refresh and Deep-power modes are issued to the DDR-SDRAM device.
1	SELFREFRESH	The MPDDRC issues a self-refresh command to the DDR-SDRAM device, the clock(s) is/are deactivated and the CKE signal is set low. The DDR-SDRAM device leaves the Self-refresh mode when accessed and reenters it after the access.
2	POWERDOWN	The MPDDRC issues a Powerdown command to the DDR-SDRAM device after each access, the CKE signal is set low. The DDR-SDRAM device leaves the Powerdown mode when accessed and reenters it after the access.
3	DEEPPOWERDOWN	The MPDDRC issues a Deep Powerdown command to the low-power DDR-SDRAM device.

- **CLK_FR: Clock Frozen Command Bit**

Reset value is 0.

This field sets the clock low during Powerdown mode. Some DDR-SDRAM devices do not support freezing the clock during Powerdown mode. Refer to the relevant DDR-SDRAM device datasheet for details.

Value	Name	Description
0	DISABLED	Clock(s) is/are not frozen.
1	ENABLED	Clock(s) is/are frozen.

- **LPDDR2_PWOFF: LPDDR2 - LPDDR3 Power Off Bit**

Reset value is 0.

LPDDR2 power off sequence must be controlled to preserve the LPDDR2 device. The power failure is handled at system level (IRQ or FIQ) and the LPDDR2 power off sequence is applied using the LPDDR2_PWOFF bit.

LPDDR2_PWOFF bit is used to impose CKE low before a power off sequence. Uncontrolled power off sequence can be applied only up to 400 times in the life of a LPDDR2 device.

Value	Name	Description
0	DISABLED	No power-off sequence applied to LPDDR2.
1	ENABLED	A power-off sequence is applied to the LPDDR2 device. CKE is forced low.

- **PASR: Partial Array Self-refresh**

Reset value is 0.

This field is unique to low-power DDR1-SDRAM. It is used to specify whether only one-quarter, one-half or all banks of the DDR-SDRAM array are enabled. Disabled banks are not refreshed in Self-refresh mode.

The values of this field are dependent on the low-power DDR-SDRAM devices.

After the initialization sequence, as soon as the PASR field is modified, the Extended Mode Register in the external device memory is accessed automatically and PASR bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

- **DS: Drive Strength**

Reset value is 0.

This field is unique to low-power DDR1-SDRAM. It selects the output drive strength.

Value	Name	Description
0	DS_FULL	Full drive strength
1	DS_HALF	Half drive strength
2	DS_QUARTER	Quarter drive strength
3	DS_OCTANT	Octant drive strength
4–7	–	Reserved

After the initialization sequence, as soon as the DS field is modified, the Extended Mode Register is accessed automatically and DS bits are updated. Depending on the UPD_MR bit, update is done before entering self-refresh mode or during a refresh command and a pending read or write access.

- **TIMEOUT: Time Between Last Transfer and Low-Power Mode**

Reset value is 0.

This field defines when Low-power mode is activated.

Value	Name	Description
0	NONE	SDRAM Low-power mode is activated immediately after the end of the last transfer.
1	DELAY_64_CLK	SDRAM Low-power mode is activated 64 clock cycles after the end of the last transfer.
2	DELAY_128_CLK	SDRAM Low-power mode is activated 128 clock cycles after the end of the last transfer.
3	–	Reserved

- **APDE: Active Powerdown Exit Time**

Reset value is 1.

This mode is unique to the DDR2-SDRAM devices.

This mode manages the active Powerdown mode which determines performance versus power saving.

Value	Name	Description
0	DDR2_FAST_EXIT	Fast Exit from Powerdown. DDR2-SDRAM devices only.
1	DDR2_SLOW_EXIT	Slow Exit from Powerdown. DDR2-SDRAM devices only.

After the initialization sequence, as soon as the APDE field is modified, the Extended Mode Register (located in the memory of the external device) is accessed automatically and APDE bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access

- **UPD_MR: Update Load Mode Register and Extended Mode Register**

Reset value is 0.

This bit is used to enable or disable automatic update of the Load Mode Register and Extended Mode Register. This update depends on the MPDDRC integration in a system. MPDDRC can either share or not an external bus with another controller.

Value	Name	Description
0	NO_UPDATE	Update of Load Mode and Extended Mode registers is disabled.
1	UPDATE_SHAREDBUS	MPDDRC shares an external bus. Automatic update is done during a refresh command and a pending read or write access in the SDRAM device.
2	UPDATE_NOSHAREDBUS	MPDDRC does not share an external bus. Automatic update is done before entering Self-refresh mode.
3	–	Reserved

28.7.8 MPDDRC Memory Device Register

Name: MPDDRC_MD

Address: 0xF0010020

Access: Read/Write

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
–	–	–	DBW	–	MD		

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

- **MD: Memory Device**

Indicates the type of memory used.

Reset value is that for the DDR-SDRAM device.

Value	Name	Description
3	LPDDR_SDRAM	Low-power DDR1-SDRAM
6	DDR2_SDRAM	DDR2-SDRAM
7	LPDDR2_SDRAM	Low-power DDR2-SDRAM

- **DBW: Data Bus Width**

Value	Name	Description
0	DBW_32_BITS	Data bus width is 32 bits
1	DBW_16_BITS	Data bus width is 16 bits.

28.7.9 MPDDRC Low-power DDR2 Low-power Register

Name: MPDDRC_LPDDR2_LPR

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	DS			

23	22	21	20	19	18	17	16
SEG_MASK							

15	14	13	12	11	10	9	8
SEG_MASK							

7	6	5	4	3	2	1	0
BK_MASK_PASR							

- **BK_MASK_PASR: Bank Mask Bit/PASR**

Partial Array Self-Refresh (low-power DDR2-SDRAM-S4 devices only)

Reset value is 0.

After the initialization sequence, as soon as the BK_MASK_PASR field is modified, Mode Register 16 is accessed automatically and BK_MASK_PASR bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

0: Refresh is enabled (= unmasked).

1: Refresh is disabled (= masked).

This mode is unique to the low-power DDR2-SDRAM-S4 devices. In Self-refresh mode, each bank of LPDDR2 can be independently configured whether a self-refresh operation is taking place or not.

After the initialization sequence, as soon as the BK_MASK_PASR field is modified, the Extended Mode Register is accessed automatically and BK_MASK_PASR bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

- **SEG_MASK: Segment Mask Bit**

Reset value is 0.

After the initialization sequence, as soon as the SEG_MASK field is modified, Mode Register 17 is accessed automatically and SEG_MASK bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

0: Segment is refreshed (= unmasked).

1: Segment is not refreshed (= masked).

This mode is unique to the low-power DDR2-SDRAM-S4 devices. The number of Segment Mask bits differs with the density. For 1 Gbit density, 8 segments are used. In Self-refresh mode, when the Segment Mask bit is configured, the refresh operation is masked in the segment.

- **DS: Drive Strength**

Reset value is 2.

After the initialization sequence, as soon as the DS field is modified, Mode Register 3 is accessed automatically and DS bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

This field is unique to low-power DDR2-SDRAM. It selects the I/O drive strength:

Value	Name	Description
0	–	Reserved
1	DS_34_3	34.3 ohm typical
2	DS_40	40 ohm typical (default)
3	DS_48	48 ohm typical
4	DS_60	60 ohm typical
5	–	Reserved
6	DS_80	80 ohm typical
7	DS_120	120 ohm typical
8–15	–	Reserved

In case of low-power DDR2-SDRAM the RDIV field in the MPDDRC_IO_CALIBR register must be set to same value of DS field.

28.7.10 MPDDRC Low-power DDR2 Calibration and MR4 Register

Name: MPDDRC_LPDDR2_CAL_MR4

Access: Read/Write

31	30	29	28	27	26	25	24
MR4_READ							
23	22	21	20	19	18	17	16
MR4_READ							
15	14	13	12	11	10	9	8
COUNT_CAL							
7	6	5	4	3	2	1	0
COUNT_CAL							

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

- **COUNT_CAL: LPDDR2Calibration Timer Count**

This 16-bit field is loaded into a timer which generates the calibration pulse. Each time the calibration pulse is generated, a ZQCS calibration sequence is initiated.

The ZQCS Calibration command is used to calibrate DRAM Ron values over PVT.

One method for calculating the interval between ZQCS commands gives the temperature ($T_{driftrate}$) and voltage ($V_{driftrate}$) drift rates that the SDRAM is subject to in the application. The interval could be defined by the following formula: ZQCorrection/((T_{Sens} x $T_{driftrate}$) + (V_{Sens} x $V_{driftrate}$))

Where T_{Sens} = max(dRONdTM) and V_{Sens} = max(dRONdVM) define the SDRAM temperature and voltage sensitivities.

For example, if T_{Sens} = 0.75%/C, V_{Sens} = 0.2%/mV, $T_{driftrate}$ = 1C/sec and $V_{driftrate}$ = 15 mV/s, then the interval between ZQCS commands is calculated as:

1.5/((0.75 x 1) + (0.2 x 15)) = 0.4s

In this example, the devices require a calibration every 0.4s. The value to be loaded depends on average time between REFRESH commands, t_{REF}.

For example, for a device with the time between refresh of 7.8 µs, the value of the Calibration Timer Count field is programmed: (0.4/7.8 x 10^{-6}) = 0xC852.

- **MR4_READ: Mode Register 4 Read Interval**

MR4_READ defines the time period between MR4 reads (for LPDDR2-SDRAM). The formula is (MR4_READ+1) × t_{REF}. The value to be loaded depends on the average time between REFRESH commands, t_{REF}. For example, for an LPDDR2-SDRAM with the time between refresh of 7.8 µs, if the MR4_READ value is 2, the time period between MR4 reads is 23.4 µs.

The LPDDR2-SDRAM devices feature a temperature sensor whose status can be read from MR4 register. This sensor can be used to determine an appropriate refresh rate. Temperature sensor data may be read from MR4 register using the Mode Register Read protocol. The Adjust Refresh Rate bit (ADJ_REF) in the Refresh Timer Register (MPDDRC_RTR) must be written to a one to activate these reads.

28.7.11 MPDDRC Low-power DDR2Timing Calibration Register

Name: MPDDRC_LPDDR2_TIM_CAL

Access: Read/Write

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
ZQCS							

- **ZQCS: ZQ Calibration Short**

Reset value is 6 DDRCK(1) clock cycles.

This field defines the delay between the ZQ Calibration command and any valid command in number of DDRCK(1) clock cycles.

The number of cycles is between 0 and 255.

Note: 1. DDRCK is the clock that drives the SDRAM device.

28.7.12 MPDDRC I/O Calibration Register

Name: MPDDRC_IO_CALIBR

Address: 0xF0010034

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CALCODEN				CALCODEP			

15	14	13	12	11	10	9	8
–	–	–	–	–	TZQIO		

7	6	5	4	3	2	1	0
–	–	–	EN_CALIB	–	RDIV		

- **RDIV: Resistor Divider, Output Driver Impedance**

Reset value is 0.

With the LPDDR2-SDRAM device, the RDIV field must be equal to the DS (Drive Strength) field of the MPDDRC Low-power DDR2 Low-power Register.

RDIV is used with the external precision resistor RZQ to define the output driver impedance. The value of RZQ is either 240 ohms (LPDDR2 device) or 200 ohms (DDR2/LPDDR1 device).

Value	Name	Description
1	RZQ_34	LPDDR2 serial impedance line = 34.3 ohms, DDR2/LPDDR1 serial impedance line: Not applicable
2	RZQ_40_RZQ_33_3	LPDDR2 serial impedance line = 40 ohms, DDR2/LPDDR1 serial impedance line = 33.3 ohms
3	RZQ_48_RZQ_40	LPDDR2 serial impedance line = 48 ohms, DDR2/LPDDR1 serial impedance line = 40 ohms
4	RZQ_60_RZQ_50	LPDDR2 serial impedance line = 60 ohms, DDR2/LPDDR1 serial impedance line = 50 ohms
6	RZQ_80_RZQ_66_7	LPDDR2 serial impedance line = 80 ohms, DDR2/LPDDR1 serial impedance line = 66.7 ohms
7	RZQ_120_RZQ_100	LPDDR2 serial impedance line = 120 ohms, DDR2/LPDDR1 serial impedance line = 100 ohms

- **TZQIO: IO Calibration**

This field defines the delay between an IO Calibration command and any valid command in number of DDRCK(1) clock cycles.

The number of cycles is between 0 and 7.

The TZQIO configuration code must be set correctly depending on the clock frequency using the following formula:

TZQIO = (DDRCK × 20e-9) + 1

where DDRCK frequency is in Hz.

For example, for a frequency of 176 MHz, the value of the TZQIO field is configured (176 × 10e6) × (20 × 10e-9) + 1.

- **EN_CALIB: Enable Calibration**

Reset value is 0.

This field enables calibration for the LPDDR1 and DDR2 devices. When the calibration is enabled, it is recommended to define the COUNT_CAL field (refer to “COUNT_CAL: LPDDR2Calibration Timer Count”).

This 16-bit field is loaded into a timer which generates the calibration pulse. Each time the calibration pulse is generated, a calibration sequence is initiated.

Value	Name	Description
0	DISABLE_CALIBRATIO N	Calibration is disabled.
1	ENABLE_CALIBRATION	Calibration is enabled.

- **CALCODEP: Number of Transistor P (read-only)**

Reset value is 7.

This value gives the number of transistor P to perform the calibration.

- **CALCODEN: Number of Transistor N (read-only)**

Reset value is 8.

This value gives the number of transistor N to perform the calibration.

Note: 1. DDRCK is the clock that drives the SDRAM device.

28.7.13 MPDDRC OCMS Register

Name: MPDDRC_OCMS

Access: Read/Write

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
–	–	–	–	–	–	–	SCR_EN

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

- **SCR_EN: Scrambling Enable**

0: Disables “Off-chip” scrambling for SDRAM access.

1: Enables “Off-chip” scrambling for SDRAM access.

28.7.14 MPDDRC OCMS KEY1 Register

Name: MPDDRC_OCMS_KEY1

Access: Write once

31	30	29	28	27	26	25	24
KEY1							

23	22	21	20	19	18	17	16
KEY1							

15	14	13	12	11	10	9	8
KEY1							

7	6	5	4	3	2	1	0
KEY1							

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

- **KEY1: Off-chip Memory Scrambling (OCMS) Key Part 1**

When Off-chip Memory Scrambling is enabled, the data scrambling depends on KEY1 and KEY2 values.

28.7.15 MPDDRC OCMS KEY2 Register

Name: MPDDRC_OCMS_KEY2

Access: Write once

31	30	29	28	27	26	25	24
KEY2							

23	22	21	20	19	18	17	16
KEY2							

15	14	13	12	11	10	9	8
KEY2							

7	6	5	4	3	2	1	0
KEY2							

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

- **KEY2: Off-chip Memory Scrambling (OCMS) Key Part 2**

When Off-chip Memory Scrambling is enabled, the data scrambling depends on KEY1 and KEY2 values.

28.7.16 MPDDRC Configuration Arbiter Register

Name: MPDDRC_CONF_ARBITER

Access: Read/Write

31	30	29	28	27	26	25	24
BDW_BURST_P7	BDW_BURST_P6	BDW_BURST_P5	BDW_BURST_P4	BDW_BURST_P3	BDW_BURST_P2	BDW_BURST_P1	BDW_BURST_P0
23	22	21	20	19	18	17	16
MA_PR_P7	MA_PR_P6	MA_PR_P5	MA_PR_P4	MA_PR_P3	MA_PR_P2	MA_PR_P1	MA_PR_P0
15	14	13	12	11	10	9	8
RQ_WD_P7	RQ_WD_P6	RQ_WD_P5	RQ_WD_P4	RQ_WD_P3	RQ_WD_P2	RQ_WD_P1	RQ_WD_P0
7	6	5	4	3	2	1	0
–	–	–	–	BDW_MAX_CUR	–	ARB	

- **ARB: Type of Arbitration**

Reset value is 0.

This field allows to choose the type of arbitration: round-robin, number of requests per port or bandwidth per port.

Value	Name	Description
0	ROUND	Round Robin
1	NB_REQUEST	Request Policy
2	BANDWIDTH	Bandwidth Policy
3	–	Reserved

- **RQ_WD_Px: Request or Word from Port X**

Reset value is 0.

0: Number of requests is selected.

1: Number of words is selected.

- **BDW_BURST_Px: Bandwidth is Reached or Bandwidth and Current Burst Access is Ended on port X**

Reset value is 0.

0: The arbitration is done when bandwidth is reached and burst access is ended.

1: The arbitration is done when bandwidth is reached.

- **BDW_MAX_CUR: Bandwidth Max or Current**

This field displays the maximum of the bandwidth or the current bandwidth for each port.

The maximum of the bandwidth is computed when at least two ports of MPDDRC are used.

That information is given in Section 28.7.20 “MPDDRC Current/Maximum Bandwidth Port 0-1-2-3 Register” and Section 28.7.21 “MPDDRC Current/Maximum Bandwidth Port 4-5-6-7 Register”.

Reset value is 0.

0: Current bandwidth is displayed.

1: Maximum of the bandwidth is displayed.

- **MA_PR_Px: Master or Software Provide Information**

Reset value is 0.

0: Number of requests or words is provided by the master, if the master supports this feature.

1: Number of requests or words is provided by software, refer to “NRQ_NWD_BDW_Px: Number of Requests, Number of Words or Bandwidth Allocation from Port 0-1-2-3”.

28.7.17 MPDDRC Timeout Register

Name: MPDDRC_TIMEOUT

Access: Read/Write

31	30	29	28	27	26	25	24
TIMEOUT_P7				TIMEOUT_P6			
23	22	21	20	19	18	17	16
TIMEOUT_P5				TIMEOUT_P4			
15	14	13	12	11	10	9	8
TIMEOUT_P3				TIMEOUT_P2			
7	6	5	4	3	2	1	0
TIMEOUT_P1				TIMEOUT_P0			

- **TIMEOUT_Px: Timeout for Ports 0, 1, 2, 3, 4, 5, 6 and 7**

Reset value is 0.

Some masters have the particularity to insert idle state between two accesses. This field defines the delay between two accesses on the same port in number of DDRCK(1) clock cycles before arbitration and handling the access over to another port.

This field is not used with round-robin and bandwidth arbitrations.

The number of cycles is between 1 and 15.

Note: 1. DDRCK is the clock that drives the SDRAM device.

28.7.18 MPDDRC Request Port 0-1-2-3 Register

Name: MPDDRC_REQ_PORT_0123

Access: Read/Write

31	30	29	28	27	26	25	24
NRQ_NWD_BDW_P3							
23	22	21	20	19	18	17	16
NRQ_NWD_BDW_P2							
15	14	13	12	11	10	9	8
NRQ_NWD_BDW_P1							
7	6	5	4	3	2	1	0
NRQ_NWD_BDW_P0							

- **NRQ_NWD_BDW_Px: Number of Requests, Number of Words or Bandwidth Allocation from Port 0-1-2-3**

Reset value is 0.

The number of requests corresponds to the number of start transfers. For example, setting this field to 2 performs two burst accesses regardless of the burst type (INCR4, INCR8, etc.). The number of words corresponds exactly to the number of accesses; setting this field to 2 performs two accesses. In this example, burst accesses will be broken.

These values depend on scheme arbitration (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

In case of round-robin arbitration, this field is not used. In case of “bandwidth arbitration”, this field corresponds to percentage allocated for each port. In case of “request” arbitration, this field corresponds to number of start transfers or to number of accesses allocated for each port.

28.7.19 MPDDRC Request Port 4-5-6-7 Register

Name: MPDDRC_REQ_PORT_4567

Access: Read/Write

31	30	29	28	27	26	25	24
NRQ_NWD_BDW_P7							
23	22	21	20	19	18	17	16
NRQ_NWD_BDW_P6							
15	14	13	12	11	10	9	8
NRQ_NWD_BDW_P5							
7	6	5	4	3	2	1	0
NRQ_NWD_BDW_P4							

- **NRQ_NWD_BDW_Px: Number of Requests, Number of Words or Bandwidth allocation from port 4-5-6-7**

Reset value is 0.

The number of requests corresponds to the number of start transfers. For example, setting this field to 2 performs two burst accesses regardless of the burst type (INCR4, INCR8, etc.). The number of words corresponds exactly to the number of accesses; setting this field to 2 performs two accesses. In this example, burst accesses will be broken.

These values depend on scheme arbitration (refer to Section 28.7.16 “MPDDRC Configuration Arbiter Register”).

In case of round-robin arbitration, this field is not used. In case of “bandwidth arbitration”, this field corresponds to percentage allocated for each port. In case of “request” arbitration, this field corresponds to number of start transfers or to number of accesses allocated for each port.

28.7.20 MPDDRC Current/Maximum Bandwidth Port 0-1-2-3 Register

Name: MPDDRC_BDW_PORT_0123

Access: Read-only

31	30	29	28	27	26	25	24
–	BDW_P3						

23	22	21	20	19	18	17	16
–	BDW_P2						

15	14	13	12	11	10	9	8
–	BDW_P1						

7	6	5	4	3	2	1	0
–	BDW_P0						

- **BDW_Px: Current/Maximum Bandwidth from Port 0-1-2-3**

Reset value is 0.

This field displays the current bandwidth or the maximum bandwidth for each port. This information is given in the "BDW_MAX_CUR: Bandwidth Max or Current" field description.

28.7.21 MPDDRC Current/Maximum Bandwidth Port 4-5-6-7 Register

Name: MPDDRC_BDW_PORT_4567

Access: Read-only

31	30	29	28	27	26	25	24
–	BDW_P7						

23	22	21	20	19	18	17	16
–	BDW_P6						

15	14	13	12	11	10	9	8
–	BDW_P5						

7	6	5	4	3	2	1	0
–	BDW_P4						

- **BDW_Px: Current/Maximum Bandwidth from Port 4-5-6-7**

Reset value is 0.

This field displays the current bandwidth or the maximum bandwidth for each port. This information is given in the "BDW_MAX_CUR: Bandwidth Max or Current" field description.

28.7.22 MPDDRC Read Data Path Register

Name: MPDDRC_RD_DATA_PATH

Access: Read/Write

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
–	–	–	–	–	–	SHIFT_SAMPLING	

- **SHIFT_SAMPLING: Shift Sampling Point of Data**

Reset value is 0.

This field shifts the sampling point of data that comes from the memory device. This sampling point depends on the external bus frequency. The higher the frequency, the more the sampling point will be shifted.

Value	Name	Description
0	NO_SHIFT	Initial sampling point.
1	SHIFT_ONE_CYCLE	Sampling point is shifted by one cycle.
2	SHIFT_TWO_CYCLES	Sampling point is shifted by two cycles.
3	SHIFT_THREE_CYCLES	Sampling point is shifted by three cycles, unique for LPDDR2. Not applicable for DDR2 and LPDDR1 devices.

28.7.23 MPDDRC Write Protection Mode Register

Name: MPDDRC_WPMR

Address: 0xF00100E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x444452 (“DDR” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x444452 (“DDR” in ASCII).

Refer to Section 28.7 “AHB Multiport DDR-SDRAM Controller (MPDDRC) User Interface” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x444452	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

28.7.24 MPDDRC Write Protection Status Register

Name: MPDDRC_WPSR

Address: 0xF00100E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Enable**

0: No write protection violation occurred since the last read of this register (MPDDRC_WPSR).

1: A write protection violation occurred since the last read of this register (MPDDRC_WPSR). If this violation is an unauthorized attempt to write a control register, the associated violation is reported into the WPVSRC field.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

28.7.25 MPDDRC DLL Offset Selection Register

Name: MPDDRC_DLL_OS

Access: Read/Write

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
–	–	–	–	–	–	–	SELOFF

- **SELOFF: Offset Selection**

0: The hardcoded offsets are selected.

1: The programmable offsets are selected.

28.7.26 MPDDRC DLL Master Offset Register

Name: MPDDRC_DLL_MAO

Access: Read/Write

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
MAOFF							

Only the first 5 bits of the MAOFF field are significant.

- **MAOFF: Master Delay Line Offset**

The value stored by this field is unsigned.

When this field is written, the programmable Master delay line offset is written.

When this field is read:

– DQSDELAY_OSR.SELOFF = 0: The hardcoded Master delay line offset is read.

– DQSDELAY_OSR.SELOFF = 1: The programmable Master delay line offset is read.

28.7.27 MPDDRC DLL Slave Offset 0 Register

Name: MPDDRC_DLL_SO0

Access: Read/Write

31	30	29	28	27	26	25	24
S3OFF							
23	22	21	20	19	18	17	16
S2OFF							
15	14	13	12	11	10	9	8
S1OFF							
7	6	5	4	3	2	1	0
S0OFF							

Only the first 6 bits of the S0OFF, S1OFF, S2OFF, and S3OFF fields are significant.

- **SxOFF: SLAVEx Delay Line Offset**

The value stored by this field is signed.

When this field is written, the programmable SLAVEx delay line offset is written.

When this field is read:

– DQSDELAY_OSR.SELOFF = 0: The hardcoded SLAVEx delay line offset is read.

– DQSDELAY_OSR.SELOFF = 1: The programmable SLAVEx delay line offset is read.

28.7.28 MPDDRC DLL Slave Offset 1 Register

Name: MPDDRC_DLL_SO1

Access: Read/Write

31	30	29	28	27	26	25	24
S7OFF							
23	22	21	20	19	18	17	16
S6OFF							
15	14	13	12	11	10	9	8
S5OFF							
7	6	5	4	3	2	1	0
S4OFF							

Only the first 6 bits of the S4OFF, S5OFF, S6OFF, and S7OFF fields are significant.

- **SxOFF: SLAVEx Delay Line Offset**

The value stored by this field is signed.

When this field is written, the programmable SLAVEx delay line offset is written.

When this field is read:

– DQSDELAY_OSR.SELOFF = 0: The hardcoded SLAVEx delay line offset is read.

– DQSDELAY_OSR.SELOFF = 1: The programmable SLAVEx delay line offset is read.

28.7.29 MPDDRC DLL CLKWR Offset Register

Name: MPDDRC_DLL_WRO

Access: Read/Write

31	30	29	28	27	26	25	24
WR3OFF							
23	22	21	20	19	18	17	16
WR2OFF							
15	14	13	12	11	10	9	8
WR1OFF							
7	6	5	4	3	2	1	0
WR0OFF							

Only the first 6 bits of the WR0OFF, WR1OFF, WR2OFF, and WR3OFF fields are significant.

- **WRxOFF: CLKWRx Delay Line Offset**

The value stored by this field is signed.

When this field is written, the programmable CLKWRx delay line offset is written.

When this field is read:

– DQSDELAY_OSR.SELOFF = 0: The hardcoded CLKWRx delay line offset is read.

– DQSDELAY_OSR.SELOFF = 1: The programmable CLKWRx delay line offset is read.

28.7.30 MPDDRC DLL CLKAD Offset Register

Name: MPDDRC_DLL_ADO

Access: Read/Write

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
ADOFF							

Only the first 6 bits of the ADOFF field are significant.

- **ADOFF: CLKAD Delay Line Offset**

The value stored by this field is signed.

When this field is written, the programmable CLKAD delay line offset is written.

When this field is read:

– DQSDELAY_OSR.SELOFF = 0: The hardcoded CLKAD delay line offset is read.

– DQSDELAY_OSR.SELOFF = 1: The programmable CLKAD delay line offset is read.

28.7.31 MPDDRC DLL Status Master x Register

Name: MPDDRC_DLL_SMx [x = 0..3]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	MDCNT			

23	22	21	20	19	18	17	16
MDCNT				–	–	–	–

15	14	13	12	11	10	9	8
MDLVAL							

7	6	5	4	3	2	1	0
–	–	–	–	–	MDOVF	MDDEC	MDINC

- **MDINC: MASTERx Delay Increment**

0: The DQSDELAY is not incrementing the MASTERx delay counter.

1: The DQSDELAY is incrementing the MASTERx delay counter.

- **MDDEC: MASTERx Delay Decrement**

0: The DQSDELAY is not decrementing the MASTERx delay counter.

1: The DQSDELAY is decrementing the MASTERx delay counter.

- **MDOVF: MASTERx Delay Overflow Flag**

0: The MASTERx delay counter has not reached its maximum value, or the MASTERx is not locked yet

1: The MASTERx delay counter has reached its maximum value, the MASTERx delay counter increment is stopped and the DQSDELAY forces the MASTERx lock. If this flag is set, it means the DDR-SDRAM controller clock frequency is too low compared to MASTERx delay line number of elements.

- **MDLVAL: MASTERx Delay Lock Value**

Value of the MASTERx delay counter at the last lock of the MASTERx (the MASTERx offset is taken into account in this value).

- **MDCNT: MASTERx Delay Counter Value**

Current value of the MASTERx delay counter.

28.7.32 MPDDRC DLL Status Slave x Register

Name: MPDDRC_DLL_SSLx [x = 0..7]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	SDCVAL			

23	22	21	20	19	18	17	16
SDCVAL				–	–	–	–

15	14	13	12	11	10	9	8
SDCNT							

7	6	5	4	3	2	1	0
–	–	–	–	–	SDERF	SDCUDF	SDCOVF

- **SDCOVF: SLAVEx Delay Correction Overflow Flag**

0: Due to the correction, the SLAVEx delay counter has not reached its maximum value, or the SLAVEx is not locked yet.

1: Due to the correction, the SLAVEx delay counter has reached its maximum value, the correction is not optimal because it is not applied entirely

- **SDCUDF: SLAVEx Delay Correction Underflow Flag**

0: Due to the correction, the SLAVEx delay counter has not reached its minimum value, or the SLAVEx is not locked yet.

1: Due to the correction, the SLAVEx delay counter has reached its minimum value, the correction is not optimal because it has not been entirely applied.

- **SDERF: SLAVEx Delay Correction Error Flag**

0: The DQSDELAY has succeeded in computing the SLAVEx delay correction, or the Slave is not locked yet.

1: The DQSDELAY has not succeeded in computing the SLAVEx delay correction.

- **SDCNT: SLAVEx Delay Counter Value**

Current value of the SLAVEx delay counter.

- **SDCVAL: SLAVEx Delay Correction Value**

Value of the correction applied to the SLAVEx delay (the SLAVEx offset is taken into account in this value).

28.7.33 MPDDRC DLL Status CLKWR x Register

Name: MPDDRC_DLL_SWRx [x = 0..3]

Access: Read-only

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
WRDCNT							

- **WRDCNT: CLKWRx Delay Counter Value**

Current value of the CLKWRx delay counter.

28.7.34 MPDDRC DLL Status CLKAD Register

Name: MPDDRC_DLL_SAD

Access: Read-only

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
ADDCNT							

- **ADDCNT: CLKAD Delay Counter Value**

Current value of the CLKAD delay counter.

29. Static Memory Controller (SMC)

29.1 Description

This Static Memory Controller (SMC) is capable of handling several types of external memory and peripheral devices, such as SRAM, PSRAM, PROM, EPROM, EEPROM, LCD Module, NOR Flash and NAND Flash.

The SMC generates the signals that control the access to external memory devices or peripheral devices. It has 4 Chip Selects and a 26-bit address bus. The 16-bit data bus can be configured to interface with 8- or 16-bit external devices. Separate read and write control signals allow for direct memory and peripheral interfacing. Read and write signal waveforms are fully parametrizable.

The SMC can manage wait requests from external devices to extend the current access. The SMC is provided with an automatic Slow Clock mode. In Slow Clock mode, it switches from user-programmed waveforms to slow-rate specific waveforms on read and write signals.

The SMC embeds a NAND Flash Controller (NFC). The NFC can handle automatic transfers, sending the commands and address cycles to the NAND Flash and transferring the contents of the page (for read and write) to the NFC SRAM. It minimizes the CPU overhead.

The SMC includes programmable hardware error correcting code with one-bit error correction capability and supports two-bit error detection. In order to improve the overall system performance, the DATA phase of the transfer can be DMA-assisted.

The External Data Bus can be scrambled/unscrambled by means of user keys.

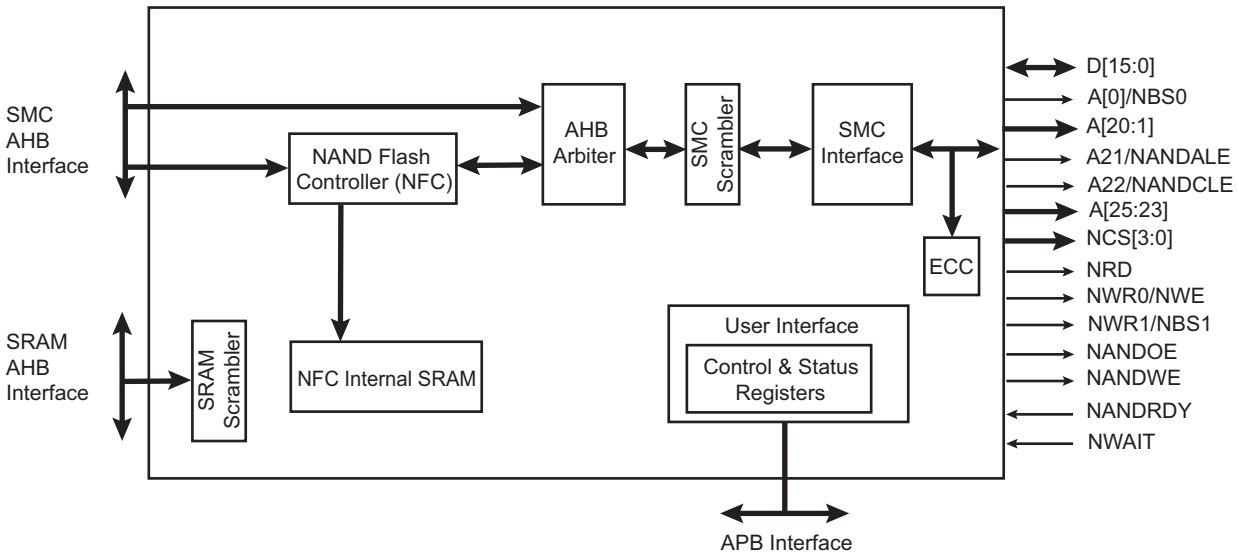
29.2 Embedded Characteristics

- 64-Mbyte Address Space per Chip Select
- 8- or 16-bit Data Bus
- Word, Halfword, Byte Transfers
- Byte Write or Byte Select 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
- Programmable Data Float Time per Chip Select
- External Data Bus Scrambling/Unscrambling Function
- External Wait Request
- Automatic Switch to Slow Clock Mode
- Hardware Configurable Number of Chip Selects from 1 to 4
- Programmable Timing on a per Chip Select Basis
- NAND Flash Controller Supporting NAND Flash with Multiplexed Data/Address Buses
- Supports SLC and MLC NAND Flash Technology
- Supports NAND Flash Devices with 8 or 16-bit Data Paths
- Multibit Error Correcting Code (ECC) supporting NAND Flash devices with 8-bit only Data Path
- ECC Algorithm Based on Binary Shortened Bose, Chaudhuri and Hocquenghem (BCH) Codes
- Programmable Error Correcting Capability: 2, 4, 8, 12 and 24 bits of Errors per Block
- 9 Kbytes NFC SRAM
- Programmable Block Size: 512 bytes or 1024 bytes
- Programmable Number of Block per Page: 1, 2, 4 or 8 Blocks of Data per Page
- Programmable Spare Area Size up to 512 bytes
- Supports Spare Area ECC Protection

- Supports 8 Kbytes Page Size Using 1024 bytes/block and 4 Kbytes Page Size Using 512 bytes/block
- Multibit Error Detection Is Interrupt Driven
- Provides Hardware Acceleration for Determining Roots of Polynomials Defined over a Finite Field
- Programmable Finite Field GF(2^13) or GF(2^14)
- Finds Roots of Error-locator Polynomial
- Programmable Number of Roots
- Register Write Protection

29.3 Block Diagram

Figure 29-1. Block Diagram



29.4 I/O Lines Description

Table 29-1. I/O Line Description

Name	Description	Type	Active Level
NCS[3:0]	Static Memory Controller Chip Select Lines	Output	Low
NRD	Read Signal	Output	Low
NWR0/NWE	Write 0/Write Enable Signal	Output	Low
A0/NBS0	Address Bit 0/Byte 0 Select Signal	Output	Low
NWR1/NBS1	Write 1/Byte 1 Select Signal	Output	Low
A[25:1]	Address Bus	Output	–
D[15:0]	Data Bus	I/O	–
NWAIT	External Wait Signal	Input	Low
NANDRDY	NAND Flash Ready/Busy	Input	–
NANDWE	NAND Flash Write Enable	Output	Low

Table 29-1. I/O Line Description (Continued)

Name	Description	Type	Active Level
NANDOE	NAND Flash Output Enable	Output	Low
NANDALE	NAND Flash Address Latch Enable	Output	–
NANDCLE	NAND Flash Command Latch Enable	Output	–

29.5 Multiplexed Signals

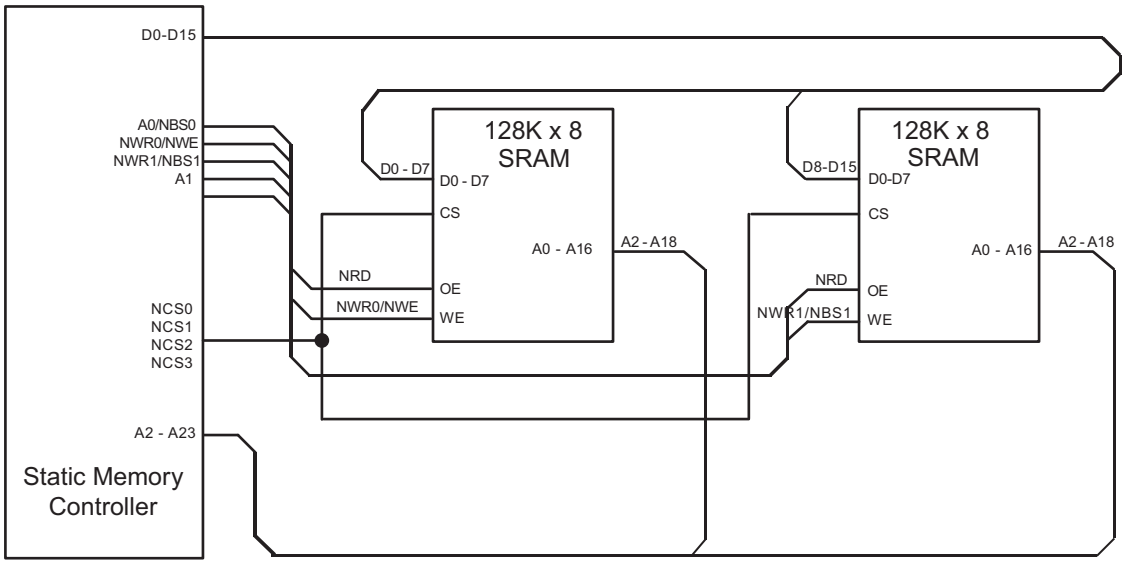
Table 29-2. Static Memory Controller (SMC) Multiplexed Signals

Multiplexed Signals		Related Function
NWR0	NWE	Byte-write or Byte-select access, refer to Section 29.9.2.1 “Byte Write Access” and Section 29.9.2.2 “Byte Select Access”
A0	NBS0	8-bit or 16-bit data bus, refer to Section 29.9.1 “Data Bus Width”
A22	NANDCLE	NAND Flash Command Latch Enable
A21	NANDALE	NAND Flash Address Latch Enable
NWR1	NBS1	Byte-write or Byte-select access, refer to Section 29.9.2.1 “Byte Write Access” and Section 29.9.2.2 “Byte Select Access”
A1	–	8-/16-bit data bus, refer to Section 29.9.1 “Data Bus Width” Byte-write or Byte-select access, refer to Section 29.9.2.1 “Byte Write Access” and Section 29.9.2.2 “Byte Select Access”

29.6 Application Example

29.6.1 Hardware Interface

Figure 29-2. SMC Connections to Static Memory Devices



29.7 Product Dependencies

29.7.1 I/O Lines

The pins used for interfacing the Static Memory Controller are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the Static Memory Controller pins to their peripheral function. If I/O lines of the SMC are not used by the application, they can be used for other purposes by the PIO controller.

29.7.2 Power Management

The SMC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SMC clock.

29.7.3 Interrupt Sources

The SMC has an interrupt line connected to the interrupt controller. Handling the SMC interrupt requires programming the interrupt controller before configuring the SMC.

Table 29-3. Peripheral IDs

Instance	ID
HSMC	22

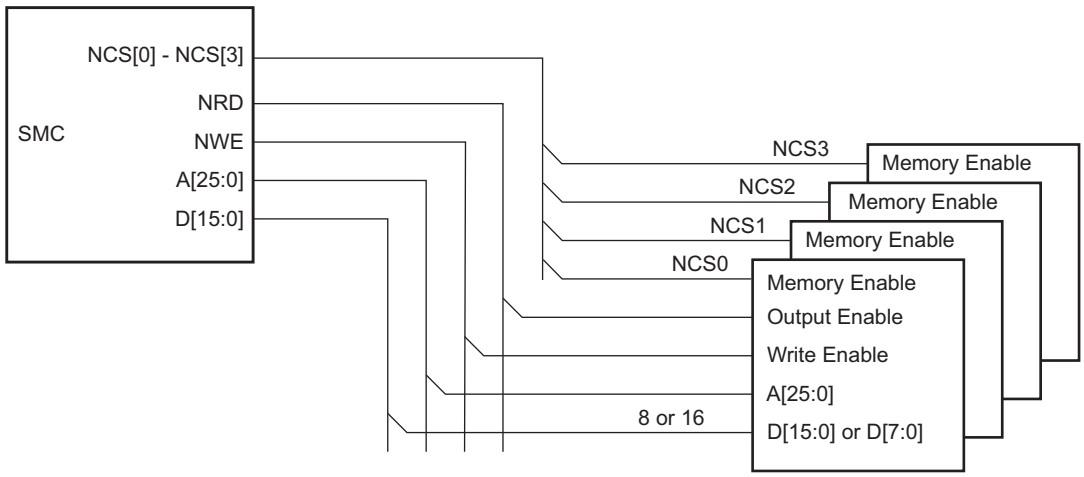
29.8 External Memory Mapping

The SMC provides up to 26 address lines, A[25:0]. This allows each chip select line to address up to 64 Mbytes of memory.

If the physical memory device connected on one chip select is smaller than 64 Mbytes, it wraps around and appears to be repeated within this space. The SMC correctly handles any valid access to the memory device within the page (refer to Figure 29-3).

A[25:0] is only significant for 8-bit memory; A[25:1] is used for 16-bit memory.

Figure 29-3. Memory Connections for External Devices



29.9 Connection to External Devices

29.9.1 Data Bus Width

A data bus width of 8 or 16 bits can be selected for each chip select. This option is controlled by the bit DBW in the SMC Mode Register (HSMC_MODE) for the corresponding chip select.

Figure 29-4 shows how to connect a 512 KB x 8-bit memory on NCS2. Figure 29-5 shows how to connect a 512 KB x 16-bit memory on NCS2.

Figure 29-4. Memory Connection for an 8-bit Data Bus

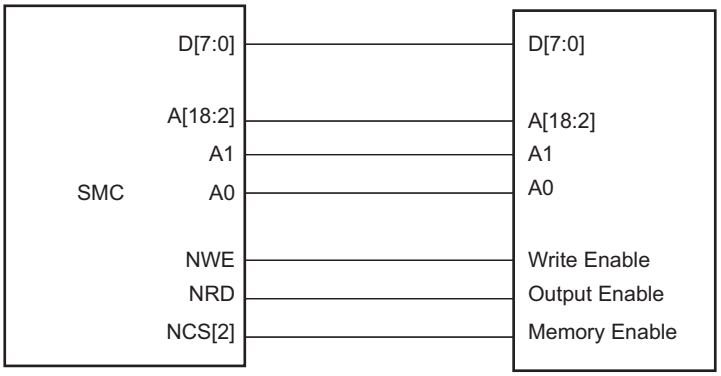
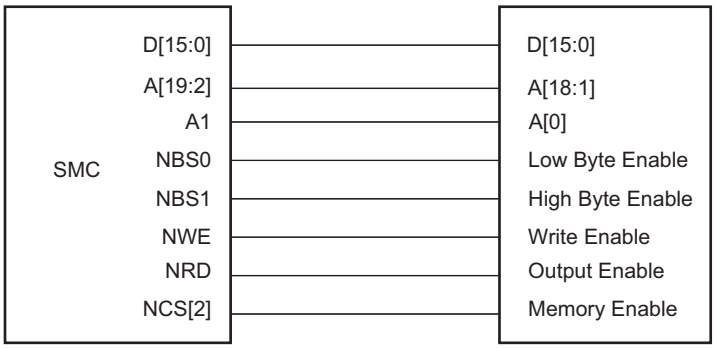


Figure 29-5. Memory Connection for a 16-bit Data Bus



29.9.2 Byte Write or Byte Select Access

Each chip select with a 16-bit data bus can operate with one of two different types of write access: Byte Write or Byte Select. This is controlled by the BAT bit of the HSMC_MODE register for the corresponding chip select.

29.9.2.1 Byte Write Access

Byte Write Access is used to connect 2 x 8-bit devices as a 16-bit memory, and supports one write signal per byte of the data bus and a single read signal.

Note that the SMC does not allow boot in Byte Write Access mode.

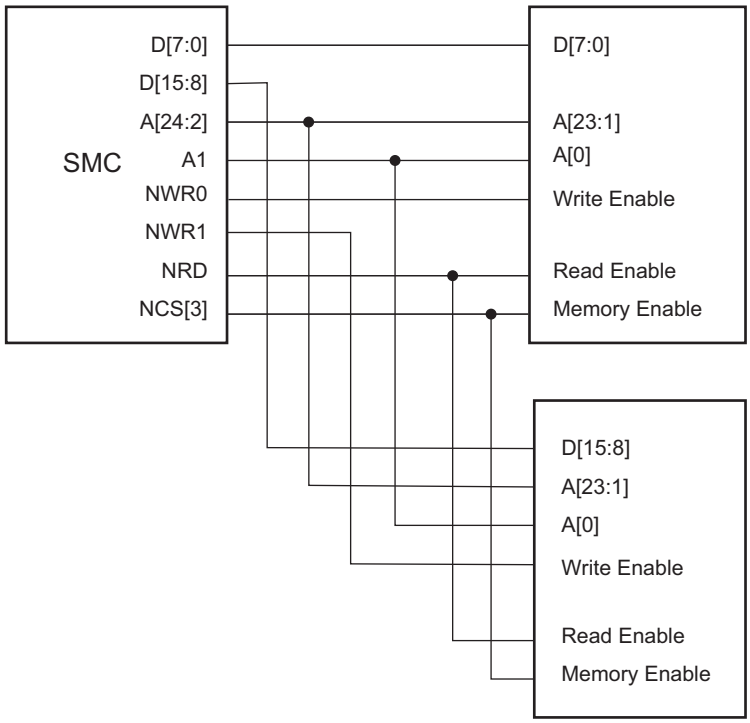
For 16-bit devices, the SMC provides NWR0 and NWR1 write signals for respectively Byte0 (lower byte) and Byte1 (upper byte) of a 16-bit bus. One single read signal (NRD) is provided.

29.9.2.2 Byte Select Access

Byte Select Access is used to connect one 16-bit device. In this mode, read/write operations can be enabled/disabled at Byte level. One Byte-select line per byte of the data bus is provided. One NRD and one NWE signal control read and write.

For 16-bit devices, the SMC provides NBS0 and NBS1 selection signals for respectively Byte0 (lower byte) and Byte1 (upper byte) of a 16-bit bus.

Figure 29-6. Connection of 2 x 8-bit Devices on a 16-bit Bus: Byte Write Option



29.9.2.3 Signal Multiplexing

Depending on the Byte Access Type (BAT), only the write signals or the byte select signals are used. To save IOs at the external bus interface, control signals at the SMC interface are multiplexed. Table 29-4 shows signal multiplexing depending on the data bus width and the Byte Access Type.

For 16-bit devices, bit A0 of address is unused. When Byte Select Option is selected, NWR1 is unused. When Byte Write option is selected, NBS0 is unused.

Table 29-4. SMC Multiplexed Signal Translation

Device Type	Signal Name		
	16-bit Bus		8-bit Bus
	1 x 16-bit	2 x 8-bit	1 x 8-bit
Byte Access Type (BAT)	Byte Select	Byte Write	–
NBS0_A0	NBS0	–	A0
NWE_NWR0	NWE	NWR0	NWE
NBS1_NWR1	NBS1	NWR1	–
A1	A1	A1	A1

29.10 Standard Read and Write Protocols

In the following sections, the Byte Access Type is not considered. Byte select lines (NBS0 to NBS1) always have the same timing as the A address bus. NWE represents either the NWE signal in byte select access type or one of the byte write lines (NWR0 to NWR1) in byte write access type. NWR0 to NWR1 have the same timings and protocol as NWE. In the same way, NCS represents one of the NCS[0..3] chip select lines.

29.10.1 Read Waveforms

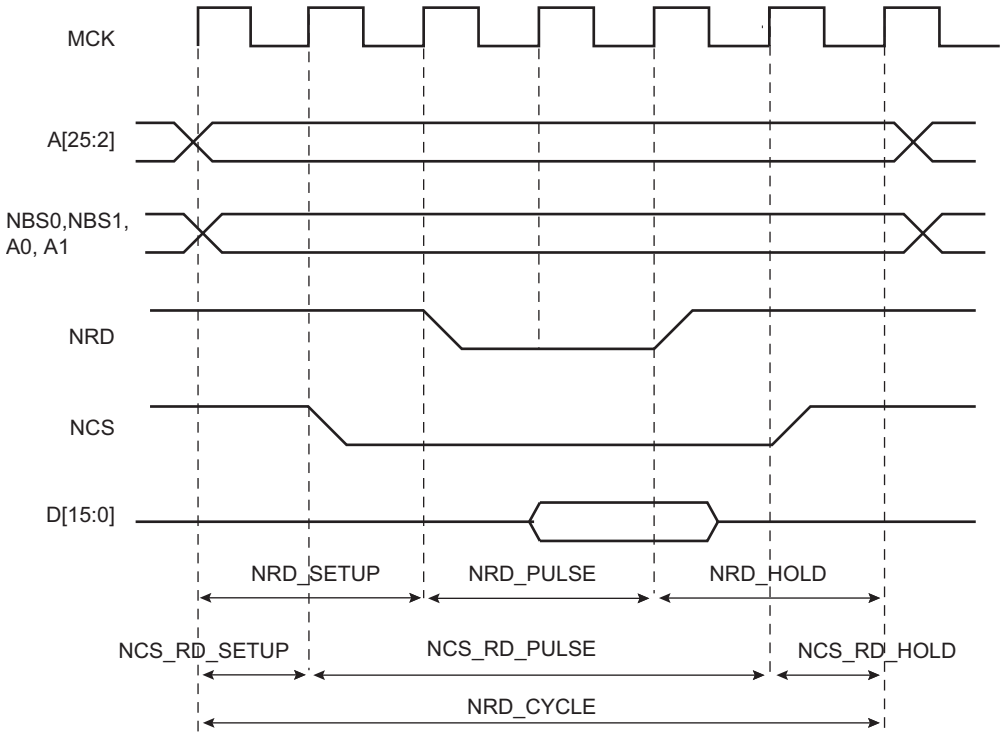
The read cycle is shown on Figure 29-7.

The read cycle starts with the address setting on the memory address bus, i.e.,:

{A[25:2], A1, A0} for 8-bit devices

{A[25:2], A1} for 16-bit devices

Figure 29-7. Standard Read Cycle



29.10.1.1NRD Waveform

The NRD signal is characterized by a setup timing, a pulse width and a hold timing:

1. NRD_SETUP: The NRD setup time is defined as the setup of address before the NRD falling edge.
2. NRD_PULSE: The NRD pulse length is the time between NRD falling edge and NRD rising edge.
3. NRD_HOLD: The NRD hold time is defined as the hold time of address after the NRD rising edge.

29.10.1.2NCS Waveform

Similar to the NRD signal, the NCS signal can be divided into a setup time, pulse length and hold time:

- NCS_RD_SETUP: The NCS setup time is defined as the setup time of address before the NCS falling edge.
- NCS_RD_PULSE: The NCS pulse length is the time between NCS falling edge and NCS rising edge.
- NCS_RD_HOLD: The NCS hold time is defined as the hold time of address after the NCS rising edge.

29.10.1.3Read Cycle

The NRD_CYCLE time is defined as the total duration of the read cycle, that is, 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

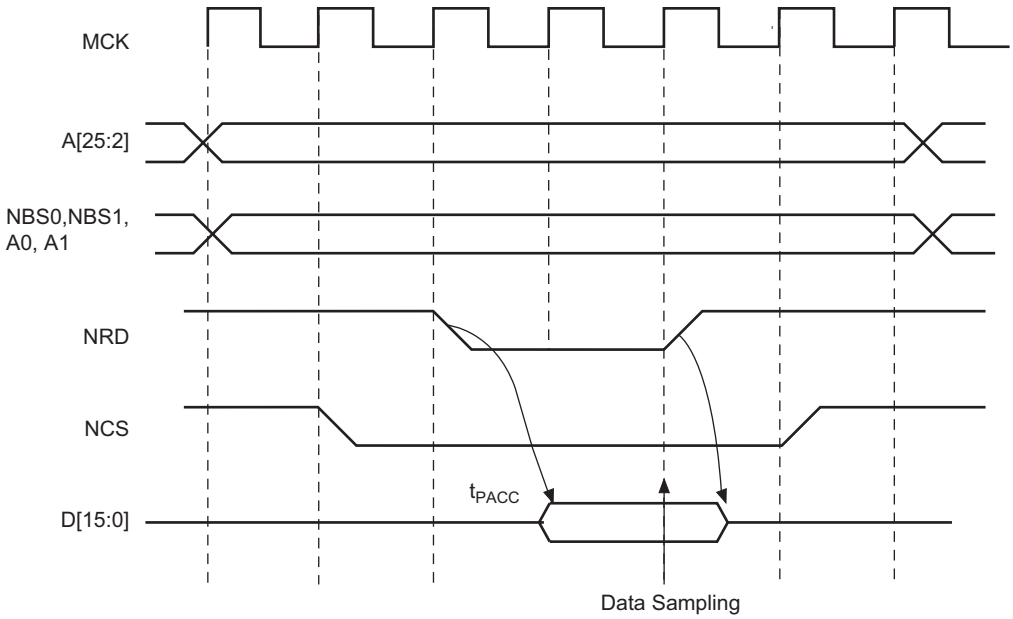
29.10.2 Read Mode

As NCS and NRD waveforms are defined independently of one another, the SMC needs to know when the read data is available on the data bus. The SMC does not compare NCS and NRD timings to know which signal rises first. The READ_MODE parameter in the HSMC_MODE register of the corresponding chip select indicates which signal of NRD and NCS controls the read operation.

29.10.2.1Read is Controlled by NRD (READ_MODE = 1)

Figure 29-8 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, the 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 SMC samples the read data internally on the rising edge of the Master Clock that generates the rising edge of NRD, whatever the programmed waveform of NCS.

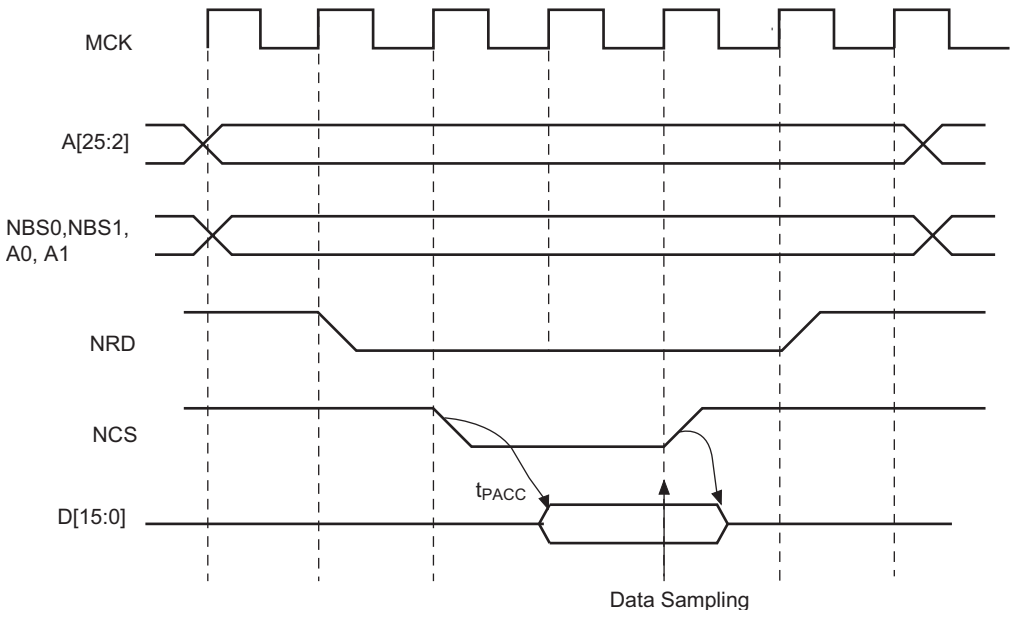
Figure 29-8. READ_MODE = 1: Data is Sampled by SMC before the Rising Edge of NRD



29.10.2.2Read is Controlled by NCS (READ_MODE = 0)

Figure 29-9 shows the typical read cycle. 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 that case, the READ_MODE must be configured to 0 (read is controlled by NCS): the SMC internally samples the data on the rising edge of the Master Clock that generates the rising edge of NCS, whatever the programmed waveform of NRD.

Figure 29-9. READ_MODE = 0: Data is Sampled by SMC before the Rising Edge of NCS



29.10.3 Write Waveforms

The write protocol is similar to the read protocol. It is depicted in Figure 29-10. The write cycle starts with the address setting on the memory address bus.

29.10.3.1NWE 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.

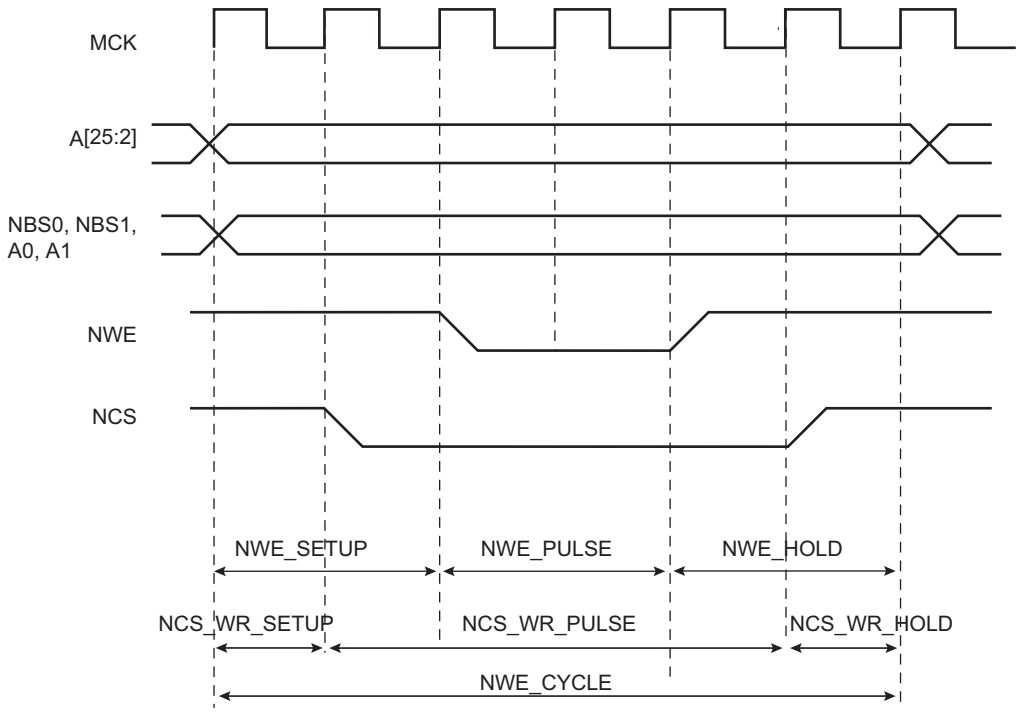
The NWE waveforms apply to all byte-write lines in Byte Write Access mode: NWR0 to NWR3.

29.10.3.2NCS Waveforms

The NCS signal waveforms in write operations are not the same as 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.

Figure 29-10. Write Cycle



29.10.3.3Write 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 equal to:

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

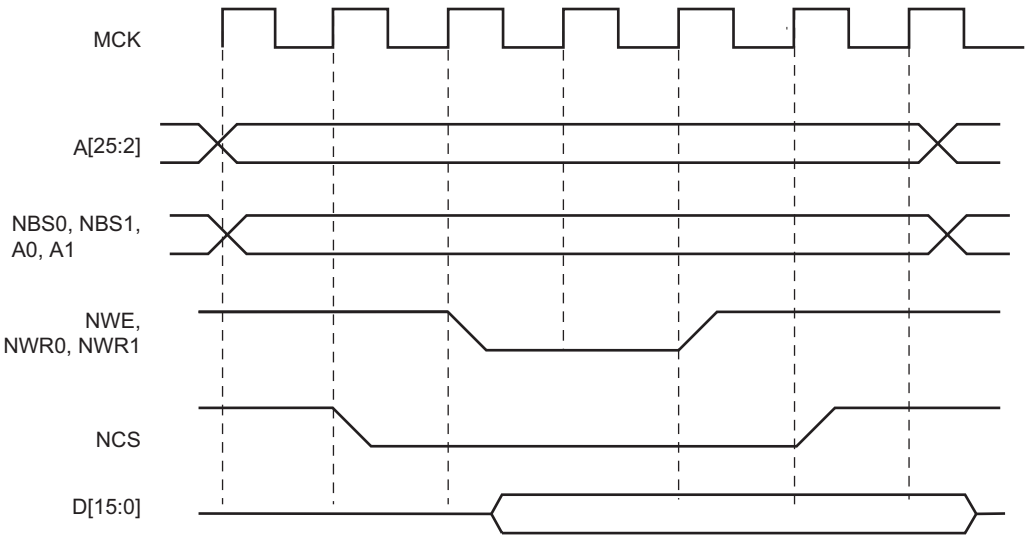
29.10.4 Write Mode

The WRITE_MODE parameter in the HSMC_MODE register of the corresponding chip select indicates which signal controls the write operation.

29.10.4.1 Write is Controlled by NWE (WRITE_MODE = 1)

Figure 29-11 shows the waveforms of a write operation with WRITE_MODE set to 1. 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.

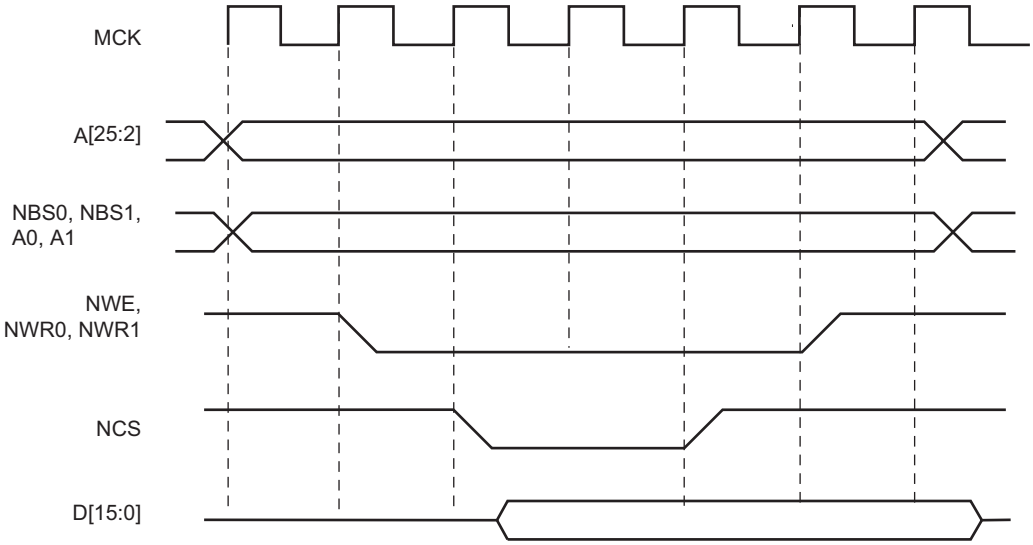
Figure 29-11. WRITE_MODE = 1. The write operation is controlled by NWE



29.10.4.2 Write is Controlled by NCS (WRITE_MODE = 0)

Figure 29-12 shows the waveforms of a write operation with WRITE_MODE configured to 0. 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 29-12. WRITE_MODE = 0. The write operation is controlled by NCS



29.10.5 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

Table 29-5 shows how the timing parameters are coded and their permitted range.

Table 29-5. Coding and Range of Timing Parameters

Coded Value	Number of Bits	Effective Value	Permitted Range	
			Coded Value	Effective Value
setup [5:0]	6	128 x setup[5] + setup[4:0]	0 ≤ setup ≤ 31	0..31
			32 ≤ setup ≤ 63	128..(128 + 31)
pulse [6:0]	7	256 x pulse[6] + pulse[5:0]	0 ≤ pulse ≤ 63	0..63
			64 ≤ pulse ≤ 127	256..(256 + 63)
cycle[8:0]	9	256 x cycle[8:7] + cycle[6:0]	0 ≤ cycle ≤ 127	0..127
			128 ≤ cycle ≤ 255	256..(256 + 127)
			256 ≤ cycle ≤ 383	512..(512 + 127)
			384 ≤ cycle ≤ 511	768..(768 + 127)

29.10.6 Reset Values of Timing Parameters

Table 29-6 gives the default value of timing parameters at reset.

Table 29-6. Reset Values of Timing Parameters

Register	Reset Value	Description
HSMC_SETUP	0x0101_0101	All setup timings are set to 1
HSMC_PULSE	0x0101_0101	All pulse timings are set to 1
HSMC_CYCLE	0x0003_0003	The read and write operations last three Master Clock cycles and provide one hold cycle.
WRITE_MODE	1	Write is controlled with NWE
READ_MODE	1	Read is controlled with NRD

29.10.7 Usage Restriction

The SMC 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 an unpredictable behavior of the SMC.

29.10.7.1For Read Operations

Null but positive setup and hold of address and NRD and/or NCS cannot be guaranteed at the memory interface because of the propagation delay of these signals through external logic and pads. When 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.

29.10.7.2For Write Operations

If a null hold value is programmed on NWE, the SMC can guarantee a positive hold of address, byte select lines, and NCS signal after the rising edge of NWE. This is true for WRITE_MODE = 1 only. Refer to Section 29.12.2 “Early Read Wait State”.

29.10.7.3For Read and Write Operations

A null value for pulse parameters is forbidden and may lead to an 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.

29.11 Scrambling/Unscrambling Function

The external data bus D[15:0] 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 memory device.

The scrambling and unscrambling are performed on-the-fly without additional wait states.

The scrambling method depends on two user-configurable key registers, HSMC_KEY1 and HSMC_KEY2. These key registers are only accessible in Write mode.

The key 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.

The scrambling/unscrambling function is enabled or disabled by configuring specific bits in the HSMC_OCMS and the HSMC_TIMINGSx registers. The bit configuration values to enable memory scrambling are summarized in Table 29-7.

Table 29-7. Scrambling Function Bit Encoding

Memories	Bit Values		
	HSMC_OCMS.SMSE	HSMC_OCMS.SRSE	HSMC_TIMINGSx.OCMS
Off-chip Memories	1	0	1
NAND Flash with NFC	0	1	0

When the NAND Flash memory content is scrambled, the on-chip NFC SRAM page buffer associated for the transfer is also scrambled.

29.12 Automatic Wait States

Under certain circumstances, the SMC automatically inserts idle cycles between accesses to avoid bus contention or operation conflict.

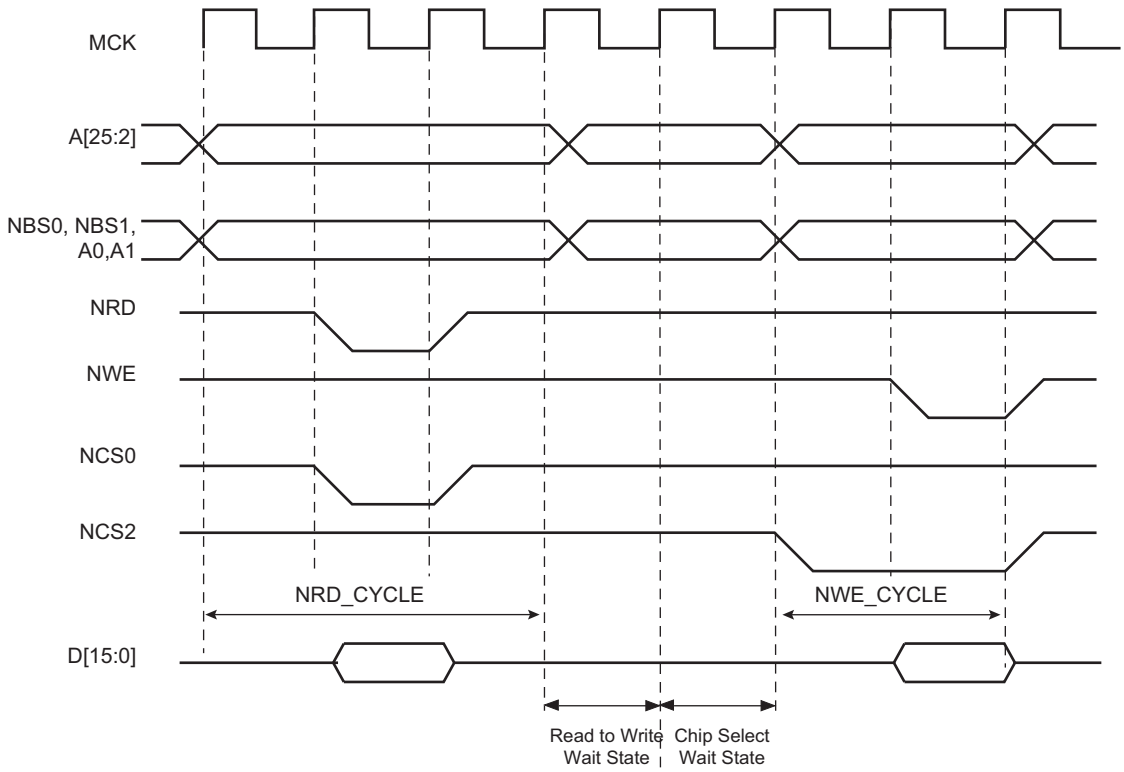
29.12.1 Chip Select Wait States

The SMC 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: NBS0 to NBS1, NWR0 to NWR1, NCS[0..3], and NRD lines. They are all set to 1.

Figure 29-13 illustrates a chip select wait state between access on Chip Select 0 and Chip Select 2.

Figure 29-13. Chip Select Wait State between a Read Access on NCS0 and a Write Access on NCS2



29.12.2 Early Read Wait State

In some cases, the SMC 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 29-14).
- in NCS Write Controlled mode (WRITE_MODE = 0), if there is no hold timing on the NCS signal and the NCS_RD_SETUP parameter is configured to 0, regardless of the Read mode (Figure 29-15). The write operation must end with an NCS rising edge. Without an Early Read Wait State, the write operation could not complete properly.
- in NWE Controlled 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, chip select and byte 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. Refer to Figure 29-16.

Figure 29-14. Early Read Wait State: Write with No Hold Followed by Read with No Setup

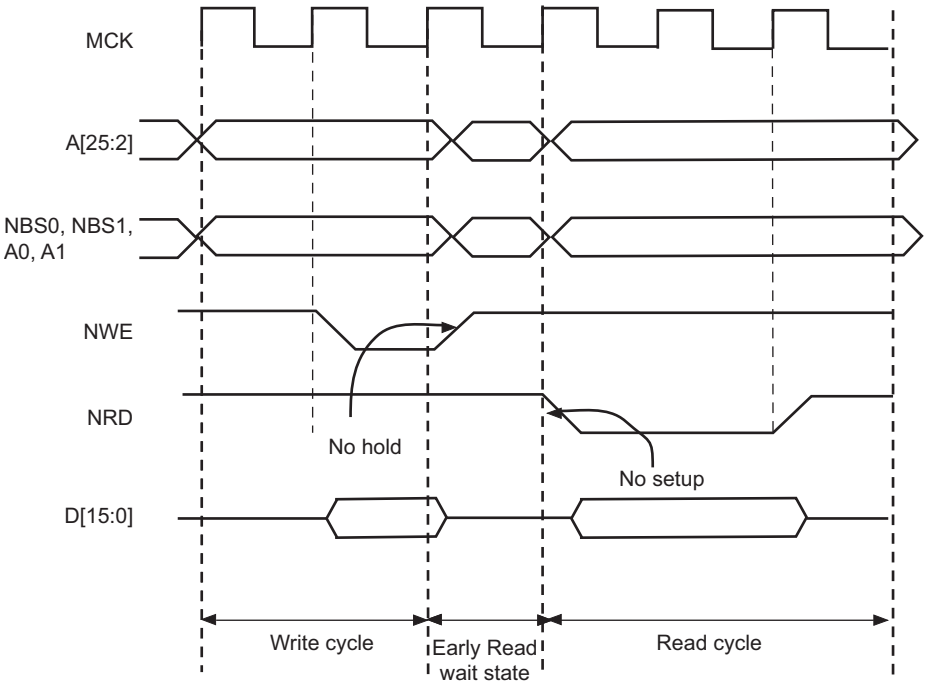


Figure 29-15. Early Read Wait State: NCS Controlled Write with No Hold Followed by a Read with No NCS Setup

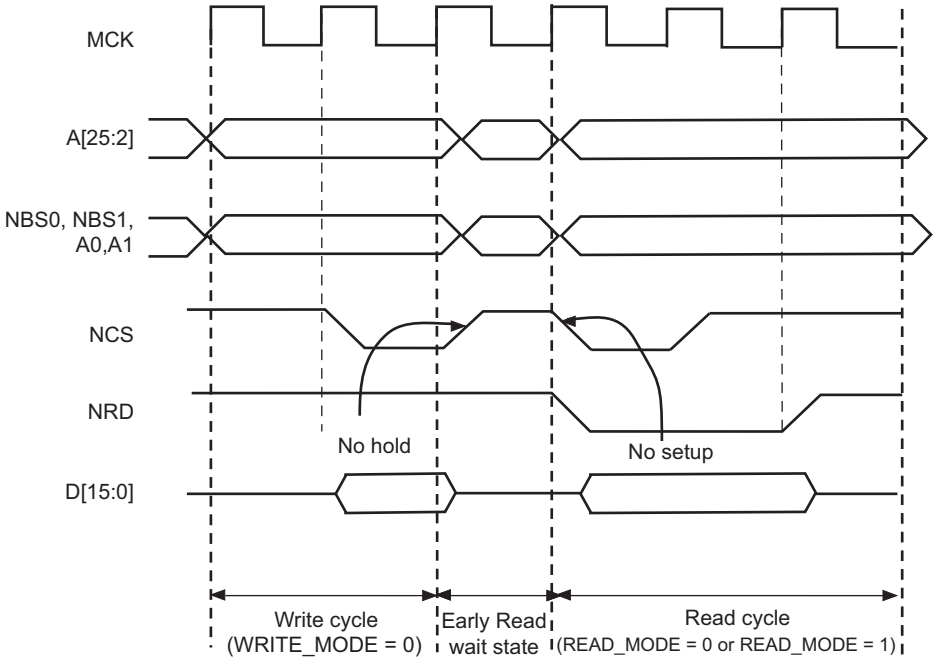
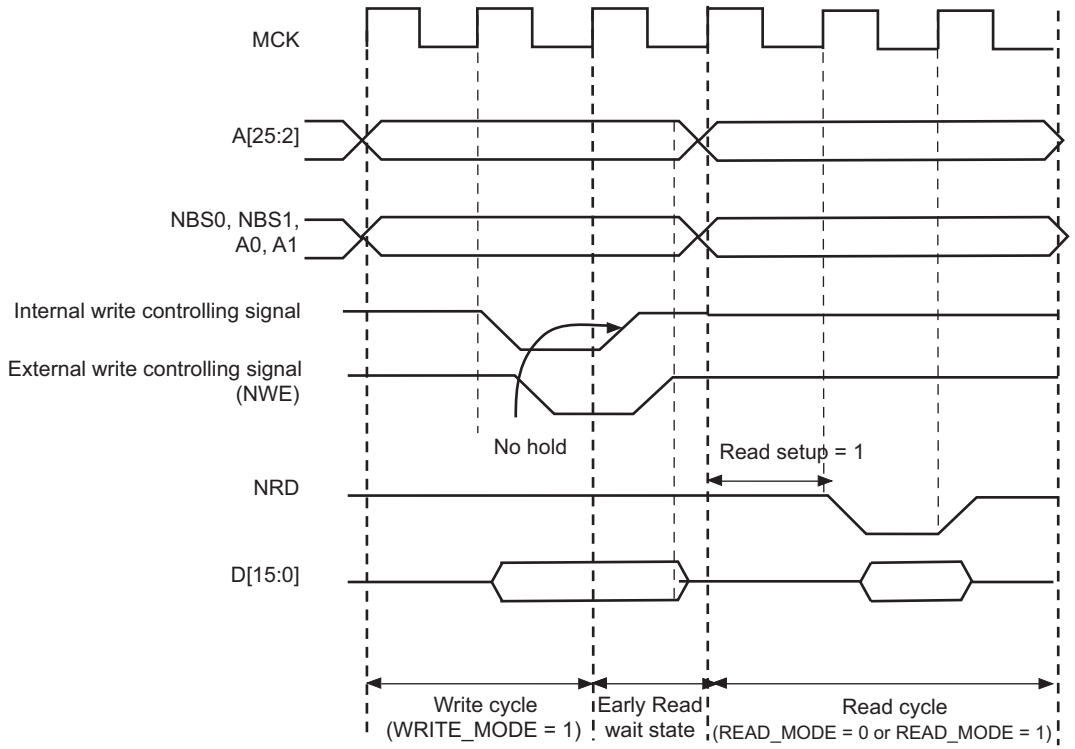


Figure 29-16. Early Read Wait State: NWE-controlled Write with No Hold Followed by a Read with One Setup Cycle



29.12.3 Reload User Configuration Wait State

The user may change any of the configuration parameters by writing the SMC user interface.

When detecting that a new user configuration has been written in the user interface, the SMC inserts a wait state before starting the next access. The so called “Reload User Configuration Wait State” is used by the SMC 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 reprogramming 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.

29.12.3.1User Procedure

To insert a Reload Configuration Wait State, the SMC detects a write access to any HSMC_MODE register of the user interface. If only the timing registers are modified (HSMC_SETUP, HSMC_PULSE, HSMC_CYCLE registers) in the user interface, the user must validate the modification by writing the HSMC_MODE register, even if no change was made on the mode parameters.

29.12.3.2Slow Clock Mode Transition

A Reload Configuration Wait State is also inserted when the Slow Clock Mode is entered or exited, after the end of the current transfer (refer to Section 29.15 “Slow Clock Mode”).

29.12.4 Read to Write Wait State

Due to an internal mechanism, a wait cycle is always inserted between consecutive read and write SMC 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. Refer to Figure 29-13.

29.13 Data Float Wait States

Some memory devices are slow to release the external bus. For such devices, it is necessary to add wait states (data float wait states) after a read access:

- before starting a read access to a different external memory
- before starting a write access to the same device or to a different external one.

The Data Float Output Time (t_{DF}) for each external memory device is programmed in the TDF_CYCLES field of the HSMC_MODE register for the corresponding chip select. The value of TDF_CYCLES indicates the number of data float wait cycles (between 0 and 15) before the external device releases the bus, and represents the time allowed for the data output to go to high impedance after the memory is disabled.

Data float wait states do not delay internal memory accesses. Hence, a single access to an external memory with long t_{DF} will not slow down the execution of a program from internal memory.

The data float wait states management depends on the READ_MODE and the TDF_MODE bits of the HSMC_MODE register for the corresponding chip select.

29.13.1 READ_MODE

Setting READ_MODE to 1 indicates to the SMC that the NRD signal is responsible for turning off the tri-state buffers of the external memory device. The Data Float Period then begins after the rising edge of the NRD signal and lasts TDF_CYCLES MCK cycles.

When the read operation is controlled by the NCS signal (READ_MODE = 0), the TDF_CYCLES field in HSMC_MODEx gives the number of MCK cycles during which the data bus remains busy after the rising edge of NCS.

Figure 29-17 illustrates the Data Float Period in NRD-controlled mode (READ_MODE = 1), assuming a data float period of two cycles (TDF_CYCLES = 2). Figure 29-18 shows the read operation when controlled by NCS (READ_MODE = 0) and the TDF_CYCLES parameter equals 3.

Figure 29-17. TDF Period in NRD Controlled Read Access (TDF = 2)

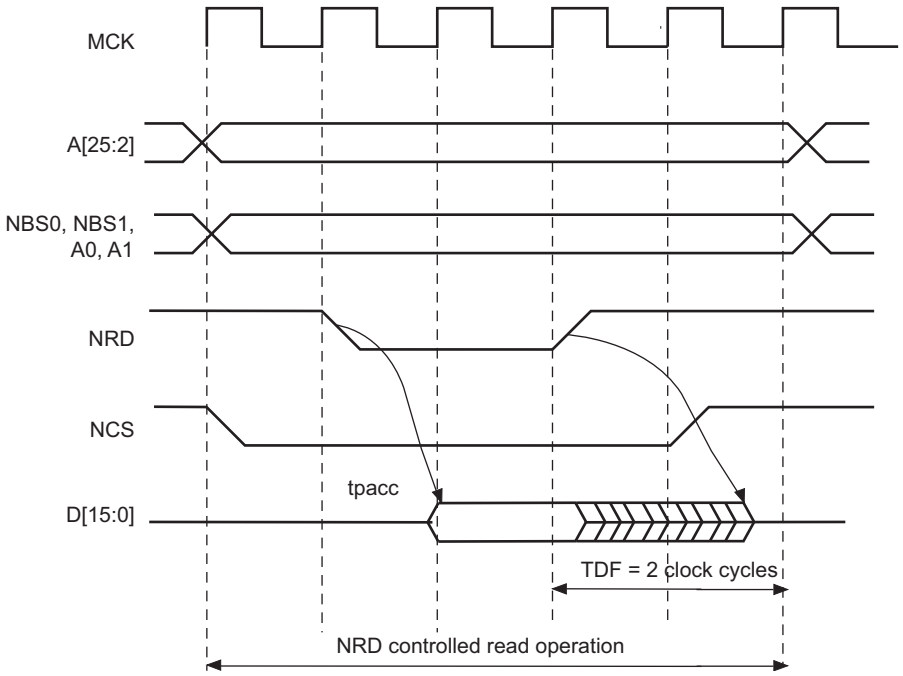
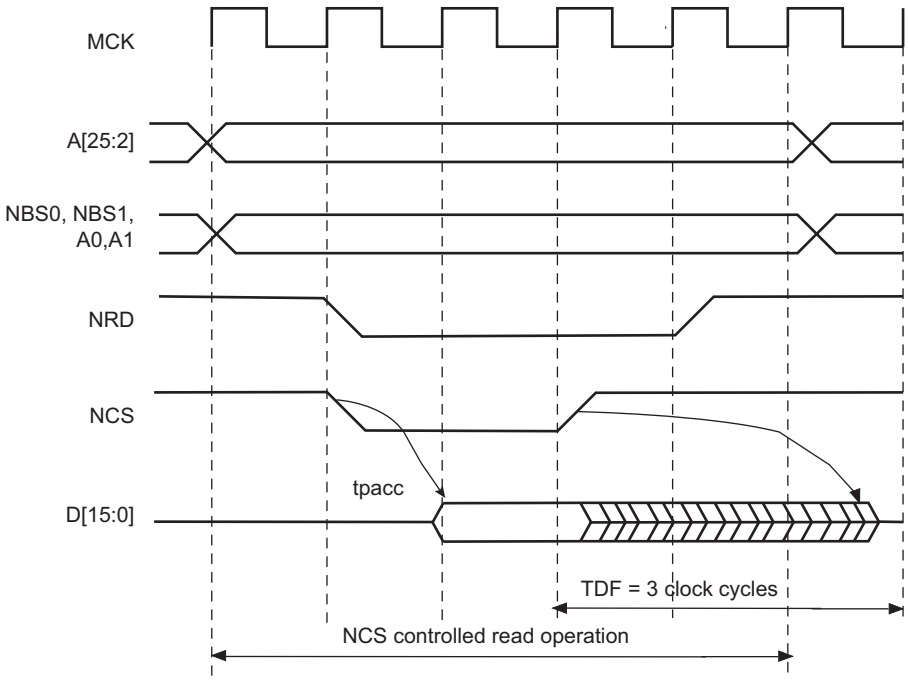


Figure 29-18. TDF Period in NCS Controlled Read Operation (TDF = 3)



29.13.2 TDF Optimization Enabled (TDF_MODE = 1)

When the TDF_MODE of the HSMC_MODE register is set to 1 (TDF optimization is enabled), the SMC takes advantage of the setup period of the next access to optimize the number of wait states cycle to insert.

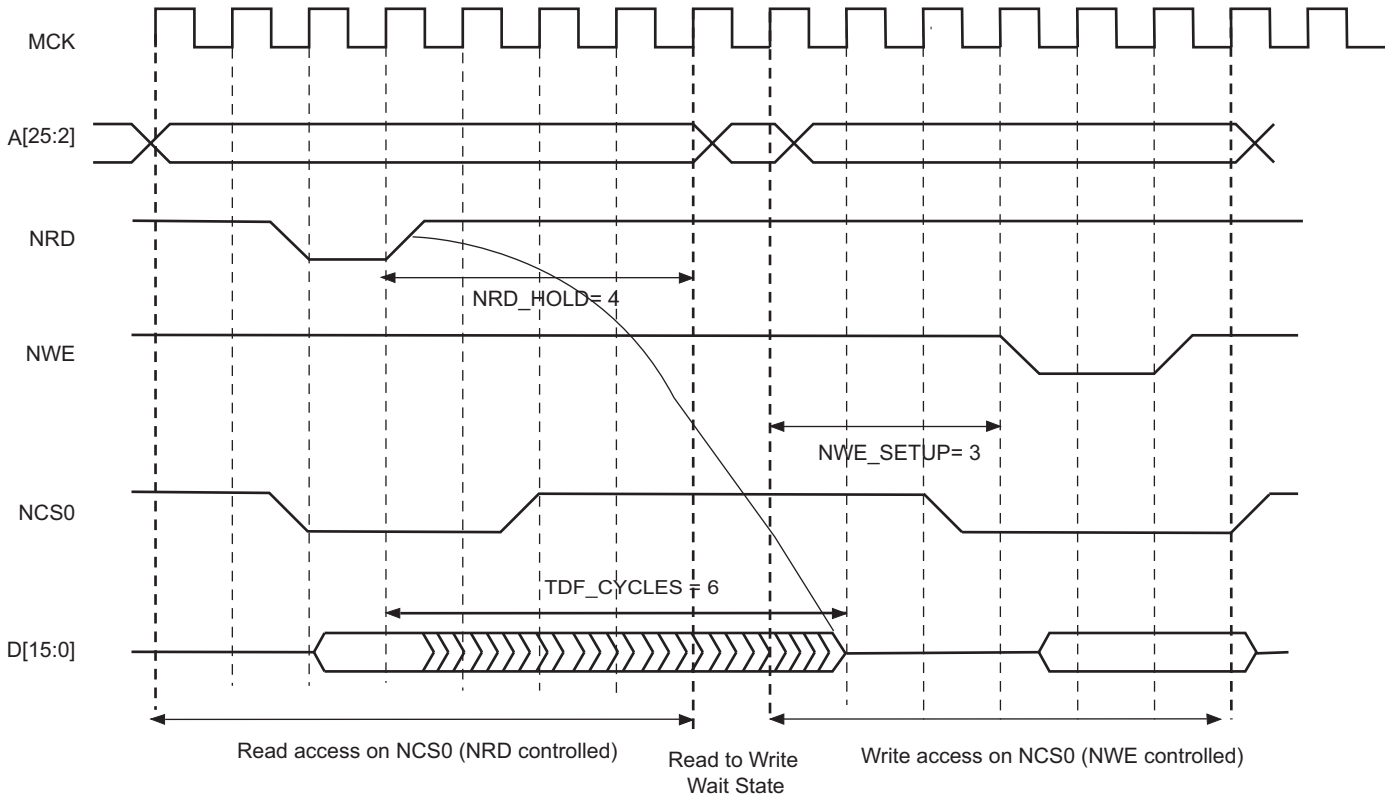
Figure 29-19 shows a read access controlled by NRD, followed by a write access controlled by NWE, on Chip Select 0. Chip Select 0 has been programmed with:

NRD_HOLD = 4; READ_MODE = 1 (NRD controlled)

NWE_SETUP = 3; WRITE_MODE = 1 (NWE controlled)

TDF_CYCLES = 6; TDF_MODE = 1 (optimization enabled).

Figure 29-19. TDF Optimization: No TDF wait states are inserted if the TDF period is over when the next access begins



29.13.3 TDF Optimization Disabled (TDF_MODE = 0)

When optimization is disabled, TDF wait states are inserted at the end of the read transfer, so that the data float period ends when the second access begins. If the hold period of the read1 controlling signal overlaps the data float period, no additional TDF wait states will be inserted.

Figure 29-20, Figure 29-21 and Figure 29-22 illustrate the cases:

- read access followed by a read access on another chip select,
- read access followed by a write access on another chip select,
- read access followed by a write access on the same chip select,

with no TDF optimization.

Figure 29-20. TDF Optimization Disabled (TDF Mode = 0). TDF wait states between 2 read accesses on different chip selects

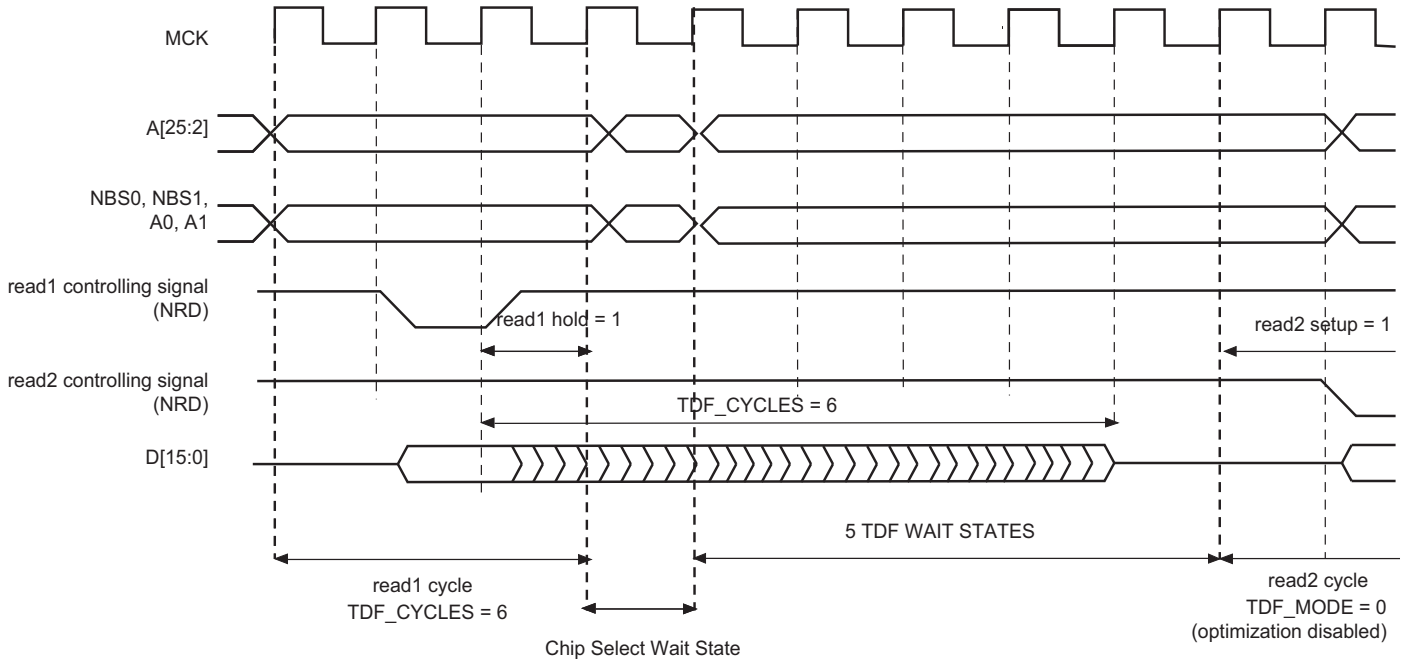


Figure 29-21. TDF Mode = 0: TDF wait states between a read and a write access on different chip selects

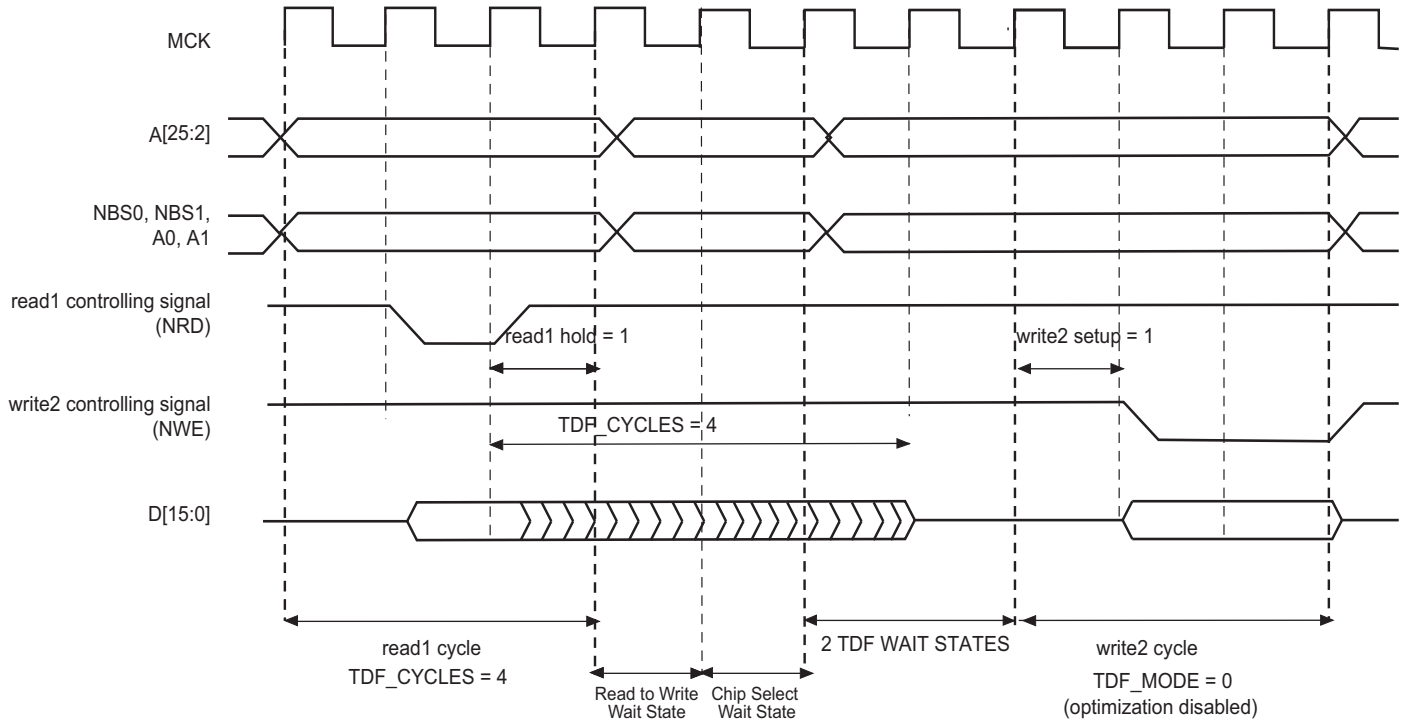
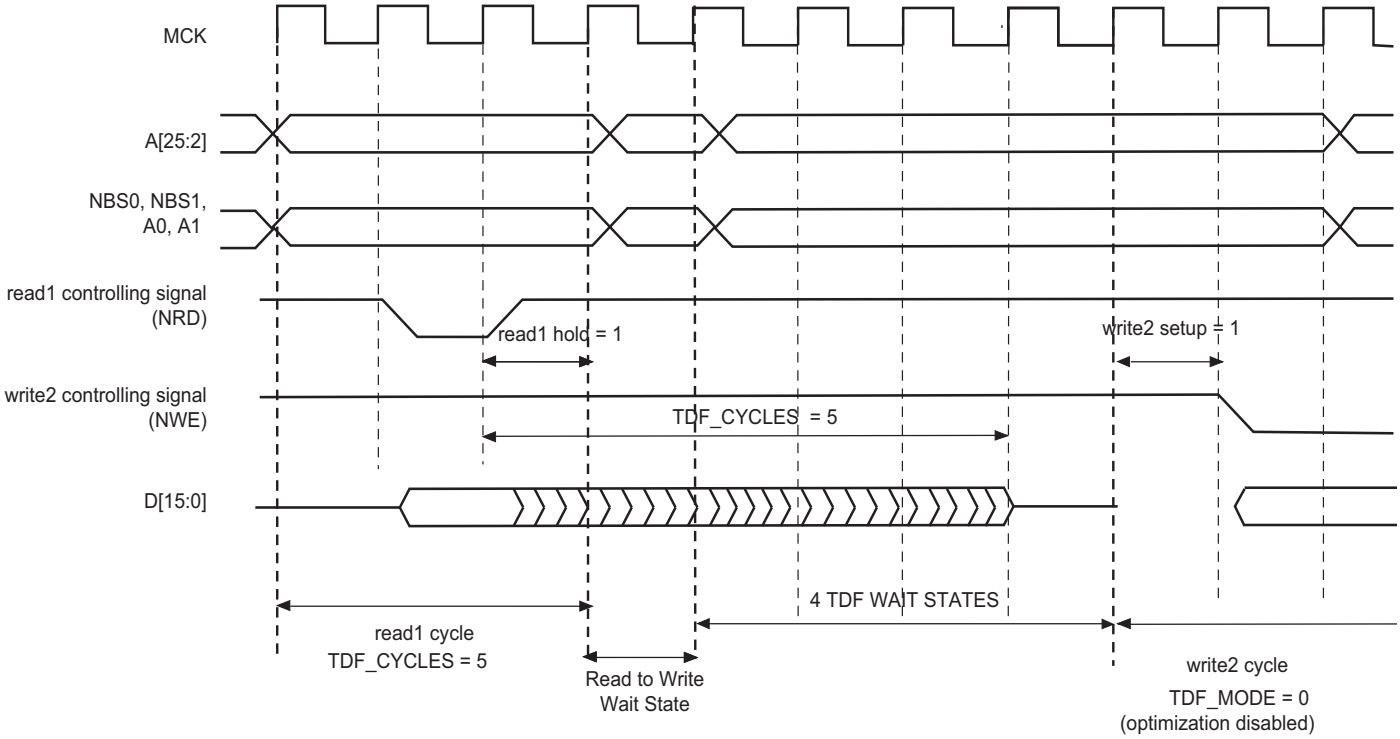


Figure 29-22. TDF Mode = 0: TDF wait states between read and write accesses on the same chip select



29.14 External Wait

Any access can be extended by an external device using the NWAIT input signal of the SMC. The EXNW_MODE field of the HSMC_MODE register on the corresponding chip select must be set to either ‘10’ (Frozen mode) or ‘11’ (Ready mode). When the 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.

29.14.1 Restriction

When one of the EXNW_MODE is enabled, it is mandatory to program at least one hold cycle for the read/write controlling signal. For that reason, the NWAIT signal cannot be used in Slow Clock Mode (Section 29.15 “Slow Clock Mode”).

The NWAIT signal is assumed to be a response of the external device to the read/write request of the SMC. NWAIT is then examined by the SMC 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 the SMC behavior.

29.14.2 Frozen Mode

When the external device asserts the NWAIT signal (active low), and after an internal synchronization of this signal, the SMC state is frozen, i.e., SMC internal counters are frozen, and all control signals remain unchanged. When the resynchronized NWAIT signal is deasserted, the SMC completes the access, resuming the access from the point where it was stopped. Refer to Figure 29-23. This mode must be selected when the external device uses the NWAIT signal to delay the access and to freeze the SMC.

The assertion of the NWAIT signal outside the expected period is ignored as illustrated in Figure 29-24.

Figure 29-23. Write Access with NWAIT Assertion in Frozen Mode (EXNW_MODE = 10)

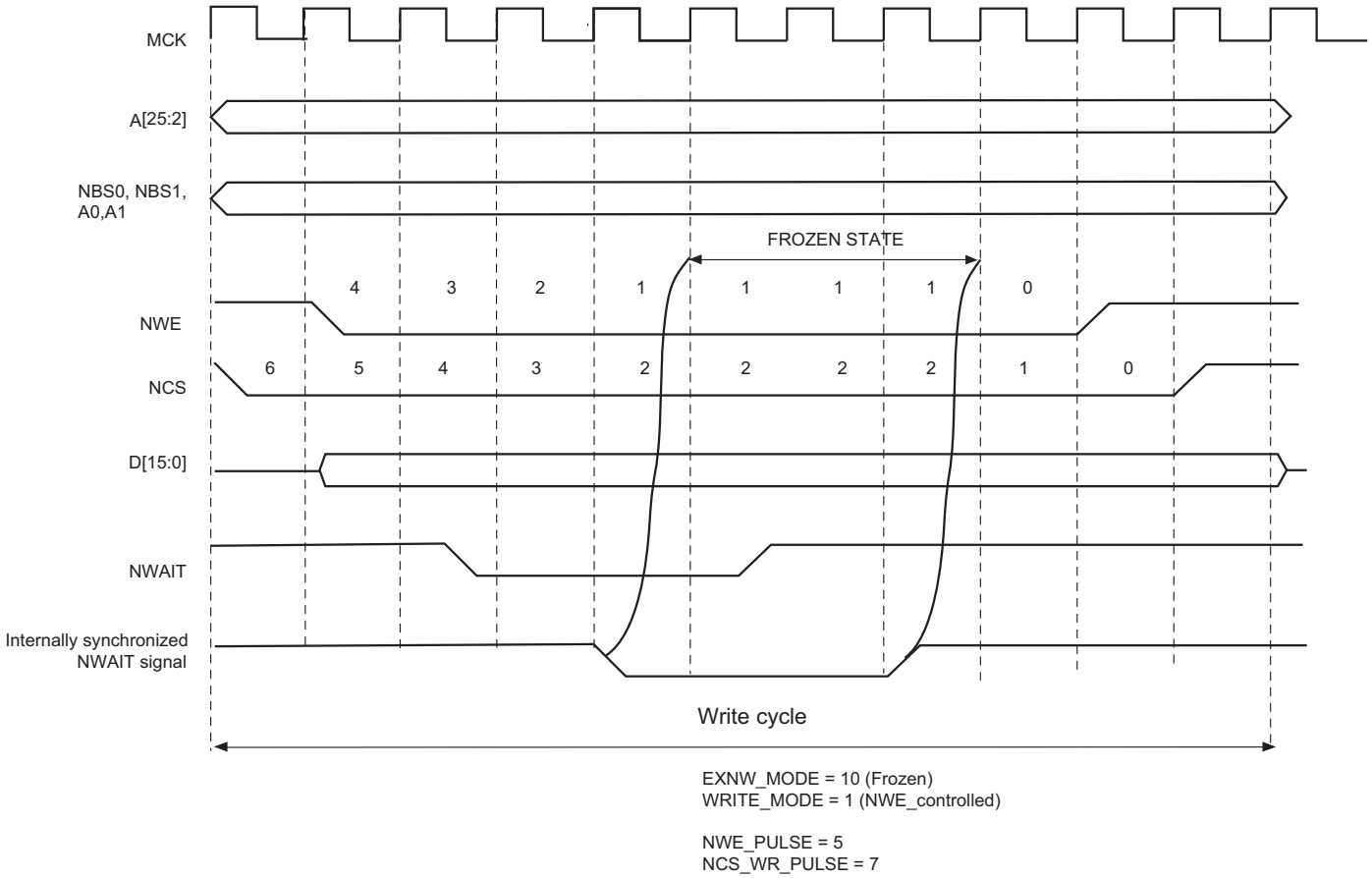
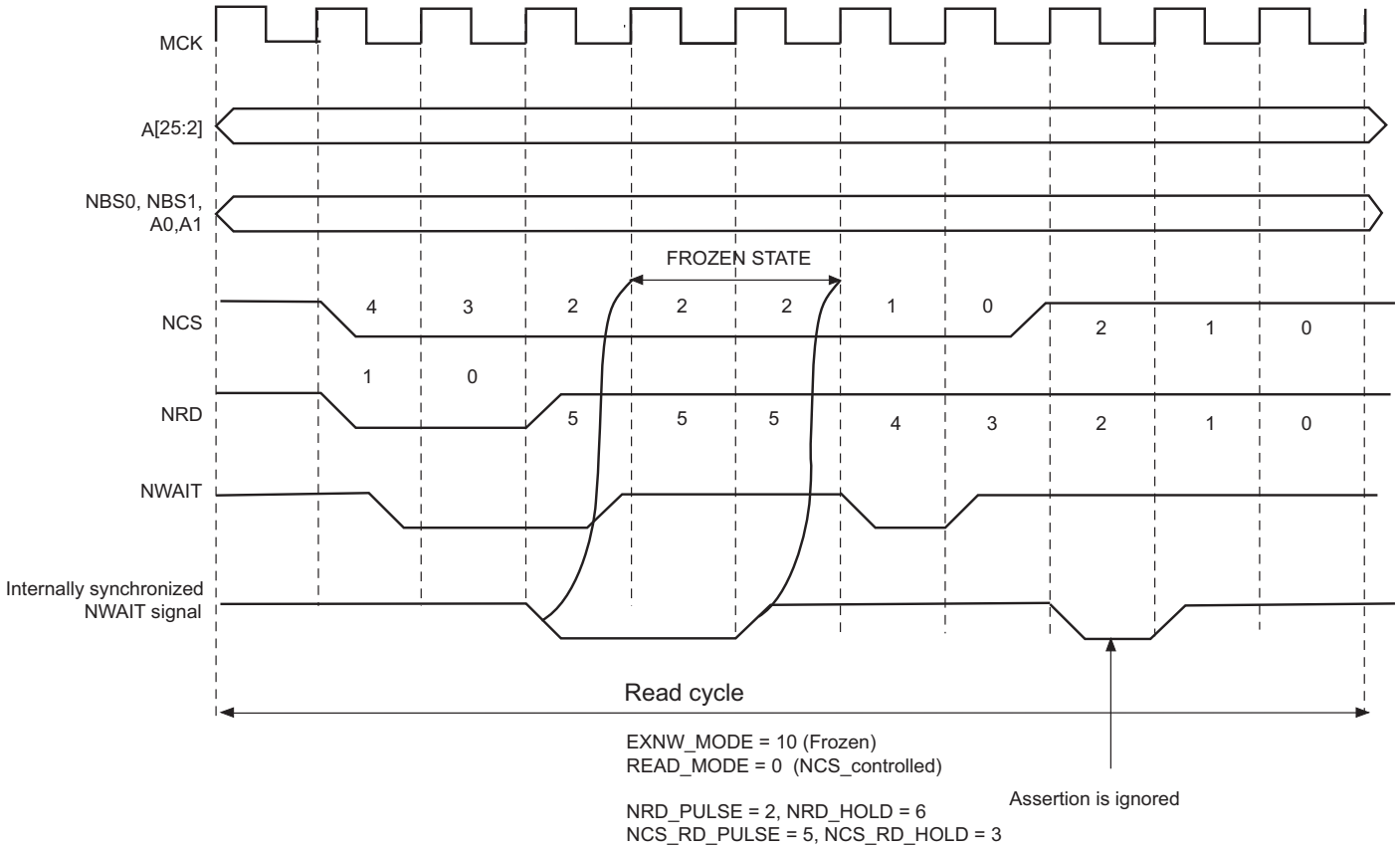


Figure 29-24. Read Access with NWAIT Assertion in Frozen Mode (EXNW_MODE = 10)



29.14.3 Ready Mode

In Ready mode (EXNW_MODE = 11), the SMC behaves differently. Normally, the SMC 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 SMC suspends the access as shown in Figure 29-25 and Figure 29-26. 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 29-26.

Figure 29-25. NWAIT Assertion in Write Access: Ready Mode (EXNW_MODE = 11)

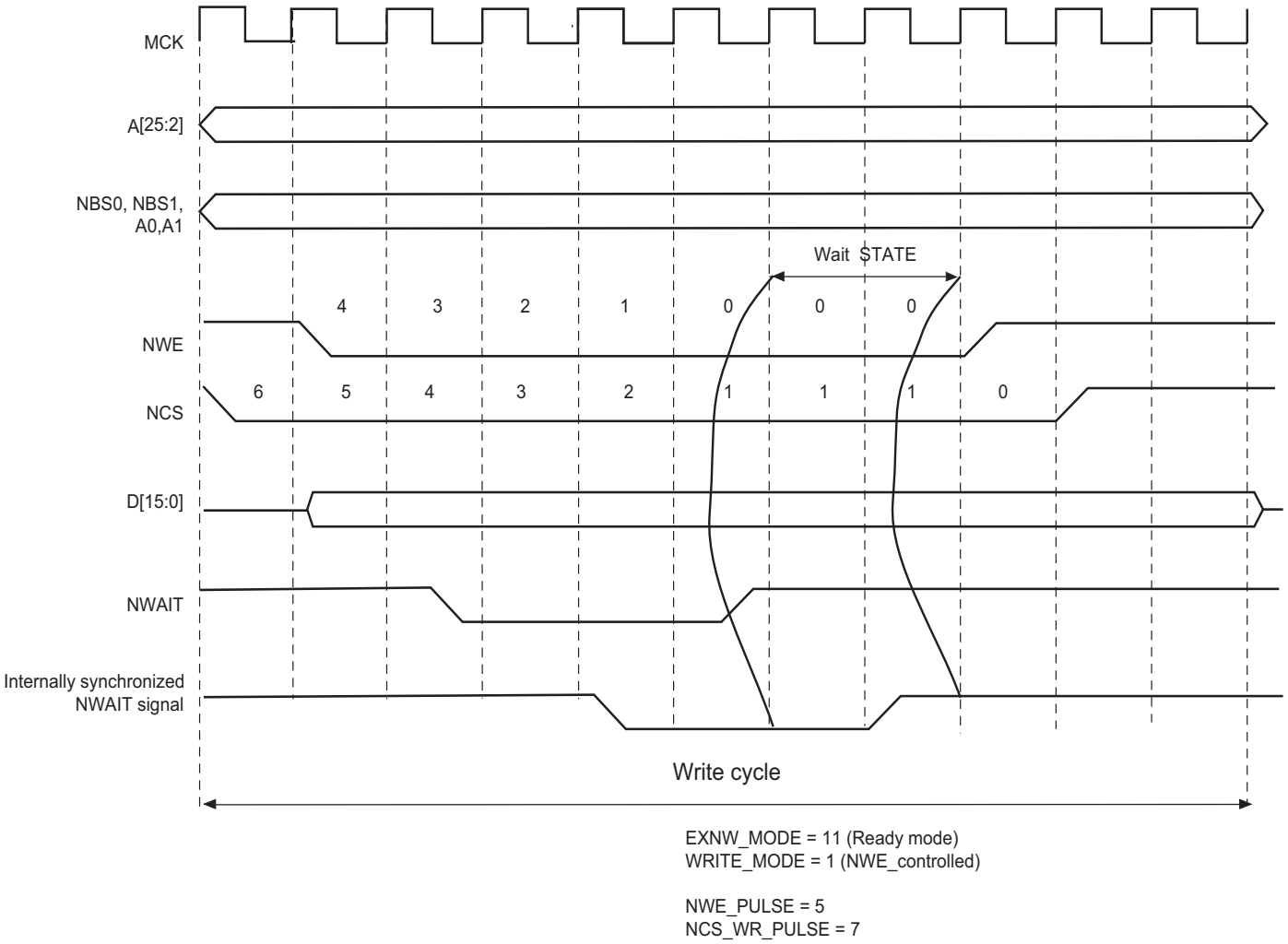
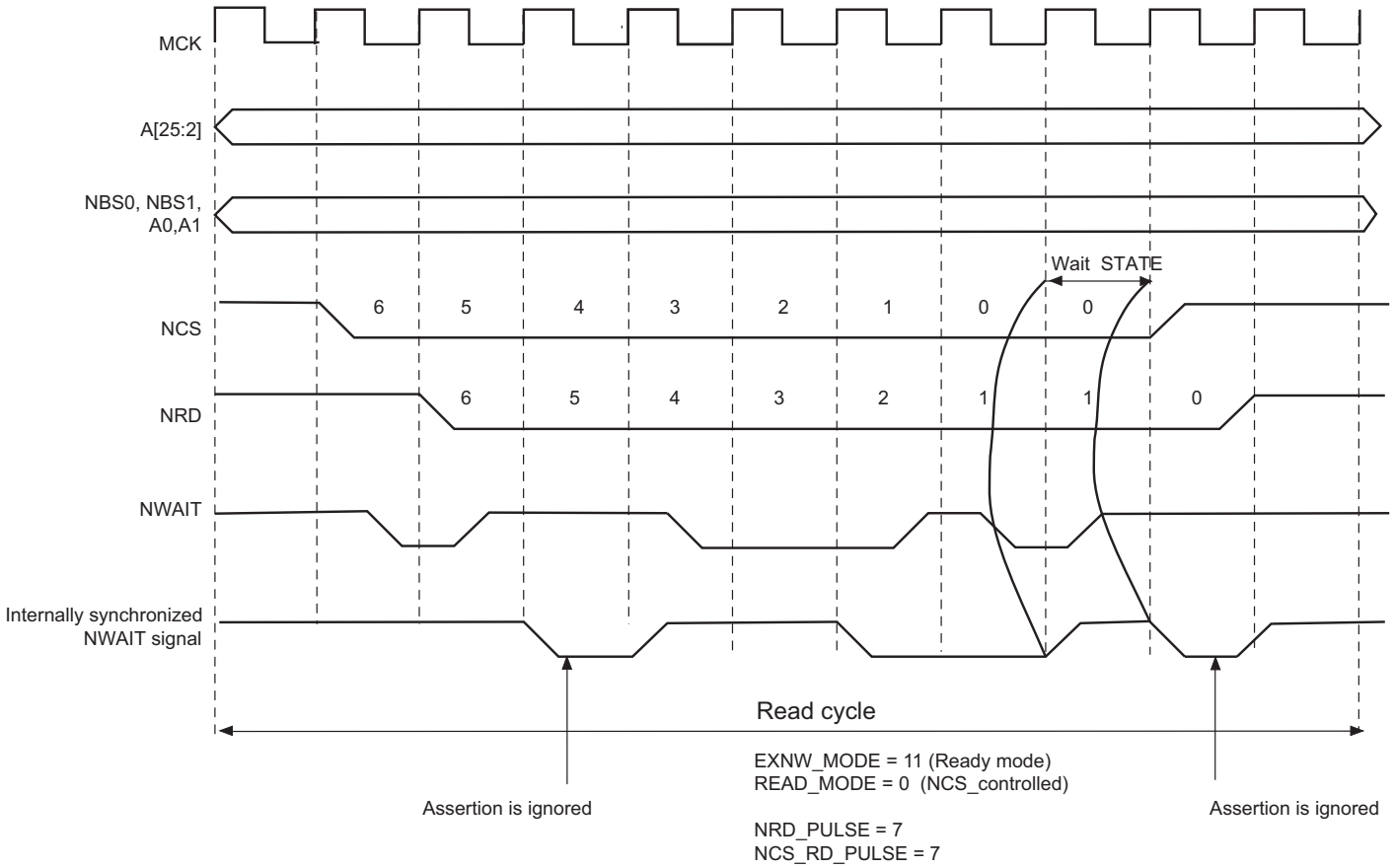


Figure 29-26. NWAIT Assertion in Read Access: Ready Mode (EXNW_MODE = 11)



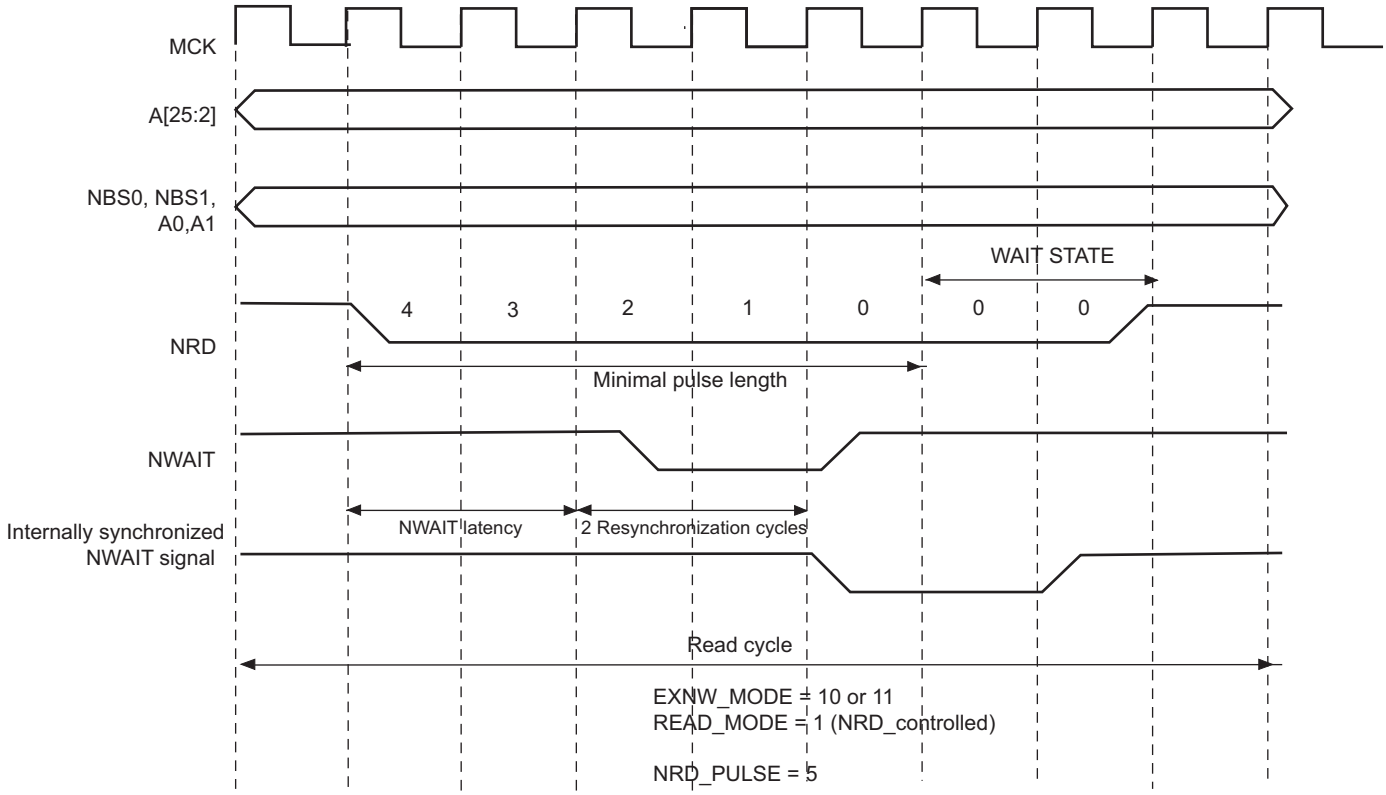
29.14.4 NWAIT Latency and Read/Write Timings

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 + 1 cycle. Otherwise, the SMC 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 on Figure 29-27.

When 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 29-27. NWAIT Latency



29.15 Slow Clock Mode

The SMC is able to automatically apply a set of “Slow Clock mode” read/write waveforms when an internal signal driven by the Power Management Controller is asserted because MCK has been turned to a very slow clock rate (typically 32 kHz clock rate). In this mode, the user-programmed waveforms are ignored and the Slow Clock mode waveforms are applied. This mode is provided so as to avoid reprogramming the User Interface with appropriate waveforms at very slow clock rate. When activated, the Slow mode is active on all chip selects.

29.15.1 Slow Clock Mode Waveforms

Figure 29-28 illustrates the read and write operations in Slow Clock mode. They are valid on all chip selects. Table 29-8 indicates the value of read and write parameters in Slow Clock mode.

Figure 29-28. Write/Read Cycles in Slow Clock Mode

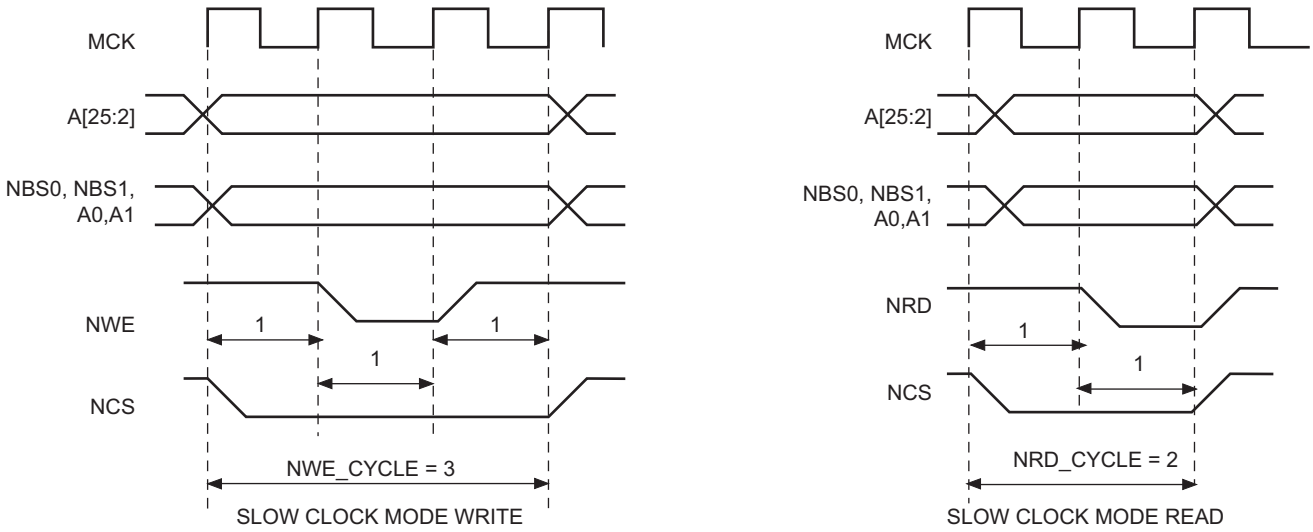


Table 29-8. Read and Write Timing Parameters in Slow Clock Mode

Read Parameters	Duration (cycles)	Write Parameters	Duration (cycles)
NRD_SETUP	1	NWE_SETUP	1
NRD_PULSE	1	NWE_PULSE	1
NCS_RD_SETUP	0	NCS_WR_SETUP	0
NCS_RD_PULSE	2	NCS_WR_PULSE	3
NRD_CYCLE	2	NWE_CYCLE	3

29.15.2 Switching from (to) Slow Clock Mode to (from) Normal Mode

When switching from Slow Clock mode to Normal mode, the current Slow Clock mode transfer is completed at high clock rate, with the set of Slow Clock mode parameters. Refer to Figure 29-29. The external device may not be fast enough to support such timings.

Figure 29-30 illustrates the recommended procedure to properly switch from one mode to the other.

Figure 29-29. Clock Rate Transition occurs while the SMC is performing a Write Operation

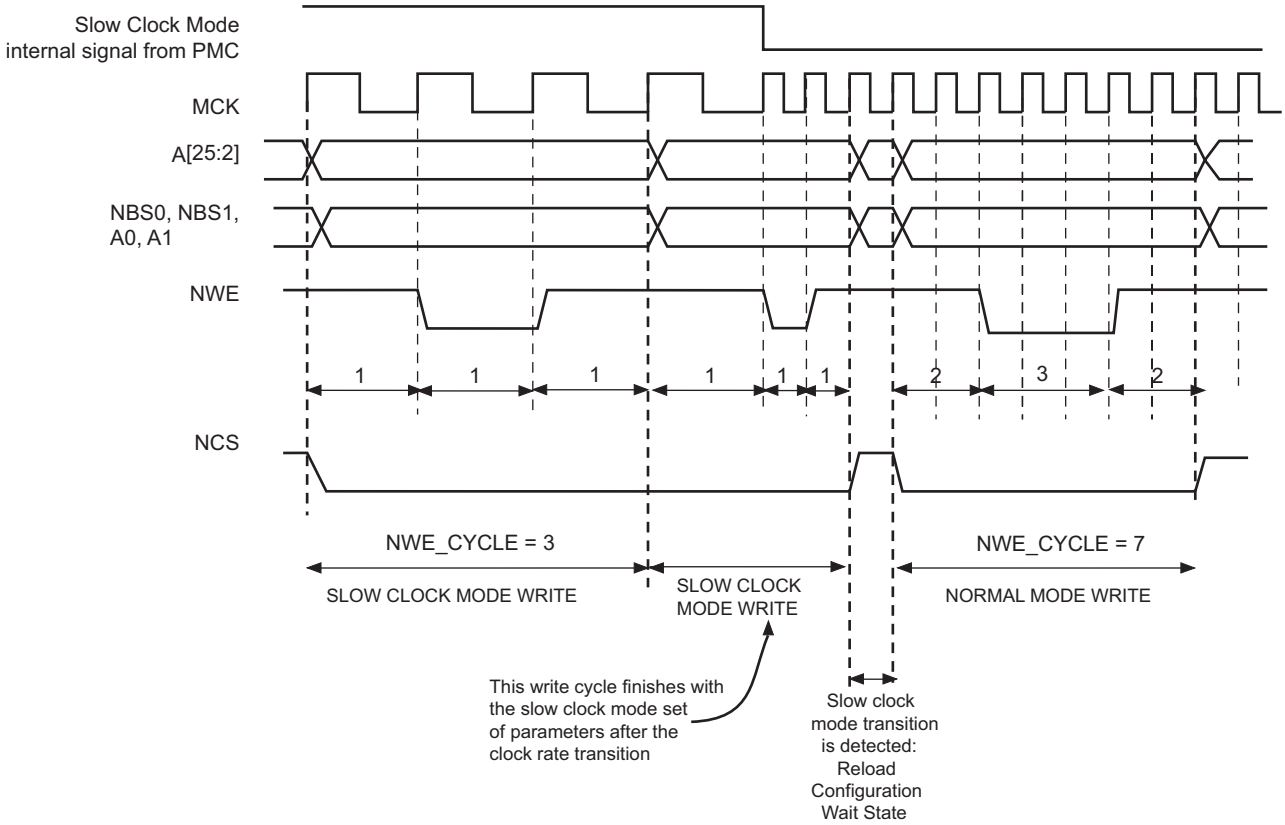
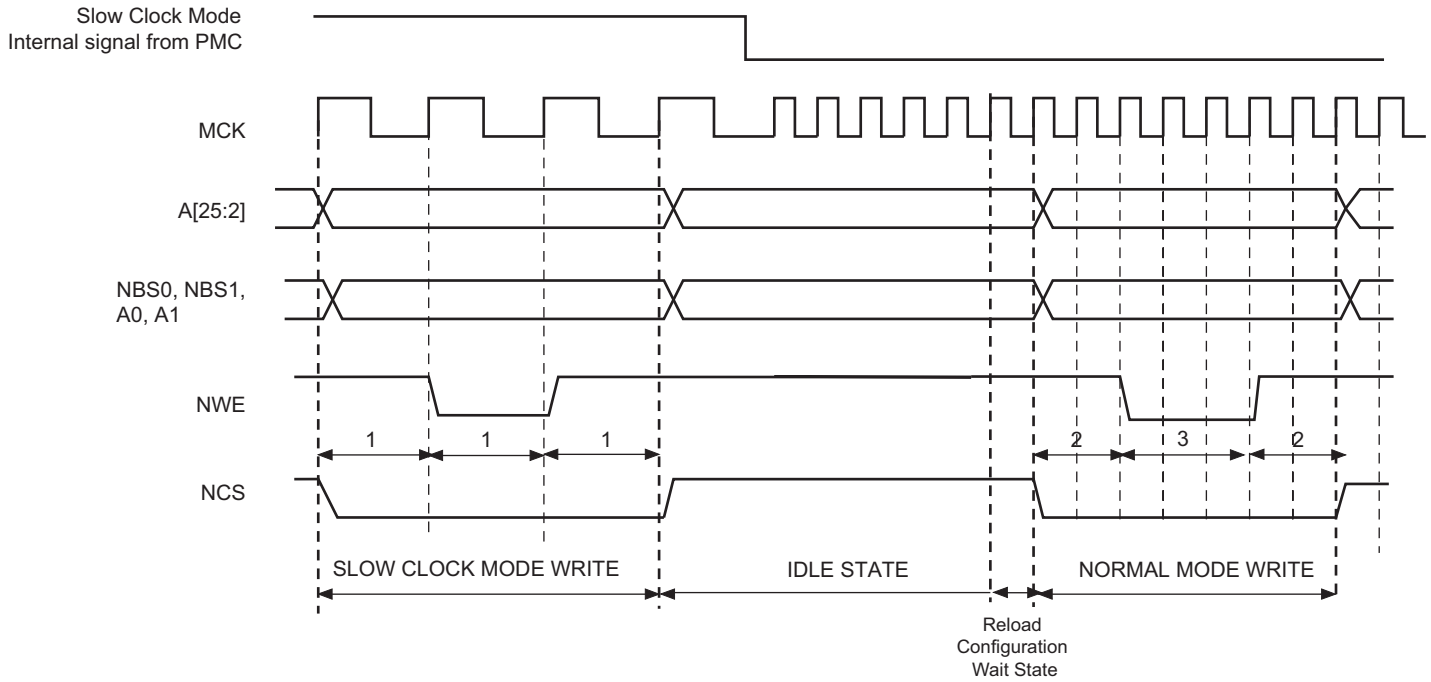


Figure 29-30. Recommended Procedure to Switch from Slow Clock Mode to Normal Mode or from Normal Mode to Slow Clock Mode



29.16 Register Write Protection

To prevent any single software error that may corrupt SMC behavior, selected registers can be write-protected by setting the WPEN bit in the Write Protection Mode Register (HSMC_WPMR).

If a write access in a write-protected register is detected, then the WPVS flag in the 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 reset after reading the HSMC_WPSR.

The following registers can be write-protected:

- Setup Register
- Pulse Register
- Cycle Register
- Timings Register
- Mode Register

29.17 NFC Operations

29.17.1 NFC Overview

The NFC handles all the command, address and data sequences of the NAND low level protocol. An SRAM is used as an internal read/write buffer when data is transferred from or to the NAND.

29.17.2 NFC Control Registers

NAND Flash Read and NAND Flash Program operations can be performed through the NFC Command Registers. In order to minimize CPU intervention and latency, commands are posted in a command buffer. This buffer provides zero wait state latency. The detailed description of the command encoding scheme is explained below.

The NFC handles an automatic transfer between the external NAND Flash and the chip via the NFC SRAM. It is done via NFC Command Registers.

The NFC Command Registers are very efficient to use. When writing to these registers:

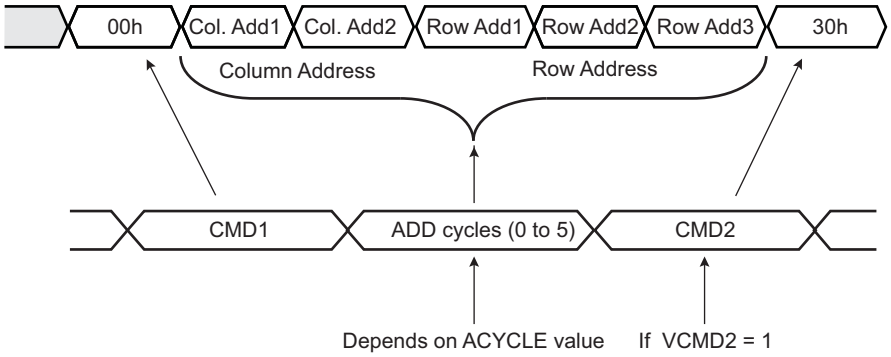
- the address of the register (NFCADDR_CMD) is the command used
- the data of the register (NFCDATA_ADDT) is the address to be sent to the NAND Flash

So, in one single access the command is sent and immediately executed by the NFC. Two commands can even be programmed within a single access (CMD1, CMD2) depending on the VCMD2 value.

The NFC can send up to five address cycles.

Figure 29-31 shows a typical NAND Flash Page Read Command of a NAND Flash Memory and correspondence with NFC Address Command Register.

Figure 29-31. NFC/NAND Flash Access Example



For more details refer to Section 29.17.2.2 “NFC Address Command”.

Reading the NFC Command Register (to any address) will give the status of the NFC. This is especially useful to know if the NFC is busy, for example.

29.17.2.1Building NFC Address Command Example

The base address is made of HOST_ADDR address.

Page read operation example:

```
// Build the Address Command (NFCADDR_CMD)
AddressCommand = (HOST_ADDR                          |
                  NFCWR=0                            |// NFC Read Data from NAND
Flash
                  DATAEN=1                           |    // NFC Data phase
Enable.
                  CSID=1                             |       // Chip Select ID =
1
                 ACYCLE= 5                           |     // Number of address
cycle.
                 VCMD2=1                             | // CMD2 is sent after
Address Cycles
                 CMD2=0x30                           | // CMD2 = 30h
                 CMD1=0x0)    // CMD1 = Read Command = 00h

  // Set the Address for Cycle 0
  HSMC_ADDR = Col. Add1

  // Write command with the Address Command built above
*AddressCommand = (Col. Add2   |// ADDR_CYCLE1
           Row Add1  | // ADDR_CYCLE2
           Row Add2  |// ADDR_CYCLE3
           Row Add3  )// ADDR_CYCLE4
```

29.17.2.2NFC Address Command

Name: NFCADDR_CMD

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	NFCWR	DATAEN	CSID

23	22	21	20	19	18	17	16
CSID		ACYCLE			VCMD2	CMD2	

15	14	13	12	11	10	9	8
CMD2						CMD1	

7	6	5	4	3	2	1	0
CMD1						–	–

- **CMD1: Command Register Value for Cycle 1**

When a write access occurs, the NFC sends this command.

- **CMD2: Command Register Value for Cycle 2**

When a write access occurs with the VCMD2 field set, the NFC sends this command after CMD1.

- **VCMD2: Valid Cycle 2 Command**

When set to true, the CMD2 field is issued after the address cycle.

- **ACYCLE: Number of Address Required for the Current Command**

When ACYCLE field is different from zero, ACYCLE Address cycles are performed after Command Cycle 1. The maximum number of cycles is 5.

- **CSID: Chip Select Identifier**

Chip select used

- **DATAEN: NFC Data Phase Enable**

When set to true, the NFC will automatically read or write data after the command.

- **NFCWR: NFC Write Enable**

0: NFC reads data from the NAND Flash.

1: NFC writes data into the NAND Flash.

29.17.2.3NFC Data Address

Name: NFCDATA_ADDT

Access: Write-only

31	30	29	28	27	26	25	24
ADDR_CYCLE4							
23	22	21	20	19	18	17	16
ADDR_CYCLE3							
15	14	13	12	11	10	9	8
ADDR_CYCLE2							
7	6	5	4	3	2	1	0
ADDR_CYCLE1							

- **ADDR_CYCLE1: NAND Flash Array Address Cycle 1**

When less than five address cycles are used, ADDR_CYCLE1 is the first byte written to the NAND Flash.

When five address cycles are used, ADDR_CYCLE1 is the second byte written to NAND Flash.

- **ADDR_CYCLE2: NAND Flash Array Address Cycle 2**

When less than five address cycles are used, ADDR_CYCLE2 is the second byte written to the NAND Flash.

When five address cycles are used, ADDR_CYCLE2 is the third byte written to the NAND Flash.

- **ADDR_CYCLE3: NAND Flash Array Address Cycle 3**

When less than five address cycles are used, ADDR_CYCLE3 is the third byte written to the NAND Flash.

When five address cycles are used, ADDR_CYCLE3 is the fourth byte written to the NAND Flash.

- **ADDR_CYCLE4: NAND Flash Array Address Cycle 4**

When less than five address cycles are used, ADDR_CYCLE4 is the fourth byte written to the NAND Flash.

When five address cycles are used, ADDR_CYCLE4 is the fifth byte written to the NAND Flash.

Note: If five address cycles are used, the first address cycle is ADDR_CYCLE0. Refer to “NFC Address Cycle Zero Register”.

29.17.2.4NFC DATA Status

Name: NFCDATA_STATUS

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	NFCBUSY	NFCWR	DATAEN	CSID

23	22	21	20	19	18	17	16
CSID		ACYCLE			VCMD2	CMD2	

15	14	13	12	11	10	9	8
CMD2						CMD1	

7	6	5	4	3	2	1	0
CMD1						–	–

- **CMD1: Command Register Value for Cycle 1**

When a Read or Write Access occurs, the Physical Memory Interface drives the IO bus with CMD1 field during the Command Latch cycle 1.

- **CMD2: Command Register Value for Cycle 2**

When VCMD2 bit is set to true, the Physical Memory Interface drives the IO bus with CMD2 field during the Command Latch cycle 2.

- **VCMD2: Valid Cycle 2 Command**

When set to true, the CMD2 field is issued after addressing cycle.

- **ACYCLE: Number of Address Required for the Current Command**

When ACYCLE field is different from zero, ACYCLE Address cycles are performed after Command Cycle 1.

- **CSID: Chip Select Identifier**

Chip select used

- **DATAEN: NFC Data Phase Enable**

When set to true, the NFC data phase is enabled.

- **NFCWR: NFC Write Enable**

0: NFC is in Read mode.

1: NFC is in Write mode.

- **NFCBUSY: NFC Busy Status Flag**

If set to true, it indicates that the NFC is busy.

29.17.3 NFC Initialization

Prior to any Command and Data Transfer, the SMC User Interface must be configured to meet the device timing requirements.

- Write enable Configuration

Use NWE_SETUP, NWE_PULSE and NWE_CYCLE to define the write enable waveform according to the external device datasheet.

Use TADL field in the HSMC_TIMINGS register to configure the timing between the last address latch cycle and the first rising edge of WEN for data input.

Figure 29-32. Write Enable Timing Configuration

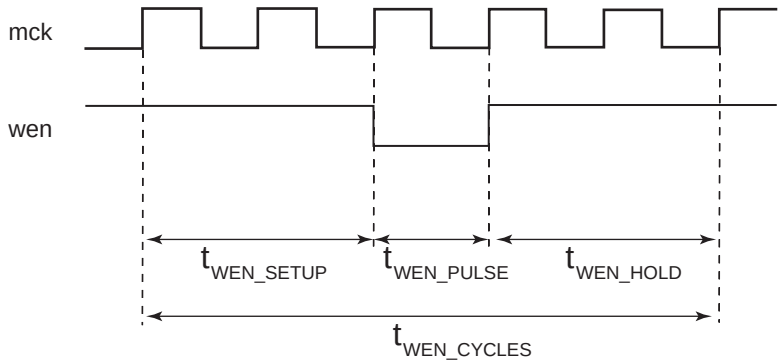
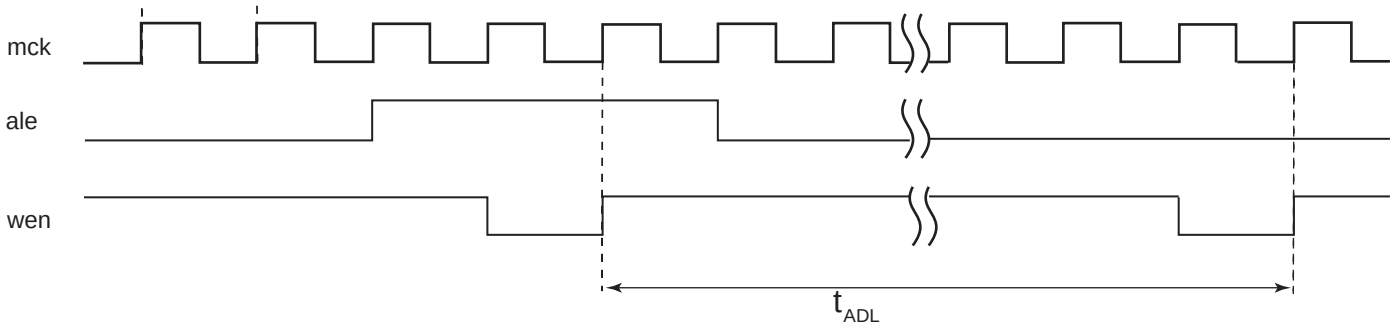


Figure 29-33. Write Enable Timing for NAND Flash Device Data Input Mode



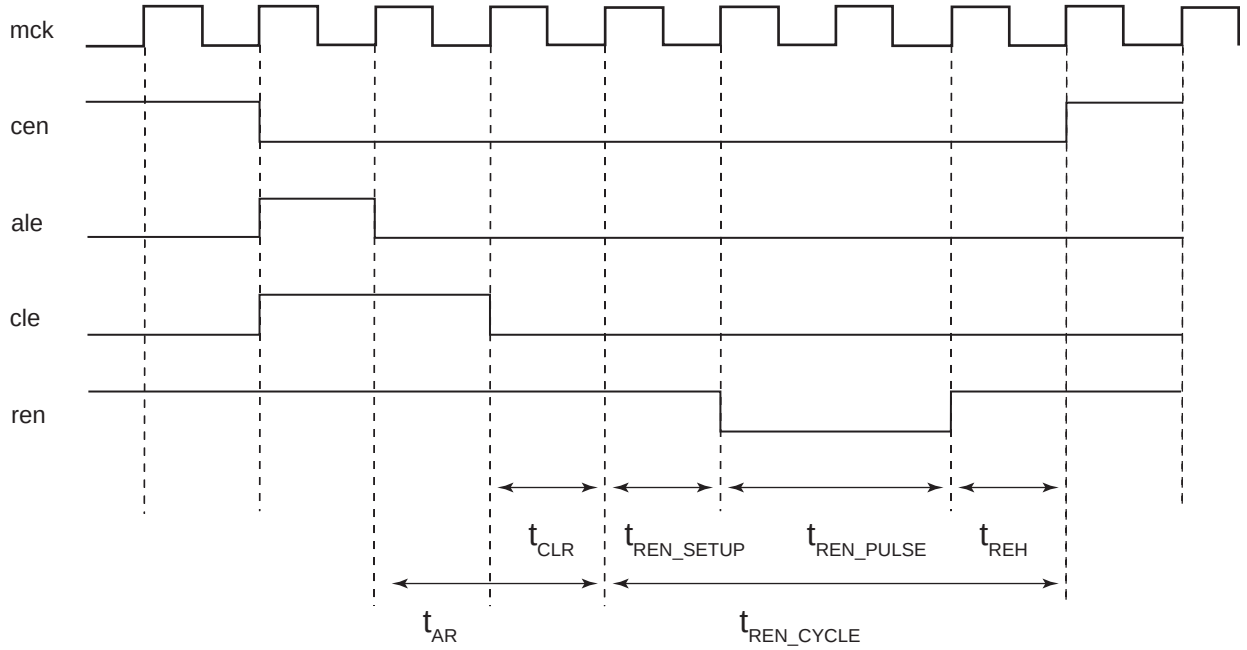
- Read Enable Configuration

Use NRD_SETUP, NRD_PULSE and NRD_CYCLE to define the read enable waveform according to the external device datasheet.

Use TAR field in the HSMC_TIMINGS register to configure the timings between the address latch enable falling edge to read the enable falling edge.

Use TCLR field in the HSMC_TIMINGS register to configure the timings between the command latch enable falling edge to read the enable falling edge.

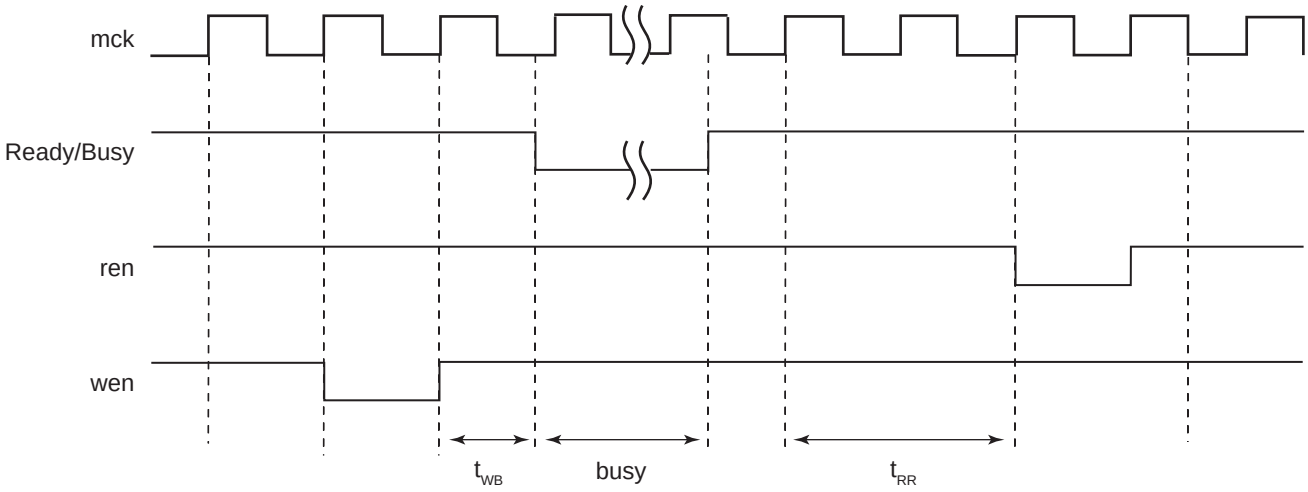
Figure 29-34. Read Enable Timing Configuration Working with NAND Flash Device



- Ready/Busy Signal Timing configuration working with a NAND Flash device

Use TWB field in HSMC_TIMINGS register to configure the maximum elapsed time between the rising edge of the wen signal and the falling edge of the Ready/Busy signal. Use TRR field in the HSMC_TIMINGS register to program the number of clock cycles between the rising edge of the Ready/Busy signal and the falling edge of the ren signal.

Figure 29-35. Ready/Busy Timing Configuration

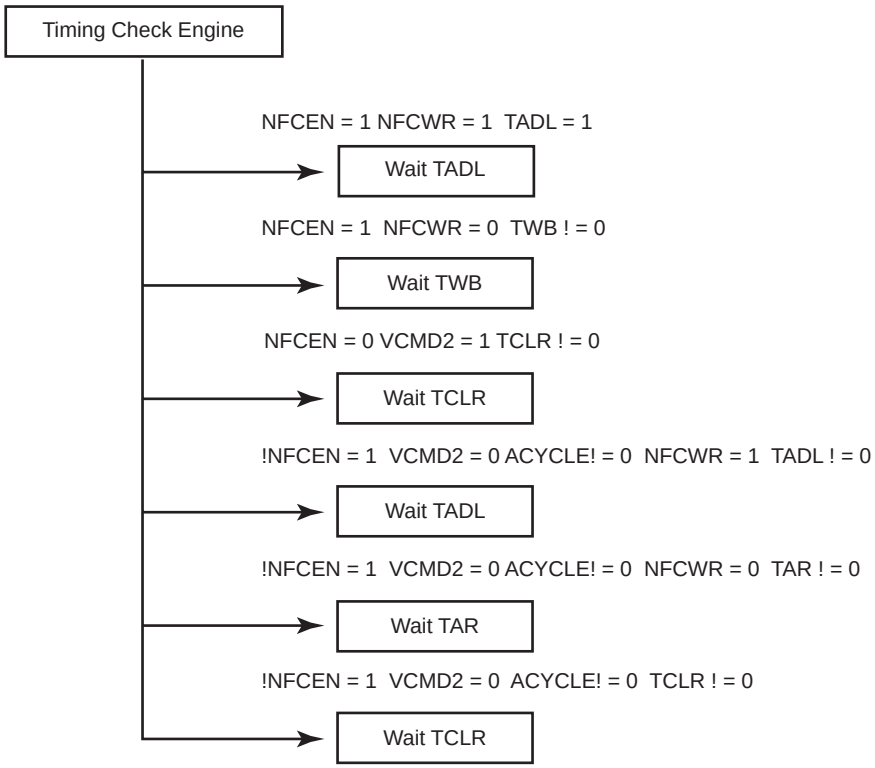


29.17.3.1NFC Timing Engine

When the NFC Command register is written, the NFC issues a NAND Flash Command and optionally performs a data transfer between the NFC SRAM and the NAND Flash device. The NFC Timing Engine guarantees valid NAND Flash timings, depending on the set of parameters decoded from the address bus. These timings are defined in the HSMC_TIMINGS register.

For information on the timing used depending on the command, refer to Figure 29-36.

Figure 29-36. NFC Timing Engine



Refer to the NFC Address Command register description and the Timings Register.

29.17.4 NFC SRAM

29.17.4.1NFC SRAM Mapping

If the NFC is used to read and write data from and to the NAND Flash, the configuration depends on the page size (PAGESIZE field in HSMC_CFG register). Refer to Table 29-9 to Table 29-13 for detailed mapping.

The NFC can handle the NAND Flash with a page size of 8 Kbytes or lower (such as 2 Kbytes, for example). In case of a 4 Kbyte or lower page size, the NFC SRAM can be split into two banks. The BANK bit in the HSMC_BANK register is used to select where NAND flash data are written or read. For an 8 Kbyte page size this field is not relevant.

Note that a “Ping-Pong” mode (write or read to a bank while the NFC writes or reads to another bank) is accessible with the NFC (using two different banks).

If the NFC is not used, the NFC SRAM can be used for a general purpose by the application.

Table 29-9. NFC SRAM Bank Mapping for 512 bytes

Offset	Use	Access
0x00000000–0x000001FF	Main Area Bank 0	Read/Write
0x00000200–0x000003FF	Spare Area Bank 0	Read/Write
0x00001200–0x000013FF	Main Area Bank 1	Read/Write
0x00001400–0x000015FF	Spare Area Bank 1	Read/Write

Table 29-10. NFC SRAM Bank Mapping for 1 Kbyte

Offset	Use	Access
0x00000000–0x000003FF	Main Area Bank 0	Read/Write
0x00000400–0x000005FF	Spare Area Bank 0	Read/Write
0x00001200–0x000015FF	Main Area Bank 1	Read/Write
0x00001600–0x000017FF	Spare Area Bank 1	Read/Write

Table 29-11. NFC SRAM Bank Mapping for 2 Kbytes

Offset	Use	Access
0x00000000–0x000007FF	Main Area Bank 0	Read/Write
0x00000800–0x000009FF	Spare Area Bank 0	Read/Write
0x00001200–0x000019FF	Main Area Bank 1	Read/Write
0x00001A00–0x00001BFF	Spare Area Bank 1	Read/Write

Table 29-12. NFC SRAM Bank Mapping for 4 Kbytes

Offset	Use	Access
0x00000000–0x00000FFF	Main Area Bank 0	Read/Write
0x00001000–0x000011FF	Spare Area Bank 0	Read/Write
0x00001200–0x000021FF	Main Area Bank 1	Read/Write
0x00002200–0x000023FF	Spare Area Bank 1	Read/Write

Table 29-13. NFC SRAM Bank Mapping for 8 Kbytes, only one bank is available

Offset	Use	Access
0x00000000–0x00001FFF	Main Area Bank 0	Read/Write
0x00002000–0x000023FF	Spare Area Bank 0	Read/Write

29.17.4.2NFC SRAM Access Prioritization Algorithm

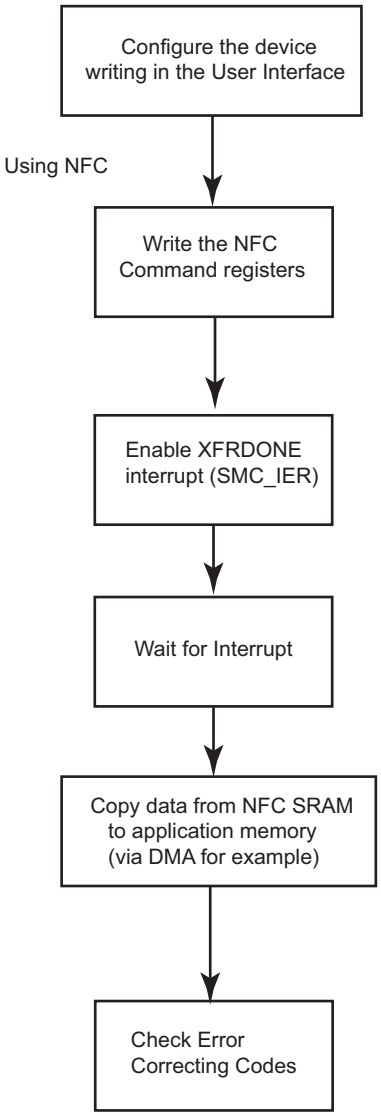
When the NFC is reading from or writing to an NFC SRAM bank, the other bank is available. If an NFC SRAM access occurs when the NFC performs a read or write operation in the same bank, then the access is discarded. The write operation is not performed. The read operation returns undefined data. If this situation is encountered, the AWB status flag located in the NFC Status Register is raised and indicates that a shared resource access violation has occurred.

29.17.5 NAND Flash Operations

This section describes the software operations needed to issue commands to the NAND Flash device and to perform data transfers using the NFC.

29.17.5.1Page Read

Figure 29-37. Page Read Flow Chart

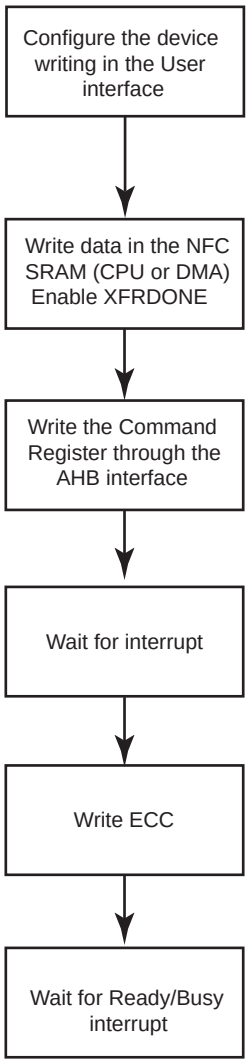


Note that, instead of using the interrupt, one can poll the NFCBUSY flag.

For more information on the NFC Control Register, refer to Section 29.17.2.2 “NFC Address Command”.

29.17.5.2Program Page

Figure 29-38. Program Page Flow Chart



Writing the ECC cannot be done using the NFC; it needs to be done “manually”.

Note that, instead of using the interrupt, one can poll the NFCBUSY flag.

For more information on the NFC Control Register, refer to Section 29.17.2.2 “NFC Address Command”.

29.18 PMECC Controller Functional Description

The Programmable Multibit Error Correcting Code (PMECC) controller is a programmable binary BCH (Bose, Chaudhuri and Hocquenghem) encoder/decoder. This controller can be used to generate redundancy information for both SLC and MLC NAND devices. It supports redundancy for correction of 2, 4, 8, 12, or 24 errors per sector of data. The sector size is programmable and can be set to 512 bytes or 1024 bytes. The PMECC module generates redundancy at encoding time, when a NAND write page operation is performed. The redundancy is appended to the page and written in the spare area. This operation is performed by the processor. It moves the content of the PMECCX registers into the NAND flash memory. The number of registers depends on the selected error correction capability (refer to Table 29-14 "Relevant Redundancy Registers"). This operation shall be executed for each sector. At decoding time, the PMECC module generates the remainders of the received codeword by the minimal polynomials. When all remainders for a given sector are set to zero, no error occurred. When the remainders are different from zero, the codeword is corrupted and further processing is required.

The PMECC module generates an interrupt indicating that an error occurred. The processor must read the PMECC Interrupt Status Register (HSMC_PMECCISR). This register indicates which sector is corrupted.

The processor must execute the following decoding steps to find the error location within a sector:

1. Syndrome computation.
2. Finding the error location polynomial.
3. Finding the roots of the error location polynomial.

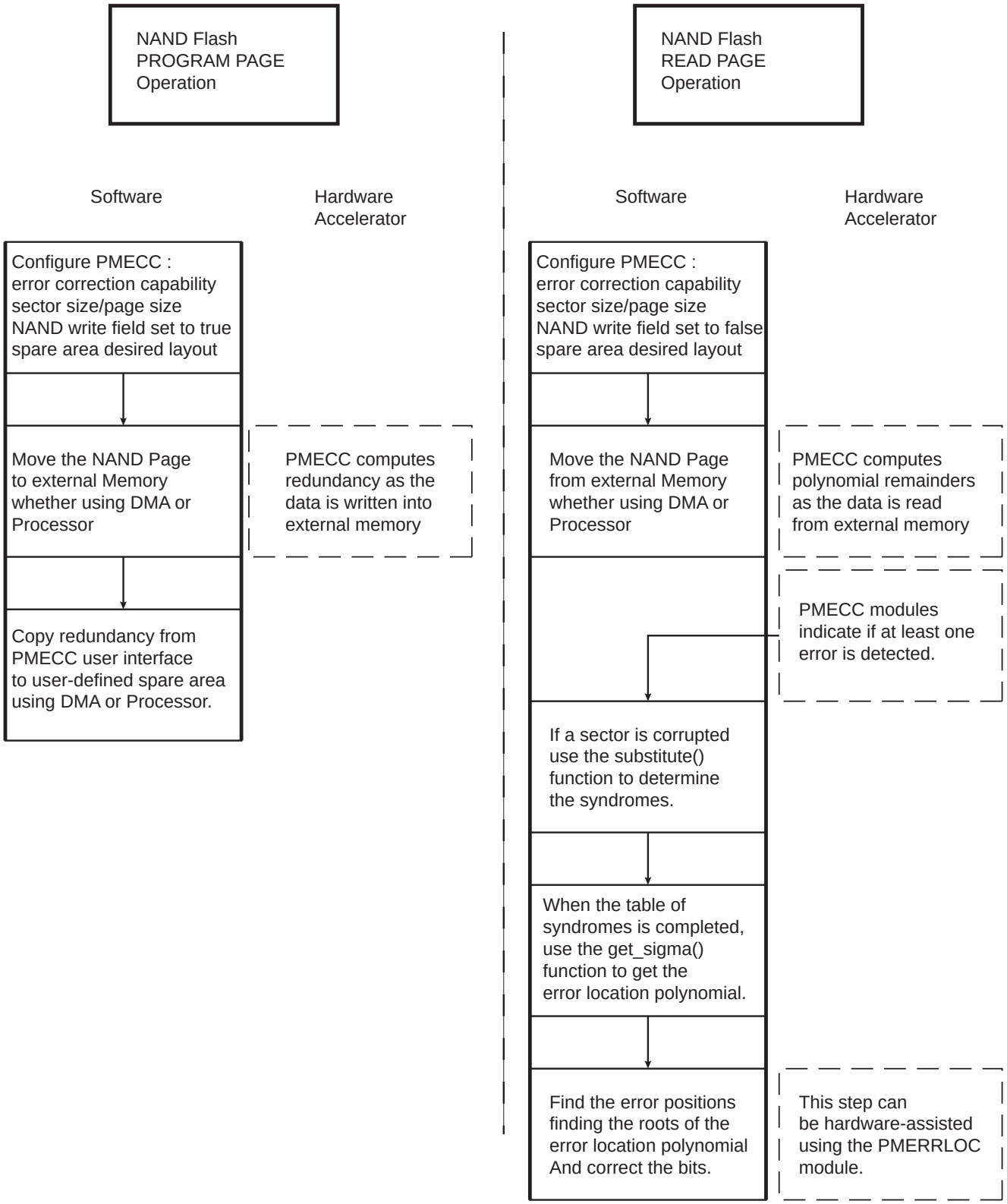
All decoding steps involve finite field computation. It means that a library of finite field arithmetic must be available to perform addition, multiplication and inversion. These arithmetic operations can be performed through the use of a memory mapped lookup table, or direct software implementation. The software implementation presented is based on lookup tables. Two tables named gf_log and gf_antilog are used. If alpha is the primitive element of the field, then a power of alpha is in the field. Assuming that beta = alpha ^ index, then beta belongs to the field, and gf_log(beta) = gf_log(alpha ^ index) = index. The gf_antilog table provides exponent inverse of the element; if beta = alpha ^ index, then gf_antilog(index) = beta.

The first step consists in the syndrome computation. The PMECC module computes the remainders and the software must substitute the power of the primitive element. The procedure implementation is given in Section 29.19.1 "Remainder Substitution Procedure".

The second step is the most software intensive. It is the Berlekamp's iterative algorithm for finding the error-location polynomial. The procedure implementation is given in Section 29.19.2 "Finding the Error Location Polynomial Sigma(x)".

The Last step is finding the root of the error location polynomial. This step can be very software intensive. Indeed there is no straightforward method of finding the roots, except evaluating each element of the field in the error location polynomial. However, a hardware accelerator can be used to find the roots of the polynomial. The PMERRLOC module provides this kind of hardware acceleration.

Figure 29-39. Software Hardware Multibit Error Correction Dataflow



29.18.1 MLC/SLC Write Page Operation Using PMECC

When an MLC write page operation is performed, the PMECC controller is configured with the NANDWR bit of the PMECCFG register set to one. When the NAND spare area contains file system information and redundancy (PMECCx), the spare area is error protected, then the SPAREEN bit of the PMECCFG register is set. When the NAND spare area contains only redundancy information, the SPAREEN bit is cleared.

When the write page operation is terminated, the user writes the redundancy in the NAND spare area. This operation can be done with DMA assistance.

Table 29-14. Relevant Redundancy Registers

BCH_ERR Field	Sector Size Set to 512 Bytes	Sector Size Set to 1024 Bytes
0	PMECC0	PMECC0
1	PMECC0, PMECC1	PMECC0, PMECC1
2	PMECC0, PMECC1, PMECC2, PMECC3	PMECC0, PMECC1, PMECC2, PMECC3
3	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6
4	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9, PMECC10

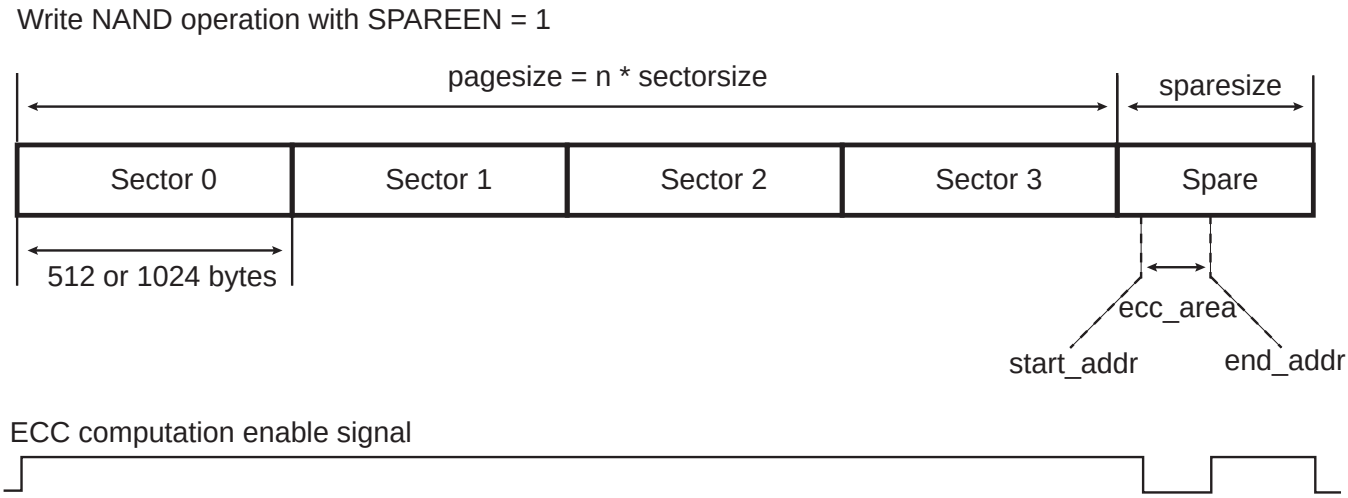
Table 29-15. Number of Relevant ECC Bytes per Sector, Copied from LSByte to MSByte

BCH_ERR Field	Sector Size Set to 512 Bytes	Sector Size Set to 1024 Bytes
0	4 bytes	4 bytes
1	7 bytes	7 bytes
2	13 bytes	14 bytes
3	20 bytes	21 bytes
4	39 bytes	42 bytes

29.18.1.1SLC/MLC Write Operation with Spare Enable Bit Set

When the SPAREEN bit of the PMECCFG register is set, the spare area of the page is encoded with the stream of data of the last sector of the page. This mode is entered by setting the DATA bit of the PMECCTRL register. When the encoding process is over, the redundancy shall be written to the spare area in User mode. The USER bit of the PMECCTRL register must be set.

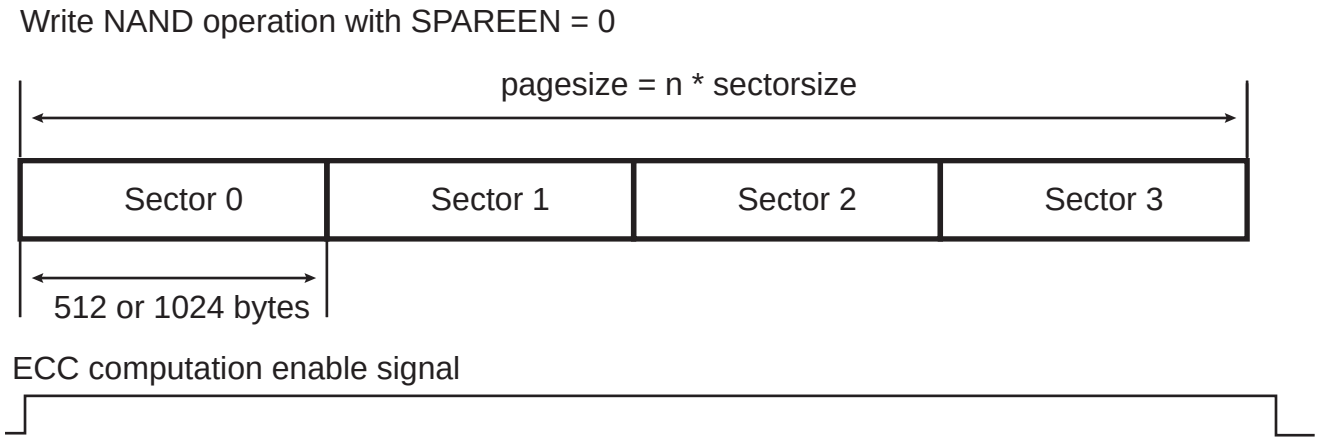
Figure 29-40. NAND Write Operation with Spare Encoding



29.18.1.2SLC/MLC Write Operation with Spare Disable

When the SPAREEN bit of PMECCFG is cleared, the spare area is not encoded with the stream of data. This mode is entered by setting the DATA bit of the PMECCTRL register.

Figure 29-41. NAND Write Operation



29.18.2 MLC/SLC Read Page Operation Using PMECC

Table 29-16. Relevant Remainder Registers

BCH_ERR Field	Sector Size Set to 512 Bytes	Sector Size Set to 1024 Bytes
0	PMECCREM0	PMECCREM0
1	PMECCREM0, PMECCREM1	PMECCREM0, PMECCREM1
2	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3,	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3
3	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7

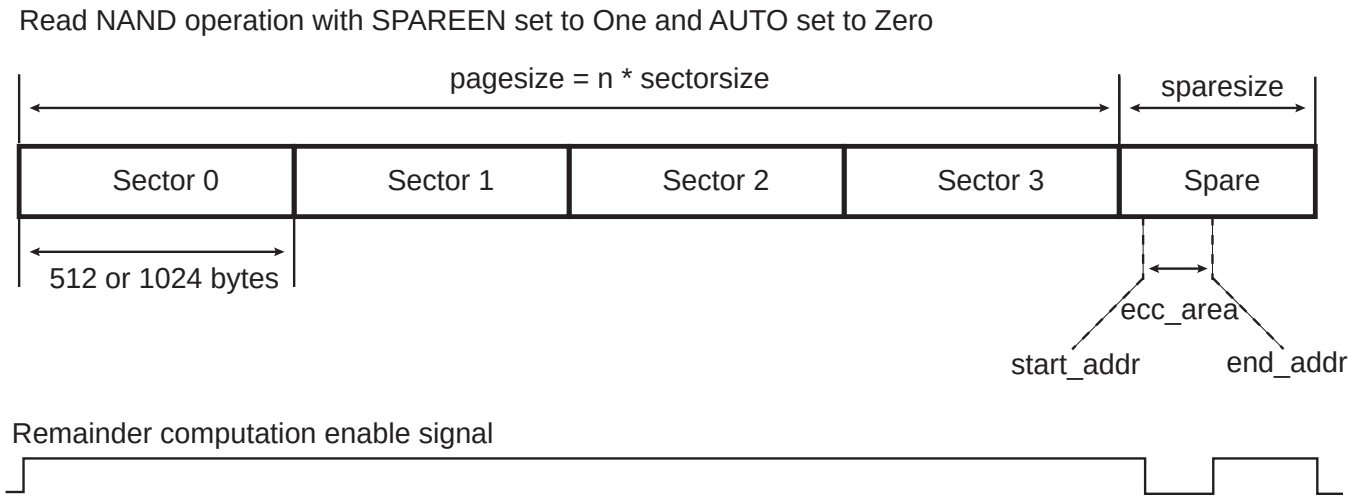
Table 29-16. Relevant Remainder Registers (Continued)

BCH_ERR Field	Sector Size Set to 512 Bytes	Sector Size Set to 1024 Bytes
4	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11

29.18.2.1MLC/SLC Read Operation with Spare Decoding

When the spare area is protected, it contains valid data. As the redundancy may be included in the middle of the information stream, the user shall program the start address and the end address of the ECC area. The controller will automatically skip the ECC area. This mode is entered writing a 1 in the DATA bit of the PMECCTRL register. When the page has been fully retrieved from the NAND, the ECC area shall be read using the User mode, writing a 1 to the USER bit of the PMECCTRL register.

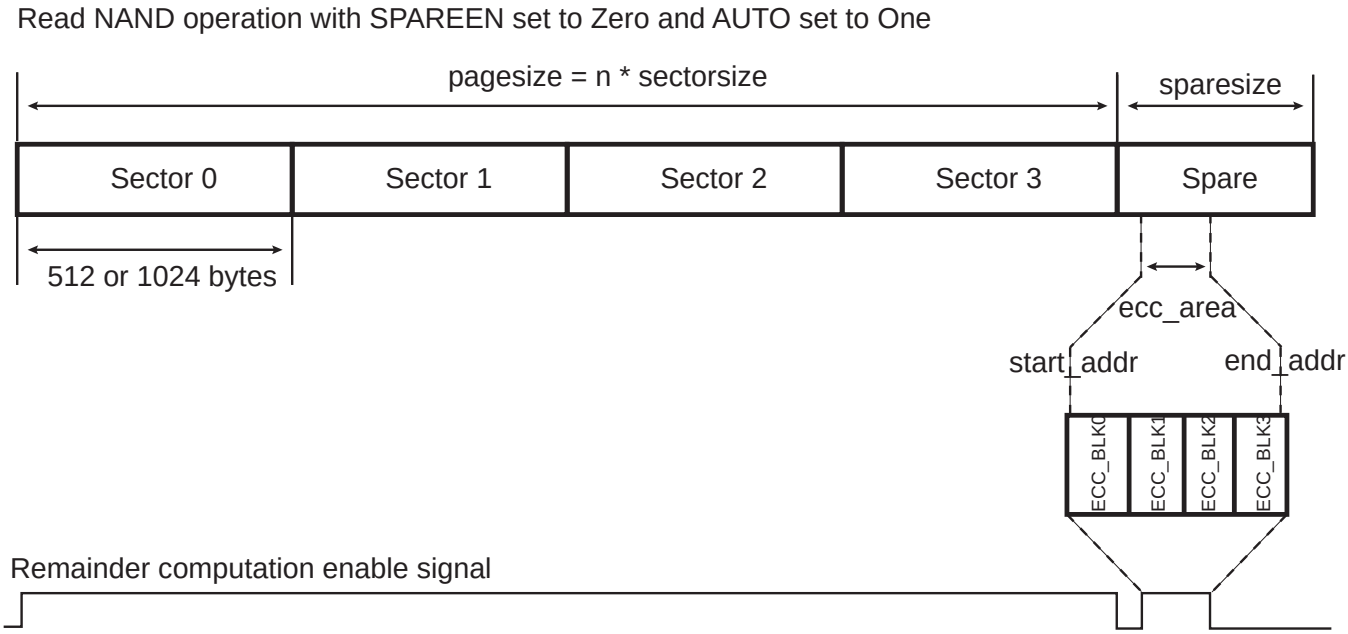
Figure 29-42. Read Operation with Spare Decoding



29.18.2.2MLC/SLC Read Operation

If the spare area is not protected with the error correcting code, the redundancy area is retrieved directly. This mode is entered writing a 1 in the DATA bit of the PMECCTRL register. When AUTO field is set to one, the ECC is retrieved automatically; otherwise, the ECC must be read using the User mode.

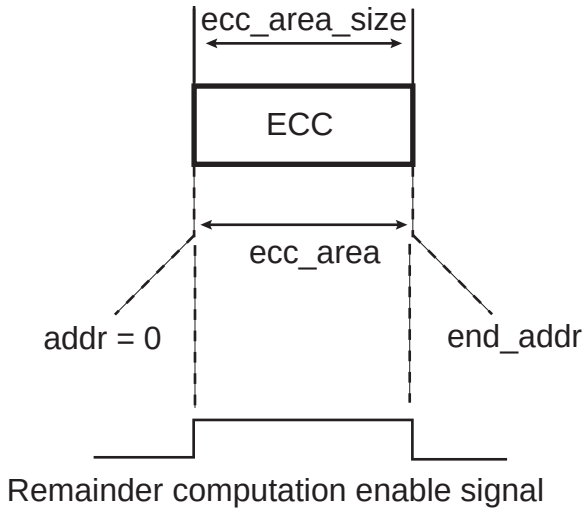
Figure 29-43. Read Operation



29.18.2.3MLC/SLC User Read ECC Area

This mode allows a manual retrieve of the ECC. It is entered writing a 1 in the USER field of the PMECCTRL register.

Figure 29-44. Read User Mode



29.18.2.4MLC Controller Working with NFC

Table 29-17. MLC Controller Configuration when the Host Controller is Used

Transfer Type	NFC		PMECC		
	RSPARE	WSPARE	SPAREEN	AUTO	User Mode
Program Page main area is protected, spare is not protected, spare is written manually	0	0	0	0	Not used
Program Page main area is protected, spare is protected, spare is written by NFC	0	1	1	0	Not used
Read Page main area is protected, spare is not protected, spare is not retrieved by NFC	0	0	0	0	Used
Read Page main area is protected, spare is not protected, spare is retrieved by NFC	1	0	0	1	Not used
Read Page main area is protected, spare is protected, spare is retrieved by NFC	1	0	1	0	Used

29.19 Software Implementation

29.19.1 Remainder Substitution Procedure

The substitute function evaluates the remainder polynomial, with different values of the field primitive element. The addition arithmetic operation is performed with the exclusive OR. The multiplication arithmetic operation is performed through the gf_log and gf_antilog lookup tables.

The REM2NP1 and REMN2NP3 fields of the PMECCREMN registers contain only odd remainders. Each bit indicates whether the coefficient of the remainder polynomial is set to zero or not.

NB_ERROR_MAX defines the maximum value of the error correcting capability.

NB_ERROR defines the error correcting capability selected at encoding/decoding time.

NB_FIELD_ELEMENTS defines the number of elements in the field.

si[] is a table that holds the current syndrome value. An element of that table belongs to the field. This is also a shared variable for the next step of the decoding operation.

oo[] is a table that contains the degree of the remainders.

```
int substitute()
{
int i;
int j;
for (i = 1; i < 2 * NB_ERROR_MAX; i++)
{
      si[i] = 0;
}
for (i = 1; i < 2*NB_ERROR; i++)
{
      for (j = 0; j < oo[i]; j++)
      {
            if (REM2NPX[i][j])
            {
```

```
                        si[i] = gf_antilog[(i * j)%NB_FIELD_ELEMENTS] ^ si[i];
                }
        }
}
return 0;
}
```

29.19.2 Finding the Error Location Polynomial Sigma(x)

The sample code below gives a Berlekamp iterative procedure for finding the value of the error location polynomial.

The input of the procedure is the si[] table defined in the remainder substitution procedure.

The output of the procedure is the error location polynomial named smu (sigma mu). The polynomial coefficients belong to the field. The smu[NB_ERROR+1][] is a table that contains all these coefficients.

NB_ERROR_MAX defines the maximum value of the error correcting capability.

NB_ERROR defines the error correcting capability selected at encoding/decoding time.

NB_FIELD_ELEMENTS defines the number of elements in the field.

```
int get_sigma()
{
int i;
int j;
int k;
/* mu          */
int mu[NB_ERROR_MAX+2];
/* sigma ro   */
int sro[2*NB_ERROR_MAX+1];
/* discrepancy */
int dmu[NB_ERROR_MAX+2];
/* delta order   */
int delta[NB_ERROR_MAX+2];
/* index of largest delta */
int ro;
int largest;
int diff;
/*                    */
/*     First Row      */
/*                    */
/* Mu */
mu[0]  = -1; /* Actually -1/2 */
/* Sigma(x) set to 1 */
for (i = 0; i < (2*NB_ERROR_MAX+1); i++)
      smu[0][i] = 0;
smu[0][0] = 1;
/* discrepancy set to 1 */
dmu[0] = 1;
/* polynom order set to 0 */
lmu[0] = 0;
/* delta set to -1 */
delta[0]  = (mu[0] * 2 - lmu[0]) >> 1;
/*                     */
/*     Second Row      */
/*                     */
/* Mu */
```

```
mu[1]  = 0;
/* Sigma(x) set to 1 */
for (i = 0; i < (2*NB_ERROR_MAX+1); i++)
      smu[1][i] = 0;
smu[1][0] = 1;
/* discrepancy set to Syndrome 1 */
dmu[1] = si[1];
/* polynom order set to 0 */
lmu[1] = 0;
/* delta set to 0 */
delta[1]  = (mu[1] * 2 - lmu[1]) >> 1;
for (i=1; i <= NB_ERROR; i++)
{
      mu[i+1] = i << 1;
      /*********************************************/
      /*                                           */
      /*                                           */
      /*          Compute Sigma (Mu+1)             */
      /*          And L(mu)                        */
      /* check if discrepancy is set to 0 */
      if (dmu[i] == 0)
      {
      /* copy polynom */
      for (j=0; j<2*NB_ERROR_MAX+1; j++)
      {
            smu[i+1][j] = smu[i][j];
      }
      /* copy previous polynom order to the next */
      lmu[i+1] = lmu[i];
      }
      else
      {
      ro      = 0;
      largest = -1;
      /* find largest delta with dmu != 0 */
      for (j=0; j<i; j++)
            {
            if (dmu[j])
                  {
                  if (delta[j] > largest)
                        {
                        largest = delta[j];
                        ro   = j;
                        }
                  }
            }
      /* initialize signal ro */
      for (k = 0; k < 2*NB_ERROR_MAX+1; k ++)
      {
            sro[k] = 0;
      }
      /* compute difference */
      diff = (mu[i] - mu[ro]);
      /* compute X ^ (2(mu-ro)) */
      for (k = 0; k < (2*NB_ERROR_MAX+1); k ++)
```

```
        {
                sro[k+diff] = smu[ro][k];
        }
        /* multiply by dmu * dmu[ro]^-1 */
        for (k = 0; k < 2*NB_ERROR_MAX+1; k ++)
        {
                /* dmu[ro] is not equal to zero by definition */
                /* check that operand are different from 0    */
                if (sro[k] && dmu[i])
                {
                        /* galois inverse */
                        sro[k] = gf_antilog[(gf_log[dmu[i]] + (NB_FIELD_ELEMENTS-
gf_log[dmu[ro]]) + gf_log[sro[k]]) % NB_FIELD_ELEMENTS];
                }
        }
        /* multiply by dmu * dmu[ro]^-1 */
        for (k = 0; k < 2*NB_ERROR_MAX+1; k++)
        {
                smu[i+1][k] = smu[i][k] ^ sro[k];
                if (smu[i+1][k])
                {
                        /* find the order of the polynom */
                        lmu[i+1] = k << 1;
                }
        }
}
/*                                                 */
/*                                                 */
/*      End Compute Sigma (Mu+1)                   */
/*      And L(mu)                                  */
/*************************************************/
/* In either case compute delta */
delta[i+1]  = (mu[i+1] * 2 - lmu[i+1]) >> 1;
/* In either case compute the discrepancy */
for (k = 0 ; k <= (lmu[i+1]>>1); k++)
{
        if (k == 0)
        dmu[i+1] = si[2*(i-1)+3];
        /* check if one operand of the multiplier is null, its index is -1 */
        else if (smu[i+1][k] && si[2*(i-1)+3-k])
        dmu[i+1] = gf_antilog[(gf_log[smu[i+1][k]] + gf_log[si[2*(i-1)+3-
k]])%nn] ^ dmu[i+1];
}
}
return 0;
}
```

29.19.3 Finding the Error Position

The output of the get_sigma() procedure is a polynomial stored in the smu[NB_ERROR+1][] table. The error positions are the roots of that polynomial. The degree of that polynomial is a very important information, as it gives the number of errors. PMERRLOC module provides hardware accelerator for that step.

29.19.3.1Error Location

The PMECC Error Location controller provides hardware acceleration for determining roots of polynomials over two finite fields: GF(2^13) and GF(2^14). It integrates 24 fully programmable coefficients. These coefficients belong to GF(2^13) or GF(2^14). The coefficient programmed in the PMERRLOC{*i*} is the coefficient of X ^ *i* in the polynomial.

The search operation is started as soon as a write access is detected in the ELEN register and can be disabled writing to the ELDIS register. The ENINIT field of the ELEN register shall be initialized with the number of galois field elements to test. The set of the roots can be limited to a valid range.

Table 29-18. ENINIT Field Value for a Sector Size of 512 Bytes

Error Correcting Capability	ENINIT Value
2	4122
4	4148
8	4200
12	4252
24	4408

Table 29-19. ENINIT Field Value for a Sector Size of 1024 Bytes

Error Correcting Capability	ENINIT Value
2	8220
4	8248
8	8304
12	8360
24	8528

When the PMECC engine is searching for roots, the BUSY field of the ELSR register remains asserted. An interrupt is asserted at the end of the computation, and the DONE bit of the PMECC Error Location Interrupt Status Register (HSMC_ELSIR) is set. The ERR_CNT field of the HSMC_ELISR indicates the number of errors. The error position can be read in the PMERRLOCX registers.

29.20 Static Memory Controller (SMC) User Interface

The SMC is programmed using the registers listed in Table 29-20. For each chip select, a set of four registers is used to program the parameters of the external device. In Table 29-20, “CS_number” denotes the chip select number. Sixteen bytes per chip select are required.

Table 29-20. Register Mapping

Offset	Register	Name	Access	Reset
0x000	NFC Configuration Register	HSMC_CFG	Read/Write	0x0
0x004	NFC Control Register	HSMC_CTRL	Write-only	–
0x008	NFC Status Register	HSMC_SR	Read-only	0x0
0x00C	NFC Interrupt Enable Register	HSMC_IER	Write-only	–
0x010	NFC Interrupt Disable Register	HSMC_IDR	Write-only	–
0x014	NFC Interrupt Mask Register	HSMC_IMR	Read-only	0x0
0x018	NFC Address Cycle Zero Register	HSMC_ADDR	Read/Write	0x0
0x01C	Bank Address Register	HSMC_BANK	Read/Write	0x0
0x020–0x06C	Reserved	–	–	–
0x070	PMECC Configuration Register	HSMC_PMECCFG	Read/Write	0x0
0x074	PMECC Spare Area Size Register	HSMC_PMECCSAREA	Read/Write	0x0
0x078	PMECC Start Address Register	HSMC_PMECCSADDR	Read/Write	0x0
0x07C	PMECC End Address Register	HSMC_PMECCEADDR	Read/Write	0x0
0x080	Reserved	–	–	–
0x084	PMECC Control Register	HSMC_PMECCTRL	Write-only	–
0x088	PMECC Status Register	HSMC_PMECCSR	Read-only	0x0
0x08C	PMECC Interrupt Enable register	HSMC_PMECCIER	Write-only	–
0x090	PMECC Interrupt Disable Register	HSMC_PMECCIDR	Write-only	–
0x094	PMECC Interrupt Mask Register	HSMC_PMECCIMR	Read-only	0x0
0x098	PMECC Interrupt Status Register	HSMC_PMECCISR	Read-only	0x0
0x09C–0x0AC	Reserved	–	–	–
0x0B0+sec_num*(0x40)+0x00	PMECC Redundancy 0 Register	HSMC_PMECC0	Read-only	0x0
0x0B0+sec_num*(0x40)+0x04	PMECC Redundancy 1 Register	HSMC_PMECC1	Read-only	0x0
0x0B0+sec_num*(0x40)+0x08	PMECC Redundancy 2 Register	HSMC_PMECC2	Read-only	0x0
0x0B0+sec_num*(0x40)+0x0C	PMECC Redundancy 3 Register	HSMC_PMECC3	Read-only	0x0
0x0B0+sec_num*(0x40)+0x10	PMECC Redundancy 4 Register	HSMC_PMECC4	Read-only	0x0
0x0B0+sec_num*(0x40)+0x14	PMECC Redundancy 5 Register	HSMC_PMECC5	Read-only	0x0
0x0B0+sec_num*(0x40)+0x18	PMECC Redundancy 6 Register	HSMC_PMECC6	Read-only	0x0
0x0B0+sec_num*(0x40)+0x1C	PMECC Redundancy 7 Register	HSMC_PMECC7	Read-only	0x0
0x0B0+sec_num*(0x40)+0x20	PMECC Redundancy 8 Register	HSMC_PMECC8	Read-only	0x0
0x0B0+sec_num*(0x40)+0x24	PMECC Redundancy 9 Register	HSMC_PMECC9	Read-only	0x0
0x0B0+sec_num*(0x40)+0x28	PMECC Redundancy 10 Register	HSMC_PMECC10	Read-only	0x0
0x2B0+sec_num*(0x40)+0x00	PMECC Remainder 0 Register	HSMC_REM0	Read-only	0x0

Table 29-20. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x2B0+sec_num*(0x40)+0x04	PMECC Remainder 1 Register	HSMC_REM1	Read-only	0x0
0x2B0+sec_num*(0x40)+0x08	PMECC Remainder 2 Register	HSMC_REM2	Read-only	0x0
0x2B0+sec_num*(0x40)+0x0C	PMECC Remainder 3 Register	HSMC_REM3	Read-only	0x0
0x2B0+sec_num*(0x40)+0x10	PMECC Remainder 4 Register	HSMC_REM4	Read-only	0x0
0x2B0+sec_num*(0x40)+0x14	PMECC Remainder 5 Register	HSMC_REM5	Read-only	0x0
0x2B0+sec_num*(0x40)+0x18	PMECC Remainder 6 Register	HSMC_REM6	Read-only	0x0
0x2B0+sec_num*(0x40)+0x1C	PMECC Remainder 7 Register	HSMC_REM7	Read-only	0x0
0x2B0+sec_num*(0x40)+0x20	PMECC Remainder 8 Register	HSMC_REM8	Read-only	0x0
0x2B0+sec_num*(0x40)+0x24	PMECC Remainder 9 Register	HSMC_REM9	Read-only	0x0
0x2B0+sec_num*(0x40)+0x28	PMECC Remainder 10 Register	HSMC_REM10	Read-only	0x0
0x2B0+sec_num*(0x40)+0x2C	PMECC Remainder 11 Register	HSMC_REM11	Read-only	0x0
0x4A0–0x4FC	Reserved	–	–	–
0x500	PMECC Error Location Configuration Register	HSMC_ELCFG	Read/Write	0x0
0x504	PMECC Error Location Primitive Register	HSMC_ELPRIM	Read-only	0x401A
0x508	PMECC Error Location Enable Register	HSMC_ELEN	Write-only	–
0x50C	PMECC Error Location Disable Register	HSMC_ELDIS	Write-only	–
0x510	PMECC Error Location Status Register	HSMC_ELSR	Read-only	0x0
0x514	PMECC Error Location Interrupt Enable register	HSMC_ELIER	Write-only	–
0x518	PMECC Error Location Interrupt Disable Register	HSMC_ELIDR	Write-only	–
0x51C	PMECC Error Location Interrupt Mask Register	HSMC_ELIMR	Read-only	0x0
0x520	PMECC Error Location Interrupt Status Register	HSMC_ELISR	Read-only	0x0
0x524	Reserved	–	–	–
0x528	PMECC Error Location SIGMA 0 Register	HSMC_SIGMA0	Read-only	0x1
0x52C	PMECC Error Location SIGMA 1 Register	HSMC_SIGMA1	Read/Write	0x0
0x530	PMECC Error Location SIGMA 2 Register	HSMC_SIGMA2	Read/Write	0x0
0x534	PMECC Error Location SIGMA 3 Register	HSMC_SIGMA3	Read/Write	0x0
0x538	PMECC Error Location SIGMA 4 Register	HSMC_SIGMA4	Read/Write	0x0
0x53C	PMECC Error Location SIGMA 5 Register	HSMC_SIGMA5	Read/Write	0x0

Table 29-20. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x540	PMECC Error Location SIGMA 6 Register	HSMC_SIGMA6	Read/Write	0x0
0x544	PMECC Error Location SIGMA 7 Register	HSMC_SIGMA7	Read/Write	0x0
0x548	PMECC Error Location SIGMA 8 Register	HSMC_SIGMA8	Read/Write	0x0
0x54C	PMECC Error Location SIGMA 9 Register	HSMC_SIGMA9	Read/Write	0x0
0x550	PMECC Error Location SIGMA 10 Register	HSMC_SIGMA10	Read/Write	0x0
0x554	PMECC Error Location SIGMA 11 Register	HSMC_SIGMA11	Read/Write	0x0
0x558	PMECC Error Location SIGMA 12 Register	HSMC_SIGMA12	Read/Write	0x0
0x55C	PMECC Error Location SIGMA 13 Register	HSMC_SIGMA13	Read/Write	0x0
0x560	PMECC Error Location SIGMA 14 Register	HSMC_SIGMA14	Read/Write	0x0
0x564	PMECC Error Location SIGMA 15 Register	HSMC_SIGMA15	Read/Write	0x0
0x568	PMECC Error Location SIGMA 16 Register	HSMC_SIGMA16	Read/Write	0x0
0x56C	PMECC Error Location SIGMA 17 Register	HSMC_SIGMA17	Read/Write	0x0
0x570	PMECC Error Location SIGMA 18 Register	HSMC_SIGMA18	Read/Write	0x0
0x574	PMECC Error Location SIGMA 19 Register	HSMC_SIGMA19	Read/Write	0x0
0x578	PMECC Error Location SIGMA 20 Register	HSMC_SIGMA20	Read/Write	0x0
0x57C	PMECC Error Location SIGMA 21 Register	HSMC_SIGMA21	Read/Write	0x0
0x580	PMECC Error Location SIGMA 22 Register	HSMC_SIGMA22	Read/Write	0x0
0x584	PMECC Error Location SIGMA 23 Register	HSMC_SIGMA23	Read/Write	0x0
0x588	PMECC Error Location SIGMA 24 Register	HSMC_SIGMA24	Read/Write	0x0
0x58C	PMECC Error Location 0 Register	HSMC_ERRLOC0	Read-only	0x0
...	...	...	...	...
0x5E8	PMECC Error Location 23 Register	HSMC_ERRLOC23	Read-only	0x0
0x5EC–0x5FC	Reserved	–	–	–
0x14*CS_number+0x600	Setup Register	HSMC_SETUP	Read/Write	0x0101_0101
0x14*CS_number+0x604	Pulse Register	HSMC_PULSE	Read/Write	0x0101_0101

Table 29-20. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x14*CS_number+0x608	Cycle Register	HSMC_CYCLE	Read/Write	0x0003_0003
0x14*CS_number+0x60C	Timings Register	HSMC_TIMINGS	Read/Write	0x0000_0000
0x14*CS_number+0x610	Mode Register	HSMC_MODE	Read/Write	0x0000_1003
0x6A0	Off Chip Memory Scrambling Register	HSMC_OCMS	Read/Write	0x0
0x6A4	Off Chip Memory Scrambling KEY1 Register	HSMC_KEY1	Write-once	0x0
0x6A8	Off Chip Memory Scrambling KEY2 Register	HSMC_KEY2	Write-once	0x0
0x6AC–0x6E0	Reserved	–	–	–
0x6E4	Write Protection Mode Register	HSMC_WPMR	Read/Write	0x0
0x6E8	Write Protection Status Register	HSMC_WPSR	Read-only	0x0
0x6EC–0x6FC	Reserved	–	–	–

29.20.1 NFC Configuration Register

Name: HSMC_CFG

Address: 0xFC05C000

Access: Read/Write

31	30	29	28	27	26	25	24
–	NFCSPARESIZE						

23	22	21	20	19	18	17	16
–	DTOMUL			DTOCYC			

15	14	13	12	11	10	9	8
–	–	RBEDGE	EDGECTRL	–	–	RSPARE	WSPARE

7	6	5	4	3	2	1	0
–	–	–	–	–	PAGESIZE		

- **PAGESIZE: Page Size of the NAND Flash Device**

Value	Name	Description
0	PS512	Main area 512 bytes
1	PS1024	Main area 1024 bytes
2	PS2048	Main area 2048 bytes
3	PS4096	Main area 4096 bytes
4	PS8192	Main area 8192 bytes

- **WSPARE: Write Spare Area**

0: The NFC skips the spare area in Write mode.

1: The NFC writes both main area and spare area in Write mode.

- **RSPARE: Read Spare Area**

0: The NFC skips the spare area in Read mode.

1: The NFC reads both main area and spare area in Read mode.

- **EDGECTRL: Rising/Falling Edge Detection Control**

0: Rising edge is detected

1: Falling edge is detected

- **RBEDGE: Ready/Busy Signal Edge Detection**

0: When configured to zero, RB_EDGE fields indicate the level of the Ready/Busy lines.

1: When set to one, RB_EDGE fields indicate only transition on Ready/Busy lines.

- **DTOCYC: Data Timeout Cycle Number**

- **DTOMUL: Data Timeout Multiplier**

These fields determine the maximum number of Master Clock cycles that the SMC waits until the detection of a rising edge on Ready/Busy signal.

Data Timeout Multiplier is defined by DTOMUL as shown in the following table:

Value	Name	Description
0	X1	DTOCYC
1	X16	DTOCYC x 16
2	X128	DTOCYC x 128
3	X256	DTOCYC x 256
4	X1024	DTOCYC x 1024
5	X4096	DTOCYC x 4096
6	X65536	DTOCYC x 65536
7	X1048576	DTOCYC x 1048576

If the data timeout set by DTOCYC and DTOMUL has been exceeded, the Data Timeout Error flag (DTOE) in the NFC Status Register (NFC_SR) raises.

- **NFCSPARESIZE: NAND Flash Spare Area Size Retrieved by the Host Controller**

The spare size is set to (NFCSPARESIZE + 1) * 4 bytes. The spare area is only retrieved when RSPARE or WSPARE is activated.

29.20.2 NFC Control Register

Name: HSMC_CTRL

Address: 0xFC05C004

Access: Write-only

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
–	–	–	–	–	–	NFCDIS	NFCEN

- **NFCEN: NAND Flash Controller Enable**

0: No effect

1: Enable the NAND Flash controller.

- **NFCDIS: NAND Flash Controller Disable**

0: No effect

1: Disable the NAND Flash controller.

29.20.3 NFC Status Register

Name: HSMC_SR

Address: 0xFC05C008

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	RB_EDGE0
23	22	21	20	19	18	17	16
NFCASE	AWB	UNDEF	DTOE	–	–	CMDDONE	XFRDONE
15	14	13	12	11	10	9	8
–	NFCSID			NFCWR	–	–	NFCBUSY
7	6	5	4	3	2	1	0
–	–	RB_FALL	RB_RISE	–	–	–	SMCSTS

- **SMCSTS: NAND Flash Controller Status (this field cannot be reset)**

0: NAND Flash Controller disabled

1: NAND Flash Controller enabled

- **RB_RISE: Selected Ready Busy Rising Edge Detected**

When set to one, this flag indicates that a rising edge on the Ready/Busy Line has been detected. This flag is reset after a status read operation. The Ready/Busy line is selected through the decoding of field HSMC_SR.NFCSID.

- **RB_FALL: Selected Ready Busy Falling Edge Detected**

When set to one, this flag indicates that a falling edge on the Ready/Busy Line has been detected. This flag is reset after a status read operation. The Ready/Busy line is selected through the decoding of field HSMC_SR.NFCSID.

- **NFCBUSY: NFC Busy (this field cannot be reset)**

When set to one, this flag indicates that the Controller is activated and accesses the memory device.

- **NFCWR: NFC Write/Read Operation (this field cannot be reset)**

When a command is issued, this field indicates the current Read or Write Operation.

- **NFCSID: NFC Chip Select ID (this field cannot be reset)**

When a command is issued, this field indicates the value of the targeted chip select.

- **XFRDONE: NFC Data Transfer Terminated**

When set to one, this flag indicates that the NFC has terminated the Data Transfer. This flag is reset after a status read operation.

- **CMDDONE: Command Done**

When set to one, this flag indicates that the NFC has terminated the Command. This flag is reset after a status read operation.

- **DTOE: Data Timeout Error**

When set to one, this flag indicates that the Data timeout set be by DTOMUL and DTOCYC has been exceeded. This flag is reset after a status read operation.

- **UNDEF: Undefined Area Error**

When set to one, this flag indicates that the processor performed an access in an undefined memory area. This flag is reset after a status read operation.

- **AWB: Accessing While Busy**

If set to one, this flag indicates that an AHB master has performed an access during the busy phase. This flag is reset after a status read operation.

- **NFCASE: NFC Access Size Error**

If set to one, this flag indicates that an illegal access has been detected in the NFC Memory Area. Only Word Access is allowed within the NFC memory area. This flag is reset after a status read operation.

- **RB_EDGEx: Ready/Busy Line x Edge Detected**

If set to one, this flag indicates that an edge has been detected on the Ready/Busy Line x. Depending on the EDGE CTRL field located in the HSMC_CFG register, only rising or falling edge is detected. This flag is reset after a status read operation.

29.20.4 NFC Interrupt Enable Register

Name: HSMC_IER

Address: 0xFC05C00C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	RB_EDGE0
23	22	21	20	19	18	17	16
NFCASE	AWB	UNDEF	DTOE	–	–	CMDDONE	XFRDONE
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RB_FALL	RB_RISE	–	–	–	–

- **RB_RISE: Ready Busy Rising Edge Detection Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **RB_FALL: Ready Busy Falling Edge Detection Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **XFRDONE: Transfer Done Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **CMDDONE: Command Done Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **DTOE: Data Timeout Error Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **UNDEF: Undefined Area Access Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **AWB: Accessing While Busy Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **NFCASE: NFC Access Size Error Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **RB_EDGEx: Ready/Busy Line x Interrupt Enable**

0: No effect

1: Interrupt source enabled

29.20.5 NFC Interrupt Disable Register

Name: HSMC_IDR

Address: 0xFC05C010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	RB_EDGE0
23	22	21	20	19	18	17	16
NFCASE	AWB	UNDEF	DTOE	–	–	CMDDONE	XFRDONE
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RB_FALL	RB_RISE	–	–	–	–

- **RB_RISE: Ready Busy Rising Edge Detection Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **RB_FALL: Ready Busy Falling Edge Detection Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **XFRDONE: Transfer Done Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **CMDDONE: Command Done Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **DTOE: Data Timeout Error Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **UNDEF: Undefined Area Access Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **AWB: Accessing While Busy Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **NFCASE: NFC Access Size Error Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **RB_EDGEx: Ready/Busy Line x Interrupt Disable**

0: No effect

1: Interrupt source disabled

29.20.6 NFC Interrupt Mask Register

Name: HSMC_IMR

Address: 0xFC05C014

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	RB_EDGE0
23	22	21	20	19	18	17	16
NFCASE	AWB	UNDEF	DTOE	–	–	CMDDONE	XFRDONE
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RB_FALL	RB_RISE	–	–	–	–

- **RB_RISE: Ready Busy Rising Edge Detection Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **RB_FALL: Ready Busy Falling Edge Detection Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **XFRDONE: Transfer Done Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **CMDDONE: Command Done Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **DTOE: Data Timeout Error Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **UNDEF: Undefined Area Access Interrupt Mask5**

0: Interrupt source disabled

1: Interrupt source enabled

- **AWB: Accessing While Busy Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **NFCASE: NFC Access Size Error Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **RB_EDGEx: Ready/Busy Line x Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

29.20.7 NFC Address Cycle Zero Register

Name: HSMC_ADDR

Address: 0xFC05C018

Access: Read/Write

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
ADDR_CYCLE0							

- **ADDR_CYCLE0: NAND Flash Array Address Cycle 0**

When five address cycles are used, ADDR_CYCLE0 is the first byte written to the NAND Flash (used by the NFC).

29.20.8 NFC Bank Register

Name: HSMC_BANK

Address: 0xFC05C01C

Access: Read/Write

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
–	–	–	–	–	–	–	BANK

- **BANK: Bank Identifier**

0: Bank 0 is used.

1: Bank 1 is used.

29.20.9 PMECC Configuration Register

Name: HSMC_PMECCFG

Address: 0xFC05C070

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	AUTO	–	–	–	SPAREEN
15	14	13	12	11	10	9	8
–	–	–	NANDWR	–	–	PAGESIZE	
7	6	5	4	3	2	1	0
–	–	–	SECTORSZ	–	BCH_ERR		

- **BCH_ERR: Error Correcting Capability**

Value	Name	Description
0	BCH_ERR2	2 errors
1	BCH_ERR4	4 errors
2	BCH_ERR8	8 errors
3	BCH_ERR12	12 errors
4	BCH_ERR24	24 errors

- **SECTORSZ: Sector Size**

0: The ECC computation is based on a sector of 512 bytes.

1: The ECC computation is based on a sector of 1024 bytes.

- **PAGESIZE: Number of Sectors in the Page**

Value	Name	Description
0	PAGESIZE_1SEC	1 sector for main area (512 or 1024 bytes)
1	PAGESIZE_2SEC	2 sectors for main area (1024 or 2048 bytes)
2	PAGESIZE_4SEC	4 sectors for main area (2048 or 4096 bytes)
3	PAGESIZE_8SEC	8 sectors for main area (4096 or 8192 bytes)

- **NANDWR: NAND Write Access**

0: NAND read access

1: NAND write access

- **SPAREEN: Spare Enable**
  - for NAND write access:

0: The spare area is skipped

1: The spare area is protected with the last sector of data.

  - for NAND read access:

0: The spare area is skipped.

1: The spare area contains protected data or only redundancy information.

- **AUTO: Automatic Mode Enable**

This bit is only relevant in NAND Read Mode, when spare enable is activated.

0: Indicates that the spare area is not protected. In that case, the ECC computation takes into account the ECC area located in the spare area. (within the start address and the end address).

1: Indicates that the spare area is error protected. In this case, the ECC computation takes into account the whole spare area minus the ECC area in the ECC computation operation.

29.20.10PMECC Spare Area Size Register

Name: HSMC_PMECCSAREA

Address: 0xFC05C074

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	SPARESIZE

7	6	5	4	3	2	1	0
SPARESIZE							

- **SPARESIZE: Spare Area Size**

Number of bytes in the spare area. The spare area size is equal to (SPARESIZE + 1) bytes.

29.20.11PMECC Start Address Register

Name: HSMC_PMECCSADDR

Address: 0xFC05C078

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	STARTADDR

7	6	5	4	3	2	1	0
STARTADDR							

- **STARTADDR: ECC Area Start Address**

This register is programmed with the start ECC start address. When STARTADDR is equal to 0, then the first ECC byte is located at the first byte of the spare area.

29.20.12PMECC End Address Register

Name: HSMC_PMECCEADDR

Address: 0xFC05C07C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	ENDADDR

7	6	5	4	3	2	1	0
ENDADDR							

- **ENDADDR: ECC Area End Address**

This register is programmed with the start ECC end address. When ENDADDR is equal to *N*, then the first ECC byte is located at byte *N* of the spare area.

29.20.13PMECC Control Register

Name: HSMC_PMECCTRL

Address: 0xFC05C084

Access: Write-only

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
–	–	DISABLE	ENABLE	–	USER	DATA	RST

- **RST: Reset the PMECC Module**

0: No effect

1: Reset the PMECC controller.

- **DATA: Start a Data Phase**

0: No effect

1: The PMECC controller enters a Data phase.

- **USER: Start a User Mode Phase**

0: No effect

1: The PMECC controller enters a User mode phase.

- **ENABLE: PMECC Enable**

0: No effect

1: Enable the PMECC controller.

- **DISABLE: PMECC Enable**

0: No effect

1: Disable the PMECC controller.

29.20.14PMECC Status Register

Name: HSMC_PMECCSR

Address: 0xFC05C088

Access: Read-only

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
–	–	–	ENABLE	–	–	–	BUSY

- **BUSY: The kernel of the PMECC is busy**

0: PMECC controller finite state machine reached idle state

1: PMECC controller finite state machine is processing the incoming byte stream

- **ENABLE: PMECC Enable bit**

0: PMECC controller disabled

1: PMECC controller enabled

29.20.15PMECC Interrupt Enable Register

Name: HSMC_PMECCIER

Address: 0xFC05C08C

Access: Write-only

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
–	–	–	–	–	–	–	ERRIE

- **ERRIE: Error Interrupt Enable**

0: No effect

1: The Multibit Error interrupt is enabled. An interrupt will be raised if at least one error is detected in at least one sector.

29.20.16PMECC Interrupt Disable Register

Name: HSMC_PMECCIDR

Address: 0xFC05C090

Access: Write-only

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
–	–	–	–	–	–	–	ERRID

- **ERRID: Error Interrupt Disable**

0: No effect

1: Multibit Error interrupt disabled

29.20.17PMECC Interrupt Mask Register

Name: HSMC_PMECCIMR

Address: 0xFC05C094

Access: Read-only

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
–	–	–	–	–	–	–	ERRIM

- **ERRIM: Error Interrupt Mask**

0: Multibit Error disabled

1: Multibit Error enabled

29.20.18PMECC Interrupt Status Register

Name: HSMC_PMECCISR

Address: 0xFC05C098

Access: Read-only

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
ERRIS							

- **ERRIS: Error Interrupt Status Register**

When set to one, bit *i* of the HSMC_PMECCISR indicates that sector *i* is corrupted.

29.20.19PMECC Redundancy x Register

Name: HSMC_PMECCx [x=0..10] [sec_num=0..7]

Address: 0xFC05C0B0 [0][0] .. 0xFC05C0D8 [10][0]
0xFC05C0F0 [0][1] .. 0xFC05C118 [10][1]
0xFC05C130 [0][2] .. 0xFC05C158 [10][2]
0xFC05C170 [0][3] .. 0xFC05C198 [10][3]
0xFC05C1B0 [0][4] .. 0xFC05C1D8 [10][4]
0xFC05C1F0 [0][5] .. 0xFC05C218 [10][5]
0xFC05C230 [0][6] .. 0xFC05C258 [10][6]
0xFC05C270 [0][7] .. 0xFC05C298 [10][7]

Access: Read-only

31	30	29	28	27	26	25	24
ECC							

23	22	21	20	19	18	17	16
ECC							

15	14	13	12	11	10	9	8
ECC							

7	6	5	4	3	2	1	0
ECC							

- **ECC: BCH Redundancy**

This register contains the remainder of the division of the codeword by the generator polynomial.

29.20.20PMECC Remainder x Register

Name: HSMC_REMx [x=0..11] [sec_num=0..7]

Address: 0xFC05C2B0 [0][0] .. 0xFC05C2DC [11][0]
0xFC05C2F0 [0][1] .. 0xFC05C31C [11][1]
0xFC05C330 [0][2] .. 0xFC05C35C [11][2]
0xFC05C370 [0][3] .. 0xFC05C39C [11][3]
0xFC05C3B0 [0][4] .. 0xFC05C3DC [11][4]
0xFC05C3F0 [0][5] .. 0xFC05C41C [11][5]
0xFC05C430 [0][6] .. 0xFC05C45C [11][6]
0xFC05C470 [0][7] .. 0xFC05C49C [11][7]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	REM2NP3					

23	22	21	20	19	18	17	16
REM2NP3							

15	14	13	12	11	10	9	8
–	–	REM2NP1					

7	6	5	4	3	2	1	0
REM2NP1							

- **REM2NP1: BCH Remainder 2 * N + 1**

When sector size is set to 512 bytes, bit REM2NP1[13] is not used and read as zero.

If bit *i* of the REM2NP1 field is set to one, then the coefficient of the X ^ *i* is set to one; otherwise, the coefficient is zero.

- **REM2NP3: BCH Remainder 2 * N + 3**

When sector size is set to 512 bytes, bit REM2NP3[29] is not used and read as zero.

If bit *i* of the REM2NP3 field is set to one, then the coefficient of the X ^ *i* is set to one; otherwise, the coefficient is zero.

29.20.21PMECC Error Location Configuration Register

Name: HSMC_ELCFG

Address: 0xFC05C500

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	ERRNUM				

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SECTORSZ

- **ERRNUM: Number of Errors**

- **SECTORSZ: Sector Size**

0: The ECC computation is based on a 512 bytes sector.

1: The ECC computation is based on a 1024 bytes sector.

29.20.22PMECC Error Location Primitive Register

Name: HSMC_ELPRIM

Address: 0xFC05C504

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PRIMITIV							

7	6	5	4	3	2	1	0
PRIMITIV							

- **PRIMITIV: Primitive Polynomial**

This field indicates the Primitive Polynomial used in the ECC computation.

29.20.23PMECC Error Location Enable Register

Name: HSMC_ELEN

Address: 0xFC05C508

Access: Write-only

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

- **ENINIT: Error Location Enable**

Initial bit number in the codeword.

29.20.24PMECC Error Location Disable Register

Name: HSMC_ELDIS

Address: 0xFC05C50C

Access: Write-only

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
–	–	–	–	–	–	–	DIS

- **DIS: Disable Error Location Engine**

0: No effect

1: Disable the Error location engine.

29.20.25PMECC Error Location Status Register

Name: HSMC_ELSR

Address: 0xFC05C510

Access: Read-only

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
–	–	–	–	–	–	–	BUSY

- **BUSY: Error Location Engine Busy**

0: Error location engine is disabled.

1: Error location engine is enabled and is finding roots of the polynomial.

29.20.26PMECC Error Location Interrupt Enable Register

Name: HSMC_ELIER

Address: 0xFC05C514

Access: Write-only

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
–	–	–	–	–	–	–	DONE

- **DONE: Computation Terminated Interrupt Enable**

0: No effect

1: Interrupt enable

29.20.27PMECC Error Location Interrupt Disable Register

Name: HSMC_ELIDR

Address: 0xFC05C518

Access: Write-only

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
–	–	–	–	–	–	–	DONE

- **DONE: Computation Terminated Interrupt Disable**

0: No effect

1: Interrupt disable.

29.20.28PMECC Error Location Interrupt Mask Register

Name: HSMC_ELIMR

Address: 0xFC05C51C

Access: Read-only

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
–	–	–	–	–	–	–	DONE

- **DONE: Computation Terminated Interrupt Mask**

0: Computation Terminated interrupt disabled

1: Computation Terminated interrupt enabled

29.20.29PMECC Error Location Interrupt Status Register

Name: HSMC_ELISR

Address: 0xFC05C520

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	ERR_CNT				

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DONE

- **DONE: Computation Terminated Interrupt Status**

When set to one, this indicates that the error location engine has completed the root finding algorithm.

- **ERR_CNT: Error Counter value**

This field indicates the number of roots of the polynomial.

29.20.30PMECC Error Location SIGMA0 Register

Name: HSMC_SIGMA0

Address: 0xFC05C52C [1] .. 0xFC05C588 [24], 0xFC05C528 [0] .. 0xFC05C588 [24]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	SIGMA0					

7	6	5	4	3	2	1	0
SIGMA0							

- **SIGMA0: Coefficient of degree 0 in the SIGMA polynomial**

SIGMA0 belongs to the finite field GF(2^13) when the sector size is set to 512 bytes.

SIGMA0 belongs to the finite field GF(2^14) when the sector size is set to 1024 bytes.

29.20.31PMECC Error Location SIGMAx Register

Name: HSMC_SIGMAx [x=1..24]

Address: 0xFC05C52C [1] .. 0xFC05C588 [24], 0xFC05C528 [0] .. 0xFC05C588 [24]

Access: Read/Write

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

- **SIGMAx: Coefficient of degree x in the SIGMA polynomial**

SIGMAx belongs to the finite field GF(2^13) when the sector size is set to 512 bytes.

SIGMAx belongs to the finite field GF(2^14) when the sector size is set to 1024 bytes.

29.20.32PMECC Error Location x Register

Name: HSMC_ERRLOCx [x=0..23]

Address: 0xFC05C58C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	ERRLOCN					

7	6	5	4	3	2	1	0
ERRLOCN							

- **ERRLOCN: Error Position within the Set {sector area, spare area}**

ERRLOCN points to 1 when the first bit of the main area is corrupted.

If the sector size is set to 512 bytes, the ERRLOCN points to 4096 when the last bit of the sector area is corrupted.

If the sector size is set to 1024 bytes, the ERRLOCN points to 8192 when the last bit of the sector area is corrupted.

If the sector size is set to 512 bytes, the ERRLOCN points to 4097 when the first bit of the spare area is corrupted.

If the sector size is set to 1024 bytes, the ERRLOCN points to 8193 when the first bit of the spare area is corrupted.

29.20.33Setup Register

Name: HSMC_SETUPx [x=0..3]

Address: 0xFC05C600 [0], 0xFC05C614 [1], 0xFC05C628 [2], 0xFC05C63C [3]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	NCS_RD_SETUP					

23	22	21	20	19	18	17	16
–	–	NRD_SETUP					

15	14	13	12	11	10	9	8
–	–	NCS_WR_SETUP					

7	6	5	4	3	2	1	0
–	–	NWE_SETUP					

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

- **NWE_SETUP: NWE Setup Length**

The NWE signal setup length is defined as:

NWE setup length = (128 * NWE_SETUP[5] + NWE_SETUP[4:0]) clock cycles.

- **NCS_WR_SETUP: 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.

- **NRD_SETUP: NRD Setup Length**

The NRD signal setup length is defined as:

NRD setup length = (128 * NRD_SETUP[5] + NRD_SETUP[4:0]) clock cycles.

- **NCS_RD_SETUP: 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.

29.20.34Pulse Register

Name: HSMC_PULSEx [x=0..3]

Address: 0xFC05C604 [0], 0xFC05C618 [1], 0xFC05C62C [2], 0xFC05C640 [3]

Access: Write-only

31	30	29	28	27	26	25	24
–	NCS_RD_PULSE						

23	22	21	20	19	18	17	16
–	NRD_PULSE						

15	14	13	12	11	10	9	8
–	NCS_WR_PULSE						

7	6	5	4	3	2	1	0
–	NWE_PULSE						

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

- **NWE_PULSE: 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 must be at least one clock cycle.

- **NCS_WR_PULSE: 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 must be at least one clock cycle.

- **NRD_PULSE: NRD Pulse Length**

The NRD signal pulse length is defined as:

NRD pulse length = (256 * NRD_PULSE[6] + NRD_PULSE[5:0]) clock cycles.

The NRD pulse width must be as least 1 clock cycle.

- **NCS_RD_PULSE: NCS Pulse Length in READ Access**

In READ mode, The NCS signal pulse length is defined as:

NCS pulse length = (256 * NCS_RD_PULSE[6] + NCS_RD_PULSE[5:0]) clock cycles.

29.20.35Cycle Register

Name: HSMC_CYCLEx [x=0..3]

Address: 0xFC05C608 [0], 0xFC05C61C [1], 0xFC05C630 [2], 0xFC05C644 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	NRD_CYCLE

23	22	21	20	19	18	17	16
NRD_CYCLE							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	NWE_CYCLE

7	6	5	4	3	2	1	0
NWE_CYCLE							

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

- **NWE_CYCLE: 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.

- **NRD_CYCLE: 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.

29.20.36Timings Register

Name: HSMC_TIMINGSx [x=0..3]

Address: 0xFC05C60C [0], 0xFC05C620 [1], 0xFC05C634 [2], 0xFC05C648 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
NFSEL	–	–	–	TWB			

23	22	21	20	19	18	17	16
–	–	–	–	TRR			

15	14	13	12	11	10	9	8
–	–	–	OCMS	TAR			

7	6	5	4	3	2	1	0
TADL				TCLR			

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

- **TCLR: CLE to REN Low Delay**

Command Latch Enable falling edge to Read Enable falling edge timing.

Latch Enable Falling to Read Enable Falling = (TCLR[3] * 64) + TCLR[2:0] clock cycles.

- **TADL: ALE to Data Start**

Last address latch cycle to the first rising edge of WEN for data input.

Last address latch to first rising edge of WEN = (TADL[3] * 64) + TADL[2:0] clock cycles.

- **TAR: ALE to REN Low Delay**

Address Latch Enable falling edge to Read Enable falling edge timing.

Address Latch Enable to Read Enable = (TAR[3] * 64) + TAR[2:0] clock cycles.

- **OCMS: Off Chip Memory Scrambling Enable**

When set to one, the memory scrambling is activated. (Value must be zero if external memory is NAND Flash and NFC is used).

- **TRR: Ready to REN Low Delay**

Ready/Busy signal to Read Enable falling edge timing.

Read to REN = (TRR[3] * 64) + TRR[2:0] clock cycles.

- **TWB: WEN High to REN to Busy**

Write Enable rising edge to Ready/Busy falling edge timing.

Write Enable to Read/Busy = (TWB[3] * 64) + TWB[2:0] clock cycles.

- **NFSEL: NAND Flash Selection**

If this bit is set to one, the chip select is assigned to NAND Flash write enable and read enable lines drive the Error Correcting Code module.

29.20.37Mode Register

Name: HSMC_MODEx [x=0..3]

Address: 0xFC05C610 [0], 0xFC05C624 [1], 0xFC05C638 [2], 0xFC05C64C [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	TDF_MODE	TDF_CYCLES			
15	14	13	12	11	10	9	8
–	–	–	DBW	–	–	–	BAT
7	6	5	4	3	2	1	0
–	–	EXNW_MODE		–	–	WRITE_MODE	READ_MODE

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

- **READ_MODE: Selection of the Control Signal for Read Operation**

Value	Name	Description
0	NCS_CTRL	The Read operation is controlled by the NCS signal.
1	NRD_CTRL	The Read operation is controlled by the NRD signal.

- **WRITE_MODE: Selection of the Control Signal for Write Operation**

Value	Name	Description
0	NCS_CTRL	The Write operation is controller by the NCS signal.
1	NWE_CTRL	The Write operation is controlled by the NWE signal

- **EXNW_MODE: 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 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.

- **BAT: Byte Access Type**

This field is used only if DBW defines a 16-bit data bus.

Value	Name	Description
0	BYTE_SELECT	Byte select access type: - Write operation is controlled using NCS, NWE, NBS0, NBS1. - Read operation is controlled using NCS, NRD, NBS0, NBS1.
1	BYTE_WRITE	Byte write access type: - Write operation is controlled using NCS, NWR0, NWR1. - Read operation is controlled using NCS and NRD.

- **DBW: Data Bus Width**

Value	Name	Description
0	BIT_8	8-bit bus
1	BIT_16	16-bit bus

- **TDF_CYCLES: Data Float Time**

This field gives the integer number of clock cycles required by the external device to release the data after the rising edge of the read controlling signal. The SMC always provide one full cycle of bus turnaround after the TDF_CYCLES period. The external bus cannot be used by another chip select during TDF_CYCLES + 1 cycles. From 0 up to 15 TDF_CYCLES can be set.

- **TDF_MODE: TDF Optimization**

1: TDF optimization enabled

– The number of TDF wait states is optimized using the setup period of the next read/write access.

0: TDF optimization disabled

– The number of TDF wait states is inserted before the next access begins.

29.20.38Off Chip Memory Scrambling Register

Name: HSMC_OCMS

Address: 0xFC05C6A0

Access: Read/Write

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
–	–	–	–	–	–	SRSE	SMSE

- **SMSE: Static Memory Controller Scrambling Enable**

0: Disable “Off Chip” Scrambling for SMC access.

1: Enable “Off Chip” Scrambling for SMC access. (If OCMS bit is set in the corresponding HSMC_TIMINGSx register.)

- **SRSE: NFC Internal SRAM Scrambling Enable**

0: Disable Scrambling for NFC internal SRAM access.

1: Enable Scrambling for NFC internal SRAM access. (OCMS bit must be cleared in the corresponding HSMC_TIMINGSx register.)

29.20.39Off Chip Memory Scrambling Key1 Register

Name: HSMC_KEY1

Address: 0xFC05C6A4

Access: Write-once

31	30	29	28	27	26	25	24
KEY1							

23	22	21	20	19	18	17	16
KEY1							

15	14	13	12	11	10	9	8
KEY1							

7	6	5	4	3	2	1	0
KEY1							

- **KEY1: Off Chip Memory Scrambling (OCMS) Key Part 1**

When Off Chip Memory Scrambling is enabled by setting the HSMC_OCMS and HSMC_TIMINGS registers in accordance, the data scrambling depends on KEY1 and KEY2 values.

29.20.40Off Chip Memory Scrambling Key2 Register

Name: HSMC_KEY2

Address: 0xFC05C6A8

Access: Write-once

31	30	29	28	27	26	25	24
KEY2							

23	22	21	20	19	18	17	16
KEY2							

15	14	13	12	11	10	9	8
KEY2							

7	6	5	4	3	2	1	0
KEY2							

- **KEY2: Off Chip Memory Scrambling (OCMS) Key Part 2**

When Off Chip Memory Scrambling is enabled by setting the HSMC_OCMS and HSMC_TIMINGS registers in accordance, the data scrambling depends on KEY2 and KEY1 values.

29.20.41Write Protection Mode Register

Name: HSMC_WPMR

Address: 0xFC05C6E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables write protection if WPKEY value corresponds to 0x534D43 (“SMC” in ASCII)

1: Enables write protection if WPKEY value corresponds to 0x534D43 (“SMC” in ASCII)

Refer to Section 29.16 “Register Write Protection” for a list of write-protected registers.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x534D43	PASSWD	Writing any other value in this field aborts the write operation of bit WPEN. Always reads as 0.

29.20.42Write Protection Status Register

Name: HSMC_WPSR

Address: 0xFC05C6E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protect violation has occurred since the last read of the HSMC_WPSR.

1: A write protect violation has occurred since the last read of the HSMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

30. DMA Controller (XDMAC)

30.1 Description

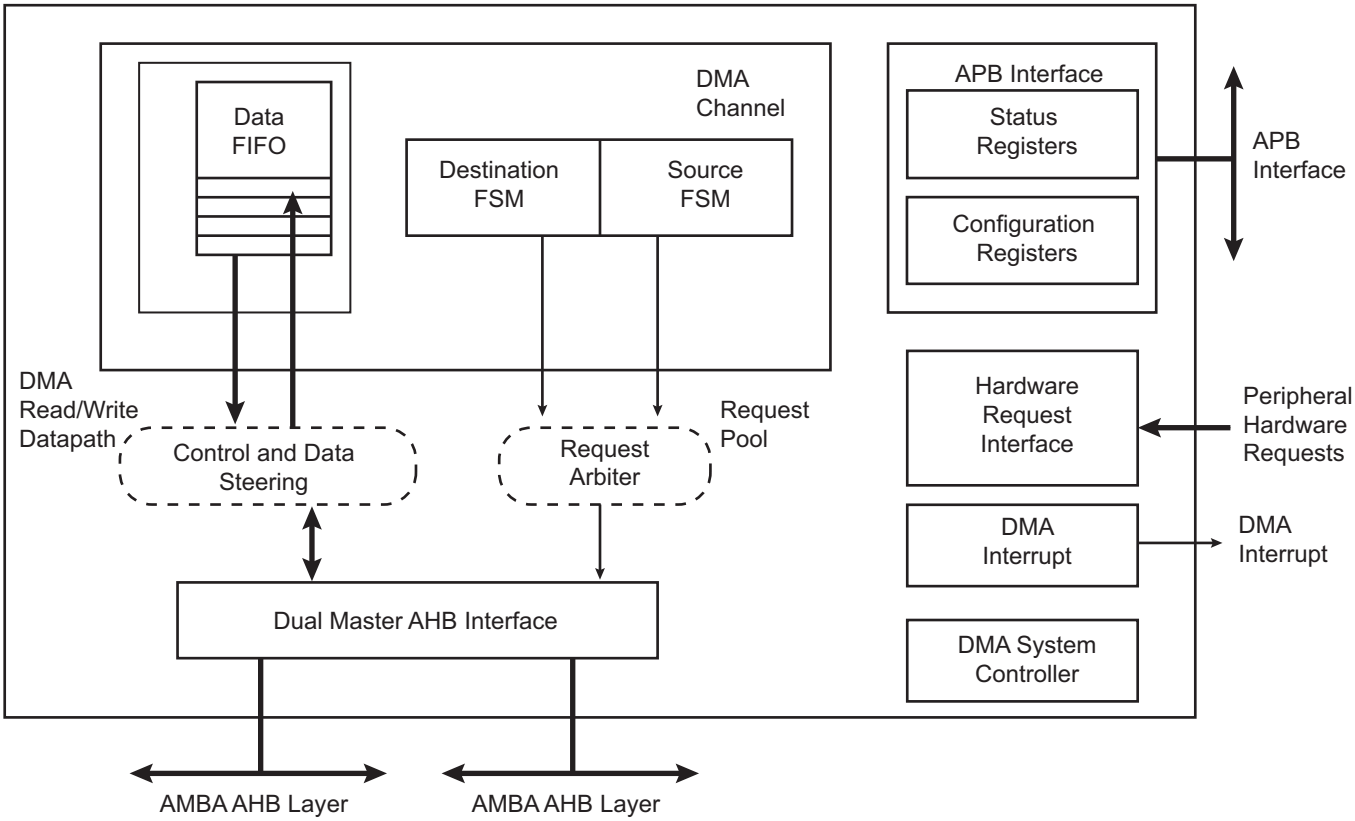
The DMA Controller (XDMAC) is a AHB-protocol central direct memory access controller. It performs peripheral data transfer and memory move operations over one or two bus ports through the unidirectional communication channel. Each channel is fully programmable and provides both peripheral or memory-to-memory transfers. The channel features are configurable at implementation.

30.2 Embedded Characteristics

- 2 AHB Master Interfaces
- 16 DMA Channels
- 48 Hardware Requests
- 1 Kbyte 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), Word (32-bit) and Double-Word (64-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

30.3 Block Diagram

Figure 30-1. DMA Controller (XDMAC) Block Diagram



30.4 DMA Controller Peripheral Connections

30.4.1 DMA Controller 0

- Dedicated to Always Secured accesses
- Can handle Non-secured accesses
- Two 64-bit Master Buses
- Embeds 16 channels and 46 hardware requests
- Thirty-two 64-bit word FIFO on all channels
- Supports:
  - Linked List support with Status Write Back operation at End of Transfer
  - Word, Half-word, Byte transfer support
  - Memory to memory transfer
  - Peripheral to memory
  - Memory to peripheral

DMA Controller 0 manages the transfer between peripherals and memory and receives the triggers from the peripherals listed in Table 30-1.

Table 30-1. DMA Channels Definition (XDMAC0)

Instance Name	Channel Type	Interface Number
HSMCI0	Receive/Transmit	0
HSMCI1	Receive/Transmit	1
TWI0	Transmit	2
TWI0	Receive	3
TWI1	Transmit	4
TWI1	Receive	5
TWI2	Transmit	6
TWI2	Receive	7
TWI3	Transmit	8
TWI3	Receive	9
SPI0	Transmit	10
SPI0	Receive	11
SPI1	Transmit	12
SPI1	Receive	13
SPI2	Transmit	14
SPI2	Receive	15
USART2	Transmit	16
USART2	Receive	17
USART3	Transmit	18
USART3	Receive	19
USART4	Transmit	20
USART4	Receive	21

Table 30-1. DMA Channels Definition (XDMAC0) (Continued)

Instance Name	Channel Type	Interface Number
UART0	Transmit	22
UART0	Receive	23
UART1	Transmit	24
UART1	Receive	25
SSC0	Transmit	26
SSC0	Receive	27
SSC1	Transmit	28
SSC1	Receive	29
DBGU	Transmit	30
DBGU	Receive	31
ADC	Receive	32
SMD	Transmit	33
SMD	Receive	34
USART0	Transmit	36
USART0	Receive	37
USART1	Transmit	38
USART1	Receive	39
AES	Receive	40
AES	Transmit	41
TDES	Transmit	42
TDES	Receive	43
SHA	Transmit	44

30.4.2 DMA Controller 1

- Always Not Secure
- Two 64-bit Master Buses
- Embeds 16 channels and 35 hardware requests
- Thirty-two 64-bit word FIFO on all channels
- Supports:
  - Linked List support with Status Write Back operation at End of Transfer
  - Word, Half-word, Byte transfer support
  - Peripheral to memory
  - Memory to peripheral

DMA Controller 1 manages the transfer between peripherals and memory and receives the triggers from the peripherals listed in Table 30-2.

Table 30-2. DMA Channels Definition (XDMAC1)

Instance Name	Channel Type	Interface Number
HSMCI0	Receive/Transmit	0
HSMCI1	Receive/Transmit	1
TWI0	Transmit	2
TWI0	Receive	3
TWI1	Transmit	4
TWI1	Receive	5
TWI2	Transmit	6
TWI2	Receive	7
TWI3	Transmit	8
TWI3	Receive	9
SPI0	Transmit	10
SPI0	Receive	11
SPI1	Transmit	12
SPI1	Receive	13
SPI2	Transmit	14
SPI2	Receive	15
USART2	Transmit	16
USART2	Receive	17
USART3	Transmit	18
USART3	Receive	19
USART4	Transmit	20
USART4	Receive	21
UART0	Transmit	22
UART0	Receive	23
UART1	Transmit	24
UART1	Receive	25

Table 30-2. DMA Channels Definition (XDMAC1) (Continued)

Instance Name	Channel Type	Interface Number
SSC0	Transmit	26
SSC0	Receive	27
SSC1	Transmit	28
SSC1	Receive	29
DBGU	Transmit	30
DBGU	Receive	31
ADC	Receive	32
SMD	Transmit	33
SMD	Receive	34

30.5 Functional Description

30.5.1 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).

30.5.2 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 30-2. XDMAC Memory Transfer Hierarchy

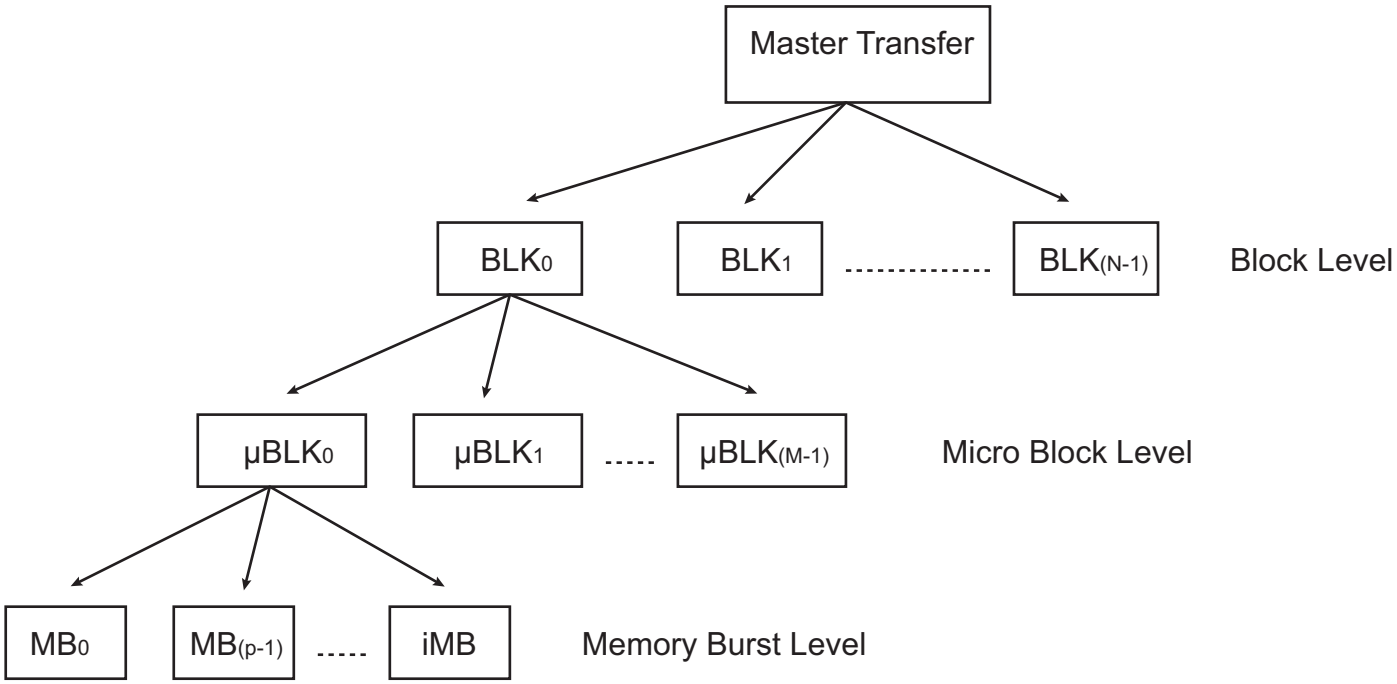
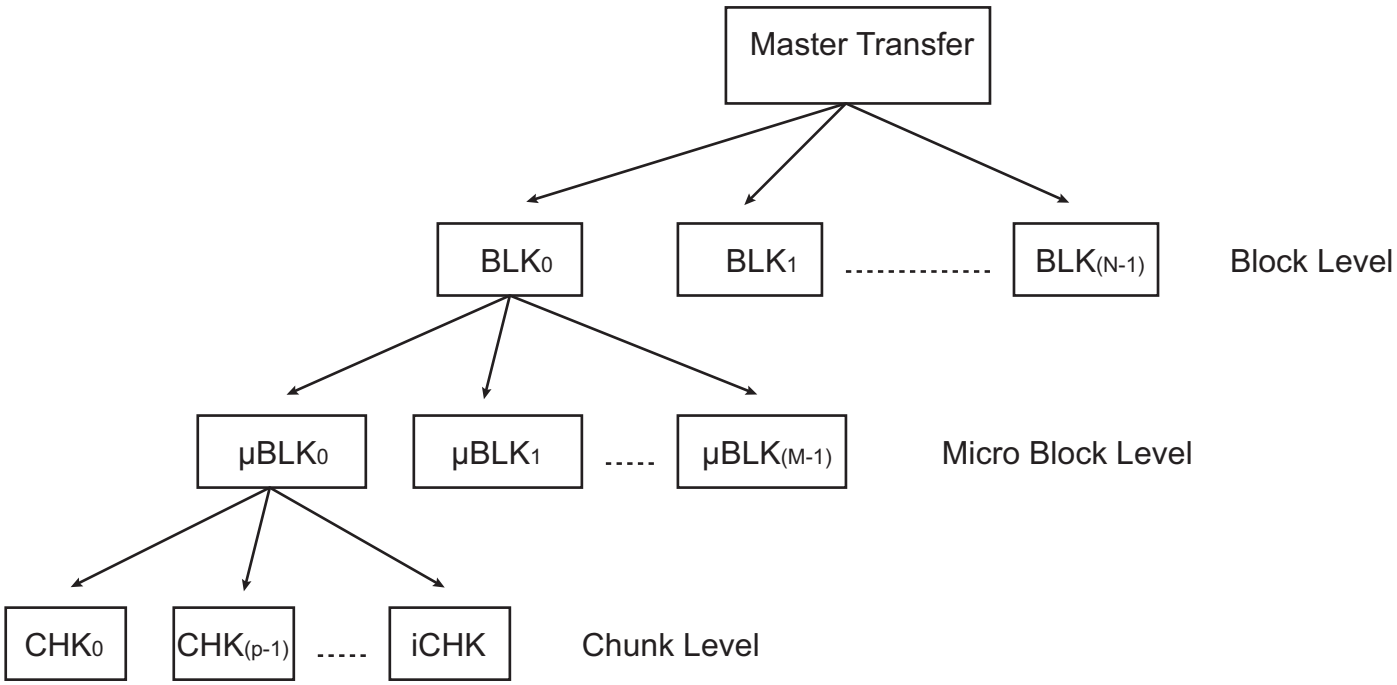


Figure 30-3. XDAMC Peripheral Transfer Hierarchy



30.5.3 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.

30.5.3.1 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.

30.5.4 XDMAC Transfer Software Operation

30.5.4.1 Single Block With Single Microblock Transfer

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):
    5. Clear XDMAC_CCx.TYPE for a memory-to-memory transfer, otherwise set this bit.
    6. Configure XDMAC_CCx.MBSIZE to the memory burst size used.
    7. Configure XDMAC_CCx.SAM and DAM to Memory Addressing mode.
    8. Configure XDMAC_CCx.DSYNC to select the peripheral transfer direction.
    9. Set XDMAC_CCx.PROT to activate a secure channel.
    10. Configure XDMAC_CCx.CSIZE to configure the channel chunk size (only relevant for peripheral synchronized transfer).
    11. Configure XDMAC_CCx.DWIDTH to configure the transfer data width.
    12. Configure XDMAC_CCx.SIF, XDMAC_CCx.DIF to configure the master interface used to read data and write data, respectively.
    13. Configure XDMAC_CCx.PERID to select the active hardware request line (only relevant for a peripheral synchronized transfer).
    14. 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.

30.5.4.2 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 configuration).
7. Program XDMAC_CBCx.BLEN with the number of microblocks of data.
8. Clear the following four 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.

30.5.4.3 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:
   1. Set XDMAC_CNDCx.NDE to enable the descriptor fetch.
   2. Set XDMAC_CNDCx.NDSUP to update the source address at the descriptor fetch time, otherwise clear this bit.
   3. Set XDMAC_CNDCx.NDDUP to update the destination address at the descriptor fetch time, otherwise clear this bit.
   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.

30.5.4.4 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.

30.6 Linked List Descriptor Operation

30.6.1 Linked List Descriptor View

30.6.1.1 Channel Next Descriptor View 0–3 Structures

Table 30-3. 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

30.6.2 Descriptor Structure Members Description

30.6.2.1 Descriptor Structure Microblock Control Member

Name: MBR_UBC

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	NVIEW		NDEN	NSEN	NDE
23	22	21	20	19	18	17	16
UBLEN							
15	14	13	12	11	10	9	8
UBLEN							
7	6	5	4	3	2	1	0
UBLEN							

- **UBLEN: Microblock Length**

This field indicates the number of data in the microblock. The microblock contains UBLEN data.

- **NDE: Next Descriptor Enable**

0: Descriptor fetch is disabled.

1: Descriptor fetch is enabled.

- **NSEN: Next Descriptor Source Update**

0: Source parameters remain unchanged.

1: Source parameters are updated when the descriptor is retrieved.

- **NDEN: Next Descriptor Destination Update**

0: Destination parameters remain unchanged.

1: Destination parameters are updated when the descriptor is retrieved.

- **NVIEW: 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

30.7 XDMAC Maintenance Software Operations

30.7.1 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.

30.7.2 Suspending a Channel

A read request suspend command is issued by writing to the XDMAC_GRS register. A write request suspend command is issued by writing to the XDMAC_GWS register. A read write suspend channel is issued by writing to the XDMAC_GRWS register. These commands have an immediate effect on the scheduling of both read and write transactions. If a transaction is already in progress, it is terminated normally. The channel is not disabled. The FIFO content is preserved. The scheduling can resume normally, clearing the bit in the same registers. Pending bytes located in the FIFO are not written out to memory. The write suspend command does not affect read request operations, i.e., read operations can still occur until the FIFO is full.

30.7.3 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.

30.7.4 Maintenance Operation Priority

30.7.4.1 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.

30.7.4.2 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.

30.7.4.3 Suspend Operation Priority

If the suspend operation is performed after a disable request, the write suspend operation is ignored.

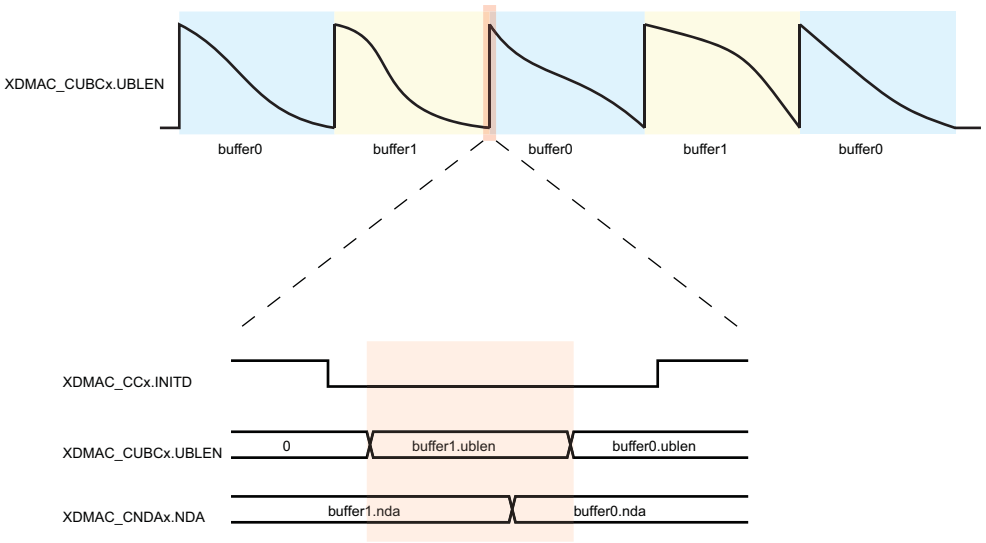
30.8 XDMAC Software Requirements

- 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, word or double-word aligned address depending on the Channel x Data Width field (DWIDTH) of the XDMAC Channel x Configuration Register.
- 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_CUBCx.UBLEN(ublen)
Read XDMAC_CCx.INITD(initd1)
Read XDMA_CNDAx.NDA(nda1)
If (nda0 == nda1 && initd0 == 1 && initd1 == 1).
Then the ublenis correct, the buffer id is nda.
Else retry
```

  Refer to Figure 30-4.

Figure 30-4. INITD Timing Diagram



30.9 Extensible DMA Controller (XDMAC) User Interface

Table 30-4. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Global Type Register	XDMAC_GTYPE	Read-only	0x00000000
0x04	Global Configuration Register	XDMAC_GCFG	Read/Write	0x00000000
0x08	Global Weighted Arbiter Configuration Register	XDMAC_GWAC	Read/Write	0x00000000
0x0C	Global Interrupt Enable Register	XDMAC_GIE	Write-only	–
0x10	Global Interrupt Disable Register	XDMAC_GID	Write-only	–
0x14	Global Interrupt Mask Register	XDMAC_GIM	Read-only	0x00000000
0x18	Global Interrupt Status Register	XDMAC_GIS	Read-only	0x00000000
0x1C	Global Channel Enable Register	XDMAC_GE	Write-only	–
0x20	Global Channel Disable Register	XDMAC_GD	Write-only	–
0x24	Global Channel Status Register	XDMAC_GS	Read-only	0x00000000
0x28	Global Channel Read Suspend Register	XDMAC_GRS	Read/Write	0x00000000
0x2C	Global Channel Write Suspend Register	XDMAC_GWS	Read/Write	0x00000000
0x30	Global Channel Read Write Suspend Register	XDMAC_GRWS	Write-only	–
0x34	Global Channel Read Write Resume Register	XDMAC_GRWR	Write-only	–
0x38	Global Channel Software Request Register	XDMAC_GSWR	Write-only	–
0x3C	Global Channel Software Request Status Register	XDMAC_GSWS	Read-only	0x00000000
0x40	Global Channel Software Flush Request Register	XDMAC_GSWF	Write-only	–
0x44–0x4C	Reserved	–	–	–
0x50+chid*0x40	Channel Interrupt Enable Register	XDMAC_CIE	Write-only	–
0x54+chid*0x40	Channel Interrupt Disable Register	XDMAC_CID	Write-only	–
0x58+chid*0x40	Channel Interrupt Mask Register	XDMAC_CIM	Read-only	0x00000000
0x5C+chid*0x40	Channel Interrupt Status Register	XDMAC_CIS	Read-only	0x00000000
0x60+chid*0x40	Channel Source Address Register	XDMAC_CSA	Read/Write	0x00000000
0x64+chid*0x40	Channel Destination Address Register	XDMAC_CDA	Read/Write	0x00000000
0x68+chid*0x40	Channel Next Descriptor Address Register	XDMAC_CNDA	Read/Write	0x00000000
0x6C+chid*0x40	Channel Next Descriptor Control Register	XDMAC_CNDC	Read/Write	0x00000000
0x70+chid*0x40	Channel Microblock Control Register	XDMAC_CUBC	Read/Write	0x00000000
0x74+chid*0x40	Channel Block Control Register	XDMAC_CBC	Read/Write	0x00000000
0x78+chid*0x40	Channel Configuration Register	XDMAC_CC	Read/Write	0x00000000
0x7C+chid*0x40	Channel Data Stride Memory Set Pattern	XDMAC_CDS_MSP	Read/Write	0x00000000
0x80+chid*0x40	Channel Source Microblock Stride	XDMAC_CSUS	Read/Write	0x00000000
0x84+chid*0x40	Channel Destination Microblock Stride	XDMAC_CDUS	Read/Write	0x00000000
0x88+chid*0x40	Reserved	–	–	–
0x8C+chid*0x40	Reserved	–	–	–
0xFEC–0xFFC	Reserved	–	–	–

30.9.1 XDMAC Global Type Register

Name: XDMAC_GTYPE

Address: 0xF0014000 (0), 0xF0004000 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	NB_REQ						

15	14	13	12	11	10	9	8
FIFO_SZ							

7	6	5	4	3	2	1	0
FIFO_SZ			NB_CH				

- **NB_CH: Number of Channels Minus One**
- **FIFO_SZ: Number of Bytes**
- **NB_REQ: Number of Peripheral Requests Minus One**

30.9.2 XDMAC Global Configuration Register

Name: XDMAC_GCFG

Address: 0xF0014004 (0), 0xF0004004 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	BXKBEN
7	6	5	4	3	2	1	0
–	–	–	–	CGDISIF	CGDISFIFO	CGDISPIPE	CGDISREG

- **CGDISREG: Configuration Registers Clock Gating Disable**

0: The automatic clock gating is enabled for the configuration registers.

1: The automatic clock gating is disabled for the configuration registers.

- **CGDISPIPE: Pipeline Clock Gating Disable**

0: The automatic clock gating is enabled for the main pipeline.

1: The automatic clock gating is disabled for the main pipeline.

- **CGDISFIFO: FIFO Clock Gating Disable**

0: The automatic clock gating is enabled for the main FIFO.

1: The automatic clock gating is disabled for the main FIFO.

- **CGDISIF: Bus Interface Clock Gating Disable**

0: The automatic clock gating is enabled for the system bus interface.

1: The automatic clock gating is disabled for the system bus interface.

- **BXKBEN: Boundary X Kilobyte Enable**

0: The 1 Kbyte boundary is used.

1: The controller does not meet the AHB specification.

30.9.3 XDMAC Global Weighted Arbiter Configuration Register

Name: XDMAC_GWAC

Address: 0xF0014008 (0), 0xF0004008 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PW3				PW2			

7	6	5	4	3	2	1	0
PW1				PW0			

- **PW0: Pool Weight 0**

This field indicates the weight of the pool 0 in the arbitration scheme of the DMA scheduler.

- **PW1: Pool Weight 1**

This field indicates the weight of the pool 1 in the arbitration scheme of the DMA scheduler.

- **PW2: Pool Weight 2**

This field indicates the weight of the pool 2 in the arbitration scheme of the DMA scheduler.

- **PW3: Pool Weight 3**

This field indicates the weight of the pool 3 in the arbitration scheme of the DMA scheduler.

30.9.4 XDMAC Global Interrupt Enable Register

Name: XDMAC_GIE

Address: 0xF001400C (0), 0xF000400C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
IE15	IE14	IE13	IE12	IE11	IE10	IE9	IE8
7	6	5	4	3	2	1	0
IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0

- **IEx: XDMAC Channel x Interrupt Enable Bit**

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.

30.9.5 XDMAC Global Interrupt Disable Register

Name: XDMAC_GID

Address: 0xF0014010 (0), 0xF0004010 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
7	6	5	4	3	2	1	0
ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0

- **IDx: XDMAC Channel x Interrupt Disable Bit**

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.

30.9.6 XDMAC Global Interrupt Mask Register

Name: XDMAC_GIM

Address: 0xF0014014 (0), 0xF0004014 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8
7	6	5	4	3	2	1	0
IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0

- **IMx: XDMAC Channel x Interrupt Mask Bit**

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.

30.9.7 XDMAC Global Interrupt Status Register

Name: XDMAC_GIS

Address: 0xF0014018 (0), 0xF0004018 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
IS15	IS14	IS13	IS12	IS11	IS10	IS9	IS8
7	6	5	4	3	2	1	0
IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0

- **ISx: XDMAC Channel x Interrupt Status Bit**

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.

30.9.8 XDMAC Global Channel Enable Register

Name: XDMAC_GE

Address: 0xF001401C (0), 0xF000401C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8
7	6	5	4	3	2	1	0
EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0

- **ENx: XDMAC Channel x Enable Bit**

0: This bit has no effect.

1: Enables channel x. This operation is permitted if the Channel x Status bit (XDMAC_GS.STx) was read as 0.

30.9.9 XDMAC Global Channel Disable Register

Name: XDMAC_GD

Address: 0xF0014020 (0), 0xF0004020 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
7	6	5	4	3	2	1	0
DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0

- **DIx: XDMAC Channel x Disable Bit**

0: This bit has no effect.

1: Disables channel x.

30.9.10 XDMAC Global Channel Status Register

Name: XDMAC_GS

Address: 0xF0014024 (0), 0xF0004024 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
ST15	ST14	ST13	ST12	ST11	ST10	ST9	ST8

7	6	5	4	3	2	1	0
ST7	ST6	ST5	ST4	ST3	ST2	ST1	ST0

- **STx: XDMAC Channel x Status Bit**

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.

30.9.11 XDMAC Global Channel Read Suspend Register

Name: XDMAC_GRS

Address: 0xF0014028 (0), 0xF0004028 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RS15	RS14	RS13	RS12	RS11	RS10	RS9	RS8

7	6	5	4	3	2	1	0
RS7	RS6	RS5	RS4	RS3	RS2	RS1	RS0

- **RSx: XDMAC Channel x Read Suspend Bit**

0: The read channel is not suspended.

1: The source requests for channel x are no longer serviced by the system scheduler.

30.9.12 XDMAC Global Channel Write Suspend Register

Name: XDMAC_GWS

Address: 0xF001402C (0), 0xF000402C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
WS15	WS14	WS13	WS12	WS11	WS10	WS9	WS8

7	6	5	4	3	2	1	0
WS7	WS6	WS5	WS4	WS3	WS2	WS1	WS0

- **WSx: XDMAC Channel x Write Suspend Bit**

0: The write channel is not suspended.

1: Destination requests are no longer routed to the scheduler.

30.9.13 XDMAC Global Channel Read Write Suspend Register

Name: XDMAC_GRWS

Address: 0xF0014030 (0), 0xF0004030 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RWS15	RWS14	RWS13	RWS12	RWS11	RWS10	RWS9	RWS8
7	6	5	4	3	2	1	0
RWS7	RWS6	RWS5	RWS4	RWS3	RWS2	RWS1	RWS0

- **RWSx: XDMAC Channel x Read Write Suspend Bit**

0: No effect.

1: Read and write requests are suspended.

30.9.14 XDMAC Global Channel Read Write Resume Register

Name: XDMAC_GRWR

Address: 0xF0014034 (0), 0xF0004034 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RWR15	RWR14	RWR13	RWR12	RWR11	RWR10	RWR9	RWR8
7	6	5	4	3	2	1	0
RWR7	RWR6	RWR5	RWR4	RWR3	RWR2	RWR1	RWR0

- **RWRx: XDMAC Channel x Read Write Resume Bit**

0: No effect.

1: Read and write requests are serviced.

30.9.15 XDMAC Global Channel Software Request Register

Name: XDMAC_GSWR

Address: 0xF0014038 (0), 0xF0004038 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
SWREQ15	SWREQ14	SWREQ13	SWREQ12	SWREQ11	SWREQ10	SWREQ9	SWREQ8
7	6	5	4	3	2	1	0
SWREQ7	SWREQ6	SWREQ5	SWREQ4	SWREQ3	SWREQ2	SWREQ1	SWREQ0

- **SWREQx: XDMAC Channel x Software Request Bit**

0: No effect.

1: Requests a DMA transfer for channel x.

30.9.16 XDMAC Global Channel Software Request Status Register

Name: XDMAC_GSWS

Address: 0xF001403C (0), 0xF000403C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	-	–	–	–	–	–	–

15	14	13	12	11	10	9	8
SWRS15	SWRS14	SWRS13	SWRS12	SWRS11	SWRS10	SWRS9	SWRS8

7	6	5	4	3	2	1	0
SWRS7	SWRS6	SWRS5	SWRS4	SWRS3	SWRS2	SWRS1	SWRS0

- **SWRSx: XDMAC Channel x Software Request Status Bit**

0: Channel x source request is serviced.

1: Channel x source request is pending.

30.9.17 XDMAC Global Channel Software Flush Request Register

Name: XDMAC_GSWF

Address: 0xF0014040 (0), 0xF0004040 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
SWF15	SWF14	SWF13	SWF12	SWF11	SWF10	SWF9	SWF8
7	6	5	4	3	2	1	0
SWF7	SWF6	SWF5	SWF4	SWF3	SWF2	SWF1	SWF0

- **SWFx: XDMAC Channel x Software Flush Request Bit**

0: No effect.

1: Requests a DMA transfer flush for channel x. This bit is only relevant when the transfer is source peripheral synchronized.

30.9.18 XDMAC Channel x [x = 0..15] Interrupt Enable Register

Name: XDMAC_CIEx [x = 0..15]

Address: 0xF0004050 (1)[0], 0xF0004090 (1)[1], 0xF00040D0 (1)[2], 0xF0004110 (1)[3], 0xF0004150 (1)[4], 0xF0004190 (1)[5], 0xF00041D0 (1)[6], 0xF0004210 (1)[7], 0xF0004250 (1)[8], 0xF0004290 (1)[9], 0xF00042D0 (1)[10], 0xF0004310 (1)[11], 0xF0004350 (1)[12], 0xF0004390 (1)[13], 0xF00043D0 (1)[14], 0xF0004410 (1)[15], 0xF0014050 (0)[0], 0xF0014090 (0)[1], 0xF00140D0 (0)[2], 0xF0014110 (0)[3], 0xF0014150 (0)[4], 0xF0014190 (0)[5], 0xF00141D0 (0)[6], 0xF0014210 (0)[7], 0xF0014250 (0)[8], 0xF0014290 (0)[9], 0xF00142D0 (0)[10], 0xF0014310 (0)[11], 0xF0014350 (0)[12], 0xF0014390 (0)[13], 0xF00143D0 (0)[14], 0xF0014410 (0)[15]

Access: Write-only

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
–	ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE

- **BIE: End of Block Interrupt Enable Bit**

0: No effect.

1: Enables end of block interrupt.

- **LIE: End of Linked List Interrupt Enable Bit**

0: No effect.

1: Enables end of linked list interrupt.

- **DIE: End of Disable Interrupt Enable Bit**

0: No effect.

1: Enables end of disable interrupt.

- **FIE: End of Flush Interrupt Enable Bit**

0: No effect.

1: Enables end of flush interrupt.

- **RBIE: Read Bus Error Interrupt Enable Bit**

0: No effect.

1: Enables read bus error interrupt.

- **WBIE: Write Bus Error Interrupt Enable Bit**

0: No effect.

1: Enables write bus error interrupt.

- **ROIE: Request Overflow Error Interrupt Enable Bit**

0: No effect.

1: Enables request overflow error interrupt.

30.9.19 XDMAC Channel x [x = 0..15] Interrupt Disable Register

Name: XDMAC_CIDx [x = 0..15]

Address: 0xF0004054 (1)[0], 0xF0004094 (1)[1], 0xF00040D4 (1)[2], 0xF0004114 (1)[3], 0xF0004154 (1)[4], 0xF0004194 (1)[5], 0xF00041D4 (1)[6], 0xF0004214 (1)[7], 0xF0004254 (1)[8], 0xF0004294 (1)[9], 0xF00042D4 (1)[10], 0xF0004314 (1)[11], 0xF0004354 (1)[12], 0xF0004394 (1)[13], 0xF00043D4 (1)[14], 0xF0004414 (1)[15], 0xF0014054 (0)[0], 0xF0014094 (0)[1], 0xF00140D4 (0)[2], 0xF0014114 (0)[3], 0xF0014154 (0)[4], 0xF0014194 (0)[5], 0xF00141D4 (0)[6], 0xF0014214 (0)[7], 0xF0014254 (0)[8], 0xF0014294 (0)[9], 0xF00142D4 (0)[10], 0xF0014314 (0)[11], 0xF0014354 (0)[12], 0xF0014394 (0)[13], 0xF00143D4 (0)[14], 0xF0014414 (0)[15]

Access: Write-only

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
–	ROID	WBEID	RBEID	FID	DID	LID	BID

- **BID: End of Block Interrupt Disable Bit**

0: No effect.

1: Disables end of block interrupt.

- **LID: End of Linked List Interrupt Disable Bit**

0: No effect.

1: Disables end of linked list interrupt.

- **DID: End of Disable Interrupt Disable Bit**

0: No effect.

1: Disables end of disable interrupt.

- **FID: End of Flush Interrupt Disable Bit**

0: No effect.

1: Disables end of flush interrupt.

- **RBEID: Read Bus Error Interrupt Disable Bit**

0: No effect.

1: Disables bus error interrupt.

- **WBEID: Write Bus Error Interrupt Disable Bit**

0: No effect.

1: Disables bus error interrupt.

- **ROID: Request Overflow Error Interrupt Disable Bit**

0: No effect.

1: Disables request overflow error interrupt.

30.9.20 XDMAC Channel x [x = 0..15] Interrupt Mask Register

Name: XDMAC_CIMx [x = 0..15]

Address: 0xF0004058 (1)[0], 0xF0004098 (1)[1], 0xF00040D8 (1)[2], 0xF0004118 (1)[3], 0xF0004158 (1)[4], 0xF0004198 (1)[5], 0xF00041D8 (1)[6], 0xF0004218 (1)[7], 0xF0004258 (1)[8], 0xF0004298 (1)[9], 0xF00042D8 (1)[10], 0xF0004318 (1)[11], 0xF0004358 (1)[12], 0xF0004398 (1)[13], 0xF00043D8 (1)[14], 0xF0004418 (1)[15], 0xF0014058 (0)[0], 0xF0014098 (0)[1], 0xF00140D8 (0)[2], 0xF0014118 (0)[3], 0xF0014158 (0)[4], 0xF0014198 (0)[5], 0xF00141D8 (0)[6], 0xF0014218 (0)[7], 0xF0014258 (0)[8], 0xF0014298 (0)[9], 0xF00142D8 (0)[10], 0xF0014318 (0)[11], 0xF0014358 (0)[12], 0xF0014398 (0)[13], 0xF00143D8 (0)[14], 0xF0014418 (0)[15]

Access: Read-only

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
–	ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM

- **BIM: End of Block Interrupt Mask Bit**

0: Block interrupt is masked.

1: Block interrupt is activated.

- **LIM: End of Linked List Interrupt Mask Bit**

0: End of linked list interrupt is masked.

1: End of linked list interrupt is activated.

- **DIM: End of Disable Interrupt Mask Bit**

0: End of disable interrupt is masked.

1: End of disable interrupt is activated.

- **FIM: End of Flush Interrupt Mask Bit**

0: End of flush interrupt is masked.

1: End of flush interrupt is activated.

- **RBEIM: Read Bus Error Interrupt Mask Bit**

0: Bus error interrupt is masked.

1: Bus error interrupt is activated.

- **WBEIM: Write Bus Error Interrupt Mask Bit**

0: Bus error interrupt is masked.

1: Bus error interrupt is activated.

- **ROIM: Request Overflow Error Interrupt Mask Bit**

0: Request overflow interrupt is masked.

1: Request overflow interrupt is activated.

30.9.21 XDMAC Channel x [x = 0..15] Interrupt Status Register

Name: XDMAC_CISx [x = 0..15]

Address: 0xF000405C (1)[0], 0xF000409C (1)[1], 0xF00040DC (1)[2], 0xF000411C (1)[3], 0xF000415C (1)[4], 0xF000419C (1)[5], 0xF00041DC (1)[6], 0xF000421C (1)[7], 0xF000425C (1)[8], 0xF000429C (1)[9], 0xF00042DC (1)[10], 0xF000431C (1)[11], 0xF000435C (1)[12], 0xF000439C (1)[13], 0xF00043DC (1)[14], 0xF000441C (1)[15], 0xF001405C (0)[0], 0xF001409C (0)[1], 0xF00140DC (0)[2], 0xF001411C (0)[3], 0xF001415C (0)[4], 0xF001419C (0)[5], 0xF00141DC (0)[6], 0xF001421C (0)[7], 0xF001425C (0)[8], 0xF001429C (0)[9], 0xF00142DC (0)[10], 0xF001431C (0)[11], 0xF001435C (0)[12], 0xF001439C (0)[13], 0xF00143DC (0)[14], 0xF001441C (0)[15]

Access: Read-only

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
–	ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS

- **BIS: End of Block Interrupt Status Bit**

0: End of block interrupt has not occurred.

1: End of block interrupt has occurred since the last read of the Status register.

- **LIS: End of Linked List Interrupt Status Bit**

0: End of linked list condition has not occurred.

1: End of linked list condition has occurred since the last read of the Status register.

- **DIS: End of Disable Interrupt Status Bit**

0: End of disable condition has not occurred.

1: End of disable condition has occurred since the last read of the Status register.

- **FIS: End of Flush Interrupt Status Bit**

0: End of flush condition has not occurred.

1: End of flush condition has occurred since the last read of the Status register.

- **RBEIS: Read Bus Error Interrupt Status Bit**

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.

- **WBEIS: Write Bus Error Interrupt Status Bit**

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.

- **ROIS: Request Overflow Error Interrupt Status Bit**

0: Overflow condition has not occurred.

1: Overflow condition has occurred at least once. (This information is only relevant for peripheral synchronized transfers.)

30.9.22 XDMAC Channel x [x = 0..15] Source Address Register

Name: XDMAC_CSAx [x = 0..15]

Address: 0xF0004060 (1)[0], 0xF00040A0 (1)[1], 0xF00040E0 (1)[2], 0xF0004120 (1)[3], 0xF0004160 (1)[4], 0xF00041A0 (1)[5], 0xF00041E0 (1)[6], 0xF0004220 (1)[7], 0xF0004260 (1)[8], 0xF00042A0 (1)[9], 0xF00042E0 (1)[10], 0xF0004320 (1)[11], 0xF0004360 (1)[12], 0xF00043A0 (1)[13], 0xF00043E0 (1)[14], 0xF0004420 (1)[15], 0xF0014060 (0)[0], 0xF00140A0 (0)[1], 0xF00140E0 (0)[2], 0xF0014120 (0)[3], 0xF0014160 (0)[4], 0xF00141A0 (0)[5], 0xF00141E0 (0)[6], 0xF0014220 (0)[7], 0xF0014260 (0)[8], 0xF00142A0 (0)[9], 0xF00142E0 (0)[10], 0xF0014320 (0)[11], 0xF0014360 (0)[12], 0xF00143A0 (0)[13], 0xF00143E0 (0)[14], 0xF0014420 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
SA							

23	22	21	20	19	18	17	16
SA							

15	14	13	12	11	10	9	8
SA							

7	6	5	4	3	2	1	0
SA							

- **SA: 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.

30.9.23 XDMAC Channel x [x = 0..15] Destination Address Register

Name: XDMAC_CDAx [x = 0..15]

Address: 0xF0004064 (1)[0], 0xF00040A4 (1)[1], 0xF00040E4 (1)[2], 0xF0004124 (1)[3], 0xF0004164 (1)[4], 0xF00041A4 (1)[5], 0xF00041E4 (1)[6], 0xF0004224 (1)[7], 0xF0004264 (1)[8], 0xF00042A4 (1)[9], 0xF00042E4 (1)[10], 0xF0004324 (1)[11], 0xF0004364 (1)[12], 0xF00043A4 (1)[13], 0xF00043E4 (1)[14], 0xF0004424 (1)[15], 0xF0014064 (0)[0], 0xF00140A4 (0)[1], 0xF00140E4 (0)[2], 0xF0014124 (0)[3], 0xF0014164 (0)[4], 0xF00141A4 (0)[5], 0xF00141E4 (0)[6], 0xF0014224 (0)[7], 0xF0014264 (0)[8], 0xF00142A4 (0)[9], 0xF00142E4 (0)[10], 0xF0014324 (0)[11], 0xF0014364 (0)[12], 0xF00143A4 (0)[13], 0xF00143E4 (0)[14], 0xF0014424 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
DA							

23	22	21	20	19	18	17	16
DA							

15	14	13	12	11	10	9	8
DA							

7	6	5	4	3	2	1	0
DA							

- **DA: 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.

30.9.24 XDMAC Channel x [x = 0..15] Next Descriptor Address Register

Name: XDMAC_CNDAx [x = 0..15]

Address: 0xF0004068 (1)[0], 0xF00040A8 (1)[1], 0xF00040E8 (1)[2], 0xF0004128 (1)[3], 0xF0004168 (1)[4], 0xF00041A8 (1)[5], 0xF00041E8 (1)[6], 0xF0004228 (1)[7], 0xF0004268 (1)[8], 0xF00042A8 (1)[9], 0xF00042E8 (1)[10], 0xF0004328 (1)[11], 0xF0004368 (1)[12], 0xF00043A8 (1)[13], 0xF00043E8 (1)[14], 0xF0004428 (1)[15], 0xF0014068 (0)[0], 0xF00140A8 (0)[1], 0xF00140E8 (0)[2], 0xF0014128 (0)[3], 0xF0014168 (0)[4], 0xF00141A8 (0)[5], 0xF00141E8 (0)[6], 0xF0014228 (0)[7], 0xF0014268 (0)[8], 0xF00142A8 (0)[9], 0xF00142E8 (0)[10], 0xF0014328 (0)[11], 0xF0014368 (0)[12], 0xF00143A8 (0)[13], 0xF00143E8 (0)[14], 0xF0014428 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
NDA							

23	22	21	20	19	18	17	16
NDA							

15	14	13	12	11	10	9	8
NDA							

7	6	5	4	3	2	1	0
NDA						–	NDAIF

- **NDAIF: Channel x Next Descriptor Interface**

0: The channel descriptor is retrieved through the system interface 0.

1: The channel descriptor is retrieved through the system interface 1.

- **NDA: 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.

30.9.25 XDMAC Channel x [x = 0..15] Next Descriptor Control Register

Name: XDMAC_CNDCx [x = 0..15]

Address: 0xF000406C (1)[0], 0xF00040AC (1)[1], 0xF00040EC (1)[2], 0xF000412C (1)[3], 0xF000416C (1)[4], 0xF00041AC (1)[5], 0xF00041EC (1)[6], 0xF000422C (1)[7], 0xF000426C (1)[8], 0xF00042AC (1)[9], 0xF00042EC (1)[10], 0xF000432C (1)[11], 0xF000436C (1)[12], 0xF00043AC (1)[13], 0xF00043EC (1)[14], 0xF000442C (1)[15], 0xF001406C (0)[0], 0xF00140AC (0)[1], 0xF00140EC (0)[2], 0xF001412C (0)[3], 0xF001416C (0)[4], 0xF00141AC (0)[5], 0xF00141EC (0)[6], 0xF001422C (0)[7], 0xF001426C (0)[8], 0xF00142AC (0)[9], 0xF00142EC (0)[10], 0xF001432C (0)[11], 0xF001436C (0)[12], 0xF00143AC (0)[13], 0xF00143EC (0)[14], 0xF001442C (0)[15]

Access: Read/Write

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
–	–	–	NDVIEW		NDDUP	NDSUP	NDE

- **NDE: Channel x Next Descriptor Enable**

0 (DSCR_FETCH_DIS): Descriptor fetch is disabled.

1 (DSCR_FETCH_EN): Descriptor fetch is enabled.

- **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.

- **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.

- **NDVIEW: 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

30.9.26 XDMAC Channel x [x = 0..15] Microblock Control Register

Name: XDMAC_CUBCx [x = 0..15]

Address: 0xF0004070 (1)[0], 0xF00040B0 (1)[1], 0xF00040F0 (1)[2], 0xF0004130 (1)[3], 0xF0004170 (1)[4], 0xF00041B0 (1)[5], 0xF00041F0 (1)[6], 0xF0004230 (1)[7], 0xF0004270 (1)[8], 0xF00042B0 (1)[9], 0xF00042F0 (1)[10], 0xF0004330 (1)[11], 0xF0004370 (1)[12], 0xF00043B0 (1)[13], 0xF00043F0 (1)[14], 0xF0004430 (1)[15], 0xF0014070 (0)[0], 0xF00140B0 (0)[1], 0xF00140F0 (0)[2], 0xF0014130 (0)[3], 0xF0014170 (0)[4], 0xF00141B0 (0)[5], 0xF00141F0 (0)[6], 0xF0014230 (0)[7], 0xF0014270 (0)[8], 0xF00142B0 (0)[9], 0xF00142F0 (0)[10], 0xF0014330 (0)[11], 0xF0014370 (0)[12], 0xF00143B0 (0)[13], 0xF00143F0 (0)[14], 0xF0014430 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
UBLEN							

15	14	13	12	11	10	9	8
UBLEN							

7	6	5	4	3	2	1	0
UBLEN							

- **UBLEN: Channel x Microblock Length**

This field indicates the number of data in the microblock. The microblock contains UBLEN data.

30.9.27 XDMAC Channel x [x = 0..15] Block Control Register

Name: XDMAC_CBCx [x = 0..15]

Address: 0xF0004074 (1)[0], 0xF00040B4 (1)[1], 0xF00040F4 (1)[2], 0xF0004134 (1)[3], 0xF0004174 (1)[4], 0xF00041B4 (1)[5], 0xF00041F4 (1)[6], 0xF0004234 (1)[7], 0xF0004274 (1)[8], 0xF00042B4 (1)[9], 0xF00042F4 (1)[10], 0xF0004334 (1)[11], 0xF0004374 (1)[12], 0xF00043B4 (1)[13], 0xF00043F4 (1)[14], 0xF0004434 (1)[15], 0xF0014074 (0)[0], 0xF00140B4 (0)[1], 0xF00140F4 (0)[2], 0xF0014134 (0)[3], 0xF0014174 (0)[4], 0xF00141B4 (0)[5], 0xF00141F4 (0)[6], 0xF0014234 (0)[7], 0xF0014274 (0)[8], 0xF00142B4 (0)[9], 0xF00142F4 (0)[10], 0xF0014334 (0)[11], 0xF0014374 (0)[12], 0xF00143B4 (0)[13], 0xF00143F4 (0)[14], 0xF0014434 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	BLEN			

7	6	5	4	3	2	1	0
BLEN							

- **BLEN: Channel x Block Length**

The length of the block is (BLEN+1) microblocks.

30.9.28 XDMAC Channel x [x = 0..15] Configuration Register

Name: XDMAC_CCx[x = 0..15]

Address: 0xF0004078 (1)[0], 0xF00040B8 (1)[1], 0xF00040F8 (1)[2], 0xF0004138 (1)[3], 0xF0004178 (1)[4], 0xF00041B8 (1)[5], 0xF00041F8 (1)[6], 0xF0004238 (1)[7], 0xF0004278 (1)[8], 0xF00042B8 (1)[9], 0xF00042F8 (1)[10], 0xF0004338 (1)[11], 0xF0004378 (1)[12], 0xF00043B8 (1)[13], 0xF00043F8 (1)[14], 0xF0004438 (1)[15], 0xF0014078 (0)[0], 0xF00140B8 (0)[1], 0xF00140F8 (0)[2], 0xF0014138 (0)[3], 0xF0014178 (0)[4], 0xF00141B8 (0)[5], 0xF00141F8 (0)[6], 0xF0014238 (0)[7], 0xF0014278 (0)[8], 0xF00142B8 (0)[9], 0xF00142F8 (0)[10], 0xF0014338 (0)[11], 0xF0014378 (0)[12], 0xF00143B8 (0)[13], 0xF00143F8 (0)[14], 0xF0014438 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	PERID						

23	22	21	20	19	18	17	16
WRIP	RDIP	INITD	–	DAM		SAM	

15	14	13	12	11	10	9	8
–	DIF	SIF	DWIDTH		CSIZE		

7	6	5	4	3	2	1	0
MEMSET	SWREQ	PROT	DSYNC	–	MBSIZE		TYPE

- **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).

- **MBSIZE: 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.

- **DSYNC: Channel x Synchronization**

0 (PER2MEM): Peripheral-to-memory transfer.

1 (MEM2PER): Memory-to-peripheral transfer.

- **PROT: Channel x Protection**

0 (SEC): Channel is secured.

1 (UNSEC): Channel is unsecured.

- **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.

- **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.

- **CSIZE: 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

- **DWIDTH: 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
3	DWORD	The data size is set to 64 bits

- **SIF: Channel x Source Interface Identifier**

0 (AHB_IF0): The data is read through the system bus interface 0.

1 (AHB_IF1): The data is read through the system bus interface 1.

- **DIF: Channel x Destination Interface Identifier**

0 (AHB_IF0): The data is written through the system bus interface 0.

1 (AHB_IF1): The data is written though the system bus interface 1.

- **SAM: 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.

- **DAM: 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.

- **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. Refer to Section 30.8 “XDMAC Software Requirements”.

- **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.

- **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.

- **PERID: Channel x Peripheral Hardware Request Line Identifier**

This field contains the peripheral hardware request line identifier. PERID refers to identifiers defined in Section 30.4 “DMA Controller Peripheral Connections”.

30.9.29 XDMAC Channel x [x = 0..15] Data Stride Memory Set Pattern Register

Name: XDMAC_CDS_MSPx [x = 0..15]

Address: 0xF000407C (1)[0], 0xF00040BC (1)[1], 0xF00040FC (1)[2], 0xF000413C (1)[3], 0xF000417C (1)[4], 0xF00041BC (1)[5], 0xF00041FC (1)[6], 0xF000423C (1)[7], 0xF000427C (1)[8], 0xF00042BC (1)[9], 0xF00042FC (1)[10], 0xF000433C (1)[11], 0xF000437C (1)[12], 0xF00043BC (1)[13], 0xF00043FC (1)[14], 0xF000443C (1)[15], 0xF001407C (0)[0], 0xF00140BC (0)[1], 0xF00140FC (0)[2], 0xF001413C (0)[3], 0xF001417C (0)[4], 0xF00141BC (0)[5], 0xF00141FC (0)[6], 0xF001423C (0)[7], 0xF001427C (0)[8], 0xF00142BC (0)[9], 0xF00142FC (0)[10], 0xF001433C (0)[11], 0xF001437C (0)[12], 0xF00143BC (0)[13], 0xF00143FC (0)[14], 0xF001443C (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
DDS_MSP							

23	22	21	20	19	18	17	16
DDS_MSP							

15	14	13	12	11	10	9	8
SDS_MSP							

7	6	5	4	3	2	1	0
SDS_MSP							

- **SDS_MSP: 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.

- **DDS_MSP: 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.

30.9.30 XDMAC Channel x [x = 0..15] Source Microblock Stride Register

Name: XDMAC_CSUSx [x = 0..15]

Address: 0xF0004080 (1)[0], 0xF00040C0 (1)[1], 0xF0004100 (1)[2], 0xF0004140 (1)[3], 0xF0004180 (1)[4], 0xF00041C0 (1)[5], 0xF0004200 (1)[6], 0xF0004240 (1)[7], 0xF0004280 (1)[8], 0xF00042C0 (1)[9], 0xF0004300 (1)[10], 0xF0004340 (1)[11], 0xF0004380 (1)[12], 0xF00043C0 (1)[13], 0xF0004400 (1)[14], 0xF0004440 (1)[15], 0xF0014080 (0)[0], 0xF00140C0 (0)[1], 0xF0014100 (0)[2], 0xF0014140 (0)[3], 0xF0014180 (0)[4], 0xF00141C0 (0)[5], 0xF0014200 (0)[6], 0xF0014240 (0)[7], 0xF0014280 (0)[8], 0xF00142C0 (0)[9], 0xF0014300 (0)[10], 0xF0014340 (0)[11], 0xF0014380 (0)[12], 0xF00143C0 (0)[13], 0xF0014400 (0)[14], 0xF0014440 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
SUBS							

15	14	13	12	11	10	9	8
SUBS							

7	6	5	4	3	2	1	0
SUBS							

- **SUBS: Channel x Source Microblock Stride**

Two’s complement microblock stride for channel x.

30.9.31 XDMAC Channel x [x = 0..15] Destination Microblock Stride Register

Name: XDMAC_CDUSx [x = 0..15]

Address: 0xF0004084 (1)[0], 0xF00040C4 (1)[1], 0xF0004104 (1)[2], 0xF0004144 (1)[3], 0xF0004184 (1)[4], 0xF00041C4 (1)[5], 0xF0004204 (1)[6], 0xF0004244 (1)[7], 0xF0004284 (1)[8], 0xF00042C4 (1)[9], 0xF0004304 (1)[10], 0xF0004344 (1)[11], 0xF0004384 (1)[12], 0xF00043C4 (1)[13], 0xF0004404 (1)[14], 0xF0004444 (1)[15], 0xF0014084 (0)[0], 0xF00140C4 (0)[1], 0xF0014104 (0)[2], 0xF0014144 (0)[3], 0xF0014184 (0)[4], 0xF00141C4 (0)[5], 0xF0014204 (0)[6], 0xF0014244 (0)[7], 0xF0014284 (0)[8], 0xF00142C4 (0)[9], 0xF0014304 (0)[10], 0xF0014344 (0)[11], 0xF0014384 (0)[12], 0xF00143C4 (0)[13], 0xF0014404 (0)[14], 0xF0014444 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
DUBS							

15	14	13	12	11	10	9	8
DUBS							

7	6	5	4	3	2	1	0
DUBS							

- **DUBS: Channel x Destination Microblock Stride**

Two’s complement microblock stride for channel x.

31. LCD Controller (LCDC)

31.1 Description

The LCD Controller (LCDC) consists of logic for transferring LCD image data from an external display buffer to an LCD module. The LCD has one display input buffer per overlay that fetches pixels through the dual AHB master interface and a lookup table to allow palletized display configurations. The LCD controller is programmable on a per overlay basis, and supports different LCD resolutions, window sizes, image formats and pixel depths.

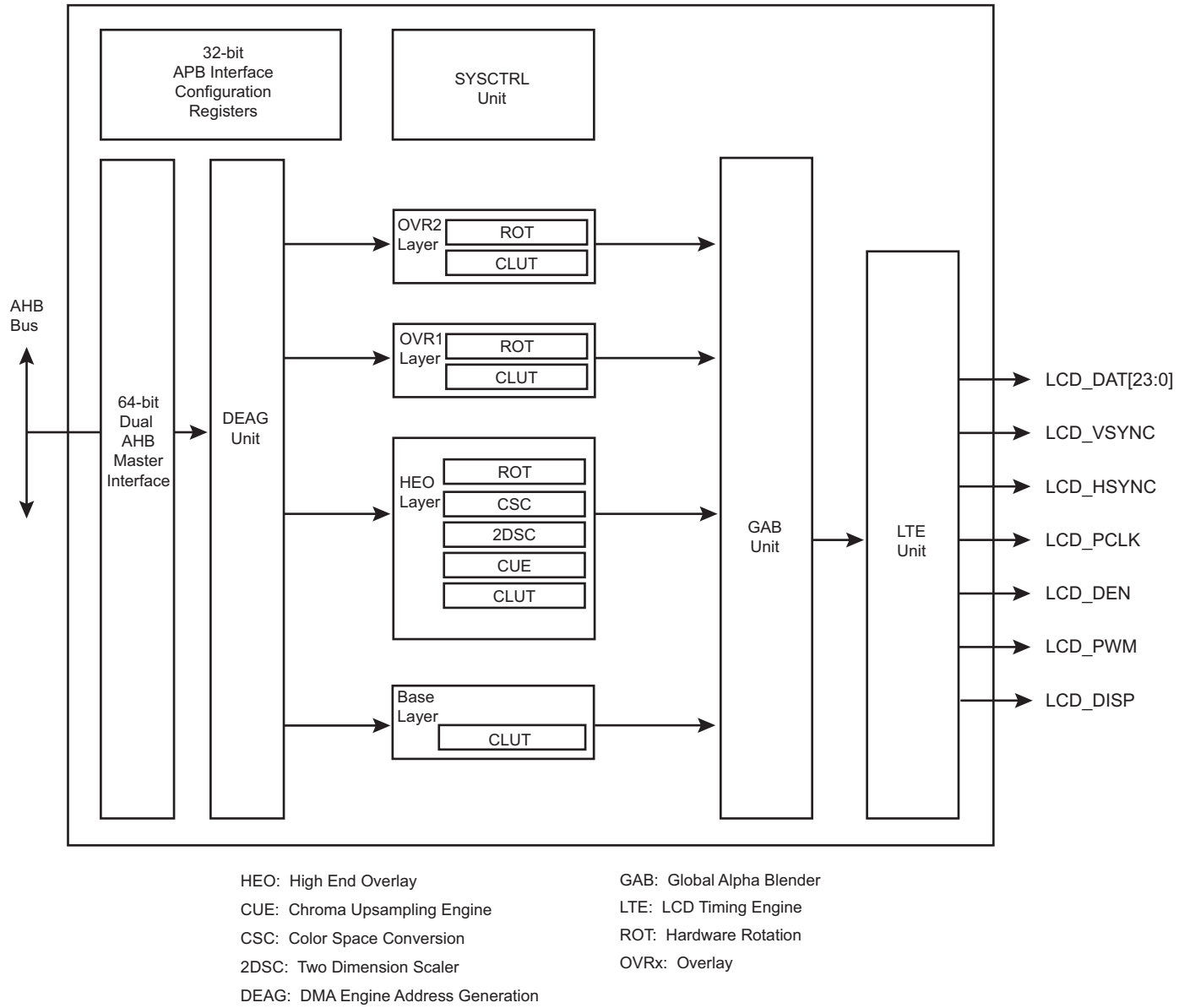
The LCD is connected to the ARM Advanced High Performance Bus (AHB) as a master for reading pixel data. It also integrates an APB interface to configure its registers.

31.2 Embedded Characteristics

- Dual AHB Master Interface
- Supports Single Scan Active TFT Display
- Supports 12-, 16-, 18- and 24-bit Output Mode through the Spatial Dithering Unit
- Asynchronous Output Mode Supported (at synthesis time)
- 1, 2, 4, 8 bits per Pixel (Palletized)
- 12, 16, 18, 19, 24, 25 and 32 bits per Pixel (Non-palletized)
- Supports One Base Layer (Background)
- Supports One Overlay 1 Layer Window
- Supports One High End Overlay (HEO) Window
- Little Endian Memory Organization
- Programmable Timing Engine, with Integer Clock Divider
- Programmable Polarity for Data, Line Synchro and Frame Synchro
- Display Size up to 2048x2048, or up to 720p in Video Format
- Color Lookup Table with up to 256 Entries and Predefined 8-bit Alpha
- Programmable Negative and Positive Row Striding for all Layers
- Programmable Negative and Positive Pixel Striding for Layers
- High End Overlay supports 4:2:0 Planar Mode and Semiplanar Mode
- High End Overlay supports 4:2:2 Planar Mode, Semiplanar Mode and Packed
- High End Overlay includes Chroma Upsampling Unit
- Horizontal and Vertical Rescaling Unit with Edge Interpolation and Independent Non-Integer Ratio, up to 1024x768
- Hidden Layer Removal supported
- Integrates Fully Programmable Color Space Conversion
- Blender Function Supports Arbitrary 8-bit Alpha Value and Chroma Keying
- DMA User Interface uses Linked List Structure and Add-to-queue Structure

31.3 Block Diagram

Figure 31-1. Block Diagram



HEO: High End Overlay
CUE: Chroma Upsampling Engine
CSC: Color Space Conversion
2DSC: Two Dimension Scaler
DEAG: DMA Engine Address Generation

GAB: Global Alpha Blender
LTE: LCD Timing Engine
ROT: Hardware Rotation
OVRx: Overlay

31.4 I/O Lines Description

Table 31-1. I/O Lines Description

Name	Description	Type
LCD_PWM	Contrast control signal, using Pulse Width Modulation	**Output**
LCD_HSYNC	Horizontal Synchronization Pulse	Output
LCD_VSYNC	Vertical Synchronization Pulse	Output
LCD_DAT[23:0]	LCD 24-bit data bus	Output
LCD_DEN	Data Enable	Output

Table 31-1. I/O Lines Description (Continued)

Name	Description	Type
LCD_DISP	Display Enable signal	Output
LCD_PCLK	Pixel Clock	Output

31.5 Product Dependencies

31.5.1 I/O Lines

The pins used for interfacing the LCD Controller 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 LCD Controller are not used by the application, they can be used for other purposes by the PIO Controller.

Table 31-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
LCDC	LCDDAT0	PA0	A
LCDC	LCDDAT1	PA1	A
LCDC	LCDDAT2	PA2	A
LCDC	LCDDAT3	PA3	A
LCDC	LCDDAT4	PA4	A
LCDC	LCDDAT5	PA5	A
LCDC	LCDDAT6	PA6	A
LCDC	LCDDAT7	PA7	A
LCDC	LCDDAT8	PA8	A
LCDC	LCDDAT9	PA9	A
LCDC	LCDDAT10	PA10	A
LCDC	LCDDAT11	PA11	A
LCDC	LCDDAT12	PA12	A
LCDC	LCDDAT13	PA13	A
LCDC	LCDDAT14	PA14	A
LCDC	LCDDAT15	PA15	A
LCDC	LCDDAT16	PA16	A
LCDC	LCDDAT17	PA17	A
LCDC	LCDDAT18	PA18	A
LCDC	LCDDAT19	PA19	A
LCDC	LCDDAT20	PA20	A
LCDC	LCDDAT21	PA21	A
LCDC	LCDDAT22	PA22	A
LCDC	LCDDAT23	PA23	A
LCDC	LCDDEN	PA29	A
LCDC	LCDDISP	PA25	A
LCDC	LCDHSYNC	PA27	A

Table 31-2. I/O Lines (Continued)

LCDC	LCDPCK	PA28	A
LCDC	LCDPWM	PA24	A
LCDC	LCDVSYNC	PA26	A

31.5.2 Power Management

The LCD Controller is not continuously clocked. The user must first enable the LCD Controller clock in the Power Management Controller (PMC_PCER) before using it.

31.5.3 Interrupt Sources

The LCD Controller interrupt line is connected to one of the internal sources of the interrupt con-troller. Using the

Table 31-3. Peripheral IDs

Instance	ID
LCDC	51

LCD Controller interrupt requires prior programming of the interrupt controller.

31.6 Functional Description

The LCD module integrates the following digital blocks:

- DMA Engine Address Generation (DEAG)—This block performs data prefetch and requests access to the AHB interface.
- Input Overlay FIFO—Stores the stream of pixels
- Color Lookup Table (CLUT)—These 256 RAM-based lookup table entries are selected when the color depth is set to 1, 2, 4 or 8 bpp.
- Chroma Upsampling Engine (CUE)—This block is selected when the input image sampling format is YUV (Y'CbCr) 4:2:0 and converts it to higher quality 4:4:4 image.
- Color Space Conversion (CSC)—changes the color space from YUV to RGB
- Two Dimension Scaler (2DSC)—Resizes the image
- Global Alpha Blender (GAB)—Performs programmable 256-level alpha blending
- Output FIFO—Stores the blended pixel prior to display
- LCD Timing Engine—Provides a fully programmable HSYNC-VSYNC interface

The DMA controller reads the image through the AHB master interface. The LCD controller engine formats the display data, then the GAB performs alpha blending if required, and writes the final pixel into the output FIFO. The programmable timing engine drives a valid pixel onto the LCD_DAT[23:0] display bus.

31.6.1 Timing Engine Configuration

31.6.1.1 Pixel Clock Period Configuration

The pixel clock (LCD_PCLK) generated by the timing engine is the source clock divided by the field CLKDIV in the LCDC_LCDCFG0 register. The source clock can be selected between the system clock and the 2x system clock with the field CLKSEL located in the LCDC_LCDCFG0 register.

Pixel clock period formula:

$$\text{LCD\_PCLK} = \frac{\text{source clock}}{\text{CLKDIV} + 2}$$

The pixel clock polarity is also programmable.

31.6.1.2 Horizontal and Vertical Synchronization Configuration

The following fields are used to configure the timing engine:

- LCDC_LCDCFG1.HSPW
- LCDC_LCDCFG1.VSPW
- LCDC_LCDCFG2.VFPW
- LCDC_LCDCFG2.VBPW
- LCDC_LCDCFG3.HFPW
- LCDC_LCDCFG3.HBPW
- LCDC_LCDCFG4.PPL
- LCDC_LCDCFG4.RPF

The polarity of output signals is also programmable.

31.6.1.3 Timing Engine Powerup Software Operation

The following sequence is used to enable the display:

1. Configure LCD timing parameters, signal polarity and clock period.
2. Enable the pixel clock by writing a one to bit LCDC_LCDEN.CLKEN.
3. Poll bit LCDC_LCDSR.CLKSTS to check that the clock is running.
4. Enable Horizontal and Vertical Synchronization by writing a one to bit LCDC_LCDEN.SYNCEN.
5. Poll bit LCDC_LCDSR.LCDSTS to check that the synchronization is up.
6. Enable the display power signal by writing a one to bit LCDC_LCDEN.DISPEN.

31.6.1.4 Poll bit LCDC_LCDSR.DISPSTS to check that the power signal is activated.

The field LCDC_LCDCFG5.GUARDTIME is used to configure the number of frames before the assertion of the DISP signal.

31.6.1.5 Timing Engine Powerdown Software Operation

The following sequence is used to disable the display:

1. Disable the DISP signal by writing bit LCDC_LCDDIS.DISPDIS.
2. Poll bit LCDC_LCDSR.DISPSTS to verify that the DISP is no longer activated.
3. Disable the HSYNC and VSYNC signals by writing a one to bit LCDC_LCDDIS.SYNCDIS.
4. Poll bit LCDC_LCDSR.LCDSTS to check that the synchronization is off.
5. Disable the pixel clock by writing a one to bit LCDC_LCDDIS.CLKDIS.

31.6.2 DMA Software Operations

31.6.2.1 DMA Channel Descriptor (DSCR) Alignment and Structure

The DMA Channel Descriptor (DSCR) must be aligned on a 64-bit boundary.

The DMA Channel Descriptor structure contains three fields:

- DSCR.CHXADDR: Frame Buffer base address register
- DSCR.CHXCTRL: Transfer Control register
- DSCR.CHXNEXT: Next Descriptor Address register

Table 31-4. DMA Channel Descriptor Structure

System Memory	Structure Field for Channel CHX
DSCR + 0x0	ADDR
DSCR + 0x4	CTRL
DSCR + 0x8	NEXT

31.6.2.2 Enabling a DMA Channel

Follow the steps below to enable a DMA channel:

1. Check the status of the channel by reading the CHXCHSR register.
2. Write the channel descriptor (DSCR) structure in the system memory by writing DSCR.CHXADDR Frame base address, DSCR.CHXCTRL channel control and DSCR.CHXNEXT next descriptor location.
3. If more than one descriptor is expected, the field DFETCH of DSCR.CHXCTRL is set to ‘1’ to enable the descriptor fetch operation.
4. Write the DSCR.CHXNEXT register with the address location of the descriptor structure and set DFETCH field of the DSCR.CHXCTRL register to ‘1’.
5. Enable the relevant channel by writing one to the CHEN field of the CHXCHER register.
6. An interrupt may be raised if unmasked when the descriptor has been loaded.

31.6.2.3 Disabling a DMA Channel

Follow the steps below to disable a DMA channel:

1. Clearing the DFETCH bit in the DSCR.CHXCTRL field of the DSCR structure disables the channel at the end of the frame.
2. Setting the DSCR.CHXNEXT field of the DSCR structure disables the channel at the end of the frame.
3. Writing one to the CHDIS field of the CHXCHDR register disables the channel at the end of the frame.
4. Writing one to the CHRST field of the CHXCHDR register disables the channel immediately. This may occur in the middle of the image.
5. Polling CHSR field in the CHXCHSR register until the channel is successfully disabled.

31.6.2.4 DMA Dynamic Linking of a New Transfer Descriptor

1. Write the new descriptor structure in the system memory.
2. Write the address of the new structure in the CHXHEAD register.
3. Add the new structure to the queue of descriptors by writing one to the A2QEN field of the CHXCHER register.
4. The new descriptor will be added to the queue on the next frame.
5. An interrupt will be raised if unmasked, when the head descriptor structure has been loaded by the DMA channel.

31.6.2.5 DMA Interrupt Generation

The DMA Controller operation sets the following interrupt flags in the Interrupt Status register CHXISR:

- DMA field indicates that the DMA transfer is completed.
- DSCR field indicates that the descriptor structure is loaded in the DMA controller.
- ADD field indicates that a descriptor has been added to the descriptor queue.
- DONE field indicates that the channel transfer has terminated and the channel is automatically disabled.

31.6.2.6 DMA Address Alignment Requirements

When programming the DSCR.CHXADDR field of the DSCR structure, the following requirement must be met.

Table 31-5. DMA Address Alignment when CLUT Mode is Selected

CLUT Mode	DMA Address Alignment
1 bpp	8 bits
2 bpp	8 bits
4 bpp	8 bits
8 bpp	8 bits

Table 31-6. DMA Address Alignment when RGB Mode is Selected

RGB Mode	DMA Address Alignment
12 bpp RGB 444	16 bits
16 bpp ARGB 4444	16 bits
16 bpp RGBA 4444	16 bits
16 bpp RGB 565	16 bits
16 bpp TRGB 1555	16 bits
18 bpp RGB 666	32 bits
18 bpp RGB 666 PACKED	8 bits
19 bpp TRGB 1666	32 bits
19 bpp TRGB 1666	8 bits
24 bpp RGB 888	32 bits
24 bpp RGB 888 PACKED	8 bits
25 bpp TRGB 1888	32 bits
32 bpp ARGB 8888	32 bits
32 bpp RGBA 8888	32 bits

Table 31-7. DMA Address Alignment when YUV Mode is Selected

YUV Mode	DMA Address Alignment
32 bpp AYCrCb	32 bits
16 bpp YCrCb 4:2:2	32 bits
16 bpp semiplanar YCrCb 4:2:2	Y 8 bits
	CrCb 16 bits
16 bpp planar YCrCb 4:2:2	Y 8 bits
	Cr 8 bits
	Cb 8 bits
12 bpp YCrCb 4:2:0	Y 8 bits
	CrCb 16 bits
12 bpp YCrCb 4:2:0	Y 8 bits
	Cr 8 bits
	Cb 8 bits

31.6.3 Overlay Software Configuration

31.6.3.1 System Bus Access Attributes

These attributes are defined to improve bandwidth of the overlay.

- LOCKDIS bit—When set to ‘1’, the AHB lock signal is not asserted when the PSTRIDE value is different from zero (rotation in progress).
- ROTDIS bit—When set to ‘1’, the Pixel Striding optimization is disabled.
- DLBO bit—When set to ‘1’, only defined burst lengths are performed when the DMA channel retrieves the data from the memory.

- BLEN field—Defines the maximum burst length of the DMA channel.
- SIF bit—Defines the targeted DMA interface.

31.6.3.2 Color Attributes

- CLUTMODE field—Selects the Color Lookup Table mode.
- RGBMODE field—Selects the RGB mode.
- YUVMODE field—Selects the Luminance Chrominance mode.

31.6.3.3 Window Position, Size, Scaling and Striding Attributes

- XPOS and YPOS fields—Define the position of the overlay window.
- XSIZE and YSIZE fields—Define the size of the displayed window.
- XMEMSIZE and YMEMSIZE fields—Define the size of the image frame buffer.
- XSTRIDE and PSTRIDE fields—Define the line and pixel striding.
- XFACTOR and YFACTOR fields—Define the scaling ratio.

The position and size attributes are to be programmed to keep the window within the display area.

When the Color Lookup Table mode is enabled, the restrictions detailed in the following table apply on the horizontal and vertical window sizes.

Table 31-8. Color Lookup Table Mode and Window Size

CLUT Mode	X-Y Size Requirement
1 bpp	Multiple of 8 pixels
2 bpp	Multiple of 4 pixels
4 bpp	Multiple of 2 pixels
8 bpp	Free size

Pixel striding is disabled when CLUT mode is enabled.

When YUV mode is enabled, the restrictions detailed in the following table apply on the window size.

Table 31-9. YUV Mode and Window Size

YUV Mode	X-Y Requirement, Scaling Turned Off	X-Y Requirement, Scaling Turned On
AYUV	Free size	X-Y size is greater than 5
YUV 4:2:2 packed	XSIZE is greater than 2 pixels	X-Y size is greater than 5
YUV 4:2:2 semiplanar	XSIZE is greater than 2 pixels	X-Y size is greater than 5
YUV 4:2:2 planar	XSIZE is greater than 2 pixels	X-Y size is greater than 5
YUV 4:2:0 semiplanar	XSIZE is greater that 2 pixels	X-Y size is greater than 5
YUV 4:2:0 planar	XSIZE is greater than 2 pixels	X-Y size is greater than 5

In RGB mode, there is no restriction on the line length.

31.6.3.4 Overlay Blender Attributes

When two or more video layers are used, alpha blending is performed to define the final image displayed. Each window has its own blending attributes.

- CRKEY bit—Enables the chroma keying and match logic.
- INV bit—Performs bit inversion at pixel level.
- ITER2BL bit—When written to ‘1’, the iterated data path is selected.

- ITER bit—When written to ‘1’, the iterated value is used in the iterated data path, otherwise the iterated value is set to 0.
- REVALPHA bit—Uses the reverse alpha value.
- GAEN bit—Enables the global alpha value in the data path.
- LAEN bit—Enables the local alpha value from the pixel.
- OVR bit—When written to ‘1’, the overlay is selected as an input of the blender.
- DMA bit—The DMA data path is activated.
- REP bit—Enables the bit replication to fill the 24-bit internal data path.
- DSTKEY bit—When written to ‘1’, Destination keying is enabled.
- GA field—Defines the global alpha value.

31.6.3.5 Overlay Attributes Software Operation

1. When required, write the overlay attributes configuration registers.
2. Set UPDATEEN field of the CHXCHER register.
3. Poll UPDATESR field in the CHXCHSR, the update applies when that field is reset.

31.6.4 RGB Frame Buffer Memory Bitmap

31.6.4.1 1 bpp Through Color Lookup Table

Table 31-10. 1 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 1 bpp	p31	p30	p29	p28	p27	p26	p25	p24	p23	p22	p21	p20	p19	p18	p17	p16	p15	p14	p13	p12	p11	p10	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0

31.6.4.2 2 bpp Through Color Lookup Table

Table 31-11. 2 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 2 bpp	p15		p14		p13		p12		p11		p10		p9		p8		p7		p6		p5		p4		p3		p2		p1		p0	

31.6.4.3 4 bpp Through Color Lookup Table

Table 31-12. 4 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 4 bpp	p7				p6				p5				p4				p3				p2				p1				p0			

31.6.4.4 8 bpp Through Color Lookup Table

Table 31-13. 8 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 8 bpp	p3								p2								p1								p0							

31.6.4.5 12 bpp Memory Mapping, RGB 4:4:4

Table 31-14. 12 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	–				R1[3:0]				G1[3:0]				B1[3:0]				–				R0[3:0]				G0[3:0]				B0[3:0]			

31.6.4.6 16 bpp Memory Mapping with Alpha Channel, ARGB 4:4:4:4

Table 31-15. 16 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	A1[3:0]				R1[3:0]				G1[3:0]				B1[3:0]				A0[3:0]				R0[3:0]				G0[3:0]				B0[3:0]			

31.6.4.7 16 bpp Memory Mapping with Alpha Channel, RGBA 4:4:4:4

Table 31-16. 16 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	R1[3:0]				G13:0]				B1[3:0]				A1[3:0]				R0[3:0]				G0[3:0]				B0[3:0]				A0[3:0]			

31.6.4.8 16 bpp Memory Mapping with Alpha Channel, RGB 5:6:5

Table 31-17. 16 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16bpp	R1[4:0]								G1[5:0]								B1[4:0]								R0[4:0]							
	G0[5:0]								B0[4:0]																							

31.6.4.9 16 bpp Memory Mapping with Transparency Bit, ARGB 1:5:5:5

Table 31-18. 16 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 4 bpp	A1	R1[4:0]							G1[4:0]								B1[4:0]								A0	R0[4:0]						
	G0[4:0]								B0[4:0]																							

31.6.4.10 18 bpp Unpacked Memory Mapping with Transparency Bit, RGB 6:6:6

Table 31-19. 18 bpp Unpacked Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 18 bpp	–								–						R0[5:0]						G0[5:0]						B0[5:0]					

31.6.4.11 18 bpp Packed Memory Mapping with Transparency Bit, RGB 6:6:6

Table 31-20. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 18 bpp	G1[1:0]		B1[5:0]						–						R0[5:0]						G0[5:0]						B0[5:0]					

Table 31-21. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 18 bpp	R2[3:0]				G2[5:0]				B2[5:0]								–						R1[5:2]						G1[5:2]			

Table 31-22. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x8, 0x9, 0xA, 0xB

Mem addr	0xB								0xA								0x9								0x8							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 18 bpp	G4[1:0]		B4[5:0]						–				R3[5:0]				G3[5:0]								B3[3:0]						R2[5:4]	

31.6.4.12 19 bpp Unpacked Memory Mapping with Transparency Bit, RGB 1:6:6:6

Table 31-23. 19 bpp Unpacked Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 19 bpp	–								–					A0	R0[5:0]						G0[5:0]				B0[5:0]							

31.6.4.13 19 bpp Packed Memory Mapping with Transparency Bit, ARGB 1:6:6:6

Table 31-24. 19 bpp Packed Memory Mapping, Little Endian Organization at Address 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 19 bpp	G1[1:0]		B1[5:0]						–					A0	R0[5:0]						G0[5:0]				B0[5:0]							

Table 31-25. 19 bpp Packed Memory Mapping, Little Endian Organization at Address 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 19 bpp	R2[3:0]				G2[5:0]				B2[5:0]								–					A1	R1[5:2]				G1[5:2]					

Table 31-26. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x8, 0x9, 0xA, 0xB

Mem addr	0xB								0xA								0x9								0x8							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 19 bpp	G4[1:0]		B4[5:0]						–			A3	R3[5:0]				G3[5:0]								B3[3:0]						R2[5:4]	

31.6.4.14 24 bpp Unpacked Memory Mapping, RGB 8:8:8

Table 31-27. 24 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 24 bpp	–								R0[7:0]								G0[7:0]								B0[7:0]							

31.6.4.15 24 bpp Packed Memory Mapping, RGB 8:8:8

Table 31-28. 24 bpp Packed Memory Mapping, Little Endian Organization at Address 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 24 bpp	B1[7:0]								R0[7:0]								G0[7:0]								B0[7:0]							

Table 31-29. 24 bpp Packed Memory Mapping, Little Endian Organization at Address 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 24 bpp	G2[7:0]								B2[7:0]								R1[7:0]								G1[7:0]							

31.6.4.16 25 bpp Memory Mapping, ARGB 1:8:8:8

Table 31-30. 25 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 25 bpp	–							A0	R0[7:0]								G0[7:0]								B0[7:0]							

31.6.4.17 32 bpp Memory Mapping, ARGB 8:8:8:8

Table 31-31. 32 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 32 bpp	A0[7:0]								R0[7:0]								G0[7:0]								B0[7:0]							

31.6.4.18 32 bpp Memory Mapping, RGBA 8:8:8:8

Table 31-32. 32 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 32 bpp	R0[7:0]								G0[7:0]								B0[7:0]								A0[7:0]							

31.6.5 YUV Frame Buffer Memory Mapping

31.6.5.1 AYCbCr 4:4:4 Interleaved Frame Buffer Memory Mapping

Table 31-33. 32 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	A0[7:0]								Y0[7:0]								Cb0[7:0]								Cr0[7:0]							

31.6.5.2 4:2:2 Interleaved Mode Frame Buffer Memory Mapping

Table 31-34. 16 bpp 4:2:2 interleaved Mode 0

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Cr0[7:0]								Y1[7:0]								Cb0[7:0]								Y0[7:0]							

Table 31-35. 16 bpp 4:2:2 interleaved Mode 1

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Y1[7:0]								Cr0[7:0]								Y0[7:0]								Cb0[7:0]							

Table 31-36. 16 bpp 4:2:2 interleaved Mode 2

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Cb0[7:0]								Y1[7:0]								Cr0[7:0]								Y0[7:0]							

Table 31-37. 16 bpp 4:2:2 interleaved Mode 3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Y1[7:0]								Cb0[7:0]								Y0[7:0]								Cr0[7:0]							

31.6.5.3 4:2:2 Semiplanar Mode Frame Buffer Memory Mapping

Table 31-38. 4:2:2 Semiplanar Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 31-39. 4:2:2 Semiplanar Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Cb2[7:0]								Cr2[7:0]								Cb0[7:0]								Cr0[7:0]							

31.6.5.4 4:2:2 Planar Mode Frame Buffer Memory Mapping

Table 31-40. 4:2:2 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 31-41. 4:2:2 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	C3[7:0]								C2[7:0]								C1[7:0]								C0[7:0]							

31.6.5.5 4:2:0 Planar Mode Frame Buffer Memory Mapping

In Planar Mode, the three video components Y, Cr and Cb are split into three memory areas and stored in a raster-scan order. These three memory planes are contiguous and always aligned on a 32-bit boundary.

Table 31-42. 4:2:0 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 31-43. 4:2:0 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Y7[7:0]								Y6[7:0]								Y5[7:0]								Y4[7:0]							

Table 31-44. 4:2:0 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	C3[7:0]								C2[7:0]								C1[7:0]								C0[7:0]							

Table 31-45. 4:2:0 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	C7[7:0]								C6:[7:0]								C5[7:0]								C4[7:0]							

31.6.5.6 4:2:0 Semiplanar Frame Buffer Memory Mapping

Table 31-46. 4:2:0 Semiplanar Mode Luminance Memory Mapping, Little Endian Organization

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 31-47. 4:2:0 Semiplanar Mode Chrominance Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Cb1[7:0]								Cr1[7:0]								Cb0[7:0]								Cr0[7:0]							

31.6.6 Chrominance Upsampling Unit

Both 4:2:2 and 4:2:0 input formats are supported by the LCD module. In 4:2:2, the two chrominance components are sampled at half the sample rate of the luminance. The horizontal chrominance resolution is halved. When this input format is selected, the chrominance upsampling unit uses two chrominances to interpolate the missing component.

In 4:2:0, Cr and Cb components are subsampled at a factor of two vertically and horizontally. When this input mode is selected, the chrominance upsampling unit uses two and four chroma components to generate the missing horizontal and vertical components.

Figure 31-2. 4:2:2 Upsampling Algorithm

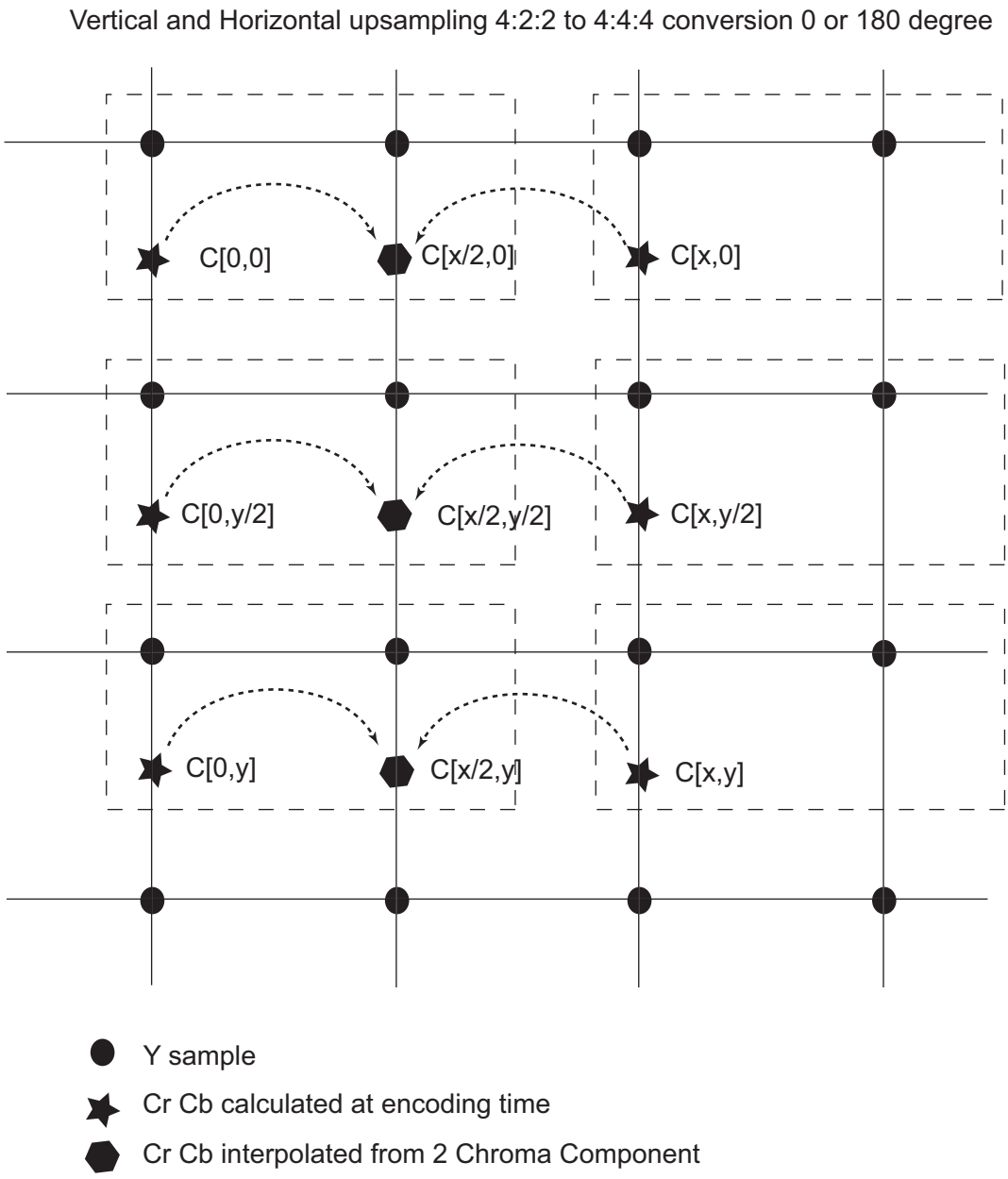


Figure 31-3. 4:2:2 Packed Upsampling Algorithm

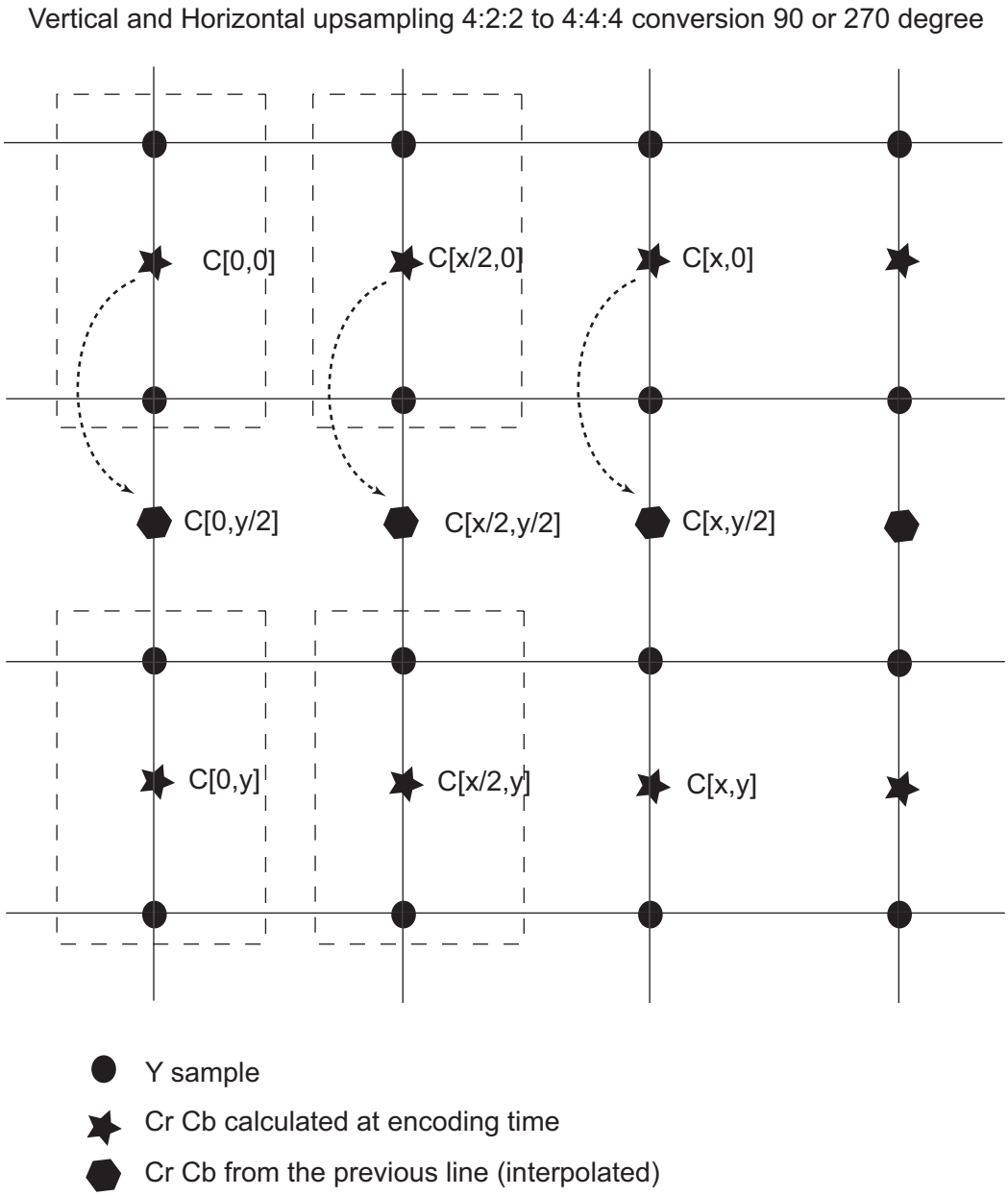


Figure 31-4. 4:2:2 Semiplanar and Planar Upsampling Algorithm - 90 or 270 Degree R Rotation Activated

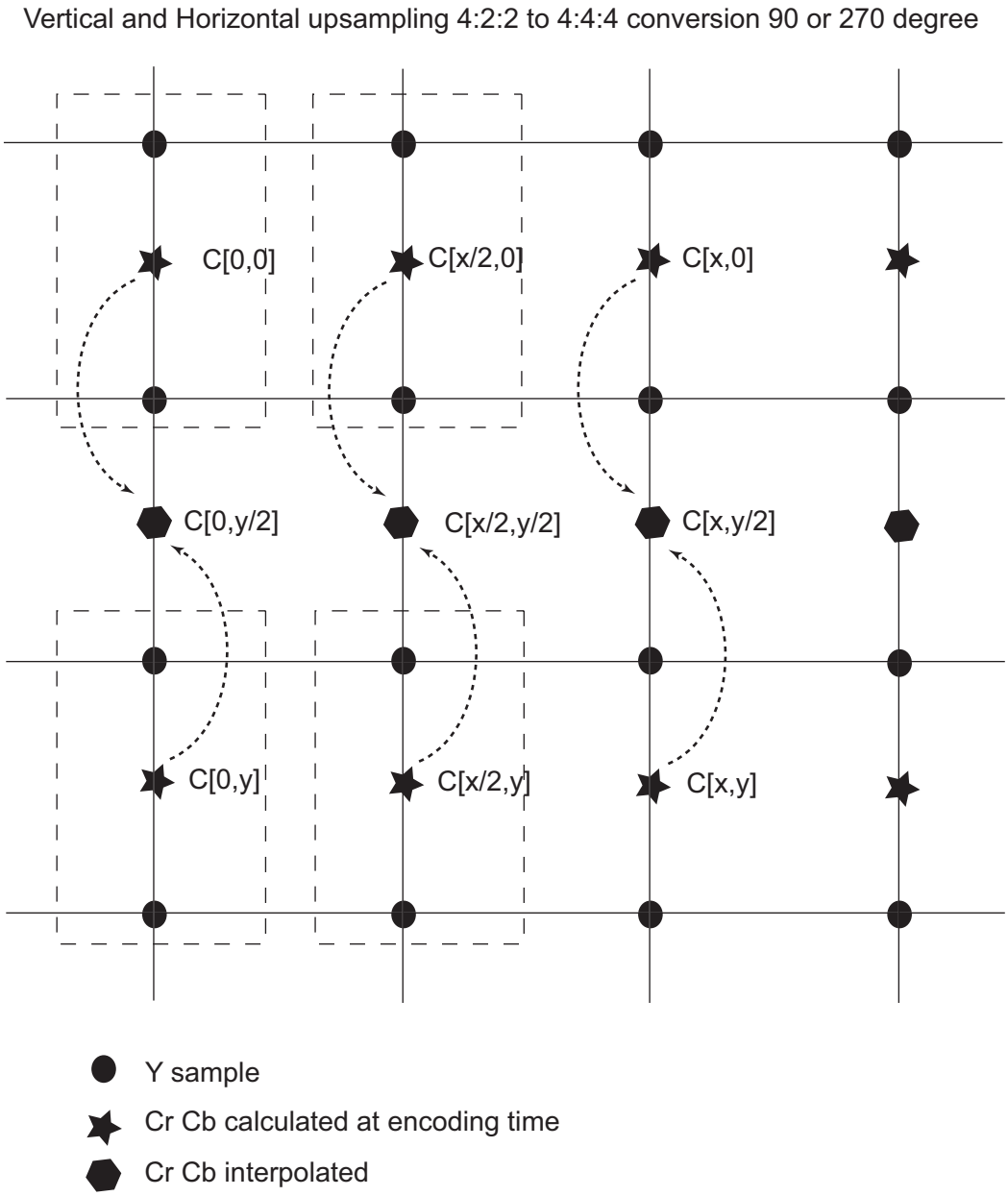
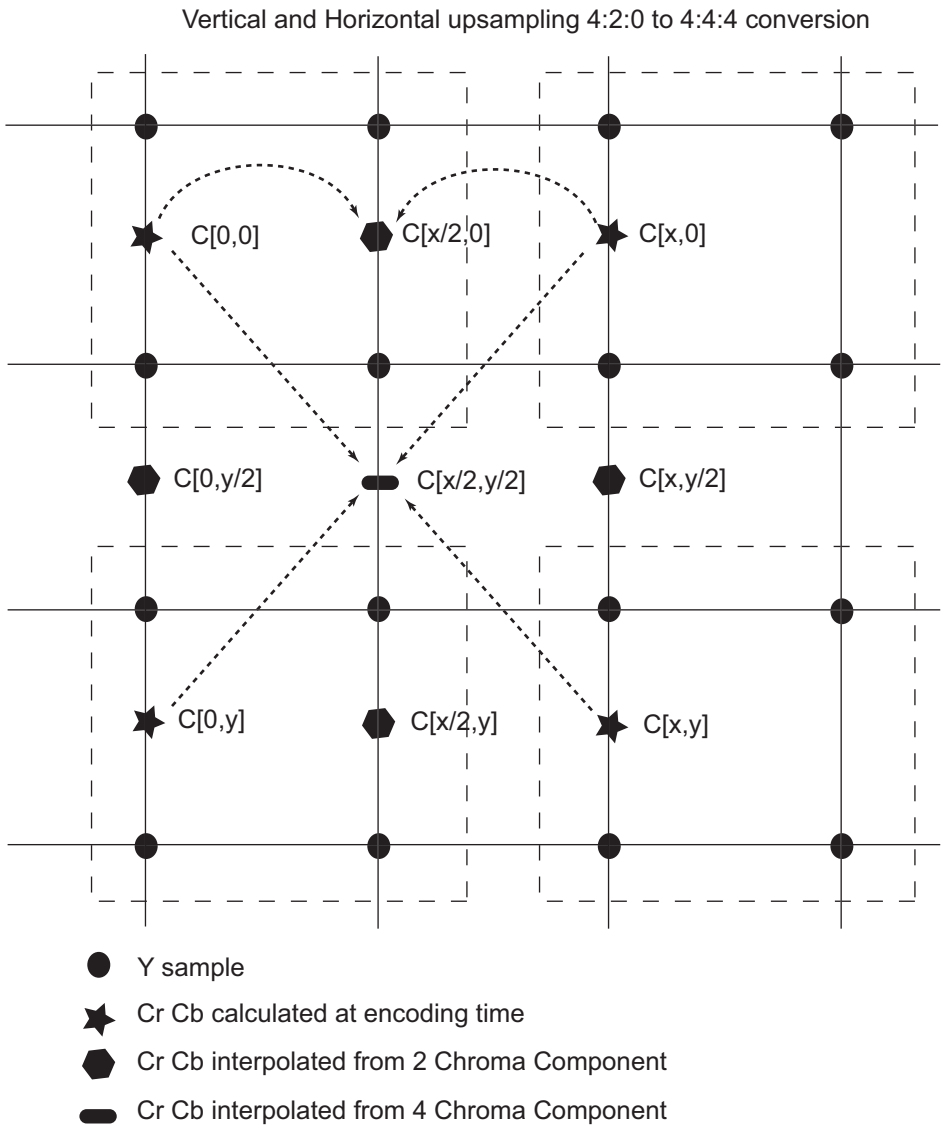


Figure 31-5. 4:2:0 Upsampling Algorithm



$$Chroma\left[\frac{x}{2}, 0\right] = \frac{Cr[0,0] + Cr[0,x]}{2}$$

$$Chroma\left[0, \frac{y}{2}\right] = \frac{Cr[0,0] + C[0,y]}{2}$$

$$Chroma\left[\frac{x}{2}, \frac{y}{2}\right] = \frac{Cr[0,0] + Cr[x,0] + Cr[y,0] + Cr[x,y]}{4}$$

$$Chroma\left[x, \frac{y}{2}\right] = \frac{Cr[x,0] + Cr[x,y]}{2}$$

$$Chroma\left[\frac{x}{2}, y\right] = \frac{Cr[0,y] + Cr[x,y]}{2}$$

31.6.6.1 Chrominance Upsampling Algorithm

1. Read line n from chrominance cache and interpolate [x/2,0] chrominance component filling the 1 x 2 kernel with line n. If the chrominance cache is empty, then fetch the first line from external memory and interpolate from the external memory. Duplicate the last chrominance at the end of line.
2. Fetch line n+1 from external memory, write line n + 1 to chrominance cache, read line n from the chrominance cache. interpolate [0,y/2], [x/2,y/2] and [x, y/2] filling the 2x2 kernel with line n and n+1. Duplicate the last chrominance line to generate the last interpolated line.
3. Repeat step 1 and step 2.

31.6.7 Line and Pixel Striding

The LCD module includes a technique to increment the memory address by a programmable amount when the end of line has been reached. This offset is referred to as XSTRIDE and is defined on a per overlay basis. Additionally, the PSTRIDE field allows a programmable jump at the pixel level. Pixel stride is the value from one pixel to the next.

31.6.7.1 Line Striding

When the end of line has been reached, the DMA address counter points to the next pixel address. The channel DMA address register is added to the XSTRIDE field, and then updated. If XSTRIDE is set to '0', the DMA address register remains unchanged. The XSTRIDE field of the channel configuration register is aligned to the pixel size boundary. The XSTRIDE field is a two's complement number. The following formula applies at the line boundary and indicates how the DMA controller computes the next pixel address. The function Sizeof() returns the number of bytes required to store a pixel.

$$NextPixelAddress = CurrentPixelAddress + Sizeof(pixel) + XSTRIDE$$

31.6.7.2 Pixel Striding

The DMA channel engine may optionally fetch non-contiguous pixels. The channel DMA address register is added to the PSTRIDE field and then updated. If PSTRIDE is set to zero, the DMA address register remains unchanged and pixels are contiguous. The PSTRIDE field of the channel configuration register is aligned to the pixel size boundary. The PSTRIDE is a two's complement number. The following formula applies at the pixel boundary and indicates how the DMA controller computes the next pixel address. The function Sizeof() returns the number of bytes required to store a pixel.

$$NextPixelAddress = CurrentPixelAddress + Sizeof(pixel) + PSTRIDE$$

31.6.8 Color Space Conversion Unit

The color space conversion unit converts Luminance Chrominance color space into the Red Green Blue color space. The conversion matrix is defined below and is fully programmable through the LCD user interface.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} CSCRY & CSCRU & CSCRV \\ CSCGY & CSCGU & CSCGV \\ CSCBY & CSCBU & CSCBV \end{bmatrix} \bullet \begin{bmatrix} Y - Yoff \\ Cb - Cboff \\ Cr - Croff \end{bmatrix}$$

Color space conversion coefficients are defined with the following equation:

$$CSC_{ij} = \frac{1}{2^7} \cdot \left[-2^9 \cdot c_9 + \sum_{n=0}^{8} c_n \cdot 2^n \right]$$

Color space conversion coefficients are defined with one sign bit, 2 integer bits and 7 fractional bits. The range of the CSCij coefficients is defined below with a step of 1/128.

$$-4 \leq CSC_{ij} \leq 3.9921875$$

Additionally, a set scaling factor {Yoff, Cboff, Croff} can be applied.

31.6.9 Two Dimension Scaler

The High End Overlay (HEO) data path includes a hardware scaler that allows an image resize in both horizontal and vertical directions.

31.6.9.1 Video Scaler Description

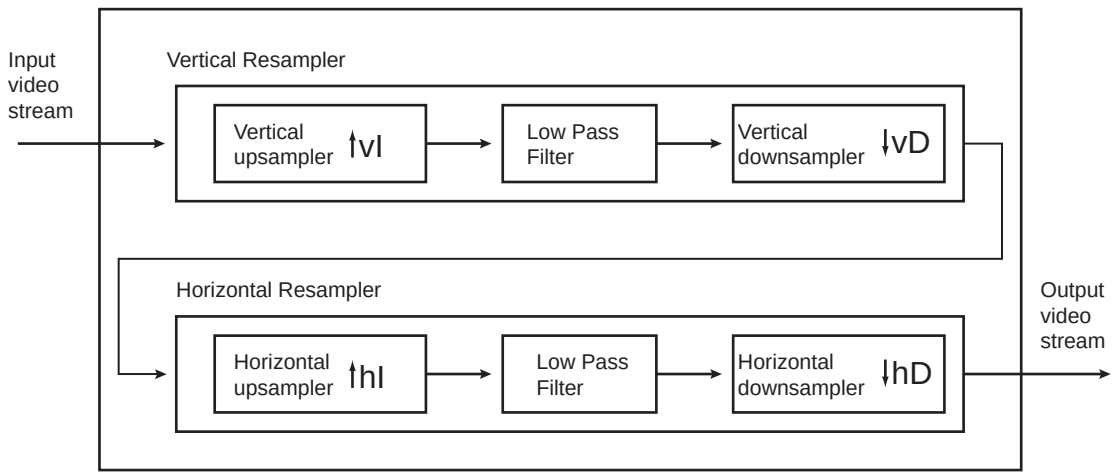
The scaling operation is based on a vertical and horizontal resampling algorithm. The sampling rate of the original image is increased when the video is upscaled, and decreased when the video is downscaled. A Vertical resampler is used to perform a vertical interpolation by a factor of *vI,* and a decimation by a factor of *vD*. A Horizontal resampler is used to perform a vertical interpolation by a factor of *hI,* and a decimation by a factor of *hD.* Both horizontal and vertical low pass filters are designed to minimize the aliasing effect. The frequency response of the low pass filter has the following characteristics:

$$H(\omega) = \begin{cases} I & \text{when } 0 \leq |\omega| \leq min(\frac{\pi}{I}, \frac{\pi}{D}) \\ 0 & \text{otherwise} \end{cases}$$

Taking into account the linear phase condition and anticipating the filter length M, the desired frequency response is modified.

$$H(\omega) = \begin{cases} Ie^{-j\omega\frac{M}{2}} & \text{when } 0 \leq |\omega| \leq min(\frac{\pi}{I}, \frac{\pi}{D}) \\ 0 & \text{otherwise} \end{cases}$$

Figure 31-6. Video Resampler Architecture



The impulse response of the low pass filter defined is:

$$h(n) = \begin{cases} I \times \frac{\omega_c}{\pi} \text{ when } n = 0 \\ I \times \frac{\omega_c}{\pi} \times \frac{\sin(\omega_c n)}{\omega_c n} \text{otherwise} \end{cases}$$

Or, for the filter of length M:

$$h(n) = \begin{cases} I \times \frac{\omega_c}{\pi} \text{ when } n = \frac{M}{2} \\ I \times \frac{\omega_c}{\pi} \times \frac{\sin\left(\omega_c\left(n - \frac{M}{2}\right)\right)}{\omega_c\left(n - \frac{M}{2}\right)} \text{otherwise} \end{cases}$$

This ideal filter is non-causal and cannot be realized. The unit sample response *h(n)* is infinite in duration and must be truncated depending on the expected length *M* of the filter. This truncation is equivalent to the multiplication of the impulse response by a window function *w(n)*.

Table 31-48. Window Function for a Filter Length M

Name of Window Function	Time Domain Sequence *w(n)*
Barlett	$1 - \frac{2 \times \left\lvert n - \frac{M-1}{2} \right\rvert}{M-1}$
Blackman	$0.42 - 0.5 \times \cos\frac{2\pi n}{M-1} + 0.08 \times \cos\frac{4\pi n}{M-1}$
Hamming	$0.54 - 0.46 \times \cos\frac{2\pi n}{M-1}$
Hanning	$0.5 - 0.5 \times \cos\frac{2\pi n}{M-1}$

The horizontal resampler includes an 8-phase 5-tap filter equivalent to a 40-tap FIR described in Figure 31-7.

The vertical resampler includes an 8-phase 3-tap filter equivalent to a 24-tap FIR described in Figure 31-8.

Figure 31-7. Horizontal Resampler Filter Architecture

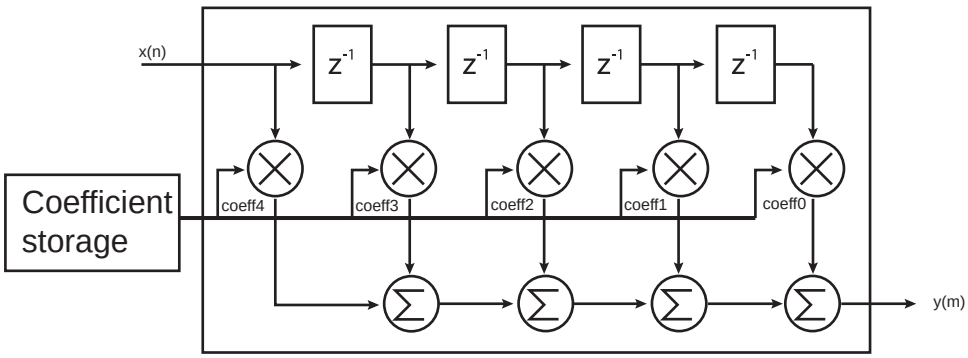
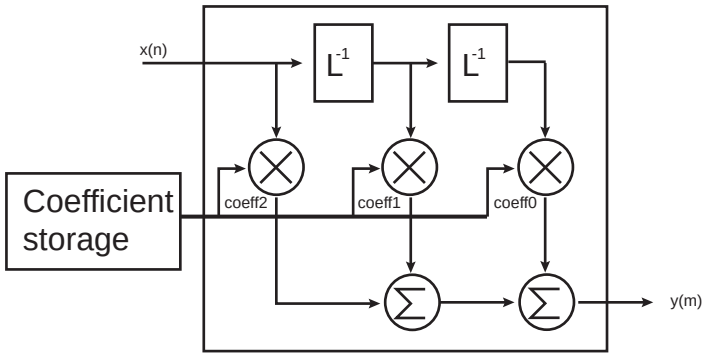


Figure 31-8. Vertical Resampler Filter Architecture



31.6.9.2 Horizontal Scaler

The XMEMSIZE field of the LCDC_HEOCFG4 register indicates the horizontal size minus one of the image in the system memory. The XSIZE field of the LCDC_HEOCFG3 register contains the horizontal size minus one of the window. The SCALEN bit of the LCDC_HEOCFG13 register is set to '1'. The scaling factor is programmed in the XFACTOR field of the LCDC_HEOCFG13 register. Use the following algorithm to find the XFACTOR value:

$$XFACTOR_{1st} = floor\left(\frac{8 \times 256 \times XMEMSIZE - 256 \times XPHIDEF}{XSIZE}\right)$$

$$XFACTOR_{1st} = XFACTOR_{1st} + 1$$

$$XMEMSIZE_{max} = floor\left(\frac{XFACTOR_{1st} \times XSIZE + 256 \times XPHIDEF}{2048}\right)$$

$$\begin{cases} XFACTOR = XFACTOR_{1st} - 1 & when(XMEMSIZE_{max} > XMEMSIZE) \\ XFACTOR = XFACTOR_{1st} & otherwise \end{cases}$$

31.6.9.3 Vertical Scaler

The YMEMSIZE field of the LCDC_HEOCFG4 register indicates the vertical size minus one of the image in the system memory. The YSIZE field of the LCDC_HEOCFG3 register contains the vertical size minus one of the window. The SCALEN bit of the LCDC_HEOCFG13 register is set to one. The scaling factor is programmed in the YFACTOR field of the LCDC_HEOCFG13 register.

$$YFACTOR_{1st} = floor\left(\frac{8 \times 256 \times YMEMSIZE - 256 \times YPHIDEF}{YSIZE}\right)$$

$$YFACTOR_{1st} = YFACTOR_{1st} + 1$$

$$YMEMSIZE_{max} = floor\left(\frac{YFACTOR_{1st} \times YSIZE + 256 \times YPHIDEF}{2048}\right)$$

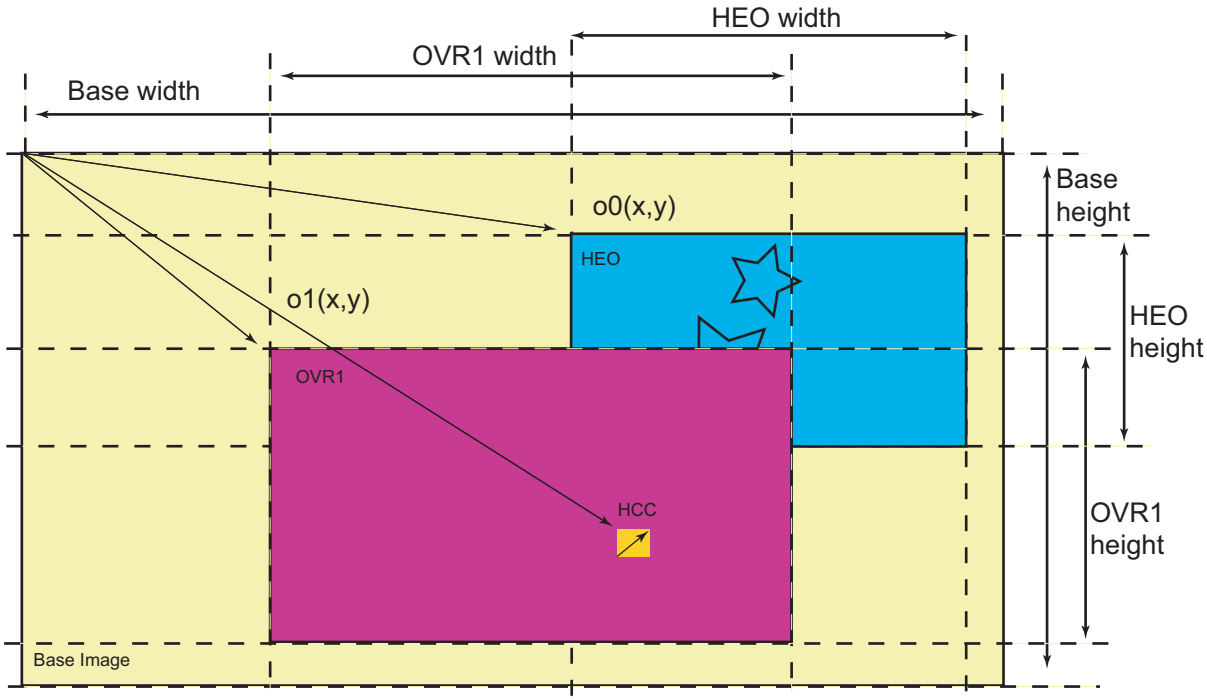
$$\begin{cases} YFACTOR = YFACTOR_{1st} - 1 & when(YMEMSIZE_{max} > YMEMSIZE) \\ YFACTOR = YFACTOR_{1st} & otherwise \end{cases}$$

31.6.10 Color Combine Unit

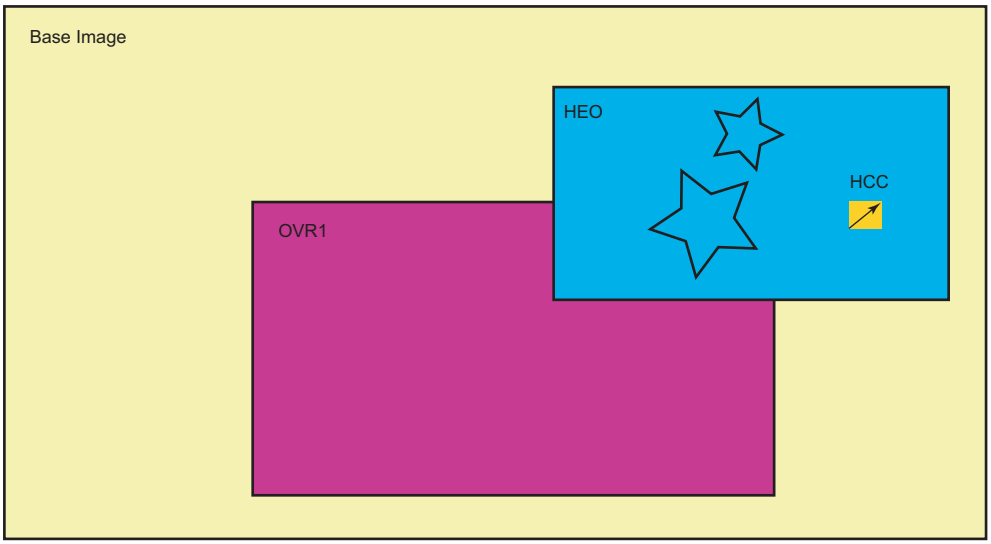
31.6.10.1 Window Overlay

The LCD module provides hardware support for multiple “overlay plane” that can be used to display windows on top of the image without destroying the image located below. The overlay image can use any color depth. Using the overlay alleviates the need to re-render the occluded portion of the image. When pixels are combined together through the alpha blending unit, a new color is created. This new pixel is called an iterated pixel and is passed to the next blending stage. Then, this pixel may be combined again with another pixel. The VIDPRI bit located in the LCDC_HEOCFG12 register configures the video priority algorithm used to display the layers. When the VIDPRI bit is written to ‘0’, the OVR1 layer is located above the HEO layer. When the VIDPRI bit is written to ‘1’, OVR1 is located below the HEO layer.

Figure 31-9. Overlay Example with Two Different Video Prioritization Algorithms



Video Prioritization Algorithm 1 : HCC > OVR1 > HEO > BASE



Video Prioritization Algorithm 2 : HCC > HEO > OVR1 > BASE

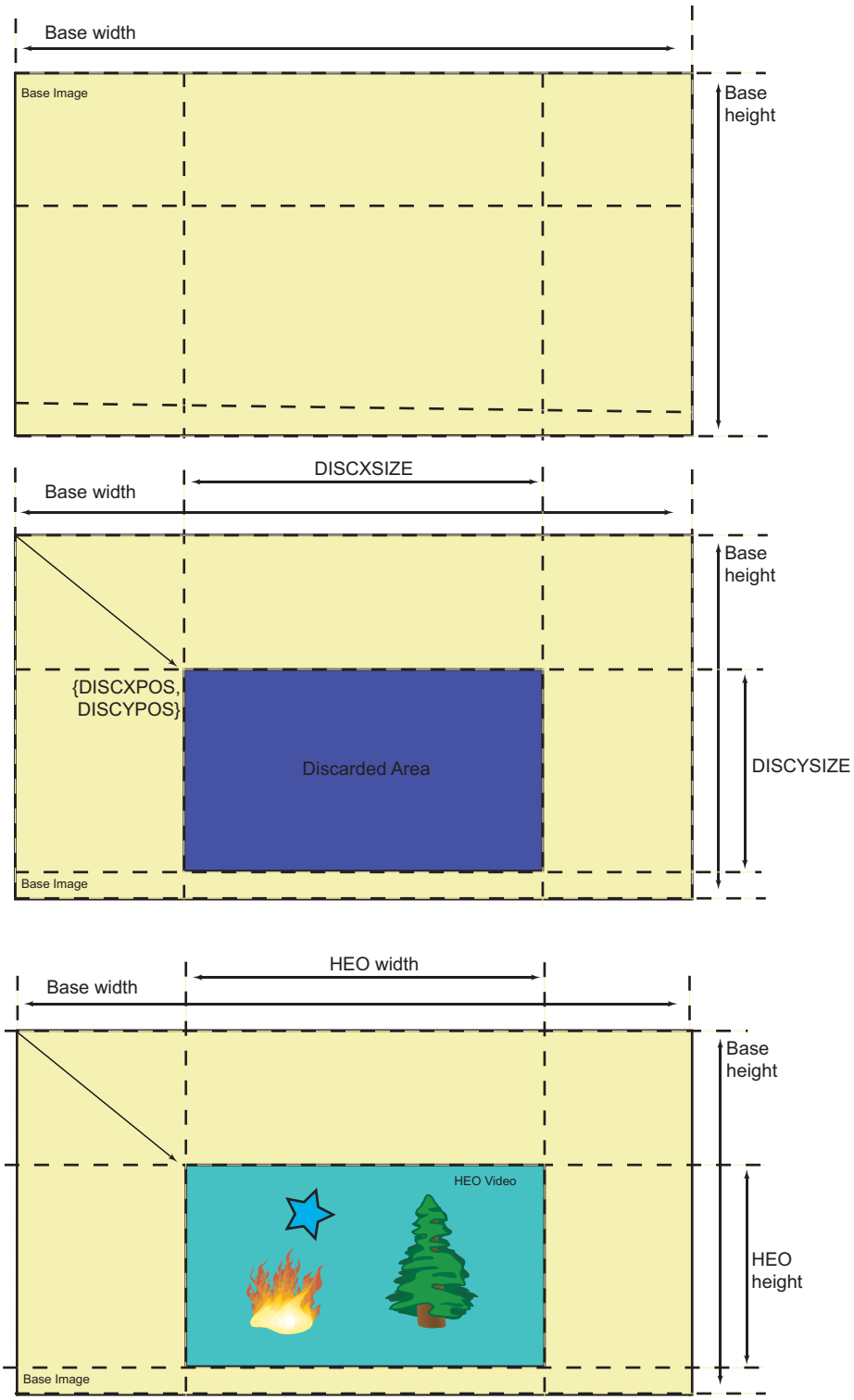
31.6.10.2 Base Layer with Window Overlay Optimization

When the base layer is combined with at least one active overlay, the whole base layer frame is retrieved from the memory though it is not visible. A set of registers is used to disable the Base DMA when this condition is met. These registers are the following:

- LCDC_BASECFG5:
  - field DISCXPOS (Discard Area Horizontal Position)
  - field DISCYPOS (Discard Area Vertical Position)

- LCDC_BASECFG6:
  - field DISCXSIZE (Discard Area Horizontal Size)
  - field DISCYSIZE (Discard Area Vertical Size)
- LCDC_BASECFG4: bit DISCEN (Discard Area Enable)

Figure 31-10. Base Layer Discard Area



31.6.10.3 Overlay Blending

The blending function requires two pixels (one iterated from the previous blending stage and one from the current overlay color) and a set of blending configuration parameters. These parameters define the color operation.

Figure 31-11. Alpha Blender Function

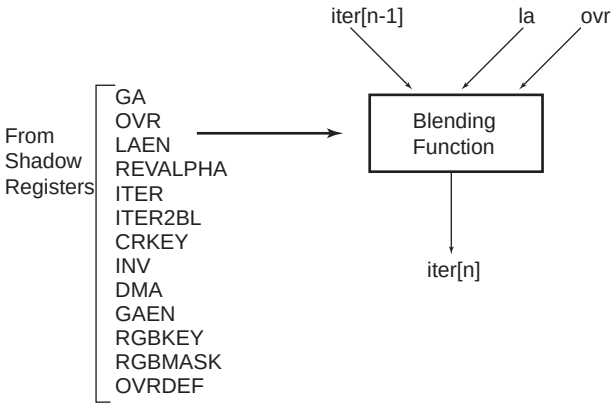
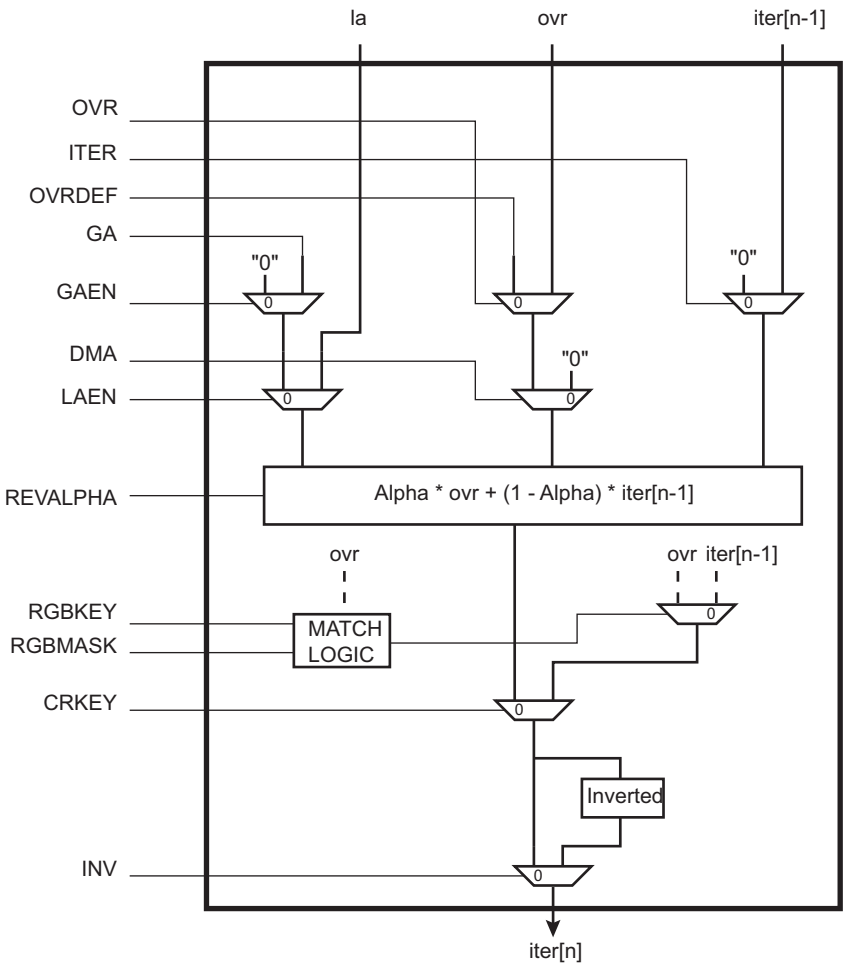
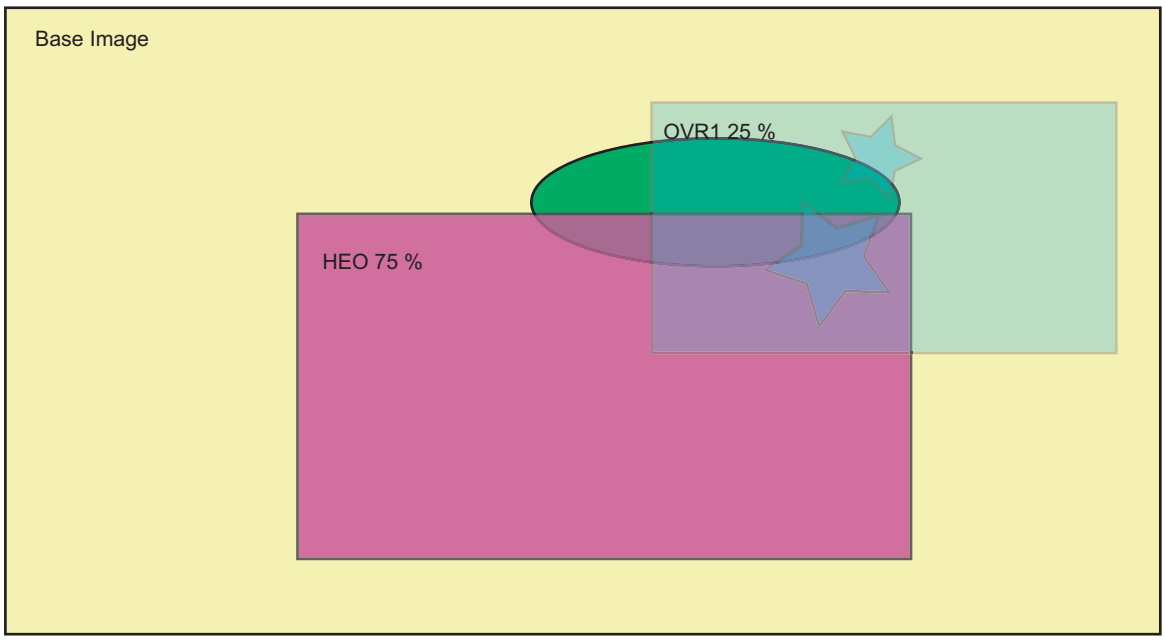


Figure 31-12. Alpha Blender Database



31.6.10.4 Window Blending

Figure 31-13. 256-level Alpha Blending



Video Prioritization Algorithm 1: OVR1 > HEO > BASE

31.6.10.5 Color Keying

Color keying involves a method of bit-block image transfer (Blit). This entails blitting one image onto another where not all the pixels are copied. Blitting usually involves two bitmaps: a source bitmap and a destination bitmap. A raster operation (ROP) is performed to define whether the iterated color or the overlay color is to be visible or not.

Source Color Keying

If the masked overlay color matches the color key, the iterated color is selected and Source Color Keying is activated using the following configuration sequence:

1. Select the overlay to blit.
2. Write a ‘0’ to DSTKEY.
3. Activate Color Keying by writing a ‘1’ to CRKEY.
4. Configure the Color Key by writing RKEY, GKEY and BKEY fields.
5. Configure the Color Mask by writing RKEY, GKEY and BKEY fields.

When the field RMASK, GMASK, or BMASK is configured to ‘0’, the comparison is disabled and the raster operation is activated.

Destination Color Keying

If the iterated masked color matches the color key then the overlay color is selected, Destination Color Keying is activated using the following configuration sequence:

1. Select the overlay to blit.
2. Write a ‘1’ to DSTKEY.
3. Activate Color Keying by writing a ‘1’ to CRKEY bit.
4. Configure the Color Key by writing RKEY, GKEY and BKEY fields.
5. Configure the Color Mask by writing RKEY, GKEY and BKEY fields.

When the field RMASK, GMASK, or BMASK is configured to ‘0’, the comparison is disabled and the raster operation is activated.

31.6.11 LCDC PWM Controller

This block generates the LCD contrast control signal (LCD_PWM) to make possible the control of the display's contrast by software. This is an 8-bit PWM (Pulse Width Modulation) signal that can be converted to an analog voltage with a simple passive filter.

The PWM module has a free-running counter whose value is compared against a compare register (PWMCVAL field of the LCDC_LCDCFG6 register). If the value in the counter is less than that in the register, the output brings the value of the signal polarity (PWMPOL) bit in the PWM control register: LCDC_LCDCFG6. Otherwise, the opposite value is output. Thus, a periodic waveform with a pulse width proportional to the value in the compare register is generated.

Due to the comparison mechanism, the output pulse has a width between zero and 255 PWM counter cycles. Thus by adding a simple passive filter outside the chip, an analog voltage between 0 and (255/256) × V_{DD} can be obtained (for the positive polarity case, or between (1/256) × V_{DD} and V_{DD} for the negative polarity case). Other voltage values can be obtained by adding active external circuitry.

For PWM mode, the counter frequency can be adjusted to four different values using the PWMPS field of the LCDC_LCDCFG6 register.

The PWM module can be fed with the slow clock or the system clock, depending on the CLKPWMSEL bit of the LCDC_CFG0 register.

31.6.12 LCD Overall Performance

31.6.12.1 Color Lookup Table (CLUT)

Table 31-49. CLUT Pixel Performance

CLUT Mode	Pixels/Cycle	Rotation	Scaling
1 bpp	64	Not supported	Supported
2 bpp	32	Not supported	Supported
3 bpp	16	Not supported	Supported
4 bpp	8	Not supported	Supported

31.6.12.2 RGB Mode Fetch Performance

Table 31-50. RGB Mode Performance

RGB Mode	Pixels/Cycle Memory Burst Mode	Rotation Peak Random Memory Access (pixels/cycle)		Scaling Burst Mode or Rotation Optimization Available
		Rotation Optimization (1)	Normal Mode	
12 bpp	4	1	0.2	Supported
16 bpp	4	1	0.2	Supported
18 bpp	2	1	0.2	Supported
18 bpp RGB PACKED	2.666	Not supported	0.2	Supported
19 bpp	2	1	0.2	Supported
19 bpp PACKED	2.666	Not Supported	0.2	Supported
24 bpp	2	1	0.2	Supported
24 bpp PACKED	2.666	Not Supported	0.2	Supported
25 bpp	2	1	0.2	Supported
32 bpp	2	1	0.2	Supported

Note: 1. Rotation optimization = AHB lock asserted on consecutive single access.

31.6.12.3 YUV Mode Fetch Performance

Table 31-51. Single Stream for 0 Wait State Memory

YUV Mode	Pixels/Cycle Memory Burst Mode	Rotation Peak Random Memory Access (pixels/cycle)		Scaling Burst Mode or Rotation Optimization Is Available
		Rotation Optimization (1)	Normal Mode	
32 bpp AYUV	2	1	0.2	Supported
16 bpp 422	4	Not Supported	Not Supported	Supported

Note: 1. Rotation optimization = AHB lock asserted on consecutive single access

Table 31-52. Multiple Stream for 0 Wait State Memory

YUV Mode	Comp/Cycle Memory Burst Mode	Rotation Peak Random Memory Access (pixels/cycle)		Scaling Burst Mode or Rotation Optimization Is Available
		Rotation Optimization	Normal Mode	
16 bpp 422 semiplanar	8 Y, 4 UV	1 Y, 1 UV (2 streams)	0.2 Y 0.2 UV (2 streams)	Supported
16 bpp 422 planar	8 Y, 8 U, 8 V	1 Y, 1 U, 1 V (3 streams)	0.2 Y, 0.2 U, 0.2 V (3 streams)	Supported
12 bpp 4:2:0 semiplanar	8 Y, 4 UV	1 Y, 1 UV (2 streams)	0.2 Y 0.2 UV (2 streams)	Supported
12 bpp 4:2:0 planar	8 Y, 8 U, 8 V	1 Y, 1 U, 1 V (3 streams)	0.2 Y, 0.2 U, 0.2 V (3 streams)	Supported

Note: In order to provide more bandwidth, when multiple streams are used to transfer Y, UV, U or V components, two AHB interfaces are recommended or multiple AXI ID are required.

Table 31-53. YUV Planar Overall Performance 1 AHB Interface for 0 Wait State Memory

YUV Mode	Pix/Cycle Memory Burst Mode	Rotation Peak Random Memory Access (pixels/cycle)		Scaling Burst Mode or Rotation Optimization Is Available
		Rotation Optimization	Normal Mode	
16 bpp 422 semiplanar	4	0.66	0.132	Supported
16 bpp 422 planar	4	0.5	0.1	Supported
12 bpp 4:2:0 semiplanar	5.32	0.8	0.16	Supported
12 bpp 4:2:0 planar	5.32	0.66	0.132	Supported

Note: In order to provide more bandwidth, when multiple streams are used to transfer Y, UV, U or V components, two AHB interfaces are recommended or multiple AXI ID are required.

31.6.13 Input FIFO

The LCD module includes one input FIFO per overlay. These input FIFOs are used to buffer the AHB burst and serialize the stream of pixels.

31.6.14 Output FIFO

The LCD module includes one output FIFO that stores the blended pixel.

31.6.15 Output Timing Generation

31.6.15.1 Active Display Timing Mode

Figure 31-14. Active Display Timing

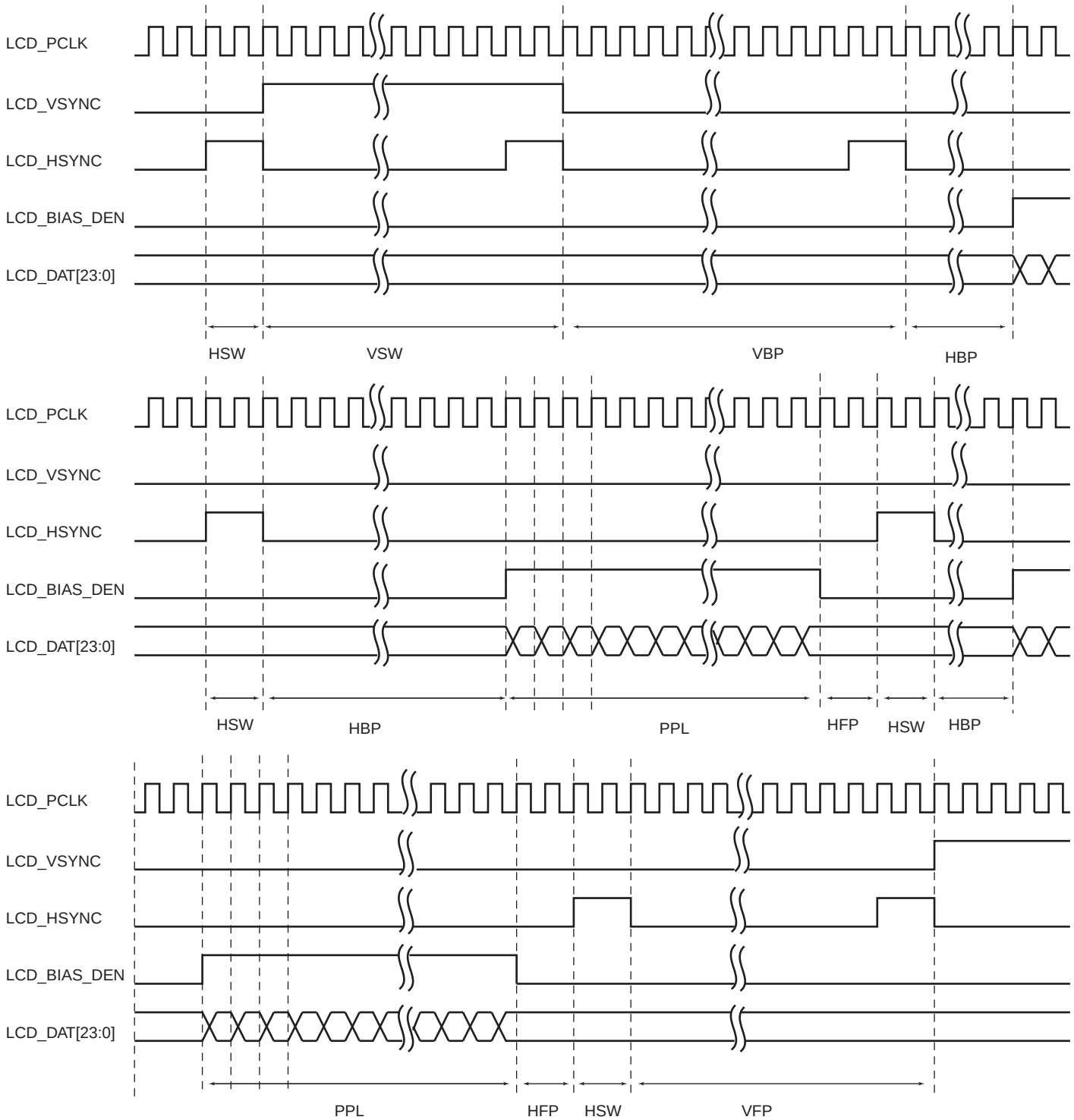


Figure 31-15. Vertical Synchronization Timing (part 1)

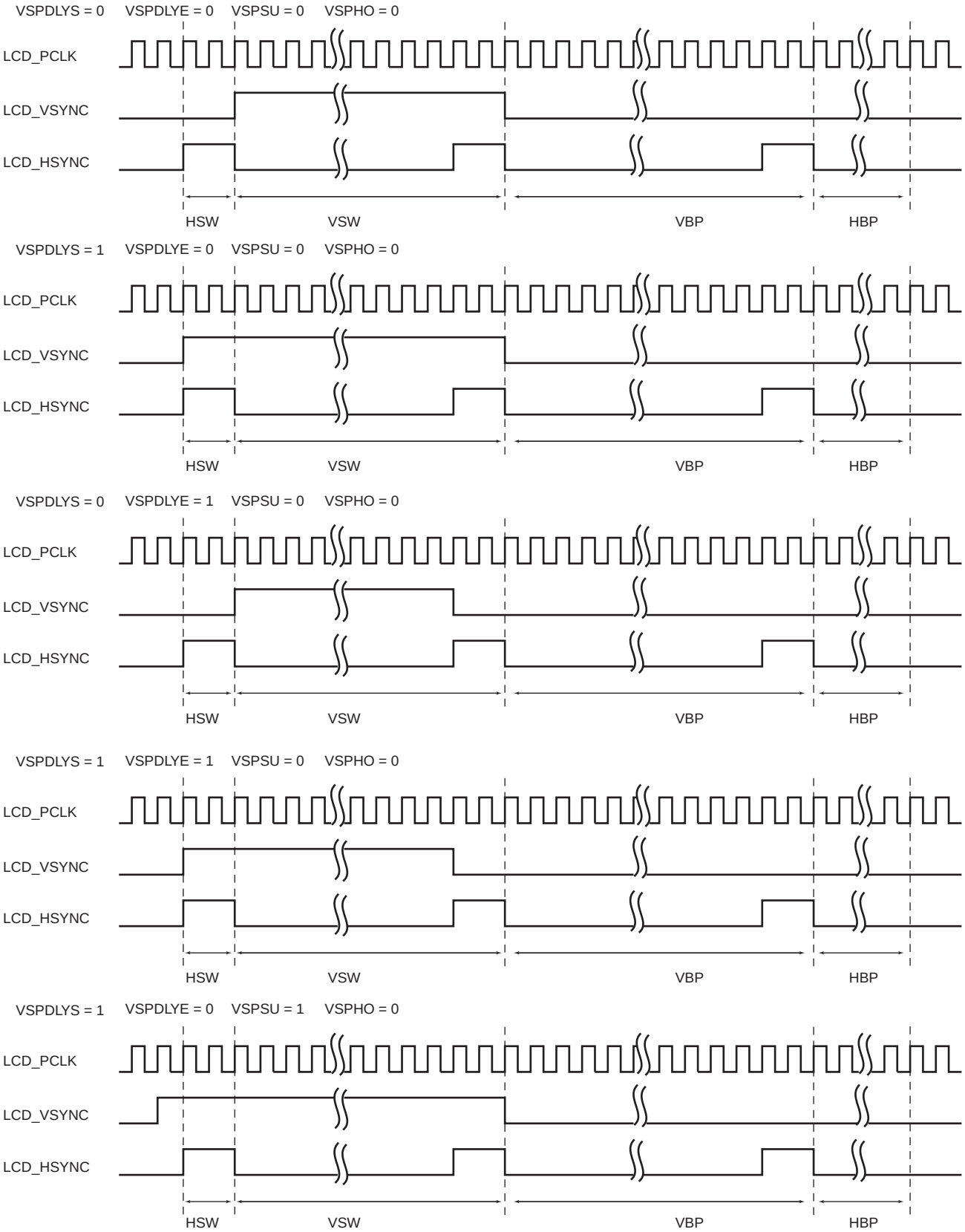


Figure 31-16. Vertical Synchronization Timing (part 2)

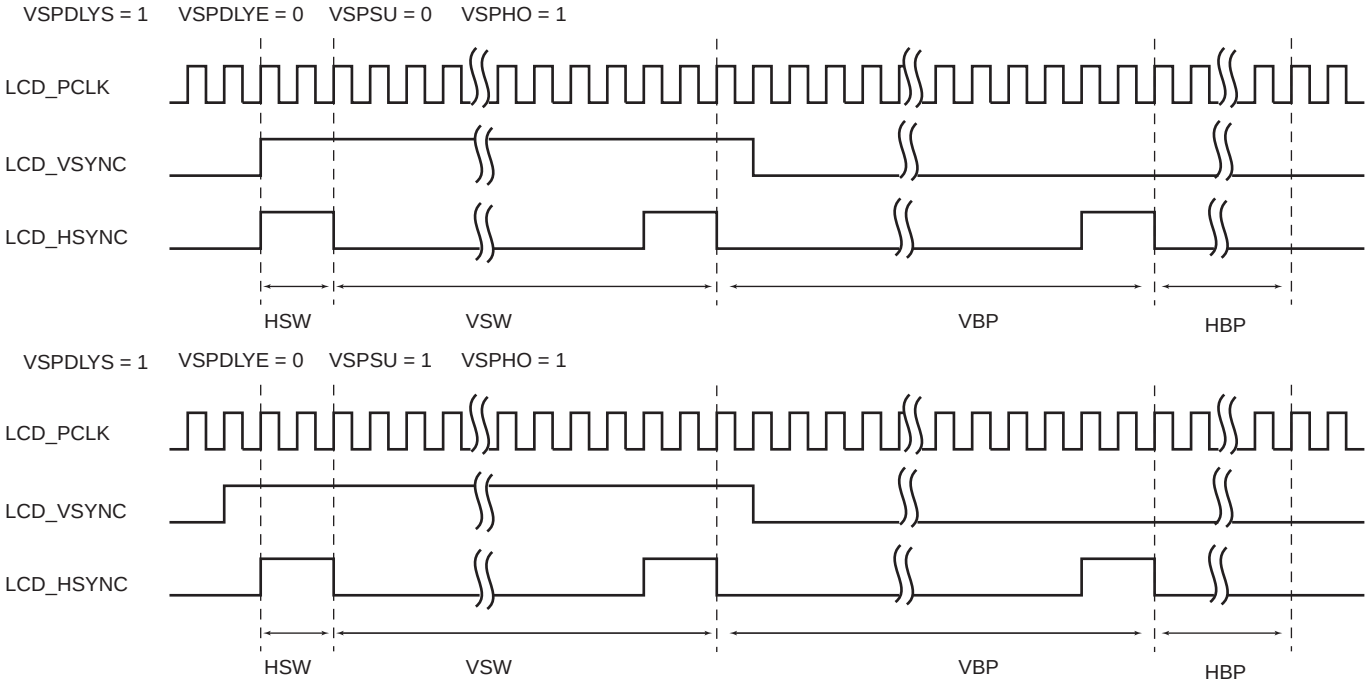
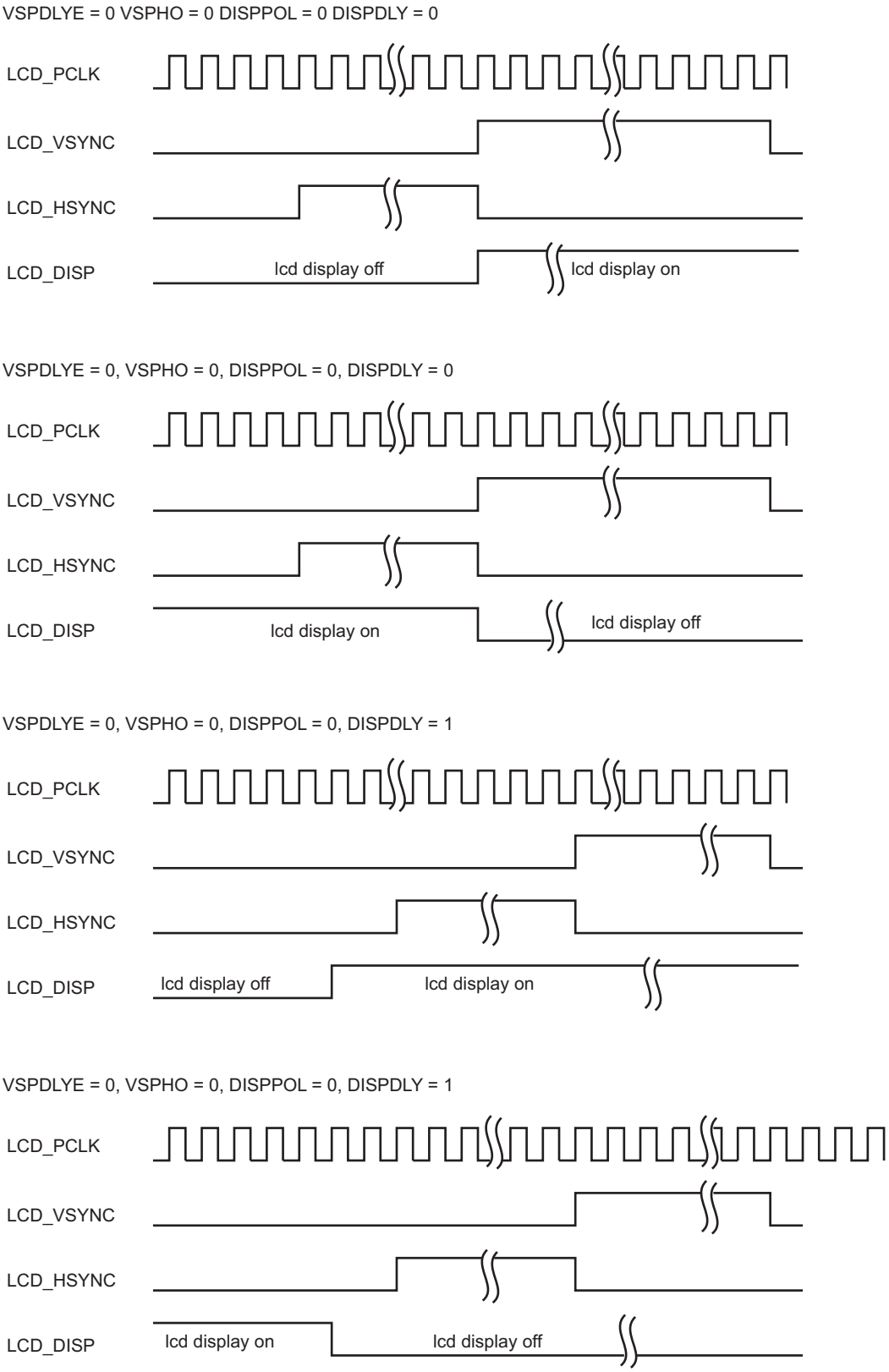


Figure 31-17. DISP Signal Timing Diagram



31.6.16 Output Format

31.6.16.1 Active Mode Output Pin Assignment

Table 31-54. Active Mode Output with 24-bit Bus Interface Configuration

Pin ID	TFT 24 bits	TFT 18 bits	TFT 16 bits	TFT 12 bits
LCD_DAT[23]	R[7]	R[5]	R[4]	R[3]
LCD_DAT[22]	R[6]	R[4]	R[3]	R[2]
LCD_DAT[21]	R[5]	R[3]	R[2]	R[1]
LCD_DAT[20]	R[4]	R[2]	R[1]	R[0]
LCD_DAT[19]	R[3]	R[1]	R[0]	–
LCD_DAT[18]	R[2]	R[0]	–	–
LCD_DAT[17]	R[1]	–	–	–
LCD_DAT[16]	R[0]	–	–	–
LCD_DAT[15]	G[7]	G[5]	G[5]	G[3]
LCD_DAT[14]	G[6]	G[4]	G[4]	G[2]
LCD_DAT[13]	G[5]	G[3]	G[3]	G[1]
LCD_DAT[12]	G[4]	G[2]	G[2]	G[0]
LCD_DAT[11]	G[3]	G[1]	G[1]	–
LCD_DAT[10]	G[2]	G[0]	G[0]	–
LCD_DAT[9]	G[1]	–	–	–
LCD_DAT[8]	G[0]	–	–	–
LCD_DAT[7]	B[7]	B[5]	B[4]	B[3]
LCD_DAT[6]	B[6]	B[4]	B[3]	B[2]
LCD_DAT[5]	B[5]	B[3]	B[2]	B[1]
LCD_DAT[4]	B[4]	B[2]	B[1]	B[0]
LCD_DAT[3]	B[3]	B[1]	B[0]	–
LCD_DAT[2]	B[2]	B[0]	–	–
LCD_DAT[1]	B[1]	–	–	–
LCD_DAT[0]	B[0]	–	–	–

31.7 LCD Controller (LCDC) User Interface

Table 31-55. Register Mapping

Offset	Register	Name	Access	Reset
0x00000000	LCD Controller Configuration Register 0	LCDC_LCDCFG0	Read/Write	0x00000000
0x00000004	LCD Controller Configuration Register 1	LCDC_LCDCFG1	Read/Write	0x00000000
0x00000008	LCD Controller Configuration Register 2	LCDC_LCDCFG2	Read/Write	0x00000000
0x0000000C	LCD Controller Configuration Register 3	LCDC_LCDCFG3	Read/Write	0x00000000
0x00000010	LCD Controller Configuration Register 4	LCDC_LCDCFG4	Read/Write	0x00000000
0x00000014	LCD Controller Configuration Register 5	LCDC_LCDCFG5	Read/Write	0x00000000
0x00000018	LCD Controller Configuration Register 6	LCDC_LCDCFG6	Read/Write	0x00000000
0x0000001C	Reserved	–	–	–
0x00000020	LCD Controller Enable Register	LCDC_LCDEN	Write-only	–
0x00000024	LCD Controller Disable Register	LCDC_LCDDIS	Write-only	–
0x00000028	LCD Controller Status Register	LCDC_LCDSR	Read-only	0x00000000
0x0000002C	LCD Controller Interrupt Enable Register	LCDC_LCDIER	Write-only	–
0x00000030	LCD Controller Interrupt Disable Register	LCDC_LCDIDR	Write-only	–
0x00000034	LCD Controller Interrupt Mask Register	LCDC_LCDIMR	Read-only	0x00000000
0x00000038	LCD Controller Interrupt Status Register	LCDC_LCDISR	Read-only	0x00000000
0x0000003C	LCD Controller Attribute Register	LCDC_ATTR	Write-only	–
0x00000040	Base Layer Channel Enable Register	LCDC_BASECHER	Write-only	–
0x00000044	Base Layer Channel Disable Register	LCDC_BASECHDR	Write-only	–
0x00000048	Base Layer Channel Status Register	LCDC_BASECHSR	Read-only	0x00000000
0x0000004C	Base Layer Interrupt Enable Register	LCDC_BASEIER	Write-only	–
0x00000050	Base Layer Interrupt Disabled Register	LCDC_BASEIDR	Write-only	–
0x00000054	Base Layer Interrupt Mask Register	LCDC_BASEIMR	Read-only	0x00000000
0x00000058	Base Layer Interrupt Status Register	LCDC_BASEISR	Read-only	0x00000000
0x0000005C	Base DMA Head Register	LCDC_BASEHEAD	Read/Write	0x00000000
0x00000060	Base DMA Address Register	LCDC_BASEADDR	Read/Write	0x00000000
0x00000064	Base DMA Control Register	LCDC_BASECTRL	Read/Write	0x00000000
0x00000068	Base DMA Next Register	LCDC_BASENEXT	Read/Write	0x00000000
0x0000006C	Base Layer Configuration Register 0	LCDC_BASECFG0	Read/Write	0x00000000
0x00000070	Base Layer Configuration Register 1	LCDC_BASECFG1	Read/Write	0x00000000
0x00000074	Base Layer Configuration Register 2	LCDC_BASECFG2	Read/Write	0x00000000
0x00000078	Base Layer Configuration Register 3	LCDC_BASECFG3	Read/Write	0x00000000
0x0000007C	Base Layer Configuration Register 4	LCDC_BASECFG4	Read/Write	0x00000000
0x00000080	Base Layer Configuration Register 5	LCDC_BASECFG5	Read/Write	0x00000000
0x00000084	Base Layer Configuration Register 6	LCDC_BASECFG6	Read/Write	0x00000000
0x00000088–0x0000013C	Reserved	–	–	–

Table 31-55. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x00000140	Overlay 1 Channel Enable Register	LCDC_OVR1CHER	Write-only	–
0x00000144	Overlay 1 Channel Disable Register	LCDC_OVR1CHDR	Write-only	–
0x00000148	Overlay 1 Channel Status Register	LCDC_OVR1CHSR	Read-only	0x00000000
0x0000014C	Overlay 1 Interrupt Enable Register	LCDC_OVR1IER	Write-only	–
0x00000150	Overlay 1 Interrupt Disable Register	LCDC_OVR1IDR	Write-only	–
0x00000154	Overlay 1 Interrupt Mask Register	LCDC_OVR1IMR	Read-only	0x00000000
0x00000158	Overlay 1 Interrupt Status Register	LCDC_OVR1ISR	Read-only	0x00000000
0x0000015C	Overlay 1 DMA Head Register	LCDC_OVR1HEAD	Read/Write	0x00000000
0x00000160	Overlay 1 DMA Address Register	LCDC_OVR1ADDR	Read/Write	0x00000000
0x00000164	Overlay 1 DMA Control Register	LCDC_OVR1CTRL	Read/Write	0x00000000
0x00000168	Overlay 1 DMA Next Register	LCDC_OVR1NEXT	Read/Write	0x00000000
0x0000016C	Overlay 1 Configuration Register 0	LCDC_OVR1CFG0	Read/Write	0x00000000
0x00000170	Overlay 1 Configuration Register 1	LCDC_OVR1CFG1	Read/Write	0x00000000
0x00000174	Overlay 1 Configuration Register 2	LCDC_OVR1CFG2	Read/Write	0x00000000
0x00000178	Overlay 1 Configuration Register 3	LCDC_OVR1CFG3	Read/Write	0x00000000
0x0000017C	Overlay 1 Configuration Register 4	LCDC_OVR1CFG4	Read/Write	0x00000000
0x00000180	Overlay 1 Configuration Register 5	LCDC_OVR1CFG5	Read/Write	0x00000000
0x00000184	Overlay 1 Configuration Register 6	LCDC_OVR1CFG6	Read/Write	0x00000000
0x00000188	Overlay 1 Configuration Register 7	LCDC_OVR1CFG7	Read/Write	0x00000000
0x0000018C	Overlay 1 Configuration Register 8	LCDC_OVR1CFG8	Read/Write	0x00000000
0x00000190	Overlay 1 Configuration Register 9	LCDC_OVR1CFG9	Read/Write	0x00000000
0x00000194–0x0000023C	Reserved	–	–	–
0x00000294–0x0000033C	Reserved	–	–	–
0x00000340	High End Overlay Channel Enable Register	LCDC_HEOCHER	Write-only	–
0x00000344	High End Overlay Channel Disable Register	LCDC_HEOCHDR	Write-only	–
0x00000348	High End Overlay Channel Status Register	LCDC_HEOCHSR	Read-only	0x00000000
0x0000034C	High End Overlay Interrupt Enable Register	LCDC_HEOIER	Write-only	–
0x00000350	High End Overlay Interrupt Disable Register	LCDC_HEOIDR	Write-only	–
0x00000354	High End Overlay Interrupt Mask Register	LCDC_HEOIMR	Read-only	0x00000000
0x00000358	High End Overlay Interrupt Status Register	LCDC_HEOISR	Read-only	0x00000000
0x0000035C	High End Overlay DMA Head Register	LCDC_HEOHEAD	Read/Write	0x00000000
0x00000360	High End Overlay DMA Address Register	LCDC_HEOADDR	Read/Write	0x00000000
0x00000364	High End Overlay DMA Control Register	LCDC_HEOCTRL	Read/Write	0x00000000
0x00000368	High End Overlay DMA Next Register	LCDC_HEONEXT	Read/Write	0x00000000
0x0000036C	High End Overlay U-UV DMA Head Register	LCDC_HEOUHEAD	Read/Write	0x00000000
0x00000370	High End Overlay U-UV DMA Address Register	LCDC_HEOUADDR	Read/Write	0x00000000

Table 31-55. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x00000374	High End Overlay U-UV DMA Control Register	LCDC_HEOUCTRL	Read/Write	0x00000000
0x00000378	High End Overlay U-UV DMA Next Register	LCDC_HEOUNEXT	Read/Write	0x00000000
0x0000037C	High End Overlay V DMA Head Register	LCDC_HEOVHEAD	Read/Write	0x00000000
0x00000380	High End Overlay V DMA Address Register	LCDC_HEOVADDR	Read/Write	0x00000000
0x00000384	High End Overlay V DMA Control Register	LCDC_HEOVCTRL	Read/Write	0x00000000
0x00000388	High End Overlay V DMA Next Register	LCDC_HEOVNEXT	Read/Write	0x00000000
0x0000038C	High End Overlay Configuration Register 0	LCDC_HEOCFG0	Read/Write	0x00000000
0x00000390	High End Overlay Configuration Register 1	LCDC_HEOCFG1	Read/Write	0x00000000
0x00000394	High End Overlay Configuration Register 2	LCDC_HEOCFG2	Read/Write	0x00000000
0x00000398	High End Overlay Configuration Register 3	LCDC_HEOCFG3	Read/Write	0x00000000
0x0000039C	High End Overlay Configuration Register 4	LCDC_HEOCFG4	Read/Write	0x00000000
0x000003A0	High End Overlay Configuration Register 5	LCDC_HEOCFG5	Read/Write	0x00000000
0x000003A4	High End Overlay Configuration Register 6	LCDC_HEOCFG6	Read/Write	0x00000000
0x000003A8	High End Overlay Configuration Register 7	LCDC_HEOCFG7	Read/Write	0x00000000
0x000003AC	High End Overlay Configuration Register 8	LCDC_HEOCFG8	Read/Write	0x00000000
0x000003B0	High End Overlay Configuration Register 9	LCDC_HEOCFG9	Read/Write	0x00000000
0x000003B4	High End Overlay Configuration Register 10	LCDC_HEOCFG10	Read/Write	0x00000000
0x000003B8	High End Overlay Configuration Register 11	LCDC_HEOCFG11	Read/Write	0x00000000
0x000003BC	High End Overlay Configuration Register 12	LCDC_HEOCFG12	Read/Write	0x00000000
0x000003C0	High End Overlay Configuration Register 13	LCDC_HEOCFG13	Read/Write	0x00000000
0x000003C4	High End Overlay Configuration Register 14	LCDC_HEOCFG14	Read/Write	0x00000000
0x000003C8	High End Overlay Configuration Register 15	LCDC_HEOCFG15	Read/Write	0x00000000
0x000003CC	High End Overlay Configuration Register 16	LCDC_HEOCFG16	Read/Write	0x00000000
0x000003D0	High End Overlay Configuration Register 17	LCDC_HEOCFG17	Read/Write	0x00000000
0x000003D4	High End Overlay Configuration Register 18	LCDC_HEOCFG18	Read/Write	0x00000000
0x000003D8	High End Overlay Configuration Register 19	LCDC_HEOCFG19	Read/Write	0x00000000
0x000003DC	High End Overlay Configuration Register 20	LCDC_HEOCFG20	Read/Write	0x00000000
0x000003E0	High End Overlay Configuration Register 21	LCDC_HEOCFG21	Read/Write	0x00000000
0x000003E4	High End Overlay Configuration Register 22	LCDC_HEOCFG22	Read/Write	0x00000000
0x000003E8	High End Overlay Configuration Register 23	LCDC_HEOCFG23	Read/Write	0x00000000
0x000003EC	High End Overlay Configuration Register 24	LCDC_HEOCFG24	Read/Write	0x00000000
0x000003F0	High End Overlay Configuration Register 25	LCDC_HEOCFG25	Read/Write	0x00000000
0x000003F4	High End Overlay Configuration Register 26	LCDC_HEOCFG26	Read/Write	0x00000000
0x000003F8	High End Overlay Configuration Register 27	LCDC_HEOCFG27	Read/Write	0x00000000
0x000003FC	High End Overlay Configuration Register 28	LCDC_HEOCFG28	Read/Write	0x00000000
0x00000400	High End Overlay Configuration Register 29	LCDC_HEOCFG29	Read/Write	0x00000000

Table 31-55. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x00000404	High End Overlay Configuration Register 30	LCDC_HEOCFG30	Read/Write	0x00000000
0x00000408	High End Overlay Configuration Register 31	LCDC_HEOCFG31	Read/Write	0x00000000
0x0000040C	High End Overlay Configuration Register 32	LCDC_HEOCFG32	Read/Write	0x00000000
0x00000410	High End Overlay Configuration Register 33	LCDC_HEOCFG33	Read/Write	0x00000000
0x00000414	High End Overlay Configuration Register 34	LCDC_HEOCFG34	Read/Write	0x00000000
0x00000418	High End Overlay Configuration Register 35	LCDC_HEOCFG35	Read/Write	0x00000000
0x0000041C	High End Overlay Configuration Register 36	LCDC_HEOCFG36	Read/Write	0x00000000
0x00000420	High End Overlay Configuration Register 37	LCDC_HEOCFG37	Read/Write	0x00000000
0x00000424	High End Overlay Configuration Register 38	LCDC_HEOCFG38	Read/Write	0x00000000
0x00000428	High End Overlay Configuration Register 39	LCDC_HEOCFG39	Read/Write	0x00000000
0x0000042C	High End Overlay Configuration Register 40	LCDC_HEOCFG40	Read/Write	0x00000000
0x00000430	High End Overlay Configuration Register 41	LCDC_HEOCFG41	Read/Write	0x00000000
0x00000434–0x0000053C	Reserved	–	–	–
0x00000584–0x000005FC	Reserved	–	–	–
0x00000600	Base CLUT Register 0	LCDC_BASECLUT0	Read/Write	0x00000000
...	...	...	...	...
0x000008FC	Base CLUT Register 255	LCDC_BASECLUT255	Read/Write	0x00000000
0x00000A00	Overlay 1 CLUT Register 0	LCDC_OVR1CLUT0	Read/Write	0x00000000
...	...	...	...	...
0x00000DFC	Overlay 1 CLUT Register 255	LCDC_OVR1CLUT255	Read/Write	0x00000000
0x00001200	High End Overlay CLUT Register 0	LCDC_HEOCLUT0	Read/Write	0x00000000
...	...	...	...	...
0x000015FC	High End Overlay CLUT Register 255	LCDC_HEOCLUT255	Read/Write	0x00000000
0x00001600–0x00001FFC	Reserved	–	–	–

Note: 1. The CLUT registers are located in embedded RAM.

31.7.1 LCD Controller Configuration Register 0

Name: LCDC_LCDCFG0

Address: 0xF0000000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CLKDIV							

15	14	13	12	11	10	9	8
–	–	–	–	CGDISHEO		CGDISOVR1	CGDISBASE

7	6	5	4	3	2	1	0
–	–	–	–	CLKPWMSEL	CLKSEL	–	CLKPOL

- **CLKPOL: LCD Controller Clock Polarity**

0: Data/Control signals are launched on the rising edge of the pixel clock.

1: Data/Control signals are launched on the falling edge of the pixel clock.

- **CLKSEL: LCD Controller Clock Source Selection**

0: The asynchronous output stage of the LCD controller is fed by the System Clock.

1: The asynchronous output state of the LCD controller is fed by the 2x System Clock.

- **CLKPWMSEL: LCD Controller PWM Clock Source Selection**

0: The slow clock is selected and feeds the PWM module.

1: The system clock is selected and feeds the PWM module.

- **CGDISBASE: Clock Gating Disable Control for the Base Layer**

0: Automatic Clock Gating is enabled for the Base Layer.

1: Clock is running continuously.

- **CGDISOVR1: Clock Gating Disable Control for the Overlay 1 Layer**

0: Automatic Clock Gating is enabled for the Overlay 1 Layer.

1: Clock is running continuously.

- **CGDISHEO: Clock Gating Disable Control for the High End Overlay**

0: Automatic Clock Gating is enabled for the High End Overlay Layer.

1: Clock is running continuously.

- **CLKDIV: LCD Controller Clock Divider**

8-bit width clock divider for pixel clock (LCD_PCLK). The pixel clock period formula is:

LCD_PCLK = source clock / (CLKDIV+2)

where source clock is the system clock when CLKSEL is written to ‘0’ and to the 2x system_clock when CLKSEL is written to ‘1’.

31.7.2 LCD Controller Configuration Register 1

Name: LCDC_LCDCFG1

Address: 0xF0000004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	VSPW	

23	22	21	20	19	18	17	16
VSPW							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	HSPW	

7	6	5	4	3	2	1	0
HSPW							

- **HSPW: Horizontal Synchronization Pulse Width**

Width of the LCD_HSYNC pulse, given in pixel clock cycles. Width is (HSPW+1) LCD_PCLK cycles.

- **VSPW: Vertical Synchronization Pulse Width**

Width of the LCD_VSYNC pulse, given in number of lines. Width is (VSPW+1) lines.

31.7.3 LCD Controller Configuration Register 2

Name: LCDC_LCDCFG2

Address: 0xF0000008

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	VBPW	

23	22	21	20	19	18	17	16
VBPW							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	VFPW	

7	6	5	4	3	2	1	0
VFPW							

- **VFPW: Vertical Front Porch Width**

This field indicates the number of lines at the end of the Frame. The blanking interval is equal to (VFPW+1) lines.

- **VBPW: Vertical Back Porch Width**

This field indicates the number of lines at the beginning of the Frame. The blanking interval is equal to VBPW lines.

31.7.4 LCD Controller Configuration Register 3

Name: LCDC_LCDCFG3

Address: 0xF000000C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	HBPW	

23	22	21	20	19	18	17	16
HBPW							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	HFPW	

7	6	5	4	3	2	1	0
HFPW							

- **HFPW: Horizontal Front Porch Width**

Number of pixel clock cycles inserted at the end of the active line. The interval is equal to (HFPW+1) LCD_PCLK cycles.

- **HBPW: Horizontal Back Porch Width**

Number of pixel clock cycles inserted at the beginning of the line. The interval is equal to (HBPW+1) LCD_PCLK cycles.

31.7.5 LCD Controller Configuration Register 4

Name: LCDC_LCDCFG4

Address: 0xF0000010

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	RPF		

23	22	21	20	19	18	17	16
RPF							

15	14	13	12	11	10	9	8
–	–	–	–	–	PPL		

7	6	5	4	3	2	1	0
PPL							

- **RPF: Number of Active Row Per Frame**

Number of active lines in the frame. The frame height is equal to (RPF+1) lines.

- **PPL: Number of Pixels Per Line**

Number of pixel in the frame. The number of active pixels in the frame is equal to (PPL+1) pixels.

31.7.6 LCD Controller Configuration Register 5

Name: LCDC_LCDCFG5

Address: 0xF0000014

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
GUARDTIME							

15	14	13	12	11	10	9	8
–	–	VSPHO	VSPSU	–	–	MODE	

7	6	5	4	3	2	1	0
DISPDLY	DITHER	–	DISPPOL	VSPDLYE	VSPDLYS	VSPOL	HSPOL

- **HSPOL: Horizontal Synchronization Pulse Polarity**

0: Active High

1: Active Low

- **VSPOL: Vertical Synchronization Pulse Polarity**

0: Active High

1: Active Low

- **VSPDLYS: Vertical Synchronization Pulse Start**

0: The first active edge of the Vertical synchronization pulse is synchronous with the second edge of the horizontal pulse.

1: The first active edge of the Vertical synchronization pulse is synchronous with the first edge of the horizontal pulse.

- **VSPDLYE: Vertical Synchronization Pulse End**

0: The second active edge of the Vertical synchronization pulse is synchronous with the second edge of the horizontal pulse.

1: The second active edge of the Vertical synchronization pulse is synchronous with the first edge of the horizontal pulse.

- **DISPPOL: Display Signal Polarity**

0: Active High

1: Active Low

- **DITHER: LCD Controller Dithering**

0: Dithering logical unit is disabled

1: Dithering logical unit is activated

- **DISPDLY: LCD Controller Display Power Signal Synchronization**

0: The LCD_DISP signal is asserted synchronously with the second active edge of the horizontal pulse.

1: The LCD_DISP signal is asserted asynchronously with both edges of the horizontal pulse.

- **MODE: LCD Controller Output Mode**

Value	Name	Description
0	OUTPUT_12BPP	LCD Output mode is set to 12 bits per pixel
1	OUTPUT_16BPP	LCD Output mode is set to 16 bits per pixel
2	OUTPUT_18BPP	LCD Output mode is set to 18 bits per pixel
3	OUTPUT_24BPP	LCD Output mode is set to 24 bits per pixel

- **VSPSU: LCD Controller Vertical synchronization Pulse Setup Configuration**

0: The vertical synchronization pulse is asserted synchronously with horizontal pulse edge.

1: The vertical synchronization pulse is asserted one pixel clock cycle before the horizontal pulse.

- **VSPHO: LCD Controller Vertical synchronization Pulse Hold Configuration**

0: The vertical synchronization pulse is asserted synchronously with horizontal pulse edge.

1: The vertical synchronization pulse is held active one pixel clock cycle after the horizontal pulse.

- **GUARDTIME: LCD DISPLAY Guard Time**

Number of frames inserted during start up before LCD_DISP assertion.

Number of frames inserted after LCD_DISP reset.

31.7.7 LCD Controller Configuration Register 6

Name: LCDC_LCDCFG6

Address: 0xF0000018

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PWMCVAL							

7	6	5	4	3	2	1	0
–	–	–	PWMPOL	–	PWMPS		

- **PWMPS: PWM Clock Prescaler**

Selects the configuration of the counter prescaler module.

Value	Name	Description
000	DIV_1	The counter advances at a rate of $f_{COUNTER} = f_{PWM\_SELECTED\_CLOCK}$
001	DIV_2	The counter advances at a rate of $f_{COUNTER} = f_{PWM\_SELECTED\_CLOCK/2}$
010	DIV_4	The counter advances at a rate of $f_{COUNTER} = f_{PWM\_SELECTED\_CLOCK/4}$
011	DIV_8	The counter advances at a rate of $f_{COUNTER} = f_{PWM\_SELECTED\_CLOCK/8}$
100	DIV_16	The counter advances at a rate of $f_{COUNTER} = f_{PWM\_SELECTED\_CLOCK/16}$
101	DIV_32	The counter advances at a of rate $f_{COUNTER} = f_{PWM\_SELECTED\_CLOCK/32}$
110	DIV_64	The counter advances at a of rate $f_{COUNTER} = f_{PWM\_SELECTED\_CLOCK/64}$

- **PWMPOL: LCD Controller PWM Signal Polarity**

This bit defines the polarity of the PWM output signal.

0: The output pulses are low level.

1: The output pulses are high level (the output will be high whenever the value in the counter is less than the value CVAL).

- **PWMCVAL: LCD Controller PWM Compare Value**

PWM compare value. Used to adjust the analog value obtained after an external filter to control the contrast of the display.

31.7.8 LCD Controller Enable Register

Name: LCDC_LCDEN

Address: 0xF0000020

Access: Write-only

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
–	–	–	–	PWMEN	DISPEN	SYNCEN	CLKEN

- **CLKEN: LCD Controller Pixel Clock Enable**

0: No effect

1: Pixel clock logical unit is activated.

- **SYNCEN: LCD Controller Horizontal and Vertical Synchronization Enable**

0: No effect

1: Both horizontal and vertical synchronization (LCD_VSYNC and LCD_HSYNC) signals are generated.

- **DISPEN: LCD Controller DISP Signal Enable**

0: No effect

1: LCD_DISP signal is generated.

- **PWMEN: LCD Controller Pulse Width Modulation Enable**

0: No effect

1: PWM is enabled.

31.7.9 LCD Controller Disable Register

Name: LCDC_LCDDIS

Address: 0xF0000024

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	PWMRST	DISPRST	SYNCRST	CLKRST
7	6	5	4	3	2	1	0
–	–	–	–	PWMDIS	DISPDIS	SYNCDIS	CLKDIS

- **CLKDIS: LCD Controller Pixel Clock Disable**

0: No effect.

1: Disables the pixel clock.

- **SYNCDIS: LCD Controller Horizontal and Vertical Synchronization Disable**

0: No effect.

1: Disables the synchronization signals after the end of the frame.

- **DISPDIS: LCD Controller DISP Signal Disable**

0: No effect.

1: Disables the DISP signal.

- **PWMDIS: LCD Controller Pulse Width Modulation Disable**

0: No effect.

1: Disables the pulse width modulation signal.

- **CLKRST: LCD Controller Clock Reset**

0: No effect.

1: Resets the pixel clock generator module. The pixel clock duty cycle may be violated.

- **SYNCRST: LCD Controller Horizontal and Vertical Synchronization Reset**

0: No effect.

1: Resets the timing engine. Both Horizontal and vertical pulse width are violated.

- **DISPRST: LCD Controller DISP Signal Reset**

0: No effect.

1: Resets the DISP signal.

- **PWMRST: LCD Controller PWM Reset**

0: No effect.

1: Resets the PWM module. The duty cycle may be violated.

31.7.10 LCD Controller Status Register

Name: LCDC_LCDSR

Address: 0xF0000028

Access: Read-only

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
–	–	–	SIPSTS	PWMSTS	DISPSTS	LCDSTS	CLKSTS

- **CLKSTS: Clock Status**

0: Pixel clock is disabled.

1: Pixel clock is running.

- **LCDSTS: LCD Controller Synchronization status**

0: Timing engine is disabled.

1: Timing engine is running.

- **DISPSTS: LCD Controller DISP Signal Status**

0: DISP is disabled.

1: DISP signal is activated.

- **PWMSTS: LCD Controller PWM Signal Status**

0: PWM is disabled.

1: PWM signal is activated.

- **SIPSTS: Synchronization In Progress**

0: Clock domain synchronization is terminated.

1: Synchronization is in progress. Access to the registers LCDC_LCDCCFG[0..6], LCDC_LCDEN and LCDC_LCDDIS has no effect.

31.7.11 LCD Controller Interrupt Enable Register

Name: LCDC_LCDIER

Address: 0xF000002C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	HEOIE		OVR1IE	BASEIE
7	6	5	4	3	2	1	0
–	–	–	FIFOERRIE	–	DISPIE	DISIE	SOFIE

- **SOFIE: Start of Frame Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **DISIE: LCD Disable Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **DISPIE: Powerup/Powerdown Sequence Terminated Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **FIFOERRIE: Output FIFO Error Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **BASEIE: Base Layer Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **OVR1IE: Overlay 1 Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **HEOIE: High End Overlay Interrupt Enable**

0: No effect.

1: Enables the interrupt.

31.7.12 LCD Controller Interrupt Disable Register

Name: LCDC_LCDIDR

Address: 0xF0000030

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	HEOID		OVR1ID	BASEID
7	6	5	4	3	2	1	0
–	–	–	FIFOERRID	–	DISPID	DISID	SOFID

- **SOFID: Start of Frame Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **DISID: LCD Disable Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **DISPID: Powerup/Powerdown Sequence Terminated Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **FIFOERRID: Output FIFO Error Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **BASEID: Base Layer Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **OVR1ID: Overlay 1 Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **HEOID: High End Overlay Interrupt Disable**

0: No effect.

1: Disables the interrupt.

31.7.13 LCD Controller Interrupt Mask Register

Name: LCDC_LCDIMR

Address: 0xF0000034

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	HEOIM		OVR1IM	BASEIM

7	6	5	4	3	2	1	0
–	–	–	FIFOERRIM	–	DISPIM	DISIM	SOFIM

- **SOFIM: Start of Frame Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **DISIM: LCD Disable Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **DISPIM: Powerup/Powerdown Sequence Terminated Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **FIFOERRIM: Output FIFO Error Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **BASEIM: Base Layer Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **OVR1IM: Overlay 1 Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **HEOIM: High End Overlay Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

31.7.14 LCD Controller Interrupt Status Register

Name: LCDC_LCDISR

Address: 0xF0000038

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	HEO		OVR1	BASE

7	6	5	4	3	2	1	0
–	–	–	FIFOERR	–	DISP	DIS	SOF

- **SOF: Start of Frame Interrupt Status**

0: No detection since last read of LCDC_LCDISR.

1: Indicates that a start of frame event has been detected. This flag is reset after a read operation.

- **DIS: LCD Disable Interrupt Status**

0: Horizontal and vertical timing generator has not yet been disabled.

1: Indicates that the horizontal and vertical timing generator has been disabled. This flag is reset after a read operation.

- **DISP: Powerup/Powerdown Sequence Terminated Interrupt Status**

0: Powerup sequence or powerdown sequence has not yet terminated.

1: Indicates the powerup sequence or powerdown sequence has terminated. This flag is reset after a read operation.

- **FIFOERR: Output FIFO Error**

0: No underflow has occurred in the output FIFO since last read of LCDC_LCDISR.

1: Indicates that an underflow has occurred in the output FIFO. This flag is reset after a read operation.

- **BASE: Base Layer Raw Interrupt Status**

0: No base layer interrupt detected since last read of LCDC_BASEISR.

1: Indicates that a base layer interrupt is pending. This flag is reset as soon as the LCDC_BASEISR is read.

- **OVR1: Overlay 1 Raw Interrupt Status**

0: No Overlay 1 layer interrupt detected since last read of LCDC_OVR1ISR.

1: Indicates that an Overlay 1 layer interrupt is pending. This flag is reset as soon as the LCDC_OVR1ISR is read.

- **HEO: High End Overlay Raw Interrupt Status**

0: No High End layer interrupt detected since last read of LCDC_HEOISR.

1: Indicates that a High End layer interrupt is pending. This flag is reset as soon as the LCDC_HEOISR is read.

31.7.15 LCD Controller Attribute Register

Name: LCDC_ATTR

Address: 0xF000003C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	HEOA2Q		OVR1A2Q	BASEA2Q
7	6	5	4	3	2	1	0
–	–	–	–	HEO		OVR1	BASE

- **BASE: Base Layer Update Attribute**

0: No effect.

1: Update the BASE window attributes.

- **OVR1: Overlay 1 Update Attribute**

0: No effect.

1: Update the OVR1 window attribute.

- **HEO: High End Overlay Update Attribute**

0: No effect.

1: Update the HEO window attribute.

- **BASEA2Q: Base Layer Update Add To Queue**

0: No effect.

1: Add the descriptor pointed to by the LCDC_BASEHEAD register to the descriptor list.

- **OVR1A2Q: Overlay 1 Update Add To Queue**

0: No effect.

1: Add the descriptor pointed to by the LCDC_OVR1HEAD register to the descriptor list.

- **HEOA2Q: High End Overlay Update Add To Queue**

0: No effect.

1: Add the descriptor pointed to by the LCDC_HEOHEAD register to the descriptor list.

31.7.16 Base Layer Channel Enable Register

Name: LCDC_BASECHER

Address: 0xF0000040

Access: Write-only

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
–	–	–	–	–	A2QEN	UPDATEEN	CHEN

- **CHEN: Channel Enable**

0: No effect

1: Enables the DMA channel

- **UPDATEEN: Update Overlay Attributes Enable**

0: No effect

1: Updates windows attributes on the next start of frame.

- **A2QEN: Add To Queue Enable**

0: No effect

1: Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list.

31.7.17 Base Layer Channel Disable Register

Name: LCDC_BASECHDR

Address: 0xF0000044

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CHRST

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CHDIS

- **CHDIS: Channel Disable**

0: No effect

1: Disables the layer at the end of the current frame. The frame is completed.

- **CHRST: Channel Reset**

0: No effect

1: Resets the layer immediately. The frame is aborted.

31.7.18 Base Layer Channel Status Register

Name: LCDC_BASECHSR

Address: 0xF0000048

Access: Read-only

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
–	–	–	–	–	A2QSR	UPDATESR	CHSR

- **CHSR: Channel Status**

0: Layer disabled

1: Layer enabled

- **UPDATESR: Update Overlay Attributes In Progress Status**

0: No update pending

1: Overlay attributes will be updated on the next frame

- **A2QSR: Add To Queue Status**

0: Add to queue not pending

1: Add to queue pending

31.7.19 Base Layer Interrupt Enable Register

Name: LCDC_BASEIER

Address: 0xF000004C

Access: Write-only

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
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DONE: End of List Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Enable**

0: No effect

1: Interrupt source is enabled

31.7.20 Base Layer Interrupt Disable Register

Name: LCDC_BASEIDR

Address: 0xF0000050

Access: Write-only

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
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DSCR: Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **ADD: Head Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DONE: End of List Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **OVR: Overflow Interrupt Disable**

0: No effect

1: Interrupt source is disabled

31.7.21 Base Layer Interrupt Mask Register

Name: LCDC_BASEIMR

Address: 0xF0000054

Access: Read-only

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
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DONE: End of List Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

31.7.22 Base Layer Interrupt Status Register

Name: LCDC_BASEISR

Address: 0xF0000058

Access: Read-only

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
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer**

0: No end of DMA transfer has been detected since last read of LCDC_BASEISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **DSCR: DMA Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_BASEISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **ADD: Head Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_BASEISR

1: The descriptor pointed to by the LCDC_BASEHEAD register has been loaded successfully. This flag is reset after a read operation.

- **DONE: End of List Detected**

0: No End of List condition occurred since last read of LCDC_BASEISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **OVR: Overflow Detected**

0: No overflow occurred since last read of LCDC_BASEISR

1: An overflow occurred. This flag is reset after a read operation.

31.7.23 Base DMA Head Register

Name: LCDC_BASEHEAD

Address: 0xF000005C

Access: Read/Write

31	30	29	28	27	26	25	24
HEAD							

23	22	21	20	19	18	17	16
HEAD							

15	14	13	12	11	10	9	8
HEAD							

7	6	5	4	3	2	1	0
HEAD						–	–

- **HEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

31.7.24 Base DMA Address Register

Name: LCDC_BASEADDR

Address: 0xF0000060

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: DMA Transfer Start Address**

Frame buffer base address

31.7.25 Base DMA Control Register

Name: LCDC_BASECTRL

Address: 0xF0000064

Access: Read/Write

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
–	–	DONEIEN	ADDIEN	DSCRIEN	DMAIEN	LFETCH	DFETCH

- **DFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled

1: Transfer Descriptor fetch is enabled

- **LFETCH: Lookup Table Fetch Enable**

0: Lookup Table DMA fetch is disabled

1: Lookup Table DMA fetch is enabled

- **DMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled

1: DMA transfer completed interrupt is disabled

- **DSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled

1: Transfer descriptor loaded interrupt is disabled

- **ADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled

1: Transfer descriptor added to queue interrupt is enabled

- **DONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled

1: End of list interrupt is enabled

31.7.26 Base DMA Next Register

Name: LCDC_BASENEXT

Address: 0xF0000068

Access: Read/Write

31	30	29	28	27	26	25	24
NEXT							

23	22	21	20	19	18	17	16
NEXT							

15	14	13	12	11	10	9	8
NEXT							

7	6	5	4	3	2	1	0
NEXT							

- **NEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

31.7.27 Base Layer Configuration Register 0

Name: LCDC_BASECFG0

Address: 0xF000006C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DLBO

7	6	5	4	3	2	1	0
–	–	BLEN		–	–	–	SIF

- **SIF: Source Interface**

0: Base Layer data is retrieved through AHB interface 0.

1: Base Layer data is retrieved through AHB interface 1.

- **BLEN: AHB Burst Length**

Value	Name	Description
0	AHB_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **DLBO: Defined Length Burst Only For Channel Bus Transaction**

0: Undefined length INCR burst is used for a burst of 2 and 3 beats.

1: Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16).

31.7.28 Base Layer Configuration Register 1

Name: LCDC_BASECFG1

Address: 0xF0000070

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	-	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–				–	–	CLUTMODE	

7	6	5	4	3	2	1	0
RGBMODE				–	–	–	CLUTEN

- **CLUTEN: Color Lookup Table Mode Enable**

0: RGB mode is selected.

1: Color Lookup Table mode is selected.

- **RGBMODE: RGB Mode Input Selection**

Value	Name	Description
0	12BPP_RGB_444	12 bpp RGB 444
1	16BPP_ARGB_4444	16 bpp ARGB 4444
2	16BPP_RGBA_4444	16 bpp RGBA 4444
3	16BPP_RGB_565	16 bpp RGB 565
4	16BPP_TRGB_1555	16 bpp TRGB 1555
5	18BPP_RGB_666	18 bpp RGB 666
6	18BPP_RGB_666PACKED	18 bpp RGB 666 PACKED
7	19BPP_TRGB_1666	19 bpp TRGB 1666
8	19BPP_TRGB_PACKED	19 bpp TRGB 1666 PACKED
9	24BPP_RGB_888	24 bpp RGB 888
10	24BPP_RGB_888_PACKED	24 bpp RGB 888 PACKED
11	25BPP_TRGB_1888	25 bpp TRGB 1888
12	32BPP_ARGB_8888	32 bpp ARGB 8888
13	32BPP_RGBA_8888	32 bpp RGBA 8888

- **CLUTMODE: Color Lookup Table Mode Input Selection**

Value	Name	Description
0	CLUT_1BPP	Color Lookup Table mode set to 1 bit per pixel
1	CLUT_2BPP	Color Lookup Table mode set to 2 bits per pixel
2	CLUT_4BPP	Color Lookup Table mode set to 4 bits per pixel
3	CLUT_8BPP	Color Lookup Table mode set to 8 bits per pixel

31.7.29 Base Layer Configuration Register 2

Name: LCDC_BASECFG2

Address: 0xF0000074

Access: Read/Write

31	30	29	28	27	26	25	24
XSTRIDE							

23	22	21	20	19	18	17	16
XSTRIDE							

15	14	13	12	11	10	9	8
XSTRIDE							

7	6	5	4	3	2	1	0
XSTRIDE							

- **XSTRIDE: Horizontal Stride**

XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

31.7.30 Base Layer Configuration Register 3

Name: LCDC_BASECFG3

Address: 0xF0000078

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RDEF							

15	14	13	12	11	10	9	8
GDEF							

7	6	5	4	3	2	1	0
BDEF							

- **RDEF: Red Default**

Default Red color when the Base DMA channel is disabled

- **GDEF: Green Default**

Default Green color when the Base DMA channel is disabled

- **BDEF: Blue Default**

Default Blue color when the Base DMA channel is disabled

31.7.31 Base Layer Configuration Register 4

Name: LCDC_BASECFG4

Address: 0xF000007C

Access: Read/Write

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

- **DMA: Use DMA Data Path**

0: The default color is used on the Base Layer.

1: The DMA channel retrieves the pixels stream from the memory.

- **REP: Use Replication logic to expand RGB color to 24 bits**

0: When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0.

1: When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb.

- **DISCEN: Discard Area Enable**

0: The whole frame is retrieved from memory.

1: The DMA channel discards the area located at screen coordinate {DISCXPOS, DISCYPOS}.

31.7.32 Base Layer Configuration Register 5

Name: LCDC_BASECFG5

Address: 0xF0000080

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	DISCYPOS		

23	22	21	20	19	18	17	16
DISCYPOS							

15	14	13	12	11	10	9	8
–	–	–	–		DISCXPOS		

7	6	5	4	3	2	1	0
DISCXPOS							

- **DISCXPOS: Discard Area Horizontal Coordinate**

Horizontal Position of the Discard Area

- **DISCYPOS: Discard Area Vertical Coordinate**

Vertical Position of the Discard Area

31.7.33 Base Layer Configuration Register 6

Name: LCDC_BASECFG6

Address: 0xF0000084

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	DISCYSIZE		

23	22	21	20	19	18	17	16
DISCYSIZE							

15	14	13	12	11	10	9	8
–	–	–	–		DISCXSIZE		

7	6	5	4	3	2	1	0
DISCXSIZE							

- **DISCXSIZE: Discard Area Horizontal Size**

Discard Horizontal size in pixels. The Discard size is set to (DISCXSIZE + 1) pixels horizontally.

- **DISCYSIZE: Discard Area Vertical Size**

Discard Vertical size in pixels. The Discard size is set to (DISCYSIZE + 1) pixels vertically.

31.7.34 Overlay 1 Channel Enable Register

Name: LCDC_OVR1CHER

Address: 0xF0000140

Access: Write-only

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
–	–	–	–	–	A2QEN	UPDATEEN	CHEN

- **CHEN: Channel Enable**

0: No effect

1: Enables the DMA channel

- **UPDATEEN: Update Overlay Attributes Enable**

0: No effect

1: Updates window attributes (size, alpha blending, etc.) on the next start of frame.

- **A2QEN: Add To Queue Enable**

0: No effect

1: Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list.

31.7.35 Overlay 1 Channel Disable Register

Name: LCDC_OVR1CHDR

Address: 0xF0000144

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CHRST

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CHDIS

- **CHDIS: Channel Disable**

0: No effect

1: Disables the layer at the end of the current frame. The frame is completed.

- **CHRST: Channel Reset**

0: No effect

1: Resets the layer immediately. The frame is aborted.

31.7.36 Overlay 1 Channel Status Register

Name: LCDC_OVR1CHSR

Address: 0xF0000148

Access: Read-only

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
–	–	–	–	–	A2QSR	UPDATESR	CHSR

- **CHSR: Channel Status**

0: Layer disabled

1: Layer enabled

- **UPDATESR: Update Overlay Attributes In Progress Status**

0: No update pending

1: Overlay attributes will be updated on the next frame

- **A2QSR: Add To Queue Status**

0: Add to queue not pending

1: Add to queue pending

31.7.37 Overlay 1 Interrupt Enable Register

Name: LCDC_OVR1IER

Address: 0xF000014C

Access: Write-only

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
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DONE: End of List Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Enable**

0: No effect

1: Interrupt source is enabled

31.7.38 Overlay 1 Interrupt Disable Register

Name: LCDC_OVR1IDR

Address: 0xF0000150

Access: Write-only

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
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DSCR: Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **ADD: Head Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DONE: End of List Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **OVR: Overflow Interrupt Disable**

0: No effect

1: Interrupt source is disabled

31.7.39 Overlay 1 Interrupt Mask Register

Name: LCDC_OVR1IMR

Address: 0xF0000154

Access: Read-only

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
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DONE: End of List Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

31.7.40 Overlay 1 Interrupt Status Register

Name: LCDC_OVR1ISR

Address: 0xF0000158

Access: Read-only

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
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer**

0: No End of Transfer has been detected since last read of LCDC_OVR1ISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **DSCR: DMA Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_OVR1ISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **ADD: Head Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_OVR1ISR

1: The descriptor pointed to by the LCDC_OVR1HEAD register has been loaded successfully. This flag is reset after a read operation.

- **DONE: End of List Detected**

0: No End of List condition has occurred since last read of LCDC_OVR1ISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **OVR: Overflow Detected**

0: No overflow occurred since last read of LCDC_OVR1ISR

1: An overflow occurred. This flag is reset after a read operation.

31.7.41 Overlay 1 Head Register

Name: LCDC_OVR1HEAD

Address: 0xF000015C

Access: Read/Write

31	30	29	28	27	26	25	24
HEAD							

23	22	21	20	19	18	17	16
HEAD							

15	14	13	12	11	10	9	8
HEAD							

7	6	5	4	3	2	1	0
HEAD						–	–

- **HEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

31.7.42 Overlay 1 Address Register

Name: LCDC_OVR1ADDR

Address: 0xF0000160

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: DMA Transfer Overlay 1 Address**

Overlay 1 frame buffer base address

31.7.43 Overlay 1 Control Register

Name: LCDC_OVR1CTRL

Address: 0xF0000164

Access: Read/Write

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
–	–	DONEIEN	ADDIEN	DSCRIEN	DMAIEN	LFETCH	DFETCH

- **DFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled

1: Transfer Descriptor fetch is enabled

- **LFETCH: Lookup Table Fetch Enable**

0: Lookup Table DMA fetch is disabled

1: Lookup Table DMA fetch is enabled

- **DMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled

1: DMA transfer completed interrupt is disabled

- **DSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled

1: Transfer descriptor loaded interrupt is disabled

- **ADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled

1: Transfer descriptor added to queue interrupt is enabled

- **DONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled

1: End of list interrupt is enabled

31.7.44 Overlay 1 Next Register

Name: LCDC_OVR1NEXT

Address: 0xF0000168

Access: Read/Write

31	30	29	28	27	26	25	24
NEXT							

23	22	21	20	19	18	17	16
NEXT							

15	14	13	12	11	10	9	8
NEXT							

7	6	5	4	3	2	1	0
NEXT							

- **NEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

31.7.45 Overlay 1 Configuration Register 0

Name: LCDC_OVR1CFG0

Address: 0xF000016C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	LOCKDIS	ROTDIS	–	–	–	DLBO

7	6	5	4	3	2	1	0
–	–	BLEN		–	–	–	SIF

- **SIF: Source Interface**

0: Base Layer data is retrieved through AHB interface 0.

1: Base Layer data is retrieved through AHB interface 1.

- **BLEN: AHB Burst Length**

Value	Name	Description
0	AHB_BLEN_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_BLEN_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_BLEN_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_BLEN_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **DLBO: Defined Length Burst Only for Channel Bus Transaction**

0: Undefined length INCR burst is used for a burst of 2 and 3 beats.

1: Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16).

- **ROTDIS: Hardware Rotation Optimization Disable**

0: Rotation optimization is enabled.

1: Rotation optimization is disabled.

- **LOCKDIS: Hardware Rotation Lock Disable**

0: AHB lock signal is asserted when a rotation is performed.

1: AHB lock signal is cleared when a rotation is performed.

31.7.46 Overlay 1 Configuration Register 1

Name: LCDC_OVR1CFG1

Address: 0xF0000170

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	CLUTMODE	

7	6	5	4	3	2	1	0
RGBMODE				–	–	–	CLUTEN

- **CLUTEN: Color Lookup Table Mode Enable**

0: RGB mode is selected.

1: Color Lookup Table mode is selected.

- **RGBMODE: RGB Mode Input Selection**

Value	Name	Description
0	12BPP_RGB_444	12 bpp RGB 444
1	16BPP_ARGB_4444	16 bpp ARGB 4444
2	16BPP_RGBA_4444	16 bpp RGBA 4444
3	16BPP_RGB_565	16 bpp RGB 565
4	16BPP_TRGB_1555	16 bpp TRGB 1555
5	18BPP_RGB_666	18 bpp RGB 666
6	18BPP_RGB_666PACKED	18 bpp RGB 666 PACKED
7	19BPP_TRGB_1666	19 bpp TRGB 1666
8	19BPP_TRGB_PACKED	19 bpp TRGB 1666 PACKED
9	24BPP_RGB_888	24 bpp RGB 888
10	24BPP_RGB_888_PACKED	24 bpp RGB 888 PACKED
11	25BPP_TRGB_1888	25 bpp TRGB 1888
12	32BPP_ARGB_8888	32 bpp ARGB 8888
13	32BPP_RGBA_8888	32 bpp RGBA 8888

- **CLUTMODE: Color Lookup Table Mode Input Selection**

Value	Name	Description
0	CLUT_1BPP	Color Lookup Table mode set to 1 bit per pixel
1	CLUT_2BPP	Color Lookup Table mode set to 2 bits per pixel
2	CLUT_4BPP	Color Lookup Table mode set to 4 bits per pixel
3	CLUT_8BPP	Color Lookup Table mode set to 8 bits per pixel

31.7.47 Overlay 1 Configuration Register 2

Name: LCDC_OVR1CFG2

Address: 0xF0000174

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YPOS		

23	22	21	20	19	18	17	16
YPOS							

15	14	13	12	11	10	9	8
–	–	–	–	–	XPOS		

7	6	5	4	3	2	1	0
XPOS							

- **XPOS: Horizontal Window Position**

Overlay 1 Horizontal window position.

- **YPOS: Vertical Window Position**

Overlay 1 Vertical window position.

31.7.48 Overlay 1 Configuration Register 3

Name: LCDC_OVR1CFG3

Address: 0xF0000178

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YSIZE		

23	22	21	20	19	18	17	16
YSIZE							

15	14	13	12	11	10	9	8
–	–	–	–	–	XSIZE		

7	6	5	4	3	2	1	0
XSIZE							

- **XSIZE: Horizontal Window Size**

Overlay 1 window width in pixels. The window width is set to (XSIZE + 1).

The following constraint must be met: XPOS + XSIZE ≤ PPL

- **YSIZE: Vertical Window Size**

Overlay 1 window height in pixels. The window height is set to (YSIZE + 1).

The following constraint must be met: YPOS + YSIZE ≤ RPF

31.7.49 Overlay 1 Configuration Register 4

Name: LCDC_OVR1CFG4

Address: 0xF000017C

Access: Read/Write

31	30	29	28	27	26	25	24
XSTRIDE							

23	22	21	20	19	18	17	16
XSTRIDE							

15	14	13	12	11	10	9	8
XSTRIDE							

7	6	5	4	3	2	1	0
XSTRIDE							

- **XSTRIDE: Horizontal Stride**

XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

31.7.50 Overlay 1 Configuration Register 5

Name: LCDC_OVR1CFG5

Address: 0xF0000180

Access: Read/Write

31	30	29	28	27	26	25	24
PSTRIDE							

23	22	21	20	19	18	17	16
PSTRIDE							

15	14	13	12	11	10	9	8
PSTRIDE							

7	6	5	4	3	2	1	0
PSTRIDE							

- **PSTRIDE: Pixel Stride**

PSTRIDE represents the memory offset, in bytes, between two pixels of the image.

31.7.51 Overlay 1 Configuration Register 6

Name: LCDC_OVR1CFG6

Address: 0xF0000184

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RDEF							

15	14	13	12	11	10	9	8
GDEF							

7	6	5	4	3	2	1	0
BDEF							

- **RDEF: Red Default**

Default Red color when the Overlay 1 DMA channel is disabled.

- **GDEF: Green Default**

Default Green color when the Overlay 1 DMA channel is disabled.

- **BDEF: Blue Default**

Default Blue color when the Overlay 1 DMA channel is disabled.

31.7.52 Overlay 1 Configuration Register 7

Name: LCDC_OVR1CFG7

Address: 0xF0000188

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RKEY							

15	14	13	12	11	10	9	8
GKEY							

7	6	5	4	3	2	1	0
BKEY							

- **RKEY: Red Color Component Chroma Key**

Reference Red chroma key used to match the Red color of the current overlay.

- **GKEY: Green Color Component Chroma Key**

Reference Green chroma key used to match the Green color of the current overlay.

- **BKEY: Blue Color Component Chroma Key**

Reference Blue chroma key used to match the Blue color of the current overlay.

31.7.53 Overlay 1 Configuration Register 8

Name: LCDC_OVR1CFG8

Address: 0xF000018C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RMASK							

15	14	13	12	11	10	9	8
GMASK							

7	6	5	4	3	2	1	0
BMASK							

- **RMASK: Red Color Component Chroma Key Mask**

Red Mask used when the compare function is used. If a bit is set then this bit is compared.

- **GMASK: Green Color Component Chroma Key Mask**

Green Mask used when the compare function is used. If a bit is set then this bit is compared.

- **BMASK: Blue Color Component Chroma Key Mask**

Blue Mask used when the compare function is used. If a bit is set then this bit is compared.

31.7.54 Overlay 1 Configuration Register 9

Name: LCDC_OVR1CFG9

Address: 0xF0000190

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
GA							

15	14	13	12	11	10	9	8
–	–	–	–	–	DSTKEY	REP	DMA

7	6	5	4	3	2	1	0
OVR	LAEN	GAEN	REVALPHA	ITER	ITER2BL	INV	CRKEY

- **CRKEY: Blender Chroma Key Enable**

0: Chroma key matching is disabled.

1: Chroma key matching is enabled.

- **INV: Blender Inverted Blender Output Enable**

0: Iterated pixel is the blended pixel.

1: Iterated pixel is the inverted pixel.

- **ITER2BL: Blender Iterated Color Enable**

0: Final adder stage operand is set to 0.

1: Final adder stage operand is set to the iterated pixel value.

- **ITER: Blender Use Iterated Color**

0: Pixel difference is set to 0.

1: Pixel difference is set to the iterated pixel value.

- **REVALPHA: Blender Reverse Alpha**

0: Pixel difference is multiplied by alpha.

1: Pixel difference is multiplied by 1 - alpha.

- **GAEN: Blender Global Alpha Enable**

0: Global alpha blending coefficient is disabled.

1: Global alpha blending coefficient is enabled.

- **LAEN: Blender Local Alpha Enable**

0: Local alpha blending coefficient is disabled.

1: Local alpha blending coefficient is enabled.

- **OVR: Blender Overlay Layer Enable**

0: Overlay pixel color is set to the default overlay pixel color.

1: Overlay pixel color is set to the DMA channel pixel color.

- **DMA: Blender DMA Layer Enable**

0: The default color is used on the Overlay 1 Layer.

1: The DMA channel retrieves the pixels stream from the memory.

- **REP: Use Replication logic to expand RGB color to 24 bits**

0: When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0.

1: When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb.

- **DSTKEY: Destination Chroma Keying**

0: Source Chroma keying is enabled.

1: Destination Chroma keying is used.

- **GA: Blender Global Alpha**

Global alpha blender for the current layer.

31.7.55 High End Overlay Channel Enable Register

Name: LCDC_HEOCHER

Address: 0xF0000340

Access: Write-only

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
–	–	–	–	–	A2QEN	UPDATEEN	CHEN

- **CHEN: Channel Enable**

0: No effect

1: Enables the DMA channel

- **UPDATEEN: Update Overlay Attributes Enable**

0: No effect

1: Updates windows attributes on the next start of frame.

- **A2QEN: Add To Queue Enable**

0: No effect

1: Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list.

31.7.56 High End Overlay Channel Disable Register

Name: LCDC_HEOCHDR

Address: 0xF0000344

Access: Write-only

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

- **CHDIS: Channel Disable**

0: No effect

1: Disables the layer at the end of the current frame. The frame is completed.

- **CHRST: Channel Reset**

0: No effect

1: Resets the layer immediately. The frame is aborted.

31.7.57 High End Overlay Channel Status Register

Name: LCDC_HEOCHSR

Address: 0xF0000348

Access: Read-only

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
–	–	–	–	–	A2QSR	UPDATESR	CHSR

- **CHSR: Channel Status**

0: Layer disabled

1: Layer enabled

- **UPDATESR: Update Overlay Attributes In Progress Status**

0: No update pending

1: Overlay attributes will be updated on the next frame

- **A2QSR: Add To Queue Status**

0: Add to queue not pending

1: Add to queue pending

31.7.58 High End Overlay Interrupt Enable Register

Name: LCDC_HEOIER

Address: 0xF000034C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	VOVR	VDONE	VADD	VDSCR	VDMA	–	–

15	14	13	12	11	10	9	8
–	UOVR	UDONE	UADD	UDSCR	UDMA	–	–

7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DONE: End of List Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UDMA: End of DMA Transfer for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UDSCR: Descriptor Loaded for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UADD: Head Descriptor Loaded for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UDONE: End of List for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UOVR: Overflow for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VDMA: End of DMA for V Chrominance Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VDSCR: Descriptor Loaded for V Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VADD: Head Descriptor Loaded for V Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VDONE: End of List for V Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VOVR: Overflow for V Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

31.7.59 High End Overlay Interrupt Disable Register

Name: LCDC_HEOIDR

Address: 0xF0000350

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	VOVR	VDONE	VADD	VDSCR	VDMA	–	–
15	14	13	12	11	10	9	8
–	UOVR	UDONE	UADD	UDSCR	UDMA	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DSCR: Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **ADD: Head Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DONE: End of List Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **OVR: Overflow Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **UDMA: End of DMA Transfer for U or UV Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **UDSCR: Descriptor Loaded for U or UV Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **UADD: Head Descriptor Loaded for U or UV Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **UDONE: End of List Interrupt for U or UV Chrominance Component Disable**

0: No effect

1: Interrupt source is disabled

- **UOVR: Overflow Interrupt for U or UV Chrominance Component Disable**

0: No effect

1: Interrupt source is disabled

- **VDMA: End of DMA Transfer for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **VDSCR: Descriptor Loaded for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **VADD: Head Descriptor Loaded for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **VDONE: End of List for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **VOVR: Overflow for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

31.7.60 High End Overlay Interrupt Mask Register

Name: LCDC_HEOIMR

Address: 0xF0000354

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	VOVR	VDONE	VADD	VDSCR	VDMA	–	–

15	14	13	12	11	10	9	8
–	UOVR	UDONE	UADD	UDSCR	UDMA	–	–

7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DONE: End of List Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UDMA: End of DMA Transfer for U or UV Chrominance Component Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UDSCR: Descriptor Loaded for U or UV Chrominance Component Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UADD: Head Descriptor Loaded for U or UV Chrominance Component Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UDONE: End of List for U or UV Chrominance Component Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UOVR: Overflow for U Chrominance Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VDMA: End of DMA Transfer for V Chrominance Component Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VDSCR: Descriptor Loaded for V Chrominance Component Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VADD: Head Descriptor Loaded for V Chrominance Component Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VDONE: End of List for V Chrominance Component Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VOVR: Overflow for V Chrominance Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

31.7.61 High End Overlay Interrupt Status Register

Name: LCDC_HEOISR

Address: 0xF0000358

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	VOVR	VDONE	VADD	VDSCR	VDMA	–	–

15	14	13	12	11	10	9	8
–	UOVR	UDONE	UADD	UDSCR	UDMA	–	–

7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer**

0: No end of transfer has been detected since last read of LCDC_HEOISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **DSCR: DMA Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **ADD: Head Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: The descriptor pointed to by the LCDC_HEOHEAD register has been loaded successfully. This flag is reset after a read operation.

- **DONE: End of List Detected**

0: No End of List condition occurred since last read of LCDC_HEOISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **OVR: Overflow Detected**

0: No overflow occurred since last read of LCDC_HEOISR

1: An overflow occurred. This flag is reset after a read operation.

- **UDMA: End of DMA Transfer for U Component**

0: No End of Transfer has been detected since last read of LCDC_HEOISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **UDSCR: DMA Descriptor Loaded for U Component**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **UADD: Head Descriptor Loaded for U Component**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: The descriptor pointed to by the LCDC_HEOUHEAD register has been loaded successfully. This flag is reset after a read operation.

- **UDONE: End of List Detected for U Component**

0: No End of List condition occurred since last read of LCDC_HEOISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **UOVR: Overflow Detected for U Component**

0: No overflow occurred since last read of LCDC_HEOISR

1: An overflow occurred. This flag is reset after a read operation.

- **VDMA: End of DMA Transfer for V Component**

0: No End of Transfer has been detected since last read of LCDC_HEOISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **VDSCR: DMA Descriptor Loaded for V Component**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **VADD: Head Descriptor Loaded for V Component**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: The descriptor pointed to by the LCDC_HEOVHEAD register has been loaded successfully. This flag is reset after a read operation.

- **VDONE: End of List Detected for V Component**

0: No End of List condition occurred since last read of LCDC_HEOISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **VOVR: Overflow Detected for V Component**

0: No overflow occurred since last read of LCDC_HEOISR

1: An overflow occurred. This flag is reset after a read operation.

31.7.62 High End Overlay DMA Head Register

Name: LCDC_HEOHEAD

Address: 0xF000035C

Access: Read/Write

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

- **HEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

31.7.63 High End Overlay DMA Address Register

Name: LCDC_HEOADDR

Address: 0xF0000360

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: DMA Transfer Start Address**

Frame Buffer Base Address.

31.7.64 High End Overlay DMA Control Register

Name: LCDC_HEOCTRL

Address: 0xF0000364

Access: Read/Write

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
–	–	DONEIEN	ADDIEN	DSCRIEN	DMAIEN	LFETCH	DFETCH

- **DFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled.

1: Transfer Descriptor fetch is enabled.

- **LFETCH: Lookup Table Fetch Enable**

0: Lookup Table DMA fetch is disabled.

1: Lookup Table DMA fetch is enabled.

- **DMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled.

1: DMA transfer completed interrupt is disabled.

- **DSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled.

1: Transfer descriptor loaded interrupt is disabled.

- **ADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled.

1: Transfer descriptor added to queue interrupt is enabled.

- **DONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled.

1: End of list interrupt is enabled.

31.7.65 High End Overlay DMA Next Register

Name: LCDC_HEONEXT

Address: 0xF0000368

Access: Read/Write

31	30	29	28	27	26	25	24
NEXT							

23	22	21	20	19	18	17	16
NEXT							

15	14	13	12	11	10	9	8
NEXT							

7	6	5	4	3	2	1	0
NEXT							

- **NEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

31.7.66 High End Overlay U-UV DMA Head Register

Name: LCDC_HEOUHEAD

Address: 0xF000036C

Access: Read/Write

31	30	29	28	27	26	25	24
UHEAD							

23	22	21	20	19	18	17	16
UHEAD							

15	14	13	12	11	10	9	8
UHEAD							

7	6	5	4	3	2	1	0
UHEAD							

- **UHEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

31.7.67 High End Overlay U-UV DMA Address Register

Name: LCDC_HEOUADDR

Address: 0xF0000370

Access: Read/Write

31	30	29	28	27	26	25	24
UADDR							

23	22	21	20	19	18	17	16
UADDR							

15	14	13	12	11	10	9	8
UADDR							

7	6	5	4	3	2	1	0
UADDR							

- **UADDR: DMA Transfer Start Address for U or UV Chrominance**

U or UV frame buffer address.

31.7.68 High End Overlay U-UV DMA Control Register

Name: LCDC_HEOUCTRL

Address: 0xF0000374

Access: Read/Write

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
–	–	UDONEIEN	UADDIEN	UDSCRIEN	UDMAIEN	–	UDFETCH

- **UDFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled.

1: Transfer Descriptor fetch is enabled.

- **UDMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled.

1: DMA transfer completed interrupt is disabled.

- **UDSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled.

1: Transfer descriptor loaded interrupt is disabled.

- **UADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled.

1: Transfer descriptor added to queue interrupt is enabled.

- **UDONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled.

1: End of list interrupt is enabled.

31.7.69 High End Overlay U-UV DMA Next Register

Name: LCDC_HEOUNEXT

Address: 0xF0000378

Access: Read/Write

31	30	29	28	27	26	25	24
UNEXT							

23	22	21	20	19	18	17	16
UNEXT							

15	14	13	12	11	10	9	8
UNEXT							

7	6	5	4	3	2	1	0
UNEXT							

- **UNEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

31.7.70 High End Overlay V DMA Head Register

Name: LCDC_HEOVHEAD

Address: 0xF000037C

Access: Read/Write

31	30	29	28	27	26	25	24
VHEAD							

23	22	21	20	19	18	17	16
VHEAD							

15	14	13	12	11	10	9	8
VHEAD							

7	6	5	4	3	2	1	0
VHEAD							

- **VHEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

31.7.71 High End Overlay V DMA Address Register

Name: LCDC_HEOVADDR

Address: 0xF0000380

Access: Read/Write

31	30	29	28	27	26	25	24
VADDR							

23	22	21	20	19	18	17	16
VADDR							

15	14	13	12	11	10	9	8
VADDR							

7	6	5	4	3	2	1	0
VADDR							

- **VADDR: DMA Transfer Start Address for V Chrominance**

Frame Buffer Base Address.

31.7.72 High End Overlay V DMA Control Register

Name: LCDC_HEOVCTRL

Address: 0xF0000384

Access: Read/Write

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
–	–	VDONEIEN	VADDIEN	VDSCRIEN	VDMAIEN	–	VDFETCH

- **VDFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled.

1: Transfer Descriptor fetch is enabled.

- **VDMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled.

1: DMA transfer completed interrupt is disabled.

- **VDSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled.

1: Transfer descriptor loaded interrupt is disabled.

- **VADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled.

1: Transfer descriptor added to queue interrupt is enabled.

- **VDONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled.

1: End of list interrupt is enabled.

31.7.73 High End Overlay V DMA Next Register

Name: LCDC_HEOVNEXT

Address: 0xF0000388

Access: Read/Write

31	30	29	28	27	26	25	24
VNEXT							

23	22	21	20	19	18	17	16
VNEXT							

15	14	13	12	11	10	9	8
VNEXT							

7	6	5	4	3	2	1	0
VNEXT							

- **VNEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

31.7.74 High End Overlay Configuration Register 0

Name: LCDC_HEOCFG0

Address: 0xF000038C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	LOCKDIS	ROTDIS	–	–	–	DLBO

7	6	5	4	3	2	1	0
BLENUV		BLEN		–	–	–	SIF

- **SIF: Source Interface**

0: Base Layer data is retrieved through AHB interface 0.

1: Base Layer data is retrieved through AHB interface 1.

- **BLEN: AHB Burst Length**

Value	Name	Description
0	AHB_BLEN_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_BLEN_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_BLEN_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_BLEN_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **BLENUV: AHB Burst Length for U-V Channel**

Value	Name	Description
0	AHB_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **DLBO: Defined Length Burst Only For Channel Bus Transaction**

0: Undefined length INCR burst is used for a burst of 2 and 3 beats.

1: Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16).

- **ROTDIS: Hardware Rotation Optimization Disable**

0: Rotation optimization is enabled.

1: Rotation optimization is disabled.

- **LOCKDIS: Hardware Rotation Lock Disable**

0: AHB lock signal is asserted when a rotation is performed.

1: AHB lock signal is cleared when a rotation is performed.

31.7.75 High End Overlay Configuration Register 1

Name: LCDC_HEOCFG1

Address: 0xF0000390

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	DSCALEOPT	–	–	YUV422SWP	YUV422ROT

15	14	13	12	11	10	9	8
YUVMODE				–	–	CLUTMODE	

7	6	5	4	3	2	1	0
RGBMODE				–	–	YUVEN	CLUTEN

- **CLUTEN: Color Lookup Table Mode Enable**

0: RGB mode is selected.

1: Color Lookup Table mode is selected.

- **YUVEN: YUV Color Space Enable**

0: Color space is RGB

1: Color Space is YUV

- **RGBMODE: RGB Mode Input Selection**

Value	Name	Description
0	12BPP_RGB_444	12 bpp RGB 444
1	16BPP_ARGB_4444	16 bpp ARGB 4444
2	16BPP_RGBA_4444	16 bpp RGBA 4444
3	16BPP_RGB_565	16 bpp RGB 565
4	16BPP_TRGB_1555	16 bpp TRGB 1555
5	18BPP_RGB_666	18 bpp RGB 666
6	18BPP_RGB_666PACKED	18 bpp RGB 666 PACKED
7	19BPP_TRGB_1666	19 bpp TRGB 1666
8	19BPP_TRGB_PACKED	19 bpp TRGB 1666 PACKED
9	24BPP_RGB_888	24 bpp RGB 888
10	24BPP_RGB_888_PACKED	24 bpp RGB 888 PACKED
11	25BPP_TRGB_1888	25 bpp TRGB 1888
12	32BPP_ARGB_8888	32 bpp ARGB 8888
13	32BPP_RGBA_8888	32 bpp RGBA 8888

- **CLUTMODE: Color Lookup Table Mode Input Selection**

Value	Name	Description
0	CLUT_1BPP	Color Lookup Table mode set to 1 bit per pixel
1	CLUT_2BPP	Color Lookup Table mode set to 2 bits per pixel
2	CLUT_4BPP	Color Lookup Table mode set to 4 bits per pixel
3	CLUT_8BPP	Color Lookup Table mode set to 8 bits per pixel

- **YUVMODE: YUV Mode Input Selection**

Value	Name	Description
0	32BPP_AYCBCR	32 bpp AYCbCr 444
1	16BPP_YCBCR_MODE0	16 bpp Cr(n)Y(n+1)Cb(n)Y(n) 422
2	16BPP_YCBCR_MODE1	16 bpp Y(n+1)Cr(n)Y(n)Cb(n) 422
3	16BPP_YCBCR_MODE2	16 bpp Cb(n)Y(+1)Cr(n)Y(n) 422
4	16BPP_YCBCR_MODE3	16 bpp Y(n+1)Cb(n)Y(n)Cr(n) 422
5	16BPP_YCBCR_SEMIPLANAR	16 bpp Semiplanar 422 YCbCr
6	16BPP_YCBCR_PLANAR	16 bpp Planar 422 YCbCr
7	12BPP_YCBCR_SEMIPLANAR	12 bpp Semiplanar 420 YCbCr
8	12BPP_YCBCR_PLANAR	12 bpp Planar 420 YCbCr

- **YUV422ROT: YUV 4:2:2 Rotation**

0: Chroma Upsampling kernel is configured to use 0 and 180 degrees algorithm

1: Indicates that the Chroma Upsampling kernel is configured to use the 4:2:2 Rotation Algorithm. This bit is relevant only when a rotation angle of 90 degrees or 270 degrees is used.

- **YUV422SWP: YUV 4:2:2 Swap**

0: The two Y components of the YUV 4:2:2 packed data stream are not swapped.

1: The two Y components of the YUV 4:2:2 packed data stream are swapped.

- **DSCALEOPT: Down Scaling Bandwidth Optimization**

0: Scaler Optimization is disabled.

1: Scaler Optimization is enabled, only relevant pixels are retrieved from memory to fill the scaler filter.

31.7.76 High End Overlay Configuration Register 2

Name: LCDC_HEOCFG2

Address: 0xF0000394

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YPOS		

23	22	21	20	19	18	17	16
YPOS							

15	14	13	12	11	10	9	8
–	–	–	–	–	XPOS		

7	6	5	4	3	2	1	0
XPOS							

- **XPOS: Horizontal Window Position**

High End Overlay Horizontal window position.

- **YPOS: Vertical Window Position**

High End Overlay Vertical window position.

31.7.77 High End Overlay Configuration Register 3

Name: LCDC_HEOCFG3

Address: 0xF0000398

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YSIZE		

23	22	21	20	19	18	17	16
YSIZE							

15	14	13	12	11	10	9	8
–	–	–	–	–	XSIZE		

7	6	5	4	3	2	1	0
XSIZE							

- **XSIZE: Horizontal Window Size**

High End Overlay window width in pixels. The window width is set to (XSIZE + 1).

The following constraint must be met: XPOS + XSIZE ≤ PPL

- **YSIZE: Vertical Window Size**

High End Overlay window height in pixels. The window height is set to (YSIZE + 1).

The following constraint must be met: YPOS + YSIZE ≤ RPF

31.7.78 High End Overlay Configuration Register 4

Name: LCDC_HEOCFG4

Address: 0xF000039C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YMEMSIZE		

23	22	21	20	19	18	17	16
YMEMSIZE							

15	14	13	12	11	10	9	8
–	–	–	–	–	XMEMSIZE		

7	6	5	4	3	2	1	0
XMEMSIZE							

- **XMEMSIZE: Horizontal image Size in Memory**

High End Overlay image width in pixels. The image width is set to (XMEMSIZE + 1).

- **YMEMSIZE: Vertical image Size in Memory**

High End Overlay image height in pixels. The image height is set to (YMEMSIZE + 1).

31.7.79 High End Overlay Configuration Register 5

Name: LCDC_HEOCFG5

Address: 0xF00003A0

Access: Read/Write

31	30	29	28	27	26	25	24
XSTRIDE							

23	22	21	20	19	18	17	16
XSTRIDE							

15	14	13	12	11	10	9	8
XSTRIDE							

7	6	5	4	3	2	1	0
XSTRIDE							

- **XSTRIDE: Horizontal Stride**

XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

31.7.80 High End Overlay Configuration Register 6

Name: LCDC_HEOCFG6

Address: 0xF00003A4

Access: Read/Write

31	30	29	28	27	26	25	24
PSTRIDE							

23	22	21	20	19	18	17	16
PSTRIDE							

15	14	13	12	11	10	9	8
PSTRIDE							

7	6	5	4	3	2	1	0
PSTRIDE							

- **PSTRIDE: Pixel Stride**

PSTRIDE represents the memory offset, in bytes, between two pixels of the image memory.

31.7.81 High End Overlay Configuration Register 7

Name: LCDC_HEOCFG7

Address: 0xF00003A8

Access: Read/Write

31	30	29	28	27	26	25	24
UVXSTRIDE							

23	22	21	20	19	18	17	16
UVXSTRIDE							

15	14	13	12	11	10	9	8
UVXSTRIDE							

7	6	5	4	3	2	1	0
UVXSTRIDE							

- **UVXSTRIDE: UV Horizontal Stride**

UVXSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

31.7.82 High End Overlay Configuration Register 8

Name: LCDC_HEOCFG8

Address: 0xF00003AC

Access: Read/Write

31	30	29	28	27	26	25	24
UVPSTRIDE							

23	22	21	20	19	18	17	16
UVPSTRIDE							

15	14	13	12	11	10	9	8
UVPSTRIDE							

7	6	5	4	3	2	1	0
UVPSTRIDE							

- **UVPSTRIDE: UV Pixel Stride**

UVPSTRIDE represents the memory offset, in bytes, between two pixels of the image memory.

31.7.83 High End Overlay Configuration Register 9

Name: LCDC_HEOCFG9

Address: 0xF00003B0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RDEF							

15	14	13	12	11	10	9	8
GDEF							

7	6	5	4	3	2	1	0
BDEF							

- **RDEF: Red Default**

Default Red color when the High End Overlay DMA channel is disabled.

- **GDEF: Green Default**

Default Green color when the High End Overlay DMA channel is disabled.

- **BDEF: Blue Default**

Default Blue color when the High End Overlay DMA channel is disabled.

31.7.84 High End Overlay Configuration Register 10

Name: LCDC_HEOCFG10

Address: 0xF00003B4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RKEY							

15	14	13	12	11	10	9	8
GKEY							

7	6	5	4	3	2	1	0
BKEY							

- **RKEY: Red Color Component Chroma Key**

Reference Red chroma key used to match the Red color of the current overlay.

- **GKEY: Green Color Component Chroma Key**

Reference Green chroma key used to match the Green color of the current overlay.

- **BKEY: Blue Color Component Chroma Key**

Reference Blue chroma key used to match the Blue color of the current overlay.

31.7.85 High End Overlay Configuration Register 11

Name: LCDC_HEOCFG11

Address: 0xF00003B8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RMASK							

15	14	13	12	11	10	9	8
GMASK							

7	6	5	4	3	2	1	0
BMASK							

- **RMASK: Red Color Component Chroma Key Mask**

Red Mask used when the compare function is used. If a bit is set then this bit is compared.

- **GMASK: Green Color Component Chroma Key Mask**

Green Mask used when the compare function is used. If a bit is set then this bit is compared.

- **BMASK: Blue Color Component Chroma Key Mask**

Blue Mask used when the compare function is used. If a bit is set then this bit is compared.

31.7.86 High End Overlay Configuration Register 12

Name: LCDC_HEOCFG12

Address: 0xF00003BC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
GA							

15	14	13	12	11	10	9	8
–	–	–	VIDPRI	–	DSTKEY	REP	DMA

7	6	5	4	3	2	1	0
OVR	LAEN	GAEN	REVALPHA	ITER	ITER2BL	INV	CRKEY

- **CRKEY: Blender Chroma Key Enable**

0: Chroma key matching is disabled.

1: Chroma key matching is enabled.

- **INV: Blender Inverted Blender Output Enable**

0: Iterated pixel is the blended pixel.

1: Iterated pixel is the inverted pixel.

- **ITER2BL: Blender Iterated Color Enable**

0: Final adder stage operand is set to 0.

1: Final adder stage operand is set to the iterated pixel value.

- **ITER: Blender Use Iterated Color**

0: Pixel difference is set to 0.

1: Pixel difference is set to the iterated pixel value.

- **REVALPHA: Blender Reverse Alpha**

0: Pixel difference is multiplied by alpha.

1: Pixel difference is multiplied by 1 - alpha.

- **GAEN: Blender Global Alpha Enable**

0: Global alpha blending coefficient is disabled.

1: Global alpha blending coefficient is enabled.

- **LAEN: Blender Local Alpha Enable**

0: Local alpha blending coefficient is disabled.

1: Local alpha blending coefficient is enabled.

- **OVR: Blender Overlay Layer Enable**

0: Overlay pixel color is set to the default overlay pixel color.

1: Overlay pixel color is set to the DMA channel pixel color.

- **DMA: Blender DMA Layer Enable**

0: The default color is used on the Overlay 1 Layer.

1: The DMA channel retrieves the pixels stream from the memory.

- **REP: Use Replication logic to expand RGB color to 24 bits**

0: When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0.

1: When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb.

- **DSTKEY: Destination Chroma Keying**

0: Source Chroma keying is enabled.

1: Destination Chroma keying is used.

- **VIDPRI: Video Priority**

0: OVR1 layer is above HEO layer.

1: OVR1 layer is below HEO layer.

- **GA: Blender Global Alpha**

Global alpha blender for the current layer.

31.7.87 High End Overlay Configuration Register 13

Name: LCDC_HEOCFG13

Address: 0xF00003C0

Access: Read/Write

31	30	29	28	27	26	25	24
SCALEN	–	YFACTOR					

23	22	21	20	19	18	17	16
YFACTOR							

15	14	13	12	11	10	9	8
–	–	XFACTOR					

7	6	5	4	3	2	1	0
XFACTOR							

- **SCALEN: Hardware Scaler Enable**

0: Scaler is disabled

1: Scaler is enabled.

- **YFACTOR: Vertical Scaling Factor**

Scaler Vertical Factor.

- **XFACTOR: Horizontal Scaling Factor**

Scaler Horizontal Factor.

31.7.88 High End Overlay Configuration Register 14

Name: LCDC_HEOCFG14

Address: 0xF00003C4

Access: Read/Write

31	30	29	28	27	26	25	24
–	CSCYOFF	CSCRV					

23	22	21	20	19	18	17	16
CSCRV				CSCRU			

15	14	13	12	11	10	9	8
CSCRU						CSCRY	

7	6	5	4	3	2	1	0
CSCRY							

- **CSCRY: Color Space Conversion Y coefficient for Red Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCRU: Color Space Conversion U coefficient for Red Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCRV: Color Space Conversion V coefficient for Red Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCYOFF: Color Space Conversion Offset**

0: Offset is set to 0

1: Offset is set to 16

31.7.89 High End Overlay Configuration Register 15

Name: LCDC_HEOCFG15

Address: 0xF00003C8

Access: Read/Write

31	30	29	28	27	26	25	24
–	CSCUOFF	CSCGV					

23	22	21	20	19	18	17	16
CSCGV				CSCGU			

15	14	13	12	11	10	9	8
CSCGU						CSCGY	

7	6	5	4	3	2	1	0
CSCGY							

- **CSCGY: Color Space Conversion Y coefficient for Green Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCGU: Color Space Conversion U coefficient for Green Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCGV: Color Space Conversion V coefficient for Green Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCUOFF: Color Space Conversion Offset**

0: Offset is set to 0

1: Offset is set to 128

31.7.90 High End Overlay Configuration Register 16

Name: LCDC_HEOCFG16

Address: 0xF00003CC

Access: Read/Write

31	30	29	28	27	26	25	24
–	CSCVOFF	CSCBV					

23	22	21	20	19	18	17	16
CSCBV				CSCBU			

15	14	13	12	11	10	9	8
CSCBU						CSCBY	

7	6	5	4	3	2	1	0
CSCBY							

- **CSCBY: Color Space Conversion Y coefficient for Blue Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCBU: Color Space Conversion U coefficient for Blue Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCBV: Color Space Conversion V coefficient for Blue Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCVOFF: Color Space Conversion Offset**

0: Offset is set to 0

1: Offset is set to 128

31.7.91 High End Overlay Configuration Register 17

Name: LCDC_HEOCFG17

Address: 0xF00003D0

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI0COEFF3							

23	22	21	20	19	18	17	16
XPHI0COEFF2							

15	14	13	12	11	10	9	8
XPHI0COEFF1							

7	6	5	4	3	2	1	0
XPHI0COEFF0							

- **XPHI0COEFF0: Horizontal Coefficient for phase 0 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI0COEFF1: Horizontal Coefficient for phase 0 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI0COEFF2: Horizontal Coefficient for phase 0 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI0COEFF3: Horizontal Coefficient for phase 0 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.92 High End Overlay Configuration Register 18

Name: LCDC_HEOCFG18

Address: 0xF00003D4

Access: Read/Write

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
XPHI0COEFF4							

- **XPHI0COEFF4: Horizontal Coefficient for phase 0 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.93 High End Overlay Configuration Register 19

Name: LCDC_HEOCFG19

Address: 0xF00003D8

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI1COEFF3							

23	22	21	20	19	18	17	16
XPHI1COEFF2							

15	14	13	12	11	10	9	8
XPHI1COEFF1							

7	6	5	4	3	2	1	0
XPHI1COEFF0							

- **XPHI1COEFF0: Horizontal Coefficient for phase 1 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI1COEFF1: Horizontal Coefficient for phase 1 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI1COEFF2: Horizontal Coefficient for phase 1 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI1COEFF3: Horizontal Coefficient for phase 1 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.94 High End Overlay Configuration Register 20

Name: LCDC_HEOCFG20

Address: 0xF00003DC

Access: Read/Write

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
XPHI1COEFF4							

- **XPHI1COEFF4: Horizontal Coefficient for phase 1 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.95 High End Overlay Configuration Register 21

Name: LCDC_HEOCFG21

Address: 0xF00003E0

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI2COEFF3							

23	22	21	20	19	18	17	16
XPHI2COEFF2							

15	14	13	12	11	10	9	8
XPHI2COEFF1							

7	6	5	4	3	2	1	0
XPHI2COEFF0							

- **XPHI2COEFF0: Horizontal Coefficient for phase 2 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI2COEFF1: Horizontal Coefficient for phase 2 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI2COEFF2: Horizontal Coefficient for phase 2 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI2COEFF3: Horizontal Coefficient for phase 2 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.96 High End Overlay Configuration Register 22

Name: LCDC_HEOCFG22

Address: 0xF00003E4

Access: Read/Write

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
XPHI2COEFF4							

- **XPHI2COEFF4: Horizontal Coefficient for phase 2 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.97 High End Overlay Configuration Register 23

Name: LCDC_HEOCFG23

Address: 0xF00003E8

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI3COEFF3							

23	22	21	20	19	18	17	16
XPHI3COEFF2							

15	14	13	12	11	10	9	8
XPHI3COEFF1							

7	6	5	4	3	2	1	0
XPHI3COEFF0							

- **XPHI3COEFF0: Horizontal Coefficient for phase 3 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI3COEFF1: Horizontal Coefficient for phase 3 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI3COEFF2: Horizontal Coefficient for phase 3 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI3COEFF3: Horizontal Coefficient for phase 3 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.98 High End Overlay Configuration Register 24

Name: LCDC_HEOCFG24

Address: 0xF00003EC

Access: Read/Write

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
XPHI3COEFF4							

- **XPHI3COEFF4: Horizontal Coefficient for phase 3 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.99 High End Overlay Configuration Register 25

Name: LCDC_HEOCFG25

Address: 0xF00003F0

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI4COEFF3							

23	22	21	20	19	18	17	16
XPHI4COEFF2							

15	14	13	12	11	10	9	8
XPHI4COEFF1							

7	6	5	4	3	2	1	0
XPHI4COEFF0							

- **XPHI4COEFF0: Horizontal Coefficient for phase 4 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI4COEFF1: Horizontal Coefficient for phase 4 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI4COEFF2: Horizontal Coefficient for phase 4 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI4COEFF3: Horizontal Coefficient for phase 4 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.100High End Overlay Configuration Register 26

Name: LCDC_HEOCFG26

Address: 0xF00003F4

Access: Read/Write

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
XPHI4COEFF4							

- **XPHI4COEFF4: Horizontal Coefficient for phase 4 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.101High End Overlay Configuration Register 27

Name: LCDC_HEOCFG27

Address: 0xF00003F8

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI5COEFF3							
23	22	21	20	19	18	17	16
XPHI5COEFF2							
15	14	13	12	11	10	9	8
XPHI5COEFF1							
7	6	5	4	3	2	1	0
XPHI5COEFF0							

- **XPHI5COEFF0: Horizontal Coefficient for phase 5 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI5COEFF1: Horizontal Coefficient for phase 5 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI5COEFF2: Horizontal Coefficient for phase 5 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI5COEFF3: Horizontal Coefficient for phase 5 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.102High End Overlay Configuration Register 28

Name: LCDC_HEOCFG28

Address: 0xF00003FC

Access: Read/Write

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
XPHI5COEFF4							

- **XPHI5COEFF4: Horizontal Coefficient for phase 5 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.103High End Overlay Configuration Register 29

Name: LCDC_HEOCFG29

Address: 0xF0000400

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI6COEFF3							

23	22	21	20	19	18	17	16
XPHI6COEFF2							

15	14	13	12	11	10	9	8
XPHI6COEFF1							

7	6	5	4	3	2	1	0
XPHI6COEFF0							

- **XPHI6COEFF0: Horizontal Coefficient for phase 6 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI6COEFF1: Horizontal Coefficient for phase 6 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI6COEFF2: Horizontal Coefficient for phase 6 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI6COEFF3: Horizontal Coefficient for phase 6 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.104High End Overlay Configuration Register 30

Name: LCDC_HEOCFG30

Address: 0xF0000404

Access: Read/Write

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
XPHI6COEFF4							

- **XPHI6COEFF4: Horizontal Coefficient for phase 6 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.105High End Overlay Configuration Register 31

Name: LCDC_HEOCFG31

Address: 0xF0000408

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI7COEFF3							
23	22	21	20	19	18	17	16
XPHI7COEFF2							
15	14	13	12	11	10	9	8
XPHI7COEFF1							
7	6	5	4	3	2	1	0
XPHI7COEFF0							

- **XPHI7COEFF0: Horizontal Coefficient for phase 7 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI7COEFF1: Horizontal Coefficient for phase 7 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI7COEFF2: Horizontal Coefficient for phase 7 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI7COEFF3: Horizontal Coefficient for phase 7 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.106High End Overlay Configuration Register 32

Name: LCDC_HEOCFG32

Address: 0xF000040C

Access: Read/Write

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
XPHI7COEFF4							

- **XPHI7COEFF4: Horizontal Coefficient for phase 7 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.107High End Overlay Configuration Register 33

Name: LCDC_HEOCFG33

Address: 0xF0000410

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI0COEFF2							

15	14	13	12	11	10	9	8
YPHI0COEFF1							

7	6	5	4	3	2	1	0
YPHI0COEFF0							

- **YPHI0COEFF0: Vertical Coefficient for phase 0 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI0COEFF1: Vertical Coefficient for phase 0 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI0COEFF2: Vertical Coefficient for phase 0 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.108High End Overlay Configuration Register 34

Name: LCDC_HEOCFG34

Address: 0xF0000414

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI1COEFF2							

15	14	13	12	11	10	9	8
YPHI1COEFF1							

7	6	5	4	3	2	1	0
YPHI1COEFF0							

- **YPHI1COEFF0: Vertical Coefficient for phase 1 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI1COEFF1: Vertical Coefficient for phase 1 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI1COEFF2: Vertical Coefficient for phase 1 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.109High End Overlay Configuration Register 35

Name: LCDC_HEOCFG35

Address: 0xF0000418

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
YPHI2COEFF2							
15	14	13	12	11	10	9	8
YPHI2COEFF1							
7	6	5	4	3	2	1	0
YPHI2COEFF0							

- **YPHI2COEFF0: Vertical Coefficient for phase 2 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI2COEFF1: Vertical Coefficient for phase 2 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI2COEFF2: Vertical Coefficient for phase 2 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.110 High End Overlay Configuration Register 36

Name: LCDC_HEOCFG36

Address: 0xF000041C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI3COEFF2							

15	14	13	12	11	10	9	8
YPHI3COEFF1							

7	6	5	4	3	2	1	0
YPHI3COEFF0							

- **YPHI3COEFF0: Vertical Coefficient for phase 3 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI3COEFF1: Vertical Coefficient for phase 3 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI3COEFF2: Vertical Coefficient for phase 3 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.111High End Overlay Configuration Register 37

Name: LCDC_HEOCFG37

Address: 0xF0000420

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI4COEFF2							

15	14	13	12	11	10	9	8
YPHI4COEFF1							

7	6	5	4	3	2	1	0
YPHI4COEFF0							

- **YPHI4COEFF0: Vertical Coefficient for phase 4 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI4COEFF1: Vertical Coefficient for phase 4 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI4COEFF2: Vertical Coefficient for phase 4 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.112High End Overlay Configuration Register 38

Name: LCDC_HEOCFG38

Address: 0xF0000424

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI5COEFF2							

15	14	13	12	11	10	9	8
YPHI5COEFF1							

7	6	5	4	3	2	1	0
YPHI5COEFF0							

- **YPHI5COEFF0: Vertical Coefficient for phase 5 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI5COEFF1: Vertical Coefficient for phase 5 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI5COEFF2: Vertical Coefficient for phase 5 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.113High End Overlay Configuration Register 39

Name: LCDC_HEOCFG39

Address: 0xF0000428

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI6COEFF2							

15	14	13	12	11	10	9	8
YPHI6COEFF1							

7	6	5	4	3	2	1	0
YPHI6COEFF0							

- **YPHI6COEFF0: Vertical Coefficient for phase 6 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI6COEFF1: Vertical Coefficient for phase 6 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI6COEFF2: Vertical Coefficient for phase 6 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.114 High End Overlay Configuration Register 40

Name: LCDC_HEOCFG40

Address: 0xF000042C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI7COEFF2							

15	14	13	12	11	10	9	8
YPHI7COEFF1							

7	6	5	4	3	2	1	0
YPHI7COEFF0							

- **YPHI7COEFF0: Vertical Coefficient for phase 7 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI7COEFF1: Vertical Coefficient for phase 7 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI7COEFF2: Vertical Coefficient for phase 7 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

31.7.115High End Overlay Configuration Register 41

Name: LCDC_HEOCFG41

Address: 0xF0000430

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	YPHIDEF		

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	XPHIDEF		

- **XPHIDEF: Horizontal Filter Phase Offset**

XPHIDEF defines the index of the first coefficient set used when the horizontal resampling operation is started.

- **YPHIDEF: Vertical Filter Phase Offset**

XPHIDEF defines the index of the first coefficient set used when the vertical resampling operation is started.

31.7.116Base CLUT Register x

Name: LCDC_BASECLUTx [x=0..255]

Address: 0xF0000600

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RCLUT							

15	14	13	12	11	10	9	8
GCLUT							

7	6	5	4	3	2	1	0
BCLUT							

- **BCLUT: Blue Color Entry**

This field indicates the 8-bit width Blue color of the color lookup table.

- **GCLUT: Green Color Entry**

This field indicates the 8-bit width Green color of the color lookup table.

- **RCLUT: Red Color Entry**

This field indicates the 8-bit width Red color of the color lookup table.

31.7.117Overlay 1 CLUT Register x

Name: LCDC_OVR1CLUTx [x=0..255]

Address: 0xF0000A00

Access: Read/Write

31	30	29	28	27	26	25	24
ACLUT							

23	22	21	20	19	18	17	16
RCLUT							

15	14	13	12	11	10	9	8
GCLUT							

7	6	5	4	3	2	1	0
BCLUT							

- **BCLUT: Blue Color Entry**

This field indicates the 8-bit width Blue color of the color lookup table.

- **GCLUT: Green Color Entry**

This field indicates the 8-bit width Green color of the color lookup table.

- **RCLUT: Red Color Entry**

This field indicates the 8-bit width Red color of the color lookup table.

- **ACLUT: Alpha Color Entry**

This field indicates the 8-bit width Alpha channel of the color lookup table.

31.7.118 High End Overlay CLUT Register x

Name: LCDC_HEOCLUTx [x=0..255]

Address: 0xF0001200

Access: Read/Write

31	30	29	28	27	26	25	24
ACLUT							

23	22	21	20	19	18	17	16
RCLUT							

15	14	13	12	11	10	9	8
GCLUT							

7	6	5	4	3	2	1	0
BCLUT							

- **BCLUT: Blue Color Entry**

This field indicates the 8-bit width Blue color of the color lookup table.

- **GCLUT: Green Color Entry**

This field indicates the 8-bit width Green color of the color lookup table.

- **RCLUT: Red Color Entry**

This field indicates the 8-bit width Red color of the color lookup table.

- **ACLUT: Alpha Color Entry**

This field indicates the 8-bit width Alpha channel of the color lookup table.

32. Video Decoder (VDEC)

32.1 Description

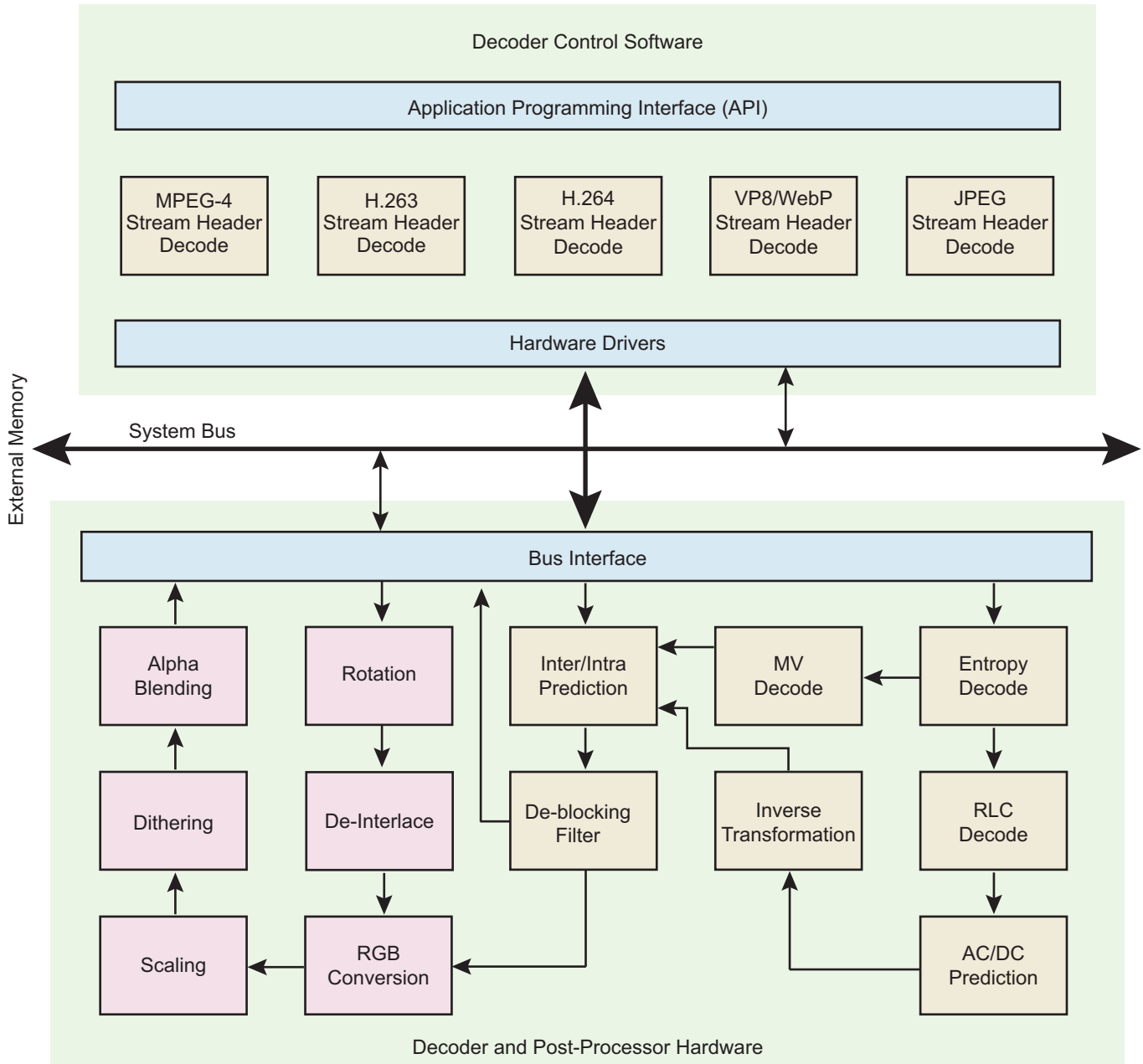
The multi-format Video Decoder (VDEC) enables compression or decompression of digital video signals for the formats MPEG-4, H.264, H.263, VP8, and JPEG. A prior introduction to these format standards is recommended and helpful in understanding the functionality of the decoder.

32.2 Embedded Characteristics

- Supported standards:
  - MPEG-4 Simple Profile, Levels 0, 0B, 1, 2, 3, 4a, 5, 6
  - MPEG-4 Advanced Simple Profile, Levels 0, 1, 2, 3, 3b, 4, 5
  - H.264 Baseline Profile, Main Profile, High Profile, Levels 1–4.2
  - H.263 Profile 0, Levels 10–70
  - VP8/WebP Decoding, Versions 0–3
  - JPEG decoding, Baseline, Interleaved, Versions 0–3
- Supports Power Management
- Little-endian and Big-endian Support for Stream Data

32.3 Block Diagram

Figure 32-1. Block Diagram



32.4 Decoding Features

32.4.1 H.264 Features

Table 32-1. H.264 Features

Feature	Decoder Support
Input data format	H.264 byte or NAL unit stream/SVC stream/MVC stream
Decoding scheme	Frame by frame (or field by field) Slice by slice
Output data format	YCbCr 4:2:0 semi-planar raster scan(1) YCbCr 4:2:0 semi-planar 8x4 tiled(2) YCbCr 4:0:0 (monochrome)
Supported image size	48 x 48 to 1280 x 720 Step size 16 pixels(3)
Maximum frame rate	30 fps at 1280 x 720(4)
Maximum bit rate	62.5 Mbps, as specified by H.264 High Profile level 4.2(4)
Error detection and concealment	Supported

Notes:
1. In semi-planar raster scan format the Cb and Cr components are interleaved pixel by pixel in a separate plane. This allows more efficient bus usage compared to the planar YCbCr format due to longer bursts that can be used in chrominance data transferring.
2. The semi-planar 8x4 tiled format is otherwise similar to the raster-scan format, but the data is organized in tiles. The tile size is 8x4 pixels in both luminance and chrominance planes. The tiled mode allows more efficient bus usage compared to the raster-scan format due to longer bursts that can be used in all picture data transferring. Tiled mode is not supported in JPEG decoding.
3. Decoder performs cropping for video fields that can be multiple of eight pixels in vertical direction.
4. Actual maximum frame rate and bit rate will depend on the logic clock frequency and system bus performance.

32.4.2 MPEG-4/H.263 Features

Table 32-2. MPEG-4/H.263 Features

Feature	Decoder Support
Input data format	MPEG-4/H.263
Decoding scheme	Frame by frame (or field by field) Video packet by video packet
Output data format	YCbCr 4:2:0 semi-planar raster scan YCbCr 4:2:0 semi-planar 8x4 tiled
Supported image size	48 x 48 to 1280 x 720 (MPEG-4) 48 x 48 to 720 x 576 (H.263) Step size 16 pixels
Maximum frame rate	30 fps at 1280 x 720(1)
Maximum bit rate	As specified by MPEG-4 ASP level 5
Error detection and concealment	Supported

Notes:
1. Actual maximum frame rate and bit rate will depend on the logic clock frequency and system bus performance.

32.4.3 JPEG Features

Table 32-3. JPEG Features

Feature	Decoder Support
Input data format	JFIF file format 1.02 YCbCr 4:0:0, 4:2:0, 4:2:2, 4:4:0, 4:1:1 and 4:4:4 sampling formats
Decoding scheme	Input: buffer by buffer, from 5kB to 8MB at a time(1) Output: from 1MB row to 16 Mpixels at a time(2)
Output data format	YCbCr 4:0:0, 4:2:0, 4:2:2, 4:4:0, 4:1:1 and 4:4:4 semi-planar raster-scan
Supported image size	48x48 to 16368 x 16368 (256 Mpixels) Step size 8 pixels(3)
Maximum data rate	Up to 76 million pixels per second (4)
Thumbnail decoding	JPEG compressed thumbnails supported
Error detection	Supported

Notes:
1. Programmable buffer size for optimizing performance and memory consumption. Interrupt will be issued when buffer runs empty, and the control software will load more stream to external memory.
2. Programmable output slice size for optimizing performance and memory consumption. Interrupt will be issued when the requested area decoded. The control software can be used to switch the decoder output picture base address each time.
3. Non-8x8 dividable resolutions will be filled to 8 pixel boundary.
4. Actual maximum data rate will depend on the logic clock frequency and JPEG compression rate. The given figure applies to high-quality JPEG with logic running at 200 MHz.

32.4.4 VP8/WebP Features

Table 32-4. VP8/WebP Features

Feature	Decoder Support
Input data format	VP8 stream
Decoding scheme	Frame by frame
Output data format	YCbCr 4:2:0 semi-planar raster scan YCbCr 4:2:0 semi-planar 8x4 tiled
Supported image size	48 x 48 to 1280 x 720 Step size 16 pixels
Maximum frame rate	30 fps at 1280 x 720(1)
Maximum bit rate	As specified by VP8 specification
Error detection and concealment	Supported

Notes:
1. Actual maximum frame rate and bit rate will depend on the logic clock frequency and system bus performance.

32.5 Post-Processing Features

Table 32-5. Post-Processing Features

Feature	Post-Processor Support
Input data format	Any format generated by the decoder in combined mode In standalone mode: - YCbCr 4:2:0 semi-planar raster-scan - YCbCr 4:2:0 semi-planar tiled (16x16) - YCbCr 4:2:0 planar - YCbYCr 4:2:2, YCrYCb 4:2:2 - CbYCrY 4:2:2, CrYCbY 4:2:2
Post-processing scheme	Frame by frame. Post-processor handles the image macroblock by macroblock, also in standalone mode.
Input image source	Internal source (combined mode): VDEC decoder External source (standalone mode): e.g., a software decoder or camera interface
Output data format	YCbCr 4:2:0 semi-planar YCbYCr 4:2:2 raster-scan or 4x4 tiled YCrYCb 4:2:2 raster-scan or 4x4 tiled CbYCrY 4:2:2 raster-scan or 4x4 tiled CrYCbY 4:2:2 raster-scan or 4x4 tiled Fully configurable ARGB channel lengths and locations inside 32 bits, e.g., ARGB 32-bit (8-8-8-8), RGB 16-bit (5-6-5), ARGB 16-bit (4-4-4-4)
Input image size (combined mode)	48 x 48 to 1280 x 720 Step size 16 pixels
Input image size (stand-alone mode)	48 x 48 to 1280 x 720 Step size 16 pixels
Output image size	16 x 16 to 1280 x 720 Horizontal step size 8 Vertical step size 2
Image up-scaling	Bicubic polynomial interpolation with a four-tap horizontal kernel and a two-tap vertical kernel Arbitrary, non-integer scaling ratio separately for both dimensions Maximum output width is 3x the input width (within the maximum output image size limit) Maximum output height is 3x the input height - 2 pixels (within the maximum output image size limit)
Image down-scaling	Proprietary averaging filter Arbitrary, non-integer scaling ratio separately for both dimensions Unlimited down-scaling ratio (e.g., from 16 Mpixels to QVGA)
YCbCr to RGB color conversion	BT.601-5 compliant BT.709 compliant User definable conversion coefficient
Dithering	2x2 ordered dithering for 4-, 5- and 6-bit RGB channel precision
Deinterlacing (2)	Conditional spatial deinterlace filtering. Supports only YCbCr 4:2:0 input format.

Table 32-5. Post-Processing Features (Continued)

Feature	Post-Processor Support
Programmable alpha channel	Constant eight bit value can be set to the alpha channel of the 24-bit RGB output data to control the transparency of the output picture. The resulting 32-bit ARGB data can be used as input data for later alpha blending.
Alpha blending	Output image can be alpha blended with two rectangular areas.(1) YCbCr 4:2:0 PP output format is not supported when performing alpha blending. The supported overlay input formats are following: - alpha value + YCbCr 4:4:4, 8 bit each. - 8 bit alpha value + 24 bit RGB Alpha blending input data can be cropped from a larger area by defining the scan line length.
RGB image contrast adjustment	Segmented linear
RGB image brightness adjustment	Linear
RGB image color saturation adjustment	Linear
De-blocking filter for MPEG-4 simple profile/H.263	Using a modified H.264 in-loop filter as a post-processing filter. Filtering has to be performed in combined mode.
Image cropping/digital zoom	User definable start position, height and width. Can be used with scaling to perform digital zoom. Usable only for JPEG or standalone mode.
Picture in picture	Output image can be written to any location inside video memory (1)
Supported display size for picture in picture	Up to 1280 x 720
Output image masking	Output image writing can be prevented on two rectangular areas in the image (3). The masking feature is exclusive with alpha blending however it is possible to have one masking area and one blending area.
Image rotation(2)	Rotation 90, 180 or 270 degrees Horizontal and Vertical flip

Notes:
1. Step size in the start and end coordinates depends on the output picture format and the width of the data bus.
2. Usable in stand-alone post-processing mode only. While performing this operation, any other operation, such as color conversion or scaling, can be performed at the same time.
3. It is not allowed to perform horizontal up-scaling and vertical down-scaling (or vice versa) at the same. If needed, this kind of operation can be performed in two phases.

32.6 H.264/MVC Decoder

The decoder has two operating modes:

- Primary—Hardware performs entropy decoding
- Secondary—Software performs entropy decoding; used in H.264 ASO or Slice Group stream decoding

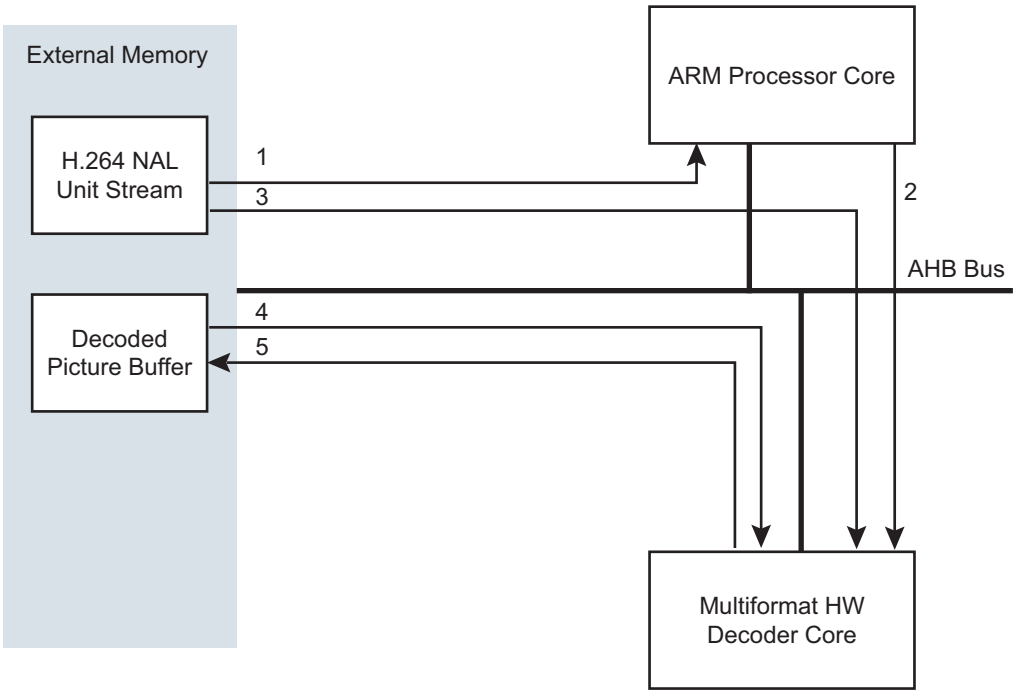
32.6.1 Decoder Data Flow, Hardware Performs Entropy Decoding

Numbered references in the following text are illustrated in Figure 32-2.

The decoder software starts decoding the first picture by parsing the stream headers (**1**). Software then setups the hardware control registers (picture size, stream start address, etc.) and enables the hardware (**2**).

Hardware decodes the picture by reading stream, VLC and QP tables for JPEG (**3**), and the reference pictures (**4**) (required for inter picture decoding) from the external memory. Hardware writes the decoded output picture to memory (**5**) one macroblock at a time. When the picture has been fully decoded, or the hardware has run out of stream data, it gives an interrupt with a proper status flag and provides stream end address for software to continue.

Figure 32-2. Multi-format Decoder and External Memory Data Flow in VLC Mode



32.6.2 Decoder Data Flow, Software Performs Entropy Decoding (RLC Mode)

Numbered references in the following text are illustrated in Figure 32-3.

In this case, the decoder software starts decoding the first picture by parsing the stream headers (**1**), and by performing entropy decoding. Software then writes the following items to external memory:

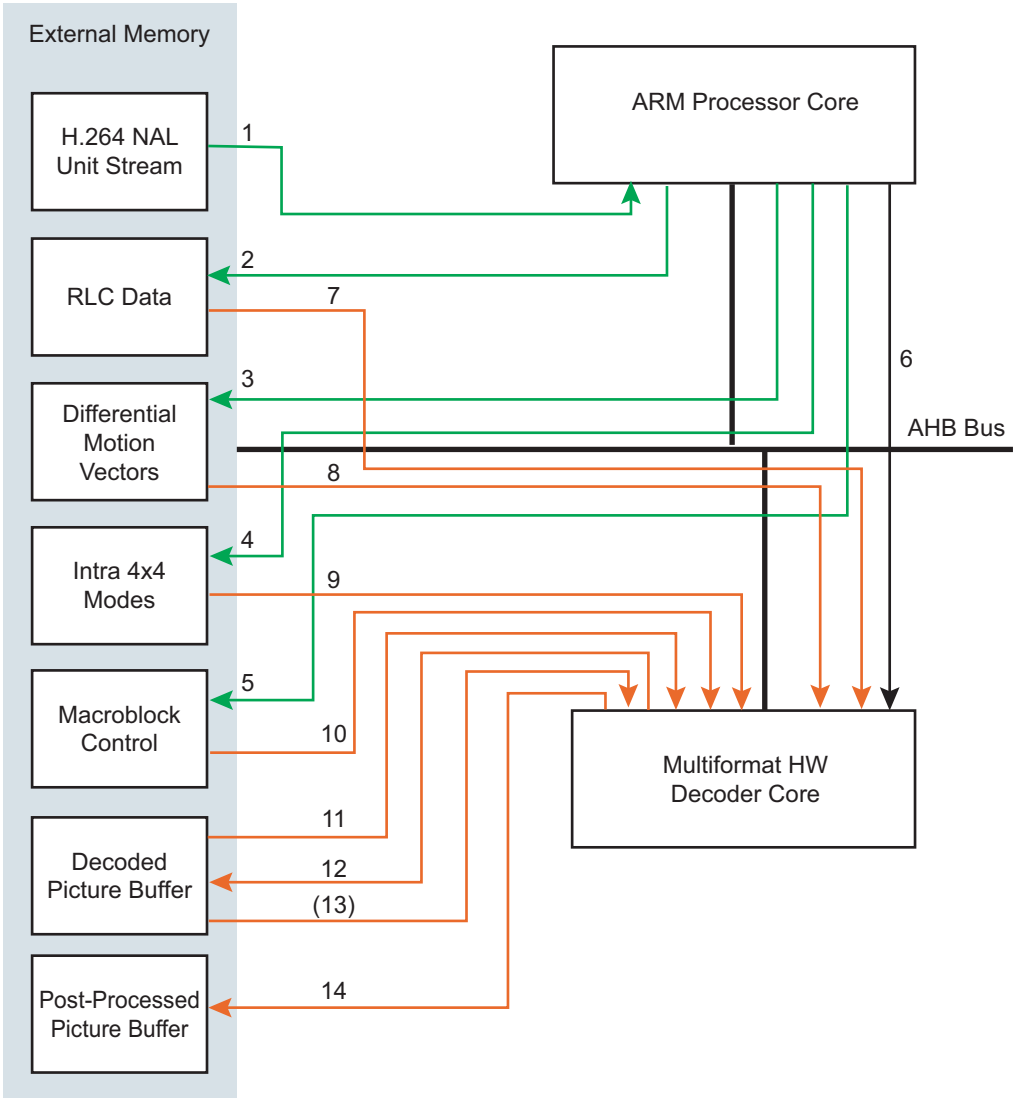
- Run-length-coded (RLC) data (**2**)
- Differential motion vectors (**3**)
- Intra 4x4 prediction modes (**4**)
- Macroblock control data (**5**)

Last step for the software is to write the hardware control registers and to enable the hardware (**6**).

Hardware decodes the picture by buffering control data for several macroblocks at a time, and then reads appropriate amount of RLC data, differential motion vectors and intra modes depending on each macroblock type (**7**)–(**10**). Hardware will also read the reference pictures (previously decoded pictures) as required (**11**). Hardware writes decoded (in-loop filtered, if H.264) output picture to memory one macroblock at a time (**12**). When the picture has been fully decoded, hardware can raise an interrupt and write the status bits in the status register.

If post-processing is enabled, one or two additional image transfer operations will take place. If the decoded images are in display order (i.e., no picture re-ordering has been made when encoding the sequence), and rotation is not used, PP will process the pictures in pipeline with the decoder. Otherwise, it will first have to read the decoded image that is to be displayed next from the memory (**13**), and then write back the processed image (**14**).

Figure 32-3. Multi-format Decoder and External Memory Data Flow in RLC Mode



32.7 MPEG-4/H.263 Decoder

The decoder has two operating modes:

- Primary—Hardware performs entropy decoding
- Secondary—Software performs entropy decoding; used in MPEG-4 data partitioned stream decoding

32.7.1 Decoder Data Flow, Hardware Performs Entropy Decoding

Numbered references in the following text are illustrated in Figure 32-2.

The decoder software starts decoding the first picture by parsing the stream headers (**1**). Software then setups the hardware control registers (picture size, stream start address, etc.) and enables the hardware (**2**).

Hardware decodes the picture by reading stream, VLC and QP tables for JPEG (**3**), and the reference pictures (**4**) (required for inter picture decoding) from the external memory. For B-pictures another reference picture may be required. Hardware writes the decoded output picture to memory (**5**) one macroblock at a time. When the picture has been fully decoded, or the hardware has run out of stream data, it gives an interrupt with a proper status flag and provides stream end address for software to continue.

32.7.2 Decoder Data Flow, Software Performs Entropy Decoding

Numbered references in the following text are illustrated in Figure 32-3.

In this case, the decoder software starts decoding the first picture by parsing the stream headers (**1**), and by performing entropy decoding. Software then writes the following items to external memory:

- Run-length-coded (RLC) data (**2**)
- Differential motion vectors (**3**)
- separate DC coefficients (in MPEG-4, if the stream is using data partitioning) (**4**)
- Macroblock control data (**5**)

Last step for the software is to write the hardware control registers and to enable the hardware (**6**).

Hardware decodes the picture by buffering control data for several macroblocks at a time, and reading then appropriate amount of AC and DC RLC data and differential motion vectors depending on the macroblock type (**7**)–(**10**). Hardware will also read the reference pictures (previously decoded pictures) as required (**11**). Hardware writes decoded (in-loop filtered, if H.264) output picture to memory one macroblock at a time (**12**). When the picture has been fully decoded, hardware can raise an interrupt and write the status bits in the status register.

If post-processing is enabled, one or two additional image transfer operations will take place. If the decoded images are in display order (i.e., no picture re-ordering has been made when encoding the sequence), and rotation is not used, PP will process the pictures in pipeline with the decoder. Otherwise, it will first have to read the decoded image that is to be displayed next from the memory (**13**), and then write back the processed image (**14**).

32.8 Functionality of the JPEG Decoder

The decoder software starts decoding the picture by parsing the stream headers and then writes the following items to external memory:

- VLC tables
- Quantization tables

Last step for the software is to write the hardware control registers and to enable the hardware. After starting hardware, software waits interrupt from hardware.

Hardware decodes the picture by reading stream, VLC and QP tables. Hardware writes the decoded output picture to memory one macroblock at a time. When the picture has been fully decoded, or the hardware has run out of stream data, it gives an interrupt with a proper status flag and provides stream end address for software to continue and returns to initial state.

The JPEG decoder supports slice mode decoding. In slice mode, software passes information to hardware of how many macro block rows should be decoded in one slice. After slice decoding, hardware will raise a slice ready interrupt. The software manages of output buffer updates.

JPEG can also be decoded by buffering the input stream. When the stream in a buffer is processed, hardware will raise an interrupt. The user needs to load the input buffer and call software again.

32.9 Functionality of the VP8/WebP Decoder

The decoder software starts decoding the first picture by parsing the stream headers.

Software then sets up the hardware control registers (picture size, stream start address, etc.) and enables the hardware.

Hardware decodes the picture by reading stream and the reference picture (required for inter-picture decoding) from the external memory. Hardware writes the decoded output picture to memory one macroblock at a time. When the picture has been fully decoded, or the hardware has run out of stream data, it gives an interrupt with a proper status flag and provides stream end address for software to continue and returns to initial state.

Note: WebP stream contains no reference pictures.

32.10 Functionality of the Post-processor

All post-processing functions are fully hardware implemented. The control software part provides an API for using the PP in applications and translates the user parameters to the hardware interface.

The PP can process images from an external source (standalone mode), or it can be initialized to a combined processing mode with the decoder. In combined mode the PP software internally communicates with the proper decoder library. This communication is hidden from the API user, who only has to setup the PP to work in combined mode.

32.11 Product Dependencies

32.11.1 Power Management

The Video Decoder requires a peripheral clock. The user has to enable the VDEC peripheral clock by setting the corresponding PIDx bit in the PMC Peripheral Clock Enable Register (PMC_PCER).

Software can reset the hardware synchronically by writing separate decoder and post-processor enable bits to zero. These enable bits are located in the memory-mapped registers and they can be used for terminating or restarting the decoding or post-processing at any time.

32.11.2 Interrupt Sources

The Video Decoder has an interrupt line connected to the interrupt controller. The interrupt controller must be programmed prior to handling Video Decoder interrupts.

33. Image Sensor Interface (ISI)

33.1 Description

The Image Sensor Interface (ISI) connects a CMOS-type image sensor to the processor and provides image capture in various formats.The ISI performs data conversion, if necessary, before the storage in memory through DMA.

The ISI supports color CMOS image sensor and grayscale image sensors with a reduced set of functionalities.

In Grayscale mode, the data stream is stored in memory without any processing and so is not compatible with the LCD controller.

Internal FIFOs on the preview and codec paths are used to store the incoming data. The RGB output on the preview path is compatible with the LCD controller. This module outputs the data in RGB format (LCD compatible) and has scaling capabilities to make it compliant to the LCD display resolution (refer to Table 33-5).

Several input formats such as preprocessed RGB or YCbCr are supported through the data bus interface.

The ISI supports two synchronization modes:

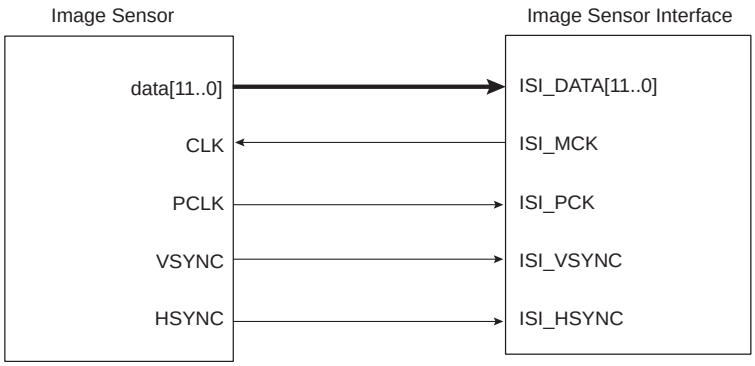
- Hardware with ISI_VSYNC and ISI_HSYNC signals
- International Telecommunication Union Recommendation *ITU-R BT.656-4* Start-of-Active-Video (SAV) and End-of-Active-Video (EAV) synchronization sequence

Using EAV/SAV for synchronization reduces the pin count (ISI_VSYNC, ISI_HSYNC not used). The polarity of the synchronization pulse is programmable to comply with the sensor signals.

Table 33-1. I/O Description

Signal	Direction	Description
ISI_VSYNC	In	Vertical Synchronization
ISI_HSYNC	In	Horizontal Synchronization
ISI_DATA[11..0]	In	Sensor Pixel Data
ISI_MCK	Out	Master Clock provided to the Image Sensor. Refer to Section 33.5.3 “Clocks”.
ISI_PCK	In	Pixel Clock provided by the Image Sensor

Figure 33-1. ISI Connection Example

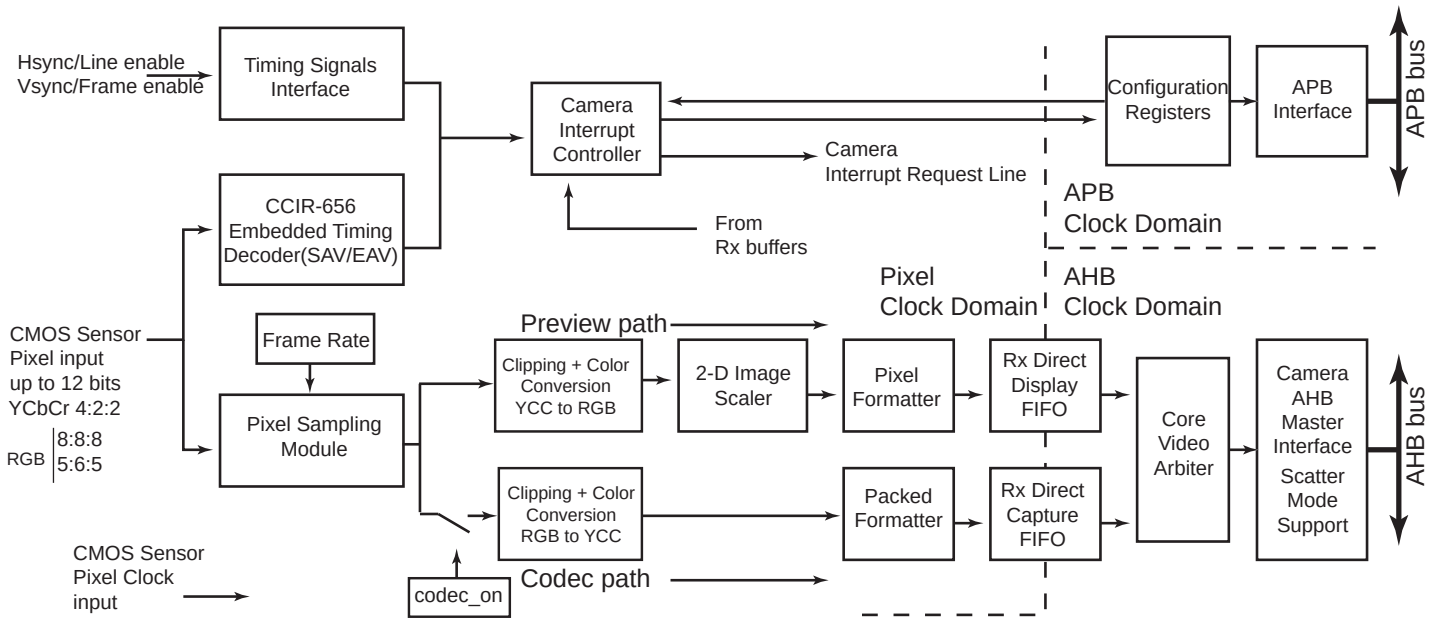


33.2 Embedded Characteristics

- ITU-R BT. 601/656 8-bit Mode External Interface Support
- Supports up to 12-bit Grayscale CMOS Sensors
- Support for ITU-R BT.656-4 SAV and EAV Synchronization
- Vertical and Horizontal Resolutions up to 2048 × 2048
- Preview Path up to 640 × 480 in RGB Mode
- Codec Path up to 2048 × 2048
- 32-byte FIFO on Codec Path
- 32-byte FIFO on Preview Path
- Support for Packed Data Formatting for YCbCr 4:2:2 Formats
- Preview Scaler to Generate Smaller Size image
- Programmable Frame Capture Rate
- VGA, QVGA, CIF, QCIF Formats Supported for LCD Preview
- Custom Formats with Horizontal and Vertical Preview Size as Multiples of 16 Also Supported for LCD Preview

33.3 Block Diagram

Figure 33-2. ISI Block Diagram



33.4 Product Dependencies

33.4.1 I/O Lines

The pins used for interfacing the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the ISI pins to their peripheral functions.

Table 33-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
ISI	ISI_D0	PC19	A
ISI	ISI_D1	PC20	A
ISI	ISI_D2	PC21	A
ISI	ISI_D3	PC22	A
ISI	ISI_D4	PC23	A
ISI	ISI_D5	PC24	A
ISI	ISI_D6	PC25	A
ISI	ISI_D7	PC26	A
ISI	ISI_D8	PC0	C
ISI	ISI_D9	PC1	C
ISI	ISI_D10	PC2	C
ISI	ISI_D11	PC3	C
ISI	ISI_HSYNC	PB4	C
ISI	ISI_PCK	PB1	C
ISI	ISI_VSYNC	PB3	C

33.4.2 Power Management

The ISI can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the ISI clock.

33.4.3 Interrupt Sources

The ISI interface has an interrupt line connected to the interrupt controller. Handling the ISI interrupt requires programming the interrupt controller before configuring the ISI.

Table 33-3. Peripheral IDs

Instance	ID
ISI	52

33.5 Functional Description

The Image Sensor Interface (ISI) supports direct connection to the ITU-R BT. 601/656 8-bit mode compliant sensors and up to 12-bit grayscale sensors. It receives the image data stream from the image sensor on the 12-bit data bus.

This module receives up to 12 bits for data, the horizontal and vertical synchronizations and the pixel clock. The reduced pin count alternative for synchronization is supported for sensors that embed SAV (start of active video) and EAV (end of active video) delimiters in the data stream.

The Image Sensor Interface interrupt line is connected to the Advanced Interrupt Controller and can trigger an interrupt at the beginning of each frame and at the end of a DMA frame transfer. If the SAV/EAV synchronization is used, an interrupt can be triggered on each delimiter event.

For 8-bit color sensors, the data stream received can be in several possible formats: YCbCr 4:2:2, RGB 8:8:8, RGB 5:6:5 and may be processed before the storage in memory. When the preview DMA channel is configured and enabled, the preview path is activated and an 'RGB frame' is moved to memory. The preview path frame rate is configured with the FRATE field of the ISI_CFG1 register. When the codec DMA channel is configured and enabled, the codec path is activated and a 'YCbCr 4:2:2 frame' is captured as soon as the ISI_CDC bit of the ISI Control Register (ISI_CR) is set.

When the FULL bit of the ISI_CFG1 register is set, both preview DMA channel and codec DMA channel can operate simultaneously. When a zero is written to the FULL bit of the ISI_CFG1 register, a hardware scheduler checks the FRATE field. If its value is zero, a preview frame is skipped and a codec frame is moved to memory instead. If its value is other than zero, at least one free frame slot is available. The scheduler postpones the codec frame to that free available frame slot.

The data stream may be sent on both preview path and codec path if the value of bit ISI_CDC in the ISI_CR is one. To optimize the bandwidth, the codec path should be enabled only when a capture is required.

In Grayscale mode, the input data stream is stored in memory without any processing. The 12-bit data, which represent the grayscale level for the pixel, is stored in memory one or two pixels per word, depending on the GS_MODE bit in the ISI_CFG2 register. The codec datapath is not available when grayscale image is selected.

A frame rate counter allows users to capture all frames or 1 out of every 2 to 8 frames.

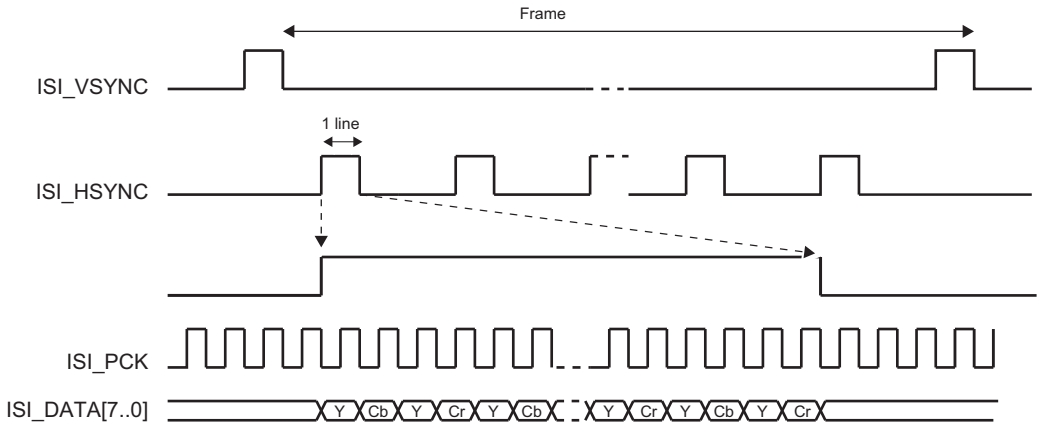
33.5.1 Data Timing

33.5.1.1 VSYNC/HSYNC Data Timing

In the VSYNC/HSYNC synchronization, the valid data is captured with the active edge of the pixel clock (ISI_PCK), after SFD lines of vertical blanking and SLD pixel clock periods delay programmed in the ISI_CR.

The data timing using horizontal and vertical synchronization are shown in Figure 33-4.

Figure 33-3. HSYNC and VSYNC Synchronization



33.5.1.2 SAV/EAV Data Timing

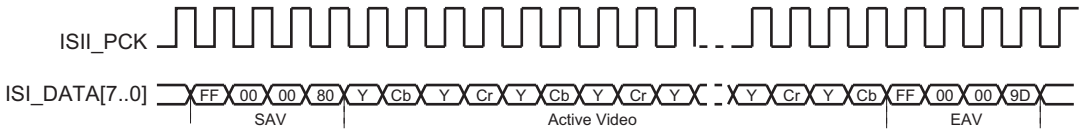
The ITU-RBT.656-4 standard defines the functional timing for an 8-bit wide interface.

There are two timing reference signals, one at the beginning of each video data block SAV (0xFF000080) and one at the end of each video data block EAV (0xFF00009D). Only data sent between EAV and SAV is captured. Horizontal blanking and vertical blanking are ignored. Use of the SAV and EAV synchronization eliminates the

ISI_VSYNC and ISI_HSYNC signals from the interface, thereby reducing the pin count. In order to retrieve both frame and line synchronization properly, at least one line of vertical blanking is mandatory.

The data timing using EAV/SAV sequence synchronization are shown in Figure 33-4.

Figure 33-4. SAV and EAV Sequence Synchronization



33.5.2 Data Ordering

The RGB color space format is required for viewing images on a display screen preview, and the YCbCr color space format is required for encoding.

All the sensors do not output the YCbCr or RGB components in the same order. The ISI allows the user to program the same component order as the sensor, reducing software treatments to restore the right format.

Table 33-4. Data Ordering in YCbCr Mode

Mode	Byte 0	Byte 1	Byte 2	Byte 3
Default	Cb(i)	Y(i)	Cr(i)	Y(i+1)
Mode 1	Cr(i)	Y(i)	Cb(i)	Y(i+1)
Mode 2	Y(i)	Cb(i)	Y(i+1)	Cr(i)
Mode 3	Y(i)	Cr(i)	Y(i+1)	Cb(i)

Table 33-5. RGB Format in Default Mode, RGB_CFG = 00, No Swap

Mode	Byte	D7	D6	D5	D4	D3	D2	D1	D0
RGB 8:8:8	Byte 0	R7(i)	R6(i)	R5(i)	R4(i)	R3(i)	R2(i)	R1(i)	R0(i)
	Byte 1	G7(i)	G6(i)	G5(i)	G4(i)	G3(i)	G2(i)	G1(i)	G0(i)
	Byte 2	B7(i)	B6(i)	B5(i)	B4(i)	B3(i)	B2(i)	B1(i)	B0(i)
	Byte 3	R7(i+1)	R6(i+1)	R5(i+1)	R4(i+1)	R3(i+1)	R2(i+1)	R1(i+1)	R0(i+1)
RGB 5:6:5	Byte 0	R4(i)	R3(i)	R2(i)	R1(i)	R0(i)	G5(i)	G4(i)	G3(i)
	Byte 1	G2(i)	G1(i)	G0(i)	B4(i)	B3(i)	B2(i)	B1(i)	B0(i)
	Byte 2	R4(i+1)	R3(i+1)	R2(i+1)	R1(i+1)	R0(i+1)	G5(i+1)	G4(i+1)	G3(i+1)
	Byte 3	G2(i+1)	G1(i+1)	G0(i+1)	B4(i+1)	B3(i+1)	B2(i+1)	B1(i+1)	B0(i+1)

Table 33-6. RGB Format, RGB_CFG = 10 (Mode 2), No Swap

Mode	Byte	D7	D6	D5	D4	D3	D2	D1	D0
RGB 5:6:5	Byte 0	G2(i)	G1(i)	G0(i)	R4(i)	R3(i)	R2(i)	R1(i)	R0(i)
	Byte 1	B4(i)	B3(i)	B2(i)	B1(i)	B0(i)	G5(i)	G4(i)	G3(i)
	Byte 2	G2(i+1)	G1(i+1)	G0(i+1)	R4(i+1)	R3(i+1)	R2(i+1)	R1(i+1)	R0(i+1)
	Byte 3	B4(i+1)	B3(i+1)	B2(i+1)	B1(i+1)	B0(i+1)	G5(i+1)	G4(i+1)	G3(i+1)

Table 33-7. RGB Format in Default Mode, RGB_CFG = 00, Swap Activated

Mode	Byte	D7	D6	D5	D4	D3	D2	D1	D0
RGB 8:8:8	Byte 0	R0(i)	R1(i)	R2(i)	R3(i)	R4(i)	R5(i)	R6(i)	R7(i)
	Byte 1	G0(i)	G1(i)	G2(i)	G3(i)	G4(i)	G5(i)	G6(i)	G7(i)
	Byte 2	B0(i)	B1(i)	B2(i)	B3(i)	B4(i)	B5(i)	B6(i)	B7(i)
	Byte 3	R0(i+1)	R1(i+1)	R2(i+1)	R3(i+1)	R4(i+1)	R5(i+1)	R6(i+1)	R7(i+1)
RGB 5:6:5	Byte 0	G3(i)	G4(i)	G5(i)	R0(i)	R1(i)	R2(i)	R3(i)	R4(i)
	Byte 1	B0(i)	B1(i)	B2(i)	B3(i)	B4(i)	G0(i)	G1(i)	G2(i)
	Byte 2	G3(i+1)	G4(i+1)	G5(i+1)	R0(i+1)	R1(i+1)	R2(i+1)	R3(i+1)	R4(i+1)
	Byte 3	B0(i+1)	B1(i+1)	B2(i+1)	B3(i+1)	B4(i+1)	G0(i+1)	G1(i+1)	G2(i+1)

The RGB 5:6:5 input format is processed to be displayed as RGB 5:6:5 format, compliant with the 16-bit mode of the LCD controller.

33.5.3 Clocks

The sensor master clock (ISI_MCK) can be generated either by the Advanced Power Management Controller (APMC) through a Programmable Clock output or by an external oscillator connected to the sensor.

None of the sensors embed a power management controller, so providing the clock by the APMC is a simple and efficient way to control power consumption of the system.

Care must be taken when programming the system clock. The ISI has two clock domains, the sensor master clock and the pixel clock provided by sensor. The two clock domains are not synchronized, but the sensor master clock must be faster than the pixel clock.

33.5.4 Preview Path

33.5.4.1 Scaling, Decimation (Subsampling)

This module resizes captured 8-bit color sensor images to fit the LCD display format. The resize module performs only downscaling. The same ratio is applied for both horizontal and vertical resize, then a fractional decimation algorithm is applied.

The decimation factor is a multiple of 1/16; values 0 to 15 are forbidden.

Table 33-8. Decimation Factor

Decimation Value	0–15	16	17	18	19	...	124	125	126	127
Decimation Factor	—	1	1.063	1.125	1.188	...	7.750	7.813	7.875	7.938

Table 33-9. Decimation and Scaler Offset Values

OUTPUT	INPUT	352 × 288	640 × 480	800 × 600	1280 × 1024	1600 × 1200	2048 × 1536
VGA 640 × 480	F	—	16	20	32	40	51
QVGA 320 × 240	F	16	32	40	64	80	102
CIF 352 × 288	F	16	26	33	56	66	85

Table 33-9. Decimation and Scaler Offset Values (Continued)

OUTPUT	INPUT	352 × 288	640 × 480	800 × 600	1280 × 1024	1600 × 1200	2048 × 1536
QCIF 176 × 144	F	32	53	66	113	133	170

Example:

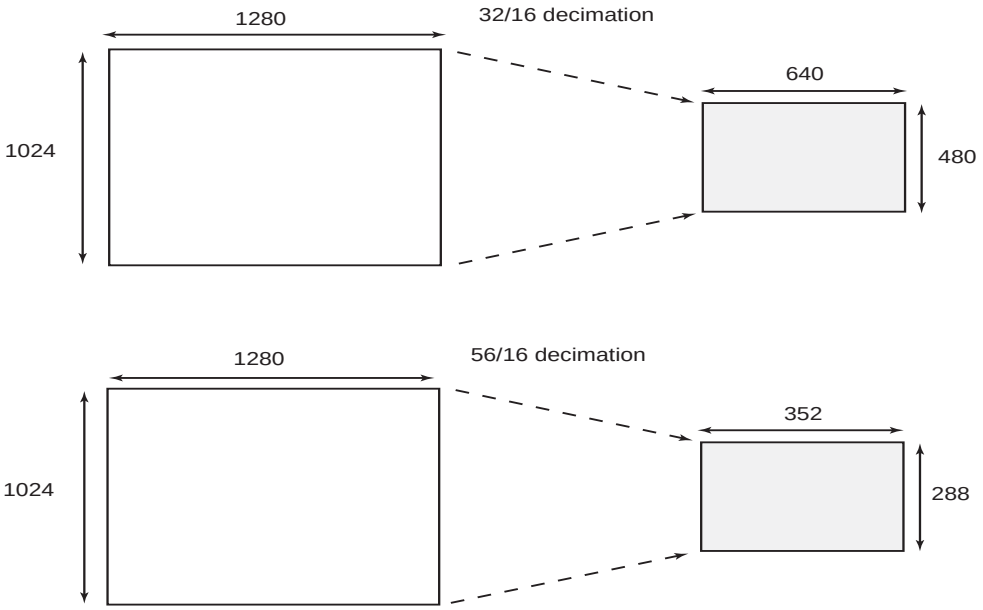
Input 1280 × 1024 Output = 640 × 480

Hratio = 1280/640 = 2

Vratio = 1024/480 = 2.1333

The decimation factor is 2 so 32/16.

Figure 33-5. Resize Examples



33.5.4.2 Color Space Conversion

This module converts YCrCb or YUV pixels to RGB color space. Clipping is performed to ensure that the samples value do not exceed the allowable range. The conversion matrix is defined below and is fully programmable:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} C_0 & 0 & C_1 \\ C_0 & -C_2 & -C_3 \\ C_0 & C_4 & 0 \end{bmatrix} \times \begin{bmatrix} Y - Y_{off} \\ C_b - C_{boff} \\ C_r - C_{roff} \end{bmatrix}$$

Example of programmable value to convert YCrCb to RGB:

$$\begin{cases} R = 1.164 \cdot (Y - 16) + 1.596 \cdot (C_r - 128) \\ G = 1.164 \cdot (Y - 16) - 0.813 \cdot (C_r - 128) - 0.392 \cdot (C_b - 128) \\ B = 1.164 \cdot (Y - 16) + 2.107 \cdot (C_b - 128) \end{cases}$$

An example of programmable value to convert from YUV to RGB:

$$\begin{cases} R = Y + 1.596 \cdot V \\ G = Y - 0.394 \cdot U - 0.436 \cdot V \\ B = Y + 2.032 \cdot U \end{cases}$$

33.5.4.3 Memory Interface

RGB Mode

The preview datapath contains a data formatter that converts 8:8:8 pixel to RGB 5:6:5 format compliant with the 16-bit format of the LCD controller. In general, when converting from a color channel with more bits to one with fewer bits, the formatter module discards the lower-order bits.

For example, converting from RGB 8:8:8 to RGB 5:6:5, the formatter module discards the three LSBs from the red and blue channels, and two LSBs from the green channel.

12-bit Grayscale Mode

ISI_DATA[11:0] is the physical interface to the ISI. These bits are sampled and written to memory.

When 12-bit Grayscale mode is enabled, two memory formats are supported:

ISI_CFG2.GS_MODE = 0: two pixels per word

ISI_CFG2.GS_MODE = 1: one pixel per word

The following tables illustrate the memory mapping for the two formats.

Table 33-10. Grayscale Memory Mapping Configuration for 12-bit Data (ISI_CFG2.GS_MODE = 0: two pixels per word)

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

Table 33-11. Grayscale Memory Mapping Configuration for 12-bit Data (ISI_CFG2.GS_MODE = 1: one pixel per word)

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

8-bit Grayscale Mode

For 8-bit Grayscale mode, ISI_DATA[7:0] on the 12-bit data bus is the physical interface to the ISI. These bits are sampled and written to memory.

To enable 8-bit Grayscale mode, configure ISI_CFG2 as follows:

- Clear ISI_CFG2.GRAYSCALE.
- Clear ISI_CFG2.RGB_SWAP.
- Clear ISI_CFG2.COL_SPACE.

- Configure the field ISI_CFG2.YCC_SWAP to value 0.
- Configure the field ISI_CFG2.IM_VSIZE with the vertical resolution of the image minus 1.
- Configure the field ISI_CFG2.IM_HSIZE with the horizontal resolution of the image divided by 2. The horizontal resolution must be a multiple of 2.

The codec datapath is used to capture the 8-bit grayscale image. Use the following configuration:

- Set ISI_DMA_C_CTRL.C_FETCH.
- Configure ISI_DMA_C_DSCR.C_DSCR with the descriptor address.
- Write a one to the bit ISI_DMA_CHER.C_CH_EN.

Table 33-12. Memory Mapping for 8-bit Grayscale Mode

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

33.5.4.4 FIFO and DMA Features

Both preview and codec datapaths contain FIFOs. These asynchronous buffers are used to safely transfer formatted pixels from the pixel clock domain to the AHB clock domain. A video arbiter is used to manage FIFO thresholds and triggers a relevant DMA request through the AHB master interface. Thus, depending on the FIFO state, a specified length burst is asserted. Regarding AHB master interface, it supports Scatter DMA mode through linked list operation. This mode of operation improves flexibility of image buffer location and allows the user to allocate two or more frame buffers. The destination frame buffers are defined by a series of Frame Buffer Descriptors (FBD). Each FBD controls the transfer of one entire frame and then optionally loads a further FBD to switch the DMA operation at another frame buffer address. The FBD is defined by a series of three words. The first word defines the current frame buffer address (named DMA_X_ADDR register), the second defines control information (named DMA_X_CTRL register) and the third defines the next descriptor address (named DMA_X_DSCR). DMA Transfer mode with linked list support is available for both codec and preview datapaths. The data to be transferred described by an FBD requires several burst accesses. In the following example, the use of two ping-pong frame buffers is described.

Example:

The first FBD, stored at address 0x00030000, defines the location of the first frame buffer. This address is programmed in the ISI user interface DMA_P_DSCR. To enable the descriptor fetch operation, the value 0x00000001 must be written to the DMA_P_CTRL register. LLI_0 and LLI_1 are the two descriptors of the linked list.

Destination address: frame buffer ID0 0x02A000 (LLI_0.DMA_P_ADDR)

Transfer 0 Control Information, fetch and writeback: 0x00000003 (LLI_0.DMA_P_CTRL)

Next FBD address: 0x00030010 (LLI_0.DMA_P_DSCR)

The second FBD, stored at address 0x00030010, defines the location of the second frame buffer.

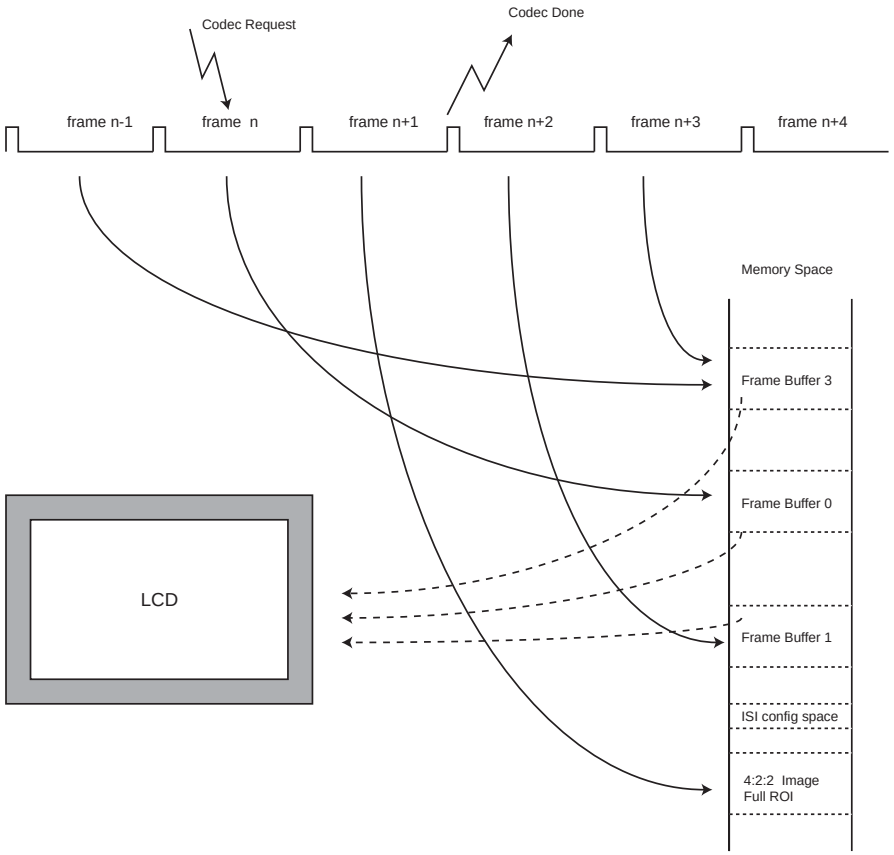
Destination address: frame buffer ID1 0x0003A000 (LLI_1.DMA_P_ADDR)

Transfer 1 Control information fetch and writeback: 0x00000003 (LLI_1.DMA_P_CTRL)

The third FBD address: 0x00030000, wrapping to first FBD (LLI_1.DMA_P_DSCR)

Using this technique, several frame buffers can be configured through the linked list. Figure 33-6 illustrates a typical three-frame buffer application. Frame n is mapped to frame buffer 0, frame n+1 is mapped to frame buffer 1, frame n+2 is mapped to frame buffer 2 and further frames wrap. A codec request occurs, and the full-size 4:2:2 encoded frame is stored in a dedicated memory space.

Figure 33-6. Three Frame Buffers Application and Memory Mapping



33.5.5 Codec Path

33.5.5.1 Color Space Conversion

Depending on user selection, this module can be bypassed so that input YCrCb stream is directly connected to the format converter module. If the RGB input stream is selected, this module converts RGB to YCrCb color space with the formulas given below:

$$\begin{bmatrix} Y \\ C_r \\ C_b \end{bmatrix} = \begin{bmatrix} C_0 & C_1 & C_2 \\ C_3 & -C_4 & -C_5 \\ -C_6 & -C_7 & C_8 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} Y_{off} \\ Cr_{off} \\ Cb_{off} \end{bmatrix}$$

An example of coefficients is given below:

$$\begin{cases} Y = 0.257 \cdot R + 0.504 \cdot G + 0.098 \cdot B + 16 \\ C_r = 0.439 \cdot R - 0.368 \cdot G - 0.071 \cdot B + 128 \\ C_b = -0.148 \cdot R - 0.291 \cdot G + 0.439 \cdot B + 128 \end{cases}$$

33.5.5.2 Memory Interface

Dedicated FIFOs are used to support packed memory mapping. YCrCb pixel components are sent in a single 32-bit word in a contiguous space (packed). Data is stored in the order of natural scan lines. Planar mode is not supported.

33.5.5.3 DMA Features

Like preview datapath, codec datapath DMA mode uses linked list operation.

33.6 Image Sensor Interface (ISI) User Interface

Table 33-13. Register Mapping

Offset	Register	Name	Access	Reset Value
0x00	ISI Configuration 1 Register	ISI_CFG1	Read/Write	0x00000000
0x04	ISI Configuration 2 Register	ISI_CFG2	Read/Write	0x00000000
0x08	ISI Preview Size Register	ISI_PSIZE	Read/Write	0x00000000
0x0C	ISI Preview Decimation Factor Register	ISI_PDECF	Read/Write	0x00000010
0x10	ISI Color Space Conversion YCrCb To RGB Set 0 Register	ISI_Y2R_SET0	Read/Write	0x6832CC95
0x14	ISI Color Space Conversion YCrCb To RGB Set 1 Register	ISI_Y2R_SET1	Read/Write	0x00007102
0x18	ISI Color Space Conversion RGB To YCrCb Set 0 Register	ISI_R2Y_SET0	Read/Write	0x01324145
0x1C	ISI Color Space Conversion RGB To YCrCb Set 1 Register	ISI_R2Y_SET1	Read/Write	0x01245E38
0x20	ISI Color Space Conversion RGB To YCrCb Set 2 Register	ISI_R2Y_SET2	Read/Write	0x01384A4B
0x24	ISI Control Register	ISI_CR	Write-only	–
0x28	ISI Status Register	ISI_SR	Read-only	0x00000000
0x2C	ISI Interrupt Enable Register	ISI_IER	Write-only	–
0x30	ISI Interrupt Disable Register	ISI_IDR	Write-only	–
0x34	ISI Interrupt Mask Register	ISI_IMR	Read-only	0x00000000
0x38	DMA Channel Enable Register	ISI_DMA_CHER	Write-only	–
0x3C	DMA Channel Disable Register	ISI_DMA_CHDR	Write-only	–
0x40	DMA Channel Status Register	ISI_DMA_CHSR	Read-only	0x00000000
0x44	DMA Preview Base Address Register	ISI_DMA_P_ADDR	Read/Write	0x00000000
0x48	DMA Preview Control Register	ISI_DMA_P_CTRL	Read/Write	0x00000000
0x4C	DMA Preview Descriptor Address Register	ISI_DMA_P_DSCR	Read/Write	0x00000000
0x50	DMA Codec Base Address Register	ISI_DMA_C_ADDR	Read/Write	0x00000000
0x54	DMA Codec Control Register	ISI_DMA_C_CTRL	Read/Write	0x00000000
0x58	DMA Codec Descriptor Address Register	ISI_DMA_C_DSCR	Read/Write	0x00000000
0x5C–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	ISI_WPMR	Read/Write	0x00000000
0xE8	Write Protection Status Register	ISI_WPSR	Read-only	0x00000000
0xEC–0xF8	Reserved	–	–	–
0xFC	Reserved	–	–	–

Note: Several parts of the ISI controller use the pixel clock provided by the image sensor (ISI_PCK). Thus the user must first program the image sensor to provide this clock (ISI_PCK) before programming the Image Sensor Controller.

33.6.1 ISI Configuration 1 Register

Name: ISI_CFG1

Address: 0xF0008000

Access: Read/Write

31	30	29	28	27	26	25	24
SFD							

23	22	21	20	19	18	17	16
SLD							

15	14	13	12	11	10	9	8
–	THMASK		FULL	DISCR	FRATE		

7	6	5	4	3	2	1	0
CRC_SYNC	EMB_SYNC	–	PIXCLK_POL	VSYNC_POL	HSYNC_POL	–	–

- **HSYNC_POL: Horizontal Synchronization Polarity**

0: HSYNC active high.

1: HSYNC active low.

- **VSYNC_POL: Vertical Synchronization Polarity**

0: VSYNC active high.

1: VSYNC active low.

- **PIXCLK_POL: Pixel Clock Polarity**

0: Data is sampled on rising edge of pixel clock.

1: Data is sampled on falling edge of pixel clock.

- **EMB_SYNC: Embedded Synchronization**

0: Synchronization by HSYNC, VSYNC.

1: Synchronization by embedded synchronization sequence SAV/EAV.

- **CRC_SYNC: Embedded Synchronization Correction**

0: No CRC correction is performed on embedded synchronization.

1: CRC correction is performed. If the correction is not possible, the current frame is discarded and the CRC_ERR bit is set in the ISI_SR.

- **FRATE: Frame Rate [0..7]**

0: All the frames are captured, else one frame every FRATE + 1 is captured.

- **DISCR: Disable Codec Request**

0: Codec datapath DMA interface requires a request to restart.

1: Codec datapath DMA automatically restarts.

- **FULL: Full Mode is Allowed**

0: The codec frame is transferred to memory when an available frame slot is detected.

1: Both preview and codec DMA channels are operating simultaneously.

- **THMASK: Threshold Mask**

Value	Name	Description
0	BEATS_4	Only 4 beats AHB burst allowed
1	BEATS_8	Only 4 and 8 beats AHB burst allowed
2	BEATS_16	4, 8 and 16 beats AHB burst allowed

- **SLD: Start of Line Delay**

SLD pixel clock periods to wait before the beginning of a line.

- **SFD: Start of Frame Delay**

SFD lines are skipped at the beginning of the frame.

33.6.2 ISI Configuration 2 Register

Name: ISI_CFG2

Address: 0xF0008004

Access: Read/Write

31	30	29	28	27	26	25	24
RGB_CFG		YCC_SWAP		–	IM_HSIZE		

23	22	21	20	19	18	17	16
IM_HSIZE							

15	14	13	12	11	10	9	8
COL_SPACE	RGB_SWAP	GRAYSCALE	RGB_MODE	GS_MODE	IM_VSIZE		

7	6	5	4	3	2	1	0
IM_VSIZE							

- **IM_VSIZE: Vertical Size of the Image Sensor [0..2047]**

IM_VSIZE = Vertical size - 1

- **GS_MODE: Grayscale Pixel Format Mode**

0: 2 pixels per word.

1: 1 pixel per word.

- **RGB_MODE: RGB Input Mode**

0: RGB 8:8:8 24 bits.

1: RGB 5:6:5 16 bits.

- **GRAYSCALE: Grayscale Mode Format Enable**

0: Grayscale mode is disabled.

1: Input image is assumed to be grayscale-coded.

- **RGB_SWAP: RGB Format Swap Mode**

0: D7 → R7.

1: D0 → R7.

The RGB_SWAP has no effect when Grayscale mode is enabled.

- **COL_SPACE: Color Space for the Image Data**

0: YCbCr.

1: RGB.

- **IM_HSIZE: Horizontal Size of the Image Sensor [0..2047]**

If 8-bit Grayscale mode is enabled, IM_HSIZE = (Horizontal size/2) - 1.

Else IM_HSIZE = Horizontal size - 1.

- **YCC_SWAP: YCrCb Format Swap Mode**

Defines the YCC image data.

Value	Name	Description
0	DEFAULT	Byte 0 Cb(i) Byte 1 Y(i) Byte 2 Cr(i) Byte 3 Y(i+1)
1	MODE1	Byte 0 Cr(i) Byte 1 Y(i) Byte 2 Cb(i) Byte 3 Y(i+1)
2	MODE2	Byte 0 Y(i) Byte 1 Cb(i) Byte 2 Y(i+1) Byte 3 Cr(i)
3	MODE3	Byte 0 Y(i) Byte 1 Cr(i) Byte 2 Y(i+1) Byte 3 Cb(i)

- **RGB_CFG: RGB Pixel Mapping Configuration**

Defines RGB pattern when RGB_MODE is set to 1.

Value	Name	Description
0	DEFAULT	Byte 0 R/G(MSB) Byte 1 G(LSB)/B Byte 2 R/G(MSB) Byte 3 G(LSB)/B
1	MODE1	Byte 0 B/G(MSB) Byte 1 G(LSB)/R Byte 2 B/G(MSB) Byte 3 G(LSB)/R
2	MODE2	Byte 0 G(LSB)/R Byte 1 B/G(MSB) Byte 2 G(LSB)/R Byte 3 B/G(MSB)
3	MODE3	Byte 0 G(LSB)/B Byte 1 R/G(MSB) Byte 2 G(LSB)/B Byte 3 R/G(MSB)

If RGB_MODE is set to RGB 8:8:8, then RGB_CFG = 0 implies RGB color sequence, else it implies BGR color sequence.

33.6.3 ISI Preview Size Register

Name: ISI_PSIZE

Address: 0xF0008008

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	PREV_HSIZE	

23	22	21	20	19	18	17	16
PREV_HSIZE							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	PREV_VSIZE	

7	6	5	4	3	2	1	0
PREV_VSIZE							

- **PREV_VSIZE: Vertical Size for the Preview Path**

PREV_VSIZE = Vertical Preview size - 1 (480 max only in RGB mode).

- **PREV_HSIZE: Horizontal Size for the Preview Path**

PREV_HSIZE = Horizontal Preview size - 1 (640 max only in RGB mode).

33.6.4 ISI Preview Decimation Factor Register

Name: ISI_PDECF

Address: 0xF000800C

Access: Read/Write

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
DEC_FACTOR							

- **DEC_FACTOR: Decimation Factor**

DEC_FACTOR is 8-bit width, range is from 16 to 255. Values from 0 to 16 do not perform any decimation.

33.6.5 ISI Color Space Conversion YCrCb to RGB Set 0 Register

Name: ISI_Y2R_SET0

Address: 0xF0008010

Access: Read/Write

31	30	29	28	27	26	25	24
C3							
23	22	21	20	19	18	17	16
C2							
15	14	13	12	11	10	9	8
C1							
7	6	5	4	3	2	1	0
C0							

- **C0: Color Space Conversion Matrix Coefficient C0**

C0 element default step is 1/128, ranges from 0 to 1.9921875.

- **C1: Color Space Conversion Matrix Coefficient C1**

C1 element default step is 1/128, ranges from 0 to 1.9921875.

- **C2: Color Space Conversion Matrix Coefficient C2**

C2 element default step is 1/128, ranges from 0 to 1.9921875.

- **C3: Color Space Conversion Matrix Coefficient C3**

C3 element default step is 1/128, ranges from 0 to 1.9921875.

33.6.6 ISI Color Space Conversion YCrCb to RGB Set 1 Register

Name: ISI_Y2R_SET1

Address: 0xF0008014

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	Cboff	Croff	Yoff	–	–	–	C4
7	6	5	4	3	2	1	0
C4							

- **C4: Color Space Conversion Matrix Coefficient C4**

C4 element default step is 1/128, ranges from 0 to 3.9921875.

- **Yoff: Color Space Conversion Luminance Default Offset**

0: No offset.

1: Offset = 128.

- **Croff: Color Space Conversion Red Chrominance Default Offset**

0: No offset.

1: Offset = 16.

- **Cboff: Color Space Conversion Blue Chrominance Default Offset**

0: No offset.

1: Offset = 16.

33.6.7 ISI Color Space Conversion RGB to YCrCb Set 0 Register

Name: ISI_R2Y_SET0

Address: 0xF0008018

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	Roff

23	22	21	20	19	18	17	16
–	C2						

15	14	13	12	11	10	9	8
–	C1						

7	6	5	4	3	2	1	0
–	C0						

- **C0: Color Space Conversion Matrix Coefficient C0**

C0 element default step is 1/256, from 0 to 0.49609375.

- **C1: Color Space Conversion Matrix Coefficient C1**

C1 element default step is 1/128, from 0 to 0.9921875.

- **C2: Color Space Conversion Matrix Coefficient C2**

C2 element default step is 1/512, from 0 to 0.2480468875.

- **Roff: Color Space Conversion Red Component Offset**

0: No offset

1: Offset = 16

33.6.8 ISI Color Space Conversion RGB to YCrCb Set 1 Register

Name: ISI_R2Y_SET1

Address: 0xF000801C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	Goff

23	22	21	20	19	18	17	16
–	C5						

15	14	13	12	11	10	9	8
–	C4						

7	6	5	4	3	2	1	0
–	C3						

- **C3: Color Space Conversion Matrix Coefficient C3**

C0 element default step is 1/128, ranges from 0 to 0.9921875.

- **C4: Color Space Conversion Matrix Coefficient C4**

C1 element default step is 1/256, ranges from 0 to 0.49609375.

- **C5: Color Space Conversion Matrix Coefficient C5**

C1 element default step is 1/512, ranges from 0 to 0.2480468875.

- **Goff: Color Space Conversion Green Component Offset**

0: No offset.

1: Offset = 128.

33.6.9 ISI Color Space Conversion RGB to YCrCb Set 2 Register

Name: ISI_R2Y_SET2

Address: 0xF0008020

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	Boff

23	22	21	20	19	18	17	16
–	C8						

15	14	13	12	11	10	9	8
–	C7						

7	6	5	4	3	2	1	0
–	C6						

- **C6: Color Space Conversion Matrix Coefficient C6**

C6 element default step is 1/512, ranges from 0 to 0.2480468875.

- **C7: Color Space Conversion Matrix Coefficient C7**

C7 element default step is 1/256, ranges from 0 to 0.49609375.

- **C8: Color Space Conversion Matrix Coefficient C8**

C8 element default step is 1/128, ranges from 0 to 0.9921875.

- **Boff: Color Space Conversion Blue Component Offset**

0: No offset.

1: Offset = 128.

33.6.10 ISI Control Register

Name: ISI_CR

Address: 0xF0008024

Access: Write-only

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

- **ISI_EN: ISI Module Enable Request**

Write a one to this bit to enable the module. Software must poll the ENABLE bit in the ISI_SR to verify that the command has successfully completed.

- **ISI_DIS: ISI Module Disable Request**

Write a one to this bit to disable the module. If both ISI_EN and ISI_DIS are asserted at the same time, the disable request is not taken into account. Software must poll the DIS_DONE bit in the ISI_SR to verify that the command has successfully completed.

- **ISI_SRST: ISI Software Reset Request**

Write a one to this bit to request a software reset of the module. Software must poll the SRST bit in the ISI_SR to verify that the software request command has terminated.

- **ISI_CDC: ISI Codec Request**

Write a one to this bit to enable the codec datapath and capture a full resolution frame. A new request cannot be taken into account while CDC_PND bit is active in the ISI_SR.

33.6.11 ISI Status Register

Name: ISI_SR

Address: 0xF0008028

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	FR_OVR	CRC_ERR	C_OVR	P_OVR
23	22	21	20	19	18	17	16
–	–	–	–	SIP	–	CXFR_DONE	PXFR_DONE
15	14	13	12	11	10	9	8
–	–	–	–	–	VSYNC	–	CDC_PND
7	6	5	4	3	2	1	0
–	–	–	–	–	SRST	DIS_DONE	ENABLE

- **ENABLE: Module Enable**

0: Module is disabled.

1: Module is enabled.

- **DIS_DONE: Module Disable Request has Terminated (cleared on read)**

0: Indicates that the request is not completed (if a request was issued).

1: Disable request has completed. This flag is reset after a read operation.

- **SRST: Module Software Reset Request has Terminated (cleared on read)**

0: Indicates that the request is not completed (if a request was issued).

1: Software reset request has completed. This flag is reset after a read operation.

- **CDC_PND: Pending Codec Request**

0: Indicates that no codec request is pending

1: Indicates that the request has been taken into account but cannot be serviced within the current frame. The operation is postponed to the next frame.

- **VSYNC: Vertical Synchronization (cleared on read)**

0: Indicates that the vertical synchronization has not been detected since the last read of the ISI_SR.

1: Indicates that a vertical synchronization has been detected since the last read of the ISI_SR.

- **PXFR_DONE: Preview DMA Transfer has Terminated (cleared on read)**

0: Preview transfer done not detected.

1: Preview transfer done detected. When set, this bit indicates that the data transfer on the preview channel has completed since the last read of ISI_SR.

- **CXFR_DONE: Codec DMA Transfer has Terminated (cleared on read)**

0: Codec transfer done not detected.

1: Codec transfer done detected. When set, this bit indicates that the data transfer on the codec channel has completed since the last read of ISI_SR.

- **SIP: Synchronization in Progress**

When the status of the preview or codec DMA channel is modified, a minimum amount of time is required to perform the clock domain synchronization.

0: The clock domain synchronization process is terminated.

1: This bit is set when the clock domain synchronization operation occurs. No modification of the channel status is allowed when this bit is set, to guarantee data integrity.

- **P_OVR: Preview Datapath Overflow (cleared on read)**

0: No overflow

1: An overrun condition has occurred in input FIFO on the preview path. The overrun happens when the FIFO is full and an attempt is made to write a new sample to the FIFO since the last read of ISI_SR.

- **C_OVR: Codec Datapath Overflow (cleared on read)**

0: No overflow

1: An overrun condition has occurred in input FIFO on the codec path. The overrun happens when the FIFO is full and an attempt is made to write a new sample to the FIFO since the last read of ISI_SR.

- **CRC_ERR: CRC Synchronization Error (cleared on read)**

0: No CRC error in the embedded synchronization frame (SAV/EAV)

1: Embedded Synchronization Correction is enabled (CRC_SYNC bit is set) in the ISI_CR and an error has been detected and not corrected since the last read of ISI_SR. The frame is discarded and the ISI waits for a new one.

- **FR_OVR: Frame Rate Overrun (cleared on read)**

0: No frame overrun

1: Frame overrun. The current frame is being skipped because a vsync signal has been detected while flushing FIFOs since the last read of ISI_SR.

33.6.12 ISI Interrupt Enable Register

Name: ISI_IER

Address: 0xF000802C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	FR_OVR	CRC_ERR	C_OVR	P_OVR
23	22	21	20	19	18	17	16
–	–	–	–	–	–	CXFR_DONE	PXFR_DONE
15	14	13	12	11	10	9	8
–	–	–	–	–	VSYNC	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	SRST	DIS_DONE	–

- **DIS_DONE: Disable Done Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **SRST: Software Reset Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **VSYNC: Vertical Synchronization Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **PXFR_DONE: Preview DMA Transfer Done Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **CXFR_DONE: Codec DMA Transfer Done Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **P_OVR: Preview Datapath Overflow Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **C_OVR: Codec Datapath Overflow Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **CRC_ERR: Embedded Synchronization CRC Error Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **FR_OVR: Frame Rate Overflow Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

33.6.13 ISI Interrupt Disable Register

Name: ISI_IDR

Address: 0xF0008030

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	FR_OVR	CRC_ERR	C_OVR	P_OVR
23	22	21	20	19	18	17	16
–	–	–	–	–	–	CXFR_DONE	PXFR_DONE
15	14	13	12	11	10	9	8
–	–	–	–	–	VSYNC	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	SRST	DIS_DONE	–

- **DIS_DONE: Disable Done Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **SRST: Software Reset Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **VSYNC: Vertical Synchronization Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **PXFR_DONE: Preview DMA Transfer Done Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **CXFR_DONE: Codec DMA Transfer Done Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **P_OVR: Preview Datapath Overflow Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **C_OVR: Codec Datapath Overflow Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **CRC_ERR: Embedded Synchronization CRC Error Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **FR_OVR: Frame Rate Overflow Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

33.6.14 ISI Interrupt Mask Register

Name: ISI_IMR

Address: 0xF0008034

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	FR_OVR	CRC_ERR	C_OVR	P_OVR
23	22	21	20	19	18	17	16
–	–	–	–	–	–	CXFR_DONE	PXFR_DONE
15	14	13	12	11	10	9	8
–	–	–	–	–	VSYNC	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	SRST	DIS_DONE	–

- **DIS_DONE: Module Disable Operation Completed**

0: The Module Disable Operation Completed interrupt is disabled.

1: The Module Disable Operation Completed interrupt is enabled.

- **SRST: Software Reset Completed**

0: The Software Reset Completed interrupt is disabled.

1: The Software Reset Completed interrupt is enabled.

- **VSYNC: Vertical Synchronization**

0: The Vertical Synchronization interrupt is disabled.

1: The Vertical Synchronization interrupt is enabled.

- **PXFR_DONE: Preview DMA Transfer Completed**

0: The Preview DMA Transfer Completed interrupt is disabled.

1: The Preview DMA Transfer Completed interrupt is enabled.

- **CXFR_DONE: Codec DMA Transfer Completed**

0: The Codec DMA Transfer Completed interrupt is disabled.

1: The Codec DMA Transfer Completed interrupt is enabled.

- **P_OVR: Preview FIFO Overflow**

0: The Preview FIFO Overflow interrupt is disabled.

1: The Preview FIFO Overflow interrupt is enabled.

- **C_OVR: Codec FIFO Overflow**

0: The Codec FIFO Overflow interrupt is disabled.

1: The Codec FIFO Overflow interrupt is enabled.

- **CRC_ERR: CRC Synchronization Error**

0: The CRC Synchronization Error interrupt is disabled.

1: The CRC Synchronization Error interrupt is enabled.

- **FR_OVR: Frame Rate Overrun**

0: The Frame Rate Overrun interrupt is disabled.

1: The Frame Rate Overrun is enabled.

33.6.15 DMA Channel Enable Register

Name: ISI_DMA_CHER

Address: 0xF0008038

Access: Write-only

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
–	–	–	–	–	–	C_CH_EN	P_CH_EN

- **P_CH_EN: Preview Channel Enable**

Write a one to this bit to enable the preview DMA channel.

- **C_CH_EN: Codec Channel Enable**

Write a one to this bit to enable the codec DMA channel.

33.6.16 DMA Channel Disable Register

Name: ISI_DMA_CHDR

Address: 0xF000803C

Access: Read/Write

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
–	–	–	–	–	–	C_CH_DIS	P_CH_DIS

- **P_CH_DIS: Preview Channel Disable Request**

0: No effect.

1: Disables the channel. Poll P_CH_S in DMA_CHSR to verify that the preview channel status has been successfully modified.

- **C_CH_DIS: Codec Channel Disable Request**

0: No effect.

1: Disables the channel. Poll C_CH_S in DMA_CHSR to verify that the codec channel status has been successfully modified.

33.6.17 DMA Channel Status Register

Name: ISI_DMA_CHSR

Address: 0xF0008040

Access: Read-only

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
–	–	–	–	–	–	C_CH_S	P_CH_S

- **P_CH_S: Preview DMA Channel Status**

0: Indicates that the Preview DMA channel is disabled.

1: Indicates that the Preview DMA channel is enabled.

- **C_CH_S: Code DMA Channel Status**

0: Indicates that the Codec DMA channel is disabled.

1: Indicates that the Codec DMA channel is enabled.

33.6.18 DMA Preview Base Address Register

Name: ISI_DMA_P_ADDR

Address: 0xF0008044

Access: Read/Write

31	30	29	28	27	26	25	24
P_ADDR							

23	22	21	20	19	18	17	16
P_ADDR							

15	14	13	12	11	10	9	8
P_ADDR							

7	6	5	4	3	2	1	0
P_ADDR						–	–

- **P_ADDR: Preview Image Base Address**

This address is word-aligned.

33.6.19 DMA Preview Control Register

Name: ISI_DMA_P_CTRL

Address: 0xF0008048

Access: Read/Write

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
–	–	–	–	P_DONE	P_IEN	P_WB	P_FETCH

- **P_FETCH: Descriptor Fetch Control Bit**

0: Preview channel fetch operation is disabled.

1: Preview channel fetch operation is enabled.

- **P_WB: Descriptor Writeback Control Bit**

0: Preview channel writeback operation is disabled.

1: Preview channel writeback operation is enabled.

- **P_IEN: Transfer Done Flag Control**

0: Preview transfer done flag generation is enabled.

1: Preview transfer done flag generation is disabled.

- **P_DONE: Preview Transfer Done**

This bit is only updated in the memory.

0: The transfer related to this descriptor has not been performed.

1: The transfer related to this descriptor has completed. This bit is updated in memory at the end of the transfer, when writeback operation is enabled.

33.6.20 DMA Preview Descriptor Address Register

Name: ISI_DMA_P_DSCR

Address: 0xF000804C

Access: Read/Write

31	30	29	28	27	26	25	24
P_DSCR							

23	22	21	20	19	18	17	16
P_DSCR							

15	14	13	12	11	10	9	8
P_DSCR							

7	6	5	4	3	2	1	0
P_DSCR						–	–

- **P_DSCR: Preview Descriptor Base Address**

This address is word-aligned.

33.6.21 DMA Codec Base Address Register

Name: ISI_DMA_C_ADDR

Address: 0xF0008050

Access: Read/Write

31	30	29	28	27	26	25	24
C_ADDR							

23	22	21	20	19	18	17	16
C_ADDR							

15	14	13	12	11	10	9	8
C_ADDR							

7	6	5	4	3	2	1	0
C_ADDR						–	–

- **C_ADDR: Codec Image Base Address**

This address is word-aligned.

33.6.22 DMA Codec Control Register

Name: ISI_DMA_C_CTRL

Address: 0xF0008054

Access: Read/Write

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
–	–	–	–	C_DONE	C_IEN	C_WB	C_FETCH

- **C_FETCH: Descriptor Fetch Control Bit**

0: Codec channel fetch operation is disabled.

1: Codec channel fetch operation is enabled.

- **C_WB: Descriptor Writeback Control Bit**

0: Codec channel writeback operation is disabled.

1: Codec channel writeback operation is enabled.

- **C_IEN: Transfer Done Flag Control**

0: Codec transfer done flag generation is enabled.

1: Codec transfer done flag generation is disabled.

- **C_DONE: Codec Transfer Done**

This bit is only updated in the memory.

0: The transfer related to this descriptor has not been performed.

1: The transfer related to this descriptor has completed. This bit is updated in memory at the end of the transfer when write-back operation is enabled.

33.6.23 DMA Codec Descriptor Address Register

Name: ISI_DMA_C_DSCR

Address: 0xF0008058

Access: Read/Write

31	30	29	28	27	26	25	24
C_DSCR							

23	22	21	20	19	18	17	16
C_DSCR							

15	14	13	12	11	10	9	8
C_DSCR							

7	6	5	4	3	2	1	0
C_DSCR						–	–

- **C_DSCR: Codec Descriptor Base Address**

This address is word-aligned.

33.6.24 ISI Write Protection Mode Register

Name: ISI_WPMR

Address: 0xF00080E4

Access: Read/Write

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

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x495349 (“ISI” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x495349 (“ISI” in ASCII).

- **WPKEY: Write Protection Key Password**

Value	Name	Description
0x495349	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

33.6.25 ISI Write Protection Status Register

Name: ISI_WPSR

Address: 0xF00080E8

Access: Read/Write

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

- **WPVS: Write Protection Violation Status**

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

- **WPVSRC: Write Protection Violation Source**

Value	Description
0	No Write Protection Violation occurred since the last read of this register (ISI_WPSR).
1	Write access in ISI_CFG1 while Write Protection was enabled (since the last read).
2	Write access in ISI_CFG2 while Write Protection was enabled (since the last read).
3	Write access in ISI_PSIZE while Write Protection was enabled (since the last read).
4	Write access in ISI_PDECF while Write Protection was enabled (since the last read).
5	Write access in ISI_Y2R_SET0 while Write Protection was enabled (since the last read).
6	Write access in ISI_Y2R_SET1 while Write Protection was enabled (since the last read).
7	Write access in ISI_R2Y_SET0 while Write Protection was enabled (since the last read).
8	Write access in ISI_R2Y_SET1 while Write Protection was enabled (since the last read).
9	Write access in ISI_R2Y_SET2 while Write Protection was enabled (since the last read).

34. USB High Speed Device Port (UDPHS)

34.1 Description

The USB High Speed Device Port (UDPHS) is compliant with the Universal Serial Bus (USB), rev 2.0 High Speed device specification.

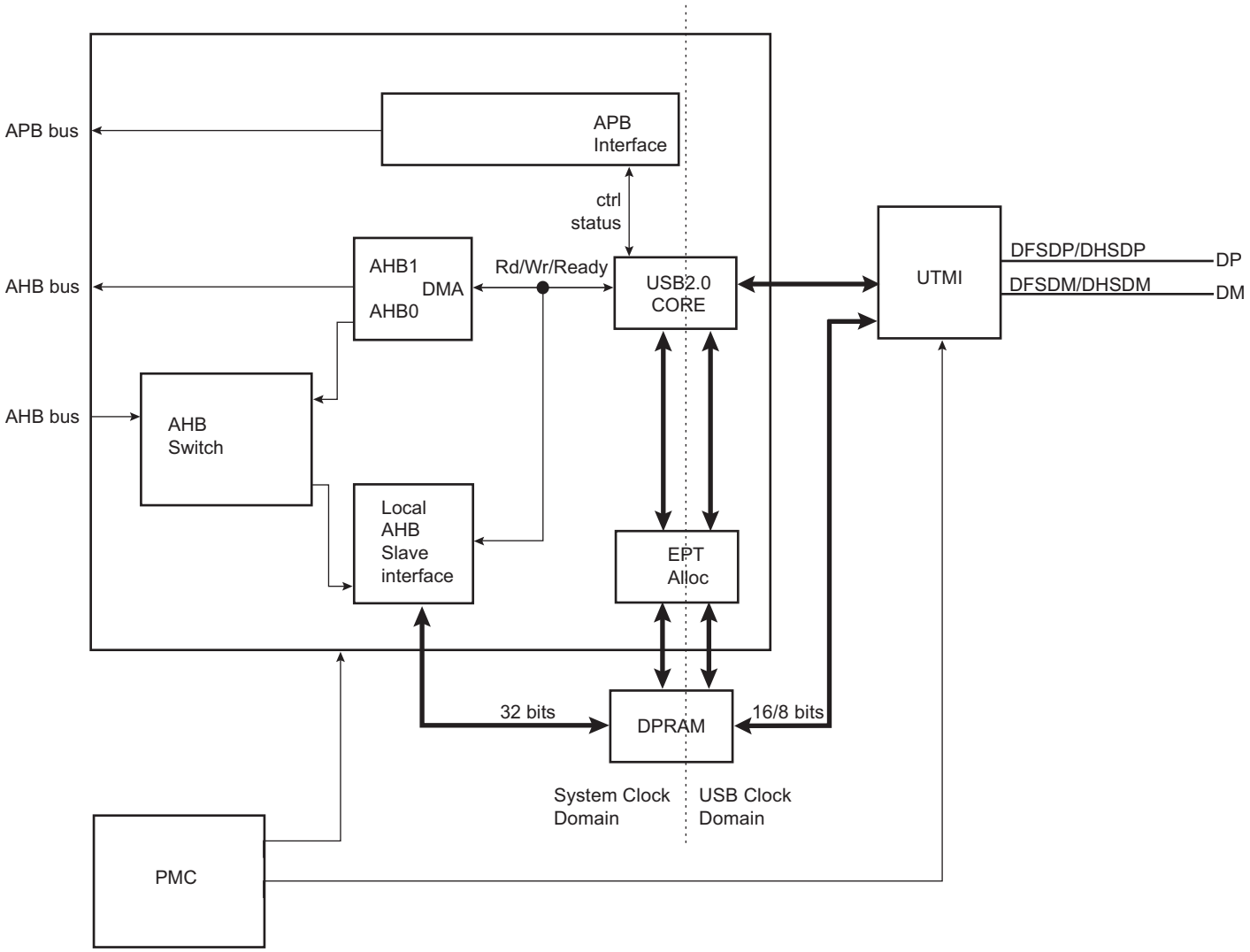
Each endpoint can be configured in one of several USB transfer types. It can be associated with one, two or three banks of a Dual-port RAM used to store the current data payload. If two or three banks are used, one DPR bank is read or written by the processor, while the other is read or written by the USB device peripheral. This feature is mandatory for isochronous endpoints.

34.2 Embedded Characteristics

- 1 Device High Speed
- 1 UTMI transceiver shared between Host and Device
- USB v2.0 High Speed Compliant, 480 Mbit/s
- 16 Endpoints up to 1024 bytes
- Embedded Dual-port RAM for Endpoints
- Suspend/Resume Logic (Command of UTMI)
- Up to Three Memory Banks for Endpoints (Not for Control Endpoint)
- 8 Kbytes of DPRAM

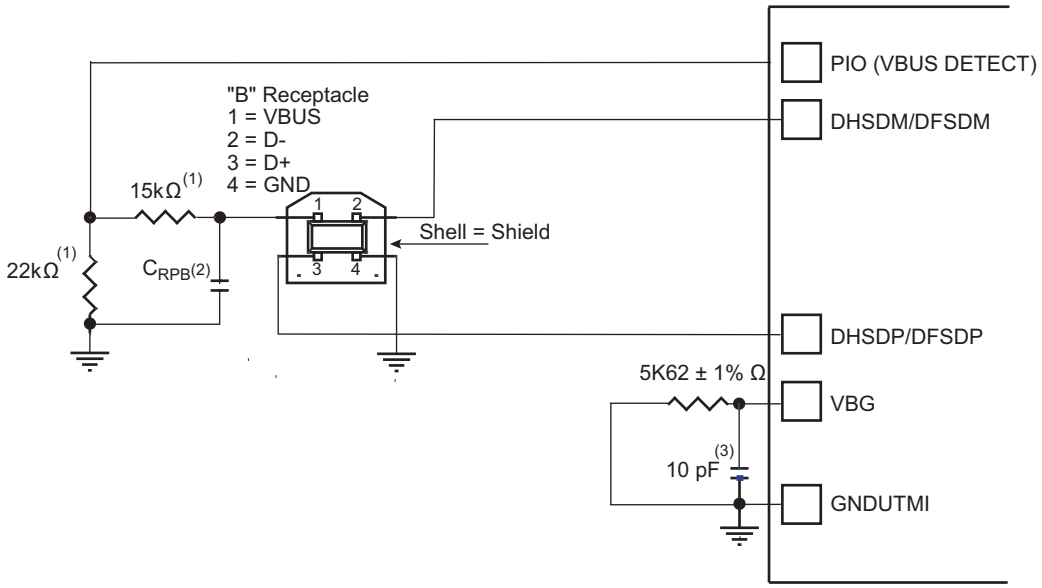
34.3 Block Diagram

Figure 34-1. Block Diagram



34.4 Typical Connection

Figure 34-2. Board Schematic



Notes:
1. The values shown on the 22 kΩ and 15 kΩ resistors are only valid with 3V3-supplied PIOs.
2. CRPB: Upstream Facing Port Bypass Capacitance of 1 µF to 10 µF (refer to “DC Electrical Characteristics” in Universal Serial Bus Specification Rev. 2)
3. 10 pF capacitor on VBG is a provision and may not be populated.

34.5 Product Dependencies

34.5.1 Power Management

The UDPHS is not continuously clocked.

For using the UDPHS, the programmer must first enable the UDPHS Clock in the Power Management Controller Peripheral Clock Enable Register (PMC_PCER). Then enable the PLL in the PMC UTMI Clock Configuration Register (CKGR_UCKR). Finally, enable BIAS in CKGR_UCKR.

However, if the application does not require UDPHS operations, the UDPHS clock can be stopped when not needed and restarted later.

34.5.2 Interrupt Sources

The UDPHS interrupt line is connected on one of the internal sources of the interrupt controller. Using the UDPHS interrupt requires the interrupt controller to be programmed first.

Table 34-1. Peripheral IDs

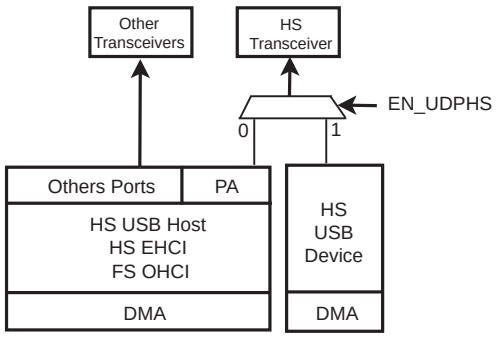
Instance	ID
UDPHS	47

34.6 Functional Description

34.6.1 UTMI transceivers Sharing

The High Speed USB Host Port A is shared with the High Speed USB Device port and connected to the second UTMI transceiver. The selection between Host Port A and USB Device is controlled by the UDPHS enable bit (EN_UDPHS) located in the UDPHS_CTRL register.

Figure 34-3. USB Selection



34.6.2 USB V2.0 High Speed Device Port Introduction

The USB V2.0 High Speed Device Port provides communication services between host and attached USB devices. Each device is offered with a collection of communication flows (pipes) associated with each endpoint. Software on the host communicates with a USB Device through a set of communication flows.

34.6.3 USB V2.0 High Speed Transfer Types

A communication flow is carried over one of four transfer types defined by the USB device.

A device provides several logical communication pipes with the host. To each logical pipe is associated an endpoint. Transfer through a pipe belongs to one of the four transfer types:

- Control Transfers: Used to configure a device at attach time and can be used for other device-specific purposes, including control of other pipes on the device.
- Bulk Data Transfers: Generated or consumed in relatively large burst quantities and have wide dynamic latitude in transmission constraints.
- Interrupt Data Transfers: Used for timely but reliable delivery of data, for example, characters or coordinates with human-perceptible echo or feedback response characteristics.
- Isochronous Data Transfers: Occupy a prenegotiated amount of USB bandwidth with a prenegotiated delivery latency. (Also called streaming real time transfers.)

As indicated below, transfers are sequential events carried out on the USB bus.

Endpoints must be configured according to the transfer type they handle.

Table 34-2. USB Communication Flow

Transfer	Direction	Bandwidth	Endpoint Size	Error Detection	Retrying
Control	Bidirectional	Not guaranteed	8, 16, 32, 64	Yes	Automatic
Isochronous	Unidirectional	Guaranteed	8–1024	Yes	No
Interrupt	Unidirectional	Not guaranteed	8–1024	Yes	Yes
Bulk	Unidirectional	Not guaranteed	8–512	Yes	Yes

34.6.4 USB Transfer Event Definitions

A transfer is composed of one or several transactions as shown in the following table.

Table 34-3. USB Transfer Events

Transfer		Transaction
Direction	**Type**	
CONTROL (bidirectional)	Control Transfer (1)	• Setup transaction → Data IN transactions → Status OUT transaction • Setup transaction → Data OUT transactions → Status IN transaction • Setup transaction → Status IN transaction
IN (device toward host)	Bulk IN Transfer	• Data IN transaction → Data IN transaction
	Interrupt IN Transfer	• Data IN transaction → Data IN transaction
	Isochronous IN Transfer (2)	• Data IN transaction → Data IN transaction
OUT (host toward device)	Bulk OUT Transfer	• Data OUT transaction → Data OUT transaction
	Interrupt OUT Transfer	• Data OUT transaction → Data OUT transaction
	Isochronous OUT Transfer (2)	• Data OUT transaction → Data OUT transaction

Notes: 1. Control transfer must use endpoints with one bank and can be aborted using a stall handshake.
2. Isochronous transfers must use endpoints configured with two or three banks.

An endpoint handles all transactions related to the type of transfer for which it has been configured.

Table 34-4. UDPHS Endpoint Description

Endpoint #	Mnemonic	Nb Bank	DMA	High Band Width	Max. Endpoint Size	Endpoint Type
0	EPT_0	1	N	N	64	Control
1	EPT_1	3	Y	Y	1024	Ctrl/Bulk/Iso(1)/Interrupt
2	EPT_2	3	Y	Y	1024	Ctrl/Bulk/Iso(1)/Interrupt
3	EPT_3	2	Y	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
4	EPT_4	2	Y	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
5	EPT_5	2	Y	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
6	EPT_6	2	Y	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
7	EPT_7	2	Y	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
8	EPT_8	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
9	EPT_9	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
10	EPT_10	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
11	EPT_11	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
12	EPT_12	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
13	EPT_13	2	N	N	1024	Ctrl/Bulk/Isov/Interrupt
14	EPT_14	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
15	EPT_15	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt

Note: 1. In Isochronous (Iso) mode, it is preferable that High Band Width capability is available.

The size of internal DPRAM is 8 KB.

Suspend and resume are automatically detected by the UDPHS device, which notifies the processor by raising an interrupt.

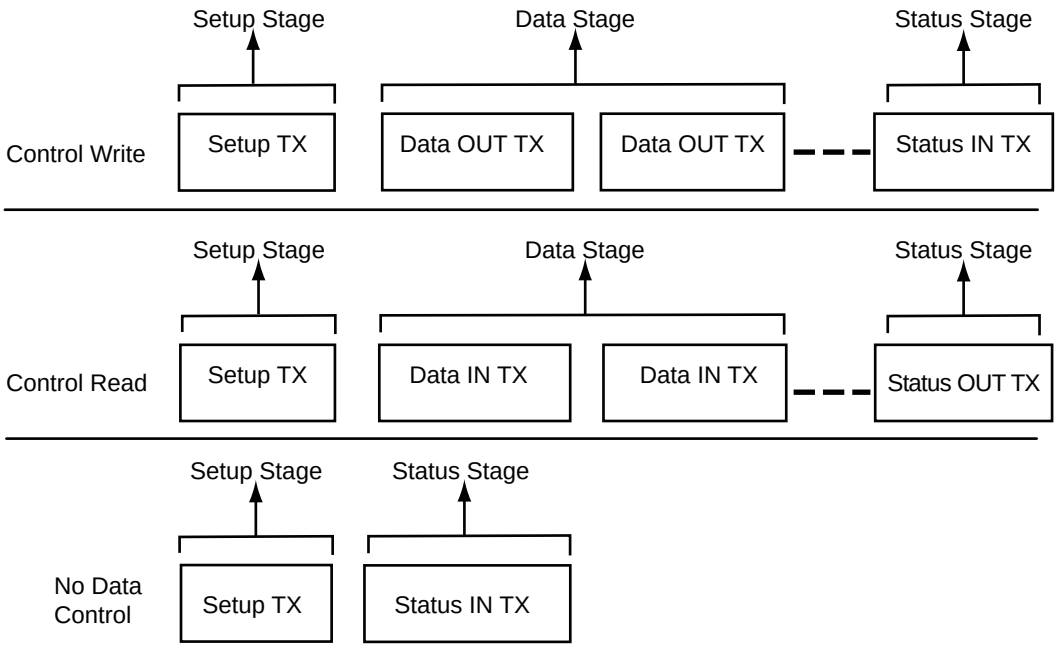
34.6.5 USB V2.0 High Speed BUS Transactions

Each transfer results in one or more transactions over the USB bus.

There are five kinds of transactions flowing across the bus in packets:

1. Setup Transaction
2. Data IN Transaction
3. Data OUT Transaction
4. Status IN Transaction
5. Status OUT Transaction

Figure 34-4. Control Read and Write Sequences



A status IN or OUT transaction is identical to a data IN or OUT transaction.

34.6.6 Endpoint Configuration

The endpoint 0 is always a control endpoint, it must be programmed and active in order to be enabled when the End Of Reset interrupt occurs.

To configure the endpoints:

- Fill the configuration register (UDPHS_EPTCFG) with the endpoint size, direction (IN or OUT), type (CTRL, Bulk, IT, ISO) and the number of banks.
- Fill the number of transactions (NB_TRANS) for isochronous endpoints.

Note: For control endpoints the direction has no effect.

- Verify that the EPT_MAPD flag is set. This flag is set if the endpoint size and the number of banks are correct compared to the FIFO maximum capacity and the maximum number of allowed banks.
- Configure control flags of the endpoint and enable it in UDPHS_EPTCTLENBx according to Section 34.7.12 “UDPHS Endpoint Control Disable Register (Isochronous Endpoint)”.

Control endpoints can generate interrupts and use only 1 bank.

All endpoints (except endpoint 0) can be configured either as Bulk, Interrupt or Isochronous. Refer to Table 34-4. UDPHS Endpoint Description.

The maximum packet size they can accept corresponds to the maximum endpoint size.

Note: The endpoint size of 1024 is reserved for isochronous endpoints.

The size of the DPRAM is 8 KB. The DPR is shared by all active endpoints. The memory size required by the active endpoints must not exceed the size of the DPRAM.

SIZE_DPRAM = SIZE _EPT0

+ NB_BANK_EPT1 x SIZE_EPT1

+ NB_BANK_EPT2 x SIZE_EPT2

+ NB_BANK_EPT3 x SIZE_EPT3

+ NB_BANK_EPT4 x SIZE_EPT4

+ NB_BANK_EPT5 x SIZE_EPT5

+ NB_BANK_EPT6 x SIZE_EPT6

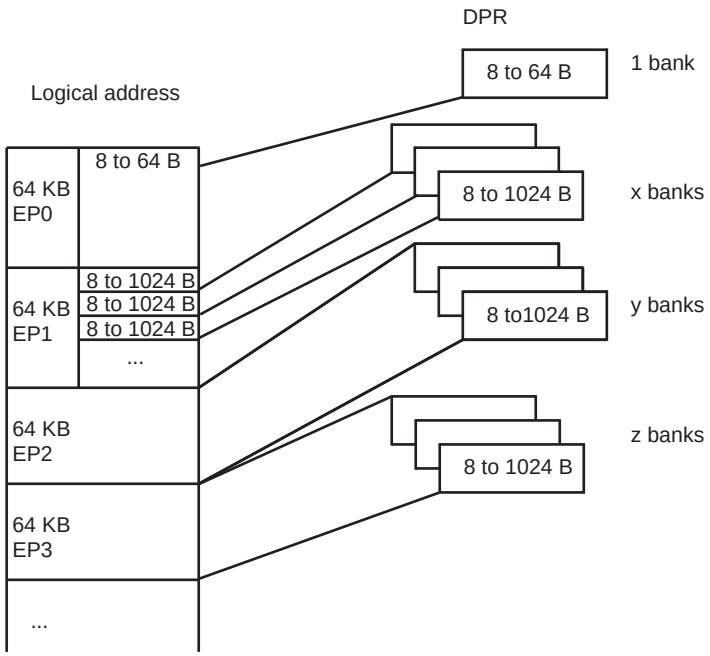
+... (refer to 34.7.8 UDPHS Endpoint Configuration Register)

If a user tries to configure endpoints with a size the sum of which is greater than the DPRAM, then the EPT_MAPD is not set.

The application has access to the physical block of DPR reserved for the endpoint through a 64 KB logical address space.

The physical block of DPR allocated for the endpoint is remapped all along the 64 KB logical address space. The application can write a 64 KB buffer linearly.

Figure 34-5. Logical Address Space for DPR Access



Configuration examples of UDPHS_EPTCTLx (UDPHS Endpoint Control Disable Register (Isochronous Endpoint)) for Bulk IN endpoint type follow below.

- With DMA
  - AUTO_VALID: Automatically validate the packet and switch to the next bank.
  - EPT_ENABL: Enable endpoint.
- Without DMA:
  - TXRDY: An interrupt is generated after each transmission.
  - EPT_ENABL: Enable endpoint.

Configuration examples of Bulk OUT endpoint type follow below.

- With DMA
  - AUTO_VALID: Automatically validate the packet and switch to the next bank.
  - EPT_ENABL: Enable endpoint.
- Without DMA
  - RXRDY_TXKL: An interrupt is sent after a new packet has been stored in the endpoint FIFO.
  - EPT_ENABL: Enable endpoint.

34.6.7 DPRAM Management

Endpoints can only be allocated in ascending order, from the endpoint 0 to the last endpoint to be allocated. The user shall therefore configure them in the same order.

The allocation of an endpoint x starts when the Number of Banks field in the UDPHS Endpoint Configuration Register (UDPHS_EPTCFGx.BK_NUMBER) is different from zero. Then, the hardware allocates a memory area in the DPRAM and inserts it between the x-1 and x+1 endpoints. The x+1 endpoint memory window slides up and its data is lost. Note that the following endpoint memory windows (from x+2) do not slide.

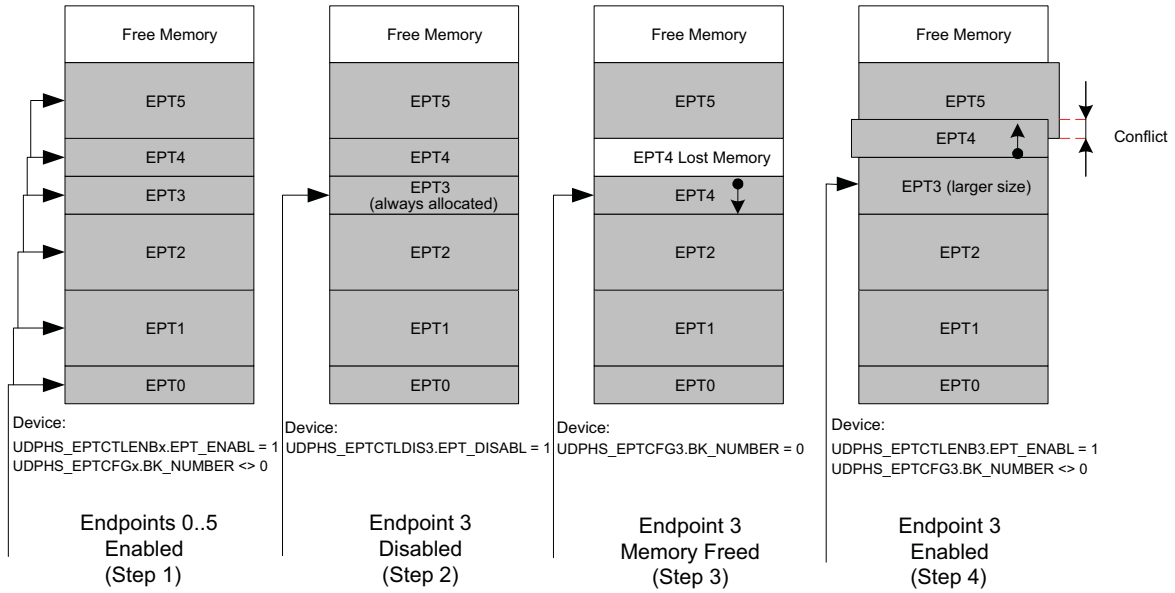
Disabling an endpoint, by writing a one to the Endpoint Disable bit in the UDPHS Endpoint Control Disable Register (UDPHS_EPTCTLDISx.EPT_DISABL), does not reset its configuration:

- Endpoint Banks (UDPHS_EPTCFGx.BK_NUMBER)
- Endpoint Size (UDPHS_EPTCFGx.EPT_SIZE)
- Endpoint Direction (UDPHS_EPTCFGx.EPT_DIR)
- Endpoint Type (UDPHS_EPTCFGx.EPT_TYPE)

To free its memory, the user shall write a zero to the UDPHS_EPTCFGx.BK_NUMBER field. The x+1 endpoint memory window then slides down and its data is lost. Note that the following endpoint memory windows (from x+2) do not slide.

Figure 34-6 illustrates the allocation and reorganization of the DPRAM in a typical example.

Figure 34-6. Example of DPRAM Allocation and Reorganization



DPRAM allocation sequence:

1. The endpoints 0 to 5 are enabled, configured and allocated in ascending order. Each endpoint then owns a memory area in the DPRAM.
2. The endpoint 3 is disabled, but its memory is kept allocated by the controller.
3. In order to free its memory, its UDPHS_EPTCFGx.BK_NUMBER field is written to zero. The endpoint 4 memory window slides down, but the endpoint 5 does not move.
4. If the user chooses to reconfigure the endpoint 3 with a larger size, the controller allocates a memory area after the endpoint 2 memory area and automatically slides up the endpoint 4 memory window. The endpoint 5 does not move and a memory conflict appears as the memory windows of the endpoints 4 and 5 overlap. The data of these endpoints is potentially lost.

Notes:
1. There is no way the data of the endpoint 0 can be lost (except if it is de-allocated) as the memory allocation and de-allocation may affect only higher endpoints.
2. Deactivating then reactivating the same endpoint with the same configuration only modifies temporarily the controller DPRAM pointer and size for this endpoint. Nothing changes in the DPRAM, higher endpoints seem not to have been moved and their data is preserved as far as nothing has been written or received into them while changing the allocation state of the first endpoint.
3. When the user writes a value different from zero to the UDPHS_EPTCFGx.BK_NUMBER field, the Endpoint Mapped bit (UDPHS_EPTCFGx.EPT_MAPD) is set only if the configured size and number of banks are correct as compared to the endpoint maximal allowed values and to the maximal FIFO size (i.e., the DPRAM size). The UDPHS_EPTCFGx.EPT_MAPD value does not consider memory allocation conflicts.

34.6.8 Transfer With DMA

USB packets of any length may be transferred when required by the UDPHS device. These transfers always feature sequential addressing.

Packet data AHB bursts may be locked on a DMA buffer basis for drastic overall AHB bus bandwidth performance boost with paged memories. These clock-cycle consuming memory row (or bank) changes will then likely not occur, or occur only once instead of several times, during a single big USB packet DMA transfer in case another AHB master addresses the memory. The locked bursts result in up to 128-word single-cycle unbroken AHB bursts for bulk endpoints and 256-word single-cycle unbroken bursts for isochronous endpoints.

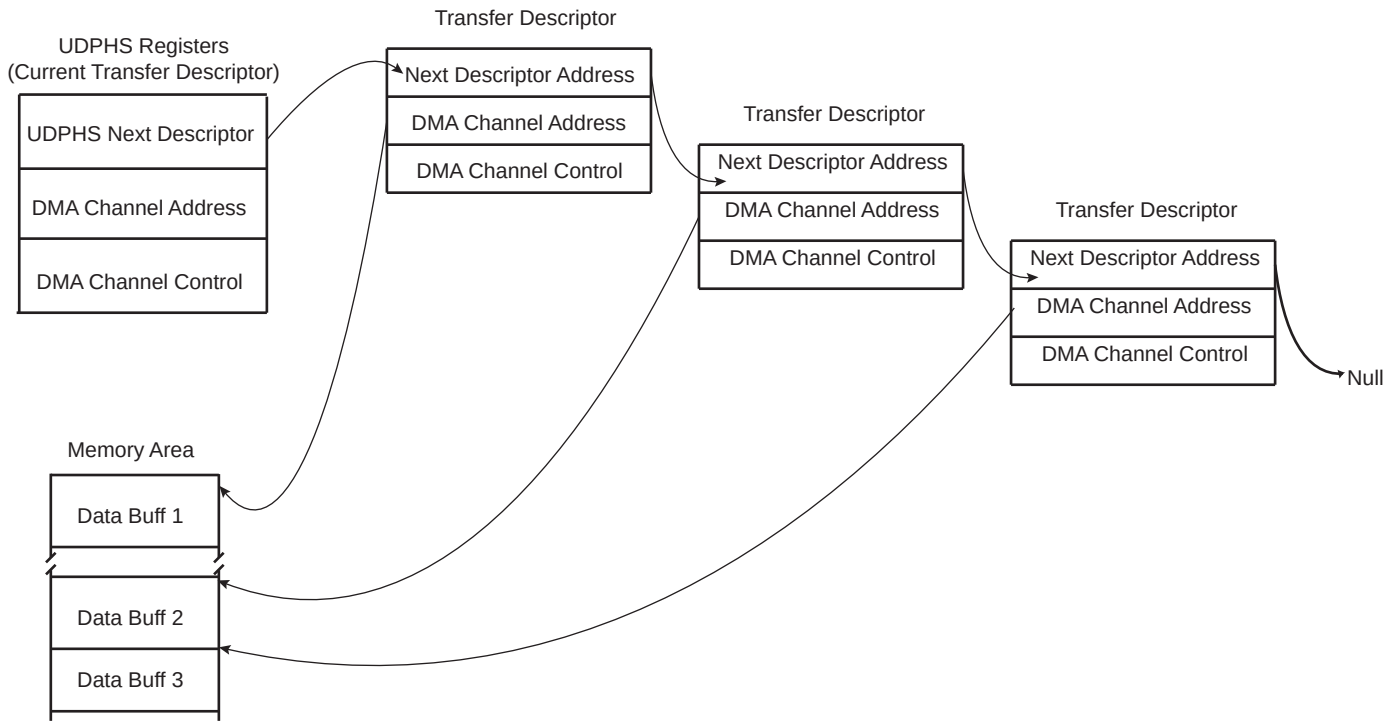
This maximum burst length is then controlled by the lowest programmed USB endpoint size (EPT_SIZE field in the UDPHS_EPTCFGx register) and DMA Size (BUFF_LENGTH field in the UDPHS_DMACONTROLx register).

The USB 2.0 device average throughput may be up to nearly 60 Mbyte/s. Its internal slave average access latency decreases as burst length increases due to the 0 wait-state side effect of unchanged endpoints. If at least 0 wait-state word burst capability is also provided by the external DMA AHB bus slaves, each of both DMA AHB busses need less than 50% bandwidth allocation for full USB 2.0 bandwidth usage at 30 MHz, and less than 25% at 60 MHz.

The UDPHS DMA Channel Transfer Descriptor is described in Section 34.7.21 “UDPHS DMA Channel Transfer Descriptor”.

Note: In case of debug, be careful to address the DMA to an SRAM address even if a remap is done.

Figure 34-7. Example of DMA Chained List



34.6.9 Transfer Without DMA

Important: If the DMA is not to be used, it is necessary to disable it, otherwise it can be enabled by previous versions of software **without warning**. If this should occur, the DMA can process data before an interrupt without knowledge of the user.

The recommended means to disable DMA are as follows:

```
// Reset IP UDPHS
    AT91C_BASE_UDPHS->UDPHS_CTRL &= ~AT91C_UDPHS_EN_UDPHS;
    AT91C_BASE_UDPHS->UDPHS_CTRL |= AT91C_UDPHS_EN_UDPHS;
// With OR without DMA !!!
    for( i=1; i<=((AT91C_BASE_UDPHS->UDPHS_IPFEATURES &
AT91C_UDPHS_DMA_CHANNEL_NBR)>>4); i++ ) {
// RESET endpoint canal DMA:
        // DMA stop channel command
        AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMACONTROL = 0;  // STOP
command
// Disable endpoint
```

```
        AT91C_BASE_UDPHS->UDPHS_EPT[i].UDPHS_EPTCTLDIS |= 0XFFFFFFFF;
// Reset endpoint config
        AT91C_BASE_UDPHS->UDPHS_EPT[i].UDPHS_EPTCTLCFG = 0;
// Reset DMA channel (Buff count and Control field)
        AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMACONTROL = 0x02;  // NON
STOP command
// Reset DMA channel 0 (STOP)
        AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMACONTROL = 0;  // STOP
command
// Clear DMA channel status (read the register for clear it)
        AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMASTATUS =
AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMASTATUS;
}
```

34.6.10 Handling Transactions with USB V2.0 Device Peripheral

34.6.10.1Setup Transaction

The setup packet is valid in the DPR while RX_SETUP is set. Once RX_SETUP is cleared by the application, the UDPHS accepts the next packets sent over the device endpoint.

When a valid setup packet is accepted by the UDPHS:

- The UDPHS device automatically acknowledges the setup packet (sends an ACK response)
- Payload data is written in the endpoint
- Sets the RX_SETUP interrupt
- The BYTE_COUNT field in the UDPHS_EPTSTAx register is updated

An endpoint interrupt is generated while RX_SETUP in the UDPHS_EPTSTAx register is not cleared. This interrupt is carried out to the microcontroller if interrupts are enabled for this endpoint.

Thus, firmware must detect RX_SETUP polling UDPHS_EPTSTAx or catching an interrupt, read the setup packet in the FIFO, then clear the RX_SETUP bit in the UDPHS_EPTCLRSTA register to acknowledge the setup stage.

If STALL_SNT was set to 1, then this bit is automatically reset when a setup token is detected by the device. Then, the device still accepts the setup stage. (Refer to Section 34.6.10.5 “STALL”).

34.6.10.2NYET

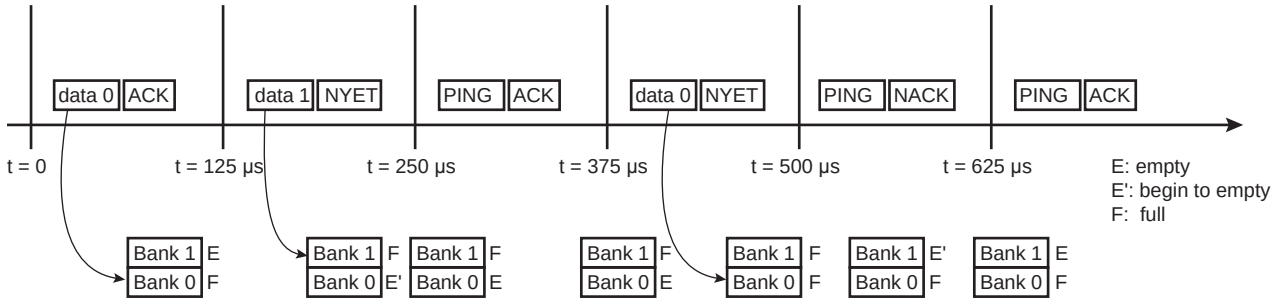
NYET is a High Speed only handshake. It is returned by a High Speed endpoint as part of the PING protocol.

High Speed devices must support an improved NAK mechanism for Bulk OUT and control endpoints (except setup stage). This mechanism allows the device to tell the host whether it has sufficient endpoint space for the next OUT transfer (refer to USB 2.0 spec 8.5.1 NAK Limiting via Ping Flow Control).

The NYET/ACK response to a High Speed Bulk OUT transfer and the PING response are automatically handled by hardware in the UDPHS_EPTCTLx register (except when the user wants to force a NAK response by using the NYET_DIS bit).

If the endpoint responds instead to the OUT/DATA transaction with an NYET handshake, this means that the endpoint accepted the data but does not have room for another data payload. The host controller must return to using a PING token until the endpoint indicates it has space available.

Figure 34-8. NYET Example with Two Endpoint Banks



34.6.10.3Data IN

Bulk IN or Interrupt IN

Data IN packets are sent by the device during the data or the status stage of a control transfer or during an (interrupt/bulk/isochronous) IN transfer. Data buffers are sent packet by packet under the control of the application or under the control of the DMA channel.

There are three ways for an application to transfer a buffer in several packets over the USB:

- packet by packet
  (refer to ”Bulk IN or Interrupt IN: Sending a Packet Under Application Control (Device to Host)”)
- 64 KB
  (refer to ”Bulk IN or Interrupt IN: Sending a Packet Under Application Control (Device to Host)”)
- DMA
  (refer to ”Bulk IN or Interrupt IN: Sending a Buffer Using DMA (Device to Host)”)

Bulk IN or Interrupt IN: Sending a Packet Under Application Control (Device to Host)

The application can write one or several banks.

A simple algorithm can be used by the application to send packets regardless of the number of banks associated to the endpoint.

Algorithm Description for Each Packet:

- The application waits for TXRDY flag to be cleared in the UDPHS_EPTSTAx register before it can perform a write access to the DPR.
- The application writes one USB packet of data in the DPR through the 64 KB endpoint logical memory window.
- The application sets TXRDY flag in the UDPHS_EPTSETSTAx register.

The application is notified that it is possible to write a new packet to the DPR by the TXRDY interrupt. This interrupt can be enabled or masked by setting the TXRDY bit in the UDPHS_EPTCTLENB/UDPHS_EPTCTLDIS register.

Algorithm Description to Fill Several Packets:

Using the previous algorithm, the application is interrupted for each packet. It is possible to reduce the application overhead by writing linearly several banks at the same time. The AUTO_VALID bit in the UDPHS_EPTCTLx must be set by writing the AUTO_VALID bit in the UDPHS_EPTCTLENBx register.

The auto-valid-bank mechanism allows the transfer of data (IN and OUT) without the intervention of the CPU. This means that bank validation (set TXRDY or clear the RXRDY_TXKL bit) is done by hardware.

- The application checks the BUSY_BANK_STA field in the UDPHS_EPTSTAx register. The application must wait that at least one bank is free.
- The application writes a number of bytes inferior to the number of free DPR banks for the endpoint. Each time the application writes the last byte of a bank, the TXRDY signal is automatically set by the UDPHS.

- If the last packet is incomplete (i.e., the last byte of the bank has not been written) the application must set the TXRDY bit in the UDPHS_EPTSETSTAx register.

The application is notified that all banks are free, so that it is possible to write another burst of packets by the BUSY_BANK interrupt. This interrupt can be enabled or masked by setting the BUSY_BANK flag in the UDPHS_EPTCTLENB and UDPHS_EPTCTLDIS registers.

This algorithm must not be used for isochronous transfer. In this case, the ping-pong mechanism does not operate.

A Zero Length Packet can be sent by setting just the TXRDY flag in the UDPHS_EPTSETSTAx register.

Bulk IN or Interrupt IN: Sending a Buffer Using DMA (Device to Host)

The UDPHS integrates a DMA host controller. This DMA controller can be used to transfer a buffer from the memory to the DPR or from the DPR to the processor memory under the UDPHS control. The DMA can be used for all transfer types except control transfer.

Example DMA configuration:

1. Program UDPHS_DMAADDRESS x with the address of the buffer that should be transferred.
2. Enable the interrupt of the DMA in UDPHS_IEN
3. Program UDPHS_ DMACONTROLx:
   - Size of buffer to send: size of the buffer to be sent to the host.
   - END_B_EN: The endpoint can validate the packet (according to the values programmed in the AUTO_VALID and SHRT_PCKT fields of UDPHS_EPTCTLx.) (Refer to Section 34.7.12 “UDPHS Endpoint Control Disable Register (Isochronous Endpoint)” and Figure 34-13)
   - END_BUFFIT: generate an interrupt when the BUFF_COUNT in UDPHS_DMASTATUSx reaches 0.
   - CHANN_ENB: Run and stop at end of buffer

The auto-valid-bank mechanism allows the transfer of data (IN & OUT) without the intervention of the CPU. This means that bank validation (set TXRDY or clear the RXRDY_TXKL bit) is done by hardware.

A transfer descriptor can be used. Instead of programming the register directly, a descriptor should be programmed and the address of this descriptor is then given to UDPHS_DMANXTDSC to be processed after setting the LDNXT_DSC field (Load Next Descriptor Now) in UDPHS_DMACONTROLx register.

The structure that defines this transfer descriptor must be aligned.

Each buffer to be transferred must be described by a DMA Transfer descriptor (refer to Section 34.7.21 “UDPHS DMA Channel Transfer Descriptor”). Transfer descriptors are chained. Before executing transfer of the buffer, the UDPHS may fetch a new transfer descriptor from the memory address pointed by the UDPHS_DMANXTDSCx register. Once the transfer is complete, the transfer status is updated in the UDPHS_DMASTATUSx register.

To chain a new transfer descriptor with the current DMA transfer, the DMA channel must be stopped. To do so, INTDIS_DMA and TXRDY may be set in the UDPHS_EPTCTLENBx register. It is also possible for the application to wait for the completion of all transfers. In this case the LDNXT_DSC bit in the last transfer descriptor UDPHS_DMACONTROLx register must be set to 0 and the CHANN_ENB bit set to 1.

Then the application can chain a new transfer descriptor.

The INTDIS_DMA can be used to stop the current DMA transfer if an enabled interrupt is triggered. This can be used to stop DMA transfers in case of errors.

The application can be notified at the end of any buffer transfer (ENB_BUFFIT bit in the UDPHS_DMACONTROLx register).

Figure 34-9. Data IN Transfer for Endpoint with One Bank

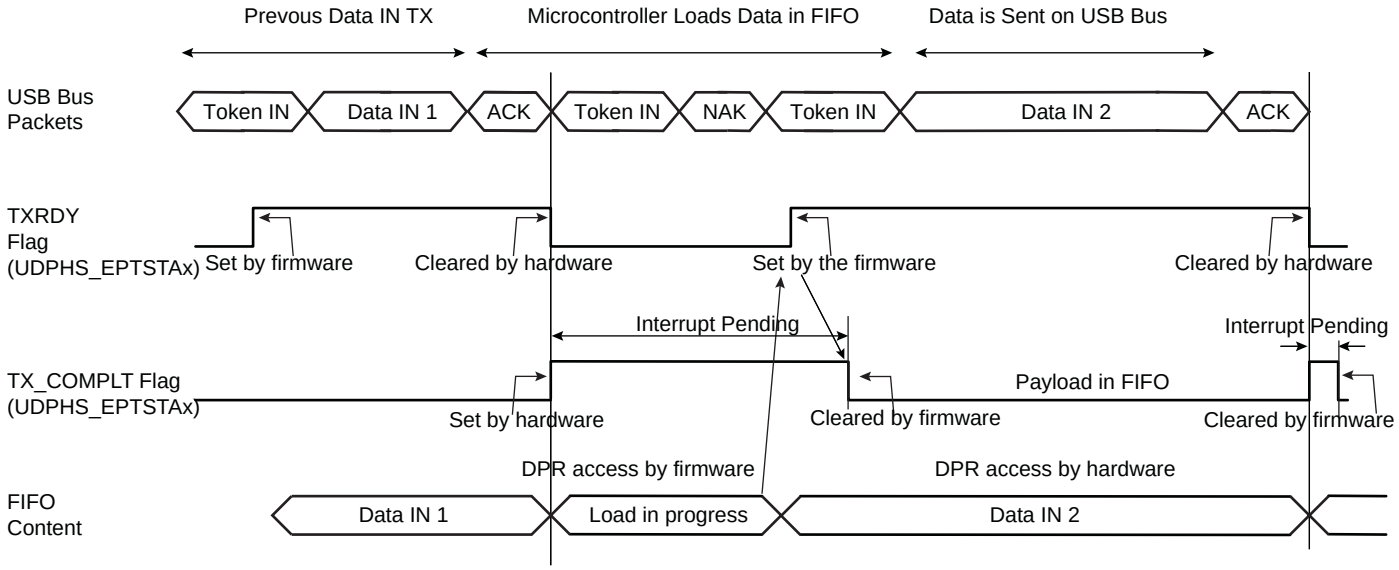


Figure 34-10. Data IN Transfer for Endpoint with Two Banks

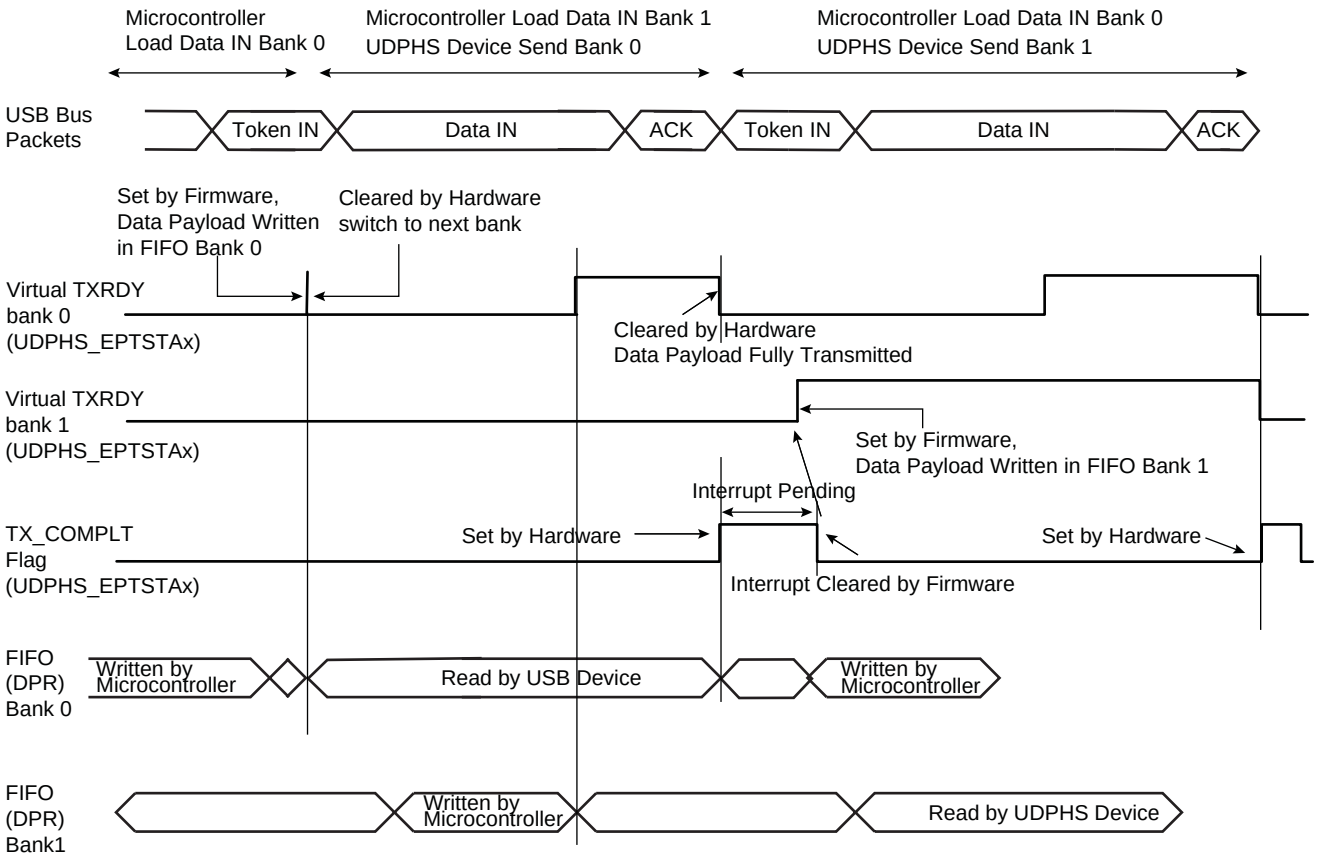
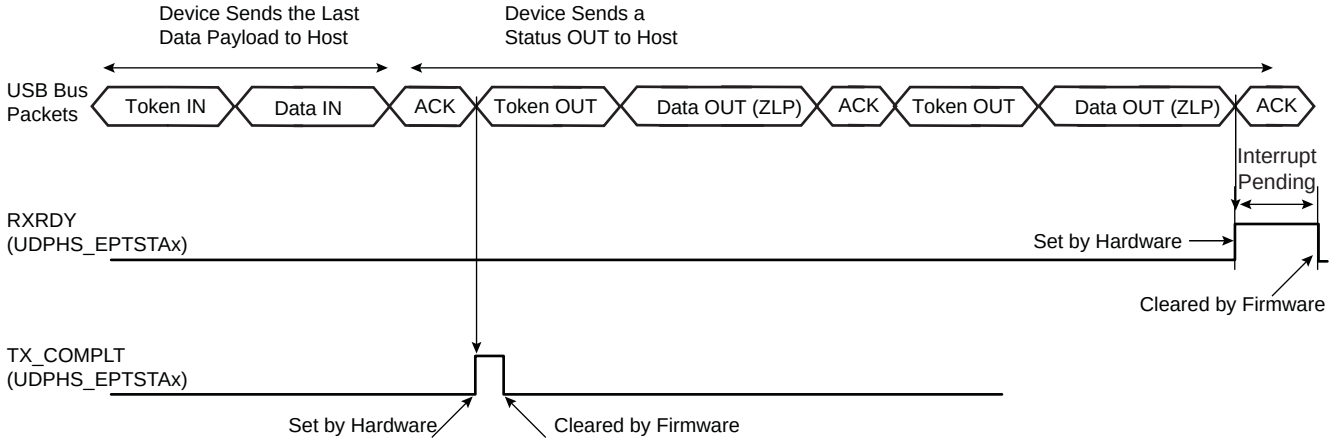
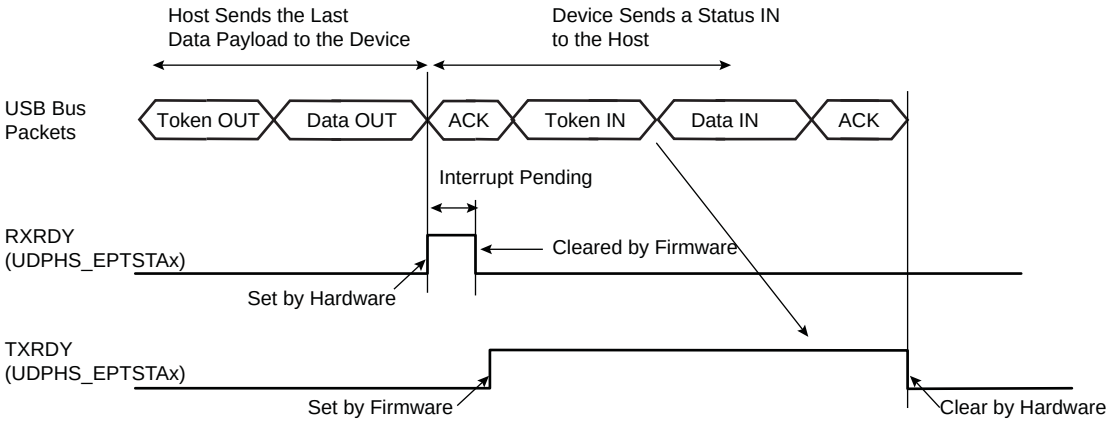


Figure 34-11. Data IN Followed By Status OUT Transfer at the End of a Control Transfer



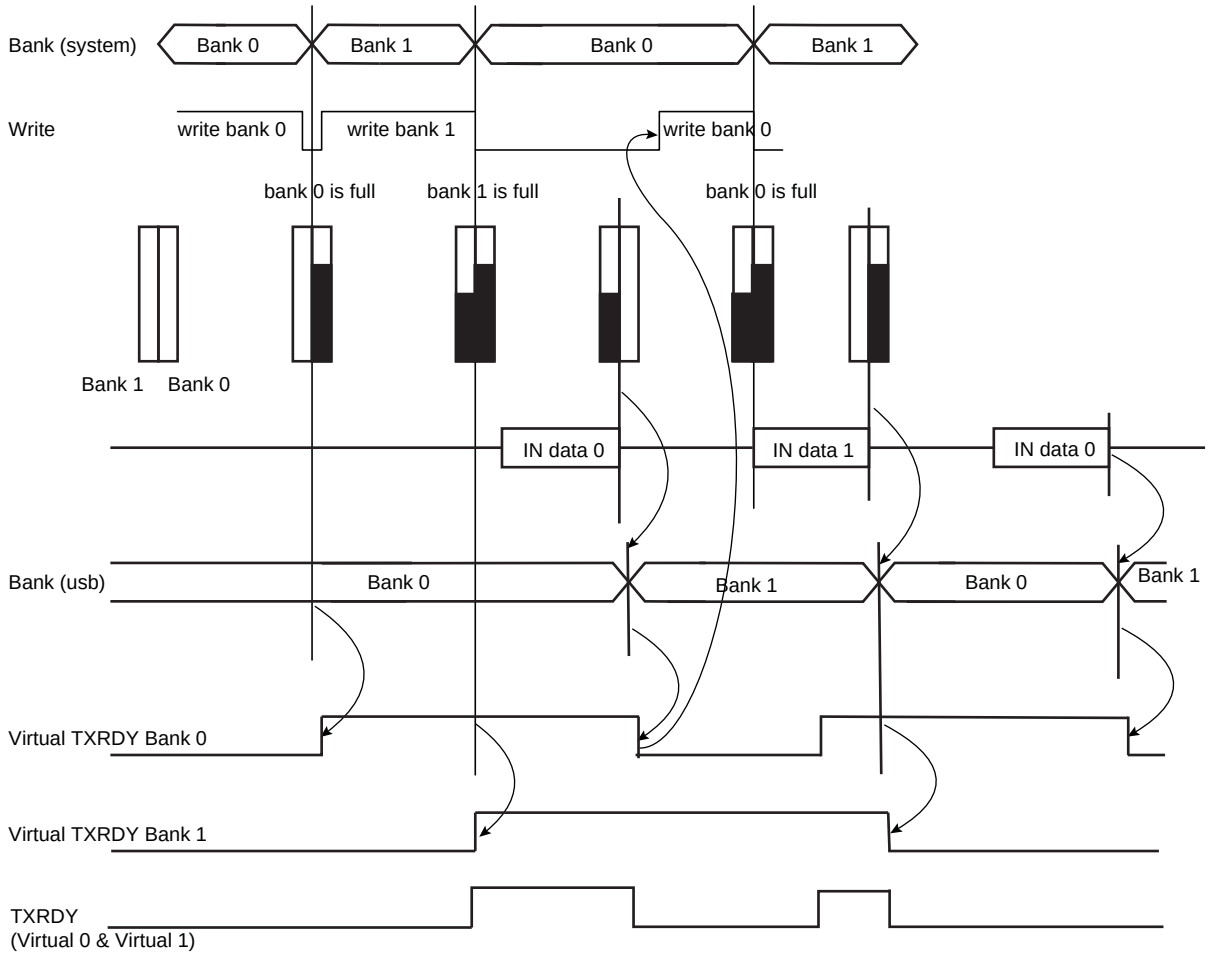
Note: A NAK handshake is always generated at the first status stage token.

Figure 34-12. Data OUT Followed by Status IN Transfer



Note: Before proceeding to the status stage, the software should determine that there is no risk of extra data from the host (data stage). If not certain (non-predictable data stage length), then the software should wait for a NAK-IN interrupt before proceeding to the status stage. This precaution should be taken to avoid collision in the FIFO.

Figure 34-13. Autovalid with DMA



Note: In the illustration above Autovalid validates a bank as full, although this might not be the case, in order to continue processing data and to send to DMA.

Isochronous IN

Isochronous-IN is used to transmit a stream of data whose timing is implied by the delivery rate. Isochronous transfer provides periodic, continuous communication between host and device.

It guarantees bandwidth and low latencies appropriate for telephony, audio, video, etc.

If the endpoint is not available (TXRDY_TRER = 0), then the device does not answer to the host. An ERR_FL_ISO interrupt is generated in the UDPHS_EPTSTAx register and once enabled, then sent to the CPU.

The STALL_SNT command bit is not used for an ISO-IN endpoint.

High Bandwidth Isochronous Endpoint Handling: IN Example

For high bandwidth isochronous endpoints, the DMA can be programmed with the number of transactions (BUFF_LENGTH field in UDPHS_DMACONTROLx) and the system should provide the required number of packets per microframe, otherwise, the host will notice a sequencing problem.

A response should be made to the first token IN recognized inside a microframe under the following conditions:

- If at least one bank has been validated, the correct DATAx corresponding to the programmed Number Of Transactions per Microframe (NB_TRANS) should be answered. In case of a subsequent missed or corrupted token IN inside the microframe, the USB 2.0 Core available data bank(s) that should normally have been transmitted during that microframe shall be flushed at its end. If this flush occurs, an error condition is flagged (ERR_FLUSH is set in UDPHS_EPTSTAx).

- If no bank is validated yet, the default DATA0 ZLP is answered and underflow is flagged (ERR_FL_ISO is set in UDPHS_EPTSTAx). Then, no data bank is flushed at microframe end.
- If no data bank has been validated at the time when a response should be made for the second transaction of NB_TRANS = 3 transactions microframe, a DATA1 ZLP is answered and underflow is flagged (ERR_FL_ISO is set in UDPHS_EPTSTAx). If and only if remaining untransmitted banks for that microframe are available at its end, they are flushed and an error condition is flagged (ERR_FLUSH is set in UDPHS_EPTSTAx).
- If no data bank has been validated at the time when a response should be made for the last programmed transaction of a microframe, a DATA0 ZLP is answered and underflow is flagged (ERR_FL_ISO is set in UDPHS_EPTSTAx). If and only if the remaining untransmitted data bank for that microframe is available at its end, it is flushed and an error condition is flagged (ERR_FLUSH is set in UDPHS_EPTSTAx).
- If at the end of a microframe no valid token IN has been recognized, no data bank is flushed and no error condition is reported.

At the end of a microframe in which at least one data bank has been transmitted, if less than NB_TRANS banks have been validated for that microframe, an error condition is flagged (ERR_TRANS is set in UDPHS_EPTSTAx).

Cases of Error (in UDPHS_EPTSTAx)

- ERR_FL_ISO: There was no data to transmit inside a microframe, so a ZLP is answered by default.
- ERR_FLUSH: At least one packet has been sent inside the microframe, but the number of token INs received is less than the number of transactions actually validated (TXRDY_TRER) and likewise with the NB_TRANS programmed.
- ERR_TRANS: At least one packet has been sent inside the microframe, but the number of token INs received is less than the number of programmed NB_TRANS transactions and the packets not requested were not validated.
- ERR_FL_ISO + ERR_FLUSH: At least one packet has been sent inside the microframe, but the data has not been validated in time to answer one of the following token INs.
- ERR_FL_ISO + ERR_TRANS: At least one packet has been sent inside the microframe, but the data has not been validated in time to answer one of the following token INs and the data can be discarded at the microframe end.
- ERR_FLUSH + ERR_TRANS: The first token IN has been answered and it was the only one received, a second bank has been validated but not the third, whereas NB_TRANS was waiting for three transactions.
- ERR_FL_ISO + ERR_FLUSH + ERR_TRANS: The first token IN has been treated, the data for the second Token IN was not available in time, but the second bank has been validated before the end of the microframe. The third bank has not been validated, but three transactions have been set in NB_TRANS.

34.6.10.4Data OUT

Bulk OUT or Interrupt OUT

Like data IN, data OUT packets are sent by the host during the data or the status stage of control transfer or during an interrupt/bulk/isochronous OUT transfer. Data buffers are sent packet by packet under the control of the application or under the control of the DMA channel.

Bulk OUT or Interrupt OUT: Receiving a Packet Under Application Control (Host to Device)

Algorithm Description for Each Packet:

- The application enables an interrupt on RXRDY_TXKL.
- When an interrupt on RXRDY_TXKL is received, the application knows that UDPHS_EPTSTAx register BYTE_COUNT bytes have been received.
- The application reads the BYTE_COUNT bytes from the endpoint.
- The application clears RXRDY_TXKL.

Note: If the application does not know the size of the transfer, it may **not** be a good option to use AUTO_VALID. Because if a zero-length-packet is received, the RXRDY_TXKL is automatically cleared by the AUTO_VALID hardware and if the endpoint interrupt is triggered, the software will not find its originating flag when reading the UDPHS_EPTSTAx register.

Algorithm to Fill Several Packets:

- The application enables the interrupts of BUSY_BANK and AUTO_VALID.
- When a BUSY_BANK interrupt is received, the application knows that all banks available for the endpoint have been filled. Thus, the application can read all banks available.

If the application does not know the size of the receive buffer, instead of using the BUSY_BANK interrupt, the application must use RXRDY_TXKL.

Bulk OUT or Interrupt OUT: Sending a Buffer Using DMA (Host To Device)

To use the DMA setting, the AUTO_VALID field is mandatory.

Refer to Bulk IN or Interrupt IN: Sending a Buffer Using DMA (Device to Host) for more information.

DMA Configuration Example:

1. First program UDPHS_DMAADDRESSx with the address of the buffer that should be transferred.
2. Enable the interrupt of the DMA in UDPHS_IEN
3. Program the DMA Channelx Control Register:
   - Size of buffer to be sent.
   - END_B_EN: Can be used for OUT packet truncation (discarding of unbuffered packet data) at the end of DMA buffer.
   - END_BUFFIT: Generate an interrupt when BUFF_COUNT in the UDPHS_DMASTATUSx register reaches 0.
   - END_TR_EN: End of transfer enable, the UDPHS device can put an end to the current DMA transfer, in case of a short packet.
   - END_TR_IT: End of transfer interrupt enable, an interrupt is sent after the last USB packet has been transferred by the DMA, if the USB transfer ended with a short packet. (Beneficial when the receive size is unknown.)
   - CHANN_ENB: Run and stop at end of buffer.

For OUT transfer, the bank will be automatically cleared by hardware when the application has read all the bytes in the bank (the bank is empty).

Notes:
1. When a zero-length-packet is received, RXRDY_TXKL bit in UDPHS_EPTSTAx is cleared automatically by AUTO_VALID, and the application knows of the end of buffer by the presence of the END_TR_IT.
2. If the host sends a zero-length packet, and the endpoint is free, then the device sends an ACK. No data is written in the endpoint, the RXRDY_TXKL interrupt is generated, and the BYTE_COUNT field in UDPHS_EPTSTAx is null.

Figure 34-14. Data OUT Transfer for Endpoint with One Bank

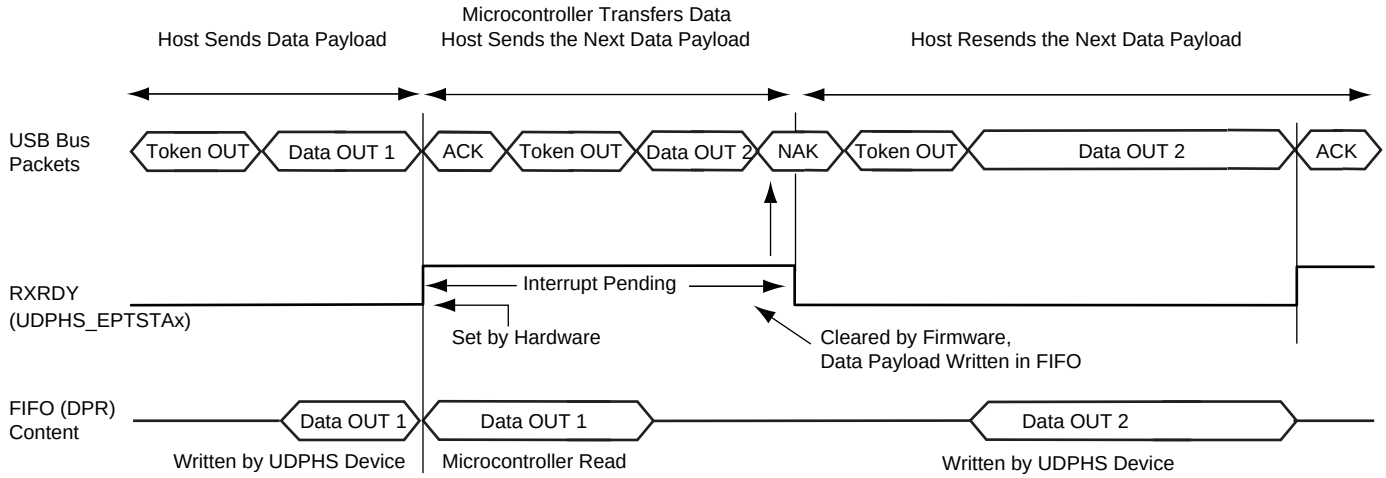
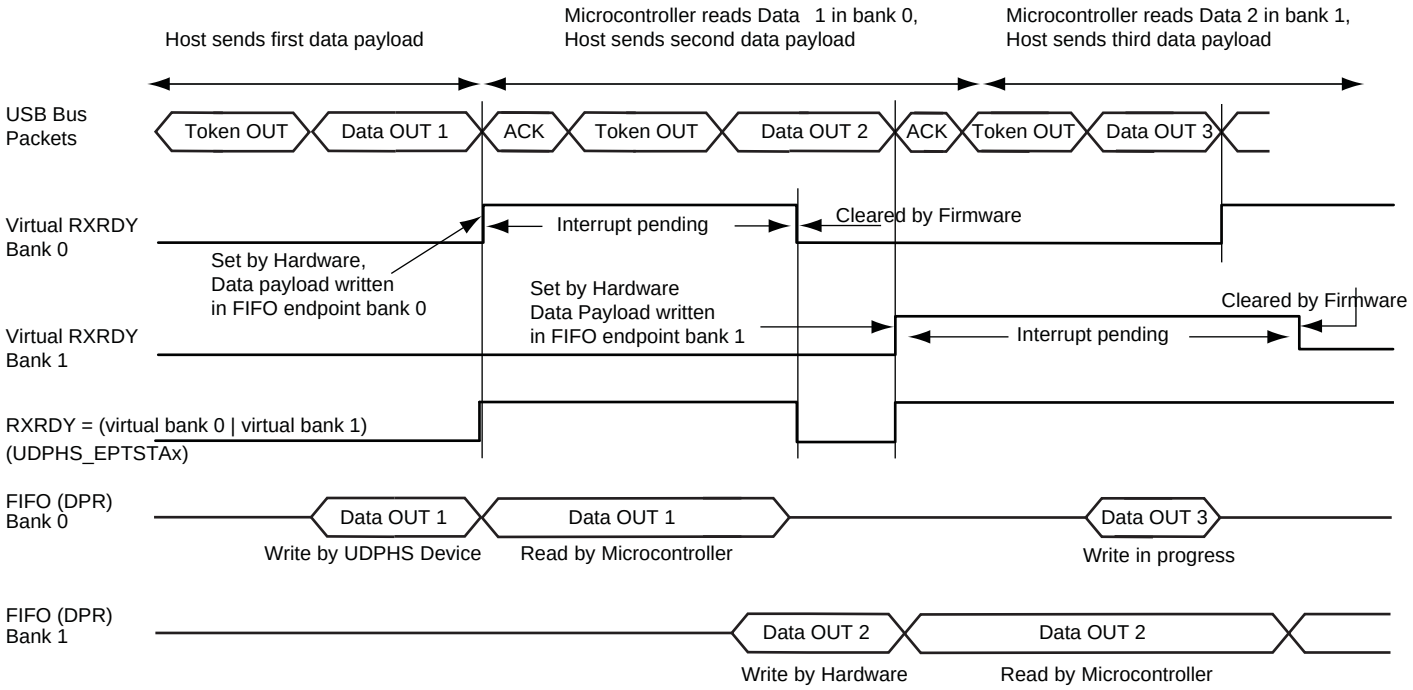


Figure 34-15. Data OUT Transfer for an Endpoint with Two Banks



High Bandwidth Isochronous Endpoint OUT

USB 2.0 supports individual High Speed isochronous endpoints that require data rates up to 192 Mb/s (24 MB/s): 3x1024 data bytes per microframe.

To support such a rate, two or three banks may be used to buffer the three consecutive data packets. The microcontroller (or the DMA) should be able to empty the banks very rapidly (at least 24 MB/s on average).

NB_TRANS field in UDPHS_EPTCFGx register = Number Of Transactions per Microframe.

If NB_TRANS > 1 then it is High Bandwidth.

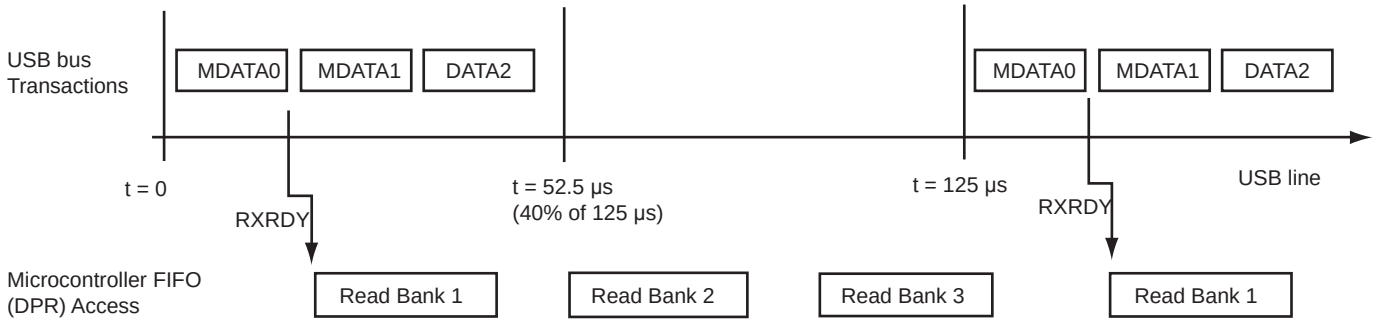
Example:

- If NB_TRANS = 3, the sequence should be either
  - MData0

- MData0/Data1
- MData0/Data1/Data2

- If NB_TRANS = 2, the sequence should be either
  - MData0
  - MData0/Data1
- If NB_TRANS = 1, the sequence should be
  - Data0

Figure 34-16. Bank Management, Example of Three Transactions per Microframe



Isochronous Endpoint Handling: OUT Example

The user can ascertain the bank status (free or busy), and the toggle sequencing of the data packet for each bank with the UDPHS_EPTSTAx register in the three fields as follows:

- TOGGLESQ_STA: PID of the data stored in the current bank
- CURBK: Number of the bank currently being accessed by the microcontroller.
- BUSY_BANK_STA: Number of busy bank

This is particularly useful in case of a missing data packet.

If the inter-packet delay between the OUT token and the Data is greater than the USB standard, then the ISO-OUT transaction is ignored. (Payload data is not written, no interrupt is generated to the CPU.)

If there is a data CRC (Cyclic Redundancy Check) error, the payload is, none the less, written in the endpoint. The ERR_CRC_NTR flag is set in UDPHS_EPTSTAx register.

If the endpoint is already full, the packet is not written in the DPRAM. The ERR_FL_ISO flag is set in UDPHS_EPTSTAx.

If the payload data is greater than the maximum size of the endpoint, then the ERR_OVFLW flag is set. It is the task of the CPU to manage this error. The data packet is written in the endpoint (except the extra data).

If the host sends a Zero Length Packet, and the endpoint is free, no data is written in the endpoint, the RXRDY_TXKL flag is set, and the BYTE_COUNT field in UDPHS_EPTSTAx register is null.

The FRCESTALL command bit is unused for an isochronous endpoint.

Otherwise, payload data is written in the endpoint, the RXRDY_TXKL interrupt is generated and the BYTE_COUNT in UDPHS_EPTSTAx register is updated.

34.6.10.5STALL

STALL is returned by a function in response to an IN token or after the data phase of an OUT or in response to a PING transaction. STALL indicates that a function is unable to transmit or receive data, or that a control pipe request is not supported.

- OUT

To stall an endpoint, set the FRCESTALL bit in UDPHS_EPTSETSTAx register and after the STALL_SNT flag has been set, set the TOGGLE_SEG bit in the UDPHS_EPTCLRSTAx register.

- IN

Set the FRCESTALL bit in UDPHS_EPTSETSTAx register.

Figure 34-17. Stall Handshake Data OUT Transfer

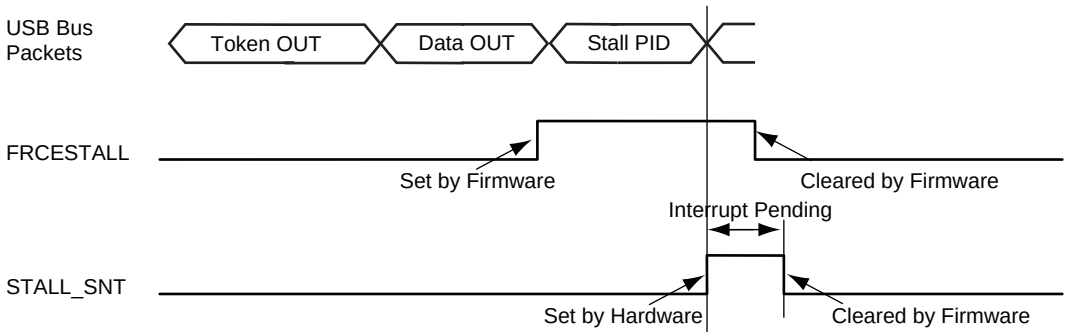
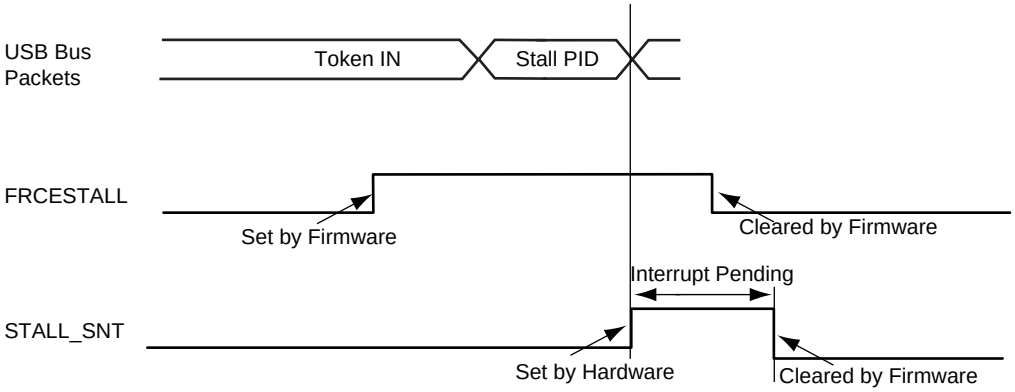


Figure 34-18. Stall Handshake Data IN Transfer



34.6.11 Speed Identification

The high speed reset is managed by hardware.

At the connection, the host makes a reset which could be a classic reset (full speed) or a high speed reset.

At the end of the reset process (full or high), the ENDRESET interrupt is generated.

Then the CPU should read the SPEED bit in UDPHS_INTSTAx to ascertain the speed mode of the device.

34.6.12 USB V2.0 High Speed Global Interrupt

Interrupts are defined in Section 34.7.3 “UDPHS Interrupt Enable Register” (UDPHS_IEN) and in Section 34.7.4 “UDPHS Interrupt Status Register” (UDPHS_INTSTA).

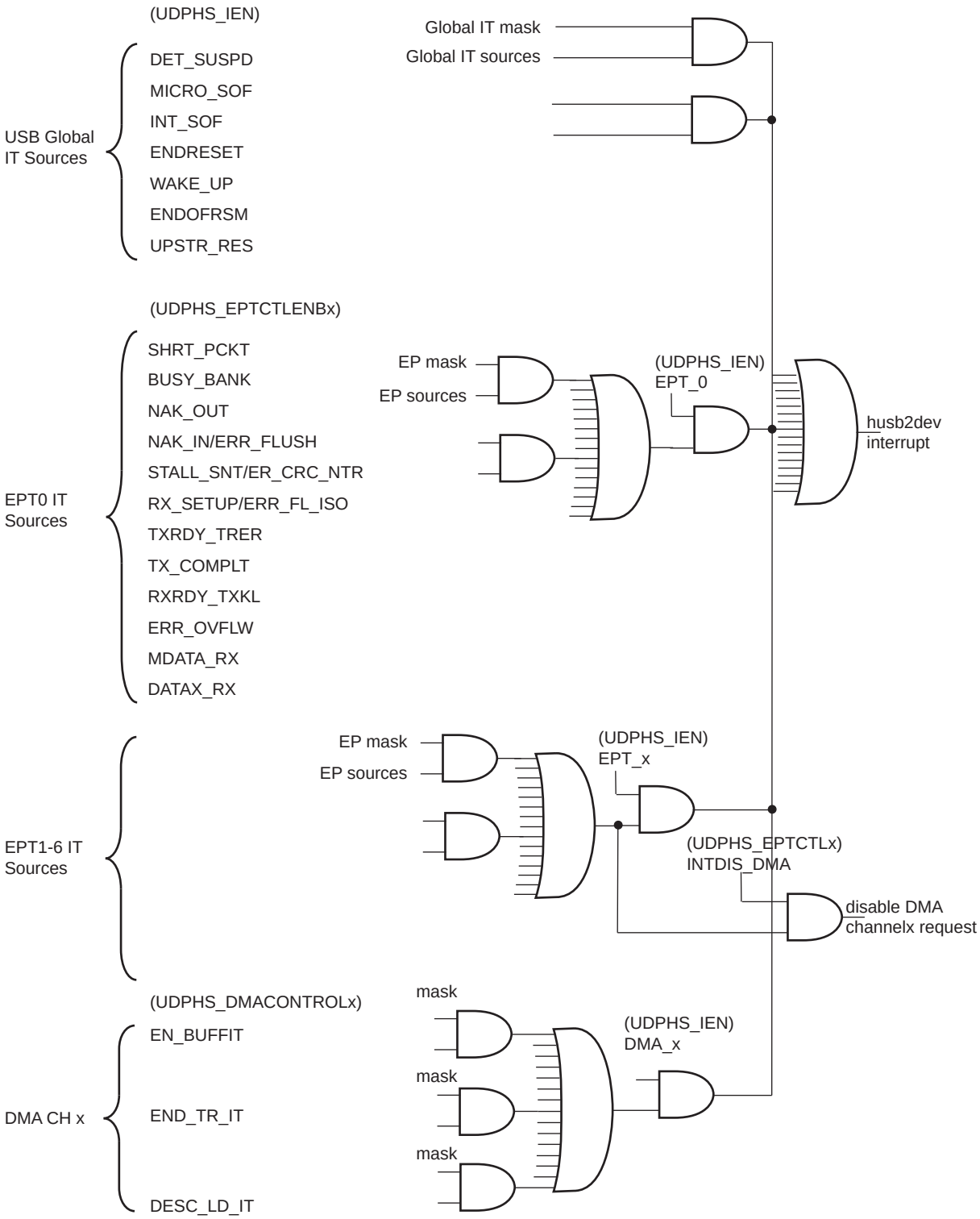
34.6.13 Endpoint Interrupts

Interrupts are enabled in UDPHS_IEN (refer to Section 34.7.3 “UDPHS Interrupt Enable Register”) and individually masked in UDPHS_EPTCTLENBx (refer to Section 34.7.9 “UDPHS Endpoint Control Enable Register (Control, Bulk, Interrupt Endpoints)”).

Table 34-5. Endpoint Interrupt Source Masks

SHRT_PCKT	Short Packet Interrupt
BUSY_BANK	Busy Bank Interrupt
NAK_OUT	NAKOUT Interrupt
NAK_IN/ERR_FLUSH	NAKIN/Error Flush Interrupt
STALL_SNT/ERR_CRC_NTR	Stall Sent/CRC error/Number of Transaction Error Interrupt
RX_SETUP/ERR_FL_ISO	Received SETUP/Error Flow Interrupt
TXRDY_TRER	TX Packet Read/Transaction Error Interrupt
TX_COMPLT	Transmitted IN Data Complete Interrupt
RXRDY_TXKL	Received OUT Data Interrupt
ERR_OVFLW	Overflow Error Interrupt
MDATA_RX	MDATA Interrupt
DATAX_RX	DATAx Interrupt

Figure 34-19. UDPHS Interrupt Control Interface

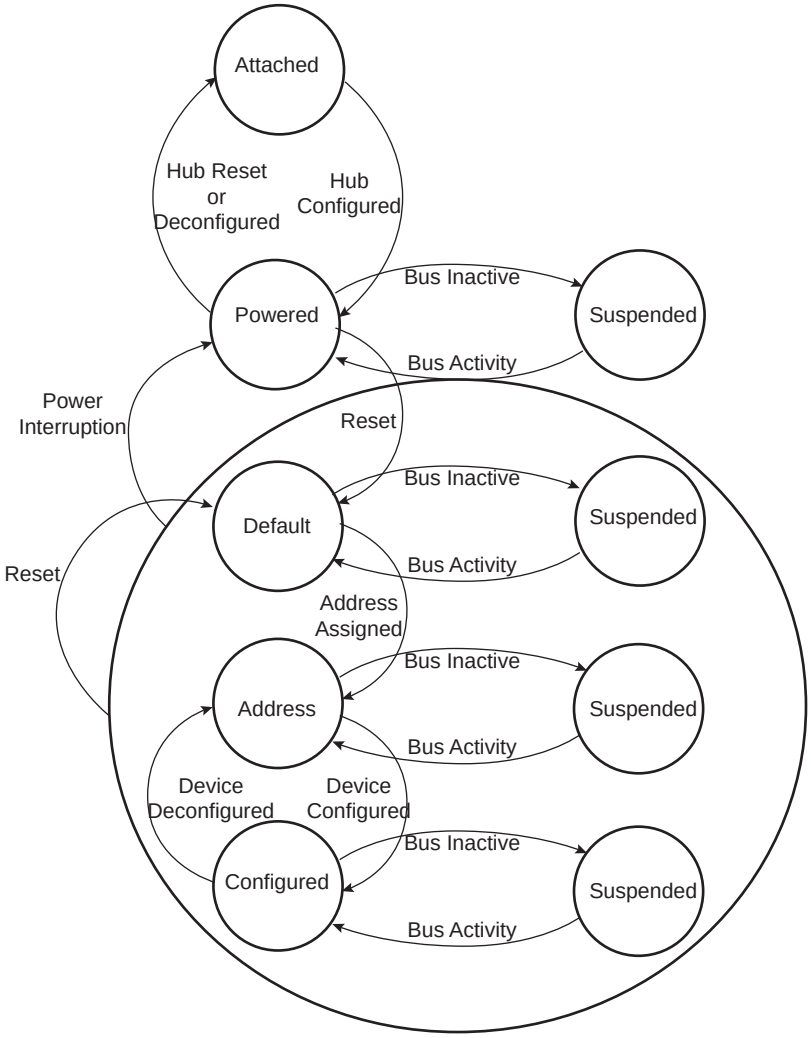


34.6.14 Power Modes

34.6.14.1Controlling Device States

A USB device has several possible states. Refer to Chapter 9 (USB Device Framework) of the Universal Serial Bus Specification, Rev 2.0.

Figure 34-20. UDPHS Device State Diagram



Movement from one state to another depends on the USB bus state or on standard requests sent through control transactions via the default endpoint (endpoint 0).

After a period of bus inactivity, the USB device enters Suspend mode. Accepting Suspend/Resume requests from the USB host is mandatory. Constraints in Suspend mode are very strict for bus-powered applications; devices may not consume more than 500 µA on the USB bus.

While in Suspend mode, the host may wake up a device by sending a resume signal (bus activity) or a USB device may send a wakeup request to the host, e.g., waking up a PC by moving a USB mouse.

The wakeup feature is not mandatory for all devices and must be negotiated with the host.

34.6.14.2Not Powered State

Self powered devices can detect 5V VBUS using a PIO. When the device is not connected to a host, device power consumption can be reduced by the DETACH bit in UDPHS_CTRL. Disabling the transceiver is automatically done. HSDM, HSDP, FSDP and FSDP lines are tied to GND pull-downs integrated in the hub downstream ports.

34.6.14.3Entering Attached State

When no device is connected, the USB FSDP and FSDM signals are tied to GND by 15 KΩ pull-downs integrated in the hub downstream ports. When a device is attached to an hub downstream port, the device connects a 1.5 KΩ

pull-up on FSDP. The USB bus line goes into IDLE state, FSDP is pulled-up by the device 1.5 KΩ resistor to 3.3V and FSDM is pulled-down by the 15 KΩ resistor to GND of the host.

After pull-up connection, the device enters the powered state. The transceiver remains disabled until bus activity is detected.

In case of low power consumption need, the device can be stopped. When the device detects the VBUS, the software must enable the USB transceiver by enabling the EN_UDPHS bit in UDPHS_CTRL register.

The software can detach the pull-up by setting DETACH bit in UDPHS_CTRL register.

34.6.14.4From Powered State to Default State (Reset)

After its connection to a USB host, the USB device waits for an end-of-bus reset. The unmasked flag ENDRESET is set in the UDPHS_IEN register and an interrupt is triggered.

Once the ENDRESET interrupt has been triggered, the device enters Default State. In this state, the UDPHS software must:

- Enable the default endpoint, setting the EPT_ENABL flag in the UDPHS_EPTCTLENB[0] register and, optionally, enabling the interrupt for endpoint 0 by writing 1 in EPT_0 of the UDPHS_IEN register. The enumeration then begins by a control transfer.
- Configure the Interrupt Mask Register which has been reset by the USB reset detection
- Enable the transceiver.

In this state, the EN_UDPHS bit in UDPHS_CTRL register must be enabled.

34.6.14.5From Default State to Address State (Address Assigned)

After a Set Address standard device request, the USB host peripheral enters the address state.

Warning: before the device enters address state, it must achieve the Status IN transaction of the control transfer, i.e., the UDPHS device sets its new address once the TX_COMPLT flag in the UDPHS_EPTCTL[0] register has been received and cleared.

To move to address state, the driver software sets the DEV_ADDR field and the FADDR_EN flag in the UDPHS_CTRL register.

34.6.14.6From Address State to Configured State (Device Configured)

Once a valid Set Configuration standard request has been received and acknowledged, the device enables endpoints corresponding to the current configuration. This is done by setting the BK_NUMBER, EPT_TYPE, EPT_DIR and EPT_SIZE fields in the UDPHS_EPTCFGx registers and enabling them by setting the EPT_ENABL flag in the UDPHS_EPTCTLENBx registers, and, optionally, enabling corresponding interrupts in the UDPHS_IEN register.

34.6.14.7Entering Suspend State (Bus Activity)

When a Suspend (no bus activity on the USB bus) is detected, the DET_SUSPD signal in the UDPHS_STA register is set. This triggers an interrupt if the corresponding bit is set in the UDPHS_IEN register. This flag is cleared by writing to the UDPHS_CLRINT register. Then the device enters Suspend mode.

In this state bus powered devices must drain less than 500 µA from the 5V VBUS. As an example, the microcontroller switches to slow clock, disables the PLL and main oscillator, and goes into Idle mode. It may also switch off other devices on the board.

The UDPHS device peripheral clocks can be switched off. Resume event is asynchronously detected.

34.6.14.8Receiving a Host Resume

In Suspend mode, a resume event on the USB bus line is detected asynchronously, transceiver and clocks disabled (however the pull-up should not be removed).

Once the resume is detected on the bus, the signal WAKE_UP in the UDPHS_INTSTA is set. It may generate an interrupt if the corresponding bit in the UDPHS_IEN register is set. This interrupt may be used to wake up the core, enable PLL and main oscillators and configure clocks.

34.6.14.9Sending an External Resume

In Suspend State it is possible to wake up the host by sending an external resume.

The device waits at least 5 ms after being entered in Suspend State before sending an external resume.

The device must force a K state from 1 to 15 ms to resume the host.

34.6.15 Test Mode

A device must support the TEST_MODE feature when in the Default, Address or Configured High Speed device states.

TEST_MODE can be:

- Test_J
- Test_K
- Test_Packet
- Test_SEO_NAK

(Refer to Section 34.7.7 “UDPHS Test Register” for definitions of each test mode.)

```
const char test_packet_buffer[] = {
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,                // JKJKJKJK * 9
  0xAA,0xAA,0xAA,0xAA,0xAA,0xAA,0xAA,0xAA,                     // JJKKJJKK * 8
  0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,                     // JJKKJJKK * 8
  0xFE,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF, //
JJJJJJJKKKKKKK * 8
  0x7F,0xBF,0xDF,0xEF,0xF7,0xFB,0xFD,                          // JJJJJJJK * 8
  0xFC,0x7E,0xBF,0xDF,0xEF,0xF7,0xFB,0xFD,0x7E                 // {JKKKKKKK *
10}, JK
};
```

34.7 USB High Speed Device Port (UDPHS) User Interface

Table 34-6. Register Mapping

Offset	Register	Name	Access	Reset
0x00	UDPHS Control Register	UDPHS_CTRL	Read/Write	0x0000_0200
0x04	UDPHS Frame Number Register	UDPHS_FNUM	Read-only	0x0000_0000
0x08–0x0C	Reserved	–	–	–
0x10	UDPHS Interrupt Enable Register	UDPHS_IEN	Read/Write	0x0000_0010
0x14	UDPHS Interrupt Status Register	UDPHS_INTSTA	Read-only	0x0000_0000
0x18	UDPHS Clear Interrupt Register	UDPHS_CLRINT	Write-only	–
0x1C	UDPHS Endpoints Reset Register	UDPHS_EPTRST	Write-only	–
0x20–0xCC	Reserved	–	–	–
0xE0	UDPHS Test Register	UDPHS_TST	Read/Write	0x0000_0000
0xE4–0xFC	Reserved	–	–	–
0x100 + endpoint * 0x20 + 0x00	UDPHS Endpoint Configuration Register	UDPHS_EPTCFG	Read/Write	0x0000_0000
0x100 + endpoint * 0x20 + 0x04	UDPHS Endpoint Control Enable Register	UDPHS_EPTCTLENB	Write-only	–
0x100 + endpoint * 0x20 + 0x08	UDPHS Endpoint Control Disable Register	UDPHS_EPTCTLDIS	Write-only	–
0x100 + endpoint * 0x20 + 0x0C	UDPHS Endpoint Control Register	UDPHS_EPTCTL	Read-only	0x0000_0000(1)
0x100 + endpoint * 0x20 + 0x10	Reserved (for endpoint)	–	–	–
0x100 + endpoint * 0x20 + 0x14	UDPHS Endpoint Set Status Register	UDPHS_EPTSETSTA	Write-only	–
0x100 + endpoint * 0x20 + 0x18	UDPHS Endpoint Clear Status Register	UDPHS_EPTCLRSTA	Write-only	–
0x100 + endpoint * 0x20 + 0x1C	UDPHS Endpoint Status Register	UDPHS_EPTSTA	Read-only	0x0000_0040
0x120–0x2FC	UDPHS Endpoint1 to 15 (2) Registers	–	–	–
0x300 + channel * 0x10 + 0x00	UDPHS DMA Next Descriptor Address Register	UDPHS_DMANXTDSC	Read/Write	0x0000_0000
0x300 + channel * 0x10 + 0x04	UDPHS DMA Channel Address Register	UDPHS_DMAADDRESS	Read/Write	0x0000_0000
0x300 + channel * 0x10 + 0x08	UDPHS DMA Channel Control Register	UDPHS_DMACONTROL	Read/Write	0x0000_0000
0x300 + channel * 0x10 + 0x0C	UDPHS DMA Channel Status Register	UDPHS_DMASTATUS	Read/Write	0x0000_0000
0x310–0x36C	DMA Channel1 to 6 (3) Registers	–	–	–

Notes:
1. The reset value for UDPHS_EPTCTL0 is 0x0000_0001.
2. The addresses for the UDPHS Endpoint registers shown here are for UDPHS Endpoint0. The structure of this group of registers is repeated successively for each endpoint according to the sequence of endpoint registers located between 0x120 and 0x2FC.
3. The DMA channel index refers to the corresponding EP number. When no DMA channel is assigned to one EP, the associated registers are reserved. This is the case for EP0, so DMA Channel 0 registers are reserved.

34.7.1 UDPHS Control Register

Name: UDPHS_CTRL

Address: 0xFC02C000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	PULLD_DIS	REWAKEUP	DETACH	EN_UDPHS
7	6	5	4	3	2	1	0
FADDR_EN	DEV_ADDR						

- **DEV_ADDR: UDPHS Address (cleared upon USB reset)**

This field contains the default address (0) after powerup or UDPHS bus reset (read), or it is written with the value set by a SET_ADDRESS request received by the device firmware (write).

- **FADDR_EN: Function Address Enable (cleared upon USB reset)**

0: Device is not in address state (read), or only the default function address is used (write).

1: Device is in address state (read), or this bit is set by the device firmware after a successful status phase of a SET_ADDRESS transaction (write). When set, the only address accepted by the UDPHS controller is the one stored in the UDPHS Address field. It will not be cleared afterwards by the device firmware. It is cleared by hardware on hardware reset, or when UDPHS bus reset is received.

- **EN_UDPHS: UDPHS Enable**

0: UDPHS is disabled (read), or this bit disables and resets the UDPHS controller (write). Switch the host to UTMI.

1: UDPHS is enabled (read), or this bit enables the UDPHS controller (write). Switch the host to UTMI.

- **DETACH: Detach Command**

0: UDPHS is attached (read), or this bit pulls up the DP line (attach command) (write).

1: UDPHS is detached, UTMI transceiver is suspended (read), or this bit simulates a detach on the UDPHS line and forces the UTMI transceiver into suspend state (Suspend M = 0) (write).

Refer to “PULLD_DIS: Pull-Down Disable (cleared upon USB reset)”.

- **REWAKEUP: Send Remote Wakeup (cleared upon USB reset)**

0: Remote Wakeup is disabled (read), or this bit has no effect (write).

1: Remote Wakeup is enabled (read), or this bit forces an external interrupt on the UDPHS controller for Remote Wakeup purposes.

An Upstream Resume is sent only after the UDPHS bus has been in SUSPEND state for at least 5 ms.

This bit is automatically cleared by hardware at the end of the Upstream Resume.

- **PULLD_DIS: Pull-Down Disable (cleared upon USB reset)**

When set, there is no pull-down on DP & DM. (DM Pull-Down = DP Pull-Down = 0).

Note: If the DETACH bit is also set, device DP & DM are left in high impedance state.

(Refer to “DETACH: Detach Command”.)

DETACH	PULLD_DIS	DP	DM	Condition
0	0	Pull up	Pull down	Not recommended
0	1	Pull up	High impedance state	VBUS present
1	0	Pull down	Pull down	No VBUS
1	1	High impedance state	High impedance state	VBUS present & software disconnect

34.7.2 UDPHS Frame Number Register

Name: UDPHS_FNUM

Address: 0xFC02C004

Access: Read-only

31	30	29	28	27	26	25	24
FNUM_ERR	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	FRAME_NUMBER					

7	6	5	4	3	2	1	0
FRAME_NUMBER					MICRO_FRAME_NUM		

- **MICRO_FRAME_NUM: Microframe Number (cleared upon USB reset)**

Number of the received microframe (0 to 7) in one frame.This field is reset at the beginning of each new frame (1 ms). One microframe is received each 125 microseconds (1 ms/8).

- **FRAME_NUMBER: Frame Number as defined in the Packet Field Formats (cleared upon USB reset)**

This field is provided in the last received SOF packet (refer to INT_SOF in the UDPHS Interrupt Status Register).

- **FNUM_ERR: Frame Number CRC Error (cleared upon USB reset)**

This bit is set by hardware when a corrupted Frame Number in Start of Frame packet (or Micro SOF) is received. This bit and the INT_SOF (or MICRO_SOF) interrupt are updated at the same time.

34.7.3 UDPHS Interrupt Enable Register

Name: UDPHS_IEN

Address: 0xFC02C010

Access: Read/Write

31	30	29	28	27	26	25	24
DMA_7	DMA_6	DMA_5	DMA_4	DMA_3	DMA_2	DMA_1	–
23	22	21	20	19	18	17	16
EPT_15	EPT_14	EPT_13	EPT_12	EPT_11	EPT_10	EPT_9	EPT_8
15	14	13	12	11	10	9	8
EPT_7	EPT_6	EPT_5	EPT_4	EPT_3	EPT_2	EPT_1	EPT_0
7	6	5	4	3	2	1	0
UPSTR_RES	ENDOFRSM	WAKE_UP	ENDRESET	INT_SOF	MICRO_SOF	DET_SUSPD	–

- **DET_SUSPD: Suspend Interrupt Enable (cleared upon USB reset)**

0: Disable Suspend Interrupt.

1: Enable Suspend Interrupt.

- **MICRO_SOF: Micro-SOF Interrupt Enable (cleared upon USB reset)**

0: Disable Micro-SOF Interrupt.

1: Enable Micro-SOF Interrupt.

- **INT_SOF: SOF Interrupt Enable (cleared upon USB reset)**

0: Disable SOF Interrupt.

1: Enable SOF Interrupt.

- **ENDRESET: End Of Reset Interrupt Enable (cleared upon USB reset)**

0: Disable End Of Reset Interrupt.

1: Enable End Of Reset Interrupt. Automatically enabled after USB reset.

- **WAKE_UP: Wakeup CPU Interrupt Enable (cleared upon USB reset)**

0: Disable Wakeup CPU Interrupt.

1: Enable Wakeup CPU Interrupt.

- **ENDOFRSM: End Of Resume Interrupt Enable (cleared upon USB reset)**

0: Disable Resume Interrupt.

1: Enable Resume Interrupt.

- **UPSTR_RES: Upstream Resume Interrupt Enable (cleared upon USB reset)**

0: Disable Upstream Resume Interrupt.

1: Enable Upstream Resume Interrupt.

- **EPT_x: Endpoint x Interrupt Enable (cleared upon USB reset)**

0: Disable the interrupts for this endpoint.

1: Enable the interrupts for this endpoint.

- **DMA_x: DMA Channel x Interrupt Enable (cleared upon USB reset)**

0: Disable the interrupts for this channel.

1: Enable the interrupts for this channel.

34.7.4 UDPHS Interrupt Status Register

Name: UDPHS_INTSTA

Address: 0xFC02C014

Access: Read-only

31	30	29	28	27	26	25	24
DMA_7	DMA_6	DMA_5	DMA_4	DMA_3	DMA_2	DMA_1	–
23	22	21	20	19	18	17	16
EPT_15	EPT_14	EPT_13	EPT_12	EPT_11	EPT_10	EPT_9	EPT_8
15	14	13	12	11	10	9	8
EPT_7	EPT_6	EPT_5	EPT_4	EPT_3	EPT_2	EPT_1	EPT_0
7	6	5	4	3	2	1	0
UPSTR_RES	ENDOFRSM	WAKE_UP	ENDRESET	INT_SOF	MICRO_SOF	DET_SUSPD	SPEED

- **SPEED: Speed Status**

0: Reset by hardware when the hardware is in Full Speed mode.

1: Set by hardware when the hardware is in High Speed mode.

- **DET_SUSPD: Suspend Interrupt**

0: Cleared by setting the DET_SUSPD bit in UDPHS_CLRINT register.

1: Set by hardware when a UDPHS Suspend (Idle bus for three frame periods, a J state for 3 ms) is detected. This triggers a UDPHS interrupt when the DET_SUSPD bit is set in UDPHS_IEN register.

- **MICRO_SOF: Micro Start Of Frame Interrupt**

0: Cleared by setting the MICRO_SOF bit in UDPHS_CLRINT register.

1: Set by hardware when an UDPHS micro start of frame PID (SOF) has been detected (every 125 us) or synthesized by the macro. This triggers a UDPHS interrupt when the MICRO_SOF bit is set in UDPHS_IEN. In case of detected SOF, the MICRO_FRAME_NUM field in UDPHS_FNUM register is incremented and the FRAME_NUMBER field does not change.

Note: The Micro Start Of Frame Interrupt (MICRO_SOF), and the Start Of Frame Interrupt (INT_SOF) are not generated at the same time.

- **INT_SOF: Start Of Frame Interrupt**

0: Cleared by setting the INT_SOF bit in UDPHS_CLRINT.

1: Set by hardware when an UDPHS Start Of Frame PID (SOF) has been detected (every 1 ms) or synthesized by the macro. This triggers a UDPHS interrupt when the INT_SOF bit is set in UDPHS_IEN register. In case of detected SOF, in High Speed mode, the MICRO_FRAME_NUMBER field is cleared in UDPHS_FNUM register and the FRAME_NUMBER field is updated.

- **ENDRESET: End Of Reset Interrupt**

0: Cleared by setting the ENDRESET bit in UDPHS_CLRINT.

1: Set by hardware when an End Of Reset has been detected by the UDPHS controller. This triggers a UDPHS interrupt when the ENDRESET bit is set in UDPHS_IEN.

- **WAKE_UP: Wakeup CPU Interrupt**

0: Cleared by setting the WAKE_UP bit in UDPHS_CLRINT.

1: Set by hardware when the UDPHS controller is in SUSPEND state and is re-activated by a filtered non-idle signal from the UDPHS line (not by an upstream resume). This triggers a UDPHS interrupt when the WAKE_UP bit is set in UDPHS_IEN register. When receiving this interrupt, the user has to enable the device controller clock prior to operation.

Note: this interrupt is generated even if the device controller clock is disabled.

- **ENDOFRSM: End Of Resume Interrupt**

0: Cleared by setting the ENDOFRSM bit in UDPHS_CLRINT.

1: Set by hardware when the UDPHS controller detects a good end of resume signal initiated by the host. This triggers a UDPHS interrupt when the ENDOFRSM bit is set in UDPHS_IEN.

- **UPSTR_RES: Upstream Resume Interrupt**

0: Cleared by setting the UPSTR_RES bit in UDPHS_CLRINT.

1: Set by hardware when the UDPHS controller is sending a resume signal called “upstream resume”. This triggers a UDPHS interrupt when the UPSTR_RES bit is set in UDPHS_IEN.

- **EPT_x: Endpoint x Interrupt (cleared upon USB reset)**

0: Reset when the UDPHS_EPTSTAx interrupt source is cleared.

1: Set by hardware when an interrupt is triggered by the UDPHS_EPTSTAx register and this endpoint interrupt is enabled by the EPT_x bit in UDPHS_IEN.

- **DMA_x: DMA Channel x Interrupt**

0: Reset when the UDPHS_DMASTATUSx interrupt source is cleared.

1: Set by hardware when an interrupt is triggered by the DMA Channelx and this endpoint interrupt is enabled by the DMA_x bit in UDPHS_IEN.

34.7.5 UDPHS Clear Interrupt Register

Name: UDPHS_CLRINT

Address: 0xFC02C018

Access: Write-only

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
UPSTR_RES	ENDOFRSM	WAKE_UP	ENDRESET	INT_SOF	MICRO_SOF	DET_SUSPD	–

- **DET_SUSPD: Suspend Interrupt Clear**

0: No effect.

1: Clear the DET_SUSPD bit in UDPHS_INTSTA.

- **MICRO_SOF: Micro Start Of Frame Interrupt Clear**

0: No effect.

1: Clear the MICRO_SOF bit in UDPHS_INTSTA.

- **INT_SOF: Start Of Frame Interrupt Clear**

0: No effect.

1: Clear the INT_SOF bit in UDPHS_INTSTA.

- **ENDRESET: End Of Reset Interrupt Clear**

0: No effect.

1: Clear the ENDRESET bit in UDPHS_INTSTA.

- **WAKE_UP: Wakeup CPU Interrupt Clear**

0: No effect.

1: Clear the WAKE_UP bit in UDPHS_INTSTA.

- **ENDOFRSM: End Of Resume Interrupt Clear**

0: No effect.

1: Clear the ENDOFRSM bit in UDPHS_INTSTA.

- **UPSTR_RES: Upstream Resume Interrupt Clear**

0: No effect.

1: Clear the UPSTR_RES bit in UDPHS_INTSTA.

34.7.6 UDPHS Endpoints Reset Register

Name: UDPHS_EPTRST

Address: 0xFC02C01C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
EPT_15	EPT_14	EPT_13	EPT_12	EPT_11	EPT_10	EPT_9	EPT_8
7	6	5	4	3	2	1	0
EPT_7	EPT_6	EPT_5	EPT_4	EPT_3	EPT_2	EPT_1	EPT_0

- **EPT_x: Endpoint x Reset**

0: No effect.

1: Reset the Endpointx state.

Setting this bit clears all bits in the Endpoint status UDPHS_EPTSTAx register except the TOGGLESQ_STA field.

34.7.7 UDPHS Test Register

Name: UDPHS_TST

Address: 0xFC02C0E0

Access: Read/Write

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
–	–	OPMODE2	TST_PKT	TST_K	TST_J	SPEED_CFG	

- **SPEED_CFG: Speed Configuration**

Value	Name	Description
0	NORMAL	Normal mode: The macro is in Full Speed mode, ready to make a High Speed identification, if the host supports it and then to automatically switch to High Speed mode.
1	–	Reserved
2	HIGH_SPEED	Force High Speed: Set this value to force the hardware to work in High Speed mode. Only for debug or test purpose.
3	FULL_SPEED	Force Full Speed: Set this value to force the hardware to work only in Full Speed mode. In this configuration, the macro will not respond to a High Speed reset handshake.

- **TST_J: Test J Mode**

0: No effect.

1: Set to send the J state on the UDPHS line. This enables the testing of the high output drive level on the D+ line.

- **TST_K: Test K Mode**

0: No effect.

1: Set to send the K state on the UDPHS line. This enables the testing of the high output drive level on the D- line.

- **TST_PKT: Test Packet Mode**

0: No effect.

1: Set to repetitively transmit the packet stored in the current bank. This enables the testing of rise and fall times, eye patterns, jitter, and any other dynamic waveform specifications.

- **OPMODE2: OpMode2**

0: No effect.

1: Set to force the OpMode signal (UTMI interface) to “10”, to disable the bit-stuffing and the NRZI encoding.

Note: For the Test mode, Test_SE0_NAK (refer to Universal Serial Bus Specification, Revision 2.0: 7.1.20, Test Mode Support). Force the device in High Speed mode, and configure a bulk-type endpoint. Do not fill this endpoint for sending NAK to the host. Upon command, a port’s transceiver must enter the High Speed Receive mode and remain in that mode until the exit action is taken. This enables the testing of output impedance, low level output voltage and loading characteristics. In addition, while in this mode, upstream facing ports (and only upstream facing ports) must respond to any IN token packet with a NAK handshake (only

if the packet CRC is determined to be correct) within the normal allowed device response time. This enables testing of the device squelch level circuitry and, additionally, provides a general purpose stimulus/response test for basic functional testing.

34.7.8 UDPHS Endpoint Configuration Register

Name: UDPHS_EPTCFGx [x=0..15]

Address: 0xFC02C100 [0], 0xFC02C120 [1], 0xFC02C140 [2], 0xFC02C160 [3], 0xFC02C180 [4], 0xFC02C1A0 [5], 0xFC02C1C0 [6], 0xFC02C1E0 [7], 0xFC02C200 [8], 0xFC02C220 [9], 0xFC02C240 [10], 0xFC02C260 [11], 0xFC02C280 [12], 0xFC02C2A0 [13], 0xFC02C2C0 [14], 0xFC02C2E0 [15]

Access: Read/Write

31	30	29	28	27	26	25	24
EPT_MAPD	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	NB_TRANS	
7	6	5	4	3	2	1	0
BK_NUMBER		EPT_TYPE		EPT_DIR	EPT_SIZE		

- **EPT_SIZE: Endpoint Size (cleared upon USB reset)**

Set this field according to the endpoint size(1) in bytes (refer to Section 34.6.6 “Endpoint Configuration”).

Value	Name	Description
0	8	8 bytes
1	16	16 bytes
2	32	32 bytes
3	64	64 bytes
4	128	128 bytes
5	256	256 bytes
6	512	512 bytes
7	1024	1024 bytes

Note: 1. 1024 bytes is only for isochronous endpoint.

- **EPT_DIR: Endpoint Direction (cleared upon USB reset)**

0: Clear this bit to configure OUT direction for Bulk, Interrupt and Isochronous endpoints.

1: Set this bit to configure IN direction for Bulk, Interrupt and Isochronous endpoints.

For Control endpoints this bit has no effect and should be left at zero.

- **EPT_TYPE: Endpoint Type (cleared upon USB reset)**

Set this field according to the endpoint type (refer to Section 34.6.6 “Endpoint Configuration”).

(Endpoint 0 should always be configured as control)

Value	Name	Description
0	CTRL8	Control endpoint
1	ISO	Isochronous endpoint
2	BULK	Bulk endpoint

Value	Name	Description
3	INT	Interrupt endpoint

- **BK_NUMBER: Number of Banks (cleared upon USB reset)**

Set this field according to the endpoint's number of banks (refer to Section 34.6.6 "Endpoint Configuration").

Value	Name	Description
0	0	Zero bank, the endpoint is not mapped in memory
1	1	One bank (bank 0)
2	2	Double bank (Ping-Pong: bank0/bank1)
3	3	Triple bank (bank0/bank1/bank2)

- **NB_TRANS: Number Of Transaction per Microframe (cleared upon USB reset)**

The Number of transactions per microframe is set by software.

Note: Meaningful for high bandwidth isochronous endpoint only.

- **EPT_MAPD: Endpoint Mapped (cleared upon USB reset)**

0: The user should reprogram the register with correct values.

1: Set by hardware when the endpoint size (EPT_SIZE) and the number of banks (BK_NUMBER) are correct regarding:

- The FIFO max capacity (FIFO_MAX_SIZE in UDPHS_IPFEATURES register)
- The number of endpoints/banks already allocated
- The number of allowed banks for this endpoint

34.7.9 UDPHS Endpoint Control Enable Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTCTLENBx [x=0..15]

Address: 0xFC02C104 [0], 0xFC02C124 [1], 0xFC02C144 [2], 0xFC02C164 [3], 0xFC02C184 [4], 0xFC02C1A4 [5], 0xFC02C1C4 [6], 0xFC02C1E4 [7], 0xFC02C204 [8], 0xFC02C224 [9], 0xFC02C244 [10], 0xFC02C264 [11], 0xFC02C284 [12], 0xFC02C2A4 [13], 0xFC02C2C4 [14], 0xFC02C2E4 [15]

Access: Write-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	TXRDY	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
–	–	–	NYET_DIS	INTDIS_DMA	–	AUTO_VALID	EPT_ENABL

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register”.

For additional information, refer to “UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints)”.

- **EPT_ENABL: Endpoint Enable**

0: No effect.

1: Enable endpoint according to the device configuration.

- **AUTO_VALID: Packet Auto-Valid Enable**

0: No effect.

1: Enable this bit to automatically validate the current packet and switch to the next bank for both IN and OUT transfers.

- **INTDIS_DMA: Interrupts Disable DMA**

0: No effect.

1: If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled.

- **NYET_DIS: NYET Disable (Only for High Speed Bulk OUT endpoints)**

0: No effect.

1: Forces an ACK response to the next High Speed Bulk OUT transfer instead of a NYET response.

- **ERR_OVFLW: Overflow Error Interrupt Enable**

0: No effect.

1: Enable Overflow Error Interrupt.

- **RXRDY_TXKL: Received OUT Data Interrupt Enable**

0: No effect.

1: Enable Received OUT Data Interrupt.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Enable**

0: No effect.

1: Enable Transmitted IN Data Complete Interrupt.

- **TXRDY: TX Packet Ready Interrupt Enable**

0: No effect.

1: Enable TX Packet Ready/Transaction Error Interrupt.

- **RX_SETUP: Received SETUP**

0: No effect.

1: Enable RX_SETUP Interrupt.

- **STALL_SNT: Stall Sent Interrupt Enable**

0: No effect.

1: Enable Stall Sent Interrupt.

- **NAK_IN: NAKIN Interrupt Enable**

0: No effect.

1: Enable NAKIN Interrupt.

- **NAK_OUT: NAKOUT Interrupt Enable**

0: No effect.

1: Enable NAKOUT Interrupt.

- **BUSY_BANK: Busy Bank Interrupt Enable**

0: No effect.

1: Enable Busy Bank Interrupt.

- **SHRT_PCKT: Short Packet Send/Short Packet Interrupt Enable**

For OUT endpoints:

0: No effect.

1: Enable Short Packet Interrupt.

For IN endpoints: Guarantees short packet at end of DMA Transfer if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTOVALID bits are also set.

34.7.10 UDPHS Endpoint Control Enable Register (Isochronous Endpoints)

Name: UDPHS_EPTCTLENBx [x=0..15] (ISOENDPT)

Address: 0xFC02C104 [0], 0xFC02C124 [1], 0xFC02C144 [2], 0xFC02C164 [3], 0xFC02C184 [4], 0xFC02C1A4 [5], 0xFC02C1C4 [6], 0xFC02C1E4 [7], 0xFC02C204 [8], 0xFC02C224 [9], 0xFC02C244 [10], 0xFC02C264 [11], 0xFC02C284 [12], 0xFC02C2A4 [13], 0xFC02C2C4 [14], 0xFC02C2E4 [15]

Access: Write-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	TXRDY_TRER	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
MDATA_RX	DATAX_RX	–	–	INTDIS_DMA	–	AUTO_VALID	EPT_ENABL

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register”.

For additional information, refer to “UDPHS Endpoint Control Register (Isochronous Endpoint)”.

- **EPT_ENABL: Endpoint Enable**

0: No effect.

1: Enable endpoint according to the device configuration.

- **AUTO_VALID: Packet Auto-Valid Enable**

0: No effect.

1: Enable this bit to automatically validate the current packet and switch to the next bank for both IN and OUT transfers.

- **INTDIS_DMA: Interrupts Disable DMA**

0: No effect.

1: If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled.

- **DATAX_RX: DATAx Interrupt Enable (Only for high bandwidth Isochronous OUT endpoints)**

0: No effect.

1: Enable DATAx Interrupt.

- **MDATA_RX: MDATA Interrupt Enable (Only for high bandwidth Isochronous OUT endpoints)**

0: No effect.

1: Enable MDATA Interrupt.

- **ERR_OVFLW: Overflow Error Interrupt Enable**

0: No effect.

1: Enable Overflow Error Interrupt.

- **RXRDY_TXKL: Received OUT Data Interrupt Enable**

0: No effect.

1: Enable Received OUT Data Interrupt.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Enable**

0: No effect.

1: Enable Transmitted IN Data Complete Interrupt.

- **TXRDY_TRER: TX Packet Ready/Transaction Error Interrupt Enable**

0: No effect.

1: Enable TX Packet Ready/Transaction Error Interrupt.

- **ERR_FL_ISO: Error Flow Interrupt Enable**

0: No effect.

1: Enable Error Flow ISO Interrupt.

- **ERR_CRC_NTR: ISO CRC Error/Number of Transaction Error Interrupt Enable**

0: No effect.

1: Enable Error CRC ISO/Error Number of Transaction Interrupt.

- **ERR_FLUSH: Bank Flush Error Interrupt Enable**

0: No effect.

1: Enable Bank Flush Error Interrupt.

- **BUSY_BANK: Busy Bank Interrupt Enable**

0: No effect.

1: Enable Busy Bank Interrupt.

- **SHRT_PCKT: Short Packet Send/Short Packet Interrupt Enable**

For OUT endpoints:

0: No effect.

1: Enable Short Packet Interrupt.

For IN endpoints: Guarantees short packet at end of DMA Transfer if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTOVALID bits are also set.

34.7.11 UDPHS Endpoint Control Disable Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTCTLDISx [x=0..15]

Address: 0xFC02C108 [0], 0xFC02C128 [1], 0xFC02C148 [2], 0xFC02C168 [3], 0xFC02C188 [4], 0xFC02C1A8 [5], 0xFC02C1C8 [6], 0xFC02C1E8 [7], 0xFC02C208 [8], 0xFC02C228 [9], 0xFC02C248 [10], 0xFC02C268 [11], 0xFC02C288 [12], 0xFC02C2A8 [13], 0xFC02C2C8 [14], 0xFC02C2E8 [15]

Access: Write-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	TXRDY	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
–	–	–	NYET_DIS	INTDIS_DMA	–	AUTO_VALID	EPT_DISABL

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register”.

For additional information, refer to “UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints)”.

- **EPT_DISABL: Endpoint Disable**

0: No effect.

1: Disable endpoint.

- **AUTO_VALID: Packet Auto-Valid Disable**

0: No effect.

1: Disable this bit to not automatically validate the current packet.

- **INTDIS_DMA: Interrupts Disable DMA**

0: No effect.

1: Disable the “Interrupts Disable DMA”.

- **NYET_DIS: NYET Enable (Only for High Speed Bulk OUT endpoints)**

0: No effect.

1: Let the hardware handle the handshake response for the High Speed Bulk OUT transfer.

- **ERR_OVFLW: Overflow Error Interrupt Disable**

0: No effect.

1: Disable Overflow Error Interrupt.

- **RXRDY_TXKL: Received OUT Data Interrupt Disable**

0: No effect.

1: Disable Received OUT Data Interrupt.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Disable**

0: No effect.

1: Disable Transmitted IN Data Complete Interrupt.

- **TXRDY: TX Packet Ready Interrupt Disable**

0: No effect.

1: Disable TX Packet Ready/Transaction Error Interrupt.

- **RX_SETUP: Received SETUP Interrupt Disable**

0: No effect.

1: Disable RX_SETUP Interrupt.

- **STALL_SNT: Stall Sent Interrupt Disable**

0: No effect.

1: Disable Stall Sent Interrupt.

- **NAK_IN: NAKIN Interrupt Disable**

0: No effect.

1: Disable NAKIN Interrupt.

- **NAK_OUT: NAKOUT Interrupt Disable**

0: No effect.

1: Disable NAKOUT Interrupt.

- **BUSY_BANK: Busy Bank Interrupt Disable**

0: No effect.

1: Disable Busy Bank Interrupt.

- **SHRT_PCKT: Short Packet Interrupt Disable**

For OUT endpoints:

0: No effect.

1: Disable Short Packet Interrupt.

For IN endpoints: Never automatically add a zero length packet at end of DMA transfer.

34.7.12 UDPHS Endpoint Control Disable Register (Isochronous Endpoint)

Name: UDPHS_EPTCTLDISx [x=0..15] (ISOENDPT)

Address: 0xFC02C108 [0], 0xFC02C128 [1], 0xFC02C148 [2], 0xFC02C168 [3], 0xFC02C188 [4], 0xFC02C1A8 [5], 0xFC02C1C8 [6], 0xFC02C1E8 [7], 0xFC02C208 [8], 0xFC02C228 [9], 0xFC02C248 [10], 0xFC02C268 [11], 0xFC02C288 [12], 0xFC02C2A8 [13], 0xFC02C2C8 [14], 0xFC02C2E8 [15]

Access: Write-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	TXRDY_TRER	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
MDATA_RX	DATAX_RX	–	–	INTDIS_DMA	–	AUTO_VALID	EPT_DISABL

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register”.

For additional information, refer to “UDPHS Endpoint Control Register (Isochronous Endpoint)”.

- **EPT_DISABL: Endpoint Disable**

0: No effect.

1: Disable endpoint.

- **AUTO_VALID: Packet Auto-Valid Disable**

0: No effect.

1: Disable this bit to not automatically validate the current packet.

- **INTDIS_DMA: Interrupts Disable DMA**

0: No effect.

1: Disable the “Interrupts Disable DMA”.

- **DATAX_RX: DATAx Interrupt Disable (Only for High Bandwidth Isochronous OUT endpoints)**

0: No effect.

1: Disable DATAx Interrupt.

- **MDATA_RX: MDATA Interrupt Disable (Only for High Bandwidth Isochronous OUT endpoints)**

0: No effect.

1: Disable MDATA Interrupt.

- **ERR_OVFLW: Overflow Error Interrupt Disable**

0: No effect.

1: Disable Overflow Error Interrupt.

- **RXRDY_TXKL: Received OUT Data Interrupt Disable**

0: No effect.

1: Disable Received OUT Data Interrupt.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Disable**

0: No effect.

1: Disable Transmitted IN Data Complete Interrupt.

- **TXRDY_TRER: TX Packet Ready/Transaction Error Interrupt Disable**

0: No effect.

1: Disable TX Packet Ready/Transaction Error Interrupt.

- **ERR_FL_ISO: Error Flow Interrupt Disable**

0: No effect.

1: Disable Error Flow ISO Interrupt.

- **ERR_CRC_NTR: ISO CRC Error/Number of Transaction Error Interrupt Disable**

0: No effect.

1: Disable Error CRC ISO/Error Number of Transaction Interrupt.

- **ERR_FLUSH: bank flush error Interrupt Disable**

0: No effect.

1: Disable Bank Flush Error Interrupt.

- **BUSY_BANK: Busy Bank Interrupt Disable**

0: No effect.

1: Disable Busy Bank Interrupt.

- **SHRT_PCKT: Short Packet Interrupt Disable**

For OUT endpoints:

0: No effect.

1: Disable Short Packet Interrupt.

For IN endpoints: Never automatically add a zero length packet at end of DMA transfer.

34.7.13 UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTCTLx [x=0..15]

Address: 0xFC02C10C [0], 0xFC02C12C [1], 0xFC02C14C [2], 0xFC02C16C [3], 0xFC02C18C [4], 0xFC02C1AC [5], 0xFC02C1CC [6], 0xFC02C1EC [7], 0xFC02C20C [8], 0xFC02C22C [9], 0xFC02C24C [10], 0xFC02C26C [11], 0xFC02C28C [12], 0xFC02C2AC [13], 0xFC02C2CC [14], 0xFC02C2EC [15]

Access: Read-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	TXRDY	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
–	–	–	NYET_DIS	INTDIS_DMA	–	AUTO_VALID	EPT_ENABL

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register”.

- **EPT_ENABL: Endpoint Enable (cleared upon USB reset)**

0: The endpoint is disabled according to the device configuration. Endpoint 0 should always be enabled after a hardware or UDPHS bus reset and participate in the device configuration.

1: The endpoint is enabled according to the device configuration.

- **AUTO_VALID: Packet Auto-Valid Enabled (Not for CONTROL Endpoints) (cleared upon USB reset)**

Set this bit to automatically validate the current packet and switch to the next bank for both IN and OUT endpoints.

For IN Transfer:

If this bit is set, the UDPHS_EPTSTAx register TXRDY bit is set automatically when the current bank is full and at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set.

The user may still set the UDPHS_EPTSTAx register TXRDY bit if the current bank is not full, unless the user needs to send a Zero Length Packet by software.

For OUT Transfer:

If this bit is set, the UDPHS_EPTSTAx register RXRDY_TXKL bit is automatically reset for the current bank when the last packet byte has been read from the bank FIFO or at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set. For example, to truncate a padded data packet when the actual data transfer size is reached.

The user may still clear the UDPHS_EPTSTAx register RXRDY_TXKL bit, for example, after completing a DMA buffer by software if UDPHS_DMACONTROLx register END_B_EN bit was disabled or in order to cancel the read of the remaining data bank(s).

- **INTDIS_DMA: Interrupt Disables DMA (cleared upon USB reset)**

If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled regardless of the UDPHS_IEN register EPT_x bit for this endpoint. Then, the firmware will have to clear or disable the interrupt source or clear this bit if transfer completion is needed.

If the exception raised is associated with the new system bank packet, then the previous DMA packet transfer is normally completed, but the new DMA packet transfer is not started (not requested).

If the exception raised is not associated to a new system bank packet (NAK_IN, NAK_OUT, etc.), then the request cancellation may happen at any time and may immediately stop the current DMA transfer.

This may be used, for example, to identify or prevent an erroneous packet to be transferred into a buffer or to complete a DMA buffer by software after reception of a short packet.

- **NYET_DIS: NYET Disable (Only for High Speed Bulk OUT Endpoints) (cleared upon USB reset)**

0: Lets the hardware handle the handshake response for the High Speed Bulk OUT transfer.

1: Forces an ACK response to the next High Speed Bulk OUT transfer instead of a NYET response.

Note: According to the *Universal Serial Bus Specification, Rev 2.0* (8.5.1.1 NAK Responses to OUT/DATA During PING Protocol), a NAK response to an HS Bulk OUT transfer is expected to be an unusual occurrence.

- **ERR_OVFLW: Overflow Error Interrupt Enabled (cleared upon USB reset)**

0: Overflow Error Interrupt is masked.

1: Overflow Error Interrupt is enabled.

- **RXRDY_TXKL: Received OUT Data Interrupt Enabled (cleared upon USB reset)**

0: Received OUT Data Interrupt is masked.

1: Received OUT Data Interrupt is enabled.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Enabled (cleared upon USB reset)**

0: Transmitted IN Data Complete Interrupt is masked.

1: Transmitted IN Data Complete Interrupt is enabled.

- **TXRDY: TX Packet Ready Interrupt Enabled (cleared upon USB reset)**

0: TX Packet Ready Interrupt is masked.

1: TX Packet Ready Interrupt is enabled.

Caution: Interrupt source is active as long as the corresponding UDPHS_EPTSTAx register TXRDY flag remains low. If there are no more banks available for transmitting after the software has set UDPHS_EPTSTAx/TXRDY for the last transmit packet, then the interrupt source remains inactive until the first bank becomes free again to transmit at UDPHS_EPTSTAx/TXRDY hardware clear.

- **RX_SETUP: Received SETUP Interrupt Enabled (cleared upon USB reset)**

0: Received SETUP is masked.

1: Received SETUP is enabled.

- **STALL_SNT: Stall Sent Interrupt Enabled (cleared upon USB reset)**

0: Stall Sent Interrupt is masked.

1: Stall Sent Interrupt is enabled.

- **NAK_IN: NAKIN Interrupt Enabled (cleared upon USB reset)**

0: NAKIN Interrupt is masked.

1: NAKIN Interrupt is enabled.

- **NAK_OUT: NAKOUT Interrupt Enabled (cleared upon USB reset)**

0: NAKOUT Interrupt is masked.

1: NAKOUT Interrupt is enabled.

- **BUSY_BANK: Busy Bank Interrupt Enabled (cleared upon USB reset)**

0: BUSY_BANK Interrupt is masked.

1: BUSY_BANK Interrupt is enabled.

For OUT endpoints: an interrupt is sent when all banks are busy.

For IN endpoints: an interrupt is sent when all banks are free.

- **SHRT_PCKT: Short Packet Interrupt Enabled (cleared upon USB reset)**

For OUT endpoints: send an Interrupt when a Short Packet has been received.

0: Short Packet Interrupt is masked.

1: Short Packet Interrupt is enabled.

For IN endpoints: a Short Packet transmission is guaranteed upon end of the DMA Transfer, thus signaling a BULK or INTERRUPT end of transfer, but only if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTO_VALID bits are also set.

34.7.14 UDPHS Endpoint Control Register (Isochronous Endpoint)

Name: UDPHS_EPTCTLx [x=0..15] (ISOENDPT)

Address: 0xFC02C10C [0], 0xFC02C12C [1], 0xFC02C14C [2], 0xFC02C16C [3], 0xFC02C18C [4], 0xFC02C1AC [5], 0xFC02C1CC [6], 0xFC02C1EC [7], 0xFC02C20C [8], 0xFC02C22C [9], 0xFC02C24C [10], 0xFC02C26C [11], 0xFC02C28C [12], 0xFC02C2AC [13], 0xFC02C2CC [14], 0xFC02C2EC [15]

Access: Read-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	TXRDY_TRER	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
MDATA_RX	DATAX_RX	–	–	INTDIS_DMA	–	AUTO_VALID	EPT_ENABL

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register”.

- **EPT_ENABL: Endpoint Enable (cleared upon USB reset)**

0: The endpoint is disabled according to the device configuration. Endpoint 0 should always be enabled after a hardware or UDPHS bus reset and participate in the device configuration.

1: The endpoint is enabled according to the device configuration.

- **AUTO_VALID: Packet Auto-Valid Enabled (cleared upon USB reset)**

Set this bit to automatically validate the current packet and switch to the next bank for both IN and OUT endpoints.

For IN Transfer:

If this bit is set, the UDPHS_EPTSTAx register TXRDY_TRER bit is set automatically when the current bank is full and at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set.

The user may still set the UDPHS_EPTSTAx register TXRDY_TRER bit if the current bank is not full, unless the user needs to send a Zero Length Packet by software.

For OUT Transfer:

If this bit is set, the UDPHS_EPTSTAx register RXRDY_TXKL bit is automatically reset for the current bank when the last packet byte has been read from the bank FIFO or at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set. For example, to truncate a padded data packet when the actual data transfer size is reached.

The user may still clear the UDPHS_EPTSTAx register RXRDY_TXKL bit, for example, after completing a DMA buffer by software if UDPHS_DMACONTROLx register END_B_EN bit was disabled or in order to cancel the read of the remaining data bank(s).

- **INTDIS_DMA: Interrupt Disables DMA (cleared upon USB reset)**

If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled regardless of the UDPHS_IEN register EPT_x bit for this endpoint. Then, the firmware will have to clear or disable the interrupt source or clear this bit if transfer completion is needed.

If the exception raised is associated with the new system bank packet, then the previous DMA packet transfer is normally completed, but the new DMA packet transfer is not started (not requested).

If the exception raised is not associated to a new system bank packet (ex: ERR_FL_ISO), then the request cancellation may happen at any time and may immediately stop the current DMA transfer.

This may be used, for example, to identify or prevent an erroneous packet to be transferred into a buffer or to complete a DMA buffer by software after reception of a short packet, or to perform buffer truncation on ERR_FL_ISO interrupt for adaptive rate.

- **DATAX_RX: DATAx Interrupt Enabled (Only for High Bandwidth Isochronous OUT endpoints) (cleared upon USB reset)**

0: No effect.

1: Send an interrupt when a DATA2, DATA1 or DATA0 packet has been received meaning the whole microframe data payload has been received.

- **MDATA_RX: MDATA Interrupt Enabled (Only for High Bandwidth Isochronous OUT endpoints) (cleared upon USB reset)**

0: No effect.

1: Send an interrupt when an MDATA packet has been received and so at least one packet of the microframe data payload has been received.

- **ERR_OVFLW: Overflow Error Interrupt Enabled (cleared upon USB reset)**

0: Overflow Error Interrupt is masked.

1: Overflow Error Interrupt is enabled.

- **RXRDY_TXKL: Received OUT Data Interrupt Enabled (cleared upon USB reset)**

0: Received OUT Data Interrupt is masked.

1: Received OUT Data Interrupt is enabled.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Enabled (cleared upon USB reset)**

0: Transmitted IN Data Complete Interrupt is masked.

1: Transmitted IN Data Complete Interrupt is enabled.

- **TXRDY_TRER: TX Packet Ready/Transaction Error Interrupt Enabled (cleared upon USB reset)**

0: TX Packet Ready/Transaction Error Interrupt is masked.

1: TX Packet Ready/Transaction Error Interrupt is enabled.

Caution: Interrupt source is active as long as the corresponding UDPHS_EPTSTAx register TXRDY_TRER flag remains low. If there are no more banks available for transmitting after the software has set UDPHS_EPTSTAx/TXRDY_TRER for the last transmit packet, then the interrupt source remains inactive until the first bank becomes free again to transmit at UDPHS_EPTSTAx/TXRDY_TRER hardware clear.

- **ERR_FL_ISO: Error Flow Interrupt Enabled (cleared upon USB reset)**

0: Error Flow Interrupt is masked.

1: Error Flow Interrupt is enabled.

- **ERR_CRC_NTR: ISO CRC Error/Number of Transaction Error Interrupt Enabled (cleared upon USB reset)**

0: ISO CRC error/number of Transaction Error Interrupt is masked.

1: ISO CRC error/number of Transaction Error Interrupt is enabled.

- **ERR_FLUSH: Bank Flush Error Interrupt Enabled (cleared upon USB reset)**

0: Bank Flush Error Interrupt is masked.

1: Bank Flush Error Interrupt is enabled.

- **BUSY_BANK: Busy Bank Interrupt Enabled (cleared upon USB reset)**

0: BUSY_BANK Interrupt is masked.

1: BUSY_BANK Interrupt is enabled.

For OUT endpoints: An interrupt is sent when all banks are busy.

For IN endpoints: An interrupt is sent when all banks are free.

- **SHRT_PCKT: Short Packet Interrupt Enabled (cleared upon USB reset)**

For OUT endpoints: send an Interrupt when a Short Packet has been received.

0: Short Packet Interrupt is masked.

1: Short Packet Interrupt is enabled.

For IN endpoints: A Short Packet transmission is guaranteed upon end of the DMA Transfer, thus signaling an end of isochronous (micro-)frame data, but only if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTO_VALID bits are also set.

34.7.15 UDPHS Endpoint Set Status Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTSETSTAx [x=0..15]

Address: 0xFC02C114 [0], 0xFC02C134 [1], 0xFC02C154 [2], 0xFC02C174 [3], 0xFC02C194 [4], 0xFC02C1B4 [5], 0xFC02C1D4 [6], 0xFC02C1F4 [7], 0xFC02C214 [8], 0xFC02C234 [9], 0xFC02C254 [10], 0xFC02C274 [11], 0xFC02C294 [12], 0xFC02C2B4 [13], 0xFC02C2D4 [14], 0xFC02C2F4 [15]

Access: Write-only

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

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register”.

For additional information, refer to “UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints)”.

- **FRCESTALL: Stall Handshake Request Set**

0: No effect.

1: Set this bit to request a STALL answer to the host for the next handshake

Refer to chapters 8.4.5 (Handshake Packets) and 9.4.5 (Get Status) of the *Universal Serial Bus Specification, Rev 2.0* for more information on the STALL handshake.

- **RXRDY_TXKL: KILL Bank Set (for IN Endpoint)**

0: No effect.

1: Kill the last written bank.

- **TXRDY: TX Packet Ready Set**

0: No effect.

1: Set this bit after a packet has been written into the endpoint FIFO for IN data transfers

  – This flag is used to generate a Data IN transaction (device to host).
  – Device firmware checks that it can write a data payload in the FIFO, checking that TXRDY is cleared.
  – Transfer to the FIFO is done by writing in the “Buffer Address” register.
  – Once the data payload has been transferred to the FIFO, the firmware notifies the UDPHS device setting TXRDY to one.
  – UDPHS bus transactions can start.
  – TXCOMP is set once the data payload has been received by the host.
  – Data should be written into the endpoint FIFO only after this bit has been cleared.
  – Set this bit without writing data to the endpoint FIFO to send a Zero Length Packet.

34.7.16 UDPHS Endpoint Set Status Register (Isochronous Endpoint)

Name: UDPHS_EPTSETSTAx [x=0..15] (ISOENDPT)

Address: 0xFC02C114 [0], 0xFC02C134 [1], 0xFC02C154 [2], 0xFC02C174 [3], 0xFC02C194 [4], 0xFC02C1B4 [5], 0xFC02C1D4 [6], 0xFC02C1F4 [7], 0xFC02C214 [8], 0xFC02C234 [9], 0xFC02C254 [10], 0xFC02C274 [11], 0xFC02C294 [12], 0xFC02C2B4 [13], 0xFC02C2D4 [14], 0xFC02C2F4 [15]

Access: Write-only

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

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register”.

For additional information, refer to “UDPHS Endpoint Status Register (Isochronous Endpoint)”.

- **RXRDY_TXKL: KILL Bank Set (for IN Endpoint)**

0: No effect.

1: Kill the last written bank.

- **TXRDY_TRER: TX Packet Ready Set**

0: No effect.

1: Set this bit after a packet has been written into the endpoint FIFO for IN data transfers

  – This flag is used to generate a Data IN transaction (device to host).
  – Device firmware checks that it can write a data payload in the FIFO, checking that TXRDY_TRER is cleared.
  – Transfer to the FIFO is done by writing in the “Buffer Address” register.
  – Once the data payload has been transferred to the FIFO, the firmware notifies the UDPHS device setting TXRDY_TRER to one.
  – UDPHS bus transactions can start.
  – TXCOMP is set once the data payload has been sent.
  – Data should be written into the endpoint FIFO only after this bit has been cleared.
  – Set this bit without writing data to the endpoint FIFO to send a Zero Length Packet.

34.7.17 UDPHS Endpoint Clear Status Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTCLRSTAx [x=0..15]

Address: 0xFC02C118 [0], 0xFC02C138 [1], 0xFC02C158 [2], 0xFC02C178 [3], 0xFC02C198 [4], 0xFC02C1B8 [5], 0xFC02C1D8 [6], 0xFC02C1F8 [7], 0xFC02C218 [8], 0xFC02C238 [9], 0xFC02C258 [10], 0xFC02C278 [11], 0xFC02C298 [12], 0xFC02C2B8 [13], 0xFC02C2D8 [14], 0xFC02C2F8 [15]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	–	TX_COMPLT	RXRDY_TXKL	–

7	6	5	4	3	2	1	0
–	TOGGLESQ	FRCESTALL	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register”.

For additional information, refer to “UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints)”.

- **FRCESTALL: Stall Handshake Request Clear**

0: No effect.

1: Clear the STALL request. The next packets from host will not be STALLed.

- **TOGGLESQ: Data Toggle Clear**

0: No effect.

1: Clear the PID data of the current bank

For OUT endpoints, the next received packet should be a DATA0.

For IN endpoints, the next packet will be sent with a DATA0 PID.

- **RXRDY_TXKL: Received OUT Data Clear**

0: No effect.

1: Clear the RXRDY_TXKL flag of UDPHS_EPTSTAx.

- **TX_COMPLT: Transmitted IN Data Complete Clear**

0: No effect.

1: Clear the TX_COMPLT flag of UDPHS_EPTSTAx.

- **RX_SETUP: Received SETUP Clear**

0: No effect.

1: Clear the RX_SETUP flags of UDPHS_EPTSTAx.

- **STALL_SNT: Stall Sent Clear**

0: No effect.

1: Clear the STALL_SNT flags of UDPHS_EPTSTAx.

- **NAK_IN: NAKIN Clear**

0: No effect.

1: Clear the NAK_IN flags of UDPHS_EPTSTAx.

- **NAK_OUT: NAKOUT Clear**

0: No effect.

1: Clear the NAK_OUT flag of UDPHS_EPTSTAx.

34.7.18 UDPHS Endpoint Clear Status Register (Isochronous Endpoint)

Name: UDPHS_EPTCLRSTAx [x=0..15] (ISOENDPT)

Address: 0xFC02C118 [0], 0xFC02C138 [1], 0xFC02C158 [2], 0xFC02C178 [3], 0xFC02C198 [4], 0xFC02C1B8 [5], 0xFC02C1D8 [6], 0xFC02C1F8 [7], 0xFC02C218 [8], 0xFC02C238 [9], 0xFC02C258 [10], 0xFC02C278 [11], 0xFC02C298 [12], 0xFC02C2B8 [13], 0xFC02C2D8 [14], 0xFC02C2F8 [15]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	–	TX_COMPLT	RXRDY_TXKL	–
7	6	5	4	3	2	1	0
–	TOGGLESQ	–	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register”.

For additional information, refer to “UDPHS Endpoint Status Register (Isochronous Endpoint)”.

- **TOGGLESQ: Data Toggle Clear**

0: No effect.

1: Clear the PID data of the current bank

For OUT endpoints, the next received packet should be a DATA0.

For IN endpoints, the next packet will be sent with a DATA0 PID.

- **RXRDY_TXKL: Received OUT Data Clear**

0: No effect.

1: Clear the RXRDY_TXKL flag of UDPHS_EPTSTAx.

- **TX_COMPLT: Transmitted IN Data Complete Clear**

0: No effect.

1: Clear the TX_COMPLT flag of UDPHS_EPTSTAx.

- **ERR_FL_ISO: Error Flow Clear**

0: No effect.

1: Clear the ERR_FL_ISO flags of UDPHS_EPTSTAx.

- **ERR_CRC_NTR: Number of Transaction Error Clear**

0: No effect.

1: Clear the ERR_CRC_NTR flags of UDPHS_EPTSTAx.

- **ERR_FLUSH: Bank Flush Error Clear**

0: No effect.

1: Clear the ERR_FLUSH flags of UDPHS_EPTSTAx.

34.7.19 UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTSTAx [x=0..15]

Address: 0xFC02C11C [0], 0xFC02C13C [1], 0xFC02C15C [2], 0xFC02C17C [3], 0xFC02C19C [4], 0xFC02C1BC [5], 0xFC02C1DC [6], 0xFC02C1FC [7], 0xFC02C21C [8], 0xFC02C23C [9], 0xFC02C25C [10], 0xFC02C27C [11], 0xFC02C29C [12], 0xFC02C2BC [13], 0xFC02C2DC [14], 0xFC02C2FC [15]

Access: Read-only

31	30	29	28	27	26	25	24
SHRT_PCKT	BYTE_COUNT						
23	22	21	20	19	18	17	16
BYTE_COUNT				BUSY_BANK_STA		CURBK_CTLDIR	
15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	TXRDY	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
TOGGLESQ_STA		FRCESTALL	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register”.

- **FRCESTALL: Stall Handshake Request (cleared upon USB reset)**

0: No effect.

1: If set a STALL answer will be done to the host for the next handshake.

This bit is reset by hardware upon received SETUP.

- **TOGGLESQ_STA: Toggle Sequencing (cleared upon USB reset)**

Toggle Sequencing:

– **IN endpoint:** It indicates the PID Data Toggle that will be used for the next packet sent. This is not relative to the current bank.

– CONTROL and OUT endpoint:

These bits are set by hardware to indicate the PID data of the current bank:

Value	Name	Description
0	DATA0	DATA0
1	DATA1	DATA1
2	DATA2	Reserved for High Bandwidth Isochronous Endpoint
3	MDATA	Reserved for High Bandwidth Isochronous Endpoint

Notes:
1. In OUT transfer, the Toggle information is meaningful only when the current bank is busy (Received OUT Data = 1).
2. These bits are updated for OUT transfer:
   - A new data has been written into the current bank.
   - The user has just cleared the Received OUT Data bit to switch to the next bank.
3. This field is reset to DATA1 by the UDPHS_EPTCLRSTAx register TOGGLESQ bit, and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_OVFLW: Overflow Error (cleared upon USB reset)**

This bit is set by hardware when a new too-long packet is received.

Example: If the user programs an endpoint 64 bytes wide and the host sends 128 bytes in an OUT transfer, then the Overflow Error bit is set.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **RXRDY_TXKL: Received OUT Data/KILL Bank (cleared upon USB reset)**
  - **Received OUT Data** (for OUT endpoint or Control endpoint):

This bit is set by hardware after a new packet has been stored in the endpoint FIFO.

This bit is cleared by the device firmware after reading the OUT data from the endpoint.

For multi-bank endpoints, this bit may remain active even when cleared by the device firmware, this if an other packet has been received meanwhile.

Hardware assertion of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register RXRDY_TXKL bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

  - **KILL Bank** (for IN endpoint):
  - The bank is really cleared or the bank is sent, BUSY_BANK_STA is decremented.
  - The bank is not cleared but sent on the IN transfer, TX_COMPLT
  - The bank is not cleared because it was empty. The user should wait that this bit is cleared before trying to clear another packet.

Note: “Kill a packet” may be refused if at the same time, an IN token is coming and the current packet is sent on the UDPHS line. In this case, the TX_COMPLT bit is set. Take notice however, that if at least two banks are ready to be sent, there is no problem to kill a packet even if an IN token is coming. In fact, in that case, the current bank is sent (IN transfer) and the last bank is killed.

- **TX_COMPLT: Transmitted IN Data Complete (cleared upon USB reset)**

This bit is set by hardware after an IN packet has been accepted (ACK’ed) by the host.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **TXRDY: TX Packet Ready (cleared upon USB reset)**

This bit is cleared by hardware after the host has acknowledged the packet.

For Multi-bank endpoints, this bit may remain clear even after software is set if another bank is available to transmit.

Hardware clear of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register TXRDY bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **RX_SETUP: Received SETUP (cleared upon USB reset)**
  - (for Control endpoint only)

This bit is set by hardware when a valid SETUP packet has been received from the host.

It is cleared by the device firmware after reading the SETUP data from the endpoint FIFO.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **STALL_SNT: Stall Sent (cleared upon USB reset)**
  – (for Control, Bulk and Interrupt endpoints)

This bit is set by hardware after a STALL handshake has been sent as requested by the UDPHS_EPTSTAx register FRCESTALL bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **NAK_IN: NAK IN (cleared upon USB reset)**

This bit is set by hardware when a NAK handshake has been sent in response to an IN request from the Host.

This bit is cleared by software.

- **NAK_OUT: NAK OUT (cleared upon USB reset)**

This bit is set by hardware when a NAK handshake has been sent in response to an OUT or PING request from the Host.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by EPT_CTL_DISx (disable endpoint).

- **CURBK_CTLDIR: Current Bank/Control Direction (cleared upon USB reset)**
  – **Current Bank** (not relevant for Control endpoint):

These bits are set by hardware to indicate the number of the current bank.

Value	Name	Description
0	BANK0	Bank 0 (or single bank)
1	BANK1	Bank 1
2	BANK2	Bank 2

Note: The current bank is updated each time the user:
- Sets the TX Packet Ready bit to prepare the next IN transfer and to switch to the next bank.
- Clears the received OUT data bit to access the next bank.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

  – **Control Direction** (for Control endpoint only):

0: A Control Write is requested by the Host.

1: A Control Read is requested by the Host.

Notes: 1. This bit corresponds with the 7th bit of the bmRequestType (Byte 0 of the Setup Data).
2. This bit is updated after receiving new setup data.

- **BUSY_BANK_STA: Busy Bank Number (cleared upon USB reset)**

These bits are set by hardware to indicate the number of busy banks.

IN endpoint: It indicates the number of busy banks filled by the user, ready for IN transfer.

OUT endpoint: It indicates the number of busy banks filled by OUT transaction from the Host.

Value	Name	Description
0	0BUSYBANK	All banks are free
1	1BUSYBANK	1 busy bank
2	2BUSYBANKS	2 busy banks
3	3BUSYBANKS	3 busy banks

- **BYTE_COUNT: UDPHS Byte Count (cleared upon USB reset)**

Byte count of a received data packet.

This field is incremented after each write into the endpoint (to prepare an IN transfer).

This field is decremented after each reading into the endpoint (OUT transfer).

This field is also updated at RXRDY_TXKL flag clear with the next bank.

This field is also updated at TXRDY flag set with the next bank.

This field is reset by EPT_x of UDPHS_EPTRST register.

- **SHRT_PCKT: Short Packet (cleared upon USB reset)**

An OUT Short Packet is detected when the receive byte count is less than the configured UDPHS_EPTCFGx register EPT_Size.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

34.7.20 UDPHS Endpoint Status Register (Isochronous Endpoint)

Name: UDPHS_EPTSTAx [x=0..15] (ISOENDPT)

Address: 0xFC02C11C [0], 0xFC02C13C [1], 0xFC02C15C [2], 0xFC02C17C [3], 0xFC02C19C [4], 0xFC02C1BC [5], 0xFC02C1DC [6], 0xFC02C1FC [7], 0xFC02C21C [8], 0xFC02C23C [9], 0xFC02C25C [10], 0xFC02C27C [11], 0xFC02C29C [12], 0xFC02C2BC [13], 0xFC02C2DC [14], 0xFC02C2FC [15]

Access: Read-only

31	30	29	28	27	26	25	24
SHRT_PCKT	BYTE_COUNT						
23	22	21	20	19	18	17	16
BYTE_COUNT				BUSY_BANK_STA		CURBK	
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	TXRDY_TRER	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
TOGGLESQ_STA		–	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register”.

- **TOGGLESQ_STA: Toggle Sequencing (cleared upon USB reset)**

Toggle Sequencing:

– **IN endpoint:** It indicates the PID Data Toggle that will be used for the next packet sent. This is not relative to the current bank.

– OUT endpoint:

These bits are set by hardware to indicate the PID data of the current bank:

Value	Name	Description
0	DATA0	DATA0
1	DATA1	DATA1
2	DATA2	Data2 (only for High Bandwidth Isochronous Endpoint)
3	MDATA	MData (only for High Bandwidth Isochronous Endpoint)

Notes:
1. In OUT transfer, the Toggle information is meaningful only when the current bank is busy (Received OUT Data = 1).
2. These bits are updated for OUT transfer:
   - A new data has been written into the current bank.
   - The user has just cleared the Received OUT Data bit to switch to the next bank.
3. For High Bandwidth Isochronous Out endpoint, it is recommended to check the UDPHS_EPTSTAx/TXRDY_TRER bit to know if the toggle sequencing is correct or not.
4. This field is reset to DATA1 by the UDPHS_EPTCLRSTAx register TOGGLESQ bit, and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_OVFLW: Overflow Error (cleared upon USB reset)**

This bit is set by hardware when a new too-long packet is received.

Example: If the user programs an endpoint 64 bytes wide and the host sends 128 bytes in an OUT transfer, then the Overflow Error bit is set.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **RXRDY_TXKL: Received OUT Data/KILL Bank (cleared upon USB reset)**
  - **Received OUT Data** (for OUT endpoint or Control endpoint):

This bit is set by hardware after a new packet has been stored in the endpoint FIFO.

This bit is cleared by the device firmware after reading the OUT data from the endpoint.

For multi-bank endpoints, this bit may remain active even when cleared by the device firmware, this if an other packet has been received meanwhile.

Hardware assertion of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register RXRDY_TXKL bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

  - **KILL Bank** (for IN endpoint):
  - The bank is really cleared or the bank is sent, BUSY_BANK_STA is decremented.
  - The bank is not cleared but sent on the IN transfer, TX_COMPLT
  - The bank is not cleared because it was empty. The user should wait that this bit is cleared before trying to clear another packet.

Note: “Kill a packet” may be refused if at the same time, an IN token is coming and the current packet is sent on the UDPHS line. In this case, the TX_COMPLT bit is set. Take notice however, that if at least two banks are ready to be sent, there is no problem to kill a packet even if an IN token is coming. In fact, in that case, the current bank is sent (IN transfer) and the last bank is killed.

- **TX_COMPLT: Transmitted IN Data Complete (cleared upon USB reset)**

This bit is set by hardware after an IN packet has been sent.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **TXRDY_TRER: TX Packet Ready/Transaction Error (cleared upon USB reset)**
  - **TX Packet Ready**:

This bit is cleared by hardware, as soon as the packet has been sent.

For Multi-bank endpoints, this bit may remain clear even after software is set if another bank is available to transmit.

Hardware clear of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register TXRDY_TRER bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

  - **Transaction Error** (for high bandwidth isochronous OUT endpoints) (Read-Only):

This bit is set by hardware when a transaction error occurs inside one microframe.

If one toggle sequencing problem occurs among the n-transactions (n = 1, 2 or 3) inside a microframe, then this bit is still set as long as the current bank contains one “bad” n-transaction (refer to “CURBK: Current Bank (cleared upon USB reset)”). As soon as the current bank is relative to a new “good” n-transactions, then this bit is reset.

Notes:
1. A transaction error occurs when the toggle sequencing does not comply with the *Universal Serial Bus Specification, Rev 2.0* (5.9.2 High Bandwidth Isochronous endpoints) (Bad PID, missing data, etc.)
2. When a transaction error occurs, the user may empty all the “bad” transactions by clearing the Received OUT Data flag (RXRDY_TXKL).

If this bit is reset, then the user should consider that a new n-transaction is coming.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_FL_ISO: Error Flow (cleared upon USB reset)**

This bit is set by hardware when a transaction error occurs.

  - Isochronous IN transaction is missed, the micro has no time to fill the endpoint (underflow).
  - Isochronous OUT data is dropped because the bank is busy (overflow).

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_CRC_NTR: CRC ISO Error/Number of Transaction Error (cleared upon USB reset)**
  – **CRC ISO Error** (for Isochronous OUT endpoints) (Read-only):

This bit is set by hardware if the last received data is corrupted (CRC error on data).

This bit is updated by hardware when new data is received (Received OUT Data bit).

  – **Number of Transaction Error** (for High Bandwidth Isochronous IN endpoints):

This bit is set at the end of a microframe in which at least one data bank has been transmitted, if less than the number of transactions per micro-frame banks (UDPHS_EPTCFGx register NB_TRANS) have been validated for transmission inside this microframe.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_FLUSH: Bank Flush Error (cleared upon USB reset)**
  – (for High Bandwidth Isochronous IN endpoints)

This bit is set when flushing unsent banks at the end of a microframe.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by EPT_CTL_DISx (disable endpoint).

- **CURBK: Current Bank (cleared upon USB reset)**
  – **Current Bank**:

These bits are set by hardware to indicate the number of the current bank.

Value	Name	Description
0	BANK0	Bank 0 (or single bank)
1	BANK1	Bank 1
2	BANK2	Bank 2

Note: The current bank is updated each time the user:
- Sets the TX Packet Ready bit to prepare the next IN transfer and to switch to the next bank.
- Clears the received OUT data bit to access the next bank.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **BUSY_BANK_STA: Busy Bank Number (cleared upon USB reset)**

These bits are set by hardware to indicate the number of busy banks.

  – **IN endpoint**: It indicates the number of busy banks filled by the user, ready for IN transfer.

  – **OUT endpoint**: It indicates the number of busy banks filled by OUT transaction from the Host.

Value	Name	Description
0	0BUSYBANK	All banks are free
1	1BUSYBANK	1 busy bank
2	2BUSYBANKS	2 busy banks
3	3BUSYBANKS	3 busy banks

- **BYTE_COUNT: UDPHS Byte Count (cleared upon USB reset)**

Byte count of a received data packet.

This field is incremented after each write into the endpoint (to prepare an IN transfer).

This field is decremented after each reading into the endpoint (OUT transfer).

This field is also updated at RXRDY_TXKL flag clear with the next bank.

This field is also updated at TXRDY_TRER flag set with the next bank.

This field is reset by EPT_x of UDPHS_EPTRST register.

- **SHRT_PCKT: Short Packet (cleared upon USB reset)**

An OUT Short Packet is detected when the receive byte count is less than the configured UDPHS_EPTCFGx register EPT_Size.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

34.7.21 UDPHS DMA Channel Transfer Descriptor

The DMA channel transfer descriptor is loaded from the memory.

Be careful with the alignment of this buffer.

The structure of the DMA channel transfer descriptor is defined by three parameters as described below:

Offset 0:

The address must be aligned: 0xXXXX0

Next Descriptor Address Register: UDPHS_DMANXTDSCx

Offset 4:

The address must be aligned: 0xXXXX4

DMA Channelx Address Register: UDPHS_DMAADDRESSx

Offset 8:

The address must be aligned: 0xXXXX8

DMA Channelx Control Register: UDPHS_DMACONTROLx

To use the DMA channel transfer descriptor, fill the structures with the correct value (as described in the following pages).

Then write directly in UDPHS_DMANXTDSCx the address of the descriptor to be used first.

Then write 1 in the LDNXT_DSC bit of UDPHS_DMACONTROLx (load next channel transfer descriptor). The descriptor is automatically loaded upon Endpointx request for packet transfer.

34.7.22 UDPHS DMA Next Descriptor Address Register

Name: UDPHS_DMANXTDSCx [x = 0..6]

Address: 0xFC02C300 [0], 0xFC02C310 [1], 0xFC02C320 [2], 0xFC02C330 [3], 0xFC02C340 [4], 0xFC02C350 [5], 0xFC02C360 [6]

Access: Read/Write

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

Note: Channel 0 is not used.

- **NXT_DSC_ADD: Next Descriptor Address**

This field points to the next channel descriptor to be processed. This channel descriptor must be aligned, so bits 0 to 3 of the address must be equal to zero.

34.7.23 UDPHS DMA Channel Address Register

Name: UDPHS_DMAADDRESSx [x = 0..6]

Address: 0xFC02C304 [0], 0xFC02C314 [1], 0xFC02C324 [2], 0xFC02C334 [3], 0xFC02C344 [4], 0xFC02C354 [5], 0xFC02C364 [6]

Access: Read/Write

31	30	29	28	27	26	25	24
BUFF_ADD							

23	22	21	20	19	18	17	16
BUFF_ADD							

15	14	13	12	11	10	9	8
BUFF_ADD							

7	6	5	4	3	2	1	0
BUFF_ADD							

Note: Channel 0 is not used.

- **BUFF_ADD: Buffer Address**

This field determines the AHB bus starting address of a DMA channel transfer.

Channel start and end addresses may be aligned on any byte boundary.

The firmware may write this field only when the UDPHS_DMASTATUS register CHANN_ENB bit is clear.

This field is updated at the end of the address phase of the current access to the AHB bus. It is incrementing of the access byte width. The access width is 4 bytes (or less) at packet start or end, if the start or end address is not aligned on a word boundary.

The packet start address is either the channel start address or the next channel address to be accessed in the channel buffer.

The packet end address is either the channel end address or the latest channel address accessed in the channel buffer.

The channel start address is written by software or loaded from the descriptor, whereas the channel end address is either determined by the end of buffer or the UDPHS device, USB end of transfer if the UDPHS_DMACONTROLx register END_TR_EN bit is set.

34.7.24 UDPHS DMA Channel Control Register

Name: UDPHS_DMACONTROLx [x = 0..6]

Address: 0xFC02C308 [0], 0xFC02C318 [1], 0xFC02C328 [2], 0xFC02C338 [3], 0xFC02C348 [4], 0xFC02C358 [5], 0xFC02C368 [6]

Access: Read/Write

31	30	29	28	27	26	25	24
BUFF_LENGTH							
23	22	21	20	19	18	17	16
BUFF_LENGTH							
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
BURST_LCK	DESC_LD_IT	END_BUFFIT	END_TR_IT	END_B_EN	END_TR_EN	LDNXT_DSC	CHANN_ENB

Note: Channel 0 is not used.

- **CHANN_ENB: (Channel Enable Command)**

0: DMA channel is disabled at and no transfer will occur upon request. This bit is also cleared by hardware when the channel source bus is disabled at end of buffer.

If the UDPHS_DMACONTROL register LDNXT_DSC bit has been cleared by descriptor loading, the firmware will have to set the corresponding CHANN_ENB bit to start the described transfer, if needed.

If the UDPHS_DMACONTROL register LDNXT_DSC bit is cleared, the channel is frozen and the channel registers may then be read and/or written reliably as soon as both UDPHS_DMASTATUS register CHANN_ENB and CHANN_ACT flags read as 0.

If a channel request is currently serviced when this bit is cleared, the DMA FIFO buffer is drained until it is empty, then the UDPHS_DMASTATUS register CHANN_ENB bit is cleared.

If the LDNXT_DSC bit is set at or after this bit clearing, then the currently loaded descriptor is skipped (no data transfer occurs) and the next descriptor is immediately loaded.

1: UDPHS_DMASTATUS register CHANN_ENB bit will be set, thus enabling DMA channel data transfer. Then any pending request will start the transfer. This may be used to start or resume any requested transfer.

- **LDNXT_DSC: Load Next Channel Transfer Descriptor Enable (Command)**

0: No channel register is loaded after the end of the channel transfer.

1: The channel controller loads the next descriptor after the end of the current transfer, i.e., when the UDPHS_DMASTATUS/CHANN_ENB bit is reset.

If the UDPHS_DMA CONTROL/CHANN_ENB bit is cleared, the next descriptor is immediately loaded upon transfer request.

DMA Channel Control Command Summary

LDNXT_DSC	CHANN_ENB	Description
0	0	Stop now
0	1	Run and stop at end of buffer
1	0	Load next descriptor now
1	1	Run and link at end of buffer

- **END_TR_EN: End of Transfer Enable (Control)**

Used for OUT transfers only.

0: USB end of transfer is ignored.

1: UDPHS device can put an end to the current buffer transfer.

When set, a BULK or INTERRUPT short packet or the last packet of an ISOCHRONOUS (micro) frame (DATAX) will close the current buffer and the UDPHS_DMASTATUSx register END_TR_ST flag will be raised.

This is intended for UDPHS non-prenegotiated end of transfer (BULK or INTERRUPT) or ISOCHRONOUS microframe data buffer closure.

- **END_B_EN: End of Buffer Enable (Control)**

0: DMA Buffer End has no impact on USB packet transfer.

1: Endpoint can validate the packet (according to the values programmed in the UDPHS_EPTCTLx register AUTO_VALID and SHRT_PCKT fields) at DMA Buffer End, i.e., when the UDPHS_DMASTATUS register BUFF_COUNT reaches 0.

This is mainly for short packet IN validation initiated by the DMA reaching end of buffer, but could be used for OUT packet truncation (discarding of unwanted packet data) at the end of DMA buffer.

- **END_TR_IT: End of Transfer Interrupt Enable**

0: UDPHS device initiated buffer transfer completion will not trigger any interrupt at UDPHS_STATUSx/END_TR_ST rising.

1: An interrupt is sent after the buffer transfer is complete, if the UDPHS device has ended the buffer transfer.

Use when the receive size is unknown.

- **END_BUFFIT: End of Buffer Interrupt Enable**

0: UDPHS_DMA_STATUSx/END_BF_ST rising will not trigger any interrupt.

1: An interrupt is generated when the UDPHS_DMASTATUSx register BUFF_COUNT reaches zero.

- **DESC_LD_IT: Descriptor Loaded Interrupt Enable**

0: UDPHS_DMASTATUSx/DESC_LDST rising will not trigger any interrupt.

1: An interrupt is generated when a descriptor has been loaded from the bus.

- **BURST_LCK: Burst Lock Enable**

0: The DMA never locks bus access.

1: USB packets AHB data bursts are locked for maximum optimization of the bus bandwidth usage and maximization of fly-by AHB burst duration.

- **BUFF_LENGTH: Buffer Byte Length (Write-only)**

This field determines the number of bytes to be transferred until end of buffer. The maximum channel transfer size (64 KBytes) is reached when this field is 0 (default value). If the transfer size is unknown, this field should be set to 0, but the transfer end may occur earlier under UDPHS device control.

When this field is written, The UDPHS_DMASTATUSx register BUFF_COUNT field is updated with the write value.

Notes:
1. Bits [31:2] are only writable when issuing a channel Control Command other than “Stop Now”.
2. For reliability it is highly recommended to wait for both UDPHS_DMASTATUSx register CHAN_ACT and CHAN_ENB flags are at 0, thus ensuring the channel has been stopped before issuing a command other than “Stop Now”.

34.7.25 UDPHS DMA Channel Status Register

Name: UDPHS_DMASTATUSx [x = 0..6]

Address: 0xFC02C30C [0], 0xFC02C31C [1], 0xFC02C32C [2], 0xFC02C33C [3], 0xFC02C34C [4], 0xFC02C35C [5], 0xFC02C36C [6]

Access: Read/Write

31	30	29	28	27	26	25	24
BUFF_COUNT							
23	22	21	20	19	18	17	16
BUFF_COUNT							
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	DESC_LDST	END_BF_ST	END_TR_ST	–	–	CHANN_ACT	CHANN_ENB

Note: Channel 0 is not used.

- **CHANN_ENB: Channel Enable Status**

0: The DMA channel no longer transfers data, and may load the next descriptor if the UDPHS_DMACONTROLx register LDNXT_DSC bit is set.

When any transfer is ended either due to an elapsed byte count or a UDPHS device initiated transfer end, this bit is automatically reset.

1: The DMA channel is currently enabled and transfers data upon request.

This bit is normally set or cleared by writing into the UDPHS_DMACONTROLx register CHANN_ENB bit either by software or descriptor loading.

If a channel request is currently serviced when the UDPHS_DMACONTROLx register CHANN_ENB bit is cleared, the DMA FIFO buffer is drained until it is empty, then this status bit is cleared.

- **CHANN_ACT: Channel Active Status**

0: The DMA channel is no longer trying to source the packet data.

When a packet transfer is ended this bit is automatically reset.

1: The DMA channel is currently trying to source packet data, i.e., selected as the highest-priority requesting channel.

When a packet transfer cannot be completed due to an END_BF_ST, this flag stays set during the next channel descriptor load (if any) and potentially until UDPHS packet transfer completion, if allowed by the new descriptor.

- **END_TR_ST: End of Channel Transfer Status**

0: Cleared automatically when read by software.

1: Set by hardware when the last packet transfer is complete, if the UDPHS device has ended the transfer.

Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

- **END_BF_ST: End of Channel Buffer Status**

0: Cleared automatically when read by software.

1: Set by hardware when the BUFF_COUNT countdown reaches zero.

Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

- **DESC_LDST: Descriptor Loaded Status**

0: Cleared automatically when read by software.

1: Set by hardware when a descriptor has been loaded from the system bus.

Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

- **BUFF_COUNT: Buffer Byte Count**

This field determines the current number of bytes still to be transferred for this buffer.

This field is decremented from the AHB source bus access byte width at the end of this bus address phase.

The access byte width is 4 by default, or less, at DMA start or end, if the start or end address is not aligned on a word boundary.

At the end of buffer, the DMA accesses the UDPHS device only for the number of bytes needed to complete it.

This field value is reliable (stable) only if the channel has been stopped or frozen (UDPHS_EPTCTLx register NT_DIS_DMA bit is used to disable the channel request) and the channel is no longer active CHANN_ACT flag is 0.

Note: For OUT endpoints, if the receive buffer byte length (BUFF_LENGTH) has been defaulted to zero because the USB transfer length is unknown, the actual buffer byte length received will be 0x10000-BUFF_COUNT.

35. USB Host High Speed Port (UHPHS)

35.1 Description

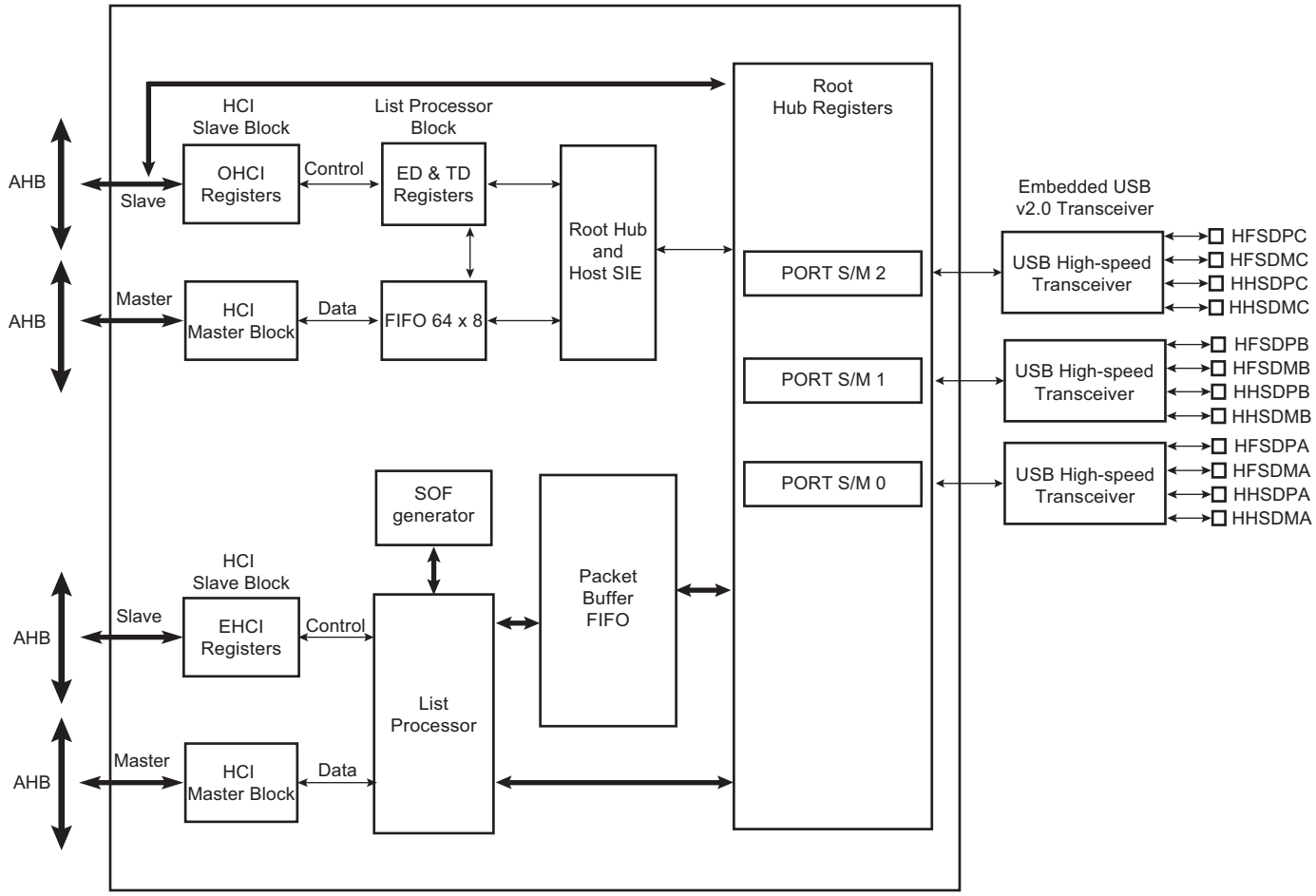
The USB Host High Speed Port (UHPHS) interfaces the USB with the host application. It handles Open HCI protocol (Open Host Controller Interface) as well as Enhanced HCI protocol (Enhanced Host Controller Interface).

35.2 Embedded Characteristics

- Compliant with Enhanced HCI Rev 1.0 Specification
  - Compliant with USB V2.0 High-speed
  - Supports High-speed 480 Mbps
- Compliant with Open HCI Rev 1.0 Specification
  - Compliant with USB V2.0 Full-speed and Low-speed Specification
  - Supports both Low-speed 1.5 Mbps and Full-speed 12 Mbps USB devices
- Root Hub Integrated with 3 Downstream USB HS Ports
- Embedded USB Transceivers
- Supports Power Management
- 3 Hosts (A, B, and C) High Speed (EHCI), Port A shared with UDPHS

35.3 Block Diagram

Figure 35-1. Block Diagram



Access to the USB host operational registers is achieved through the AHB bus slave interface. The Open HCI host controller and Enhanced HCI host controller initialize master DMA transfers through the AHB bus master interface as follows:

- Fetches endpoint descriptors and transfer descriptors
- Access to endpoint data from system memory
- Access to the HC communication area
- Write status and retire transfer descriptor

Memory access errors (abort, misalignment) lead to an “Unrecoverable Error” indicated by the corresponding flag in the host controller operational registers.

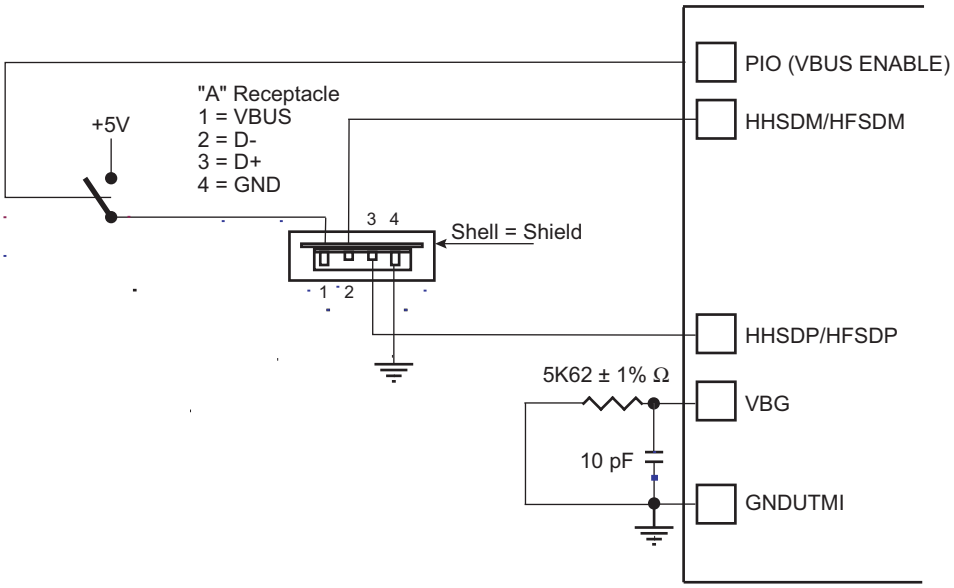
The USB root hub is integrated in the USB host. Several USB downstream ports are available. The number of downstream ports can be determined by the software driver reading the root hub’s operational registers. Device connection is automatically detected by the USB host port logic.

USB physical transceivers are integrated in the product and driven by the root hub’s ports.

Over current protection on ports can be activated by the USB host controller. Atmel’s standard product does not dedicate pads to external over current protection.

35.4 Typical Connection

Figure 35-2. Board Schematic to Interface UHP High-speed Host Controller



Note: 1. 10 pF capacitor on VBG is a provision and may not be populated.

35.5 Product Dependencies

35.5.1 I/O Lines

HFSDPs, HFSDMs, HHSDPs and HHSDMs are not controlled by any PIO controllers. The embedded USB High Speed physical transceivers are controlled by the USB host controller.

One transceiver is shared with the USB High Speed Device (port A). The selection between Host Port A and USB Device is controlled by the UDPHS enable bit (EN_UDPHS) located in the UDPHS_CTRL register.

In the case the port A is driven by the USB High Speed Device, the output signals are DFSDP, DFSDM, DHSDP and DHSDM. The transceiver is automatically selected for Device operation once the USB High Speed Device is enabled.

In the case the port A is driven by the USB High Speed Host, the output signals are HFSDPA, HFSDMA, HHSDPA and HHSDMA.

35.5.2 Power Management

The system embeds 3 transceivers.

The USB Host High Speed requires a 480 MHz clock for the embedded High-speed transceivers. This clock (UPLLCK) is provided by the UTMI PLL.

In case power consumption is saved by stopping the UTMI PLL, high-speed operations are not possible. Nevertheless, OHCI Full-speed operations remain possible by selecting PLLACK as the input clock of OHCI.

The High-speed transceiver returns a 30 MHz clock to the USB Host controller.

The USB Host controller requires 48 MHz and 12 MHz clocks for OHCI full-speed operations. These clocks must be generated by a PLL with a correct accuracy of ± 0.25% using the USBDIV field.

Thus the USB Host peripheral receives three clocks from the Power Management Controller (PMC): the Peripheral Clock (MCK domain), the UHP48M and the UHP12M (built-in UHP48M divided by four) used by the OHCI to interface with the bus USB signals (recovered 12 MHz domain) in Full-speed operations.

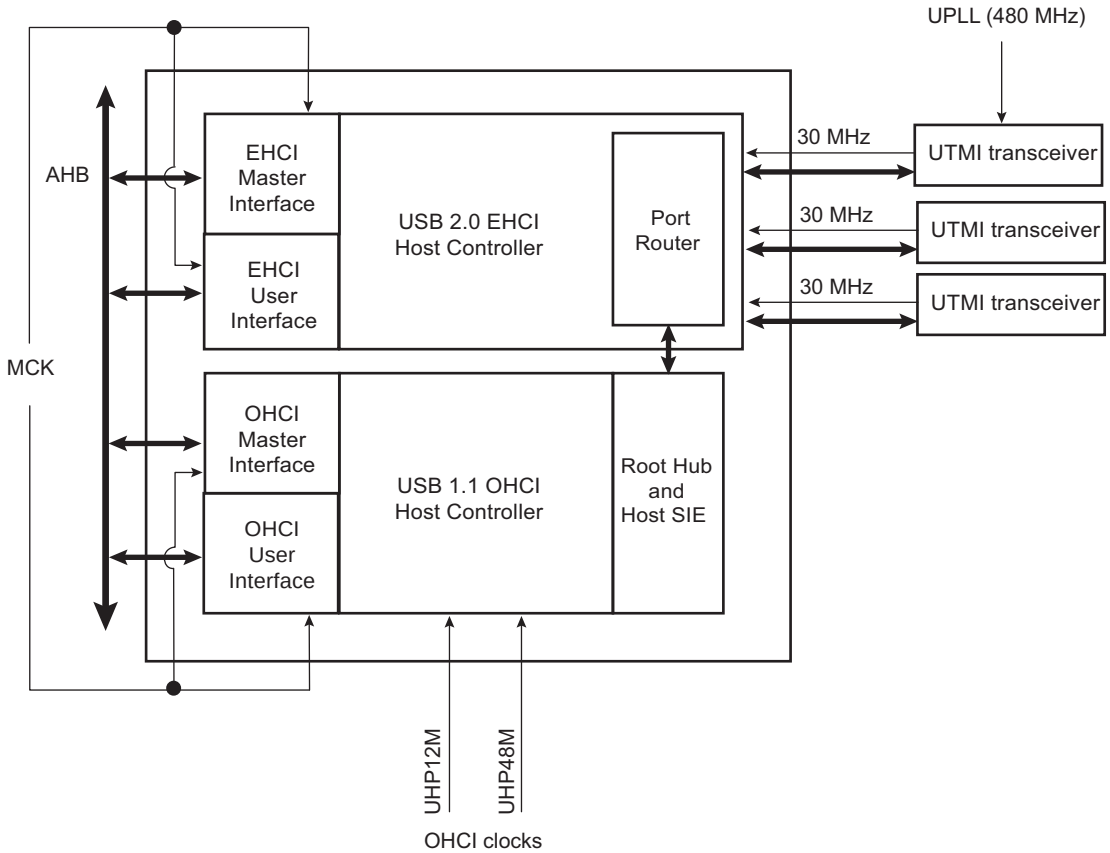
For High-speed operations, the user has to perform the following:

- Enable UHP peripheral clock in PMC_PCER.
- Write PLLCOUNT field in CKGR_UCKR.
- Enable UPLL with UPLLEN bit in CKGR_UCKR.
- Wait until UTMI_PLL is locked (LOCKU bit in PMC_SR).
- Enable BIAS with BIASEN bit in CKGR_UCKR.
- Select UPLLCK as Input clock of OHCI part (USBS bit in PMC_USB register).
- Program OHCI clocks (UHP48M and UHP12M) with USBDIV field in PMC_USB register. USBDIV must be 9 (division by 10) if UPLLCK is selected.
- Enable OHCI clocks with UHP bit in PMC_SCER.

For OHCI Full-speed operations only, the user has to perform the following:

- Enable UHP peripheral clock in PMC_PCER.
- Select PLLACK as Input clock of OHCI part (USBS bit in PMC_USB register).
- Program OHCI clocks (UHP48M and UHP12M) with USBDIV field in PMC_USB register. USBDIV value is to be calculated according to the PLLACK value and USB Full-speed accuracy.
- Enable the OHCI clocks with UHP bit in PMC_SCER.

Figure 35-3. UHP Clock Trees



35.5.3 Interrupt Sources

The USB host interface has an interrupt line connected to the interrupt controller.

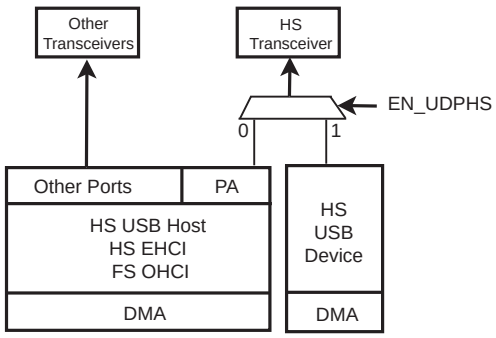
Handling USB host interrupts requires programming the interrupt controller before configuring the UHPHS.

35.6 Functional Description

35.6.1 UTMI Transceivers Sharing

The High Speed USB Host Port A is shared with the High Speed USB Device port and connected to the second UTMI transceiver. The selection between Host Port A and USB device is controlled by the UDPHS enable bit (EN_UDPHS) located in the UDPHS_CTRL register.

Figure 35-4. USB Selection



35.6.2 EHCI

The USB Host Port controller is fully compliant with the Enhanced HCI specification. The USB Host Port User Interface (registers description) can be found in the Enhanced HCI Rev 1.0 Specification available on www.usb.org. The standard EHCI USB stack driver can be easily ported to Atmel's architecture in the same way all existing class drivers run, without hardware specialization.

35.6.3 OHCI

The USB Host Port integrates a root hub and transceivers on downstream ports. It provides several Full-speed half-duplex serial communication ports at a baud rate of 12 Mbps. Up to 127 USB devices (printer, camera, mouse, keyboard, disk, etc.) and the USB hub can be connected to the USB host in the USB "tiered star" topology.

The USB Host Port controller is fully compliant with the Open HCI specification. The USB Host Port User Interface (registers description) can be found in the Open HCI Rev 1.0 Specification available on www.usb.org. The standard OHCI USB stack driver can be easily ported to Atmel's architecture, in the same way all existing class drivers run without hardware specialization.

This means that all standard class devices are automatically detected and available to the user's application. As an example, integrating an HID (Human Interface Device) class driver provides a plug & play feature for all USB keyboards and mouses.

35.7 USB Host High Speed Port (UHPHS) User Interface

The Enhanced USB Host Controller contains two sets of software-accessible hardware registers: memory-mapped Host Controller Registers and optional PCI configuration registers. Note that the PCI configuration registers are only needed for PCI devices that implement the Host Controller.

- Memory-mapped USB Host Controller Registers—This block of registers is memory-mapped into non-cacheable memory. This memory space must begin on a DWord (32-bit) boundary. This register space is divided into two sections: a set of read-only capability registers and a set of read/write operational registers. Table 35-1 describes each register space.

Note: Host controllers are not required to support exclusive-access mechanisms (such as PCI LOCK) for accesses to the memory-mapped register space. Therefore, if software attempts exclusive-access mechanisms to the host controller memory-mapped register space, the results are undefined.

- PCI Configuration Registers (for PCI devices)—In addition to the normal PCI header, power management, and device-specific registers, two registers are needed in the PCI configuration space to support USB. The normal PCI header and device-specific registers are beyond the scope of this document (the UHPHS_CLASSC register is shown in this document). Note that HCD does not interact with the PCI configuration space. This space is used only by the PCI enumerator to identify the USB Host Controller, and assign the appropriate system resources.

Table 35-1. Enhanced Interface Register Sets

Offset	Register Set	Explanation
0 to N-1	Capability Registers	The capability registers specify the limits, restrictions, and capabilities of a host controller implementation. These values are used as parameters to the host controller driver.
N to N+M-1	Operational Registers	The operational registers are used by system software to control and monitor the operational state of the host controller.

Table 35-2. Register Mapping

Offset	Register	Name	Access	Reset
Host Controller Capability Registers				
0x00	UHPHS Host Controller Capability Register	UHPHS_HCCAPBASE	Read-only	0x0100 0010
0x04	UHPHS Host Controller Structural Parameters Register	UHPHS_HCSPARAMS	Read-only	0x0000 1116
0x08	UHPHS Host Controller Capability Parameters Register	UHPHS_HCCPARAMS	Read-only	0x0000 A010
0x0C	Reserved	–	–	–
Host Controller Operational Registers				
0x10	UHPHS USB Command Register	UHPHS_USBCMD	Read/Write(1)	0x0008 0000 or 0x0008 0B00(2)
0x14	UHPHS USB Status Register	UHPHS_USBSTS	Read/Write(1)	0x0000 1000
0x18	UHPHS USB Interrupt Enable Register	UHPHS_USBINTR	Read/Write	0x0000 0000
0x1C	UHPHS USB Frame Index Register	UHPHS_FRINDEX	Read/Write	0x0000 0000
0x20	UHPHS Control Data Structure Segment Register	UHPHS_CTRLDSSEGMENT	Read/Write	0x0000 0000

Table 35-2. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x24	UHPHS Periodic Frame List Base Address Register	UHPHS_PERIODICLISTBASE	Read/Write	0x0000 0000
0x28	UHPHS Asynchronous List Address Register	UHPHS_ASYNCLISTADDR	Read/Write	0x0000 0000
0x2C–0x4F	Reserved	–	–	–
0x50	UHPHS Configured Flag Register	UHPHS_CONFIGFLAG	Read/Write	0x0000 0000
0x54	UHPHS Port Status and Control Register 0	UHPHS_PORTSC_0	Read/Write(1)	0x0000 2000 or 0x0000 3000(3)
0x58	UHPHS Port Status and Control Register 1	UHPHS_PORTSC_1	Read/Write(1)	0x0000 2000 or 0x0000 3000(3)
0x5C	UHPHS Port Status and Control Register 2	UHPHS_PORTSC_2	Read/Write(1)	0x0000 2000 or 0x0000 3000(3)
0x90	EHCI Synopsys-Specific Registers 00	UHPHS_INSNREG00	Read/Write(1)	0x0000 0000
0x94	EHCI Synopsys-Specific Registers 01	UHPHS_INSNREG01	Read/Write(1)	0x0020 0020
0x98	EHCI Synopsys-Specific Registers 02	UHPHS_INSNREG02	Read/Write(1)	(5)
0x9C	EHCI Synopsys-Specific Registers 03	UHPHS_INSNREG03	Read/Write(1)	0x0000 0001
0xA0	EHCI Synopsys-Specific Registers 04	UHPHS_INSNREG04	Read/Write(1)	0x0000 0000
0xA4	EHCI Synopsys-Specific Registers 05	UHPHS_INSNREG05	Read/Write(1)	0x0000 1000
0xA8	EHCI Synopsys-Specific Registers 06	UHPHS_INSNREG06	Read/Write(1)	0x0000 0000
0xAC	EHCI Synopsys-Specific Registers 07	UHPHS_INSNREG07	Read/Write(1)	0x0000 0000
0xB0	EHCI Synopsys-Specific Registers 08	UHPHS_INSNREG08	Read/Write(1)	0x0000 0000

Notes:
1. Field-dependent.
2. The default value depends on whether the Asynchronous Schedule Park Capability (ASPC) field in the UHPHS_HCCPARAMS register is enabled: Disabled (set to 0) = 0x0008 0000h; Enabled (set to 1) = 0x0008 0B00h.
3. The default value depends on the value of the Port Power Control (PPC) field in the UHPHS_HCSPARAMS register: 0x0000 2000h (with PPC set to 1); 0x0000 3000h (with PPC set to 0).
4. Software should not assume reserved bits are always 0 and should preserve these bits when writing to modifiable registers.
5. This value is determined by coreConsultant.

35.7.1 UHPHS Host Controller Capability Register

Name: UHPHS_HCCAPBASE

Access: Read-only

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

- **CAPLENGTH: Capability Registers Length**

10h: Default value.

This field is used as an offset to add to register base to find the beginning of the Operational Register Space.

- **HCIVERSION: Host Controller Interface Version Number**

0100h: Default value.

This is a two-byte field containing a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this field represents a major revision and the least significant byte is the minor revision.

35.7.2 UHPHS Host Controller Structural Parameters Register

Name: UHPHS_HCSPARAMS

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
N_DP				–	–	–	P_INDICATOR

15	14	13	12	11	10	9	8
N_CC				N_PCC			

7	6	5	4	3	2	1	0
–	–	–	PPC	N_PORTS			

This is a set of fields that are structural parameters: number of downstream ports, etc.

- **N_PORTS: Number of Ports**

This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register Space. Valid values are in the range of 1H to FH.

A zero in this field is undefined.

- **PPC: Port Power Control**

This field indicates whether the host controller implementation includes port power control. A one in this bit indicates the ports have port power switches. A zero in this bit indicates the ports do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register (refer to Section 35.7.12).

- **N_PCC: Number of Ports per Companion Controller**

This field indicates the number of ports supported per companion host controller. It is used to indicate the port routing configuration to system software.

For example, if N_PORTS has a value of 6 and N_CC has a value of 2, then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first four are routed to companion controller 1 and the last two are routed to companion controller 2.

The number in this field must be consistent with N_PORTS and N_CC.

- **N_CC: Number of Companion Controllers**

This field indicates the number of companion controllers associated with this USB 2.0 host controller.

A zero in this field indicates there are no companion host controllers. Port-ownership hand-off is not supported. Only high-speed devices are supported on the host controller root ports.

A value larger than zero in this field indicates there are companion USB 1.1 host controller(s). Port-ownership hand-offs are supported. High, Full- and Low-speed devices are supported on the host controller root ports.

- **P_INDICATOR: Port Indicators**

This bit indicates whether the ports support port indicator control. When this bit is a 1, the port status and control registers include a read/writeable field for controlling the state of the port indicator. Refer to Section 35.7.12 for definition of the port indicator control field.

- **N_DP: Debug Port Number**

Optional. This register identifies which of the host controller ports is the debug port. The value is the port number (1-based) of the debug port. A non-zero value in this field indicates the presence of a debug port. The value in this register must not be greater than N_PORTS (refer to "N_PORTS: Number of Ports").

35.7.3 UHPHS Host Controller Capability Parameters Register

Name: UHPHS_HCCPARAMS

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
EECP							

7	6	5	4	3	2	1	0
IST				–	ASPC	PFLF	AC

This is a set of fields that are capability parameters: Multiple Mode control (time-base bit functionality), addressing capability, etc.

- **AC: 64-bit Addressing Capability**

This field documents the addressing range capability of this implementation. The value of this field determines whether software should use 32-bit or 64-bit data structures.

Values for this field have the following interpretation:

0: Data structures using 32-bit address memory pointers

1: Data structures using 64-bit address memory pointers

Note: This is not tightly coupled with the UHPHS_USBBASE address register mapping control. The 64-bit Addressing Capability bit indicates whether the host controller can generate 64-bit addresses as a master. The UHPHS_USBBASE register indicates the host controller only needs to decode 32-bit addresses as a slave.

- **PFLF: Programmable Frame List Flag**

The default value is implementation-dependent.

If this bit is set to 0, then system software must use a frame list length of 1024 elements with this host controller. The UHPHS_USBCMD register Frame List Size field is a read-only register and should be set to 0.

If set to 1, then system software can specify and use a smaller frame list and configure the host controller via the UHPHS_USBCMD register Frame List Size field. The frame list must always be aligned on a 4-Kbyte page boundary. This requirement ensures that the frame list is always physically contiguous.

- **ASPC: Asynchronous Schedule Park Capability**

The default value is Implementation dependent.

If this bit is set to 1, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. The feature can be disabled or enabled and set to a specific level by using the Asynchronous Schedule Park Mode Enable and Asynchronous Schedule Park Mode Count fields in the UHPHS_USBCMD register.

- **IST: Isochronous Scheduling Threshold**

The default value is Implementation dependent.

This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is 0, the value of the least significant 3 bits indicates the number of microframes a host

controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is set to 1, then host software assumes the host controller may cache an isochronous data structure for an entire frame.

- **EECP: EHCI Extended Capabilities Pointer**

The default value is Implementation dependent.

This optional field indicates the existence of a capabilities list. A value of 00h indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. The pointer value must be 40h or greater if implemented to maintain the consistency of the PCI header defined for this class of device.

35.7.4 UHPHS USB Command Register

Name: UHPHS_USBCMD

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
ITC							

15	14	13	12	11	10	9	8
–	–	–	–	ASPME	–	ASPMC	

7	6	5	4	3	2	1	0
LHCR	IAAD	ASE	PSE	FLS		HCRESET	RS

The Command Register indicates the command to be executed by the serial bus host controller. Writing to the register causes a command to be executed.

- **RS: Run/Stop (read/write)**

0: Stop (default value).

1: Run.

When set to 1, the Host Controller proceeds with execution of the schedule. The Host Controller continues execution as long as this bit is set to 1. When this bit is set to 0, the Host Controller completes the current and any actively pipelined transactions on the USB and then halts. The Host Controller must halt within 16 microframes after software clears the Run bit. The HC Halted bit in the status register indicates when the Host Controller has finished its pending pipelined transactions and has entered the stopped state. Software must not write 1 to this field unless the host controller is in the Halted state (i.e., HCHalted in the UHPHS_USBSTS register is 1). Doing so will yield undefined results.

- **HCRESET: Host Controller Reset (read/write)**

This control bit is used by software to reset the host controller. The effects of this on Root Hub registers are similar to a Chip Hardware Reset.

When software writes a 1 to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports.

PCI Configuration registers are not affected by this reset. All operational registers, including port registers and port state machines, are set to their initial values. Port ownership reverts to the companion host controller(s) with side effects. Software must reinitialize the host controller in order to return the host controller to an operational state.

This bit is set to 0 by the Host Controller when the reset process is complete. Software cannot terminate the reset process early by writing a 0 to this register.

Software should not set this bit to 1 when the HCHalted bit in the UHPHS_USBSTS register is 0. Attempting to reset an actively running host controller will result in undefined behavior.

- **FLS: Frame List Size (read/write or read-only)**

This field is R/W only if Programmable Frame List Flag in the UHPHS_HCCPARAMS registers is set to 1. This field specifies the size of the frame list. The size of the frame list controls which bits in the Frame Index Register should be used for the Frame List Current index.

00b: 1024 elements (4096 bytes) (default value).

01b: 512 elements (2048 bytes).

10b: 256 elements (1024 bytes), for resource-constrained environments.

11b: Reserved.

- **PSE: Periodic Schedule Enable (read/write)**

This bit controls whether the host controller skips processing the Periodic Schedule.

0: Do not process the Periodic Schedule (default value).

1: Use the UHPHS_PERIODICLISTBASE register to access the Periodic Schedule.

- **ASE: Asynchronous Schedule Enable (read/write)**

This bit controls whether the host controller skips processing the Asynchronous Schedule.

0: Do not process the Asynchronous Schedule (default value).

1: Use the UHPHS_ASYNCLISTADDR register to access the Asynchronous Schedule.

- **IAAD: Interrupt on Async Advance Doorbell (read/write)**

This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell.

When the host controller has evicted all appropriate cached schedule state, it sets the Interrupt on Async Advance status bit in the UHPHS_USBSTS register. If the Interrupt on Async Advance Enable bit in the UHPHS_USBINTR register is set to 1, then the host controller will assert an interrupt at the next interrupt threshold.

The host controller sets this bit to 0 after it has set the Interrupt on Async Advance status bit in the UHPHS_USBSTS register to 1.

Software should not write a 1 to this bit when the asynchronous schedule is disabled. Doing so will yield undefined results.

- **LHCR: Light Host Controller Reset (optional) (read/write)**

This control bit is not required. If implemented, it allows the driver to reset the EHCI controller without affecting the state of the ports or the relationship to the companion host controllers. For example, the UHPHS_PORTSC registers should not be reset to their default values and the CF bit setting should not go to 0 (retaining port ownership relationships).

A host software read of this bit as 0 indicates the Light Host Controller Reset has completed and it is safe for host software to re-initialize the host controller. A host software read of this bit as 1 indicates the Light Host Controller Reset has not yet completed.

If not implemented, a read of this field will always return a 0.

- **ASPMC: Asynchronous Schedule Park Mode Count (optional) (read/write or read-only)**

If the Asynchronous Park Capability bit in the UHPHS_HCCPARAMS register is set to 1, then this field defaults to 3h and is R/W. Otherwise it defaults to 0 and is RO. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 1h to 3h. Software must not write a 0 to this bit when Park Mode Enable is set to 1 as this will result in undefined behavior.

- **ASPME: Asynchronous Schedule Park Mode Enable (optional) (read/write or read-only)**

If the Asynchronous Park Capability bit in the UHPHS_HCCPARAMS register is set to 1, then this bit defaults to a 1h and is R/W. Otherwise the bit must be a 0 and is RO. Software uses this bit to enable or disable Park mode. When this bit is set to 1, Park mode is enabled. When this bit is set to 0, Park mode is disabled.

- **ITC: Interrupt Threshold Control (read/write)**

This field is used by system software to select the maximum rate at which the host controller will issue interrupts. The only valid values are defined below. If software writes an invalid value to this register, the results are undefined.

Value	Maximum Interrupt Interval
00h	Reserved
01h	1 microframe
02h	2 microframes
04h	4 microframes
08h	8 microframes (default, equates to 1 ms)
10h	16 microframes (2 ms)
20h	32 microframes (4 ms)
40h	64 microframes (8 ms)

Any other value in this register yields undefined results.

Software modifications to this bit while HCHalted bit is equal to 0 results in undefined behavior.

35.7.5 UHPHS USB Status Register

Name: UHPHS_USBSTS

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
ASS	PSS	RCM	HCHLT	–	–	–	–
7	6	5	4	3	2	1	0
–	–	IAA	HSE	FLR	PCD	USBERRINT	USBINT

This register indicates pending interrupts and various states of the Host Controller. The status resulting from a transaction on the serial bus is not indicated in this register. Software sets a bit to 0 in this register by writing a 1 to it.

- **USBINT: USB Interrupt (read/write clear)**

The Host Controller sets this bit to 1 on the completion of a USB transaction, which results in the retirement of a Transfer Descriptor that had its IOC bit set.

The Host Controller also sets this bit to 1 when a short packet is detected (the actual number of bytes received was less than the expected number of bytes).

- **USBERRINT: USB Error Interrupt (read/write clear)**

The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and USBINT bit are set.

- **PCD: Port Change Detect (read/write clear)**

The Host Controller sets this bit to 1 when any port for which the Port Owner bit is set to 0 (refer to Section 35.7.12) has a change bit transition from 0 to 1 or a Force Port Resume bit transition from 0 to 1 as a result of a J-K transition detected on a suspended port. This bit will also be set as a result of the Connect Status Change being set to 1 after system software has relinquished ownership of a connected port by writing 1 to a port's Port Owner bit.

This bit is allowed to be maintained in the Auxiliary power well. Alternatively, it is also acceptable that on a D3 to D0 transition of the EHCI HC device, this bit is loaded with the OR of all of the PORTSC change bits (including: Force Port Resume, Over-Current Change, Enable/Disable Change and Connect Status Change).

- **FLR: Frame List Rollover (read/write clear)**

The Host Controller sets this bit to 1 when the Frame List Index (refer to Section 35.7.7) rolls over from its maximum value to 0. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the Frame List Size field of the UHPHS_USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX[13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to 1 every time FRINDEX[12] toggles.

- **HSE: Host System Error (read/write clear)**

The Host Controller sets this bit to 1 when a serious error occurs during a host system access involving the Host Controller module. In a PCI system, conditions that set this bit to 1 include PCI Parity error, PCI Master Abort, and PCI Target Abort. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs.

- **IAA: Interrupt on Async Advance (read/write clear)**

0: Default.

System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing 1 to the Interrupt on the Async Advance Doorbell bit in the UHPHS_USBCMD register. This status bit indicates the assertion of that interrupt source.

- **HCHLT: HCHalted (read-only)**

1: Default.

This bit is 0 whenever the Run/Stop bit is 1. The Host Controller sets this bit to 1 after it has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (e.g. internal error).

- **RCM: Reclamation (read-only)**

0: Default.

This is a read-only status bit used to detect any empty asynchronous schedule.

- **PSS: Periodic Schedule Status (read-only)**

0: Default.

The bit reports the current real status of the Periodic Schedule. If this bit is set to 0, then the status of the Periodic Schedule is disabled. If this bit is set to 1, then the status of the Periodic Schedule is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the UHPHS_USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0).

- **ASS: Asynchronous Schedule Status (read-only)**

0: Default.

The bit reports the current real status of the Asynchronous Schedule. If this bit is set to 0, then the status of the Asynchronous Schedule is disabled. If this bit is set to 1, then the status of the Asynchronous Schedule is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the UHPHS_USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0).

35.7.6 UHPHS USB Interrupt Enable Register

Name: UHPHS_USBINTR

Access: Read/Write

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
–	–	IAAE	HSEE	FLRE	PCIE	USBEIE	USBIE

This register enables and disables reporting of the corresponding interrupt to the software. When a bit is set and the corresponding interrupt is active, an interrupt is generated to the host. Interrupt sources that are disabled in this register still appear in the UHPHS_USBSTS to allow the software to poll for events.

Each interrupt enable bit description indicates whether it is dependent on the interrupt threshold mechanism.

For all enable register bits, 1= Enabled, 0= Disabled.

- **USBIE: USB Interrupt Enable**

When this bit is set to 1, and the USBINT bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBINT bit.

- **USBEIE: USB Error Interrupt Enable**

When this bit is set to 1, and the USBERRINT bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBERRINT bit.

- **PCIE: Port Change Interrupt Enable**

When this bit is set to 1, and the Port Change Detect bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Port Change Detect bit.

- **FLRE: Frame List Rollover Enable**

When this bit is set to 1, and the Frame List Rollover bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Frame List Rollover bit.

- **HSEE: Host System Error Enable**

When this bit is set to 1, and the Host System Error Status bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Host System Error bit.

- **IAAE: Interrupt on Async Advance Enable**

When this bit is set to 1, and the Interrupt on Async Advance bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the Interrupt on Async Advance bit.

35.7.7 UHPHS USB Frame Index Register

Name: UHPHS_FRINDEX

Access: Read/Write

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

This register is used by the host controller to index into the periodic frame list. The register updates every 125 µs (once each microframe). Bits [N:3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the UHPHS_USBCMD register (refer to Section 35.7.4).

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the Halted state as indicated by the HCHalted bit (UHPHS_USBSTS register, Section 35.7.5). A write to this register while the Run/Stop bit is set to 1 (UHPHS_USBCMD register, Section 35.7.4) produces undefined results. Writes to this register also affect the SOF value.

- **FI: Frame Index**

The value in this register increments at the end of each time frame (e.g., microframe). Bits [N:3] are used for the Frame List current index. This means that each location of the frame list is accessed eight times (frames or microframes) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size field in the UHPHS_USBCMD register.

USBCMD [Frame List Size]	Number Elements	N
00b	(1024)	12
01b	(512)	11
10b	(256)	10
11b	Reserved	–

The SOF frame number value for the bus SOF token is derived or alternatively managed from this register. The value of FRINDEX must be 125 µs (1 microframe) ahead of the SOF token value. The SOF value may be implemented as an 11-bit shadow register. For this discussion, this shadow register is 11 bits and is named SOFV. SOFV updates every eight microframes (1 millisecond). An example implementation to achieve this behavior is to increment SOFV each time the FRINDEX[2:0] increments from 0 to 1.

Software must use the value of FRINDEX to derive the current microframe number, both for high-speed isochronous scheduling purposes and to provide the “get microframe number” function required for client drivers. Therefore, the value of FRINDEX and the value of SOFV must be kept consistent if chip is reset or software writes to FRINDEX. Writes to FRINDEX must also write-through FRINDEX[13:3] to SOFV[10:0]. In order to keep the update as simple as possible, software should never write a FRINDEX value where the three least significant bits are 111b or 000b.

35.7.8 UHPHS Control Data Structure Segment Register

Name: UHPHS_CTRLDSSEGMENT

Access: Read/Write

This 32-bit register corresponds to the most significant address bits [63:32] for all EHCI data structures. If the 64-bit Addressing Capability field in UHPHS_HCCPARAMS is set to 0, then this register is not used. Software cannot write to it and a read from this register will return zeros.

If the 64-bit Addressing Capability field in UHPHS_HCCPARAMS is 1, then this register is used with the link pointers to construct 64-bit addresses to EHCI control data structures. This register is concatenated with the link pointer from either the UHPHS_PERIODICLISTBASE, UHPHS_ASYNCLISTADDR, or any control data structure link field to construct a 64-bit address.

This register must be written as a DWord. Byte writes produce undefined results. This register allows the host software to locate all control data structures within the same 4-Gigabyte memory segment.

35.7.9 UHPHS Periodic Frame List Base Address Register

Name: UHPHS_PERIODICLISTBASE

Access: Read/Write

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

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. If the host controller is in 64-bit mode (as indicated by a 1 in the 64-bit Addressing Capability field in the UHPHS_HCCSPARAMS register), then the most significant 32 bits of every control data structure address comes from the UHPHS_CTRLDSSEGMENT register (refer to Section 35.7.8). System software loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4-Kbyte aligned. The contents of this register are combined with the Frame Index Register (UHPHS_FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence. This register must be written as a DWord. Byte writes produce undefined results.

- **BA: Base Address (Low)**

These bits correspond to memory address signals [31:12], respectively.

35.7.10 UHPHS Asynchronous List Address Register

Name: UHPHS_ASYNCLISTADDR

Access: Read/Write

31	30	29	28	27	26	25	24
LPL							

23	22	21	20	19	18	17	16
LPL							

15	14	13	12	11	10	9	8
LPL							

7	6	5	4	3	2	1	0
LPL			–	–	–	–	–

This 32-bit register contains the address of the next asynchronous queue head to be executed. If the host controller is in 64-bit mode (as indicated by a 1 in the 64-bit Addressing Capability field in the UHPHS_HCCPARAMS register), then the most significant 32 bits of every control data structure address comes from the UHPHS_CTRLDSSEGMENT register (refer to Section 35.7.8). Bits [4:0] of this register cannot be modified by system software and will always return a zero when read. The memory structure referenced by this physical memory pointer is assumed to be 32-byte (cache line) aligned. This register must be written as a DWord. Byte writes produce undefined results.

- **LPL: Link Pointer Low**

These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH).

35.7.11 UHPHS Configure Flag Register

Name: UHPHS_CONFIGFLAG

Access: Read/Write

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
–	–	–	–	–	–	–	CF

This register is in the auxiliary power well. It is only reset by hardware when the auxiliary power is initially applied or in response to a host controller reset.

- **CF: Configure Flag (read/write)**

Host software sets this bit as the last action in its process of configuring the Host Controller. This bit controls the default port-routing control logic. Bit values and side-effects are listed below.

0: Port routing control logic default-routes each port to an implementation-dependent classic host controller (default value).

1: Port routing control logic default-routes all ports to this host controller.

35.7.12 UHPHS Port Status and Control Register

Name: UHPHS_PORTSC_x[x = 0..2]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	WKOC_E	WKDSCNNT_E	WKCNNT_E	PTC			

15	14	13	12	11	10	9	8
PIC		PO	PP	LS		–	PR

7	6	5	4	3	2	1	0
SUS	FPR	OCC	OCA	PEDC	PED	CSC	CCS

A host controller must implement one or more port registers. The number of port registers implemented by a particular instantiation of a host controller is documented in the UHPHS_HCSPARAMS register (Section 35.7.2). Software uses this information as an input parameter to determine how many ports need to be serviced. All ports have the structure defined below.

This register is in the auxiliary power well. It is only reset by hardware when the auxiliary power is initially applied or in response to a host controller reset. The initial conditions of a port are:

- No device connected
- Port disabled

If the port has port power control, software cannot change the state of the port until after it applies power to the port by setting port power to a 1. Software must not attempt to change the state of the port until after power is stable on the port. The host is required to have power stable to the port within 20 milliseconds of the 0 to 1 transition.

Notes:
1. When a device is attached, the port state transitions to the connected state and system software will process this as with any status change notification.
2. If a port is being used as the Debug Port, then the port may report device connected and enabled when the Configured Flag is set to 0.

- **CCS: Current Connect Status (read-only)**

0: No device is present (default value).

1: Device is present on port.

This value reflects the current state of the port, and may not correspond directly to the event that caused the Connect Status Change bit (Bit 1) to be set.

This field is 0 if Port Power is 0.

- **CSC: Connect Status Change (read/write clear)**

0: No change (default value).

1: Change in Current Connect Status.

Indicates a change has occurred in the port's Current Connect Status. The host controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be "setting" an already-set bit (i.e., the bit will remain set). Software sets this bit to 0 by writing a 1 to it.

This field is 0 if Port Power is 0.

- **PED: Port Enabled/Disabled (read/write)**

0: Disable (default value).

1: Enable.

Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a 1 to this field. The host controller will only set this bit to 1 when the reset sequence determines that the attached device is a high-speed device.

Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events.

When the port is disabled (0b), downstream propagation of data is blocked on this port, except for reset.

This field is 0 if Port Power is 0.

- **PEDC: Port Enable/Disable Change (read/write clear)**

0: No change (default value).

1: Port enabled/disabled status has changed.

For the root hub, this bit gets set to 1 only when a port is disabled due to the appropriate conditions existing at the EOF2 point (refer to Chapter 11 of the USB Specification for the definition of a Port Error). Software clears this bit by writing a 1 to it.

This field is 0 if Port Power is 0.

- **OCA: Over-current Active (read-only)**

0: This port does not have an over-current condition (default value).

1: This port currently has an over-current condition.

This bit will automatically transition from 1 to 0 when the over current condition is removed.

- **OCC: Over-current Change (read/write clear)**

0: Default value.

1: This bit gets set to 1 when there is a change to Over-current Active.

Software clears this bit by writing 1 to this bit position.

- **FPR: Force Port Resume (read/write)**

0: No resume (K-state) detected/driven on port (default value).

1: Resume detected/driven on port.

This functionality defined for manipulating this bit depends on the value of the Suspend bit. For example, if the port is not suspended (Suspend and Enabled bits are set to 1) and software transitions this bit to 1, then the effects on the bus are undefined.

Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to 1 if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to 1 because a J-to-K transition is detected, the Port Change Detect bit in the UHPHS_USBSTS register is also set to 1. If software sets this bit to 1, the host controller must not set the Port Change Detect bit.

Note that when the EHCI controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains set to 1. Software must appropriately time the Resume and set this bit to 0 when the appropriate amount of time has elapsed. Writing a 0 (from 1) causes the port to return to High-Speed mode (forcing the bus below the port into a high-speed idle).

This bit will remain set to 1 until the port has switched to the high-speed idle. The host controller must complete this transition within 2 milliseconds of software setting this bit to 0.

This field is 0 if Port Power is 0.

- **SUS: Suspend (read/write)**

0: Port not in suspend state (default value).

1: Port in suspend state.

Port Enabled Bit and Suspend bit of this register define the port states as follows:

Bits [Port Enabled, Suspend]	Port State
0X	Disable
10	Enable
11	Suspend

When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB.

A write of 0 to this bit is ignored by the host controller. The host controller will unconditionally set this bit to 0 when:

- Software sets the Force Port Resume bit to 0 (from 1).
- Software sets the Port Reset bit to 1 (from 0).

If host software sets this bit to 1 when the port is not enabled (i.e., Port Enabled bit set to 0), the results are undefined.

This field is 0 if Port Power is set to 0.

- **PR: Port Reset (read/write)**

0: Port is not in Reset (default value).

1: Port is in Reset.

When software writes a 1 to this bit (from 0), the bus reset sequence as defined in the USB Specification Revision 2.0 is started. Software writes a 0 to this bit to terminate the bus reset sequence. Software must keep this bit set to 1 long enough to ensure the reset sequence, as specified in the USB Specification Revision 2.0, completes.

Note: When software writes this bit to 1, it must also write 0 to the Port Enable bit.

When software writes a 0 to this bit, there may be a delay before the bit status changes to 0. The bit status will not read as 0 until after the reset has completed. If the port is in High-Speed mode after reset is complete, the host controller will automatically enable this port (e.g., set the Port Enable bit to 1). A host controller must terminate the reset and stabilize the state of the port within 2 milliseconds of software transitioning this bit from 1 to 0. For example: if the port detects that the attached device is high-speed during reset, then the host controller must have the port in the enabled state within 2 ms of software writing this bit to 0.

The HCHalted bit in the UHPHS_USBSTS register should be set to 0 before software attempts to use this bit. The host controller may hold Port Reset asserted to 1 when the HCHalted bit is 1.

This field is 0 if Port Power is 0.

- **LS: Line Status (read-only)**

These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. These bits are used for detection of low-speed USB devices prior to the port reset and enable sequence. This field is valid only when the port enable bit is 0 and the current connect status bit is set to 1.

Bits are encoded as follows:

Value	USB State	Interpretation
00b	SE0	Not a low-speed device, perform EHCI reset
10b	J-state	Not a low-speed device, perform EHCI reset
01b	K-state	Low-speed device, release ownership of port
11b	Undefined	Not a low-speed device, perform EHCI reset

This value of this field is undefined if Port Power is 0.

- **PP: Port Power (read/write or read-only)**

The function of this bit depends on the value of the Port Power Control (PPC) field in the UHPHS_HCSPARAMS register. The behavior is as follows:

PPC	PP	Operation
0b	1b	Read-only. Host controller does not have port power control switches. Each port is hard-wired to power.
1b	1b/0b	Read/write. Host controller has port power control switches. This bit represents the current setting of the switch (0 = off, 1 = on). When power is not available on a port (i.e., PP at 0), the port is non-functional and will not report attaches, detaches, etc.

When an over-current condition is detected on a powered port and PPC is set to 1, the PP bit in each affected port may be transitioned by the host controller from 1 to 0 (removing power from the port).

- **PO: Port Owner (read/write)**

0: This bit unconditionally goes to a 0 when the Configured bit in the UHPHS_CONFIGFLAG register makes a 0 to 1 transition.

1: This bit unconditionally goes to 1 whenever the Configured bit is 0 (default value).

System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes 1 to this bit when the attached device is not a high-speed device. A 1 in this bit means that a companion host controller owns and controls the port.

- **PIC: Port Indicator Control (read/write)**

00b: Default value.

Writing to these bits has no effect if the P_INDICATOR bit in the UHPHS_HCSPARAMS register is set to 0. If the P_INDICATOR bit is set to 1, then the bits are encoded as follows:

Value	Meaning
00b	Port indicators are off
01b	Amber
10b	Green
11b	Undefined

Refer to the USB Specification Revision 2.0 for a description on how these bits are to be used.

This field is 0 if Port Power is 0.

- **PTC: Port Test Control (read/write)**

0000b: Default value.

When this field is set to 0, the port is NOT operating in a test mode. A non-zero value indicates that it is operating in test mode and the specific test mode is indicated by the specific value.

Test mode bits are encoded as follows (0110b - 1111b are reserved):

Value	Test Mode
0000b	Test mode not enabled
0001b	Test J_STATE
0010b	Test K_STATE
0011b	Test SE0_NAK
0100b	Test Packet
0101b	Test FORCE_ENABLE

Refer to the USB Specification Revision 2.0, Chapter 7, for details on each test mode.

- **WKCNNT_E: Wake on Connect Enable (read/write)**

0: Default value.

Writing this bit to 1 enables the port to be sensitive to device connects as wakeup events.

This field is 0 if Port Power is 0.

- **WKDSCNNT_E: Wake on Disconnect Enable (read/write)**

0: Default value.

Writing this bit to 1 enables the port to be sensitive to device disconnects as wakeup events.

This field is 0 if Port Power is 0.

- **WKOC_E: Wake on Over-current Enable (read/write)**

0: Default value.

Writing this bit to 1 enables the port to be sensitive to over-current conditions as wakeup events.

This field is 0 if Port Power is 0.

35.7.13 EHCI: REG00 - Programmable Microframe Base Value

Name: UHPHS_INSNREG00

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	Debug			

15	14	13	12	11	10	9	8
Debug		MFC_8		MFC_16			

7	6	5	4	3	2	1	0
MFC_16							En

The Programmable Microframe Base Value is used to change the microframe length value (default is microframe SOF = 125 µs) in order to reduce simulation time.

- **En: Enable this Register**

0: Register disabled (default value).

1: Register enabled.

Note: Do not enable this register for the gate-level netlist.

- **MFC_16: Microframe Counter with Word Byte Interface**

This value is used as the 1-microframe counter with 16-bit interface.

- **MFC_8: Microframe Counter with Byte Interface**

This value is used as the 1-microframe counter with 8-bit interface.

- **Debug: Debug Purposes**

This field is used for debug purposes only.

In Heterogeneous mode, if the per port clock gets out of sync (but still within the ppm limits) of the phy_clk, then the per port SOF counter needs some correction relative to the global SOF counter. The RTL corrects itself if this happens.

This field controls the SOF correction, in case some debugging is required for the correction.

If bit 14 is set to 1, then it enables the RTL to use the value in bits 19:15 to perform the correction.

In normal operating mode, these bits should not be written.

Note: The “value” in bits [31:1] must be programmed as follows: (value + 32/64) * Clock Period = microframe timer duration
Factor 32 is used for a 16-bit interface and factor 64 is used for an 8-bit interface. For example, for the full (125 µs) microframe duration:
- In 8-bit, 60 MHz mode, the value is h1D0C (=7436), so (7436 + 64) * 16.67 ns = 125 µs
- In 16-bit, 30 MHz mode, the value is hE86 (=3718), so (3718 + 32) * 33.33 ns = 125 µs
For a 50 µs microframe duration:
- In 8-bit, 60 MHz mode, the value is hB77 (=2395), so (2395 + 64) * 16.67 ns = 50 µs
- In 16-bit, 30 MHz mode, the value is h5BC (=1468), so (1468 + 32) * 33.33 ns = 50 µs

35.7.14 EHCI: REG01 - Programmable Packet Buffer OUT/IN Thresholds

Name: UHPHS_INSNREG01

Access: Read/Write

31	30	29	28	27	26	25	24
Out_Threshold							

23	22	21	20	19	18	17	16
Out_Threshold							

15	14	13	12	11	10	9	8
In_Threshold							

7	6	5	4	3	2	1	0
In_Threshold							

Programmable Packet Buffer OUT/IN thresholds (in CONFIG1 mode only, not applicable in Config2 mode).

The value specified here is the number of DWORDs (32-bit entries).

- **In_Threshold: Amount of Data Available in the IN Packet Buffer**

The IN threshold is used to start the memory transfer as soon as the IN threshold amount of data is available in the Packet Buffer. It is also used to disconnect the data write, if the threshold amount of data is not available in the Packet Buffer.

- **Out_Threshold: Amount of Data Available in the OUT Packet Buffer**

The OUT threshold is used to start the USB transfer as soon as the OUT threshold amount of data is fetched from system memory. It is also used to disconnect the data fetch, if the threshold amount of space is not available in the Packet Buffer.

The minimum OUT and IN threshold amount that can be programmed through INSN registers is 16 bytes.

For INCRX configurations, the minimum threshold amount that can be programmed is the highest possible INCRX burst value. For example, if the value of the strap signals {ss_ena_incr16_i, ss_ena_incr8_i, ss_ena_incr4_i} is 3'b011 (for example, INCR16 burst is disabled, INCR8/INCR4 bursts are enabled), then the minimum OUT and IN threshold values should be 32 bytes (8 DWords).

OUT and IN threshold values can be equal to the packet buffer depth only when one of the following conditions is met:

- The packet buffer depth is equal to 512 bytes and isochronous/interrupt transactions are not initiated by the host controller.
- The packet buffer depth is equal to 1024 bytes.

The threshold default value depends on one of the following packet buffer configurations:

- 1024 bytes depth, 256 bytes IN and OUT thresholds
- 512 bytes depth, 128 bytes IN and OUT thresholds
- 256 bytes depth, 64 bytes IN and OUT thresholds
- 128 bytes depth, 64 bytes IN and OUT thresholds
- 64 bytes depth, 60 bytes IN and OUT thresholds

For INCRX configurations, the Break Memory Transfer bit is always enabled.

Depending on the different packet buffer settings, not all MSB bits are used.

35.7.15 EHCI: REG02 - Programmable Packet Buffer Depth

Name: UHPHS_INSNREG02

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	Dwords			

7	6	5	4	3	2	1	0
Dwords							

Programmable Packet Buffer Depth (in CONFIG1 mode only, not applicable in Config2 mode).

The value specified here is the number of DWORDs (32-bit entries).

- **Dwords: Number of Entries**

For a maximum 256 entries for 1-Kbyte packet buffer, bits [8:0] are sufficient.

35.7.16 EHCI: REG03

Name: UHPHS_INSNREG03

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	EN_CK256	Ignore_LS	Tx_Tx			Per_Frame	TA_Offset

7	6	5	4	3	2	1	0
TA_Offset							Break_Mem

The default value for INSNREG03[0] depends on the host core configuration. So, if INCRx support is enabled, this bit is 1 after reset. Otherwise, it should stay at 0.

- **Break_Mem: Break Memory Transfer (in CONFIG1 mode only, not applicable in CONFIG2 mode)**

0: Disables this function.

1: Enables this function.

Used in conjunction with INSNREG01 to enable breaking memory transactions into chunks once the OUT/IN threshold value is reached.

- **TA_Offset: Time-Available Offset**

This value indicates the additional number of bytes to be accommodated for the time-available calculation. The USB traffic on the bus can be started only when sufficient time is available to complete the packet within the EOF1 point.

Refer to the USB 2.0 specification for details of the EOF1 point. This time-available calculation is done in the hardware, and can be further offset by programming a value in this location.

Note: Time-available calculation is added for future flexibility. The application is not required to program this field by default.

- **Per_Frame: Periodic Frame List Fetch**

In CONFIG1 mode only (“EHCI Descriptor/Data Prefetching” is disabled in core configuration), setting this bit forces the host controller to fetch the periodic frame list in every microframe of a frame. If not set, then the periodic frame list is fetched only in microframe 0 of every frame.

The default is 0 (not set). This bit can be changed only during core initialization and should not be changed afterwards.

- **Tx_Tx: Tx-Tx Turnaround Delay Add-on**

This field specifies the extra delays in phy_clks to be added to the “Transmit to Transmit turnaround delay” value maintained in the core. The default value of this register field is 0. This default value of 0 is sufficient for most PHYs. But for some PHYs which enter wait states during the token packet, it may be required to program a value greater than 0 to meet the transmit-to-transmit minimum turnaround time.

It is recommended to use default value 0 and to change it only if there is an issue with minimum transmit-to-transmit turnaround time.

This value should be programmed during core initialization and should not be changed afterwards.

- **Ignore_LS: Ignore Linestate During TestSE0 Nak**

When set to 1 (default), the core ignores the linestate checking when transmitting SOF in SE0_NAK Test mode.

When set to 0, the port state machine disables the port if it does not find the linestate to be in SE0 when transmitting SOF during the SE0_NAK test.

While performing impedance measurement during the SE0_NAK test, the linestate could go to non SE0 forcing the core to disable the port. This bit is used to control the port behavior during this operation.

- **EN_CK256: Enable 256 Clock Checking**

This bit controls the End of Resume sequence of the EHCI host controller.

By default, the value of this bit is 0 and during the End of Resume sequence, the host controller waits for SE0 on the linestate before switching the PHY to High-Speed.

When set to 1, during the End of Resume sequence, the controller waits for SE0 or 256 clocks before switching the PHY to High-Speed.

Setting this bit to 1 enables the 256-clock check. Some of the UTMI PHYs do not present SE0 on the linestate during the End of Resume sequence. For such PHYs, this bit should be set, so that the core does not wait forever for SE0.

This bit should be set only during initialization.

35.7.17 EHCI: REG04

Name: UHPHS_INSNREG04

Access: Read/Write

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
–	–	EN_AutoFunc	NAK_RF	–	SDPE_TIME	HCCPARAMS_BW	HCSPARAMS_W

Bits [2:0] are used for debug purposes. Bits [(5+UHC2_N_PORTS):4] are functional bits where UHC2_N_PORTS indicates the number of physical USB ports.

- **HCSPARAMS_W: HCSPARAMS Write**

When set, the HCSPARAMS register becomes writable. Upon system reset, this bit is 0.

- **HCCPARAMS_BW: HCCPARAMS Bits Write**

When set, the HCCPARAMS register's bits 17, 15:4, and 2:0 become writable. Upon system reset, these bits are 0.

- **SDPE_TIME: Scales Down Port Enumeration Time**

When set, Scales Down Port Enumeration Time is enabled. Reset value is 1’b0.

Note: This bit can be used for both RTL and Gate level simulations.

- **NAK_RF: NAK Reload Fix (Read/Write)**

0: Enables this function.

1: Disables this function

Incorrect NAK reload transition at the end of a microframe for backward compatibility with Release 2.40c. For more information, refer to the *USB 2.0 Host-AHB Release Notes*. Reset value is 1’b0.

- **EN_AutoFunc: Enable Automatic Feature**

0: Enables the automatic feature.

The Suspend signal is deasserted (logic level 1'b1) when run/stop is reset by software, but the hchalted bit is not set yet.

1: Disables the automatic feature, which takes all ports out of suspend when software clears the run/stop bit. This is for backward compatibility.

Bit [5] has an added functionality in release 2.80a and later. For systems where the host is halted without waking up all ports out of suspend, the port can remain suspended because the PHYCLK is not running when the halt is programmed. To avoid this, the DWC H20AHB host core automatically pulls ports out of suspend when the host is halted by software.

This bit is used to disable this automatic function.

Reset value is 0.

35.7.18 EHCI: REG05 - UTMI Configuration

Name: UHPHS_INSNREG05

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	VBusy	VPort

15	14	13	12	11	10	9	8
VPort			VControlLoadM	VControl			

7	6	5	4	3	2	1	0
VStatus							

Control and Status Register, used to read the UTMI registers from the following signals:

- **VStatus: Vendor Status (Software RO)**
- **VControl: Vendor Control (Software R/W)**
- **VControlLoadM: Vendor Control Load Microframe**

0: Load.

1: NOP (software R/W)

- **VPort: Vendor Port (Software R/W)**

Valid values range from 1 to 15 depending on coreConsultant configuration.

For example, if the number of ports is 3, then software should only write values 1, 2, and 3 to this field and not any other values in the range, that is, 0 or 4 to 15. For example, if the software writes value 4 to VPort, from that write onwards, any write to this register is ignored and the read value will always be 4.

- **VBusy: Vendor Busy (Software RO)**

Hardware indicator that a write to this register has occurred and the hardware is currently processing the operation defined by the data written. When processing is finished, this bit is cleared.

35.7.19 EHCI: REG06 - AHB Error Status

Name: UHPHS_INSNREG06

Access: Read/Write

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

Control and Status Register, used to read the UTMI registers from the following signals:

- **Nb_Success_Burst: Number of Successful Bursts (read-only)**(1)

Number of successfully completed beats in the current burst before the AHB error occurred.

- **Nb_Burst: Number of Bursts (read-only)**(1)

Number of beats expected in the burst at which the AHB error occurred. Valid values are 0 to 16.

5'b10001–5b11111: Reserved

5'b00000–5b10000: Valid

- **HBURST: Burst Value (read-only)**(1)

Value of the control phase at which the AHB error occurred.

Note: 1. This field applies to AHB INCRX-enabled configurations only.

- **AHB_ERR: AHB Error**

AHB Error Captured Indicator that an AHB error was encountered and values were captured. To clear this field the application must write a 0 to it.

EHCI:

– When no error, 0 is written to INSNREG06[8:4].
– When INCR4 and an error occur, 4 is written to INSNREG06[8:4].
– When INCR8 and an error occur, 8 is written to INSNREG06[8:4].
– When INCR16 and an error occur, 16 is written to INSNREG06[8:4].
– Other values except 4, 8, and 16 are not written to INSNREG06[8:4].

OHCI:

– When no error, 0 is written to INSNREG06[8:4].
– When INCR4 and error occur, 4 is written to INSNREG06[8:4].
– Other values except 4 are not written to INSNREG06[8:4].

35.7.20 EHCI: REG07 - AHB Master Error Address

Name: UHPHS_INSNREG07

Access: Read Only

31	30	29	28	27	26	25	24
AHB_ADDR							

23	22	21	20	19	18	17	16
AHB_ADDR							

15	14	13	12	11	10	9	8
AHB_ADDR							

7	6	5	4	3	2	1	0
AHB_ADDR							

- **AHB_ADDR: AHB Address (read only)**

AHB address of the control phase at which the AHB error occurred.

36. Ethernet MAC (GMAC)

36.1 Description

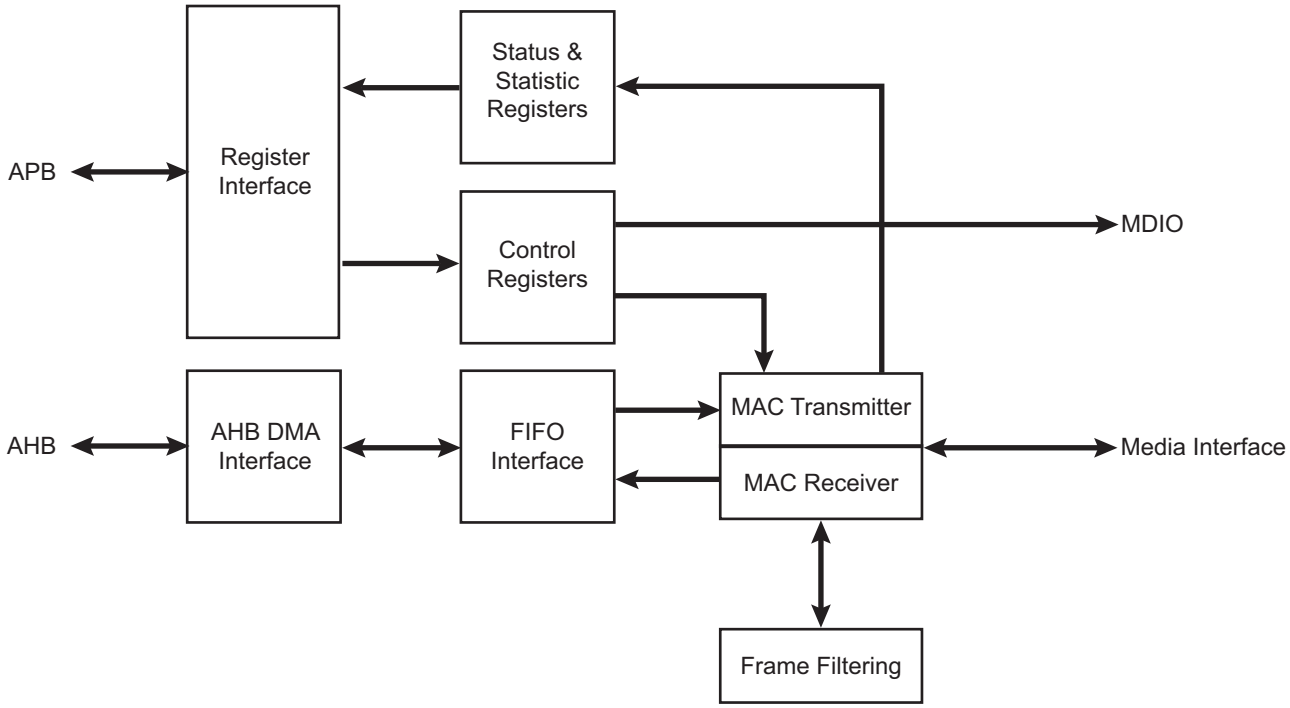
The Ethernet MAC (GMAC) module implements a 10/100 Mbps Ethernet MAC compatible with the IEEE 802.3 standard. The GMAC can operate in either half or full duplex mode at all supported speeds. The GMAC Network Configuration Register is used to select the speed, duplex mode and interface type (MII, RMII).

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
- 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 Time Stamp Unit (TSU)
- Support for 802.1AS Timing and Synchronization

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

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
GCRS	Carrier Sense and Data Valid	CRS	Not Used

Table 36-1. GMAC Connections in Different Modes (Continued)

Signal Name	Function	MII	RMII
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.

Table 36-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
GMAC0	G0_COL	PB5	A
GMAC0	G0_CRS	PB4	A
GMAC0	G0_MDC	PB16	A
GMAC0	G0_MDIO	PB17	A
GMAC0	G0_RXCK	PB1	A
GMAC0	G0_RXDV	PB6	A
GMAC0	G0_RXER	PB7	A
GMAC0	G0_RX0	PB8	A
GMAC0	G0_RX1	PB9	A
GMAC0	G0_RX2	PB10	A
GMAC0	G0_RX3	PB11	A
GMAC0	G0_TXCK	PB0	A
GMAC0	G0_TXEN	PB2	A
GMAC0	G0_TXER	PB3	A
GMAC0	G0_TX0	PB12	A
GMAC0	G0_TX1	PB13	A
GMAC0	G0_TX2	PB14	A
GMAC0	G0_TX3	PB15	A
GMAC1	G1_COL	PA9	B
GMAC1	G1_CRS	PA6	B
GMAC1	G1_MDC	PA22	B
GMAC1	G1_MDIO	PA23	B
GMAC1	G1_RXCK	PA3	B
GMAC1	G1_RXDV	PA10	B
GMAC1	G1_RXER	PA11	B

Table 36-2. I/O Lines (Continued)

GMAC1	G1_RX0	PA12	B
GMAC1	G1_RX1	PA13	B
GMAC1	G1_RX2	PA18	B
GMAC1	G1_RX3	PA19	B
GMAC1	G1_TXCK	PA2	B
GMAC1	G1_TXEN	PA4	B
GMAC1	G1_TXER	PA5	B
GMAC1	G1_TX0	PA14	B
GMAC1	G1_TX1	PA15	B
GMAC1	G1_TX2	PA20	B
GMAC1	G1_TX3	PA21	B

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 1 interrupt sources. Refer to Section 8.2 “Peripheral Identifiers” for the interrupt numbers for GMAC priority queues.

Table 36-3. Peripheral IDs

Instance	ID
GMAC0	54
GMAC1	55

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 Time Stamp Unit

The 1588 time stamp 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 performs six types of operations on the AHB bus. The order of priority of these operations is:

1. Receive buffer manager write
2. Receive buffer manager read
3. Transmit buffer manager write
4. Transmit buffer manager read
5. Receive data DMA write
6. Transmit data DMA read

36.6.3.1 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 Table 36-4 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 36-4. Receive Buffer Descriptor Entry

Bit	Function
Word 0	
31:2	Address of beginning of buffer

Table 36-4. Receive Buffer Descriptor Entry (Continued)

Bit	Function
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.
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).

Table 36-4. Receive Buffer Descriptor Entry (Continued)

Bit	Function
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.

To function properly, a 10/100 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.

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.

36.6.3.2 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 FIFO-based DMA configured with a 32-bit data path the address of the buffer is a byte address.

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 Table 36-5.

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.

The DMA transmission will automatically restart from the first buffer of the frame.

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 36-5. 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	Transmit underrun—occurs when the start of packet data has been written into the FIFO and either HRESP is not OK, or the transmit data could not be fetched in time, or when buffers are exhausted.
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).
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

36.6.3.3 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.

36.6.4 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.

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.

If the back pressure bit is set in the Network Control register, or if the HDFC configuration bit is set in the GMAC_UR register (10M or 100M half duplex mode), the transmit block transmits 64 bits of data, which can consist of 16 nibbles of 1011 or in bit rate mode 64 1s, whenever it sees an incoming frame to force a collision. This provides a way of implementing flow control in half duplex mode.

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.

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 Table 36-4 “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.

36.6.7 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 36-6. 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 Wake on LAN Support

The receive block supports Wake on LAN by detecting the following events on incoming receive frames:

- Magic packet
- Address Resolution Protocol (ARP) request to the device IP address
- Specific address 1 filter match
- Multicast hash filter match

These events can be individually enabled through bits [19:16] of the Wake on LAN register. Also, for Wake on LAN detection to occur, receive enable must be set in the Network Control register, however a receive buffer does not have to be available.

In case of an ARP request, specific address 1 or multicast filter events will occur even if the frame is errored. For magic packet events, the frame must be correctly formed and error free.

A magic packet event is detected if all of the following are true:

- Magic packet events are enabled through bit 16 of the Wake on LAN register
- The frame's destination address matches specific address 1
- The frame is correctly formed with no errors
- The frame contains at least 6 bytes of 0xFF for synchronization
- There are 16 repetitions of the contents of Specific Address 1 register immediately following the synchronization

An ARP request event is detected if all of the following are true:

- ARP request events are enabled through bit 17 of the Wake on LAN register
- Broadcasts are allowed by bit 5 in the Network Configuration register
- The frame has a broadcast destination address (bytes 1 to 6)
- The frame has a type ID field of 0x0806 (bytes 13 and 14)
- The frame has an ARP operation field of 0x0001 (bytes 21 and 22)
- The least significant 16 bits of the frame's ARP target protocol address (bytes 41 and 42) match the value programmed in bits[15:0] of the Wake on LAN register

The decoding of the ARP fields adjusts automatically if a VLAN tag is detected within the frame. The reserved value of 0x0000 for the Wake on LAN target address value will not cause an ARP request event, even if matched by the frame.

A specific address 1 filter match event will occur if all of the following are true:

- Specific address 1 events are enabled through bit 18 of the Wake on LAN register
- The frame's destination address matches the value programmed in the Specific Address 1 registers

A multicast filter match event will occur if all of the following are true:

- Multicast hash events are enabled through bit 19 of the Wake on LAN register
- Multicast hash filtering is enabled through bit 6 of the Network Configuration register
- The frame destination address matches against the multicast hash filter
- The frame destination address is not a broadcast

36.6.14 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 36-7. 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 36-8. 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)	—
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 36-9. 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 36-10. 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

Table 36-10. Example of Pdelay_Req Frame in 1588 Version 2 (UDP/IPv4) Format (Continued)

Frame Segment	Value
Version PTP (Octet 43)	02

Table 36-11. 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 36-12. 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)	—
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 36-13. 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 36-14. 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

36.6.15 Time Stamp 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. 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.

36.6.16 MAC 802.3 Pause Frame Support

Note: Refer to 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 36-15. 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.

36.6.16.1802.3 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.

36.6.16.2802.3 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.

36.6.17 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 36-16. 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.

36.6.17.1PFC 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.

36.6.17.2PFC 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.

36.6.18 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 GMII 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.
- The sleep, quiet and refresh periods are defined in Table 78-2 of 802.3az. For 1000BASE-X the sleep period is 20 microseconds, the quiet period 2.5 milliseconds and the refresh period 20 microseconds.
- 1000BASE-X is required to go quiet after sleep is signaled. The easiest way to do this is to write to a control register to disable transmit in the SerDes.
- SGMII and XFI are not part of 802.3az and should not go quiet after sleep is signaled.
- 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.

36.6.19 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 LPI bit in the Network Control register.
2. If connected to 1000BASE-T PHY using SGMII or RGMII, there is nothing more to do.
3. If connected to a backplane using a 1000BASE-KX PHY, use firmware to periodically disable the SerDes transmit path. (Write to bit 1.160.0 for 1000BASE-KX.)
4. Wake up by clearing the LPI bit in the Network Control register.

For the receive path:

1. Wait for an interrupt to indicate that LPI has been received.
2. Disable relevant parts of the receive path if desired but keep the PCS and SerDes active.
3. Wait for an interrupt to indicate that regular idle has been received and then 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 10/100 Mbps speed bit in the Network Configuration register is used to select between 10 Mbps and 100 Mbps.

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. Refer to “GMAC Network Control Register” and “GMAC Network Configuration Register”.

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.

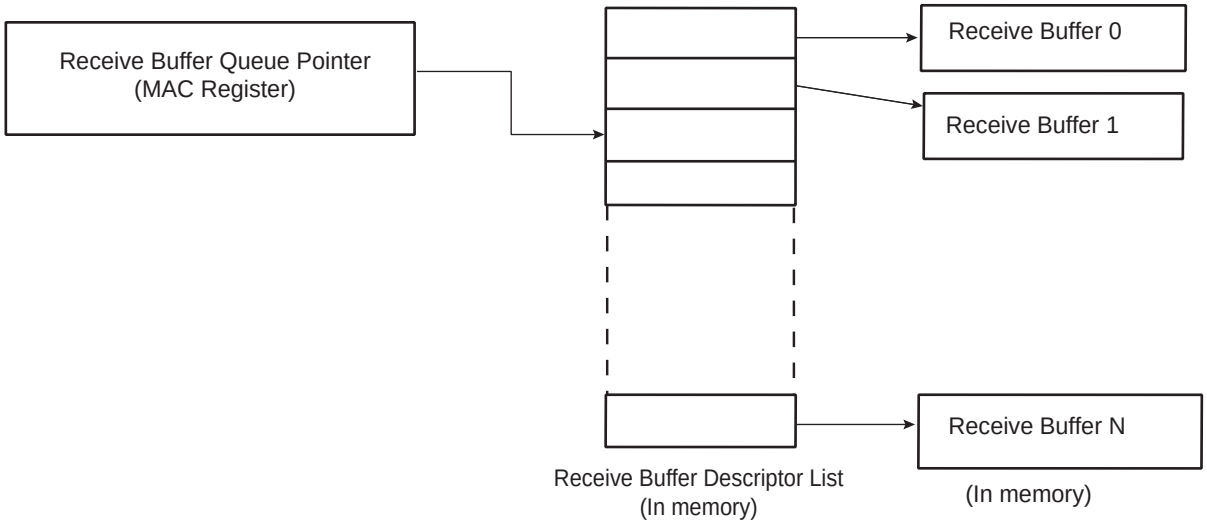
Note: These writes to the Network Control register cannot be combined in any way.

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 Table 36-4 “Receive Buffer Descriptor Entry”.

The Receive Buffer Queue Pointer register points to this data structure.

Figure 36-2. 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 Table 36-5 “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.

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 MDIO pin and the LSB updated from the MDIO pin with each Management Data Clock (MDC) cycle. This causes the transmission of a PHY management frame on MDIO. Refer to 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 a single interrupt. 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 (refer to Table 36-4 “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 Section 36.8.45 “GMAC Octets Transmitted Low Register” and ending with Section 36.8.89 “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 Ethernet MAC (GMAC) User Interface

Table 36-17. Register Mapping

Offset[1] [2]	Register	Name	Access	Reset
0x000	Network Control Register	GMAC_NCR	Read/Write	0x0000_0000
0x004	Network Configuration Register	GMAC_NCFGR	Read/Write	0x0008_0000
0x008	Network Status Register	GMAC_NSR	Read-only	0b01x0
0x00C	User Register	GMAC_UR	Read/Write	0x0000_0000
0x010	DMA Configuration Register	GMAC_DCFGR	Read/Write	0x0002_0004
0x014	Transmit Status Register	GMAC_TSR	Read/Write	0x0000_0000
0x018	Receive Buffer Queue Base Address Register	GMAC_RBQB	Read/Write	0x0000_0000
0x01C	Transmit Buffer Queue Base Address Register	GMAC_TBQB	Read/Write	0x0000_0000
0x020	Receive Status Register	GMAC_RSR	Read/Write	0x0000_0000
0x024	Interrupt Status Register	GMAC_ISR	Read-only	0x0000_0000
0x028	Interrupt Enable Register	GMAC_IER	Write-only	–
0x02C	Interrupt Disable Register	GMAC_IDR	Write-only	–
0x030	Interrupt Mask Register	GMAC_IMR	Read/Write	0x07FF_FFFF
0x034	PHY Maintenance Register	GMAC_MAN	Read/Write	0x0000_0000
0x038	Received Pause Quantum Register	GMAC_RPQ	Read-only	0x0000_0000
0x03C	Transmit Pause Quantum Register	GMAC_TPQ	Read/Write	0x0000_FFFF
0x048	RX Jumbo Frame Max Length Register	GMAC_RJFML	Read/Write	0x0000_3FFF
0x040–0x07C	Reserved	–	–	–
0x080	Hash Register Bottom	GMAC_HRB	Read/Write	0x0000_0000
0x084	Hash Register Top	GMAC_HRT	Read/Write	0x0000_0000
0x088	Specific Address 1 Bottom Register	GMAC_SAB1	Read/Write	0x0000_0000
0x08C	Specific Address 1 Top Register	GMAC_SAT1	Read/Write	0x0000_0000
0x090	Specific Address 2 Bottom Register	GMAC_SAB2	Read/Write	0x0000_0000
0x094	Specific Address 2 Top Register	GMAC_SAT2	Read/Write	0x0000_0000
0x098	Specific Address 3 Bottom Register	GMAC_SAB3	Read/Write	0x0000_0000
0x09C	Specific Address 3 Top Register	GMAC_SAT3	Read/Write	0x0000_0000
0x0A0	Specific Address 4 Bottom Register	GMAC_SAB4	Read/Write	0x0000_0000
0x0A4	Specific Address 4 Top Register	GMAC_SAT4	Read/Write	0x0000_0000
0x0A8	Type ID Match 1 Register	GMAC_TIDM1	Read/Write	0x0000_0000
0x0AC	Type ID Match 2 Register	GMAC_TIDM2	Read/Write	0x0000_0000
0x0B0	Type ID Match 3 Register	GMAC_TIDM3	Read/Write	0x0000_0000
0x0B4	Type ID Match 4 Register	GMAC_TIDM4	Read/Write	0x0000_0000
0x0B8	Wake on LAN Register	GMAC_WOL	Read/Write	0x0000_0000
0x0BC	IPG Stretch Register	GMAC_IPGS	Read/Write	0x0000_0000
0x0C0	Stacked VLAN Register	GMAC_SVLAN	Read/Write	0x0000_0000

Table 36-17. Register Mapping (Continued)

Offset[1] [2]	Register	Name	Access	Reset
0x0C4	Transmit PFC Pause Register	GMAC_TPFCP	Read/Write	0x0000_0000
0x0C8	Specific Address 1 Mask Bottom Register	GMAC_SAMB1	Read/Write	0x0000_0000
0x0CC	Specific Address 1 Mask Top Register	GMAC_SAMT1	Read/Write	0x0000_0000
0x0D0–0x0D8	Reserved	–	–	–
0x0DC	1588 Timer Nanosecond Comparison Register	GMAC_NSC	Read/Write	0x0000_0000
0x0E0	1588 Timer Second Comparison Low Register	GMAC_SCL	Read/Write	0x0000_0000
0x0E4	1588 Timer Second Comparison High Register	GMAC_SCH	Read/Write	0x0000_0000
0x0E8	PTP Event Frame Transmitted Seconds High Register	GMAC_EFTSH	Read-only	0x0000_0000
0x0EC	PTP Event Frame Received Seconds High Register	GMAC_EFRSH	Read-only	0x0000_0000
0x0F0	PTP Peer Event Frame Transmitted Seconds High Register	GMAC_PEFTSH	Read-only	0x0000_0000
0x0F4	PTP Peer Event Frame Received Seconds High Register	GMAC_PEFRSH	Read-only	0x0000_0000
0x0F8–0x0FC	Reserved	–	–	–
0x100	Octets Transmitted Low Register	GMAC_OTLO	Read-only	0x0000_0000
0x104	Octets Transmitted High Register	GMAC_OTHI	Read-only	0x0000_0000
0x108	Frames Transmitted Register	GMAC_FT	Read-only	0x0000_0000
0x10C	Broadcast Frames Transmitted Register	GMAC_BCFT	Read-only	0x0000_0000
0x110	Multicast Frames Transmitted Register	GMAC_MFT	Read-only	0x0000_0000
0x114	Pause Frames Transmitted Register	GMAC_PFT	Read-only	0x0000_0000
0x118	64 Byte Frames Transmitted Register	GMAC_BFT64	Read-only	0x0000_0000
0x11C	65 to 127 Byte Frames Transmitted Register	GMAC_TBFT127	Read-only	0x0000_0000
0x120	128 to 255 Byte Frames Transmitted Register	GMAC_TBFT255	Read-only	0x0000_0000
0x124	256 to 511 Byte Frames Transmitted Register	GMAC_TBFT511	Read-only	0x0000_0000
0x128	512 to 1023 Byte Frames Transmitted Register	GMAC_TBFT1023	Read-only	0x0000_0000
0x12C	1024 to 1518 Byte Frames Transmitted Register	GMAC_TBFT1518	Read-only	0x0000_0000
0x130	Greater Than 1518 Byte Frames Transmitted Register	GMAC_GTBFT1518	Read-only	0x0000_0000
0x134	Transmit Underruns Register	GMAC_TUR	Read-only	0x0000_0000
0x138	Single Collision Frames Register	GMAC_SCF	Read-only	0x0000_0000
0x13C	Multiple Collision Frames Register	GMAC_MCF	Read-only	0x0000_0000
0x140	Excessive Collisions Register	GMAC_EC	Read-only	0x0000_0000
0x144	Late Collisions Register	GMAC_LC	Read-only	0x0000_0000
0x148	Deferred Transmission Frames Register	GMAC_DTF	Read-only	0x0000_0000
0x14C	Carrier Sense Errors Register	GMAC_CSE	Read-only	0x0000_0000
0x150	Octets Received Low Received Register	GMAC_ORLO	Read-only	0x0000_0000
0x154	Octets Received High Received Register	GMAC_ORHI	Read-only	0x0000_0000

Table 36-17. Register Mapping (Continued)

Offset[1] [2]	Register	Name	Access	Reset
0x158	Frames Received Register	GMAC_FR	Read-only	0x0000_0000
0x15C	Broadcast Frames Received Register	GMAC_BCFR	Read-only	0x0000_0000
0x160	Multicast Frames Received Register	GMAC_MFR	Read-only	0x0000_0000
0x164	Pause Frames Received Register	GMAC_PFR	Read-only	0x0000_0000
0x168	64 Byte Frames Received Register	GMAC_BFR64	Read-only	0x0000_0000
0x16C	65 to 127 Byte Frames Received Register	GMAC_TBFR127	Read-only	0x0000_0000
0x170	128 to 255 Byte Frames Received Register	GMAC_TBFR255	Read-only	0x0000_0000
0x174	256 to 511 Byte Frames Received Register	GMAC_TBFR511	Read-only	0x0000_0000
0x178	512 to 1023 Byte Frames Received Register	GMAC_TBFR1023	Read-only	0x0000_0000
0x17C	1024 to 1518 Byte Frames Received Register	GMAC_TBFR1518	Read-only	0x0000_0000
0x180	1519 to Maximum Byte Frames Received Register	GMAC_TMXBFR	Read-only	0x0000_0000
0x184	Undersize Frames Received Register	GMAC_UFR	Read-only	0x0000_0000
0x188	Oversize Frames Received Register	GMAC_OFR	Read-only	0x0000_0000
0x18C	Jabbers Received Register	GMAC_JR	Read-only	0x0000_0000
0x190	Frame Check Sequence Errors Register	GMAC_FCSE	Read-only	0x0000_0000
0x194	Length Field Frame Errors Register	GMAC_LFFE	Read-only	0x0000_0000
0x198	Receive Symbol Errors Register	GMAC_RSE	Read-only	0x0000_0000
0x19C	Alignment Errors Register	GMAC_AE	Read-only	0x0000_0000
0x1A0	Receive Resource Errors Register	GMAC_RRE	Read-only	0x0000_0000
0x1A4	Receive Overrun Register	GMAC_ROE	Read-only	0x0000_0000
0x1A8	IP Header Checksum Errors Register	GMAC_IHCE	Read-only	0x0000_0000
0x1AC	TCP Checksum Errors Register	GMAC_TCE	Read-only	0x0000_0000
0x1B0	UDP Checksum Errors Register	GMAC_UCE	Read-only	0x0000_0000
0x1B4–0x1B8	Reserved	–	–	–
0x1BC	1588 Timer Increment Sub-nanoseconds Register	GMAC_TISUBN	Read/Write	0x0000_0000
0x1C0	1588 Timer Seconds High Register	GMAC_TSH	Read/Write	0x0000_0000
0x1C4–0x1CC	Reserved	–	–	–
0x1D0	1588 Timer Seconds Low Register	GMAC_TSL	Read/Write	0x0000_0000
0x1D4	1588 Timer Nanoseconds Register	GMAC_TN	Read/Write	0x0000_0000
0x1D8	1588 Timer Adjust Register	GMAC_TA	Write-only	–
0x1DC	1588 Timer Increment Register	GMAC_TI	Read/Write	0x0000_0000
0x1E0	PTP Event Frame Transmitted Seconds Low Register	GMAC_EFTSL	Read-only	0x0000_0000
0x1E4	PTP Event Frame Transmitted Nanoseconds Register	GMAC_EFTN	Read-only	0x0000_0000
0x1E8	PTP Event Frame Received Seconds Low Register	GMAC_EFRSL	Read-only	0x0000_0000
0x1EC	PTP Event Frame Received Nanoseconds Register	GMAC_EFRN	Read-only	0x0000_0000

Table 36-17. Register Mapping (Continued)

Offset[1] [2]	Register	Name	Access	Reset
0x1F0	PTP Peer Event Frame Transmitted Seconds Low Register	GMAC_PEFTSL	Read-only	0x0000_0000
0x1F4	PTP Peer Event Frame Transmitted Nanoseconds Register	GMAC_PEFTN	Read-only	0x0000_0000
0x1F8	PTP Peer Event Frame Received Seconds Low Register	GMAC_PEFRSL	Read-only	0x0000_0000
0x1FC	PTP Peer Event Frame Received Nanoseconds Register	GMAC_PEFRN	Read-only	0x0000_0000
0x200–0x26C	Reserved	–	–	–
0x270	Received LPI Transitions	GMAC_RXLPI	Read-only	0x0000_0000
0x274	Received LPI Time	GMAC_RXLPITIME	Read-only	0x0000_0000
0x278	Transmit LPI Transitions	GMAC_TXLPI	Read-only	0x0000_0000
0x27C	Transmit LPI Time	GMAC_TXLPITIME	Read-only	0x0000_0000
0x280–0x7FC	Reserved	–	–	–

Notes: 1. If an offset is not listed in the Register Mapping, it must be considered as ‘reserved’.
2. Some register groups are not continuous in memory.

36.8.1 GMAC Network Control Register

Name: GMAC_NCR

Address: 0xF8020000 (0), 0xFC028000 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	TXLPIEN	FNP	TXPBPF	ENPBPR
15	14	13	12	11	10	9	8
SRTSM	–	–	TXZQPF	TXPF	THALT	TSTART	BP
7	6	5	4	3	2	1	0
WESTAT	INCSTAT	CLRSTAT	MPE	TXEN	RXEN	LBL	–

- **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.

- **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.

- **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.

- **MPE: Management Port Enable**

Set to one to enable the management port. When zero, forces MDIO to high impedance state and MDC low.

- **CLRSTAT: Clear Statistics Registers**

This bit is write-only. Writing a one clears the statistics registers.

- **INCSTAT: Increment Statistics Registers**

This bit is write-only. Writing a one increments all the statistics registers by one for test purposes.

- **WESTAT: Write Enable for Statistics Registers**

Setting this bit to one makes the statistics registers writable for functional test purposes.

- **BP: Back pressure**

If set in 10M or 100M half duplex mode, forces collisions on all received frames.

- **TSTART: Start Transmission**

Writing one to this bit starts transmission.

- **THALT: Transmit Halt**

Writing one to this bit halts transmission as soon as any ongoing frame transmission ends.

- **TXPF: Transmit Pause Frame**

Writing one to this bit causes a pause frame to be transmitted.

- **TXZQPF: Transmit Zero Quantum Pause Frame**

Writing one to this bit causes a pause frame with zero quantum to be transmitted.

- **SRTSM: Store Receive Time Stamp to Memory**

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.

- **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.

- **TXPBPF: Transmit PFC Priority-based Pause Frame**

Takes the values stored in the Transmit PFC Pause Register.

- **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.

- **TXLPIEN: Enable LPI Transmission**

When set, LPI (low power idle) is immediately transmitted.

36.8.2 GMAC Network Configuration Register

Name: GMAC_NCFGR

Address: 0xF8020004 (0), 0xFC028004 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	IRXER	RXBP	IPGSEN	–	IRXFCS	EFRHD	RXCOEN
23	22	21	20	19	18	17	16
DCPF	DBW		CLK			RFCS	LFERD
15	14	13	12	11	10	9	8
RXBUFO		PEN	RTY	–	–	–	MAXFS
7	6	5	4	3	2	1	0
UNIHEN	MTI HEN	NBC	CAF	JFRAME	DNVLAN	FD	SPD

- **SPD: Speed**

Set to logic one to indicate 100 Mbps operation, logic zero for 10 Mbps.

- **FD: Full Duplex**

If set to logic one, the transmit block ignores the state of collision and carrier sense and allows receive while transmitting.

- **DNVLAN: Discard Non-VLAN FRAMES**

When set only VLAN tagged frames will be passed to the address matching logic.

- **JFRAME: Jumbo Frame Size**

Set to one to enable jumbo frames up to 10240 bytes to be accepted. The default length is 10240 bytes.

- **CAF: Copy All Frames**

When set to logic one, all valid frames will be accepted.

- **NBC: No Broadcast**

When set to logic one, frames addressed to the broadcast address of all ones will not be accepted.

- **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.

- **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.

- **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.

- **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.

- **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.

- **RXBUFO: Receive Buffer Offset**

Indicates the number of bytes by which the received data is offset from the start of the receive buffer

- **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.

- **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.

- **CLK: 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)

- **DBW: Data Bus Width**

Should always be written to ‘0’.

- **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.

- **RXCOEN: Receive Checksum Offload Enable**

When set, the receive checksum engine is enabled. Frames with bad IP, TCP or UDP checksums are discarded.

- **EFRHD: Enable Frames Received in Half Duplex**

Enable frames to be received in half-duplex mode while transmitting.

- **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.

- **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.

- **RXBP: Receive Bad Preamble**

When set, frames with non-standard preamble are not rejected.

- **IRXER: Ignore IPG GRXER**

When set, GRXER has no effect on the GMAC's operation when GRXDV is low.

36.8.3 GMAC Network Status Register

Name: GMAC_NSR

Address: 0xF8020008 (0), 0xFC028008 (1)

Access: Read-only

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
RXLPIS	–	–	–	–	IDLE	MDIO	–

- **MDIO: MDIO Input Status**

Returns status of the MDIO pin.

- **IDLE: PHY Management Logic Idle**

The PHY management logic is idle (i.e., has completed).

- **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.

36.8.4 GMAC User Register

Name: GMAC_UR

Address: 0xF802000C (0), 0xFC02800C (1)

Access: Read/Write

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
–	–	–	–	–	–	–	RMII

- **RMII: Reduced MII Mode**

0: MII mode is selected (default).

1: RMII mode is selected.

36.8.5 GMAC DMA Configuration Register

Name: GMAC_DCFGR

Address: 0xF8020010 (0), 0xFC028010 (1)

Access: Read/Write

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

- **FBLDO: 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.

Upper bits become non-writable if the configured DMA TX and RX FIFO sizes are smaller than required to support the selected burst size.

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

- **ESMA: Endian Swap Mode Enable for Management Descriptor Accesses**

When set, selects swapped endianism for AHB transfers. When clear, selects little endian mode.

- **ESPA: Endian Swap Mode Enable for Packet Data Accesses**

When set, selects swapped endianism for AHB transfers. When clear, selects little endian mode.

- **DRBS: 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.

36.8.6 GMAC Transmit Status Register

Name: GMAC_TSR

Address: 0xF8020014 (0), 0xFC028014 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	HRESP
7	6	5	4	3	2	1	0
–	UND	TXCOMP	TFC	TXGO	RLE	COL	UBR

- **UBR: Used Bit Read**

Set when a transmit buffer descriptor is read with its used bit set. Writing a one clears this bit.

- **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.

- **RLE: Retry Limit Exceeded**

Writing a one clears this bit.

- **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.

- **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).

Writing a one clears this bit.

- **TXCOMP: Transmit Complete**

Set when a frame has been transmitted. Writing a one clears this bit.

- **UND: Transmit Underrun**

This bit is set if the transmitter was forced to terminate a frame that it had already began transmitting due to further data being unavailable.

This bit is set if a transmitter status write back has not completed when another status write back is attempted.

When using the DMA interface configured for internal FIFO mode, this bit 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 a used bit was read.

Writing a one clears this bit.

- **HRESP: HRESP Not OK**

Set when the DMA block sees HRESP not OK. Writing a one clears this bit.

36.8.7 GMAC Receive Buffer Queue Base Address Register

Name: GMAC_RBQB

Address: 0xF8020018 (0), 0xFC028018 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR						–	–

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.

- **ADDR: 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

Address: 0xF802001C (0), 0xFC02801C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR						–	–

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.

- **ADDR: 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

Address: 0xF8020020 (0), 0xFC028020 (1)

Access: Read/Write

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
–	–	–	–	HNO	RXOVR	REC	BNA

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.

- **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.

- **REC: Frame Received**

One or more frames have been received and placed in memory. Writing a one clears this bit.

- **RXOVR: Receive Overrun**

This bit is set if RX FIFO is not able to store the receive frame due to a FIFO overflow, or if the receive status was not taken at the end of the frame. The buffer will be recovered if an overrun occurs. Writing a one clears this bit.

- **HNO: HRESP Not OK**

Set when the DMA block sees HRESP not OK. Writing a one clears this bit.

36.8.10 GMAC Interrupt Status Register

Name: GMAC_ISR

Address: 0xF8020024 (0), 0xFC028024 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
23	22	21	20	19	18	17	16
PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR	–	–
15	14	13	12	11	10	9	8
–	PFTR	PTZ	PFNZ	HRESP	ROVR	–	–
7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS

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.

- **MFS: Management Frame Sent**

The PHY Maintenance Register has completed its operation. Cleared on read.

- **RCOMP: Receive Complete**

A frame has been stored in memory. Cleared on read.

- **RXUBR: RX Used Bit Read**

Set when a receive buffer descriptor is read with its used bit set. Cleared on read.

- **TXUBR: TX Used Bit Read**

Set when a transmit buffer descriptor is read with its used bit set. Cleared on read.

- **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.

- **RLEX: Retry Limit Exceeded**

Transmit error. Cleared on read.

- **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.

- **TCOMP: Transmit Complete**

Set when a frame has been transmitted. Cleared on read.

- **ROVR: Receive Overrun**

Set when the receive overrun status bit is set. Cleared on read.

- **HRESP: HRESP Not OK**

Set when the DMA block sees HRESP not OK. Cleared on read.

- **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.

- **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.

- **PFTR: Pause Frame Transmitted**

Indicates a pause frame has been successfully transmitted after being initiated from the Network Control register. Cleared on read.

- **DRQFR: PTP Delay Request Frame Received**

Indicates a PTP delay_req frame has been received. Cleared on read.

- **SFR: PTP Sync Frame Received**

Indicates a PTP sync frame has been received. Cleared on read.

- **DRQFT: PTP Delay Request Frame Transmitted**

Indicates a PTP delay_req frame has been transmitted. Cleared on read.

- **SFT: PTP Sync Frame Transmitted**

Indicates a PTP sync frame has been transmitted. Cleared on read.

- **PDRQFR: PDelay Request Frame Received**

Indicates a PTP pdelay_req frame has been received. Cleared on read.

- **PDRSFR: PDelay Response Frame Received**

Indicates a PTP pdelay_resp frame has been received. Cleared on read.

- **PDRQFT: PDelay Request Frame Transmitted**

Indicates a PTP pdelay_req frame has been transmitted. Cleared on read.

- **PDRSFT: PDelay Response Frame Transmitted**

Indicates a PTP pdelay_resp frame has been transmitted. Cleared on read.

- **SRI: TSU Seconds Register Increment**

Indicates the register has incremented. Cleared on read.

- **RXLPISBC: Receive LPI indication Status Bit Change**

Receive LPI indication status bit change. Cleared on read.

- **WOL: Wake On LAN**

WOL interrupt. Indicates a WOL event has been received.

- **TSUTIMCOMP: TSU Timer Comparison**

Indicates when TSU timer count value is equal to programmed value. Cleared on read.

36.8.11 GMAC Interrupt Enable Register

Name: GMAC_IER

Address: 0xF8020028 (0), 0xFC028028 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
23	22	21	20	19	18	17	16
PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR	–	–
15	14	13	12	11	10	9	8
EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR	–	–
7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS

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.

- **MFS: Management Frame Sent**
- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **TXUBR: TX Used Bit Read**
- **TUR: Transmit Underrun**
- **RLEX: Retry Limit Exceeded or Late Collision**
- **TFC: Transmit Frame Corruption Due to AHB Error**
- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**
- **PFNZ: Pause Frame with Non-zero Pause Quantum Received**
- **PTZ: Pause Time Zero**
- **PFTR: Pause Frame Transmitted**
- **EXINT: External Interrupt**
- **DRQFR: PTP Delay Request Frame Received**

- **SFR: PTP Sync Frame Received**
- **DRQFT: PTP Delay Request Frame Transmitted**
- **SFT: PTP Sync Frame Transmitted**
- **PDRQFR: PDelay Request Frame Received**
- **PDRSFR: PDelay Response Frame Received**
- **PDRQFT: PDelay Request Frame Transmitted**
- **PDRSFT: PDelay Response Frame Transmitted**
- **SRI: TSU Seconds Register Increment**
- **RXLPISBC: Enable RX LPI Indication**
- **WOL: Wake On LAN**
- **TSUTIMCOMP: TSU Timer Comparison**

36.8.12 GMAC Interrupt Disable Register

Name: GMAC_IDR

Address: 0xF802002C (0), 0xFC02802C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
23	22	21	20	19	18	17	16
PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR	–	–
15	14	13	12	11	10	9	8
EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR	–	–
7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS

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.

- **MFS: Management Frame Sent**
- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **TXUBR: TX Used Bit Read**
- **TUR: Transmit Underrun**
- **RLEX: Retry Limit Exceeded or Late Collision**
- **TFC: Transmit Frame Corruption Due to AHB Error**
- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**
- **PFNZ: Pause Frame with Non-zero Pause Quantum Received**
- **PTZ: Pause Time Zero**
- **PFTR: Pause Frame Transmitted**
- **EXINT: External Interrupt**
- **DRQFR: PTP Delay Request Frame Received**

- **SFR: PTP Sync Frame Received**
- **DRQFT: PTP Delay Request Frame Transmitted**
- **SFT: PTP Sync Frame Transmitted**
- **PDRQFR: PDelay Request Frame Received**
- **PDRSFR: PDelay Response Frame Received**
- **PDRQFT: PDelay Request Frame Transmitted**
- **PDRSFT: PDelay Response Frame Transmitted**
- **SRI: TSU Seconds Register Increment**
- **RXLPISBC: Enable RX LPI Indication**
- **WOL: Wake On LAN**
- **TSUTIMCOMP: TSU Timer Comparison**

36.8.13 GMAC Interrupt Mask Register

Name: GMAC_IMR

Address: 0xF8020030 (0), 0xFC028030 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	TSUTIMCOMP	WOL	RXLPISBC	SRI	PDRSFT	PDRQFT
23	22	21	20	19	18	17	16
PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR	–	–
15	14	13	12	11	10	9	8
EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR	–	–
7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS

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.

- **MFS: Management Frame Sent**
- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **TXUBR: TX Used Bit Read**
- **TUR: Transmit Underrun**
- **RLEX: Retry Limit Exceeded**
- **TFC: Transmit Frame Corruption Due to AHB Error**
- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**
- **PFNZ: Pause Frame with Non-zero Pause Quantum Received**
- **PTZ: Pause Time Zero**

- **PFTR: Pause Frame Transmitted**
- **EXINT: External Interrupt**
- **DRQFR: PTP Delay Request Frame Received**
- **SFR: PTP Sync Frame Received**
- **DRQFT: PTP Delay Request Frame Transmitted**
- **SFT: PTP Sync Frame Transmitted**
- **PDRQFR: PDelay Request Frame Received**
- **PDRSFR: PDelay Response Frame Received**
- **PDRQFT: PDelay Request Frame Transmitted**
- **PDRSFT: PDelay Response Frame Transmitted**
- **SRI: TSU Seconds Register Increment**
- **RXLPISBC: Enable RX LPI Indication**
- **WOL: Wake On LAN**
- **TSUTIMCOMP: TSU Timer Comparison**

36.8.14 GMAC PHY Maintenance Register

Name: GMAC_MAN

Address: 0xF8020034 (0), 0xFC028034 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WZO	CLTTO	OP		PHYA			
23	22	21	20	19	18	17	16
PHYA	REGA					WTN	
15	14	13	12	11	10	9	8
DATA							
7	6	5	4	3	2	1	0
DATA							

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 MDIO pin and the LSB updated from the MDIO pin with each MDC cycle. This causes transmission of a PHY management frame on MDIO. Refer to *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. Refer to Table 36-18.

Table 36-18. 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, refer to Section 36.8.2 “GMAC Network Configuration Register”.

- **DATA: 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.

- **WTN: Write Ten**

Must be written to 10.

- **REGA: Register Address**

Specifies the register in the PHY to access.

- **PHYA: PHY Address**

- **OP: Operation**

01: Write

10: Read

- **CLTTO: Clause 22 Operation**

0: Clause 45 operation

1: Clause 22 operation

- **WZO: Write ZERO**

Must be written with 0.

36.8.15 GMAC Receive Pause Quantum Register

Name: GMAC_RPQ

Address: 0xF8020038 (0), 0xFC028038 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RPQ							

7	6	5	4	3	2	1	0
RPQ							

- **RPQ: 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

Address: 0xF802003C (0), 0xFC02803C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TPQ							

7	6	5	4	3	2	1	0
TPQ							

- **TPQ: Transmit Pause Quantum**

Written with the pause quantum value for pause frame transmission.

36.8.17 GMAC RX Jumbo Frame Max Length Register

Name: GMAC_RJFML

Address: 0xF8020048 (0), 0xFC028048 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	FML					

7	6	5	4	3	2	1	0
FML							

- **FML: Frame Max Length**

Rx jumbo frame maximum length.

36.8.18 GMAC Hash Register Bottom

Name: GMAC_HRB

Address: 0xF8020080 (0), 0xFC028080 (1)

Access: Read-only

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The unicast hash enable (UNIHEN) and the multicast hash enable (MITIHEN) bits in the Network Configuration Register (Section 36.8.2 “GMAC Network Configuration Register”) enable the reception of hash matched frames. Refer to Section 36.6.9 “Hash Addressing”.

- **ADDR: Hash Address**

The first 32 bits of the Hash Address Register.

36.8.19 GMAC Hash Register Top

Name: GMAC_HRT

Address: 0xF8020084 (0), 0xFC028084 (1)

Access: Read-only

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

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. Refer to Section 36.6.9 “Hash Addressing”.

- **ADDR: Hash Address**

Bits 63 to 32 of the Hash Address Register.

36.8.20 GMAC Specific Address 1 Bottom Register

Name: GMAC_SAB1

Address: 0xF8020088 (0), 0xFC028088 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

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.

- **ADDR: 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.

36.8.21 GMAC Specific Address 1 Top Register

Name: GMAC_SAT1

Address: 0xF802008C (0), 0xFC02808C (1)

Access: Read/Write

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

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.

- **ADDR: Specific Address 1**

The most significant bits of the destination address, that is, bits 47:32.

36.8.22 GMAC Specific Address 2 Bottom Register

Name: GMAC_SAB2

Address: 0xF8020090 (0), 0xFC028090 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

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.

- **ADDR: 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.

36.8.23 GMAC Specific Address 2 Top Register

Name: GMAC_SAT2

Address: 0xF8020094 (0), 0xFC028094 (1)

Access: Read/Write

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

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.

- **ADDR: Specific Address 2**

The most significant bits of the destination address, that is, bits 47:32.

36.8.24 GMAC Specific Address 3 Bottom Register

Name: GMAC_SAB3

Address: 0xF8020098 (0), 0xFC028098 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

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.

- **ADDR: 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.

36.8.25 GMAC Specific Address 3 Top Register

Name: GMAC_SAT3

Address: 0xF802009C (0), 0xFC02809C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

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.

- **ADDR: Specific Address 3**

The most significant bits of the destination address, that is, bits 47:32.

36.8.26 GMAC Specific Address 4 Bottom Register

Name: GMAC_SAB4

Address: 0xF80200A0 (0), 0xFC0280A0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

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.

- **ADDR: 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.

36.8.27 GMAC Specific Address 4 Top Register

Name: GMAC_SAT4

Address: 0xF80200A4 (0), 0xFC0280A4 (1)

Access: Read/Write

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

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.

- **ADDR: Specific Address 4**

The most significant bits of the destination address, that is, bits 47:32.

36.8.28 GMAC Type ID Match 1 Register

Name: GMAC_TIDM1

Address: 0xF80200A8 (0), 0xFC0280A8 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ENID1	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TID							

7	6	5	4	3	2	1	0
TID							

- **TID: Type ID Match 1**

For use in comparisons with received frames type ID/length frames.

- **ENID1: Enable Copying of TID Matched Frames**

0: TID is not part of the comparison match.

1: TID is processed for the comparison match.

36.8.29 GMAC Type ID Match 2 Register

Name: GMAC_TIDM2

Address: 0xF80200AC (0), 0xFC0280AC (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ENID2	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TID							

7	6	5	4	3	2	1	0
TID							

- **TID: Type ID Match 2**

For use in comparisons with received frames type ID/length frames.

- **ENID2: Enable Copying of TID Matched Frames**

0: TID is not part of the comparison match.

1: TID is processed for the comparison match.

36.8.30 GMAC Type ID Match 3 Register

Name: GMAC_TIDM3

Address: 0xF80200B0 (0), 0xFC0280B0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ENID3	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
TID							
7	6	5	4	3	2	1	0
TID							

- **TID: Type ID Match 3**

For use in comparisons with received frames type ID/length frames.

- **ENID3: Enable Copying of TID Matched Frames**

0: TID is not part of the comparison match.

1: TID is processed for the comparison match.

36.8.31 GMAC Type ID Match 4 Register

Name: GMAC_TIDM4

Address: 0xF80200B4 (0), 0xFC0280B4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ENID4	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TID							

7	6	5	4	3	2	1	0
TID							

- **TID: Type ID Match 4**

For use in comparisons with received frames type ID/length frames.

- **ENID4: Enable Copying of TID Matched Frames**

0: TID is not part of the comparison match.

1: TID is processed for the comparison match.

36.8.32 GMAC Wake on LAN Register

Name: GMAC_WOL

Address: 0xF80200B8 (0), 0xFC0280B8 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
–				MTI	SA1	ARP	MAG

15	14	13	12	11	10	9	8
IP							

7	6	5	4	3	2	1	0
IP							

- **IP: ARP Request IP Address**

Wake on LAN ARP request IP address. Written to define the least significant 16 bits of the target IP address that is matched to generate a Wake on LAN event. A value of zero will not generate an event, even if this is matched by the received frame.

- **MAG: Magic Packet Event Enable**

Wake on LAN magic packet event enable.

- **ARP: ARP Request Event Enable**

Wake on LAN ARP request event enable.

- **SA1: Specific Address Register 1 Event Enable**

Wake on LAN Specific Address Register 1 event enable.

- **MTI: Multicast Hash Event Enable**

Wake on LAN multicast hash event enable.

36.8.33 GMAC IPG Stretch Register

Name: GMAC_IPGS

Address: 0xF80200BC (0), 0xFC0280BC (1)

Access: Read/Write

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

- **FL: 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. Refer to Section 36.6.4 “MAC Transmit Block”.

36.8.34 GMAC Stacked VLAN Register

Name: GMAC_SVLAN

Address: 0xF80200C0 (0), 0xFC0280C0 (1)

Access: Read/Write

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

- **VLAN_TYPE: 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.

- **ESVLAN: Enable Stacked VLAN Processing Mode**

0: Disable the stacked VLAN processing mode

1: Enable the stacked VLAN processing mode

36.8.35 GMAC Transmit PFC Pause Register

Name: GMAC_TPFCP

Address: 0xF80200C4 (0), 0xFC0280C4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PQ							

7	6	5	4	3	2	1	0
PEV							

- **PEV: 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].

- **PQ: 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.

36.8.36 GMAC Specific Address 1 Mask Bottom Register

Name: GMAC_SAMB1

Address: 0xF80200C8 (0), 0xFC0280C8 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: Specific Address 1 Mask**

Setting a bit to one masks the corresponding bit in the Specific Address 1 Register.

36.8.37 GMAC Specific Address Mask 1 Top Register

Name: GMAC_SAMT1

Address: 0xF80200CC (0), 0xFC0280CC (1)

Access: Read/Write

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

- **ADDR: Specific Address 1 Mask**

Setting a bit to one masks the corresponding bit in the Specific Address 1 Register.

36.8.38 GMAC 1588 Timer Nanosecond Comparison Register

Name: GMAC_NSC

Address: 0xF80200DC (0), 0xFC0280DC (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	NANOSEC					

15	14	13	12	11	10	9	8
NANOSEC							

7	6	5	4	3	2	1	0
NANOSEC							

- **NANOSEC: 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).

36.8.39 GMAC 1588 Timer Second Comparison Low Register

Name: GMAC_SCL

Address: 0xF80200E0 (0), 0xFC0280E0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
SEC							

23	22	21	20	19	18	17	16
SEC							

15	14	13	12	11	10	9	8
SEC							

7	6	5	4	3	2	1	0
SEC							

- **SEC: 1588 Timer Second Comparison Value**

Value is compared to seconds value bits [31:0] of the TSU timer count value.

36.8.40 GMAC 1588 Timer Second Comparison High Register

Name: GMAC_SCH

Address: 0xF80200E4 (0), 0xFC0280E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
SEC							

7	6	5	4	3	2	1	0
SEC							

- **SEC: 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.

36.8.41 GMAC PTP Event Frame Transmitted Seconds High Register

Name: GMAC_EFTSH

Address: 0xF80200E8 (0), 0xFC0280E8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.42 GMAC PTP Event Frame Received Seconds High Register

Name: GMAC_EFRSH

Address: 0xF80200EC (0), 0xFC0280EC (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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 Peer Event Frame Transmitted Seconds High Register

Name: GMAC_PEFTSH

Address: 0xF80200F0 (0), 0xFC0280F0 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.44 GMAC PTP Peer Event Frame Received Seconds High Register

Name: GMAC_PEFRSH

Address: 0xF80200F4 (0), 0xFC0280F4 (1)

Access: Read-only

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

- **RUD: 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 Octets Transmitted Low Register

Name: GMAC_OTLO

Address: 0xF8020100 (0), 0xFC028100 (1)

Access: Read-only

31	30	29	28	27	26	25	24
TXO							

23	22	21	20	19	18	17	16
TXO							

15	14	13	12	11	10	9	8
TXO							

7	6	5	4	3	2	1	0
TXO							

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

- **TXO: 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.

36.8.46 GMAC Octets Transmitted High Register

Name: GMAC_OTHI

Address: 0xF8020104 (0), 0xFC028104 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TXO							

7	6	5	4	3	2	1	0
TXO							

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

- **TXO: 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.

36.8.47 GMAC Frames Transmitted Register

Name: GMAC_FT

Address: 0xF8020108 (0), 0xFC028108 (1)

Access: Read-only

31	30	29	28	27	26	25	24
FTX							

23	22	21	20	19	18	17	16
FTX							

15	14	13	12	11	10	9	8
FTX							

7	6	5	4	3	2	1	0
FTX							

- **FTX: 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.

36.8.48 GMAC Broadcast Frames Transmitted Register

Name: GMAC_BCFT

Address: 0xF802010C (0), 0xFC02810C (1)

Access: Read-only

31	30	29	28	27	26	25	24
BFTX							

23	22	21	20	19	18	17	16
BFTX							

15	14	13	12	11	10	9	8
BFTX							

7	6	5	4	3	2	1	0
BFTX							

- **BFTX: 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.

36.8.49 GMAC Multicast Frames Transmitted Register

Name: GMAC_MFT

Address: 0xF8020110 (0), 0xFC028110 (1)

Access: Read-only

31	30	29	28	27	26	25	24
MFTX							

23	22	21	20	19	18	17	16
MFTX							

15	14	13	12	11	10	9	8
MFTX							

7	6	5	4	3	2	1	0
MFTX							

- **MFTX: 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.

36.8.50 GMAC Pause Frames Transmitted Register

Name: GMAC_PFT

Address: 0xF8020114 (0), 0xFC028114 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PFTX							

7	6	5	4	3	2	1	0
PFTX							

- **PFTX: 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.

36.8.51 GMAC 64 Byte Frames Transmitted Register

Name: GMAC_BFT64

Address: 0xF8020118 (0), 0xFC028118 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 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.

36.8.52 GMAC 65 to 127 Byte Frames Transmitted Register

Name: GMAC_TBFT127

Address: 0xF802011C (0), 0xFC02811C (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 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.

36.8.53 GMAC 128 to 255 Byte Frames Transmitted Register

Name: GMAC_TBFT255

Address: 0xF8020120 (0), 0xFC028120 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 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.

36.8.54 GMAC 256 to 511 Byte Frames Transmitted Register

Name: GMAC_TBFT511

Address: 0xF8020124 (0), 0xFC028124 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 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.

36.8.55 GMAC 512 to 1023 Byte Frames Transmitted Register

Name: GMAC_TBFT1023

Address: 0xF8020128 (0), 0xFC028128 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 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.

36.8.56 GMAC 1024 to 1518 Byte Frames Transmitted Register

Name: GMAC_TBFT1518

Address: 0xF802012C (0), 0xFC02812C (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 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.

36.8.57 GMAC Greater Than 1518 Byte Frames Transmitted Register

Name: GMAC_GTBFT1518

Address: 0xF8020130 (0), 0xFC028130 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 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.

36.8.58 GMAC Transmit Underruns Register

Name: GMAC_TUR

Address: 0xF8020134 (0), 0xFC028134 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXUNR	

7	6	5	4	3	2	1	0
TXUNR							

- **TXUNR: 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.

36.8.59 GMAC Single Collision Frames Register

Name: GMAC_SCF

Address: 0xF8020138 (0), 0xFC028138 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	SCOL	

15	14	13	12	11	10	9	8
SCOL							

7	6	5	4	3	2	1	0
SCOL							

- **SCOL: Single Collision**

This register counts the number of frames experiencing a single collision before being successfully transmitted i.e., no underrun.

36.8.60 GMAC Multiple Collision Frames Register

Name: GMAC_MCF

Address: 0xF802013C (0), 0xFC02813C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	MCOL	

15	14	13	12	11	10	9	8
MCOL							

7	6	5	4	3	2	1	0
MCOL							

- **MCOL: 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.

36.8.61 GMAC Excessive Collisions Register

Name: GMAC_EC

Address: 0xF8020140 (0), 0xFC028140 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	XCOL	

7	6	5	4	3	2	1	0
XCOL							

- **XCOL: Excessive Collisions**

This register counts the number of frames that failed to be transmitted because they experienced 16 collisions.

36.8.62 GMAC Late Collisions Register

Name: GMAC_LC

Address: 0xF8020144 (0), 0xFC028144 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	LCOL	

7	6	5	4	3	2	1	0
LCOL							

- **LCOL: 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.

36.8.63 GMAC Deferred Transmission Frames Register

Name: GMAC_DTF

Address: 0xF8020148 (0), 0xFC028148 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	DEFT	

15	14	13	12	11	10	9	8
DEFT							

7	6	5	4	3	2	1	0
DEFT							

- **DEFT: 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.

36.8.64 GMAC Carrier Sense Errors Register

Name: GMAC_CSE

Address: 0xF802014C (0), 0xFC02814C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	CSR	

7	6	5	4	3	2	1	0
CSR							

- **CSR: 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.

36.8.65 GMAC Octets Received Low Register

Name: GMAC_ORLO

Address: 0xF8020150 (0), 0xFC028150 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RXO							

23	22	21	20	19	18	17	16
RXO							

15	14	13	12	11	10	9	8
RXO							

7	6	5	4	3	2	1	0
RXO							

When reading the Octets Transmitted and Octets Received Registers, bits [31:0] should be read prior to bits [47:32] to ensure reliable operation.

- **RXO: 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.

36.8.66 GMAC Octets Received High Register

Name: GMAC_ORHI

Address: 0xF8020154 (0), 0xFC028154 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RXO							

7	6	5	4	3	2	1	0
RXO							

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

- **RXO: 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.

36.8.67 GMAC Frames Received Register

Name: GMAC_FR

Address: 0xF8020158 (0), 0xFC028158 (1)

Access: Read-only

31	30	29	28	27	26	25	24
FRX							

23	22	21	20	19	18	17	16
FRX							

15	14	13	12	11	10	9	8
FRX							

7	6	5	4	3	2	1	0
FRX							

- **FRX: 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.

36.8.68 GMAC Broadcast Frames Received Register

Name: GMAC_BCFR

Address: 0xF802015C (0), 0xFC02815C (1)

Access: Read-only

31	30	29	28	27	26	25	24
BFRX							

23	22	21	20	19	18	17	16
BFRX							

15	14	13	12	11	10	9	8
BFRX							

7	6	5	4	3	2	1	0
BFRX							

- **BFRX: 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.

36.8.69 GMAC Multicast Frames Received Register

Name: GMAC_MFR

Address: 0xF8020160 (0), 0xFC028160 (1)

Access: Read-only

31	30	29	28	27	26	25	24
MFRX							

23	22	21	20	19	18	17	16
MFRX							

15	14	13	12	11	10	9	8
MFRX							

7	6	5	4	3	2	1	0
MFRX							

- **MFRX: 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.

36.8.70 GMAC Pause Frames Received Register

Name: GMAC_PFR

Address: 0xF8020164 (0), 0xFC028164 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PFRX							

7	6	5	4	3	2	1	0
PFRX							

- **PFRX: Pause Frames Received Register**

This register counts the number of pause frames received without error.

36.8.71 GMAC 64 Byte Frames Received Register

Name: GMAC_BFR64

Address: 0xF8020168 (0), 0xFC028168 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 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.

36.8.72 GMAC 65 to 127 Byte Frames Received Register

Name: GMAC_TBFR127

Address: 0xF802016C (0), 0xFC02816C (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 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.

36.8.73 GMAC 128 to 255 Byte Frames Received Register

Name: GMAC_TBFR255

Address: 0xF8020170 (0), 0xFC028170 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 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.

36.8.74 GMAC 256 to 511 Byte Frames Received Register

Name: GMAC_TBFR511

Address: 0xF8020174 (0), 0xFC028174 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 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.

36.8.75 GMAC 512 to 1023 Byte Frames Received Register

Name: GMAC_TBFR1023

Address: 0xF8020178 (0), 0xFC028178 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 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.

36.8.76 GMAC 1024 to 1518 Byte Frames Received Register

Name: GMAC_TBFR1518

Address: 0xF802017C (0), 0xFC02817C (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 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.

36.8.77 GMAC 1519 to Maximum Byte Frames Received Register

Name: GMAC_TMXBFR

Address: 0xF8020180 (0), 0xFC028180 (1)

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 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. Refer to Section 36.8.2 “GMAC Network Configuration Register”.

36.8.78 GMAC Undersized Frames Received Register

Name: GMAC_UFR

Address: 0xF8020184 (0), 0xFC028184 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	UFRX	

7	6	5	4	3	2	1	0
UFRX							

- **UFRX: 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.

36.8.79 GMAC Oversized Frames Received Register

Name: GMAC_OFR

Address: 0xF8020188 (0), 0xFC028188 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	OFRX	

7	6	5	4	3	2	1	0
OFRX							

- **OFRX: 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. Refer to Section 36.8.2 “GMAC Network Configuration Register”.

36.8.80 GMAC Jabbers Received Register

Name: GMAC_JR

Address: 0xF802018C (0), 0xFC02818C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	JRX	

7	6	5	4	3	2	1	0
JRX							

- **JRX: 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. Refer to Section 36.8.2 “GMAC Network Configuration Register”.

36.8.81 GMAC Frame Check Sequence Errors Register

Name: GMAC_FCSE

Address: 0xF8020190 (0), 0xFC028190 (1)

Access: Read-only

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

- **FCKR: 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. Refer to Section 36.8.2 “GMAC Network Configuration Register”.

36.8.82 GMAC Length Field Frame Errors Register

Name: GMAC_LFFE

Address: 0xF8020194 (0), 0xFC028194 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	LFER	

7	6	5	4	3	2	1	0
LFER							

- **LFER: 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. Refer to Section 36.8.2 “GMAC Network Configuration Register”.

36.8.83 GMAC Receive Symbol Errors Register

Name: GMAC_RSE

Address: 0xF8020198 (0), 0xFC028198 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXSE	

7	6	5	4	3	2	1	0
RXSE							

- **RXSE: 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. Refer to Section 36.8.2 “GMAC Network Configuration Register”.

36.8.84 GMAC Alignment Errors Register

Name: GMAC_AE

Address: 0xF802019C (0), 0xFC02819C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	AER	

7	6	5	4	3	2	1	0
AER							

- **AER: 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. Refer to Section 36.8.2 “GMAC Network Configuration Register”.

36.8.85 GMAC Receive Resource Errors Register

Name: GMAC_RRE

Address: 0xF80201A0 (0), 0xFC0281A0 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	RXRER	

15	14	13	12	11	10	9	8
RXRER							

7	6	5	4	3	2	1	0
RXRER							

- **RXRER: Receive Resource Errors**

This register counts 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. Refer to Section 36.8.2 “GMAC Network Configuration Register”.

36.8.86 GMAC Receive Overruns Register

Name: GMAC_ROE

Address: 0xF80201A4 (0), 0xFC0281A4 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXOVR	

7	6	5	4	3	2	1	0
RXOVR							

- **RXOVR: Receive Overruns**

This register counts the number of frames that are address recognized but were not copied to memory due to a receive overrun.

36.8.87 GMAC IP Header Checksum Errors Register

Name: GMAC_IHCE

Address: 0xF80201A8 (0), 0xFC0281A8 (1)

Access: Read-only

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
HCKER							

- **HCKER: 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.

36.8.88 GMAC TCP Checksum Errors Register

Name: GMAC_TCE

Address: 0xF80201AC (0), 0xFC0281AC (1)

Access: Read-only

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
TCKER							

- **TCKER: 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.

36.8.89 GMAC UDP Checksum Errors Register

Name: GMAC_UCE

Address: 0xF80201B0 (0), 0xFC0281B0 (1)

Access: Read-only

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
UCKER							

- **UCKER: 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.

36.8.90 GMAC 1588 Timer Increment Sub-nanoseconds Register

Name: GMAC_TISUBN

Address: 0xF80201BC (0), 0xFC0281BC (1)

Access: Read/Write

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

- **LSBTIR: 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.

36.8.91 GMAC 1588 Timer Seconds High Register

Name: GMAC_TSH

Address: 0xF80201C0 (0), 0xFC0281C0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TCS							

7	6	5	4	3	2	1	0
TCS							

- **TCS: 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.

36.8.92 GMAC 1588 Timer Seconds Low Register

Name: GMAC_TSL

Address: 0xF80201D0 (0), 0xFC0281D0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
TCS							

23	22	21	20	19	18	17	16
TCS							

15	14	13	12	11	10	9	8
TCS							

7	6	5	4	3	2	1	0
TCS							

- **TCS: 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 Nanoseconds Register

Name: GMAC_TN

Address: 0xF80201D4 (0), 0xFC0281D4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	TNS					

23	22	21	20	19	18	17	16
TNS							

15	14	13	12	11	10	9	8
TNS							

7	6	5	4	3	2	1	0
TNS							

- **TNS: 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.

36.8.94 GMAC 1588 Timer Adjust Register

Name: GMAC_TA

Address: 0xF80201D8 (0), 0xFC0281D8 (1)

Access: Write-only

31	30	29	28	27	26	25	24
ADJ	–	ITDT					

23	22	21	20	19	18	17	16
ITDT							

15	14	13	12	11	10	9	8
ITDT							

7	6	5	4	3	2	1	0
ITDT							

- **ITDT: 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.

- **ADJ: Adjust 1588 Timer**

Write as one to subtract from the 1588 timer. Write as zero to add to it.

36.8.95 GMAC 1588 Timer Increment Register

Name: GMAC_TI

Address: 0xF80201DC (0), 0xFC0281DC (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
NIT							

15	14	13	12	11	10	9	8
ACNS							

7	6	5	4	3	2	1	0
CNS							

- **CNS: Count Nanoseconds**

A count of nanoseconds by which the 1588 Timer Nanoseconds Register will be incremented each clock cycle.

- **ACNS: Alternative Count Nanoseconds**

Alternative count of nanoseconds by which the 1588 Timer Nanoseconds Register will be incremented each clock cycle.

- **NIT: Number of Increments**

The number of increments after which the alternative increment is used.

36.8.96 GMAC PTP Event Frame Transmitted Seconds Low Register

Name: GMAC_EFTSL

Address: 0xF80201E0 (0), 0xFC0281E0 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RUD							

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.97 GMAC PTP Event Frame Transmitted Nanoseconds Register

Name: GMAC_EFTN

Address: 0xF80201E4 (0), 0xFC0281E4 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	RUD					

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.98 GMAC PTP Event Frame Received Seconds Low Register

Name: GMAC_EFRSL

Address: 0xF80201E8 (0), 0xFC0281E8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RUD							

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.99 GMAC PTP Event Frame Received Nanoseconds Register

Name: GMAC_EFRN

Address: 0xF80201EC (0), 0xFC0281EC (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	RUD					

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.100GMAC PTP Peer Event Frame Transmitted Seconds Low Register

Name: GMAC_PEFTSL

Address: 0xF80201F0 (0), 0xFC0281F0 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RUD							

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.101GMAC PTP Peer Event Frame Transmitted Nanoseconds Register

Name: GMAC_PEFTN

Address: 0xF80201F4 (0), 0xFC0281F4 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	RUD					

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.102GMAC PTP Peer Event Frame Received Seconds Low Register

Name: GMAC_PEFRSL

Address: 0xF80201F8 (0), 0xFC0281F8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RUD							

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.103GMAC PTP Peer Event Frame Received Nanoseconds Register

Name: GMAC_PEFRN

Address: 0xF80201FC (0), 0xFC0281FC (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	RUD					

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: 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.

36.8.104GMAC Received LPI Transitions

Name: GMAC_RXLPI

Address: 0xF8020270 (0), 0xFC028270 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
COUNT							

7	6	5	4	3	2	1	0
COUNT							

- **COUNT: 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.

36.8.105GMAC Received LPI Time

Name: GMAC_RXLPITIME

Address: 0xF8020274 (0), 0xFC028274 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
LPITIME							

15	14	13	12	11	10	9	8
LPITIME							

7	6	5	4	3	2	1	0
LPITIME							

- **LPITIME: Time in LPI (cleared on read)**

This field increments once every 16 PCLK cycles when the bit LPI Indication (bit 7) is set in the Network Status register.

36.8.106GMAC Transmit LPI Transitions

Name: GMAC_TXLPI

Address: 0xF8020278 (0), 0xFC028278 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
COUNT							

7	6	5	4	3	2	1	0
COUNT							

- **COUNT: 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.

36.8.107GMAC Transmit LPI Time

Name: GMAC_TXLPITIME

Address: 0xF802027C (0), 0xFC02827C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
LPITIME							

15	14	13	12	11	10	9	8
LPITIME							

7	6	5	4	3	2	1	0
LPITIME							

- **LPITIME: Time in LPI (cleared on read)**

This field increments once every 16 PCLK cycles when the bit Enable LPI Transmission (bit 19) is set in the Network Control register.

37. High Speed Multimedia Card Interface (HSMCI)

37.1 Description

The High Speed Multimedia Card Interface (HSMCI) supports the MultiMedia Card (MMC) Specification V4.3, the SD Memory Card Specification V2.0, the SDIO V2.0 specification and CE-ATA V1.1.

The HSMCI includes a command register, response registers, data registers, timeout counters and error detection logic that automatically handle the transmission of commands and, when required, the reception of the associated responses and data with a limited processor overhead.

The HSMCI operates at a rate of up to Master Clock divided by 2 and supports the interfacing of 1 slot(s). Each slot may be used to interface with a High Speed MultiMedia Card bus (up to 30 Cards) or with an SD Memory Card. Only one slot can be selected at a time (slots are multiplexed). A bit field in the SD Card Register performs this selection.

The SD Memory Card communication is based on a 9-pin interface (clock, command, four data and three power lines) and the High Speed MultiMedia Card on a 7-pin interface (clock, command, one data, three power lines and one reserved for future use).

The SD Memory Card interface also supports High Speed MultiMedia Card operations. The main differences between SD and High Speed MultiMedia Cards are the initialization process and the bus topology.

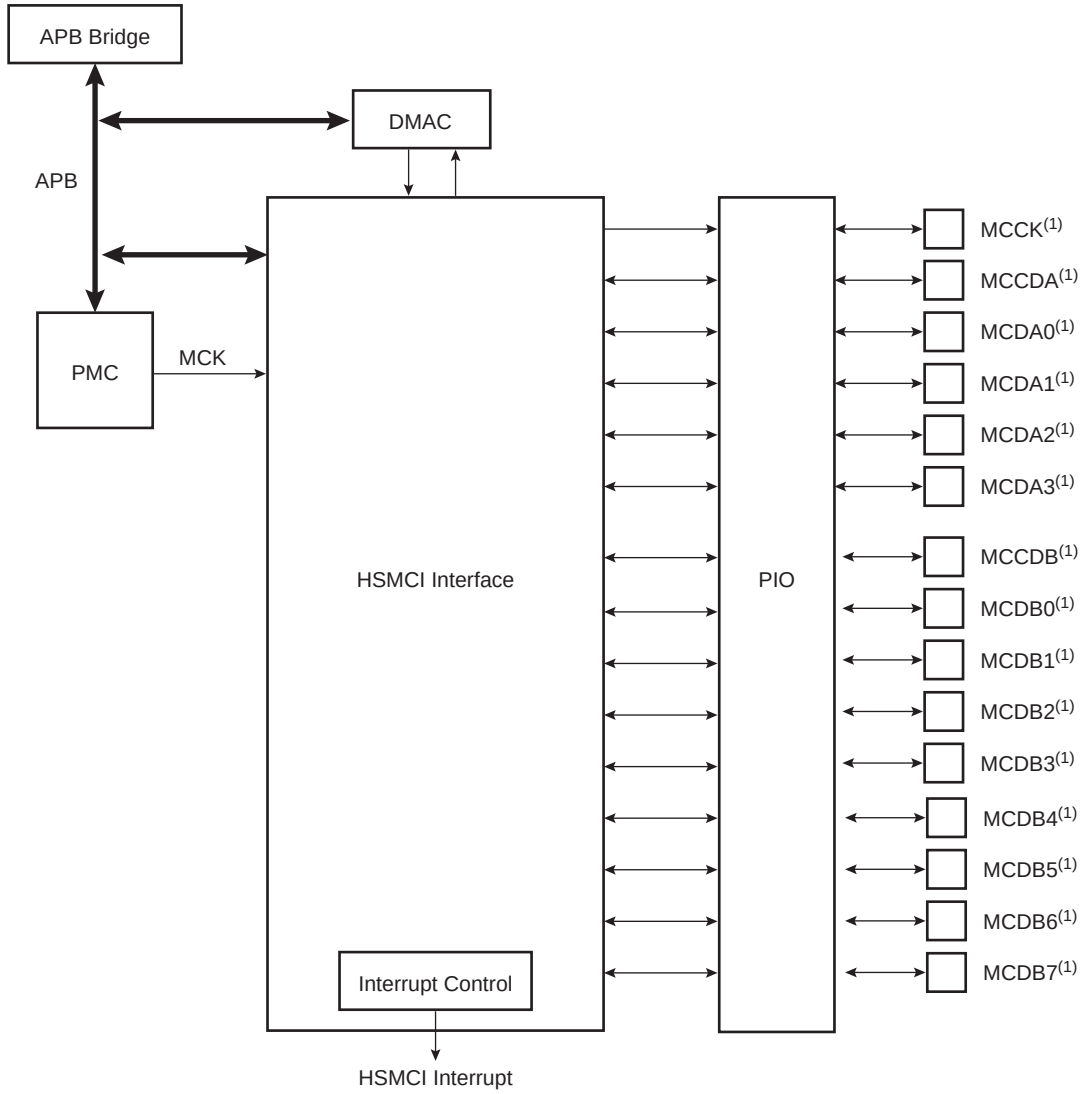
HSMCI fully supports CE-ATA Revision 1.1, built on the MMC System Specification v4.0. The module includes dedicated hardware to issue the command completion signal and capture the host command completion signal disable.

37.2 Embedded Characteristics

- Compatible with MultiMedia Card Specification Version 4.3
- Compatible with SD Memory Card Specification Version 2.0
- Compatible with SDIO Specification Version 2.0
- Compatible with CE-ATA Specification 1.1
- Cards Clock Rate Up to Master Clock Divided by 2
- Boot Operation Mode Support
- High Speed Mode Support
- Embedded Power Management to Slow Down Clock Rate When Not Used
- Supports 1 Multiplexed Slot(s)
  - Each Slot for either a High Speed MultiMedia Card Bus (Up to 30 Cards) or an SD Memory Card
- Support for Stream, Block and Multi-block Data Read and Write
  - Minimizes Processor Intervention for Large Buffer Transfers
- Built in FIFO (from 16 to 256 bytes) with Large Memory Aperture Supporting Incremental Access
- Support for CE-ATA Completion Signal Disable Command
- Protection Against Unexpected Modification On-the-Fly of the Configuration Registers

37.3 Block Diagram

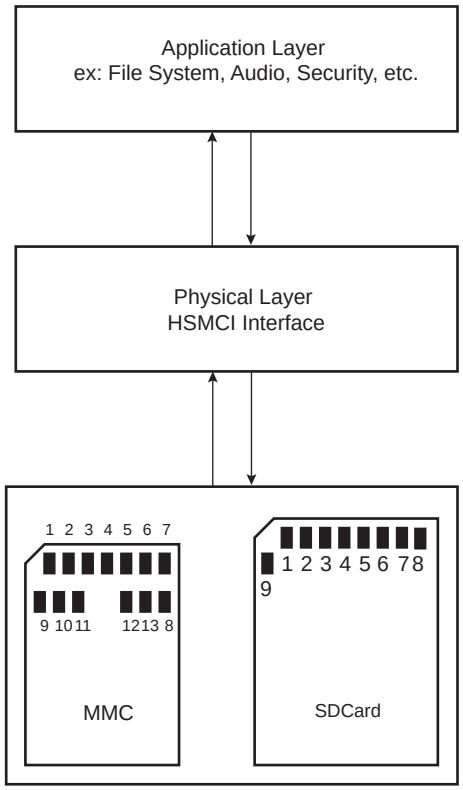
Figure 37-1. Block Diagram



Note: 1. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCCDB to HSMCIx_CDB, MCDAy to HSMCIx_DAy, MCDBy to HSMCIx_DBy.

37.4 Application Block Diagram

Figure 37-2. Application Block Diagram



37.5 Pin Name List

Table 37-1. I/O Lines Description for 8-bit Configuration

Pin Name[1]	Pin Description	Type[2]	Comments
MCCDA/MCCDB	Command/response	I/O/PP/OD	CMD of an MMC or SDCard/SDIO
MCCK	Clock	I/O	CLK of an MMC or SD Card/SDIO
MCDA0–MCDA7	Data 0..7 of Slot A	I/O/PP	DAT[0..7] of an MMC DAT[0..3] of an SD Card/SDIO
MCDB0–MCDB7	Data 0..7 of Slot B	I/O/PP	DAT[0..7] of an MMC DAT[0..3] of an SD Card/SDIO

Notes:
1. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCCDB to HSMCIx_CDB, MCCDC to HSMCIx_CDC, MCCDD to HSMCIx_CDD, MCDAy to HSMCIx_DAy, MCDBy to HSMCIx_DBy, MCDCy to HSMCIx_DCy, MCDDy to HSMCIx_DDy.
2. I: Input, O: Output, PP: Push/Pull, OD: Open Drain.

Table 37-2. I/O Lines Description for 4-bit Configuration

Pin Name[1]	Pin Description	Type[2]	Comments
MCCDA/MCCDB	Command/response	I/O/PP/OD	CMD of an MMC or SDCard/SDIO
MCCK	Clock	I/O	CLK of an MMC or SD Card/SDIO
MCDA0–MCDA3	Data 0..3 of Slot A	I/O/PP	DAT[0..3] of an MMC DAT[0..3] of an SD Card/SDIO

Notes: 1. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCDAy to HSMCIx_DAy.
2. I: Input, O: Output, PP: Push/Pull, OD: Open Drain.

37.6 Product Dependencies

37.6.1 I/O Lines

The pins used for interfacing the High Speed MultiMedia Cards or SD Cards are multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the peripheral functions to HSMCI pins.

Table 37-3. I/O Lines

Instance	Signal	I/O Line	Peripheral
HSMCI0	MCI0_CDA	PC5	B
HSMCI0	MCI0_CDB	PE0	B
HSMCI0	MCI0_CK	PC4	B
HSMCI0	MCI0_DA0	PC6	B
HSMCI0	MCI0_DA1	PC7	B
HSMCI0	MCI0_DA2	PC8	B
HSMCI0	MCI0_DA3	PC9	B
HSMCI0	MCI0_DA4	PC10	B
HSMCI0	MCI0_DA5	PC11	B
HSMCI0	MCI0_DA6	PC12	B
HSMCI0	MCI0_DA7	PC13	B
HSMCI0	MCI0_DB0	PE1	B
HSMCI0	MCI0_DB1	PE2	B
HSMCI0	MCI0_DB2	PE3	B
HSMCI0	MCI0_DB3	PE4	B
HSMCI1	MCI1_CDA	PE19	C
HSMCI1	MCI1_CK	PE18	C
HSMCI1	MCI1_DA0	PE20	C
HSMCI1	MCI1_DA1	PE21	C
HSMCI1	MCI1_DA2	PE22	C
HSMCI1	MCI1_DA3	PE23	C

37.6.2 Power Management

The HSMCI is clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the HSMCI clock.

37.6.3 Interrupt Sources

The HSMCI has an interrupt line connected to the interrupt controller.

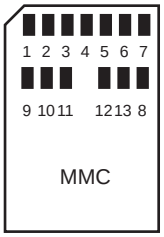
Handling the HSMCI interrupt requires programming the interrupt controller before configuring the HSMCI.

Table 37-4. Peripheral IDs

Instance	ID
HSMCI0	35
HSMCI1	36

37.7 Bus Topology

Figure 37-3. High Speed MultiMedia Memory Card Bus Topology



The High Speed MultiMedia Card communication is based on a 13-pin serial bus interface. It has three communication lines and four supply lines.

Table 37-5. Bus Topology

Pin Number	Name	Type[1]	Description	HSMCI Pin Name[2] (Slot z)
1	DAT[3]	I/O/PP	Data	MCDz3
2	CMD	I/O/PP/OD	Command/response	MCCDz
3	VSS1	S	Supply voltage ground	VSS
4	VDD	S	Supply voltage	VDD
5	CLK	I/O	Clock	MCCK
6	VSS2	S	Supply voltage ground	VSS
7	DAT[0]	I/O/PP	Data 0	MCDz0
8	DAT[1]	I/O/PP	Data 1	MCDz1
9	DAT[2]	I/O/PP	Data 2	MCDz2

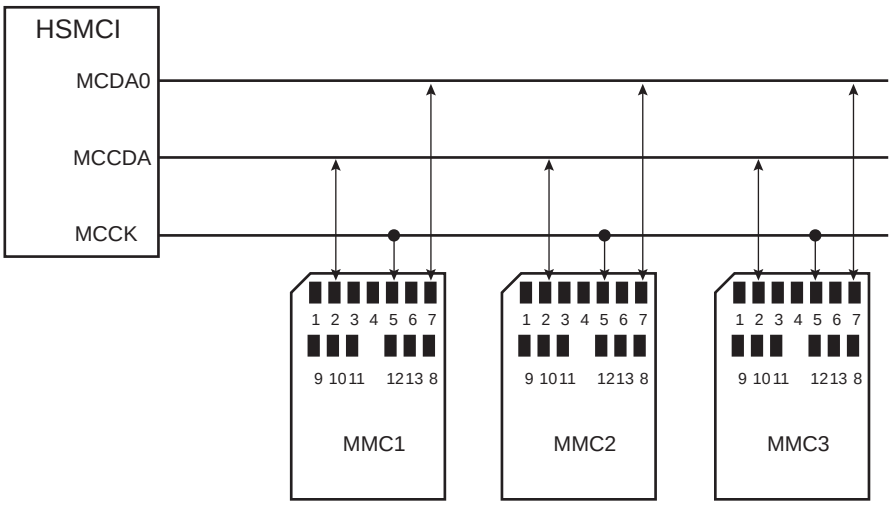
Table 37-5. Bus Topology (Continued)

Pin Number	Name	Type[1]	Description	HSMCI Pin Name[2] (Slot z)
10	DAT[4]	I/O/PP	Data 4	MCDz4
11	DAT[5]	I/O/PP	Data 5	MCDz5
12	DAT[6]	I/O/PP	Data 6	MCDz6
13	DAT[7]	I/O/PP	Data 7	MCDz7

Notes: 1. I: Input, O: Output, PP: Push/Pull, OD: Open Drain, S: Supply

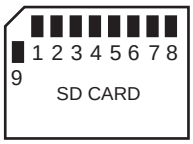
2. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCCDB to HSMCIx_CDB, MCDAy to HSMCIx_DAy, MCDBy to HSMCIx_DBy.

Figure 37-4. MMC Bus Connections (One Slot)



Note: When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA MCDAy to HSMCIx_DAy.

Figure 37-5. SD Memory Card Bus Topology



The SD Memory Card bus includes the signals listed in Table 37-6.

Table 37-6. SD Memory Card Bus Signals

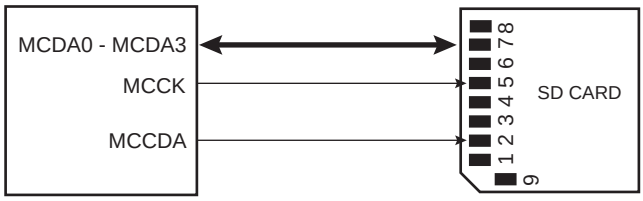
Pin Number	Name	Type[1]	Description	HSMCI Pin Name[2] (Slot z)
1	CD/DAT[3]	I/O/PP	Card detect/ Data line Bit 3	MCDz3
2	CMD	PP	Command/response	MCCDz
3	VSS1	S	Supply voltage ground	VSS
4	VDD	S	Supply voltage	VDD
5	CLK	I/O	Clock	MCCK
6	VSS2	S	Supply voltage ground	VSS

Table 37-6. SD Memory Card Bus Signals (Continued)

Pin Number	Name	Type[1]	Description	HSMCI Pin Name[2] (Slot z)
7	DAT[0]	I/O/PP	Data line Bit 0	MCDz0
8	DAT[1]	I/O/PP	Data line Bit 1 or Interrupt	MCDz1
9	DAT[2]	I/O/PP	Data line Bit 2	MCDz2

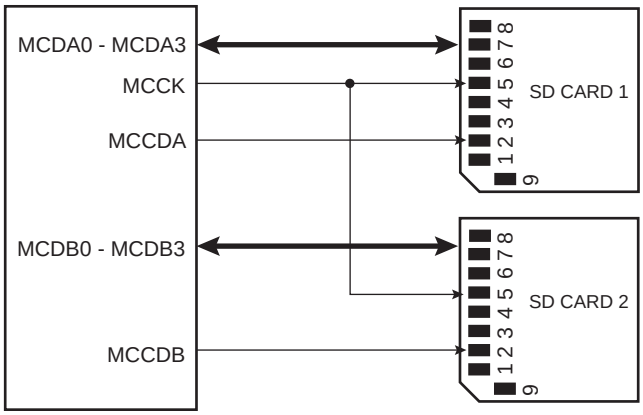
Notes: 1. I: input, O: output, PP: Push Pull, OD: Open Drain.
2. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCCDB to HSMCIx_CDB, MCDAy to HSMCIx_DAy, MCDBy to HSMCIx_DBy.

Figure 37-6. SD Card Bus Connections with One Slot



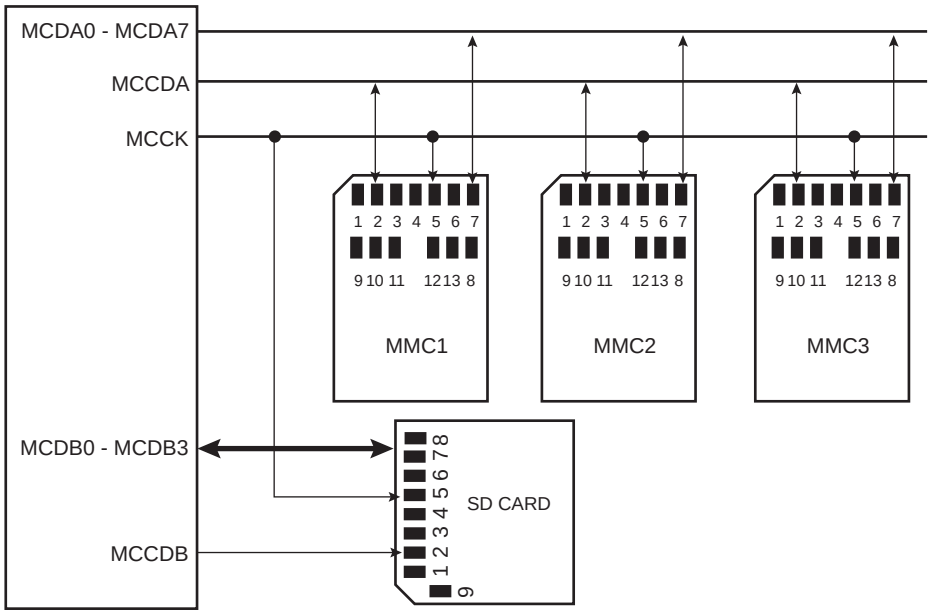
Note: When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA MCDAy to HSMCIx_DAy.

Figure 37-7. SD Card Bus Connections with Two Slots



Note: When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK,MCCDA to HSMCIx_CDA, MCDAy to HSMCIx_DAy, MCCDB to HSMCIx_CDB, MCDBy to HSMCIx_DBy.

Figure 37-8. Mixing High Speed MultiMedia and SD Memory Cards with Two Slots



Note: When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCDAy to HSMCIx_DAy, MCCDB to HSMCIx_CDB, MCDBy to HSMCIx_DBy.

When the HSMCI is configured to operate with SD memory cards, the width of the data bus can be selected in the HSMCI_SDCR. Clearing the SDCBUS bit in this register means that the width is one bit; setting it means that the width is four bits. In the case of High Speed MultiMedia cards, only the data line 0 is used. The other data lines can be used as independent PIOs.

37.8 High Speed MultiMedia Card Operations

After a power-on reset, the cards are initialized by a special message-based High Speed MultiMedia Card bus protocol. Each message is represented by one of the following tokens:

- Command—A command is a token that starts an operation. A command is sent from the host either to a single card (addressed command) or to all connected cards (broadcast command). A command is transferred serially on the CMD line.
- Response—A response is a token which is sent from an addressed card or (synchronously) from all connected cards to the host as an answer to a previously received command. A response is transferred serially on the CMD line.
- Data—Data can be transferred from the card to the host or vice versa. Data is transferred via the data line.

Card addressing is implemented using a session address assigned during the initialization phase by the bus controller to all currently connected cards. Their unique CID number identifies individual cards.

The structure of commands, responses and data blocks is described in the High Speed MultiMedia Card System Specification. Refer to Table 37-7.

High Speed MultiMedia Card bus data transfers are composed of these tokens.

There are different types of operations. Addressed operations always contain a command and a response token. In addition, some operations have a data token; the others transfer their information directly within the command or response structure. In this case, no data token is present in an operation. The bits on the DAT and the CMD lines are transferred synchronous to the clock HSMCI clock.

Two types of data transfer commands are defined:

- Sequential commands—These commands initiate a continuous data stream. They are terminated only when a stop command follows on the CMD line. This mode reduces the command overhead to an absolute minimum.
- Block-oriented commands—These commands send a data block succeeded by CRC bits.

Both read and write operations allow either single or multiple block transmission. A multiple block transmission is terminated when a stop command follows on the CMD line similarly to the sequential read or when a multiple block transmission has a predefined block count (refer to Section 37.8.2 “Data Transfer Operation”).

The HSMCI provides a set of registers to perform the entire range of High Speed MultiMedia Card operations.

37.8.1 Command - Response Operation

After reset, the HSMCI is disabled and becomes valid after setting the MCIEN bit in the HSMCI_CR.

The PWSEN bit saves power by dividing the HSMCI clock by $2^{PWSDIV} + 1$ when the bus is inactive.

The two bits, RDPROOF and WRPROOF in the HSMCI Mode Register (HSMCI_MR) allow stopping the HSMCI clock during read or write access if the internal FIFO is full. This will guarantee data integrity, not bandwidth.

All the timings for High Speed MultiMedia Card are defined in the High Speed MultiMedia Card System Specification.

The two bus modes (open drain and push/pull) needed to process all the operations are defined in the HSMCI Command Register (HSMCI_CMDR). The HSMCI_CMDR allows a command to be carried out.

For example, to perform an ALL_SEND_CID command:

	Host Command					N_{ID} Cycles			Response			High Impedance State		
CMD	S	T	Content	CRC	E	Z	******	Z	S	T	CID Content	Z	Z	Z

The command ALL_SEND_CID and the fields and values for the HSMCI_CMDR are described in Table 37-7 and Table 37-8.

Table 37-7. ALL_SEND_CID Command Description

CMD Index	Type	Argument	Response	Abbreviation	Command Description
CMD2	bcr(1)	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all cards to send their CID numbers on the CMD line

Note: 1. bcr means broadcast command with response.

Table 37-8. Fields and Values for HSMCI_CMDR

Field	Value
CMDNB (command number)	2 (CMD2)
RSPTYP (response type)	2 (R2: 136 bits response)
SPCMD (special command)	0 (not a special command)
OPCMD (open drain command)	1
MAXLAT (max latency for command to response)	0 (NID cycles ==> 5 cycles)
TRCMD (transfer command)	0 (No transfer)

Table 37-8. Fields and Values for HSMCI_CMDR (Continued)

Field	Value
TRDIR (transfer direction)	X (available only in transfer command)
TRTYP (transfer type)	X (available only in transfer command)
IOSPCMD (SDIO special command)	0 (not a special command)

The HSMCI_ARGR contains the argument field of the command.

To send a command, the user must perform the following steps:

- Fill the argument register (HSMCI_ARGR) with the command argument.
- Set the command register (HSMCI_CMDR) (refer to Table 37-8).

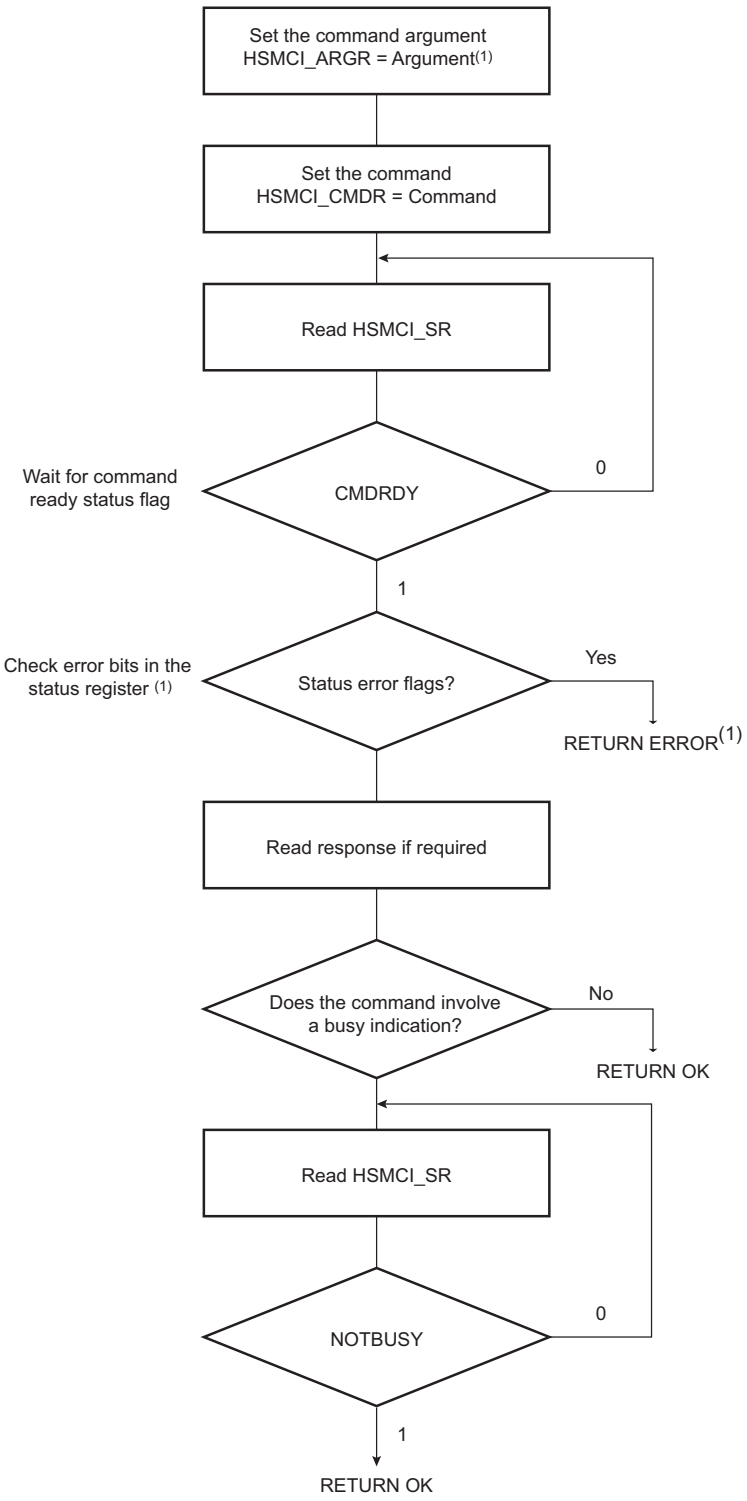
The command is sent immediately after writing the command register.

While the card maintains a busy indication (at the end of a STOP_TRANSMISSION command CMD12, for example), a new command shall not be sent. The NOTBUSY flag in the Status Register (HSMCI_SR) is asserted when the card releases the busy indication.

If the command requires a response, it can be read in the HSMCI Response Register (HSMCI_RSPR). The response size can be from 48 bits up to 136 bits depending on the command. The HSMCI embeds an error detection to prevent any corrupted data during the transfer.

The following flowchart shows how to send a command to the card and read the response if needed. In this example, the status register bits are polled but setting the appropriate bits in the HSMCI Interrupt Enable Register (HSMCI_IER) allows using an interrupt method.

Figure 37-9. Command/Response Functional Flow Diagram



Note: If the command is SEND_OP_COND, the CRC error flag is always present (refer to R3 response in the High Speed MultiMedia Card specification).

37.8.2 Data Transfer Operation

The High Speed MultiMedia Card allows several read/write operations (single block, multiple blocks, stream, etc.). These kinds of transfer can be selected setting the Transfer Type (TRTYP) field in the HSMCI Command Register (HSMCI_CMDR).

In all cases, the block length (BLKLEN field) must be defined either in the HSMCI Mode Register (HSMCI_MR) or in the HSMCI Block Register (HSMCI_BLKR). This field determines the size of the data block.

Consequent to MMC Specification 3.1, two types of multiple block read (or write) transactions are defined (the host can use either one at any time):

- Open-ended/Infinite Multiple block read (or write):

  The number of blocks for the read (or write) multiple block operation is not defined. The card will continuously transfer (or program) data blocks until a stop transmission command is received.

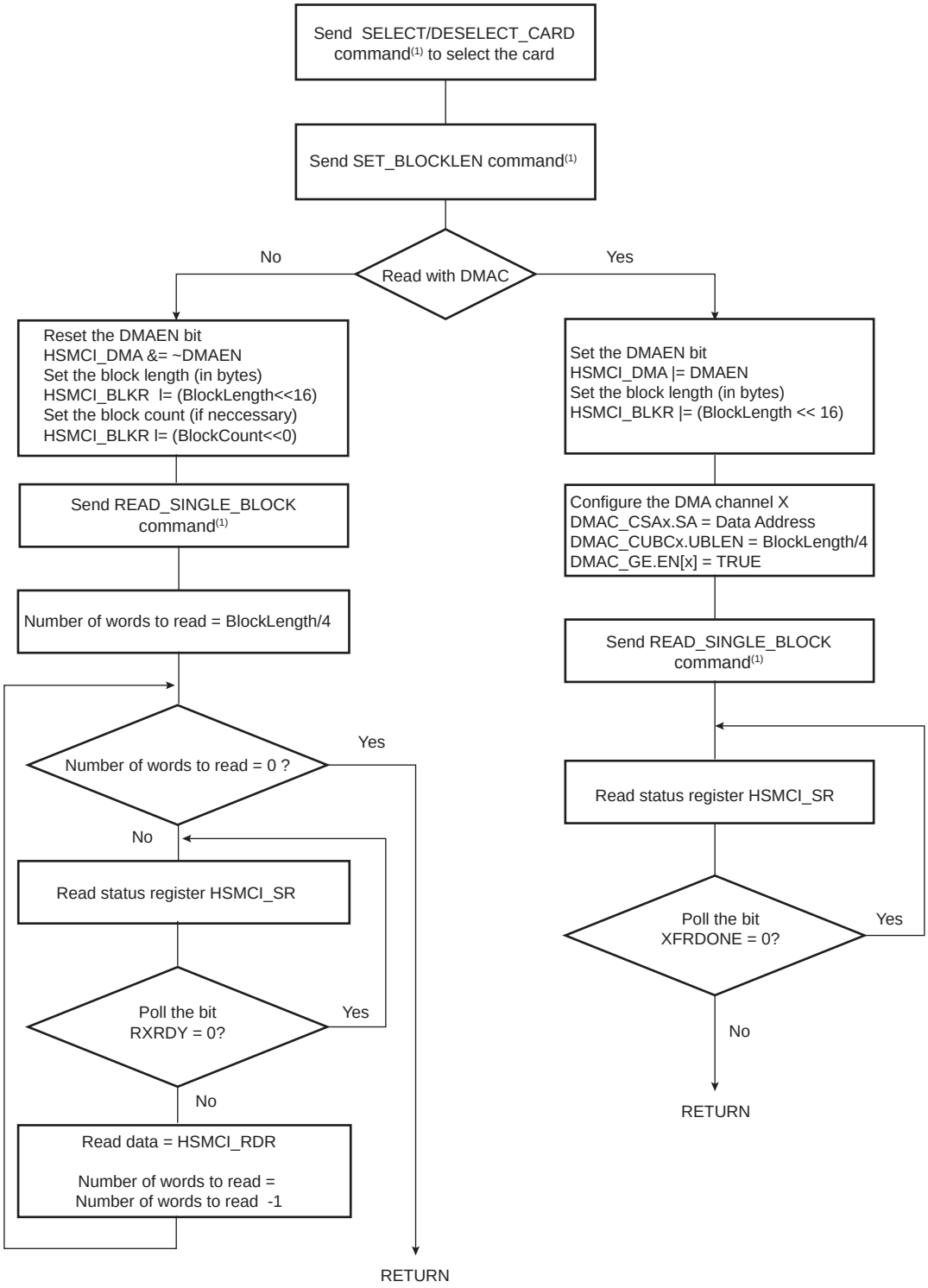
- Multiple block read (or write) with predefined block count (since version 3.1 and higher):

  The card will transfer (or program) the requested number of data blocks and terminate the transaction. The stop command is not required at the end of this type of multiple block read (or write), unless terminated with an error. In order to start a multiple block read (or write) with predefined block count, the host must correctly program the HSMCI Block Register (HSMCI_BLKR). Otherwise the card will start an open-ended multiple block read. The BCNT field of the HSMCI_BLKR defines the number of blocks to transfer (from 1 to 65535 blocks). Programming the value 0 in the BCNT field corresponds to an infinite block transfer.

37.8.3 Read Operation

The following flowchart (Figure 37-10) shows how to read a single block with or without use of DMAC facilities. In this example, a polling method is used to wait for the end of read. Similarly, the user can configure the HSMCI Interrupt Enable Register (HSMCI_IER) to trigger an interrupt at the end of read.

Figure 37-10. Read Functional Flow Diagram



Note: 1. It is assumed that this command has been correctly sent (refer to Figure 37-9).

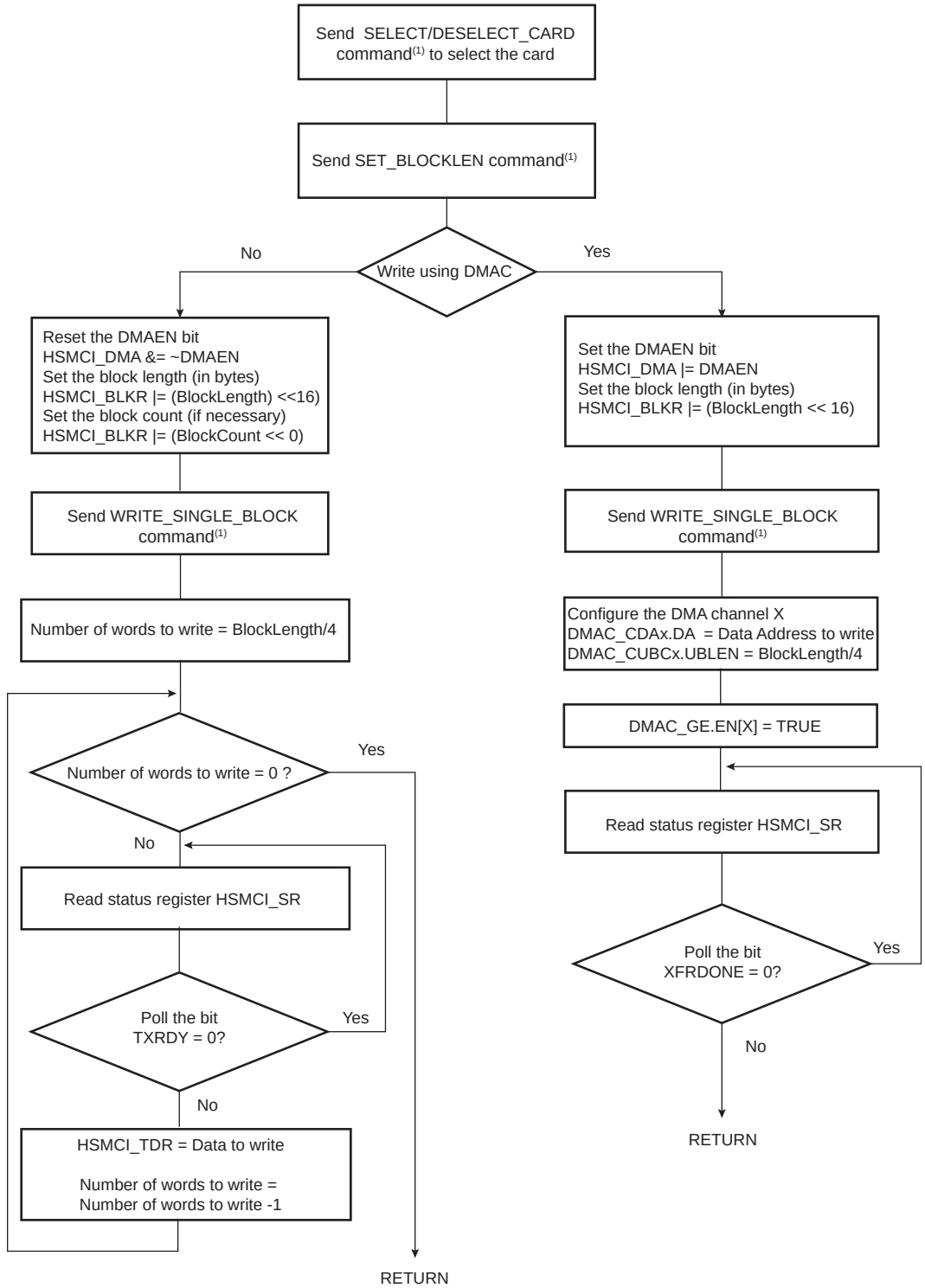
37.8.4 Write Operation

In write operation, the HSMCI Mode Register (HSMCI_MR) is used to define the padding value when writing non-multiple block size. If the bit PADV is 0, then 0x00 value is used when padding data, otherwise 0xFF is used.

If set, the bit DMAEN in the HSMCI DMA Configuration Register (HSMCI_DMA) enables DMA transfer.

The flowchart in Figure 37-11 shows how to write a single block with or without use of DMA facilities. Polling or interrupt method can be used to wait for the end of write according to the contents of the HSMCI Interrupt Mask Register (HSMCI_IMR).

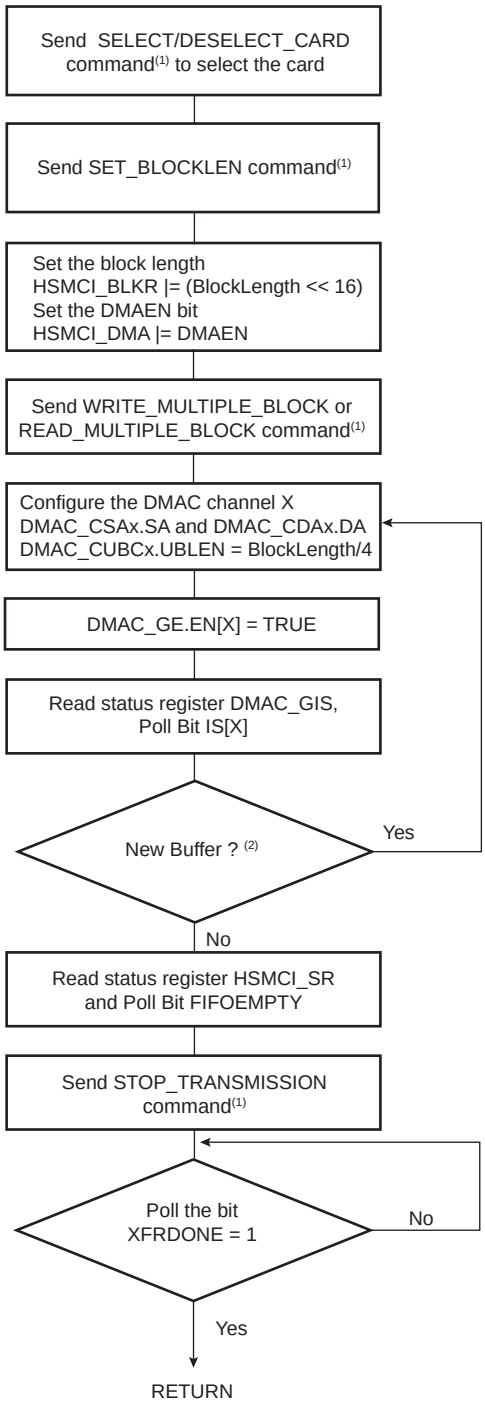
Figure 37-11. Write Functional Flow Diagram



Note: 1. It is assumed that this command has been correctly sent (refer to Figure 37-9).

The flowchart in Figure 37-12 shows how to manage read multiple block and write multiple block transfers with the DMA Controller. Polling or interrupt method can be used to wait for the end of write according to the contents of the HSMCI_IMR.

Figure 37-12. Read and Write Multiple Block



Notes: 1. It is assumed that this command has been correctly sent (refer to Figure 37-9).
2. Handle errors reported in HSMCI_SR.

37.8.5 WRITE_SINGLE_BLOCK/WRITE_MULTIPLE_BLOCK Operation using DMA Controller

1. Wait until the current command execution has successfully terminated.
    1. Check that CMDRDY and NOTBUSY fields are asserted in HSMCI_SR
2. Program the block length in the card. This value defines the value *block_length.*
3. Program the block length in the HSMCI Configuration Register with *block_length* value.
4. Configure the fields of the HSMCI_MR as follows:
    1. Program FBYTE to one when the transfer is not multiple of 4, zero otherwise.
5. Issue a WRITE_SINGLE_BLOCK command writing HSMCI_ARGR then HSMCI_CMDR.
6. Program the DMA Controller.
    1. Read the Channel Status Register to choose an available (disabled) channel.
    2. Clear any pending interrupts on the channel from the previous DMAC transfer by reading the DMAC_CISx register.
    3. Program the channel registers.
    4. The DMAC_CSAx register for Channel x must be set to the location of the source data.
    5. The DMAC_CDAx register for Channel x must be set with the starting address of the HSMCI_FIFO address.
    6. Configure the fields of DMAC_CCx of Channel x as follows:

    –DWIDTH is set to WORD when the transfer is multiple of 4, otherwise it is set to BYTE

    –CSIZE must be set according to the value of HSMCI_DMA.CHKSIZE.

    7. Configure the fields of DMAC_CUBCx for Channel x as follows:

    –UBLEN is programmed with *block_length/4* when the transfer length is multiple of 4, *block_length* otherwise.

    8. Enable Channel x, writing one to DMAC_GE.EN[x]. The DMAC is ready and waiting for request.
7. Wait for XFRDONE in the HSMCI_SR.

37.8.6 READ_SINGLE_BLOCK/READ_MULTIPLE_BLOCK Operation using DMA Controller

1. Wait until the current command execution has successfully completed.
    1. Check that CMDRDY and NOTBUSY are asserted in HSMCI_SR.
2. Program the block length in the card. This value defines the value *block_length.*
3. Program the block length in the HSMCI Configuration Register with *block_length* value.
4. Set RDPROOF bit in HSMCI_MR to avoid overflow.
5. Configure the fields of the HSMCI_MR as follows:
    1. Program FBYTE to one when the transfer is not multiple of 4, zero otherwise.
6. Issue a READ_SINGLE_BLOCK/WRITE_MULTIPLE_BLOCK command.
7. Program the DMA controller.
    1. Read the Channel Status Register to choose an available (disabled) channel.
    2. Clear any pending interrupts on the channel from the previous DMA transfer by reading the DMAC_CISx register.
    3. Program the channel registers.
    4. The DMAC_CSAx register for Channel x must be set with the starting address of the HSMCI_FIFO address.
    5. The DMAC_CDAx register for Channel x must be word aligned.
    6. Configure the fields of DMAC_CCx for Channel x as follows:

    –DWIDTH is set to WORD when the length is a multiple of 4, otherwise it is set to BYTE.

    –CSIZE must be set according to the value of HSMCI_DMA.CHKSIZE.

7. Configure the fields of the DMAC_CUBCx register of Channel x as follows:

–UBLEN is programmed with *block_length/4* when the transfer length is multiple of 4, *block_length* otherwise.

8. Enable Channel x, writing one to DMAC_GE.EN[x]. The DMAC is ready and waiting for request.

8. Wait for XFRDONE in the HSMCI_SR.

37.9 SD/SDIO Card Operation

The High Speed MultiMedia Card Interface allows processing of SD Memory (Secure Digital Memory Card) and SDIO (SD Input Output) Card commands.

SD/SDIO cards are based on the MultiMedia Card (MMC) format, but are physically slightly thicker and feature higher data transfer rates, a lock switch on the side to prevent accidental overwriting and security features. The physical form factor, pin assignment and data transfer protocol are forward-compatible with the High Speed MultiMedia Card with some additions. SD slots can actually be used for more than flash memory cards. Devices that support SDIO can use small devices designed for the SD form factor, such as GPS receivers, Wi-Fi or Bluetooth adapters, modems, barcode readers, IrDA adapters, FM radio tuners, RFID readers, digital cameras and more.

SD/SDIO is covered by numerous patents and trademarks, and licensing is only available through the Secure Digital Card Association.

The SD/SDIO Card communication is based on a 9-pin interface (Clock, Command, 4 x Data and 3 x Power lines). The communication protocol is defined as a part of this specification. The main difference between the SD/SDIO Card and the High Speed MultiMedia Card is the initialization process.

The SD/SDIO Card Register (HSMCI_SDCR) allows selection of the Card Slot and the data bus width.

The SD/SDIO Card bus allows dynamic configuration of the number of data lines. After powerup, by default, the SD/SDIO Card uses only DAT0 for data transfer. After initialization, the host can change the bus width (number of active data lines).

37.9.1 SDIO Data Transfer Type

SDIO cards may transfer data in either a multi-byte (1 to 512 bytes) or an optional block format (1 to 511 blocks), while the SD memory cards are fixed in the block transfer mode. The TRTYP field in the HSMCI Command Register (HSMCI_CMDR) allows to choose between SDIO Byte or SDIO Block transfer.

The number of bytes/blocks to transfer is set through the BCNT field in the HSMCI Block Register (HSMCI_BLKR). In SDIO Block mode, the field BLKLEN must be set to the data block size while this field is not used in SDIO Byte mode.

An SDIO Card can have multiple I/O or combined I/O and memory (called Combo Card). Within a multi-function SDIO or a Combo card, there are multiple devices (I/O and memory) that share access to the SD bus. In order to allow the sharing of access to the host among multiple devices, SDIO and combo cards can implement the optional concept of suspend/resume (Refer to the SDIO Specification for more details). To send a suspend or a resume command, the host must set the SDIO Special Command field (IOSPCMD) in the HSMCI Command Register.

37.9.2 SDIO Interrupts

Each function within an SDIO or Combo card may implement interrupts (Refer to the SDIO Specification for more details). In order to allow the SDIO card to interrupt the host, an interrupt function is added to a pin on the DAT[1] line to signal the card's interrupt to the host. An SDIO interrupt on each slot can be enabled through the HSMCI Interrupt Enable Register. The SDIO interrupt is sampled regardless of the currently selected slot.

37.10 CE-ATA Operation

CE-ATA maps the streamlined ATA command set onto the MMC interface. The ATA task file is mapped onto MMC register space.

CE-ATA utilizes five MMC commands:

- GO_IDLE_STATE (CMD0): used for hard reset.
- STOP_TRANSMISSION (CMD12): causes the ATA command currently executing to be aborted.
- FAST_IO (CMD39): Used for single register access to the ATA taskfile registers, 8-bit access only.
- RW_MULTIPLE_REGISTERS (CMD60): used to issue an ATA command or to access the control/status registers.
- RW_MULTIPLE_BLOCK (CMD61): used to transfer data for an ATA command.

CE-ATA utilizes the same MMC command sequences for initialization as traditional MMC devices.

37.10.1 Executing an ATA Polling Command

1. Issue READ_DMA_EXT with RW_MULTIPLE_REGISTER (CMD60) for 8 KB of DATA.
2. Read the ATA status register until DRQ is set.
3. Issue RW_MULTIPLE_BLOCK (CMD61) to transfer DATA.
4. Read the ATA status register until DRQ && BSY are configured to 0.

37.10.2 Executing an ATA Interrupt Command

1. Issue READ_DMA_EXT with RW_MULTIPLE_REGISTER (CMD60) for 8 KB of DATA with nIEN field set to zero to enable the command completion signal in the device.
2. Issue RW_MULTIPLE_BLOCK (CMD61) to transfer DATA.
3. Wait for Completion Signal Received Interrupt.

37.10.3 Aborting an ATA Command

If the host needs to abort an ATA command prior to the completion signal it must send a special command to avoid potential collision on the command line. The SPCMD field of the HSMCI_CMDR must be set to 3 to issue the CE-ATA completion Signal Disable Command.

37.10.4 CE-ATA Error Recovery

Several methods of ATA command failure may occur, including:

- No response to an MMC command, such as RW_MULTIPLE_REGISTER (CMD60).
- CRC is invalid for an MMC command or response.
- CRC16 is invalid for an MMC data packet.
- ATA Status register reflects an error by setting the ERR bit to one.
- The command completion signal does not arrive within a host-specified timeout period.

Error conditions are expected to happen infrequently. Thus, a robust error recovery mechanism may be used for each error event. The recommended error recovery procedure after a timeout is:

- Issue the command completion signal disable if nIEN was cleared to zero and the RW_MULTIPLE_BLOCK (CMD61) response has been received.
- Issue STOP_TRANSMISSION (CMD12) and successfully receive the R1 response.
- Issue a software reset to the CE-ATA device using FAST_IO (CMD39).

If STOP_TRANMISSION (CMD12) is successful, then the device is again ready for ATA commands. However, if the error recovery procedure does not work as expected or there is another timeout, the next step is to issue GO_IDLE_STATE (CMD0) to the device. GO_IDLE_STATE (CMD0) is a hard reset to the device and completely resets all device states.

Note that after issuing GO_IDLE_STATE (CMD0), all device initialization needs to be completed again. If the CE-ATA device completes all MMC commands correctly but fails the ATA command with the ERR bit set in the ATA Status register, no error recovery action is required. The ATA command itself failed implying that the device could not complete the action requested, however, there was no communication or protocol failure. After the device signals an error by setting the ERR bit to one in the ATA Status register, the host may attempt to retry the command.

37.11 HSMCI Boot Operation Mode

In boot operation mode, the processor can read boot data from the slave (MMC device) by keeping the CMD line low after power-on before issuing CMD1. The data can be read from either the boot area or user area, depending on register setting.

37.11.1 Boot Procedure, Processor Mode

1. Configure the HSMCI data bus width programming SDCBUS Field in the HSMCI_SDCR. The BOOT_BUS_WIDTH field located in the device Extended CSD register must be set accordingly.
2. Set the byte count to 512 bytes and the block count to the desired number of blocks, writing BLKLEN and BCNT fields of the HSMCI_BLKR.
3. Issue the Boot Operation Request command by writing to the HSMCI_CMDR with SPCMD field set to BOOTREQ, TRDIR set to READ and TRCMD set to “start data transfer”.
4. The BOOT_ACK field located in the HSMCI_CMDR must be set to one, if the BOOT_ACK field of the MMC device located in the Extended CSD register is set to one.
5. Host processor can copy boot data sequentially as soon as the RXRDY flag is asserted.
6. When Data transfer is completed, host processor shall terminate the boot stream by writing the HSMCI_CMDR with SPCMD field set to BOOTEND.

37.11.2 Boot Procedure DMA Mode

1. Configure the HSMCI data bus width by programming SDCBUS Field in the HSMCI_SDCR. The BOOT_BUS_WIDTH field in the device Extended CSD register must be set accordingly.
2. Set the byte count to 512 bytes and the block count to the desired number of blocks by writing BLKLEN and BCNT fields of the HSMCI_BLKR.
3. Enable DMA transfer in the HSMCI_DMA register.
4. Configure DMA controller, program the total amount of data to be transferred and enable the relevant channel.
5. Issue the Boot Operation Request command by writing to the HSMCI_CMDR with SPCND set to BOOTREQ, TRDIR set to READ and TRCMD set to “start data transfer”.
6. DMA controller copies the boot partition to the memory.
7. When DMA transfer is completed, host processor shall terminate the boot stream by writing the HSMCI_CMDR with SPCMD field set to BOOTEND.

37.12 HSMCI Transfer Done Timings

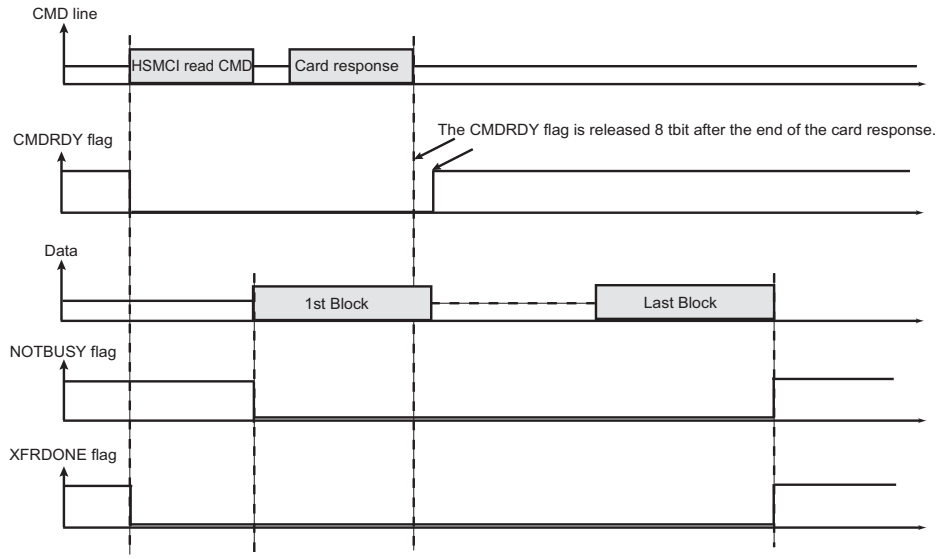
37.12.1 Definition

The XFRDONE flag in the HSMCI_SR indicates exactly when the read or write sequence is finished.

37.12.2 Read Access

During a read access, the XFRDONE flag behaves as shown in Figure 37-13.

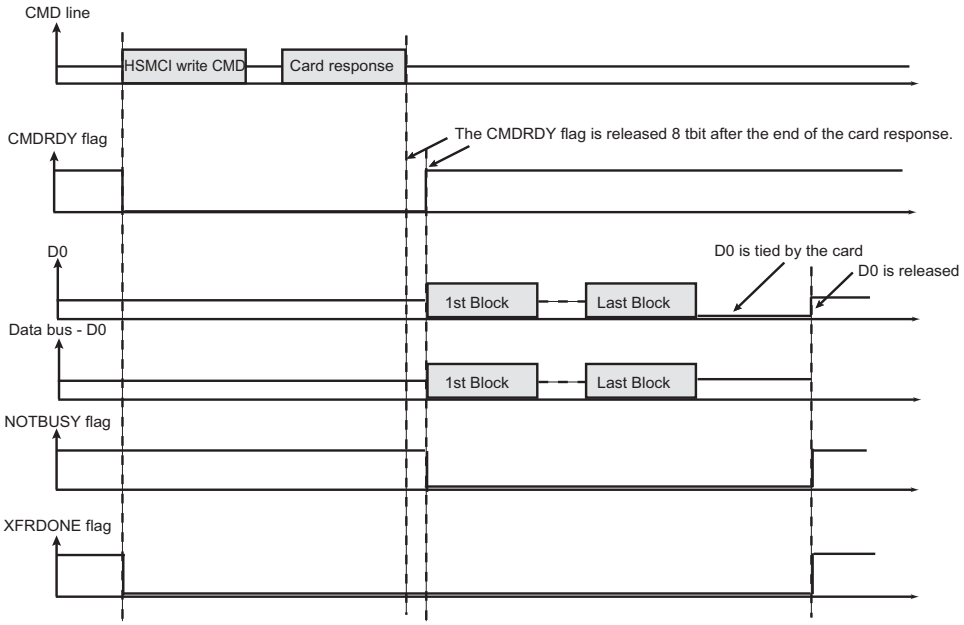
Figure 37-13. XFRDONE During a Read Access



37.12.3 Write Access

During a write access, the XFRDONE flag behaves as shown in Figure 37-14.

Figure 37-14. XFRDONE During a Write Access



37.13 Register Write Protection

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

If a write access to a write-protected register is detected, the WPVS bit in the HSMCI Write Protection Status Register (HSMCI_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 HSMCI_WPSR.

The following registers can be protected:

- HSMCI Mode Register
- HSMCI Data Timeout Register
- HSMCI SDCard/SDIO Register
- HSMCI Completion Signal Timeout Register
- HSMCI DMA Configuration Register
- HSMCI Configuration Register

37.14 High Speed MultiMedia Card Interface (HSMCI) User Interface

Table 37-9. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	HSMCI_CR	Write-only	–
0x04	Mode Register	HSMCI_MR	Read/Write	0x0
0x08	Data Timeout Register	HSMCI_DTOR	Read/Write	0x0
0x0C	SD/SDIO Card Register	HSMCI_SDCR	Read/Write	0x0
0x10	Argument Register	HSMCI_ARGR	Read/Write	0x0
0x14	Command Register	HSMCI_CMDR	Write-only	–
0x18	Block Register	HSMCI_BLKR	Read/Write	0x0
0x1C	Completion Signal Timeout Register	HSMCI_CSTOR	Read/Write	0x0
0x20	Response Register(1)	HSMCI_RSPR	Read-only	0x0
0x24	Response Register(1)	HSMCI_RSPR	Read-only	0x0
0x28	Response Register(1)	HSMCI_RSPR	Read-only	0x0
0x2C	Response Register(1)	HSMCI_RSPR	Read-only	0x0
0x30	Receive Data Register	HSMCI_RDR	Read-only	0x0
0x34	Transmit Data Register	HSMCI_TDR	Write-only	–
0x38–0x3C	Reserved	–	–	–
0x40	Status Register	HSMCI_SR	Read-only	0xC0E5
0x44	Interrupt Enable Register	HSMCI_IER	Write-only	–
0x48	Interrupt Disable Register	HSMCI_IDR	Write-only	–
0x4C	Interrupt Mask Register	HSMCI_IMR	Read-only	0x0
0x50	DMA Configuration Register	HSMCI_DMA	Read/Write	0x0
0x54	Configuration Register	HSMCI_CFG	Read/Write	0x0
0x58–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	HSMCI_WPMR	Read/Write	0x0
0xE8	Write Protection Status Register	HSMCI_WPSR	Read-only	0x0
0xEC–0xFC	Reserved	–	–	–
0x100–0x1FC	Reserved	–	–	–
0x200	FIFO Memory Aperture0	HSMCI_FIFO0	Read/Write	0x0
...	...	...	...	...
0x5FC	FIFO Memory Aperture255	HSMCI_FIFO255	Read/Write	0x0

Notes: 1. The Response Register can be read by N accesses at the same HSMCI_RSPR or at consecutive addresses (0x20 to 0x2C). N depends on the size of the response.

37.14.1 HSMCI Control Register

Name: HSMCI_CR

Address: 0xF8000000 (0), 0xFC000000 (1)

Access: Write-only

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
SWRST	–	–	–	PWSDIS	PWSEN	MCIDIS	MCIEN

- **MCIEN: Multi-Media Interface Enable**

0: No effect.

1: Enables the Multi-Media Interface if MCDIS is 0.

- **MCIDIS: Multi-Media Interface Disable**

0: No effect.

1: Disables the Multi-Media Interface.

- **PWSEN: Power Save Mode Enable**

0: No effect.

1: Enables the Power Saving Mode if PWSDIS is 0.

WARNING: Before enabling this mode, the user must set a value different from 0 in the PWSDIV field of the HSMCI_MR.

- **PWSDIS: Power Save Mode Disable**

0: No effect.

1: Disables the Power Saving Mode.

- **SWRST: Software Reset**

0: No effect.

1: Resets the HSMCI. A software triggered hardware reset of the HSMCI is performed.

37.14.2 HSMCI Mode Register

Name: HSMCI_MR

Address: 0xF8000004 (0), 0xFC000004 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	CLKODD
15	14	13	12	11	10	9	8
–	PADV	FBYTE	WRPROOF	RDPROOF	PWSDIV		
7	6	5	4	3	2	1	0
CLKDIV							

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

- **CLKDIV: Clock Divider**

High Speed MultiMedia Card Interface clock (MCCK or HSMCI_CK) is Master Clock (MCK) divided by 2 × CLKDIV + CLKODD + 2.

- **PWSDIV: Power Saving Divider**

High Speed MultiMedia Card Interface clock is divided by $2^{(PWSDIV)}$ + 1 when entering Power Saving Mode.

WARNING: This value must be different from 0 before enabling the Power Save Mode in the HSMCI_CR (PWSEN bit).

- **RDPROOF: Read Proof Enable**

Enabling Read Proof allows to stop the HSMCI Clock during read access if the internal FIFO is full. This will guarantee data integrity, not bandwidth.

0: Disables Read Proof.

1: Enables Read Proof.

- **WRPROOF: Write Proof Enable**

Enabling Write Proof allows to stop the HSMCI Clock during write access if the internal FIFO is full. This will guarantee data integrity, not bandwidth.

0: Disables Write Proof.

1: Enables Write Proof.

- **FBYTE: Force Byte Transfer**

Enabling Force Byte Transfer allow byte transfers, so that transfer of blocks with a size different from modulo 4 can be supported.

WARNING: BLKLEN value depends on FBYTE.

0: Disables Force Byte Transfer.

1: Enables Force Byte Transfer.

- **PADV: Padding Value**

0: 0x00 value is used when padding data in write transfer.

1: 0xFF value is used when padding data in write transfer.

PADV may be only in manual transfer.

- **CLKODD: Clock divider is odd**

This bit is the least significant bit of the clock divider and indicates the clock divider parity.

37.14.3 HSMCI Data Timeout Register

Name: HSMCI_DTOR

Address: 0xF8000008 (0), 0xFC000008 (1)

Access: Read/Write

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
–	DTOMUL			DTOCYC			

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

- **DTOCYC: Data Timeout Cycle Number**

This field determines the maximum number of Master Clock cycles that the HSMCI waits between two data block transfers. It equals (DTOCYC x Multiplier).

- **DTOMUL: Data Timeout Multiplier**

Value	Name	Description
0	1	DTOCYC
1	16	DTOCYC x 16
2	128	DTOCYC x 128
3	256	DTOCYC x 256
4	1024	DTOCYC x 1024
5	4096	DTOCYC x 4096
6	65536	DTOCYC x 65536
7	1048576	DTOCYC x 1048576

If the data timeout set by DTOCYC and DTOMUL has been exceeded, the Data TimeOut Error flag (DTOE) in the HSMCI Status Register (HSMCI_SR) rises.

37.14.4 HSMCI SDCard/SDIO Register

Name: HSMCI_SDCR

Address: 0xF800000C (0), 0xFC00000C (1)

Access: Read/Write

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
SDCBUS		–	–	–	–	SDCSEL	

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

- **SDCSEL: SDCard/SDIO Slot**

Value	Name	Description
0	SLOTA	Slot A is selected.
1	SLOTB	Slot B is selected.
2	SLOTC	Reserved
3	SLOTD	Reserved

- **SDCBUS: SDCard/SDIO Bus Width**

Value	Name	Description
0	1	1 bit
1	–	Reserved
2	4	4 bits
3	8	8 bits

37.14.5 HSMCI Argument Register

Name: HSMCI_ARGR

Address: 0xF8000010 (0), 0xFC000010 (1)

Access: Read/Write

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

- **ARG: Command Argument**

37.14.6 HSMCI Command Register

Name: HSMCI_CMDR

Address: 0xF8000014 (0), 0xFC000014 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	BOOT_ACK	ATACS	IOSPCMD	
23	22	21	20	19	18	17	16
–	–	TRTYP			TRDIR	TRCMD	
15	14	13	12	11	10	9	8
–	–	–	MAXLAT	OPDCMD	SPCMD		
7	6	5	4	3	2	1	0
RSPTYP		CMDNB					

This register is write-protected while CMDRDY is 0 in HSMCI_SR. If an Interrupt command is sent, this register is only writable by an interrupt response (field SPCMD). This means that the current command execution cannot be interrupted or modified.

- **CMDNB: Command Number**

This is the command index.

- **RSPTYP: Response Type**

Value	Name	Description
0	NORESP	No response
1	48_BIT	48-bit response
2	136_BIT	136-bit response
3	R1B	R1b response type

- **SPCMD: Special Command**

Value	Name	Description
0	STD	Not a special CMD.
1	INIT	Initialization CMD: 74 clock cycles for initialization sequence.
2	SYNC	Synchronized CMD: Wait for the end of the current data block transfer before sending the pending command.
3	CE_ATA	CE-ATA Completion Signal disable Command. The host cancels the ability for the device to return a command completion signal on the command line.
4	IT_CMD	Interrupt command: Corresponds to the Interrupt Mode (CMD40).
5	IT_RESP	Interrupt response: Corresponds to the Interrupt Mode (CMD40).
6	BOR	Boot Operation Request. Start a boot operation mode, the host processor can read boot data from the MMC device directly.
7	EBO	End Boot Operation. This command allows the host processor to terminate the boot operation mode.

- **OPDCMD: Open Drain Command**

0 (PUSHPULL): Push pull command.

1 (OPENDRAIN): Open drain command.

- **MAXLAT: Max Latency for Command to Response**

0 (5): 5-cycle max latency.

1 (64): 64-cycle max latency.

- **TRCMD: Transfer Command**

Value	Name	Description
0	NO_DATA	No data transfer
1	START_DATA	Start data transfer
2	STOP_DATA	Stop data transfer
3	–	Reserved

- **TRDIR: Transfer Direction**

0 (WRITE): Write.

1 (READ): Read.

- **TRTYP: Transfer Type**

Value	Name	Description
0	SINGLE	MMC/SD Card Single Block
1	MULTIPLE	MMC/SD Card Multiple Block
2	STREAM	MMC Stream
4	BYTE	SDIO Byte
5	BLOCK	SDIO Block

- **IOSPCMD: SDIO Special Command**

Value	Name	Description
0	STD	Not an SDIO Special Command
1	SUSPEND	SDIO Suspend Command
2	RESUME	SDIO Resume Command

- **ATACS: ATA with Command Completion Signal**

0 (NORMAL): Normal operation mode.

1 (COMPLETION): This bit indicates that a completion signal is expected within a programmed amount of time (HSMCI_CSTOR).

- **BOOT_ACK: Boot Operation Acknowledge**

The master can choose to receive the boot acknowledge from the slave when a Boot Request command is issued. When set to one this field indicates that a Boot acknowledge is expected within a programmable amount of time defined with DTOMUL and DTOCYC fields located in the HSMCI_DTOR. If the acknowledge pattern is not received then an acknowledge timeout error is raised. If the acknowledge pattern is corrupted then an acknowledge pattern error is set.

37.14.7 HSMCI Block Register

Name: HSMCI_BLKR

Address: 0xF8000018 (0), 0xFC000018 (1)

Access: Read/Write

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

- **BCNT: MMC/SDIO Block Count - SDIO Byte Count**

This field determines the number of data byte(s) or block(s) to transfer.

The transfer data type and the authorized values for BCNT field are determined by the TRTYP field in the HSMCI Command Register (HSMCI_CMDR).

When TRTYP = 1 (MMC/SDCARD Multiple Block), BCNT can be programmed from 1 to 65535, 0 corresponds to an infinite block transfer.

When TRTYP = 4 (SDIO Byte), BCNT can be programmed from 1 to 511, 0 corresponds to 512-byte transfer. Values in range 512 to 65536 are forbidden.

When TRTYP = 5 (SDIO Block), BCNT can be programmed from 1 to 511, 0 corresponds to an infinite block transfer. Values in range 512 to 65536 are forbidden.

WARNING: In SDIO Byte and Block modes (TRTYP = 4 or 5), writing the 7 last bits of BCNT field with a value which differs from 0 is forbidden and may lead to unpredictable results.

- **BLKLEN: Data Block Length**

This field determines the size of the data block.

Bits 16 and 17 must be configured to 0 if FBYTE is disabled.

Note: In SDIO Byte mode, BLKLEN field is not used.

37.14.8 HSMCI Completion Signal Timeout Register

Name: HSMCI_CSTOR

Address: 0xF800001C (0), 0xFC00001C (1)

Access: Read/Write

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
–	CSTOMUL			CSTOCYC			

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

- **CSTOCYC: Completion Signal Timeout Cycle Number**

This field determines the maximum number of Master Clock cycles that the HSMCI waits between two data block transfers. Its value is calculated by (CSTOCYC x Multiplier).

- **CSTOMUL: Completion Signal Timeout Multiplier**

This field determines the maximum number of Master Clock cycles that the HSMCI waits between two data block transfers. Its value is calculated by (CSTOCYC x Multiplier).

These fields determine the maximum number of Master Clock cycles that the HSMCI waits between the end of the data transfer and the assertion of the completion signal. The data transfer comprises data phase and the optional busy phase. If a non-DATA ATA command is issued, the HSMCI starts waiting immediately after the end of the response until the completion signal.

Multiplier is defined by CSTOMUL as shown in the following table:

Value	Name	Description
0	1	CSTOCYC x 1
1	16	CSTOCYC x 16
2	128	CSTOCYC x 128
3	256	CSTOCYC x 256
4	1024	CSTOCYC x 1024
5	4096	CSTOCYC x 4096
6	65536	CSTOCYC x 65536
7	1048576	CSTOCYC x 1048576

If the data timeout set by CSTOCYC and CSTOMUL has been exceeded, the Completion Signal TimeOut Error flag (CSTOE) in the HSMCI Status Register (HSMCI_SR) rises.

37.14.9 HSMCI Response Register

Name: HSMCI_RSPR

Address: 0xF8000020 (0), 0xFC000020 (1)

Access: Read-only

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

- **RSP: Response**

Note: 1. The response register can be read by N accesses at the same HSMCI_RSPR or at consecutive addresses (0x20 to 0x2C). N depends on the size of the response.

37.14.10HSMCI Receive Data Register

Name: HSMCI_RDR

Address: 0xF8000030 (0), 0xFC000030 (1)

Access: Read-only

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

- **DATA: Data to Read**

37.14.11 HSMCI Transmit Data Register

Name: HSMCI_TDR

Address: 0xF8000034 (0), 0xFC000034 (1)

Access: Write-only

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

- **DATA: Data to Write**

37.14.12HSMCI Status Register

Name: HSMCI_SR

Address: 0xF8000040 (0), 0xFC000040 (1)

Access: Read-only

31	30	29	28	27	26	25	24
UNRE	OVRE	ACKRCVE	ACKRCV	XFRDONE	FIFOEMPTY	–	BLKOVRE
23	22	21	20	19	18	17	16
CSTOE	DTOE	DCRCE	RTOE	RENDE	RCRCE	RDIRE	RINDE
15	14	13	12	11	10	9	8
–	–	CSRCV	SDIOWAIT	–	–	SDIOIRQB	SDIOIRQA
7	6	5	4	3	2	1	0
–	–	NOTBUSY	DTIP	BLKE	TXRDY	RXRDY	CMDRDY

- **CMDRDY: Command Ready (cleared by writing in HSMCI_CMDR)**

0: A command is in progress.

1: The last command has been sent.

- **RXRDY: Receiver Ready (cleared by reading HSMCI_RDR)**

0: Data has not yet been received since the last read of HSMCI_RDR.

1: Data has been received since the last read of HSMCI_RDR.

- **TXRDY: Transmit Ready (cleared by writing in HSMCI_TDR)**

0: The last data written in HSMCI_TDR has not yet been transferred in the Shift Register.

1: The last data written in HSMCI_TDR has been transferred in the Shift Register.

- **BLKE: Data Block Ended (cleared on read)**

This flag must be used only for Write Operations.

0: A data block transfer is not yet finished.

1: A data block transfer has ended, including the CRC16 Status transmission. The flag is set for each transmitted CRC Status.

Refer to the MMC or SD Specification for more details concerning the CRC Status.

- **DTIP: Data Transfer in Progress (cleared at the end of CRC16 calculation)**

0: No data transfer in progress.

1: The current data transfer is still in progress, including CRC16 calculation.

- **NOTBUSY: HSMCI Not Busy**

A block write operation uses a simple busy signalling of the write operation duration on the data (DAT0) line: during a data transfer block, if the card does not have a free data receive buffer, the card indicates this condition by pulling down the data line (DAT0) to LOW. The card stops pulling down the data line as soon as at least one receive buffer for the defined data transfer block length becomes free.

Refer to the MMC or SD Specification for more details concerning the busy behavior.

For all the read operations, the NOTBUSY flag is cleared at the end of the host command.
For the Infinite Read Multiple Blocks, the NOTBUSY flag is set at the end of the STOP_TRANSMISSION host command (CMD12).

For the Single Block Reads, the NOTBUSY flag is set at the end of the data read block.

For the Multiple Block Reads with predefined block count, the NOTBUSY flag is set at the end of the last received data block.

The NOTBUSY flag allows to deal with these different states.

0: The HSMCI is not ready for new data transfer. Cleared at the end of the card response.

1: The HSMCI is ready for new data transfer. Set when the busy state on the data line has ended. This corresponds to a free internal data receive buffer of the card.

- **SDIOIRQA: SDIO Interrupt for Slot A (cleared on read)**

0: No interrupt detected on SDIO Slot A.

1: An SDIO Interrupt on Slot A occurred.

- **SDIOIRQB: SDIO Interrupt for Slot B (cleared on read)**

0: No interrupt detected on SDIO Slot B.

1: An SDIO Interrupt on Slot B occurred.

- **SDIOWAIT: SDIO Read Wait Operation Status**

0: Normal Bus operation.

1: The data bus has entered IO wait state.

- **CSRCV: CE-ATA Completion Signal Received (cleared on read)**

0: No completion signal received since last status read operation.

1: The device has issued a command completion signal on the command line.

- **RINDE: Response Index Error (cleared by writing in HSMCI_CMDR)**

0: No error.

1: A mismatch is detected between the command index sent and the response index received.

- **RDIRE: Response Direction Error (cleared by writing in HSMCI_CMDR)**

0: No error.

1: The direction bit from card to host in the response has not been detected.

- **RCRCE: Response CRC Error (cleared by writing in HSMCI_CMDR)**

0: No error.

1: A CRC7 error has been detected in the response.

- **RENDE: Response End Bit Error (cleared by writing in HSMCI_CMDR)**

0: No error.

1: The end bit of the response has not been detected.

- **RTOE: Response TimeOut Error (cleared by writing in HSMCI_CMDR)**

0: No error.

1: The response timeout set by MAXLAT in the HSMCI_CMDR has been exceeded.

- **DCRCE: Data CRC Error (cleared on read)**

0: No error.

1: A CRC16 error has been detected in the last data block.

- **DTOE: Data TimeOut Error (cleared on read)**

0: No error.

1: The data timeout set by DTOCYC and DTOMUL in HSMCI_DTOR has been exceeded.

- **CSTOE: Completion Signal TimeOut Error (cleared on read)**

0: No error.

1: The completion signal timeout set by CSTOCYC and CSTOMUL in HSMCI_CSTOR has been exceeded.

- **BLKOVRE: DMA Block Overrun Error (cleared on read)**

0: No error.

1: A new block of data is received and the DMA controller has not started to move the current pending block, a block overrun is raised.

- **FIFOEMPTY: FIFO empty flag**

0: FIFO contains at least one byte.

1: FIFO is empty.

- **XFRDONE: Transfer Done flag**

0: A transfer is in progress.

1: Command Register is ready to operate and the data bus is in the idle state.

- **ACKRCV: Boot Operation Acknowledge Received (cleared on read)**

0: No Boot acknowledge received since the last read of the HSMCI_SR.

1: A Boot acknowledge signal has been received since the last read of HSMCI_SR.

- **ACKRCVE: Boot Operation Acknowledge Error (cleared on read)**

0: No boot operation error since the last read of HSMCI_SR

1: Corrupted Boot Acknowledge signal received since the last read of HSMCI_SR.

- **OVRE: Overrun (if FERRCTRL = 1, cleared by writing in HSMCI_CMDR or cleared on read if FERRCTRL = 0)**

0: No error.

1: At least one 8-bit received data has been lost (not read).

If FERRCTRL = 1 in HSMCI_CFG, OVRE is cleared on read.

If FERRCTRL = 0 in HSMCI_CFG, OVRE is cleared by writing HSMCI_CMDR.

- **UNRE: Underrun (if FERRCTRL = 1, cleared by writing in HSMCI_CMDR or cleared on read if FERRCTRL = 0)**

0: No error.

1: At least one 8-bit data has been sent without valid information (not written).

If FERRCTRL = 1 in HSMCI_CFG, OVRE is cleared on read.

If FERRCTRL = 0 in HSMCI_CFG, OVRE is cleared by writing HSMCI_CMDR.

37.14.13HSMCI Interrupt Enable Register

Name: HSMCI_IER

Address: 0xF8000044 (0), 0xFC000044 (1)

Access: Write-only

31	30	29	28	27	26	25	24
UNRE	OVRE	ACKRCVE	ACKRCV	XFRDONE	FIFOEMPTY	–	BLKOVRE
23	22	21	20	19	18	17	16
CSTOE	DTOE	DCRCE	RTOE	RENDE	RCRCE	RDIRE	RINDE
15	14	13	12	11	10	9	8
–	–	CSRCV	SDIOWAIT	–	–	SDIOIRQB	SDIOIRQA
7	6	5	4	3	2	1	0
–	–	NOTBUSY	DTIP	BLKE	TXRDY	RXRDY	CMDRDY

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

0: No effect.

1: Enables the corresponding interrupt.

- **CMDRDY: Command Ready Interrupt Enable**
- **RXRDY: Receiver Ready Interrupt Enable**
- **TXRDY: Transmit Ready Interrupt Enable**
- **BLKE: Data Block Ended Interrupt Enable**
- **DTIP: Data Transfer in Progress Interrupt Enable**
- **NOTBUSY: Data Not Busy Interrupt Enable**
- **SDIOIRQA: SDIO Interrupt for Slot A Interrupt Enable**
- **SDIOIRQB: SDIO Interrupt for Slot B Interrupt Enable**
- **SDIOWAIT: SDIO Read Wait Operation Status Interrupt Enable**
- **CSRCV: Completion Signal Received Interrupt Enable**
- **RINDE: Response Index Error Interrupt Enable**
- **RDIRE: Response Direction Error Interrupt Enable**
- **RCRCE: Response CRC Error Interrupt Enable**
- **RENDE: Response End Bit Error Interrupt Enable**
- **RTOE: Response TimeOut Error Interrupt Enable**
- **DCRCE: Data CRC Error Interrupt Enable**
- **DTOE: Data TimeOut Error Interrupt Enable**

- **CSTOE: Completion Signal Timeout Error Interrupt Enable**
- **BLKOVRE: DMA Block Overrun Error Interrupt Enable**
- **FIFOEMPTY: FIFO empty Interrupt enable**
- **XFRDONE: Transfer Done Interrupt enable**
- **ACKRCV: Boot Acknowledge Interrupt Enable**
- **ACKRCVE: Boot Acknowledge Error Interrupt Enable**
- **OVRE: Overrun Interrupt Enable**
- **UNRE: Underrun Interrupt Enable**

37.14.14HSMCI Interrupt Disable Register

Name: HSMCI_IDR

Address: 0xF8000048 (0), 0xFC000048 (1)

Access: Write-only

31	30	29	28	27	26	25	24
UNRE	OVRE	ACKRCVE	ACKRCV	XFRDONE	FIFOEMPTY	–	BLKOVRE
23	22	21	20	19	18	17	16
CSTOE	DTOE	DCRCE	RTOE	RENDE	RCRCE	RDIRE	RINDE
15	14	13	12	11	10	9	8
–	–	CSRCV	SDIOWAIT	–	–	SDIOIRQB	SDIOIRQA
7	6	5	4	3	2	1	0
–	–	NOTBUSY	DTIP	BLKE	TXRDY	RXRDY	CMDRDY

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

0: No effect.

1: Disables the corresponding interrupt.

- **CMDRDY: Command Ready Interrupt Disable**
- **RXRDY: Receiver Ready Interrupt Disable**
- **TXRDY: Transmit Ready Interrupt Disable**
- **BLKE: Data Block Ended Interrupt Disable**
- **DTIP: Data Transfer in Progress Interrupt Disable**
- **NOTBUSY: Data Not Busy Interrupt Disable**
- **SDIOIRQA: SDIO Interrupt for Slot A Interrupt Disable**
- **SDIOIRQB: SDIO Interrupt for Slot B Interrupt Disable**
- **SDIOWAIT: SDIO Read Wait Operation Status Interrupt Disable**
- **CSRCV: Completion Signal received interrupt Disable**
- **RINDE: Response Index Error Interrupt Disable**
- **RDIRE: Response Direction Error Interrupt Disable**
- **RCRCE: Response CRC Error Interrupt Disable**
- **RENDE: Response End Bit Error Interrupt Disable**
- **RTOE: Response TimeOut Error Interrupt Disable**
- **DCRCE: Data CRC Error Interrupt Disable**
- **DTOE: Data TimeOut Error Interrupt Disable**

- **CSTOE: Completion Signal TimeOut Error Interrupt Disable**
- **BLKOVRE: DMA Block Overrun Error Interrupt Disable**
- **FIFOEMPTY: FIFO empty Interrupt Disable**
- **XFRDONE: Transfer Done Interrupt Disable**
- **ACKRCV: Boot Acknowledge Interrupt Disable**
- **ACKRCVE: Boot Acknowledge Error Interrupt Disable**
- **OVRE: Overrun Interrupt Disable**
- **UNRE: Underrun Interrupt Disable**

37.14.15HSMCI Interrupt Mask Register

Name: HSMCI_IMR

Address: 0xF800004C (0), 0xFC00004C (1)

Access: Read-only

31	30	29	28	27	26	25	24
UNRE	OVRE	ACKRCVE	ACKRCV	XFRDONE	FIFOEMPTY	–	BLKOVRE
23	22	21	20	19	18	17	16
CSTOE	DTOE	DCRCE	RTOE	RENDE	RCRCE	RDIRE	RINDE
15	14	13	12	11	10	9	8
–	–	CSRCV	SDIOWAIT	–	–	SDIOIRQB	SDIOIRQA
7	6	5	4	3	2	1	0
–	–	NOTBUSY	DTIP	BLKE	TXRDY	RXRDY	CMDRDY

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.

- **CMDRDY: Command Ready Interrupt Mask**
- **RXRDY: Receiver Ready Interrupt Mask**
- **TXRDY: Transmit Ready Interrupt Mask**
- **BLKE: Data Block Ended Interrupt Mask**
- **DTIP: Data Transfer in Progress Interrupt Mask**
- **NOTBUSY: Data Not Busy Interrupt Mask**
- **SDIOIRQA: SDIO Interrupt for Slot A Interrupt Mask**
- **SDIOIRQB: SDIO Interrupt for Slot B Interrupt Mask**
- **SDIOWAIT: SDIO Read Wait Operation Status Interrupt Mask**
- **CSRCV: Completion Signal Received Interrupt Mask**
- **RINDE: Response Index Error Interrupt Mask**
- **RDIRE: Response Direction Error Interrupt Mask**
- **RCRCE: Response CRC Error Interrupt Mask**
- **RENDE: Response End Bit Error Interrupt Mask**
- **RTOE: Response TimeOut Error Interrupt Mask**
- **DCRCE: Data CRC Error Interrupt Mask**
- **DTOE: Data TimeOut Error Interrupt Mask**

- **CSTOE: Completion Signal TimeOut Error Interrupt Mask**
- **BLKOVRE: DMA Block Overrun Error Interrupt Mask**
- **FIFOEMPTY: FIFO Empty Interrupt Mask**
- **XFRDONE: Transfer Done Interrupt Mask**
- **ACKRCV: Boot Operation Acknowledge Received Interrupt Mask**
- **ACKRCVE: Boot Operation Acknowledge Error Interrupt Mask**
- **OVRE: Overrun Interrupt Mask**
- **UNRE: Underrun Interrupt Mask**

37.14.16HSMCI DMA Configuration Register

Name: HSMCI_DMA

Address: 0xF8000050 (0), 0xFC000050 (1)

Access: Read/Write

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

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

- **CHKSIZE: DMA Channel Read and Write Chunk Size**

The CHKSIZE field indicates the number of data available when the DMA chunk transfer request is asserted.

Value	Name	Description
0	1	1 data available
1	2	2 data available
2	4	4 data available
3	8	8 data available
4	16	16 data available

- **DMAEN: DMA Hardware Handshaking Enable**

0: DMA interface is disabled.

1: DMA Interface is enabled.

Note: To avoid unpredictable behavior, DMA hardware handshaking must be disabled when CPU transfers are performed.

37.14.17HSMCI Configuration Register

Name: HSMCI_CFG

Address: 0xF8000054 (0), 0xFC000054 (1)

Access: Read/Write

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

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

- **FIFOMODE: HSMCI Internal FIFO control mode**

0: A write transfer starts when a sufficient amount of data is written into the FIFO.

When the block length is greater than or equal to 3/4 of the HSMCI internal FIFO size, then the write transfer starts as soon as half the FIFO is filled. When the block length is greater than or equal to half the internal FIFO size, then the write transfer starts as soon as one quarter of the FIFO is filled. In other cases, the transfer starts as soon as the total amount of data is written in the internal FIFO.

1: A write transfer starts as soon as one data is written into the FIFO.

- **FERRCTRL: Flow Error flag reset control mode**

0: When an underflow/overflow condition flag is set, a new Write/Read command is needed to reset the flag.

1: When an underflow/overflow condition flag is set, a read status resets the flag.

- **HSMODE: High Speed Mode**

0: Default bus timing mode.

1: If set to one, the host controller outputs command line and data lines on the rising edge of the card clock. The Host driver shall check the high speed support in the card registers.

- **LSYNC: Synchronize on the last block**

0: The pending command is sent at the end of the current data block.

1: The pending command is sent at the end of the block transfer when the transfer length is not infinite (block count shall be different from zero).

37.14.18HSMCI Write Protection Mode Register

Name: HSMCI_WPMR

Address: 0xF80000E4 (0), 0xFC0000E4 (1)

Access: Read/Write

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

- **WPEN: Write Protect Enable**

0: Disables the Write Protection if WPKEY corresponds to 0x4D4349 (“MCI” in ASCII).

1: Enables the Write Protection if WPKEY corresponds to 0x4D4349 (“MCI” in ASCII).

Refer to Section 37.13 “Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protect Key**

Value	Name	Description
0x4D4349	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

37.14.19HSMCI Write Protection Status Register

Name: HSMCI_WPSR

Address: 0xF80000E8 (0), 0xFC0000E8 (1)

Access: Read-only

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

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the HSMCI_WPSR.

1: A write protection violation has occurred since the last read of the HSMCI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

37.14.20HSMCI FIFOx Memory Aperture

Name: HSMCI_FIFOx [x=0..255]

Address: 0xF8000200 (0), 0xFC000200 (1)

Access: Read/Write

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

- **DATA: Data to Read or Data to Write**

38. Serial Peripheral Interface (SPI)

38.1 Description

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:

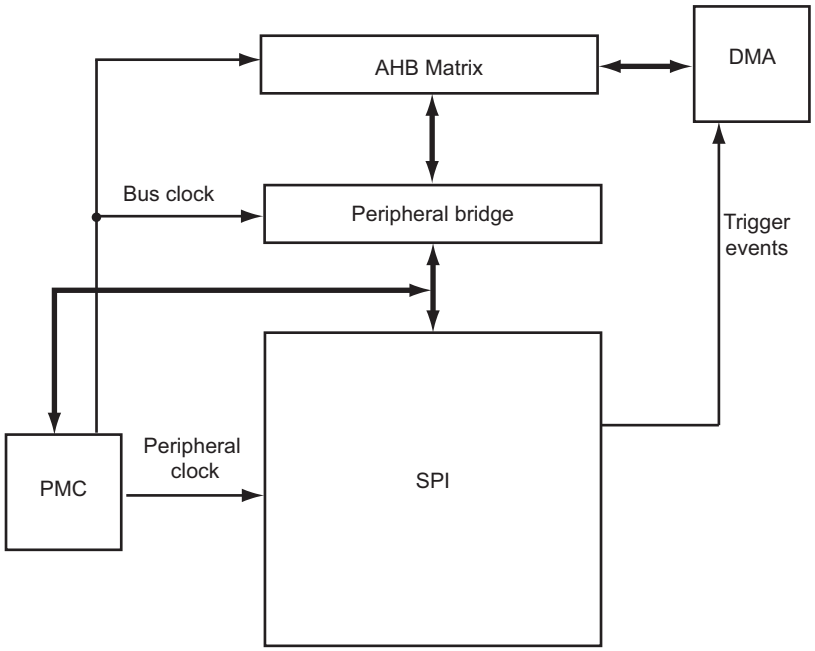
- Master Out Slave In (MOSI)—This data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO)—This data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any 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.

38.2 Embedded Characteristics

- 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 15 peripherals
- 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

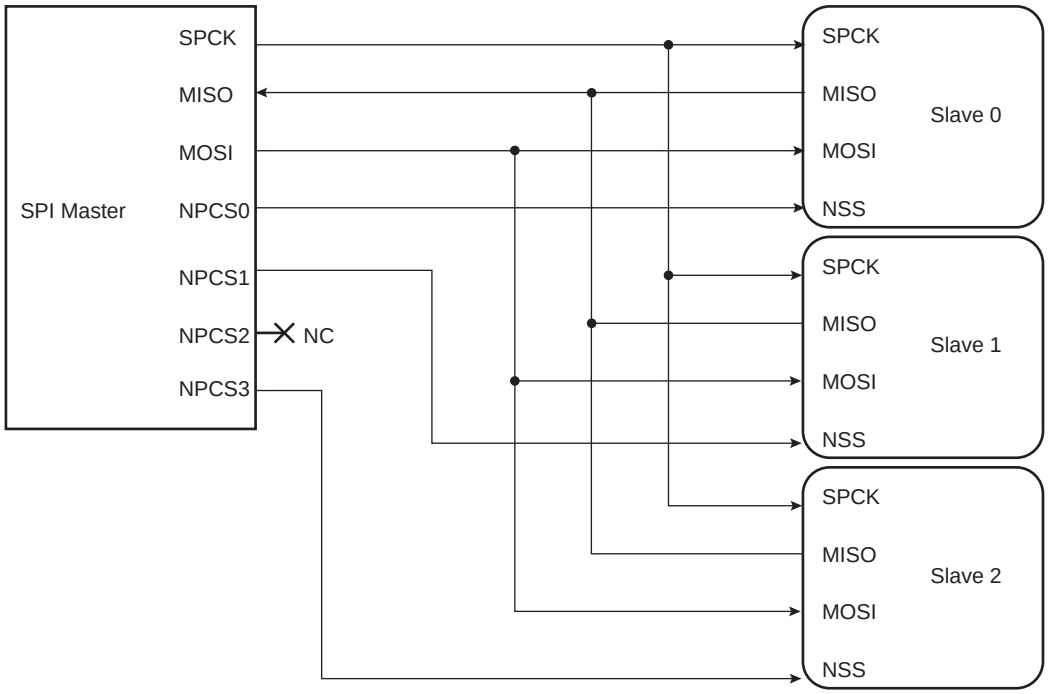
38.3 Block Diagram

Figure 38-1. Block Diagram



38.4 Application Block Diagram

Figure 38-2. Application Block Diagram: Single Master/Multiple Slave Implementation



38.5 Signal Description

Table 38-1. Signal Description

Pin Name	Pin Description	Type	
		Master	Slave
MISO	Master In Slave Out	Input	Output
MOSI	Master Out Slave In	Output	Input
SPCK	Serial Clock	Output	Input
NPCS1–NPCS3	Peripheral Chip Selects	Output	Unused
NPCS0/NSS	Peripheral Chip Select/Slave Select	Output	Input

38.6 Product Dependencies

38.6.1 I/O Lines

The pins used for interfacing the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the SPI pins to their peripheral functions.

Table 38-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
SPI0	SPI0_MISO	PC0	A
SPI0	SPI0_MOSI	PC1	A
SPI0	SPI0_NPCS0	PC3	A
SPI0	SPI0_NPCS1	PC4	A
SPI0	SPI0_NPCS1	PC27	B
SPI0	SPI0_NPCS2	PC28	B
SPI0	SPI0_NPCS2	PD31	A
SPI0	SPI0_NPCS3	PC29	B
SPI0	SPI0_SPCK	PC2	A
SPI1	SPI1_MISO	PB18	A
SPI1	SPI1_MOSI	PB19	A
SPI1	SPI1_NPCS0	PB21	A
SPI1	SPI1_NPCS1	PA26	C
SPI1	SPI1_NPCS1	PB22	A
SPI1	SPI1_NPCS2	PA27	C
SPI1	SPI1_NPCS2	PB23	A
SPI1	SPI1_NPCS3	PA28	C
SPI1	SPI1_NPCS3	PB27	A
SPI1	SPI1_SPCK	PB20	A
SPI2	SPI2_MISO	PD11	B
SPI2	SPI2_MOSI	PD13	B
SPI2	SPI2_NPCS0	PD17	B

Table 38-2. I/O Lines (Continued)

SPI2	SPI2_NPCS1	PB14	B
SPI2	SPI2_NPCS2	PB15	B
SPI2	SPI2_NPCS3	PB28	A
SPI2	SPI2_SPCK	PD15	B

38.6.2 Power Management

The SPI can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SPI clock.

38.6.3 Interrupt

The SPI interface has an interrupt line connected to the interrupt controller. Handling the SPI interrupt requires programming the interrupt controller before configuring the SPI.

Table 38-3. Peripheral IDs

Instance	ID
SPI0	37
SPI1	38
SPI2	39

38.6.4 Direct Memory Access Controller (DMAC)

The SPI interface can be used in conjunction with the DMAC in order to reduce processor overhead. For a full description of the DMAC, refer to Section 30. “DMA Controller (XDMAC)”.

38.7 Functional Description

38.7.1 Modes of Operation

The SPI operates in Master mode or in Slave mode.

- The SPI operates in Master mode by setting the MSTR bit in the SPI Mode Register (SPI_MR):
  - 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 the 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)
  - 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 baud rate generator is activated only in Master mode.

38.7.2 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 registers (SPI_CSRx). The clock phase is programmed with the NCPHA bit. 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. 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.

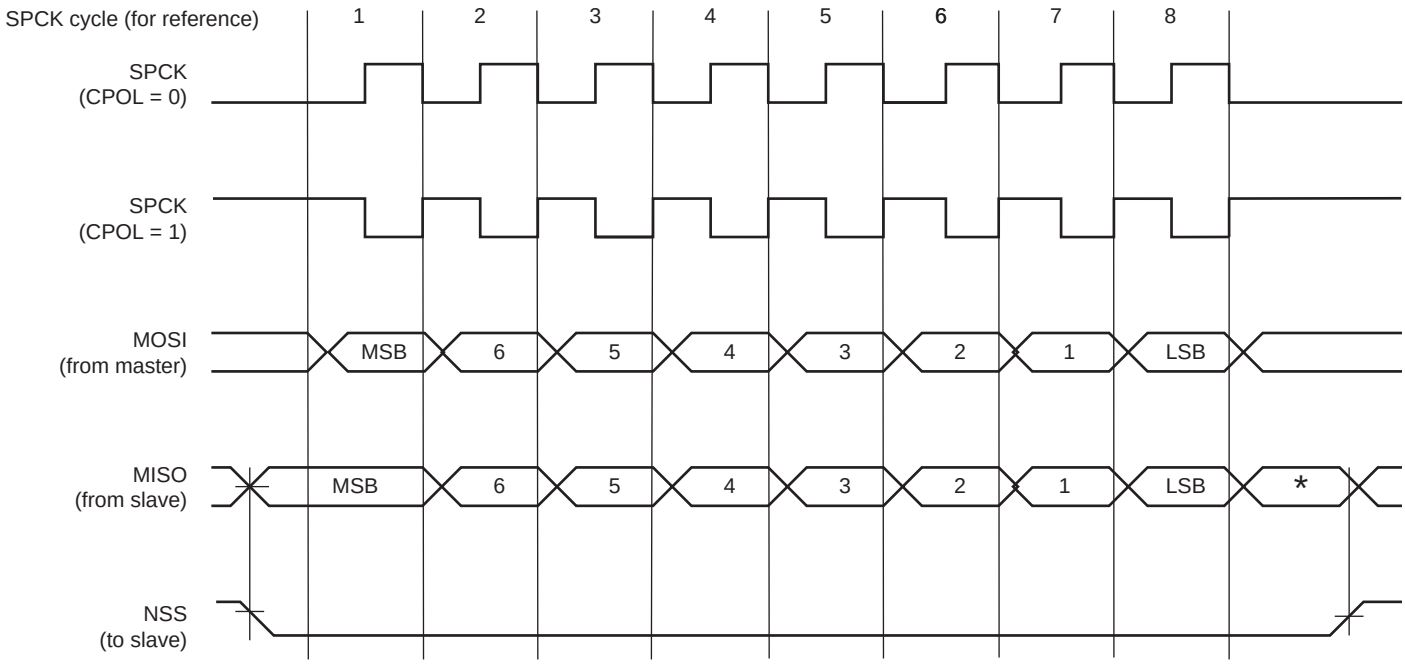
Table 38-4 shows the four modes and corresponding parameter settings.

Table 38-4. SPI Bus Protocol Modes

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

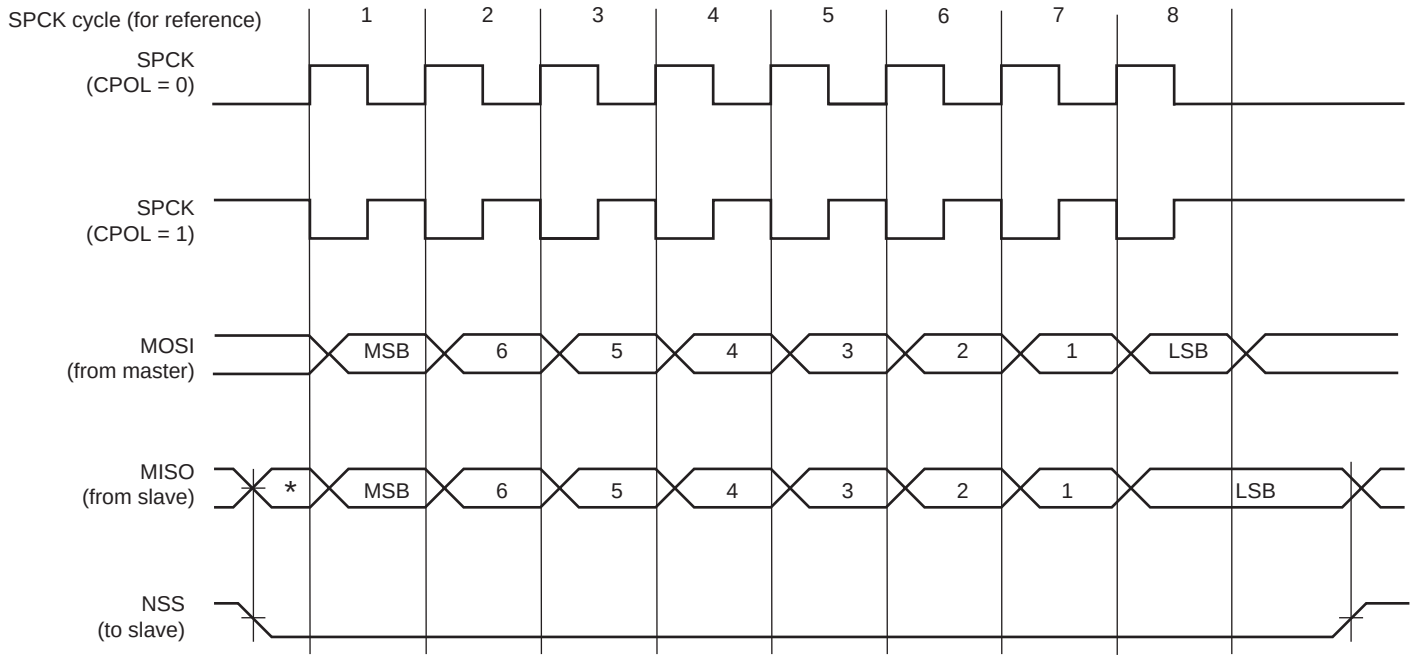
Figure 38-3 and Figure 38-4 show examples of data transfers.

Figure 38-3. SPI Transfer Format (NCPHA = 1, 8 bits per transfer)



* Not defined.

Figure 38-4. SPI Transfer Format (NCPHA = 0, 8 bits per transfer)



* Not defined.

38.7.3 Master Mode Operations

When configured in Master mode, the SPI operates on the clock generated by the internal programmable baud 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 (SPI_TDR) and the Receive Data Register (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 the SPI_TDR. The written data is 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 the SPI_RDR without transmitting data. If there is no data to transmit, dummy data can be used (SPI_TDR filled with ones). If the SPI_MR.WDRBT bit is set, transmission can occur only if the SPI_RDR has 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 (SPI_SR) can be discarded.

Before writing the SPI_TDR, the PCS field in the SPI_MR must be set in order to select a slave.

If new data is written in the SPI_TDR during the transfer, it is kept in the SPI_TDR until the current transfer is completed. Then, the received data is transferred from the Shift register to the SPI_RDR, the data in the SPI_TDR is loaded in the Shift register and a new transfer starts.

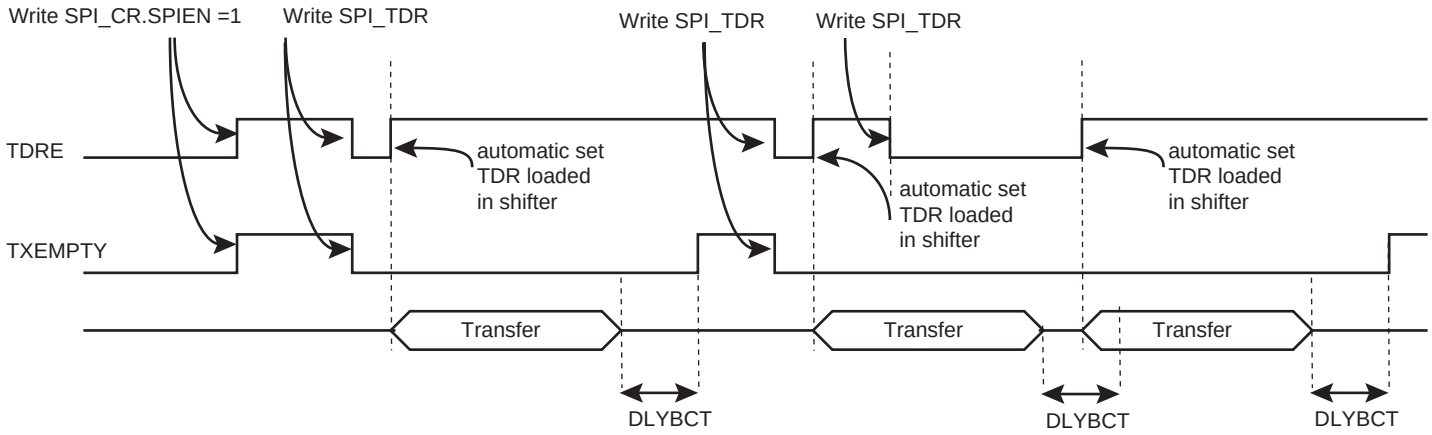
As soon as the SPI_TDR is written, the Transmit Data Register Empty (TDRE) flag in the SPI_SR is cleared. When the data written in the SPI_TDR is loaded into the Shift register, the TDRE flag in the SPI_SR is set. The TDRE bit is used to trigger the Transmit DMA channel.

Refer to Figure 38-5.

The end of transfer is indicated by the TXEMPTY flag in the SPI_SR. 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: When the SPI is enabled, the TDRE and TXEMPTY flags are set.

Figure 38-5. TDRE and TXEMPTY flag behavior



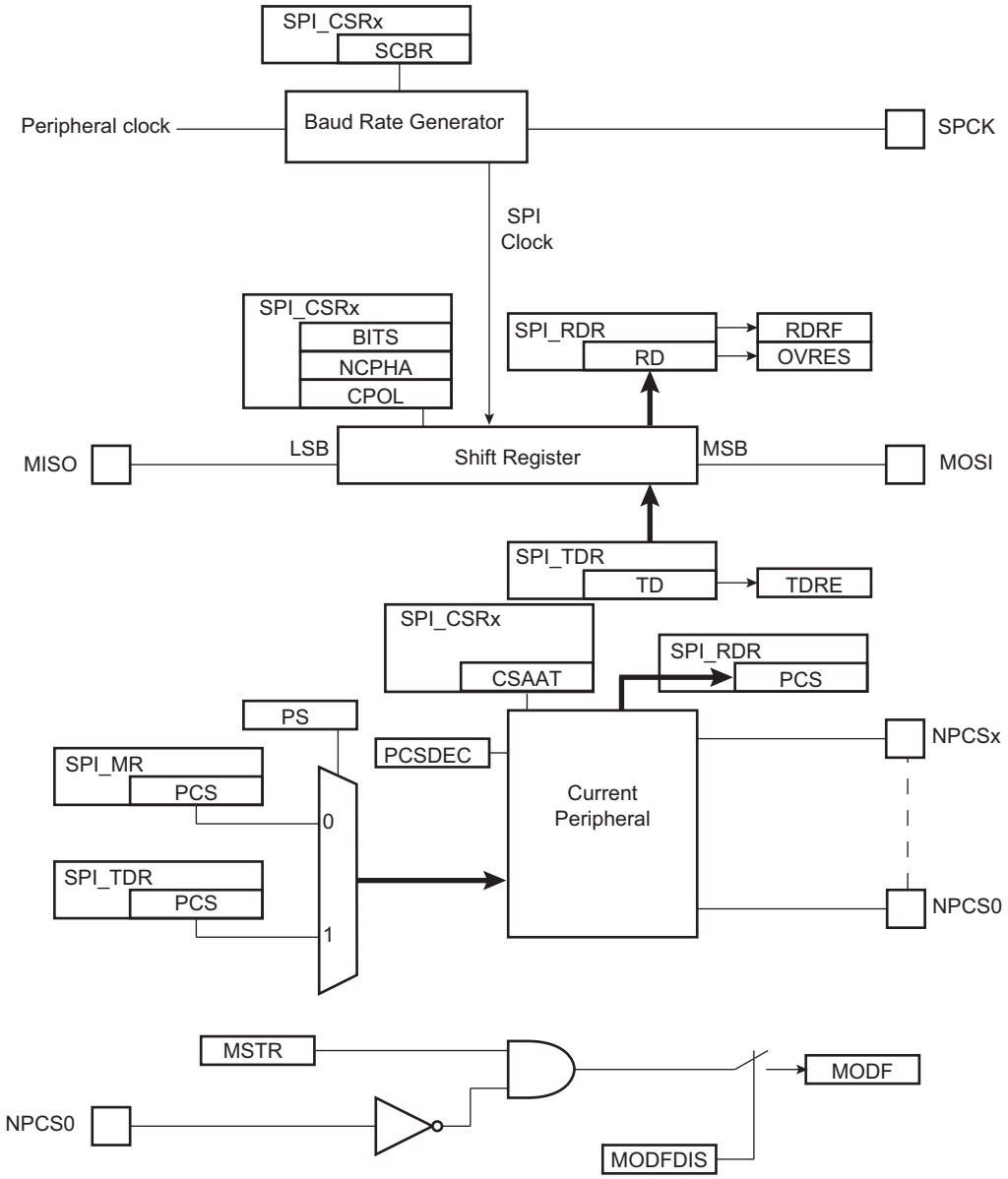
The transfer of received data from the Shift register to the SPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in the SPI_SR. When the received data is read, the RDRF bit is cleared.

If the SPI_RDR has not been read before new data is received, the Overrun Error (OVRES) bit in the SPI_SR is set. As long as this flag is set, data is loaded in the SPI_RDR. The user has to read the SPI_SR to clear the OVRES bit.

Figure 38-6 shows a block diagram of the SPI when operating in Master mode. Figure 38-7 shows a flow chart describing how transfers are handled.

38.7.3.1 Master Mode Block Diagram

Figure 38-6. Master Mode Block Diagram



38.7.3.2 Master Mode Flow Diagram

Figure 38-7. Master Mode Flow Diagram

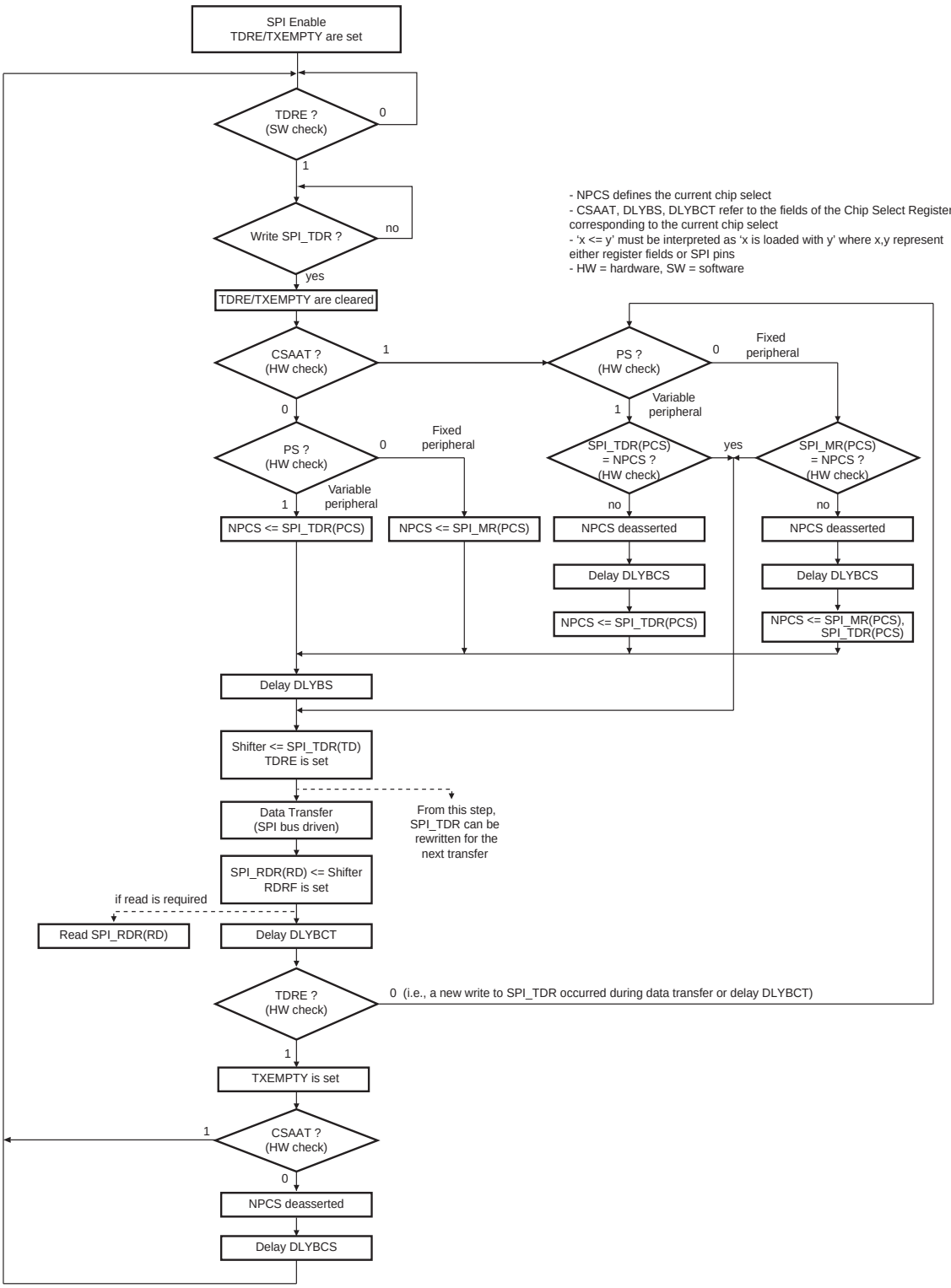
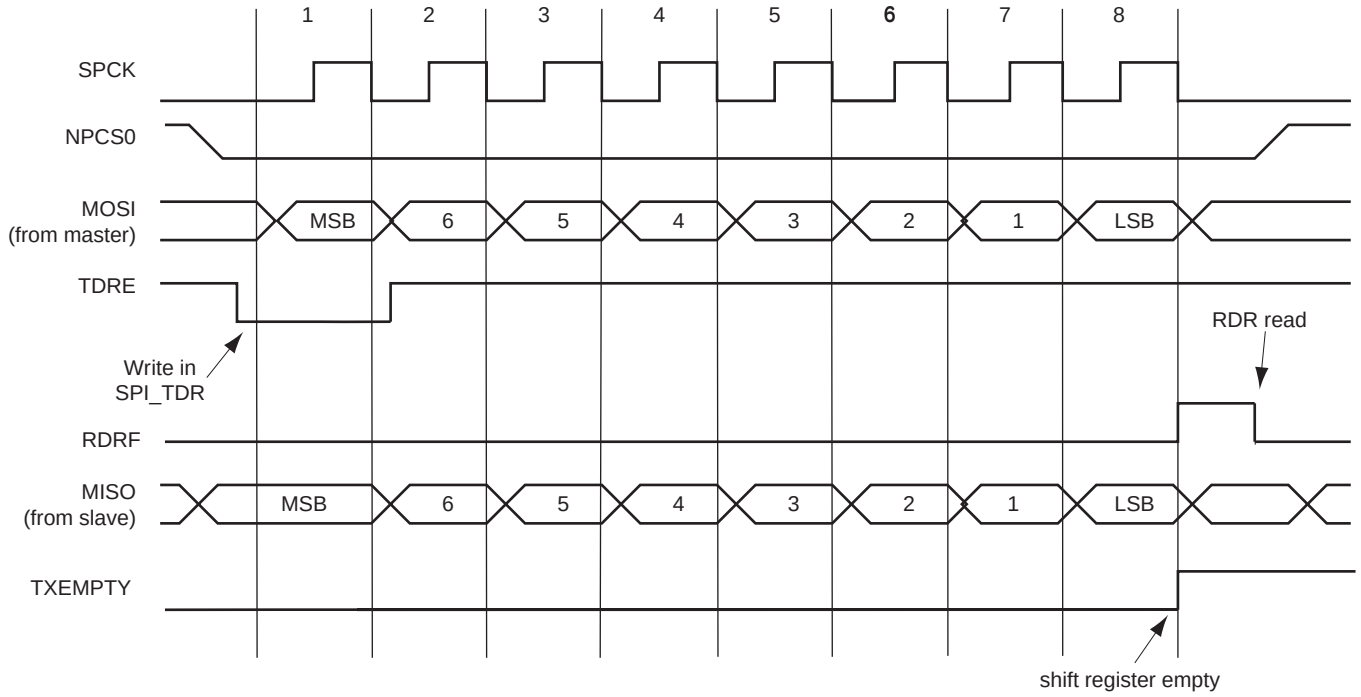


Figure 38-8 shows the behavior of Transmit Data Register Empty (TDRE), Receive Data Register (RDRF) and Transmission Register Empty (TXEMPTY) status flags within the SPI_SR during an 8-bit data transfer in Fixed mode without the DMA involved.

Figure 38-8. Status Register Flags Behavior



38.7.3.3 Clock Generation

The SPI Baud rate clock is generated by dividing the peripheral clock by a value between 1 and 255.

If the SCBR field in the SPI_CSRx is programmed to 1, the operating baud rate is peripheral clock (refer to Section 55. “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 SCBR field. This allows the SPI to automatically adapt the baud rate for each interfaced peripheral without reprogramming.

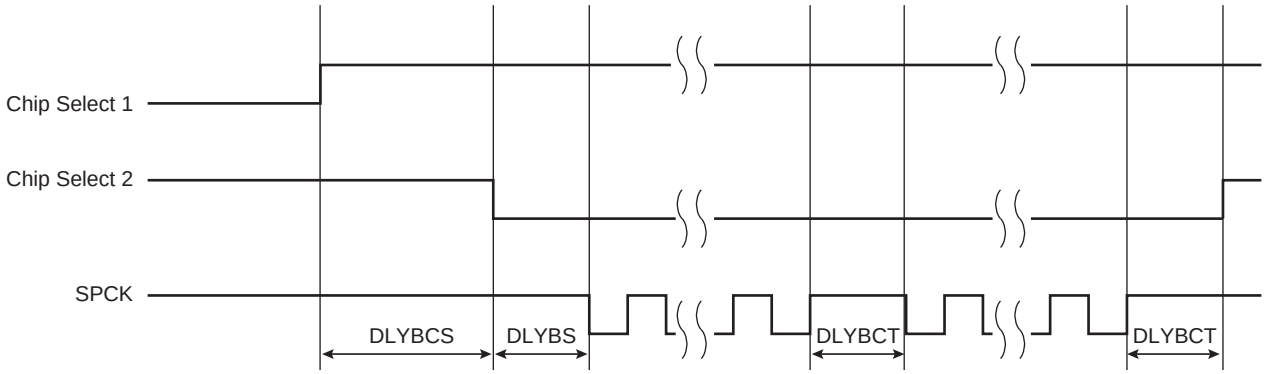
38.7.3.4 Transfer Delays

Figure 38-9 shows a chip select transfer change and consecutive transfers on the same chip select. Three delays can be programmed to modify the transfer waveforms:

- Delay between the chip selects—programmable only once for all chip selects by writing the DLYBCS field in the SPI_MR. 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.
- 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.
- 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 38-9. Programmable Delays



38.7.3.5 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 clearing the PS bit in the SPI_MR. In this case, the current peripheral is defined by the PCS field in the SPI_MR and the PCS field in the SPI_TDR has no effect.
- **Variable Peripheral Select Mode:** Data can be exchanged with more than one peripheral without having to reprogram the NPCS field in the SPI_MR.
  Variable Peripheral Select mode is enabled by setting the PS bit in the SPI_MR. The PCS field in the SPI_TDR is used to select the current peripheral. This means that the peripheral selection can be defined for each new data. The value to write in the SPI_TDR has the following format:

  [xxxxxxx(7-bit) + LASTXFER(1-bit)(1)+ xxxx(4-bit) + PCS (4-bit) + DATA (8 to 16-bit)] with PCS equals the chip select to assert, as defined in Section 38.8.4 “SPI Transmit Data Register” and LASTXFER bit at 0 or 1 depending on the CSAAT bit.

Note: 1. Optional

CSAAT, LASTXFER and CSNAAT bits are discussed in Section 38.7.3.9 “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 (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 SPIEN has previously been written in the SPI_CR.

38.7.3.6 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, the SPI_MR must be reprogrammed.

The variable peripheral selection allows buffer transfers with multiple peripherals without reprogramming the SPI_MR. Data written in the 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.

38.7.3.7 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 (refer to Figure 38-10). This can be enabled by setting the PCSDEC bit in the SPI_MR.

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.

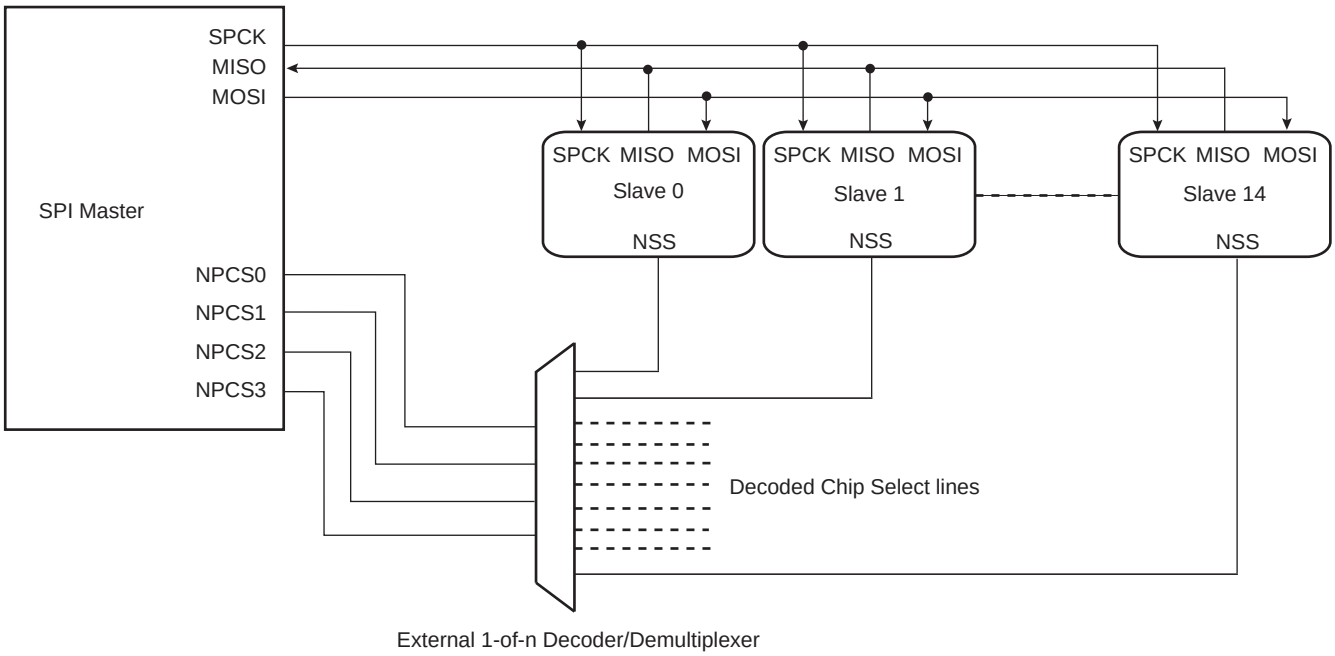
When operating with decoding, the SPI directly outputs the value defined by the PCS field on the NPCS lines of either SPI_MR or SPI_TDR (depending on PS).

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 four chip select registers (SPI_CSR0...SPI_CSR3). As a result, when external decoding is activated, each NPCS chip select defines the characteristics of up to four peripherals. As an example, 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. Figure 38-10 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 38-10. Chip Select Decoding Application Block Diagram: Single Master/Multiple Slave Implementation



38.7.3.8 Peripheral Deselection without DMA

During a transfer of more than one unit of data on a chip select without the DMA, the SPI_TDR is loaded by the processor, the TDRE flag rises as soon as the content of the SPI_TDR is transferred into the internal Shift register. When this flag is detected high, the 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 the SPI_TDR in time to keep the chip select active (low). A null DLYBCT value (delay between consecutive transfers) in the SPI_CSR, gives even less time for the processor to reload the 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 [SPI_CSR0...SPI_CSR3] can be programmed with the Chip Select Active After Transfer (CSAAT) bit at 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 the SPI_TDR is not reloaded, the chip select remains active. To deassert the chip select line at the end of the transfer, the Last Transfer (LASTXFER) bit in SPI_CR must be set after writing the last data to transmit into SPI_TDR.

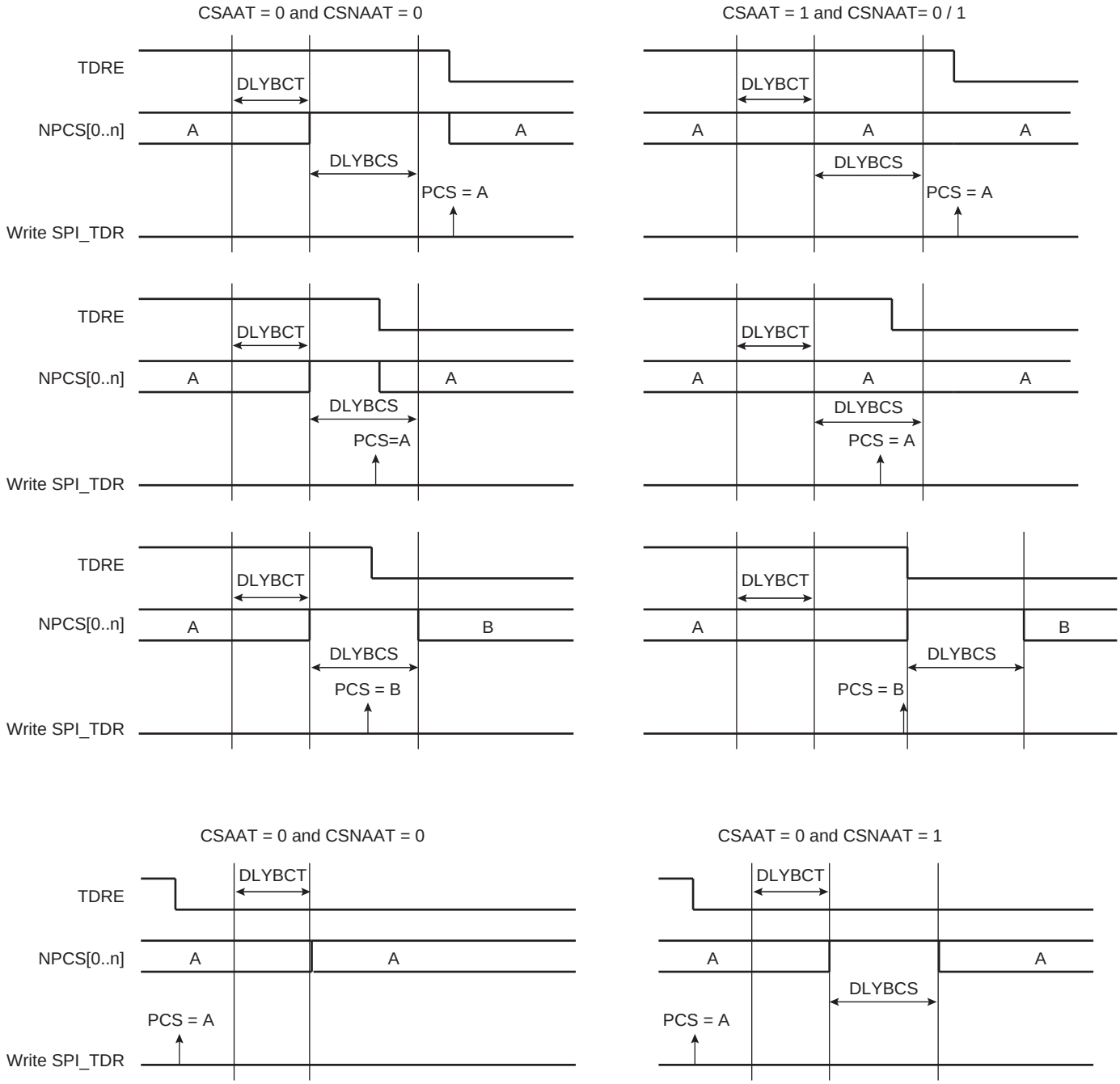
38.7.3.9 Peripheral Deselection with DMA

DMA provides faster reloads of the SPI_TDR compared to software. However, depending on the system activity, it is not guaranteed that the SPI_TDR is written with the next data before the end of the current transfer. Consequently, 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 configured to 0, 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 the SPI_TDR is transferred into the internal shift register. When this flag is detected, the 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, the SPI_CSR can be programmed with the Chip Select Not Active After Transfer (CSNAAT) bit at 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 configured to 0 for the same chip select).

Figure 38-11 shows different peripheral deselection cases and the effect of the CSAAT and CSNAAT bits.

Figure 38-11. Peripheral Deselection



38.7.3.10Mode Fault Detection

The SPI has the capability to operate in multimaster 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 any 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 multimaster environment, NPCS0, MOSI, MISO and SPCK pins must be configured in open drain (through the PIO controller). When a mode fault is detected, the SPI_SR.MODF bit is set until SPI_SR is read and the SPI is automatically disabled until it is reenabled by setting the SPI_CR.SPIEN bit.

By default, the mode fault detection is enabled. The user can disable it by setting the SPI_MR.MODFDIS bit.

38.7.4 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 is loaded in the SPI_RDR depending on the BITS field configured in SPI_CSR0. These bits are processed following a phase and a polarity defined respectively by the NCPHA and CPOL bits in SPI_CSR0. Note that the fields BITS, CPOL and NCPHA of the other chip select registers (SPI_CSR1...SPI_CSR3) 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, refer to the note below the SPI_CSRx bitmap (Section 38.8.9 “SPI Chip Select Register”).

When all bits are processed, the received data is transferred in the SPI_RDR and the RDRF bit rises. If the SPI_RDR has not been read before new data is received, the Overrun Error Status (OVRES) bit in the SPI_SR is set. As long as this flag is set, data is loaded in the SPI_RDR. The user must read SPI_SR to clear the OVRES bit.

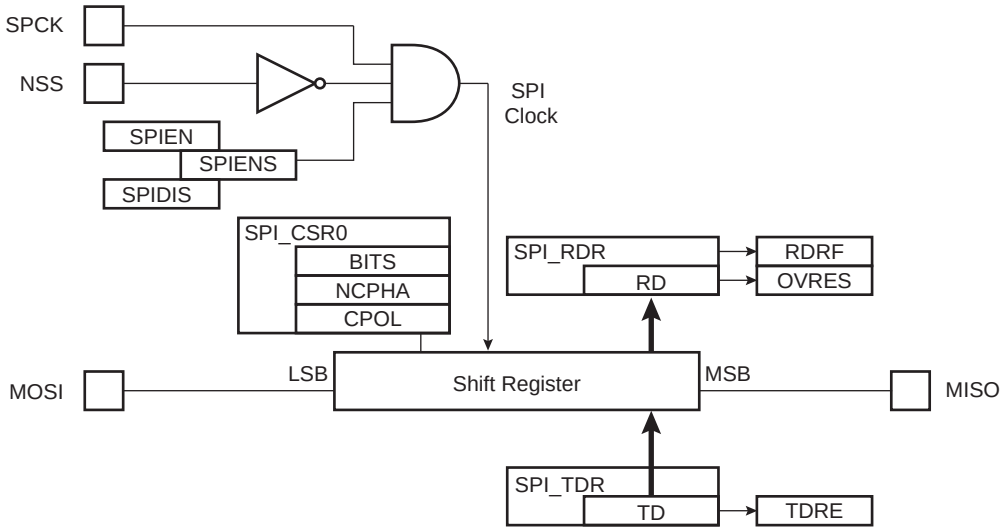
When a transfer starts, the data shifted out is the data present in the Shift register. If no data has been written in the SPI_TDR, the last data received is transferred. If no data has been received since the last reset, all bits are transmitted low, as the Shift register resets to 0.

When a first data is written in the SPI_TDR, it is transferred immediately in the Shift register and the TDRE flag rises. If new data is written, it remains in the 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 the SPI_TDR is transferred in the Shift register and the TDRE flag rises. This enables frequent updates of critical variables with single transfers.

Then, new data is loaded in the Shift register from the SPI_TDR. If no character is ready to be transmitted, i.e., no character has been written in the SPI_TDR since the last load from the SPI_TDR to the Shift register, the SPI_TDR is retransmitted. In this case the Underrun Error Status Flag (UNDES) is set in the SPI_SR.

Figure 38-12 shows a block diagram of the SPI when operating in Slave mode.

Figure 38-12. Slave Mode Functional Block Diagram



38.7.5 Register Write Protection

To prevent any single software error from corrupting SPI behavior, certain registers in the address space can be write-protected in the SPI Write Protection Mode Register (SPI_WPMR).

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

The WPVS bit is automatically cleared after reading SPI_WPSR.

The following registers are write-protected when WPEN is set in SPI_WPMR:

- SPI Mode Register
- SPI Chip Select Register

38.8 Serial Peripheral Interface (SPI) User Interface

In the “Offset” column of Table 38-5, ‘CS_number’ denotes the chip select number.

Table 38-5. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	SPI_CR	Write-only	–
0x04	Mode Register	SPI_MR	Read/Write	0x0
0x08	Receive Data Register	SPI_RDR	Read-only	0x0
0x0C	Transmit Data Register	SPI_TDR	Write-only	–
0x10	Status Register	SPI_SR	Read-only	0x0
0x14	Interrupt Enable Register	SPI_IER	Write-only	–
0x18	Interrupt Disable Register	SPI_IDR	Write-only	–
0x1C	Interrupt Mask Register	SPI_IMR	Read-only	0x0
0x20–0x2C	Reserved	–	–	–
0x30 + (CS_number * 0x04)	Chip Select Register	SPI_CSR	Read/Write	0x0
0x40–0x48	Reserved	–	–	–
0x4C–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	SPI_WPMR	Read/Write	0x0
0xE8	Write Protection Status Register	SPI_WPSR	Read-only	0x0
0xEC–0xF8	Reserved	–	–	–
0xFC	Reserved	–	–	–

38.8.1 SPI Control Register

Name: SPI_CR

Address: 0xF8010000 (0), 0xFC018000 (1), 0xFC01C000 (2)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	LASTXFER
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	REQCLR	–	–	–	–
7	6	5	4	3	2	1	0
SWRST	–	–	–	–	–	SPIDIS	SPIEN

- **SPIEN: SPI Enable**

0: No effect.

1: Enables the SPI to transfer and receive data.

- **SPIDIS: SPI Disable**

0: No effect.f

1: Disables the SPI.

All pins are set in Input mode after completion of the transmission in progress, if any.

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 SPI_THR is loaded.

Note: If both SPIEN and SPIDIS are equal to one when the SPI_CR is written, the SPI is disabled.

- **SWRST: SPI Software Reset**

0: No effect.

1: Reset the SPI. A software-triggered hardware reset of the SPI interface is performed.

The SPI is in Slave mode after software reset.

- **REQCLR: Request to Clear the Comparison Trigger**

0: No effect.

1: Restarts the comparison trigger to enable SPI_RDR loading.

- **LASTXFER: Last Transfer**

0: No effect.

1: The current NPCS is deasserted after the character written in TD has been transferred. When 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.

Refer to Section 38.7.3.5 “Peripheral Selection” for more details.

38.8.2 SPI Mode Register

Name: SPI_MR

Address: 0xF8010004 (0), 0xFC018004 (1), 0xFC01C004 (2)

Access: Read/Write

31	30	29	28	27	26	25	24
DLYBCS							
23	22	21	20	19	18	17	16
–	–	–	–	PCS			
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
LLB	–	WDRBT	MODFDIS	–	PCSDEC	PS	MSTR

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

- **MSTR: Master/Slave Mode**

0: SPI is in Slave mode

1: SPI is in Master mode

- **PS: Peripheral Select**

0: Fixed Peripheral Select

1: Variable Peripheral Select

- **PCSDEC: Chip Select Decode**

0: The chip select lines are directly connected to a peripheral device.

1: The four NPCS chip select lines are connected to a 4-bit to 16-bit decoder.

When PCSDEC = 1, up to 15 chip select signals can be generated with the four NPCS lines using an external 4-bit to 16-bit decoder. The chip select registers define the characteristics of the 15 chip selects, with the following rules:

SPI_CSR0 defines peripheral chip select signals 0 to 3.

SPI_CSR1 defines peripheral chip select signals 4 to 7.

SPI_CSR2 defines peripheral chip select signals 8 to 11.

SPI_CSR3 defines peripheral chip select signals 12 to 14.

- **MODFDIS: Mode Fault Detection**

0: Mode fault detection enabled

1: Mode fault detection disabled

- **WDRBT: Wait Data Read Before Transfer**

0: No Effect. In Master mode, a transfer can be initiated regardless of the SPI_RDR state.

1: In Master mode, a transfer can start only if the SPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception.

- **LLB: Local Loopback Enable**

0: Local loopback path disabled.

1: Local loopback path enabled.

LLB controls the local loopback on the data shift register for testing in Master mode only (MISO is internally connected on MOSI).

- **PCS: Peripheral Chip Select**

This field is only used if fixed peripheral select is active (PS = 0).

If 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 SPI_MR.PCSDEC = 1:

NPCS[3:0] output signals = PCS.

- **DLYBCS: 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 lower than 6, six peripheral clock periods are inserted by default.

Otherwise, the following equation determines the delay:

$$\text{Delay Between Chip Selects} = \frac{\text{DLYBCS}}{f_{\text{peripheral clock}}}$$

38.8.3 SPI Receive Data Register

Name: SPI_RDR

Address: 0xF8010008 (0), 0xFC018008 (1), 0xFC01C008 (2)

Access: Read-only

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

- **RD: Receive Data**

Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

- **PCS: 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 (PS = 1 in SPI_MR), it is mandatory to set the SPI_MR.WDRBT bit if the PCS field must be processed in SPI_RDR.

38.8.4 SPI Transmit Data Register

Name: SPI_TDR

Address: 0xF801000C (0), 0xFC01800C (1), 0xFC01C00C (2)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	LASTXFER

23	22	21	20	19	18	17	16
–	–	–	–	PCS			

15	14	13	12	11	10	9	8
TD							

7	6	5	4	3	2	1	0
TD							

- **TD: 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.

- **PCS: Peripheral Chip Select**

This field is only used if variable peripheral select is active (PS = 1).

If 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 SPI_MR.PCSDEC = 1:

NPCS[3:0] output signals = PCS.

- **LASTXFER: Last Transfer**

0: No effect

1: The current NPCS is deasserted after the transfer of the character written in TD. When 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.

This field is only used if variable peripheral select is active (SPI_MR.PS = 1).

38.8.5 SPI Status Register

Name: SPI_SR

Address: 0xF8010010 (0), 0xFC018010 (1), 0xFC01C010 (2)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	SPIENS
15	14	13	12	11	10	9	8
–	–	–	–	–	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full (cleared by reading SPI_RDR)**

0: No data has been received since the last read of SPI_RDR.

1: Data has been received and the received data has been transferred from the shift register to SPI_RDR since the last read of SPI_RDR.

- **TDRE: Transmit Data Register Empty (cleared by writing SPI_TDR)**

0: Data has been written to SPI_TDR and not yet transferred to the shift register.

1: The last data written in the SPI_TDR has been transferred to the shift register.

TDRE equals zero when the SPI is disabled or at reset. The SPI enable command sets this bit to 1.

- **MODF: Mode Fault Error (cleared on read)**

0: No mode fault has been detected since the last read of SPI_SR.

1: A mode fault occurred since the last read of SPI_SR.

- **OVRES: Overrun Error Status (cleared on read)**

0: No overrun has been detected since the last read of SPI_SR.

1: An overrun has occurred since the last read of SPI_SR.

An overrun occurs when SPI_RDR is loaded at least twice from the shift register since the last read of the SPI_RDR.

- **NSSR: NSS Rising (cleared on read)**

0: No rising edge detected on NSS pin since the last read of SPI_SR.

1: A rising edge occurred on NSS pin since the last read of SPI_SR.

- **TXEMPTY: Transmission Registers Empty (cleared by writing SPI_TDR)**

0: As soon as data is written in SPI_TDR.

1: SPI_TDR and internal shift register are empty. If a transfer delay has been defined, TXEMPTY is set after the end of this delay.

- **UNDES: Underrun Error Status (Slave mode only) (cleared on read)**

0: No underrun has been detected since the last read of SPI_SR.

1: A transfer starts whereas no data has been loaded in SPI_TDR.

- **SPIENS: SPI Enable Status**

0: SPI is disabled.

1: SPI is enabled.

38.8.6 SPI Interrupt Enable Register

Name: SPI_IER

Address: 0xF8010014 (0), 0xFC018014 (1), 0xFC01C014 (2)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

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

0: No effect.

1: Enables the corresponding interrupt.

- **RDRF: Receive Data Register Full Interrupt Enable**
- **TDRE: SPI Transmit Data Register Empty Interrupt Enable**
- **MODF: Mode Fault Error Interrupt Enable**
- **OVRES: Overrun Error Interrupt Enable**
- **NSSR: NSS Rising Interrupt Enable**
- **TXEMPTY: Transmission Registers Empty Enable**
- **UNDES: Underrun Error Interrupt Enable**

38.8.7 SPI Interrupt Disable Register

Name: SPI_IDR

Address: 0xF8010018 (0), 0xFC018018 (1), 0xFC01C018 (2)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

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

0: No effect.

1: Disables the corresponding interrupt.

- **RDRF: Receive Data Register Full Interrupt Disable**
- **TDRE: SPI Transmit Data Register Empty Interrupt Disable**
- **MODF: Mode Fault Error Interrupt Disable**
- **OVRES: Overrun Error Interrupt Disable**
- **NSSR: NSS Rising Interrupt Disable**
- **TXEMPTY: Transmission Registers Empty Disable**
- **UNDES: Underrun Error Interrupt Disable**

38.8.8 SPI Interrupt Mask Register

Name: SPI_IMR

Address: 0xF801001C (0), 0xFC01801C (1), 0xFC01C01C (2)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

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.

- **RDRF: Receive Data Register Full Interrupt Mask**
- **TDRE: SPI Transmit Data Register Empty Interrupt Mask**
- **MODF: Mode Fault Error Interrupt Mask**
- **OVRES: Overrun Error Interrupt Mask**
- **NSSR: NSS Rising Interrupt Mask**
- **TXEMPTY: Transmission Registers Empty Mask**
- **UNDES: Underrun Error Interrupt Mask**

38.8.9 SPI Chip Select Register

Name: SPI_CSRx [x=0..3]

Address: 0xF8010030 (0), 0xFC018030 (1), 0xFC01C030 (2)

Access: Read/Write

31	30	29	28	27	26	25	24
DLYBCT							
23	22	21	20	19	18	17	16
DLYBS							
15	14	13	12	11	10	9	8
SCBR							
7	6	5	4	3	2	1	0
BITS				CSAAT	CSNAAT	NCPHA	CPOL

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

Note: 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.

- **CPOL: Clock Polarity**

0: The inactive state value of SPCK is logic level zero.

1: The inactive state value of SPCK is logic level one.

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.

- **NCPHA: Clock Phase**

0: Data is changed on the leading edge of SPCK and captured on the following edge of SPCK.

1: Data is captured on the leading edge of SPCK and changed on the following edge of SPCK.

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.

- **CSNAAT: Chip Select Not Active After Transfer (Ignored if CSAAT = 1)**

0: The Peripheral Chip Select Line does not rise between two transfers if the 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 Line rises systematically after each transfer performed on the same slave. It remains inactive after the end of transfer for a minimal duration of:

$$\frac{DLYBCS}{f_{peripheral\ clock}}$$

(If field DLYBCS is lower than 6, a minimum of six periods is introduced.)

- **CSAAT: Chip Select Active After Transfer**

0: The Peripheral Chip Select Line rises as soon as the last transfer is achieved.

1: The Peripheral Chip Select Line does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select.

- **BITS: Bits Per Transfer**

(Refer to Note under bitmap in Section 38.8.9 “SPI Chip Select Register”.)

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	–	Reserved
10	–	Reserved
11	–	Reserved
12	–	Reserved
13	–	Reserved
14	–	Reserved
15	–	Reserved

- **SCBR: Serial Clock Bit Rate**

In Master mode, the SPI Interface uses a modulus counter to derive the SPCK bit rate from the peripheral clock. The bit rate is selected by writing a value from1 to 255 in the SCBR field. The following equation determines the SPCK bit rate:

$SCBR = f_{peripheral\ clock} / SPCK\ Bit\ Rate$

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 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 SCBR fields in SPI_CSRx is set to 1, the other SCBR fields in SPI_CSRx 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.

- **DLYBS: 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 equation determines the delay:

$DLYBS = Delay\ Before\ SPCK \times f_{peripheral\ clock}$

- **DLYBCT: 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 equation determines the delay:

DLYBCT = Delay Between Consecutive Transfers × $f_{peripheral\ clock}$ / 32

38.8.10 SPI Write Protection Mode Register

Name: SPI_WPMR

Address: 0xF80100E4 (0), 0xFC0180E4 (1), 0xFC01C0E4 (2)

Access: Read/Write

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

- **WPEN: Write Protection Enable**

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)

- **WPKEY: Write Protection Key**

Value	Name	Description
0x535049	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

Refer to Section 38.7.5 “Register Write Protection” for the list of registers that can be write-protected.

38.8.11 SPI Write Protection Status Register

Name: SPI_WPSR

Address: 0xF80100E8 (0), 0xFC0180E8 (1), 0xFC01C0E8 (2)

Access: Read-only

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

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of SPI_WPSR.

1: A write protection violation has occurred since the last read of SPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

39. Two-wire Interface (TWI)

39.1 Description

The Atmel 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, based on a byte-oriented transfer format. It can be used with any Atmel Two-wire Interface bus Serial EEPROM and I²C compatible device such as a Real Time Clock (RTC), Dot Matrix/Graphic LCD Controllers and temperature sensor. The TWI is programmable as a master or a slave with sequential or single-byte access. Multiple master capability is supported.

A configurable baud rate generator permits the output data rate to be adapted to a wide range of core clock frequencies.

Table 39-1 lists the compatibility level of the Atmel Two-wire Interface in Master mode and a full I^2C compatible device.

Table 39-1. Atmel TWI Compatibility with I^2C Standard

I^2C Standard	Atmel TWI
Standard Mode Speed (100 kHz)	Supported
Fast Mode Speed (400 kHz)	Supported
7- or 10-bit Slave Addressing	Supported
START byte(1)	Not Supported
Repeated Start (Sr) Condition	Supported
ACK and NACK Management	Supported
Slope Control and Input Filtering (Fast mode)	Not Supported
Clock Stretching/Synchronization	Supported
Multi Master Capability	Supported

Note: 1. START + b000000001 + Ack + Sr

39.2 Embedded Characteristics

- Compatible with Atmel Two-wire Interface Serial Memory and I^2C Compatible Devices(1)
- One, Two or Three Bytes for Slave Address
- Sequential Read/Write Operations
- Master, Multi-master and Slave Mode Operation
- Bit Rate: Up to 400 Kbit/s
- General Call Supported in Slave Mode
- Connection to DMA Controller (DMA) Channel Capabilities Optimizes Data Transfers
- Register Write Protection

Note: 1. Refer to Table 39-1 for details on compatibility with I²C Standard.

39.3 List of Abbreviations

Table 39-2. Abbreviations

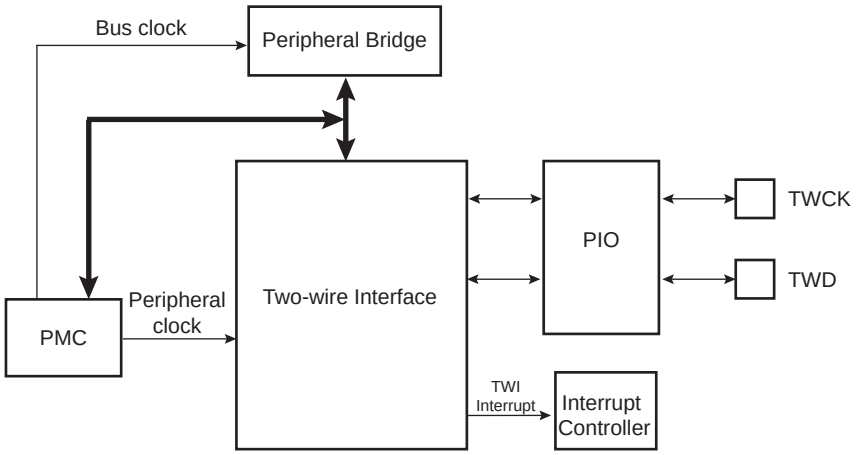
Abbreviation	Description
TWI	Two-wire Interface
A	Acknowledge
NA	Non Acknowledge

Table 39-2. Abbreviations (Continued)

Abbreviation	Description
P	Stop
S	Start
Sr	Repeated Start
SADR	Slave Address
ADR	Any address except SADR
R	Read
W	Write

39.4 Block Diagram

Figure 39-1. Block Diagram



39.5 I/O Lines Description

Table 39-3. I/O Lines Description

Name	Description	Type
TWD	Two-wire Serial Data (drives external serial data line – SDA)	Input/Output
TWCK	Two-wire Serial Clock (drives external serial clock line – SCL)	Input/Output

39.6 Product Dependencies

39.6.1 I/O Lines

Both TWD and TWCK are bidirectional lines, connected to a positive supply voltage via a current source or pull-up resistor. When the bus is free, both lines are high. The output stages of devices connected to the bus must have an open-drain or open-collector to perform the wired-AND function.

TWD and TWCK pins may be multiplexed with PIO lines. To enable the TWI, the user must program the PIO Controller to dedicate TWD and TWCK as peripheral lines.

The user must not program TWD and TWCK as open-drain. This is already done by the hardware.

Table 39-4. I/O Lines

Instance	Signal	I/O Line	Peripheral
TWI0	TWCK0	PA31	A
TWI0	TWD0	PA30	A
TWI1	TWCK1	PE30	C
TWI1	TWD1	PE29	C
TWI2	TWCK2	PB30	A
TWI2	TWD2	PB29	A
TWI3	TWCK3	PC26	B
TWI3	TWD3	PC25	B

39.6.2 Power Management

The TWI may be clocked through the Power Management Controller (PMC), thus the user must first configure the PMC to enable the TWI clock.

39.6.3 Interrupt Sources

The TWI has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TWI.

Table 39-5. Peripheral IDs

Instance	ID
TWI0	32
TWI1	33
TWI2	34
TWI3	62

39.7 Functional Description

39.7.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data is transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited (refer to Figure 39-3).

Each transfer begins with a START condition and terminates with a STOP condition (refer to Figure 39-2).

- 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 the STOP condition.

Figure 39-2. START and STOP Conditions

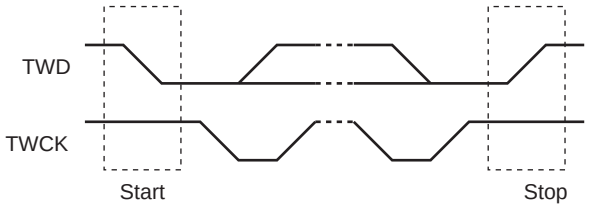
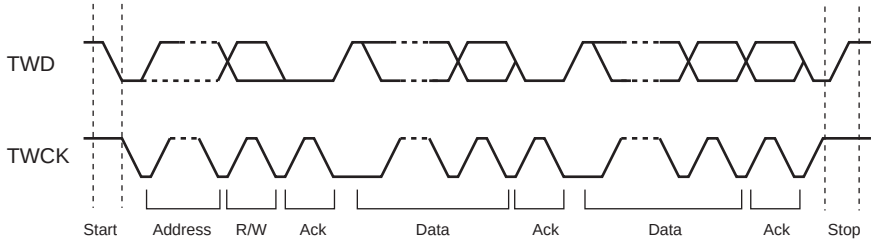


Figure 39-3. Transfer Format



39.7.2 Modes of Operation

The TWI has different modes of operations:

- Master transmitter mode
- Master receiver mode
- Multi-master transmitter mode
- Multi-master receiver mode
- Slave transmitter mode
- Slave receiver mode

These modes are described in the following sections.

39.7.3 Master Mode

39.7.3.1 Definition

The master is the device that starts a transfer, generates a clock and stops it.

39.7.3.2 Programming Master Mode

The following fields must be programmed before entering Master mode:

1. TWI_MMR.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. TWI_CWGR.CKDIV + CHDIV + CLDIV: Clock waveform.
3. TWI_CR.SVDIS: Disables the Slave mode
4. TWI_CR.MSEN: Enables the Master mode

Note: If the TWI is already in Master mode, the device address (DADR) can be configured without disabling the Master mode.

39.7.3.3 Master Transmitter Mode

After the master initiates a START condition when writing into the Transmit Holding register (TWI_THR), it sends a 7-bit slave address, configured in the Master Mode register (DADR in TWI_MMR), to notify the slave device. The bit following the slave address indicates the transfer direction—0 in this case (MREAD = 0 in TWI_MMR).

The TWI transfers require the slave to acknowledge each received byte. 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. If the slave does not acknowledge the byte, then the Not Acknowledge flag (NACK) is set in the TWI Status Register (TWI_SR) of the master and a STOP condition is sent. The NACK flag must be cleared by reading the TWI Status Register (TWI_SR) before the next write into the TWI Transmit Holding Register (TWI_THR). As with the other status bits, an interrupt can be generated if enabled in the Interrupt Enable register (TWI_IER). If the slave acknowledges the byte, the data written in the 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 the TWI_THR.

When no more data is written into the TWI_THR, the master generates a STOP condition to end the transfer. A TXCOMP bit value of one in the TWI_SR indicates that the transfer has completed. Refer to Figure 39-4, Figure 39-5, and Figure 39-6.

To clear the TXRDY flag, first set the bit TWI_CR.MSDIS, then set the bit TWI_CR.MSEN.

Figure 39-4. Master Write with One Data Byte

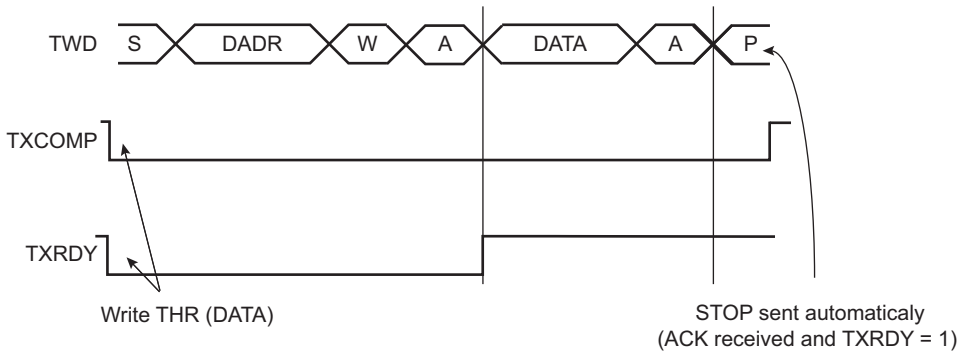


Figure 39-5. Master Write with Multiple Data Byte

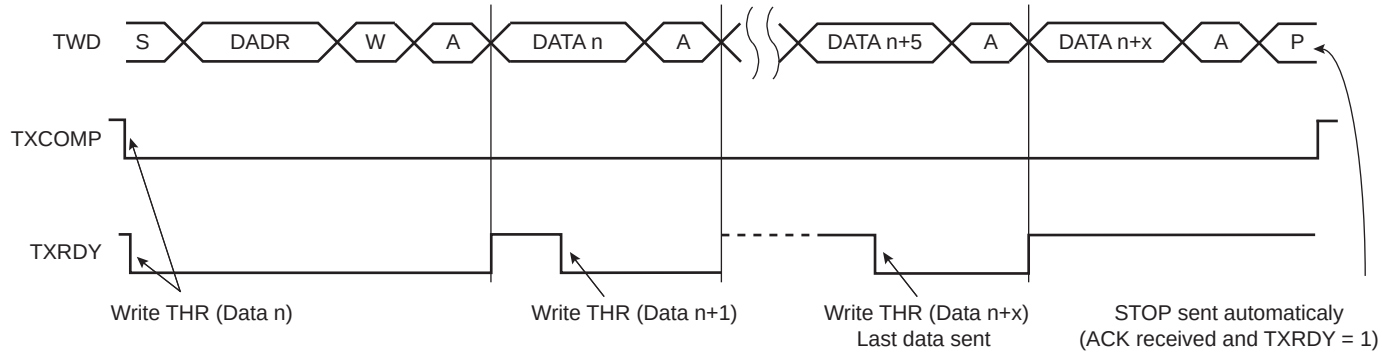
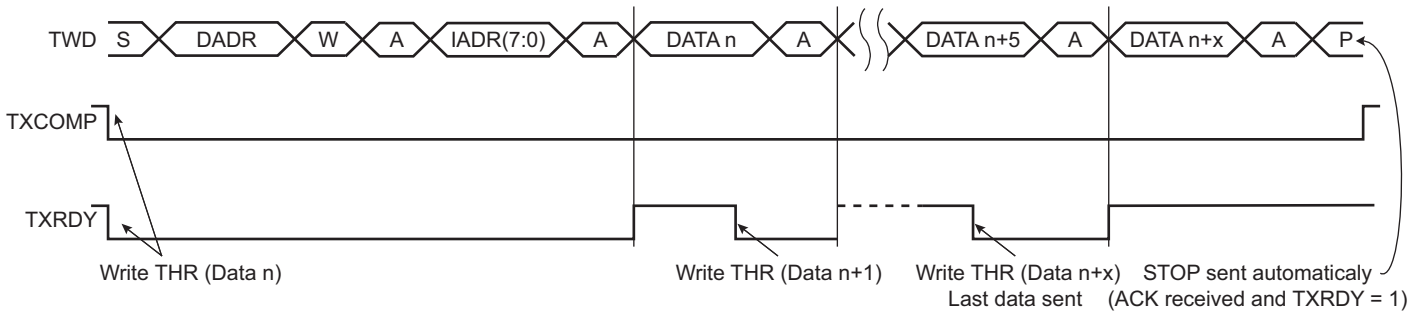


Figure 39-6. Master Write with One Byte Internal Address and Multiple Data Bytes



39.7.3.4 Master Receiver Mode

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 (MREAD = 1 in TWI_MMR). 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 NACK bit in the TWI_SR 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. Refer to Figure 39-7. When the RXRDY bit is set in the TWI_SR, a character has been received in the Receive Holding Register (TWI_RHR). The RXRDY bit is reset when reading the 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. Refer to Figure 39-7. 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. Refer to Figure 39-8. For internal address usage, refer to Section 39.7.3.5.

If the Receive Holding Register (TWI_RHR) is full (RXRDY high) and the master is receiving data, the serial clock line is tied low before receiving the last bit of the data and until the TWI_RHR is read. Once the TWI_RHR is read, the master stops stretching the serial clock line and ends the data reception. Refer to Figure 39-9.

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 is not completed until TWI_RHR is read. The last access stops on the next-to-last bit. When the TWI_RHR is read, the STOP bit command must be sent within a period of half a bit only, otherwise another read access might occur (spurious access).

A possible workaround is to set the STOP bit before reading the TWI_RHR on the next-to-last access (within the interrupt handler).

Figure 39-7. Master Read with One Data Byte

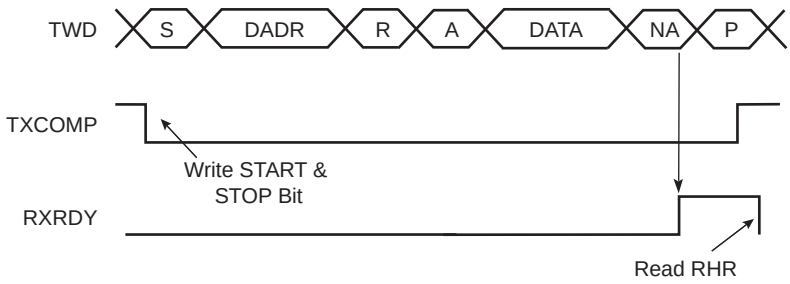


Figure 39-8. Master Read with Multiple Data Bytes

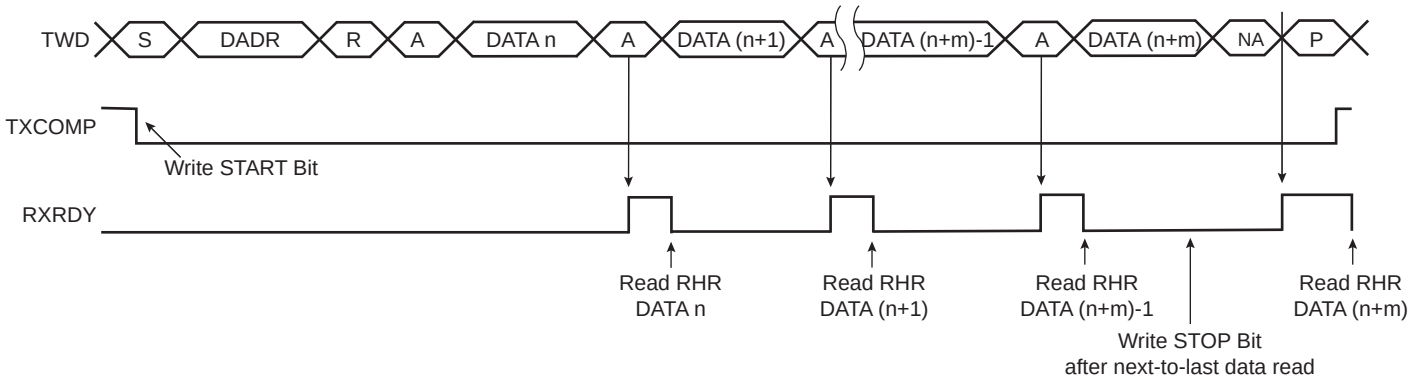
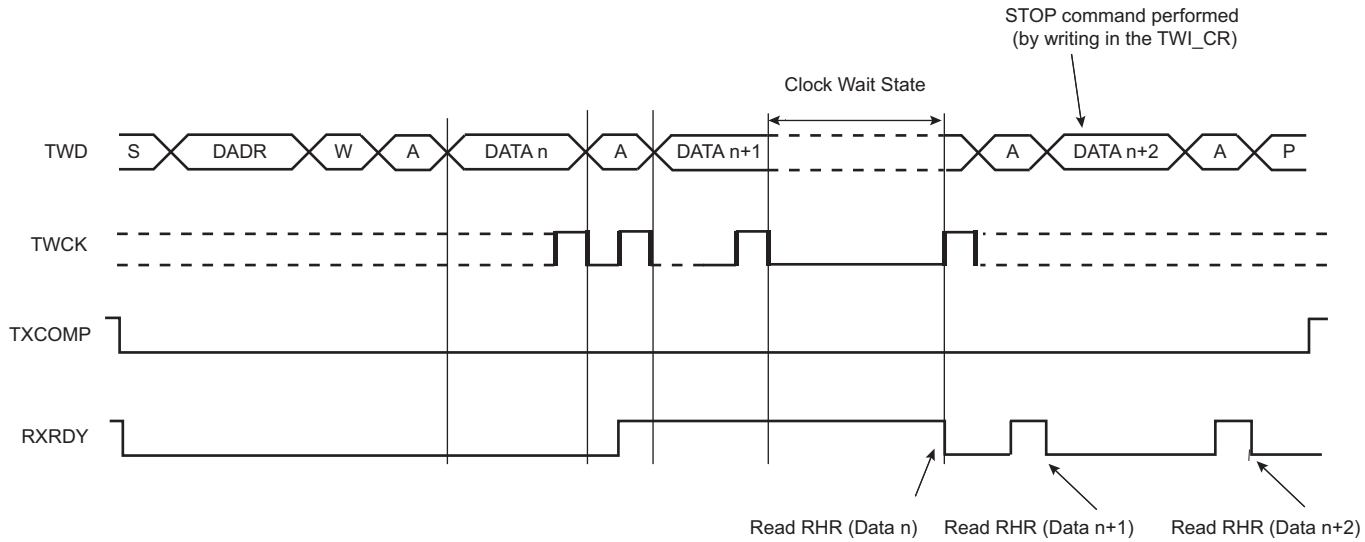


Figure 39-9. Master Read Wait State with Multiple Data Bytes



39.7.3.5 Internal Address

The TWI can perform transfers with 7-bit slave address devices and 10-bit slave address devices.

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. Refer to Figure 39-11. Refer to Figure 39-10 and Figure 39-12 for master write operation with internal address.

The three internal address bytes are configurable through the Master Mode register (TWI_MMR).

If the slave device supports only a 7-bit address, i.e., no internal address, IADRSZ must be set to 0.

Table 39-6 shows the abbreviations used in Figure 39-10 and Figure 39-11.

Table 39-6. Abbreviations

Abbreviation	Definition
S	Start
Sr	Repeated Start
P	Stop
W	Write
R	Read
A	Acknowledge
NA	Not Acknowledge
DADR	Device Address
IADR	Internal Address

Figure 39-10. Master Write with One, Two or Three Bytes Internal Address and One Data Byte

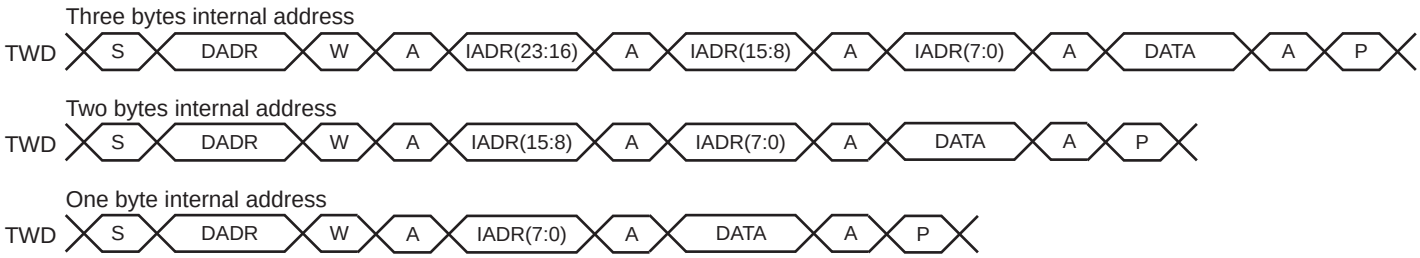
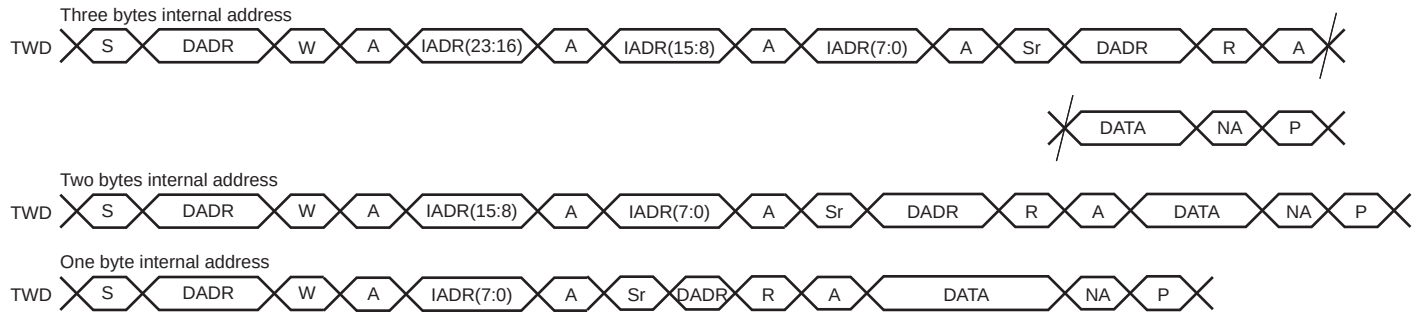


Figure 39-11. Master Read with One, Two or Three Bytes Internal Address and One Data Byte



10-bit Slave Addressing

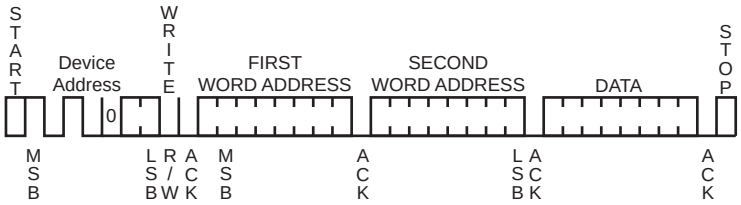
For a slave address higher than seven bits, the user must configure the address size **(**IADRSZ) and set the other slave address bits in the Internal Address register (TWI_IADR). The two remaining internal address bytes, IADR[15:8] and IADR[23:16] can be used the same way as in 7-bit slave addressing.

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 TWI_IADR with b3 b4 b5 b6 b7 b8 b9 b10 (b10 is the LSB of the 10-bit address)

Figure 39-12 below shows a byte write to a memory device. This demonstrates the use of internal addresses to access the device.

Figure 39-12. Internal Address Usage



39.7.3.6 Using the DMA Controller

The use of the DMA significantly reduces the CPU load.

To ensure correct implementation, proceed as follows.

Data Transmit with the DMA

1. Initialize the DMA (channels, memory pointers, size -1, etc.).
2. Configure the Master mode (DADR, CKDIV, etc.) or Slave mode.
3. Enable the DMA.
4. Wait for the DMA buffer transfer complete flag.
5. Disable the DMA.
6. Wait for the TXRDY flag in TWI_SR.
7. Set the STOP bit in TWI_CR.
8. Write the last character in TWI_THR.
9. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWI_SR.

Data Receive with the DMA

The DMA transfer size must be defined with the buffer size minus 2. The two remaining characters must be managed without DMA to ensure that the exact number of bytes are received whatever the system bus latency conditions encountered during the end of buffer transfer period.

In Slave mode, the number of characters to receive must be known in order to configure the DMA.

1. Initialize the DMA (channels, memory pointers, size -2, etc.);
2. Configure the Master mode (DADR, CKDIV, etc.) or Slave mode.
3. Enable the DMA.
4. (Master Only) Write the START bit in the TWI_CR to start the transfer.
5. Wait for the DMA buffer transfer complete flag.
6. Disable the DMA.
7. Wait for the RXRDY flag in the TWI_SR.
8. Set the STOP bit in TWI_CR.
9. Read the penultimate character in TWI_RHR.
10. Wait for the RXRDY flag in the TWI_SR.
11. Read the last character in TWI_RHR.
12. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWI_SR.

39.7.3.7 Read/Write Flowcharts

The flowcharts in the following figures provide examples of 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 (TWI_IER) be configured first.

Figure 39-13. TWI Write Operation with Single Data Byte without Internal Address

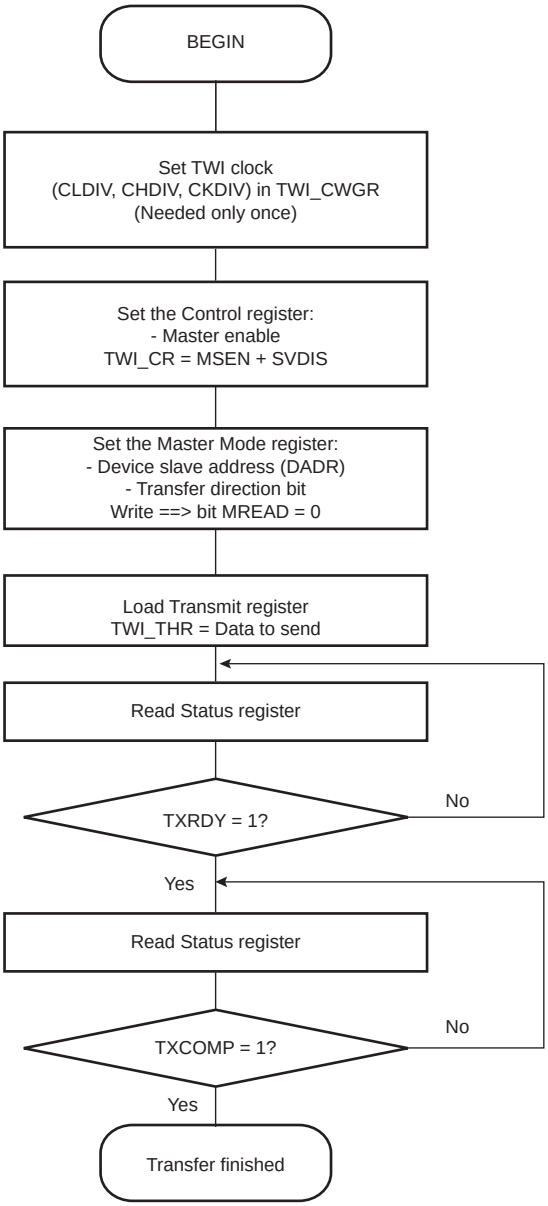


Figure 39-14. TWI Write Operation with Single Data Byte and Internal Address

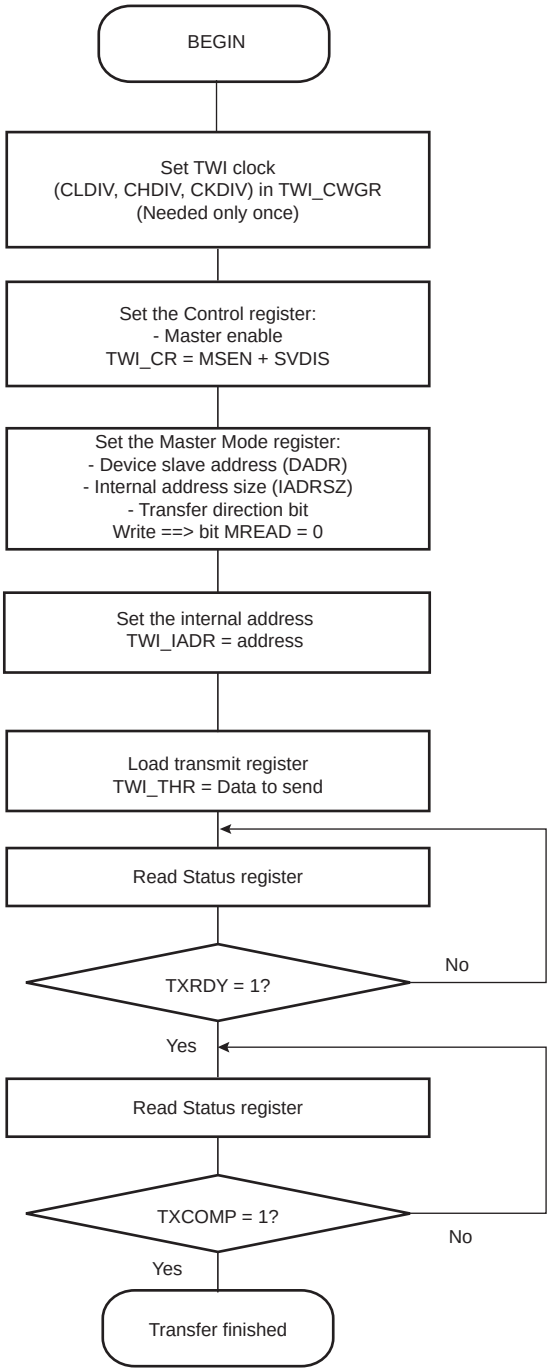


Figure 39-15. TWI Write Operation with Multiple Data Bytes with or without Internal Address

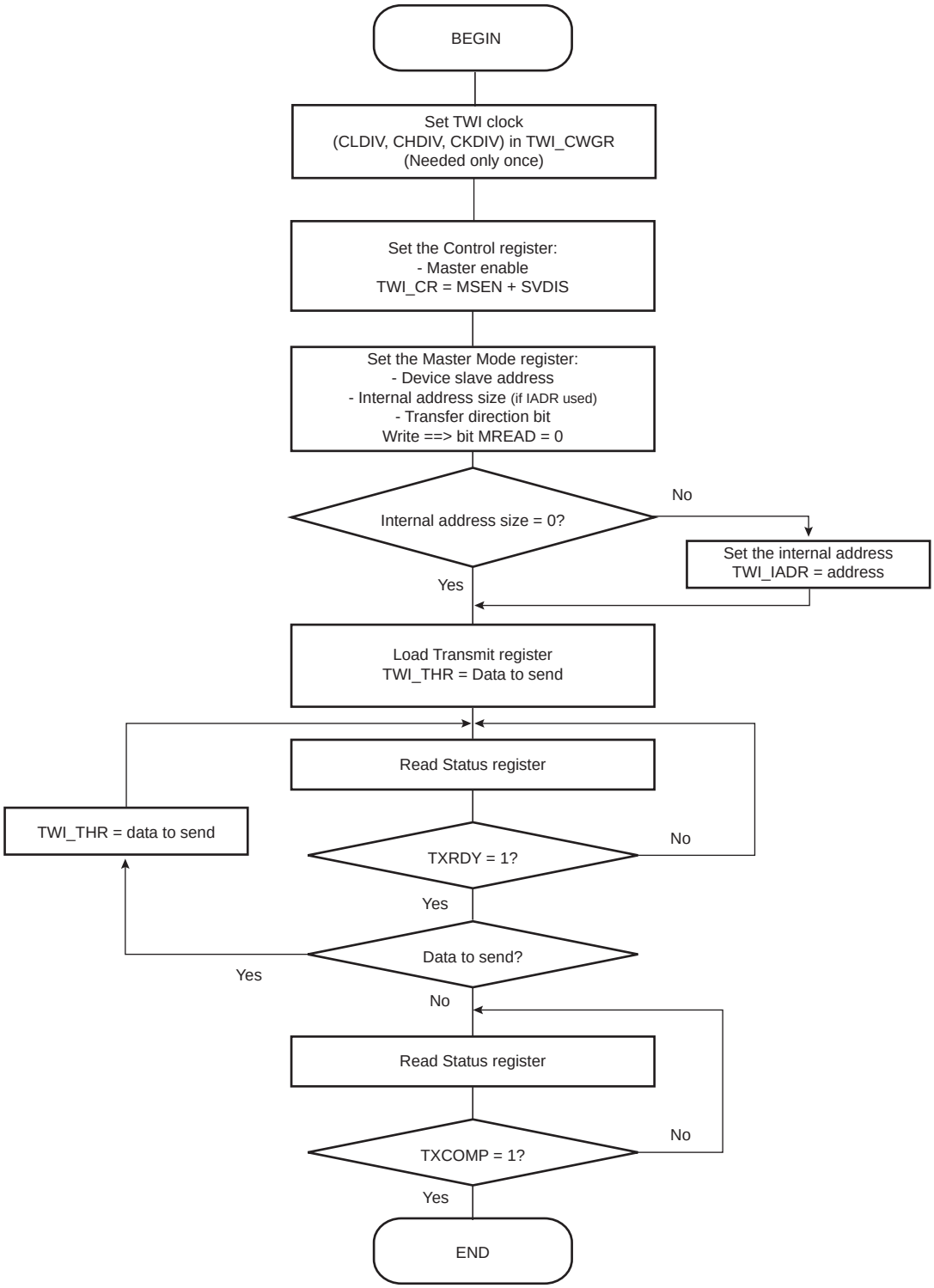


Figure 39-16. TWI Read Operation with Single Data Byte without Internal Address

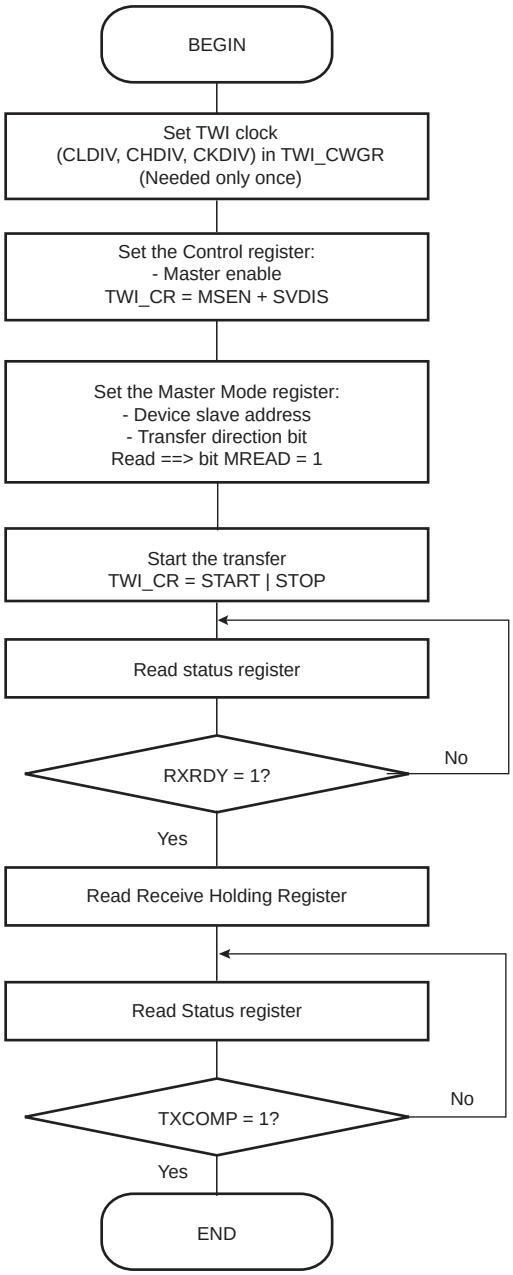


Figure 39-17. TWI Read Operation with Single Data Byte and Internal Address

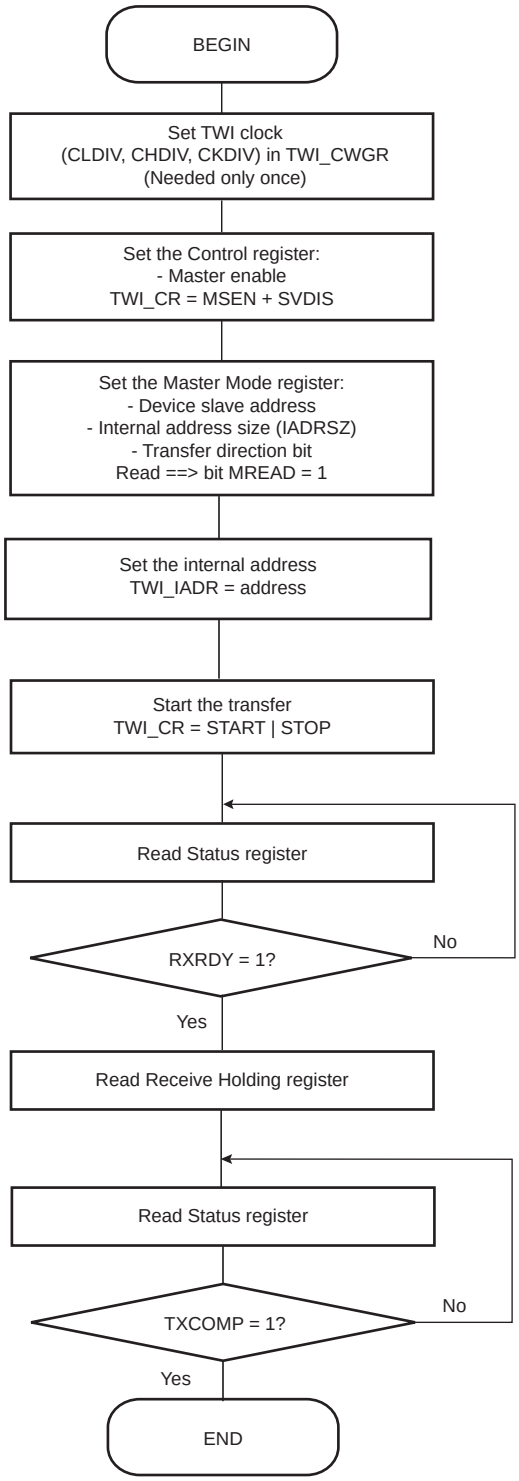
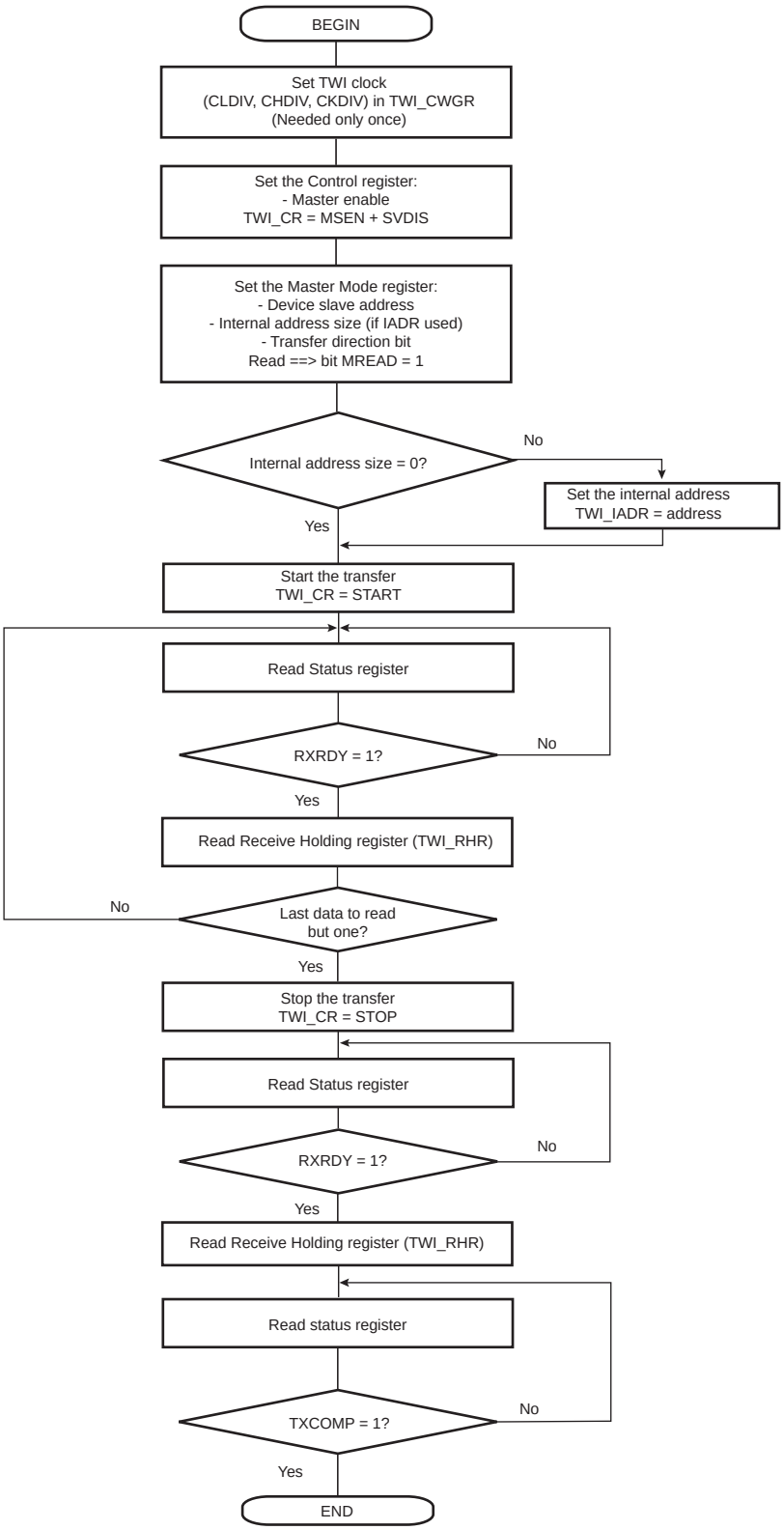


Figure 39-18. TWI Read Operation with Multiple Data Bytes with or without Internal Address



39.7.4 Multi-master Mode

39.7.4.1 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 a master lose arbitration, it stops sending data and listens to the bus in order to detect a stop. When the stop is detected, the master may put its data on the bus by performing arbitration.

Arbitration is illustrated in Figure 39-20.

39.7.4.2 Two Multi-master Modes

Two Multi-master modes may be distinguished:

1. TWI is considered as a master only and will never be addressed.
2. TWI may be either a master or a slave and may be addressed.

Note: Arbitration is supported in both Multi-master modes.

TWI as Master Only

In this mode, TWI is considered as a Master only (MSEN is always 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 (refer to Figure 39-19).

Note: The state of the bus (busy or free) is not shown in the user interface.

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 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 the 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 free, 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 set to 1), the user must program the TWI in Slave mode in case the Master that won the arbitration is required 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: If 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 is set to 1. Then the Master must repeat SADR.

Figure 39-19. Programmer Sends Data While the Bus is Busy

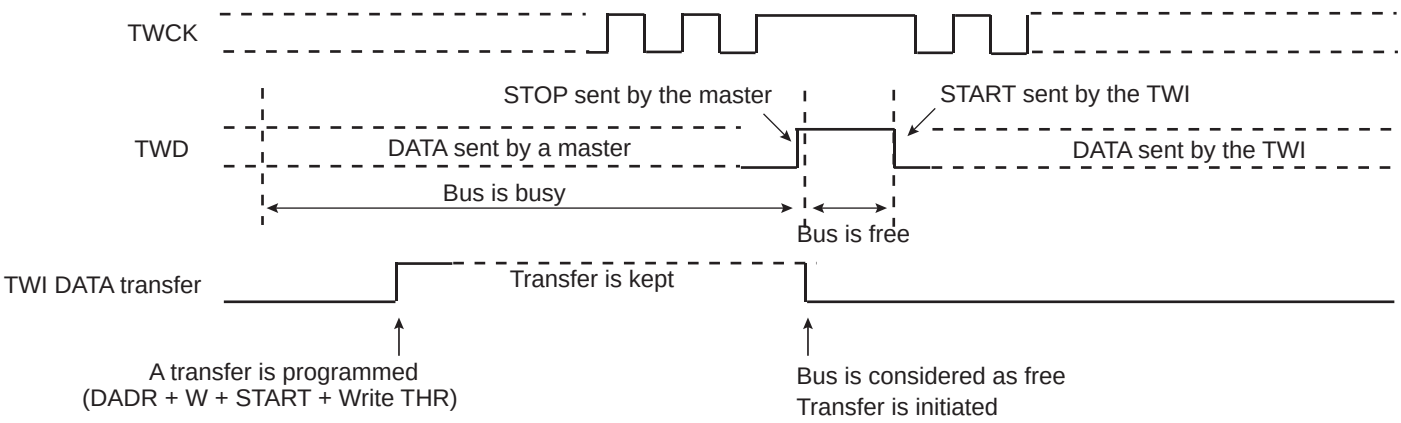
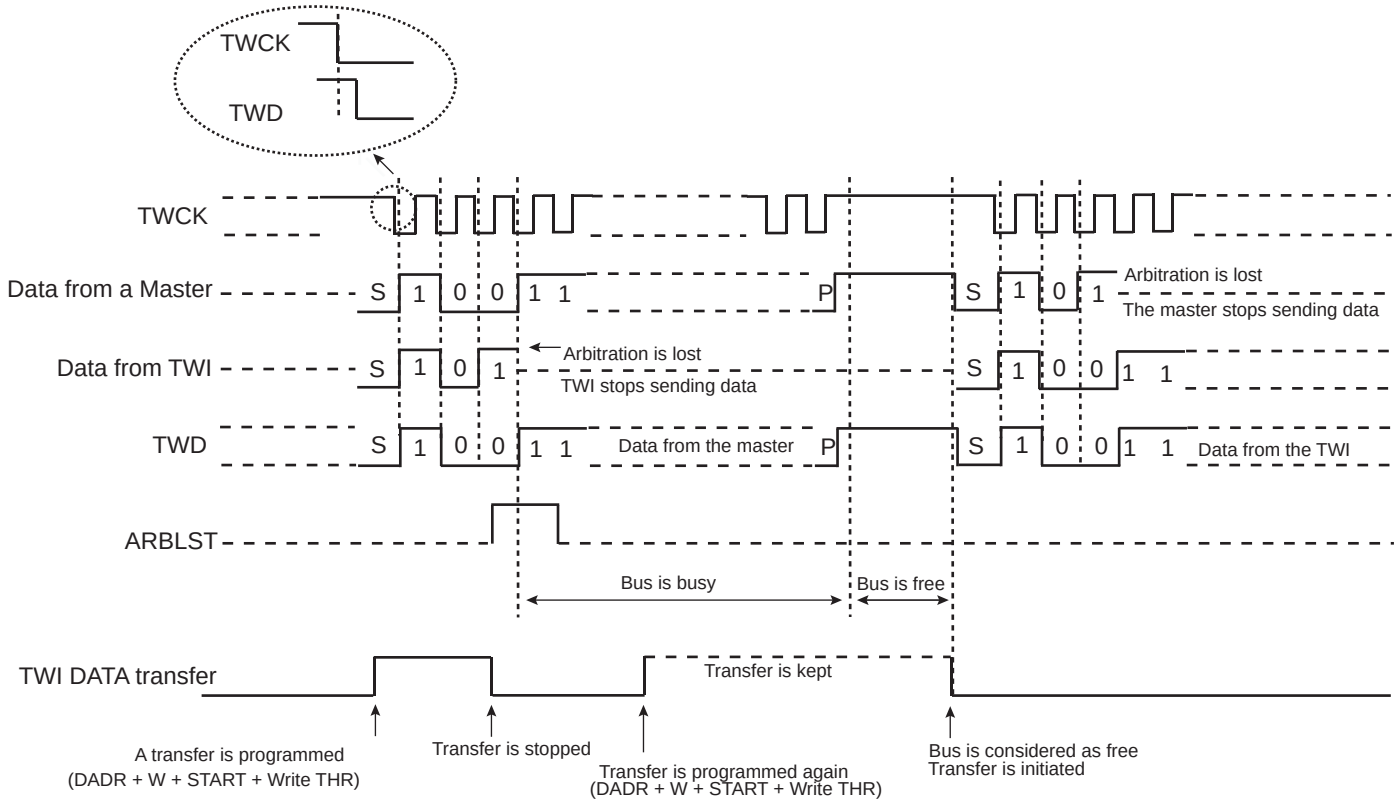
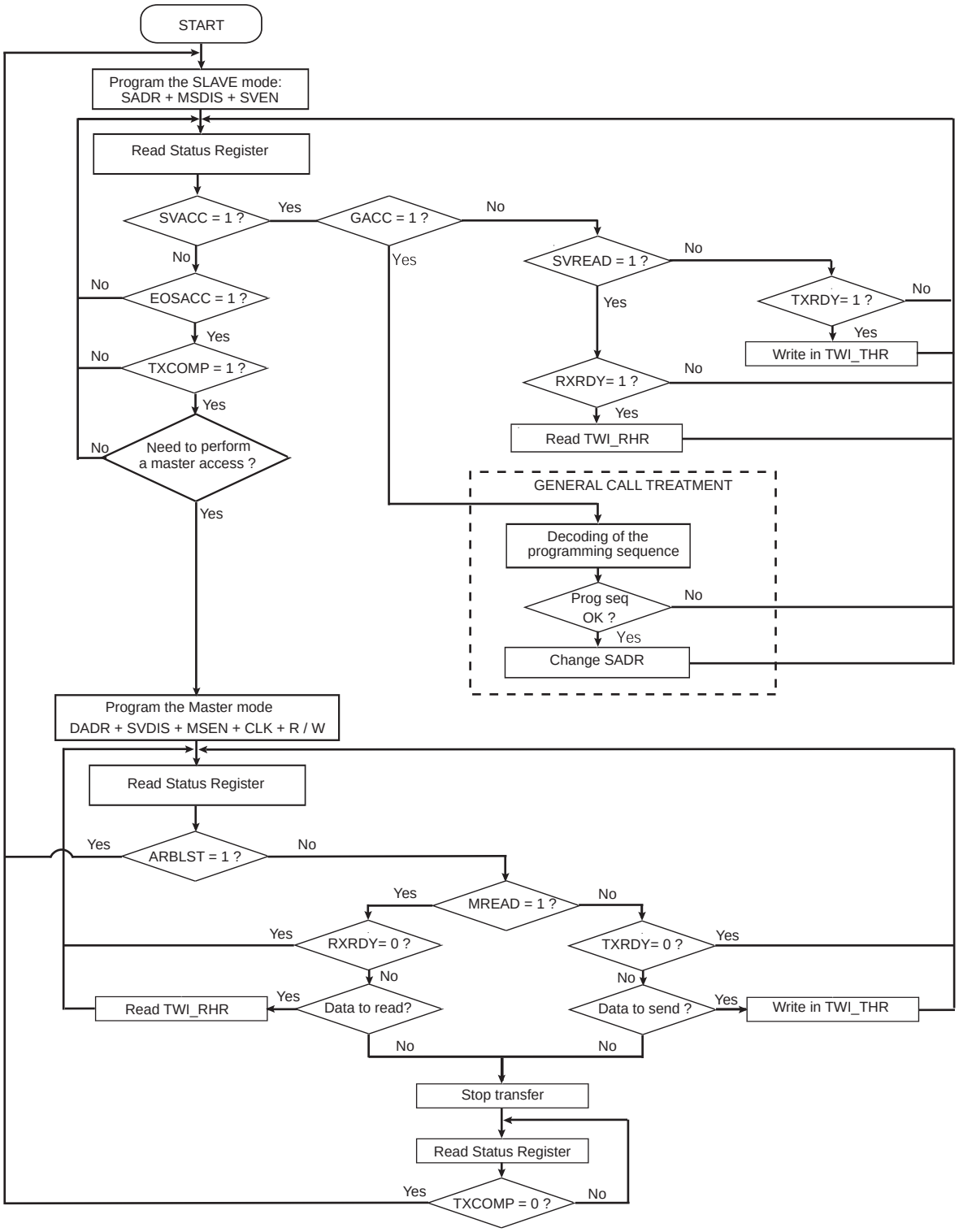


Figure 39-20. Arbitration Cases



The flowchart shown in Figure 39-21 gives an example of read and write operations in Multi-master mode.

Figure 39-21. Multi-master Flowchart



39.7.5 Slave Mode

39.7.5.1 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).

39.7.5.2 Programming Slave Mode

The following fields must be programmed before entering Slave mode:

1. TWI_SMR.SADR: The slave device address is used in order to be accessed by master devices in Read or Write mode.
2. TWI_CR.MSDIS: Disables the Master mode.
3. TWI_CR.SVEN: Enables the Slave mode.

As the device receives the clock, values written in TWI_CWGR are ignored.

39.7.5.3 Receiving Data

After a START or REPEATED START condition is detected and if the address sent by the Master matches with the Slave address programmed in the SADR (Slave Address) field, 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, the EOSACC (End Of Slave Access) flag is set.

Read Sequence

In the case of a read sequence (SVREAD is high), TWI transfers data written in TWI_THR (TWI Transmit Holding Register) until a STOP condition or a REPEATED_START and an address different from SADR is detected. Note that at the end of the read sequence TXCOMP (Transmission Complete) flag is set and SVACC reset.

As soon as data is written in 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.

To clear the TXRDY flag, first set the bit TWI_CR.SVDIS, then set the bit TWI_CR.SVEN.

Refer to Figure 39-22.

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 the TWI_RHR (TWI Receive Holding Register). RXRDY is reset when reading the 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.

Refer to Figure 39-23.

Clock Synchronization Sequence

If TWI_RHR is not read in time, the TWI performs a clock synchronization.

Clock synchronization information is given by the bit SCLWS (Clock Wait State).

Refer to Figure 39-26.

Clock Stretching Sequence

If TWI_THR is not written in time, the TWI performs a clock stretching.

Clock stretching information is given by the bit SCLWS (Clock Wait State).

Refer to Figure 39-25.

General Call

In the case where a GENERAL CALL is performed, the GACC (General Call Access) flag is set.

After GACC is set, the user must interpret the meaning of the GENERAL CALL and decode the new address programming sequence.

Refer to Figure 39-24.

39.7.5.4 Data Transfer

Read Operation

The Read mode is defined as a data requirement from the master.

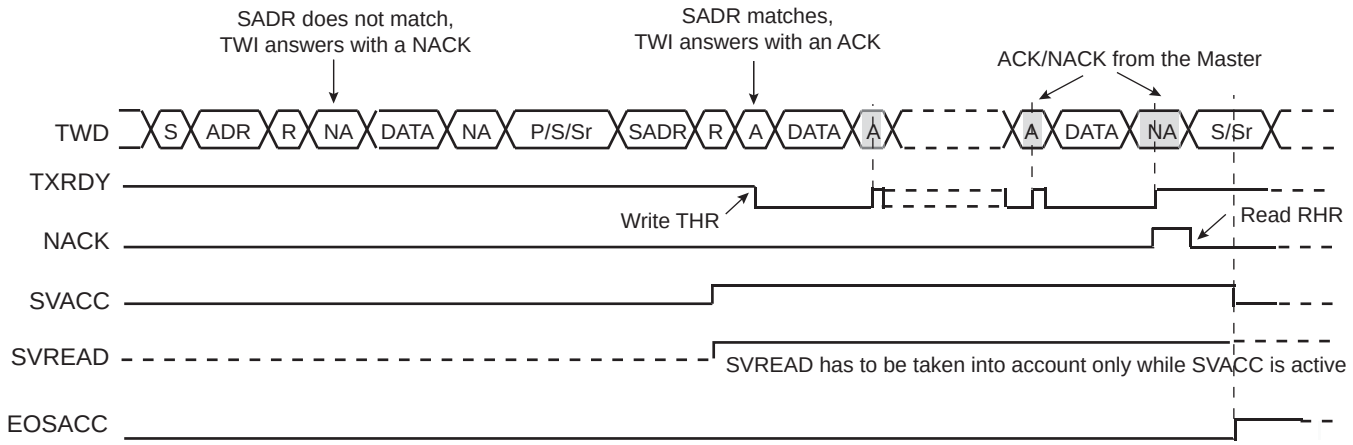
After a START or a REPEATED START condition is detected, the decoding of the address starts. If the slave address (SADR) is decoded, SVACC is set and SVREAD indicates the direction of the transfer.

Until a STOP or REPEATED START condition is detected, TWI continues sending data loaded in the TWI_THR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

Figure 39-22 describes the write operation.

Figure 39-22. Read Access Ordered by a Master



Notes:
1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. TXRDY is reset when data has been transmitted from TWI_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.

Write Operation

The Write mode is defined as a data transmission from the master.

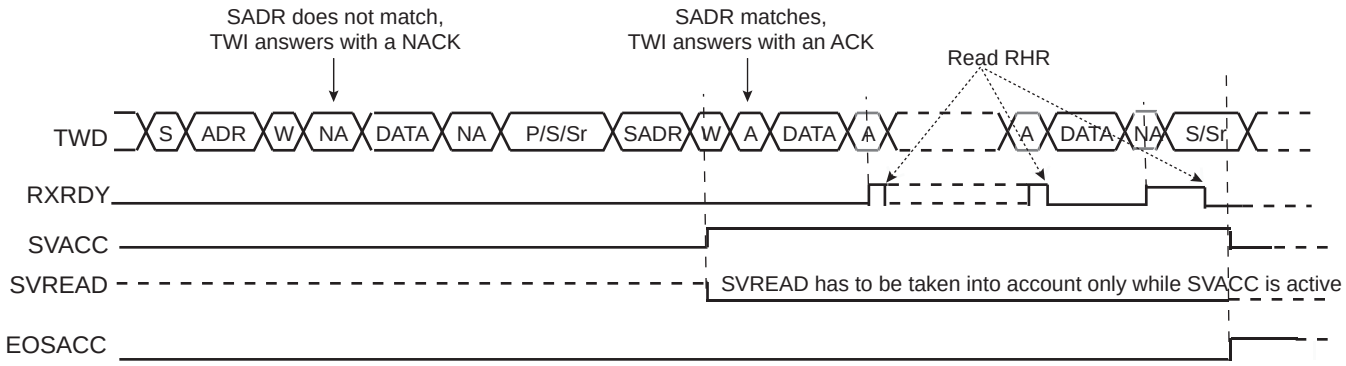
After a START or a REPEATED START, the decoding of the address starts. If the slave address is decoded, SVACC is set and SVREAD indicates the direction of the transfer (SVREAD is low in this case).

Until a STOP or REPEATED START condition is detected, TWI stores the received data in the TWI_RHR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

Figure 39-23 describes the write operation.

Figure 39-23. Write Access Ordered by a Master



Notes: 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 the TWI_RHR and reset when this data is read.

General Call

The general call is performed in order to change the address of the slave.

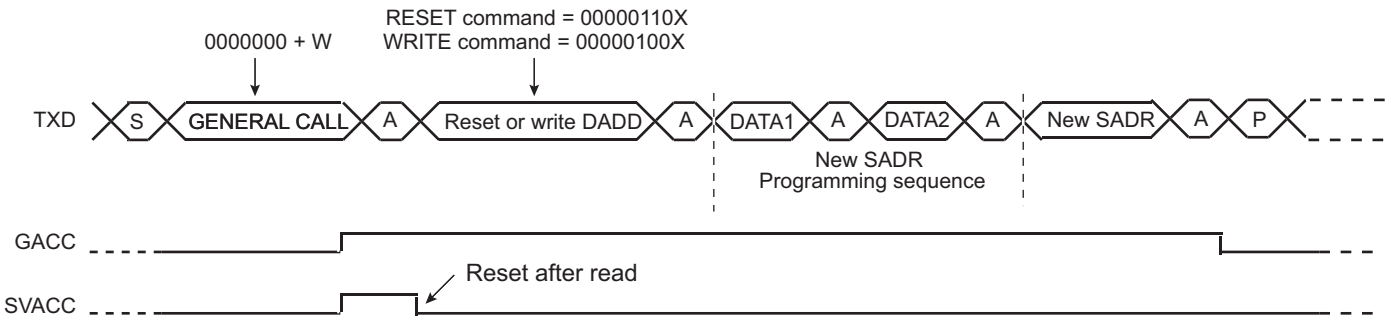
If a GENERAL CALL is detected, GACC is set.

After the detection of GENERAL CALL, it is up to the programmer to decode the commands which come afterwards.

In case of a WRITE command, the programmer has to decode the programming sequence and program a new SADR if the programming sequence matches.

Figure 39-24 describes the GENERAL CALL access.

Figure 39-24. Master Performs a General Call



Note: This method allows the user to create a personal programming sequence by choosing the programming bytes and the number of them. The programming sequence has to be provided to the master.

Clock Synchronization/Stretching

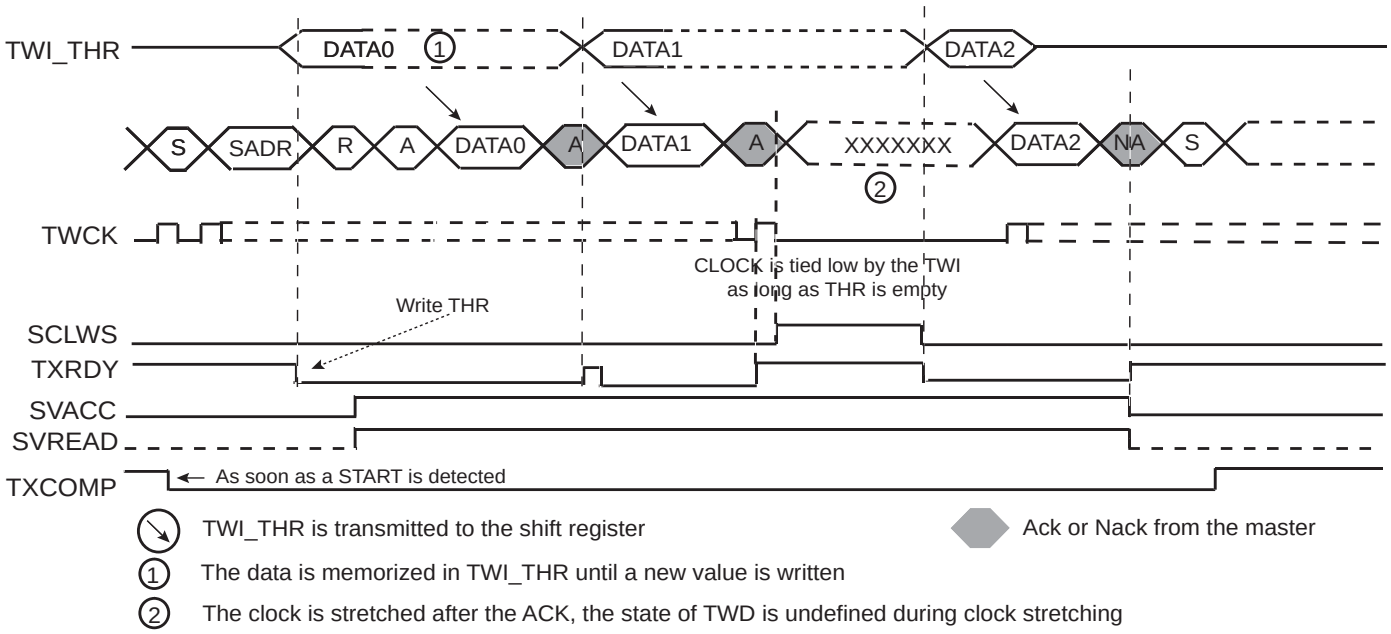
In both Read and Write modes, it may occur that TWI_THR/TWI_RHR buffer is not filled /emptied before transmission/reception of a new character. In this case, to avoid sending/receiving undesired data, a clock stretching/synchronization mechanism is implemented.

Clock Stretching in Read Mode

The clock is tied low during the acknowledge phase 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.

Figure 39-25 describes clock stretching in Read mode.

Figure 39-25. Clock Stretching in Read Mode



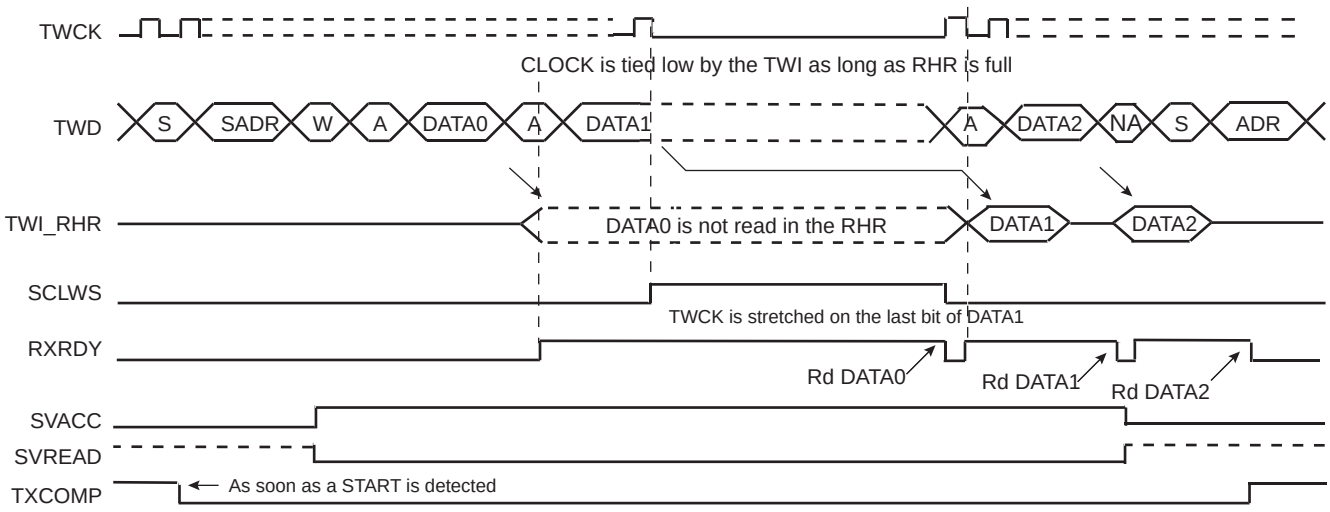
Notes:
1. TXRDY is reset when data has been written in the TWI_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.
2. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
3. SCLWS is automatically set when the clock stretching mechanism is started.

Clock Synchronization in Write Mode

The clock is tied low outside of the acknowledge phase if the internal shifter and the TWI_RHR is full. If a STOP or REPEATED_START condition was not detected, it is tied low until TWI_RHR is read.

Figure 39-26 describes the clock synchronization in Write mode.

Figure 39-26. Clock Synchronization in Write Mode



Notes:
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 synchronization mechanism is started and automatically reset when the mechanism is finished.

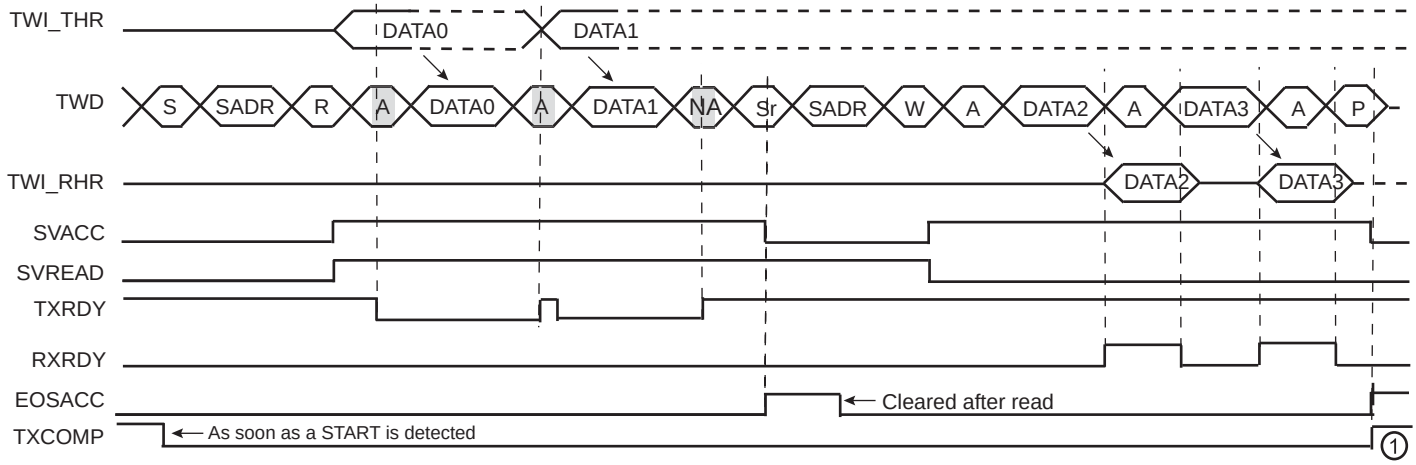
Reversal After a Repeated Start

Reversal of Read to Write

The master initiates the communication by a read command and finishes it by a write command.

Figure 39-27 describes the repeated start + reversal from Read to Write mode.

Figure 39-27. Repeated Start + Reversal from Read to Write Mode



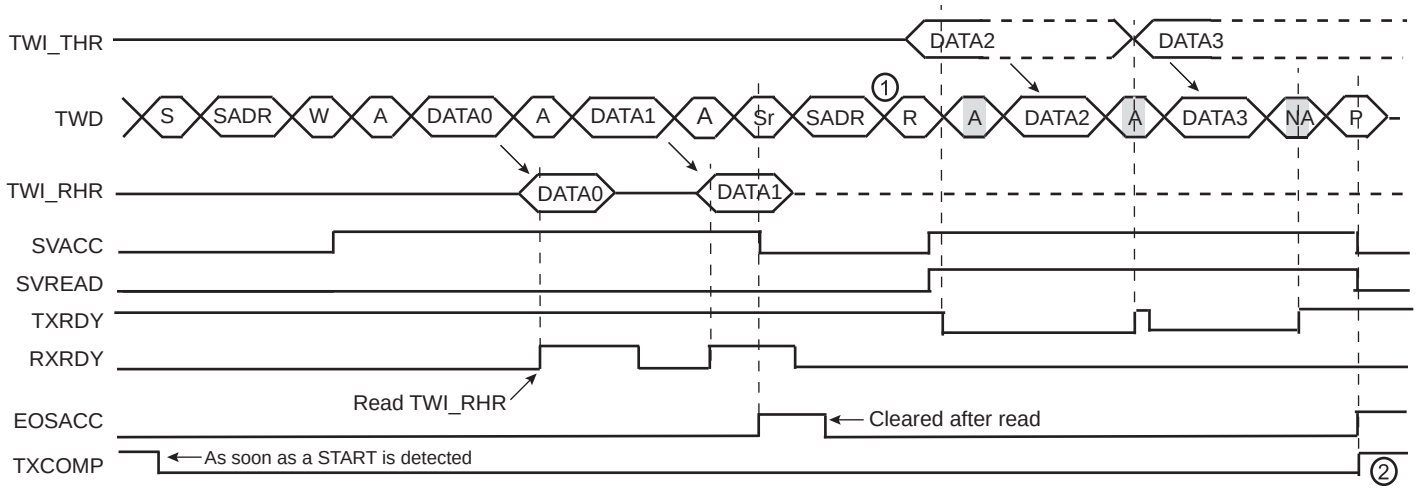
Note: 1. TXCOMP is only set at the end of the transmission because after the repeated start, SADR is detected again.

Reversal of Write to Read

The master initiates the communication by a write command and finishes it by a read command.

Figure 39-28 describes the repeated start + reversal from Write to Read mode.

Figure 39-28. Repeated Start + Reversal from Write to Read Mode



Notes: 1. In this case, if 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.

39.7.5.5 Using the DMA Controller

The use of the DMA significantly reduces the CPU load.

To ensure correct implementation, proceed as follows.

Data Transmit with the DMA

1. Initialize the DMA (channels, memory pointers, size, etc.).
2. Configure the Slave mode.
3. Enable the DMA.
4. Wait for the DMA buffer transfer complete flag.
5. Disable the DMA.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWI_SR.

Data Receive with the DMA

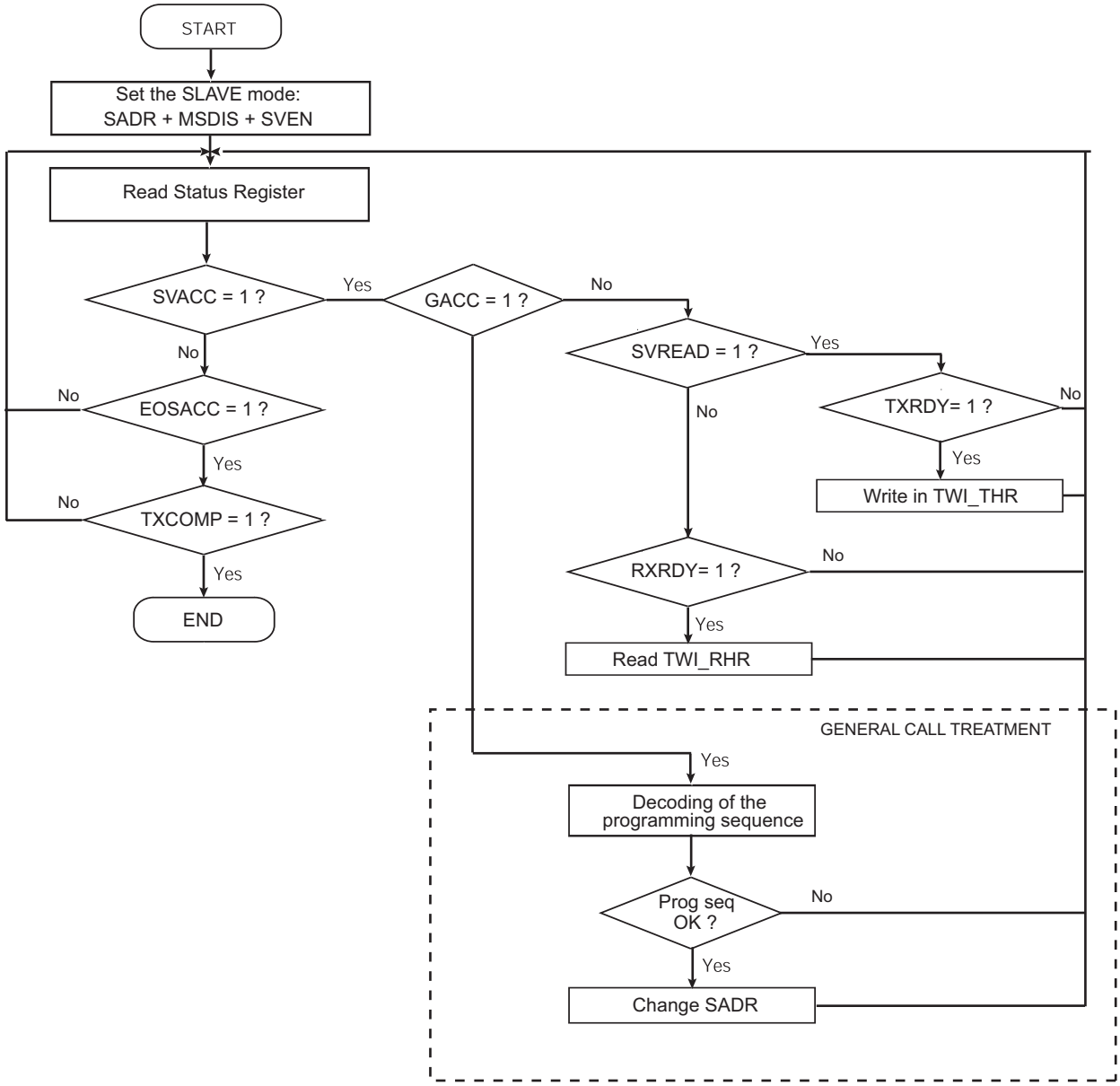
The DMA transfer size must be defined with the buffer size. In Slave mode, the number of characters to be received must be known in order to configure the DMA.

1. Initialize the DMA (channels, memory pointers, size, etc.).
2. Configure the Slave mode.
3. Enable the DMA.
4. Wait for the DMA buffer transfer complete flag.
5. Disable the DMA.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWI_SR.

39.7.5.6 Read Write Flowcharts

The flowchart shown in Figure 39-29 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 (TWI_IER) be configured first.

Figure 39-29. Read Write Flowchart in Slave Mode



39.7.6 Register Write Protection

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

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

The WPVS bit is automatically cleared after reading the TWI_WPSR.

The following registers can be write-protected:

- TWI Slave Mode Register
- TWI Clock Waveform Generator Register

39.8 Two-wire Interface (TWI) User Interface

Table 39-7. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	TWI_CR	Write-only	–
0x04	Master Mode Register	TWI_MMR	Read/Write	0x00000000
0x08	Slave Mode Register	TWI_SMR	Read/Write	0x00000000
0x0C	Internal Address Register	TWI_IADR	Read/Write	0x00000000
0x10	Clock Waveform Generator Register	TWI_CWGR	Read/Write	0x00000000
0x14–0x1C	Reserved	–	–	–
0x20	Status Register	TWI_SR	Read-only	0x0000F009
0x24	Interrupt Enable Register	TWI_IER	Write-only	–
0x28	Interrupt Disable Register	TWI_IDR	Write-only	–
0x2C	Interrupt Mask Register	TWI_IMR	Read-only	0x00000000
0x30	Receive Holding Register	TWI_RHR	Read-only	0x00000000
0x34	Transmit Holding Register	TWI_THR	Write-only	–
0x38–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	TWI_WPMR	Read/Write	0x00000000
0xE8	Write Protection Status Register	TWI_WPSR	Read-only	0x00000000
0xEC–0xFC	Reserved	–	–	–

Note: All unlisted offset values are considered as “reserved”.

39.8.1 TWI Control Register

Name: TWI_CR

Address: 0xF8014000 (0), 0xF8018000 (1), 0xF8024000 (2), 0xFC038000 (3)

Access: Write-only

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
SWRST	–	SVDIS	SVEN	MSDIS	MSEN	STOP	START

- **START: Send a START Condition**

0: No effect.

1: A frame beginning with a START bit is transmitted according to the features defined in the TWI Master Mode Register (TWI_MMR).

This action is necessary for the TWI 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 (TWI_THR).

- **STOP: Send a STOP Condition**

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, the START and STOP must both 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 multiple data write operation, when both THR and internal shifter are empty, a STOP condition is sent automatically.

- **MSEN: TWI Master Mode Enabled**

0: No effect.

1: Enables the Master mode (MSDIS must be written to 0).

Note: Switching from Slave to Master mode is only permitted when TXCOMP = 1.

- **MSDIS: TWI Master Mode Disabled**

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.

- **SVEN: TWI Slave Mode Enabled**

0: No effect.

1: Enables the Slave mode (SVDIS must be written to 0)

Note: Switching from master to Slave mode is only permitted when TXCOMP = 1.

- **SVDIS: TWI Slave Mode Disabled**

0: No effect.

1: The Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in case of read operation. In write operation, the character being transferred must be completely received before disabling.

- **SWRST: Software Reset**

0: No effect.

1: Equivalent to a system reset.

39.8.2 TWI Master Mode Register

Name: TWI_MMR

Address: 0xF8014004 (0), 0xF8018004 (1), 0xF8024004 (2), 0xFC038004 (3)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	DADR						

15	14	13	12	11	10	9	8
–	–	–	MREAD	–	–	IADRSZ	

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **IADRSZ: 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

- **MREAD: Master Read Direction**

0: Master write direction.

1: Master read direction.

- **DADR: Device Address**

The device address is used to access slave devices in Read or Write mode. These bits are only used in Master mode.

39.8.3 TWI Slave Mode Register

Name: TWI_SMR

Address: 0xF8014008 (0), 0xF8018008 (1), 0xF8024008 (2), 0xFC038008 (3)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	SADR						

15	14	13	12	11	10	9	8
–	–	–	–	–	–		

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

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

- **SADR: 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.

39.8.4 TWI Internal Address Register

Name: TWI_IADR

Address: 0xF801400C (0), 0xF801800C (1), 0xF802400C (2), 0xFC03800C (3)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
IADR							

15	14	13	12	11	10	9	8
IADR							

7	6	5	4	3	2	1	0
IADR							

- **IADR: Internal Address**

0, 1, 2 or 3 bytes depending on IADRSZ.

39.8.5 TWI Clock Waveform Generator Register

Name: TWI_CWGR

Address: 0xF8014010 (0), 0xF8018010 (1), 0xF8024010 (2), 0xFC038010 (3)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	HOLD				

23	22	21	20	19	18	17	16
–	–	–	–	–	CKDIV		

15	14	13	12	11	10	9	8
CHDIV							

7	6	5	4	3	2	1	0
CLDIV							

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

TWI_CWGR is only used in Master mode.

- **CLDIV: Clock Low Divider**

The TWCK low period is defined as follows: $t_{low} = ((CLDIV \times 2^{CKDIV}) + 4 \times t_{peripheral\ clock}$

- **CHDIV: Clock High Divider**

The TWCK high period is defined as follows: $t_{high} = ((CHDIV \times 2^{CKDIV}) + 4 \times t_{peripheral\ clock}$

- **CKDIV: Clock Divider**

The CKDIV field is used to increase both TWCK high and low periods.

- **HOLD: TWD Hold Time versus TWCK falling**

TWD is kept unchanged after TWCK falling edge for a period of (HOLD + 3) $\times$ $t_{peripheral\ clock}$.

39.8.6 TWI Status Register

Name: TWI_SR

Address: 0xF8014020 (0), 0xF8018020 (1), 0xF8024020 (2), 0xFC038020 (3)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCLWS	ARBLST	NACK
7	6	5	4	3	2	1	0
–	OVRE	GACC	SVACC	SVREAD	TXRDY	RXRDY	TXCOMP

- **TXCOMP: Transmission Completed (cleared by writing 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 Figure 39-8.

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 Figure 39-25, Figure 39-26, Figure 39-27 and Figure 39-28.

- **RXRDY: Receive Holding Register Ready (cleared by reading TWI_RHR)**

0: No character has been received since the last TWI_RHR read operation.

1: A byte has been received in the TWI_RHR since the last read.

RXRDY behavior in Master mode can be seen in Figure 39-8.

RXRDY behavior in Slave mode can be seen in Figure 39-23, Figure 39-26, Figure 39-27 and Figure 39-28.

- **TXRDY: Transmit Holding Register Ready (cleared by writing TWI_THR)**

TXRDY used in Master mode:

0: The transmit holding register has not been transferred into internal shifter. Set to 0 when writing into TWI_THR.

1: As soon as a data byte is transferred from 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 (enable TWI).

TXRDY behavior in Master mode can be seen in Figure 39.7.3.3.

TXRDY used in Slave mode:

0: As soon as data is written in the TWI_THR, until this data has been transmitted and acknowledged (ACK or NACK).

1: It indicates that the 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 programmer must not fill TWI_THR to avoid losing it.

TXRDY behavior in Slave mode can be seen in Figure 39-22, Figure 39-25, Figure 39-27 and Figure 39-28.

- **SVREAD: Slave Read**

This bit is only used in Slave mode. When SVACC is low (no Slave access has been detected) SVREAD is irrelevant.

0: Indicates that a write access is performed by a Master.

1: Indicates that a read access is performed by a Master.

SVREAD behavior can be seen in Figure 39-22, Figure 39-23, Figure 39-27 and Figure 39-28.

- **SVACC: Slave Access**

This bit is only used in Slave mode.

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.

SVACC behavior can be seen in Figure 39-22, Figure 39-23, Figure 39-27 and Figure 39-28.

- **GACC: General Call Access (cleared on read)**

This bit is only used in Slave mode.

0: No General Call has been detected.

1: A General Call has been detected. After the detection of General Call, if need be, the programmer may acknowledge this access and decode the following bytes and respond according to the value of the bytes.

GACC behavior can be seen in Figure 39-24.

- **OVRE: Overrun Error (cleared on read)**

This bit is only used in Master mode.

0: TWI_RHR has not been loaded while RXRDY was set

1: TWI_RHR has been loaded while RXRDY was set. Reset by read in TWI_SR when TXCOMP is set.

- **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 byte or an 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 programmer must not fill TWI_THR even if TXRDY is set, because that 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.

- **ARBLST: Arbitration Lost (cleared on read)**

This bit is only used in Master mode.

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.

- **SCLWS: Clock Wait State**

This bit is only used in Slave mode.

0: The clock is not stretched.

1: The clock is stretched. TWI_THR / TWI_RHR buffer is not filled / emptied before transmission / reception of a new character.

SCLWS behavior can be seen in Figure 39-25 and Figure 39-26.

- **EOSACC: End Of Slave Access (cleared on read)**

This bit is only used in Slave mode.

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.

EOSACC behavior can be seen in Figure 39-27 and Figure 39-28.

39.8.7 TWI Interrupt Enable Register

Name: TWI_IER

Address: 0xF8014024 (0), 0xF8018024 (1), 0xF8024024 (2), 0xFC038024 (3)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCL_WS	ARBLST	NACK
7	6	5	4	3	2	1	0
–	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

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

0: No effect.

1: Enables the corresponding interrupt.

- **TXCOMP: Transmission Completed Interrupt Enable**
- **RXRDY: Receive Holding Register Ready Interrupt Enable**
- **TXRDY: Transmit Holding Register Ready Interrupt Enable**
- **SVACC: Slave Access Interrupt Enable**
- **GACC: General Call Access Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**
- **NACK: Not Acknowledge Interrupt Enable**
- **ARBLST: Arbitration Lost Interrupt Enable**
- **SCL_WS: Clock Wait State Interrupt Enable**
- **EOSACC: End Of Slave Access Interrupt Enable**

39.8.8 TWI Interrupt Disable Register

Name: TWI_IDR

Address: 0xF8014028 (0), 0xF8018028 (1), 0xF8024028 (2), 0xFC038028 (3)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCL_WS	ARBLST	NACK
7	6	5	4	3	2	1	0
–	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

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

0: No effect.

1: Disables the corresponding interrupt.

- **TXCOMP: Transmission Completed Interrupt Disable**
- **RXRDY: Receive Holding Register Ready Interrupt Disable**
- **TXRDY: Transmit Holding Register Ready Interrupt Disable**
- **SVACC: Slave Access Interrupt Disable**
- **GACC: General Call Access Interrupt Disable**
- **OVRE: Overrun Error Interrupt Disable**
- **NACK: Not Acknowledge Interrupt Disable**
- **ARBLST: Arbitration Lost Interrupt Disable**
- **SCL_WS: Clock Wait State Interrupt Disable**
- **EOSACC: End Of Slave Access Interrupt Disable**

39.8.9 TWI Interrupt Mask Register

Name: TWI_IMR

Address: 0xF801402C (0), 0xF801802C (1), 0xF802402C (2), 0xFC03802C (3)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCL_WS	ARBLST	NACK
7	6	5	4	3	2	1	0
–	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

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.

- **TXCOMP: Transmission Completed Interrupt Mask**
- **RXRDY: Receive Holding Register Ready Interrupt Mask**
- **TXRDY: Transmit Holding Register Ready Interrupt Mask**
- **SVACC: Slave Access Interrupt Mask**
- **GACC: General Call Access Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **NACK: Not Acknowledge Interrupt Mask**
- **ARBLST: Arbitration Lost Interrupt Mask**
- **SCL_WS: Clock Wait State Interrupt Mask**
- **EOSACC: End Of Slave Access Interrupt Mask**

39.8.10 TWI Receive Holding Register

Name: TWI_RHR

Address: 0xF8014030 (0), 0xF8018030 (1), 0xF8024030 (2), 0xFC038030 (3)

Access: Read-only

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
RXDATA							

- **RXDATA: Master or Slave Receive Holding Data**

39.8.11 TWI Transmit Holding Register

Name: TWI_THR

Address: 0xF8014034 (0), 0xF8018034 (1), 0xF8024034 (2), 0xFC038034 (3)

Access: Write-only

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
TXDATA							

- **TXDATA: Master or Slave Transmit Holding Data**

39.8.12 TWI Write Protection Mode Register

Name: TWI_WPMR

Address: 0xF80140E4 (0), 0xF80180E4 (1), 0xF80240E4 (2), 0xFC0380E4 (3)

Access: Read/Write

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

- **WPEN: Write Protection Enable**

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).

Refer to Section 39.7.6 “Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x545749	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0

39.8.13 TWI Write Protection Status Register

Name: TWI_WPSR

Address: 0xF80140E8 (0), 0xF80180E8 (1), 0xF80240E8 (2), 0xFC0380E8 (3)

Access: Read-only

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

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the TWI_WPSR.

1: A write protection violation has occurred since the last read of the TWI_WPSR. If this violation is an unauthorized attempt to write a protected register, the violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC shows the register address offset at which a write access has been attempted.

40. Synchronous Serial Controller (SSC)

40.1 Description

The Synchronous Serial Controller (SSC) provides a synchronous communication link with external devices. It supports many serial synchronous communication protocols generally used in audio and telecom applications such as I2S, Short Frame Sync, Long Frame Sync, etc.

The SSC contains an independent receiver and transmitter and a common clock divider. The receiver and the transmitter each interface with three signals: the TD/RD signal for data, the TK/RK signal for the clock and the TF/RF signal for the Frame Sync. The transfers can be programmed to start automatically or on different events detected on the Frame Sync signal.

The SSC high-level of programmability and its use of DMA enable a continuous high bit rate data transfer without processor intervention.

Featuring connection to the DMA, the SSC enables interfacing with low processor overhead to:

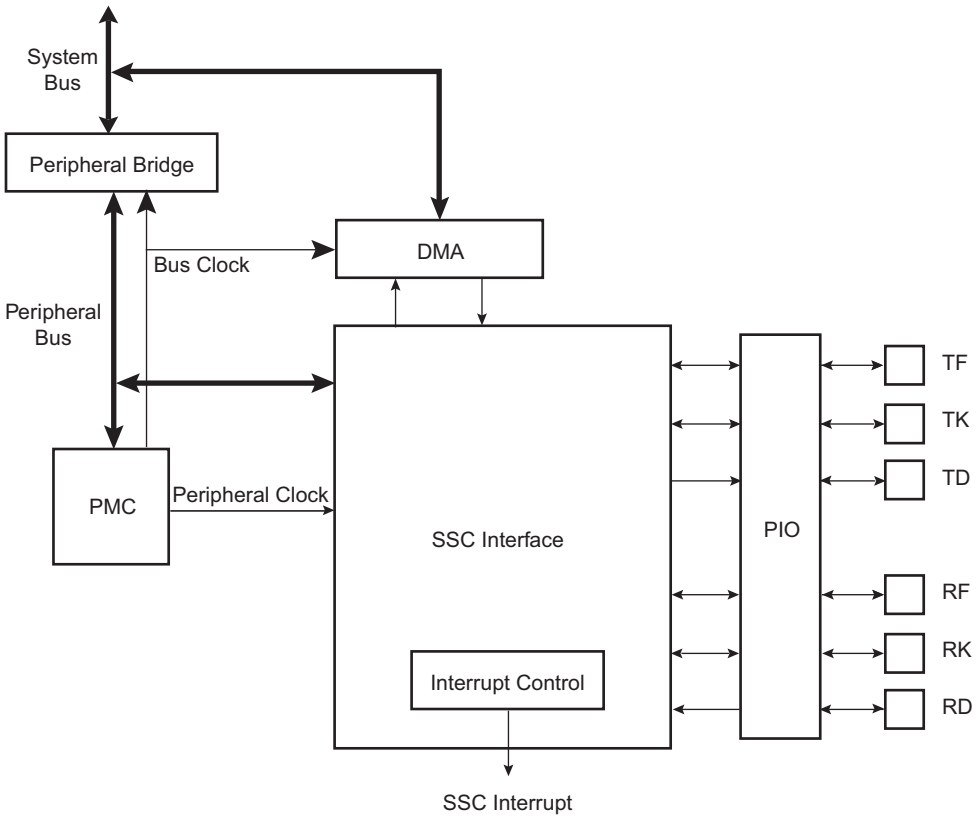
- Codecs in Master or Slave mode,
- DAC through dedicated serial interface, particularly I2S,
- Magnetic card reader.

40.2 Embedded Characteristics

- Provides Serial Synchronous Communication Links Used in Audio and Telecom Applications
- Contains an Independent Receiver and Transmitter and a Common Clock Divider
- Interfaced with the DMA Controller (DMAC) to Reduce Processor Overhead
- Offers a Configurable Frame Sync and Data Length
- Receiver and Transmitter Can be Programmed to Start Automatically or on Detection of Different Events on the Frame Sync Signal
- Receiver and Transmitter Include a Data Signal, a Clock Signal and a Frame Sync Signal

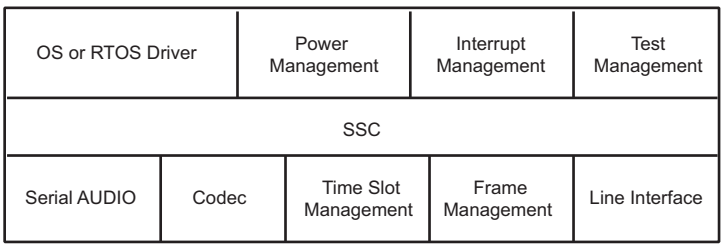
40.3 Block Diagram

Figure 40-1. Block Diagram



40.4 Application Block Diagram

Figure 40-2. Application Block Diagram



40.5 SSC Application Examples

The SSC can support several serial communication modes used in audio or high speed serial links. Some standard applications are shown in the following figures. All serial link applications supported by the SSC are not listed here.

Figure 40-3. Audio Application Block Diagram

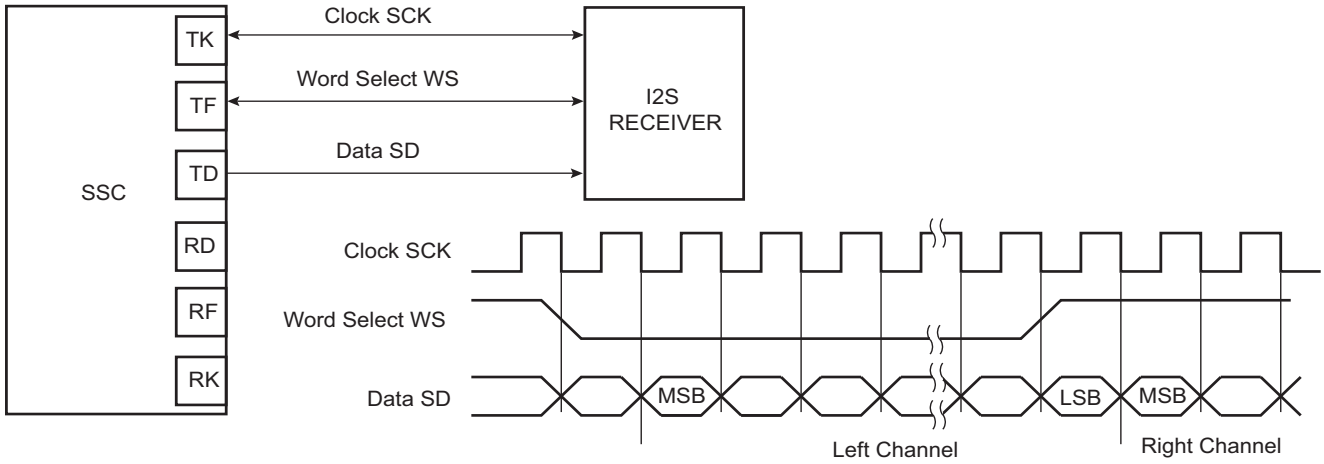


Figure 40-4. Codec Application Block Diagram

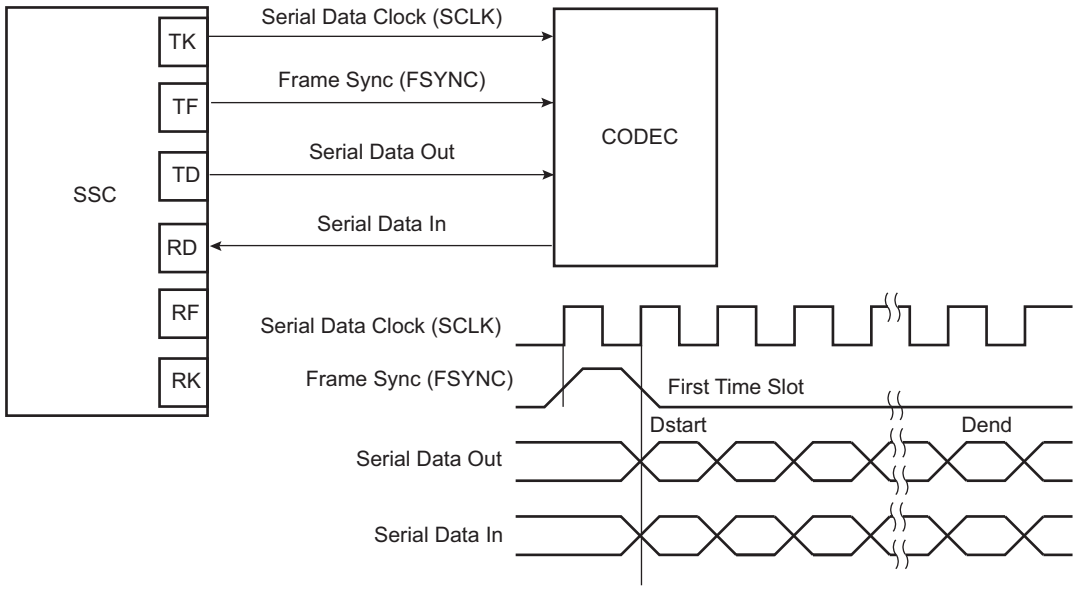
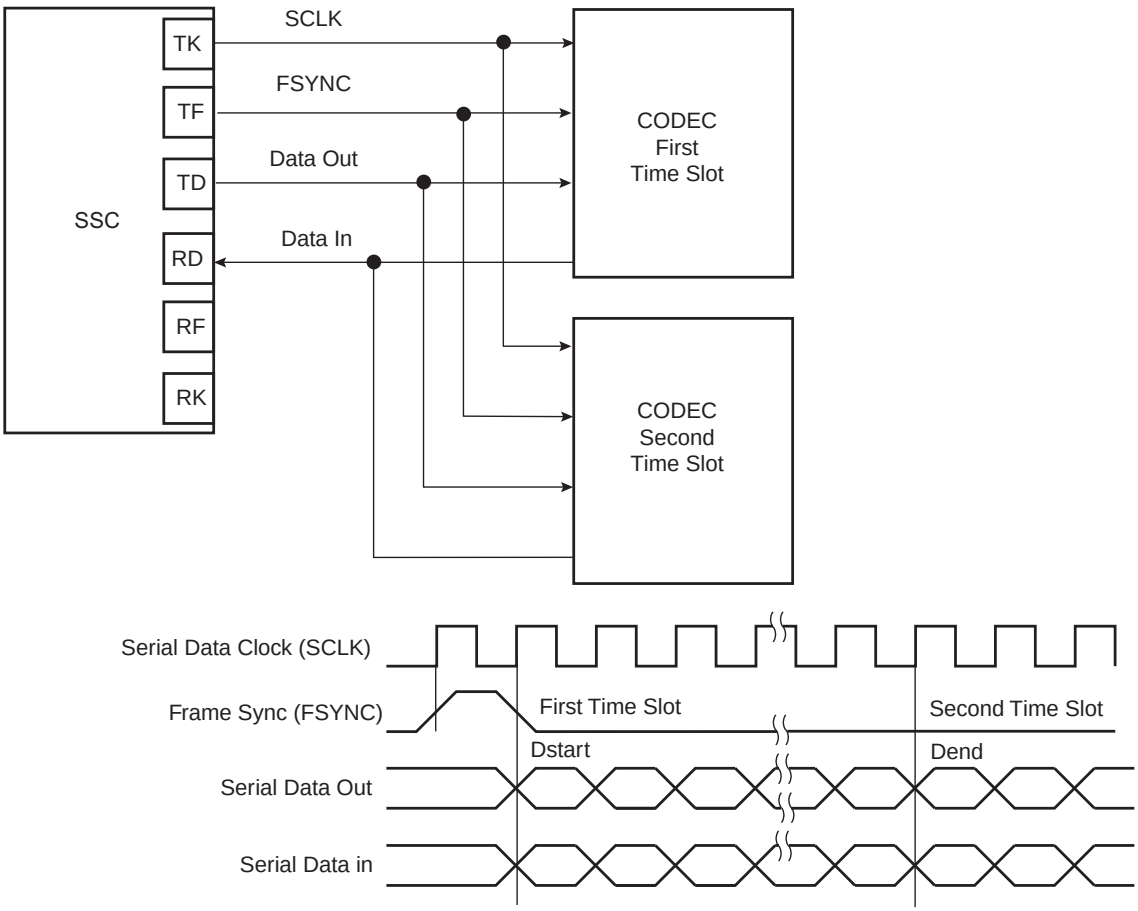


Figure 40-5. Time Slot Application Block Diagram



40.6 Pin Name List

Table 40-1. I/O Lines Description

Pin Name	Pin Description	Type
RF	Receive Frame Synchronization	Input/Output
RK	Receive Clock	Input/Output
RD	Receive Data	Input
TF	Transmit Frame Synchronization	Input/Output
TK	Transmit Clock	Input/Output
TD	Transmit Data	Output

40.7 Product Dependencies

40.7.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines.

Before using the SSC receiver, the PIO controller must be configured to dedicate the SSC receiver I/O lines to the SSC Peripheral mode.

Before using the SSC transmitter, the PIO controller must be configured to dedicate the SSC transmitter I/O lines to the SSC Peripheral mode.

Table 40-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
SSC0	RD0	PB29	B
SSC0	RF0	PB30	B
SSC0	RK0	PB26	B
SSC0	TD0	PA25	B
SSC0	TD0	PB28	B
SSC0	TF0	PB31	B
SSC0	TK0	PB27	B
SSC1	RD1	PC23	B
SSC1	RF1	PC22	B
SSC1	RK1	PC24	B
SSC1	TD1	PC21	B
SSC1	TF1	PC20	B
SSC1	TK1	PC19	B

40.7.2 Power Management

The SSC is not continuously clocked. The SSC interface may be clocked through the Power Management Controller (PMC), therefore the programmer must first configure the PMC to enable the SSC clock.

40.7.3 Interrupt

The SSC interface has an interrupt line connected to the interrupt controller. Handling interrupts requires programming the interrupt controller before configuring the SSC.

All SSC interrupts can be enabled/disabled configuring the SSC Interrupt Mask Register. Each pending and unmasked SSC interrupt asserts the SSC interrupt line. The SSC interrupt service routine can get the interrupt origin by reading the SSC Interrupt Status Register.

Table 40-3. Peripheral IDs

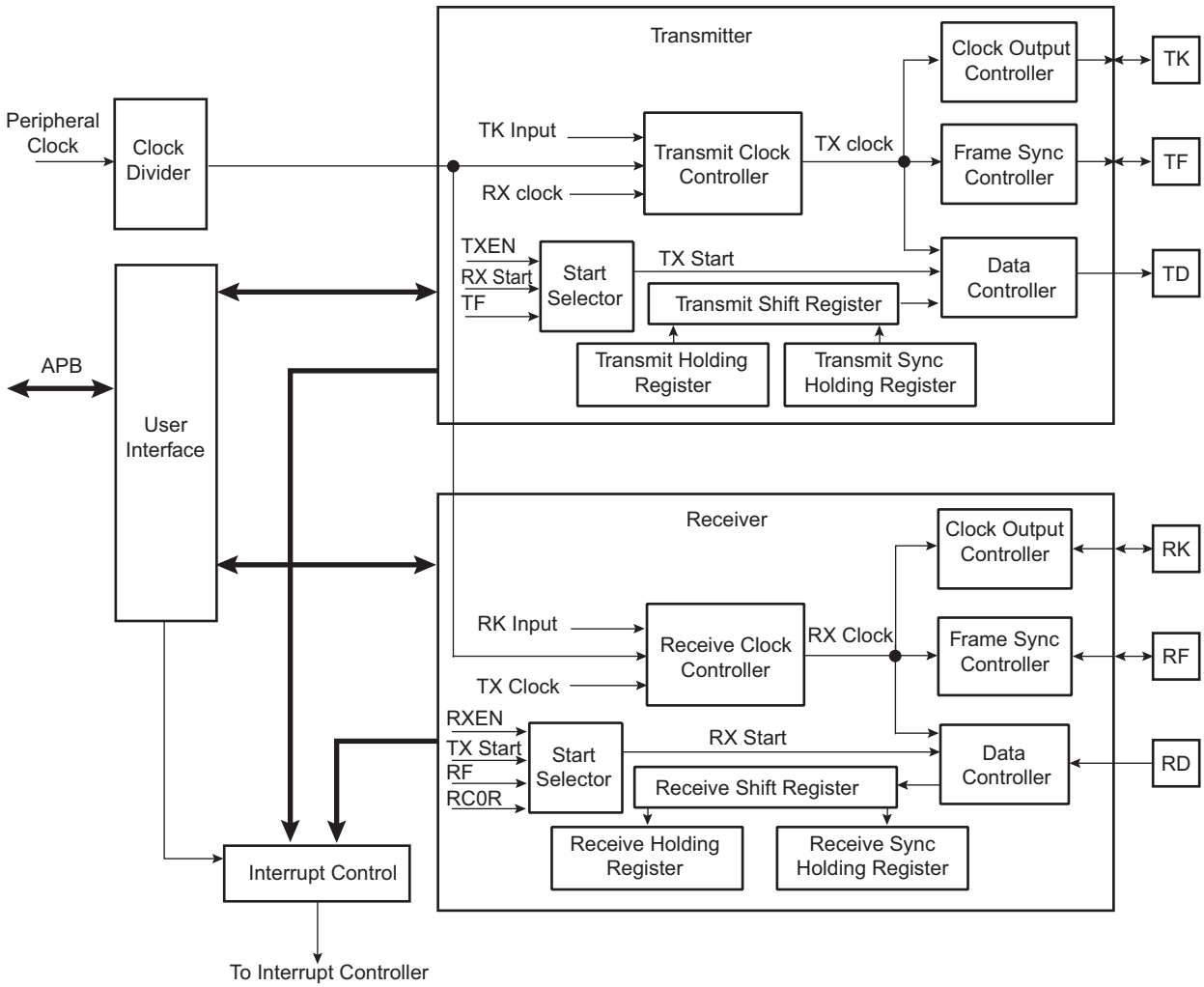
Instance	ID
SSC0	48
SSC1	49

40.8 Functional Description

This section contains the functional description of the following: SSC Functional Block, Clock Management, Data Format, Start, Transmit, Receive and Frame Synchronization.

The receiver and transmitter operate separately. However, they can work synchronously by programming the receiver to use the transmit clock and/or to start a data transfer when transmission starts. Alternatively, this can be done by programming the transmitter to use the receive clock and/or to start a data transfer when reception starts. The transmitter and the receiver can be programmed to operate with the clock signals provided on either the TK or RK pins. This allows the SSC to support many Slave mode data transfers. The maximum clock speed allowed on the TK and RK pins is the peripheral clock divided by 2.

Figure 40-6. SSC Functional Block Diagram



40.8.1 Clock Management

The transmit clock can be generated by:

- an external clock received on the TK I/O pad
- the receive clock
- the internal clock divider

The receive clock can be generated by:

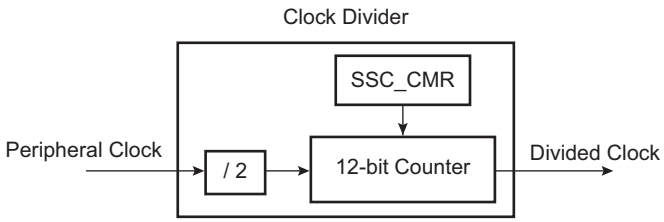
- an external clock received on the RK I/O pad
- the transmit clock
- the internal clock divider

Furthermore, the transmitter block can generate an external clock on the TK I/O pad, and the receive block can generate an external clock on the RK I/O pad.

This allows the SSC to support many Master and Slave mode data transfers.

40.8.1.1 Clock Divider

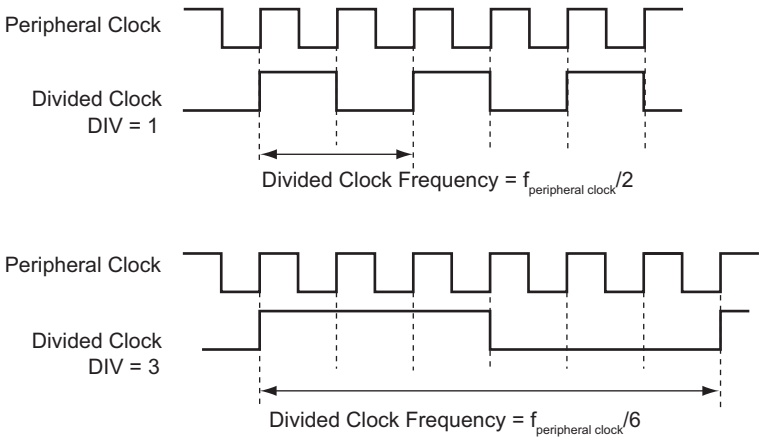
Figure 40-7. Divided Clock Block Diagram



The peripheral clock divider is determined by the 12-bit field DIV counter and comparator (so its maximal value is 4095) in the Clock Mode Register (SSC_CMR), allowing a peripheral clock division by up to 8190. The Divided Clock is provided to both the receiver and the transmitter. When this field is programmed to 0, the Clock Divider is not used and remains inactive.

When DIV is set to a value equal to or greater than 1, the Divided Clock has a frequency of peripheral clock divided by 2 times DIV. Each level of the Divided Clock has a duration of the peripheral clock multiplied by DIV. This ensures a 50% duty cycle for the Divided Clock regardless of whether the DIV value is even or odd.

Figure 40-8. Divided Clock Generation

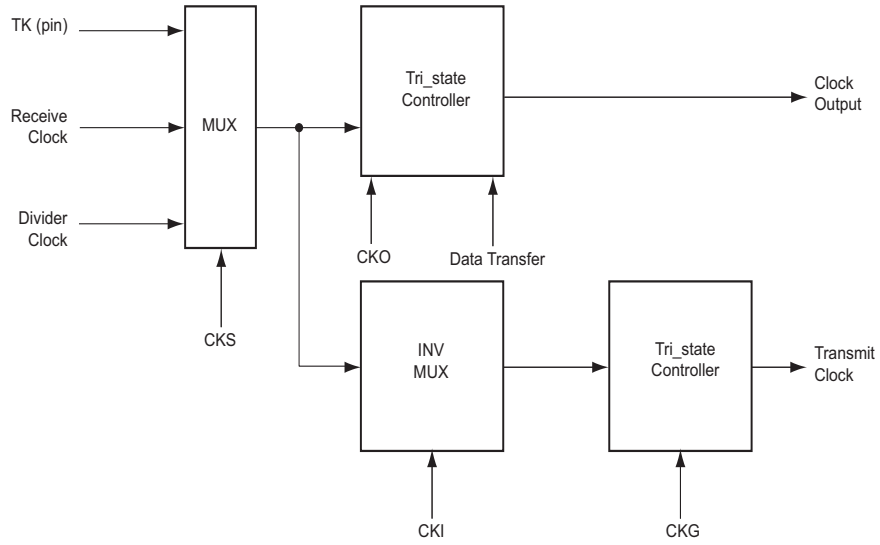


40.8.1.2 Transmit Clock Management

The transmit clock is generated from the receive clock or the divider clock or an external clock scanned on the TK I/O pad. The transmit clock is selected by the CKS field in the Transmit Clock Mode Register (SSC_TCMR). Transmit Clock can be inverted independently by the CKI bits in the SSC_TCMR.

The transmitter can also drive the TK I/O pad continuously or be limited to the current data transfer. The clock output is configured by the SSC_TCMR. The Transmit Clock Inversion (CKI) bits have no effect on the clock outputs. Programming the SSC_TCMR to select TK pin (CKS field) and at the same time Continuous Transmit Clock (CKO field) can lead to unpredictable results.

Figure 40-9. Transmit Clock Management

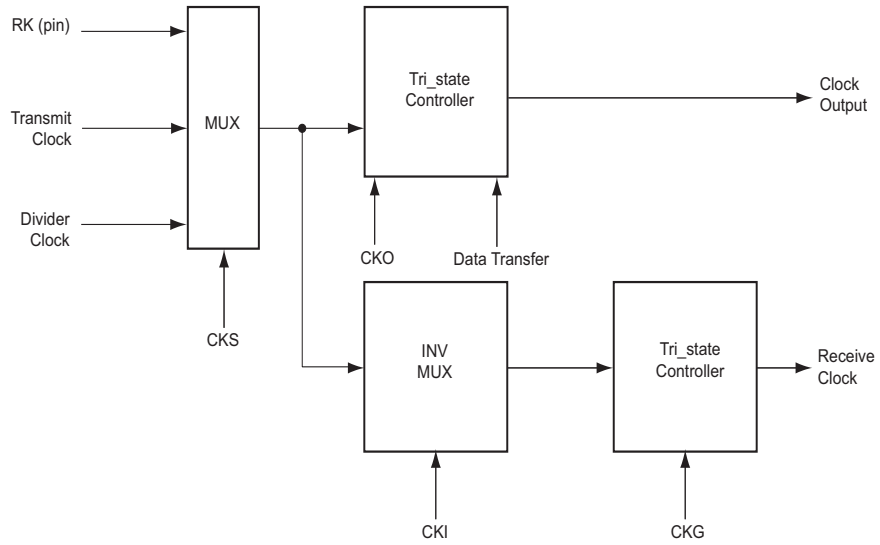


40.8.1.3 Receive Clock Management

The receive clock is generated from the transmit clock or the divider clock or an external clock scanned on the RK I/O pad. The Receive Clock is selected by the CKS field in SSC_RCMR (Receive Clock Mode Register). Receive Clocks can be inverted independently by the CKI bits in SSC_RCMR.

The receiver can also drive the RK I/O pad continuously or be limited to the current data transfer. The clock output is configured by the SSC_RCMR. The Receive Clock Inversion (CKI) bits have no effect on the clock outputs. Programming the SSC_RCMR to select RK pin (CKS field) and at the same time Continuous Receive Clock (CKO field) can lead to unpredictable results.

Figure 40-10. Receive Clock Management



40.8.1.4 Serial Clock Ratio Considerations

The transmitter and the receiver can be programmed to operate with the clock signals provided on either the TK or RK pins. This allows the SSC to support many Slave mode data transfers. In this case, the maximum clock speed allowed on the RK pin is:

- Peripheral clock divided by 2 if Receive Frame Synchronization is input

– Peripheral clock divided by 3 if Receive Frame Synchronization is output

In addition, the maximum clock speed allowed on the TK pin is:

– Peripheral clock divided by 6 if Transmit Frame Synchronization is input
– Peripheral clock divided by 2 if Transmit Frame Synchronization is output

40.8.2 Transmit Operations

A transmit frame is triggered by a start event and can be followed by synchronization data before data transmission.

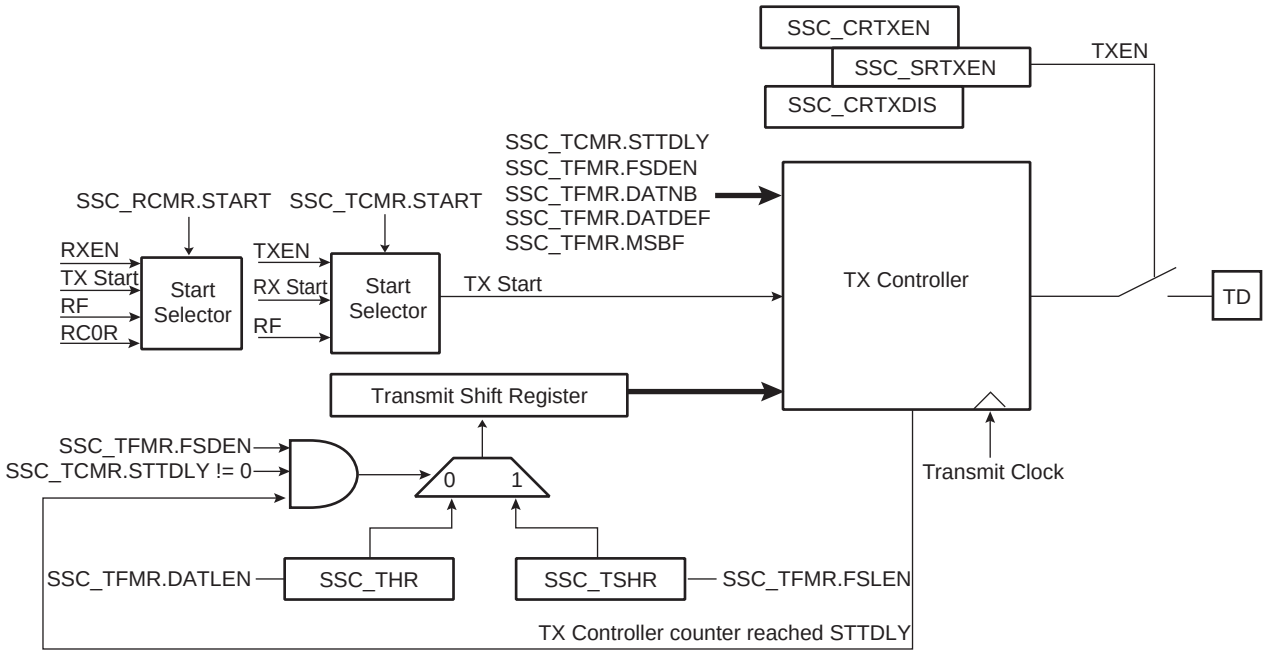
The start event is configured by setting the SSC_TCMR. Refer to Section 40.8.4 “Start”.

The frame synchronization is configured setting the Transmit Frame Mode Register (SSC_TFMR). Refer to Section 40.8.5 “Frame Synchronization”.

To transmit data, the transmitter uses a shift register clocked by the transmit clock signal and the start mode selected in the SSC_TCMR. Data is written by the application to the SSC_THR then transferred to the shift register according to the data format selected.

When both the SSC_THR and the transmit shift register are empty, the status flag TXEMPTY is set in the SSC_SR. When the Transmit Holding register is transferred in the transmit shift register, the status flag TXRDY is set in the SSC_SR and additional data can be loaded in the holding register.

Figure 40-11. Transmit Block Diagram



40.8.3 Receive Operations

A receive frame is triggered by a start event and can be followed by synchronization data before data transmission.

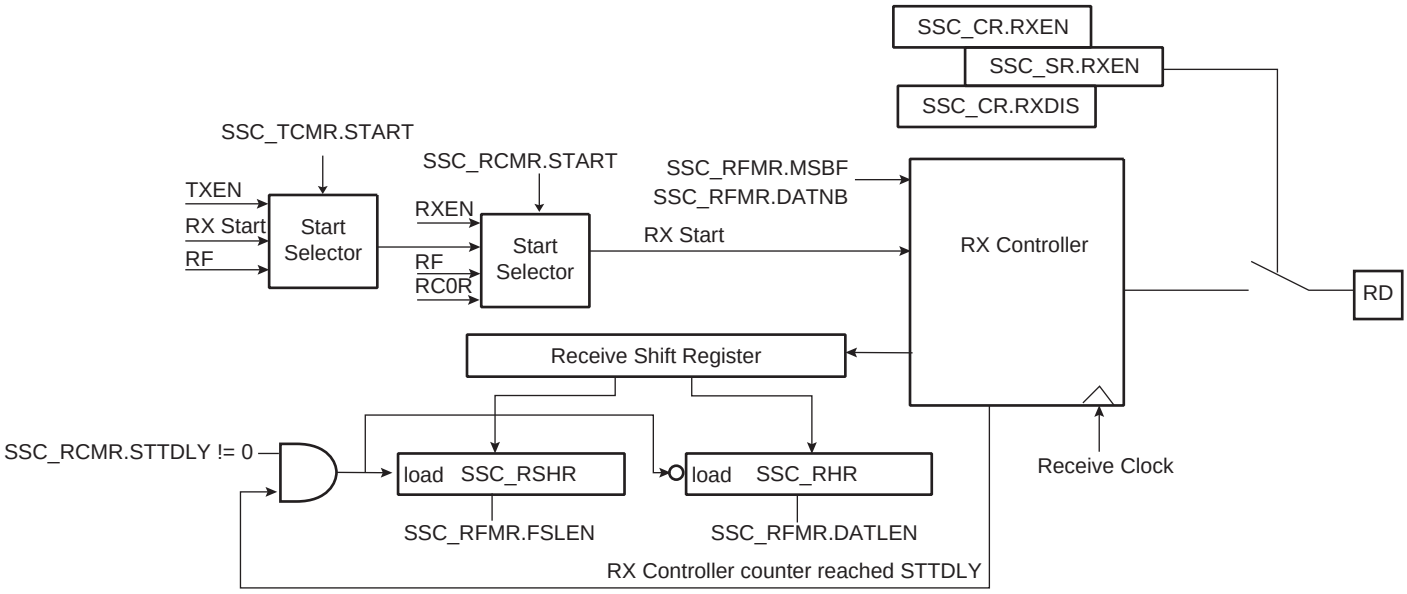
The start event is configured setting the Receive Clock Mode Register (SSC_RCMR). Refer to Section 40.8.4 “Start”.

The frame synchronization is configured by setting the Receive Frame Mode Register (SSC_RFMR). Refer to Section 40.8.5 “Frame Synchronization”.

The receiver uses a shift register clocked by the receive clock signal and the start mode selected in the SSC_RCMR. The data is transferred from the shift register depending on the data format selected.

When the receiver shift register is full, the SSC transfers this data in the holding register, the status flag RXRDY is set in the SSC_SR and the data can be read in the receiver holding register. If another transfer occurs before read of the Receive Holding Register (SSC_RHR), the status flag OVERUN is set in the SSC_SR and the receiver shift register is transferred in the SSC_RHR.

Figure 40-12. Receive Block Diagram



40.8.4 Start

The transmitter and receiver can both be programmed to start their operations when an event occurs, respectively in the Transmit Start Selection (START) field of SSC_TCMR and in the Receive Start Selection (START) field of SSC_RCMR.

Under the following conditions the start event is independently programmable:

- Continuous. In this case, the transmission starts as soon as a word is written in SSC_THR and the reception starts as soon as the receiver is enabled.
- Synchronously with the transmitter/receiver
- On detection of a falling/rising edge on TF/RF
- On detection of a low level/high level on TF/RF
- On detection of a level change or an edge on TF/RF

A start can be programmed in the same manner on either side of the Transmit/Receive Clock Register (SSC_RCMR/SSC_TCMR). Thus, the start could be on TF (Transmit) or RF (Receive).

Moreover, the receiver can start when data is detected in the bit stream with the Compare Functions.

Detection on TF/RF input/output is done by the field FSOS of the Transmit/Receive Frame Mode Register (SSC_TFMR/SSC_RFMR).

Figure 40-13. Transmit Start Mode

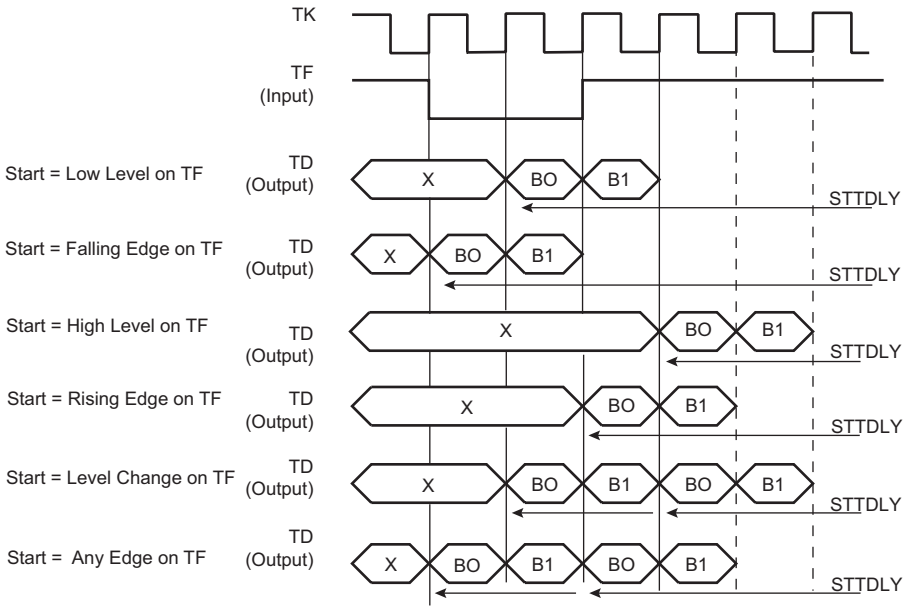
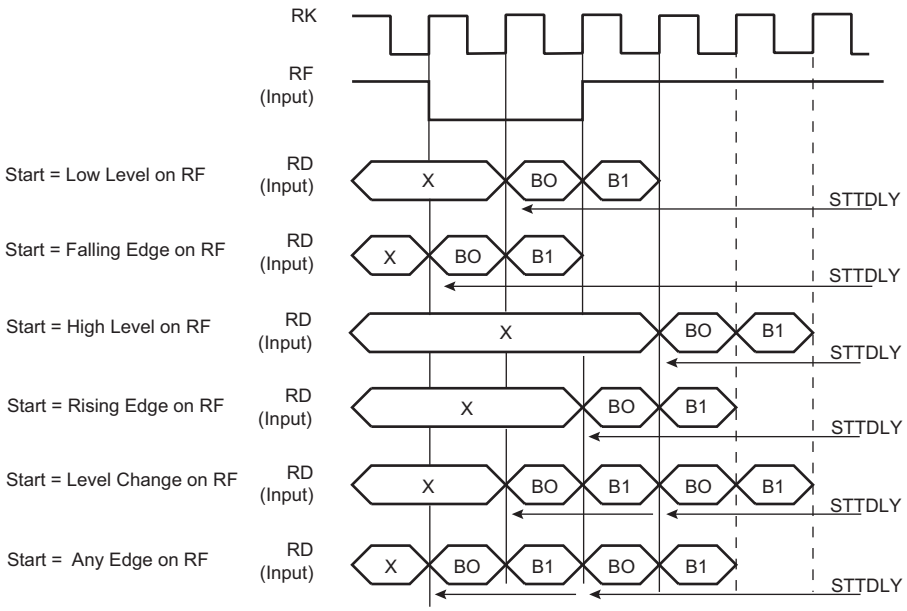


Figure 40-14. Receive Pulse/Edge Start Modes



40.8.5 Frame Synchronization

The Transmit and Receive Frame Sync pins, TF and RF, can be programmed to generate different kinds of Frame Sync signals. The Frame Sync Output Selection (FSOS) field in the Receive Frame Mode Register (SSC_RFMR) and in the Transmit Frame Mode Register (SSC_TFMR) are used to select the required waveform.

- Programmable low or high levels during data transfer are supported.
- Programmable high levels before the start of data transfers or toggling are also supported.

If a pulse waveform is selected, the Frame Sync Length (FSLEN) field in SSC_RFMR and SSC_TFMR programs the length of the pulse, from 1 bit time up to 256 bit times.

The periodicity of the Receive and Transmit Frame Sync pulse output can be programmed through the Period Divider Selection (PERIOD) field in SSC_RCMR and SSC_TCMR.

40.8.5.1 Frame Sync Data

Frame Sync Data transmits or receives a specific tag during the Frame Sync signal.

During the Frame Sync signal, the receiver can sample the RD line and store the data in the Receive Sync Holding Register and the transmitter can transfer Transmit Sync Holding Register in the shift register. The data length to be sampled/shifted out during the Frame Sync signal is programmed by the FSLEN field in SSC_RFMR/SSC_TFMR and has a maximum value of 256.

Concerning the Receive Frame Sync Data operation, if the Frame Sync Length is equal to or lower than the delay between the start event and the current data reception, the data sampling operation is performed in the Receive Sync Holding Register through the receive shift register.

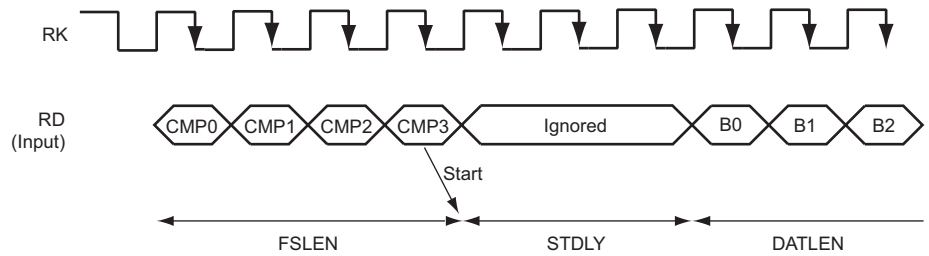
The Transmit Frame Sync Operation is performed by the transmitter only if the bit Frame Sync Data Enable (FSDEN) in SSC_TFMR is set. If the Frame Sync length is equal to or lower than the delay between the start event and the current data transmission, the normal transmission has priority and the data contained in the Transmit Sync Holding Register is transferred in the Transmit Register, then shifted out.

40.8.5.2 Frame Sync Edge Detection

The Frame Sync Edge detection is programmed by the FSEDGE field in SSC_RFMR/SSC_TFMR. This sets the corresponding flags RXSYN/TXSYN in the SSC Status Register (SSC_SR) on Frame Sync Edge detection (signals RF/TF).

40.8.6 Receive Compare Modes

Figure 40-15. Receive Compare Modes



40.8.6.1 Compare Functions

The length of the comparison patterns (Compare 0, Compare 1) and thus the number of bits they are compared to is defined by FSLEN, but with a maximum value of 256 bits. Comparison is always done by comparing the last bits received with the comparison pattern. Compare 0 can be one start event of the receiver. In this case, the receiver compares at each new sample the last bits received at the Compare 0 pattern contained in the Compare 0 Register (SSC_RC0R). When this start event is selected, the user can program the receiver to start a new data transfer either by writing a new Compare 0, or by receiving continuously until Compare 1 occurs. This selection is done with the STOP bit in the SSC_RCMR.

40.8.7 Data Format

The data framing format of both the transmitter and the receiver are programmable through the Transmitter Frame Mode Register (SSC_TFMR) and the Receive Frame Mode Register (SSC_RFMR). In either case, the user can independently select the following parameters:

- Event that starts the data transfer (START)
- Delay in number of bit periods between the start event and the first data bit (STTDLY)
- Length of the data (DATLEN)

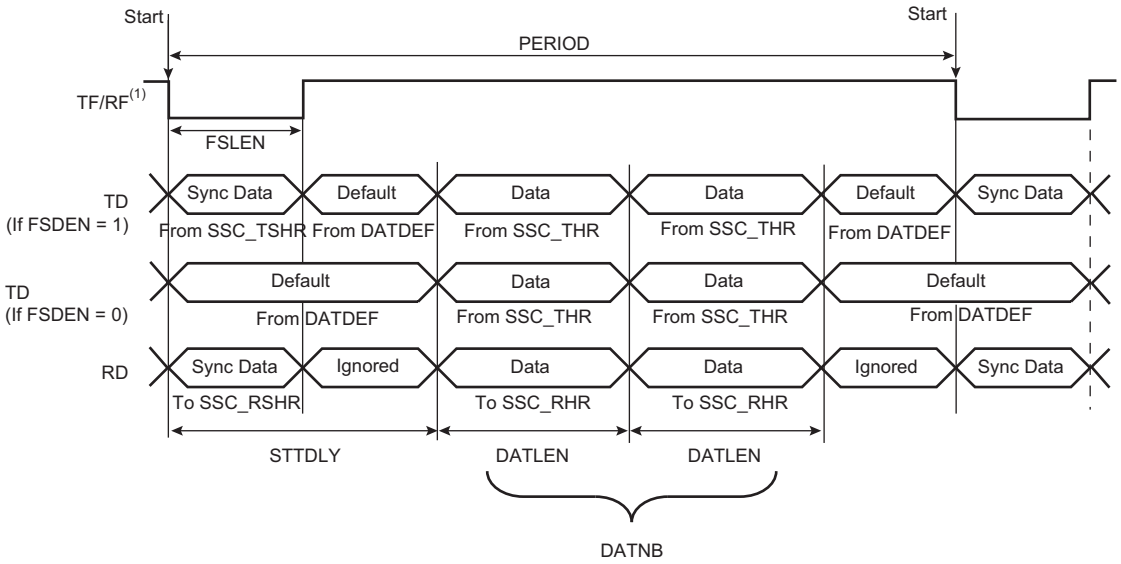
- Number of data to be transferred for each start event (DATNB)
- Length of synchronization transferred for each start event (FSLEN)
- Bit sense: most or least significant bit first (MSBF)

Additionally, the transmitter can be used to transfer synchronization and select the level driven on the TD pin while not in data transfer operation. This is done respectively by the Frame Sync Data Enable (FSDEN) and by the Data Default Value (DATDEF) bits in SSC_TFMR.

Table 40-4. Data Frame Registers

Transmitter	Receiver	Field	Length	Comment
SSC_TFMR	SSC_RFMR	DATLEN	Up to 32	Size of word
SSC_TFMR	SSC_RFMR	DATNB	Up to 16	Number of words transmitted in frame
SSC_TFMR	SSC_RFMR	MSBF	–	Most significant bit first
SSC_TFMR	SSC_RFMR	FSLEN	Up to 256	Size of Synchro data register
SSC_TFMR	–	DATDEF	0 or 1	Data default value ended
SSC_TFMR	–	FSDEN	–	Enable send SSC_TSHR
SSC_TCMR	SSC_RCMR	PERIOD	Up to 512	Frame size
SSC_TCMR	SSC_RCMR	STTDLY	Up to 255	Size of transmit start delay

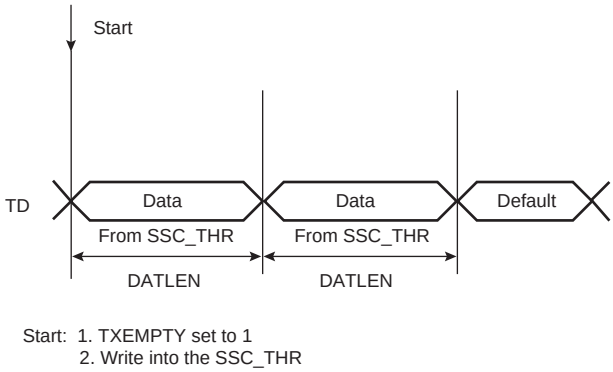
Figure 40-16. Transmit and Receive Frame Format in Edge/Pulse Start Modes



Note: 1. Example of input on falling edge of TF/RF.

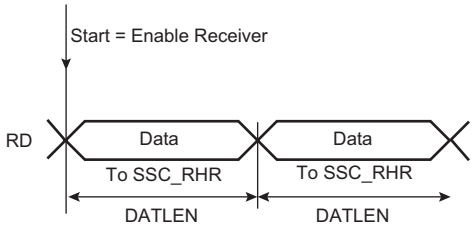
In the example illustrated in Figure 40-17, the SSC_THR is loaded twice. The FSDEN value has no effect on the transmission. SyncData cannot be output in Continuous mode.

Figure 40-17. Transmit Frame Format in Continuous Mode (STTDLY = 0)



Start: 1. TXEMPTY set to 1
2. Write into the SSC_THR

Figure 40-18. Receive Frame Format in Continuous Mode (STTDLY = 0)



40.8.8 Loop Mode

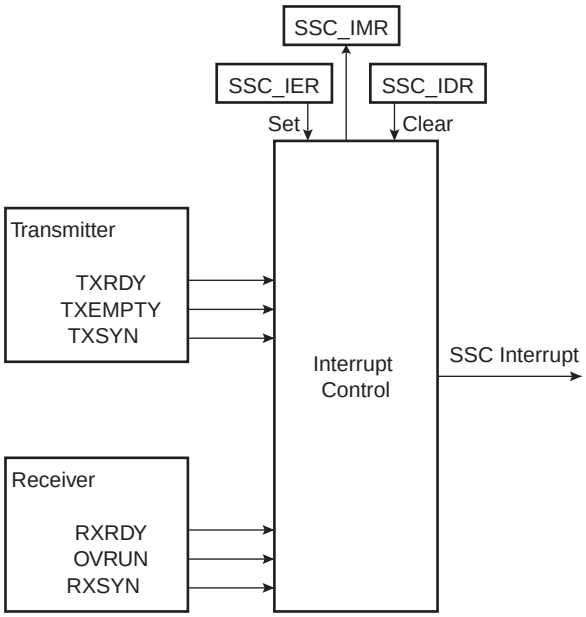
The receiver can be programmed to receive transmissions from the transmitter. This is done by setting the Loop Mode (LOOP) bit in the SSC_RFMR. In this case, RD is connected to TD, RF is connected to TF and RK is connected to TK.

40.8.9 Interrupt

Most bits in the SSC_SR have a corresponding bit in interrupt management registers.

The SSC can be programmed to generate an interrupt when it detects an event. The interrupt is controlled by writing the Interrupt Enable Register (SSC_IER) and Interrupt Disable Register (SSC_IDR). These registers enable and disable, respectively, the corresponding interrupt by setting and clearing the corresponding bit in the Interrupt Mask Register (SSC_IMR), which controls the generation of interrupts by asserting the SSC interrupt line connected to the interrupt controller.

Figure 40-19. Interrupt Block Diagram



40.8.10 Register Write Protection

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

If a write access to a write-protected register is detected, the WPVS flag in the SSC Write Protection Status Register (SSC_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 SSC_WPSR.

The following registers can be write-protected:

- SSC Clock Mode Register
- SSC Receive Clock Mode Register
- SSC Receive Frame Mode Register
- SSC Transmit Clock Mode Register
- SSC Transmit Frame Mode Register
- SSC Receive Compare 0 Register
- SSC Receive Compare 1 Register

40.9 Synchronous Serial Controller (SSC) User Interface

Table 40-5. Register Mapping

Offset	Register	Name	Access	Reset
0x0	Control Register	SSC_CR	Write-only	–
0x4	Clock Mode Register	SSC_CMR	Read/Write	0x0
0x8–0xC	Reserved	–	–	–
0x10	Receive Clock Mode Register	SSC_RCMR	Read/Write	0x0
0x14	Receive Frame Mode Register	SSC_RFMR	Read/Write	0x0
0x18	Transmit Clock Mode Register	SSC_TCMR	Read/Write	0x0
0x1C	Transmit Frame Mode Register	SSC_TFMR	Read/Write	0x0
0x20	Receive Holding Register	SSC_RHR	Read-only	0x0
0x24	Transmit Holding Register	SSC_THR	Write-only	–
0x28–0x2C	Reserved	–	–	–
0x30	Receive Sync. Holding Register	SSC_RSHR	Read-only	0x0
0x34	Transmit Sync. Holding Register	SSC_TSHR	Read/Write	0x0
0x38	Receive Compare 0 Register	SSC_RC0R	Read/Write	0x0
0x3C	Receive Compare 1 Register	SSC_RC1R	Read/Write	0x0
0x40	Status Register	SSC_SR	Read-only	0x000000CC
0x44	Interrupt Enable Register	SSC_IER	Write-only	–
0x48	Interrupt Disable Register	SSC_IDR	Write-only	–
0x4C	Interrupt Mask Register	SSC_IMR	Read-only	0x0
0x50–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	SSC_WPMR	Read/Write	0x0
0xE8	Write Protection Status Register	SSC_WPSR	Read-only	0x0
0xEC–0xFC	Reserved	–	–	–
0x100–0x124	Reserved	–	–	–

40.9.1 SSC Control Register

Name: SSC_CR

Address: 0xF8008000 (0), 0xFC014000 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
SWRST	–	–	–	–	–	TXDIS	TXEN
7	6	5	4	3	2	1	0
–	–	–	–	–	–	RXDIS	RXEN

- **RXEN: Receive Enable**

0: No effect.

1: Enables Receive if RXDIS is not set.

- **RXDIS: Receive Disable**

0: No effect.

1: Disables Receive. If a character is currently being received, disables at end of current character reception.

- **TXEN: Transmit Enable**

0: No effect.

1: Enables Transmit if TXDIS is not set.

- **TXDIS: Transmit Disable**

0: No effect.

1: Disables Transmit. If a character is currently being transmitted, disables at end of current character transmission.

- **SWRST: Software Reset**

0: No effect.

1: Performs a software reset. Has priority on any other bit in SSC_CR.

40.9.2 SSC Clock Mode Register

Name: SSC_CMR

Address: 0xF8008004 (0), 0xFC014004 (1)

Access: Read/Write

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

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

- **DIV: Clock Divider**

0: The Clock Divider is not active.

Any other value: The divided clock equals the peripheral clock divided by 2 times DIV.
The maximum bit rate is $f_{peripheral\ clock}/2$. The minimum bit rate is $f_{peripheral\ clock}/2 \times 4095 = f_{peripheral\ clock}/8190$.

40.9.3 SSC Receive Clock Mode Register

Name: SSC_RCMR

Address: 0xF8008010 (0), 0xFC014010 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
PERIOD							

23	22	21	20	19	18	17	16
STTDLY							

15	14	13	12	11	10	9	8
–	–	–	STOP	START			

7	6	5	4	3	2	1	0
CKG		CKI	CKO			CKS	

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

- **CKS: Receive Clock Selection**

Value	Name	Description
0	MCK	Divided Clock
1	TK	TK Clock signal
2	RK	RK pin

- **CKO: Receive Clock Output Mode Selection**

Value	Name	Description
0	NONE	None, RK pin is an input
1	CONTINUOUS	Continuous Receive Clock, RK pin is an output
2	TRANSFER	Receive Clock only during data transfers, RK pin is an output

- **CKI: Receive Clock Inversion**

0: The data inputs (Data and Frame Sync signals) are sampled on Receive Clock falling edge. The Frame Sync signal output is shifted out on Receive Clock rising edge.

1: The data inputs (Data and Frame Sync signals) are sampled on Receive Clock rising edge. The Frame Sync signal output is shifted out on Receive Clock falling edge.

CKI affects only the Receive Clock and not the output clock signal.

- **CKG: Receive Clock Gating Selection**

Value	Name	Description
0	CONTINUOUS	None
1	EN_RF_LOW	Receive Clock enabled only if RF Low
2	EN_RF_HIGH	Receive Clock enabled only if RF High

- **START: Receive Start Selection**

Value	Name	Description
0	CONTINUOUS	Continuous, as soon as the receiver is enabled, and immediately after the end of transfer of the previous data.
1	TRANSMIT	Transmit start
2	RF_LOW	Detection of a low level on RF signal
3	RF_HIGH	Detection of a high level on RF signal
4	RF_FALLING	Detection of a falling edge on RF signal
5	RF_RISING	Detection of a rising edge on RF signal
6	RF_LEVEL	Detection of any level change on RF signal
7	RF_EDGE	Detection of any edge on RF signal
8	CMP_0	Compare 0

- **STOP: Receive Stop Selection**

0: After completion of a data transfer when starting with a Compare 0, the receiver stops the data transfer and waits for a new compare 0.

1: After starting a receive with a Compare 0, the receiver operates in a continuous mode until a Compare 1 is detected.

- **STTDLY: Receive Start Delay**

If STTDLY is not 0, a delay of STTDLY clock cycles is inserted between the start event and the current start of reception. When the receiver is programmed to start synchronously with the transmitter, the delay is also applied.

Note: It is very important that STTDLY be set carefully. If STTDLY must be set, it should be done in relation to TAG (Receive Sync Data) reception.

- **PERIOD: Receive Period Divider Selection**

This field selects the divider to apply to the selected Receive Clock in order to generate a new Frame Sync signal. If 0, no PERIOD signal is generated. If not 0, a PERIOD signal is generated each 2 x (PERIOD + 1) Receive Clock.

40.9.4 SSC Receive Frame Mode Register

Name: SSC_RFMR

Address: 0xF8008014 (0), 0xFC014014 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
FSLEN_EXT				–	–	–	FSEDGE

23	22	21	20	19	18	17	16
–	FSOS			FSLEN			

15	14	13	12	11	10	9	8
–	–	–	–	DATNB			

7	6	5	4	3	2	1	0
MSBF	–	LOOP	DATLEN				

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

- **DATLEN: Data Length**

0: Forbidden value (1-bit data length not supported).

Any other value: The bit stream contains DATLEN + 1 data bits.

- **LOOP: Loop Mode**

0: Normal operating mode.

1: RD is driven by TD, RF is driven by TF and TK drives RK.

- **MSBF: Most Significant Bit First**

0: The lowest significant bit of the data register is sampled first in the bit stream.

1: The most significant bit of the data register is sampled first in the bit stream.

- **DATNB: Data Number per Frame**

This field defines the number of data words to be received after each transfer start, which is equal to (DATNB + 1).

- **FSLEN: Receive Frame Sync Length**

This field defines the number of bits sampled and stored in the Receive Sync Data Register. When this mode is selected by the START field in the Receive Clock Mode Register, it also determines the length of the sampled data to be compared to the Compare 0 or Compare 1 register.

This field is used with FSLEN_EXT to determine the pulse length of the Receive Frame Sync signal.

Pulse length is equal to FSLEN + (FSLEN_EXT × 16) + 1 Receive Clock periods.

- **FSOS: Receive Frame Sync Output Selection**

Value	Name	Description
0	NONE	None, RF pin is an input
1	NEGATIVE	Negative Pulse, RF pin is an output
2	POSITIVE	Positive Pulse, RF pin is an output

Value	Name	Description
3	LOW	Driven Low during data transfer, RF pin is an output
4	HIGH	Driven High during data transfer, RF pin is an output
5	TOGGLING	Toggling at each start of data transfer, RF pin is an output

- **FSEDGE: Frame Sync Edge Detection**

Determines which edge on Frame Sync will generate the interrupt RXSYN in the SSC Status Register.

Value	Name	Description
0	POSITIVE	Positive Edge Detection
1	NEGATIVE	Negative Edge Detection

- **FSLEN_EXT: FSLEN Field Extension**

Extends FSLEN field. For details, refer to ”FSLEN: Receive Frame Sync Length”.

40.9.5 SSC Transmit Clock Mode Register

Name: SSC_TCMR

Address: 0xF8008018 (0), 0xFC014018 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
PERIOD							
23	22	21	20	19	18	17	16
STTDLY							
15	14	13	12	11	10	9	8
–	–	–	–	START			
7	6	5	4	3	2	1	0
CKG		CKI	CKO			CKS	

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

- **CKS: Transmit Clock Selection**

Value	Name	Description
0	MCK	Divided Clock
1	RK	RK Clock signal
2	TK	TK pin

- **CKO: Transmit Clock Output Mode Selection**

Value	Name	Description
0	NONE	None, TK pin is an input
1	CONTINUOUS	Continuous Transmit Clock, TK pin is an output
2	TRANSFER	Transmit Clock only during data transfers, TK pin is an output

- **CKI: Transmit Clock Inversion**

0: The data outputs (Data and Frame Sync signals) are shifted out on Transmit Clock falling edge. The Frame Sync signal input is sampled on Transmit Clock rising edge.

1: The data outputs (Data and Frame Sync signals) are shifted out on Transmit Clock rising edge. The Frame Sync signal input is sampled on Transmit Clock falling edge.

CKI affects only the Transmit Clock and not the Output Clock signal.

- **CKG: Transmit Clock Gating Selection**

Value	Name	Description
0	CONTINUOUS	None
1	EN_TF_LOW	Transmit Clock enabled only if TF Low
2	EN_TF_HIGH	Transmit Clock enabled only if TF High

- **START: Transmit Start Selection**

Value	Name	Description
0	CONTINUOUS	Continuous, as soon as a word is written in the SSC_THR (if Transmit is enabled), and immediately after the end of transfer of the previous data
1	RECEIVE	Receive start
2	TF_LOW	Detection of a low level on TF signal
3	TF_HIGH	Detection of a high level on TF signal
4	TF_FALLING	Detection of a falling edge on TF signal
5	TF_RISING	Detection of a rising edge on TF signal
6	TF_LEVEL	Detection of any level change on TF signal
7	TF_EDGE	Detection of any edge on TF signal

- **STTDLY: Transmit Start Delay**

If STTDLY is not 0, a delay of STTDLY clock cycles is inserted between the start event and the current start of transmission of data. When the transmitter is programmed to start synchronously with the receiver, the delay is also applied.

Note: STTDLY must be set carefully. If STTDLY is too short in respect to TAG (Transmit Sync Data) transmission, data is transmitted instead of the end of TAG.

- **PERIOD: Transmit Period Divider Selection**

This field selects the divider to apply to the selected Transmit Clock to generate a new Frame Sync signal. If 0, no period signal is generated. If not 0, a period signal is generated at each 2 × (PERIOD + 1) Transmit Clock.

40.9.6 SSC Transmit Frame Mode Register

Name: SSC_TFMR

Address: 0xF800801C (0), 0xFC01401C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
FSLEN_EXT				–	–	–	FSEDGE
23	22	21	20	19	18	17	16
FSDEN	FSOS			FSLEN			
15	14	13	12	11	10	9	8
–	–	–	–	DATNB			
7	6	5	4	3	2	1	0
MSBF	–	DATDEF	DATLEN				

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

- **DATLEN: Data Length**

0: Forbidden value (1-bit data length not supported).

Any other value: The bit stream contains DATLEN + 1 data bits.

- **DATDEF: Data Default Value**

This bit defines the level driven on the TD pin while out of transmission. Note that if the pin is defined as multi-drive by the PIO Controller, the pin is enabled only if the SCC TD output is 1.

- **MSBF: Most Significant Bit First**

0: The lowest significant bit of the data register is shifted out first in the bit stream.

1: The most significant bit of the data register is shifted out first in the bit stream.

- **DATNB: Data Number per Frame**

This field defines the number of data words to be transferred after each transfer start, which is equal to (DATNB + 1).

- **FSLEN: Transmit Frame Sync Length**

This field defines the length of the Transmit Frame Sync signal and the number of bits shifted out from the Transmit Sync Data Register if FSDEN is 1.

This field is used with FSLEN_EXT to determine the pulse length of the Transmit Frame Sync signal.

Pulse length is equal to FSLEN + (FSLEN_EXT × 16) + 1 Transmit Clock period.

- **FSOS: Transmit Frame Sync Output Selection**

Value	Name	Description
0	NONE	None, TF pin is an input
1	NEGATIVE	Negative Pulse, TF pin is an output
2	POSITIVE	Positive Pulse, TF pin is an output

Value	Name	Description
3	LOW	Driven Low during data transfer
4	HIGH	Driven High during data transfer
5	TOGGLING	Toggling at each start of data transfer

- **FSDEN: Frame Sync Data Enable**

0: The TD line is driven with the default value during the Transmit Frame Sync signal.

1: SSC_TSHR value is shifted out during the transmission of the Transmit Frame Sync signal.

- **FSEDGE: Frame Sync Edge Detection**

Determines which edge on frame synchronization will generate the interrupt TXSYN (Status Register).

Value	Name	Description
0	POSITIVE	Positive Edge Detection
1	NEGATIVE	Negative Edge Detection

- **FSLEN_EXT: FSLEN Field Extension**

Extends FSLEN field. For details, refer to FSLEN bit description above.

40.9.7 SSC Receive Holding Register

Name: SSC_RHR

Address: 0xF8008020 (0), 0xFC014020 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RDAT							

23	22	21	20	19	18	17	16
RDAT							

15	14	13	12	11	10	9	8
RDAT							

7	6	5	4	3	2	1	0
RDAT							

- **RDAT: Receive Data**

Right aligned regardless of the number of data bits defined by DATLEN in SSC_RFMR.

40.9.8 SSC Transmit Holding Register

Name: SSC_THR

Address: 0xF8008024 (0), 0xFC014024 (1)

Access: Write-only

31	30	29	28	27	26	25	24
TDAT							

23	22	21	20	19	18	17	16
TDAT							

15	14	13	12	11	10	9	8
TDAT							

7	6	5	4	3	2	1	0
TDAT							

- **TDAT: Transmit Data**

Right aligned regardless of the number of data bits defined by DATLEN in SSC_TFMR.

40.9.9 SSC Receive Synchronization Holding Register

Name: SSC_RSHR

Address: 0xF8008030 (0), 0xFC014030 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RSDAT							

7	6	5	4	3	2	1	0
RSDAT							

- **RSDAT: Receive Synchronization Data**

40.9.10 SSC Transmit Synchronization Holding Register

Name: SSC_TSHR

Address: 0xF8008034 (0), 0xFC014034 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TSDAT							

7	6	5	4	3	2	1	0
TSDAT							

- **TSDAT: Transmit Synchronization Data**

40.9.11 SSC Receive Compare 0 Register

Name: SSC_RC0R

Address: 0xF8008038 (0), 0xFC014038 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CP0							
7	6	5	4	3	2	1	0
CP0							

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

- **CP0: Receive Compare Data 0**

40.9.12 SSC Receive Compare 1 Register

Name: SSC_RC1R

Address: 0xF800803C (0), 0xFC01403C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CP1							
7	6	5	4	3	2	1	0
CP1							

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

- **CP1: Receive Compare Data 1**

40.9.13 SSC Status Register

Name: SSC_SR

Address: 0xF8008040 (0), 0xFC014040 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	RXEN	TXEN

15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0

7	6	5	4	3	2	1	0
–	–	OVRUN	RXRDY	–	–	TXEMPTY	TXRDY

- **TXRDY: Transmit Ready**

0: Data has been loaded in SSC_THR and is waiting to be loaded in the transmit shift register (TSR).

1: SSC_THR is empty.

- **TXEMPTY: Transmit Empty**

0: Data remains in SSC_THR or is currently transmitted from TSR.

1: Last data written in SSC_THR has been loaded in TSR and last data loaded in TSR has been transmitted.

- **RXRDY: Receive Ready**

0: SSC_RHR is empty.

1: Data has been received and loaded in SSC_RHR.

- **OVRUN: Receive Overrun**

0: No data has been loaded in SSC_RHR while previous data has not been read since the last read of the Status Register.

1: Data has been loaded in SSC_RHR while previous data has not yet been read since the last read of the Status Register.

- **CP0: Compare 0**

0: A compare 0 has not occurred since the last read of the Status Register.

1: A compare 0 has occurred since the last read of the Status Register.

- **CP1: Compare 1**

0: A compare 1 has not occurred since the last read of the Status Register.

1: A compare 1 has occurred since the last read of the Status Register.

- **TXSYN: Transmit Sync**

0: A Tx Sync has not occurred since the last read of the Status Register.

1: A Tx Sync has occurred since the last read of the Status Register.

- **RXSYN: Receive Sync**

0: An Rx Sync has not occurred since the last read of the Status Register.

1: An Rx Sync has occurred since the last read of the Status Register.

- **TXEN: Transmit Enable**

0: Transmit is disabled.

1: Transmit is enabled.

- **RXEN: Receive Enable**

0: Receive is disabled.

1: Receive is enabled.

40.9.14 SSC Interrupt Enable Register

Name: SSC_IER

Address: 0xF8008044 (0), 0xFC014044 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	6	5	4	3	2	1	0
–	–	OVRUN	RXRDY	–	–	TXEMPTY	TXRDY

- **TXRDY: Transmit Ready Interrupt Enable**

0: No effect.

1: Enables the Transmit Ready Interrupt.

- **TXEMPTY: Transmit Empty Interrupt Enable**

0: No effect.

1: Enables the Transmit Empty Interrupt.

- **RXRDY: Receive Ready Interrupt Enable**

0: No effect.

1: Enables the Receive Ready Interrupt.

- **OVRUN: Receive Overrun Interrupt Enable**

0: No effect.

1: Enables the Receive Overrun Interrupt.

- **CP0: Compare 0 Interrupt Enable**

0: No effect.

1: Enables the Compare 0 Interrupt.

- **CP1: Compare 1 Interrupt Enable**

0: No effect.

1: Enables the Compare 1 Interrupt.

- **TXSYN: Tx Sync Interrupt Enable**

0: No effect.

1: Enables the Tx Sync Interrupt.

- **RXSYN: Rx Sync Interrupt Enable**

0: No effect.

1: Enables the Rx Sync Interrupt.

40.9.15 SSC Interrupt Disable Register

Name: SSC_IDR

Address: 0xF8008048 (0), 0xFC014048 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	6	5	4	3	2	1	0
–	–	OVRUN	RXRDY	–	–	TXEMPTY	TXRDY

- **TXRDY: Transmit Ready Interrupt Disable**

0: No effect.

1: Disables the Transmit Ready Interrupt.

- **TXEMPTY: Transmit Empty Interrupt Disable**

0: No effect.

1: Disables the Transmit Empty Interrupt.

- **RXRDY: Receive Ready Interrupt Disable**

0: No effect.

1: Disables the Receive Ready Interrupt.

- **OVRUN: Receive Overrun Interrupt Disable**

0: No effect.

1: Disables the Receive Overrun Interrupt.

- **CP0: Compare 0 Interrupt Disable**

0: No effect.

1: Disables the Compare 0 Interrupt.

- **CP1: Compare 1 Interrupt Disable**

0: No effect.

1: Disables the Compare 1 Interrupt.

- **TXSYN: Tx Sync Interrupt Enable**

0: No effect.

1: Disables the Tx Sync Interrupt.

- **RXSYN: Rx Sync Interrupt Enable**

0: No effect.

1: Disables the Rx Sync Interrupt.

40.9.16 SSC Interrupt Mask Register

Name: SSC_IMR

Address: 0xF800804C (0), 0xFC01404C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0

7	6	5	4	3	2	1	0
–	–	OVRUN	RXRDY	–	–	TXEMPTY	TXRDY

- **TXRDY: Transmit Ready Interrupt Mask**

0: The Transmit Ready Interrupt is disabled.

1: The Transmit Ready Interrupt is enabled.

- **TXEMPTY: Transmit Empty Interrupt Mask**

0: The Transmit Empty Interrupt is disabled.

1: The Transmit Empty Interrupt is enabled.

- **RXRDY: Receive Ready Interrupt Mask**

0: The Receive Ready Interrupt is disabled.

1: The Receive Ready Interrupt is enabled.

- **OVRUN: Receive Overrun Interrupt Mask**

0: The Receive Overrun Interrupt is disabled.

1: The Receive Overrun Interrupt is enabled.

- **CP0: Compare 0 Interrupt Mask**

0: The Compare 0 Interrupt is disabled.

1: The Compare 0 Interrupt is enabled.

- **CP1: Compare 1 Interrupt Mask**

0: The Compare 1 Interrupt is disabled.

1: The Compare 1 Interrupt is enabled.

- **TXSYN: Tx Sync Interrupt Mask**

0: The Tx Sync Interrupt is disabled.

1: The Tx Sync Interrupt is enabled.

- **RXSYN: Rx Sync Interrupt Mask**

0: The Rx Sync Interrupt is disabled.

1: The Rx Sync Interrupt is enabled.

40.9.17 SSC Write Protection Mode Register

Name: SSC_WPMR

Address: 0xF80080E4 (0), 0xFC0140E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII).

Refer to Section 40.8.10 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x535343	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

40.9.18 SSC Write Protection Status Register

Name: SSC_WPSR

Address: 0xF80080E8 (0), 0xFC0140E8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the SSC_WPSR.

1: A write protection violation has occurred since the last read of the SSC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protect Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

41. Debug Unit (DBGU)

41.1 Description

The Debug Unit (DBGU) provides a single entry point from the processor for access to all the debug capabilities of Atmel ARM-based systems.

The Debug Unit features a two-pin UART that can be used for several debug and trace purposes and offers an ideal medium for in-situ programming solutions and debug monitor communications. The Debug Unit two-pin UART can be used stand-alone for general purpose serial communication. Moreover, the association with DMA controller channels permits packet handling for these tasks with processor time reduced to a minimum.

The Debug Unit also makes the Debug Communication Channel (DCC) signals provided by the In-circuit Emulator of the ARM processor visible to the software. These signals indicate the status of the DCC read and write registers and generate an interrupt to the ARM processor, making possible the handling of the DCC under interrupt control.

Chip identifier registers permit recognition of the device and its revision. These registers indicate the sizes and types of the on-chip memories, as well as the set of embedded peripherals.

Finally, the Debug Unit features a Force NTRST capability that enables the software to decide whether to prevent access to the system via the In-circuit Emulator. This permits protection of the code, stored in ROM.

41.2 Embedded Characteristics

- System Peripheral to Facilitate Debug of Atmel ARM-based Systems
- Composed of Four Functions
  - Two-pin UART
  - Debug Communication Channel (DCC) Support
  - Chip ID Registers
  - ICE Access Prevention
- Two-pin UART
  - Implemented Features are USART Compatible
  - Independent Receiver and Transmitter with a Common Programmable Baud Rate Generator
  - Even, Odd, Mark or Space Parity Generation
  - Parity, Framing and Overrun Error Detection
  - Automatic Echo, Local Loopback and Remote Loopback Channel Modes
  - Interrupt Generation
  - Support for Two DMA Channels with Connection to Receiver and Transmitter
  - Digital Filter on Receive Line
- Debug Communication Channel Support
  - Offers Visibility of COMMRX and COMMTX Signals from the ARM Processor
  - Interrupt Generation
- Chip ID Registers
  - Identification of the Device Revision, Sizes of the Embedded Memories, Set of Peripherals
- ICE Access Prevention
  - Enables Software to Prevent System Access Through the ARM Processor’s ICE
  - Prevention is Made by Asserting the NTRST Line of the ARM Processor’s ICE

41.3 Block Diagram

Figure 41-1. Debug Unit Functional Block Diagram

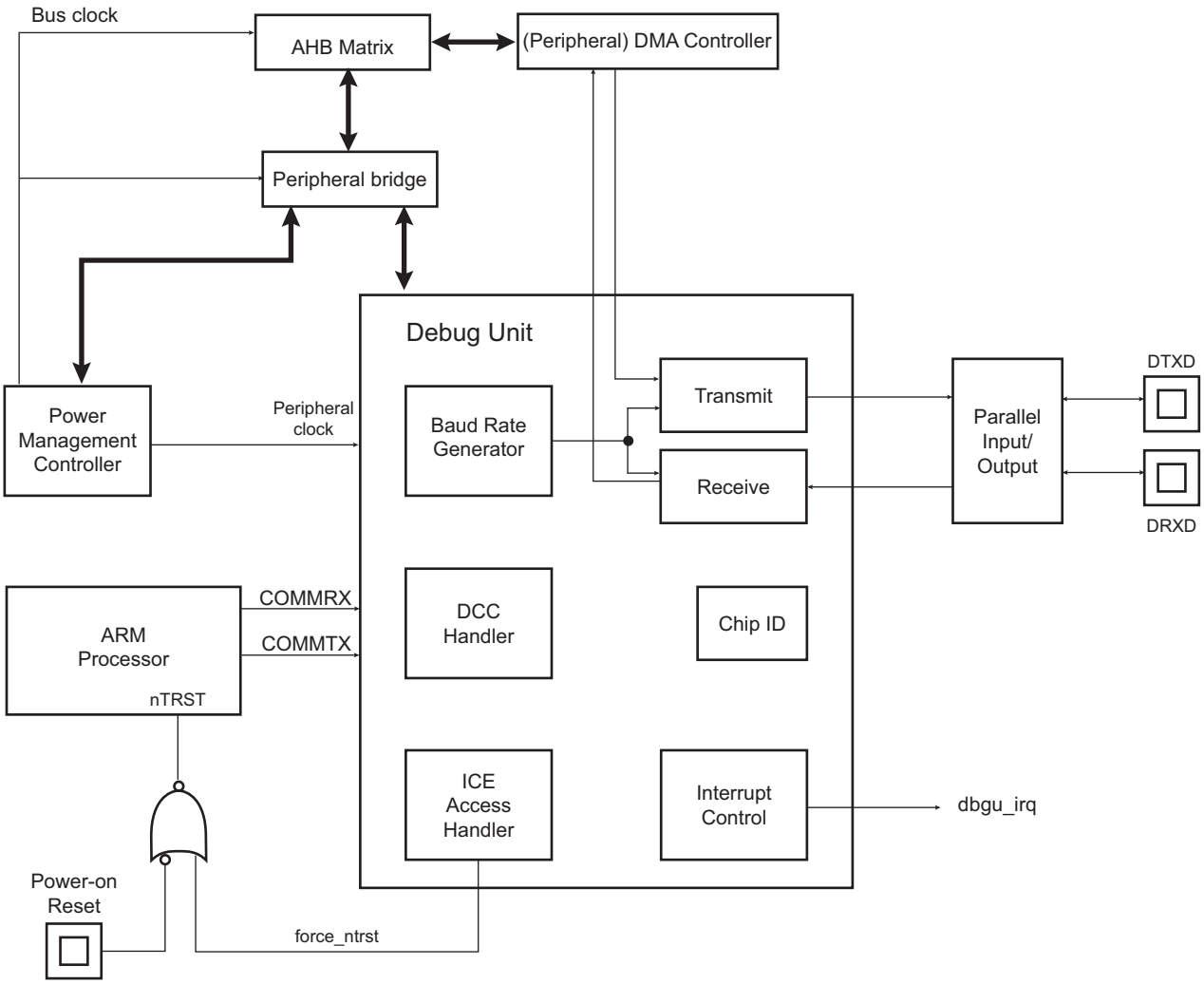
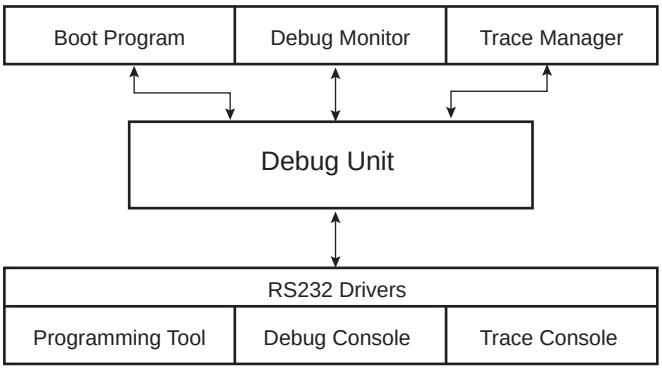


Table 41-1. Debug Unit Pin Description

Pin Name	Description	Type
DRXD	Debug Receive Data	Input
DTXD	Debug Transmit Data	Output

Figure 41-2. Debug Unit Application Example



41.4 Product Dependencies

41.4.1 I/O Lines

Depending on product integration, the Debug Unit pins may be multiplexed with PIO lines. In this case, the programmer must first configure the corresponding PIO Controller to enable I/O lines operations of the Debug Unit.

Table 41-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
DBGU	DRXD	PB24	A
DBGU	DTXD	PB25	A

41.4.2 Power Management

Depending on product integration, the Debug Unit clock may be controllable through the Power Management Controller. In this case, the programmer must first configure the PMC to enable the Debug Unit clock. Usually, the peripheral identifier used for this purpose is 1.

41.4.3 Interrupt Source

Depending on product integration, the Debug Unit interrupt line is connected to one of the interrupt sources of the Advanced Interrupt Controller. Interrupt handling requires programming of the AIC before configuring the Debug Unit. Usually, the Debug Unit interrupt line connects to the interrupt source 1 of the AIC, which may be shared with the real-time clock, the system timer interrupt lines and other system peripheral interrupts, as shown in Figure 41-1. This sharing requires the programmer to determine the source of the interrupt when the source 1 is triggered.

41.5 UART Operations

The Debug Unit operates as a UART, (asynchronous mode only) and supports only 8-bit character handling (with parity). It has no clock pin.

The Debug Unit's UART is made up of a receiver and a transmitter that operate independently, and a common baud rate generator. Receiver timeout and transmitter time guard are not implemented. However, all the implemented features are compatible with those of a standard USART.

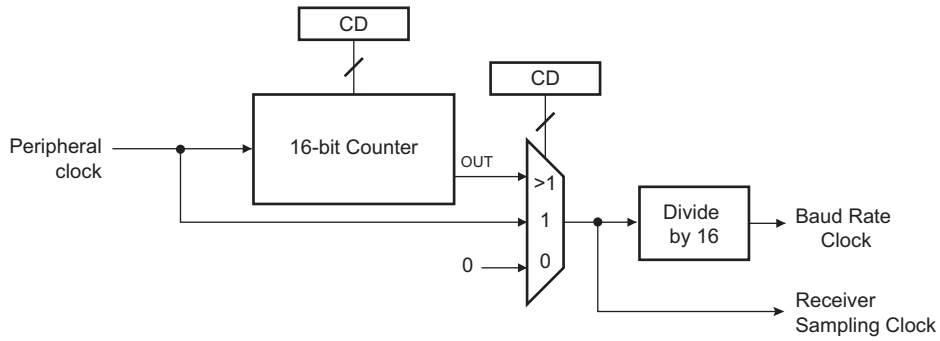
41.5.1 Baud Rate Generator

The baud rate generator provides the bit period clock named baud rate clock to both the receiver and the transmitter.

The baud rate clock is the peripheral clock divided by 16 times the value (CD) written in the Debug Unit Baud Rate Generator Register (DBGU_BRGR). If DBGU_BRGR is set to 0, the baud rate clock is disabled and the Debug Unit's UART remains inactive. The maximum allowable baud rate is peripheral clock divided by 16. The minimum allowable baud rate is peripheral clock divided by (16 x 65536).

$$\text{Baud Rate} = \frac{f_{\text{peripheral clock}}}{16 \times CD}$$

Figure 41-3. Baud Rate Generator



41.5.2 Receiver

41.5.2.1 Receiver Reset, Enable and Disable

After device reset, the Debug Unit receiver is disabled and must be enabled before being used. The receiver can be enabled by writing one to the RXEN bit in the Debug Unit Control Register (DBGU_CR). At this command, the receiver starts looking for a start bit.

The programmer can disable the receiver by writing a one to the RXDIS bit in the DBGU_CR. If the receiver is waiting for a start bit, it is immediately stopped. However, if the receiver has already detected a start bit and is receiving the data, it waits for the stop bit before actually stopping its operation.

The programmer can also put the receiver in its reset state by writing a one to the RSTRX bit in the DBGU_CR. In doing so, the receiver immediately stops its current operations and is disabled, whatever its current state. If RSTRX is applied when data is being processed, this data is lost.

41.5.2.2 Start Detection and Data Sampling

The Debug Unit only supports asynchronous operations, and this affects only its receiver. The Debug Unit receiver detects the start of a received character by sampling the DRXD signal until it detects a valid start bit. A low level (space) on DRXD is interpreted as a valid start bit if it is detected for more than 7 cycles of the sampling clock, which is 16 times the baud rate. Hence, a space that is longer than 7/16 of the bit period is detected as a valid start bit. A space which is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the DRXD at the theoretical midpoint of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (1-bit period) so the bit sampling point is eight cycles (0.5-bit period) after the start of the bit. The first sampling point is therefore 24 cycles (1.5-bit periods) after the falling edge of the start bit was detected.

Each subsequent bit is sampled 16 cycles (1-bit period) after the previous one.

Figure 41-4. Start Bit Detection

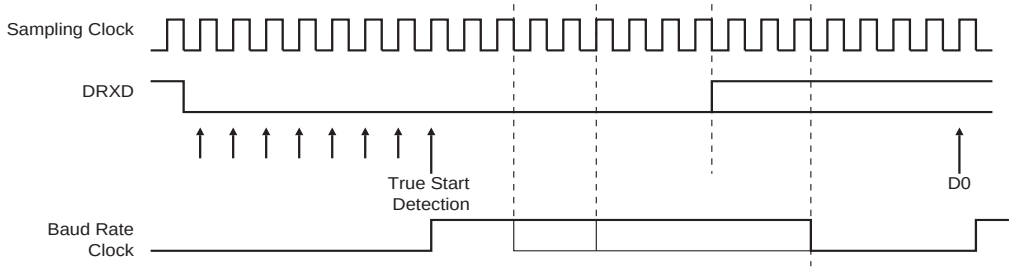
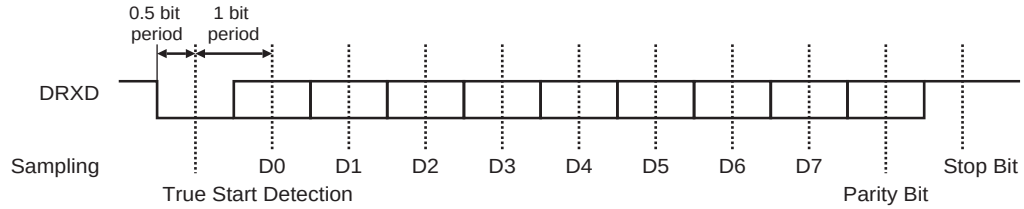


Figure 41-5. Character Reception

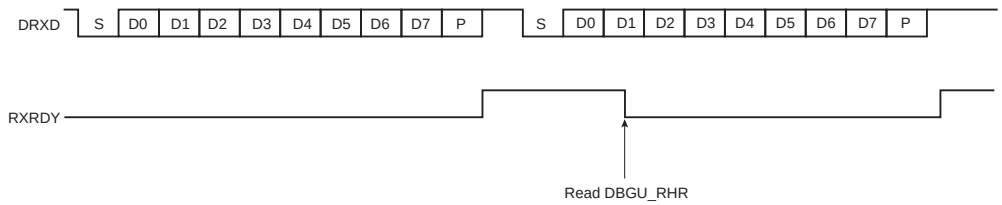
Example: 8-bit, parity enabled 1 stop



41.5.2.3 Receiver Ready

When a complete character is received, it is transferred to the Debug Unit Receive Holding Register (DBGU_RHR) and the RXRDY status bit in the Debug Unit Status Register (DBGU_SR) is set. The bit RXRDY is automatically cleared when DBGU_RHR is read.

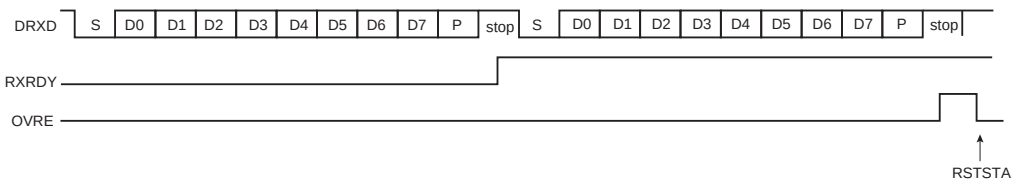
Figure 41-6. Receiver Ready



41.5.2.4 Receiver Overrun

If DBGU_RHR has not been read by the software (or the Peripheral Data Controller or DMA Controller) since the last transfer, the RXRDY bit is still set and a new character is received, the OVRE status bit in DBGU_SR is set. OVRE is cleared when the software writes a one to the bit RSTSTA (Reset Status) in the DBGU_CR.

Figure 41-7. Receiver Overrun

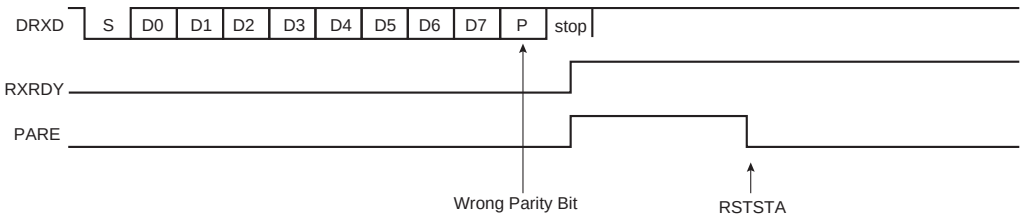


41.5.2.5 Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits, in accordance with the field PAR in the Debug Unit Mode Register (DBGU_MR). It then compares the result with the received parity bit. If different, the parity error bit PARE in DBGU_SR is set at the same time as the RXRDY is set. The parity bit is

cleared when a one is written to the bit RSTSTA (Reset Status) in the DBGU_CR. If a new character is received before the reset status command is written, the PARE bit remains at 1.

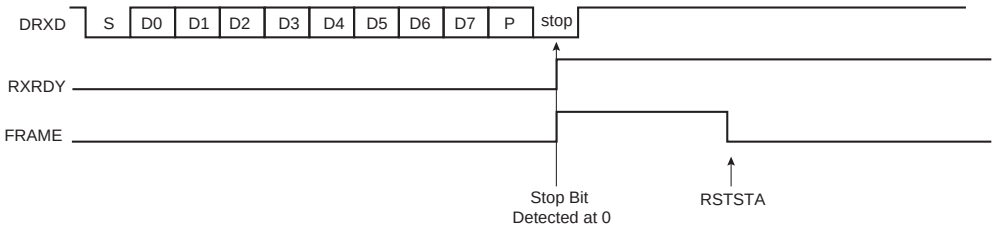
Figure 41-8. Parity Error



41.5.2.6 Receiver Framing Error

When a start bit is detected, it generates a character reception when all the data bits have been sampled. The stop bit is also sampled and when it is detected at 0, the FRAME (Framing Error) bit in DBGU_SR is set at the same time as the RXRDY bit is set. The bit FRAME remains high until a one is written to the RSTSTA bit in the DBGU_CR.

Figure 41-9. Receiver Framing Error



41.5.2.7 Receiver Digital Filter

The debug unit embeds a digital filter on the receive line. It is disabled by default and can be enabled by writing a logical 1 in the FILTER bit of DBGU_MR. When enabled, the receive line is sampled using the 16x bit clock and a three-sample filter (majority 2 over 3) determines the value of the line.

41.5.3 Transmitter

41.5.3.1 Transmitter Reset, Enable and Disable

After device reset, the Debug Unit transmitter is disabled and it must be enabled before being used. The transmitter is enabled by writing a one to the TXEN bit in DBGU_CR. From this command, the transmitter waits for a character to be written in the Transmit Holding Register (DBGU_THR) before actually starting the transmission.

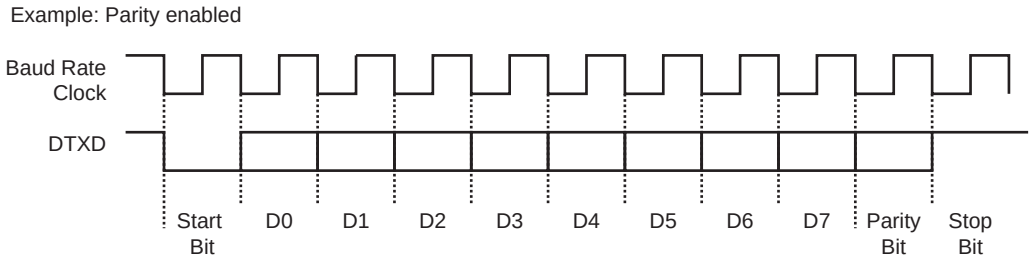
The programmer can disable the transmitter by writing a one to the TXDIS bit in the DBGU_CR. If the transmitter is not operating, it is immediately stopped. However, if a character is being processed into the Shift Register and/or a character has been written in DBGU_THR, the characters are completed before the transmitter is actually stopped.

The programmer can also put the transmitter in its reset state by writing a one to the RSTTX bit in the DBGU_CR. This immediately stops the transmitter, whether or not it is processing characters.

41.5.3.2 Transmit Format

The Debug Unit transmitter drives the pin DTXD at the baud rate clock speed. The line is driven depending on the format defined in DBGU_MR and the data stored in the Shift Register. One start bit at level 0, then the 8 data bits, from the lowest to the highest bit, one optional parity bit and one stop bit at 1 are consecutively shifted out as shown on the following figure. The field PARE in DBGU_MR defines whether or not a parity bit is shifted out. When a parity bit is enabled, it can be selected between an odd parity, an even parity, or a fixed space or mark bit.

Figure 41-10. Character Transmission

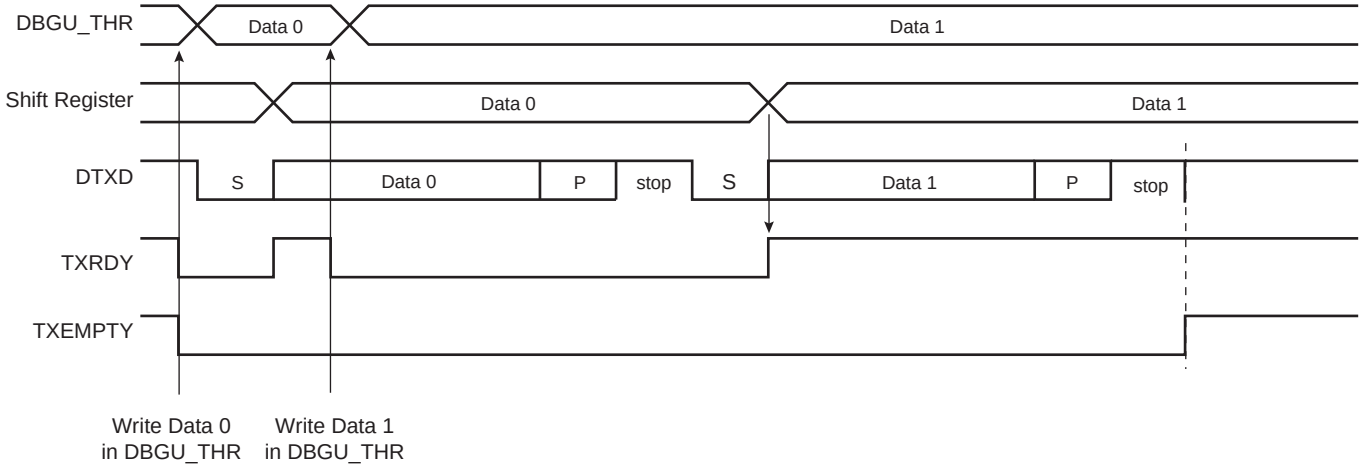


41.5.3.3 Transmitter Control

When the transmitter is enabled, the bit TXRDY (Transmitter Ready) is set in DBGU_SR. The transmission starts when the programmer writes in DBGU_THR, and after the written character is transferred from DBGU_THR to the Shift Register. The bit TXRDY remains high until a second character is written in DBGU_THR. As soon as the first character is completed, the last character written in DBGU_THR is transferred into the shift register and TXRDY rises again, showing that the holding register is empty.

When both the Shift Register and the DBGU_THR are empty, i.e., all the characters written in DBGU_THR have been processed, the bit TXEMPTY rises after the last stop bit has been completed.

Figure 41-11. Transmitter Control



41.5.4 DMA Support

Both the receiver and the transmitter of the Debug Unit's UART are connected to a DMA Controller (DMAC) channel.

The DMA Controller channels are programmed via registers that are mapped within the DMAC user interface.

41.5.5 Test Modes

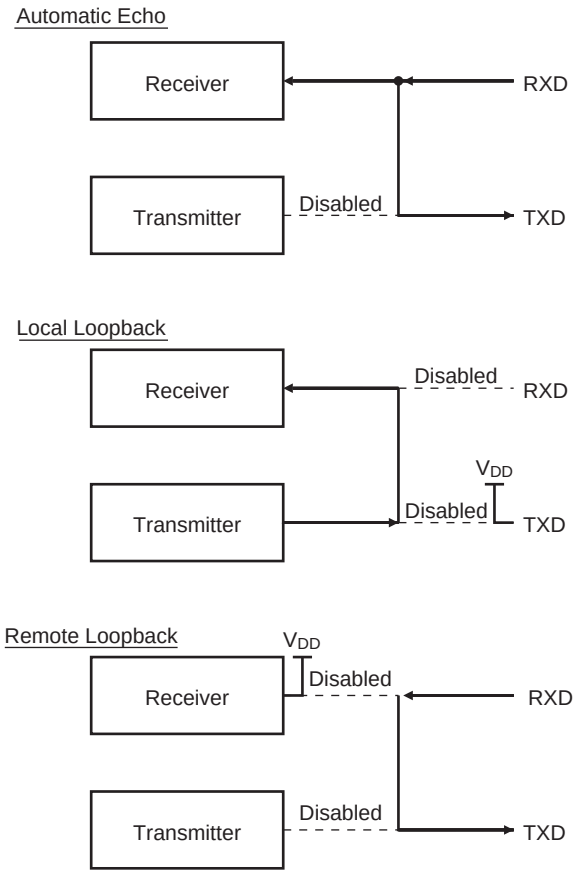
The Debug Unit supports three tests modes. These modes of operation are programmed through the field CHMODE (Channel Mode) in DBGU_MR.

The Automatic Echo mode allows bit-by-bit retransmission. When a bit is received on the DRXD line, it is sent to the DTXD line. The transmitter operates normally, but has no effect on the DTXD line.

The Local Loopback mode allows the transmitted characters to be received. DTXD and DRXD pins are not used and the output of the transmitter is internally connected to the input of the receiver. The DRXD pin level has no effect and the DTXD line is held high, as in idle state.

The Remote Loopback mode directly connects the DRXD pin to the DTXD line. The transmitter and the receiver are disabled and have no effect. This mode allows a bit-by-bit retransmission.

Figure 41-12. Test Modes



41.5.6 Debug Communication Channel Support

The Debug Unit handles the signals COMMRX and COMMTX that come from the Debug Communication Channel of the ARM processor and are driven by the In-circuit Emulator.

The Debug Communication Channel contains two registers that are accessible through the ICE Breaker on the JTAG side and through the coprocessor 0 on the ARM processor side.

As a reminder, the following instructions are used to read and write the Debug Communication Channel:

```
MRC                    p14, 0, Rd, c1, c0, 0
```

Returns the debug communication data read register into Rd

```
MCR                    p14, 0, Rd, c1, c0, 0
```

Writes the value in Rd to the debug communication data write register.

The bits COMMRX and COMMTX, which indicate, respectively, that the read register has been written by the debugger but not yet read by the processor, and that the write register has been written by the processor and not yet read by the debugger, are wired on the two highest bits of DBGU_SR. These bits can generate an interrupt. This feature permits handling under interrupt a debug link between a debug monitor running on the target system and a debugger.

41.5.7 Chip Identifier

The Debug Unit features two chip identifier registers, Debug Unit Chip ID Register (DBGU_CIDR) and Debug Unit Extension ID Register (DBGU_EXID). Both registers contain a hard-wired value that is read-only.

The first register (DBGU_CIDR) contains the following fields:

- EXT: shows the use of the extension identifier register
- NVPTYP and NVPSIZ: identifies the type of embedded non-volatile memory and its size
- ARCH: identifies the set of embedded peripherals
- SRAMSIZ: indicates the size of the embedded SRAM
- EPROC: indicates the embedded ARM processor
- VERSION: gives the revision of the silicon

The second register (DBGU_EXID) is device-dependent and is read as 0 if the bit EXT is 0 in DBGU_CIDR.

41.5.8 ICE Access Prevention

The Debug Unit allows blockage of access to the system through the ARM processor's ICE interface. This feature is implemented via the Debug Unit Force NTRST Register (DBGU_FNR), that allows assertion of the NTRST signal of the ICE Interface. Writing the bit FNTRST (Force NTRST) to 1 in this register prevents any activity on the TAP controller.

On standard devices, the bit FNTRST resets to 0 and thus does not prevent ICE access.

This feature is especially useful on custom ROM devices for customers who do not want their on-chip code to be visible.

41.6 Debug Unit (DBGU) User Interface

Table 41-3. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	Control Register	DBGU_CR	Write-only	–
0x0004	Mode Register	DBGU_MR	Read/Write	0x00000000
0x0008	Interrupt Enable Register	DBGU_IER	Write-only	–
0x000C	Interrupt Disable Register	DBGU_IDR	Write-only	–
0x0010	Interrupt Mask Register	DBGU_IMR	Read-only	0x00000000
0x0014	Status Register	DBGU_SR	Read-only	–
0x0018	Receive Holding Register	DBGU_RHR	Read-only	0x00000000
0x001C	Transmit Holding Register	DBGU_THR	Write-only	–
0x0020	Baud Rate Generator Register	DBGU_BRGR	Read/Write	0x00000000
0x0024–0x003C	Reserved	–	–	–
0x0040	Chip ID Register	DBGU_CIDR	Read-only	0x8A5C07Cx
0x0044	Chip ID Extension Register	DBGU_EXID	Read-only	–
0x0048	Force NTRST Register	DBGU_FNR	Read/Write	0x00000000
0x004C–0x00FC	Reserved	–	–	–

41.6.1 Debug Unit Control Register

Name: DBGU_CR

Address: 0xFC069000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

- **RSTRX: Reset Receiver**

0: No effect.

1: The receiver logic is reset and disabled. If a character is being received, the reception is aborted.

- **RSTTX: Reset Transmitter**

0: No effect.

1: The transmitter logic is reset and disabled. If a character is being transmitted, the transmission is aborted.

- **RXEN: Receiver Enable**

0: No effect.

1: The receiver is enabled if RXDIS is 0.

- **RXDIS: Receiver Disable**

0: No effect.

1: The receiver is disabled. If a character is being processed and RSTRX is not set, the character is completed before the receiver is stopped.

- **TXEN: Transmitter Enable**

0: No effect.

1: The transmitter is enabled if TXDIS is 0.

- **TXDIS: Transmitter Disable**

0: No effect.

1: The transmitter is disabled. If a character is being processed and a character has been written in the DBGU_THR and RSTTX is not set, both characters are completed before the transmitter is stopped.

- **RSTSTA: Reset Status Bits**

0: No effect.

1: Resets the status bits PARE, FRAME and OVRE in DBGU_SR.

41.6.2 Debug Unit Mode Register

Name: DBGU_MR

Address: 0xFC069004

Access: Read/Write

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

- **PAR: Parity Type**

Value	Name	Description
0b000	EVEN	Even Parity
0b001	ODD	Odd Parity
0b010	SPACE	Space: Parity forced to 0
0b011	MARK	Mark: Parity forced to 1
0b1xx	NONE	No Parity

- **CHMODE: Channel Mode**

Value	Name	Description
0b00	NORM	Normal Mode
0b01	AUTO	Automatic Echo
0b10	LOCLOOP	Local Loopback
0b11	REMLOOP	Remote Loopback

- **FILTER: Receiver Digital Filter**

0 (DISABLED): DBGU does not filter the receive line.

1 (ENABLED): DBGU filters the receive line using a three-sample filter (16x-bit clock) (2 over 3 majority).

41.6.3 Debug Unit Interrupt Enable Register

Name: DBGU_IER

Address: 0xFC069008

Access: Write-only

31	30	29	28	27	26	25	24
COMMRX	COMMTX	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

- **RXRDY: Enable RXRDY Interrupt**
- **TXRDY: Enable TXRDY Interrupt**
- **OVRE: Enable Overrun Error Interrupt**
- **FRAME: Enable Framing Error Interrupt**
- **PARE: Enable Parity Error Interrupt**
- **TXEMPTY: Enable TXEMPTY Interrupt**
- **COMMTX: Enable COMMTX (from ARM) Interrupt**
- **COMMRX: Enable COMMRX (from ARM) Interrupt**

0: No effect.

1: Enables the corresponding interrupt.

41.6.4 Debug Unit Interrupt Disable Register

Name: DBGU_IDR

Address: 0xFC06900C

Access: Write-only

31	30	29	28	27	26	25	24
COMMRX	COMMTX	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

- **RXRDY: Disable RXRDY Interrupt**
- **TXRDY: Disable TXRDY Interrupt**
- **OVRE: Disable Overrun Error Interrupt**
- **FRAME: Disable Framing Error Interrupt**
- **PARE: Disable Parity Error Interrupt**
- **TXEMPTY: Disable TXEMPTY Interrupt**
- **COMMTX: Disable COMMTX (from ARM) Interrupt**
- **COMMRX: Disable COMMRX (from ARM) Interrupt**

0: No effect.

1: Disables the corresponding interrupt.

41.6.5 Debug Unit Interrupt Mask Register

Name: DBGU_IMR

Address: 0xFC069010

Access: Read-only

31	30	29	28	27	26	25	24
COMMRX	COMMTX	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

- **RXRDY: Mask RXRDY Interrupt**
- **TXRDY: Disable TXRDY Interrupt**
- **OVRE: Mask Overrun Error Interrupt**
- **FRAME: Mask Framing Error Interrupt**
- **PARE: Mask Parity Error Interrupt**
- **TXEMPTY: Mask TXEMPTY Interrupt**
- **COMMTX: Mask COMMTX Interrupt**
- **COMMRX: Mask COMMRX Interrupt**

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

41.6.6 Debug Unit Status Register

Name: DBGU_SR

Address: 0xFC069014

Access: Read-only

31	30	29	28	27	26	25	24
COMMRX	COMMTX	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

- **RXRDY: Receiver Ready**

0: No character has been received since the last read of the DBGU_RHR, or the receiver is disabled.

1: At least one complete character has been received, transferred to DBGU_RHR and not yet read.

- **TXRDY: Transmitter Ready**

0: A character has been written to DBGU_THR and not yet transferred to the Shift Register, or the transmitter is disabled.

1: There is no character written to DBGU_THR not yet transferred to the Shift Register.

- **OVRE: Overrun Error**

0: No overrun error has occurred since the last RSTSTA.

1: At least one overrun error has occurred since the last RSTSTA.

- **FRAME: Framing Error**

0: No framing error has occurred since the last RSTSTA.

1: At least one framing error has occurred since the last RSTSTA.

- **PARE: Parity Error**

0: No parity error has occurred since the last RSTSTA.

1: At least one parity error has occurred since the last RSTSTA.

- **TXEMPTY: Transmitter Empty**

0: There are characters in DBGU_THR, or characters being processed by the transmitter, or the transmitter is disabled.

1: There are no characters in DBGU_THR and there are no characters being processed by the transmitter.

- **COMMTX: Debug Communication Channel Write Status**

0: COMMTX from the ARM processor is inactive.

1: COMMTX from the ARM processor is active.

- **COMMRX: Debug Communication Channel Read Status**

0: COMMRX from the ARM processor is inactive.

1: COMMRX from the ARM processor is active.

41.6.7 Debug Unit Receive Holding Register

Name: DBGU_RHR

Address: 0xFC069018

Access: Read-only

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
RXCHR							

- **RXCHR: Received Character**

Last received character if RXRDY is set.

41.6.8 Debug Unit Transmit Holding Register

Name: DBGU_THR

Address: 0xFC06901C

Access: Write-only

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
TXCHR							

- **TXCHR: Character to be Transmitted**

Next character to be transmitted after the current character if TXRDY is not set.

41.6.9 Debug Unit Baud Rate Generator Register

Name: DBGU_BRGR

Address: 0xFC069020

Access: Read/Write

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

- **CD: Clock Divisor**

Value	Name	Description
0	DISABLED	DBGU Disabled
1	MCK	Peripheral clock
2–65535	–	Peripheral clock/ (CD x 16)

41.6.10 Debug Unit Chip ID Register

Name: DBGU_CIDR

Address: 0xFC069040

Access: Read-only

31	30	29	28	27	26	25	24
EXT	NVPTYP			ARCH			

23	22	21	20	19	18	17	16
ARCH				SRAMSIZ			

15	14	13	12	11	10	9	8
NVPSIZ2				NVPSIZ			

7	6	5	4	3	2	1	0
EPROC			VERSION				

- **VERSION: Version of the Device**

Values depend on the version of the device.

- **EPROC: Embedded Processor**

Value	Name	Description
1	ARM946ES	ARM946ES
2	ARM7TDMI	ARM7TDMI
3	CM3	Cortex-M3
4	ARM920T	ARM920T
5	ARM926EJS	ARM926EJ-S
6	CA5	Cortex-A5

- **NVPSIZ: Nonvolatile Program Memory Size**

Value	Name	Description
0	NONE	None
1	8K	8 Kbytes
2	16K	16 Kbytes
3	32K	32 Kbytes
4	–	Reserved
5	64K	64 Kbytes
6	–	Reserved
7	128K	128 Kbytes
8	–	Reserved
9	256K	256 Kbytes
10	512K	512 Kbytes
11	–	Reserved
12	1024K	1024 Kbytes
13	–	Reserved

Value	Name	Description
14	2048K	2048 Kbytes
15	–	Reserved

- **NVPSIZ2: Second Nonvolatile Program Memory Size**

Value	Name	Description
0	NONE	None
1	8K	8 Kbytes
2	16K	16 Kbytes
3	32K	32 Kbytes
4	–	Reserved
5	64K	64 Kbytes
6		Reserved
7	128K	128 Kbytes
8	–	Reserved
9	256K	256 Kbytes
10	512K	512 Kbytes
11	–	Reserved
12	1024K	1024 Kbytes
13	–	Reserved
14	2048K	2048 Kbytes
15	–	Reserved

- **SRAMSIZ: Internal SRAM Size**

Value	Name	Description
0	–	Reserved
1	1K	1 Kbytes
2	2K	2 Kbytes
3	6K	6 Kbytes
4	112K	112 Kbytes
5	4K	4 Kbytes
6	80K	80 Kbytes
7	160K	160 Kbytes
8	8K	8 Kbytes
9	16K	16 Kbytes
10	32K	32 Kbytes
11	64K	64 Kbytes
12	128K	128 Kbytes

Value	Name	Description
13	256K	256 Kbytes
14	96K	96 Kbytes
15	512K	512 Kbytes

- **ARCH: Architecture Identifier**

Value	Name	Description
0xA5	ATSAMA5xx	ATSAMA5xx Series

- **NVPTYP: Nonvolatile Program Memory Type**

Value	Name	Description
0	ROM	ROM
1	ROMLESS	ROMless or on-chip Flash
4	SRAM	SRAM emulating ROM
2	FLASH	Embedded Flash Memory
3	ROM_FLASH	ROM and Embedded Flash Memory NVPSIZ is ROM size NVPSIZ2 is Flash size

- **EXT: Extension Flag**

0: Chip ID has a single register definition without extension.

1: An extended Chip ID exists.

41.6.11 Debug Unit Chip ID Extension Register

Name: DBGU_EXID

Address: 0xFC069044

Access: Read-only

31	30	29	28	27	26	25	24
EXID							

23	22	21	20	19	18	17	16
EXID							

15	14	13	12	11	10	9	8
EXID							

7	6	5	4	3	2	1	0
EXID							

- **EXID: Chip ID Extension**

Read as 0 if the bit EXT in DBGU_CIDR is 0.

41.6.12 Debug Unit Force NTRST Register

Name: DBGU_FNR

Address: 0xFC069048

Access: Read/Write

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
–	–	–	–	–	–	–	FNTRST

- **FNTRST: Force NTRST**

0: NTRST of the ARM processor’s TAP controller is driven by the power_on_reset signal.

1: NTRST of the ARM processor’s TAP controller is held low.

42. Universal Asynchronous Receiver Transmitter (UART)

42.1 Description

The Universal Asynchronous Receiver Transmitter (UART) features a two-pin UART that can be used for communication and trace purposes and offers an ideal medium for in-situ programming solutions.

Moreover, the association with a DMA controller permits packet handling for these tasks with processor time reduced to a minimum.

42.2 Embedded Characteristics

- Two-pin UART
  - Independent Receiver and Transmitter with a Common Programmable Baud Rate Generator
  - Even, Odd, Mark or Space Parity Generation
  - Parity, Framing and Overrun Error Detection
  - Automatic Echo, Local Loopback and Remote Loopback Channel Modes
  - Interrupt Generation
  - Support for Two DMA Channels with Connection to Receiver and Transmitter
  - Register Write Protection

42.3 Block Diagram

Figure 42-1. UART Block Diagram

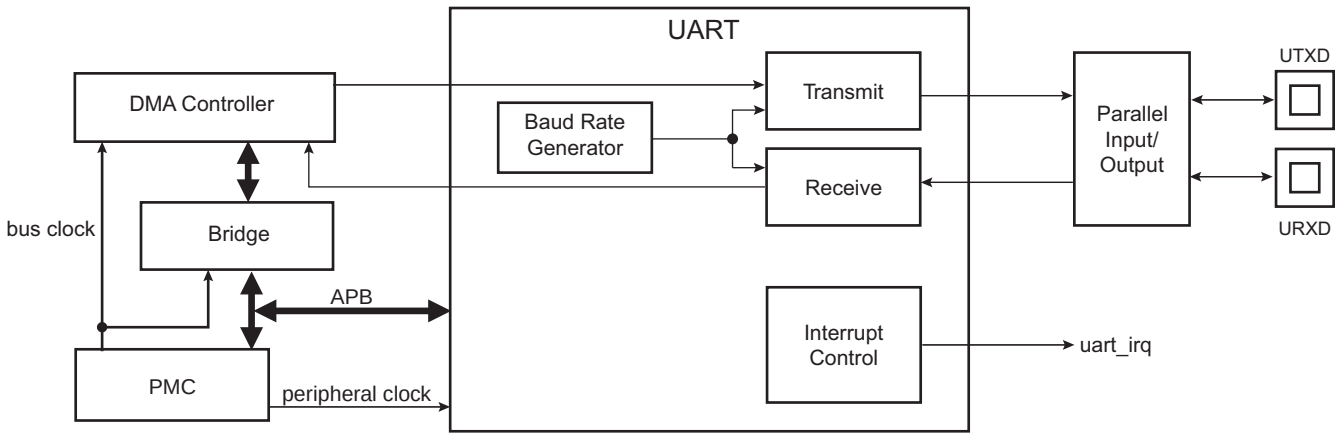


Table 42-1. UART Pin Description

Pin Name	Description	Type
URXD	UART Receive Data	Input
UTXD	UART Transmit Data	Output

42.4 Product Dependencies

42.4.1 I/O Lines

The UART pins are multiplexed with PIO lines. The user must first configure the corresponding PIO Controller to enable I/O line operations of the UART.

Table 42-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
UART0	URXD0	PE29	B
UART0	UTXD0	PE30	B
UART1	URXD1	PC25	C
UART1	UTXD1	PC26	C

42.4.2 Power Management

The UART clock can be controlled through the Power Management Controller (PMC). In this case, the user must first configure the PMC to enable the UART clock. Usually, the peripheral identifier used for this purpose is 1.

42.4.3 Interrupt Sources

The UART interrupt line is connected to one of the interrupt sources of the Interrupt Controller. Interrupt handling requires programming of the Interrupt Controller before configuring the UART.

Table 42-3. Peripheral IDs

Instance	ID
UART0	27
UART1	28

42.5 Functional Description

The UART operates in Asynchronous mode only and supports only 8-bit character handling (with parity). It has no clock pin.

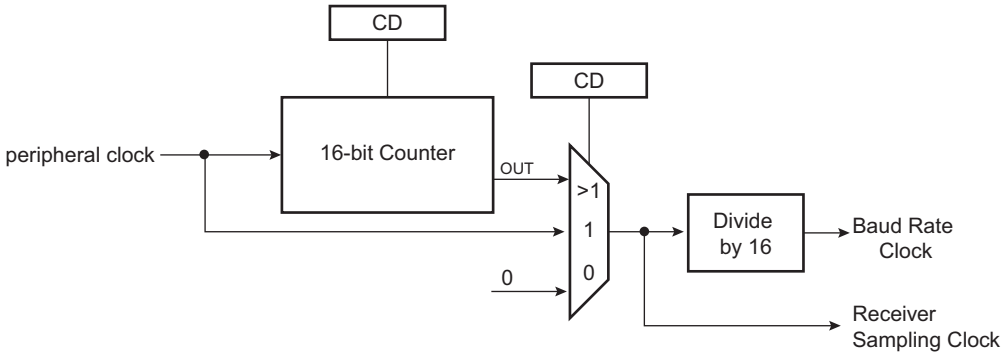
The UART is made up of a receiver and a transmitter that operate independently, and a common baud rate generator. Receiver timeout and transmitter time guard are not implemented. However, all the implemented features are compatible with those of a standard USART.

42.5.1 Baud Rate Generator

The baud rate generator provides the bit period clock named baud rate clock to both the receiver and the transmitter.

The baud rate clock is the peripheral clock divided by 16 times the clock divisor (CD) value written in the Baud Rate Generator register (UART_BRGR). If UART_BRGR is set to 0, the baud rate clock is disabled and the UART remains inactive. The maximum allowable baud rate is peripheral clock divided by 16. The minimum allowable baud rate is peripheral clock divided by (16 x 65536).

Figure 42-2. Baud Rate Generator



42.5.2 Receiver

42.5.2.1 Receiver Reset, Enable and Disable

After device reset, the UART receiver is disabled and must be enabled before being used. The receiver can be enabled by writing the Control Register (UART_CR) with the bit RXEN at 1. At this command, the receiver starts looking for a start bit.

The programmer can disable the receiver by writing UART_CR with the bit RXDIS at 1. If the receiver is waiting for a start bit, it is immediately stopped. However, if the receiver has already detected a start bit and is receiving the data, it waits for the stop bit before actually stopping its operation.

The receiver can be put in reset state by writing UART_CR with the bit RSTRX at 1. In this case, the receiver immediately stops its current operations and is disabled, whatever its current state. If RSTRX is applied when data is being processed, this data is lost.

42.5.2.2 Start Detection and Data Sampling

The UART only supports asynchronous operations, and this affects only its receiver. The UART receiver detects the start of a received character by sampling the URXD signal until it detects a valid start bit. A low level (space) on URXD is interpreted as a valid start bit if it is detected for more than seven cycles of the sampling clock, which is 16 times the baud rate. Hence, a space that is longer than 7/16 of the bit period is detected as a valid start bit. A space which is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the URXD at the theoretical midpoint of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (1-bit period) so the bit sampling point is eight cycles (0.5-bit period) after the start of the bit. The first sampling point is therefore 24 cycles (1.5-bit periods) after detecting the falling edge of the start bit.

Each subsequent bit is sampled 16 cycles (1-bit period) after the previous one.

Figure 42-3. Start Bit Detection

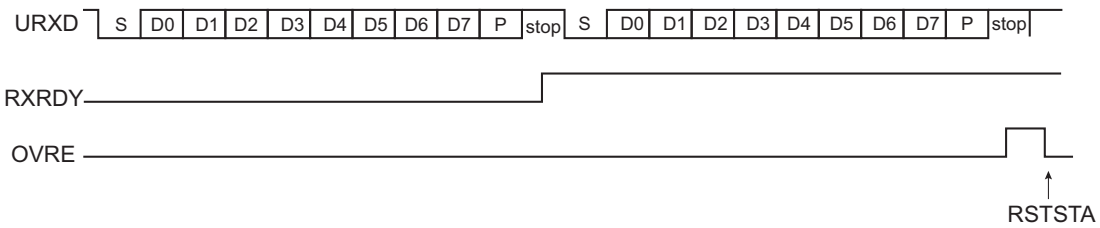
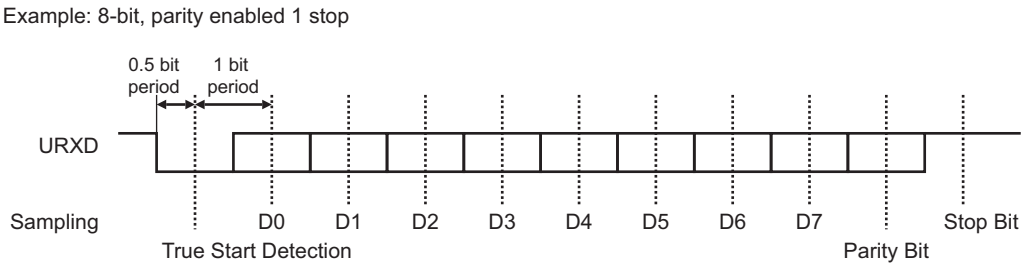


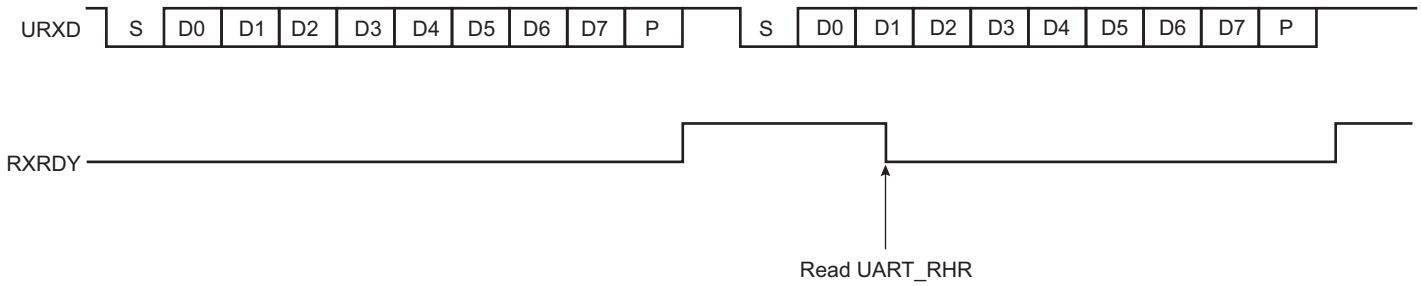
Figure 42-4. Character Reception



42.5.2.3 Receiver Ready

When a complete character is received, it is transferred to the Receive Holding Register (UART_RHR) and the RXRDY status bit in the Status Register (UART_SR) is set. The bit RXRDY is automatically cleared when UART_RHR is read.

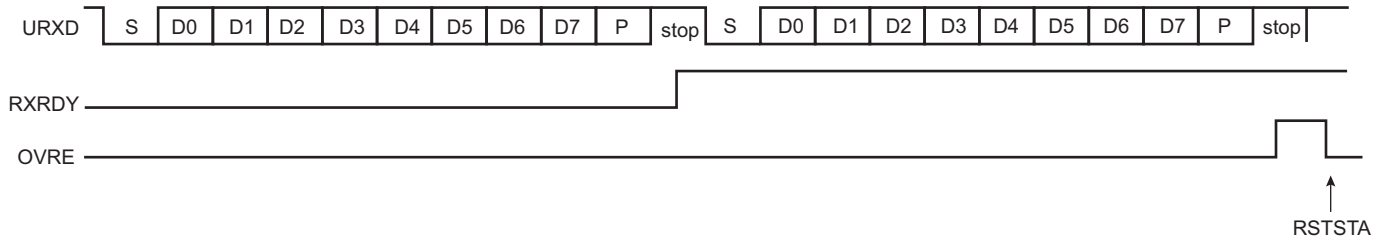
Figure 42-5. Receiver Ready



42.5.2.4 Receiver Overrun

The OVRE status bit in UART_SR is set if UART_RHR has not been read by the software (or the DMA Controller) since the last transfer, the RXRDY bit is still set and a new character is received. OVRE is cleared when the software writes a 1 to the bit RSTSTA (Reset Status) in UART_CR.

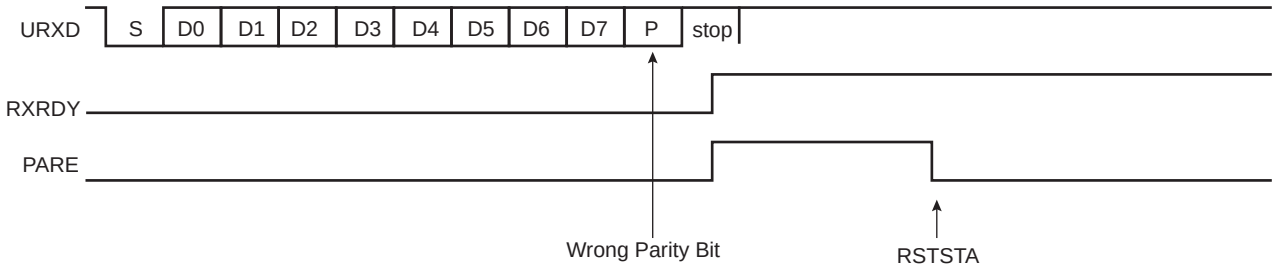
Figure 42-6. Receiver Overrun



42.5.2.5 Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits, in accordance with the field PAR in the Mode Register (UART_MR). It then compares the result with the received parity bit. If different, the parity error bit PARE in UART_SR is set at the same time RXRDY is set. The parity bit is cleared when UART_CR is written with the bit RSTSTA (Reset Status) at 1. If a new character is received before the reset status command is written, the PARE bit remains at 1.

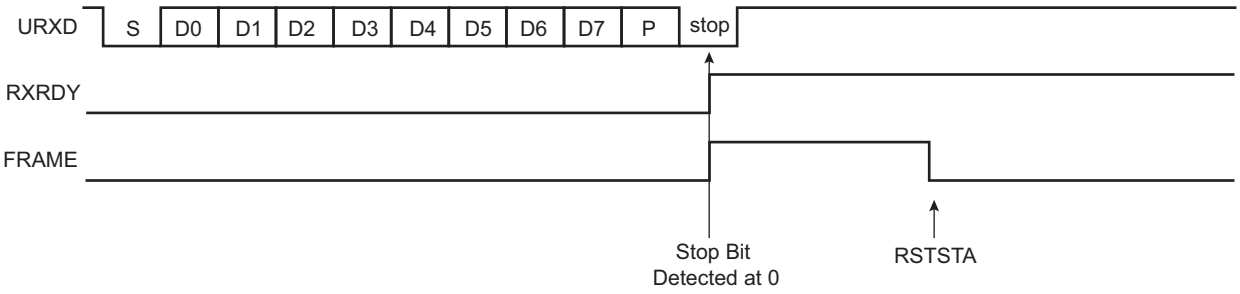
Figure 42-7. Parity Error



42.5.2.6 Receiver Framing Error

When a start bit is detected, it generates a character reception when all the data bits have been sampled. The stop bit is also sampled and when it is detected at 0, the FRAME (Framing Error) bit in UART_SR is set at the same time the RXRDY bit is set. The FRAME bit remains high until the Control Register (UART_CR) is written with the bit RSTSTA at 1.

Figure 42-8. Receiver Framing Error



42.5.3 Transmitter

42.5.3.1 Transmitter Reset, Enable and Disable

After device reset, the UART transmitter is disabled and must be enabled before being used. The transmitter is enabled by writing UART_CR with the bit TXEN at 1. From this command, the transmitter waits for a character to be written in the Transmit Holding Register (UART_THR) before actually starting the transmission.

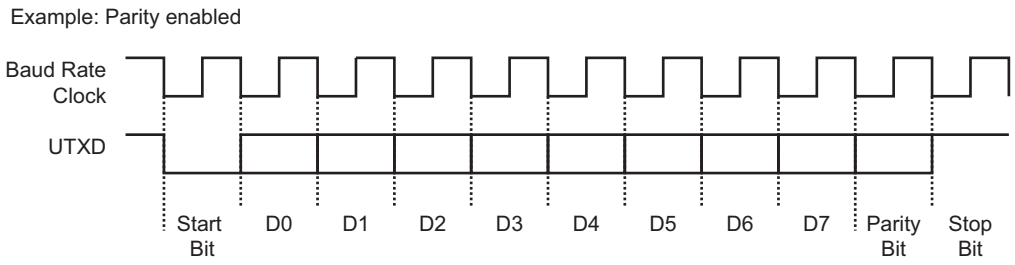
The programmer can disable the transmitter by writing UART_CR with the bit TXDIS at 1. If the transmitter is not operating, it is immediately stopped. However, if a character is being processed into the internal shift register and/or a character has been written in the UART_THR, the characters are completed before the transmitter is actually stopped.

The programmer can also put the transmitter in its reset state by writing the UART_CR with the bit RSTTX at 1. This immediately stops the transmitter, whether or not it is processing characters.

42.5.3.2 Transmit Format

The UART transmitter drives the pin UTXD at the baud rate clock speed. The line is driven depending on the format defined in UART_MR and the data stored in the internal shift register. One start bit at level 0, then the 8 data bits, from the lowest to the highest bit, one optional parity bit and one stop bit at 1 are consecutively shifted out as shown in the following figure. The field PARE in UART_MR defines whether or not a parity bit is shifted out. When a parity bit is enabled, it can be selected between an odd parity, an even parity, or a fixed space or mark bit.

Figure 42-9. Character Transmission

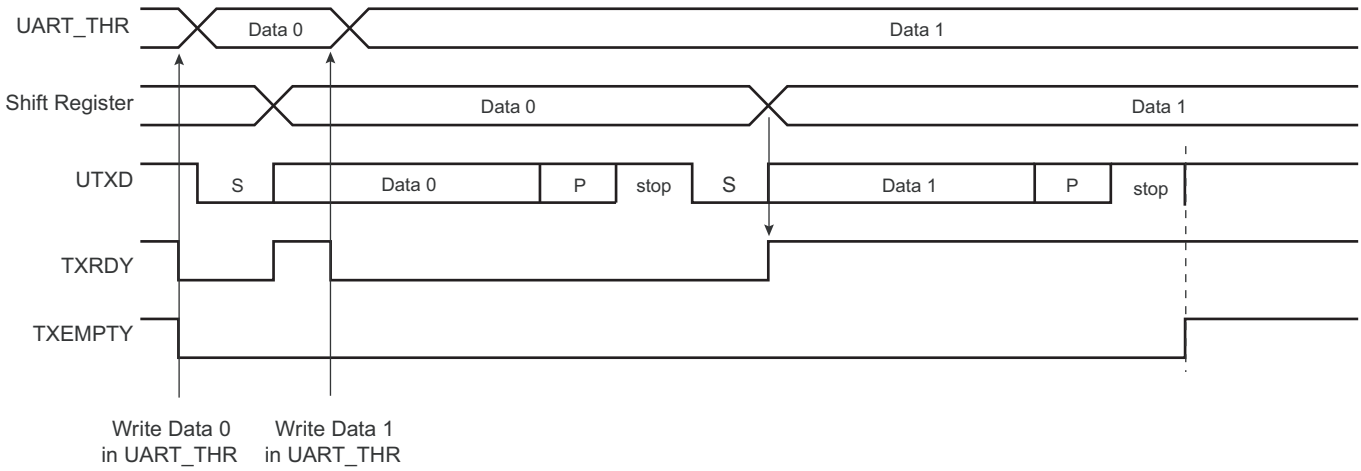


42.5.3.3 Transmitter Control

When the transmitter is enabled, the bit TXRDY (Transmitter Ready) is set in UART_SR. The transmission starts when the programmer writes in the UART_THR, and after the written character is transferred from UART_THR to the internal shift register. The TXRDY bit remains high until a second character is written in UART_THR. As soon as the first character is completed, the last character written in UART_THR is transferred into the internal shift register and TXRDY rises again, showing that the holding register is empty.

When both the internal shift register and UART_THR are empty, i.e., all the characters written in UART_THR have been processed, the TXEMPTY bit rises after the last stop bit has been completed.

Figure 42-10. Transmitter Control



42.5.4 DMA Support

Both the receiver and the transmitter of the UART are connected to a DMA Controller (DMAC) channel.

The DMA Controller channels are programmed via registers that are mapped within the DMAC user interface.

42.5.5 Register Write Protection

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

The following registers can be write-protected:

- UART Mode Register
- UART Baud Rate Generator Register

42.5.6 Test Modes

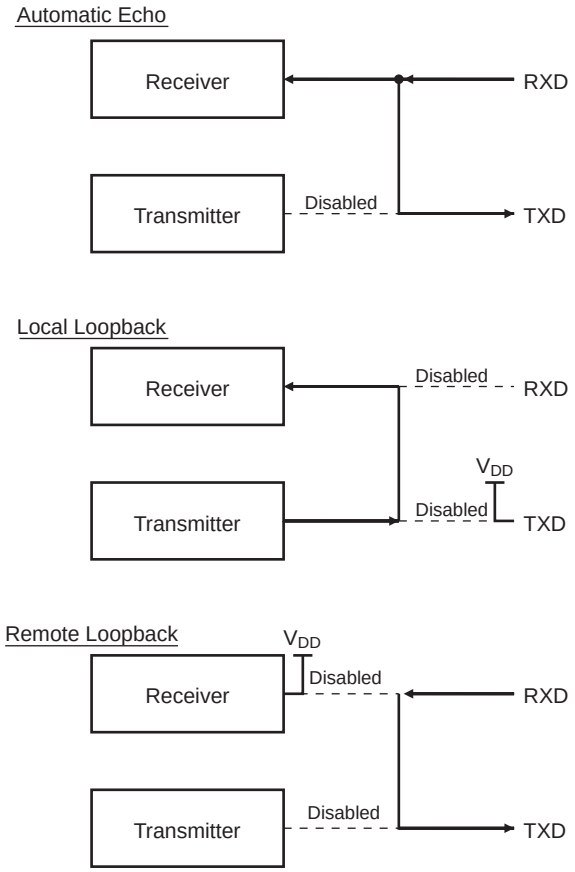
The UART supports three test modes. These modes of operation are programmed by using the CHMODE field in UART_MR.

The Automatic Echo mode allows a bit-by-bit retransmission. When a bit is received on the URXD line, it is sent to the UTXD line. The transmitter operates normally, but has no effect on the UTXD line.

The Local Loopback mode allows the transmitted characters to be received. UTXD and URXD pins are not used and the output of the transmitter is internally connected to the input of the receiver. The URXD pin level has no effect and the UTXD line is held high, as in idle state.

The Remote Loopback mode directly connects the URXD pin to the UTXD line. The transmitter and the receiver are disabled and have no effect. This mode allows a bit-by-bit retransmission.

Figure 42-11. Test Modes



42.6 Universal Asynchronous Receiver Transmitter (UART) User Interface

Table 42-4. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	Control Register	UART_CR	Write-only	–
0x0004	Mode Register	UART_MR	Read/Write	0x0
0x0008	Interrupt Enable Register	UART_IER	Write-only	–
0x000C	Interrupt Disable Register	UART_IDR	Write-only	–
0x0010	Interrupt Mask Register	UART_IMR	Read-only	0x0
0x0014	Status Register	UART_SR	Read-only	–
0x0018	Receive Holding Register	UART_RHR	Read-only	0x0
0x001C	Transmit Holding Register	UART_THR	Write-only	–
0x0020	Baud Rate Generator Register	UART_BRGR	Read/Write	0x0
0x0024	Reserved	–	–	–
0x0028–0x003C	Reserved	–	–	–
0x0040–0x00E0	Reserved	–	–	–
0x00E4	Write Protection Mode Register	UART_WPMR	Read/Write	0x0
0x00E8	Reserved	–	–	–
0x00EC–0x00FC	Reserved	–	–	–

42.6.1 UART Control Register

Name: UART_CR

Address: 0xF8004000 (0), 0xFC004000 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

- **RSTRX: Reset Receiver**

0: No effect.

1: The receiver logic is reset and disabled. If a character is being received, the reception is aborted.

- **RSTTX: Reset Transmitter**

0: No effect.

1: The transmitter logic is reset and disabled. If a character is being transmitted, the transmission is aborted.

- **RXEN: Receiver Enable**

0: No effect.

1: The receiver is enabled if RXDIS is 0.

- **RXDIS: Receiver Disable**

0: No effect.

1: The receiver is disabled. If a character is being processed and RSTRX is not set, the character is completed before the receiver is stopped.

- **TXEN: Transmitter Enable**

0: No effect.

1: The transmitter is enabled if TXDIS is 0.

- **TXDIS: Transmitter Disable**

0: No effect.

1: The transmitter is disabled. If a character is being processed and a character has been written in the UART_THR and RSTTX is not set, both characters are completed before the transmitter is stopped.

- **RSTSTA: Reset Status**

0: No effect.

1: Resets the status bits PARE, FRAME and OVRE in the UART_SR.

42.6.2 UART Mode Register

Name: UART_MR

Address: 0xF8004004 (0), 0xFC004004 (1)

Access: Read/Write

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

- **PAR: Parity Type**

Value	Name	Description
0	EVEN	Even Parity
1	ODD	Odd Parity
2	SPACE	Space: parity forced to 0
3	MARK	Mark: parity forced to 1
4	NO	No parity

- **CHMODE: Channel Mode**

Value	Name	Description
0	NORMAL	Normal mode
1	AUTOMATIC	Automatic echo
2	LOCAL_LOOPBACK	Local loopback
3	REMOTE_LOOPBACK	Remote loopback

42.6.3 UART Interrupt Enable Register

Name: UART_IER

Address: 0xF8004008 (0), 0xFC004008 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

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

0: No effect.

1: Enables the corresponding interrupt.

- **RXRDY: Enable RXRDY Interrupt**
- **TXRDY: Enable TXRDY Interrupt**
- **OVRE: Enable Overrun Error Interrupt**
- **FRAME: Enable Framing Error Interrupt**
- **PARE: Enable Parity Error Interrupt**
- **TXEMPTY: Enable TXEMPTY Interrupt**

42.6.4 UART Interrupt Disable Register

Name: UART_IDR

Address: 0xF800400C (0), 0xFC00400C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

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

0: No effect.

1: Disables the corresponding interrupt.

- **RXRDY: Disable RXRDY Interrupt**
- **TXRDY: Disable TXRDY Interrupt**
- **OVRE: Disable Overrun Error Interrupt**
- **FRAME: Disable Framing Error Interrupt**
- **PARE: Disable Parity Error Interrupt**
- **TXEMPTY: Disable TXEMPTY Interrupt**

42.6.5 UART Interrupt Mask Register

Name: UART_IMR

Address: 0xF8004010 (0), 0xFC004010 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

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.

- **RXRDY: Mask RXRDY Interrupt**
- **TXRDY: Disable TXRDY Interrupt**
- **OVRE: Mask Overrun Error Interrupt**
- **FRAME: Mask Framing Error Interrupt**
- **PARE: Mask Parity Error Interrupt**
- **TXEMPTY: Mask TXEMPTY Interrupt**

42.6.6 UART Status Register

Name: UART_SR

Address: 0xF8004014 (0), 0xFC004014 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXEMPTY	–
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

- **RXRDY: Receiver Ready**

0: No character has been received since the last read of the UART_RHR, or the receiver is disabled.

1: At least one complete character has been received, transferred to UART_RHR and not yet read.

- **TXRDY: Transmitter Ready**

0: A character has been written to UART_THR and not yet transferred to the internal shift register, or the transmitter is disabled.

1: There is no character written to UART_THR not yet transferred to the internal shift register.

- **OVRE: Overrun Error**

0: No overrun error has occurred since the last RSTSTA.

1: At least one overrun error has occurred since the last RSTSTA.

- **FRAME: Framing Error**

0: No framing error has occurred since the last RSTSTA.

1: At least one framing error has occurred since the last RSTSTA.

- **PARE: Parity Error**

0: No parity error has occurred since the last RSTSTA.

1: At least one parity error has occurred since the last RSTSTA.

- **TXEMPTY: Transmitter Empty**

0: There are characters in UART_THR, or characters being processed by the transmitter, or the transmitter is disabled.

1: There are no characters in UART_THR and there are no characters being processed by the transmitter.

42.6.7 UART Receiver Holding Register

Name: UART_RHR

Address: 0xF8004018 (0), 0xFC004018 (1)

Access: Read-only

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
RXCHR							

- **RXCHR: Received Character**

Last received character if RXRDY is set.

42.6.8 UART Transmit Holding Register

Name: UART_THR

Address: 0xF800401C (0), 0xFC00401C (1)

Access: Write-only

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
TXCHR							

- **TXCHR: Character to be Transmitted**

Next character to be transmitted after the current character if TXRDY is not set.

42.6.9 UART Baud Rate Generator Register

Name: UART_BRGR

Address: 0xF8004020 (0), 0xFC004020 (1)

Access: Read/Write

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

- **CD: Clock Divisor**

0: Baud rate clock is disabled

1 to 65,535:

$$CD = \frac{f_{peripheral\ clock}}{16 \times Baud\ Rate}$$

42.6.10 UART Write Protection Mode Register

Name: UART_WPMR

Address: 0xF80040E4 (0), 0xFC0040E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII).

Refer to Section 42.5.5 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x554152	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

43. Universal Synchronous Asynchronous Receiver Transceiver (USART)

43.1 Description

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, and SPI buses, 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.

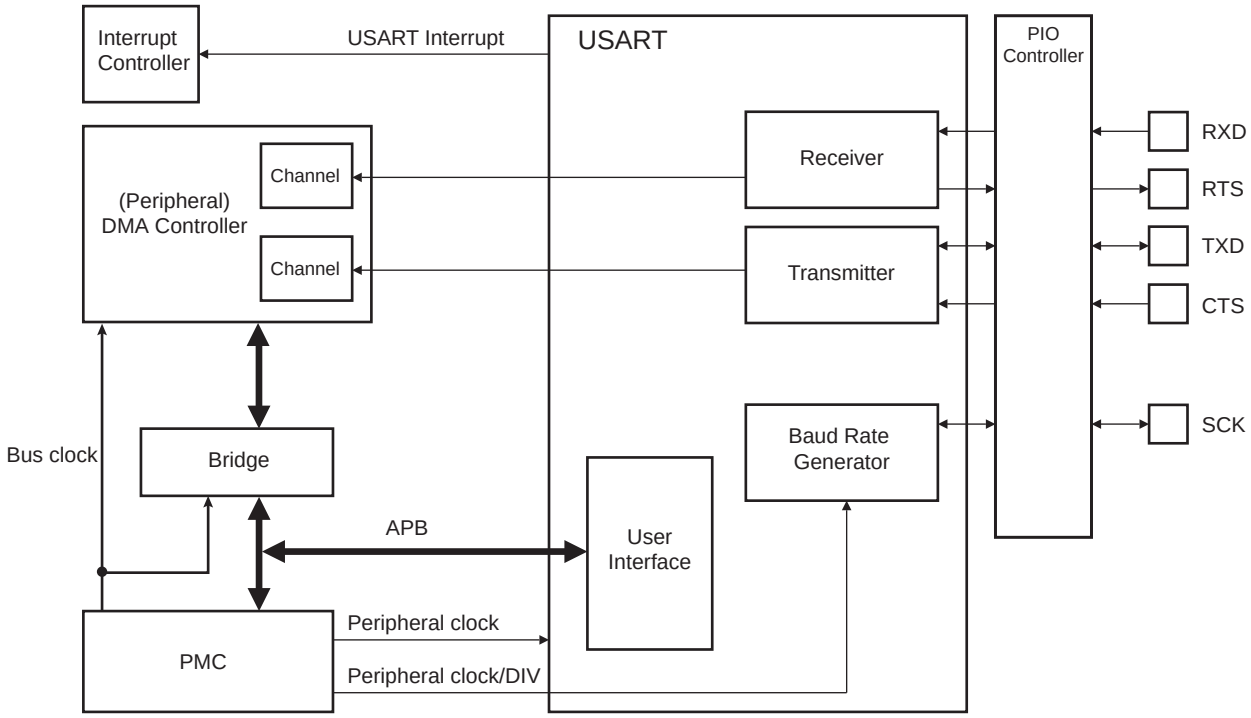
The USART supports the connection to the DMA Controller, which enables data transfers to the transmitter and from the receiver. The DMAC provides chained buffer management without any intervention of the processor.

43.2 Embedded Characteristics

- Programmable Baud Rate Generator
- 5- 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
- SPI Mode
  - Master or Slave
  - Serial Clock Programmable Phase and Polarity
  - SPI Serial Clock (SCK) Frequency up to $f_{peripheral\ clock}/6$
- Test Modes
  - Remote Loopback, Local Loopback, Automatic Echo
- Supports Connection of:
  - Two DMA Controller Channels (DMAC)
- Offers Buffer Transfer without Processor Intervention
- Register Write Protection

43.3 Block Diagram

Figure 43-1. USART Block Diagram



43.4 I/O Lines Description

Table 43-1. I/O Line Description

Name	Description	Type	Active Level
SCK	Serial Clock	I/O	—
TXD	Transmit Serial Data or Master Out Slave In (MOSI) in SPI Master mode or Master In Slave Out (MISO) in SPI Slave mode	I/O	—
RXD	Receive Serial Data or Master In Slave Out (MISO) in SPI Master mode or Master Out Slave In (MOSI) in SPI Slave mode	Input	—
CTS	Clear to Send or Slave Select (NSS) in SPI Slave mode	Input	Low
RTS	Request to Send or Slave Select (NSS) in SPI Master mode	Output	Low

43.5 Product Dependencies

43.5.1 I/O Lines

The pins used for interfacing the USART may be multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the desired USART pins to their peripheral function. If I/O lines of the USART are not used by the application, they can be used for other purposes by the PIO Controller.

Table 43-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
USART0	CTS0	PD10	A
USART0	RTS0	PD11	A
USART0	RXD0	PD12	A
USART0	SCK0	PD28	A
USART0	TXD0	PD13	A
USART1	CTS1	PD14	A
USART1	RTS1	PD15	A
USART1	RXD1	PD16	A
USART1	SCK1	PD29	A
USART1	TXD1	PD17	A
USART2	CTS2	PB3	B
USART2	RTS2	PB11	B
USART2	RXD2	PB4	B
USART2	SCK2	PB1	B
USART2	TXD2	PB5	B
USART3	CTS3	PE5	B
USART3	RTS3	PE24	B
USART3	RXD3	PE16	B
USART3	SCK3	PE15	B
USART3	TXD3	PE17	B
USART4	CTS4	PE0	C
USART4	RTS4	PE28	B
USART4	RXD4	PE26	B
USART4	SCK4	PE25	B
USART4	TXD4	PE27	B

43.5.2 Power Management

The USART is not continuously clocked. The programmer must first enable the USART clock in the Power Management Controller (PMC) before using the USART. However, if the application does not require USART operations, the USART clock can be stopped when not needed and be restarted later. In this case, the USART will resume its operations where it left off.

43.5.3 Interrupt Sources

The USART interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the USART interrupt requires the Interrupt Controller to be programmed first.

Table 43-3. Peripheral IDs

Instance	ID
USART0	6
USART1	7
USART2	29
USART3	30
USART4	31

43.6 Functional Description

43.6.1 Baud Rate Generator

The baud rate generator provides the bit period clock, also named the baud rate clock, to both the receiver and the transmitter.

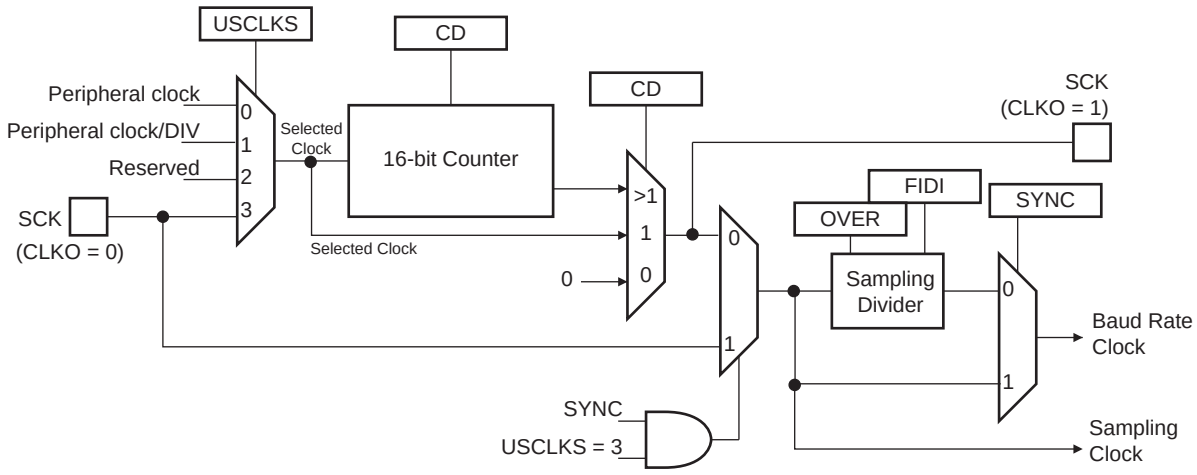
The baud rate generator clock source is selected by configuring the USCLKS field in the USART Mode register (US_MR) to one of the following:

- The peripheral clock
- A division of the peripheral clock, where the divider is product-dependent, but generally set to 8
- The external clock, available on the SCK pin

The baud rate generator is based upon a 16-bit divider, which is configured using the CD field of the Baud Rate Generator register (US_BRGR). If CD is configured to ‘0’, the baud rate generator does not generate any clocks. If CD is configured to ‘1’, 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 3 times lower than the frequency provided on the peripheral clock in USART mode (field USART_MODE differs from 0xE or 0xF), or 6 times lower in SPI mode (field USART_MODE equals 0xE or 0xF).

Figure 43-2. Baud Rate Generator



43.6.1.1 Baud Rate in Asynchronous Mode

If the USART is programmed to operate in Asynchronous mode, the selected clock is first divided by the value of US_BRGR.CD. The resulting clock is provided to the receiver as a sampling clock and then divided by 16 or 8, depending on the value of US_MR.OVER.

If OVER is set to ‘1’, the receiver sampling is eight times higher than the baud rate clock. If OVER is set to ‘0’, the sampling is performed at 16 times the baud rate clock.

The baud rate is calculated as per the following formula:

$$\text{Baud Rate} = \frac{\text{Selected Clock}}{(8(2-OVER)CD)}$$

This gives a maximum baud rate of peripheral clock divided by 8, assuming that the peripheral clock is the highest possible clock and that the OVER is written to ‘1’.

Baud Rate Calculation Example

Table 43-4 shows calculations of CD to obtain a baud rate at 38,400 bit/s for different source clock frequencies. This table also shows the actual resulting baud rate and the error.

Table 43-4. 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%
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%

In this example, the baud rate is calculated with the following formula:

$$\text{Baud Rate} = \text{Selected Clock} / CD \times 16$$

The baud rate error is calculated with the following formula. It is not recommended to work with an error higher than 5%.

$$Error = 1 - \left(\frac{\text{Expected Baud Rate}}{\text{Actual Baud Rate}}\right)$$

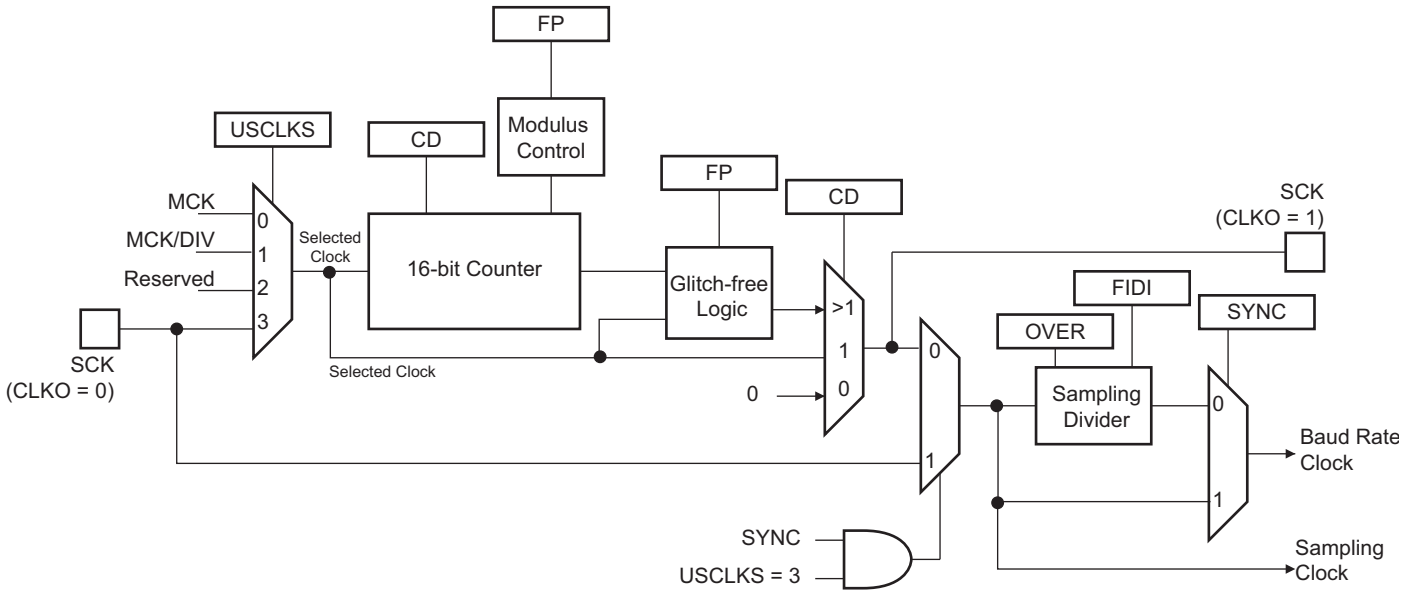
43.6.1.2 Fractional Baud Rate in Asynchronous Mode

The baud rate generator is subject to the following limitation: the output frequency changes only by 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 using US_BRGR.FP. 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:

$$\text{Baud Rate} = \frac{\text{Selected Clock}}{\left(8(2 - OVER)\left(CD + \frac{FP}{8}\right)\right)}$$

The modified architecture is presented in the following Figure 43-3.

Figure 43-3. Fractional Baud Rate Generator



Warning: When the value of US_BRGR.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 USART_BRGR.CD.

43.6.1.3 Baud Rate in Synchronous Mode or SPI Mode

If the USART is programmed to operate in Synchronous mode, the selected clock is divided by the value of US_BRGR.CD.

$$\text{Baud Rate} = \frac{\text{Selected Clock}}{CD}$$

In Synchronous mode, if the external clock is selected (USCLKS = 3), the clock is provided directly by the signal on the USART SCK pin. No division is active. The value written in US_BRGR has no effect. The external clock frequency must be at least 3 times lower than the system clock. In Master mode, Synchronous mode (USCLKS =

0 or 1, CLKO set to 1), the receive part limits the SCK maximum frequency to Selected Clock/3 in USART mode, or Selected Clock/6 in SPI mode.

When either the external clock SCK or the internal clock divided (peripheral clock/DIV) is selected, the value of CD must be even if the user has to ensure a 50:50 mark/space ratio on the SCK pin. When the peripheral clock is selected, the baud rate generator ensures a 50:50 duty cycle on the SCK pin, even if the value of CD is odd.

43.6.1.4 Baud Rate in ISO 7816 Mode

The ISO7816 specification defines the bit rate with the following formula:

$$B = \frac{Di}{Fi} \times f$$

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 Table 43-5.

Table 43-5. 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 Table 43-6.

Table 43-6. 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

Table 43-7 shows the resulting Fi/Di ratio, which is the ratio between the ISO7816 clock and the baud rate clock.

Table 43-7. 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 US_MR.USCLKS is first divided by the value programmed in US_BRGR.CD. The resulting clock can be provided to the SCK pin to feed the smart card clock inputs. This means that the US_MR.CLKO bit can be written to ‘1’.

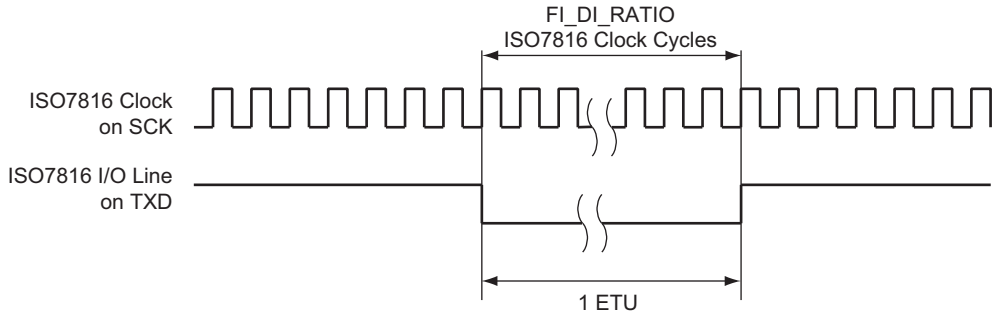
This clock is then divided by the value programmed in the FI_DI_RATIO field in the FI DI Ratio register (US_FIDI). This is performed by the Sampling Divider, which performs a division by up to 65535 in ISO7816 mode. The

noninteger values of the Fi/Di ratio are not supported and the user must program FI_DI_RATIO to a value as close as possible to the expected value.

FI_DI_RATIO 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).

Figure 43-4 shows the relation between the Elementary Time Unit, corresponding to a bit time, and the ISO 7816 clock.

Figure 43-4. Elementary Time Unit (ETU)



43.6.2 Receiver and Transmitter Control

After reset, the receiver is disabled. The user must enable the receiver by setting the RXEN bit in the Control register (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 writing a ‘1’ to US_CR.TXEN. 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 writing a ‘1’ to the corresponding bit US_CR.RSTRX and US_CR.RSTTX respectively. 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 writing a ‘1’ to US_CR.RXDIS and US_CR.TXDIS, respectively. 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 Transmit Holding register (US_THR). If a timeguard is programmed, it is handled normally.

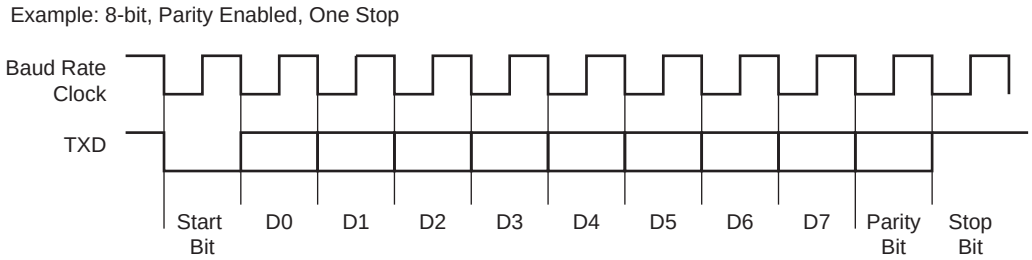
43.6.3 Synchronous and Asynchronous Modes

43.6.3.1 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, one optional parity bit and up to two 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 configured in the US_MR.CHRL and the US_MR.MODE9. Nine bits are selected by writing a ‘1’ to US_MR.MODE9 regardless of the CHRL field. The parity is selected by US_MR.PAR. Even, odd, space, marked or none parity bit can be configured. US_MR.MSBF 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 US_MR.NBSTOP. The 1.5 stop bit is supported in Asynchronous mode only.

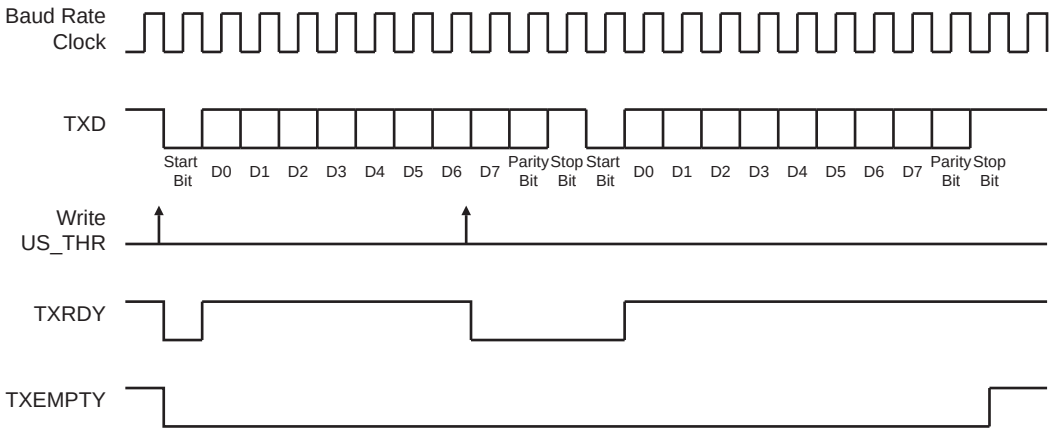
Figure 43-5. Character Transmit



The characters are sent by writing in US_THR. The transmitter reports two status bits in the Channel Status register (US_CSR): TXRDY (Transmitter Ready), which indicates that US_THR is empty, and TXEMPTY, which indicates that all the characters written in US_THR have been processed. When the current character processing is completed, the last character written in US_THR is transferred into the Shift register of the transmitter and US_THR becomes empty, thus TXRDY rises.

Both TXRDY and TXEMPTY are low when the transmitter is disabled. Writing a character in US_THR while TXRDY is low has no effect and the written character is lost.

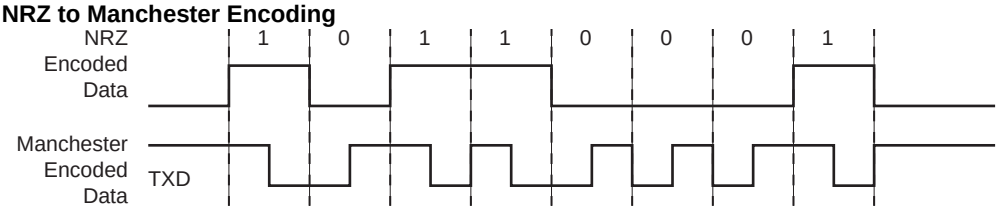
Figure 43-6. Transmitter Status



43.6.3.2 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, write a '1' to USART_MR.MAN. 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 a 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. Figure 43-7 illustrates this coding scheme.

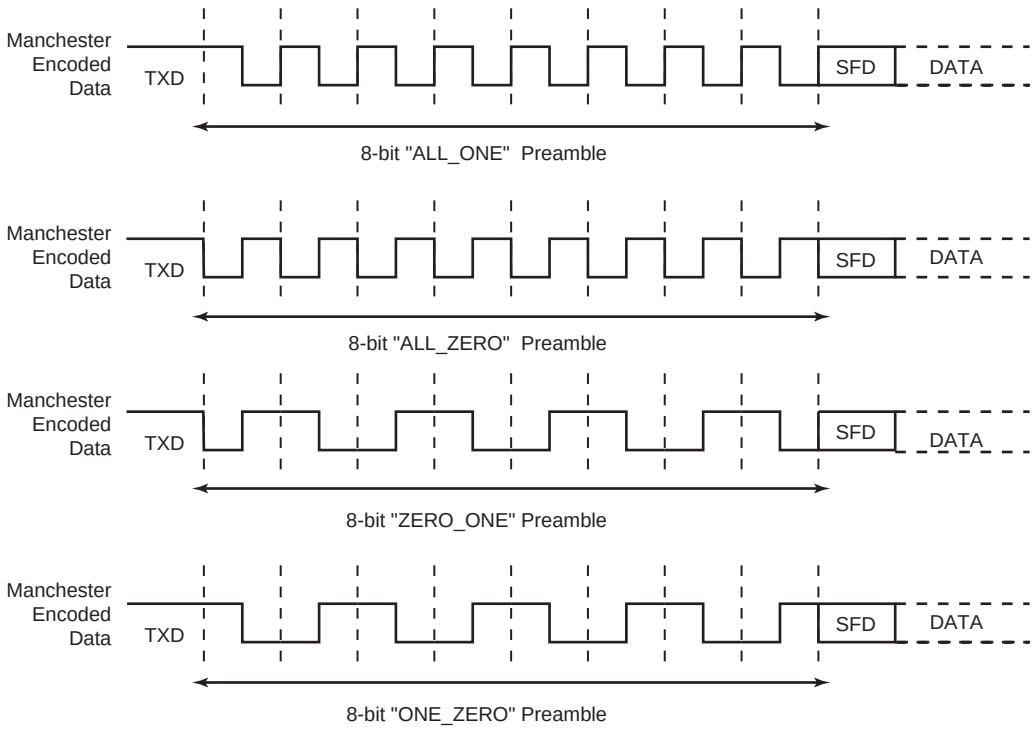
Figure 43-7. NRZ to Manchester Encoding



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

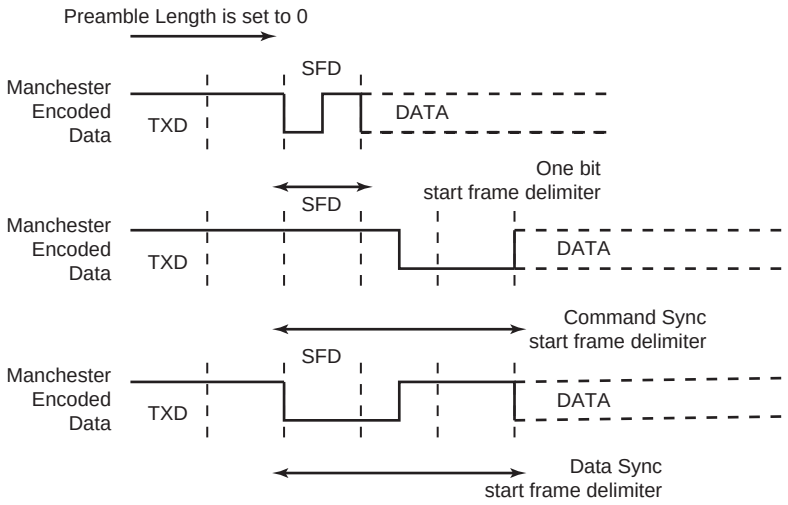
predefined pattern with a programmable length from 1 to 15 bit times. If the preamble length is set to 0, the preamble waveform is not generated prior to any character. The preamble pattern is chosen among the following sequences: ALL_ONE, ALL_ZERO, ONE_ZERO or ZERO_ONE by configuring US_MAN.TX_PP. US_MAN.TX_PL is used to configure the preamble length. Figure 43-8 illustrates and defines the valid patterns. To improve flexibility, the encoding scheme can be configured using US_MAN.TX_MPOL. If TX_MPOL is set to ‘0’ (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 TX_MPOL is set to ‘1’, a logic one is encoded with a one-to-zero transition and a logic zero is encoded with a zero-to-one transition.

Figure 43-8. Preamble Patterns, Default Polarity Assumed



A start frame delimiter is configured using US_MR.ONEBIT. It consists of a user-defined pattern that indicates the beginning of a valid data. Figure 43-9 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 US_MR.MODSYNC is written to ‘1’, the next character is a command. If it is written to ‘0’, the next character is a data. When direct memory access is used, MODSYNC can be immediately updated with a modified character located in memory. To enable this mode, US_MR.VAR_SYNC must be written to ‘1’. In this case, MODSYNC is bypassed and the sync configuration is held in US_THR.TXSYNH. The USART character format is modified and includes sync information.

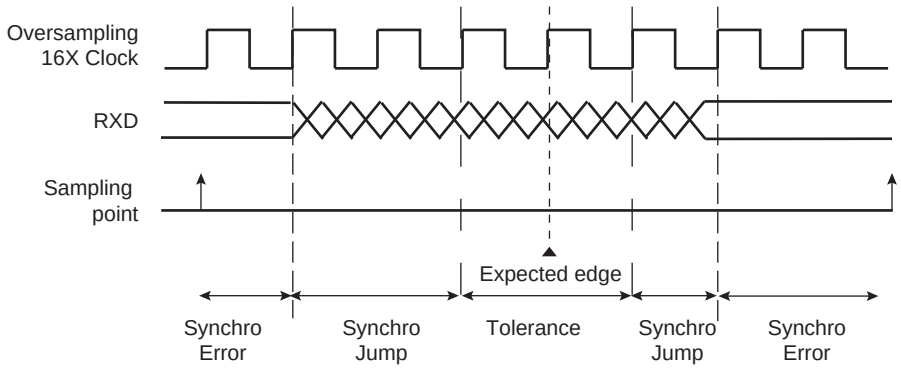
Figure 43-9. Start Frame Delimiter



Drift Compensation

Drift compensation is available only in 16X Oversampling mode. A hardware recovery system allows a larger clock drift. To enable the hardware system, USART_MAN.DRIFT must be written to ‘1’. If the RXD edge is one 16X clock cycle from the expected edge, this is considered as normal jitter and no corrective action 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 43-10. Bit Resynchronization



43.6.3.3 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 value of US_MR.OVER.

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 field NBSTOP, 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.

Figure 43-11 and Figure 43-12 illustrate start detection and character reception when USART operates in Asynchronous mode.

Figure 43-11. Asynchronous Start Detection

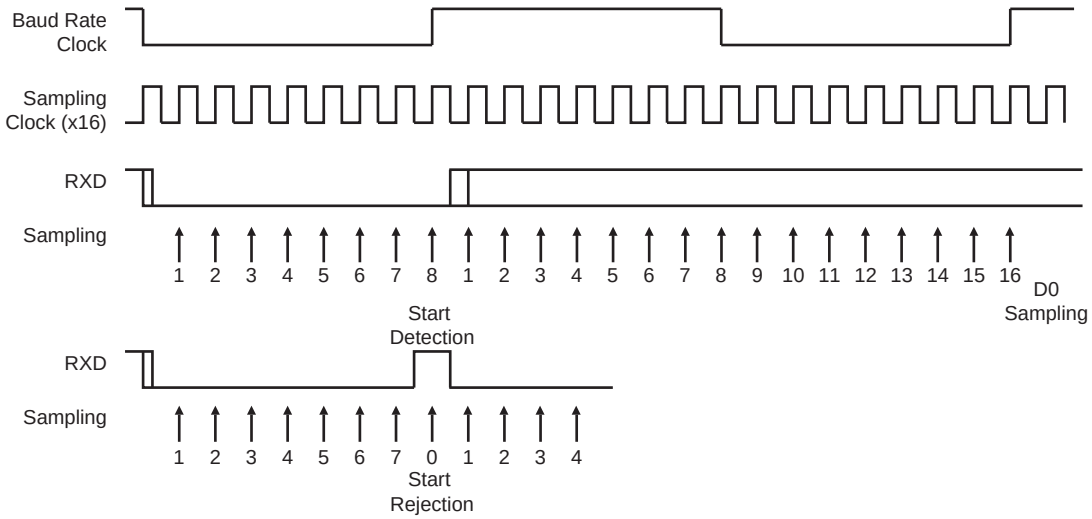
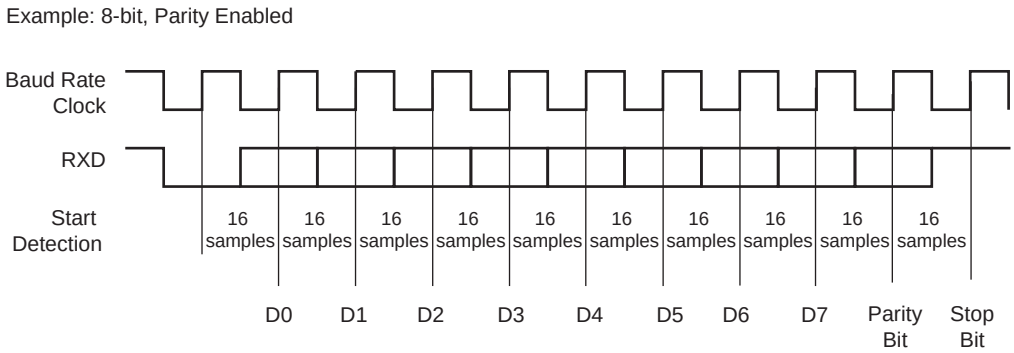


Figure 43-12. Asynchronous Character Reception



43.6.3.4 Manchester Decoder

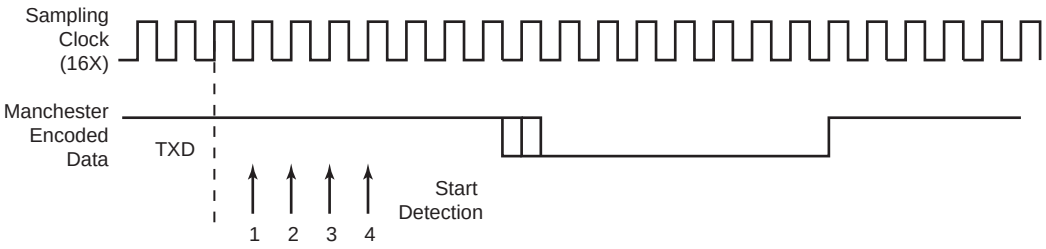
When US_MR.MAN is ‘1’, the Manchester decoder is enabled. The decoder performs both preamble and start frame delimiter detection. One input line is dedicated to Manchester-encoded input data.

An optional preamble sequence can be defined, and its length is user-defined and totally independent of the transmitter side. The length of the preamble sequence is configured using US_MAN.RX_PL. If RX_PL is ‘0’, no preamble is detected and the function is disabled. The polarity of the input stream is configured with US_MAN.RX_MPOL. Depending on the desired application, the preamble pattern matching is to be defined via the US_MAN. Refer to Figure 43-8 for available preamble patterns.

Unlike preamble, the start frame delimiter is shared between Manchester Encoder and Decoder. If US_MR.ONEBIT is written to ‘1’, only a zero-encoded Manchester can be detected as a valid start frame delimiter.

If US_MR.ONEBIT is written to ‘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. Refer to Figure 43-13. The sample pulse rejection mechanism applies.

Figure 43-13. 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 the USART for processing. Figure 43-14 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 US_CSR.MANERR flag is raised. It is cleared by writing a ‘1’ to US_CR.RSTSTA. Refer to Figure 43-15 for an example of Manchester error detection during data phase.

Figure 43-14. Preamble Pattern Mismatch

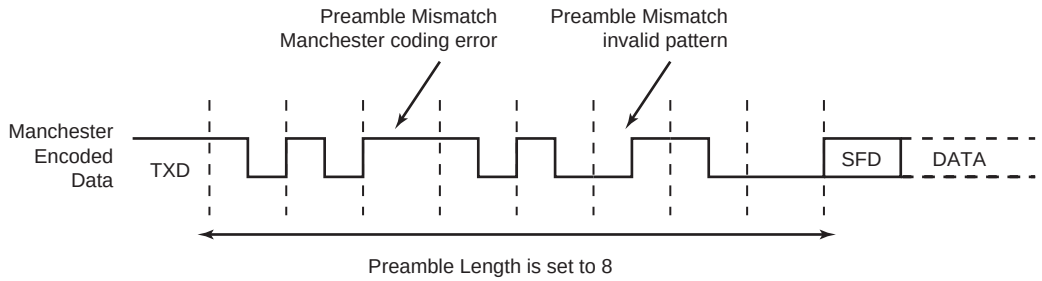
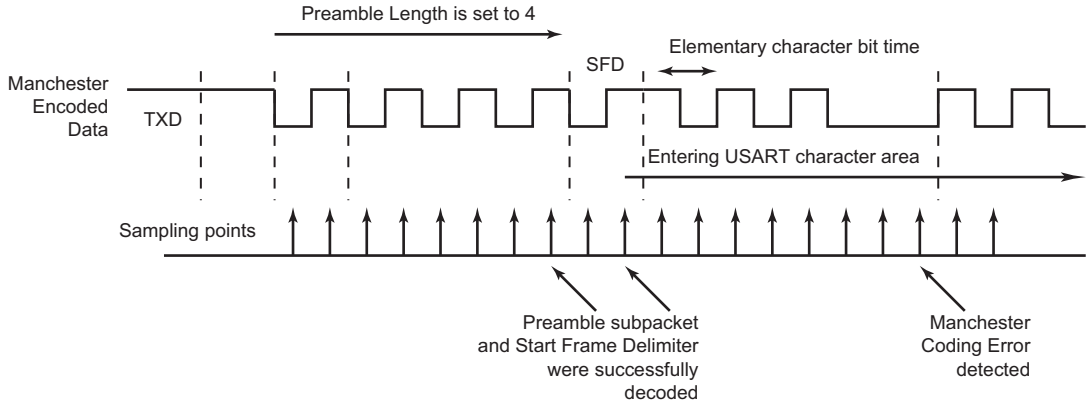


Figure 43-15. Manchester Error Flag



When the start frame delimiter is a sync pattern (US_MR.ONEBIT = 0), both command and data delimiter are supported. If a valid sync is detected, the received character is written in RXCHR in the Receive Holding register (US_RHR) and RXSYNH is updated. RXSYNH is set to ‘1’ when the received character is a command, and to ‘0’ if the received character is a data. This alleviates and simplifies the direct memory access as the character contains its own sync field in the same register.

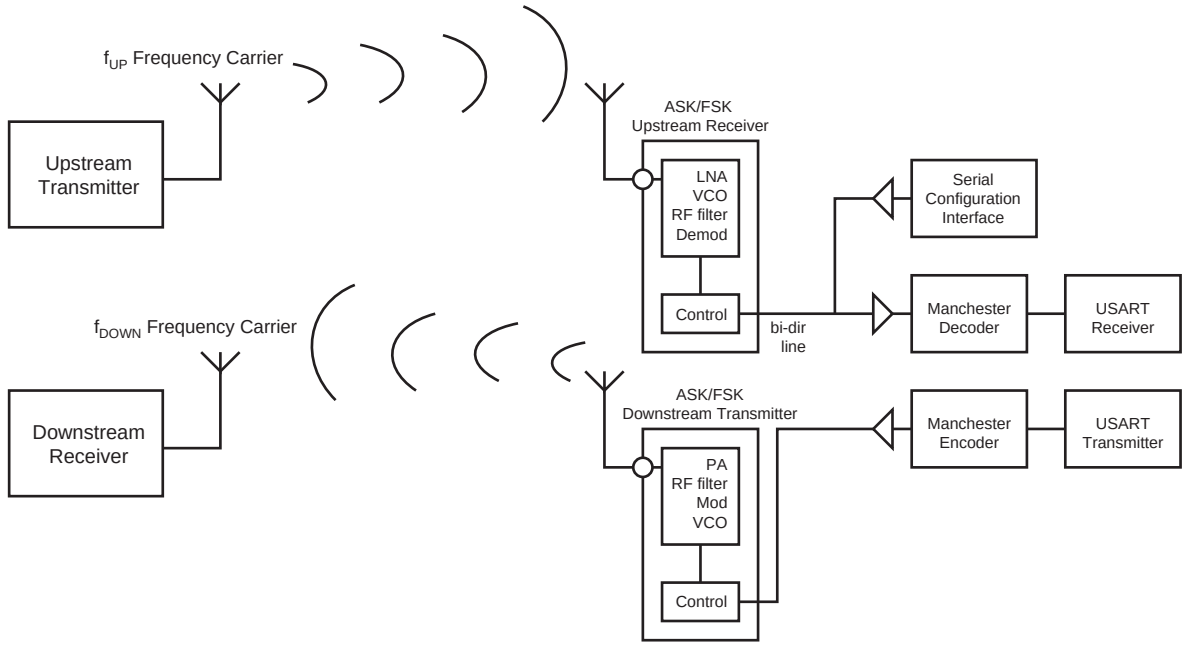
As the decoder is setup to be used in Unipolar mode, the first bit of the frame has to be a zero-to-one transition.

43.6.3.5 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. Refer to the configuration in Figure 43-16.

Figure 43-16. 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. Refer to Figure 43-17 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 one is sent, the modulator outputs an RF signal at frequency F0 and switches to F1 if the data sent is a zero. Refer to Figure 43-18.

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 43-17. ASK Modulator Output

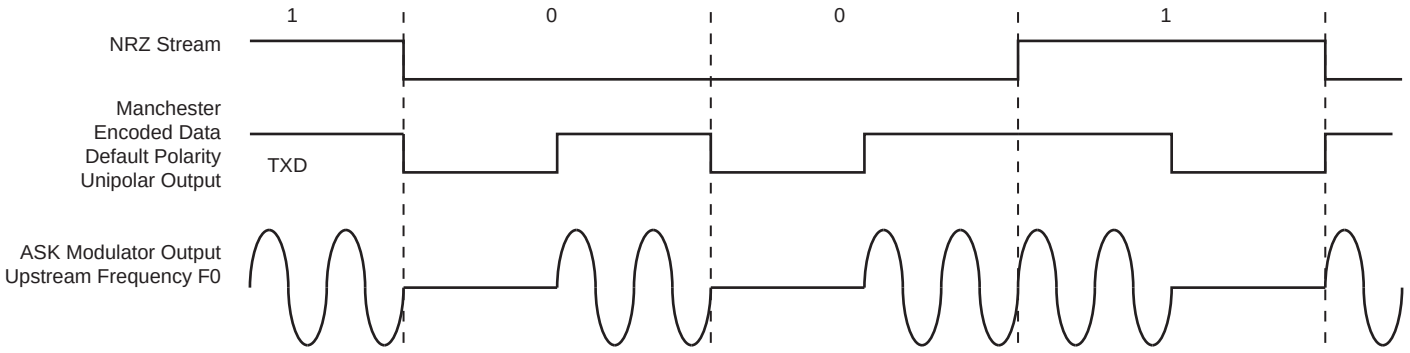
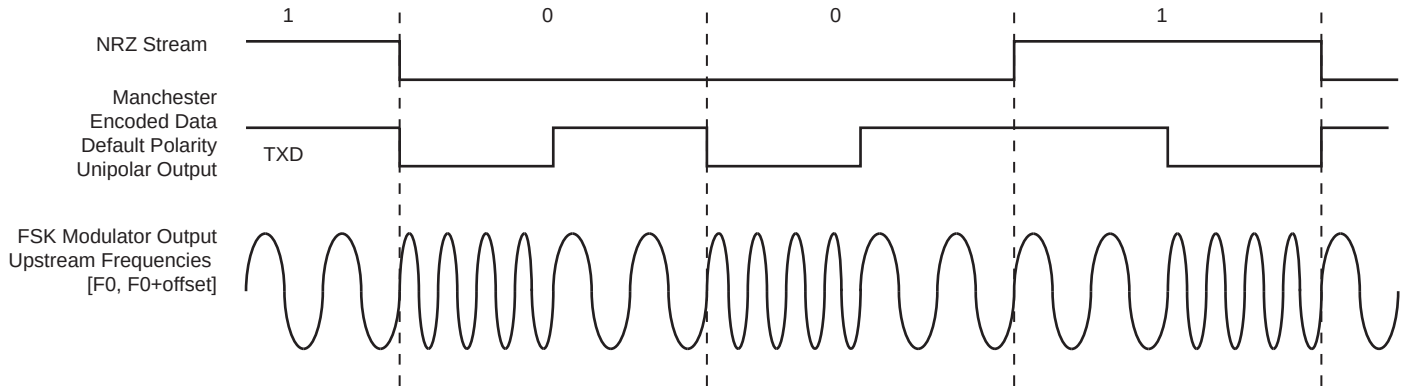


Figure 43-18. FSK Modulator Output



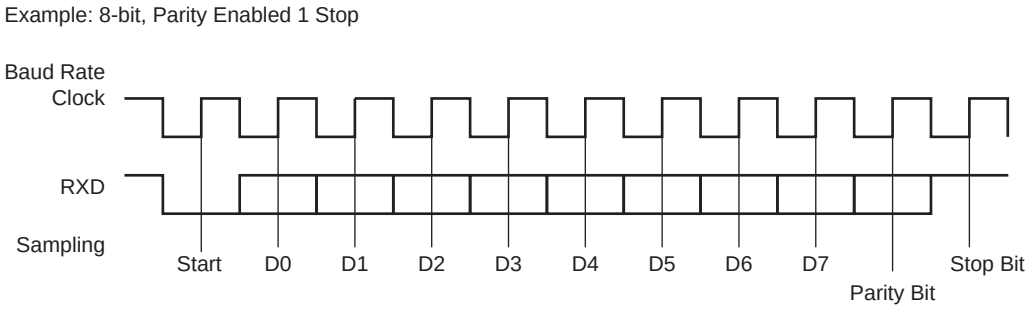
43.6.3.6 Synchronous Receiver

In Synchronous mode (US_MR.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.

Figure 43-19 illustrates a character reception in Synchronous mode.

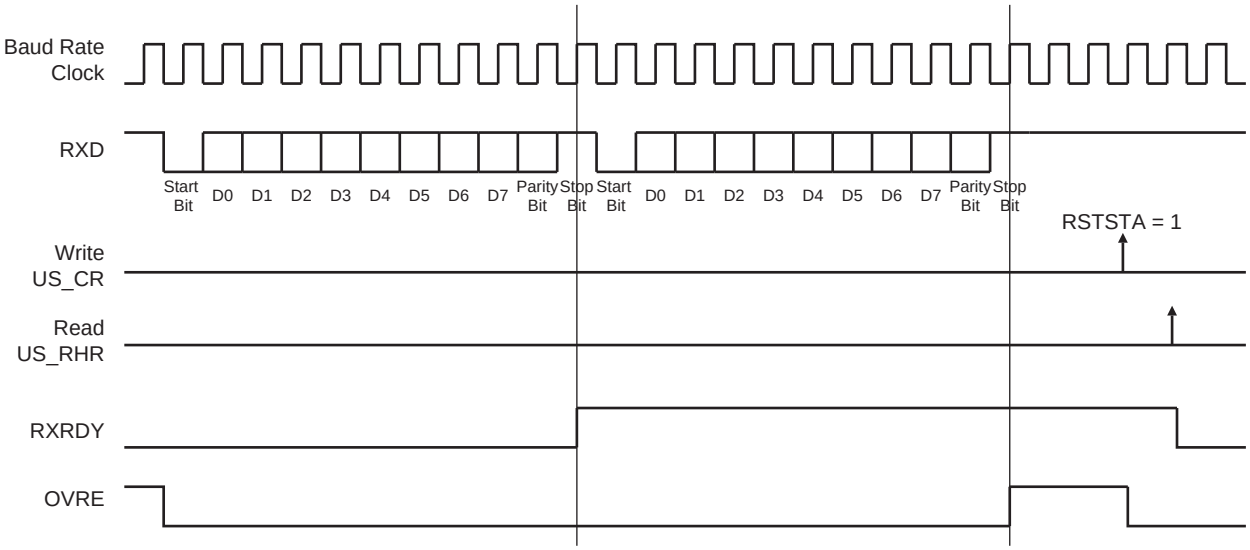
Figure 43-19. Synchronous Mode Character Reception



43.6.3.7 Receiver Operations

When a character reception is completed, it is transferred to the Receive Holding register (US_RHR) and US_CSR.RXRDY rises. If a character is completed while RXRDY is set, the OVRE (Overrun Error) bit is set. The last character is transferred into US_RHR and overwrites the previous one. The OVRE bit is cleared by writing a '1' to US_CR.RSTSTA.

Figure 43-20. Receiver Status



43.6.3.8 Parity

The USART supports five Parity modes. The PAR field also enables Multidrop mode (refer to Section 43.6.3.9 "Multidrop Mode"). Even and odd parity bit generation and error detection are supported. The configuration is done in US_MR.PAR.

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.

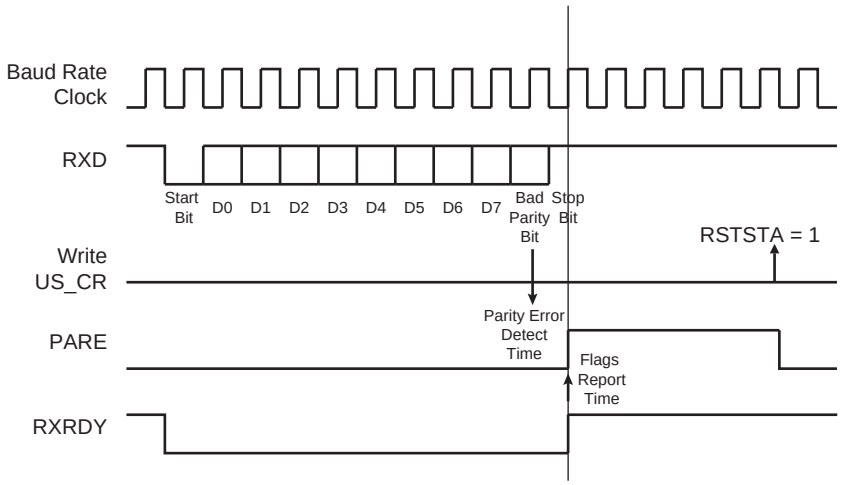
Table 43-8 shows an example of the parity bit for the character 0x41 (character ASCII “A”) depending on the configuration of the USART. Because there are two bits set to 1 in the character value, the parity bit is set to ‘1’ when the parity is odd, or configured to ‘0’ when the parity is even.

Table 43-8. 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 US_CSR.PARE (Parity Error). PARE can be cleared by writing a ‘1’ to the RSTSTA bit the US_CR. Figure 43-21 illustrates the parity bit status setting and clearing.

Figure 43-21. Parity Error



43.6.3.9 Multidrop Mode

If the value 0x6 or 0x07 is written to US_MR.PAR, the USART runs in Multidrop mode. This mode differentiates the data characters and the address characters. Data is transmitted with the parity bit at 0 and addresses are transmitted with the parity bit at 1.

If the USART is configured in Multidrop mode, the receiver sets PARE when the parity bit is high and the transmitter is able to send a character with the parity bit high when a ‘1’ is written to US_CR.SENTA.

To handle parity error, PARE is cleared when a ‘1’ is written to US_CR.RSTSTA.

The transmitter sends an address byte (parity bit set) when US_CR.SENDA = 1. In this case, the next byte written to US_THR is transmitted as an address. Any character written in the US_THR without having written SENDA is transmitted normally with the parity at 0.

43.6.3.10 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 acts as a long stop bit.

The duration of the idle state is programmed in the TG field of the Transmitter Timeguard register (US_TTGR). When this field is written to ‘0’, 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 Figure 43-22, the behavior of 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 at 0 during the timeguard transmission if a character has been written in US_THR. TXEMPTY remains low until the timeguard transmission is completed as the timeguard is part of the current character being transmitted.

Figure 43-22. Timeguard Operations

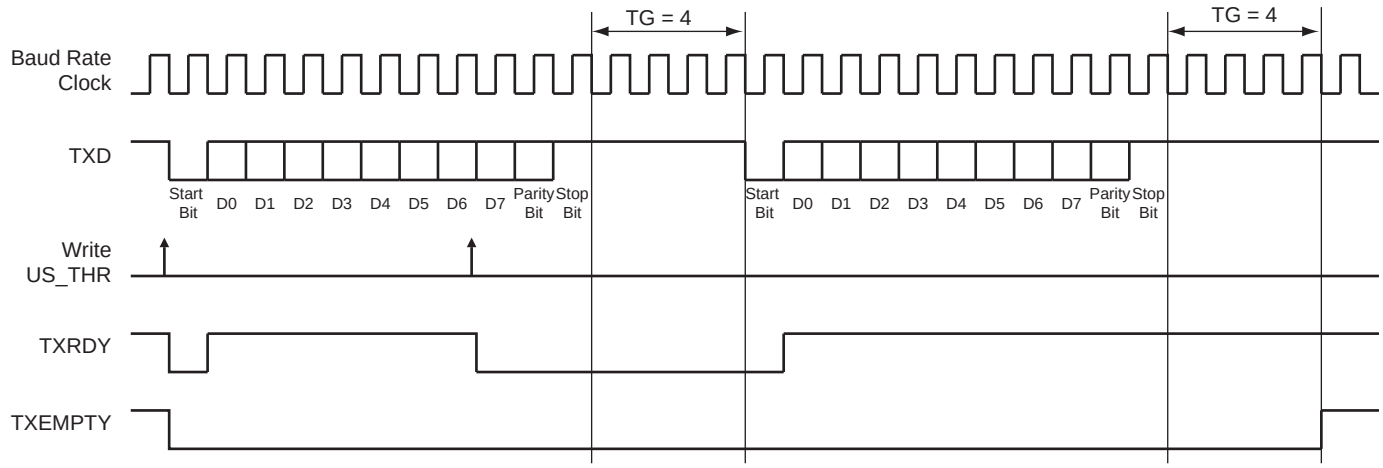


Table 43-9 indicates the maximum length of a timeguard period that the transmitter can handle depending on the baud rate.

Table 43-9. 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

43.6.3.11Receiver 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, US_CSR.TIMEOUT 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 (US_RTOR). If TO is written to ‘0’, the Receiver Timeout is disabled and no timeout is detected. US_CSR.TIMEOUT remains at ‘0’. Otherwise, the receiver loads a 16-bit counter with the value programmed in US_RTOR.TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, TIMEOUT rises. Then, the user can either:

- Stop the counter clock until a new character is received. This is performed by writing a ‘1’ to US_CR.STTTO. In this case, the idle state on RXD before a new character is received will not provide a timeout. This prevents having to handle an interrupt before a character is received and allows 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 the RETTO (Reload and Start Timeout) bit in the US_CR. 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.

Figure 43-23 shows the block diagram of the Receiver Timeout feature.

Figure 43-23. Receiver Timeout Block Diagram

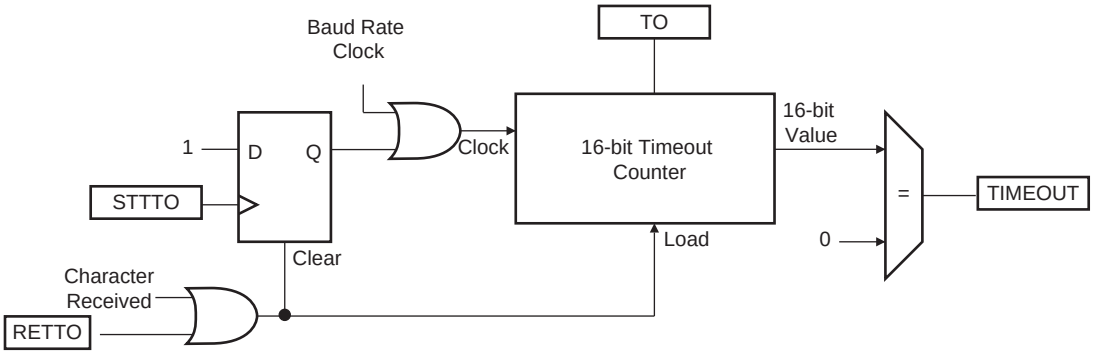


Table 43-10 gives the maximum timeout period for some standard baud rates.

Table 43-10. Maximum Timeout Period

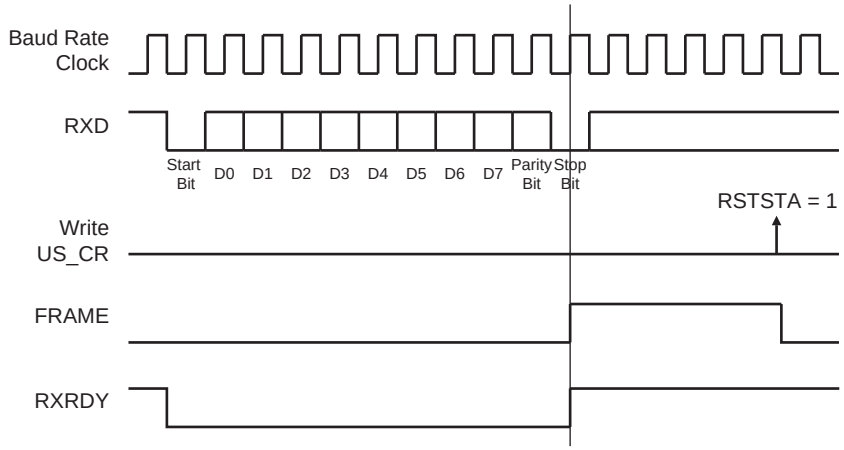
Baud Rate (bit/s)	Bit Time (µs)	Timeout (ms)
600	1,667	109,225
1,200	833	54,613
2,400	417	27,306
4,800	208	13,653
9,600	104	6,827
14,400	69	4,551
19,200	52	3,413
28,800	35	2,276
38,400	26	1,704
56,000	18	1,170
57,600	17	1,138
200,000	5	328

43.6.3.12Framing 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 in US_CSR.FRAME. FRAME is asserted in the middle of the stop bit as soon as the framing error is detected. It is cleared by writing a ‘1’ to US_CR.RSTSTA.

Figure 43-24. Framing Error Status



43.6.3.13Transmit 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 at 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 writing a ‘1’ to US_CR.STTBRK. This can be performed at any time, either while the transmitter is empty (no character in either the Shift register or in 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 STTBRK command is requested, further STTBRK commands are ignored until the end of the break is completed.

The break condition is removed by writing a ‘1’ to US_CR.STPBRK. If the STPBRK 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 STTBRK and STPBRK commands are processed only if US_CSR. TXRDY = 1 and the start of the break condition clears the TXRDY and TXEMPTY bits as if a character is processed.

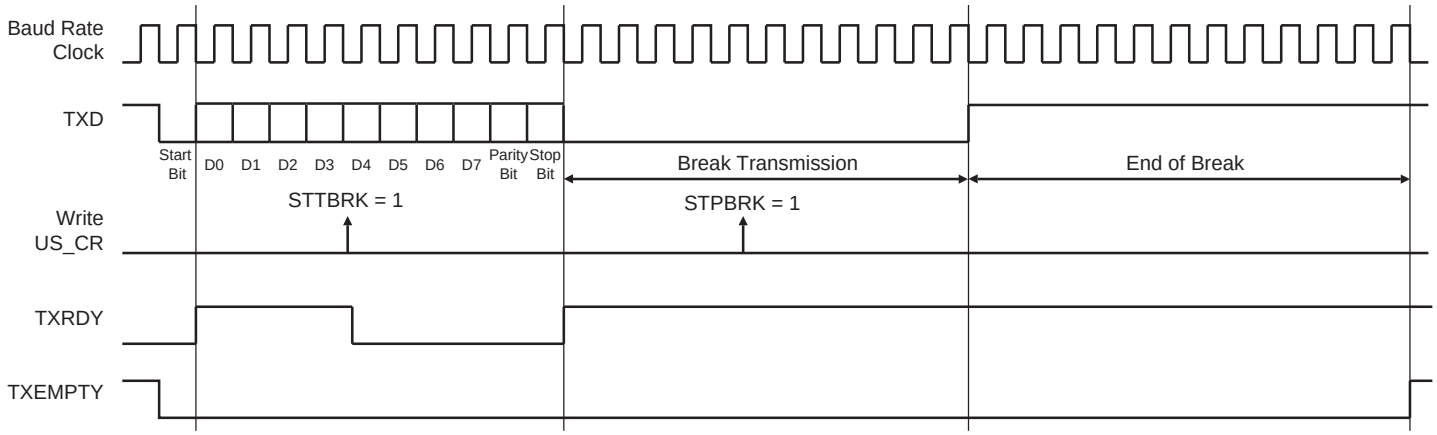
Writing US_CR with both STTBRK and STPBRK bits to ‘1’ can lead to an unpredictable result. All STPBRK commands requested without a previous STTBRK command are ignored. A byte written into US_THR while a break is pending, but not started, is ignored.

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.

Figure 43-25 illustrates the effect of both the Start Break (STTBRK) and Stop Break (STPBRK) commands on the TXD line.

Figure 43-25. Break Transmission



43.6.3.14 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.

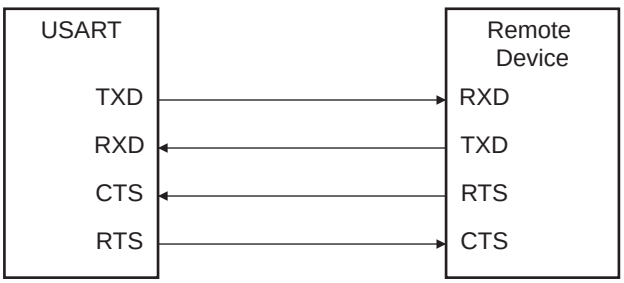
When the low stop bit is detected, the receiver asserts US_CSR.RXBRK. This bit may be cleared by writing a '1' to US_CR.RSTSTA.

An end of receive break is detected by a high level for at least 2/16 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 US_CSR.RXBRK bit.

43.6.3.15 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 Figure 43-26.

Figure 43-26. Connection with a Remote Device for Hardware Handshaking



Setting the USART to operate with hardware handshaking is performed by writing the value 0x2 to US_MR.USART_MODE.

When hardware handshaking is enabled, the USART displays similar behavior as in standard Synchronous or Asynchronous modes, with the difference that the receiver drives the RTS pin and the level on the CTS pin modifies the behavior of the transmitter, as shown in the figures below. The transmitter can handle hardware handshaking in any case.

Figure 43-27. RTS Line Software Control when US_MR.USART_MODE = 2

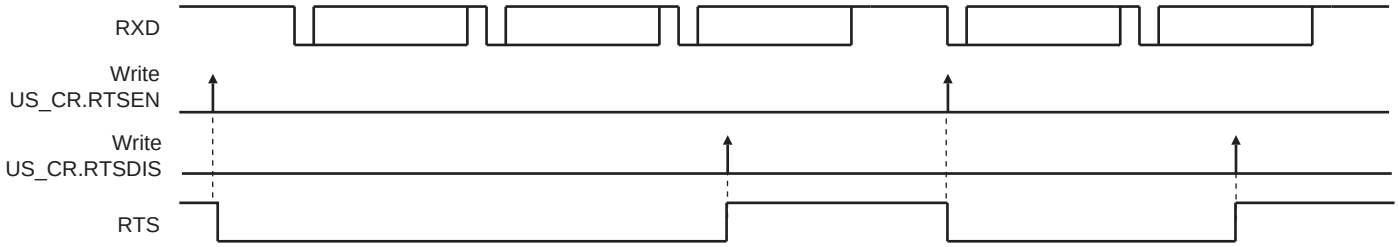
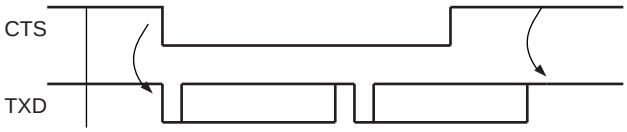


Figure 43-28 shows how the transmitter operates if hardware handshaking is enabled. The CTS pin disables the transmitter. If a character is being processed, the transmitter is disabled only after the completion of the current character and transmission of the next character occurs as soon as the pin CTS falls.

Figure 43-28. Transmitter Behavior when Operating with Hardware Handshaking



43.6.4 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.

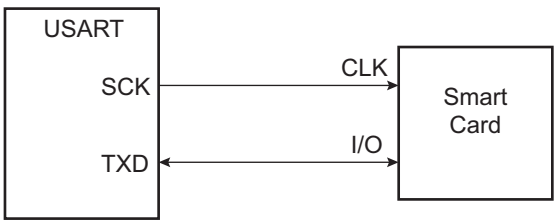
Setting the USART in ISO7816 mode is performed by writing US_MR.USART_MODE to the value 0x4 for protocol T = 0 and to the value 0x6 for protocol T = 1.

43.6.4.1 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 (refer to Section 43-2 “Baud Rate Generator”).

The USART connects to a smart card as shown in Figure 43-29. 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 43-29. 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 Mode register fields CHRL, MODE9, PAR and CHMODE. US_MR.MSBF can be used to transmit LSB or MSB first. US_MR.PAR can be used to transmit in Normal or Inverse mode. Refer to Section 43.7.3 “USART Mode Register” and “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.

43.6.4.2 Protocol T = 0

In T = 0 protocol, a character is made up of one start bit, eight data bits, one parity bit and one 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 Figure 43-30.

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 43-31. This error bit, NACK, for Non Acknowledge. In this case, the character lasts one additional bit time, as the guard time does not change and is added to the error bit time, which lasts one bit time.

When the USART is the receiver and it detects an error, it does not load the erroneous character in US_RHR. It sets US_SR.PARE so that the software can handle the error.

Figure 43-30. T = 0 Protocol without Parity Error

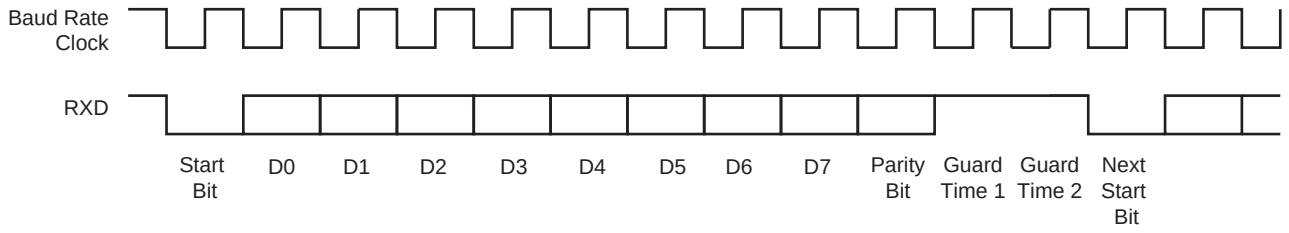
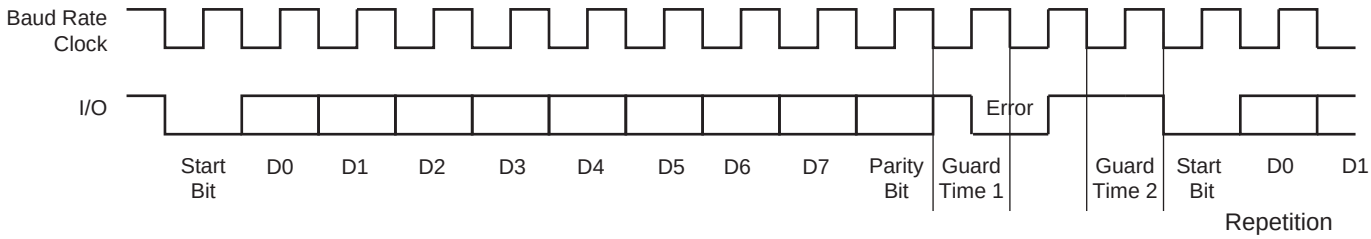


Figure 43-31. T = 0 Protocol with Parity Error



Receive Error Counter

The USART receiver also records the total number of errors. This can be read in the Number of Errors (US_NER) register. The NB_ERRORS field can record up to 255 errors. Reading US_NER automatically clears the NB_ERRORS field.

Receive NACK Inhibit

The USART can be configured to inhibit an error. This is done by writing a ‘1’ to 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.

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 US_MR.MAX_ITERATION to a value greater 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 US_CSR.ITER is set. If the repetition of the character is acknowledged by the receiver, the repetitions are stopped and the iteration counter is cleared.

US_CSR.ITER can be cleared by writing a ‘1’ to US_CR.RSTIT.

Disable Successive Receive NACK

The receiver can limit the number of successive NACKs sent back to the remote transmitter. This is programmed by setting US_MR.DSNACK. The maximum number of NACKs transmitted is configured in US_MR.MAX_ITERATION. As soon as MAX_ITERATION is reached, no error signal is driven on the I/O line and US_CSR.ITER is set.

43.6.4.3 Protocol T = 1

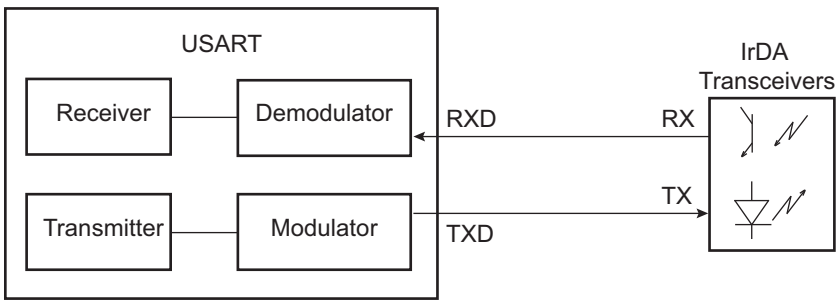
When operating in ISO7816 protocol T = 1, the transmission is similar to an asynchronous format with only one stop bit. The parity is generated when transmitting and checked when receiving. Parity error detection sets US_CSR.PARE.

43.6.5 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 Figure 43-32. 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 IrDA mode is enabled by writing the value 0x8 to US_MR.USART_MODE. The IrDA Filter register (US_IF) is used to configure 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 43-32. Connection to IrDA Transceivers



The receiver and the transmitter must be enabled or disabled depending on 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 pull-up (better for power consumption).
- Receive data

43.6.5.1 IrDA Modulation

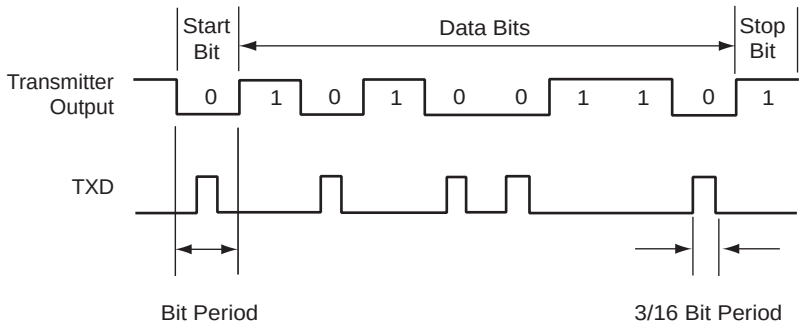
For baud rates up to and including 115.2 kbit/s, the RZI modulation scheme is used. “0” is represented by a light pulse of 3/16th of a bit time. Some examples of signal pulse duration are shown in Table 43-11.

Table 43-11. IrDA Pulse Duration

Baud Rate	Pulse Duration (3/16)
2.4 kbit/s	78.13 µs
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

Figure 43-33 shows an example of character transmission.

Figure 43-33. IrDA Modulation



43.6.5.2 IrDA Baud Rate

Table 43-12 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 43-12. 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

Table 43-12. IrDA Baud Rate Error (Continued)

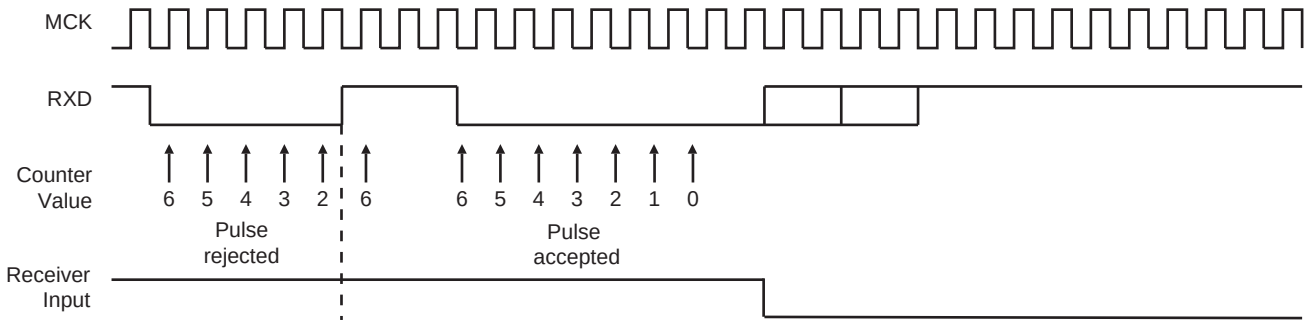
Peripheral Clock	Baud Rate (bit/s)	CD	Baud Rate Error	Pulse Time (µs)
20,000,000	19,200	65	0.16%	9.77
32,768,000	19,200	107	0.31%	9.77
40,000,000	19,200	130	0.16%	9.77
3,686,400	9,600	24	0.00%	19.53
20,000,000	9,600	130	0.16%	19.53
32,768,000	9,600	213	0.16%	19.53
40,000,000	9,600	260	0.16%	19.53
3,686,400	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

43.6.5.3 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 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 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.

Figure 43-34 illustrates the operations of the IrDA demodulator.

Figure 43-34. IrDA Demodulator Operations



The programmed value in the US_IF register must always meet the following criterion:

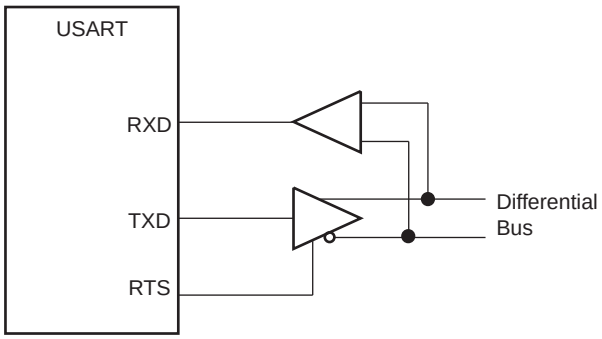
$$t_{peripheral\ clock} \times (IRDA\_FILTER + 3) < 1.41\ \mu s$$

As the IrDA mode uses the same logic as the ISO7816, note that the FI_DI_RATIO field in US_FIDI must be set to a value higher than 0 in order to ensure IrDA communications operate correctly.

43.6.6 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 Figure 43-35.

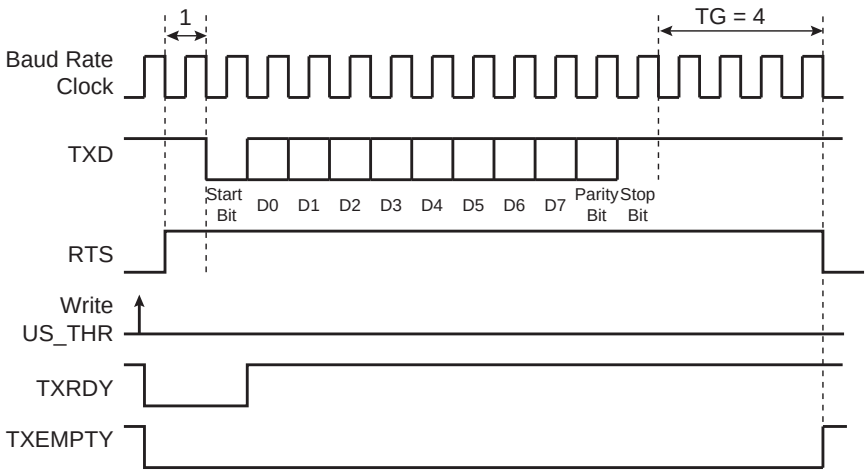
Figure 43-35. Typical Connection to a RS485 Bus



RS485 mode is enabled by writing the value 0x1 to the US_MR.USART_MODE.

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. Figure 43-36 gives an example of the RTS waveform during a character transmission when the timeguard is enabled.

Figure 43-36. Example of RTS Drive with Timeguard



43.6.7 SPI Mode

The Serial Peripheral Interface (SPI) mode 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 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 turns being masters and one master may simultaneously shift data into multiple slaves. (Multiple master protocol is the opposite of single master protocol, where one CPU is always the master while all of the others are always slaves.) However, only one slave may drive its output to write data back to the master at any given time.

A slave device is selected when its NSS signal is asserted by the master. The USART in SPI Master mode can address only one SPI slave because it can generate only one NSS signal.

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 of the slave.
- Master In Slave Out (MISO): This data line supplies the output data from a slave to the input of the master.

- Serial Clock (SCK): This control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of baud rates. The SCK line cycles once for each bit that is transmitted.
- Slave Select (NSS): This control line allows the master to select or deselect the slave.

43.6.7.1 Modes of Operation

The USART can operate in SPI Master mode or in SPI Slave mode.

SPI Master mode is enabled by writing 0xE to US_MR.USART_MODE. In this case, the SPI lines must be connected as described below:

- The MOSI line is driven by the output pin TXD
- The MISO line drives the input pin RXD
- The SCK line is driven by the output pin SCK
- The NSS line is driven by the output pin RTS

SPI Slave mode is enabled by writing to 0xF US_MR.USART_MODE. In this case, the SPI lines must be connected as described below:

- The MOSI line drives the input pin RXD
- The MISO line is driven by the output pin TXD
- The SCK line drives the input pin SCK
- The NSS line drives the input pin CTS

In order to avoid unpredictable behavior, any change of the SPI mode must be followed by a software reset of the transmitter and of the receiver (except the initial configuration after a hardware reset). (Refer to Section 43.6.7.4).

43.6.7.2 Baud Rate

In SPI mode, the baud rate generator operates in the same way as in USART Synchronous mode. Refer to Section 43.6.1.3 “Baud Rate in Synchronous Mode or SPI Mode”. However, there are some restrictions:

In SPI Master mode:

- The external clock SCK must not be selected (USCLKS ≠ 0x3), and US_MR.CLKO must be written to ‘1’, in order to generate correctly the serial clock on the SCK pin.
- To obtain correct behavior of the receiver and the transmitter, the value programmed in US_BRGR.CD must be greater than or equal to 6.
- If the divided peripheral clock is selected, the value programmed in CD must be even to ensure a 50:50 mark/space ratio on the SCK pin. This value can be odd if the peripheral clock is selected.

In SPI Slave mode:

- The external clock (SCK) selection is forced regardless of the value of the US_MR.USCLKS. Likewise, the value written in US_BRGR has no effect, because the clock is provided directly by the signal on the USART SCK pin.
- To obtain correct behavior of the receiver and the transmitter, the external clock (SCK) frequency must be at least 6 times lower than the system clock.

43.6.7.3 Data Transfer

Up to nine data bits are successively shifted out on the TXD pin at each rising or falling edge (depending on CPOL and CPHA) of the programmed serial clock. There is no Start bit, no Parity bit and no Stop bit.

The number of data bits is selected using US_MR.CHRL and US_MR.MODE9. The nine bits are selected by setting the MODE9 bit regardless of the CHRL field. The MSB data bit is always sent first in SPI mode (Master or Slave).

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed using US_MR.CPOL. The clock phase is programmed using US_MR.CPHA. These two parameters determine the

edges of the clock signal upon 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, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are used and fixed in different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

Table 43-13. SPI Bus Protocol Mode

SPI Bus Protocol Mode	CPOL	CPHA
0	0	1
1	0	0
2	1	1
3	1	0

Figure 43-37. SPI Transfer Format (CPHA = 1, 8 bits per transfer)

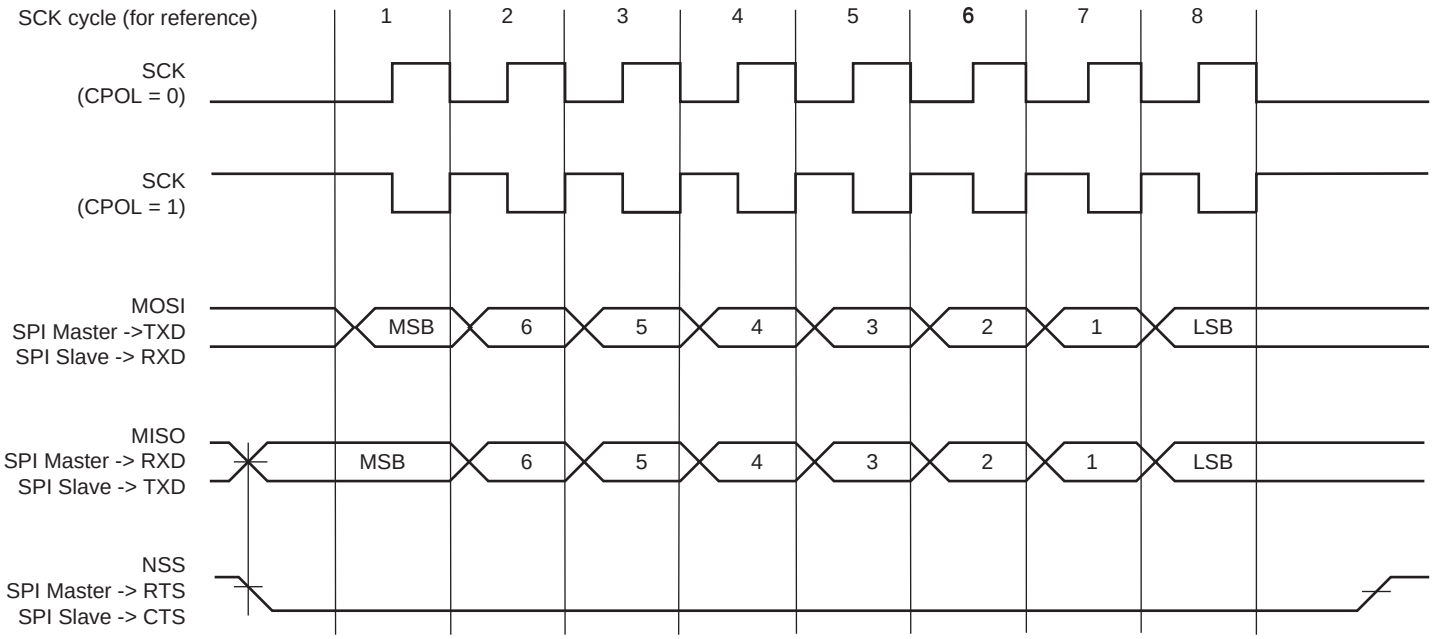
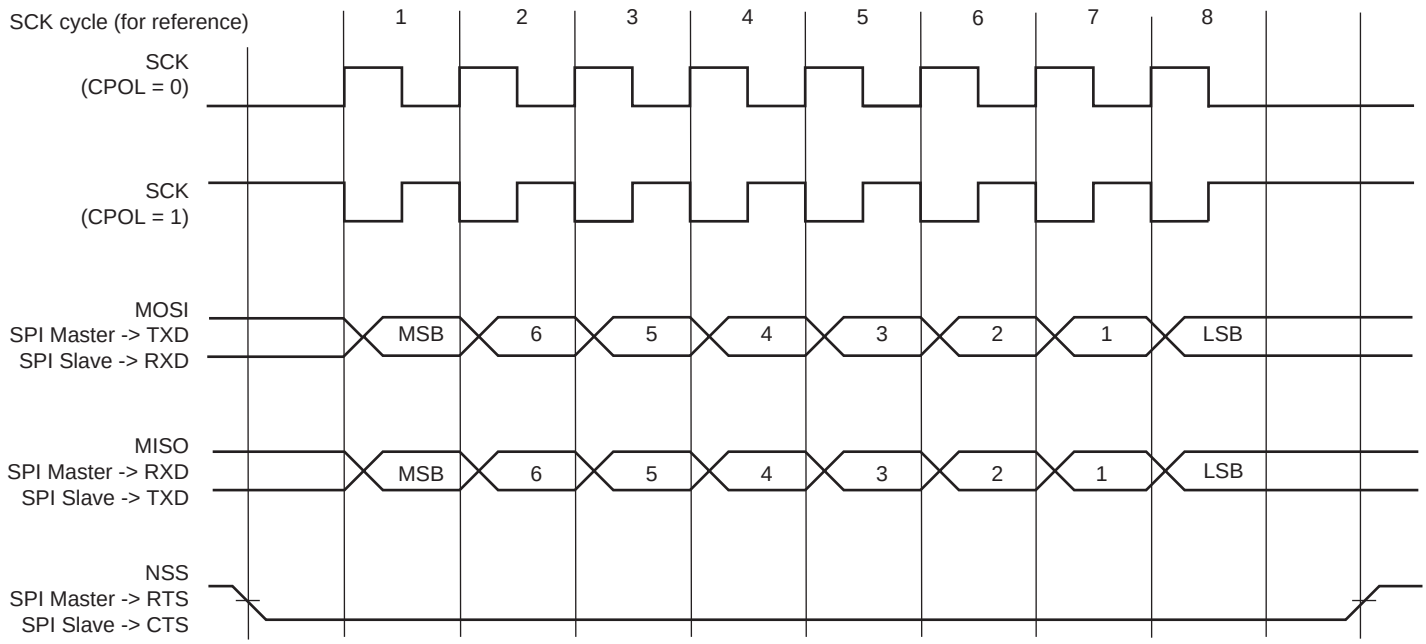


Figure 43-38. SPI Transfer Format (CPHA = 0, 8 bits per transfer)



43.6.7.4 Receiver and Transmitter Control

Refer to Section 43.6.2 “Receiver and Transmitter Control”.

43.6.7.5 Character Transmission

The characters are sent by writing in the US_THR. An additional condition for transmitting a character can be added when the USART is configured in SPI Master mode. In the USART_MR (SPI_MODE), the value of WRDBT can prevent any character transmission (even if US_THR has been written) while the receiver side is not ready (character not read). When WRDBT equals ‘0’, the character is transmitted whatever the receiver status. If WRDBT is set to ‘1’, the transmitter waits for US_RHR to be read before transmitting the character (RXRDY flag cleared), thus preventing any overflow (character loss) on the receiver side.

The chip select line is deasserted for a period equivalent to three bits between the transmission of two data.

The transmitter reports two status bits in US_CSR: TXRDY (Transmitter Ready), which indicates that US_THR is empty and TXEMPTY, which indicates that all the characters written in US_THR have been processed. When the current character processing is completed, the last character written in US_THR is transferred into the Shift register of the transmitter and US_THR becomes empty, thus TXRDY rises.

Both TXRDY and TXEMPTY bits are low when the transmitter is disabled. Writing a character in US_THR while TXRDY is low has no effect and the written character is lost.

If the USART is in SPI Slave mode and if a character must be sent while the US_THR is empty, the UNRE (Underrun Error) bit is set. The TXD transmission line stays at high level during all this time. The UNRE bit is cleared by writing a 1 to the RSTSTA (Reset Status) bit in US_CR.

In SPI Master mode, the slave select line (NSS) is asserted at low level one t_{bit} (t_{bit} being the nominal time required to transmit a bit) before the transmission of the MSB bit and released at high level one t_{bit} after the transmission of the LSB bit. So, the slave select line (NSS) is always released between each character transmission and a minimum delay of three t_{bit} always inserted. However, in order to address slave devices supporting the CSAAT mode (Chip Select Active After Transfer), the slave select line (NSS) can be forced at low level by writing a 1 to the RCS bit in the US_CR. The slave select line (NSS) can be released at high level only by writing a ‘1’ to US_CR.FCS (for example, when all data have been transferred to the slave device).

In SPI Slave mode, the transmitter does not require a falling edge of the slave select line (NSS) to initiate a character transmission but only a low level. However, this low level must be present on the slave select line (NSS) at least one t_{bit} before the first serial clock cycle corresponding to the MSB bit.

43.6.7.6 Character Reception

When a character reception is completed, it is transferred to US_RHR and US_CSR.RXRDY rises. If a character is completed while RXRDY is set, the OVRE (Overrun Error) bit is set. The last character is transferred into US_RHR and overwrites the previous one. The OVRE bit is cleared by writing a '1' to US_CR.RSTSTA.

To ensure correct behavior of the receiver in SPI Slave mode, the master device sending the frame must ensure a minimum delay of one t_{bit} between each character transmission. The receiver does not require a falling edge of the slave select line (NSS) to initiate a character reception but only a low level. However, this low level must be present on the slave select line (NSS) at least one t_{bit} before the first serial clock cycle corresponding to the MSB bit.

43.6.7.7 Receiver Timeout

Because the receiver baud rate clock is active only during data transfers in SPI mode, a receiver timeout is impossible in this mode, whatever the value is in US_RTOR.TO.

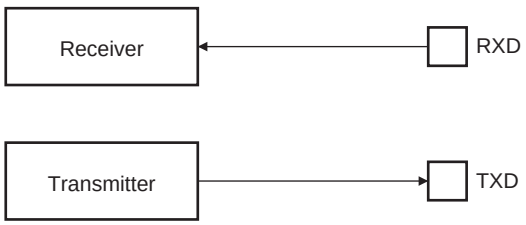
43.6.8 Test Modes

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.

43.6.8.1 Normal Mode

Normal mode connects the RXD pin on the receiver input and the transmitter output on the TXD pin.

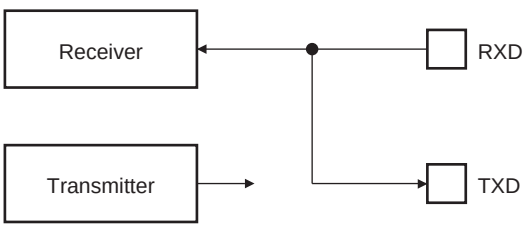
Figure 43-39. Normal Mode Configuration



43.6.8.2 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 Figure 43-40. 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.

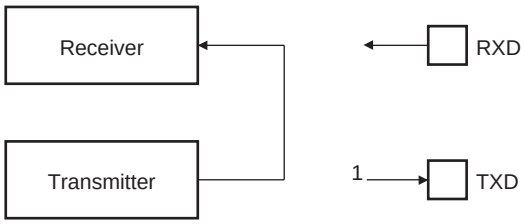
Figure 43-40. Automatic Echo Mode Configuration



43.6.8.3 Local Loopback Mode

Local Loopback mode connects the output of the transmitter directly to the input of the receiver, as shown in Figure 43-41. 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.

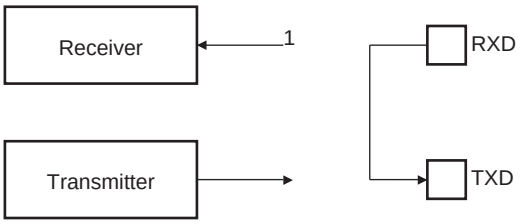
Figure 43-41. Local Loopback Mode Configuration



43.6.8.4 Remote Loopback Mode

Remote Loopback mode directly connects the RXD pin to the TXD pin, as shown in Figure 43-42. The transmitter and the receiver are disabled and have no effect. This mode allows bit-by-bit retransmission.

Figure 43-42. Remote Loopback Mode Configuration



43.6.9 Register Write Protection

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

If a write access to a write-protected register is detected, the WPVS flag in the USART Write Protection Status Register (US_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 US_WPSR.

The following registers can be write-protected:

- USART Mode Register
- USART Baud Rate Generator Register
- USART Receiver Timeout Register
- USART Transmitter Timeguard Register
- USART Manchester Configuration Register

43.7 Universal Synchronous Asynchronous Receiver Transmitter (USART) User Interface

Table 43-14. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	Control Register	US_CR	Write-only	–
0x0004	Mode Register	US_MR	Read/Write	0x0
0x0008	Interrupt Enable Register	US_IER	Write-only	–
0x000C	Interrupt Disable Register	US_IDR	Write-only	–
0x0010	Interrupt Mask Register	US_IMR	Read-only	0x0
0x0014	Channel Status Register	US_CSR	Read-only	0x0
0x0018	Receive Holding Register	US_RHR	Read-only	0x0
0x001C	Transmit Holding Register	US_THR	Write-only	–
0x0020	Baud Rate Generator Register	US_BRGR	Read/Write	0x0
0x0024	Receiver Timeout Register	US_RTOR	Read/Write	0x0
0x0028	Transmitter Timeguard Register	US_TTGR	Read/Write	0x0
0x002C–0x003C	Reserved	–	–	–
0x0040	FI DI Ratio Register	US_FIDI	Read/Write	0x174
0x0044	Number of Errors Register	US_NER	Read-only	0x0
0x0048	Reserved	–	–	–
0x004C	IrDA Filter Register	US_IF	Read/Write	0x0
0x0050	Manchester Configuration Register	US_MAN	Read/Write	0x30011004
0x0054–0x005C	Reserved	–	–	–
0x0060–0x00E0	Reserved	–	–	–
0x00E4	Write Protection Mode Register	US_WPMR	Read/Write	0x0
0x00E8	Write Protection Status Register	US_WPSR	Read-only	0x0
0x00EC–0x00FC	Reserved	–	–	–

43.7.1 USART Control Register

Name: US_CR

Address: 0xF802C000 (0), 0xF8030000 (1), 0xFC008000 (2), 0xFC00C000 (3), 0xFC010000 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	RTSDIS	RTSEN	–	–
15	14	13	12	11	10	9	8
RETTO	RSTNACK	RSTIT	SENDA	STTTO	STPBRK	STTBRK	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

For SPI control, refer to Section 43.7.2 “USART Control Register (SPI_MODE)”.

- **RSTRX: Reset Receiver**

0: No effect.

1: Resets the receiver.

- **RSTTX: Reset Transmitter**

0: No effect.

1: Resets the transmitter.

- **RXEN: Receiver Enable**

0: No effect.

1: Enables the receiver, if RXDIS is 0.

- **RXDIS: Receiver Disable**

0: No effect.

1: Disables the receiver.

- **TXEN: Transmitter Enable**

0: No effect.

1: Enables the transmitter if TXDIS is 0.

- **TXDIS: Transmitter Disable**

0: No effect.

1: Disables the transmitter.

- **RSTSTA: Reset Status Bits**

0: No effect.

1: Resets the status bits PARE, FRAME, OVRE, MANERR and RXBRK in US_CSR.

- **STTBRK: Start Break**

0: No effect.

1: Starts transmission of a break after the characters present in US_THR and the Transmit Shift Register have been transmitted. No effect if a break is already being transmitted.

- **STPBRK: Stop Break**

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.

- **STTTO: Clear TIMEOUT Flag and Start Timeout After Next Character Received**

0: No effect.

1: Starts waiting for a character before enabling the timeout counter. Immediately disables a timeout period in progress. Resets the status bit TIMEOUT in US_CSR.

- **SENDA: Send Address**

0: No effect.

1: In Multidrop mode only, the next character written to the US_THR is sent with the address bit set.

- **RSTIT: Reset Iterations**

0: No effect.

1: Resets ITER in US_CSR. No effect if the ISO7816 is not enabled.

- **RSTNACK: Reset Non Acknowledge**

0: No effect

1: Resets NACK in US_CSR.

- **RETTO: Start Timeout Immediately**

0: No effect

1: Immediately restarts timeout period.

- **RTSEN: Request to Send Pin Control**

0: No effect.

1: Drives RTS pin to 1 if US_MR.USART_MODE field = 2, else drives RTS pin to 0 if US_MR.USART_MODE field = 0.

- **RTSDIS: Request to Send Pin Control**

0: No effect.

1: Drives RTS pin to 0 if US_MR.USART_MODE field = 2 (if PDC RX buffer is not full), else drives RTS pin to 1 if US_MR.USART_MODE field = 0.

43.7.2 USART Control Register (SPI_MODE)

Name: US_CR (SPI_MODE)

Address: 0xF802C000 (0), 0xF8030000 (1), 0xFC008000 (2), 0xFC00C000 (3), 0xFC010000 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	RCS	FCS	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

- **RSTRX: Reset Receiver**

0: No effect.

1: Resets the receiver.

- **RSTTX: Reset Transmitter**

0: No effect.

1: Resets the transmitter.

- **RXEN: Receiver Enable**

0: No effect.

1: Enables the receiver, if RXDIS is 0.

- **RXDIS: Receiver Disable**

0: No effect.

1: Disables the receiver.

- **TXEN: Transmitter Enable**

0: No effect.

1: Enables the transmitter if TXDIS is 0.

- **TXDIS: Transmitter Disable**

0: No effect.

1: Disables the transmitter.

- **RSTSTA: Reset Status Bits**

0: No effect.

1: Resets the status bits OVRE, UNRE in US_CSR.

- **FCS: Force SPI Chip Select**

Applicable if USART operates in SPI Master mode (USART_MODE = 0xE):

0: No effect.

1: Forces the Slave Select Line NSS (RTS pin) to 0, even if USART is not transmitting, in order to address SPI slave devices supporting the CSAAT mode (Chip Select Active After Transfer).

- **RCS: Release SPI Chip Select**

Applicable if USART operates in SPI Master mode (USART_MODE = 0xE):

0: No effect.

1: Releases the Slave Select Line NSS (RTS pin).

43.7.3 USART Mode Register

Name: US_MR

Address: 0xF802C004 (0), 0xF8030004 (1), 0xFC008004 (2), 0xFC00C004 (3), 0xFC010004 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
ONEBIT	MODSYNC	MAN	FILTER	–	MAX_ITERATION		
23	22	21	20	19	18	17	16
INVDATA	VAR_SYNC	DSNACK	INACK	OVER	CLKO	MODE9	MSBF
15	14	13	12	11	10	9	8
CHMODE		NBSTOP		PAR			SYNC
7	6	5	4	3	2	1	0
CHRL		USCLKS		USART_MODE			

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

For SPI configuration, refer to Section 43.7.4 “USART Mode Register (SPI_MODE)”.

- **USART_MODE: USART Mode of Operation**

Value	Name	Description
0x0	NORMAL	Normal mode
0x1	RS485	RS485
0x2	HW_HANDSHAKING	Hardware Handshaking
0x3	—	Reserved
0x4	IS07816_T_0	IS07816 Protocol: T = 0
0x6	IS07816_T_1	IS07816 Protocol: T = 1
0x8	IRDA	IrDA
0xE	SPI_MASTER	SPI Master mode (CLKO must be written to 1 and USCLKS = 0, 1 or 2)
0xF	SPI_SLAVE	SPI Slave mode

- **USCLKS: Clock Selection**

Value	Name	Description
0	MCK	Peripheral clock is selected
1	DIV	Peripheral clock divided (DIV=8) is selected
2	—	Reserved
3	SCK	Serial clock (SCK) is selected

- **CHRL: 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

- **SYNC: Synchronous Mode Select**

0: USART operates in Asynchronous mode.

1: USART operates in Synchronous mode.

- **PAR: 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

- **NBSTOP: Number of Stop Bits**

Value	Name	Description
0	1_BIT	1 stop bit
1	1_5_BIT	1.5 stop bit (SYNC = 0) or reserved (SYNC = 1)
2	2_BIT	2 stop bits

- **CHMODE: 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.

- **MSBF: Bit Order**

0: Least significant bit is sent/received first.

1: Most significant bit is sent/received first.

- **MODE9: 9-bit Character Length**

0: CHRL defines character length.

1: 9-bit character length.

- **CLKO: Clock Output Select**

0: The USART does not drive the SCK pin.

1: The USART drives the SCK pin if USCLKS does not select the external clock SCK.

- **OVER: Oversampling Mode**

0: 16X Oversampling

1: 8X Oversampling

- **INACK: Inhibit Non Acknowledge**

0: The NACK is generated.

1: The NACK is not generated.

- **DSNACK: Disable Successive NACK**

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.

Note: MAX_ITERATION field must be set to 0 if DSNACK is cleared.

- **INVDATA: Inverted Data**

0: The data field transmitted on TXD line is the same as the one written in US_THR or the content read in US_RHR is the same as RXD line. Normal mode of operation.

1: The data field transmitted on TXD line is inverted (voltage polarity only) compared to the value written on US_THR or the content read in 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.

- **VAR_SYNC: Variable Synchronization of Command/Data Sync Start Frame Delimiter**

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 US_THR.

- **MAX_ITERATION: Maximum Number of Automatic Iteration**

0–7: Defines the maximum number of iterations in ISO7816 mode, protocol T = 0.

- **FILTER: Receive Line Filter**

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).

- **MAN: Manchester Encoder/Decoder Enable**

0: Manchester encoder/decoder are disabled.

1: Manchester encoder/decoder are enabled.

- **MODSYNC: Manchester Synchronization Mode**

0:The Manchester start bit is a 0 to 1 transition

1: The Manchester start bit is a 1 to 0 transition.

- **ONEBIT: Start Frame Delimiter Selector**

0: Start frame delimiter is COMMAND or DATA SYNC.

1: Start frame delimiter is one bit.

43.7.4 USART Mode Register (SPI_MODE)

Name: US_MR (SPI_MODE)

Address: 0xF802C004 (0), 0xF8030004 (1), 0xFC008004 (2), 0xFC00C004 (3), 0xFC010004 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	WRDBT	–	CLKO	–	CPOL
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CPHA
7	6	5	4	3	2	1	0
CHRL		USCLKS		USART_MODE			

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.
This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **USART_MODE: USART Mode of Operation**

Value	Name	Description
0xE	SPI_MASTER	SPI master
0xF	SPI_SLAVE	SPI Slave

- **USCLKS: Clock Selection**

Value	Name	Description
0	MCK	Peripheral clock is selected
1	DIV	Peripheral clock divided (DIV=8) is selected
3	SCK	Serial Clock (SCK) is selected

- **CHRL: Character Length**

Value	Name	Description
3	8_BIT	Character length is 8 bits

- **CPHA: SPI Clock Phase**
  – Applicable if USART operates in SPI mode (USART_MODE = 0xE or 0xF):

0: Data is changed on the leading edge of SPCK and captured on the following edge of SPCK.

1: Data is captured on the leading edge of SPCK and changed on the following edge of SPCK.

CPHA determines which edge of SPCK 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.

- **CPOL: SPI Clock Polarity**

Applicable if USART operates in SPI mode (Slave or Master, USART_MODE = 0xE or 0xF):

0: The inactive state value of SPCK is logic level zero.

1: The inactive state value of SPCK is logic level one.

CPOL is used to determine the inactive state value of the serial clock (SCK). It is used with CPHA to produce the required clock/data relationship between master and slave devices.

- **CLKO: Clock Output Select**

0: The USART does not drive the SCK pin.

1: The USART drives the SCK pin if USCLKS does not select the external clock SCK.

- **WRDBT: Wait Read Data Before Transfer**

0: The character transmission starts as soon as a character is written into US_THR (assuming TXRDY was set).

1: The character transmission starts when a character is written and only if RXRDY flag is cleared (Receive Holding Register has been read).

43.7.5 USART Interrupt Enable Register

Name: US_IER

Address: 0xF802C008 (0), 0xF8030008 (1), 0xFC008008 (2), 0xFC00C008 (3), 0xFC010008 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	MANE
23	22	21	20	19	18	17	16
–	–	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	–	–	ITER	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

For SPI specific configuration, refer to Section 43.7.6 “USART Interrupt Enable Register (SPI_MODE)”.

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

0: No effect

1: Enables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Enable**
- **TXRDY: TXRDY Interrupt Enable**
- **RXBRK: Receiver Break Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**
- **FRAME: Framing Error Interrupt Enable**
- **PARE: Parity Error Interrupt Enable**
- **TIMEOUT: Timeout Interrupt Enable**
- **TXEMPTY: TXEMPTY Interrupt Enable**
- **ITER: Max number of Repetitions Reached Interrupt Enable**
- **NACK: Non Acknowledge Interrupt Enable**
- **CTSIC: Clear to Send Input Change Interrupt Enable**
- **MANE: Manchester Error Interrupt Enable**

43.7.6 USART Interrupt Enable Register (SPI_MODE)

Name: US_IER (SPI_MODE)

Address: 0xF802C008 (0), 0xF8030008 (1), 0xFC008008 (2), 0xFC00C008 (3), 0xFC010008 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	NSSE	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

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

0: No effect

1: Enables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Enable**
- **TXRDY: TXRDY Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**
- **TXEMPTY: TXEMPTY Interrupt Enable**
- **UNRE: SPI Underrun Error Interrupt Enable**
- **NSSE: NSS Line (Driving CTS Pin) Rising or Falling Edge Event Interrupt Enable**

43.7.7 USART Interrupt Disable Register

Name: US_IDR

Address: 0xF802C00C (0), 0xF803000C (1), 0xFC00800C (2), 0xFC00C00C (3), 0xFC01000C (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	MANE
23	22	21	20	19	18	17	16
–	–	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	–	–	ITER	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

For SPI specific configuration, refer to Section 43.7.8 “USART Interrupt Disable Register (SPI_MODE)”.

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

0: No effect

1: Disables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Disable**
- **TXRDY: TXRDY Interrupt Disable**
- **RXBRK: Receiver Break Interrupt Disable**
- **OVRE: Overrun Error Interrupt Enable**
- **FRAME: Framing Error Interrupt Disable**
- **PARE: Parity Error Interrupt Disable**
- **TIMEOUT: Timeout Interrupt Disable**
- **TXEMPTY: TXEMPTY Interrupt Disable**
- **ITER: Max Number of Repetitions Reached Interrupt Disable**
- **NACK: Non Acknowledge Interrupt Disable**
- **CTSIC: Clear to Send Input Change Interrupt Disable**
- **MANE: Manchester Error Interrupt Disable**

43.7.8 USART Interrupt Disable Register (SPI_MODE)

Name: US_IDR (SPI_MODE)

Address: 0xF802C00C (0), 0xF803000C (1), 0xFC00800C (2), 0xFC00C00C (3), 0xFC01000C (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	NSSE	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

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

0: No effect

1: Disables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Disable**
- **TXRDY: TXRDY Interrupt Disable**
- **OVRE: Overrun Error Interrupt Disable**
- **TXEMPTY: TXEMPTY Interrupt Disable**
- **UNRE: SPI Underrun Error Interrupt Disable**
- **NSSE: NSS Line (Driving CTS Pin) Rising or Falling Edge Event Interrupt Disable**

43.7.9 USART Interrupt Mask Register

Name: US_IMR

Address: 0xF802C010 (0), 0xF8030010 (1), 0xFC008010 (2), 0xFC00C010 (3), 0xFC010010 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	MANE
23	22	21	20	19	18	17	16
–	–	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	–	–	ITER	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

For SPI specific configuration, refer to Section 43.7.10 “USART Interrupt Mask Register (SPI_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.

- **RXRDY: RXRDY Interrupt Mask**
- **TXRDY: TXRDY Interrupt Mask**
- **RXBRK: Receiver Break Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **FRAME: Framing Error Interrupt Mask**
- **PARE: Parity Error Interrupt Mask**
- **TIMEOUT: Timeout Interrupt Mask**
- **TXEMPTY: TXEMPTY Interrupt Mask**
- **ITER: Max Number of Repetitions Reached Interrupt Mask**
- **NACK: Non Acknowledge Interrupt Mask**
- **CTSIC: Clear to Send Input Change Interrupt Mask**
- **MANE: Manchester Error Interrupt Mask**

43.7.10 USART Interrupt Mask Register (SPI_MODE)

Name: US_IMR (SPI_MODE)

Address: 0xF802C010 (0), 0xF8030010 (1), 0xFC008010 (2), 0xFC00C010 (3), 0xFC010010 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	NSSE	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF 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.

- **RXRDY: RXRDY Interrupt Mask**
- **TXRDY: TXRDY Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **TXEMPTY: TXEMPTY Interrupt Mask**
- **UNRE: SPI Underrun Error Interrupt Mask**
- **NSSE: NSS Line (Driving CTS Pin) Rising or Falling Edge Event Interrupt Mask**

43.7.11 USART Channel Status Register

Name: US_CSR

Address: 0xF802C014 (0), 0xF8030014 (1), 0xFC008014 (2), 0xFC00C014 (3), 0xFC010014 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	MANERR
23	22	21	20	19	18	17	16
CTS	–	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	–	–	ITER	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

For SPI specific configuration, refer to Section 43.7.12 “USART Channel Status Register (SPI_MODE)”.

- **RXRDY: Receiver Ready (cleared by reading US_RHR)**

0: No complete character has been received since the last read of 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 US_RHR has not yet been read.

- **TXRDY: Transmitter Ready (cleared by writing US_THR)**

0: A character is in the US_THR 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 the US_THR.

- **RXBRK: Break Received/End of Break (cleared by writing a one to bit US_CR.RSTSTA)**

0: No break received or end of break detected since the last RSTSTA.

1: Break received or end of break detected since the last RSTSTA.

- **OVRE: Overrun Error (cleared by writing a one to bit US_CR.RSTSTA)**

0: No overrun error has occurred since the last RSTSTA.

1: At least one overrun error has occurred since the last RSTSTA.

- **FRAME: Framing Error (cleared by writing a one to bit US_CR.RSTSTA)**

0: No stop bit has been detected low since the last RSTSTA.

1: At least one stop bit has been detected low since the last RSTSTA.

- **PARE: Parity Error (cleared by writing a one to bit US_CR.RSTSTA)**

0: No parity error has been detected since the last RSTSTA.

1: At least one parity error has been detected since the last RSTSTA.

- **TIMEOUT: Receiver Timeout (cleared by writing a one to bit US_CR.STTTO)**

0: There has not been a timeout since the last Start Timeout command (STTTO in US_CR) or the Timeout Register is 0.

1: There has been a timeout since the last Start Timeout command (STTTO in US_CR).

- **TXEMPTY: Transmitter Empty (cleared by writing US_THR)**

0: There are characters in either US_THR or the Transmit Shift Register, or the transmitter is disabled.

1: There are no characters in US_THR, nor in the Transmit Shift Register.

- **ITER: Max Number of Repetitions Reached (cleared by writing a one to bit US_CR.RSTIT)**

0: Maximum number of repetitions has not been reached since the last RSTIT.

1: Maximum number of repetitions has been reached since the last RSTIT.

- **NACK: Non Acknowledge Interrupt (cleared by writing a one to bit US_CR.RSTNACK)**

0: Non acknowledge has not been detected since the last RSTNACK.

1: At least one non acknowledge has been detected since the last RSTNACK.

- **CTSIC: Clear to Send Input Change Flag (cleared on read)**

0: No input change has been detected on the CTS pin since the last read of US_CSR.

1: At least one input change has been detected on the CTS pin since the last read of US_CSR.

- **CTS: Image of CTS Input**

0: CTS input is driven low.

1: CTS input is driven high.

- **MANERR: Manchester Error (cleared by writing a one to the bit US_CR.RSTSTA)**

0: No Manchester error has been detected since the last RSTSTA.

1: At least one Manchester error has been detected since the last RSTSTA.

43.7.12 USART Channel Status Register (SPI_MODE)

Name: US_CSR (SPI_MODE)

Address: 0xF802C014 (0), 0xF8030014 (1), 0xFC008014 (2), 0xFC00C014 (3), 0xFC010014 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
NSS	–	–	–	NSSE	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

- **RXRDY: Receiver Ready (cleared by reading US_RHR)**

0: No complete character has been received since the last read of 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 US_RHR has not yet been read.

- **TXRDY: Transmitter Ready (cleared by writing US_THR)**

0: A character is in the US_THR 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 the US_THR.

- **OVRE: Overrun Error (cleared by writing a one to bit US_CR.RSTSTA)**

0: No overrun error has occurred since the last RSTSTA.

1: At least one overrun error has occurred since the last RSTSTA.

- **TXEMPTY: Transmitter Empty (cleared by writing US_THR)**

0: There are characters in either US_THR or the Transmit Shift Register, or the transmitter is disabled.

1: There are no characters in US_THR, nor in the Transmit Shift Register.

- **UNRE: Underrun Error (cleared by writing a one to bit US_CR.RSTSTA)**

0: No SPI underrun error has occurred since the last RSTSTA.

1: At least one SPI underrun error has occurred since the last RSTSTA.

- **NSSE: NSS Line (Driving CTS Pin) Rising or Falling Edge Event (cleared on read)**

0: No NSS line event has been detected since the last read of US_CSR.

1: A rising or falling edge event has been detected on NSS line since the last read of US_CSR.

- **NSS: Image of NSS Line**

0: NSS line is driven low (if NSSE = 1, falling edge occurred on NSS line).

1: NSS line is driven high (if NSSE = 1, rising edge occurred on NSS line).

43.7.13 USART Receive Holding Register

Name: US_RHR

Address: 0xF802C018 (0), 0xF8030018 (1), 0xFC008018 (2), 0xFC00C018 (3), 0xFC010018 (4)

Access: Read-only

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

- **RXCHR: Received Character**

Last character received if RXRDY is set.

- **RXSYNH: Received Sync**

0: Last character received is a data.

1: Last character received is a command.

43.7.14 USART Transmit Holding Register

Name: US_THR

Address: 0xF802C01C (0), 0xF803001C (1), 0xFC00801C (2), 0xFC00C01C (3), 0xFC01001C (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TXSYNH	–	–	–	–	–	–	TXCHR

7	6	5	4	3	2	1	0
TXCHR							

- **TXCHR: Character to be Transmitted**

Next character to be transmitted after the current character if TXRDY is not set.

- **TXSYNH: Sync Field to be Transmitted**

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.

43.7.15 USART Baud Rate Generator Register

Name: US_BRGR

Address: 0xF802C020 (0), 0xF8030020 (1), 0xFC008020 (2), 0xFC00C020 (3), 0xFC010020 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	FP		

15	14	13	12	11	10	9	8
CD							

7	6	5	4	3	2	1	0
CD							

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

- **CD: Clock Divider**

	USART_MODE ≠ ISO7816			
	SYNC = 0		**SYNC = 1 or USART_MODE = SPI (Master or Slave)**	**USART_MODE = ISO7816**
CD	**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)

- **FP: Fractional Part**

0: Fractional divider is disabled.

1–7: Baud rate resolution, defined by FP × 1/8.

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.

43.7.16 USART Receiver Timeout Register

Name: US_RTOR

Address: 0xF802C024 (0), 0xF8030024 (1), 0xFC008024 (2), 0xFC00C024 (3), 0xFC010024 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TO							

7	6	5	4	3	2	1	0
TO							

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

- **TO: Timeout Value**

0: The receiver timeout is disabled.

1–65535: The receiver timeout is enabled and TO is Timeout Delay / Bit Period.

43.7.17 USART Transmitter Timeguard Register

Name: US_TTGR

Address: 0xF802C028 (0), 0xF8030028 (1), 0xFC008028 (2), 0xFC00C028 (3), 0xFC010028 (4)

Access: Read/Write

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
TG							

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

- **TG: Timeguard Value**

0: The transmitter timeguard is disabled.

1–255: The transmitter timeguard is enabled and TG is Timeguard Delay / Bit Period.

43.7.18 USART FI DI RATIO Register

Name: US_FIDI

Address: 0xF802C040 (0), 0xF8030040 (1), 0xFC008040 (2), 0xFC00C040 (3), 0xFC010040 (4)

Access: Read/Write

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

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

- **FI_DI_RATIO: FI Over DI Ratio Value**

0: If ISO7816 mode is selected, the baud rate generator generates no signal.

1–2: Do not use.

3–65535: If ISO7816 mode is selected, the baud rate is the clock provided on SCK divided by FI_DI_RATIO.

43.7.19 USART Number of Errors Register

Name: US_NER

Address: 0xF802C044 (0), 0xF8030044 (1), 0xFC008044 (2), 0xFC00C044 (3), 0xFC010044 (4)

Access: Read-only

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
NB_ERRORS							

This register is relevant only if USART_MODE = 0x4 or 0x6 in the USART Mode Register.

- **NB_ERRORS: Number of Errors**

Total number of errors that occurred during an ISO7816 transfer. This register automatically clears when read.

43.7.20 USART IrDA Filter Register

Name: US_IF

Address: 0xF802C04C (0), 0xF803004C (1), 0xFC00804C (2), 0xFC00C04C (3), 0xFC01004C (4)

Access: Read/Write

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
IRDA_FILTER							

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.

- **IRDA_FILTER: IrDA Filter**

The IRDA_FILTER value must be defined to meet the following criteria:

$t_{peripheral\ clock} \times (IRDA\_FILTER + 3) < 1.41\ \mu s$

43.7.21 USART Manchester Configuration Register

Name: US_MAN

Address: 0xF802C050 (0), 0xF8030050 (1), 0xFC008050 (2), 0xFC00C050 (3), 0xFC010050 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	DRIFT	ONE	RX_MPOL	–	–	RX_PP	

23	22	21	20	19	18	17	16
–	–	–	–	RX_PL			

15	14	13	12	11	10	9	8
–	–	–	TX_MPOL	–	–	TX_PP	

7	6	5	4	3	2	1	0
–	–	–	–	TX_PL			

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

- **TX_PL: Transmitter Preamble Length**

0: The transmitter preamble pattern generation is disabled

1–15: The preamble length is TX_PL × Bit Period

- **TX_PP: 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

- **TX_MPOL: Transmitter Manchester Polarity**

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.

- **RX_PL: Receiver Preamble Length**

0: The receiver preamble pattern detection is disabled

1–15: The detected preamble length is RX_PL × Bit Period

- **RX_PP: Receiver Preamble Pattern detected**

The following values assume that RX_MPOL field is not set:

Value	Name	Description
00	ALL_ONE	The preamble is composed of ‘1’s
01	ALL_ZERO	The preamble is composed of ‘0’s
10	ZERO_ONE	The preamble is composed of ‘01’s
11	ONE_ZERO	The preamble is composed of ‘10’s

- **RX_MPOL: Receiver Manchester Polarity**

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.

- **ONE: Must Be Set to 1**

Bit 29 must always be set to 1 when programming the US_MAN register.

- **DRIFT: Drift Compensation**

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.

43.7.22 USART Write Protection Mode Register

Name: US_WPMR

Address: 0xF802C0E4 (0), 0xF80300E4 (1), 0xFC0080E4 (2), 0xFC00C0E4 (3), 0xFC0100E4 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x555341 (“USA” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x555341 (“USA” in ASCII).

Refer to Section 43.6.9 “Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x555341	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

43.7.23 USART Write Protection Status Register

Name: US_WPSR

Address: 0xF802C0E8 (0), 0xF80300E8 (1), 0xFC0080E8 (2), 0xFC00C0E8 (3), 0xFC0100E8 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the US_WPSR.

1: A write protection violation has occurred since the last read of the US_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

44. Software Modem Device (SMD)

44.1 Description

The Software Modem Device (SMD) is a block for communication via a modem's Digital Isolation Barrier (DIB) with a complementary Line Side Device (LSD).

SMD and LSD are two parts of the "Transformer only" solution. The transformer is the only component connecting SMD and LSD and is used for power, clock and data transfers. Power and clock are supplied by the SMD and consumed by the LSD. The data flow is bidirectional. The data transfer is based on pulse width modulation for transmission from the SMD to the LSD, and for receiving from the LSD.

There are two channels embedded into the protocol of the DIB link:

- Data channel
- Control channel

Each channel is bidirectional.

The data channel is used to transfer digitized signal samples at a constant rate of 16 bits at 16 kHz.

The control channel is used to communicate with control registers of the LSD at a maximum rate of 8 bits at 16 kHz.

The SMD performs all protocol-related data conversion for transmission and received data interpretation in both data and control channels of the link.

The SMD incorporates both RX and TX FIFOs, available through the DMAC interface. Each FIFO is able to hold eight 32-bit words (equivalent to 16 modem data samples).

44.2 Embedded Characteristics

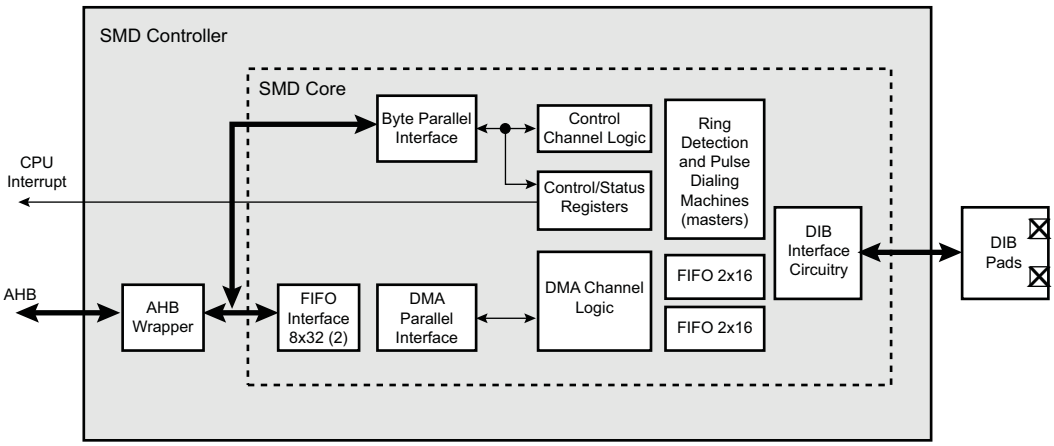
- Modulations and protocols
  - V.90
  - V.34
  - V.32bis, V.32, V.22bis, V.22, V.23, V.21
  - V.23 reverse, V.23 half-duplex
  - Bell 212A/Bell 103
  - V.29 FastPOS
  - V.22bis fast connect
  - V.80 Synchronous Access Mode
- Data compression and error correction
  - V.44 data compression (V.92 model)
  - V.42bis and MNP 5 data compression
  - V.42 LAPM and MNP 2-4 error correction
  - EIA/TIA 578 Class 1 and T.31 Class 1.0
- Call Waiting (CW) detection and Type II Caller ID decoding during data mode
- Type I Caller ID (CID) decoding
- 63 embedded and upgradable country profiles
- Embedded AT commands
- SmartDAA
  - Extension pick-up detection
  - Digital line protection
  - Line reversal detection

- Line-in-use detection
- Remote hang-up detection

- Worldwide compliance

44.3 Block Diagram

Figure 44-1. Software Modem Device Block Diagram



44.4 Software Modem Device (SMD) User Interface

The SMD presents a number of registers through the AHB interface for software control and status functions.

Table 44-1. Register Mapping

Offset	Register	Name	Access	Reset
0x0C	SMD Drive Register	SMD_DRIVE	Read/Write	0x00000002

44.4.1 SMD Drive Register

Name: SMD_DRIVE

Address: 0x0090000C

Access: Read/Write

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
PWRCLKP_PCS		PWRCLKN_PCS2		PWRCLKP_PV	PWRCLKP_PV2	DC_PWRCLKPN	MIE

- **PWRCLKP_PCS: PWRCLKP Pin Control Select**

When DC_PWRCLKPN is a 1, the usage of PWRCLKP_PCS bits for direct control of PWRCLKP pin is enabled as follows:

X1: High impedance on PWRCLKP pin.

00: Drive low on PWRCLKP pin.

10: Drive high on PWRCLKP pin.

When DC_PWRCLKPN is a 0, the protocol logic controls PWRCLKP pin.

If PWRCLKPN_FS bit is a 1, the above information is applied to PWRCLKN pin because of swapping with PWRCLKP.

- **PWRCLKN_PCS2: PWRCLKN Pin Control Select**

When DC_PWRCLKPN is a 1, the usage of PWRCLKN_PCS2 bits for direct control of PWRCLKN pin is enabled as follows:

X1: High impedance on PWRCLKN pin.

00: Drive low on PWRCLKN pin.

10: Drive high on PWRCLKN pin.

When DC_PWRCLKPN is a 0, the protocol logic controls PWRCLKN pin.

If PWRCLKPN_FS bit is a 1, the above information is applied to PWRCLKP pin because of swapping with PWRCLKN.

- **PWRCLKP_PV: PWRCLKP Pin Value**

This bit reflects the PWRCLKP pin value if PWRCLKPN_FS = 0, or the PWRCLKN pin value if PWRCLKPN_FS = 1 (because of swapping with PWRCLKP).

- **PWRCLKP_PV2: PWRCLKP Pin Value**

This bit reflects the PWRCLKN pin value if PWRCLKPN_FS = 0, or the PWRCLKP pin value if PWRCLKPN_FS = 1 (because of swapping with PWRCLKN).

- **DC_PWRCLKPN: Direct Control of PWRCLKP, PWRCLKN Pins Enable**

0: Enables protocol logic control of PWRCLKP, PWRCLKN pins.

1: Enables the use of PWRCLKP_PCS and PWRCLKN_PCS2 bits for direct control of PWRCLKP, PWRCLKN pins making them general purpose input/outputs (GPIOs).

- **MIE: MADCVS Interrupt Enable**

0: Disables smd_irq interrupt generation for MADCVS flag.

1: Enables smd_irq interrupt generation for MADCVS flag.

45. Timer Counter (TC)

45.1 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 multi-purpose 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 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

45.2 Embedded Characteristics

- Total number of TC channels implemented on this device: 9
- TC channel size: 32-bit
- 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 multi-purpose 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

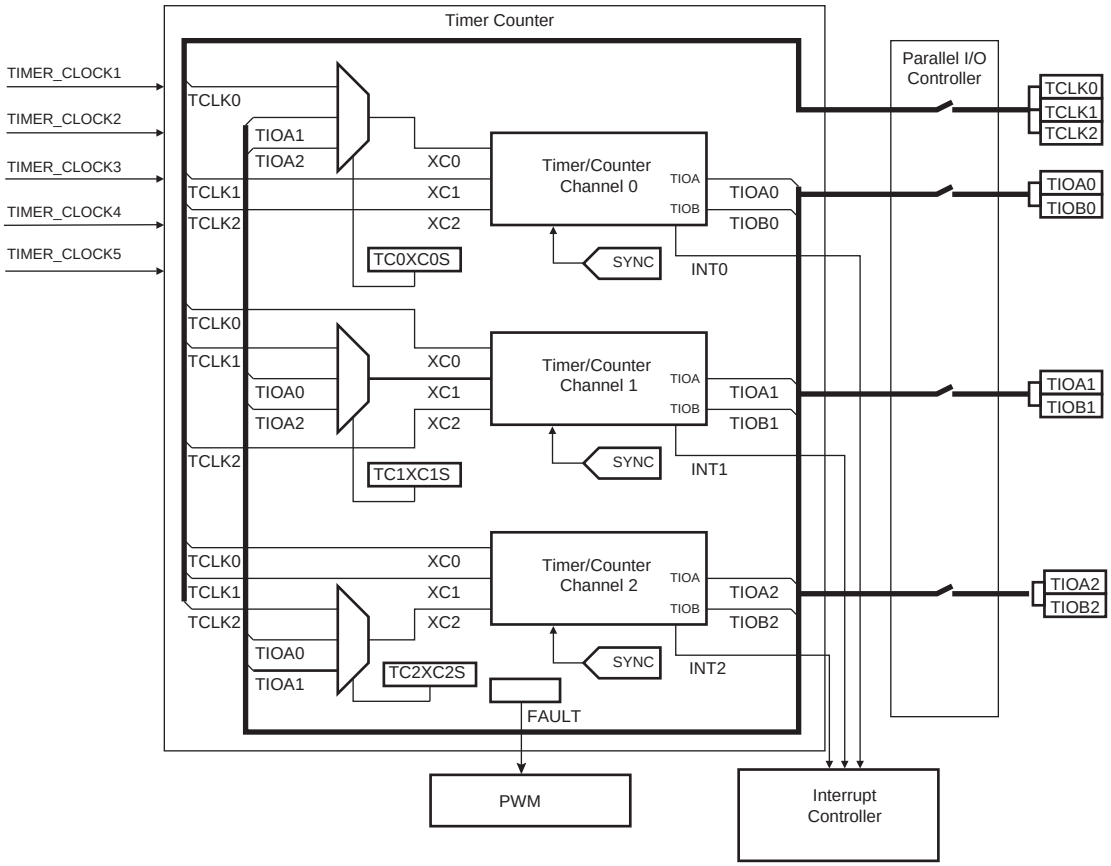
45.3 Block Diagram

Table 45-1. Timer Counter Clock Assignment

Name	Definition
TIMER_CLOCK1	div2
TIMER_CLOCK2	div8
TIMER_CLOCK3	div32
TIMER_CLOCK4	div128
TIMER_CLOCK5(1)	slow_clock

Note: 1. When slow_clock is selected for Peripheral Clock (CSS = 0 in PMC Master Clock Register), slow_clock input is equivalent to Peripheral Clock.

Figure 45-1. Timer Counter Block Diagram



Note: The QDEC connections are detailed in Figure 45-17.

Table 45-2. Channel Signal Description

Signal Name	Description
XC0, XC1, XC2	External Clock Inputs
TIOAx	Capture Mode: Timer Counter Input Waveform Mode: Timer Counter Output

Table 45-2. Channel Signal Description (Continued)

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)

45.4 Pin List

Table 45-3. 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

45.5 Product Dependencies

45.5.1 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.

Table 45-4. I/O Lines

Instance	Signal	I/O Line	Peripheral
TC0	TCLK0	PE17	C
TC0	TCLK1	PE14	B
TC0	TCLK2	PE11	B
TC0	TIOA0	PE15	C
TC0	TIOA1	PE12	B
TC0	TIOA2	PE9	B
TC0	TIOB0	PE16	C
TC0	TIOB1	PE13	B
TC0	TIOB2	PE10	B
TC1	TCLK3	PE8	B
TC1	TCLK4	PE23	B
TC1	TCLK5	PE20	B
TC1	TIOA3	PE6	B
TC1	TIOA4	PE21	B
TC1	TIOA5	PE18	B
TC1	TIOB3	PE7	B
TC1	TIOB4	PE22	B
TC1	TIOB5	PE19	B

45.5.2 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.

45.5.3 Interrupt Sources

The TC has an interrupt line connected to the interrupt controller. Handling the TC interrupt requires programming the interrupt controller before configuring the TC.

Table 45-5. Peripheral IDs

Instance	ID
TC0	40
TC1	41
TC2	42

45.5.4 Synchronization Inputs from PWM

The TC has trigger/capture inputs internally connected to the PWM. Refer to Section 45.6.14 “Synchronization with PWM” and to the implementation of the Pulse Width Modulation (PWM) in this product.

45.5.5 Fault Output

The TC has the FAULT output internally connected to the fault input of PWM. Refer to Section 45.6.18 “Fault Mode” and to the implementation of the Pulse Width Modulation (PWM) in this product.

45.6 Functional Description

45.6.1 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 Table 45-6 “Register Mapping”.

45.6.2 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 TC Status Register (TC_SR) is set.

The current value of the counter is accessible in real time by reading the TC 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.

45.6.3 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 TC Block Mode Register (TC_BMR). Refer to Figure 45-2.

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: div2, div8, div32, div128, slow_clock

This selection is made by the TCCLKS bits in the TC Channel Mode Register (TC_CMR).

The selected clock can be inverted with the CLKI bit in the TC_CMR. 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_CMR defines this signal (none, XC0, XC1, XC2). Refer to Figure 45-3.

Notes:
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_CMR.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 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 45-2. Clock Chaining Selection

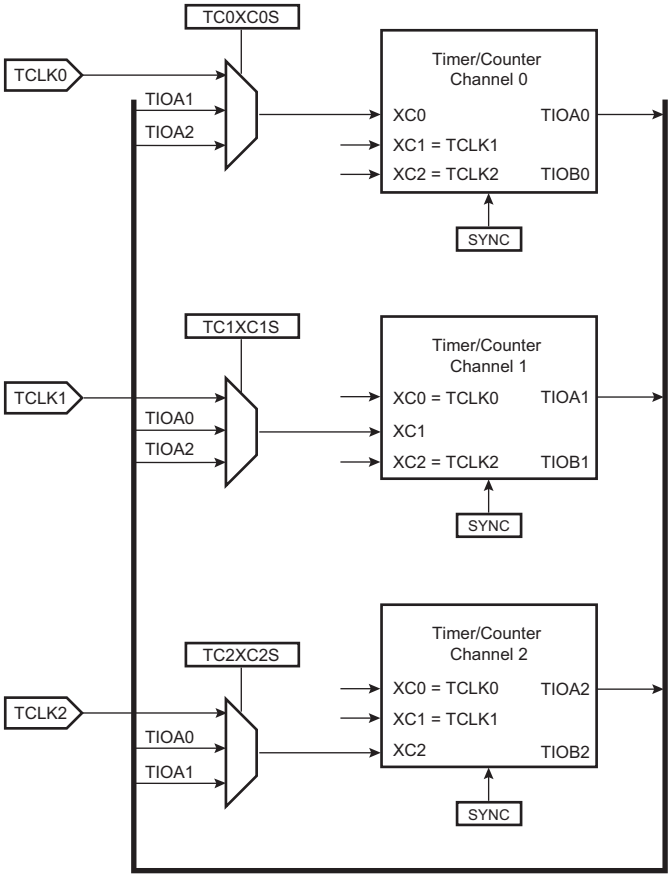
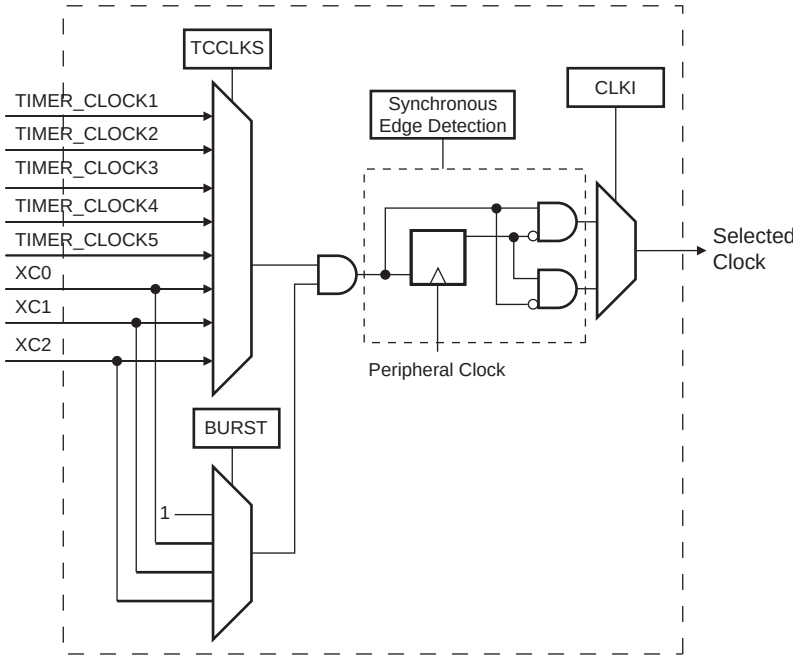


Figure 45-3. Clock Selection

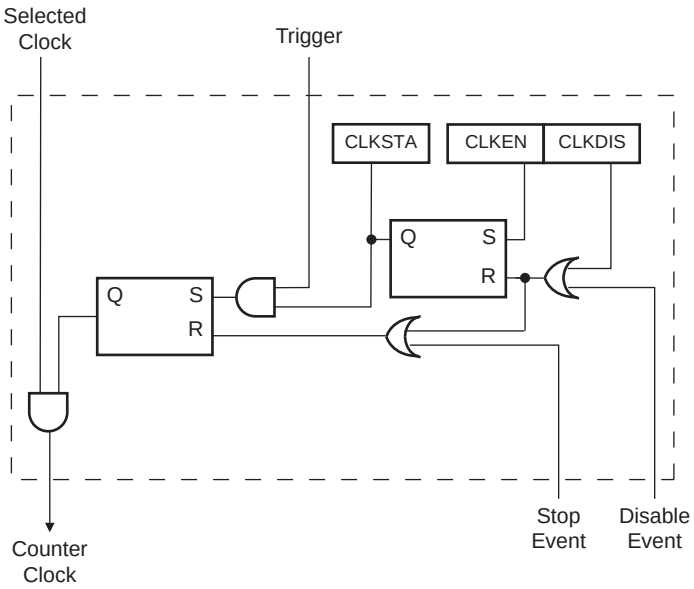


45.6.4 Clock Control

The clock of each counter can be controlled in two different ways: it can be enabled/disabled and started/stopped. Refer to Figure 45-4.

- The clock can be enabled or disabled by the user with the CLKEN and the CLKDIS commands in the TC Channel Control Register (TC_CCR). In Capture mode it can be disabled by an RB load event if LDBDIS is set to 1 in the TC_CMR. In Waveform mode, it can be disabled by an RC Compare event if CPCDIS is set to 1 in TC_CMR. When disabled, the start or the stop actions have no effect: only a CLKEN command in the TC_CCR can re-enable the clock. When the clock is enabled, the CLKSTA bit is set in the TC_SR.
- 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 (LDBSTOP = 1 in TC_CMR) or an RC compare event in Waveform mode (CPCSTOP = 1 in TC_CMR). The start and the stop commands are effective only if the clock is enabled.

Figure 45-4. Clock Control



45.6.5 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 the WAVE bit in the TC_CMR.

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.

45.6.6 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 SWTRG in TC_CCR.
- 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 (Block Control) 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 CPCTRG is set in the TC_CMR.

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 bit ENETRG in the TC_CMR.

If an external trigger is used, the duration of the pulses must be longer than the peripheral clock period in order to be detected.

45.6.7 Capture Mode

Capture mode is entered by clearing the WAVE bit in the TC_CMR.

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.

Figure 45-6 shows the configuration of the TC channel when programmed in Capture mode.

45.6.8 Capture Registers A and B

Registers A and B (RA and RB) are used as capture registers. They can be loaded with the counter value when a programmable event occurs on the signal TIOAx.

The LDRA field in the TC_CMR defines the TIOAx selected edge for the loading of register A, and the LDRB field defines the TIOAx selected edge for the loading of Register B.

The subsampling ratio defined by the SBSMPLR field in TC_CMR 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.

RA is loaded only if it has not been loaded since the last trigger or if RB has been loaded since the last loading of RA.

RB is loaded only if RA has been loaded since the last trigger or the last loading of RB.

Loading RA or RB before the read of the last value loaded sets the Overrun Error Flag (LOVRS bit) in the TC_SR. In this case, the old value is overwritten.

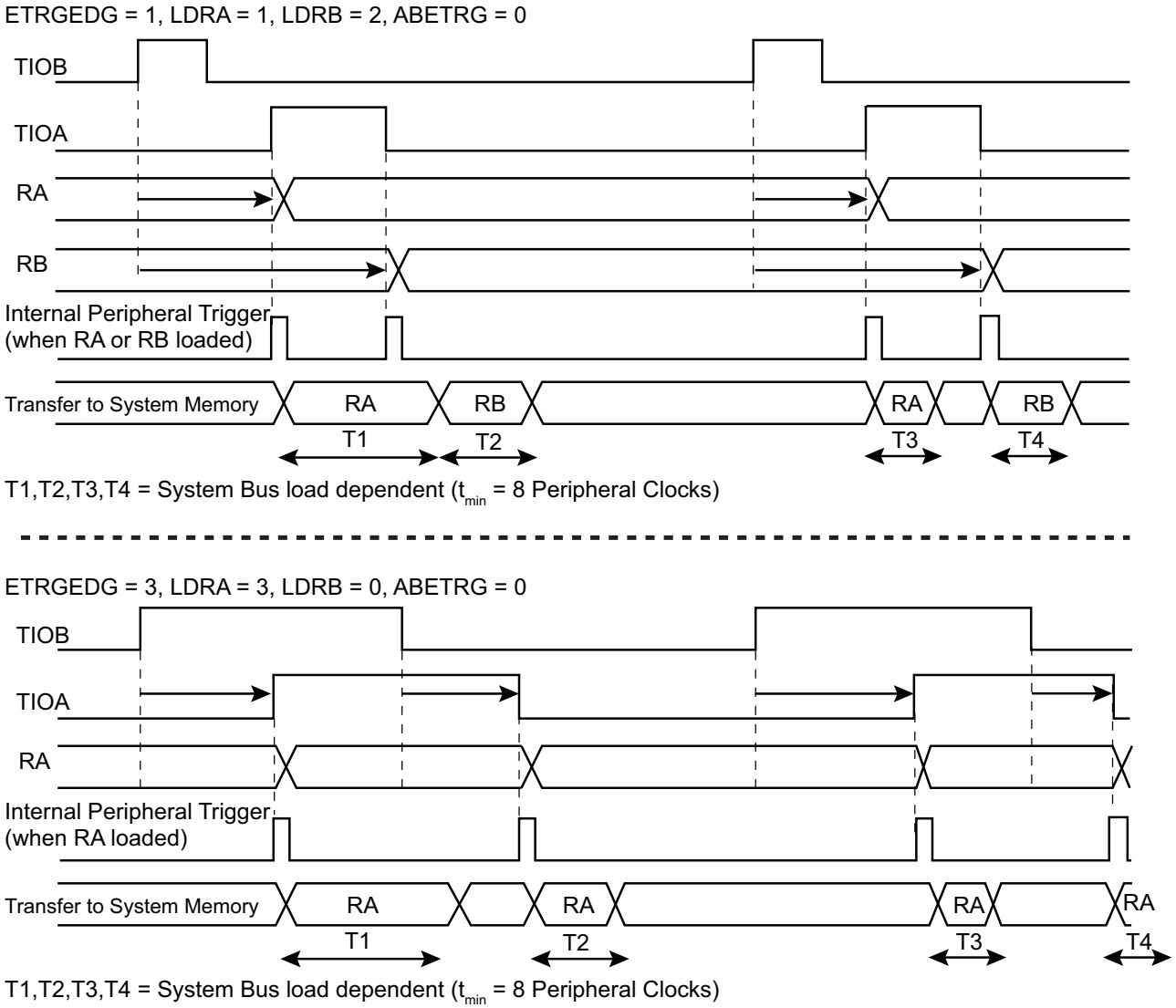
When DMA is used, the RAB register address must be configured as source address of the transfer. The RAB register provides the next unread value from Register A and Register B. It may be read by the DMA after a request has been triggered upon loading Register A or Register B.

45.6.9 Transfer with DMAC in Capture Mode

The DMAC can perform access from the TC to system memory in Capture mode only.

Figure 45-5 illustrates how TC_RA and TC_RB can be loaded in the system memory without CPU intervention.

Figure 45-5. Example of Transfer with DMAC in Capture Mode

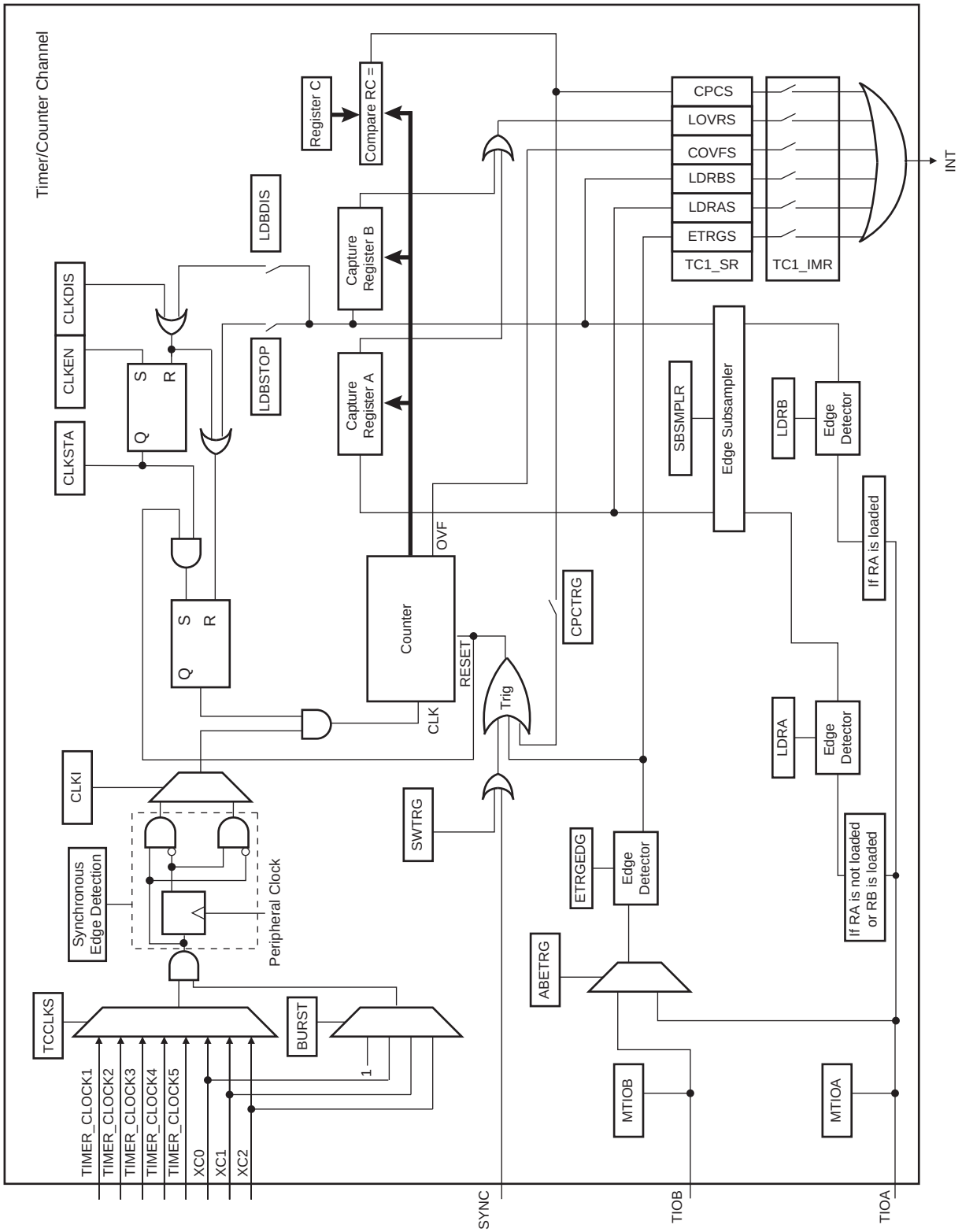


45.6.10 Trigger Conditions

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 45-6. Capture Mode



45.6.11 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).

Figure 45-7 shows the configuration of the TC channel when programmed in Waveform operating mode.

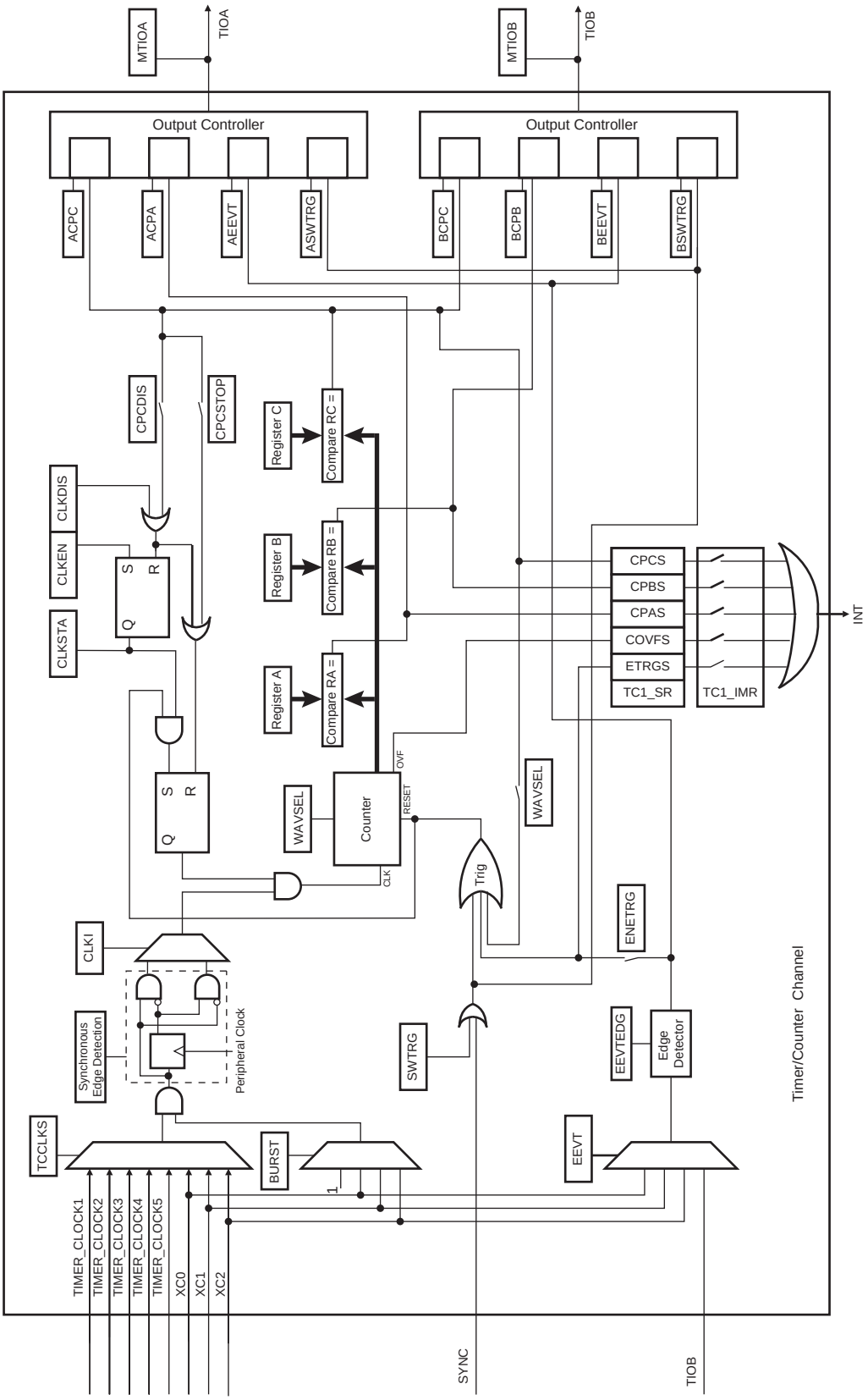
45.6.12 Waveform Selection

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 45-7. Waveform Mode



45.6.12.1WAVSEL = 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. Refer to Figure 45-8.

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 Figure 45-9.

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 45-8. WAVSEL = 00 without Trigger

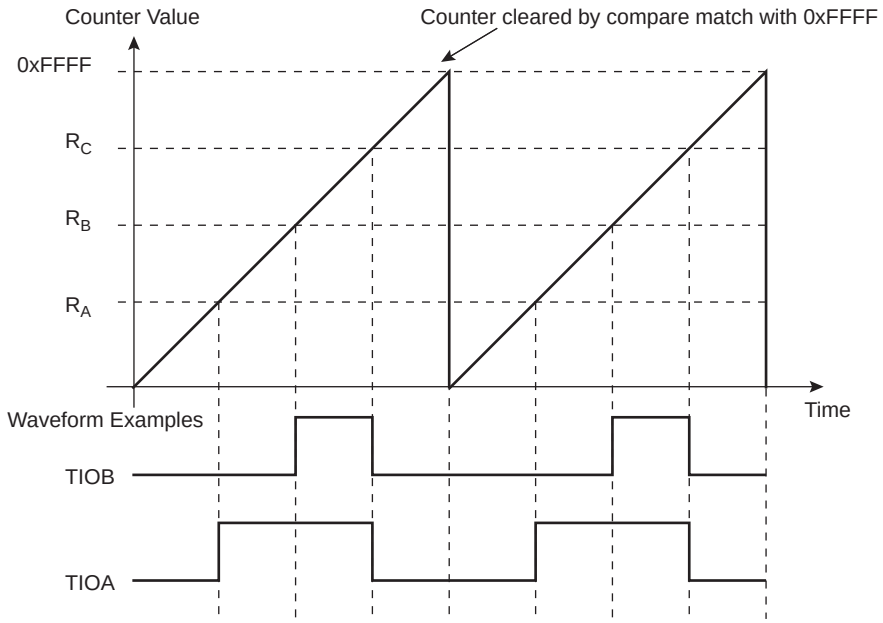
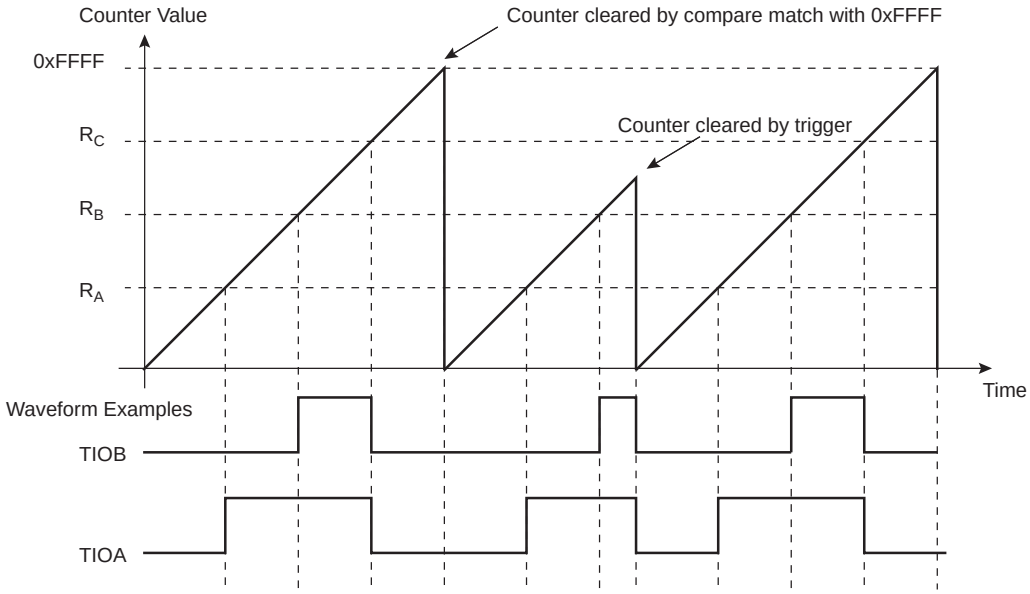


Figure 45-9. WAVSEL = 00 with Trigger



45.6.12.2WAVSEL = 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. Refer to Figure 45-10.

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 Figure 45-11.

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 45-10. WAVSEL = 10 without Trigger

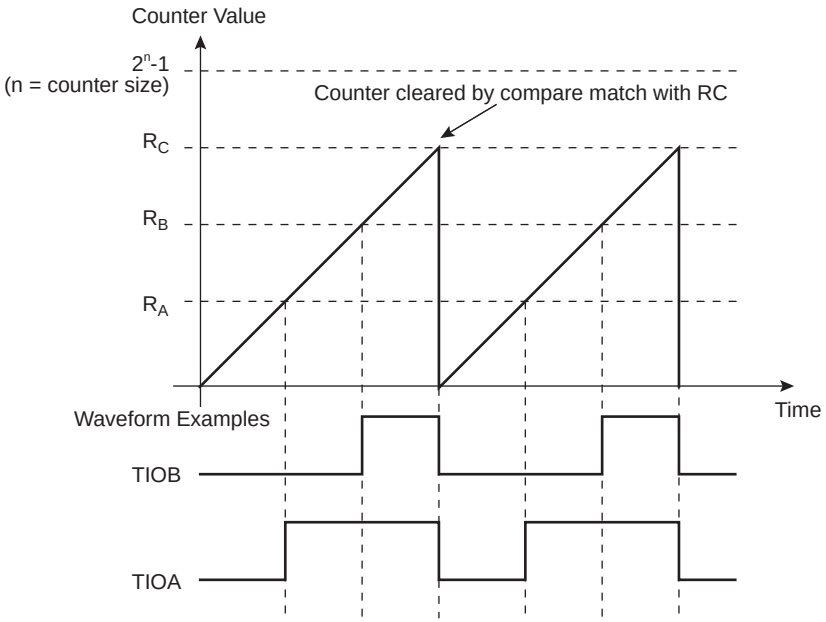
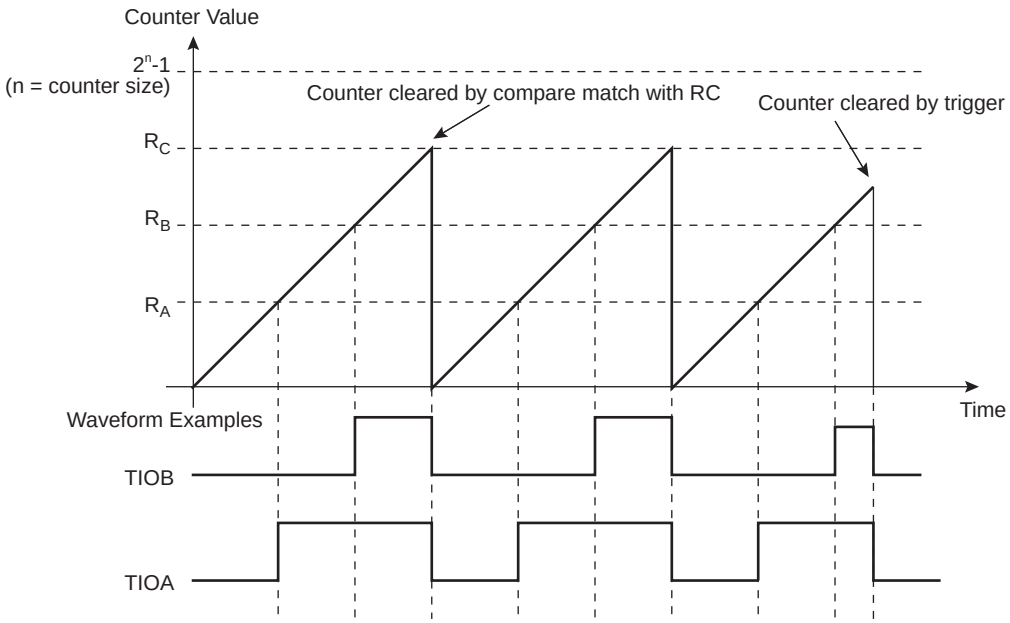


Figure 45-11. WAVSEL = 10 with Trigger



45.6.12.3WAVSEL = 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 re-incremented to 2^{32}-1 and so on. Refer to Figure 45-12.

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 Figure 45-13.

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 45-12. WAVSEL = 01 without Trigger

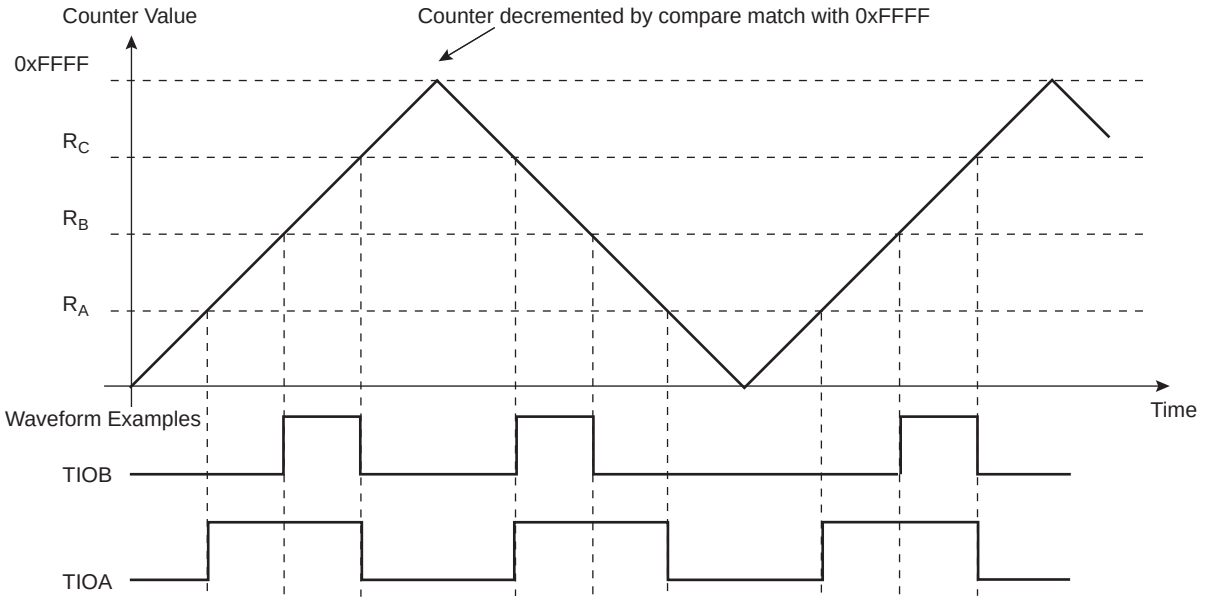
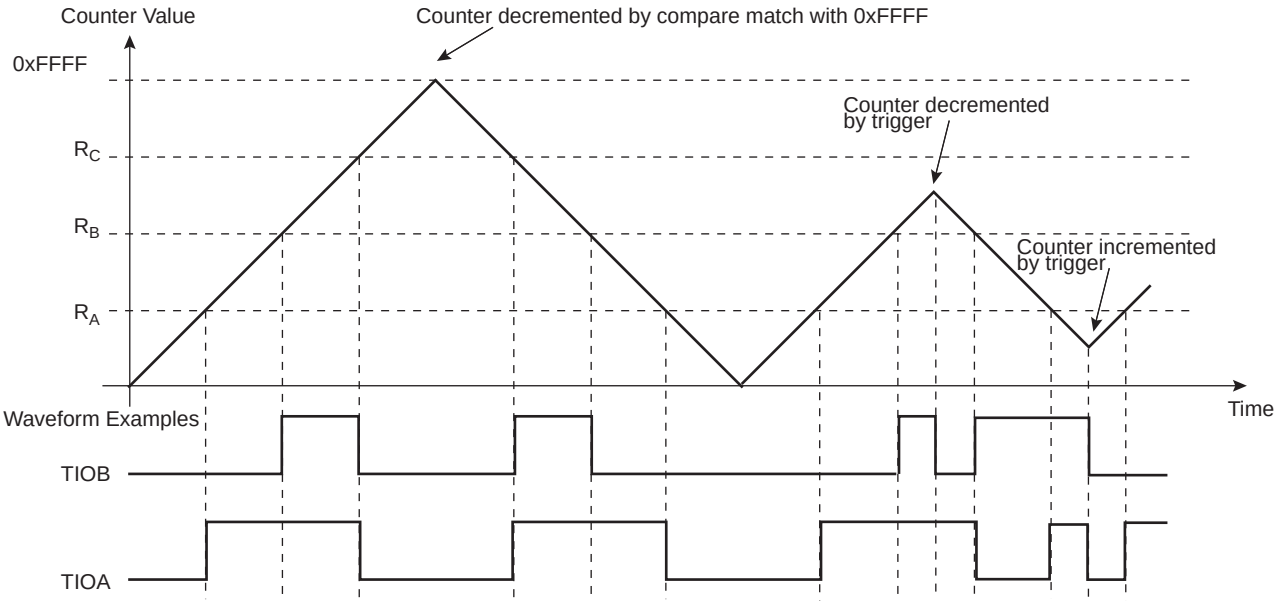


Figure 45-13. WAVSEL = 01 with Trigger



45.6.12.4WAVSEL = 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 re-incremented to RC and so on. Refer to Figure 45-14.

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 Figure 45-15.

RC Compare can stop the counter clock (CPCSTOP = 1) and/or disable the counter clock (CPCDIS = 1).

Figure 45-14. WAVSEL = 11 without Trigger

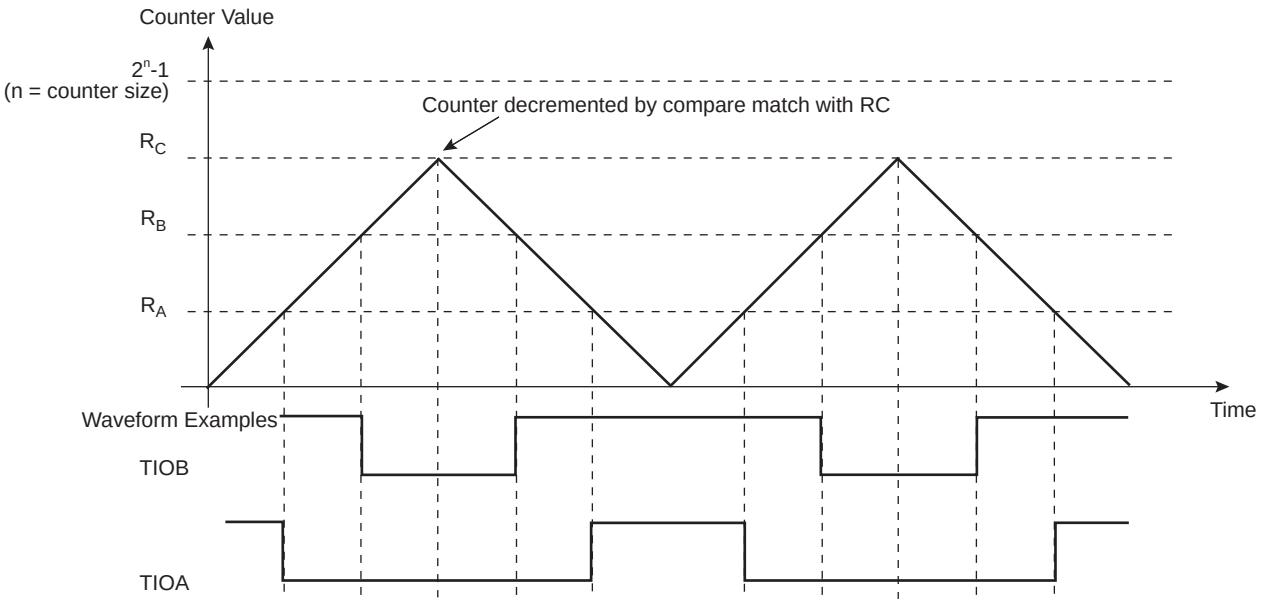
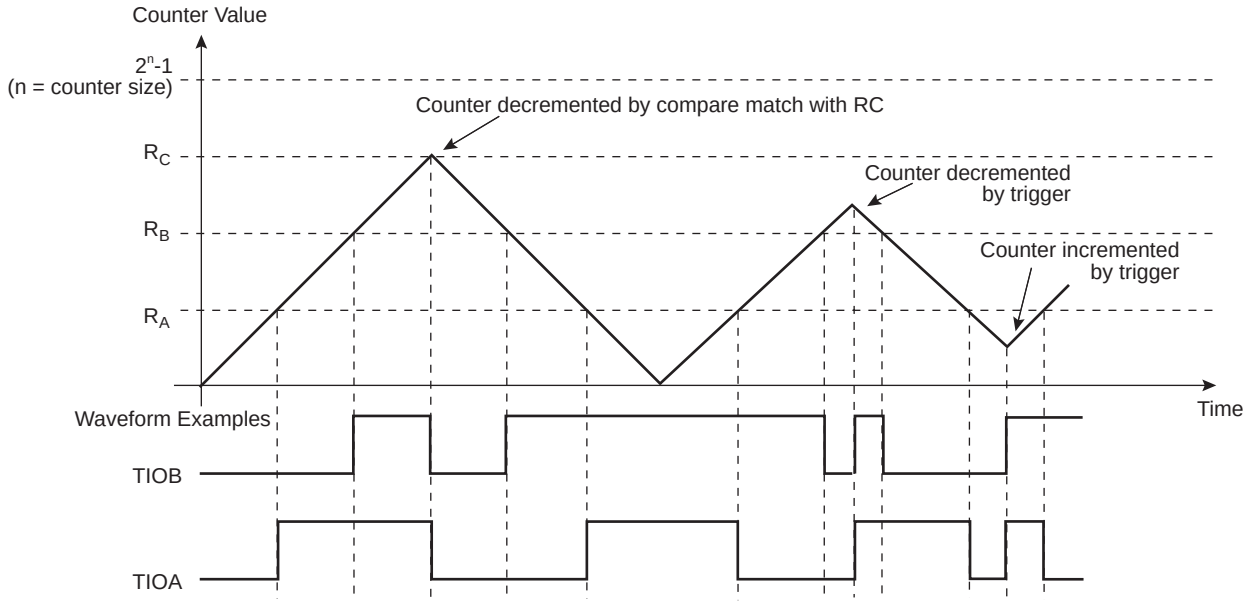


Figure 45-15. WAVSEL = 11 with Trigger



45.6.13 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 EEVT parameter in TC_CMR selects the external trigger. The EEVTEDG parameter defines the trigger edge for each of the possible external triggers (rising, falling or both). 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 bit ENETRG in the TC_CMR.

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.

45.6.14 Synchronization with PWM

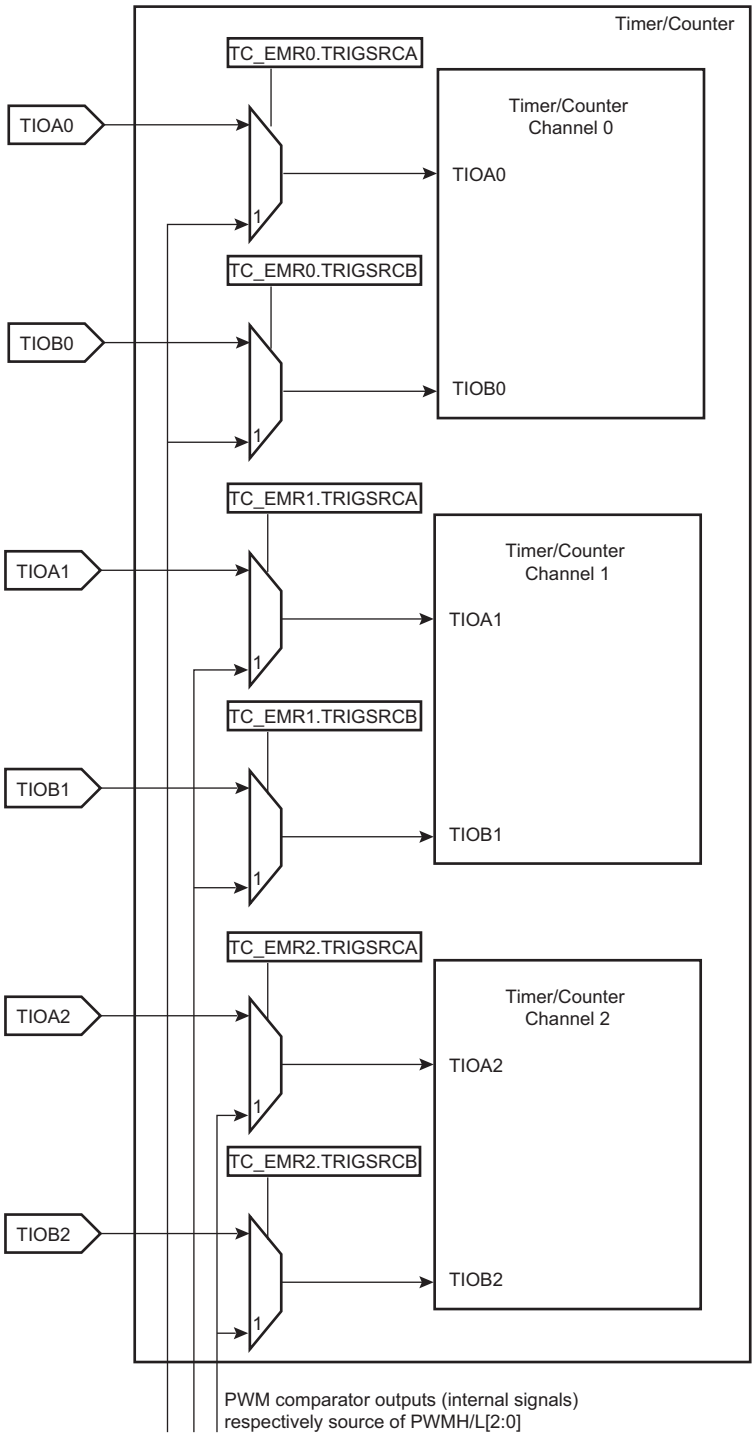
The inputs TIOAx/TIOBx can be bypassed, and thus channel trigger/capture events can be directly driven by the independent PWM module.

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 can be programmed in the Extended Mode Register (TC_EMR) fields TRIGSRCA and TRIGSRCB (refer to Section 45.7.14 “TC Extended Mode Register”).

Each channel of the TC module can be synchronized by a different PWM channel as described in Figure 45-16.

Figure 45-16. Synchronization with PWM



45.6.15 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.

45.6.16 Quadrature Decoder

45.6.16.1Description

The quadrature decoder (QDEC) is driven by TIOA0, TIOB0, 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 Figure 45-17).

When writing a 0 to bit QDEN of the TC_BMR, 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.

Field TCCLKS of TC_CMRx 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.

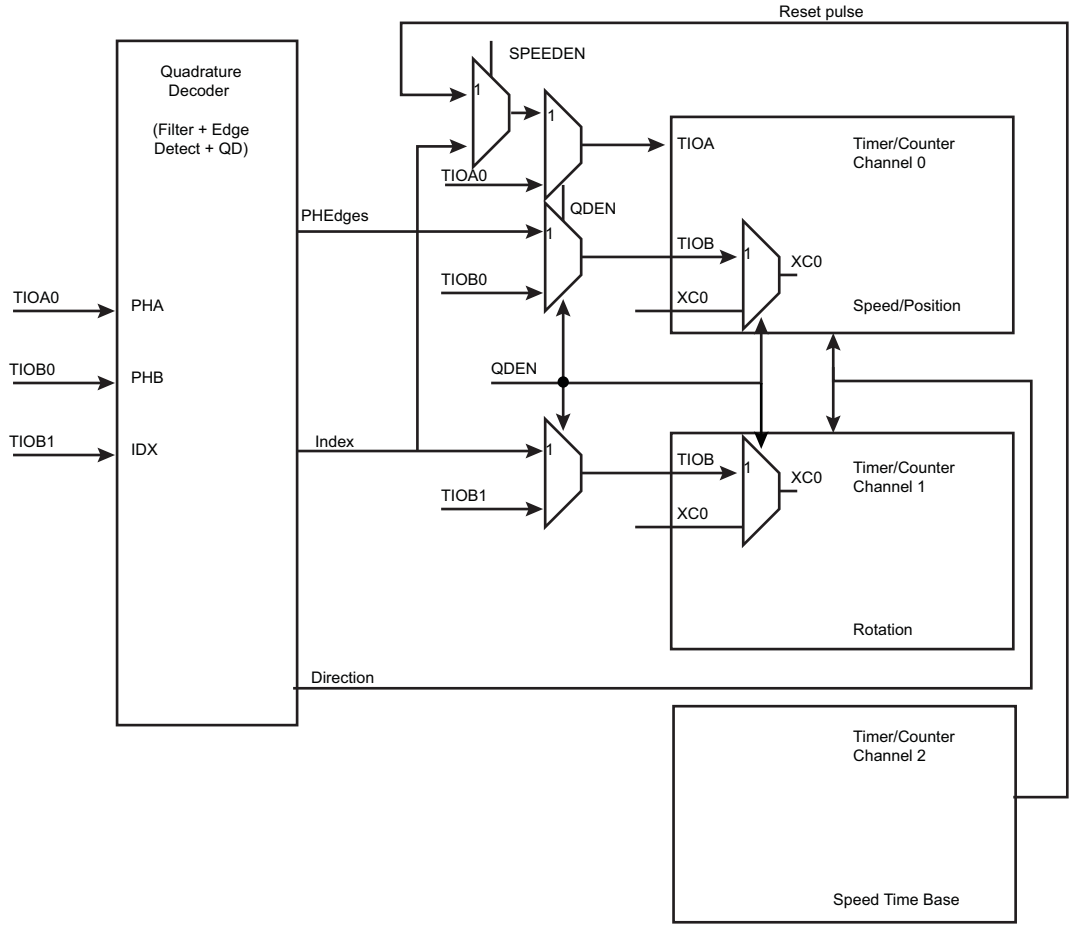
In Speed mode, position cannot be measured but revolution can be measured.

Inputs from the rotary sensor can be filtered prior to down-stream 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 the CPCS flag in the TC_SRx.

Figure 45-17. Predefined Connection of the Quadrature Decoder with Timer Counters



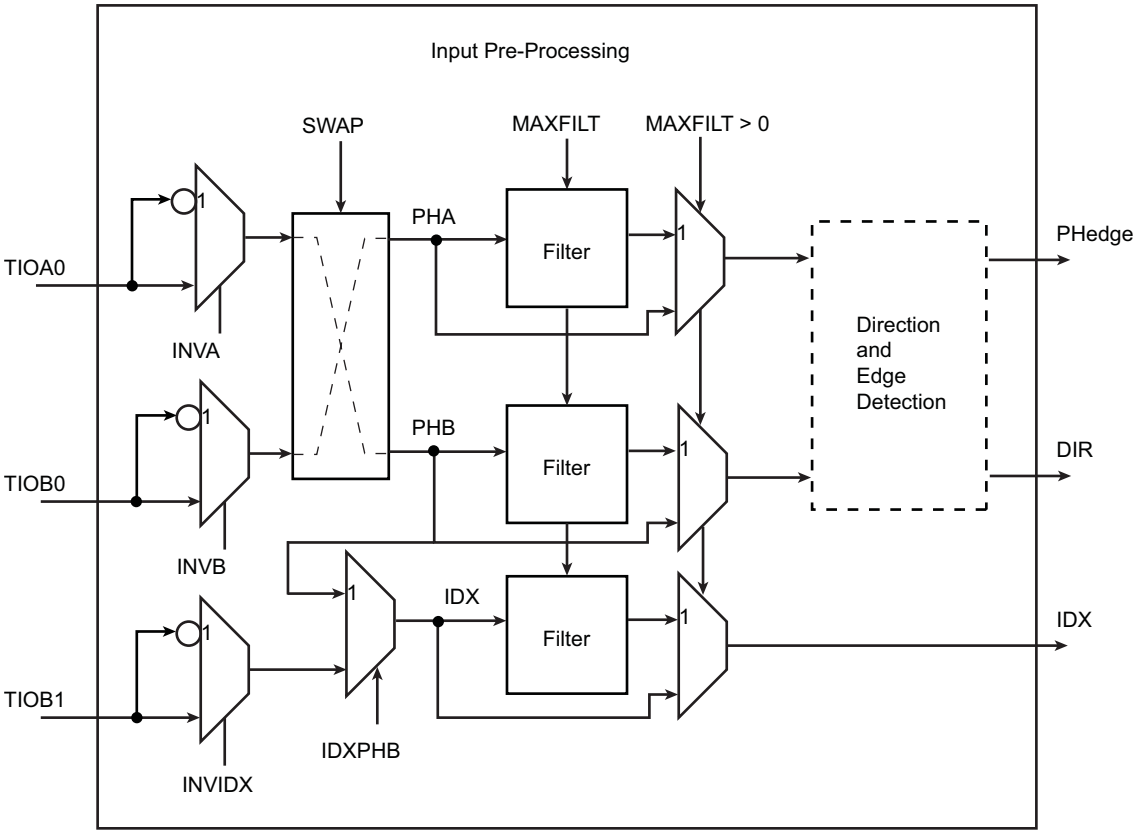
45.6.16.2 Input Pre-processing

Input pre-processing 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.

The MAXFILT field in the TC_BMR 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 $\times$ $t_{peripheral\ clock}$ ns are not passed to downstream logic.

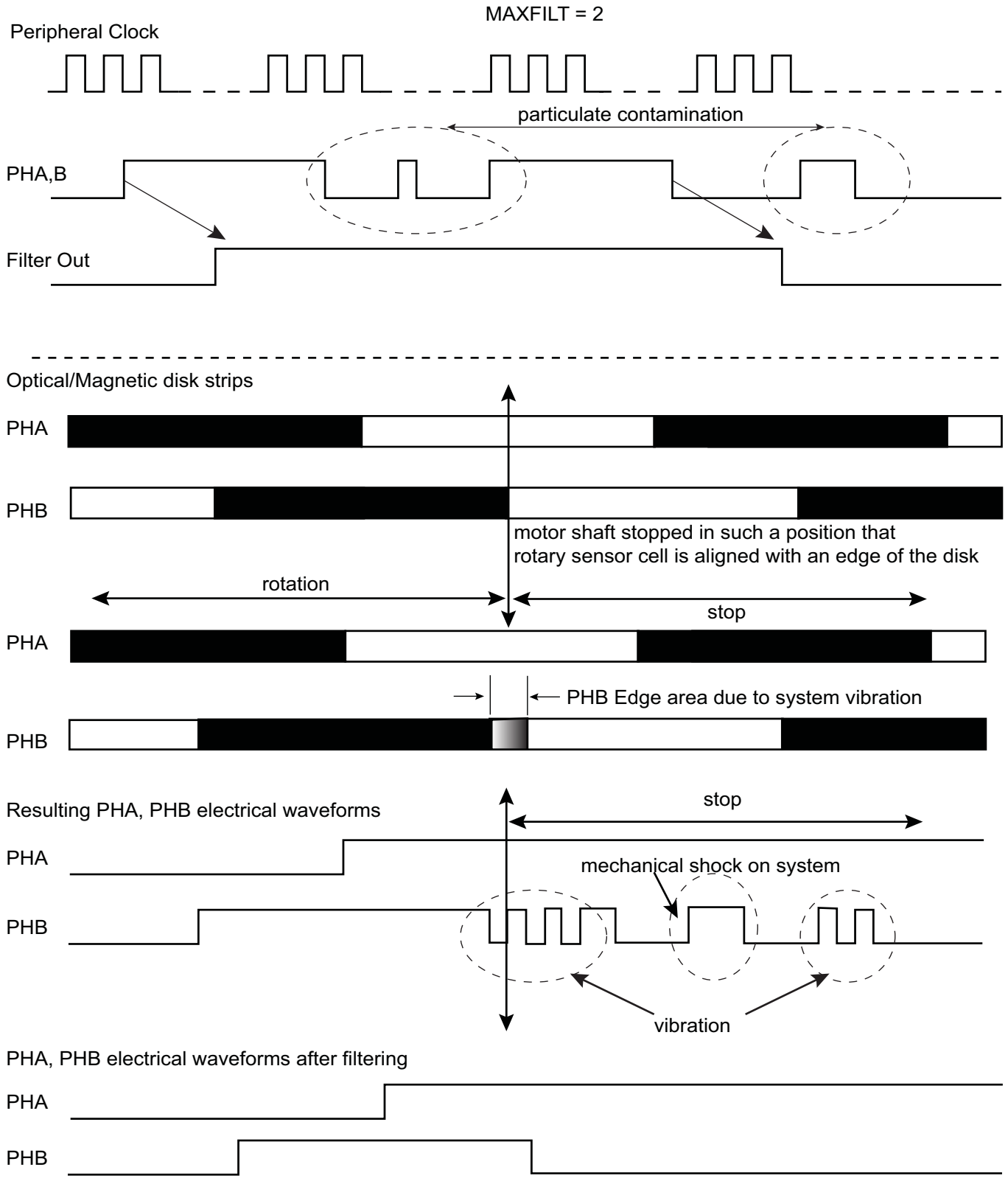
Figure 45-18. Input Stage



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 electro-magnetic 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 45-19. Filtering Examples



45.6.16.3Direction Status and Change Detection

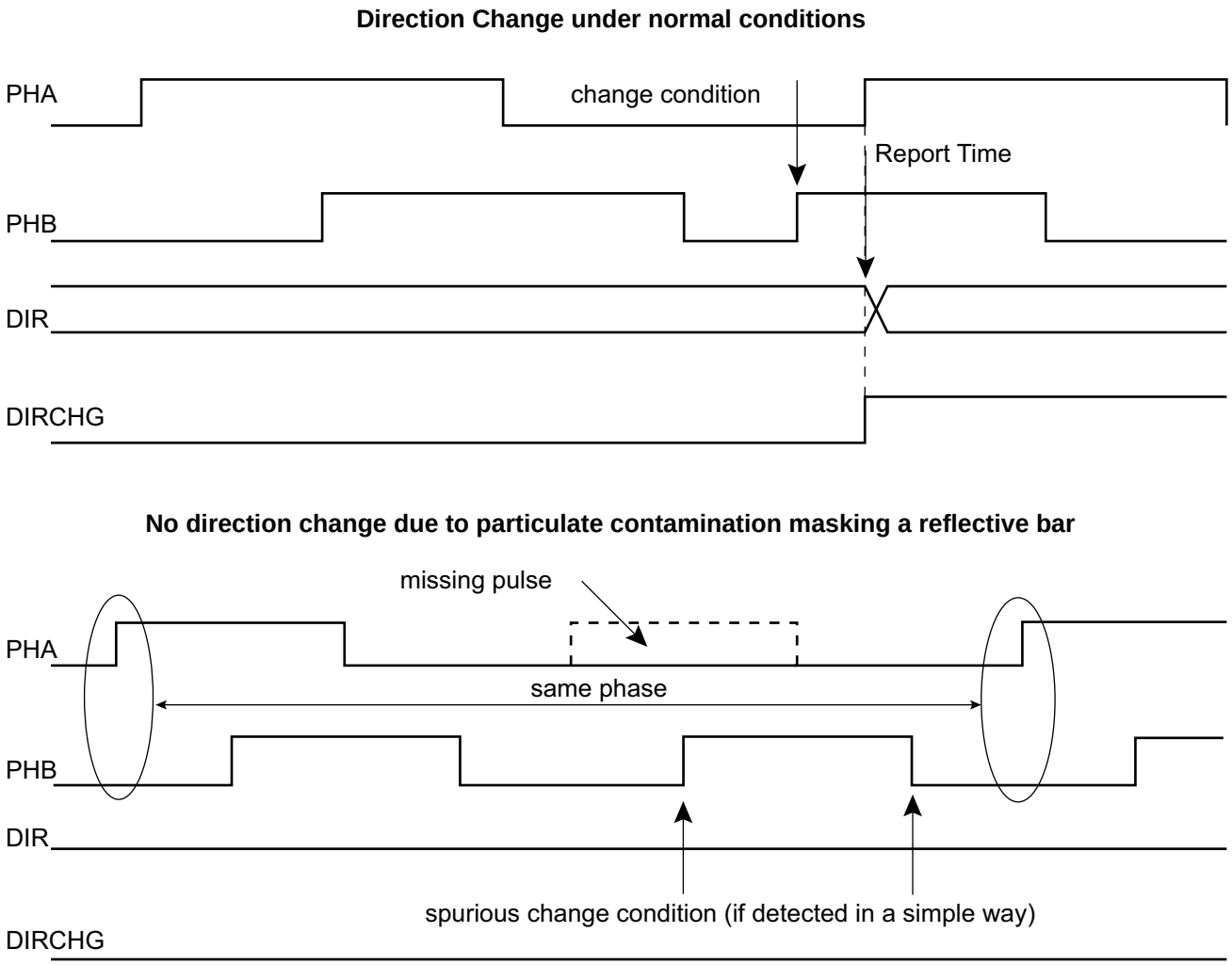
After filtering, the quadrature signals are analyzed to extract the rotation direction and edges of the two quadrature signals detected in order to be counted by timer/counter logic downstream.

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, for the reason that particulate contamination may mask one or more reflective bars on the optical or magnetic disk of the sensor. Refer to Figure 45-20 for waveforms.

Figure 45-20. Rotation Change Detection

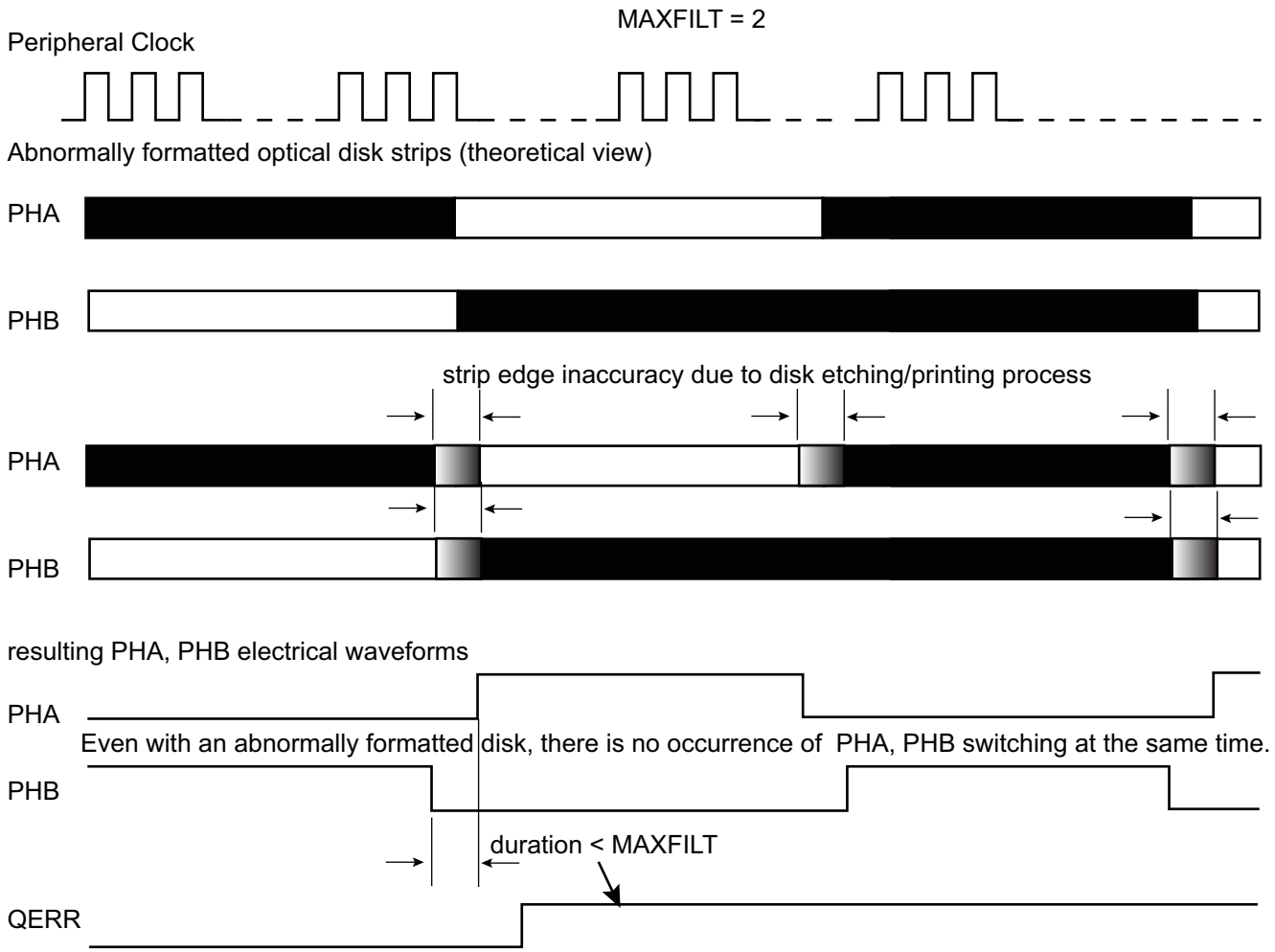


The direction change detection is disabled when QDTRANS is set in the TC_BMR. In this case, the DIR flag report must not be used.

A quadrature error is also reported by the QDEC via the QERR flag in the TC_QISR. 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 (MAXFILT + 1) × $t_{peripheral\ clock}$ ns. After being filtered there is no reason to have two edges closer than (MAXFILT + 1) × $t_{peripheral\ clock}$ ns under normal mode of operation.

Figure 45-21. 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.

45.6.16.4Position and Rotation Measurement

When the POSEN bit is set in the TC_BMR, 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).

In parallel, the number of edges are accumulated on timer/counter 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 timer/counter 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 timer/counter channels 0 and 1. The direction status is reported on TC_QISR.

45.6.16.5 Speed Measurement

When SPEEDEN is set in the TC_BMR, 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). The ABETRG bit of TC_CMR0 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.

45.6.16.6Detecting 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=1028).

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.

45.6.17 2-bit Gray Up/Down Counter for Stepper Motor

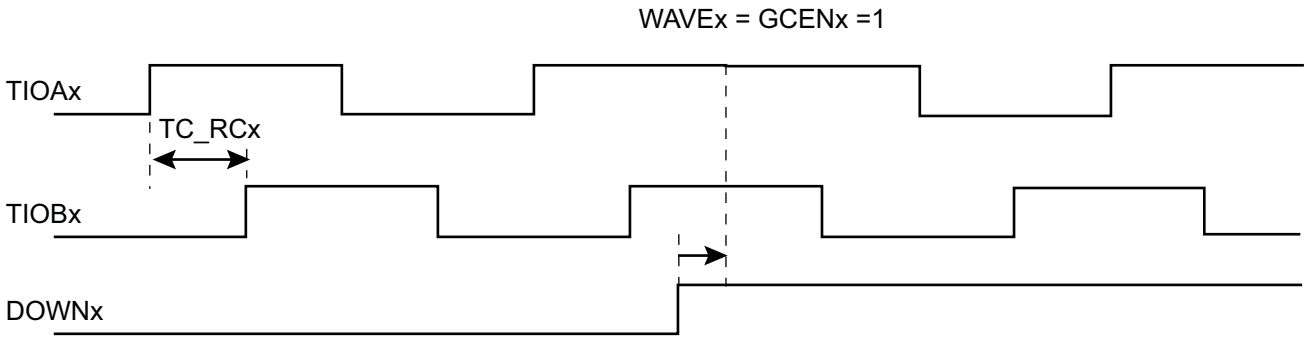
Each channel can be independently configured to generate a 2-bit Gray count waveform on corresponding TIOAx, TIOBx outputs by means of the GCEN bit in TC_SMMRx.

Up or Down count can be defined by writing bit DOWN in TC_SMMRx.

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 45-22. 2-bit Gray Up/Down Counter



45.6.18 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.

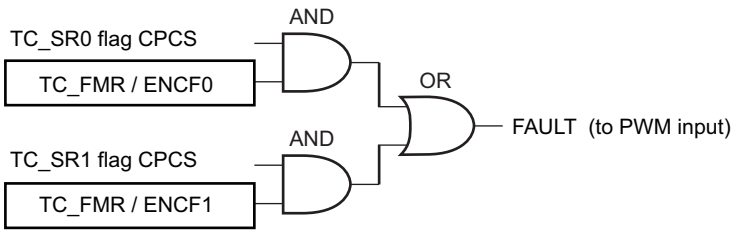
The CPCSx flags can be set accordingly and an interrupt can be generated.

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 45-23. Fault Output Generation



45.6.19 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:

- 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

45.7 Timer Counter (TC) User Interface

Table 45-6. Register Mapping

Offset[1]	Register	Name	Access	Reset
0x00 + channel * 0x40 + 0x00	Channel Control Register	TC_CCR	Write-only	–
0x00 + channel * 0x40 + 0x04	Channel Mode Register	TC_CMR	Read/Write	0
0x00 + channel * 0x40 + 0x08	Stepper Motor Mode Register	TC_SMMR	Read/Write	0
0x00 + channel * 0x40 + 0x0C	Register AB	TC_RAB	Read-only	0
0x00 + channel * 0x40 + 0x10	Counter Value	TC_CV	Read-only	0
0x00 + channel * 0x40 + 0x14	Register A	TC_RA	Read/Write[2]	0
0x00 + channel * 0x40 + 0x18	Register B	TC_RB	Read/Write[2]	0
0x00 + channel * 0x40 + 0x1C	Register C	TC_RC	Read/Write	0
0x00 + channel * 0x40 + 0x20	Status Register	TC_SR	Read-only	0
0x00 + channel * 0x40 + 0x24	Interrupt Enable Register	TC_IER	Write-only	–
0x00 + channel * 0x40 + 0x28	Interrupt Disable Register	TC_IDR	Write-only	–
0x00 + channel * 0x40 + 0x2C	Interrupt Mask Register	TC_IMR	Read-only	0
0x00 + channel * 0x40 + 0x30	Extended Mode Register	TC_EMR	Read/Write	0
0xC0	Block Control Register	TC_BCR	Write-only	–
0xC4	Block Mode Register	TC_BMR	Read/Write	0
0xC8	QDEC Interrupt Enable Register	TC_QIER	Write-only	–
0xCC	QDEC Interrupt Disable Register	TC_QIDR	Write-only	–
0xD0	QDEC Interrupt Mask Register	TC_QIMR	Read-only	0
0xD4	QDEC Interrupt Status Register	TC_QISR	Read-only	0
0xD8	Fault Mode Register	TC_FMR	Read/Write	0
0xE4	Write Protection Mode Register	TC_WPMR	Read/Write	0
0xE8–0xFC	Reserved	–	–	–

Notes: 1. Channel index ranges from 0 to 2.
2. Read-only if TC_CMRx.WAVE = 0

45.7.1 TC Channel Control Register

Name: TC_CCRx [x=0..2]

Address: 0xF801C000 (0)[0], 0xF801C040 (0)[1], 0xF801C080 (0)[2], 0xFC020000 (1)[0], 0xFC020040 (1)[1], 0xFC020080 (1)[2], 0xFC024000 (2)[0], 0xFC024040 (2)[1], 0xFC024080 (2)[2]

Access: Write-only

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
–	–	–	–	–	SWTRG	CLKDIS	CLKEN

- **CLKEN: Counter Clock Enable Command**

0: No effect.

1: Enables the clock if CLKDIS is not 1.

- **CLKDIS: Counter Clock Disable Command**

0: No effect.

1: Disables the clock.

- **SWTRG: Software Trigger Command**

0: No effect.

1: A software trigger is performed: the counter is reset and the clock is started.

45.7.2 TC Channel Mode Register: Capture Mode

Name: TC_CMRx [x=0..2] (CAPTURE_MODE)

Address: 0xF801C004 (0)[0], 0xF801C044 (0)[1], 0xF801C084 (0)[2], 0xFC020004 (1)[0], 0xFC020044 (1)[1], 0xFC020084 (1)[2], 0xFC024004 (2)[0], 0xFC024044 (2)[1], 0xFC024084 (2)[2]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	SBSMPLR			LDRB		LDRA	

15	14	13	12	11	10	9	8
WAVE	CPCTRG	–	–	–	ABETRG	ETRGEDG	

7	6	5	4	3	2	1	0
LDBDIS	LDBSTOP	BURST		CLKI	TCCLKS		

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

- **TCCLKS: Clock Selection**

Value	Name	Description
0	TIMER_CLOCK1	Clock selected: internal div2 clock signal (from PMC)
1	TIMER_CLOCK2	Clock selected: internal div8 clock signal (from PMC)
2	TIMER_CLOCK3	Clock selected: internal div32 clock signal (from PMC)
3	TIMER_CLOCK4	Clock selected: internal div128 clock signal (from PMC)
4	TIMER_CLOCK5	Clock selected: internal slow_clock clock signal (from PMC)
5	XC0	Clock selected: XC0
6	XC1	Clock selected: XC1
7	XC2	Clock selected: XC2

To operate at maximum peripheral clock frequency, refer to Section 45.7.14 “TC Extended Mode Register”.

- **CLKI: Clock Invert**

0: Counter is incremented on rising edge of the clock.

1: Counter is incremented on falling edge of the clock.

- **BURST: 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.

- **LDBSTOP: Counter Clock Stopped with RB Loading**

0: Counter clock is not stopped when RB loading occurs.

1: Counter clock is stopped when RB loading occurs.

- **LDBDIS: Counter Clock Disable with RB Loading**

0: Counter clock is not disabled when RB loading occurs.

1: Counter clock is disabled when RB loading occurs.

- **ETRGEDG: 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

- **ABETRG: TIOAx or TIOBx External Trigger Selection**

0: TIOBx is used as an external trigger.

1: TIOAx is used as an external trigger.

- **CPCTRG: RC Compare Trigger Enable**

0: RC Compare has no effect on the counter and its clock.

1: RC Compare resets the counter and starts the counter clock.

- **WAVE: Waveform Mode**

0: Capture mode is enabled.

1: Capture mode is disabled (Waveform mode is enabled).

- **LDRA: 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

- **LDRB: 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

- **SBSMPLR: 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

45.7.3 TC Channel Mode Register: Waveform Mode

Name: TC_CMRx [x=0..2] (WAVEFORM_MODE)

Address: 0xF801C004 (0)[0], 0xF801C044 (0)[1], 0xF801C084 (0)[2], 0xFC020004 (1)[0], 0xFC020044 (1)[1], 0xFC020084 (1)[2], 0xFC024004 (2)[0], 0xFC024044 (2)[1], 0xFC024084 (2)[2]

Access: Read/Write

31	30	29	28	27	26	25	24
BSWTRG		BEEVT		BCPC		BCPB	

23	22	21	20	19	18	17	16
ASWTRG		AEEVT		ACPC		ACPA	

15	14	13	12	11	10	9	8
WAVE	WAVSEL		ENETRG	EEVT		EEVTEDG	

7	6	5	4	3	2	1	0
CPCDIS	CPCSTOP	BURST		CLKI	TCCLKS		

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

- **TCCLKS: Clock Selection**

Value	Name	Description
0	TIMER_CLOCK1	Clock selected: internal div2 clock signal (from PMC)
1	TIMER_CLOCK2	Clock selected: internal div8 clock signal (from PMC)
2	TIMER_CLOCK3	Clock selected: internal div32 clock signal (from PMC)
3	TIMER_CLOCK4	Clock selected: internal div128 clock signal (from PMC)
4	TIMER_CLOCK5	Clock selected: internal slow_clock clock signal (from PMC)
5	XC0	Clock selected: XC0
6	XC1	Clock selected: XC1
7	XC2	Clock selected: XC2

To operate at maximum peripheral clock frequency, refer to Section 45.7.14 “TC Extended Mode Register”.

- **CLKI: Clock Invert**

0: Counter is incremented on rising edge of the clock.

1: Counter is incremented on falling edge of the clock.

- **BURST: 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.

- **CPCSTOP: Counter Clock Stopped with RC Compare**

0: Counter clock is not stopped when counter reaches RC.

1: Counter clock is stopped when counter reaches RC.

- **CPCDIS: Counter Clock Disable with RC Compare**

0: Counter clock is not disabled when counter reaches RC.

1: Counter clock is disabled when counter reaches RC.

- **EEVTEDG: External Event Edge Selection**

Value	Name	Description
0	NONE	None
1	RISING	Rising edge
2	FALLING	Falling edge
3	EDGE	Each edge

- **EEVT: External Event Selection**

Signal selected as external event.

Value	Name	Description	TIOB Direction
0	TIOB	TIOB(1)	Input
1	XC0	XC0	Output
2	XC1	XC1	Output
3	XC2	XC2	Output

Note: 1. If TIOB is chosen as the external event signal, it is configured as an input and no longer generates waveforms and subsequently no IRQs.

- **ENETRG: External Event Trigger Enable**

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.

Note: 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).

- **WAVSEL: 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

- **WAVE: Waveform Mode**

0: Waveform mode is disabled (Capture mode is enabled).

1: Waveform mode is enabled.

- **ACPA: RA Compare Effect on TIOAx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **ACPC: RC Compare Effect on TIOAx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **AEEVT: External Event Effect on TIOAx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **ASWTRG: Software Trigger Effect on TIOAx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **BCPB: RB Compare Effect on TIOBx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **BCPC: RC Compare Effect on TIOBx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **BEEVT: External Event Effect on TIOBx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **BSWTRG: Software Trigger Effect on TIOBx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

45.7.4 TC Stepper Motor Mode Register

Name: TC_SMMRx [x=0..2]

Address: 0xF801C008 (0)[0], 0xF801C048 (0)[1], 0xF801C088 (0)[2], 0xFC020008 (1)[0], 0xFC020048 (1)[1], 0xFC020088 (1)[2], 0xFC024008 (2)[0], 0xFC024048 (2)[1], 0xFC024088 (2)[2]

Access: Read/Write

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
–	–	–	–	–	–	DOWN	GCEN

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

- **GCEN: Gray Count Enable**

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.

- **DOWN: Down Count**

0: Up counter.

1: Down counter.

45.7.5 TC Register AB

Name: TC_RABx [x=0..2]

Address: 0xF801C00C (0)[0], 0xF801C04C (0)[1], 0xF801C08C (0)[2], 0xFC02000C (1)[0], 0xFC02004C (1)[1], 0xFC02008C (1)[2], 0xFC02400C (2)[0], 0xFC02404C (2)[1], 0xFC02408C (2)[2]

Access: Read-only

31	30	29	28	27	26	25	24
RAB							

23	22	21	20	19	18	17	16
RAB							

15	14	13	12	11	10	9	8
RAB							

7	6	5	4	3	2	1	0
RAB							

- **RAB: 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.

45.7.6 TC Counter Value Register

Name: TC_CVx [x=0..2]

Address: 0xF801C010 (0)[0], 0xF801C050 (0)[1], 0xF801C090 (0)[2], 0xFC020010 (1)[0], 0xFC020050 (1)[1], 0xFC020090 (1)[2], 0xFC024010 (2)[0], 0xFC024050 (2)[1], 0xFC024090 (2)[2]

Access: Read-only

31	30	29	28	27	26	25	24
CV							

23	22	21	20	19	18	17	16
CV							

15	14	13	12	11	10	9	8
CV							

7	6	5	4	3	2	1	0
CV							

- **CV: Counter Value**

CV contains the counter value in real time.

45.7.7 TC Register A

Name: TC_RAx [x=0..2]

Address: 0xF801C014 (0)[0], 0xF801C054 (0)[1], 0xF801C094 (0)[2], 0xFC020014 (1)[0], 0xFC020054 (1)[1], 0xFC020094 (1)[2], 0xFC024014 (2)[0], 0xFC024054 (2)[1], 0xFC024094 (2)[2]

Access: Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1

31	30	29	28	27	26	25	24
RA							

23	22	21	20	19	18	17	16
RA							

15	14	13	12	11	10	9	8
RA							

7	6	5	4	3	2	1	0
RA							

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

- **RA: Register A**

RA contains the Register A value in real time.

45.7.8 TC Register B

Name: TC_RBx [x=0..2]

Address: 0xF801C018 (0)[0], 0xF801C058 (0)[1], 0xF801C098 (0)[2], 0xFC020018 (1)[0], 0xFC020058 (1)[1], 0xFC020098 (1)[2], 0xFC024018 (2)[0], 0xFC024058 (2)[1], 0xFC024098 (2)[2]

Access: Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1

31	30	29	28	27	26	25	24
RB							

23	22	21	20	19	18	17	16
RB							

15	14	13	12	11	10	9	8
RB							

7	6	5	4	3	2	1	0
RB							

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

- **RB: Register B**

RB contains the Register B value in real time.

45.7.9 TC Register C

Name: TC_RCx [x=0..2]

Address: 0xF801C01C (0)[0], 0xF801C05C (0)[1], 0xF801C09C (0)[2], 0xFC02001C (1)[0], 0xFC02005C (1)[1], 0xFC02009C (1)[2], 0xFC02401C (2)[0], 0xFC02405C (2)[1], 0xFC02409C (2)[2]

Access: Read/Write

31	30	29	28	27	26	25	24
RC							

23	22	21	20	19	18	17	16
RC							

15	14	13	12	11	10	9	8
RC							

7	6	5	4	3	2	1	0
RC							

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

- **RC: Register C**

RC contains the Register C value in real time.

45.7.10 TC Status Register

Name: TC_SRx [x=0..2]

Address: 0xF801C020 (0)[0], 0xF801C060 (0)[1], 0xF801C0A0 (0)[2], 0xFC020020 (1)[0], 0xFC020060 (1)[1], 0xFC0200A0 (1)[2], 0xFC024020 (2)[0], 0xFC024060 (2)[1], 0xFC0240A0 (2)[2]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	MTIOB	MTIOA	CLKSTA
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow Status (cleared on read)**

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.

- **LOVRS: Load Overrun Status (cleared on read)**

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.

- **CPAS: RA Compare Status (cleared on read)**

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.

- **CPBS: RB Compare Status (cleared on read)**

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.

- **CPCS: RC Compare Status (cleared on read)**

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.

- **LDRAS: RA Loading Status (cleared on read)**

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.

- **LDRBS: RB Loading Status (cleared on read)**

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.

- **ETRGS: External Trigger Status (cleared on read)**

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.

- **CLKSTA: Clock Enabling Status**

0: Clock is disabled.

1: Clock is enabled.

- **MTIOA: TIOAx Mirror**

0: TIOAx is low. If TC_CMRx.WAVE = 0, this means that TIOAx pin is low. If TC_CMRx.WAVE = 1, this means that TIOAx is driven low.

1: TIOAx is high. If TC_CMRx.WAVE = 0, this means that TIOAx pin is high. If TC_CMRx.WAVE = 1, this means that TIOAx is driven high.

- **MTIOB: TIOBx Mirror**

0: TIOBx is low. If TC_CMRx.WAVE = 0, this means that TIOBx pin is low. If TC_CMRx.WAVE = 1, this means that TIOBx is driven low.

1: TIOBx is high. If TC_CMRx.WAVE = 0, this means that TIOBx pin is high. If TC_CMRx.WAVE = 1, this means that TIOBx is driven high.

45.7.11 TC Interrupt Enable Register

Name: TC_IERx [x=0..2]

Address: 0xF801C024 (0)[0], 0xF801C064 (0)[1], 0xF801C0A4 (0)[2], 0xFC020024 (1)[0], 0xFC020064 (1)[1], 0xFC0200A4 (1)[2], 0xFC024024 (2)[0], 0xFC024064 (2)[1], 0xFC0240A4 (2)[2]

Access: Write-only

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
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**

0: No effect.

1: Enables the Counter Overflow Interrupt.

- **LOVRS: Load Overrun**

0: No effect.

1: Enables the Load Overrun Interrupt.

- **CPAS: RA Compare**

0: No effect.

1: Enables the RA Compare Interrupt.

- **CPBS: RB Compare**

0: No effect.

1: Enables the RB Compare Interrupt.

- **CPCS: RC Compare**

0: No effect.

1: Enables the RC Compare Interrupt.

- **LDRAS: RA Loading**

0: No effect.

1: Enables the RA Load Interrupt.

- **LDRBS: RB Loading**

0: No effect.

1: Enables the RB Load Interrupt.

- **ETRGS: External Trigger**

0: No effect.

1: Enables the External Trigger Interrupt.

45.7.12 TC Interrupt Disable Register

Name: TC_IDRx [x=0..2]

Address: 0xF801C028 (0)[0], 0xF801C068 (0)[1], 0xF801C0A8 (0)[2], 0xFC020028 (1)[0], 0xFC020068 (1)[1], 0xFC0200A8 (1)[2], 0xFC024028 (2)[0], 0xFC024068 (2)[1], 0xFC0240A8 (2)[2]

Access: Write-only

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
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**

0: No effect.

1: Disables the Counter Overflow Interrupt.

- **LOVRS: Load Overrun**

0: No effect.

1: Disables the Load Overrun Interrupt (if TC_CMRx.WAVE = 0).

- **CPAS: RA Compare**

0: No effect.

1: Disables the RA Compare Interrupt (if TC_CMRx.WAVE = 1).

- **CPBS: RB Compare**

0: No effect.

1: Disables the RB Compare Interrupt (if TC_CMRx.WAVE = 1).

- **CPCS: RC Compare**

0: No effect.

1: Disables the RC Compare Interrupt.

- **LDRAS: RA Loading**

0: No effect.

1: Disables the RA Load Interrupt (if TC_CMRx.WAVE = 0).

- **LDRBS: RB Loading**

0: No effect.

1: Disables the RB Load Interrupt (if TC_CMRx.WAVE = 0).

- **ETRGS: External Trigger**

0: No effect.

1: Disables the External Trigger Interrupt.

45.7.13 TC Interrupt Mask Register

Name: TC_IMRx [x=0..2]

Address: 0xF801C02C (0)[0], 0xF801C06C (0)[1], 0xF801C0AC (0)[2], 0xFC02002C (1)[0], 0xFC02006C (1)[1], 0xFC0200AC (1)[2], 0xFC02402C (2)[0], 0xFC02406C (2)[1], 0xFC0240AC (2)[2]

Access: Read-only

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
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**

0: The Counter Overflow Interrupt is disabled.

1: The Counter Overflow Interrupt is enabled.

- **LOVRS: Load Overrun**

0: The Load Overrun Interrupt is disabled.

1: The Load Overrun Interrupt is enabled.

- **CPAS: RA Compare**

0: The RA Compare Interrupt is disabled.

1: The RA Compare Interrupt is enabled.

- **CPBS: RB Compare**

0: The RB Compare Interrupt is disabled.

1: The RB Compare Interrupt is enabled.

- **CPCS: RC Compare**

0: The RC Compare Interrupt is disabled.

1: The RC Compare Interrupt is enabled.

- **LDRAS: RA Loading**

0: The Load RA Interrupt is disabled.

1: The Load RA Interrupt is enabled.

- **LDRBS: RB Loading**

0: The Load RB Interrupt is disabled.

1: The Load RB Interrupt is enabled.

- **ETRGS: External Trigger**

0: The External Trigger Interrupt is disabled.

1: The External Trigger Interrupt is enabled.

45.7.14 TC Extended Mode Register

Name: TC_EMRx [x=0..2]

Address: 0xF801C030 (0)[0], 0xF801C070 (0)[1], 0xF801C0B0 (0)[2], 0xFC020030 (1)[0], 0xFC020070 (1)[1], 0xFC0200B0 (1)[2], 0xFC024030 (2)[0], 0xFC024070 (2)[1], 0xFC0240B0 (2)[2]

Access: Read/Write

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

- **TRIGSRCA: 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

- **TRIGSRCB: Trigger Source for Input B**

Value	Name	Description
0	EXTERNAL_TIOBx	The trigger/capture input B is driven by external pin TIOBx
1	PWMx	The trigger/capture input B is driven internally by the comparator output (refer to Figure 45-16) of the PWMx.

- **NODIVCLK: No Divided Clock**

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.

45.7.15 TC Block Control Register

Name: TC_BCR

Address: 0xF801C0C0 (0), 0xFC0200C0 (1), 0xFC0240C0 (2)

Access: Write-only

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
–	–	–	–	–	–	–	SYNC

- **SYNC: Synchro Command**

0: No effect.

1: Asserts the SYNC signal which generates a software trigger simultaneously for each of the channels.

45.7.16 TC Block Mode Register

Name: TC_BMR

Address: 0xF801C0C4 (0), 0xFC0200C4 (1), 0xFC0240C4 (2)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	MAXFILT	

23	22	21	20	19	18	17	16
MAXFILT				–	–	IDXPHB	SWAP

15	14	13	12	11	10	9	8
INVIDX	INVB	INVA	EDGPHA	QDTRANS	SPEEDEN	POSEN	QDEN

7	6	5	4	3	2	1	0
–	–	TC2XC2S		TC1XC1S		TC0XC0S	

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

- **TC0XC0S: 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

- **TC1XC1S: 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

- **TC2XC2S: 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

- **QDEN: Quadrature Decoder Enabled**

0: Disabled.

1: Enables the QDEC (filter, edge detection and quadrature decoding).

Quadrature decoding (direction change) can be disabled using QDTRANS bit.

One of the POSEN or SPEEDEN bits must be also enabled.

- **POSEN: Position Enabled**

0: Disable position.

1: Enables the position measure on channel 0 and 1.

- **SPEEDEN: Speed Enabled**

0: Disabled.

1: Enables the speed measure on channel 0, the time base being provided by channel 2.

- **QDTRANS: Quadrature Decoding Transparent**

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.

- **EDGPHA: Edge on PHA Count Mode**

0: Edges are detected on PHA only.

1: Edges are detected on both PHA and PHB.

- **INVA: Inverted PHA**

0: PHA (TIOA0) is directly driving the QDEC.

1: PHA is inverted before driving the QDEC.

- **INVB: Inverted PHB**

0: PHB (TIOB0) is directly driving the QDEC.

1: PHB is inverted before driving the QDEC.

- **INVIDX: Inverted Index**

0: IDX (TIOA1) is directly driving the QDEC.

1: IDX is inverted before driving the QDEC.

- **SWAP: Swap PHA and PHB**

0: No swap between PHA and PHB.

1: Swap PHA and PHB internally, prior to driving the QDEC.

- **IDXPHB: Index Pin is PHB Pin**

0: IDX pin of the rotary sensor must drive TIOA1.

1: IDX pin of the rotary sensor must drive TIOB0.

- **MAXFILT: Maximum Filter**

1–63: Defines the filtering capabilities.

Pulses with a period shorter than MAXFILT+1 peripheral clock cycles are discarded.

45.7.17 TC QDEC Interrupt Enable Register

Name: TC_QIER

Address: 0xF801C0C8 (0), 0xFC0200C8 (1), 0xFC0240C8 (2)

Access: Write-only

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
–	–	–	–	–	QERR	DIRCHG	IDX

- **IDX: Index**

0: No effect.

1: Enables the interrupt when a rising edge occurs on IDX input.

- **DIRCHG: Direction Change**

0: No effect.

1: Enables the interrupt when a change on rotation direction is detected.

- **QERR: Quadrature Error**

0: No effect.

1: Enables the interrupt when a quadrature error occurs on PHA, PHB.

45.7.18 TC QDEC Interrupt Disable Register

Name: TC_QIDR

Address: 0xF801C0CC (0), 0xFC0200CC (1), 0xFC0240CC (2)

Access: Write-only

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
–	–	–	–	–	QERR	DIRCHG	IDX

- **IDX: Index**

0: No effect.

1: Disables the interrupt when a rising edge occurs on IDX input.

- **DIRCHG: Direction Change**

0: No effect.

1: Disables the interrupt when a change on rotation direction is detected.

- **QERR: Quadrature Error**

0: No effect.

1: Disables the interrupt when a quadrature error occurs on PHA, PHB.

45.7.19 TC QDEC Interrupt Mask Register

Name: TC_QIMR

Address: 0xF801C0D0 (0), 0xFC0200D0 (1), 0xFC0240D0 (2)

Access: Read-only

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
–	–	–	–	–	QERR	DIRCHG	IDX

- **IDX: Index**

0: The interrupt on IDX input is disabled.

1: The interrupt on IDX input is enabled.

- **DIRCHG: Direction Change**

0: The interrupt on rotation direction change is disabled.

1: The interrupt on rotation direction change is enabled.

- **QERR: Quadrature Error**

0: The interrupt on quadrature error is disabled.

1: The interrupt on quadrature error is enabled.

45.7.20 TC QDEC Interrupt Status Register

Name: TC_QISR

Address: 0xF801C0D4 (0), 0xFC0200D4 (1), 0xFC0240D4 (2)

Access: Read-only

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

- **IDX: Index**

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.

- **DIRCHG: Direction Change**

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.

- **QERR: Quadrature Error**

0: No quadrature error since the last read of TC_QISR.

1: A quadrature error occurred since the last read of TC_QISR.

- **DIR: Direction**

Returns an image of the rotation direction.

45.7.21 TC Fault Mode Register

Name: TC_FMR

Address: 0xF801C0D8 (0), 0xFC0200D8 (1), 0xFC0240D8 (2)

Access: Read/Write

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
–	–	–	–	–	–	ENCF1	ENCF0

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

- **ENCF0: Enable Compare Fault Channel 0**

0: Disables the FAULT output source (CPCS flag) from channel 0.

1: Enables the FAULT output source (CPCS flag) from channel 0.

- **ENCF1: Enable Compare Fault Channel 1**

0: Disables the FAULT output source (CPCS flag) from channel 1.

1: Enables the FAULT output source (CPCS flag) from channel 1.

45.7.22 TC Write Protection Mode Register

Name: TC_WPMR

Address: 0xF801C0E4 (0), 0xFC0200E4 (1), 0xFC0240E4 (2)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

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).

The Timer Counter clock of the first channel must be enabled to access this register.

Refer to Section 45.6.19 “Register Write Protection” for a list of registers that can be write-protected and Timer Counter clock conditions.

- **WPKEY: 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.

46. Pulse Width Modulation Controller (PWM)

46.1 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.

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 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, to trigger pulses on the 2 independent event lines (in order to synchronize ADC conversions with a lot of flexibility independently of the PWM outputs).

PWM outputs can be overridden synchronously or asynchronously to their channel counter.

The PWM provides a fault protection mechanism with 8 fault inputs**,** capable to detect a fault condition and to override the PWM outputs asynchronously (outputs forced to ‘0’, ‘1’ or Hi-Z).

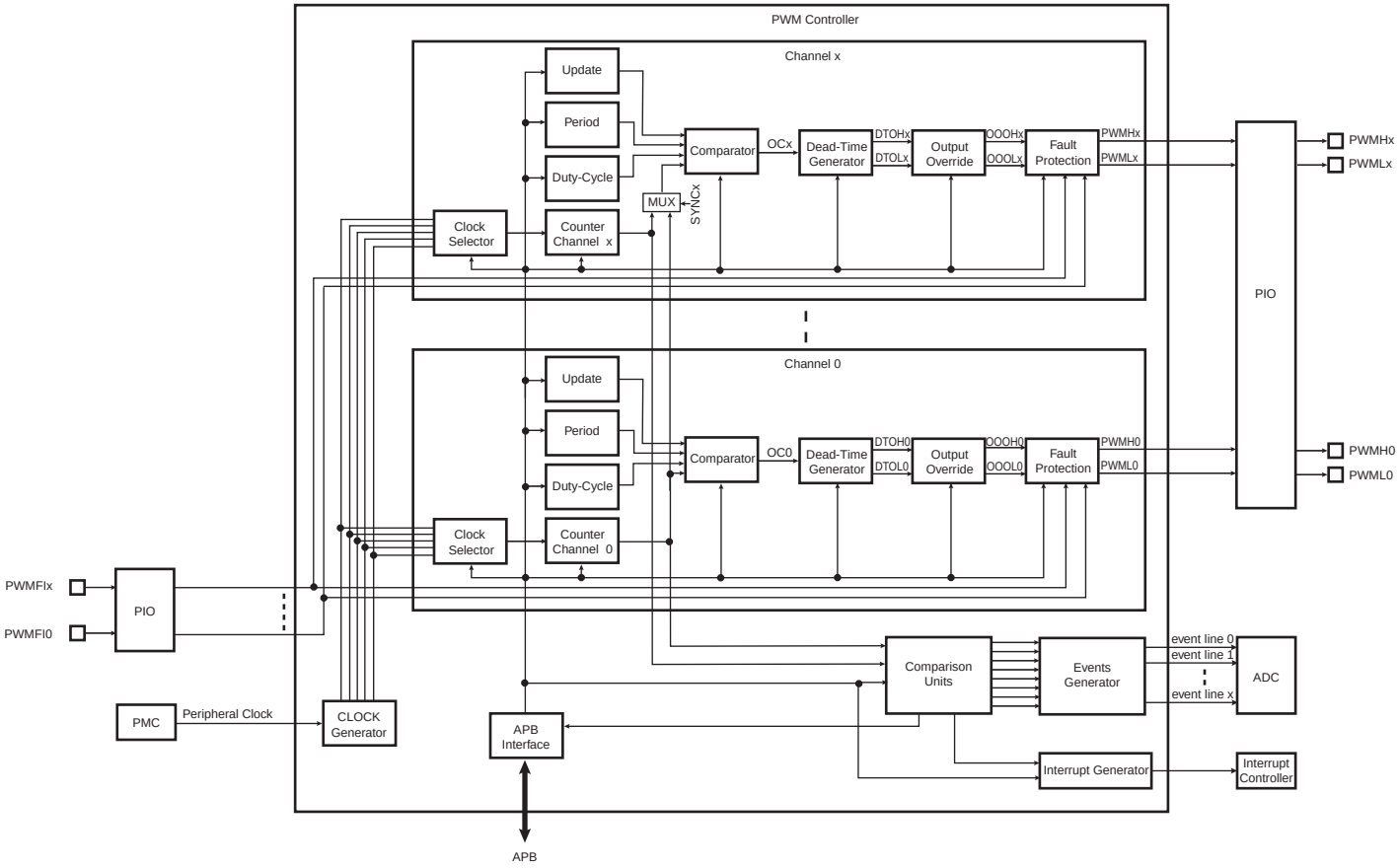
For safety usage, some configuration registers are write-protected.

46.2 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 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
- 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
- 2 Independent Events Lines Intended to Synchronize ADC Conversions
  - Programmable delay for Events Lines to delay ADC measurements
- 8 Comparison Units Intended to Generate Interrupts, Pulses on Event Lines
- 8 Programmable Fault Inputs Providing an Asynchronous Protection of PWM Outputs
  - 2 User Driven through PIO Inputs
  - PMC Driven when Crystal Oscillator Clock Fails
  - ADC Controller Driven through Configurable Comparison Function
  - Timer/Counter Driven through Configurable Comparison Function
- Register Write Protection

46.3 Block Diagram

Figure 46-1. Pulse Width Modulation Controller Block Diagram



46.4 I/O Lines Description

Each channel outputs two complementary external I/O lines.

Table 46-1. 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

46.5 Product Dependencies

46.5.1 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.

Table 46-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
PWM	PWMFI0	PC29	C
PWM	PWMFI1	PE7	C
PWM	PWMH0	PA26	B
PWM	PWMH0	PB14	C
PWM	PWMH0	PB26	C
PWM	PWMH0	PC30	C
PWM	PWMH1	PA28	B
PWM	PWMH1	PB11	C
PWM	PWMH1	PB28	C
PWM	PWMH1	PC31	C
PWM	PWMH2	PC0	B
PWM	PWMH2	PE12	C
PWM	PWMH3	PC2	B
PWM	PWMH3	PE14	C
PWM	PWML0	PA27	B
PWM	PWML0	PB15	C
PWM	PWML0	PB27	C
PWM	PWML0	PC27	C
PWM	PWML1	PA29	B
PWM	PWML1	PB10	C
PWM	PWML1	PB29	C
PWM	PWML1	PC28	C
PWM	PWML2	PC1	B
PWM	PWML2	PE13	C
PWM	PWML3	PC3	B
PWM	PWML3	PE8	C

46.5.2 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.

46.5.3 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.

Table 46-3. Peripheral IDs

Instance	ID
PWM	43

46.5.4 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
- the PMC
- the ADC controller
- Timer/Counters

Table 46-4. Fault Inputs

Fault Generator	External PWM Fault Input Number	Polarity Level(1)	Fault Input ID
PC29	PWMFI0	User-defined	0
PE7	PWMFI1	User-defined	1
Main OSC (PMC)	–	To be configured to 1	2
ADC	–	To be configured to 1	3
Timer0	–	To be configured to 1	4
Timer1	–	To be configured to 1	5
Timer2	–	To be configured to 1	6

Note: 1. FPOL field in PWMC_FMR.

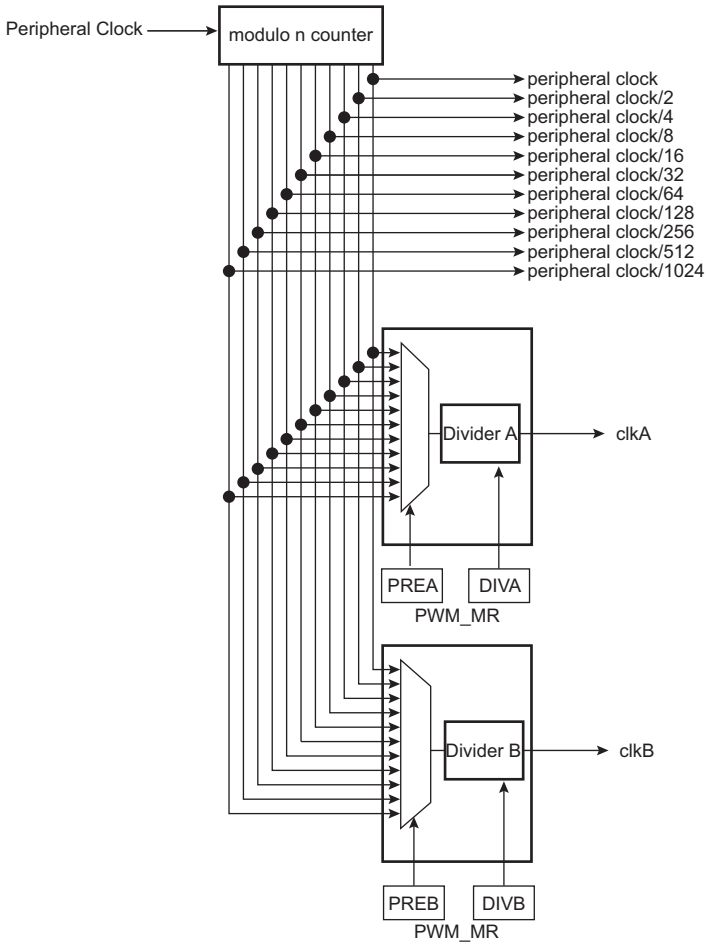
46.6 Functional Description

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.

46.6.1 PWM Clock Generator

Figure 46-2. 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.

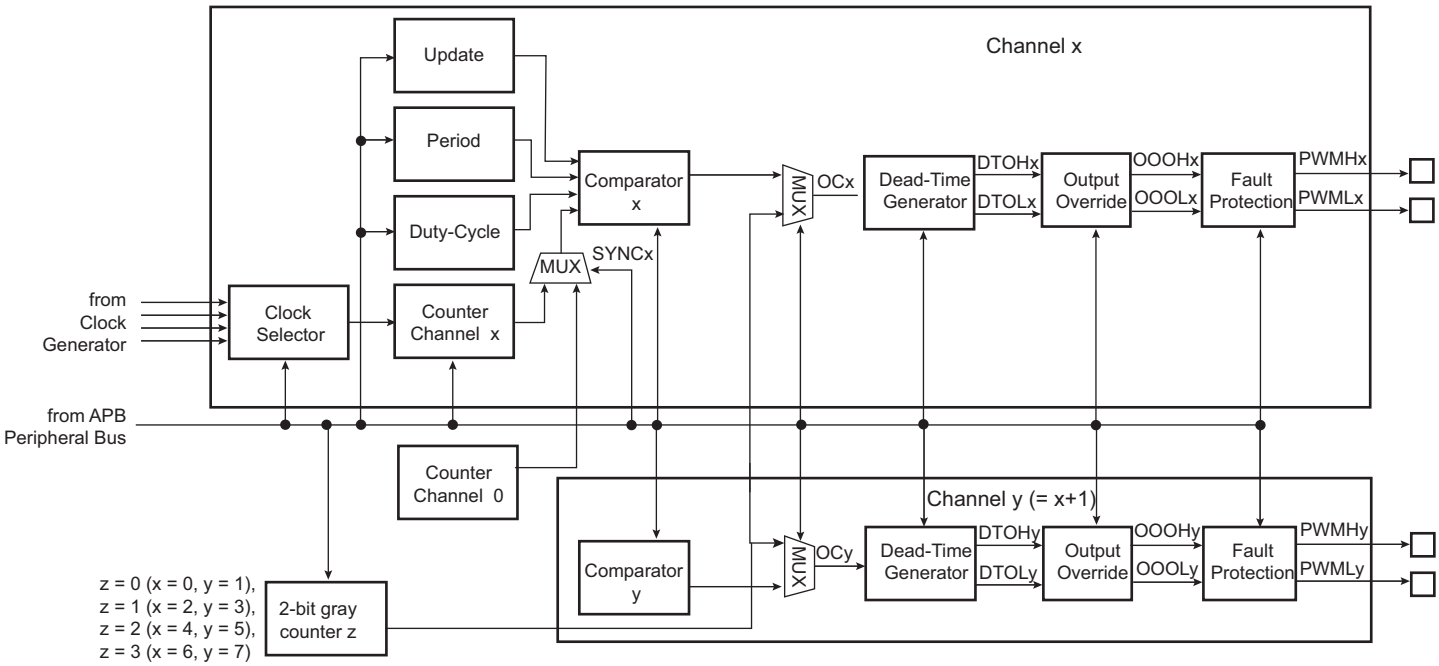
At reset, all clocks provided by the modulo n counter are turned off except the peripheral clock. This situation is also true when the PWM peripheral clock is turned off through the Power Management Controller.

CAUTION: Before using the PWM controller, the programmer must first enable the peripheral clock in the Power Management Controller (PMC).

46.6.2 PWM Channel

46.6.2.1 Channel Block Diagram

Figure 46-3. 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 Section 46.6.1 “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).

46.6.2.2 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:

  $$\frac{(X \times CPRD)}{f_{\text{peripheral clock}}}$$

  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:

  $$\frac{(X \times CRPD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(X \times CRPD \times DIVB)}{f_{\text{peripheral clock}}}$$

  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:

  $$\frac{(2 \times X \times CPRD)}{f_{\text{peripheral clock}}}$$

  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:

  $$\frac{(2 \times X \times CPRD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(2 \times X \times CPRD \times DIVB)}{f_{\text{peripheral clock}}}$$

- 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:

  $$\text{duty cycle} = \frac{(period - 1/\text{fchannel\_x\_clock} \times CDTY)}{period}$$

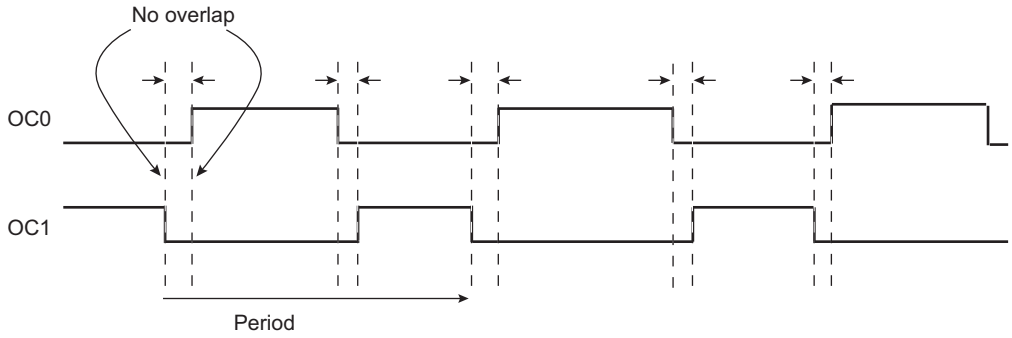
  If the waveform is center-aligned, then:

  $$\text{duty cycle} = \frac{((period/2) - 1/\text{fchannel\_x\_clock} \times CDTY))}{(period/2)}$$

- 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 ***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 46-4. Non-Overlapped Center-Aligned Waveforms



Note: Refer to Figure 46-5 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
- 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

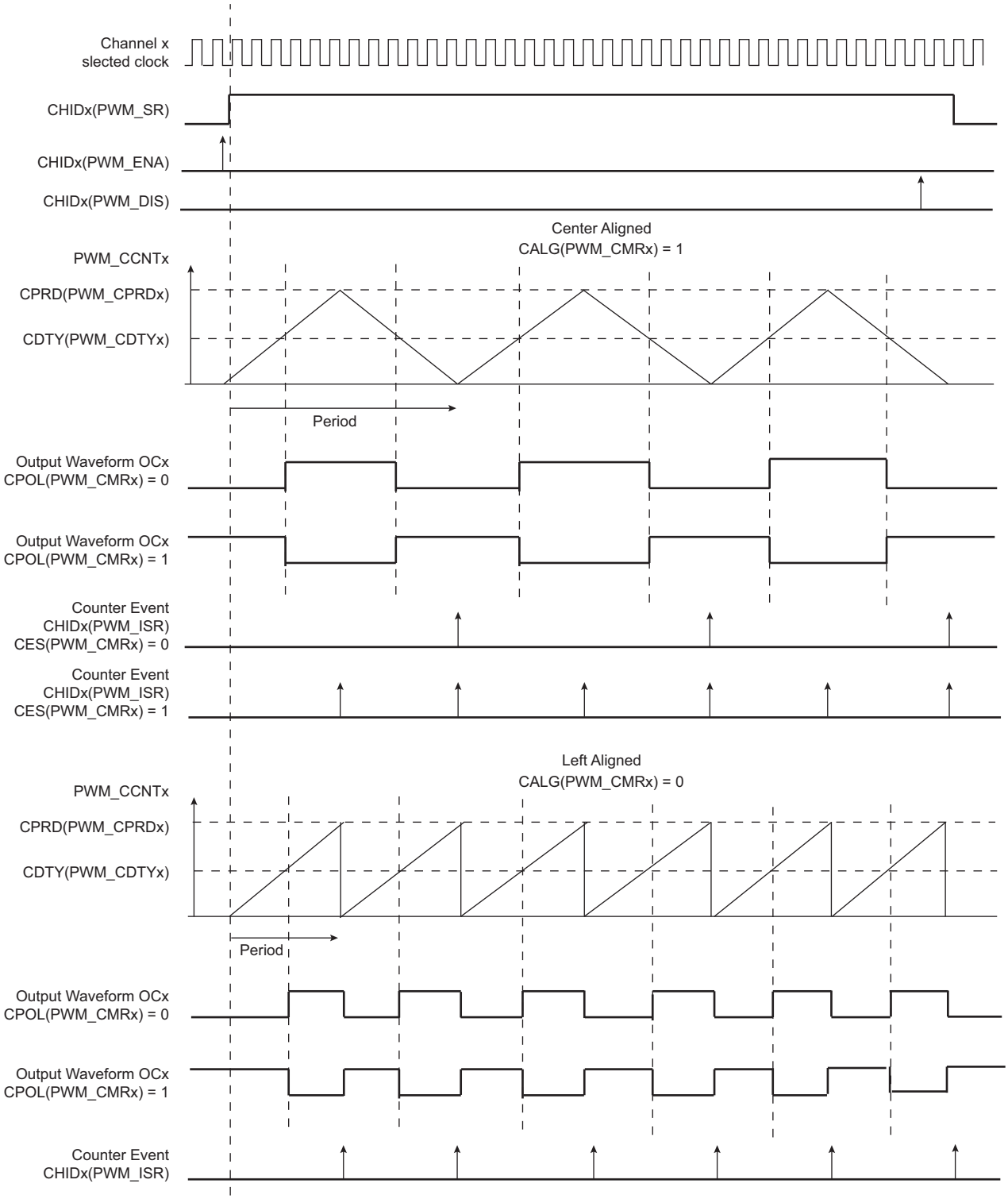
The waveform polarity must be set before enabling the channel. This immediately affects the channel output level.

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.

Figure 46-5 illustrates the counter interrupts depending on the configuration.

Figure 46-5. Waveform Properties



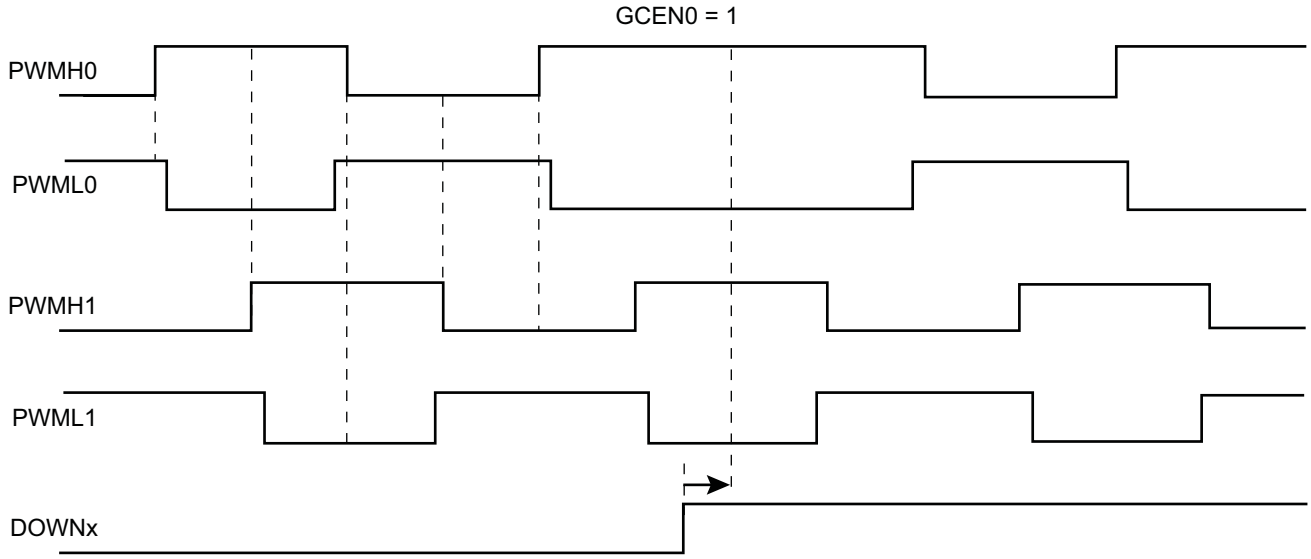
46.6.2.3 2-bit Gray Up/Down Counter for Stepper Motor

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 gray counter.

Figure 46-6. 2-bit Gray Up/Down Counter



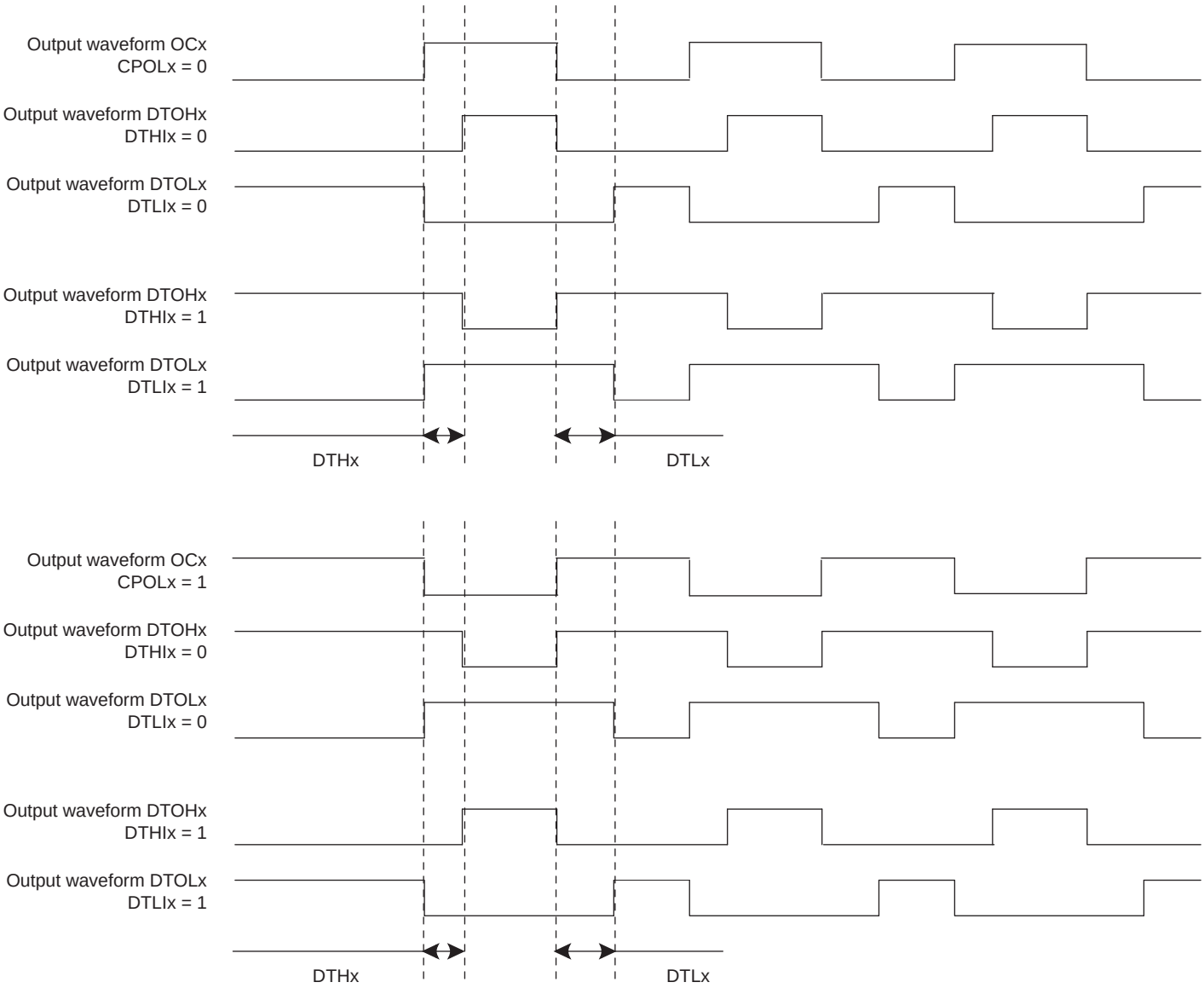
46.6.2.4 Dead-Time Generator

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.

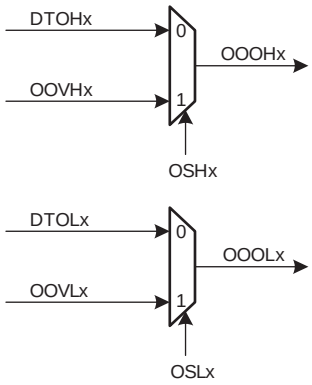
Figure 46-7. Complementary Output Waveforms



46.6.2.5 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 46-8. 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.

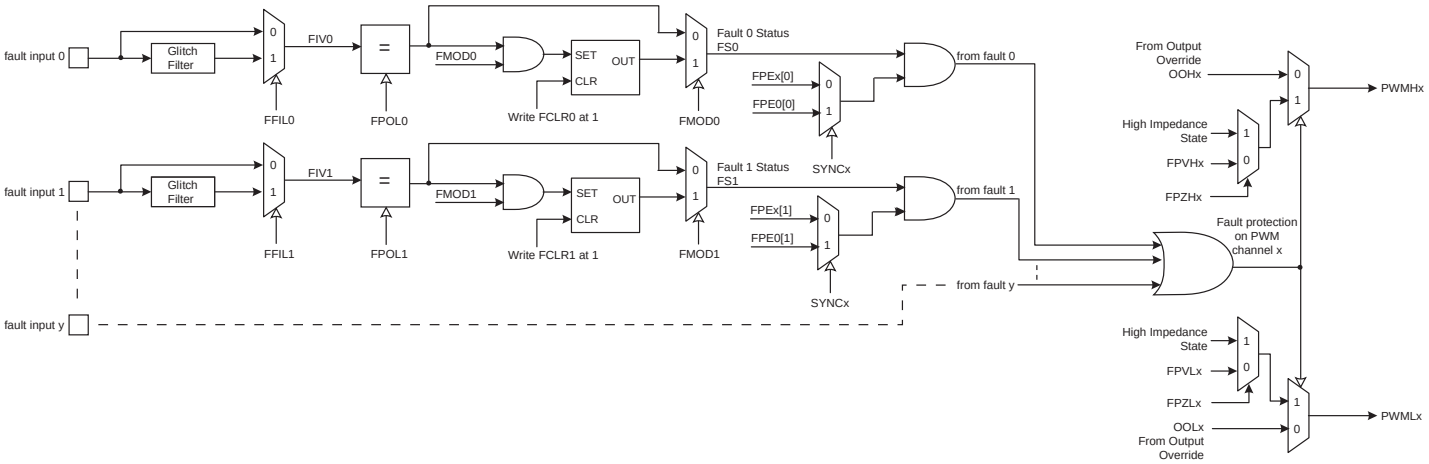
The value of the current output selection can be read in PWM_OS.

While overriding PWM outputs, the channel counters continue to run, only the PWM outputs are forced to user defined values.

46.6.2.6 Fault Protection

8 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 46-9. 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 ADC or Timer Counter, the polarity level must be FPOL = 1. For fault inputs coming from external GPIO pins the polarity level depends on the user's implementation.

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 (refer to Section 46.6.2.8 “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 Table 46-5. The output forcing is made asynchronously to the channel counter.

Table 46-5. 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 (refer to Section 46.6.3 “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.

46.6.2.7 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).

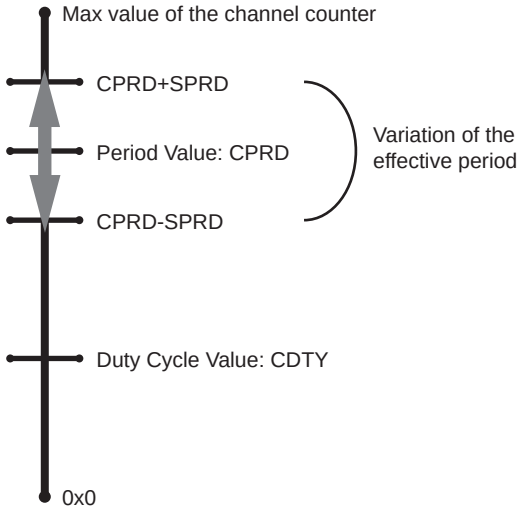
It will cause the effective period to vary from CPRD-SPRD to CPRD+SPRD. This leads to a constantly varying duty cycle on the PWM output waveform because the duty cycle value programmed is unchanged.

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 46-10. Spread Spectrum Counter



46.6.2.8 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’ (refer to “Method 1: Manual write of duty-cycle values and manual trigger of the update”).

- 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) (refer to “Method 2: Manual write of duty-cycle values and automatic trigger of the update”).

Table 46-6. Summary of the Update of Registers of Synchronous Channels

Register	UPDM = 0	UPDM = 1
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
	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

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:

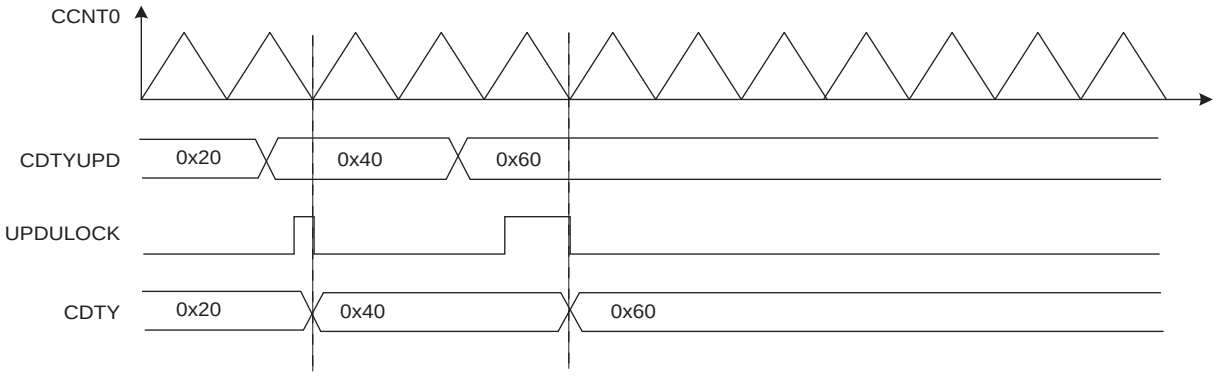
- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

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 46-11. Method 1 (UPDM = 0)



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.

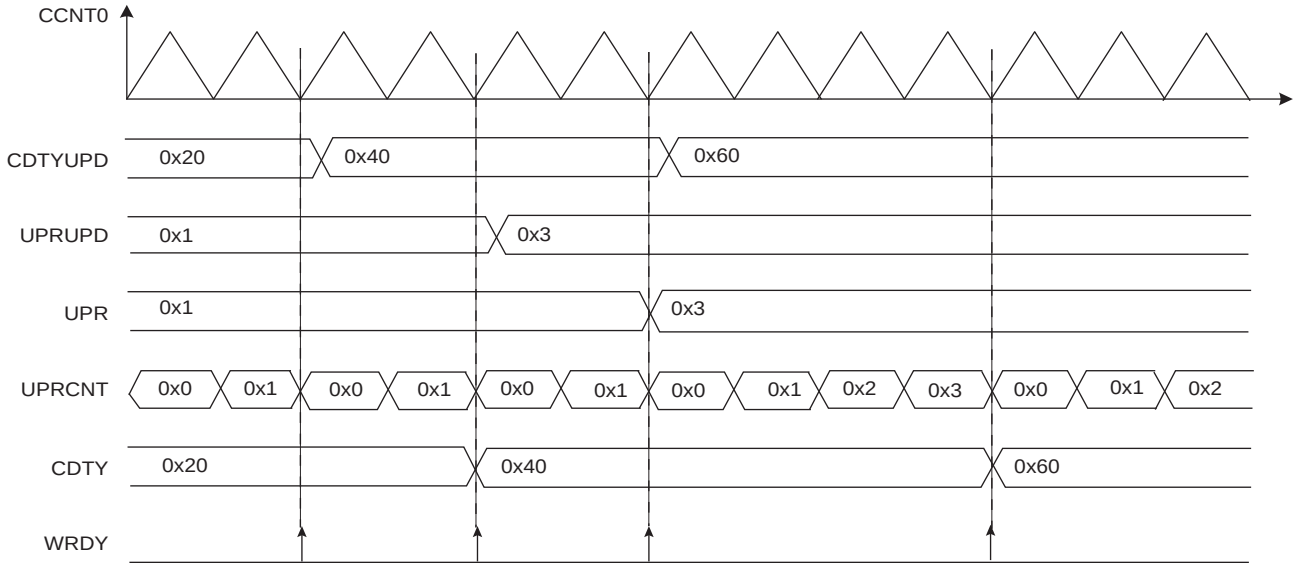
Depending on the interrupt mask in the PWM Interrupt Mask Register 2 (PWM_IMR2), an interrupt can be generated by these flags.

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 46-12. Method 2 (UPDM = 1)



46.6.2.9 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:

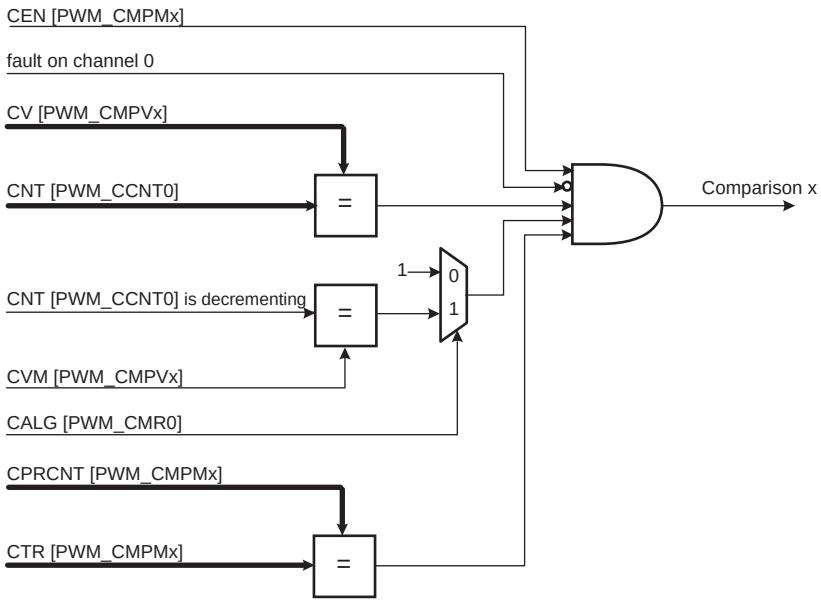
- PWM Channel Duty Cycle Update Register
- PWM Channel Mode Update Register

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.

46.6.3 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, Section 46.6.2.8 “Synchronous Channels”). These comparisons are intended to generate pulses on the event lines (used to synchronize ADC, refer to Section 46.6.4 “PWM Event Lines”), to generate software interrupts.

Figure 46-13. 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 (refer to Section 46.6.2.6 “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.

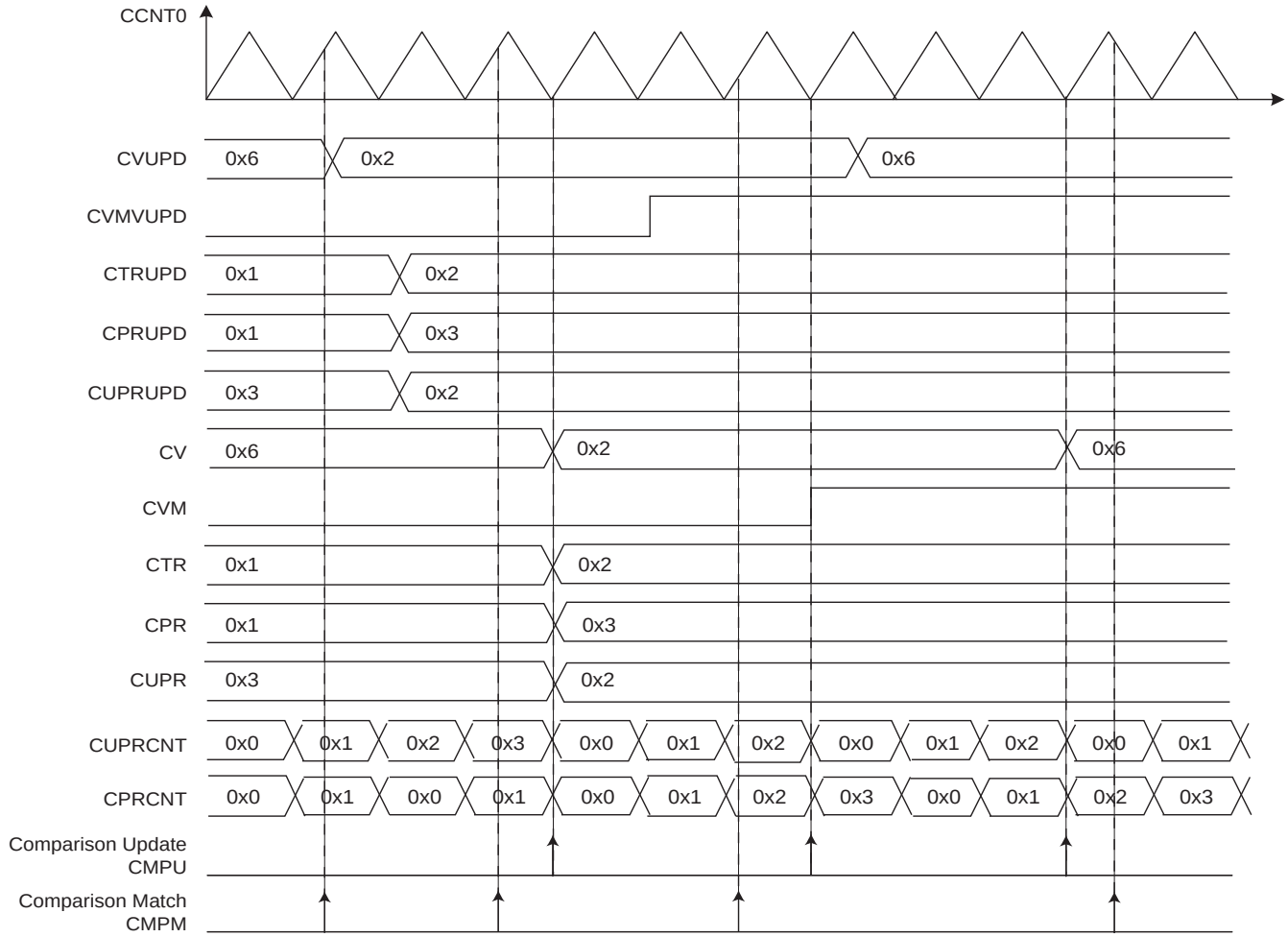
The comparison x configuration can be modified while the channel 0 is enabled by using the PWM Comparison x Mode Update Register (PWM_CMPMUPDx registers for the comparison x). In the same way, the comparison x value can be modified while the channel 0 is enabled by using the PWM Comparison x Value Update 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 46-14. Comparison Waveform



46.6.4 PWM Event Lines

The PWM provides 2 independent event lines intended to trigger actions in other peripherals (e.g., for the Analog-to-Digital Converter (ADC)).

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 Figure 46-16.

Figure 46-15. Event Line Block Diagram

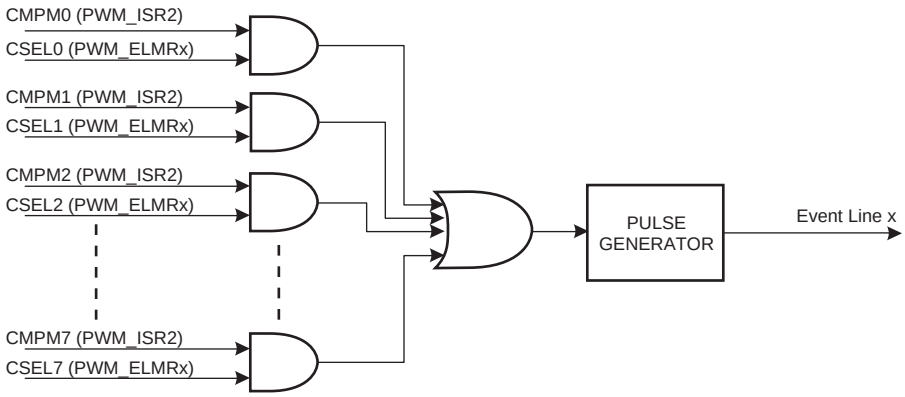
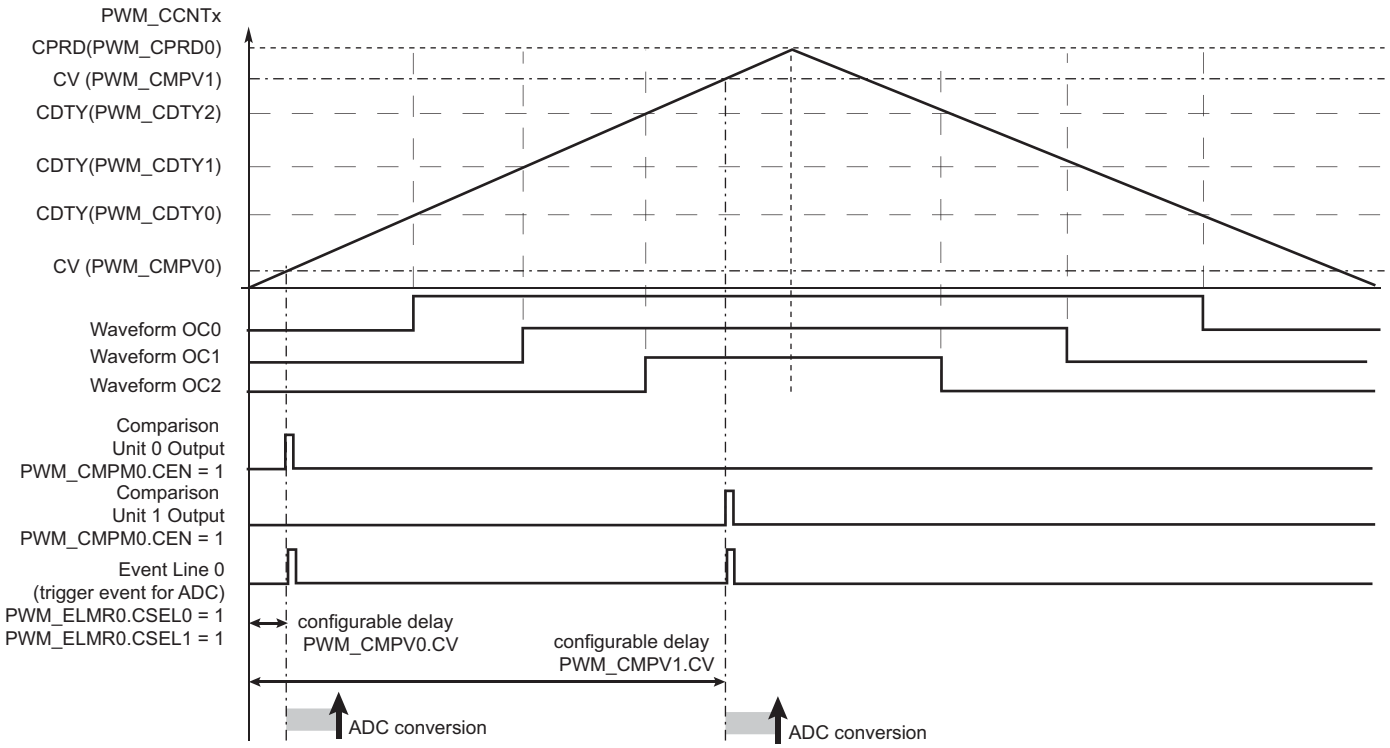


Figure 46-16. Event Line Generation Waveform (Example)



46.6.5 PWM Controller Operations

46.6.5.1 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)
- 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)

46.6.5.2 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.

46.6.5.3 Changing the Duty-Cycle, the Period and the Dead-Times

It is possible to modulate the output waveform duty-cycle, period and dead-times.

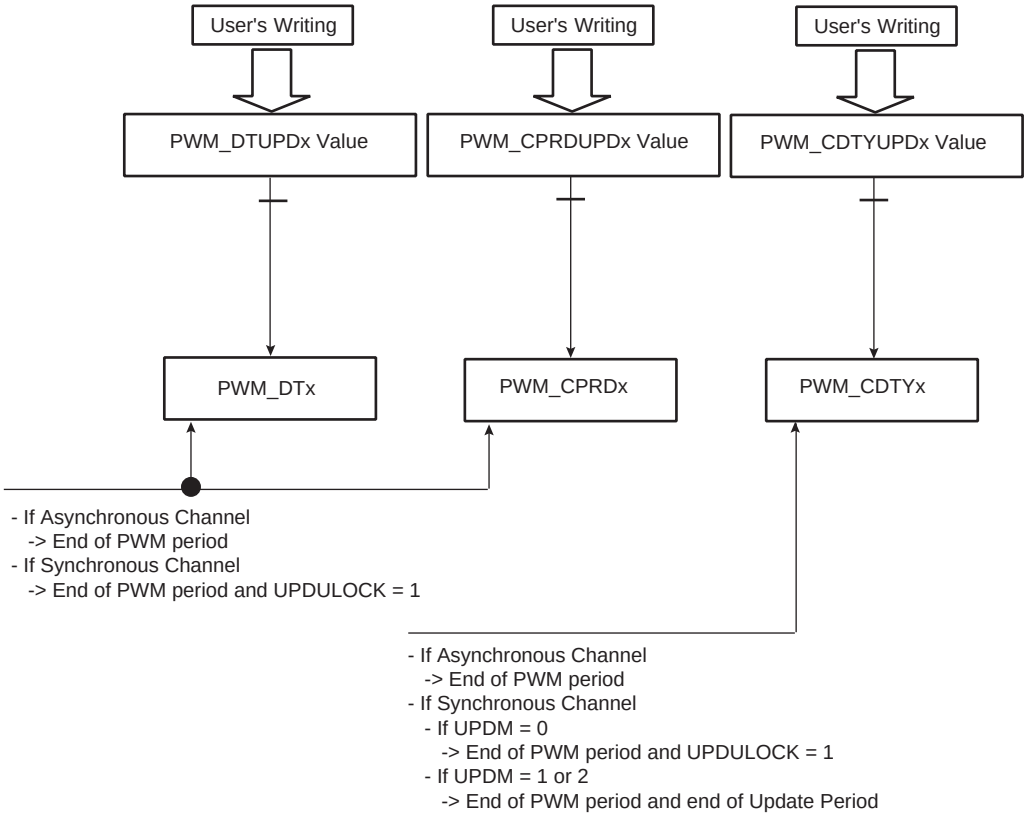
To prevent unexpected output waveform, the user must use the PWM Channel Duty Cycle Update Register (PWM_CDTYUPDx), the PWM Channel Period Update Register (PWM_CPRDUPDx) and the PWM Channel Dead Time Update Register (PWM_DTUPDx) to change waveform parameters while the channel is still enabled.

- 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 46-17. Synchronized Period, Duty-Cycle and Dead-Time Update



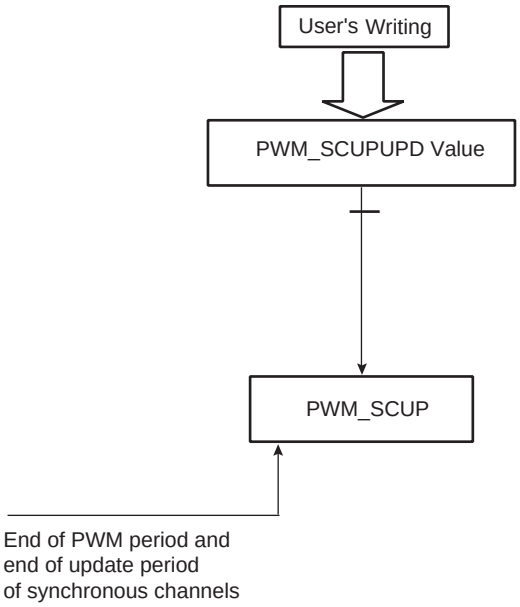
46.6.5.4 Changing the Update Period of Synchronous Channels

It is possible to change the update period of synchronous channels while they are enabled. Refer to "Method 2: Manual 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.

Notes:
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 46-18. Synchronized Update of Update Period Value of Synchronous Channels



46.6.5.5 Changing the Comparison Value and the Comparison Configuration

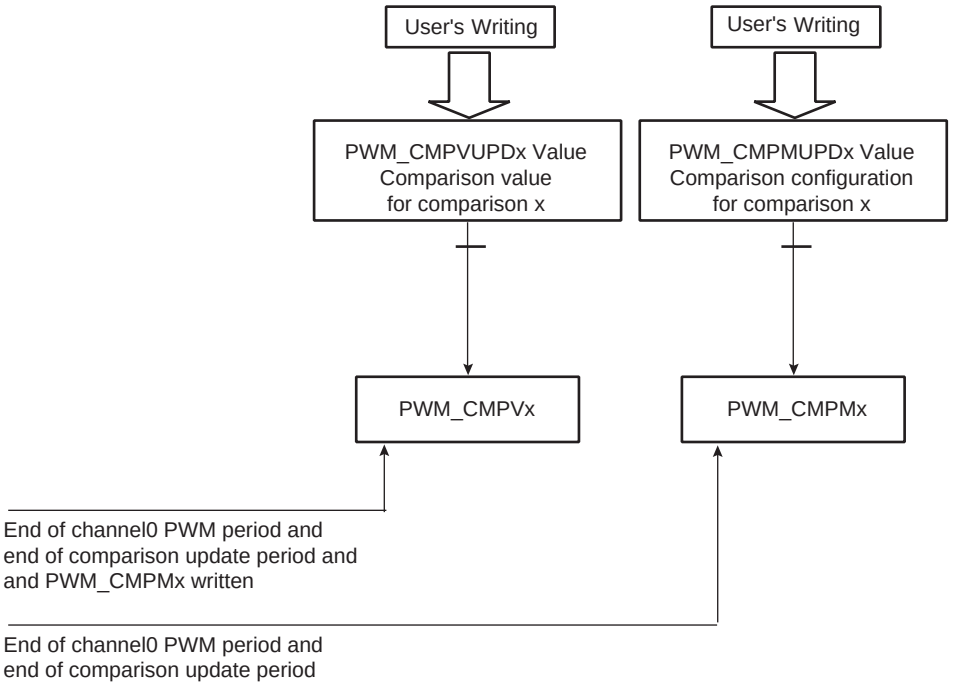
It is possible to change the comparison values and the comparison configurations while the channel 0 is enabled (refer to Section 46.6.3 “PWM Comparison Units”).

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 46-19. Synchronized Update of Comparison Values and Configurations



46.6.5.6 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.

46.6.6 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
- Register group 2:
  - PWM Sync Channels Mode Register
  - PWM Channel Mode Register
  - PWM Stepper Motor Mode 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.

46.7 Pulse Width Modulation Controller (PWM) User Interface

Table 46-7. Register Mapping

Offset	Register	Name	Access	Reset
0x00	PWM Clock Register	PWM_CLK	Read/Write	0x0
0x04	PWM Enable Register	PWM_ENA	Write-only	–
0x08	PWM Disable Register	PWM_DIS	Write-only	–
0x0C	PWM Status Register	PWM_SR	Read-only	0x0
0x10	PWM Interrupt Enable Register 1	PWM_IER1	Write-only	–
0x14	PWM Interrupt Disable Register 1	PWM_IDR1	Write-only	–
0x18	PWM Interrupt Mask Register 1	PWM_IMR1	Read-only	0x0
0x1C	PWM Interrupt Status Register 1	PWM_ISR1	Read-only	0x0
0x20	PWM Sync Channels Mode Register	PWM_SCM	Read/Write	0x0
0x24	Reserved	–	–	–
0x28	PWM Sync Channels Update Control Register	PWM_SCUC	Read/Write	0x0
0x2C	PWM Sync Channels Update Period Register	PWM_SCUP	Read/Write	0x0
0x30	PWM Sync Channels Update Period Update Register	PWM_SCUPUPD	Write-only	–
0x34	PWM Interrupt Enable Register 2	PWM_IER2	Write-only	–
0x38	PWM Interrupt Disable Register 2	PWM_IDR2	Write-only	–
0x3C	PWM Interrupt Mask Register 2	PWM_IMR2	Read-only	0x0
0x40	PWM Interrupt Status Register 2	PWM_ISR2	Read-only	0x0
0x44	PWM Output Override Value Register	PWM_OOV	Read/Write	0x0
0x48	PWM Output Selection Register	PWM_OS	Read/Write	0x0
0x4C	PWM Output Selection Set Register	PWM_OSS	Write-only	–
0x50	PWM Output Selection Clear Register	PWM_OSC	Write-only	–
0x54	PWM Output Selection Set Update Register	PWM_OSSUPD	Write-only	–
0x58	PWM Output Selection Clear Update Register	PWM_OSCUPD	Write-only	–
0x5C	PWM Fault Mode Register	PWM_FMR	Read/Write	0x0
0x60	PWM Fault Status Register	PWM_FSR	Read-only	0x0
0x64	PWM Fault Clear Register	PWM_FCR	Write-only	–
0x68	PWM Fault Protection Value Register 1	PWM_FPV1	Read/Write	0x0
0x6C	PWM Fault Protection Enable Register	PWM_FPE	Read/Write	0x0
0x70–0x78	Reserved	–	–	–
0x7C	PWM Event Line 0 Mode Register	PWM_ELMR0	Read/Write	0x0
0x80	PWM Event Line 1 Mode Register	PWM_ELMR1	Read/Write	0x0
0x84–0x9C	Reserved	–	–	–
0xA0	PWM Spread Spectrum Register	PWM_SSPR	Read/Write	0x0
0xA4	PWM Spread Spectrum Update Register	PWM_SSPUP	Write-only	–
0xA8–0xAC	Reserved	–	–	–

Table 46-7. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0xB0	PWM Stepper Motor Mode Register	PWM_SMMR	Read/Write	0x0
0xB4–0xBC	Reserved	–	–	–
0xC0	PWM Fault Protection Value 2 Register	PWM_FPV2	Read/Write	0x003F_003F
0xC4–0xE0	Reserved	–	–	–
0xE4	PWM Write Protection Control Register	PWM_WPCR	Write-only	–
0xE8	PWM Write Protection Status Register	PWM_WPSR	Read-only	0x0
0xEC–0xFC	Reserved	–	–	–
0x130	PWM Comparison 0 Value Register	PWM_CMPV0	Read/Write	0x0
0x134	PWM Comparison 0 Value Update Register	PWM_CMPVUPD0	Write-only	–
0x138	PWM Comparison 0 Mode Register	PWM_CMPM0	Read/Write	0x0
0x13C	PWM Comparison 0 Mode Update Register	PWM_CMPMUPD0	Write-only	–
0x140	PWM Comparison 1 Value Register	PWM_CMPV1	Read/Write	0x0
0x144	PWM Comparison 1 Value Update Register	PWM_CMPVUPD1	Write-only	–
0x148	PWM Comparison 1 Mode Register	PWM_CMPM1	Read/Write	0x0
0x14C	PWM Comparison 1 Mode Update Register	PWM_CMPMUPD1	Write-only	–
0x150	PWM Comparison 2 Value Register	PWM_CMPV2	Read/Write	0x0
0x154	PWM Comparison 2 Value Update Register	PWM_CMPVUPD2	Write-only	–
0x158	PWM Comparison 2 Mode Register	PWM_CMPM2	Read/Write	0x0
0x15C	PWM Comparison 2 Mode Update Register	PWM_CMPMUPD2	Write-only	–
0x160	PWM Comparison 3 Value Register	PWM_CMPV3	Read/Write	0x0
0x164	PWM Comparison 3 Value Update Register	PWM_CMPVUPD3	Write-only	–
0x168	PWM Comparison 3 Mode Register	PWM_CMPM3	Read/Write	0x0
0x16C	PWM Comparison 3 Mode Update Register	PWM_CMPMUPD3	Write-only	–
0x170	PWM Comparison 4 Value Register	PWM_CMPV4	Read/Write	0x0
0x174	PWM Comparison 4 Value Update Register	PWM_CMPVUPD4	Write-only	–
0x178	PWM Comparison 4 Mode Register	PWM_CMPM4	Read/Write	0x0
0x17C	PWM Comparison 4 Mode Update Register	PWM_CMPMUPD4	Write-only	–
0x180	PWM Comparison 5 Value Register	PWM_CMPV5	Read/Write	0x0
0x184	PWM Comparison 5 Value Update Register	PWM_CMPVUPD5	Write-only	–
0x188	PWM Comparison 5 Mode Register	PWM_CMPM5	Read/Write	0x0
0x18C	PWM Comparison 5 Mode Update Register	PWM_CMPMUPD5	Write-only	–
0x190	PWM Comparison 6 Value Register	PWM_CMPV6	Read/Write	0x0
0x194	PWM Comparison 6 Value Update Register	PWM_CMPVUPD6	Write-only	–
0x198	PWM Comparison 6 Mode Register	PWM_CMPM6	Read/Write	0x0
0x19C	PWM Comparison 6 Mode Update Register	PWM_CMPMUPD6	Write-only	–
0x1A0	PWM Comparison 7 Value Register	PWM_CMPV7	Read/Write	0x0
0x1A4	PWM Comparison 7 Value Update Register	PWM_CMPVUPD7	Write-only	–

Table 46-7. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x1A8	PWM Comparison 7 Mode Register	PWM_CMPM7	Read/Write	0x0
0x1AC	PWM Comparison 7 Mode Update Register	PWM_CMPMUPD7	Write-only	–
0x1B0–0x1FC	Reserved	–	–	–
0x200 + ch_num * 0x20 + 0x00	PWM Channel Mode Register(1)	PWM_CMR	Read/Write	0x0
0x200 + ch_num * 0x20 + 0x04	PWM Channel Duty Cycle Register(1)	PWM_CDTY	Read/Write	0x0
0x200 + ch_num * 0x20 + 0x08	PWM Channel Duty Cycle Update Register(1)	PWM_CDTYUPD	Write-only	–
0x200 + ch_num * 0x20 + 0x0C	PWM Channel Period Register(1)	PWM_CPRD	Read/Write	0x0
0x200 + ch_num * 0x20 + 0x10	PWM Channel Period Update Register(1)	PWM_CPRDUPD	Write-only	–
0x200 + ch_num * 0x20 + 0x14	PWM Channel Counter Register(1)	PWM_CCNT	Read-only	0x0
0x200 + ch_num * 0x20 + 0x18	PWM Channel Dead Time Register(1)	PWM_DT	Read/Write	0x0
0x200 + ch_num * 0x20 + 0x1C	PWM Channel Dead Time Update Register(1)	PWM_DTUPD	Write-only	–
0x400 + ch_num * 0x20 + 0x00	PWM Channel Mode Update Register(1)	PWM_CMUPD	Write-only	–

Notes: 1. Some registers are indexed with “ch_num” index ranging from 0 to 3.

46.7.1 PWM Clock Register

Name: PWM_CLK

Address: 0xF800C000

Access: Read/Write

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

This register can only be written if bits WPSWS0 and WPHWS0 are cleared in the PWM Write Protection Status Register.

- **DIVA: 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

- **DIVB: 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

- **PREA: CLKA 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

- **PREB: 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

46.7.2 PWM Enable Register

Name: PWM_ENA

Address: 0xF800C004

Access: Write-only

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
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Channel ID**

0: No effect.

1: Enable PWM output for channel x.

46.7.3 PWM Disable Register

Name: PWM_DIS

Address: 0xF800C008

Access: Write-only

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
–	–	–	–	CHID3	CHID2	CHID1	CHID0

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

- **CHIDx: Channel ID**

0: No effect.

1: Disable PWM output for channel x.

46.7.4 PWM Status Register

Name: PWM_SR

Address: 0xF800C00C

Access: Read-only

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
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Channel ID**

0: PWM output for channel x is disabled.

1: PWM output for channel x is enabled.

46.7.5 PWM Interrupt Enable Register 1

Name: PWM_IER1

Address: 0xF800C010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FCHID3	FCHID2	FCHID1	FCHID0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Counter Event on Channel x Interrupt Enable**
- **FCHIDx: Fault Protection Trigger on Channel x Interrupt Enable**

46.7.6 PWM Interrupt Disable Register 1

Name: PWM_IDR1

Address: 0xF800C014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FCHID3	FCHID2	FCHID1	FCHID0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Counter Event on Channel x Interrupt Disable**
- **FCHIDx: Fault Protection Trigger on Channel x Interrupt Disable**

46.7.7 PWM Interrupt Mask Register 1

Name: PWM_IMR1

Address: 0xF800C018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FCHID3	FCHID2	FCHID1	FCHID0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Counter Event on Channel x Interrupt Mask**
- **FCHIDx: Fault Protection Trigger on Channel x Interrupt Mask**

46.7.8 PWM Interrupt Status Register 1

Name: PWM_ISR1

Address: 0xF800C01C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FCHID3	FCHID2	FCHID1	FCHID0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Counter Event on Channel x**

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.

- **FCHIDx: Fault Protection Trigger on Channel x**

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.

Note: Reading PWM_ISR1 automatically clears CHIDx and FCHIDx flags.

46.7.9 PWM Sync Channels Mode Register

Name: PWM_SCM

Address: 0xF800C020

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	UPDM	
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	SYNC3	SYNC2	SYNC1	SYNC0

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

- **SYNCx: Synchronous Channel x**

0: Channel x is not a synchronous channel.

1: Channel x is a synchronous channel.

- **UPDM: Synchronous Channels Update Mode**

Value	Name	Description
0	MODE0	Manual write of double buffer registers and manual update of synchronous channels(1)
1	MODE1	Manual write of double buffer registers and automatic update of synchronous channels(2)

Notes: 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.

Address: 0xF800C024

46.7.10 PWM Sync Channels Update Control Register

Name: PWM_SCUC

Address: 0xF800C028

Access: Read/Write

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
–	–	–	–	–	–	–	UPDULOCK

- **UPDULOCK: Synchronous Channels Update Unlock**

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.

This bit is automatically reset when the update is done.

46.7.11 PWM Sync Channels Update Period Register

Name: PWM_SCUP

Address: 0xF800C02C

Access: Read/Write

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
UPRCNT				UPR			

- **UPR: 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.

- **UPRCNT: Update Period Counter**

Reports the value of the update period counter.

46.7.12 PWM Sync Channels Update Period Update Register

Name: PWM_SCUPUPD

Address: 0xF800C030

Access: Write-only

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
–	–	–	–	UPRUPD			

This register acts as a double buffer for the UPR value. This prevents an unexpected automatic trigger of the update of synchronous channels.

- **UPRUPD: Update Period Update**

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.

46.7.13 PWM Interrupt Enable Register 2

Name: PWM_IER2

Address: 0xF800C034

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0

15	14	13	12	11	10	9	8
CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0

7	6	5	4	3	2	1	0
–	–	–	–	UNRE			WRDY

- **WRDY: Write Ready for Synchronous Channels Update Interrupt Enable**
- **UNRE: Synchronous Channels Update Underrun Error Interrupt Enable**
- **CMPMx: Comparison x Match Interrupt Enable**
- **CMPUx: Comparison x Update Interrupt Enable**

46.7.14 PWM Interrupt Disable Register 2

Name: PWM_IDR2

Address: 0xF800C038

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
15	14	13	12	11	10	9	8
CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
7	6	5	4	3	2	1	0
–	–	–	–	UNRE			WRDY

- **WRDY: Write Ready for Synchronous Channels Update Interrupt Disable**
- **UNRE: Synchronous Channels Update Underrun Error Interrupt Disable**
- **CMPMx: Comparison x Match Interrupt Disable**
- **CMPUx: Comparison x Update Interrupt Disable**

46.7.15 PWM Interrupt Mask Register 2

Name: PWM_IMR2

Address: 0xF800C03C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0

15	14	13	12	11	10	9	8
CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0

7	6	5	4	3	2	1	0
–	–	–	–	UNRE			WRDY

- **WRDY: Write Ready for Synchronous Channels Update Interrupt Mask**
- **UNRE: Synchronous Channels Update Underrun Error Interrupt Mask**
- **CMPMx: Comparison x Match Interrupt Mask**
- **CMPUx: Comparison x Update Interrupt Mask**

46.7.16 PWM Interrupt Status Register 2

Name: PWM_ISR2

Address: 0xF800C040

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0

15	14	13	12	11	10	9	8
CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0

7	6	5	4	3	2	1	0
–	–	–	–	UNRE			WRDY

- **WRDY: Write Ready for Synchronous Channels Update**

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.

- **UNRE: Synchronous Channels Update Underrun Error**

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.

- **CMPMx: Comparison x Match**

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.

- **CMPUx: Comparison x Update**

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.

Note: Reading PWM_ISR2 automatically clears flags WRDY, UNRE and CMPSx.

46.7.17 PWM Output Override Value Register

Name: PWM_OOV

Address: 0xF800C044

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OOVL3	OOVL2	OOVL1	OOVL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OOVH3	OOVH2	OOVH1	OOVH0

- **OOVHx: Output Override Value for PWMH output of the channel x**

0: Override value is 0 for PWMH output of channel x.

1: Override value is 1 for PWMH output of channel x.

- **OOVLx: Output Override Value for PWML output of the channel x**

0: Override value is 0 for PWML output of channel x.

1: Override value is 1 for PWML output of channel x.

46.7.18 PWM Output Selection Register

Name: PWM_OS

Address: 0xF800C048

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSL3	OSL2	OSL1	OSL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSH3	OSH2	OSH1	OSH0

- **OSHx: Output Selection for PWMH output of the channel x**

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.

- **OSLx: Output Selection for PWML output of the channel x**

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.

46.7.19 PWM Output Selection Set Register

Name: PWM_OSS

Address: 0xF800C04C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSSL3	OSSL2	OSSL1	OSSL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSSH3	OSSH2	OSSH1	OSSH0

- **OSSHx: Output Selection Set for PWMH output of the channel x**

0: No effect.

1: Output override value OOVHx selected as PWMH output of channel x.

- **OSSLx: Output Selection Set for PWML output of the channel x**

0: No effect.

1: Output override value OOVLx selected as PWML output of channel x.

46.7.20 PWM Output Selection Clear Register

Name: PWM_OSC

Address: 0xF800C050

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSCL3	OSCL2	OSCL1	OSCL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSCH3	OSCH2	OSCH1	OSCH0

- **OSCHx: Output Selection Clear for PWMH output of the channel x**

0: No effect.

1: Dead-time generator output DTOHx selected as PWMH output of channel x.

- **OSCLx: Output Selection Clear for PWML output of the channel x**

0: No effect.

1: Dead-time generator output DTOLx selected as PWML output of channel x.

46.7.21 PWM Output Selection Set Update Register

Name: PWM_OSSUPD

Address: 0xF800C054

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSSUPL3	OSSUPL2	OSSUPL1	OSSUPL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSSUPH3	OSSUPH2	OSSUPH1	OSSUPH0

- **OSSUPHx: Output Selection Set for PWMH output of the channel x**

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.

- **OSSUPLx: Output Selection Set for PWML output of the channel x**

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.

46.7.22 PWM Output Selection Clear Update Register

Name: PWM_OSCUPD

Address: 0xF800C058

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSCUPL3	OSCUPL2	OSCUPL1	OSCUPL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSCUPH3	OSCUPH2	OSCUPH1	OSCUPH0

- **OSCUPHx: Output Selection Clear for PWMH output of the channel x**

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.

- **OSCUPLx: Output Selection Clear for PWML output of the channel x**

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.

46.7.23 PWM Fault Mode Register

Name: PWM_FMR

Address: 0xF800C05C

Access: Read/Write

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

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Refer to Section 46.5.4 “Fault Inputs” for details on fault generation.

- **FPOL: 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.

- **FMOD: 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(1) level AND until it is cleared in the PWM Fault Clear Register.

Note: 1. The peripheral generating the fault.

- **FFIL: 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.

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.

46.7.24 PWM Fault Status Register

Name: PWM_FSR

Address: 0xF800C060

Access: Read-only

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

Refer to Section 46.5.4 “Fault Inputs” for details on fault generation.

- **FIV: 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).

- **FS: 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.

46.7.25 PWM Fault Clear Register

Name: PWM_FCR

Address: 0xF800C064

Access: Write-only

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
FCLR							

Refer to Section 46.5.4 “Fault Inputs” for details on fault generation.

- **FCLR: 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.

46.7.26 PWM Fault Protection Value Register 1

Name: PWM_FPV1

Address: 0xF800C068

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FPVL3	FPVL2	FPVL1	FPVL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	FPVH3	FPVH2	FPVH1	FPVH0

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Refer to Section 46.5.4 “Fault Inputs” for details on fault generation.

- **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.

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.

- **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.

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.

46.7.27 PWM Fault Protection Enable Register

Name: PWM_FPE

Address: 0xF800C06C

Access: Read/Write

31	30	29	28	27	26	25	24
FPE3							
23	22	21	20	19	18	17	16
FPE2							
15	14	13	12	11	10	9	8
FPE1							
7	6	5	4	3	2	1	0
FPE0							

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Only the first 8 bits (number of fault input pins) of fields FPE0, FPE1, FPE2 and FPE3 are significant.

Refer to Section 46.5.4 “Fault Inputs” for details on fault generation.

- **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.

46.7.28 PWM Event Line x Register

Name: PWM_ELMRx

Address: 0xF800C07C

Access: Read/Write

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
CSEL7	CSEL6	CSEL5	CSEL4	CSEL3	CSEL2	CSEL1	CSEL0

- **CSELy: Comparison y Selection**

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.

46.7.29 PWM Spread Spectrum Register

Name: PWM_SSPR

Address: 0xF800C0A0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	SPRDM

23	22	21	20	19	18	17	16
SPRD							

15	14	13	12	11	10	9	8
SPRD							

7	6	5	4	3	2	1	0
SPRD							

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.

- **SPRD: 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.

- **SPRDM: Spread Spectrum Counter Mode**

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.

46.7.30 PWM Spread Spectrum Update Register

Name: PWM_SSPUP

Address: 0xF800C0A4

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
SPRDUP							

15	14	13	12	11	10	9	8
SPRDUP							

7	6	5	4	3	2	1	0
SPRDUP							

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.

- **SPRDUP: 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.

46.7.31 PWM Stepper Motor Mode Register

Name: PWM_SMMR

Address: 0xF800C0B0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	DOWN1	DOWN0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	GCEN1	GCEN0

- **GCENx: Gray Count Enable**

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].

- **DOWNx: Down Count**

0: Up counter.

1: Down counter.

46.7.32 PWM Fault Protection Value Register 2

Name: PWM_FPV2

Address: 0xF800C0C0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FPZL3	FPZL2	FPZL1	FPZL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	FPZH3	FPZH2	FPZH1	FPZH0

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

- **FPZHx: Fault Protection to Hi-Z for PWMH output on channel x**

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.

- **FPZLx: Fault Protection to Hi-Z for PWML output on channel x**

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.

46.7.33 PWM Write Protection Control Register

Name: PWM_WPCR

Address: 0xF800C0E4

Access: Write-only

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
WPRG5	WPRG4	WPRG3	WPRG2	WPRG1	WPRG0	WPCMD	

Refer to Section 46.6.6 “Register Write Protection” for the list of registers that can be write-protected.

- **WPCMD: 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.

- **WPRGx: Write Protection Register Group x**

0: The WPCMD command has no effect on the register group x.

1: The WPCMD command is applied to the register group x.

- **WPKEY: 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

46.7.34 PWM Write Protection Status Register

Name: PWM_WPSR

Address: 0xF800C0E8

Access: Read-only

31	30	29	28	27	26	25	24
WPVSRC							

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
–	–	WPHWS5	WPHWS4	WPHWS3	WPHWS2	WPHWS1	WPHWS0

7	6	5	4	3	2	1	0
WPVS	–	WPSWS5	WPSWS4	WPSWS3	WPSWS2	WPSWS1	WPSWS0

- **WPSWSx: Write Protect SW Status**

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.

- **WPHWSx: Write Protect HW Status**

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.

- **WPVS: Write Protect Violation Status**

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.

- **WPVSRC: Write Protect Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

46.7.35 PWM Comparison x Value Register

Name: PWM_CMPVx

Address: 0xF800C130 [0], 0xF800C140 [1], 0xF800C150 [2], 0xF800C160 [3], 0xF800C170 [4], 0xF800C180 [5], 0xF800C190 [6], 0xF800C1A0 [7]

Access: Read/Write

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

Only the first 16 bits (channel counter size) of field CV are significant.

- **CV: Comparison x Value**

Define the comparison x value to be compared with the counter of the channel 0.

- **CVM: Comparison x Value Mode**

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)

46.7.36 PWM Comparison x Value Update Register

Name: PWM_CMPVUPDx

Address: 0xF800C134 [0], 0xF800C144 [1], 0xF800C154 [2], 0xF800C164 [3], 0xF800C174 [4], 0xF800C184 [5], 0xF800C194 [6], 0xF800C1A4 [7]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	CVMUPD

23	22	21	20	19	18	17	16
CVUPD							

15	14	13	12	11	10	9	8
CVUPD							

7	6	5	4	3	2	1	0
CVUPD							

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.

- **CVUPD: Comparison x Value Update**

Define the comparison x value to be compared with the counter of the channel 0.

- **CVMUPD: Comparison x Value Mode Update**

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)

CAUTION: The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

46.7.37 PWM Comparison x Mode Register

Name: PWM_CMPMx

Address: 0xF800C138 [0], 0xF800C148 [1], 0xF800C158 [2], 0xF800C168 [3], 0xF800C178 [4], 0xF800C188 [5], 0xF800C198 [6], 0xF800C1A8 [7]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CUPRCNT				CUPR			

15	14	13	12	11	10	9	8
CPRCNT				CPR			

7	6	5	4	3	2	1	0
CTR				–	–	–	CEN

- **CEN: Comparison x Enable**

0: The comparison x is disabled and can not match.

1: The comparison x is enabled and can match.

- **CTR: Comparison x Trigger**

The comparison x is performed when the value of the comparison x period counter (CPRCNT) reaches the value defined by CTR.

- **CPR: 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.

- **CPRCNT: Comparison x Period Counter**

Reports the value of the comparison x period counter.

Note: The field CPRCNT is read-only

- **CUPR: 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.

- **CUPRCNT: Comparison x Update Period Counter**

Reports the value of the comparison x update period counter.

Note: The field CUPRCNT is read-only

46.7.38 PWM Comparison x Mode Update Register

Name: PWM_CMPMUPDx

Address: 0xF800C13C [0], 0xF800C14C [1], 0xF800C15C [2], 0xF800C16C [3], 0xF800C17C [4], 0xF800C18C [5], 0xF800C19C [6], 0xF800C1AC [7]

Access: Write-only

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

This register acts as a double buffer for the CEN, CTR, CPR and CUPR values. This prevents an unexpected comparison x match.

- **CENUPD: Comparison x Enable Update**

0: The comparison x is disabled and can not match.

1: The comparison x is enabled and can match.

- **CTRUPD: 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.

- **CPRUPD: 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.

- **CUPRUPD: 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.

46.7.39 PWM Channel Mode Register

Name: PWM_CMRx [x=0..3]

Address: 0xF800C200 [0], 0xF800C220 [1], 0xF800C240 [2], 0xF800C260 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	DTLI	DTHI	DTE
15	14	13	12	11	10	9	8
–	–	–	–	UPDS	CES	CPOL	CALG
7	6	5	4	3	2	1	0
–	–	–	–	CPRE			

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

- **CPRE: Channel Prescaler**

Value	Name	Description
0	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

- **CALG: Channel Alignment**

0: The period is left-aligned.

1: The period is center-aligned.

- **CPOL: Channel Polarity**

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.

- **CES: Counter Event Selection**

The bit CES defines when the channel counter event occurs when the period is center-aligned (flag CHIDx in PWM Interrupt Status Register 1).

CALG = 0 (Left Alignment):

0/1: The channel counter event occurs at the end of the PWM period.

CALG = 1 (Center Alignment):

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.

- **UPDS: Update Selection**

When the period is center aligned, the bit UPDS defines when the update of the duty cycle, the polarity value/mode occurs after writing the corresponding update registers.

CALG = 0 (Left Alignment):

0/1: The update always occurs at the end of the PWM period after writing the update register(s).

CALG = 1 (Center Alignment):

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).

- **DTE: Dead-Time Generator Enable**

0: The dead-time generator is disabled.

1: The dead-time generator is enabled.

- **DTHI: Dead-Time PWMHx Output Inverted**

0: The dead-time PWMHx output is not inverted.

1: The dead-time PWMHx output is inverted.

- **DTLI: Dead-Time PWMLx Output Inverted**

0: The dead-time PWMLx output is not inverted.

1: The dead-time PWMLx output is inverted.

46.7.40 PWM Channel Duty Cycle Register

Name: PWM_CDTYx [x=0..3]

Address: 0xF800C204 [0], 0xF800C224 [1], 0xF800C244 [2], 0xF800C264 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CDTY							

15	14	13	12	11	10	9	8
CDTY							

7	6	5	4	3	2	1	0
CDTY							

Only the first 16 bits (channel counter size) are significant.

- **CDTY: Channel Duty-Cycle**

Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

46.7.41 PWM Channel Duty Cycle Update Register

Name: PWM_CDTYUPDx [x=0..3]

Address: 0xF800C208 [0], 0xF800C228 [1], 0xF800C248 [2], 0xF800C268 [3]

Access: Write-only.

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CDTYUPD							

15	14	13	12	11	10	9	8
CDTYUPD							

7	6	5	4	3	2	1	0
CDTYUPD							

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.

- **CDTYUPD: Channel Duty-Cycle Update**

Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

46.7.42 PWM Channel Period Register

Name: PWM_CPRDx [x=0..3]

Address: 0xF800C20C [0], 0xF800C22C [1], 0xF800C24C [2], 0xF800C26C [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CPRD							

15	14	13	12	11	10	9	8
CPRD							

7	6	5	4	3	2	1	0
CPRD							

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.

- **CPRD: 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:

$$\frac{(X \times CPRD)}{f_{\text{peripheral clock}}}$$

– 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:

$$\frac{(X \times CPRD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(X \times CPRD \times DIVB)}{f_{\text{peripheral clock}}}$$

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:

$$\frac{(2 \times X \times CPRD)}{f_{\text{peripheral clock}}}$$

– 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:

$$\frac{(2 \times X \times CPRD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(2 \times X \times CPRD \times DIVB)}{f_{\text{peripheral clock}}}$$

46.7.43 PWM Channel Period Update Register

Name: PWM_CPRDUPDx [x=0..3]

Address: 0xF800C210 [0], 0xF800C230 [1], 0xF800C250 [2], 0xF800C270 [3]

Access: Write-only

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

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.

- **CPRDUPD: 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:

$$\frac{(X \times CPRDUPD)}{f_{\text{peripheral clock}}}$$

– 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:

$$\frac{(X \times CRPDUPD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(X \times CRPDUPD \times DIVB)}{f_{\text{peripheral clock}}}$$

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:

$$\frac{(2 \times X \times CPRDUPD)}{f_{\text{peripheral clock}}}$$

– 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:

$$\frac{(2 \times X \times CPRDUPD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(2 \times X \times CPRDUPD \times DIVB)}{f_{\text{peripheral clock}}}$$

46.7.44 PWM Channel Counter Register

Name: PWM_CCNTx [x=0..3]

Address: 0xF800C214 [0], 0xF800C234 [1], 0xF800C254 [2], 0xF800C274 [3]

Access: Read-only

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

Only the first 16 bits (channel counter size) are significant.

- **CNT: 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.

46.7.45 PWM Channel Dead Time Register

Name: PWM_DTx [x=0..3]

Address: 0xF800C218 [0], 0xF800C238 [1], 0xF800C258 [2], 0xF800C278 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
DTL							

23	22	21	20	19	18	17	16
DTL							

15	14	13	12	11	10	9	8
DTH							

7	6	5	4	3	2	1	0
DTH							

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.

- **DTH: 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).

- **DTL: 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).

46.7.46 PWM Channel Dead Time Update Register

Name: PWM_DTUPDx [x=0..3]

Address: 0xF800C21C [0], 0xF800C23C [1], 0xF800C25C [2], 0xF800C27C [3]

Access: Write-only

31	30	29	28	27	26	25	24
DTLUPD							

23	22	21	20	19	18	17	16
DTLUPD							

15	14	13	12	11	10	9	8
DTHUPD							

7	6	5	4	3	2	1	0
DTHUPD							

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.

- **DTHUPD: 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.

- **DTLUPD: 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.

46.7.47 PWM Channel Mode Update Register

Name: PWM_CMUPDx [x=0..3]

Address: 0xF800C400 [0], 0xF800C420 [1], 0xF800C440 [2], 0xF800C460 [3]

Access: Read/Write

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

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.

- **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.

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.

- **CPOLINVUP: Channel Polarity Inversion Update**

If this bit is written at ‘1’, the write of the bit CPOLUP is not taken into account.

0: No effect.

1: The OCx output waveform (output from the comparator) is inverted.

47. Analog-to-Digital Converter (ADC)

47.1 Description

The ADC is based on a 10-bit Analog-to-Digital Converter (ADC) managed by an ADC Controller providing enhanced resolution up to 12 bits. Refer to Figure 47-1 “Analog-to-Digital Converter Block Diagram with Touchscreen Mode”. It also integrates a 5-to-1 analog multiplexer, making possible the analog-to-digital conversions of 5 analog lines.The conversions extend from 0V to the voltage carried on pin ADVREF.

The ADC digital controller embeds circuitry to reduce the resolution down to 8 bits. The 8-bit resolution mode prevents using 16-bit Peripheral DMA transfer into memory when only 8-bit resolution is required by the application. Note that using this low resolution mode does not increase the conversion rate.

Conversion results are reported in a common register for all channels, as well as in a channel-dedicated register.

The 11-bit and 12-bit resolution modes are obtained by averaging multiple samples to decrease quantization noise. For the 11-bit mode, 4 samples are used, which gives a real sample rate of 1/4 of the actual sample frequency. For the 12-bit mode, 16 samples are used, giving a real sample rate of 1/16 of the actual sample frequency. This arrangement allows conversion speed to be traded for better accuracy.

Software trigger, external trigger on rising edge of the ADTRG pin or internal triggers from Timer Counter output(s) are configurable.

The comparison circuitry allows automatic detection of values below a threshold, higher than a threshold, in a given range or outside the range, thresholds and ranges being fully configurable.

The ADC Controller internal fault output is directly connected to PWM fault input. This input can be asserted by means of comparison circuitry in order to immediately put the PWM output in a safe state (pure combinational path).

The ADC also integrates a Sleep mode and a conversion sequencer and connects with a DMA channel. These features reduce both power consumption and processor intervention.

This ADC Controller includes a Resistive Touchscreen Controller. It supports 4-wire and 5-wire technologies.

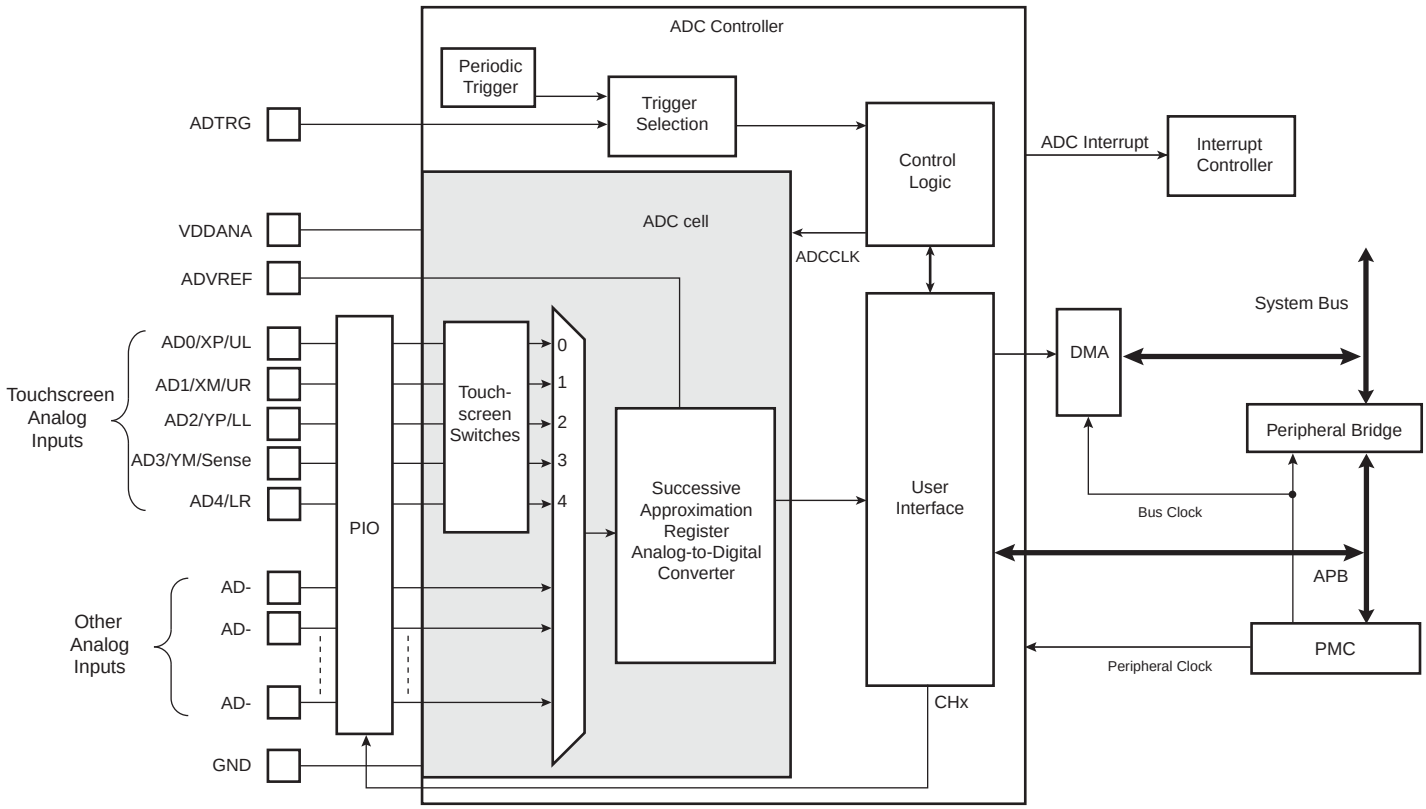
47.2 Embedded Characteristics

- 10-bit Resolution with Enhanced Mode up to 12 bits
- 320 ksps Conversion Rate
- Digital Averaging Function providing Enhanced Resolution Mode up to 12 bits
- Wide Range of Power Supply Operation
- Resistive 4-wire and 5-wire Touchscreen Controller
  - Position and Pressure Measurement for 4-wire Screens
  - Position Measurement for 5-wire Screens
  - Average of Up to 8 Measures for Noise Filtering
- Programmable Pen Detection Sensitivity
- Integrated Multiplexer Offering Up to 5 Independent Analog Inputs
- Individual Enable and Disable of Each Channel
- Hardware or Software Trigger from:
  - External Trigger Pin
  - Timer Counter Outputs (Corresponding TIOA Trigger)
  - Internal Trigger Counter
  - Trigger on Pen Contact Detection
  - PWM Event Line

- Drive of PWM Fault Input
- DMA Support
- Two Sleep Modes (Automatic Wakeup on Trigger)
  - Lowest Power Consumption (Voltage Reference OFF Between Conversions)
  - Fast Wakeup Time Response on Trigger Event (Voltage Reference ON Between Conversions)
- Channel Sequence Customizing
- Automatic Window Comparison of Converted Values
- Register Write Protection

47.3 Block Diagram

Figure 47-1. Analog-to-Digital Converter Block Diagram with Touchscreen Mode



47.4 Signal Description

Table 47-1. ADC Pin Description

Pin Name	Description
VDDANA	Analog Power Supply
ADVREF	Reference Voltage
AD0–AD4	Analog input Channels
ADTRG	External Trigger

47.5 Product Dependencies

47.5.1 Power Management

The ADC Controller is not continuously clocked. The programmer must first enable the ADC Controller peripheral clock in the Power Management Controller (PMC) before using the ADC Controller. However, if the application does not require ADC operations, the ADC Controller clock can be stopped when not needed and restarted when necessary. Configuring the ADC Controller does not require the ADC Controller clock to be enabled.

47.5.2 Interrupt Sources

The ADC interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the ADC interrupt requires the interrupt controller to be programmed first.

Table 47-2. Peripheral IDs

Instance	ID
ADC	44

47.5.3 I/O Lines

The digital input ADTRG is multiplexed with digital functions on the I/O line and the selection of ADTRG is made using the PIO controller.

The analog inputs ADC_ADx are multiplexed with digital functions on the I/O lines. ADC_ADx inputs are selected as inputs of the ADCC when writing a one in the corresponding CHx bit of ADC_CHER and the digital functions are not selected.

Table 47-3. I/O Lines

Instance	Signal	I/O Line	Peripheral
ADC	ADTRG	PE31	A
ADC	AD0	PC27	X1
ADC	AD1	PC28	X1
ADC	AD2	PC29	X1
ADC	AD3	PC30	X1
ADC	AD4	PC31	X1

47.5.4 Hardware Triggers

The ADC can use internal signals to start conversions. Refer to “TRGSEL: Trigger Selection” for the exact wiring of internal triggers.

47.5.5 Fault Output

The ADC Controller has the FAULT output connected to the FAULT input of PWM. Refer to Section 47.6.13 “Fault Output” and to Section PWM.

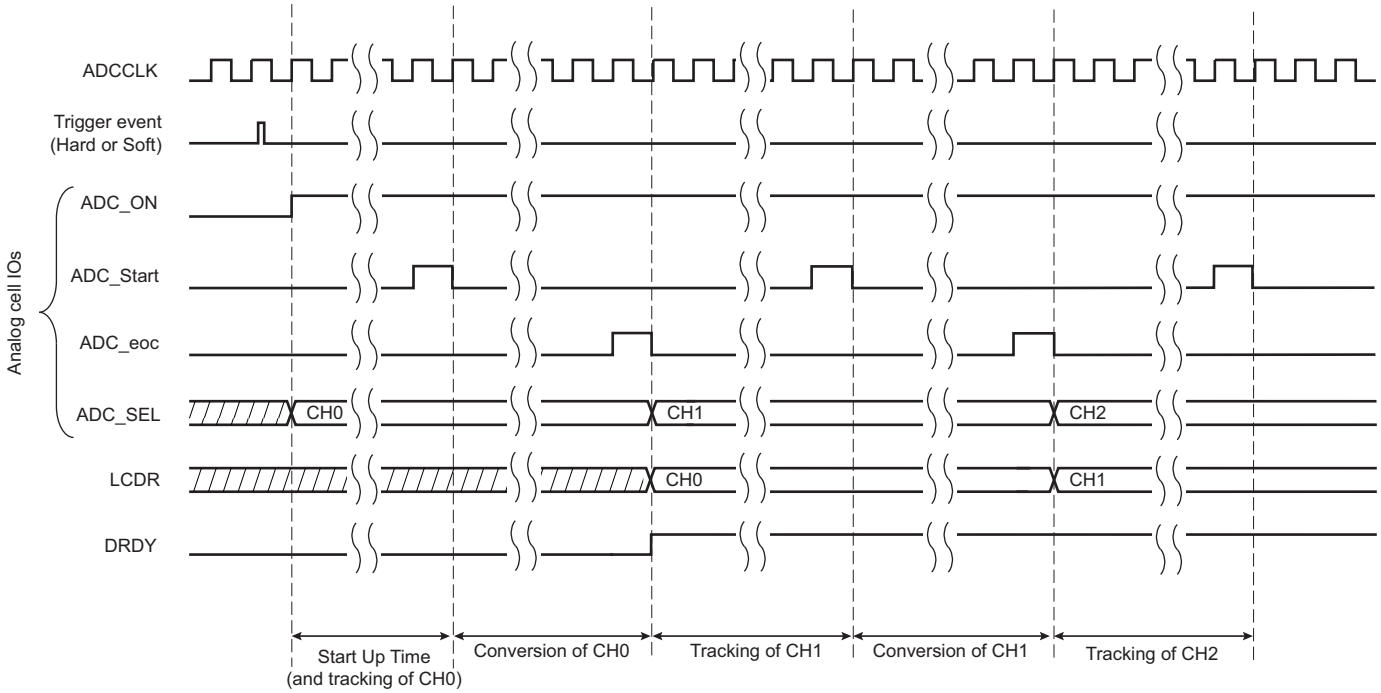
47.6 Functional Description

47.6.1 Analog-to-Digital Conversion

Once the programmed startup time (ADC_MR.STARTUP) has elapsed, ADC conversions are sequenced by three operating times:

- Tracking time—the time for the ADC to charge its input sampling capacitor to the input voltage. The tracking time is always performed before the conversion time and can be configured using the TRACKTIM field in the Mode Register (ADC_MR).
- ADC inherent conversion time—the time for the ADC to convert the sampled analog voltage. This time is constant and is defined from start of conversion to end of conversion.
- Channel conversion period—the effective time between the end of the current channel conversion and the end of the next channel conversion.

Figure 47-2. Sequence of Consecutive ADC Conversions



47.6.2 ADC Clock

The ADC uses the ADC clock (ADCCLK) to perform conversions. The ADC clock frequency is selected in the PRESCAL field of ADC_MR.

The ADC clock frequency is between $f_{peripheral\ clock}/2$, if PRESCAL is 0, and $f_{peripheral\ clock}/512$, if PRESCAL is set to 255 (0xFF).

PRESCAL must be programmed to provide the ADC clock frequency parameter given in the section “Electrical Characteristics”.

47.6.3 ADC Reference Voltage

The conversion is performed on a full range between 0V and the reference voltage pin ADVREF.

Analog inputs between these voltages convert to values based on a linear conversion.

47.6.4 Conversion Resolution

The ADC analog cell features a 10-bit resolution.

The ADC digital controller provides enhanced resolution up to 12 bits.

The ADC digital controller embeds circuitry to reduce the resolution down to 8 bits.

The 8-bit selection is performed by setting the LOWRES bit in ADC_MR. By default, after a reset, the resolution is the highest and the DATA field in the data registers is fully used. By setting the LOWRES bit, the ADC switches to the lowest resolution and the conversion results can be read in the lowest significant bits of the data registers. The two highest bits of the DATA field in the corresponding Channel Data Register (ADC_CDR) and of the LDATA field in the Last Converted Data Register (ADC_LCDR) read 0.

If ADTRG is asynchronous to the ADC peripheral clock, the internal resynchronization introduces a jitter of 1 peripheral clock. This jitter may reduce the resolution of the converted signal.

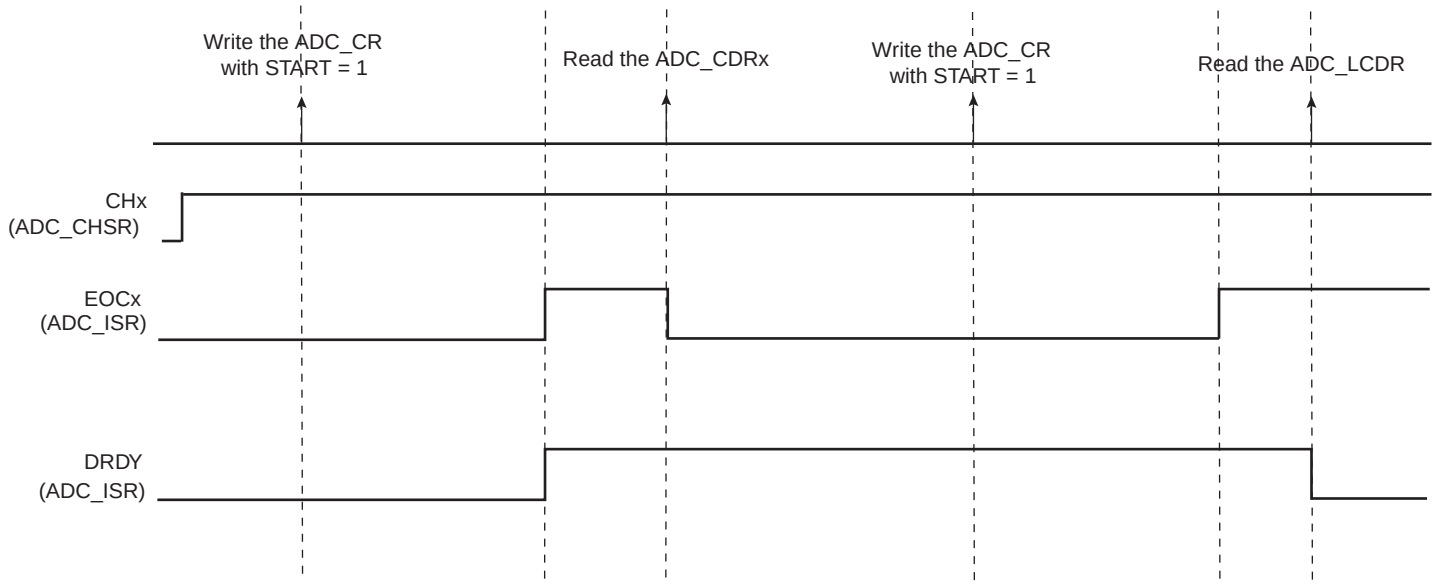
47.6.5 Conversion Results

When a conversion is completed, the resulting digital value is stored in the Channel Data Register (ADC_CDRx) of the current channel and in the ADC Last Converted Data Register (ADC_LCDR). By setting the TAG option in the Extended Mode Register (ADC_EMR), ADC_LCDR presents the channel number associated with the last converted data in the CHNB field.

When a conversion is completed, the channel EOC bit and the DRDY bit in the Interrupt Status Register (ADC_ISR) are set. In the case of a connected DMA channel, DRDY rising triggers a data request. In any case, either EOC and DRDY can trigger an interrupt.

Reading one of the ADC_CDRx clears the corresponding EOC bit. Reading ADC_LCDR clears the DRDY bit.

Figure 47-3. EOCx and DRDY Flag Behavior

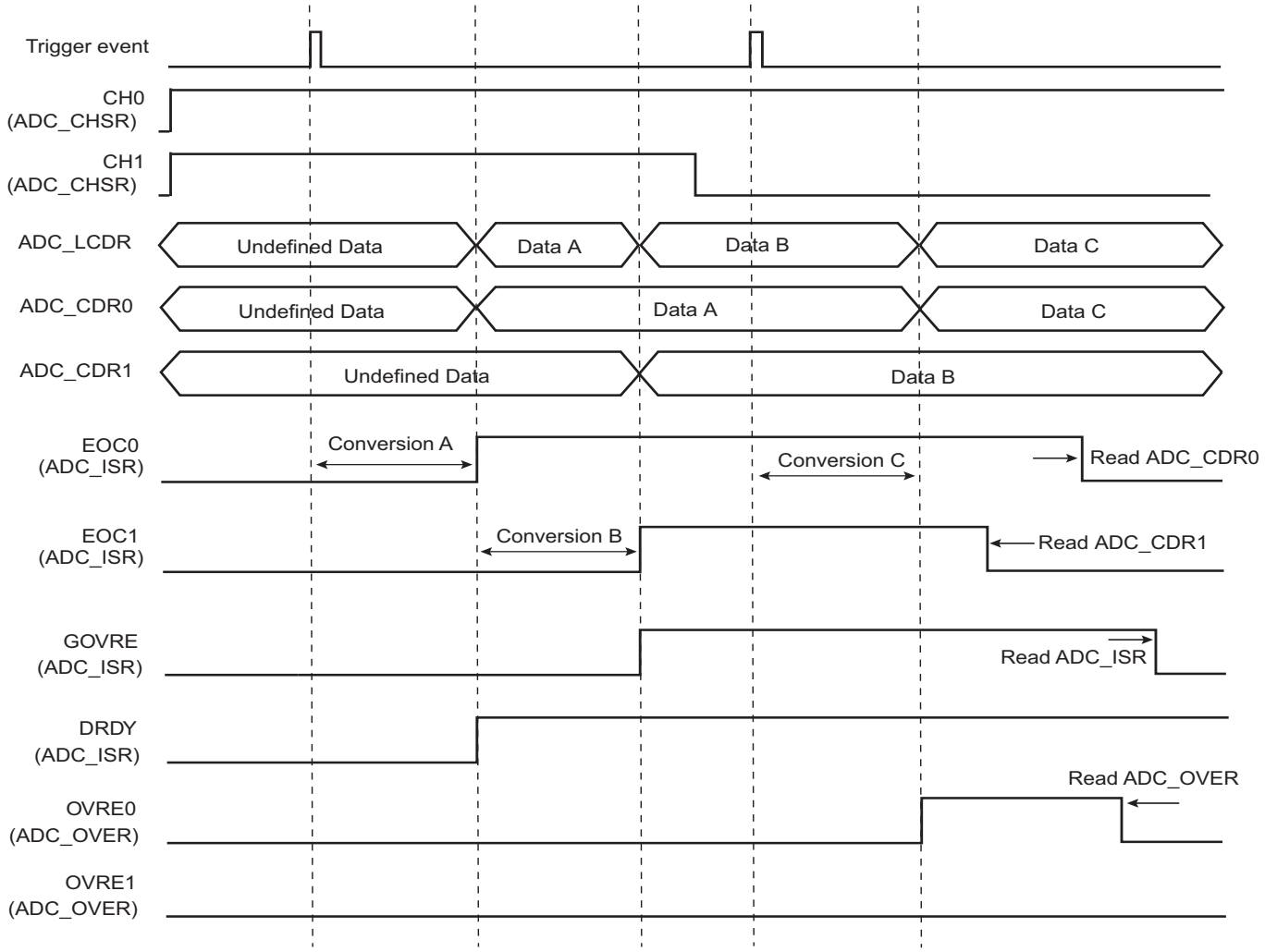


If ADC_CDR is not read before further incoming data is converted, the corresponding OVREx flag is set in the Overrun Status Register (ADC_OVER).

New data converted when DRDY is high sets the GOVRE bit in ADC_ISR.

The OVREx flag is automatically cleared when ADC_OVER is read, and the GOVRE flag is automatically cleared when ADC_ISR is read.

Figure 47-4. EOCx, OVREx and GOVREx Flag Behavior



Warning: If the corresponding channel is disabled during a conversion or if it is disabled and then reenabled during a conversion, its associated data and corresponding EOCx and GOVRE flags in ADC_ISR and OVREx flags in ADC_OVER are unpredictable.

47.6.6 Conversion Triggers

Conversions of the active analog channels are started with a software or hardware trigger. The software trigger is provided by writing the Control Register (ADC_CR) with the START bit at 1.

The list of external/internal events is provided in Section 47.7.2 “ADC Mode Register”. The hardware trigger is selected using the TRGSEL field in ADC_MR. The selected hardware trigger is enabled if TRGMOD = 1, 2 or 3 in the ADC Trigger Register (ADC_TRGR).

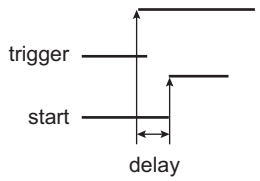
The TRGMOD field in the ADC Trigger Register (ADC_TRGR) selects the hardware trigger from the following:

- any edge, either rising or falling or both, detected on the external trigger pin ADTRG
- the Pen Detect, depending on how the PENDET bit is set in the ADC Touchscreen Mode Register (ADC_TSMR)
- a continuous trigger, meaning the ADC Controller restarts the next sequence as soon as it finishes the current one
- a periodic trigger, which is defined by programming the TRGPER field in ADC_TRGR

The minimum time between two consecutive trigger events must be strictly greater than the duration time of the longest conversion sequence according to configuration of registers ADC_MR, ADC_CHSR, ADC_SEQRx, ADC_TSMR.

If a hardware trigger is selected, the start of a conversion is triggered after a delay starting at each rising edge of the selected signal. Due to asynchronous handling, the delay may vary in a range of two peripheral clock periods to one ADC clock period. This delay introduces sampling jitter in the A/D conversion process and may therefore degrade the conversion performance (e.g., SNR, THD).

Figure 47-5. Hardware Trigger Delay



If one of the TIOA outputs is selected, the corresponding Timer Counter channel must be programmed in Waveform mode.

Only one start command is necessary to initiate a conversion sequence on all the channels. The ADC hardware logic automatically performs the conversions on the active channels, then waits for a new request. The Channel Enable (ADC_CHER) and Channel Disable (ADC_CHDR) registers enable the analog channels to be enabled or disabled independently.

If the ADC is used with a DMA, only the transfers of converted data from enabled channels are performed and the resulting data buffers should be interpreted accordingly.

47.6.7 Sleep Mode and Conversion Sequencer

The ADC Sleep mode maximizes power saving by automatically deactivating the ADC when it is not being used for conversions. Sleep mode is selected by setting the SLEEP bit in ADC_MR.

Sleep mode is managed by a conversion sequencer, which automatically processes the conversions of all channels at lowest power consumption.

This mode can be used when the minimum period of time between two successive trigger events is greater than the startup period of the ADC. See section “Electrical Characteristics”.

When a start conversion request occurs, the ADC is automatically activated. As the analog cell requires a startup time, the logic waits during this time and starts the conversion on the enabled channels. When all conversions are complete, the ADC is deactivated until the next trigger. Triggers occurring during the sequence are ignored.

The conversion sequencer allows automatic processing with minimum processor intervention and optimized power consumption. Conversion sequences can be performed periodically using the internal timer (ADC_TRGR) or the PWM event line. The periodic acquisition of several samples can be processed automatically without any intervention of the processor via the DMA.

The sequence can be customized by programming the Sequence Channel Register ADC_SEQR1 and setting the USEQ bit of the Mode Register (ADC_MR). The user can choose a specific order of channels and can program up to 5 conversions by sequence. The user is free to create a personal sequence by writing channel numbers in ADC_SEQR1. Not only can channel numbers be written in any sequence, channel numbers can be repeated several times. When the bit USEQ in ADC_MR is set, the fields USCHx in ADC_SEQR1 are used to define the sequence. Only enabled USCHx fields will be part of the sequence. Each USCHx field has a corresponding enable, CHx-1, in ADC_CHER.

If all ADC channels (i.e., 5) are used on an application board, there is no restriction of usage of the user sequence. However, if some ADC channels are not enabled for conversion but rather used as pure digital inputs, the respective indexes of these channels cannot be used in the user sequence fields (refer to ADC_SEQRx). For example, if channel 4 is disabled (ADC_CSR[4] = 0), ADC_SEQRx fields USCH1 up to USCH5 must not contain the value 4. Thus the length of the user sequence may be limited by this behavior.

As an example, if only four channels over 5 (CH0 up to CH3) are selected for ADC conversions, the user sequence length cannot exceed four channels. Each trigger event may launch up to four successive conversions of any combination of channels 0 up to 3 but no more (i.e., in this case the sequence CH0, CH0, CH1, CH1, CH1 is impossible).

A sequence that repeats the same channel several times requires more enabled channels than channels actually used for conversion. For example, the sequence CH0, CH0, CH1, CH1 requires four enabled channels (four free channels on application boards) whereas only CH0, CH1 are really converted.

Note: The reference voltage pins always remain connected in Normal mode as in Sleep mode.

47.6.8 Comparison Window

The ADC Controller features automatic comparison functions. It compares converted values to a low threshold, a high threshold or both, depending on the value of the CMPMODE bit in ADC_EMR. The comparison can be done on all channels or only on the channel specified in the CMPSEL field of ADC_EMR. To compare all channels, the CMPALL bit of ADC_EMR must be set.

Moreover, a filtering option can be set by writing the number of consecutive comparison matches needed to raise the flag. This number can be written and read in the CMPFILTER field of ADC_EMR. The filtering option is dedicated to reinforce the detection of an analog signal overpassing a predefined threshold. The filter is cleared as soon as ADC_ISR is read, so this filtering function must be used with peripheral DMA controller and works only when using Interrupt mode (no polling).

The flag can be read on the COMPE bit of the Interrupt Status Register (ADC_ISR) and can trigger an interrupt.

The high threshold and the low threshold can be read/write in the Compare Window Register (ADC_CWR).

If the comparison window is to be used with the LOWRES bit set in ADC_MR, the thresholds do not need to be adjusted, as the adjustment is done internally. However, whether the LOWRES bit is set or not, thresholds must always be configured in accordance with the maximum ADC resolution.

47.6.9 ADC Timings

The ADC startup time is programmed through the STARTUP field in ADC_MR. See section “Electrical Characteristics”.

The ADC controller provides a tracking time of ADC clock cycles.

A minimal tracking time is necessary for the ADC to guarantee the best converted final value between two channel selections. This time must be programmed in the TRACKTIM field in ADC_MR.

Warning: No input buffer amplifier to isolate the source is included in the ADC. This must be taken into consideration. See section “Electrical Characteristics”.

47.6.10 Enhanced Resolution Mode and Digital Averaging Function

47.6.10.1Enhanced Resolution Mode

The Enhanced Resolution mode is enabled if LOWRES is cleared in ADC_MR, and the OSR field is configured to 1 or 2 in ADC_EMR. The enhancement is based on a digital averaging function.

There is no averaging on the last index channel if the measure is triggered by an RTC event.

In this mode, the ADC Controller will trade off conversion speed against accuracy by averaging multiple samples, thus providing a digital low-pass filter function.

$$\text{ADC\_LCDR.LDATA} = \frac{1}{M} \times \sum_{k=0}^{k=N-1} ADC(k)$$

where N and M are given in the table below.

Table 47-4. Digital Averaging Function Configuration versus OSR Values

ADC_EMR.OSR Value	ADC_LCDR.LDATA Length	N Value	M Value	Full Scale Value	Maximum Value
0	12 bits	1	1	4095	4095
1	13 bits	4	2	8191	8190
2	14 bits	16	4	16383	16381

The average result is valid in ADC_CDRx (x corresponds to the index of the channel) only if the EOCn flag is set in ADC_ISR and if the OVREn flag is cleared in ADC_OVER. The average result for all channels is valid in ADC_LCDR only if DRDY is set and GOVRE is cleared in ADC_ISR.

Note that ADC_CDRs are not buffered. Therefore, when an averaging sequence is ongoing, the value in these registers changes after each averaging sample. However, overrun flags in ADC_OVER rise as soon as the first sample of an averaging sequence is received. Thus the previous averaged value is not read, even if the new averaged value is not ready.

Consequently, when an overrun flag rises in ADC_OVER, it means that the previous unread data is lost but it does not mean that this data has been overwritten by the new averaged value as the averaging sequence concerning this channel can still be ongoing.

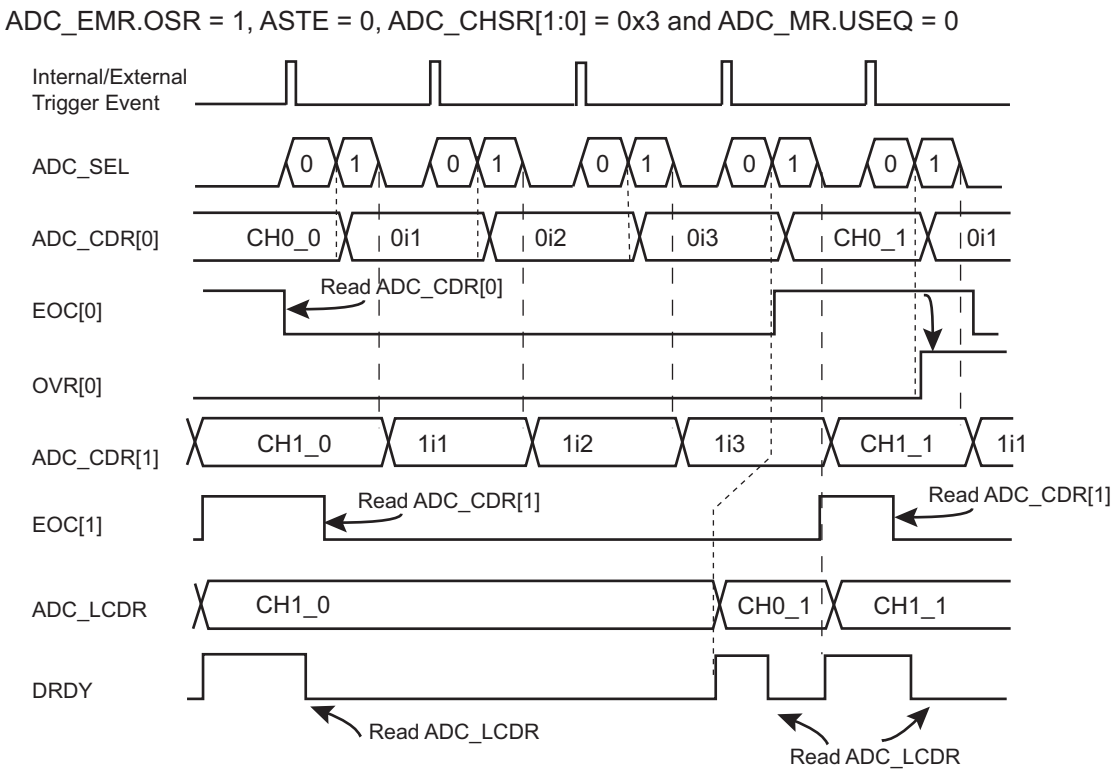
When an oversampling is performed, the maximum value that can be read on ADC_CDRx or ADC_LCDR is not the full-scale value, even if the maximum voltage is supplied on the analog input. Refer to Table 47-4 “Digital Averaging Function Configuration versus OSR Values”.

47.6.10.2Averaging Function versus Trigger Events

The samples can be defined in different ways for the averaging function depending on the configuration of the ASTE bit in ADC_EMR and the USEQ bit in ADC_MR.

When USEQ = 0, there are two possible ways to generate the averaging through the trigger event. If ASTE = 0 in ADC_EMR, every trigger event generates one sample for each enabled channel as described in Figure 47-6. Therefore four trigger events are requested to get the result of averaging if OSR = 1.

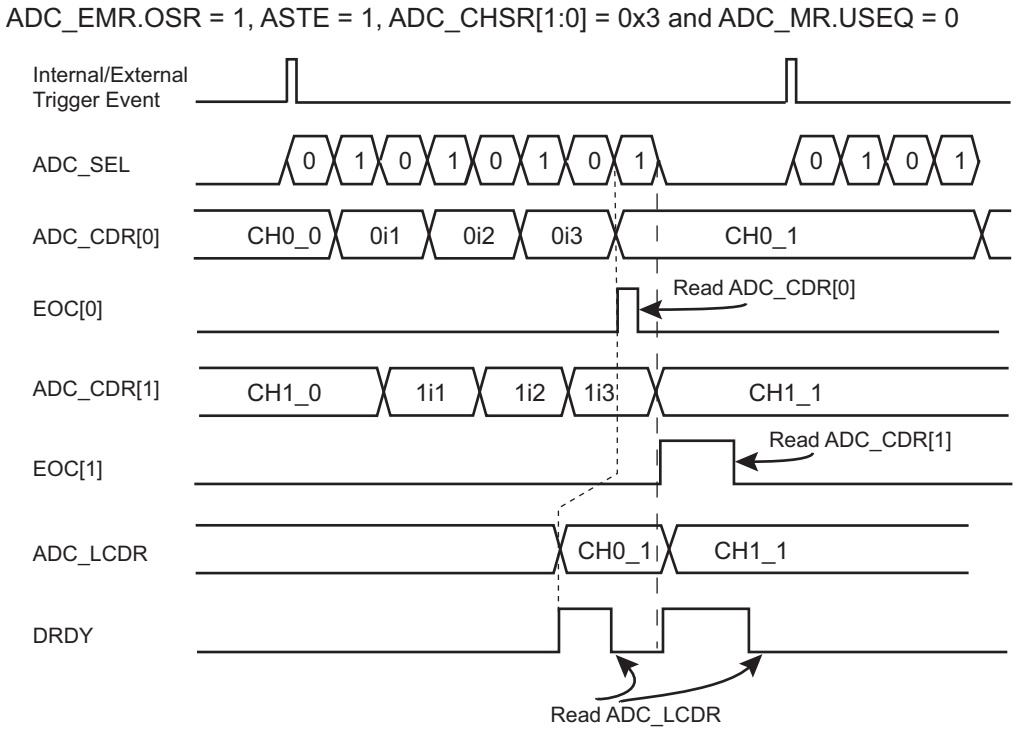
Figure 47-6. Digital Averaging Function Waveforms Over Multiple Trigger Events



Notes: ADC_SEL: Command to the ADC analog cell
0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0_0, CH0_1, CH1_0 and CH1_1 are final results of average function.

If ASTE = 1 in ADC_EMR and USEQ = 0 in ADC_MR, the sequence to be converted, defined in ADC_CHSR, is automatically repeated n times (where n corresponds to the oversampling ratio defined in the OSR field in ADC_EMR). As a result, only one trigger is required to obtain the result of the averaging function as described in Figure 47-7.

Figure 47-7. Digital Averaging Function Waveforms on a Single Trigger Event



Note: ADC_SEL: Command to the ADC analog cell
0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0_0, CH0_1, CH1_0 and CH1_1 are final results of average function.

When USEQ = 1, the user can define the channel sequence to be converted by configuring ADC_SEQRx and ADC_CHER so that channels are not interleaved during the averaging period. Under these conditions, a sample is defined for each end of conversion as described in Figure 47-8.

When USEQ = 1 and ASTE = 1, OSR can be only configured to 1. Up to three channels can be converted in this mode. The averaging result will be placed in the corresponding ADC_CDRx and in ADC_LCDR for each trigger event. The ADC real sample rate remains the maximum ADC sample rate divided by 4.

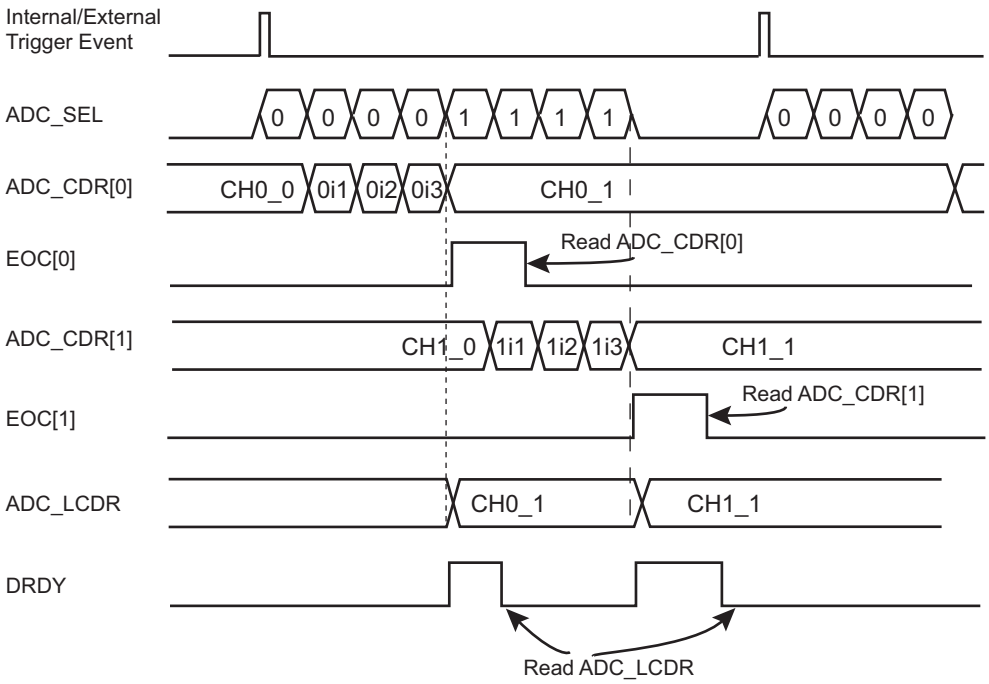
It is important that the user sequence follows a specific pattern. The user sequence must be programmed in such a way that it generates a stream of conversion, where a same channel is successively converted.

Table 47-5. Example Sequence Configurations (USEQ = 1, ASTE = 1, OSR = 1)

	Number of Channels Non-interleaved Averaging - Register Value		
Register	**1 (e.g., CH0)**	**2 (e.g., CH0, CH1)**	**3 (e.g., CH0, CH1, CH2)**
ADC_CHSR	0x0000_000F	0x0000_00FF	0x0000_0FFF
ADC_SEQR1	0x0000_0000	0x1111_0000	0x1111_0000
ADC_SEQR2	0x0000_0000	0x0000_0000	0x0000_2222

Figure 47-8. Digital Averaging Function Waveforms on a Single Trigger Event, Non-interleaved

ADC_EMR.OSR = 1, ASTE = 1, ADC_CHSR[7:0] = 0xFF and ADC_MR.USEQ = 1
ADC_SEQR1 = 0x1111_0000



Note: ADC_SEL: Command to the ADC analog cell
0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0_0, CH0_1, CH1_0 and CH1_1 are final results of average function.

47.6.11 Touchscreen

47.6.11.1Touchscreen Mode

The TSMODE parameter of the ADC Touchscreen Mode Register (ADC_TSMR) is used to enable/disable the touchscreen functionality, to select the type of screen (4-wire or 5-wire) and, in the case of a 4-wire screen and to activate (or not) the pressure measurement.

In 4-wire mode, channel 0, 1, 2 and 3 must not be used for classic ADC conversions. Likewise, in 5-wire mode, channel 0, 1, 2, 3, and 4 must not be used for classic ADC conversions.

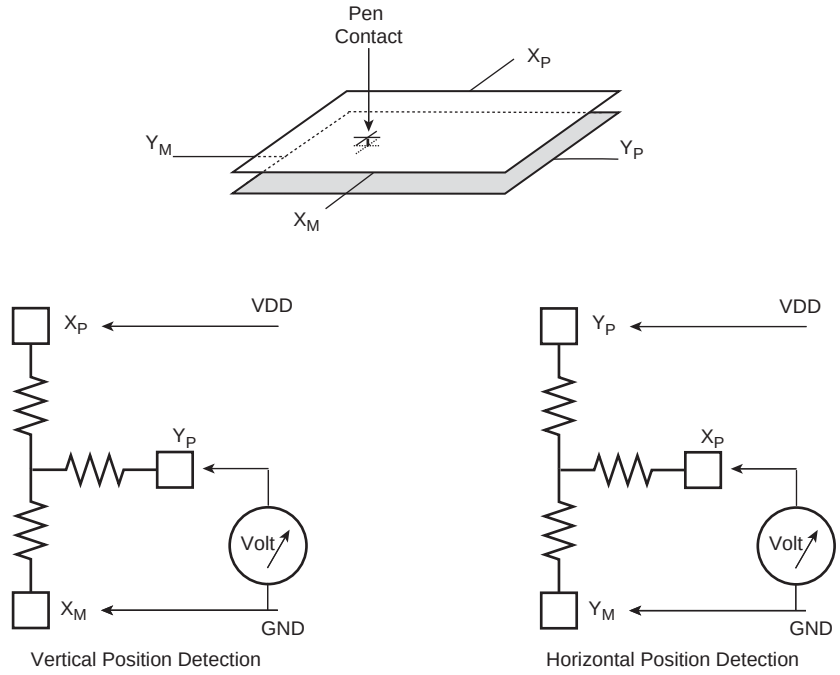
47.6.11.24-wire Resistive Touchscreen Principles

A resistive touchscreen is based on two resistive films, each one being fitted with a pair of electrodes, placed at the top and bottom on one film, and on the right and left on the other. In between, there is a layer acting as an insulator, but also enables contact when you press the screen. This is illustrated in Figure 47-9.

The ADC controller has the ability to perform without external components:

- position measurement
- pressure measurement
- pen detection

Figure 47-9. Touchscreen Position Measurement



47.6.11.3 4-wire Position Measurement Method

As shown in Figure 47-9, to detect the position of a contact, a supply is first applied from top to bottom. Due to the linear resistance of the film, there is a voltage gradient from top to bottom. When a contact is performed on the screen, the voltage propagates at the point the two surfaces come into contact with the second film. If the input impedance on the right and left electrodes sense is high enough, the film does not affect this voltage, despite its resistive nature.

For the horizontal direction, the same method is used, but by applying supply from left to right. The range depends on the supply voltage and on the loss in the switches that connect to the top and bottom electrodes.

In an ideal world (linear, with no loss through switches), the horizontal position is equal to:

VY_M / VDD or VY_P / VDD.

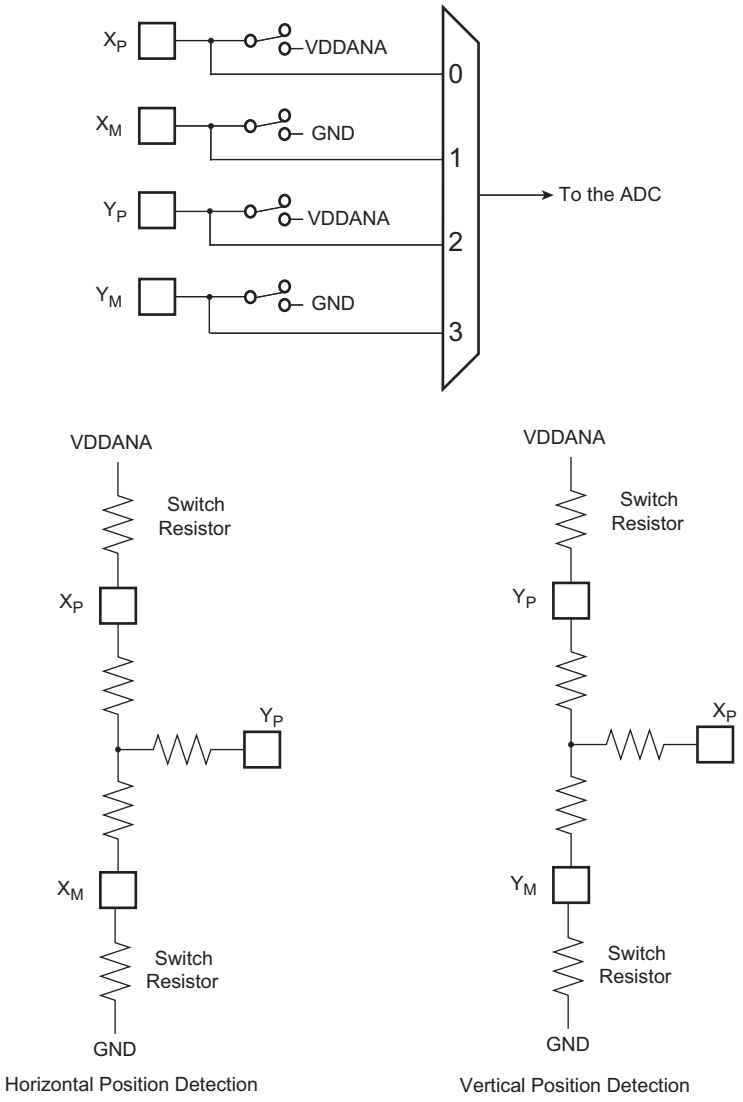
The implementation with on-chip power switches is shown in Figure 47-10. The voltage measurement at the output of the switch compensates for the switches loss.

It is possible to correct for switch loss by performing the operation:

$[VY_P - VX_M] / [VX_P - VX_M]$.

This requires additional measurements, as shown in Figure 47-10.

Figure 47-10. Touchscreen Switches Implementation



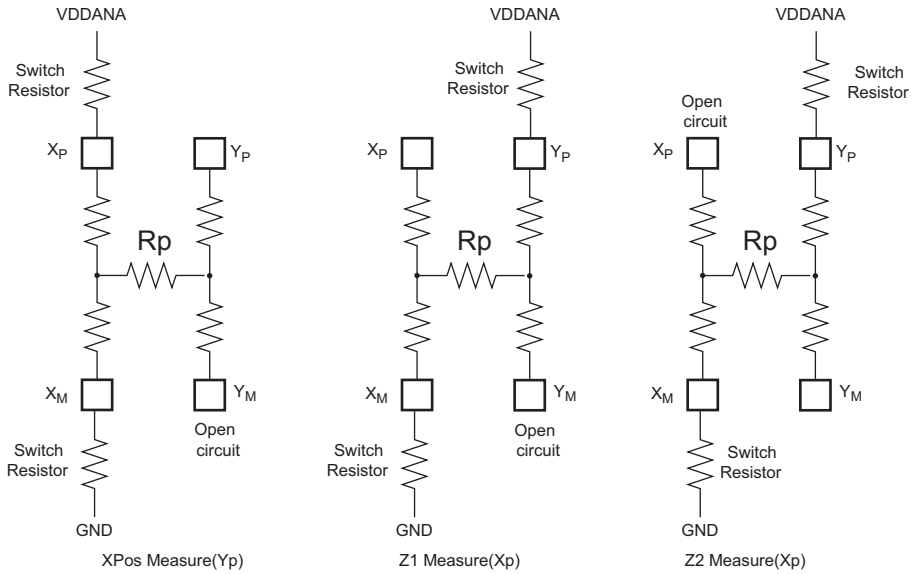
47.6.11.4 4-wire Pressure Measurement Method

The method to measure the pressure (Rp) applied to the touchscreen is based on the known resistance of the X-Panel resistance (Rxp).

Three conversions (Xpos,Z1,Z2) are necessary to determine the value of Rp (Zaxis resistance).

$Rp = Rxp \times (Xpos/1024) \times [(Z2/Z1)-1]$

Figure 47-11. Pressure Measurement



47.6.11.5 5-wire Resistive Touchscreen Principles

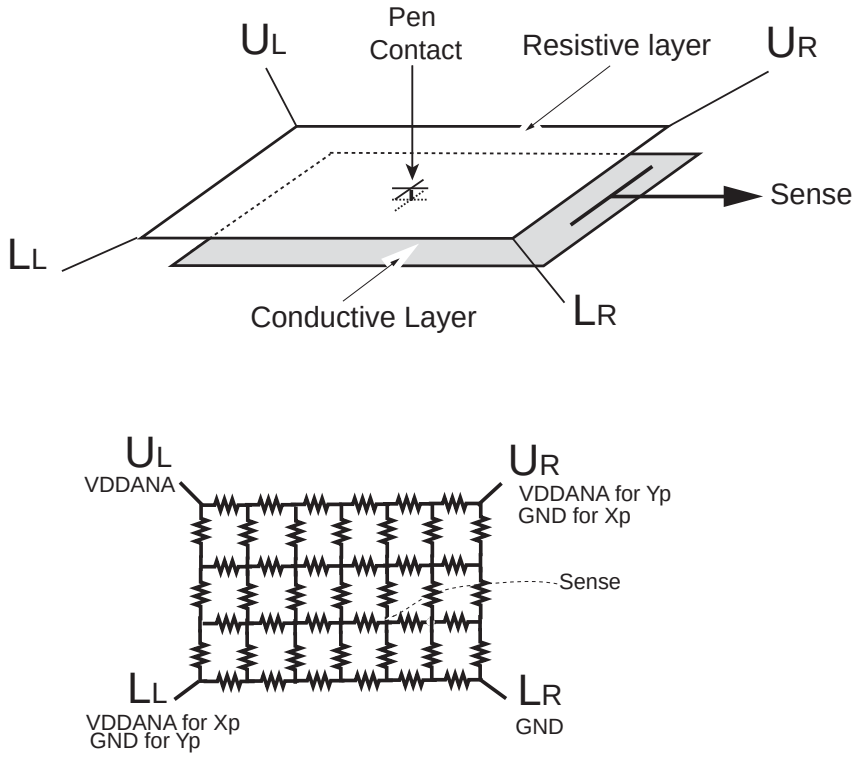
To make a 5-wire touchscreen, a resistive layer with a contact point at each corner and a conductive layer are used.

The 5-wire touchscreen differs from the 4-wire type mainly in that the voltage gradient is applied only to one layer, the resistive layer, while the other layer is the sense layer for both measurements.

The measurement of the X position is obtained by biasing the upper left corner and lower left corner to VDDANA and the upper right corner and lower right to ground.

To measure along the Y axis, bias the upper left corner and upper right corner to V_{DDANA} and bias the lower left corner and lower right corner to ground.

Figure 47-12. 5-Wire Principle



47.6.11.65-wire Position Measurement Method

In an application only monitoring clicks, 100 points per second is typically needed. For handwriting or motion detection, the number of measurements to consider is approximately 200 points per second. This must take into account that multiple measurements are included (oversampling, filtering) to compute the correct point.

The 5-wire touchscreen panel works by applying a voltage at the corners of the resistive layer and measuring the vertical or horizontal resistive network with the sense input. The ADC converts the voltage measured at the point the panel is touched.

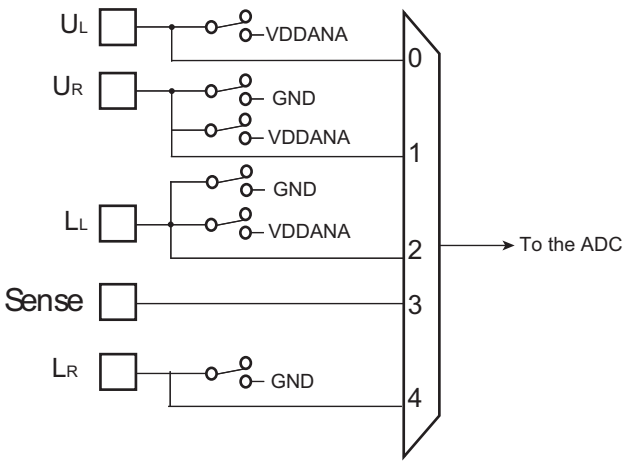
A measurement of the Y position of the pointing device is made by:

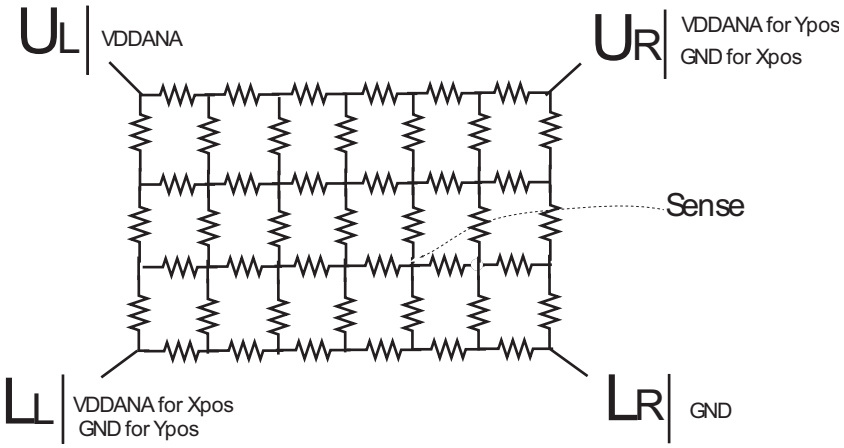
- Connecting Upper left (UL) and upper right (UR) corners to VDDANA
- Connecting Lower left (LL) and lower right (LR) corners to ground.
- The voltage measured is determined by the voltage divider developed at the point of touch (Yposition) and the SENSE input is converted by ADC.

A measurement of the X position of the pointing device is made by:

- Connecting the upper left (UL) and lower left (LL) corners to ground
- Connecting the upper right and lower right corners to VDDANA.
- The voltage measured is determined by the voltage divider developed at the point of touch (Xposition) and the SENSE input is converted by ADC.

Figure 47-13. Touchscreen Switches Implementation



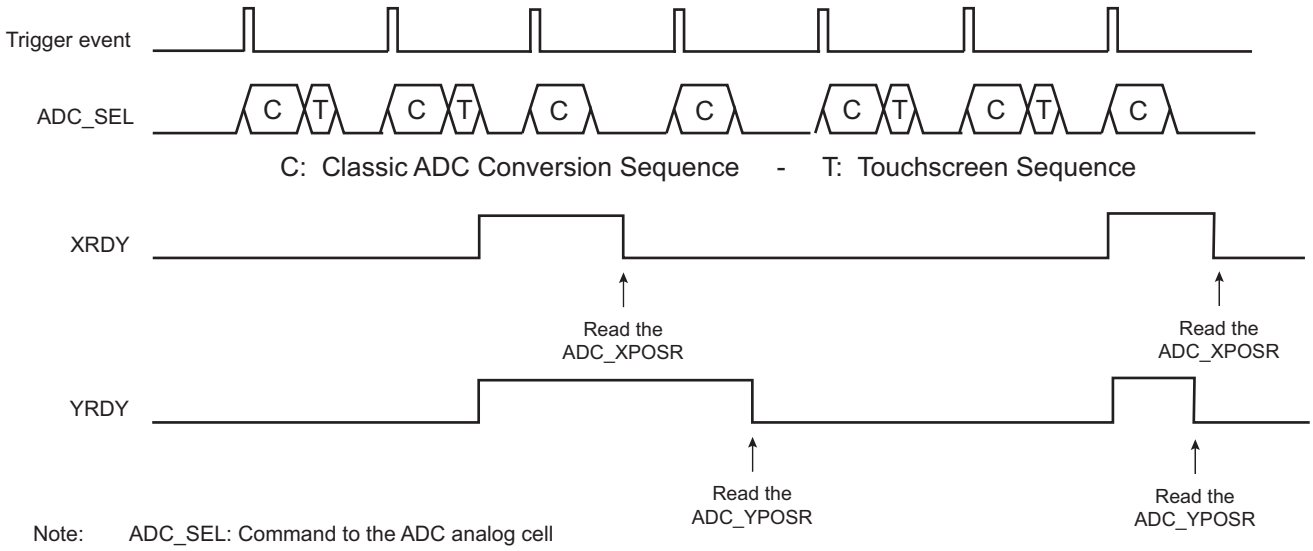


47.6.11.7 Sequence and Noise Filtering

The ADC Controller can manage ADC conversions and touchscreen measurement. On each trigger event the sequence of ADC conversions is performed as described in Section 47.6.7 “Sleep Mode and Conversion Sequencer”. The touchscreen measure frequency can be specified in number of trigger events by writing the TSFREQ parameter in ADC_TSMR. An internal counter counts triggers up to TSFREQ, and every time it rolls out, a touchscreen sequence is appended to the classic ADC conversion sequence (see Figure 47-14).

Additionally the user can average multiple touchscreen measures by writing the TSAV parameter in ADC_TSMR. This can be 1, 2, 4 or 8 measures performed on consecutive triggers as illustrated in Figure 47-14 below. Consequently, the TSFREQ parameter must be greater or equal to the TSAV parameter.

Figure 47-14. Insertion of Touchscreen Sequences (TSFREQ = 2; TSAV = 1)



47.6.11.8 Measured Values, Registers and Flags

As soon as the controller finishes the Touchscreen sequence, XRDY, YRDY and PRDY are set and can generate an interrupt. These flags can be read in the ADC Interrupt Status Register (ADC_ISR). They are reset independently by reading in the ADC Touchscreen X Position Register (ADC_XPOSR), the ADC Touchscreen Y Position Register (ADC_YPOSR) and the ADC Touchscreen Pressure Register (ADC_PRESSR).

ADC_XPOSR presents XPOS (V_X - V_{Xmin}) on its LSB and XSCALE (V_{XMAX} - V_{Xmin}) aligned on the 16th bit.

ADC_YPOSR presents YPOS (V_Y - V_{Ymin}) on its LSB and YSCALE (V_{YMAX} - V_{Ymin}) aligned on the 16th bit.

To improve the quality of the measure, the user must calculate XPOS/XSCALE and YPOS/YSCALE.

V_{XMAX}, V_{Xmin}, V_{YMAX}, and V_{Ymin} are measured at the first startup of the controller. These values can change during use, so it can be necessary to refresh them. Refresh can be done by writing ‘1’ in the TSCALIB field of the control Register (ADC_CR).

ADC_PRESSR presents Z1 on its LSB and Z2 aligned on the 16th bit. See Section 47.6.11.4 “4-wire Pressure Measurement Method”.

47.6.11.9 Pen Detect Method

When there is no contact, it is not necessary to perform a conversion. However, it is important to detect a contact by keeping the power consumption as low as possible.

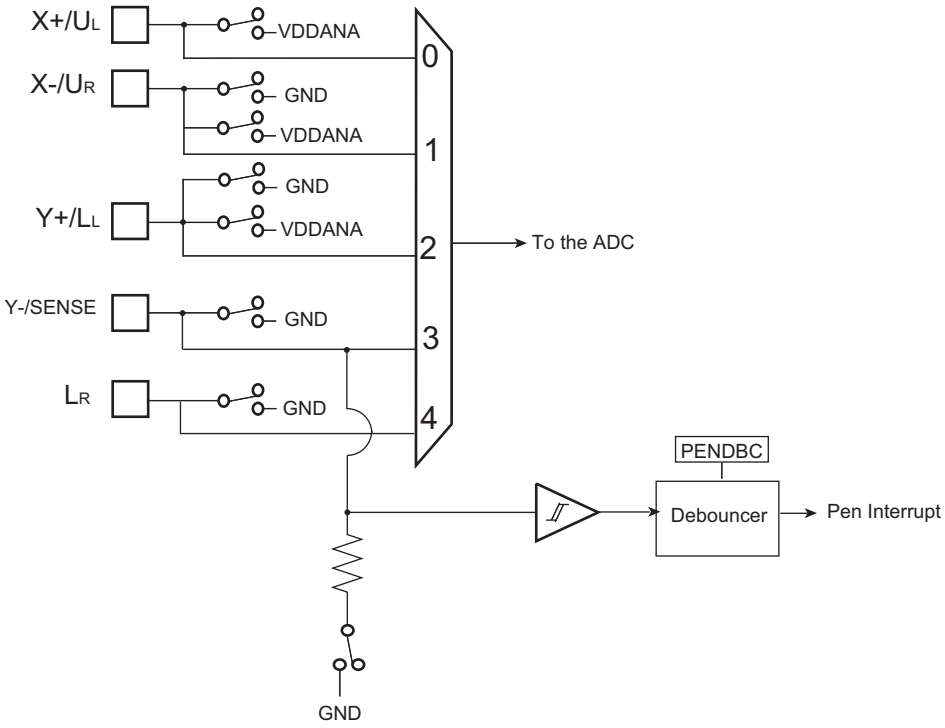
The implementation polarizes one panel by closing the switch on (X_P/U_L) and ties the horizontal panel by an embedded resistor connected to Y_M / Sense. This resistor is enabled by a fifth switch. Since there is no contact, no current is flowing and there is no related power consumption. As soon as a contact occurs, a current is flowing in the Touchscreen and a Schmitt trigger detects the voltage in the resistor.

The Touchscreen Interrupt configuration is entered by programming the PENDET bit in ADC_TSMR. If this bit is written at 1, the controller samples the pen contact state when it is not converting and waiting for a trigger.

To complete the circuit, a programmable debouncer is placed at the output of the Schmitt trigger. This debouncer is programmable up to 2^{15} ADC clock periods. The debouncer length can be selected by programming the field PENDBC in ADC_TSMR.

Due to the analog switch’s structure, the debouncer circuitry is only active when no conversion (touchscreen or classic ADC channels) is in progress. Thus, if the time between the end of a conversion sequence and the arrival of the next trigger event is lower than the debouncing time configured on PENDBC, the debouncer will not detect any contact.

Figure 47-15. Touchscreen Pen Detect



The touchscreen pen detect can be used to generate an ADC interrupt to wake up the system. The pen detect generates two types of status, reported in ADC_ISR:

- the PEN bit is set as soon as a contact exceeds the debouncing time as defined by PENDBC and remains set until ADC_ISR is read.
- the NOPEN bit is set as soon as no current flows for a time over the debouncing time as defined by PENDBC and remains set until ADC_ISR is read.

Both bits are automatically cleared as soon as ADC_ISR is read, and can generate an interrupt by writing ADC_IER.

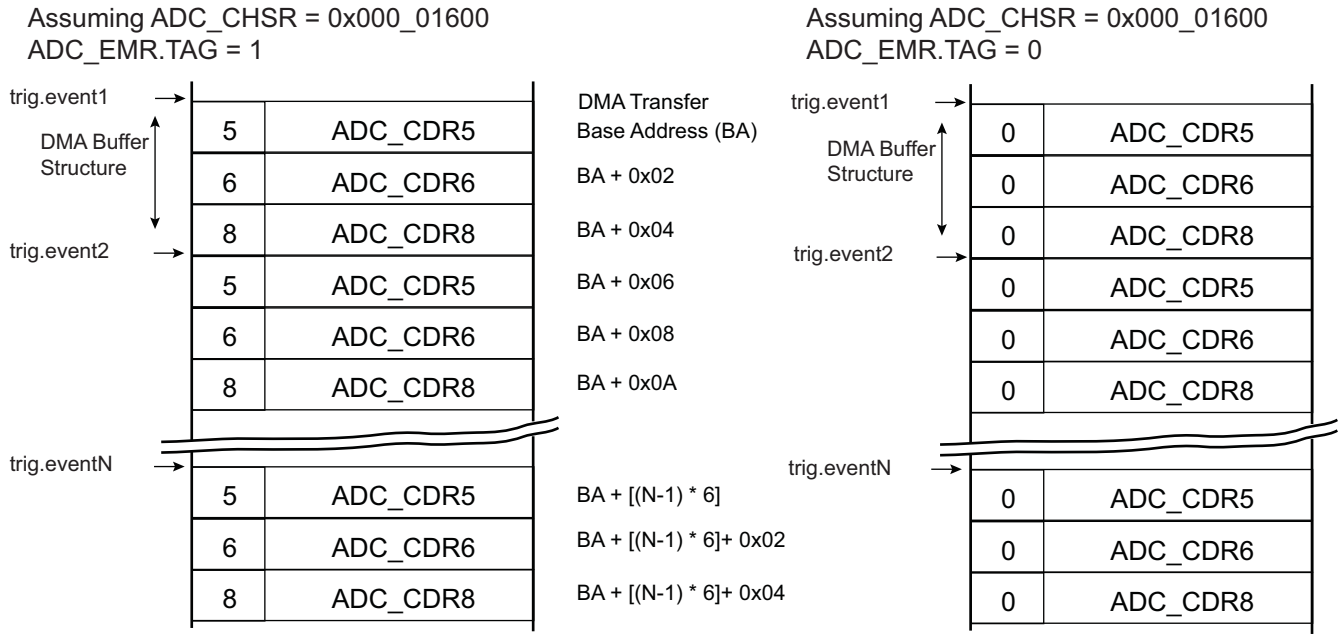
Moreover, the rising of either one of them clears the other, they cannot be set at the same time.

The PENS bit of ADC_ISR shows the current status of the pen contact.

47.6.12 Buffer Structure

The DMA read channel is triggered each time a new data is stored in ADC_LCDR. The same structure of data is repeatedly stored in ADC_LCDR each time a trigger event occurs. Depending on user mode of operation (ADC_MR, ADC_CHSR, ADC_SEQR1, ADC_TSMR) the structure differs. Each data read to DMA buffer, carried on a half-word (16-bit), consists of last converted data right aligned and when TAG is set in ADC_EMR, the four most significant bits are carrying the channel number thus allowing an easier post-processing in the DMA buffer or better checking the DMA buffer integrity.

Figure 47-16. Buffer Structure



As soon as touchscreen conversions are required, the pen detection function may help the post-processing of the buffer. Refer to Section 47.6.12.4 “Pen Detection Status”.

47.6.12.1Classic ADC Channels Only (Touchscreen Disabled)

When no touchscreen conversion is required (i.e., TSMODE = 0 in ADC_TSMR), the structure of data within the buffer is defined by ADC_MR, ADC_CHSR, ADC_SEQRx. See Figure 47-16.

If the user sequence is not used (i.e., USEQ is cleared in ADC_MR) then only the value of ADC_CHSR defines the data structure. For each trigger event, enabled channels will be consecutively stored in ADC_LCDR and automatically read to the buffer.

When the user sequence is configured (i.e., USEQ is set in ADC_MR) not only does ADC_CHSR modify the data structure of the buffer, but ADC_SEQRx registers may modify the data structure of the buffer as well.

47.6.12.2Touchscreen Channels Only

When only touchscreen conversions are required (i.e., TSMODE ≠ 0 in ADC_TSMR and ADC_CHSR equals 0), the structure of data within the buffer is defined by ADC_TSMR.

When TSMODE = 1 or 3, each trigger event adds two half-words in the buffer (assuming TSAV = 0), first half-word being XPOS of ADC_XPOSR then YPOS of ADC_YPOSR. If TSAV/TSFREQ ≠ 0, the data structure remains unchanged. Not all trigger events add data to the buffer.

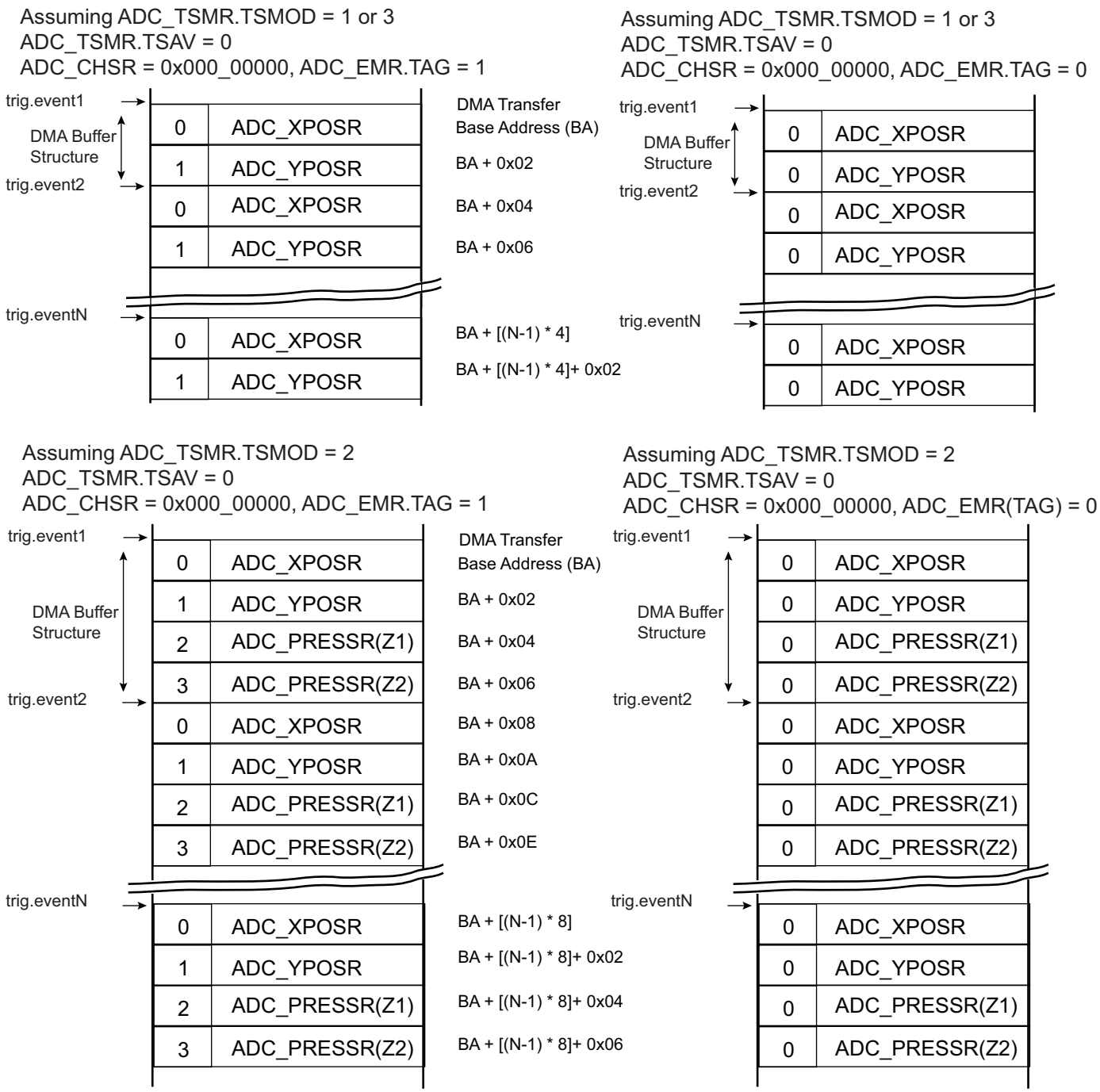
When TSMODE = 2, each trigger event adds four half-words to the buffer (assuming TSAV = 0), first half-word being XPOS of ADC_XPOSR followed by YPOS of ADC_YPOSR and finally Z1 followed by Z2, both located in ADC_PRESSR.

When TAG is set (ADC_EMR), the CHNB field (four most significant bits of ADC_LCDR) is cleared when XPOS is transmitted and set when YPOS is transmitted, allowing an easier post-processing of the buffer or a better checking of the buffer integrity. In case 4-wire with Pressure mode is selected, Z1 value is transmitted to the buffer along with tag set to 2 and Z2 is tagged with value 3.

XSCALE and YSCALE (calibration values) are not transmitted to the buffer because they are supposed to be constant and moreover only measured at the very first startup of the controller or upon user request.

There is no change in buffer structure whatever the value of PENDET bit configuration in ADC_TSMR but it is recommended to use the pen detection function for buffer post-processing (refer to Section 47.6.12.4 “Pen Detection Status”).

Figure 47-17. Buffer Structure When Only Touchscreen Channels are Enabled



47.6.12.3Interleaved Channels

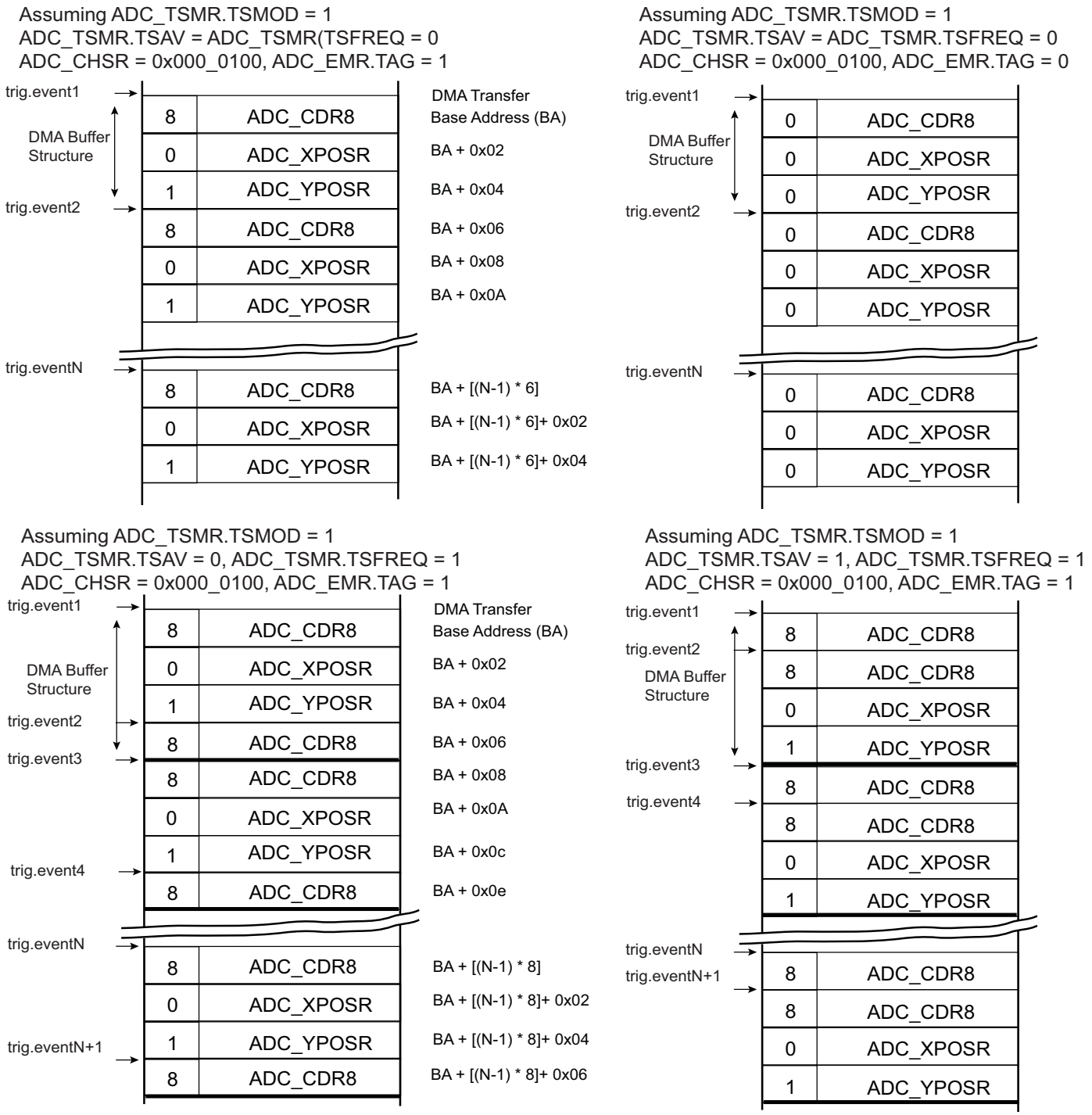
When both classic ADC channels (CH4/CH5 up to CH5 are set in ADC_CHSR) and touchscreen conversions are required (TSMODE ≠ 0 in ADC_TSMR) the structure of the buffer differs according to TSAV and TSFREQ values.

If TSFREQ ≠ 0, not all events generate touchscreen conversions, therefore the buffer structure is based on 2^{TSFREQ} trigger events. Given a TSFREQ value, the location of touchscreen conversion results depends on TSAV value.

When TSFREQ = 0, TSAV must equal 0.

There is no change in buffer structure whatever the value of PENDET bit configuration in ADC_TSMR but it is recommended to use the pen detection function for buffer post-processing (refer to Section 47.6.12.4 “Pen Detection Status”).

Figure 47-18. Buffer Structure When Classic ADC and Touchscreen Channels are Interleaved



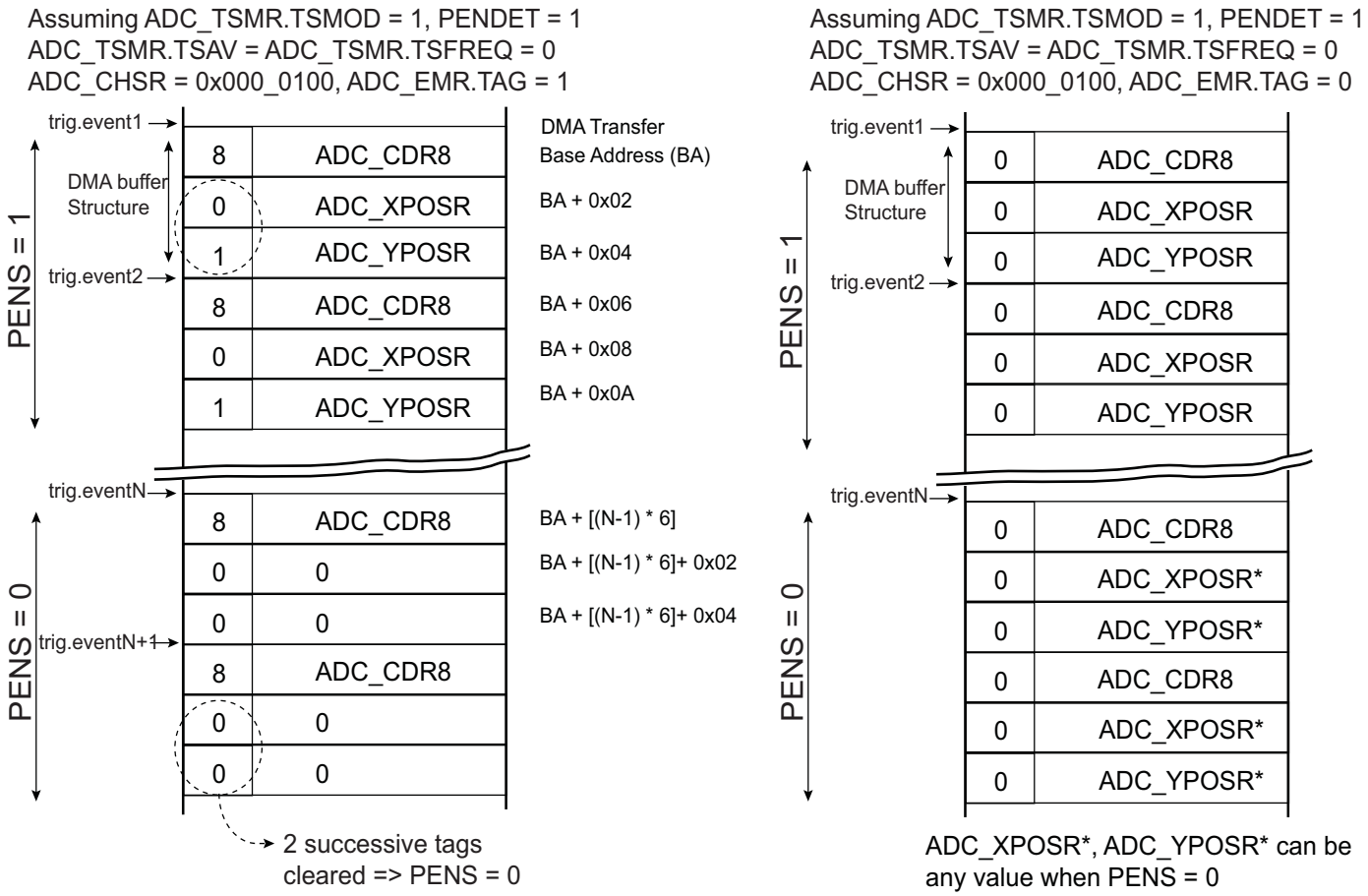
47.6.12.4Pen Detection Status

If the pen detection measure is enabled (PENDET is set in ADC_TSMR), the XPOS, YPOS, Z1, Z2 values transmitted to the buffer through ADC_LCDR are cleared (including the CHNB field), if the PENS flag of ADC_ISR is 0. When the PENS flag is set, XPOS, YPOS, Z1, Z2 are normally transmitted.

Therefore, using pen detection together with tag function eases the post-processing of the buffer, especially to determine which touchscreen converted values correspond to a period of time when the pen was in contact with the screen.

When the pen detection is disabled or the tag function is disabled, XPOS, YPOS, Z1, Z2 are normally transmitted without tag and no relationship can be found with pen status, thus post-processing may not be easy.

Figure 47-19. Buffer Structure With and Without Pen Detection Enabled



47.6.13 Fault Output

The ADC Controller internal fault output is directly connected to PWM fault input. Fault output may be asserted depending on the configuration of ADC_EMR and ADC_CWR and converted values. When the compare occurs, the ADC fault output generates a pulse of one peripheral clock cycle to the PWM fault input. This fault line can be enabled or disabled within PWM. Should it be activated and asserted by the ADC Controller, the PWM outputs are immediately placed in a safe state (pure combinational path). Note that the ADC fault output connected to the PWM is not the COMPE bit. Thus the Fault mode (FMOD) within the PWM configuration must be FMOD = 1.

47.6.14 Register Write Protection

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

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

The WPVS flag is automatically reset by reading ADC_WPSR.

The following registers are write-protected when WPEN is set in ADC_WPMR:

- ADC Mode Register
- ADC Channel Sequence 1 Register
- ADC Channel Enable Register
- ADC Channel Disable Register
- ADC Extended Mode Register
- ADC Compare Window Register
- ADC Analog Control Register
- ADC Touchscreen Mode Register
- ADC Trigger Register

47.7 Analog-to-Digital (ADC) User Interface

Table 47-6. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	ADC_CR	Write-only	–
0x04	Mode Register	ADC_MR	Read/Write	0x00000000
0x08	Channel Sequence Register 1	ADC_SEQR1	Read/Write	0x00000000
0x0C	Reserved	–	–	–
0x10	Channel Enable Register	ADC_CHER	Write-only	–
0x14	Channel Disable Register	ADC_CHDR	Write-only	–
0x18	Channel Status Register	ADC_CHSR	Read-only	0x00000000
0x1C	Reserved	–	–	–
0x20	Last Converted Data Register	ADC_LCDR	Read-only	0x00000000
0x24	Interrupt Enable Register	ADC_IER	Write-only	–
0x28	Interrupt Disable Register	ADC_IDR	Write-only	–
0x2C	Interrupt Mask Register	ADC_IMR	Read-only	0x00000000
0x30	Interrupt Status Register	ADC_ISR	Read-only	0x00000000
0x34	Reserved	–	–	–
0x38	Reserved	–	–	–
0x3C	Overrun Status Register	ADC_OVER	Read-only	0x00000000
0x40	Extended Mode Register	ADC_EMR	Read/Write	0x00000000
0x44	Compare Window Register	ADC_CWR	Read/Write	0x00000000
0x48	Reserved	–	–	–
0x4C	Reserved	–	–	–
0x50	Channel Data Register 0	ADC_CDR0	Read-only	0x00000000
0x54	Channel Data Register 1	ADC_CDR1	Read-only	0x00000000
...	...	...	...	...
0x60	Channel Data Register 4	ADC_CDR4	Read-only	0x00000000
0x64–0x90	Reserved	–	–	–
0x94	Analog Control Register	ADC_ACR	Read/Write	0x00000100
0x98–0xAC	Reserved	–	–	–
0xB0	Touchscreen Mode Register	ADC_TSMR	Read/Write	0x00000000
0xB4	Touchscreen X Position Register	ADC_XPOSR	Read-only	0x00000000
0xB8	Touchscreen Y Position Register	ADC_YPOSR	Read-only	0x00000000
0xBC	Touchscreen Pressure Register	ADC_PRESSR	Read-only	0x00000000
0xC0	Trigger Register	ADC_TRGR	Read/Write	0x00000000
0xC4–0xE0	Reserved	–	–	–

Table 47-6. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0xE4	Write Protection Mode Register	ADC_WPMR	Read/Write	0x00000000
0xE8	Write Protection Status Register	ADC_WPSR	Read-only	0x00000000
0xEC–0xFC	Reserved	–	–	–

Note: Any offset not listed in the table must be considered as “reserved”.

47.7.1 ADC Control Register

Name: ADC_CR

Address: 0xFC034000

Access: Write-only

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
–	–	–	–	–	TSCALIB	START	SWRST

- **SWRST: Software Reset**

0: No effect.

1: Resets the ADC, simulating a hardware reset.

- **START: Start Conversion**

0: No effect.

1: Begins analog-to-digital conversion.

- **TSCALIB: Touchscreen Calibration**

0: No effect.

1: Programs screen calibration (VDD/GND measurement)

If conversion is in progress, the calibration sequence starts at the beginning of a new conversion sequence. If no conversion is in progress, the calibration sequence starts at the second conversion sequence located after the TSCALIB command (Sleep mode, waiting for a trigger event).

TSCALIB measurement sequence does not affect the Last Converted Data Register (ADC_LCDR).

47.7.2 ADC Mode Register

Name: ADC_MR

Address: 0xFC034004

Access: Read/Write

31	30	29	28	27	26	25	24
USEQ	–	–	–	TRACKTIM			
23	22	21	20	19	18	17	16
–	–	–	–	STARTUP			
15	14	13	12	11	10	9	8
PRESCAL							
7	6	5	4	3	2	1	0
–	–	SLEEP	LOWRES	TRGSEL			–

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

- **TRGSEL: Trigger Selection**

Value	Name	Description
0	ADC_TRIG0	ADTRG
1	ADC_TRIG1	TIOA0
2	ADC_TRIG2	TIOA1
3	ADC_TRIG3	TIOA2
4	ADC_TRIG4	PWM event line 0
5	ADC_TRIG5	PWM_even line 1
6	ADC_TRIG6	Reserved
7	ADC_TRIG7	–

Note: The trigger selection can be performed only if TRGMOD = 1,2 or 3 in ADC Trigger Register.

- **LOWRES: Resolution**

Value	Name	Description
0	BITS_10	10-bit resolution. For higher resolution by averaging, refer to Section 47.7.13 “ADC Extended Mode Register”.
1	BITS_8	8-bit resolution

- **SLEEP: Sleep Mode**

Value	Name	Description
0	NORMAL	Normal Mode: The ADC core and reference voltage circuitry are kept ON between conversions.
1	SLEEP	Sleep Mode: The ADC core and reference voltage circuitry are OFF between conversions.

- **PRESCAL: Prescaler Rate Selection**

  $PRESCAL = (f_{peripheral\ clock} / (2 \times f_{ADCCLK})) - 1$.

- **STARTUP: Startup Time**

Value	Name	Description
0	SUT0	0 periods of ADCCLK
1	SUT8	8 periods of ADCCLK
2	SUT16	16 periods of ADCCLK
3	SUT24	24 periods of ADCCLK
4	SUT64	64 periods of ADCCLK
5	SUT80	80 periods of ADCCLK
6	SUT96	96 periods of ADCCLK
7	SUT112	112 periods of ADCCLK
8	SUT512	512 periods of ADCCLK
9	SUT576	576 periods of ADCCLK
10	SUT640	640 periods of ADCCLK
11	SUT704	704 periods of ADCCLK
12	SUT768	768 periods of ADCCLK
13	SUT832	832 periods of ADCCLK
14	SUT896	896 periods of ADCCLK
15	SUT960	960 periods of ADCCLK

- **TRACKTIM: Tracking Time**

  Tracking Time = (TRACKTIM + 1) × ADCCLK periods.

- **USEQ: Use Sequence Enable**

Value	Name	Description
0	NUM_ORDER	Normal Mode: The controller converts channels in a simple numeric order depending only on the channel index.
1	REG_ORDER	User Sequence Mode: The sequence respects what is defined in ADC_SEQR1 register and can be used to convert the same channel several times.

47.7.3 ADC Channel Sequence 1 Register

Name: ADC_SEQR1

Address: 0xFC034008

Access: Read/Write

31	30	29	28	27	26	25	24
–				–			
23	22	21	20	19	18	17	16
–				–			
15	14	13	12	11	10	9	8
USCH4				USCH3			
7	6	5	4	3	2	1	0
USCH2				USCH1			

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

- **USCHx: User Sequence Number x**

The allowed range is 0 up to 4, thus only the sequencer from CH0 to CH4 can be used.

This register activates only if the USEQ field in ADC_MR field is set to ‘1’.

Any USCHx field is processed only if the CHx-1 it in ADC_CHSR reads logical ‘1’, else any value written in USCHx does not add the corresponding channel in the conversion sequence.

Configuring the same value in different fields leads to multiple samples of the same channel during the conversion sequence. This can be done consecutively, or not, according to user needs.

When configuring consecutive fields with the same value, the associated channel is sampled as many time as the number of consecutive values, this part of the conversion sequence being triggered by a unique event.

47.7.4 ADC Channel Enable Register

Name: ADC_CHER

Address: 0xFC034010

Access: Write-only

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
–	–	–	CH4	CH3	CH2	CH1	CH0

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

- **CHx: Channel x Enable**

0: No effect.

1: Enables the corresponding channel.

Note: If USEQ = 1 in ADC_MR, CHx corresponds to the enable of sequence number x+1 described in ADC_SEQR1 (e.g. CH0 enables sequence number USCH1).

47.7.5 ADC Channel Disable Register

Name: ADC_CHDR

Address: 0xFC034014

Access: Write-only

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
–	–	–	CH4	CH3	CH2	CH1	CH0

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

- **CHx: Channel x Disable**

0: No effect.

1: Disables the corresponding channel.

Warning: If the corresponding channel is disabled during a conversion or if it is disabled and then reenabled during a conversion, its associated data and corresponding EOCx and GOVRE flags in ADC_ISR and OVREx flags in ADC_OVER are unpredictable.

47.7.6 ADC Channel Status Register

Name: ADC_CHSR

Address: 0xFC034018

Access: Read-only

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
–	–	–	CH4	CH3	CH2	CH1	CH0

- **CHx: Channel x Status**

0: The corresponding channel is disabled.

1: The corresponding channel is enabled.

47.7.7 ADC Last Converted Data Register

Name: ADC_LCDR

Address: 0xFC034020

Access: Read-only

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

- **LDATA: Last Data Converted**

The analog-to-digital conversion data is placed into this register at the end of a conversion and remains until a new conversion is completed.

- **CHNB: Channel Number**

Indicates the last converted channel when the TAG bit is set in ADC_EMR. If the TAG bit is not set, CHNB = 0.

47.7.8 ADC Interrupt Enable Register

Name: ADC_IER

Address: 0xFC034024

Access: Write-only

31	30	29	28	27	26	25	24
–	NOPEN	PEN	–	–	COMPE	GOVRE	DRDY
23	22	21	20	19	18	17	16
–	PRDY	YRDY	XRDY	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	EOC4	EOC3	EOC2	EOC1	EOC0

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

0: No effect.

1: Enables the corresponding interrupt.

- **EOCx: End of Conversion Interrupt Enable x**
- **XRDY: Touchscreen Measure XPOS Ready Interrupt Enable**
- **YRDY: Touchscreen Measure YPOS Ready Interrupt Enable**
- **PRDY: Touchscreen Measure Pressure Ready Interrupt Enable**
- **DRDY: Data Ready Interrupt Enable**
- **GOVRE: General Overrun Error Interrupt Enable**
- **COMPE: Comparison Event Interrupt Enable**
- **PEN: Pen Contact Interrupt Enable**
- **NOPEN: No Pen Contact Interrupt Enable**

47.7.9 ADC Interrupt Disable Register

Name: ADC_IDR

Address: 0xFC034028

Access: Write-only

31	30	29	28	27	26	25	24
–	NOPEN	PEN	–	–	COMPE	GOVRE	DRDY
23	22	21	20	19	18	17	16
–	PRDY	YRDY	XRDY	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	EOC4	EOC3	EOC2	EOC1	EOC0

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

0: No effect.

1: Disables the corresponding interrupt.

- **EOCx: End of Conversion Interrupt Disable x**
- **XRDY: Touchscreen Measure XPOS Ready Interrupt Disable**
- **YRDY: Touchscreen Measure YPOS Ready Interrupt Disable**
- **PRDY: Touchscreen Measure Pressure Ready Interrupt Disable**
- **DRDY: Data Ready Interrupt Disable**
- **GOVRE: General Overrun Error Interrupt Disable**
- **COMPE: Comparison Event Interrupt Disable**
- **PEN: Pen Contact Interrupt Disable**
- **NOPEN: No Pen Contact Interrupt Disable**

47.7.10 ADC Interrupt Mask Register

Name: ADC_IMR

Address: 0xFC03402C

Access: Read-only

31	30	29	28	27	26	25	24
–	NOPEN	PEN	–	–	COMPE	GOVRE	DRDY
23	22	21	20	19	18	17	16
–	PRDY	YRDY	XRDY	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	EOC4	EOC3	EOC2	EOC1	EOC0

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.

- **EOCx: End of Conversion Interrupt Mask x**
- **XRDY: Touchscreen Measure XPOS Ready Interrupt Mask**
- **YRDY: Touchscreen Measure YPOS Ready Interrupt Mask**
- **PRDY: Touchscreen Measure Pressure Ready Interrupt Mask**
- **DRDY: Data Ready Interrupt Mask**
- **GOVRE: General Overrun Error Interrupt Mask**
- **COMPE: Comparison Event Interrupt Mask**
- **PEN: Pen Contact Interrupt Mask**
- **NOPEN: No Pen Contact Interrupt Mask**

47.7.11 ADC Interrupt Status Register

Name: ADC_ISR

Address: 0xFC034030

Access: Read-only

31	30	29	28	27	26	25	24
PENS	NOPEN	PEN	–	–	COMPE	GOVRE	DRDY
23	22	21	20	19	18	17	16
–	PRDY	YRDY	XRDY	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	EOC4	EOC3	EOC2	EOC1	EOC0

- **EOCx: End of Conversion x (automatically set / cleared)**

0: The corresponding analog channel is disabled, or the conversion is not finished. This flag is cleared when reading the corresponding ADC_CDRx registers.

1: The corresponding analog channel is enabled and conversion is complete.

- **XRDY: Touchscreen XPOS Measure Ready (cleared on read)**

0: No measure has been performed since the last read of ADC_XPOSR.

1: At least one measure has been performed since the last read of ADC_ISR.

- **YRDY: Touchscreen YPOS Measure Ready (cleared on read)**

0: No measure has been performed since the last read of ADC_YPOSR.

1: At least one measure has been performed since the last read of ADC_ISR.

- **PRDY: Touchscreen Pressure Measure Ready (cleared on read)**

0: No measure has been performed since the last read of ADC_PRESSR.

1: At least one measure has been performed since the last read of ADC_ISR.

- **DRDY: Data Ready (automatically set / cleared)**

0: No data has been converted since the last read of ADC_LCDR.

1: At least one data has been converted and is available in ADC_LCDR.

- **GOVRE: General Overrun Error (cleared on read)**

0: No general overrun error occurred since the last read of ADC_ISR.

1: At least one general overrun error has occurred since the last read of ADC_ISR.

- **COMPE: Comparison Event (cleared on read)**

0: No comparison event since the last read of ADC_ISR.

1: At least one comparison event (defined in ADC_EMR and ADC_CWR) has occurred since the last read of ADC_ISR.

- **PEN: Pen contact (cleared on read)**

0: No pen contact since the last read of ADC_ISR.

1: At least one pen contact since the last read of ADC_ISR.

- **NOPEN: No Pen Contact (cleared on read)**

0: No loss of pen contact since the last read of ADC_ISR.

1: At least one loss of pen contact since the last read of ADC_ISR.

- **PENS: Pen Detect Status**

0: The pen does not press the screen.

1: The pen presses the screen.

Note: PENS is not a source of interruption.

47.7.12 ADC Overrun Status Register

Name: ADC_OVER

Address: 0xFC03403C

Access: Read-only

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
–	–	–	OVRE4	OVRE3	OVRE2	OVRE1	OVRE0

- **OVREx: Overrun Error x**

0: No overrun error on the corresponding channel since the last read of ADC_OVER.

1: An overrun error has occurred on the corresponding channel since the last read of ADC_OVER.

Note: An overrun error does not always mean that the unread data has been replaced by a new valid data. Refer to Section 47.6.10 “Enhanced Resolution Mode and Digital Averaging Function” for details.

47.7.13 ADC Extended Mode Register

Name: ADC_EMR

Address: 0xFC034040

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	TAG
23	22	21	20	19	18	17	16
–	–	–	ASTE	–	–	OSR	
15	14	13	12	11	10	9	8
–	–	CMPFILTER		–	–	CMPALL	–
7	6	5	4	3	2	1	0
CMPSEL				–	–	CMPMODE	

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

- **CMPMODE: Comparison Mode**

Value	Name	Description
0	LOW	When the converted data is lower than the low threshold of the window, generates the COMPE flag in ADC_ISR.
1	HIGH	When the converted data is higher than the high threshold of the window, generates the COMPE flag in ADC_ISR.
2	IN	When the converted data is in the comparison window, generates the COMPE flag in ADC_ISR.
3	OUT	When the converted data is out of the comparison window, generates the COMPE flag in ADC_ISR.

- **CMPSEL: Comparison Selected Channel**

If CMPALL = 0: CMPSEL indicates which channel has to be compared.

If CMPALL = 1: No effect.

- **CMPALL: Compare All Channels**

0: Only channel indicated in CMPSEL field is compared.

1: All channels are compared.

- **CMPFILTER: Compare Event Filtering**

Number of consecutive compare events necessary to raise the flag = CMPFILTER+1

When programmed to 0, the flag rises as soon as an event occurs.

Refer to Section 47.6.8 “Comparison Window” when using filtering option (CMPFILTER > 0).

- **OSR: Oversampling Rate**

Value	Name	Description
0	NO_AVERAGE	No averaging. ADC sample rate is maximum.
1	OSR4	1-bit enhanced resolution by averaging. ADC sample rate divided by 4.
2	OSR16	2-bit enhanced resolution by averaging. ADC sample rate divided by 16.

This field is active if LOWRES is cleared in the ADC Mode Register.

- **ASTE: Averaging on Single Trigger Event**

Value	Name	Description
0	MULTI_TRIG_AVERAGE	The average requests several trigger events.
1	SINGLE_TRIG_AVERAGE	The average requests only one trigger event.

- **TAG: Tag of ADC_LCDR**

0: Sets CHNB field to zero in ADC_LCDR.

1: Appends the channel number to the conversion result in ADC_LCDR.

47.7.14 ADC Compare Window Register

Name: ADC_CWR

Address: 0xFC034044

Access: Read/Write

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

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

- **LOWTHRES: Low Threshold**

Low threshold associated to compare settings of ADC_EMR.

If LOWRES is set in ADC_MR, only the 10 LSB of LOWTHRES must be programmed. The two LSB will be automatically discarded to match the value carried on ADC_CDR (8-bit).

- **HIGHTHRES: High Threshold**

High threshold associated to compare settings of ADC_EMR.

If LOWRES is set in ADC_MR, only the 10 LSB of HIGHTHRES must be programmed. The two LSB will be automatically discarded to match the value carried on ADC_CDR (8-bit).

47.7.15 ADC Channel Data Register

Name: ADC_CDRx [x=0..4]

Address: 0xFC034050

Access: Read/Write

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

- **DATA: Converted Data**

The analog-to-digital conversion data is placed into this register at the end of a conversion and remains until a new conversion is completed. ADC_CDRx is only loaded if the corresponding analog channel is enabled.

47.7.16 ADC Analog Control Register

Name: ADC_ACR

Address: 0xFC034094

Access: Read/Write

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
–	–	–	–	–	–	PENDETSENS	

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

- **PENDETSENS: Pen Detection Sensitivity**

Modifies the pen detection input pull-up resistor value. See the section ‘Electrical Characteristics’ for further details.

47.7.17 ADC Touchscreen Mode Register

Name: ADC_TSMR

Address: 0xFC0340B0

Access: Read/Write

31	30	29	28	27	26	25	24
PENDBC				–	–	–	PENDET
23	22	21	20	19	18	17	16
–	NOTSDMA	–	–	TSSCTIM			
15	14	13	12	11	10	9	8
–	–	–	–	TSFREQ			
7	6	5	4	3	2	1	0
–	–	TSAV		–	–	TSMODE	

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

- **TSMODE: Touchscreen Mode**

Value	Name	Description
0	NONE	No Touchscreen
1	4_WIRE_NO_PM	4-wire Touchscreen without pressure measurement
2	4_WIRE	4-wire Touchscreen with pressure measurement
3	5_WIRE	5-wire Touchscreen

When TSMOD equals 01 or 10 (i.e., 4-wire mode), channels 0, 1, 2 and 3 must not be used for classic ADC conversions. When TSMOD equals 11 (i.e., 5-wire mode), channels 0, 1, 2, 3, and 4 must not be used.

- **TSAV: Touchscreen Average**

Value	Name	Description
0	NO_FILTER	No Filtering. Only one ADC conversion per measure
1	AVG2CONV	Averages 2 ADC conversions
2	AVG4CONV	Averages 4 ADC conversions
3	AVG8CONV	Averages 8 ADC conversions

- **TSFREQ: Touchscreen Frequency**

Defines the touchscreen frequency compared to the trigger frequency.

TSFREQ must be greater or equal to TSAV.

The touchscreen frequency is:

$$\text{Touchscreen Frequency} = \text{Trigger Frequency} / 2^{\text{TSFREQ}}$$

- **TSSCTIM: Touchscreen Switches Closure Time**

Defines closure time of analog switches necessary to establish the measurement conditions.

The closure time is:

$$\text{Switch Closure Time} = (\text{TSSCTIM} \times 4) \text{ ADCCLK periods.}$$

- **PENDET: Pen Contact Detection Enable**

0: Pen contact detection disabled.

1: Pen contact detection enabled.

When PENDET = 1, XPOS, YPOS, Z1, Z2 values of ADC_XPOSR, ADC_YPOSR, ADC_PRESSR are automatically cleared when PENS = 0 in ADC_ISR.

- **NOTSDMA: No TouchScreen DMA**

0: XPOS, YPOS, Z1, Z2 are transmitted in ADC_LCDR.

1: XPOS, YPOS, Z1, Z2 are never transmitted in ADC_LCDR, therefore the buffer does not contains touchscreen values.

- **PENDBC: Pen Detect Debouncing Period**

Debouncing period = 2^{PENDBC} ADCCLK periods.

47.7.18 ADC Touchscreen X Position Register

Name: ADC_XPOSR

Address: 0xFC0340B4

Access: Read-only

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

- **XPOS: X Position**

The position measured is stored here. If XPOS = 0 or XPOS = XSIZE, the pen is on the border.

When pen detection is enabled (PENDET set to ‘1’ in ADC_TSMR), XPOS is tied to 0 while there is no detection of contact on the touchscreen (i.e., when PENS bit is cleared in ADC_ISR).

- **XSCALE: Scale of XPOS**

Indicates the max value that XPOS can reach. This value should be close to 2^{10}.

47.7.19 ADC Touchscreen Y Position Register

Name: ADC_YPOSR

Address: 0xFC0340B8

Access: Read-only

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

- **YPOS: Y Position**

The position measured is stored here. If YPOS = 0 or YPOS = YSIZE, the pen is on the border.

When pen detection is enabled (PENDET set to ‘1’ in ADC_TSMR), YPOS is tied to 0 while there is no detection of contact on the touchscreen (i.e., when PENS bit is cleared in ADC_ISR).

- **YSCALE: Scale of YPOS**

Indicates the max value that YPOS can reach. This value should be close to 2^{10}.

47.7.20 ADC Touchscreen Pressure Register

Name: ADC_PRESSR

Address: 0xFC0340BC

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	Z2			
23	22	21	20	19	18	17	16
Z2							
15	14	13	12	11	10	9	8
–	–	–	–	Z1			
7	6	5	4	3	2	1	0
Z1							

- **Z1: Data of Z1 Measurement**

Data Z1 necessary to calculate pen pressure.

When pen detection is enabled (PENDET set to ‘1’ in ADC_TSMR), Z1 is tied to 0 while there is no detection of contact on the touchscreen (i.e., when PENS bit is cleared in ADC_ISR).

- **Z2: Data of Z2 Measurement**

Data Z2 necessary to calculate pen pressure.

When pen detection is enabled (PENDET set to ‘1’ in ADC_TSMR), Z2 is tied to 0 while there is no detection of contact on the touchscreen (i.e., when PENS bit is cleared in ADC_ISR).

Note: These two values are unavailable if TSMODE is not set to 2 in ADC_TSMR.

47.7.21 ADC Trigger Register

Name: ADC_TRGR

Address: 0xFC0340C0

Access: Read/Write

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

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

- **TRGMOD: Trigger Mode**

Value	Name	Description
0	NO_TRIGGER	No trigger, only software trigger can start conversions
1	EXT_TRIG_RISE	External trigger rising edge
2	EXT_TRIG_FALL	External trigger falling edge
3	EXT_TRIG_ANY	External trigger any edge
4	PEN_TRIG	Pen Detect Trigger (shall be selected only if PENDET is set and TSAMOD = Touchscreen only mode)
5	PERIOD_TRIG	ADC internal periodic trigger (see field TRGPER)
6	CONTINUOUS	Continuous Mode

- **TRGPER: Trigger Period**

Effective only if TRGMOD defines a periodic trigger.

Defines the periodic trigger period, with the following equation:

Trigger Period = (TRGPER + 1) / ADCCLK

The minimum time between two consecutive trigger events must be strictly greater than the duration time of the longest conversion sequence depending on the configuration of registers ADC_MR, ADC_CHSR, ADC_SEQRx, ADC_TSMR.

When TRGMOD is set to pen detect trigger (i.e., 100) and averaging is used (i.e., field TSAV ≠ 0 in ADC_TSMR) only one measure is performed. Thus, XRDY, YRDY, PRDY, DRDY will not rise on pen contact trigger. To achieve measurement, several triggers must be provided either by software or by setting the TRGMOD on continuous trigger (i.e., 110) until flags rise.

47.7.22 ADC Write Protection Mode Register

Name: ADC_WPMR

Address: 0xFC0340E4

Access: Read/Write

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

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY value corresponds to 0x414443 (“ADC” in ASCII).

1: Enables the write protection if WPKEY value corresponds to 0x414443 (“ADC” in ASCII).

See Section 47.6.14 “Register Write Protection” for the list of write-protected registers.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x414443	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0

47.7.23 ADC Write Protection Status Register

Name: ADC_WPSR

Address: 0xFC0340E8

Access: Read-only

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

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of ADC_WPSR.

1: A write protection violation has occurred since the last read of ADC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

48. True Random Number Generator (TRNG)

48.1 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*.

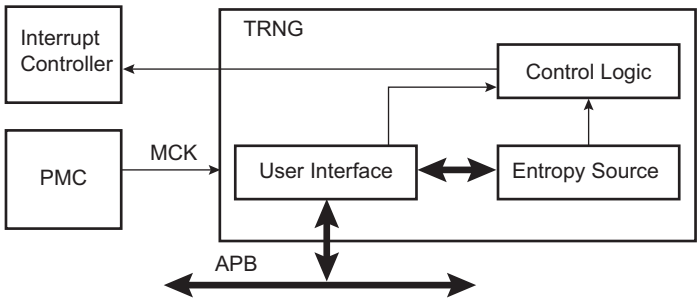
The TRNG may be used as an entropy source for seeding an NIST approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3.

48.2 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

48.3 Block Diagram

Figure 48-1. TRNG Block Diagram



48.4 Product Dependencies

48.4.1 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.

48.4.2 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.

Table 48-1. Peripheral IDs

Instance	ID
TRNG	53

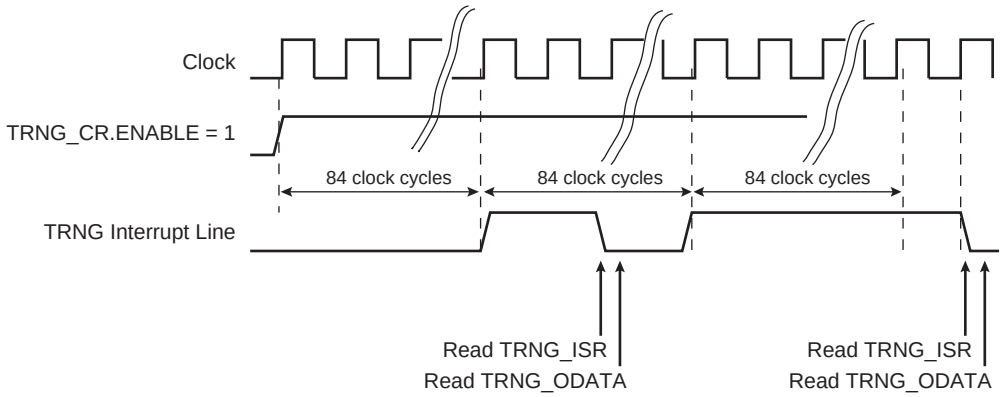
48.5 Functional Description

As soon as the TRNG is enabled in the Control register (TRNG_CR), the generator provides one 32-bit 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 48-2. TRNG Data Generation Sequence



48.6 True Random Number Generator (TRNG) User Interface

Table 48-2. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	TRNG_CR	Write-only	–
0x04–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	TRNG_IER	Write-only	–
0x14	Interrupt Disable Register	TRNG_IDR	Write-only	–
0x18	Interrupt Mask Register	TRNG_IMR	Read-only	0x0000_0000
0x1C	Interrupt Status Register	TRNG_ISR	Read-only	0x0000_0000
0x20–0x4C	Reserved	–	–	–
0x50	Output Data Register	TRNG_ODATA	Read-only	0x0000_0000
0x54–0xE0	Reserved	–	–	–
0xE4	Reserved	–	–	–
0xE8–0xFC	Reserved	–	–	–

48.6.1 TRNG Control Register

Name: TRNG_CR

Address: 0xFC030000

Access: Write-only

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

- **ENABLE: Enables the TRNG to Provide Random Values**

0: Disables the TRNG.

1: Enables the TRNG if 0x524E47 (“RNG” in ASCII) is written in KEY field at the same time.

- **KEY: Security Key**

Value	Name	Description
0x524E47	PASSWD	Writing any other value in this field aborts the write operation.

48.6.2 TRNG Interrupt Enable Register

Name: TRNG_IER

Address: 0xFC030010

Access: Write-only

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
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

48.6.3 TRNG Interrupt Disable Register

Name: TRNG_IDR

Address: 0xFC030014

Access: Write-only

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
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

48.6.4 TRNG Interrupt Mask Register

Name: TRNG_IMR

Address: 0xFC030018

Access: Read-only

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
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready Interrupt Mask**

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

48.6.5 TRNG Interrupt Status Register

Name: TRNG_ISR

Address: 0xFC03001C

Access: Read-only

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
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready**

0: Output data is not valid or TRNG is disabled.

1: New random value is completed.

DATRDY is cleared when this register is read.

48.6.6 TRNG Output Data Register

Name: TRNG_ODATA

Address: 0xFC030050

Access: Read-only

31	30	29	28	27	26	25	24
ODATA							

23	22	21	20	19	18	17	16
ODATA							

15	14	13	12	11	10	9	8
ODATA							

7	6	5	4	3	2	1	0
ODATA							

- **ODATA: Output Data**

The 32-bit Output Data register contains the 32-bit random data.

49. Advanced Encryption Standard (AES)

49.1 Description

The Advanced Encryption Standard (AES) is compliant with the American *FIPS (Federal Information Processing Standard) Publication 197* specification.

The AES supports all five confidentiality modes of operation for symmetrical key block cipher algorithms (ECB, CBC, OFB, CFB and CTR), as specified in the *NIST Special Publication 800-38A Recommendation,* as well as Galois/Counter Mode (GCM) as specified in the *NIST Special Publication 800-38D Recommendation*. It is compatible with all these modes via DMA Controller channels, minimizing processor intervention for large buffer transfers.

The 128-bit/192-bit/256-bit key is stored in four/six/eight 32-bit write-only AES Key Word registers (AES_KEYWR0–7).

The 128-bit input data and initialization vector (for some modes) are each stored in four 32-bit write-only AES Input Data registers (AES_IDATAR0–3) and AES Initialization Vector registers (AES_IVR0–3).

As soon as the initialization vector, the input data and the key are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data are ready to be read out on the four 32-bit AES Output Data registers (AES_ODATAR0–3) or through the DMA channels.

49.2 Embedded Characteristics

- Compliant with *FIPS Publication 197, Advanced Encryption Standard (AES)*
- 128-bit/192-bit/256-bit Cryptographic Key
- 10/12/14 Clock Cycles Encryption/Decryption Inherent Processing Time with a 128-bit/192-bit/256-bit Cryptographic Key
- Double Input Buffer Optimizes Runtime
- Support of the Modes of Operation Specified in the *NIST Special Publication 800-38A* and *NIST Special Publication 800-38D*:
  - Electronic Codebook (ECB)
  - Cipher Block Chaining (CBC) including CBC-MAC
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
  - Counter (CTR)
  - Galois/Counter Mode (GCM)
- 8, 16, 32, 64 and 128-bit Data Sizes Possible in CFB Mode
- Last Output Data Mode Allows Optimized Message Authentication Code (MAC) Generation
- Connection to DMA Optimizes Data Transfers for all Operating Modes

49.3 Product Dependencies

49.3.1 Power Management

The AES may be clocked through the Power Management Controller (PMC), so the programmer must first to configure the PMC to enable the AES clock.

49.3.2 Interrupt Sources

The AES interface has an interrupt line connected to the Interrupt Controller.

Handling the AES interrupt requires programming the Interrupt Controller before configuring the AES.

Table 49-1. Peripheral IDs

Instance	ID
AES	12

49.4 Functional Description

The Advanced Encryption Standard (AES) specifies a FIPS-approved cryptographic algorithm that can be used to protect electronic data. The AES algorithm is a symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information.

Encryption converts data to an unintelligible form called ciphertext. Decrypting the ciphertext converts the data back into its original form, called plaintext. The CIPHER bit in the AES Mode register (AES_MR) allows selection between the encryption and the decryption processes.

The AES is capable of using cryptographic keys of 128/192/256 bits to encrypt and decrypt data in blocks of 128 bits. This 128-bit/192-bit/256-bit key is defined in AES_KEYWRx.

The input to the encryption processes of the CBC, CFB, and OFB modes includes, in addition to the plaintext, a 128-bit data block called the initialization vector (IV), which must be set in AES_IVRx. The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message. AES_IVRx are also used by the CTR mode to set the counter value.

49.4.1 AES Register Endianness

In ARM processor-based products, the system bus and processors manipulate data in little-endian form. The AES interface requires little-endian format words. However, in accordance with the protocol of the FIPS 197 specification, data is collected, processed and stored by the AES algorithm in big-endian form.

The following example illustrates how to configure the AES:

If the first 64 bits of a message (according to FIPS 197, i.e., big-endian format) to be processed is 0xcafedeca_01234567, then AES_IDATAR0 and AES_IDATAR1 registers must be written with the following pattern:

- AES_IDATAR0 = 0xcadefeca
- AES_IDATAR1 = 0x67452301

49.4.2 Operating Modes

The AES supports the following modes of operation:

- ECB: Electronic Codebook
- CBC: Cipher Block Chaining
- OFB: Output Feedback
- CFB: Cipher Feedback
  - CFB8 (CFB where the length of the data segment is 8 bits)
  - CFB16 (CFB where the length of the data segment is 16 bits)
  - CFB32 (CFB where the length of the data segment is 32 bits)
  - CFB64 (CFB where the length of the data segment is 64 bits)
  - CFB128 (CFB where the length of the data segment is 128 bits)

- CTR: Counter
- GCM: Galois/Counter Mode

The data preprocessing, data postprocessing and data chaining for the concerned modes are performed automatically. Refer to the *NIST Special Publication 800-38A* and *NIST Special Publication 800-38D* for more complete information.

Mode selection is done by configuring the OPMOD field in AES_MR.

In CFB mode, five data sizes are possible (8, 16, 32, 64 or 128 bits), configurable by means of the CFBS field in AES_MR (Section 49.5.2 “AES Mode Register”).

In CTR mode, the size of the block counter embedded in the module is 16 bits. Therefore, there is a rollover after processing 1 Mbyte of data. If the file to be processed is greater than 1 Mbyte, this file must be split into fragments of 1 Mbyte or less for the first fragment if the initial value of the counter is greater than 0. Prior to loading the first fragment into AES_IDATARx, AES_IVRx must be fully programmed with the initial counter value. For any fragment, after the transfer is completed and prior to transferring the next fragment, AES_IVRx must be programmed with the appropriate counter value.

49.4.3 Double Input Buffer

AES_IDATARx can be double-buffered to reduce the runtime of large files.

This mode allows a new message block to be written when the previous message block is being processed. This is only possible when DMA accesses are performed (SMOD = 2).

The DUALBUFF bit in AES_MR must be set to ‘1’ to access the double buffer.

49.4.4 Start Modes

The SMOD field in AES_MR allows selection of the encryption (or decryption) Start mode.

49.4.4.1 Manual Mode

The sequence of actions is as follows:

1. Write AES_MR with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 128-bit/192-bit/256-bit key in AES_KEYWRx.
3. Write the initialization vector (or counter) in AES_IVRx.

Note: AES_IVRx concerns all modes except ECB.

4. Set the bit DATRDY (Data Ready) in the AES Interrupt Enable register (AES_IER), depending on whether an interrupt is required or not at the end of processing.
5. Write the data to be encrypted/decrypted in the authorized AES_IDATARx (refer to Table 49-2).
6. Set the START bit in the AES Control register (AES_CR) to begin the encryption or the decryption process.
7. When processing completes, the DATRDY flag in the AES Interrupt Status register (AES_ISR) is raised. If an interrupt has been enabled by setting the DATRDY bit in AES_IER, the interrupt line of the AES is activated.
8. When software reads one of AES_ODATARx, the DATRDY bit is automatically cleared.

Table 49-2. Authorized Input Data Registers

Operating Mode	Input Data Registers to Write
ECB	All
CBC	All
OFB	All
128-bit CFB	All
64-bit CFB	AES_IDATAR0 and AES_IDATAR1

Table 49-2. Authorized Input Data Registers (Continued)

Operating Mode	Input Data Registers to Write
32-bit CFB	AES_IDATAR0
16-bit CFB	AES_IDATAR0
8-bit CFB	AES_IDATAR0
CTR	All
GCM	All

Notes: 1. In 64-bit CFB mode, writing to AES_IDATAR2 and AES_IDATAR3 is not allowed and may lead to errors in processing.
2. In 32, 16, and 8-bit CFB modes, writing to AES_IDATAR1, AES_IDATAR2 and AES_IDATAR3 is not allowed and may lead to errors in processing.

49.4.4.2 Auto Mode

The Auto Mode is similar to the manual one, except that in this mode, as soon as the correct number of AES_IDATARx is written, processing is automatically started without any action in AES_CR.

49.4.4.3 DMA Mode

The DMA Controller can be used in association with the AES to perform an encryption/decryption of a buffer without any action by software during processing.

The SMOD field in AES_MR must be configured to 2 and the DMA must be configured with non-incremental addresses.

The start address of any transfer descriptor must be configured with the address of AES_IDATAR0.

The DMA chunk size configuration depends on the AES mode of operation and is listed in Table 49-3.

When writing data to AES with a first DMA channel, data are first fetched from a memory buffer (source data). It is recommended to configure the size of source data to “words” even for CFB modes. On the contrary, the destination data size depends on the mode of operation. When reading data from the AES with the second DMA channel, the source data is the data read from AES and data destination is the memory buffer. In this case, the source data size depends on the AES mode of operation and is listed in Table 49-3.

Table 49-3. DMA Data Transfer Type for the Different Operating Modes

Operating Mode	Chunk Size	Destination/Source Data Transfer Type
ECB	4	Word
CBC	4	Word
OFB	4	Word
CFB 128-bit	4	Word
CFB 64-bit	1	Word
CFB 32-bit	1	Word
CFB 16-bit	1	Half-word
CFB 8-bit	1	Byte
CTR	4	Word
GCM	4	Word

49.4.5 Last Output Data Mode

This mode is used to generate cryptographic checksums on data (MAC) by means of cipher block chaining encryption algorithm (CBC-MAC algorithm for example).

After each end of encryption/decryption, the output data are available either on AES_ODATARx for Manual and Auto mode, or at the address specified in the receive buffer pointer for DMA mode (refer to Table 49-4).

The Last Output Data (LOD) bit in AES_MR allows retrieval of only the last data of several encryption/decryption processes.

Therefore, there is no need to define a read buffer in DMA mode.

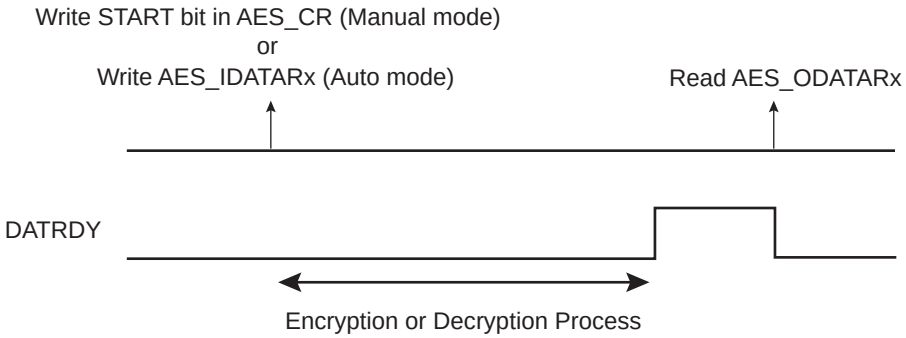
This data are only available in AES_ODATARx.

49.4.5.1 Manual and Auto Modes

If AES_MR.LOD = 0

The DATRDY flag is cleared when at least one AES_ODATARx is read (refer to Figure 49-1).

Figure 49-1. Manual and Auto Modes with AES_MR.LOD = 0



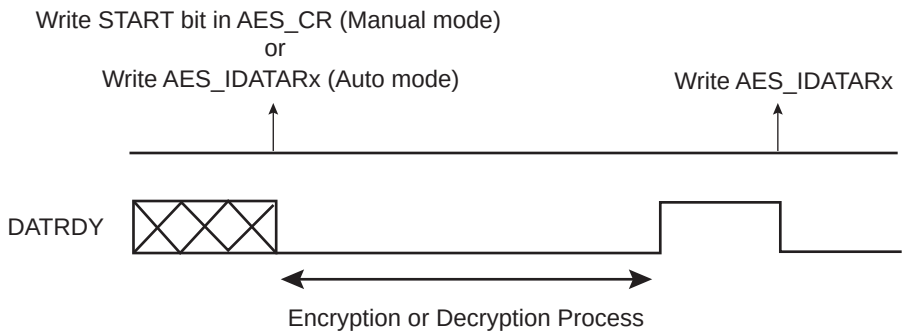
If the user does not want to read AES_ODATARx between each encryption/decryption, the DATRDY flag will not be cleared. If the DATRDY flag is not cleared, the user cannot know the end of the following encryptions/decryptions.

If AES_MR.LOD = 1

This mode is optimized to process AES CPC-MAC operating mode.

The DATRDY flag is cleared when at least one AES_IDATAR is written (refer to Figure 49-2). No additional AES_ODATAR reads are necessary between consecutive encryptions/decryptions.

Figure 49-2. Manual and Auto Modes with AES_MR.LOD = 1



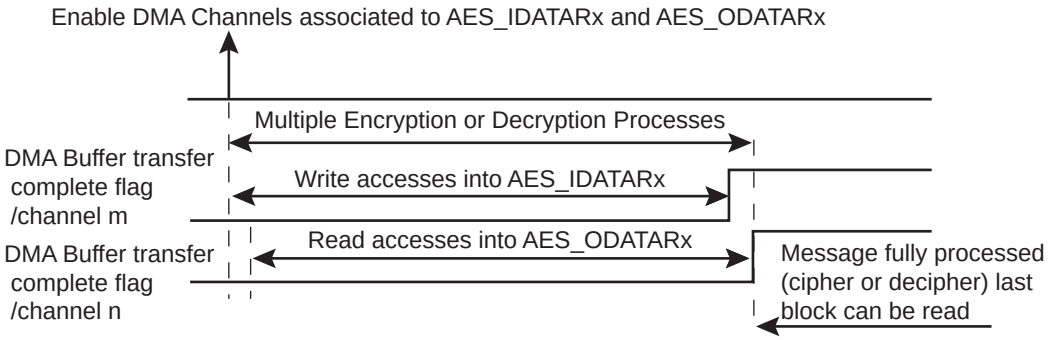
49.4.5.2 DMA Mode

If AES_MR.LOD = 0

This mode may be used for all AES operating modes except CBC-MAC where AES_MR.LOD = 1 mode is recommended.

The end of the encryption/decryption is indicated by the end of DMA transfer associated to AES_ODATARx (refer to Figure 49-3). Two DMA channels are required: one for writing message blocks to AES_IDATARx and one to obtain the result from AES_ODATARx.

Figure 49-3. DMA Transfer with AES_MR.LOD = 0



If AES_MR.LOD = 1

This mode is optimized to process AES CBC-MAC operating mode.

The user must first wait for the DMA buffer transfer complete flag, then for the flag DATRDY to rise to ensure that the encryption/decryption is completed (refer to Figure 49-4).

In this case, no receive buffers are required.

The output data are only available on AES_ODATARx.

Figure 49-4. DMA Transfer with AES_MR.LOD = 1

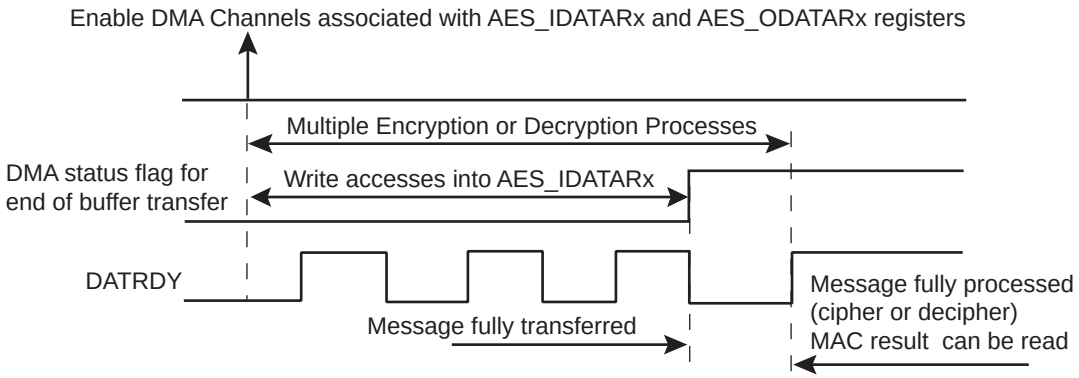


Table 49-4 summarizes the different cases.

Table 49-4. Last Output Data Mode Behavior versus Start Modes

Sequence	Manual and Auto Modes		DMA Transfer	
	AES_MR.LOD = 0	AES_MR.LOD = 1	AES_MR.LOD = 0	AES_MR.LOD = 1
DATRDY Flag Clearing Condition(1)	At least one AES_ODATAR must be read	At least one AES_IDATAR must be written	Not used	Managed by the DMA
End of Encryption/Decryption Notification	DATRDY	DATRDY	2 DMA Buffer transfer complete flags (channel m and channel n)	DMA buffer transfer complete flag, then AES DATRDY flag
Encrypted/Decrypted Data Result Location	In AES_ODATARx	In AES_ODATARx	At the address specified in the Channel Buffer Transfer Descriptor	In AES_ODATARx

Note: 1. Depending on the mode, there are other ways of clearing the DATRDY flag. Refer to Section 49.5.6 “AES Interrupt Status Register”.

Warning: In DMA mode, reading AES_ODATARx before the last data transfer may lead to unpredictable results.

49.4.6 Galois/Counter Mode (GCM)

49.4.6.1 Description

GCM comprises the AES engine in CTR mode along with a universal hash function (GHASH engine) that is defined over a binary Galois field to produce a message authentication tag (the AES CTR engine and the GHASH engine are depicted in Figure 49-5).

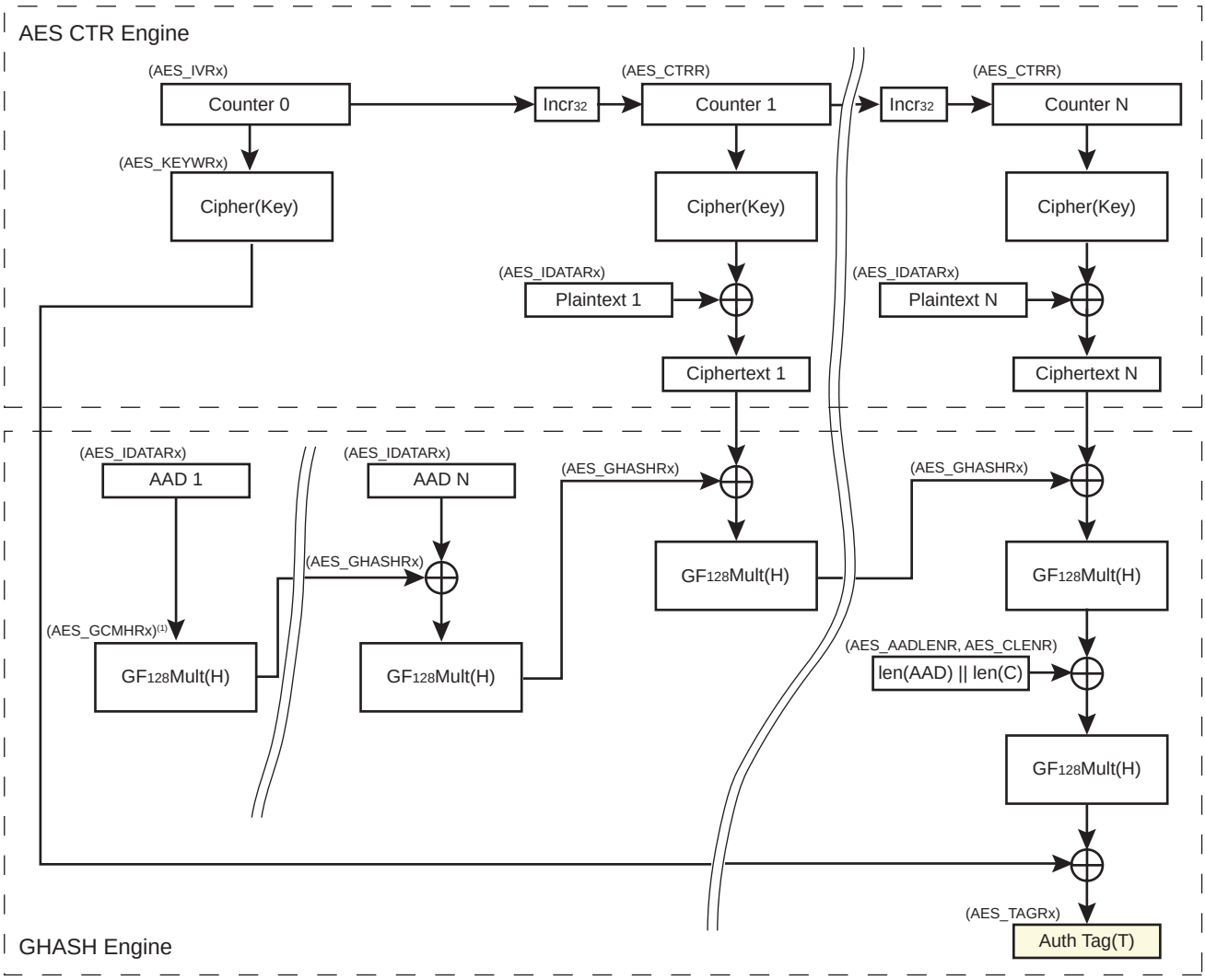
The GHASH engine processes data packets after the AES operation. GCM assures the confidentiality of data through the AES Counter mode of operation for encryption. Authenticity of the confidential data is assured through the GHASH engine. GCM can also provide assurance of data that is not encrypted. Refer to the *NIST Special Publication 800-38D* for more complete information.

GCM can be used with or without the DMA master. Messages may be processed as a single complete packet of data or they may be broken into multiple packets of data over time.

GCM processing is computed on 128-bit input data fields. There is no support for unaligned data. The AES key length can be whatever length is supported by the AES module.

The recommended programming procedure when using DMA is described in Section 49.4.6.3.

Figure 49-5. GCM Block Diagram



Note: 1. Optional

49.4.6.2 Key Writing and Automatic Hash Subkey Calculation

Whenever a new key (AES_KEYWRx) is written to the hardware, two automatic actions are processed:

- GCM Hash Subkey *H* generation—The GCM hash subkey (*H*) is automatically generated. The GCM hash subkey generation must be complete before doing any other action. The DATRDY bit of AES_ISR indicates when the subkey generation is complete (with interrupt if configured). The GCM hash subkey calculation is processed with the formula *H* = CIPHER(Key, <128 bits to zero>). The generated GCM *H* value is then available in AES_GCMHRx. If the application software requires a specific hash subkey, the automatically generated H value can be overwritten in AES_GCMHRx.
  AES_GCMHRx can be written after the end of the hash subkey generation (refer to AES_ISR.DATRDY) and prior to starting the input data feed.
- AES_GHASHRx Clear—AES_GHASHRx are automatically cleared. If a hash initial value is needed for the GHASH, it must be written to AES_GHASHRx
  - after a write to AES_KEYWRx, if any
  - before starting the input data feed

49.4.6.3 GCM Processing

GCM processing is made up of three phases:

1. Processing the Additional Authenticated Data (*AAD*), hash computation only.
2. Processing the Ciphertext (*C*), hash computation + ciphering/deciphering.
3. Generating the Tag using length of *AAD*, length of *C* and J_0 (refer to NIST documentation for details).

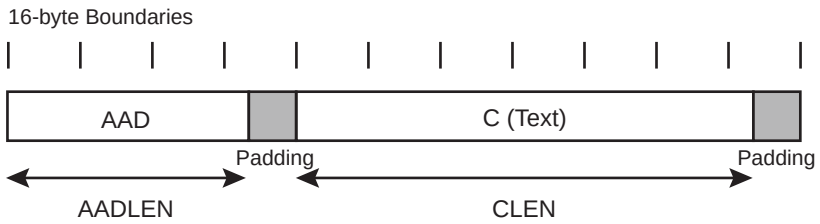
The Tag generation can be done either automatically, after the end of *AAD*/*C* processing if GTAGEN is set in AES_MR, or done manually using the GHASH field in AES_GHASHRx (Refer to subsections Processing a Complete Message with Tag Generation and Manual GCM Tag Generation for details).

Processing a Complete Message with Tag Generation

Use this procedure only if J_0 four LSB bytes ≠ 0xFFFFFFFF.

NOTE: If J_0 four LSB bytes = 0xFFFFFFFF or if the value is unknown, use the procedure described in Processing a Complete Message without Tag Generation followed by the procedure in Manual GCM Tag Generation.

Figure 49-6. Full Message Alignment



To process a complete message with Tag generation, the sequence is as follows:

1. In AES_MR, set OPMOD to GCM and GTAGEN to '1'.
2. Set KEYW in AES_KEYWRx and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. Refer to Section 49.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".
3. Calculate the J_0 value as described in NIST documentation $J_0 = IV \,||\, 0^{31} \,||\, 1$ when len(IV) = 96 and $J_0 = \text{GHASH}_H(IV \,||\, 0^{s+64} \,||\, [\text{len}(IV)]_{64})$ if len(IV) ≠ 96. Refer to Processing a Message with only AAD (GHASHH) for J_0 generation.
4. Set IV in AES_IVRx with inc32(J_0) (J_0 + 1 on 32 bits).
5. Set AADLEN field in AES_AADLENR and CLEN field in AES_CLENR.
6. Fill the IDATA field of AES_IDATARx with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Wait for TAGRDY to be set (use interrupt if needed), then read the TAG field of AES_TAGRx to obtain the authentication tag of the message.

Processing a Complete Message without Tag Generation

Processing a message without generating the Tag can be used to customize the Tag generation, or to process a fragmented message. To manually generate the GCM Tag, refer to Manual GCM Tag Generation.

To process a complete message without Tag generation, the sequence is as follows:

1. In AES_MR, set OPMOD to GCM and GTAGEN to '0'.
2. Set KEYW in AES_KEYWRx and wait until DATRDY bit of AES_ISR is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. Refer to Section 49.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".

3. Calculate the J_0 value as described in NIST documentation $J_0 = IV \,||\, 0^{31} \,||\, 1$ when len(IV) = 96 and $J_0 = \text{GHASH}_H(IV \,||\, 0^{s+64} \,||\, [\text{len}(IV)]_{64})$ if len(IV) ≠ 96. Refer to Processing a Message with only AAD (GHASHH) for J_0 generation example when len(IV) ≠ 96.
4. Set IV in AES_IVRx with inc32(J_0) (J_0 + 1 on 32 bits).
5. Set AADLEN field in AES_AADLENR and CLEN field in AES_CLENR.
6. Fill the IDATA field of AES_IDATARx with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if CLEN ≠ 0 (or wait for DATRDY), then read the GHASH field of AES_GHASHRx to obtain the hash value after the last processed data.

Processing a Fragmented Message without Tag Generation

If needed, a message can be processed by fragments, in such case automatic GCM Tag generation is not supported.

To process a message by fragments, the sequence is as follows:

- First fragment:

1. In AES_MR set OPMOD to GCM and GTAGEN to '0'.
2. Set KEYW in AES_KEYWRx and wait for DATRDY bit of AES_ISR to be set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. Refer to Section 49.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".
3. Calculate the J_0 value as described in NIST documentation $J_0 = IV \,||\, 0^{31} \,||\, 1$ when len(IV) = 96 and $J_0 = \text{GHASH}_H(IV \,||\, 0^{s+64} \,||\, [\text{len}(IV)]_{64})$ if len(IV) ≠ 96. Refer to Processing a Message with only AAD (GHASHH) for J_0 generation example when len(IV) ≠ 96.
4. Set IV in AES_IVRx with inc32(J_0) (J_0 + 1 on 32 bits).
5. Set AADLEN field in AES_AADLENR and CLEN field in AES_CLENR according to the length of the first fragment, or set the fields with the full message length, both configurations work.
6. Fill the IDATA field of AES_IDATARx with the first fragment of the message to process (aligned on 16-byte boundary) according to the SMOD configuration used. If Manual Mode or Auto Mode is used the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if the fragment ends in C phase (or wait for DATRDY if the fragment ends in *AAD* phase), then read the GHASH field of AES_GHASHRx to obtain the value of the hash after the last processed data and finally read the CTR field of AES_CTR to obtain the value of the CTR encryption counter (not needed when the fragment ends in *AAD* phase).

- Next fragment (or last fragment):

1. In AES_MR set OPMOD to GCM and GTAGEN to '0'.
2. Set KEYW in AES_KEYWRx and wait until DATRDY bit of AES_ISR is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. Refer to Section 49.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".
3. Set IV in AES_IVRx with:
   - If the first block of the fragment is a block of Additional Authenticated data, set IV in AES_IVRx with the J0 initial value
   - If the first block of the fragment is a block of Plaintext data, set IV in AES_IVRx with a value constructed as follows: 'LSB96(J0) || CTR' value, (96 bit LSB of J0 concatenated with saved CTR value from previous fragment).

4. Set AADLEN field in AES_AADLENR and CLEN field in AES_CLENR according to the length of the current fragment, or set the fields with the remaining message length, both configurations work.
5. Fill the GHASH field of AES_GHASHRx with the value stored after the previous fragment.
6. Fill the IDATA field of AES_IDATARx with the current fragment of the message to process (aligned on 16 byte boundary) according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if the fragment ends in C phase (or wait for DATRDY if the fragment ends in *AAD* phase), then read the GHASH field of AES_GHASHRx to obtain the value of the hash after the last processed data and finally read the CTR field of AES_CTR to obtain the value of the CTR encryption counter (not needed when the fragment ends in *AAD* phase).

Note: Step 1 and 2 are required only if the value of the concerned registers has been modified.

Once the last fragment has been processed, the GHASH value will allow manual generation of the GCM tag. Refer to Manual GCM Tag Generation.

Manual GCM Tag Generation

This section describes the last steps of the GCM Tag generation.

The Manual GCM Tag Generation is used to complete the GCM Tag Generation when the message has been processed without Tag Generation.

Note: The Message Processing without Tag Generation must be finished before processing the Manual GCM Tag Generation.

To generate a GCM Tag manually, the sequence is as follows:

Processing $S = \text{GHASH}_H (A \| 0^v \| C \| 0^u \| [\text{len}(A)]_{64} \| [\text{len}(C)]_{64})$:

1. In AES_MR set OPMOD to GCM and GTAGEN to ‘0’.
2. Set KEYW in AES_KEYWRx and wait for DATRDY bit of AES_ISR to be set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. Refer to Section 49.4.6.2 “Key Writing and Automatic Hash Subkey Calculation”.
3. Set AADLEN field to 0x10 (16 bytes) in AES_AADLENR and CLEN field to ‘0’ in AES_CLENR. This will allow running a single GHASH_H on a 16-byte input data (refer to Figure 49-7).
4. Fill the GHASH field of AES_GHASHRx with the state of the GHASH field stored at the end of the message processing.
5. Fill the IDATA field of AES_IDATARx according to the SMOD configuration used with ‘$\text{len}(A)_{64} \| \text{len}(C)_{64}$’ value as described in the NIST documentation and wait for DATRDY to be set; use interrupt if needed.
6. Read the GHASH field of AES_GHASHRx to obtain the current value of the hash.

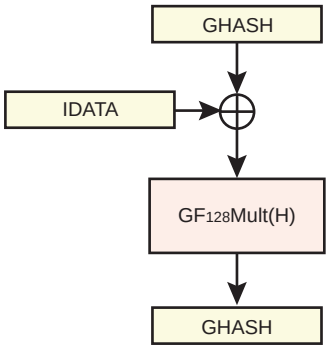
Processing $T = \text{GCTR}_K(J_0, S)$:

7. In AES_MR, set OPMOD to CTR.
8. Set the IV field in AES_IVRx with ‘J_0’ value.
9. Fill the IDATA field of AES_IDATARx with the GHASH value read at step 6 and wait for DATRDY to be set (use interrupt if needed).
10. Read the ODATA field of AES_ODATARx to obtain the GCM Tag value.

Note: Step 4 is optional if the GHASH field is to be filled with value ‘0’ (0 length packet for instance).

Processing a Message with only AAD (GHASHH)

Figure 49-7. Single GHASHH Block Diagram (AADLEN ≤ 0x10 and CLEN = 0)



It is possible to process a message with only *AAD* setting the CLEN field to '0' in AES_CLENR, this can be used for J_0 generation when len(*IV*) ≠ 96 for instance.

Example: Processing J_0 when len(*IV*) ≠ 96

To process $J_0 = GHASH_H(IV \,||\, 0^{s+64} \,||\, [len(IV)]_{64})$, the sequence is as follows:

1. In AES_MR, set OPMOD to GCM and GTAGEN to '0'.
2. Set KEYW in AES_KEYWRx and wait until DATRDY bit of AES_ISR is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. Refer to Section 49.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".
3. Set AADLEN field with '$len(IV \,||\, 0^{s+64} \,||\, [len(IV)]_{64})$' in AES_AADLENR and CLEN field to '0' in AES_CLENR. This will allow running a GHASHH only.
4. Fill the IDATA field of AES_IDATARx with the message to process ($IV \,||\, 0^{s+64} \,||\, [len(IV)]_{64}$) according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when a GHASHH step is over (use interrupt if needed).
5. Read the GHASH field of AES_GHASHRx to obtain the J_0 value.

Note: The GHASH value can be overwritten at any time by writing the GHASH field value of AES_GHASHRx, used to perform a GHASHH with an initial value for GHASH (write GHASH field between step 3 and step 4 in this case).

Processing a Single GF128 Multiplication

The AES can also be used to process a single multiplication in the Galois field on 128 bits (GF_{128}) using a single GHASHH with custom *H* value (refer to Figure 49-7).

To run a GF_{128} multiplication (A x B), the sequence is as follows:

1. In AES_MR, set OPMOD to GCM and GTAGEN to '0'.
2. Set AADLEN field with 0x10 (16 bytes) in AES_AADLENR and CLEN field to '0' in AES_CLENR. This will allow running a single GHASHH.
3. Fill the H field of AES_GCMHRx with B value.
4. Fill the IDATA field of AES_IDATARx with the A value according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when a GHASHH computation is over (use interrupt if needed).
5. Read the GHASH field of AES_GHASHRx to obtain the result.

Note: The GHASH field of AES_GHASHRx can be initialized with a value C between step 3 and step 4 to run a ((A XOR C) x B) GF_{128} multiplication.

49.4.7 Security Features

49.4.7.1 Unspecified Register Access Detection

When an unspecified register access occurs, the URAD flag in AES_ISR is raised. Its source is then reported in the Unspecified Register Access Type (URAT) field. Only the last unspecified register access is available through the URAT field.

Several kinds of unspecified register accesses can occur:

- Input Data register written during the data processing when SMOD = IDATAR0_START
- Output Data register read during data processing
- Mode register written during data processing
- Output Data register read during sub-keys generation
- Mode register written during sub-keys generation
- Write-only register read access

The URAD bit and the URAT field can only be reset by the SWRST bit in AES_CR.

49.5 Advanced Encryption Standard (AES) User Interface

Table 49-5. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	AES_CR	Write-only	–
0x04	Mode Register	AES_MR	Read/Write	0x0
0x08–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	AES_IER	Write-only	–
0x14	Interrupt Disable Register	AES_IDR	Write-only	–
0x18	Interrupt Mask Register	AES_IMR	Read-only	0x0
0x1C	Interrupt Status Register	AES_ISR	Read-only	0x0
0x20	Key Word Register 0	AES_KEYWR0	Write-only	–
0x24	Key Word Register 1	AES_KEYWR1	Write-only	–
0x28	Key Word Register 2	AES_KEYWR2	Write-only	–
0x2C	Key Word Register 3	AES_KEYWR3	Write-only	–
0x30	Key Word Register 4	AES_KEYWR4	Write-only	–
0x34	Key Word Register 5	AES_KEYWR5	Write-only	–
0x38	Key Word Register 6	AES_KEYWR6	Write-only	–
0x3C	Key Word Register 7	AES_KEYWR7	Write-only	–
0x40	Input Data Register 0	AES_IDATAR0	Write-only	–
0x44	Input Data Register 1	AES_IDATAR1	Write-only	–
0x48	Input Data Register 2	AES_IDATAR2	Write-only	–
0x4C	Input Data Register 3	AES_IDATAR3	Write-only	–
0x50	Output Data Register 0	AES_ODATAR0	Read-only	0x0
0x54	Output Data Register 1	AES_ODATAR1	Read-only	0x0
0x58	Output Data Register 2	AES_ODATAR2	Read-only	0x0
0x5C	Output Data Register 3	AES_ODATAR3	Read-only	0x0
0x60	Initialization Vector Register 0	AES_IVR0	Write-only	–
0x64	Initialization Vector Register 1	AES_IVR1	Write-only	–
0x68	Initialization Vector Register 2	AES_IVR2	Write-only	–
0x6C	Initialization Vector Register 3	AES_IVR3	Write-only	–
0x70	Additional Authenticated Data Length Register	AES_AADLENR	Read/Write	–
0x74	Plaintext/Ciphertext Length Register	AES_CLENR	Read/Write	–
0x78	GCM Intermediate Hash Word Register 0	AES_GHASHR0	Read/Write	–
0x7C	GCM Intermediate Hash Word Register 1	AES_GHASHR1	Read/Write	–
0x80	GCM Intermediate Hash Word Register 2	AES_GHASHR2	Read/Write	–
0x84	GCM Intermediate Hash Word Register 3	AES_GHASHR3	Read/Write	–
0x88	GCM Authentication Tag Word Register 0	AES_TAGR0	Read-only	–
0x8C	GCM Authentication Tag Word Register 1	AES_TAGR1	Read-only	–

Table 49-5. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x90	GCM Authentication Tag Word Register 2	AES_TAGR2	Read-only	_
0x94	GCM Authentication Tag Word Register 3	AES_TAGR3	Read-only	_
0x98	GCM Encryption Counter Value Register	AES_CTRR	Read-only	_
0x9C	GCM H Word Register 0	AES_GCMHR0	Read/Write	_
0xA0	GCM H Word Register 1	AES_GCMHR1	Read/Write	_
0xA4	GCM H Word Register 2	AES_GCMHR2	Read/Write	_
0xA8	GCM H Word Register 3	AES_GCMHR3	Read/Write	_
0xAC	Reserved	–	_	_
0xB0	Reserved	–	_	_
0xB4	Reserved	–	_	_
0xC0–0xE0	Reserved	–	_	_
0xE4–0xF8	Reserved	–	_	_
0xFC	Reserved	–	_	_

49.5.1 AES Control Register

Name: AES_CR

Address: 0xFC044000

Access: Write-only

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

- **START: Start Processing**

0: No effect.

1: Starts manual encryption/decryption process.

- **SWRST: Software Reset**

0: No effect.

1: Resets the AES. A software-triggered hardware reset of the AES interface is performed.

49.5.2 AES Mode Register

Name: AES_MR

Address: 0xFC044004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CKEY				–	CFBS		
15	14	13	12	11	10	9	8
LOD	OPMOD			KEYSIZE		SMOD	
7	6	5	4	3	2	1	0
PROCDLY				DUALBUFF	–	GTAGEN	CIPHER

- **CIPHER: Processing Mode**

0: Decrypts data.

1: Encrypts data.

- **GTAGEN: GCM Automatic Tag Generation Enable**

0: Automatic GCM Tag generation disabled.

1: Automatic GCM Tag generation enabled.

- **DUALBUFF: Dual Input Buffer**

Value	Name	Description
0	INACTIVE	AES_IDATARx cannot be written during processing of previous block.
1	ACTIVE	AES_IDATARx can be written during processing of previous block when SMOD = 2. It speeds up the overall runtime of large files.

- **PROCDLY: Processing Delay**

Processing Time = N × (PROCDLY + 1)

where

N = 10 when KEYSIZE = 0

N = 12 when KEYSIZE = 1

N = 14 when KEYSIZE = 2

The processing time represents the number of clock cycles that the AES needs in order to perform one encryption/decryption.

Note: The best performance is achieved with PROCDLY equal to 0.

- **SMOD: Start Mode**

Value	Name	Description
0	MANUAL_START	Manual Mode
1	AUTO_START	Auto Mode
2	IDATAR0_START	AES_IDATAR0 access only Auto Mode (DMA)

If a DMA transfer is used, configure SMOD to 2. Refer to Section 49.4.4.3 “DMA Mode” for more details.

- **KEYSIZE: Key Size**

Value	Name	Description
0	AES128	AES Key Size is 128 bits
1	AES192	AES Key Size is 192 bits
2	AES256	AES Key Size is 256 bits

- **OPMOD: Operating Mode**

Value	Name	Description
0	ECB	ECB: Electronic Codebook mode
1	CBC	CBC: Cipher Block Chaining mode
2	OFB	OFB: Output Feedback mode
3	CFB	CFB: Cipher Feedback mode
4	CTR	CTR: Counter mode (16-bit internal counter)
5	GCM	GCM: Galois/Counter mode

For CBC-MAC operating mode, set OPMOD to CBC and LOD to 1.

- **LOD: Last Output Data Mode**

0: No effect.

After each end of encryption/decryption, the output data are available either on the output data registers (Manual and Auto modes) or at the address specified in the Channel Buffer Transfer Descriptor for DMA mode.

In Manual and Auto modes, the DATRDY flag is cleared when at least one of the Output Data registers is read.

1: The DATRDY flag is cleared when at least one of the Input Data Registers is written.

No more Output Data Register reads is necessary between consecutive encryptions/decryptions (refer to Section 49.4.5 “Last Output Data Mode”).

Warning: In DMA mode, reading to the Output Data registers before the last data encryption/decryption process may lead to unpredictable results.

- **CFBS: Cipher Feedback Data Size**

Value	Name	Description
0	SIZE_128BIT	128-bit
1	SIZE_64BIT	64-bit
2	SIZE_32BIT	32-bit
3	SIZE_16BIT	16-bit
4	SIZE_8BIT	8-bit

- **CKEY: Key**

Value	Name	Description
0xE	PASSWD	This field must be written with 0xE the first time AES_MR is programmed. For subsequent programming of AES_MR, any value can be written, including that of 0xE. Always reads as 0.

49.5.3 AES Interrupt Enable Register

Name: AES_IER

Address: 0xFC044010

Access: Write-only

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

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

0: No effect.

1: Enables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Enable**
- **URAD: Unspecified Register Access Detection Interrupt Enable**
- **TAGRDY: GCM Tag Ready Interrupt Enable**

49.5.4 AES Interrupt Disable Register

Name: AES_IDR

Address: 0xFC044014

Access: Write-only

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

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

0: No effect.

1: Disables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Disable**
- **URAD: Unspecified Register Access Detection Interrupt Disable**
- **TAGRDY: GCM Tag Ready Interrupt Disable**

49.5.5 AES Interrupt Mask Register

Name: AES_IMR

Address: 0xFC044018

Access: Read-only

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

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.

- **DATRDY: Data Ready Interrupt Mask**
- **URAD: Unspecified Register Access Detection Interrupt Mask**
- **TAGRDY: GCM Tag Ready Interrupt Mask**

49.5.6 AES Interrupt Status Register

Name: AES_ISR

Address: 0xFC04401C

Access: Read-only

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

- **DATRDY: Data Ready (cleared by setting bit START or bit SWRST in AES_CR or by reading AES_ODATARx)**

0: Output data not valid.

1: Encryption or decryption process is completed.

Note: If AES_MR.LOD = 1: In Manual and Auto mode, the DATRDY flag can also be cleared by writing at least one AES_IDATARx.

- **URAD: Unspecified Register Access Detection Status (cleared by writing SWRST in AES_CR)**

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.

- **URAT: Unspecified Register Access (cleared by writing SWRST in AES_CR)**

Value	Name	Description
0	IDR_WR_PROCESSING	Input Data register written during the data processing when SMOD = 2 mode.
1	ODR_RD_PROCESSING	Output Data register read during the data processing.
2	MR_WR_PROCESSING	Mode register written during the data processing.
3	ODR_RD_SUBKGEN	Output Data register read during the sub-keys generation.
4	MR_WR_SUBKGEN	Mode register written during the sub-keys generation.
5	WOR_RD_ACCESS	Write-only register read access.

Only the last Unspecified Register Access Type is available through the URAT field.

- **TAGRDY: GCM Tag Ready**

0: GCM Tag is not valid.

1: GCM Tag generation is complete (cleared by reading GCM Tag, starting another processing or when writing a new key).

49.5.7 AES Key Word Register x

Name: AES_KEYWRx [x=0..7]

Address: 0xFC044020

Access: Write-only

31	30	29	28	27	26	25	24
KEYW							

23	22	21	20	19	18	17	16
KEYW							

15	14	13	12	11	10	9	8
KEYW							

7	6	5	4	3	2	1	0
KEYW							

- **KEYW: Key Word**

The four/six/eight 32-bit Key Word registers set the 128-bit/192-bit/256-bit cryptographic key used for AES encryption/decryption.

AES_KEYWR0 corresponds to the first word of the key and respectively AES_KEYWR3/AES_KEYWR5/AES_KEYWR7 to the last one.

Whenever a new key (AES_KEYWRx) is written to the hardware, two automatic actions are processed:

- GCM hash subkey generation
- AES_GHASHRx Clear

Refer to Section 49.4.6.2 “Key Writing and Automatic Hash Subkey Calculation” for details.

These registers are write-only to prevent the key from being read by another application.

49.5.8 AES Input Data Register x

Name: AES_IDATARx [x=0..3]

Address: 0xFC044040

Access: Write-only

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

- **IDATA: Input Data Word**

The four 32-bit Input Data registers set the 128-bit data block used for encryption/decryption.

AES_IDATAR0 corresponds to the first word of the data to be encrypted/decrypted, and AES_IDATAR3 to the last one.

These registers are write-only to prevent the input data from being read by another application.

49.5.9 AES Output Data Register x

Name: AES_ODATARx [x=0..3]

Address: 0xFC044050

Access: Read-only

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

- **ODATA: Output Data**

The four 32-bit Output Data registers contain the 128-bit data block that has been encrypted/decrypted.

AES_ODATAR0 corresponds to the first word, AES_ODATAR3 to the last one.

49.5.10 AES Initialization Vector Register x

Name: AES_IVRx [x=0..3]

Address: 0xFC044060

Access: Write-only

31	30	29	28	27	26	25	24
IV							

23	22	21	20	19	18	17	16
IV							

15	14	13	12	11	10	9	8
IV							

7	6	5	4	3	2	1	0
IV							

- **IV: Initialization Vector**

The four 32-bit Initialization Vector registers set the 128-bit Initialization Vector data block that is used by some modes of operation as an additional initial input.

AES_IVR0 corresponds to the first word of the Initialization Vector, AES_IVR3 to the last one.

These registers are write-only to prevent the Initialization Vector from being read by another application.

For CBC, OFB and CFB modes, the IV input value corresponds to the initialization vector.

For CTR mode, the IV input value corresponds to the initial counter value.

Note: These registers are not used in ECB mode and must not be written.

49.5.11 AES Additional Authenticated Data Length Register

Name: AES_AADLENR

Address: 0xFC044070

Access: Read/Write

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

- **AADLEN: Additional Authenticated Data Length**

Length in bytes of the Additional Authenticated Data (*AAD*) that is to be processed.

Note: The maximum byte length of the *AAD* portion of a message is limited to the 32-bit counter length.

49.5.12 AES Plaintext/Ciphertext Length Register

Name: AES_CLENR

Address: 0xFC044074

Access: Read/Write

31	30	29	28	27	26	25	24
CLEN							

23	22	21	20	19	18	17	16
CLEN							

15	14	13	12	11	10	9	8
CLEN							

7	6	5	4	3	2	1	0
CLEN							

- **CLEN: Plaintext/Ciphertext Length**

Length in bytes of the plaintext/ciphertext (*C*) data that is to be processed.

Note: The maximum byte length of the C portion of a message is limited to the 32-bit counter length.

49.5.13 AES GCM Intermediate Hash Word Register x

Name: AES_GHASHRx [x=0..3]

Address: 0xFC044078

Access: Read/Write

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

- **GHASH: Intermediate GCM Hash Word x**

The four 32-bit Intermediate Hash Word registers expose the intermediate GHASH value. May be read to save the current GHASH value so processing can later be resumed, presumably on a later message fragment. Whenever a new key (AES_KEYWRx) is written to the hardware two automatic actions are processed:

- GCM hash subkey generation
- AES_GHASHRx Clear

Refer to Section 49.4.6.2 “Key Writing and Automatic Hash Subkey Calculation” for details.

If an application software-specific hash initial value is needed for the GHASH, it must be written to AES_GHASHRx:

- after a write to AES_KEYWRx, if any,
- before starting the input data feed.

49.5.14 AES GCM Authentication Tag Word Register x

Name: AES_TAGRx [x=0..3]

Address: 0xFC044088

Access: Read-only

31	30	29	28	27	26	25	24
TAG							

23	22	21	20	19	18	17	16
TAG							

15	14	13	12	11	10	9	8
TAG							

7	6	5	4	3	2	1	0
TAG							

- **TAG: GCM Authentication Tag x**

The four 32-bit Tag registers contain the final 128-bit GCM Authentication tag (*T*) when GCM processing is complete. TAG0 corresponds to the first word, TAG3 to the last word.

49.5.15 AES GCM Encryption Counter Value Register

Name: AES_CTRR

Address: 0xFC044098

Access: Read-only

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

- **CTR: GCM Encryption Counter**

Reports the current value of the 32-bit GCM counter.

49.5.16 AES GCM H Word Register x

Name: AES_GCMHRx [x=0..3]

Address: 0xFC04409C

Access: Read/Write

31	30	29	28	27	26	25	24
H							

23	22	21	20	19	18	17	16
H							

15	14	13	12	11	10	9	8
H							

7	6	5	4	3	2	1	0
H							

- **H: GCM H Word x**

The four 32-bit H Word registers contain the 128-bit GCM hash subkey *H* value.

Whenever a new key (AES_KEYWRx) is written to the hardware, two automatic actions are processed:

- GCM hash subkey *H* generation
- AES_GHASHRx Clear

If the application software requires a specific hash subkey, the automatically-generated H value can be overwritten in AES_GCMHRx. Refer to Section 49.4.6.2 “Key Writing and Automatic Hash Subkey Calculation” for details.

Generating a GCM hash subkey *H* by a write in AES_GCMHRx enables to:

- select the GCM hash subkey *H* for GHASH operations,
- select one operand to process a single GF128 multiply.

50. Triple Data Encryption Standard (TDES)

50.1 Description

The Triple Data Encryption Standard (TDES) is compliant with the American *FIPS (Federal Information Processing Standard) Publication 46-3* specification.

The TDES supports the four different confidentiality modes of operation (ECB, CBC, OFB and CFB), specified in the *FIPS (Federal Information Processing Standard) Publication 81* and is compatible with the Peripheral Data Controller channels for all of these modes, minimizing processor intervention for large buffer transfers.

The 64-bit long keys and input data (and initialization vector for some modes) are each stored in two corresponding 32-bit write-only registers:

Key x Word Registers TDES_KEYxWR0 and TDES_KEYxWR1

Input Data Registers TDES_IDATAR0 and TDES_IDATAR1

Initialization Vector Registers TDES_IVR0 and TDES_IVR1

As soon as the initialization vector, the input data and the key are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data is ready to be read out on the two 32-bit Output Data registers (TDES_ODATARx) or through the DMA channels.

50.2 Embedded Characteristics

- Supports Single Data Encryption Standard (DES) and Triple Data Encryption Algorithm (TDEA or TDES)
- Compliant with *FIPS Publication 46-3, Data Encryption Standard (DES)*
- 64-bit Cryptographic Key for TDES
- Two-key or Three-key Algorithms for TDES
- 18-clock Cycles Encryption/Decryption Processing Time for DES
- 50-clock Cycles Encryption/Decryption Processing Time for TDES
- Supports eXtended Tiny Encryption Algorithm (XTEA)
- 128-bit key for XTEA and Programmable Round Number up to 64
- Supports the Four Standard Modes of Operation specified in the *FIPS Publication 81, DES Modes of Operation*
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
- 8-, 16-, 32- and 64-bit Data Sizes Possible in CFB Mode
- Last Output Data Mode Allowing Optimized Message (Data) Authentication Code (MAC) Generation
- Connection to DMA Optimizes Data Transfers for all Operating Modes

50.3 Product Dependencies

50.3.1 Power Management

The TDES may be clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the TDES clock.

50.3.2 Interrupt Sources

The TDES interface has an interrupt line connected to the interrupt controller. Handling the TDES interrupt requires programming the interrupt controller before configuring the TDES.

Table 50-1. Peripheral IDs

Instance	ID
TDES	14

50.4 Functional Description

The Data Encryption Standard (DES) and the Triple Data Encryption Algorithm (TDES) specify FIPS-approved cryptographic algorithms that can be used to protect electronic data. The TDES bit in the TDES Mode Register (TDES_MR) is used to select either the single DES or the Triple DES mode.

Encryption (enciphering) converts data to an unintelligible form called ciphertext. Decrypting (deciphering) the ciphertext converts the data back into its original form, called plaintext. The CIPHER bit in TDES_MR is used to choose between encryption and decryption.

A DES is capable of using cryptographic keys of 64 bits to encrypt and decrypt data in blocks of 64 bits. This 64-bit key is defined in the Key 1 Word Registers (TDES_KEY1WRx).

A TDES key consists of three DES keys, which is also referred to as a key bundle. These three 64-bit keys are defined, respectively, in the Key 1, 2 and 3 Word Registers (TDES_KEY1WRx, TDES_KEY2WRx and TDES_KEY3WRx). In Triple DES mode (TDESMOD = 1 in TDES_MR), the KEYMOD bit in TDES_MR is used to choose between a two- and a three-key algorithm, as summarized in Table 50-2.

Table 50-2. TDES Algorithms Summary

Algorithm	Mode	Data Processing Sequence Steps		
		First	Second	Third
Three-key	Encryption	Encryption with Key 1	Decryption with Key 2	Encryption with Key 3
	Decryption	Decryption with Key 3	Encryption with Key 2	Decryption with Key 1
Two-key	Encryption	Encryption with Key 1	Decryption with Key 2	Encryption with Key 1
	Decryption	Decryption with Key 1	Encryption with Key 2	Decryption with Key 1

The input to the encryption processes of the CBC, CFB, and OFB modes includes, in addition to the plaintext, a 64-bit data block called the initialization vector (IV), which must be set in the Initialization Vector Registers (TDES_IVRx). The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message.

The XTEA algorithm can be used instead of DES/TDES by configuring the TDESMOD field in TDES_MR with the appropriate value 0x2. An XTEA key consists of a 128-bit key. They are defined in the Key 1 and 2 Word Registers (TDES_KEY1WRx, TDES_KEY2WRx).

The number of rounds of XTEA is defined in TDES_XTEA_RNDR and can be programmed up to 64 (1 round = 2 Feistel network rounds).

All the start and operating modes of the TDES algorithm can be applied to the XTEA algorithm.

50.4.1 Operating Modes

The TDES supports the following operating modes:

- ECB—Electronic Code Book
- CBC—Cipher Block Chaining

- OFB—Output Feedback
- CFB—Cipher Feedback
  - CFB8 (CFB where the length of the data segment is 8 bits)
  - CFB16 (CFB where the length of the data segment is 16 bits)
  - CFB32 (CFB where the length of the data segment is 32 bits)
  - CFB64 (CFB where the length of the data segment is 64 bits)

The data pre-processing, post-processing and data chaining for each mode are automatically performed. Refer to the *FIPS Publication 81* for more complete information.

These modes are selected by setting the OPMOD field in TDES_MR.

In CFB mode, four data sizes are possible (8, 16, 32 and 64 bits), configurable by means of the CFBS field in TDES_MR (refer to Section 50.5.2 “TDES Mode Register”).

50.4.2 Start Modes

The SMOD field in TDES_MR selects the Encryption (or Decryption) start mode.

50.4.2.1 Manual Mode

The sequence is as follows:

1. Write the TDES_MR register with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 64-bit key(s) in the different Key Word Registers (TDES_KEYxWRx), depending on whether one, two or three keys are required.
3. Write the initialization vector (or counter) in the Initialization Vector Registers (TDES_IVRx).

Note: The Initialization Vector Registers concern all modes except ECB.

4. Set the bit DATRDY (Data Ready) in the TDES Interrupt Enable register (TDES_IER), depending on whether an interrupt is required or not at the end of processing.
5. Write the data to be encrypted/decrypted in the authorized Input Data Registers (refer to Table 50-3).

Note: In 32-, 16- and 8-bit CFB modes, writing to TDES_IDATAR1 is not allowed and may lead to processing errors.

6. Set the START bit in the TDES Control Register (TDES_CR) to begin the encryption or decryption process.
7. When the processing completes, the bit DATRDY in the TDES Interrupt Status Register (TDES_ISR) rises. If an interrupt has been enabled by setting the bit DATRDY in TDES_IER, the interrupt line of the TDES is activated.
8. When the software reads one of the Output Data Registers (TDES_ODATARx), the DATRDY bit is automatically cleared.

Table 50-3. Authorized Input Data Registers

Operating Mode	Input Data Registers to Write
ECB	All
CBC	All
OFB	All
CFB 64-bit	All
CFB 32-bit	TDES_IDATAR0
CFB 16-bit	TDES_IDATAR0
CFB 8-bit	TDES_IDATAR0

50.4.2.2 Auto Mode

The Auto Mode is similar to the Manual Mode, except that as soon as the correct number of Input Data registers is written, processing is automatically started without any action in TDES_CR.

50.4.2.3 DMA Mode

The DMA Controller can be used in association with the TDES to perform an encryption/decryption of a buffer without any action by the software during processing.

The SMOD field of TDES_MR must be set to 0x2 and the DMA must be configured with non-incremental addresses.

The start address of any transfer descriptor must be set in TDES_IDATAR0.

The DMA chunk size configuration depends on the TDES mode of operation and is listed in Table 50-4.

When writing data to TDES with the first DMA channel, data will be fetched from a memory buffer (source data). It is recommended to configure the size of source data to “words” even for CFB modes. On the contrary, the destination data size depends on the mode of operation. When reading data from the TDES with the second DMA channel, the source data is the data read from TDES and data destination is the memory buffer. In this case, source data size depends on the TDES mode of operation and is listed in Table 50-4.

Table 50-4. DMA Data Transfer Type for the Different Operating Modes

Operating Mode	Chunk Size	Destination/Source Data Transfer Type
ECB	1	Word
CBC	1	Word
OFB	1	Word
CFB 64-bit	1	Word
CFB 32-bit	1	Word
CFB 16-bit	1	Half-word
CFB 8-bit	1	Byte

50.4.3 Last Output Data Mode

This mode is used to generate cryptographic checksums on data (MAC) using a CBC-MAC or a CFB encryption algorithm (refer to *FIPS Publication 81 Appendix F*).

After each end of encryption/decryption, the output data is available either on the output data registers for Manual and Auto modes or at the address specified in the receive buffer pointer for DMA mode (refer to Table 50-5 “Last Output Data Mode Behavior versus Start Modes”).

The Last Output Data bit (LOD) in TDES_MR can be used to retrieve only the last data of several encryption/decryption processes.

This data is only available on the Output Data Registers (TDES_ODATARx).

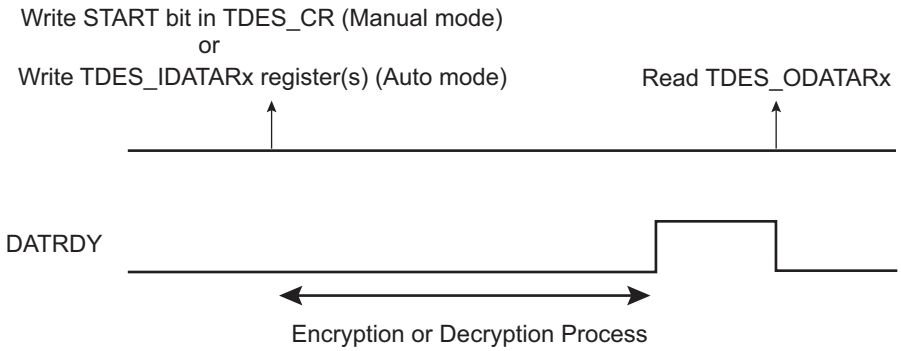
Therefore, there is no need to define a read buffer in DMA mode.

50.4.3.1 Manual and Auto Modes

TDES_MR.LOD = 0

The DATRDY flag is cleared when at least one of the Output Data Registers is read. Refer to Figure 50-1.

Figure 50-1. Manual and Auto Modes with LOD = 0

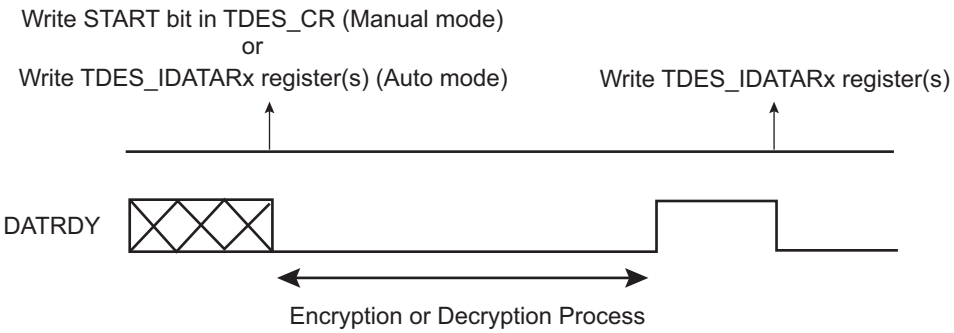


If the user does not want to read the output data registers between each encryption/decryption, the DATRDY flag will not be cleared. If the DATRDY flag is not cleared, the user will not be informed of the end of the encryptions/decryptions that follow.

TDES_MR.LOD = 1

The DATRDY flag is cleared when at least one Input Data Register is written, before the start of a new transfer. Refer to Figure 50-2. No further Output Data Register reads are necessary between consecutive encryptions/decryptions.

Figure 50-2. Manual and Auto Modes with LOD = 1



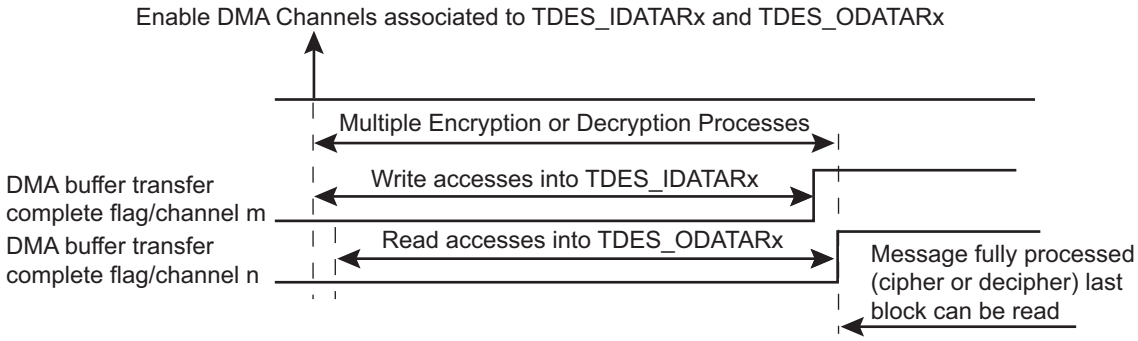
50.4.3.2 DMA Mode

TDES_MR.LOD = 0

This mode may be used for all TDES operating modes except CBC-MAC where LOD = 1 mode is recommended.

The end of the encryption/decryption is indicated by the end of DMA transfer associated to TDES_ODATARx (refer to Figure 50-3). Two DMA channels are required: one for writing message blocks to TDES_IDATARx and one to obtain the result from TDES_ODATARx.

Figure 50-3. DMA Transfer with LOD = 0



TDES_MR.LOD = 1

This mode is optimized to process the TDES CBC-MAC operating mode.

The user must first wait for the DMA buffer transfer complete flag, then for the flag DATRDY to rise to ensure that the encryption/decryption is completed (refer to Figure 50-4).

In this case, no receive buffers are required.

The output data is only available on TDES_ODATARx.

Figure 50-4. DMA Transfer with LOD = 1

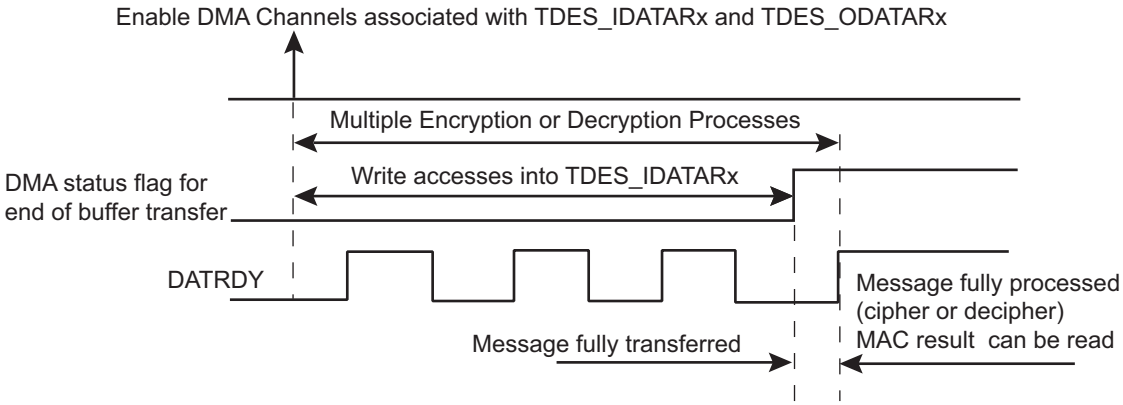


Table 50-5 summarizes the different cases.

Table 50-5. Last Output Data Mode Behavior versus Start Modes

Sequence	Manual and Auto Modes		DMA Transfer	
	LOD = 0	LOD = 1	LOD = 0	LOD = 1
DATRDY Flag Clearing Condition (1)	At least one Output Data Register must be read	At least one Input Data Register must be written	Not used	Managed by the DMA
End of Encryption/Decryption	DATRDY	DATRDY	2 DMA Buffer transfer complete flags (channel m and channel n)	DMA buffer transfer complete flag, then TDES DATRDY flag
Encrypted/Decrypted Data Result Location	In the Output Data Registers	In the Output Data Registers	Not available	In the Output Data Registers

Note: 1. Depending on the mode, there are other ways of clearing the DATRDY flag. Refer to Section 50.5.6 “TDES Interrupt Status Register”.

Warning: In DMA mode, reading to the Output Data registers before the last data transfer may lead to unpredictable results.

50.4.4 Security Features

50.4.4.1 Unspecified Register Access Detection

When an unspecified register access occurs, the URAD bit in TDES_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:

- Input Data Register written during the data processing in DMA mode
- Output Data Register read during the data processing
- Mode Register written during the data processing
- Write-only register read access

The URAD bit and the URAT field can only be reset by the SWRST bit in TDES_CR.

50.5 Triple Data Encryption Standard (TDES) User Interface

Table 50-6. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	TDES_CR	Write-only	–
0x04	Mode Register	TDES_MR	Read/Write	0x2
0x08–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	TDES_IER	Write-only	–
0x14	Interrupt Disable Register	TDES_IDR	Write-only	–
0x18	Interrupt Mask Register	TDES_IMR	Read-only	0x0
0x1C	Interrupt Status Register	TDES_ISR	Read-only	0x0000001E
0x20	Key 1 Word Register 0	TDES_KEY1WR0	Write-only	–
0x24	Key 1 Word Register 1	TDES_KEY1WR1	Write-only	–
0x28	Key 2 Word Register 0	TDES_KEY2WR0	Write-only	–
0x2C	Key 2 Word Register 1	TDES_KEY2WR1	Write-only	–
0x30	Key 3 Word Register 0	TDES_KEY3WR0	Write-only	–
0x34	Key 3 Word Register 1	TDES_KEY3WR1	Write-only	–
0x38–0x3C	Reserved	–	–	–
0x40	Input Data Register 0	TDES_IDATAR0	Write-only	–
0x44	Input Data Register 1	TDES_IDATAR1	Write-only	–
0x48–0x4C	Reserved	–	–	–
0x50	Output Data Register 0	TDES_ODATAR0	Read-only	0x0
0x54	Output Data Register 1	TDES_ODATAR1	Read-only	0x0
0x58–0x5C	Reserved	–	–	–
0x60	Initialization Vector Register 0	TDES_IVR0	Write-only	–
0x64	Initialization Vector Register 1	TDES_IVR1	Write-only	–
0x68–0x6C	Reserved	–	–	–
0x70	XTEA Rounds Register	TDES_XTEA_RNDR	Read/Write	0x0
0x74–0xFC	Reserved	–	–	–

50.5.1 TDES Control Register

Name: TDES_CR

Address: 0xFC04C000

Access: Write-only

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

- **START: Start Processing**

0: No effect

1: Starts Manual encryption/decryption process.

- **SWRST: Software Reset**

0: No effect

1: Resets the TDES. A software triggered hardware reset of the TDES interface is performed.

50.5.2 TDES Mode Register

Name: TDES_MR

Address: 0xFC04C004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	CFBS	
15	14	13	12	11	10	9	8
LOD	–	OPMOD		–	–	SMOD	
7	6	5	4	3	2	1	0
–	–	–	KEYMOD	–	TDESMOD		CIPHER

- **CIPHER: Processing Mode**

0 (DECRYPT): Decrypts data.

1 (ENCRYPT): Encrypts data.

- **TDESMOD: ALGORITHM Mode**

Value	Name	Description
0x0	SINGLE_DES	Single DES processing using TDES_KEY1WRx registers
0x1	TRIPLE_DES	Triple DES processing using TDES_KEY1WRx, TDES_KEY2WRx and TDES_KEY3WRx registers
0x2	XTEA	XTEA processing using TDES_KEY1WRx, TDES_KEY2WRx

Values which are not listed in the table must be considered as “reserved”.

- **KEYMOD: Key Mode**

0: Three-key algorithm is selected.

1: Two-key algorithm is selected. There is no need to write TDES_KEY3WRx registers.

- **SMOD: Start Mode**

Value	Name	Description
0x0	MANUAL_START	Manual Mode
0x1	AUTO_START	Auto Mode
0x2	IDATAR0_START	TDES_IDATAR0 accesses only Auto Mode

Values which are not listed in the table must be considered as “reserved”.

If a DMA transfer is used, 0x2 must be configured. Refer to Section 50.4.3.2 “DMA Mode” for more details.

- **OPMOD: Operating Mode**

Value	Name	Description
0x0	ECB	Electronic Code Book mode
0x1	CBC	Cipher Block Chaining mode
0x2	OFB	Output Feedback mode

Value	Name	Description
0x3	CFB	Cipher Feedback mode

For CBC-MAC operating mode, set OPMOD to CBC and LOD to 1.

- **LOD: Last Output Data Mode**

0: No effect.

After each end of encryption/decryption, the output data is available either on the output data registers (Manual and Auto modes).

In Manual and Auto modes, the DATRDY flag is cleared when at least one of the Output Data registers is read.

1: The DATRDY flag is cleared when at least one of the Input Data Registers is written.

No more Output Data Register reads are necessary between consecutive encryptions/decryptions (refer to Section 50.4.3 “Last Output Data Mode”).

Warning: In DMA mode, reading to the Output Data registers before the last data encryption/decryption process may lead to unpredictable result.

- **CFBS: Cipher Feedback Data Size**

Value	Name	Description
0x0	SIZE_64BIT	64-bit
0x1	SIZE_32BIT	32-bit
0x2	SIZE_16BIT	16-bit
0x3	SIZE_8BIT	8-bit

50.5.3 TDES Interrupt Enable Register

Name: TDES_IER

Address: 0xFC04C010

Access: Write-only

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

- **DATRDY: Data Ready Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **URAD: Unspecified Register Access Detection Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

50.5.4 TDES Interrupt Disable Register

Name: TDES_IDR

Address: 0xFC04C014

Access: Write-only

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

- **DATRDY: Data Ready Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **URAD: Unspecified Register Access Detection Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

50.5.5 TDES Interrupt Mask Register

Name: TDES_IMR

Address: 0xFC04C018

Access: Read-only

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

- **DATRDY: Data Ready Interrupt Mask**

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **URAD: Unspecified Register Access Detection Interrupt Mask**

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

50.5.6 TDES Interrupt Status Register

Name: TDES_ISR

Address: 0xFC04C01C

Access: Read-only

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

- **DATRDY: Data Ready (cleared by setting bit START or bit SWRST in TDES_CR or by reading TDES_ODATARx)**

0: Output data is not valid.

1: Encryption or decryption process is completed.

Note: If TDES_MR.LOD = 1: In Manual and Auto modes, the DATRDY flag can also be cleared by writing at least one TDES_IDATARx.

- **URAD: Unspecified Register Access Detection Status (cleared by setting bit TDES_CR.SWRST)**

0: No unspecified register access has been detected since the last write of bit TDES_CR.SWRST.

1: At least one unspecified register access has been detected since the last write of bit TDES_CR.SWRST.

- **URAT: Unspecified Register Access (cleared by setting bit TDES_CR.SWRST)**

Value	Name	Description
0x0	IDR_WR_PROCESSING	Input Data Register written during data processing when SMOD = 0x2 mode.
0x1	ODR_RD_PROCESSING	Output Data Register read during data processing.
0x2	MR_WR_PROCESSING	Mode Register written during data processing.
0x3	WOR_RD_ACCESS	Write-only register read access.

Only the last Unspecified Register Access Type is available through the URAT field.

50.5.7 TDES Key 1 Word Register x

Name: TDES_KEY1WRx

Address: 0xFC04C020

Access: Write-only

31	30	29	28	27	26	25	24
KEY1W							

23	22	21	20	19	18	17	16
KEY1W							

15	14	13	12	11	10	9	8
KEY1W							

7	6	5	4	3	2	1	0
KEY1W							

- **KEY1W: Key 1 Word**

The two 32-bit Key 1 Word registers are used to set the 64-bit cryptographic key used for encryption/decryption.

KEY1W0 refers to the first word of the key and KEY1W1 to the last one.

These registers are write-only to prevent the key from being read by another application.

In XTEA mode, the key is defined on 128 bits. These registers contain the 64 LSB bits of the encryption/decryption key.

50.5.8 TDES Key 2 Word Register x

Name: TDES_KEY2WRx

Address: 0xFC04C028

Access: Write-only

31	30	29	28	27	26	25	24
KEY2W							

23	22	21	20	19	18	17	16
KEY2W							

15	14	13	12	11	10	9	8
KEY2W							

7	6	5	4	3	2	1	0
KEY2W							

- **KEY2W: Key 2 Word**

The two 32-bit Key 2 Word registers are used to set the 64-bit cryptographic key used for encryption/decryption.

KEY2W0 refers to the first word of the key and KEY2W1 to the last one.

These registers are write-only to prevent the key from being read by another application.

Note: KEY2WRx registers are not used in DES mode.

In XTEA mode, the key is defined on 128 bits. These registers contain the 64 MSB bits of the encryption/decryption key.

50.5.9 TDES Key 3 Word Register x

Name: TDES_KEY3WRx

Address: 0xFC04C030

Access: Write-only

31	30	29	28	27	26	25	24
KEY3W							

23	22	21	20	19	18	17	16
KEY3W							

15	14	13	12	11	10	9	8
KEY3W							

7	6	5	4	3	2	1	0
KEY3W							

- **KEY3W: Key 3 Word**

The two 32-bit Key 3 Word registers are used to set the 64-bit cryptographic key used for encryption/decryption.

KEY3W0 refers to the first word of the key and KEY3W1 to the last one.

These registers are write-only to prevent the key from being read by another application.

Note: KEY3WRx registers are not used in DES mode, TDES with two-key algorithm selected and XTEA mode.

50.5.10 TDES Input Data Register x

Name: TDES_IDATARx

Address: 0xFC04C040

Access: Write-only

31	30	29	28	27	26	25	24
IDATA							

23	22	21	20	19	18	17	16
IDATA							

15	14	13	12	11	10	9	8
IDATA							

7	6	5	4	3	2	1	0
IDATA							

- **IDATA: Input Data**

The two 32-bit Input Data registers are used to set the 64-bit data block used for encryption/decryption.

IDATA0 refers to the first word of the data to be encrypted/decrypted, and IDATA1 to the last one.

These registers are write-only to prevent the input data from being read by another application.

50.5.11 TDES Output Data Register x

Name: TDES_ODATARx

Address: 0xFC04C050

Access: Read-only

31	30	29	28	27	26	25	24
ODATA							

23	22	21	20	19	18	17	16
ODATA							

15	14	13	12	11	10	9	8
ODATA							

7	6	5	4	3	2	1	0
ODATA							

- **ODATA: Output Data**

The two 32-bit Output Data registers contain the 64-bit data block which has been encrypted/decrypted.

ODATA1 refers to the first word, ODATA2 to the last one.

50.5.12 TDES Initialization Vector Register x

Name: TDES_IVRx

Address: 0xFC04C060

Access: Write-only

31	30	29	28	27	26	25	24
IV							

23	22	21	20	19	18	17	16
IV							

15	14	13	12	11	10	9	8
IV							

7	6	5	4	3	2	1	0
IV							

- **IV: Initialization Vector**

The two 32-bit Initialization Vector registers are used to set the 64-bit initialization vector data block, which is used by some modes of operation as an additional initial input.

IV1 refers to the first word of the Initialization Vector, IV2 to the last one.

These registers are write-only to prevent the Initialization Vector from being read by another application.

Note: These registers are not used for the ECB mode and must not be written.

50.5.13 TDES XTEA Rounds Register

Name: TDES_XTEA_RNDR

Address: 0xFC04C070

Access: Read/Write

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
–	–	XTEA_RNDS					

- **XTEA_RNDS: Number of Rounds**

This 6-bit field is used to define the number of complete rounds (1 complete round = 2 Feistel rounds) processed in XTEA algorithm.

The value of XTEA_RNDS has no effect if the TDESMOD field in TDES_MR is set to 0x0 or 0x1.

Note: 0x00 corresponds to 1 complete round, 0x01 corresponds to 2 complete rounds, etc.

51. Secure Hash Algorithm (SHA)

51.1 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.

51.2 Embedded Characteristics

- Supports Secure Hash Algorithm (SHA1, SHA224, SHA256, SHA384, SHA512)
- Compliant with *FIPS Publication 180-2*
- Supports initial hash values registers (HMAC acceleration or other)
- 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

51.3 Product Dependencies

51.3.1 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.

51.3.2 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.

Table 51-1. Peripheral IDs

Instance	ID
SHA	15

51.4 Functional Description

The Secure Hash Algorithm (SHA) module requires a padded message according to FIPS180-2 specification. 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.

51.4.1 SHA Algorithm

The SHA can process SHA1, SHA224, SHA256, SHA384, SHA512 by configuring the ALGO field in the SHA Mode register (SHA_MR).

51.4.2 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.

51.4.3 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.

51.4.4 Internal Registers for Initial Hash Value

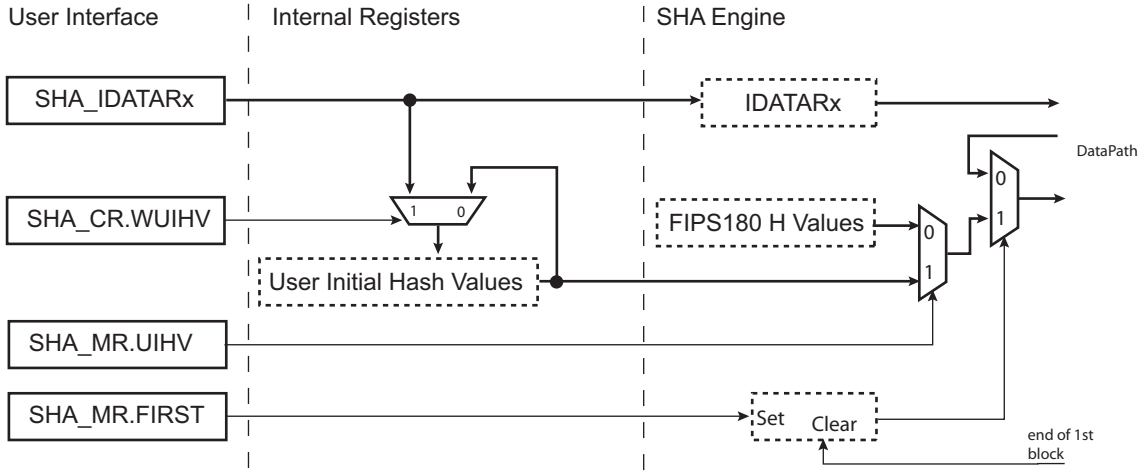
The SHA module embeds a set of initial hash value registers to store user initial hash values (refer to Figure 51-1). These registers are internal registers and are accessed through SHA Input Data registers (SHA_IDATARx). The initial hash value registers can be used to compute a custom hash algorithm with different initial constants, or to continue a hash computation by providing the intermediate hash value previously returned by the SHA module.

To write the initial hash values in the registers, follow this sequence:

1. Set bit WUIHV in SHA_CR.
2. Write the initial hash values in SHA_IDATARx. The number of registers to write depends on the hash algorithm selected:
   - SHA_IDATAR0 to SHA_IDATAR4 for the SHA1 algorithm
   - SHA_IDATAR0 to SHA_IDATAR7 for the SHA224 and SHA256 algorithms
   - SHA_IDATAR0 to SHA_IDATAR15 for the SHA384 and SHA512 algorithms
3. Clear bit WUIHV in SHA_CR.

The internal registers are selected when bit UIHV is set in SHA_MR.

Figure 51-1. User Initial Hash Value Internal Register Access



51.4.5 Start Modes

The SMOD field in the SHA_MR is used to select the Hash Processing Start mode.

51.4.5.1 Manual Mode

In Manual mode, the sequence is as follows:

1. Set the bit DATRDY (Data Ready) in the SHA_IER, depending on whether an interrupt is required at the end of processing.
2. If the initial hash values differ from the FIPS standard, set the bits UIHV in the SHA_MR depending on the configure the initial values.
   If the initial hash values comply with the FIPS180-2 specification, clear the bits UIHV in the SHA_MR.
3. For the first block of a message, the FIRST command must be set by writing a 1 into the corresponding bit of the SHA Control Register (SHA_CR). For the other blocks, there is nothing to write.
4. Write the block to be processed in the SHA_IDATARx.
5. To begin processing, set the START bit in the SHA_CR.
6. When processing is completed, the bit DATRDY in the Interrupt Status register (SHA_ISR) raises. If an interrupt has been enabled by setting the bit DATRDY in SHA_IER, the interrupt line of the SHA is activated.
7. 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.
8. After the last block is processed (DATRDY flag is set, if an interrupt has been enabled by setting the bit DATRDY in SHA_IER, 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.

51.4.5.2 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.

51.4.5.3 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.

The SMOD field in SHA_MR must be configured to 2.

The DMA must be configured with non-incremental addresses.

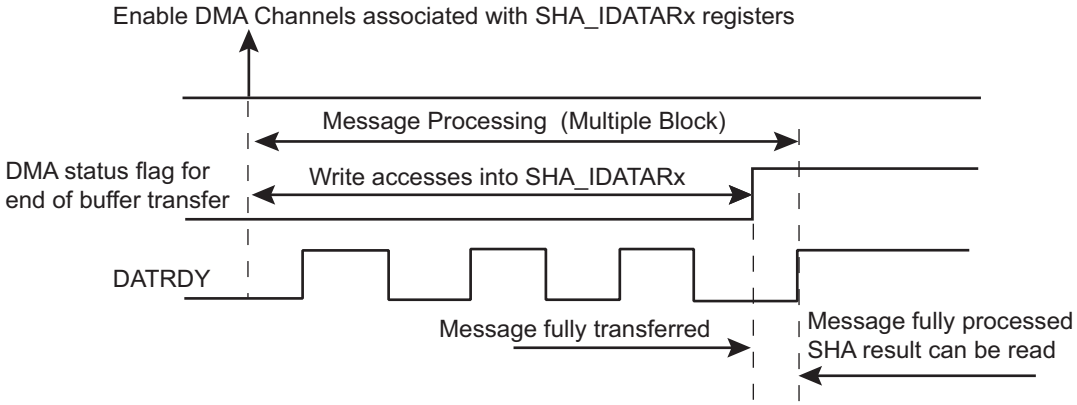
The start address of any transfer descriptor must be set to point to the SHA_IDATAR0.

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_SR.
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 (refer to Figure 51-2).

Figure 51-2. Interrupts Processing with DMA



51.4.5.4 SHA Register Endianism

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 endianism 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

51.4.6 Security Features

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.

51.5 Secure Hash Algorithm (SHA) User Interface

Table 51-2. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	SHA_CR	Write-only	–
0x04	Mode Register	SHA_MR	Read/Write	0x0000100
0x08–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	SHA_IER	Write-only	–
0x14	Interrupt Disable Register	SHA_IDR	Write-only	–
0x18	Interrupt Mask Register	SHA_IMR	Read-only	0x0
0x1C	Interrupt Status Register	SHA_ISR	Read-only	0x0
0x20–0x3C	Reserved	–	–	–
0x40	Input Data 0 Register	SHA_IDATAR0	Write-only	–
...	...	...	...	...
0x7C	Input Data 15 Register	SHA_IDATAR15	Write-only	–
0x80	Input/Output Data 0 Register	SHA_IODATAR0	Read/Write	0x0
...	...	...	...	...
0x9C	Input/Output Data 7 Register	SHA_IODATAR7	Read/Write	0x0
0xA0	Input/Output Data 8 Register	SHA_IODATAR8	Read/Write	0x0
...	...	...	...	...
0xBC	Input/Output Data 15 Register	SHA_IODATAR15	Read/Write	0x0
0xC0–0xFC	Reserved	–	–	–

51.5.1 SHA Control Register

Name: SHA_CR

Address: 0xFC050000

Access: Write-only

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

- **START: Start Processing**

0: No effect.

1: Starts manual hash algorithm process.

- **FIRST: First Block of a Message**

0: No effect.

1: Indicates that the next block to process is the first one of a message.

- **SWRST: Software Reset**

0: No effect.

1: Resets the SHA. A software-triggered hardware reset of the SHA interface is performed.

- **WUIHV: Write User Initial Hash Values**

0: SHA_IDATARx accesses are routed to the data registers.

1: SHA_IDATARx accesses are routed to the internal registers (user initial hash values).

51.5.2 SHA Mode Register

Name: SHA_MR

Address: 0xFC050004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	DUALBUFF
15	14	13	12	11	10	9	8
–	–	–	–	ALGO			
7	6	5	4	3	2	1	0
–	–	UIHV	PROCDLY	–	–	SMOD	

- **SMOD: Start Mode**

Value	Name	Description
0	MANUAL_START	Manual Mode
1	AUTO_START	Auto Mode
2	IDATAR0_START	SHA_IDATAR0 access only Auto Mode

Values not listed in the table must be considered as “reserved”.

If a DMA transfer is used, configure the SMOD value with 1 or 2. Refer to Section 51.4.5.3 “DMA Mode” for more details.

- **PROCDLY: Processing Delay**

Value	Name	Description
0	SHORTEST	SHA processing runtime is the shortest one
1	LONGEST	SHA processing runtime is the longest one (reduces the SHA bandwidth requirement, reduces the system bus overload)

When SHA1 algorithm is processed, runtime period is either 85 or 209 clock cycles.

When SHA256 or SHA224 algorithm is processed, runtime period is either 72 or 194 clock cycles.

When SHA384 or SHA512 algorithm is processed, runtime period is either 88 or 209 clock cycles.

- **UIHV: User Initial Hash Values**

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 initial hash value registers.

- **ALGO: SHA Algorithm**

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

Values not listed in the table must be considered as “reserved”.

- **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.

51.5.3 SHA Interrupt Enable Register

Name: SHA_IER

Address: 0xFC050010

Access: Write-only

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

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

0: No effect.

1: Enables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Enable**
- **URAD: Unspecified Register Access Detection Interrupt Enable**

51.5.4 SHA Interrupt Disable Register

Name: SHA_IDR

Address: 0xFC050014

Access: Write-only

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

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

0: No effect.

1: Disables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Disable**
- **URAD: Unspecified Register Access Detection Interrupt Disable**

51.5.5 SHA Interrupt Mask Register

Name: SHA_IMR

Address: 0xFC050018

Access: Read-only

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

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.

- **DATRDY: Data Ready Interrupt Mask**
- **URAD: Unspecified Register Access Detection Interrupt Mask**

51.5.6 SHA Interrupt Status Register

Name: SHA_ISR

Address: 0xFC05001C

Access: Read-only

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

- **DATRDY: Data Ready (cleared by writing a 1 to bit SWRST or START in SHA_CR, or by reading SHA_IODATARx)**

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.

- **URAD: Unspecified Register Access Detection Status (cleared by writing a 1 to SWRST bit in SHA_CR)**

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.

- **URAT: Unspecified Register Access Type (cleared by writing a 1 to SWRST bit in SHA_CR)**

Value	Description
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.

Only the last Unspecified Register Access Type is available through the URAT field.

51.5.7 SHA Input Data x Register

Name: SHA_IDATARx [x=0..15]

Address: 0xFC050040

Access: Write-only

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

- **IDATA: 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 (refer to Section 51.5.8 “SHA Input/Output Data Register x”).

51.5.8 SHA Input/Output Data Register x

Name: SHA_IODATARx [x=0..15]

Address: 0xFC050080

Access: Read/Write

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

- **IODATA: 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.

52. Advanced Encryption Standard Bridge (AESB)

52.1 Description

The Advanced Encryption Standard Bridge (AESB) is intended to provide on-the-fly off-chip memory encryption/decryption compliant with the American *FIPS (Federal Information Processing Standard) Publication 197* specification.

The AESB supports three confidentiality modes of operation for symmetrical key block cipher algorithms (ECB, CBC and CTR), as specified in the *NIST Special Publication 800-38A Recommendation*.

The 128-bit key is stored in four 32-bit registers (AESB_KEYWRx) which are all write-only.

The 128-bit input data and initialization vector (for some modes) are each stored in four 32-bit registers (AESB_IDATARx and AESB_IVRx) which are all write-only.

As soon as the initialization vector, the input data and the key are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data will be ready to be read out on the four 32-bit output data registers (AESB_ODATARx).

52.2 Embedded Characteristics

- On-the-fly off-chip memory encryption/decryption
- Compliant with *FIPS Publication 197, Advanced Encryption Standard (AES)*
- 128-bit cryptographic key
- On-The-Fly encryption/decryption
- 10 clock cycles encryption/decryption inherent processing time
- Double input buffer optimizes runtime
- Support of the three standard modes of operation specified in the *NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques:*
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC) including CBC-MAC
  - Counter (CTR)
- Last Output Data mode allows optimized Message Authentication Code (MAC) generation

52.3 Product Dependencies

52.3.1 Power Management

The AESB may be clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the AESB clock.

52.3.2 Interrupt

The AESB interface has an interrupt line connected to the Interrupt Controller.

Handling the AESB interrupt requires programming the Interrupt Controller before configuring the AESB.

Table 52-1. Peripheral IDs

Instance	ID
AESB	13

52.4 Functional Description

The Advanced Encryption Standard Bridge (AESB) specifies a FIPS-approved cryptographic algorithm that can be used to protect electronic data. The AES algorithm is a symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information.

Encryption converts data to an unintelligible form called ciphertext. Decrypting the ciphertext converts the data back into its original form, called plaintext. The CIPHER bit in the AESB Mode Register (AESB_MR) allows selection between the encryption and the decryption processes.

The AESB is capable of using cryptographic keys of 128 bits to encrypt and decrypt data in blocks of 128 bits. This 128-bit key is defined in the Key Registers (AESB_KEYWRx).

The input to the encryption processes of the CBC mode includes, in addition to the plaintext, a 128-bit data block called the initialization vector (IV), which must be set in the Initialization Vector Registers (AESB_IVRx). The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message. The Initialization Vector Registers are also used by the CTR mode to set the counter value.

52.4.1 Operating Modes

The AESB supports the following modes of operation:

- ECB—Electronic Code Book
- CBC—Cipher Block Chaining
- CTR—Counter

The data pre-processing, post-processing and data chaining for the operating modes are performed automatically. Refer to the *NIST Special Publication 800-38A Recommendation* for more complete information.

The modes are selected by the OPMOD field in AESB_MR.

In CTR mode, the size of the block counter embedded in the module is 16 bits. Therefore, there is a rollover after processing 1 megabyte of data. If the file to be processed is greater than 1 megabyte, this file must be split into fragments of 1 megabyte or less for the first fragment if the initial value of the counter is greater than 0. Prior to loading the first fragment into AESB_IDATARx registers, the AESB_IVRx registers must be cleared. For any fragment, after the transfer is completed and prior to transferring the next fragment, AESB_IVR0 must be programmed so that the fragment number (0 for the first fragment, 1 for the second one, and so on) is written in the 16 MSB of AESB_IVR0.

If the initial value of the counter is greater than 0 and the data buffer size to be processed is greater than 1 megabyte, the size of the first fragment to be processed must be 1 megabyte minus 16x(initial value) to prevent a rollover of the internal 1-bit counter.

52.4.2 Double Input Buffer

The input data register can be double-buffered to reduce the runtime of large files.

This mode allows writing a new message block when the previous message block is being processed.

The DUALBUFF bit in register AESB_MR must be set to 1 to access the double buffer.

52.4.3 Start Modes

The SMOD field in register AESB_MR allows selection of the Encryption (or Decryption) Start mode.

52.4.3.1 Manual Mode

The sequence is as follows:

1. Write AESB_MR with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 128-bit key in the Key Registers (AESB_KEYWRx).
3. Write the initialization vector (or counter) in the Initialization Vector Registers (AESB_IVRx).

Note: The Initialization Vector Registers concern all modes except ECB.

4. Set the DATRDY (Data Ready) bit in the AESB Interrupt Enable Register (AESB_IER) depending on whether an interrupt is required, or not, at the end of processing.
5. Write the data to be encrypted/decrypted in the authorized Input Data Registers (refer to Table 52-2).

Table 52-2. Authorized Input Data Registers

Operating Mode	Input Data Registers to Write
ECB	All
CBC	All
CTR	All

6. Set the START bit in the AESB Control Register (AESB_CR) to begin the encryption or decryption process.
7. When processing is complete, the DATRDY bit in the AESB Interrupt Status Register (AESB_ISR) raises. If an interrupt has been enabled by setting the DATRDY bit in AESB_IER, the interrupt line of the AESB is activated.
8. When the software reads one of the Output Data Registers (AESB_ODATARx), the AESB_ISR.DATRDY bit is automatically cleared.

52.4.3.2 Auto Mode

Auto mode is similar to Manual mode, except that in Auto mode, as soon as the correct number of Input Data registers is written, processing starts automatically without any action in the Control Register.

52.4.4 Last Output Data Mode

Last Output Data mode is used to generate cryptographic checksums on data (MAC) by means of a cipher block chaining encryption algorithm (the CBC-MAC algorithm for example).

After each end of encryption/decryption, the output data are available on the output data registers for Manual and Auto modes.

The Last Output Data (LOD) bit in AESB_MR allows retrieval of only the last data of several encryption/decryption processes.

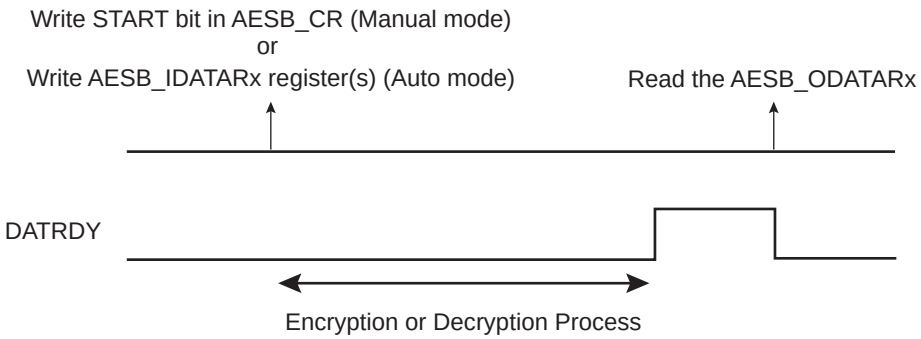
Those data are only available on the Output Data Registers (AESB_ODATARx).

52.4.5 Manual and Auto Modes

52.4.5.1 If AESB_MR.LOD = 0

The AESB_ISR.DATRDY bit is cleared when at least one of the Output Data Registers is read (refer to Figure 52-1).

Figure 52-1. Manual and Auto Modes with AESB_MR.LOD = 0

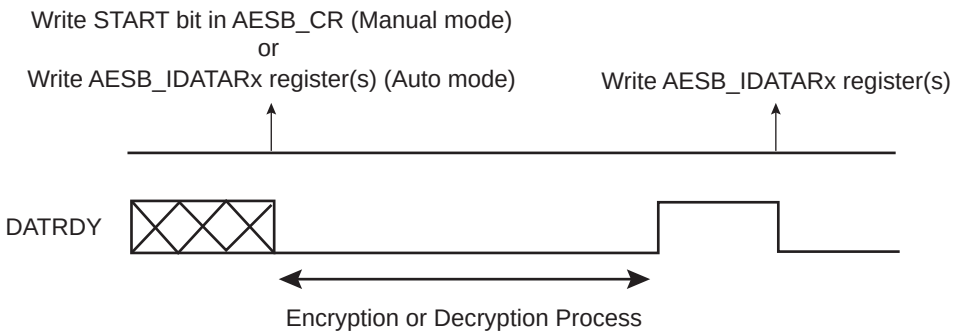


If the user does not want to read the output data registers between each encryption/decryption, the AESB_ISR.DATRDY bit will not be cleared. If the AESB_ISR.DATRDY bit is not cleared, the user cannot know the end of the following encryptions/decryptions.

52.4.5.2 If AESB_MR.LOD = 1

The AESB_ISR.DATRDY bit is cleared when at least one Input Data Register is written, so before the start of a new transfer (refer to Figure 52-2). No more Output Data Register reads are necessary between consecutive encryptions/decryptions.

Figure 52-2. Manual and Auto Modes with AESB_MR.LOD = 1



52.4.6 Automatic Bridge Mode

52.4.6.1 Description

The Automatic Bridge mode, when the AESB block is connected between the system bus and a DDR port, provides automatic encryption/decryption to/from a DDR port without any action on the part of the user. For Automatic Bridge mode, the OPMODE field must be configured to 0x4 in AESB_MR (refer to Section 52.6.2 “AESB Mode Register”). If bit AESB_MR.AAHB is set and field AESB_MR.OPMODE = 0x4, there is no compliance with the standard CTR mode of operation.

In case of write transfer, this mode automatically encrypts the data before writing it to the final slave destination. In case of read transfer, this mode automatically decrypts the data read from the target slave before putting it on the system bus.

Therefore, this mode does not work if the automatically encrypted data is moved at another address outside of the AESB IP scope. This means that for a given data, the encrypted value is not the same if written at different addresses.

52.4.6.2 Configuration

The Automatic Bridge mode can be enabled by setting bit AESB_MR.AAHB.

The IV (Initialization Vector) field of the AESB Initialization Vector Register x (AESB_IVRx) can be used to add a nonce in the encryption process in order to bring even more security (ignored if not filled). In this case, any value encrypted with a given nonce can only be decrypted with this nonce. If another nonce is set for the IV field, any value encrypted with the previous nonce cannot be decrypted anymore (refer to Section 52.6.10 “AESB Initialization Vector Register x”).

Dual buffer usage (write a 1 to bit AESB_MR.DUALBUFF) is recommended for improved performance.

52.5 Security Features

52.5.1 Unspecified Register Access Detection

When an unspecified register access occurs, the URAD bit in AESB_ISR raises. Its source is then reported in the Unspecified Register Access Type (URAT) field. Only the last unspecified register access is available through the URAT field.

Several kinds of unspecified register accesses can occur:

- Input Data Register written during the data processing when SMOD = IDATAR0_START
- Output Data Register read during data processing
- Mode Register written during data processing
- Output Data Register read during sub-keys generation
- Mode Register written during sub-keys generation
- Write-only register read access

The URAD bit and the URAT field can only be reset by the SWRST bit in AESB_CR.

52.6 Advanced Encryption Standard Bridge (AESB) User Interface

Table 52-3. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	AESB_CR	Write-only	–
0x04	Mode Register	AESB_MR	Read/Write	0x0
0x08–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	AESB_IER	Write-only	–
0x14	Interrupt Disable Register	AESB_IDR	Write-only	–
0x18	Interrupt Mask Register	AESB_IMR	Read-only	0x0
0x1C	Interrupt Status Register	AESB_ISR	Read-only	0x0
0x20	Key Word Register 0	AESB_KEYWR0	Write-only	–
0x24	Key Word Register 1	AESB_KEYWR1	Write-only	–
0x28	Key Word Register 2	AESB_KEYWR2	Write-only	–
0x2C	Key Word Register 3	AESB_KEYWR3	Write-only	–
0x30–0x3C	Reserved	–	–	–
0x40	Input Data Register 0	AESB_IDATAR0	Write-only	–
0x44	Input Data Register 1	AESB_IDATAR1	Write-only	–
0x48	Input Data Register 2	AESB_IDATAR2	Write-only	–
0x4C	Input Data Register 3	AESB_IDATAR3	Write-only	–
0x50	Output Data Register 0	AESB_ODATAR0	Read-only	0x0
0x54	Output Data Register 1	AESB_ODATAR1	Read-only	0x0
0x58	Output Data Register 2	AESB_ODATAR2	Read-only	0x0
0x5C	Output Data Register 3	AESB_ODATAR3	Read-only	0x0
0x60	Initialization Vector Register 0	AESB_IVR0	Write-only	–
0x64	Initialization Vector Register 1	AESB_IVR1	Write-only	–
0x68	Initialization Vector Register 2	AESB_IVR2	Write-only	–
0x6C	Initialization Vector Register 3	AESB_IVR3	Write-only	–
0x70–0xFC	Reserved	–	–	–

52.6.1 AESB Control Register

Name: AESB_CR

Address: 0xF0020000

Access: Write-only

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

- **START: Start Processing**

0: No effect

1: Starts manual encryption/decryption process

- **SWRST: Software Reset**

0: No effect

1: Resets the AESB. A software triggered hardware reset of the AESB interface is performed.

52.6.2 AESB Mode Register

Name: AESB_MR

Address: 0xF0020004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CKEY				–	–	–	–
15	14	13	12	11	10	9	8
LOD	OPMOD			–	–	SMOD	
7	6	5	4	3	2	1	0
PROCDLY				DUALBUFF	AAHB	–	CIPHER

- **CIPHER: Processing Mode**

0: Decrypts data

1: Encrypts data

- **AAHB: Automatic Bridge Mode**

0: Automatic Bridge mode disabled

1: Automatic Bridge mode enabled

- **DUALBUFF: Dual Input Buffer**

Value	Name	Description
0x0	INACTIVE	AESB_IDATARx cannot be written during processing of previous block.
0x1	ACTIVE	AESB_IDATARx can be written during processing of previous block when SMOD = 0x2. It speeds up the overall runtime of large files.

- **PROCDLY: Processing Delay**

Processing Time = 12 × (PROCDLY + 1)

The Processing Time represents the number of clock cycles that the AESB needs in order to perform one encryption/decryption .

Note: The best performance is achieved with PROCDLY equal to 0.

- **SMOD: Start Mode**

Value	Name	Description
0x0	MANUAL_START	Manual mode
0x1	AUTO_START	Auto mode
0x2	IDATAR0_START	AESB_IDATAR0 access only Auto mode

Values which are not listed in the table must be considered as “reserved”.

- **OPMOD: Operating Mode**

Value	Name	Description
0x0	ECB	Electronic Code Book mode
0x1	CBC	Cipher Block Chaining mode
0x2	–	Reserved
0x3	–	Reserved
0x4	CTR	Counter mode (16-bit internal counter)

Values which are not listed in the table must be considered as “reserved”.

For CBC-MAC operating mode, configure OPMOD to 0x1 (CBC) and set LOD to 1.

Note: If the OPMODE field is set to 0x4 and AAHB = 1, there is no compliance with the standard CTR mode of operation.

- **LOD: Last Output Data Mode**

0: No effect.

After each end of encryption/decryption, the output data will be available either on the output data registers (Manual and Auto modes).

In Manual and Auto modes, the AESB_ISR.DATRDY bit is cleared when at least one of the Output Data registers is read.

1: The AESB_ISR.DATRDY bit is cleared when at least one of the Input Data Registers is written.

No more Output Data Register reads are necessary between consecutive encryptions/decryptions (refer to Section 52.4.4 “Last Output Data Mode”).

- **CKEY: Key**

Value	Name	Description
0xE	PASSWD	This field must be written with 0xE the first time that AES_MR is programmed. For subsequent programming of the AES_MR register, any value can be written, including that of 0xE.Always reads as 0.

52.6.3 AESB Interrupt Enable Register

Name: AESB_IER

Address: 0xF0020010

Access: Write-only

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

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

0: No effect.

1: Enables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Enable**
- **URAD: Unspecified Register Access Detection Interrupt Enable**

52.6.4 AESB Interrupt Disable Register

Name: AESB_IDR

Address: 0xF0020014

Access: Write-only

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

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

0: No effect.

1: Disables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Disable**
- **URAD: Unspecified Register Access Detection Interrupt Disable**

52.6.5 AESB Interrupt Mask Register

Name: AESB_IMR

Address: 0xF0020018

Access: Read-only

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

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.

- **DATRDY: Data Ready Interrupt Mask**
- **URAD: Unspecified Register Access Detection Interrupt Mask**

52.6.6 AESB Interrupt Status Register

Name: AESB_ISR

Address: 0xF002001C

Access: Read-only

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

- **DATRDY: Data Ready**

0: Output data not valid.

1: Encryption or decryption process is completed.

DATRDY is cleared when a Manual encryption/decryption occurs (START bit in AESB_CR) or when a software triggered hardware reset of the AESB interface is performed (SWRST bit in AESB_CR).

AESB_MR.LOD = 0: In Manual and Auto modes, the DATRDY bit can also be cleared when at least one of the Output Data Registers is read.

AESB_MR.LOD = 1: In Manual and Auto modes, the DATRDY bit can also be cleared when at least one of the Input Data Registers is written.

- **URAD: Unspecified Register Access Detection Status**

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.

URAD bit is reset only by the SWRST bit in AESB_CR.

- **URAT: Unspecified Register Access**

Value	Name	Description
0x0	IDR_WR_PROCESSING	Input Data Register written during the data processing when SMOD = 0x2 mode
0x1	ODR_RD_PROCESSING	Output Data Register read during the data processing
0x2	MR_WR_PROCESSING	Mode Register written during the data processing
0x3	ODR_RD_SUBKGEN	Output Data Register read during the sub-keys generation
0x4	MR_WR_SUBKGEN	Mode Register written during the sub-keys generation
0x5	WOR_RD_ACCESS	Write-only register read access

Only the last Unspecified Register Access Type is available through the URAT field.

URAT field is reset only by the SWRST bit in AESB_CR.

52.6.7 AESB Key Word Register x

Name: AESB_KEYWRx

Address: 0xF0020020

Access: Write-only

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

- **KEYW: Key Word**

The four 32-bit Key Word registers set the 128-bit cryptographic key used for encryption/decryption.

AESB_KEYWR0 corresponds to the first word of the key, AESB_KEYWR3 to the last one.

These registers are write-only to prevent the key from being read by another application.

52.6.8 AESB Input Data Register x

Name: AESB_IDATARx

Address: 0xF0020040

Access: Write-only

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

- **IDATA: Input Data Word**

The four 32-bit Input Data registers set the 128-bit data block used for encryption/decryption.

AESB_IDATAR0 corresponds to the first word of the data to be encrypted/decrypted, AESB_IDATAR3 to the last one.

These registers are write-only to prevent the input data from being read by another application.

52.6.9 AESB Output Data Register x

Name: AESB_ODATARx

Address: 0xF0020050

Access: Read-only

31	30	29	28	27	26	25	24
ODATA							

23	22	21	20	19	18	17	16
ODATA							

15	14	13	12	11	10	9	8
ODATA							

7	6	5	4	3	2	1	0
ODATA							

- **ODATA: Output Data**

The four 32-bit Output Data registers contain the 128-bit data block that has been encrypted/decrypted.

AESB_ODATAR0 corresponds to the first word, AESB_ODATAR3 to the last one.

52.6.10 AESB Initialization Vector Register x

Name: AESB_IVRx

Address: 0xF0020060

Access: Write-only

31	30	29	28	27	26	25	24
IV							

23	22	21	20	19	18	17	16
IV							

15	14	13	12	11	10	9	8
IV							

7	6	5	4	3	2	1	0
IV							

- **IV: Initialization Vector**

The four 32-bit Initialization Vector registers set the 128-bit Initialization Vector data block that is used by some modes of operation as an additional initial input.

AESB_IVR0 corresponds to the first word of the Initialization Vector, AESB_IVR3 to the last one.

These registers are write-only to prevent the Initialization Vector from being read by another application.

For CBC mode, the IV input value corresponds to the initialization vector.

For CTR mode, the IV input value corresponds to the initial counter value.

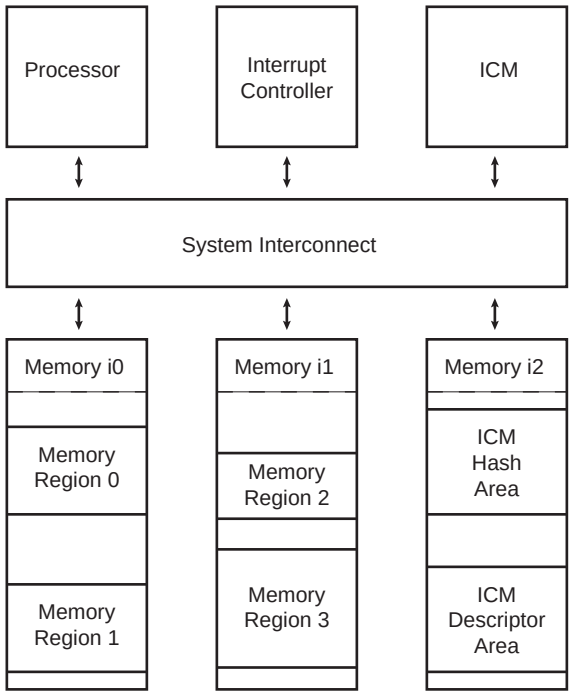
Note: These registers are not used in ECB mode and must not be written. For Automatic Bridge dedicated mode, the IV input value corresponds to the initial nonce.

53. Integrity Check Monitor (ICM)

53.1 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 controller 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. Refer to Figure 53-1 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 53-1. 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:

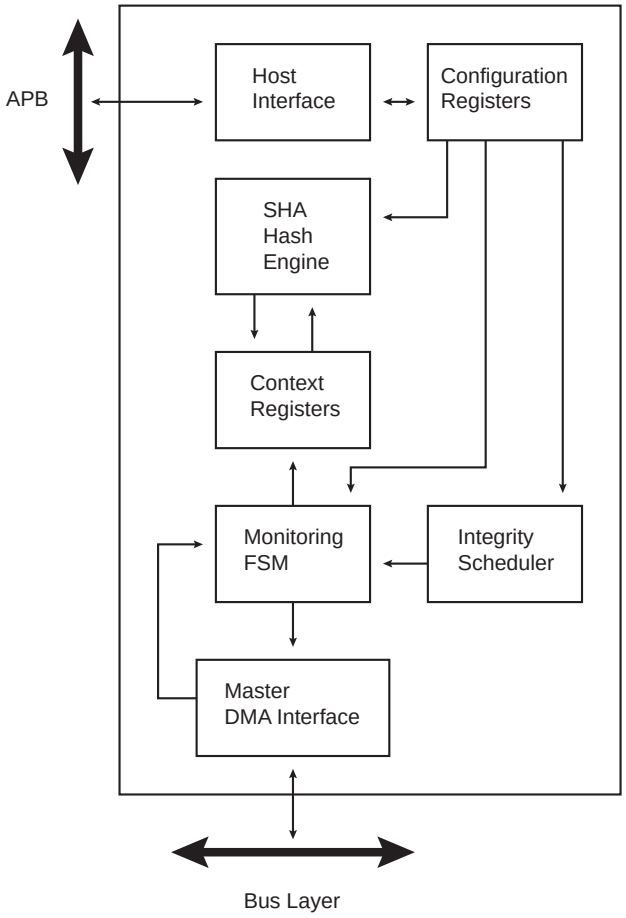
- Region—a partition of instruction or data memory space
- Region Descriptor—a data structure stored in memory, defining region attributes
- Region Attributes—region start address, region size, region SHA engine processing mode, Write Back or Compare function mode
- Context Registers—a set of ICM non-memory-mapped, 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

53.2 Embedded Characteristics

- DMA AHB master interface
- Supports monitoring of up to 4 Non-Contiguous Memory Regions
- Supports block gathering through the use of linked list
- 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

53.3 Block Diagram

Figure 53-2. Integrity Check Monitor Block Diagram



53.4 Product Dependencies

53.4.1 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.

53.4.2 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.

Table 53-1. Peripheral IDs

Instance	ID
ICM	9

53.5 Functional Description

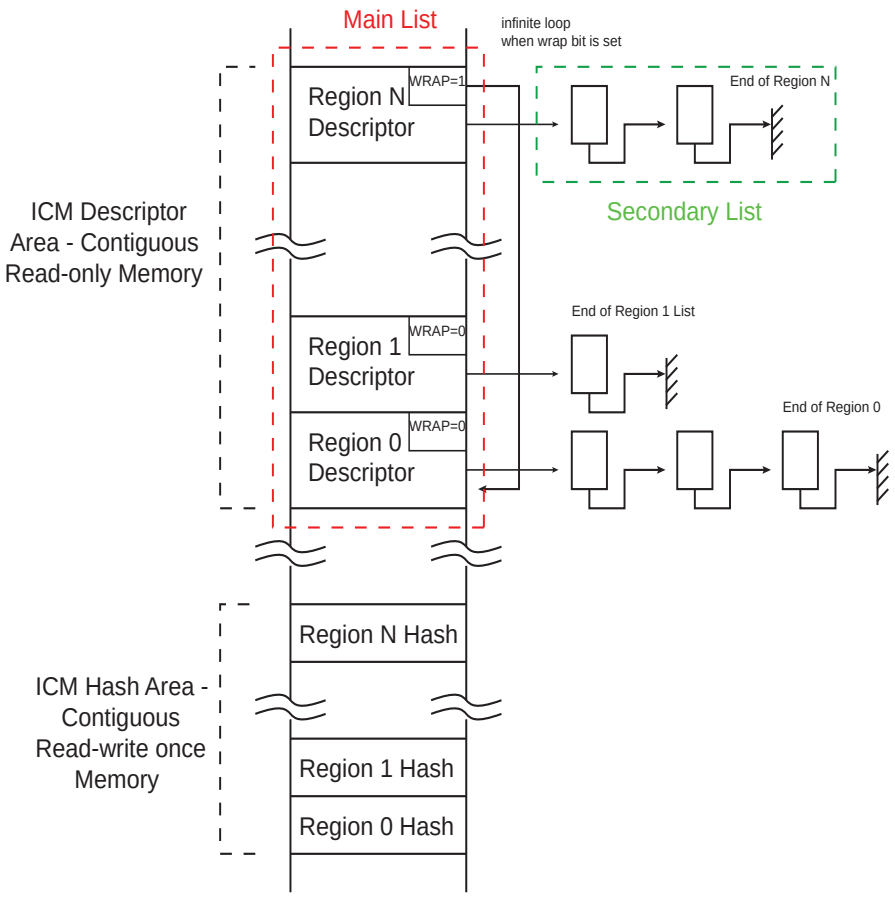
53.5.1 Overview

The Integrity Check Monitor (ICM) is a DMA controller that performs SHA-based memory hashing over memory regions. As shown in Figure 53-2, 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 53-3). Up to four regions may be monitored. Each region descriptor is composed of four words indicating the layout of the memory region (refer to Figure 53-4). 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 (refer to 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 whether 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 bit set to one. To continuously monitor the list of regions, the WRAP bit must be set to one in the last data structure.

Figure 53-3. 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 the BBC field of the ICM_CFG register to control ICM memory load.

Figure 53-4. Region Descriptor

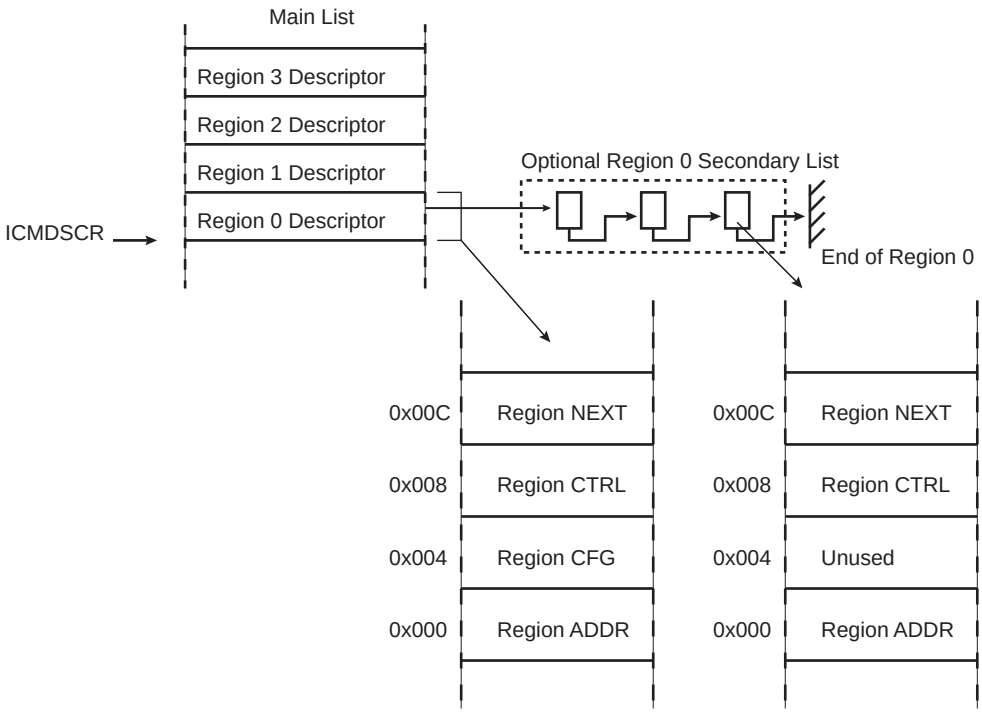
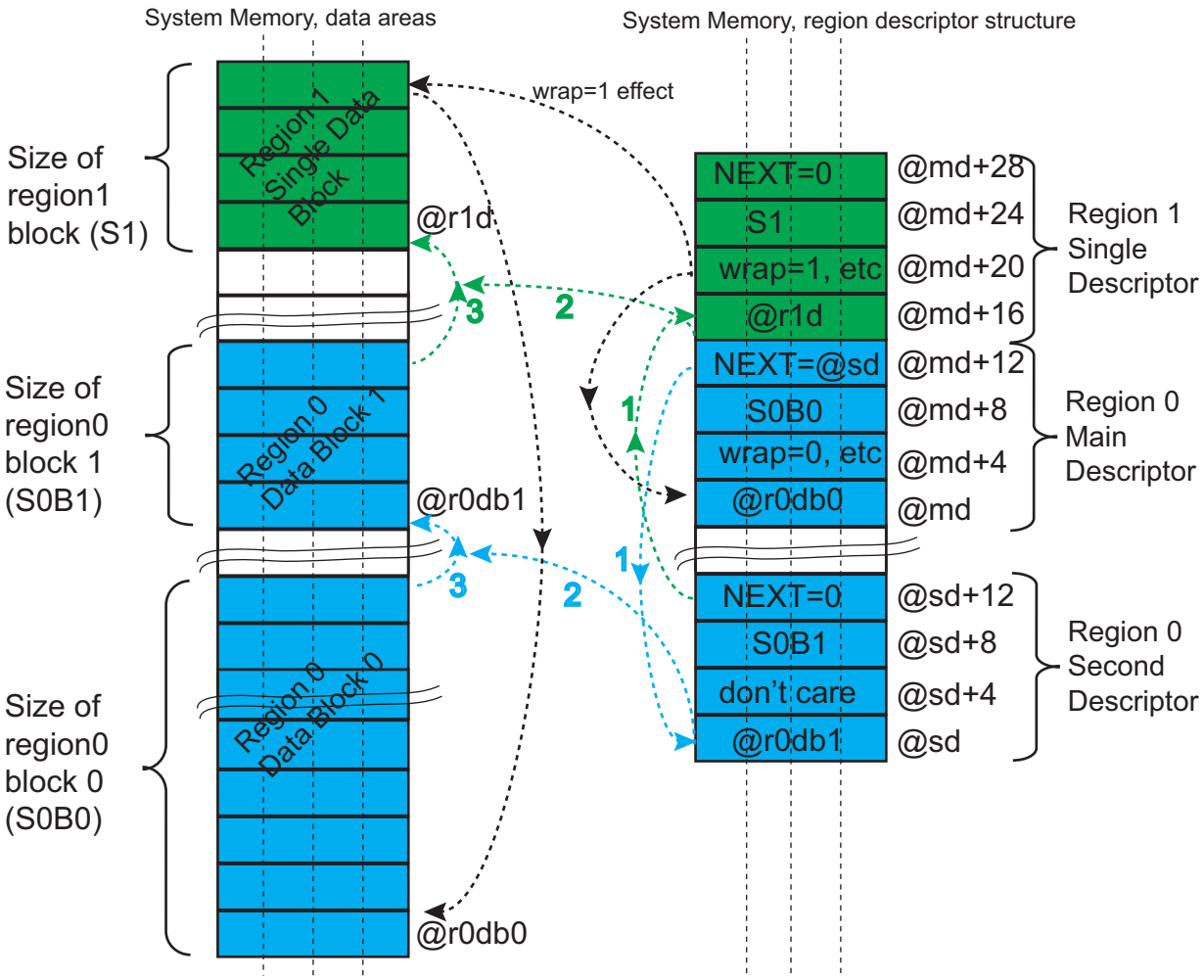


Figure 53-5 shows an example of the mandatory ICM settings to monitor three memory data blocks of the system memory (defined as two regions) with one region being not contiguous (two separate areas) and one contiguous memory area. For each said region, the SHA algorithm may be independently selected (different for each region). The wrap allows continuous monitoring.

Figure 53-5. Example: Monitoring of 3 Memory Data Blocks (Defined as 2 Regions)



53.5.2 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 controller 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 53-2. 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

53.5.2.1 ICM Region Start Address Structure Member

Name: ICM_RADDR

Address: ICM_DSCR+0x000+RID*(0x10)

Access: Read/Write

31	30	29	28	27	26	25	24
RADDR							

23	22	21	20	19	18	17	16
RADDR							

15	14	13	12	11	10	9	8
RADDR							

7	6	5	4	3	2	1	0
RADDR							

- **RADDR: Region Start Address**

This field indicates the first byte address of the region.

53.5.2.2 ICM Region Configuration Structure Member

Name: ICM_RCFG

Address: ICM_DSCR+0x004+RID*(0x10)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	ALGO			–	PROCDLY	SUIEN	ECIEN
7	6	5	4	3	2	1	0
WCIEN	BEIEN	DMIEN	RHIEN	–	EOM	WRAP	CDWBN

- **CDWBN: Compare Digest or Write Back Digest**

0: The digest is written to the Hash area.

1: The digest value is compared to the digest stored in the Hash area.

- **WRAP: Wrap Command**

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.

- **EOM: End Of Monitoring**

0: The current descriptor does not terminate the monitoring.

1: The current descriptor terminates the Main List. WRAP bit value has no effect.

- **RHIEN: Region Hash Completed Interrupt Disable (Default Enabled)**

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.

- **DMIEN: Digest Mismatch Interrupt Disable (Default Enabled)**

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.

- **BEIEN: Bus Error Interrupt Disable (Default Enabled)**

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.

- **WCIEN: Wrap Condition Interrupt Disable (Default Enabled)**

0: The ICM_ISR RWC[*i*] flag is set when the WRAP bit is set in a descriptor of the main list.

1: The ICM_ISR RWC[*i*] flag remains cleared even if the setting condition is met.

- **ECIEN: End Bit Condition Interrupt (Default Enabled)**

0: The ICM_ISR REC[*i*] flag is set when the descriptor having the EOM bit set is processed.

1: The ICM_ISR REC[*i*] flag remains cleared even if the setting condition is met.

- **SUIEN: Monitoring Status Updated Condition Interrupt (Default Enabled)**

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.

- **PROCDLY: Processing Delay**

Value	Name	Description
0	SHORTEST	SHA processing runtime is the shortest one
1	LONGEST	SHA processing runtime is the longest one

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.

- **ALGO: SHA Algorithm**

Value	Name	Description
0	SHA1	SHA1 algorithm processed
1	SHA256	SHA256 algorithm processed
4	SHA224	SHA224 algorithm processed

Values which are not listed in the table must be considered as “reserved”.

53.5.2.3 ICM Region Control Structure Member

Name: ICM_RCTRL

Address: ICM_DSCR+0x008+RID*(0x10)

Access: Read/Write

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

- **TRSIZE: Transfer Size for the Current Chunk of Data**

ICM performs a transfer of (TRSIZE + 1) blocks of 512 bits.

53.5.2.4 ICM Region Next Address Structure Member

Name: ICM_RNEXT

Address: ICM_DSCR+0x00C+RID*(0x10)

Access: Read/Write

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

- **NEXT: 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.

53.5.3 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 controller 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 controller performs a digest read operation at the same address.

53.5.3.1 Message Digest Example

Considering the following **512 bits message** (example given in FIPS 180-2):

“61626380000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000018”

The message is written to memory in a Little Endian (LE) system architecture.

Table 53-3. 512 bits Message Memory Mapping

	Address Offset / Byte Lane			
Memory Address	**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 53-4. LE Resulting SHA-160 Message Digest Memory Mapping

	Address Offset / Byte Lane			
Memory Address	**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 53-5. Resulting SHA-224 Message Digest Memory Mapping

	Address Offset / Byte Lane			
Memory Address	**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 53-6. 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 bits message** (example given in FIPS 180-2):

"6162638000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000018"

The message is written to memory in a Little Endian (LE) system architecture.

Table 53-7. 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

53.5.4 Using ICM as SHA Engine

The ICM can be configured to only calculate a SHA1, SHA224, SHA256 digest value.

53.5.4.1 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 word 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 the EOM bit of ICM_RCFG 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 the field NEXT of the ICM_RNEXT 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, the bits CDWBN and WRAP must be cleared in the region descriptor structure member ICM_RCFG. The bits WBDIS, EOMDIS, SLBDIS must be cleared in ICM_CFG.

The bits RHIEN or ECIEN must be written to 1 in the region descriptor structure member ICM_RCTRL. 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*).

53.5.4.2 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.

53.5.5 ICM Automatic Monitoring Mode

The ASCD bit of the ICM_CFG register is used to activate the ICM Automatic mode. When ICM_CFG.ASCD is set, the ICM performs the following actions:

- The ICM controller passes through the Main List once with CDWBN bit in the context register at 0 (i.e., Write Back activated) and EOM bit in context register at 0.
- When WRAP = 1 in ICM_RCFG, the ICM controller enters active monitoring with CDWBN bit in context register now set and EOM bit in context register cleared. Bits CDWBN and EOM in ICM_RCFG have no effect.

53.5.6 Programming the ICM for Multiple Regions

Table 53-8. 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.

Table 53-8. Region Attributes (Continued)

Transfer Type		Main List	ICM_RCFG			ICM_RNEXT	Comments
			CDWBN	WRAP	EOM	NEXT	
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.

53.5.7 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.

53.6 Integrity Check Monitor (ICM) User Interface

Table 53-9. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Configuration Register	ICM_CFG	Read/Write	0x0
0x04	Control Register	ICM_CTRL	Write-only	–
0x08	Status Register	ICM_SR	Read-only	–
0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	ICM_IER	Write-only	–
0x14	Interrupt Disable Register	ICM_IDR	Write-only	–
0x18	Interrupt Mask Register	ICM_IMR	Read-only	0x0
0x1C	Interrupt Status Register	ICM_ISR	Read-only	0x0
0x20	Undefined Access Status Register	ICM_UASR	Read-only	0x0
0x24–0x2C	Reserved	–	–	–
0x30	Region Descriptor Area Start Address Register	ICM_DSCR	Read/Write	0x0
0x34	Region Hash Area Start Address Register	ICM_HASH	Read/Write	0x0
0x38	User Initial Hash Value 0 Register	ICM_UIHVAL0	Write-only	–
...	...	...	...	...
0x54	User Initial Hash Value 7	ICM_UIHVAL7	Write-only	–
0x78–0xE8	Reserved	–	–	–
0xEC–0xFC	Reserved	–	–	–

53.6.1 ICM Configuration Register

Name: ICM_CFG

Address: 0xFC040000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
UALGO			UIHASH	–	–	DUALBUFF	ASCD
7	6	5	4	3	2	1	0
BBC				–	SLBDIS	EOMDIS	WBDIS

- **WBDIS: Write Back Disable**

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. The CDWBN bit of the ICM_RCFG structure member has no effect.

When ASCD bit of the ICM_CFG register is set, WBDIS bit value has no effect.

- **EOMDIS: End of Monitoring Disable**

0: End of Monitoring is permitted

1: End of Monitoring is forbidden. The EOM bit of the ICM_RCFG structure member has no effect.

- **SLBDIS: Secondary List Branching Disable**

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.

- **BBC: 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.

- **ASCD: Automatic Switch To Compare Digest**

0: Automatic mode is disabled.

1: When this mode is enabled, the ICM controller automatically switches to active monitoring after the first Main List pass. Both CDWBN and WBDIS bits have no effect. A one must be written to the EOM bit in ICM_RCFG to terminate the monitoring.

- **DUALBUFF: Dual Input Buffer**

0: Dual Input Buffer mode is disabled.

1: Dual Input Buffer mode is enabled (better performances, higher bandwidth required on system bus).

- **UIHASH: User Initial Hash Value**

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.

- **UALGO: User SHA Algorithm**

Value	Name	Description
0	SHA1	SHA1 algorithm processed
1	SHA256	SHA256 algorithm processed
4	SHA224	SHA224 algorithm processed

53.6.2 ICM Control Register

Name: ICM_CTRL

Address: 0xFC040004

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RMEN				RMDIS			
7	6	5	4	3	2	1	0
REHASH				–	SWRST	DISABLE	ENABLE

- **ENABLE: ICM Enable**

0: No effect

1: When set to one, the ICM controller is activated.

- **DISABLE: ICM Disable Register**

0: No effect

1: The ICM controller is disabled. If a region is active, this region is terminated.

- **SWRST: Software Reset**

0: No effect

1: Resets the ICM controller.

- **REHASH: Recompute Internal Hash**

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.

- **RMDIS: Region Monitoring Disable**

0: No effect

1: When bit RMDIS[*i*] is set to one, the monitoring of region with identifier *i* is disabled.

- **RMEN: Region Monitoring Enable**

0: No effect

1: When bit RMEN[*i*] is set to one, the monitoring of region with identifier *i* is activated.

Monitoring is activated by default.

53.6.3 ICM Status Register

Name: ICM_SR

Address: 0xFC040008

Access: Read-only

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

- **ENABLE: ICM Controller Enable Register**

0: ICM controller is disabled

1: ICM controller is activated

- **RAWRMDIS: Region Monitoring Disabled Raw Status**

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

- **RMDIS: Region Monitoring Disabled Status**

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

53.6.4 ICM Interrupt Enable Register

Name: ICM_IER

Address: 0xFC040010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	URAD
23	22	21	20	19	18	17	16
RSU				REC			
15	14	13	12	11	10	9	8
RWC				RBE			
7	6	5	4	3	2	1	0
RDM				RHC			

- **RHC: Region Hash Completed Interrupt Enable**

0: No effect

1: When RHC[*i*] is set to one, the Region *i* Hash Completed interrupt is enabled.

- **RDM: Region Digest Mismatch Interrupt Enable**

0: No effect

1: When RDM[*i*] is set to one, the Region *i* Digest Mismatch interrupt is enabled.

- **RBE: Region Bus Error Interrupt Enable**

0: No effect

1: When RBE[*i*] is set to one, the Region *i* Bus Error interrupt is enabled.

- **RWC: Region Wrap Condition detected Interrupt Enable**

0: No effect

1: When RWC[*i*] is set to one, the Region *i* Wrap Condition interrupt is enabled.

- **REC: Region End bit Condition Detected Interrupt Enable**

0: No effect

1: When REC[*i*] is set to one, the region *i* End bit Condition interrupt is enabled.

- **RSU: Region Status Updated Interrupt Disable**

0: No effect

1: When RSU[*i*] is set to one, the region *i* Status Updated interrupt is enabled.

- **URAD: Undefined Register Access Detection Interrupt Enable**

0: No effect

1: The Undefined Register Access interrupt is enabled.

53.6.5 ICM Interrupt Disable Register

Name: ICM_IDR

Address: 0xFC040014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	URAD
23	22	21	20	19	18	17	16
RSU				REC			
15	14	13	12	11	10	9	8
RWC				RBE			
7	6	5	4	3	2	1	0
RDM				RHC			

- **RHC: Region Hash Completed Interrupt Disable**

0: No effect

1: When RHC[*i*] is set to one, the Region *i* Hash Completed interrupt is disabled.

- **RDM: Region Digest Mismatch Interrupt Disable**

0: No effect

1: When RDM[*i*] is set to one, the Region *i* Digest Mismatch interrupt is disabled.

- **RBE: Region Bus Error Interrupt Disable**

0: No effect

1: When RBE[*i*] is set to one, the Region *i* Bus Error interrupt is disabled.

- **RWC: Region Wrap Condition Detected Interrupt Disable**

0: No effect

1: When RWC[*i*] is set to one, the Region *i* Wrap Condition interrupt is disabled.

- **REC: Region End bit Condition detected Interrupt Disable**

0: No effect

1: When REC[*i*] is set to one, the region *i* End bit Condition interrupt is disabled.

- **RSU: Region Status Updated Interrupt Disable**

0: No effect

1: When RSU[*i*] is set to one, the region *i* Status Updated interrupt is disabled.

- **URAD: Undefined Register Access Detection Interrupt Disable**

0: No effect

1: Undefined Register Access Detection interrupt is disabled.

53.6.6 ICM Interrupt Mask Register

Name: ICM_IMR

Address: 0xFC040018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	URAD

23	22	21	20	19	18	17	16
RSU				REC			

15	14	13	12	11	10	9	8
RWC				RBE			

7	6	5	4	3	2	1	0
RDM				RHC			

- **RHC: Region Hash Completed Interrupt Mask**

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.

- **RDM: Region Digest Mismatch Interrupt Mask**

0: When RDM[*i*] is set to zero, the interrupt is disabled for region i.

1: When RDM[*i*] is set to one, the interrupt is enabled for region i.

- **RBE: Region Bus Error Interrupt Mask**

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.

- **RWC: Region Wrap Condition Detected Interrupt Mask**

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.

- **REC: Region End bit Condition Detected Interrupt Mask**

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.

- **RSU: Region Status Updated Interrupt Mask**

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.

- **URAD: Undefined Register Access Detection Interrupt Mask**

0: Interrupt is disabled

1: Interrupt is enabled.

53.6.7 ICM Interrupt Status Register

Name: ICM_ISR

Address: 0xFC04001C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	URAD

23	22	21	20	19	18	17	16
RSU				REC			

15	14	13	12	11	10	9	8
RWC				RBE			

7	6	5	4	3	2	1	0
RDM				RHC			

- **RHC: Region Hash Completed**

When RHC[*i*] is set, it indicates that the ICM has completed the region with identifier *i*.

- **RDM: 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.

- **RBE: Region Bus Error**

When RBE[*i*] is set, it indicates that a bus error has been detected while hashing memory region *i*.

- **RWC: Region Wrap Condition Detected**

When RWC[*i*] is set, it indicates that a wrap condition has been detected.

- **REC: Region End bit Condition Detected**

When REC[*i*] is set, it indicates that an end bit condition has been detected.

- **RSU: Region Status Updated Detected**

When RSU[*i*] is set, it indicates that a region status updated condition has been detected.

- **URAD: Undefined Register Access Detection Status**

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.

The URAD bit is only reset by the SWRST bit in the ICM_CTRL register.

The URAT field in the ICM_UASR indicates the unspecified access type.

53.6.8 ICM Undefined Access Status Register

Name: ICM_UASR

Address: 0xFC040020

Access: Read-only

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
–	–	–	–	–	URAT		

- **URAT: Undefined Register Access Trace**

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

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.

53.6.9 ICM Descriptor Area Start Address Register

Name: ICM_DSCR

Address: 0xFC040030

Access: Read/Write

31	30	29	28	27	26	25	24
DASA							

23	22	21	20	19	18	17	16
DASA							

15	14	13	12	11	10	9	8
DASA							

7	6	5	4	3	2	1	0
DASA		–	–	–	–	–	–

- **DASA: Descriptor Area Start Address**

The start address is a multiple of the total size of the data structure (64 bytes).

53.6.10 ICM Hash Area Start Address Register

Name: ICM_HASH

Address: 0xFC040034

Access: Read/Write

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

- **HASA: Hash Area Start Address**

This field points at the Hash memory location. The address must be a multiple of 128 bytes.

53.6.11 ICM User Initial Hash Value Register

Name: ICM_UIHVALx [x=0..7]

Address: 0xFC040038

Access: Write-only

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

- **VAL: Initial Hash Value**

When UIHASH bit of IMC_CFG register 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

54. Classical Public Key Cryptography Controller (CPKCC)

54.1 Description

The Classical Public Key Cryptography Controller (CPKCC) is an Atmel macrocell that processes public key cryptography algorithm calculus in both GF(p) and GF(2^n) fields. The ROMed CPKCL, the Classical Public Key Cryptography Library, is the library built on the top of the CPKCC.

The Classical Public Key Cryptography Library includes complete implementation of the following public key cryptography algorithms:

- RSA (Rivest-Shamir-Adleman public key cryptosystem), DSA (Digital Signature Algorithm):
  - Modular Exponentiation with CRT up to 7168 bits
  - Modular Exponentiation without CRT up to 5376 bits
  - Prime generation
  - Utilities: GCD/modular Inverse, Divide, Modular reduction, Multiply, …
- Elliptic Curves:
  - ECDSA GF(p) up to 521 bits
  - ECDSA GF(2^n) up to 571 bits
  - Point Multiply
  - Point Add/Doubling
  - Choice of the curves parameters so compatibility with NIST Curves or others
- Deterministic Random Number Generation (DRNG ANSI X9.31) for DSA

54.2 Product Dependencies

54.2.1 Power Management

The CPKCC is not continuously clocked. The CPCKCC interface is clocked through the Power Management Controller (PMC).

54.2.2 Interrupt Sources

The CPKCC has an interrupt line connected to the interrupt controller. Handling interrupts requires programming the interrupt controller before configuring the CPKCC.

54.3 Functional Description

The CPKCC macrocell is managed by the CPKCL available in the ROM memory of the SAMA5D4. The user interface of the CPKCC is not described in this section.

The usage description of the CPKCC and its associated Library is provided in the application note “Using CPKCL Version 02.05.01.xx on SAMA5D4”, Atmel literature No. 11214. This document is available under Non-Disclosure Agreement (NDA). Contact an Atmel Sales Representative for further details.

55. Electrical Characteristics

55.1 Absolute Maximum Ratings

Table 55-1. Absolute Maximum Ratings*

Storage Temperature.....-60°C to +150°C
Voltage on Input Pins with Respect to Ground.....-0.3V to VDDIO+0.3V(+ 4V max)
Maximum Operating Voltage (VDDPLLA and VDDUTMIC).....1.5V
(VDDCORE and VDDIODDR).....2.0V
(VDDBU).....2.64V
(VDDIOM, VDDIOP, VDDUTMII, VDDOSC, and VDDANA).....4.0V
Total DC Output Current on all I/O lines.....350 mA

*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

55.2 DC Characteristics

The following characteristics are applicable to the operating temperature range: T_A = -40°C to +85°C, unless otherwise specified.

Table 55-2. DC Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
T_A	Operation Temperature	—	-40	—	+85	°C
T_J	Junction Temperature(1)	—	-40	—	+125	°C
V_{DDCORE}	DC Supply Core Regulator	Must be established after V_{DDIOP} or at the same time	1.62	1.8	1.98	V
V_{CCCORE}	DC Supply Core	—	1.16	1.26	1.32	V
V_{DDPLLA}	DC Supply PLLA	Connected to VCCCORE	1.1	1.2	1.32	V
$V_{DDUTMIC}$	DC Supply UDPHS and UHPHS UTMI+ Core	Connected to VCCCORE	1.1	1.2	1.32	V
$V_{DDUTMII}$	DC Supply UDPHS and UHPHS UTMI+ Interface	—	3.0	3.3	3.6	V
V_{DDBU}	DC Supply Backup	Must be established first	1.8	2.0	2.6	V
V_{DDOSC}	DC Supply Oscillator	—	3.0	—	3.6	V
V_{DDIOM}	DC Supply EBI I/Os	—	1.65/3.0	1.8/3.3	1.95/3.6	V
$V_{DDIODDR}$	DC Supply SDRAM I/Os	DDR2 or LP-DDR usage LP-DDR2 usage	1.7 1.14	1.8 1.2	1.95 1.30	V
V_{DDIOP}	DC Supply Peripheral I/Os	—	3.0	—	3.6	V
V_{DDANA}	DC Supply Analog	—	3.0	3.3	3.6	V
V_{DDFUSE}	DC Supply Fuse Box	For fuse programming only; must not be left floating	2.25	2.5	2.75	V
V_{IL}	Input Low-level Voltage	V_{DDIO} in 3.3V range	-0.3	—	0.8	V
		V_{DDIO} in 1.8V range	-0.3	—	$0.3 \times V_{DDIO}$	

Table 55-2. DC Characteristics (Continued)

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IH}	Input High-level Voltage	V_{DDIO} in 3.3V range		2	—	V_{DDIO} + 0.3	V
		V_{DDIO} in 1.8V range		0.7 × V_{DDIO}	—	V_{DDIO} + 0.3	
V_{hys}	Schmitt Trigger Hysteresis	All PIO lines V_{DDIOx} in 3.3V range		0.34	—	—	V
		All PIO lines V_{DDIOx} in 1.8V range		0.21	—	—	
V_{OL}	Output Low-level Voltage	I_O Max		—	—	0.4	V
V_{OH}	Output High-level Voltage	I_O Max		V_{DDIO} - 0.4	—	—	V
R_{PULL}	Pullup Resistance Pulldown Resistance	PC0–PC1, PC3, PC5–PC26, PE0–PE28, EBI signals V_{DDIOM} in 1.8V range		60	100	180	kΩ
		PA2–PA3, PA24, PA28, PB0–PB1, PB10, PB20, PB26, PC2, PC4, PD8, PD28–PD29, PD31, PE0–PE28, EBI signals V_{DDIOx} in 3.3V range		50	70	140	
		PA0–PA1, PA4–PA23, PA25–PA27, PA29–PA31, PB2–PB9, PB11–PB19, PB21–PB25, PB27–PB31, PC0–PC1, PC3, PC5–PC31, PD9–PD17, PD30, PE31, PIOBUx, NTRST and NRST (3.3V range only)		50	70	140	
		PD18–PD27, PE29, PE30 (3.3V range only)		200	400	600	
I_O	Output Current	PC0–PC1, PC3, PC5–PC26, PE0–PE28 V_{DDIOM} in 1.8V range		1 (LO_DRIVE)	8 (ME_DRIVE)	10 (HI_DRIVE)	mA
		PA2–PA3, PA24, PA28, PB0–PB1, PB10, PB20, PB26, PC2, PC4, PD8, PD28–PD29, PD31, PE0–PE28 V_{DDIOx} in 3.3V range		1 (LO_DRIVE)	8 (ME_DRIVE)	10 (HI_DRIVE)	
		PA0–PA1, PA4–PA23, PA25–PA27, PA29–PA31, PB2–PB9, PB11–PB19, PB21–PB25, PB27–PB31, PC0–PC1, PC3, PC5–PC31, PD9–PD17, PD30, PE31 (3.3V range only)		1 (LO_DRIVE)	6 (ME_DRIVE)	7 (HI_DRIVE)	
		PD18–PD27, PE29, PE30 (3.3V range only)		20 (LO_DRIVE)	—	34 (HI_DRIVE)	
R_{SERIAL}	Serial Resistor	PA0–PA1, PA4–PA23, PA25–PA27, PA29–PA31, PB2–PB9, PB11–PB19, PB21–PB25, PB27–PB31, PC0–PC1, PC3, PC5–PC26, PD9–PD17, PD30, PE31, PIOBUx, TST, SHDN, NRST, JTAGSEL, WKUP		—	30	—	Ω
		PA2–PA3, PA24, PA28, PB0–PB1, PB10, PB20, PB26, PC2, PC4, PD8, PD28–PD29, PD31, PE0–PE28		—	13	—	
		PC27–PC31, PD18–PD27, PE29, PE30		—	0	—	
I_{SC}	Static Current	On V_{DDCORE} = 1.8V, MCK = 0 Hz, excluding POR	T_A = 25°C	—	6	—	mA
		All inputs driven TMS, TDI, TCK, NRST = 1	T_A = 85°C	—	—	37	
		On V_{DDBU} = 2.0V, Logic cells consumption, excluding POR	T_A = 25°C	—	7	10	µA
		All inputs driven WKUP = 0	T_A = 85°C	—	—	18	

Note: 1. Junction temperature depends on ambient temperature, on-chip power dissipation, package thermal resistance, board thermal resistance and power dissipation of other components on the board.

55.3 Power Consumption

- Typical power consumption of PLLs, Slow Clock and Main Oscillator
- Power consumption of power supply in four different modes: Active, Idle, Ultra-low-power and Backup
- Power consumption by peripheral: calculated as the difference in current measurement after having enabled then disabled the corresponding clock
- Software used for power consumption measurements: Dhrystone / Coremark

55.3.1 Active Mode

Active Mode is the normal running mode with the core clock running off a PLL. The power management controller can be used to adapt the frequency and to disable the peripheral clocks.

Table 55-5 represents the power consumption estimated on the power supplies.

55.3.2 Low-power Modes

55.3.2.1 Backup Mode

The purpose of Backup Mode is to achieve the lowest power consumption possible in a system which is performing periodic wakeups to perform tasks but not requiring fast startup time.

The Zero-power Power-on Reset, RTC, Backup registers and 32 kHz oscillator (RC or Crystal Oscillator selected by software in the Supply Controller) are running. The core supply is off.

The system can be awakened from this mode through the WKUP0 pin or an RTC wakeup event.

Backup mode is entered with the help of the Shutdown Controller that asserts the SHDN output pin. The SHDN pin is to be connected to the Enable of the VDDCORE regulator.

Exit from Backup mode happens if one of the following enable wakeup events occurs:

- WKUP0 pin (level transition, configurable debouncing)
- RTC alarm

The system will restart as for a reset event.

55.3.2.2 Idle Mode

The purpose of Idle Mode is to optimize power consumption of the device versus response time. In this mode, only the core clock is stopped. The peripheral clocks, including the DDR Controller clock, can be enabled. The current consumption in this mode is application dependent.

Idle mode is entered using the WFI (Wait for Interrupt) instruction and disabling the processor clock (PCK) in the PMC_SCDR.

The processor can be awakened from an interrupt. The system will resume where it was before entering in WFI mode.

Table 55-6 represents the power consumption estimated on the power supplies.

55.3.2.3 Ultra-low-power Mode

The purpose of the Ultra-low-power mode is to reduce the power consumption of the device to the minimum without disconnecting VDDCORE power supply. It is a combination of very low frequency operations and Idle Mode.

The procedure to enter Ultra-low-power mode is the following:

1. Set the DDR in Self-refresh mode.
2. Reduce the system clock (PCK and MCK) to the minimum with the help of the PMC:
   - PCK and MCK configuration is to be defined regarding the expected power consumption and wakeup time. Refer to Table 55-7 for details.

- PLLs are disabled. CKGR_PLLAR is set to 0x3F00. CKGR_UCKR is set to 0.
- The Main Oscillator is disabled. MOSCXTEN is set to 0 in CKGR_MOR.

3. Enter in Wait for Interrupt (WFI) mode and disable the PCK clock.

The processor can be woken up from an interrupt.

Once revived, the system must reprogram the system clocks (OSC, PLL, PCK, MCK, DDRCK) to recover the previous state. Data is maintained in the external memory.

55.3.2.4 Low-power Mode Summary Table

Specific device functionalities and components can be set to ‘On’ or ‘Off’ separately and wakeup sources can be individually configured. Table 55-3 summarizes the configurations of the low-power modes.

Table 55-3. Low-power Mode Configuration Summary

Component or Parameter	Low-power Mode		
	Backup	**Ultra-low-power**	**Idle**
64 kHz RC Oscillator, 32 kHz Oscillator, RTC, POR, SHDWC, SECURAM (Backup Area)	ON	ON	ON
12 MHz RC Oscillator	OFF	ON	ON
VDDCORE Regulator	OFF	ON	ON
Core	OFF (Not powered)	Powered (Not clocked)	Powered (Not clocked)
Memory, Peripherals	OFF (Not powered)	Powered (clock under user control; clock ON/OFF selectable on peripheral basis)	Powered (clock under user control; clock ON/OFF selectable on peripheral basis)
Mode Entry	Shutdown Controller	DDR in Self-refresh PMC WFI	WFI
Potential Wakeup Sources	WKUP0 pin RTC alarm	Any interrupt	Any interrupt
Core at Wakeup	Reset	Clocked back to previous speed	Clocked back to full speed
PIO State While in Low-power Mode	Reset	Previous state saved	Previous state saved
PIO State at Wakeup	Inputs with pullups	Unchanged	Unchanged
Consumption (2)	7 µA typical (3)	Refer to Table 55-7	Refer to Table 55-6 10 mA for each PLL (4)
Wakeup Time (1)	Startup time	Refer to Table 55-7	470 ns @ 176 MHz

Notes:
1. When considering wakeup time, the time required to start the PLL is not taken into account. Once started, the device works with the Main Oscillator. The user has to add the PLL startup time if it is needed in the system. The wakeup time is defined as the time taken for wakeup until the first instruction is fetched.
2. The external loads on PIOs are not taken into account in the calculation.
3. Total current consumption
4. Depends on MCK frequency

55.3.3 Power Consumption Versus Modes

The values are measured values of the power consumption under the following operating conditions:

- Parts are from typical process
- V_{DDIOM} = 1.8 V
- V_{DDIOP} = 3.3 V
- V_{DDCORE} = 1.8 V unless specified otherwise in the table
- V_{DDBU} = 2.0 V
- T_A 25°C = unless specified otherwise in the table
- There is no consumption on the I/Os of the device
- All the peripheral clocks are disabled

Figure 55-1. Measures Schematics

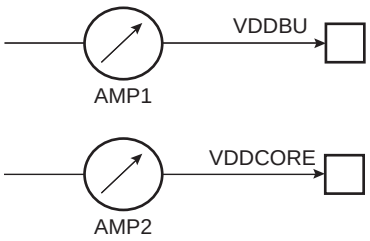


Table 55-4. Power Consumption by Peripheral in Active Mode

Peripheral	Conditions	Consumption [1]	Unit	Maximum Frequency
ADC	—	3.2	µA/MHz	MCK/2
AES	—	2.3	µA/MHz	MCK/2
AESB	—	0.2	µA/MHz	MCK
DBGU	—	1.3	µA/MHz	MCK/2
GMAC	Running ipef with UDP and bi-directional data transfer	16.5 × MCK + 734 × DR(Mbps) + 611 [2]	µA	MCK/2
HSMCI	—	8.5	µA/MHz	MCK/2
ICM	—	2.9	µA/MHz	MCK/2
ISI	Preview path enabled Codec path disabled Data from sensor in YUV422 format Color conversion from YCC to RGB565 enabled 2-D image scalar disabled	3.13 × MCK + 19.2 × Pix_CLK + 4.37 × DR + 108 [3]	µA	MCK
LCDC	Base Layer Only 24-bpp RGB888 color space Global alpha blending enabled Displayed data is random data	17.2 × MCK + 30.9 × Pix_CLK + 36.0 × DR_Base [4]	µA	MCK
	Overlay Layer Delta (for each layer) 24-bpp RGB888 color space Auxiliary functions such as rotation, scaling, color space conversion, color look-up table, and chroma up-sampling all disabled Global alpha blending enabled Displayed data is random data	3.70 × MCK + 11.1 × Pix_CLK + 26.5 × DR_Overlay [4]		

Table 55-4. Power Consumption by Peripheral in Active Mode (Continued)

Peripheral	Conditions	Consumption [1]	Unit	Maximum Frequency
MPDDRC	MPDDRC enabled and transferring data between XDMAC Power consumption of XDMAC is subtracted Transferred data is random data	216 × MCK + 4.08 × DR + 3160	µA	MCK
	MPDDRC enabled but no data transfer	194 × MCK + 3160		
PIO Controller	—	1.8	µA/MHz	MCK
CPKCC	—	3.0	µA/MHz	MCK
PWM	—	7.0	µA/MHz	MCK/2
SHA	—	4.4	µA/MHz	MCK/2
SMD	—	6.6	µA/MHz	MCK/2
SPI	—	2.2	µA/MHz	MCK/2
SSC	—	2.4	µA/MHz	MCK/2
TDES	—	1.7	µA/MHz	MCK/2
TC	—	3.9	µA/MHz	MCK/2
TRNG	—	890	µA	MCK/2
TWI	—	2.8	µA/MHz	MCK/2
UART	—	1.5	µA/MHz	MCK/2
UDPHS	Transferred file contains random data	0.012 × MCK + 0.86 × DR + 3.2 [5]	mA	MCK/2
UHPHS	Transferred file contains random data	0.014 × MCK + 1.85 × DR + 5.59 [5]	mA	MCK/2
USART	—	4.1	µA/MHz	MCK/2
VDEC	Decoding avi file without audio, 720p, 480p, H264, 30fps	100	mA	MCK
XDMAC	XDMAC enabled and transferring data	98.7 × MCK + 5.58 × DR	µA	MCK
	XDMAC enabled but no data transfer	10.1 × MCK		

Notes:
1. MCK: MHz
   DR (Theoretical Data Rate): MB/s
   DR_Base (Data Rate of Base layer): MB/s
   DR_Overlayer (Data Rate of Overlay layers): MB/s
   Pix_CLK (Pixel Clock): MHz
   For XDMAC, MPDDRC, UHPHS, and UDPHS, 1 MB/s = 1024*1024 = 1,048,576 bytes/s
   For LCDC and ISI, 1 MB/s = 1,000,000 bytes/s
   For GMAC: 1 Mbps = 1,000,000 bits/s
2. The DR is the average value of Tx and Rx speed.
3. Data from the sensor is in YUV422 format which means each pixel contains two bytes, so the Pix_CLK provided by the image sensor is twice the pixel rate (two Pix_CLKs transfer one pixel).
4. Power consumption of LCDC is the sum of the power consumption of each layer.
5. The UPLL and Transceiver consume 8 mA (measured from VDDCore), which must be taken into account once any UHPHS or UDPHS is enabled.

Table 55-5. Power Consumption in Active Mode

Conditions		Consumption		Unit
		T_A 25°C	T_A 85°C	
PLL clock = 1200 MHz ARM core clock = 600 MHz MCK = 200 MHz	Caches L1 and MMU enabled and Cache L2 disabled Code running out of internal SRAM Code speed optimization Run Dhrystone / CoreMark benchmark Peripheral clock disabled	182	213	mA

Table 55-6. Power Consumption in Idle Mode

Conditions		Consumption		Unit
		T_A 25°C	T_A 85°C	
PLL clock = 1200 MHz ARM core clock = 600 MHz MCK = 200 MHz	Core clock is stopped Peripheral clocks, including the DDR Controller clock, are disabled	37.0	57.3	mA
PLL clock = 792 MHz ARM core clock = 396 MHz MCK = 132 MHz		26.7	52.3	
PLL clock = 600 MHz ARM core clock = 300 MHz MCK = 100 MHz		22.3	47.8	

Table 55-7. VDDCORE Power Consumption in Ultra-low-Power Mode (AMP2)

Mode	Conditions	Consumption (mA)		Wakeup Time (µs)
		T_A 25°C	T_A 85°C	
ULP 12 MHz	ARM core clock disabled MCK = 12 MHz	6.3	31	9.2
ULP 750 kHz	ARM core clock disabled MCK = 750 kHz	4.9	29.4	151
ULP 187 kHz	ARM core clock disabled MCK = 187.5 kHz	4.8	29.3	592
ULP 32 kHz	ARM core clock disabled MCK = 32 kHz	4.7	29.2	3400

55.4 Clock Characteristics

55.4.1 Processor Clock Characteristics

Table 55-8. Processor Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
$1/(t_{CPPCK})$	Processor Clock Frequency	VCCCORE[1.16V, 1.39V], T_A = 85°C	250(1)	600	MHz

Note: 1. Limitation for DDR2 usage only, there are no limitations to LP-DDR and LP-DDR2.

55.4.2 Master Clock Characteristics

The master clock is the maximum clock at which the system is able to run. It is given by the smallest value of the internal bus clock and EBI clock.

Table 55-9. Master Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
$1/(t_{CPMCK})$	Master Clock Frequency	VCCCORE[1.16V, 1.39V], T_A = 85°C	125(1)	200	MHz

Note: 1. Limitation for DDR2 usage only, there are no limitations to LP-DDR, LP-DDR2.

55.5 Crystal Oscillator Characteristics

Table 55-10. Main Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OP}	Operating Frequency	—	—	12	—	MHz
C_{LOAD}(1)	Load Capacitance	—	12.5	—	17.5	pF
C_{LEXT}(1)	External Load Capacitance	—	—	—	—	pF
C_{PARA}	Parasitic Load Capacitance	—	0.6	0.7	0.8	pF
—	Duty Cycle	—	40	—	60	%
t_{START}	Startup Time	—	—	—	1	ms
P_{ON}	Drive Level	—	—	—	150	µW
$I_{DD\ ON}$	Current Dissipation	@ 12 MHz	—	0.5	1	mA

Note: 1. The external capacitors value can be determined by using the following formula:
$C_{LEXT} = (2 \times C_{LOAD}) - C_{BOARD} - C_{ROUTING} - C_{PACKAGE} - (C_{PARA} \times 2)$,
where
C_{LEXT}: external capacitor value which must be soldered from XIN to GND and XOUT to GND
C_{LOAD}: crystal targeted load. Refer to C_{LOAD} parameter in the electrical specification.
C_{BOARD}: external calculated (or measured) parasitic value due to board
$C_{ROUTING}$: parasitic capacitance due to internal chip routing, typically 1.5 pF
$C_{PACKAGE}$: parasitic capacitance due to package and bonding, typically 0.75 pF
C_{PARA}: internal parasitic load due to internal structure.

55.5.1 Crystal Characteristics

The following characteristics are applicable to the operating temperature range: T_A = -40°C to 85°C and worst case of power supply, unless otherwise specified.

Table 55-11. Crystal Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
ESR	Equivalent Series Resistance	@ 12 MHz	—	—	120	Ω
C_m	Motional Capacitance	—	5	—	9	fF
C_{SHUNT}	Shunt Capacitance	—	—	—	7	pF

55.5.2 XIN Clock Characteristics

Table 55-12. XIN Clock Electrical Characteristics

Symbol	Parameter	Conditions	Min	Max	Unit
$1/(t_{CPXIN})$	XIN Clock Frequency	—	—	50	MHz
t_{CPXIN}	XIN Clock Period	—	20	—	ns
t_{CHXIN}	XIN Clock High Half-period	—	$0.4 \times t_{CPXIN}$	$0.6 \times t_{CPXIN}$	ns
t_{CLXIN}	XIN Clock Low Half-period	—	$0.4 \times t_{CPXIN}$	$0.6 \times t_{CPXIN}$	ns
C_{IN}	XIN Input Capacitance	Main Oscillator in Bypass mode (i.e., when MOSCXTEN = 0 and MOSCXTBY = 1 in the CKGR_MOR). Refer to Section 26.19.8 “PMC Clock Generator Main Oscillator Register”.	—	25	pF
R_{IN}	XIN Pulldown Resistor		—	500	kΩ
V_{IN}	XIN Voltage		V_{DDOSC}	V_{DDOSC}	V

55.6 12 MHz RC Oscillator Characteristics

Table 55-13. 12 MHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
F0	Nominal Frequency	—	11.88	12	12.12	MHz
—	Duty Cycle	—	45	50	55	%
$I_{DD\ ON}$	Power Consumption Oscillation	—	—	100	—	µA

55.7 32 kHz Crystal Oscillator Characteristics

Table 55-14. 32 kHz Crystal Oscillator Characteristics

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
$1/(t_{CP32KHz})$	Crystal Oscillator Frequency	—		—	32.768	—	kHz
$C_{CRYSTAL32}$	Load Capacitance	Crystal @ 32.768 kHz		6		12.5	pF
C_{LEXT32}[2]	External Load Capacitance	$C_{CRYSTAL32}$ = 6 pF		—	6	—	pF
		$C_{CRYSTAL32}$ = 12.5 pF		—	19	—	pF
—	Duty Cycle	—		40	50	60	%
t_{START}	Startup Time	R_S = 50 kΩ[1]	$C_{CRYSTAL32}$ = 6 pF	—	—	400	ms
			$C_{CRYSTAL32}$ = 12.5 pF	—	—	900	ms
		R_S = 100 kΩ[1]	$C_{CRYSTAL32}$ = 6 pF	—	—	600	ms
			$C_{CRYSTAL32}$ = 12.5 pF	—	—	1200	ms
P_{ON}	Drive Level	—		—	—	0.2	µW

Notes: 1. R_S is the equivalent series resistance.
2. C_{LEXT32} is determined by taking into account internal, parasitic and package load capacitance.

55.7.1 32 kHz Crystal Characteristics

Table 55-15. 32 kHz Crystal Characteristics

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
ESR	Equivalent Series Resistor Rs	Crystal @ 32.768 kHz		—	50	100	kΩ
C_m	Motional Capacitance	Crystal @ 32.768 kHz		0.6	—	3	fF
C_{SHUNT}	Shunt Capacitance	Crystal @ 32.768 kHz		0.6	—	2	pF
$I_{DD\ ON}$	Current dissipation	R_S = 50 kΩ	$C_{CRYSTAL32}$ = 6 pF	—	0.3	0.65	µA
			$C_{CRYSTAL32}$ = 12.5 pF	—	0.45	0.7	µA
		R_S = 100 kΩ	$C_{CRYSTAL32}$ = 6 pF	—	0.45	0.75	µA
			$C_{CRYSTAL32}$ = 12.5 pF	—	0.45	0.65	µA

55.8 64 kHz RC Oscillator Characteristics

Table 55-16. 64 kHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$1/(t_{CPRCz})$	Oscillator Frequency	—	46	64	90	kHz
—	Duty Cycle[1]	—	—	50	—	%
$I_{DD\ ON}$	Power Consumption Oscillation	—	—	540	860	nA

Note: 1. The 32 kHz clock is created from a 64 kHz RC oscillator. Its frequency is divided by 2, so the duty cycle is 50/50.

55.9 PLL Characteristics

Table 55-17. PLLA Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output Frequency	—	600	—	1200	MHz
f_{IN}	Input Frequency	—	—	12	—	MHz
N	Multiplication Factor	—	45	—	112	—
I_{PLL}	Current Consumption	Active mode @ 1200 MHz	—	10.6	16.7	mA
		Standby mode	—	0.75	2	µA
t_{START}	Startup Time	—	—	30	—	µs

55.9.1 UTMI PLL Characteristics

Table 55-18. Phase Lock Loop Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{IN}	Input Frequency	—	—	12	—	MHz
f_{OUT}	Output Frequency	—	—	480	—	MHz
I_{PLL}	Current Consumption	Active mode	—	3.3	4	mA
		Standby mode	—	0.08	25	µA
t_{START}	Startup Time	—	—	—	50	µs

55.10 USB HS Characteristics

55.10.1 Electrical Characteristics

Table 55-19. Electrical Parameters

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
R_{PUI}	Bus Pullup Resistor on Upstream Port (idle bus)	In LS or FS Mode	—	1.5	—	kΩ
R_{PUA}	Bus Pullup Resistor on Upstream Port (upstream port receiving)	In LS or FS Mode	—	15	—	kΩ
t_{BIAS}	Bias settling time	—	—	—	20	µs
t_{OSC}	Oscillator settling time	With crystal 12 MHz	—	—	2	ms
$t_{SETTLING}$	Settling time	f_{IN} = 12 MHz	—	0.3	0.5	ms

55.10.2 Static Power Consumption

Table 55-20. Static Power Consumption

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
I_{BIAS}	Bias current consumption on VBG	—	—	—	1	µA
$I_{VDDUTMII}$	HS Transceiver and I/O current consumption	—	—	—	8	µA
	LS / FS Transceiver and I/O current consumption	No connection (1)	—	—	3	µA
$I_{VDDUTMIC}$	Core, PLL, and Oscillator current consumption	—	—	—	2	µA

Note: 1. If cable is connected, add 200 µA (Typical) due to pullup/pulldown current consumption.

55.10.3 Dynamic Power Consumption

Table 55-21. Dynamic Power Consumption

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
I_{BIAS}	Bias current consumption on VBG	—	—	0.7	0.8	mA
$I_{VDDUTMII}$	HS Transceiver current consumption	HS transmission	—	47	60	mA
	HS Transceiver current consumption	HS reception	—	18	27	mA
	LS / FS Transceiver current consumption	FS transmission 0m cable(1)	—	4	6	mA
	LS / FS Transceiver current consumption	FS transmission 5m cable(1)	—	26	30	mA
	LS / FS Transceiver current consumption	FS reception(1)	—	3	4.5	mA
$I_{VDDUTMIC}$	PLL, Core and Oscillator current consumption	—	—	5.5	9	mA

Note: 1. Including 1 mA due to pullup/pulldown current consumption.

55.11 10-bit ADC Characteristics

Table 55-22. Analog Power Supply Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DDANA}	ADC Analog Supply	—	2.8	3.3	3.6	V
	Maximum Voltage Ripple	RMS value, 10 kHz to 10 MHz	—	—	20	mV
I_{VDDANA}	Current Consumption	Sleep Mode	—	—	1	µA
		Normal Mode	—	—	750	

Table 55-23. Channel Conversion Time and ADC Clock

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{ADC}	ADC Clock Frequency(1)	—	—	—	10	MHz
t_{CP_ADC}	ADC Clock Period	—	100	—	—	ns
$f_{SAMPLING}$	Sampling Frequency	f_{ADC} = 10 MHz	—	—	700	ksps
t_{START}	ADC Startup time	—	—	—	40	µs
$t_{TRACKTIM}$	Track and Hold Time	Refer to Section 55.11.1.1 “Track and Hold Time versus Source Output Impedance” for more details	400	—	—	ns
t_{CONV}	Conversion Time	—	—	10	—	t_{CP_ADC}

Note: 1. f_{ADCCLK} configured in ADC_MR must be equal to $2 \times f_{ADC}$

Table 55-24. External Voltage Reference Input

Parameter	Conditions	Min	Typ	Max	Unit
ADVREF Input Voltage Range	2.8V < V_{DDANA} < 3.6V	2.8	3.3	V_{DDANA}	V
ADVREF Current	—	—	—	600	µA

55.11.1 Performance Characteristics

Table 55-25. Static Performance Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
	Resolution	—	—	10	—	bits
INL	Integral Non-linearity	—	-2	—	+2	LSB
DNL	Differential Non-linearity	—	-1	—	+1	LSB
E_G	Gain Error	—	-2	—	+2	LSB
E_O	Offset Error	—	-10	—	+10	mV

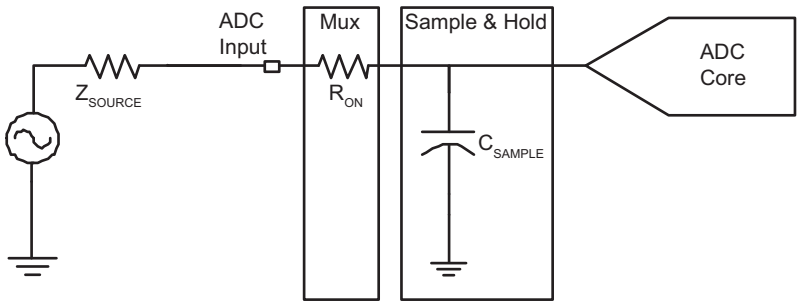
Table 55-26. Dynamic Performance Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
SNR	Signal-to-Noise Ratio	—	60	—	—	dB
THD	Total Harmonic Distortion	—	60	—	—	dB
SINAD	Signal-to-Noise and Distortion	—	56	—	—	dB
ENOB	Effective Number of Bits	—	9	—	—	bits

55.11.1.1 Track and Hold Time versus Source Output Impedance

The following figure gives a simplified acquisition path.

Figure 55-2. Simplified Acquisition Path



During the tracking phase the ADC needs to track the input signal during the tracking time shown below:

$t_{TRACK} = 0.1 \times Z_{SOURCE} + 400$

With t_{TRACK} expressed in ns and Z_{SOURCE} expressed in ohms.

Table 55-27. Analog Inputs

Parameter	Min	Typ	Max	Unit
Input Voltage Range	0	—	V_{ADVREF}	V
Input Leakage Current	—	—	±0.5	µA
Input Capacitance	—	—	8	pF
Input Source Impedance (Z_{SOURCE})	—	50	—	Ω

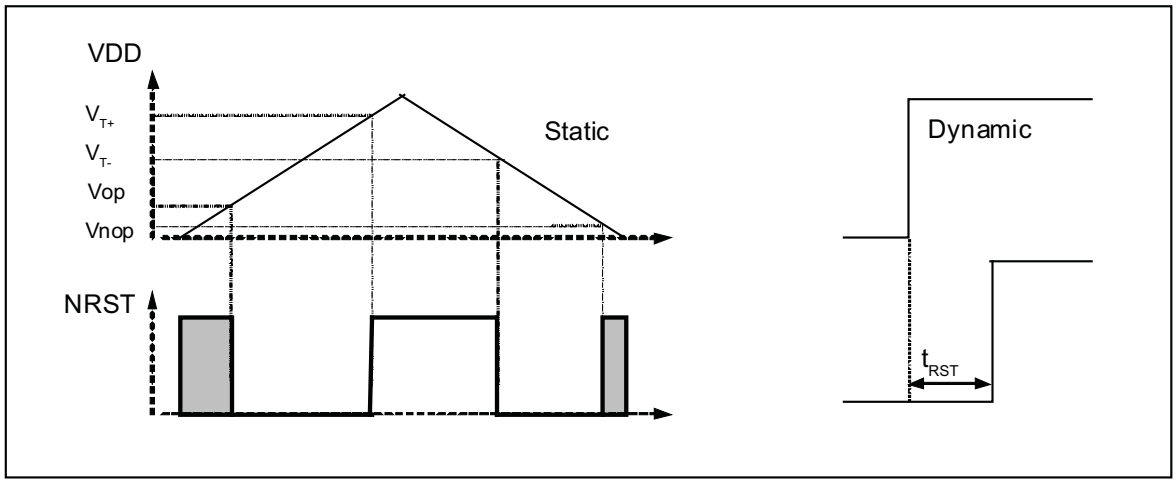
55.11.1.2 ADC Application Information

For more information on data converter terminology, refer to the application note “Data Converter Terminology”, Atmel literature No. 6022, available on www.atmel.com.

55.12 POR Characteristics

A general presentation of Power-On-Reset (POR) characteristics is provided in Figure 55-3.

Figure 55-3. General Presentation of POR Behavior



When a very slow (versus t_{RST}) supply rising slope is applied on POR VDD pin, the reset time becomes negligible and the reset signal is released when VDD rises higher than V_{T+}.

When a very fast (versus t_{RST}) supply rising slope is applied on POR VDD pin, the voltage threshold becomes negligible and the reset signal is released after t_{RST}. It is the smallest possible reset time.

Table 55-28. Backup Power Supply POR Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{T+}	Threshold Voltage Rising	—	1.78	1.83	1.88	V
V_{T-}	Threshold Voltage Falling	—	1.67	1.73	1.83	V
t_{RST}	Reset Time	—	120	260	650	µs
I_{DD}	Current Consumption	After t_{RST}	—	240	500	nA

Table 55-29. Core Power Supply POR Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{T+}	Threshold Voltage Rising	—	0.85	1.02	1.08	V
V_{T-}	Threshold Voltage Falling	—	0.8	0.96	1.06	V
t_{RST}	Reset Time	—	100	260	600	µs
I_{DD}	Current Consumption	—	—	250	500	nA

Table 55-30. IO Power Supply POR Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{T+}	Threshold Voltage Rising	Minimum Slope of +1.9 V/ms	2.8	2.9	3.0	V
V_{T-}	Threshold Voltage Falling	—	2.7	2.8	2.9	V
t_{RST}	Reset Time	—	100	300	600	µs
I_{DD}	Current Consumption	After t_{RST}	—	200	500	nA

55.13 SMC Timings

55.13.1 Timing Conditions

SMC Timings are given in MAX corners.

Timings assuming a capacitance load on data, control and address pads are given in Table 55-31.

Table 55-31. Capacitance Load

Supply	Corner	
	Max	Min
3.3V	50 pF	50 pF
1.8V	30 pF	30 pF

In the tables that follow, t_{CPMCK} is MCK period.

55.13.2 Timing Extraction

55.13.2.1 Read Timings

Table 55-32. SMC Read Signals - NRD Controlled (READ_MODE = 1)

Symbol	Parameter	Min		Max		Unit
	VDDIOM supply:	1.8V	3.3V	1.8V	3.3V	
NO HOLD SETTINGS (nrd hold = 0)						
SMC_1	Data Setup before NRD High	17.3	15.8	—	—	ns
SMC_2	Data Hold after NRD High	8.2	6.5	—	—	ns
HOLD SETTINGS (nrd hold ≠ 0)						
SMC_3	Data Setup before NRD High	13.0	11.5	—	—	ns
SMC_4	Data Hold after NRD High	8.2	6.5	—	—	ns
HOLD or NO HOLD SETTINGS (nrd hold ≠ 0, nrd hold =0)						
SMC_5	NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 Valid before NRD High	(nrd setup + nrd pulse) × t_{CPMCK} + 4.3	(nrd setup + nrd pulse) × t_{CPMCK} + 4.4	—	—	ns
SMC_6	NCS low before NRD High	(nrd setup + nrd pulse - ncs rd setup) × t_{CPMCK} + 3.6	(nrd setup + nrd pulse - ncs rd setup) × t_{CPMCK} + 4.5	—	—	ns
SMC_7	NRD Pulse Width	nrd pulse × t_{CPMCK} + 0.1	nrd pulse × t_{CPMCK}	—	—	ns

Table 55-33. SMC Read Signals - NCS Controlled (READ_MODE = 0)

Symbol	Parameter	Min		Max		Unit
	VDDIOM supply:	1.8V	3.3V	1.8V	3.3V	
NO HOLD SETTINGS (ncs rd hold = 0)						
SMC_8	Data Setup before NCS High	30.8	29.0	—	—	ns
SMC_9	Data Hold after NCS High	8.9	7.2	—	—	ns
HOLD SETTINGS (ncs rd hold ≠ 0)						
SMC_{10}	Data Setup before NCS High	26.2	24.5	—	—	ns
SMC_{11}	Data Hold after NCS High	8.9	7.2	—	—	ns
HOLD or NO HOLD SETTINGS (ncs rd hold ≠ 0, ncs rd hold = 0)						
SMC_{12}	NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS High	(ncs rd setup + ncs rd pulse) × t_{CPMCK} + 28.7	(ncs rd setup + ncs rd pulse) × t_{CPMCK} + 29.1	—	—	ns
SMC_{13}	NRD low before NCS High	(ncs rd setup + ncs rd pulse - nrd setup) × t_{CPMCK} + 23.4	(ncs rd setup + ncs rd pulse - nrd setup) × t_{CPMCK} +24.1	—	—	ns
SMC_{14}	NCS Pulse Width	ncs rd pulse length × t_{CPMCK}	ncs rd pulse length × t_{CPMCK}	—	—	ns

55.13.2.2 Write Timings

Table 55-34. SMC Write Signals - NWE Controlled (WRITE_MODE = 1)

Symbol	Parameter	Min		Max		Unit
	VDDIOM supply:	1.8V	3.3V	1.8V	3.3V	
HOLD or NO HOLD SETTINGS (nwe hold ≠ 0, nwe hold = 0)						
SMC_{15}	Data Out Valid before NWE High	nwe pulse × t_{CPMCK} + 4.6	nwe pulse × t_{CPMCK} + 5.8	—	—	ns
SMC_{16}	NWE Pulse Width	nwe pulse × t_{CPMCK} + 0.1	nwe pulse × t_{CPMCK}	—	—	ns
SMC_{17}	NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NWE low	nwe setup × t_{CPMCK} + 5.1	nwe pulse × t_{CPMCK} + 5.0	—	—	ns
SMC_{18}	NCS low before NWE high	(nwe setup - ncs rd setup + nwe pulse) × t_{CPMCK} + 4.4	(nwe setup - ncs rd setup + nwe pulse) × t_{CPMCK} + 5.2	—	—	ns
HOLD SETTINGS (nwe hold ≠ 0)						
SMC_{19}	NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 change	nwe hold × t_{CPMCK} + 2.7	nwe hold × t_{CPMCK} +1.7	—	—	ns
SMC_{20}	NWE High to NCS Inactive (1)	(nwe hold - ncs wr hold) × t_{CPMCK} + 1.0	(nwe hold - ncs wr hold) × t_{CPMCK} +1.4	—	—	ns
NO HOLD SETTINGS (nwe hold = 0)						
SMC_{21}	NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25, NCS change(1)	1.3	0.7	—	—	ns

Notes: 1. hold length = total cycle duration - setup duration - pulse duration. “hold length” is for “ncs wr hold length” or “NWE hold length”.

Table 55-35. SMC Write NCS Controlled (WRITE_MODE = 0)

Symbol	Parameter	Min		Max		Unit
	VDDIOM supply:	1.8V	3.3V	1.8V	3.3V	
SMC_{22}	Data Out Valid before NCS High	ncs wr pulse × t_{CPMCK} + 14.3	ncs wr pulse × t_{CPMCK} + 15.6	—	—	ns
SMC_{23}	NCS Pulse Width	ncs wr pulse × t_{CPMCK}	ncs wr pulse × t_{CPMCK}	—	—	ns
SMC_{24}	NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS low	ncs wr setup × t_{CPMCK} + 6.3	ncs wr setup × t_{CPMCK} + 6.4	—	—	ns
SMC_{25}	NWE low before NCS high	(ncs wr setup - nwe setup + ncs pulse) × t_{CPMCK} + 4.5	(ncs wr setup - nwe setup + ncs pulse) × t_{CPMCK} + 4.2	—	—	ns
SMC_{26}	NCS High to Data Out, NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25, change	ncs wr hold × t_{CPMCK} + 5.0	ncs wr hold × t_{CPMCK} + 4.7	—	—	ns
SMC_{27}	NCS High to NWE Inactive	(ncs wr hold - nwe hold) × t_{CPMCK} + 4.9	(ncs wr hold - nwe hold) × t_{CPMCK} + 4.7	—	—	ns

Figure 55-4. SMC Timings - NCS Controlled Read and Write

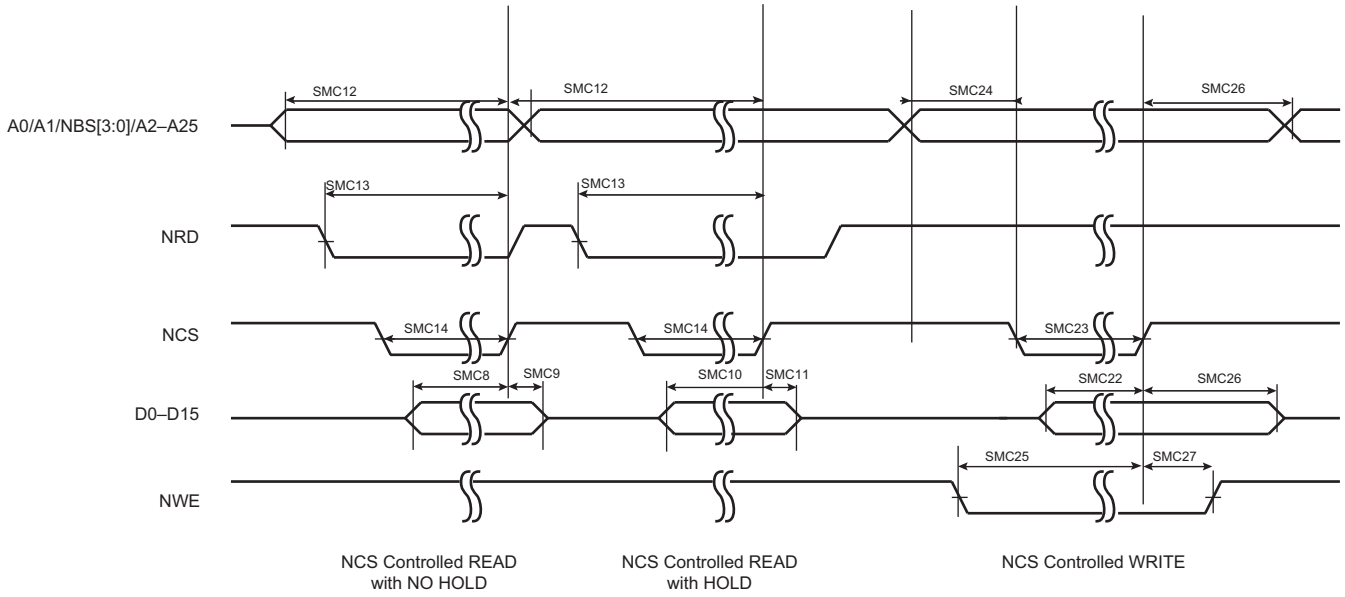
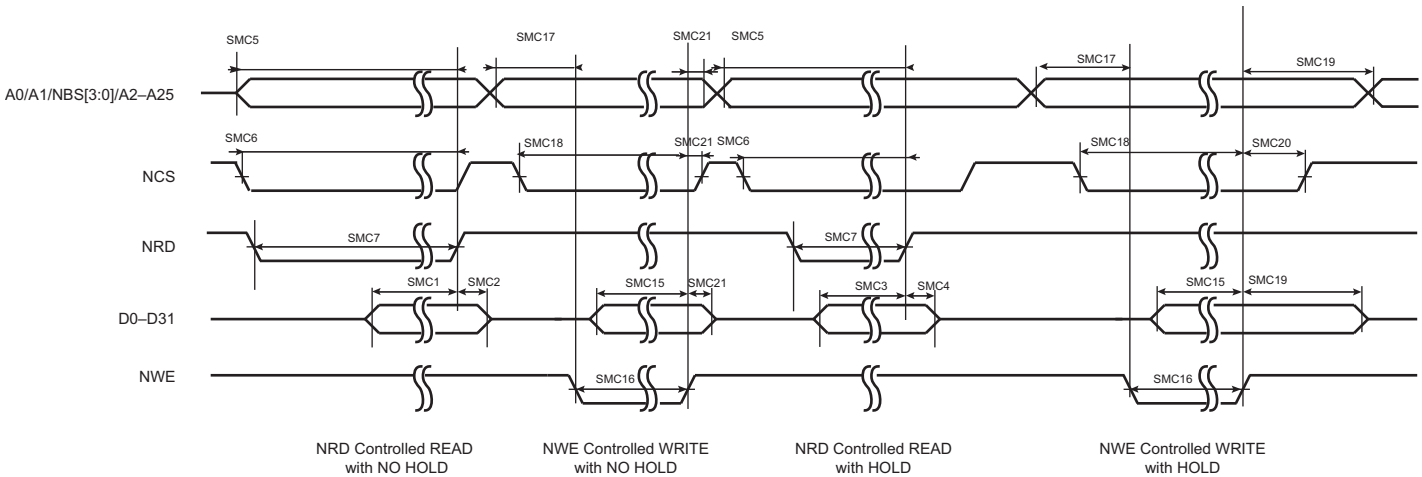


Figure 55-5. SMC Timings - NRD Controlled Read and NWE Controlled Write



55.14 SPI Timings

55.14.1 Maximum SPI Frequency

The following formulas give maximum SPI frequency in Master read and write modes and in Slave read and write modes.

Master Write Mode

The SPI is only sending data to a slave device such as an LCD, for example. The limit is given by SPI_2 (or SPI_5) timing.

Master Read Mode

$$f_{SPCK}Max = \frac{1}{SPI_0(\text{or } SPI_3) + t_{valid}}$$

t_{valid} is the slave time response to output data after deleting an SPCK edge. For a non-volatile memory with t_{valid} (or t_v) = 12 ns, f_{SPCK}max = 45 MHz at V_{DDIO} = 3.3V.

Slave Read Mode

In slave mode, SPCK is the input clock for the SPI. The max SPCK frequency is given by setup and hold timings SPI_7/SPI_8 (or SPI_{10}/SPI_{11}). Since this gives a frequency well above the pad limit, the limit in slave read mode is given by SPCK pad.

Slave Write Mode

$$f_{SPCK}Max = \frac{1}{SPI_6(\text{or } SPI_9) + t_{setup}}$$

t_{setup} is the setup time from the master before sampling data (12 ns).

This gives f_{SPCK}Max = 45 MHz @ V_{DDIO} = 3.3V.

55.14.2 Timing Conditions

Timings are given assuming a capacitance load on MISO, SPCK and MOSI as shown in Table 55-36.

Table 55-36. Capacitance Load for MISO, SPCK and MOSI

Supply	Corner	
	Max	Min
3.3V	40 pF	40 pF
1.8V (SPI0 only)	20 pF	40 pF

55.14.3 Timing Extraction

Figure 55-6. SPI Master Mode 1 and 2

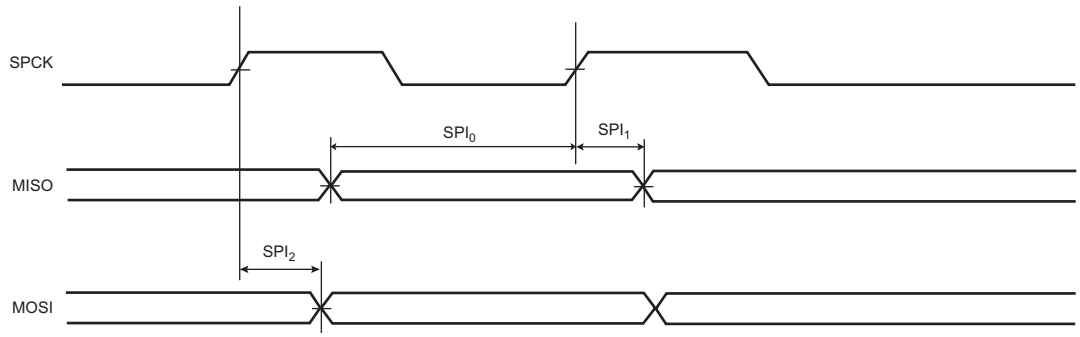


Figure 55-7. SPI Master Mode 0 and 3

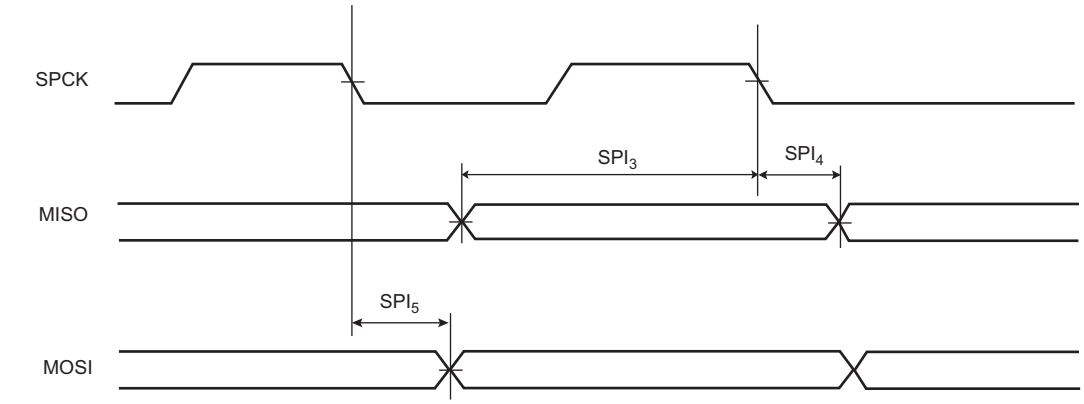


Figure 55-8. SPI Slave Mode 0 and 3

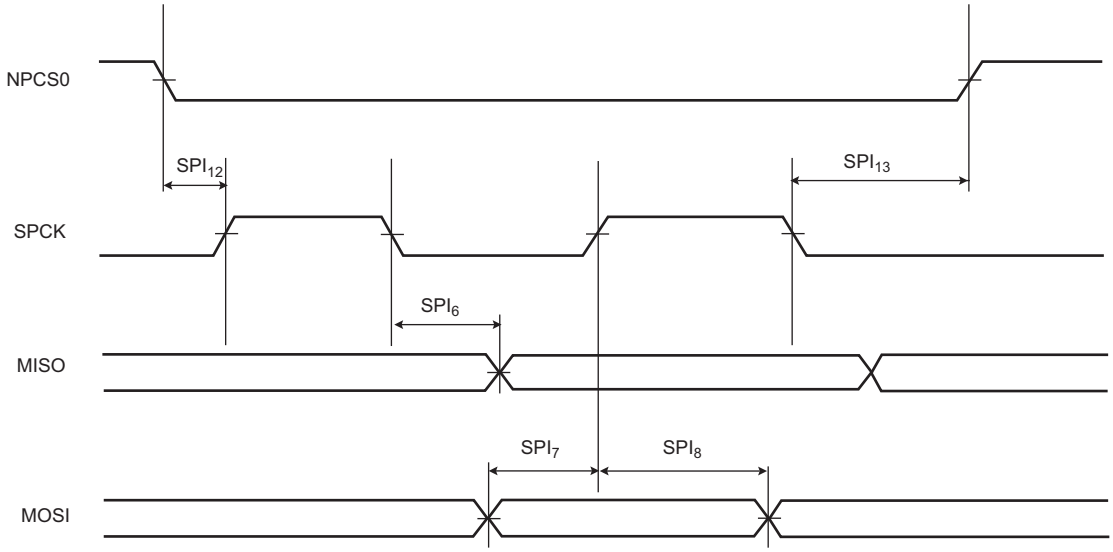


Figure 55-9. SPI Slave Mode 1 and 2

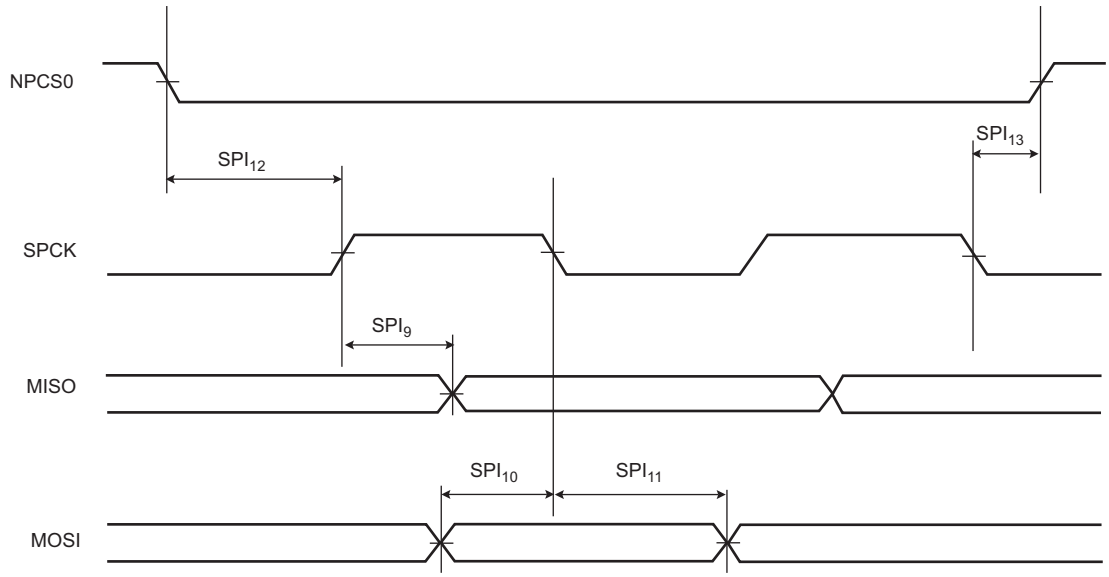


Figure 55-10. SPI Slave Mode - NPCS Timings

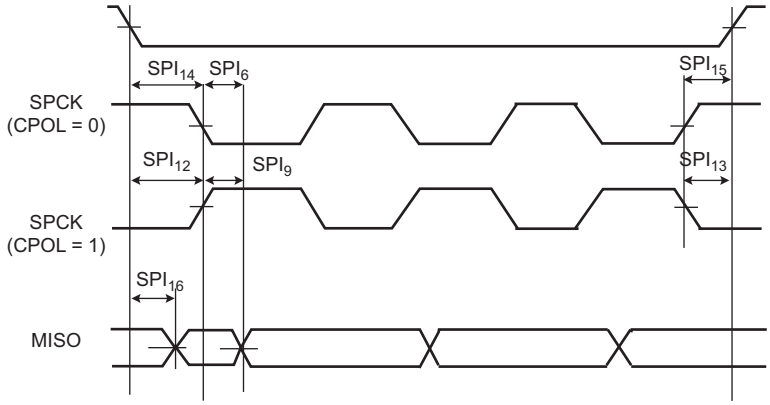


Table 55-37. SPI Timings with 3.3V Peripheral Supply

Symbol	Parameter	Conditions	Min	Max	Unit
Master Mode					
SPI_0	MISO Setup time before SPCK rises	—	9.8	—	ns
SPI_1	MISO Hold time after SPCK rises	—	6.6	—	ns
SPI_2	SPCK rising to MOSI	—	0(1)	5.4(1)	ns
SPI_3	MISO Setup time before SPCK falls	—	10.3	—	ns
SPI_4	MISO Hold time after SPCK falls	—	6.6	—	ns
SPI_5	SPCK falling to MOSI	—	0(1)	5.4(1)	ns
Slave Mode					
SPI_6	SPCK falling to MISO	—	2.6(1)	9.5(1)	ns
SPI_7	MOSI Setup time before SPCK rises	—	3.3	—	ns
SPI_8	MOSI Hold time after SPCK rises	—	0.7	—	ns
SPI_9	SPCK rising to MISO	—	2.5(1)	9.1(1)	ns
SPI_{10}	MOSI Setup time before SPCK falls	—	3.3	—	ns
SPI_{11}	MOSI Hold time after SPCK falls	—	0.6	—	ns
SPI_{12}	NPCS0 setup to SPCK rising	—	6.9	—	ns
SPI_{13}	NPCS0 hold after SPCK falling	—	20.8	—	ns
SPI_{14}	NPCS0 setup to SPCK falling	—	6.6	—	ns
SPI_{15}	NPCS0 hold after SPCK rising	—	21.3	—	ns
SPI_{16}	NPCS0 falling to MISO valid	—	—	14.9	ns

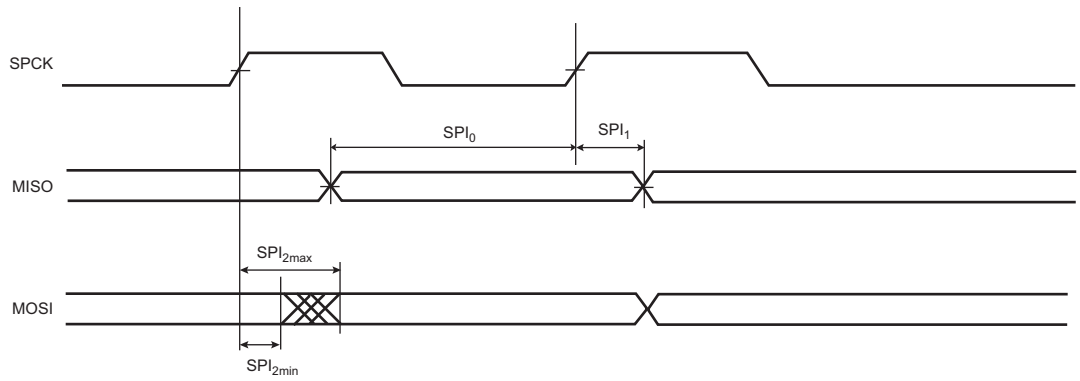
Notes: 1. For output signals, minimum and maximum access time must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access timing is the time between the SPCK rising or falling edge and the signal stabilizes. Figure 55-11 illustrates minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 55-38. SPI Timings with 1.8V Peripheral Supply (SPI0 only)

Symbol	Parameter	Conditions	Min	Max	Unit
Master Mode					
SPI_0	MISO Setup time before SPCK rises	—	11.6	—	ns
SPI_1	MISO Hold time after SPCK rises	—	8.9	—	ns
SPI_2	SPCK rising to MOSI	—	0(1)	5.1(1)	ns
SPI_3	MISO Setup time before SPCK falls	—	11.7	—	ns
SPI_4	MISO Hold time after SPCK falls	—	8.6	—	ns
SPI_5	SPCK falling to MOSI	—	0(1)	4.8(1)	ns
Slave Mode					
SPI_6	SPCK falling to MISO	—	3.5(1)	10.6(1)	ns
SPI_7	MOSI Setup time before SPCK rises	—	3.5	—	ns
SPI_8	MOSI Hold time after SPCK rises	—	0.6	—	ns
SPI_9	SPCK rising to MISO	—	3.3(1)	10.3(1)	ns
SPI_{10}	MOSI Setup time before SPCK falls	—	3.5	—	ns
SPI_{11}	MOSI Hold time after SPCK falls	—	0.6	—	ns
SPI_{12}	NPCS0 setup to SPCK rising	—	7.0	—	ns
SPI_{13}	NPCS0 hold after SPCK falling	—	20.4	—	ns
SPI_{14}	NPCS0 setup to SPCK falling	—	6.7	—	ns
SPI_{15}	NPCS0 hold after SPCK rising	—	20.8	—	ns
SPI_{16}	NPCS0 falling to MISO valid	—	—	16.6	ns

Notes: 1. For output signals, minimum and maximum access time must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access timing is the time between the SPCK rising or falling edge and the signal stabilizes. Figure 55-11 illustrates minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Figure 55-11. Minimum and Maximum Access Time for SPI Output Signal



55.15 MPDDRC Timings

55.15.1 Board Design Constraints

As SAMA5D4 embeds impedance-calibrated pads, there are no capacitive constraints on DDR signals. However, a board must be designed and equipped in order to respect DQS setup times, including propagation time and intrinsic delay in SDRAM device. In all cases, line length to memory device must not exceed 5 cm.

55.15.2 DDR2-SDRAM

Table 55-39. DDR2-SDRAM System Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
t_{DDRCK}	DDRCK cycle time	—	5.0	8.0	ns
t_{SDQS}	DQS setup time	—	—	0.6	ns

55.15.3 LPDDR1-SDRAM

Table 55-40. LPDDR1-SDRAM System Clock Waveform Parameters

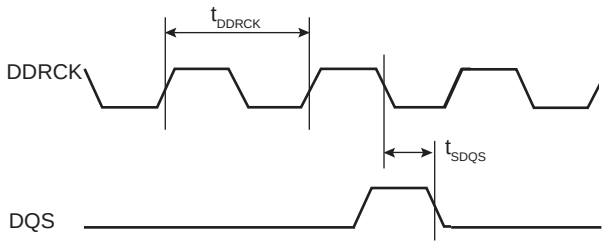
Symbol	Parameter	Conditions	Min	Max	Unit
t_{DDRCK}	DDRCK cycle time	—	5.0	—	ns
t_{SDQS}	DQS setup time	—	—	0.6	ns

55.15.4 LPDDR2-SDRAM

Table 55-41. LPDDR2-SDRAM System Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
t_{DDRCK}	DDRCK cycle time	—	5.0	—	ns
t_{SDQS}	DQS setup time	—	—	0.6	ns

Figure 55-12. DQS Timings



55.16 SSC Timings

55.16.1 Timing Conditions

Timings assuming a capacitance load are given in Table 55-42.

Table 55-42. Capacitance Load

Supply	Corner	
	Max	Min
3.3V	30 pF	30 pF
1.8V (SSC_1 only)	20 pF	20 pF

55.16.2 Timing Extraction

Figure 55-13. SSC Transmitter, TK and TF in Output

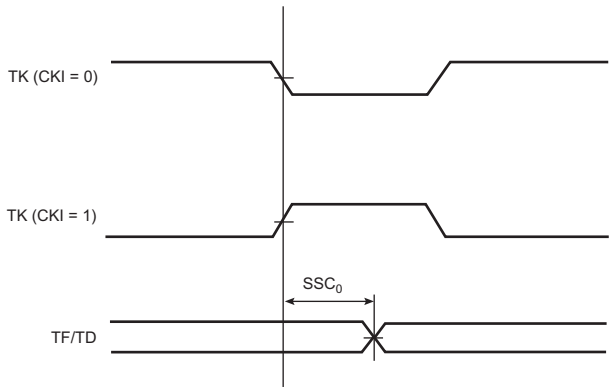


Figure 55-14. SSC Transmitter, TK in Input and TF in Output

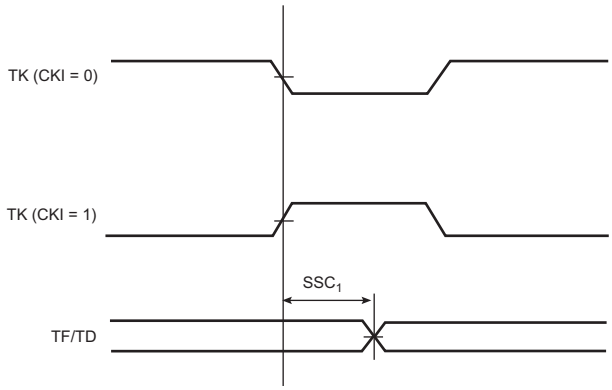


Figure 55-15. SSC Transmitter, TK in Output and TF in Input

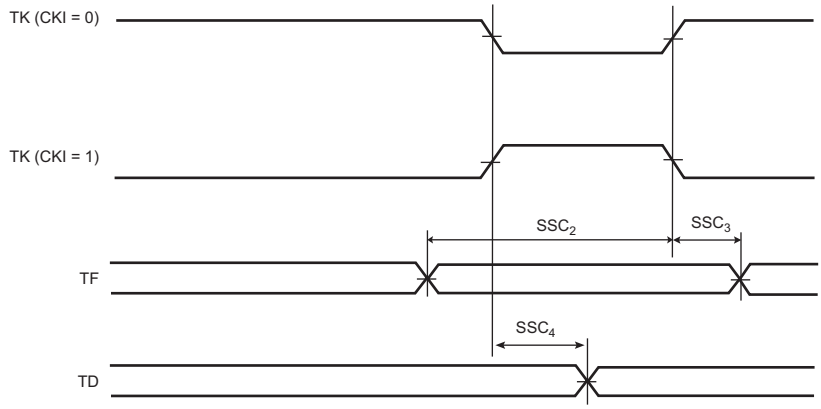


Figure 55-16. SSC Transmitter, TK and TF in Input

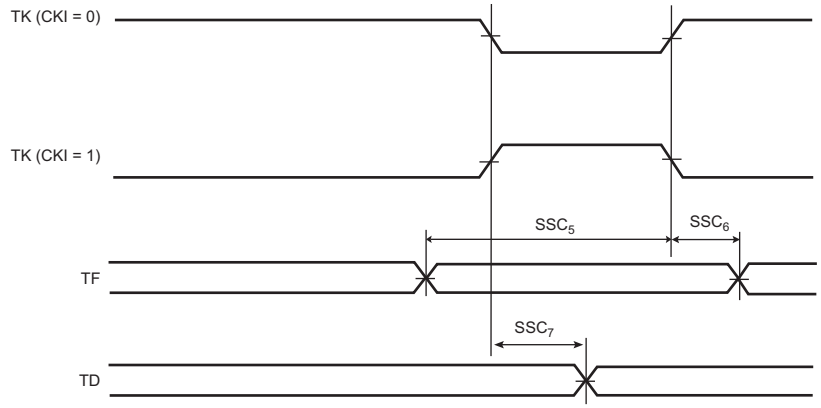


Figure 55-17. SSC Receiver RK and RF in Input

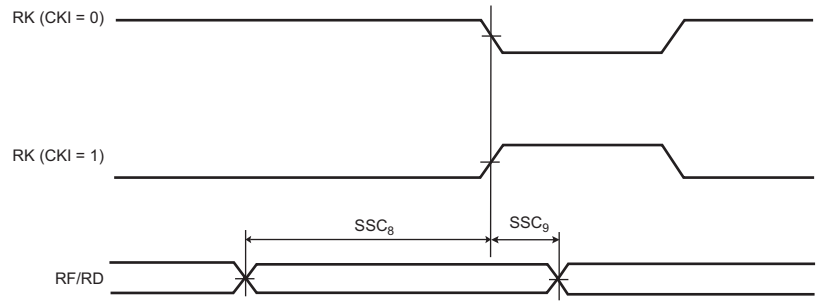


Figure 55-18. SSC Receiver, RK in Input and RF in Output

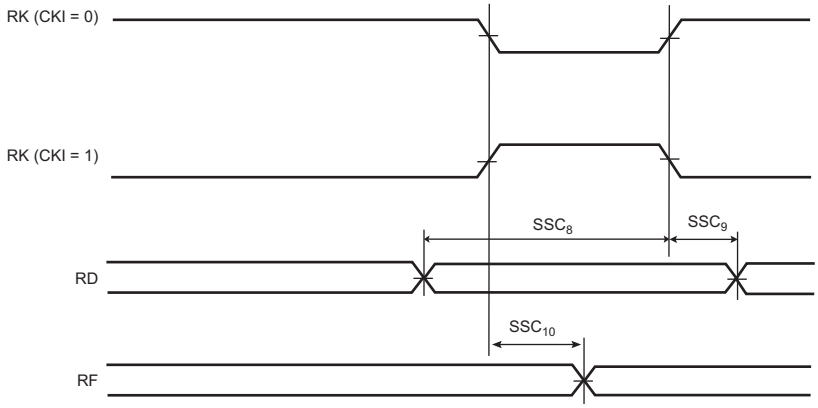


Figure 55-19. SSC Receiver, RK and RF in Output

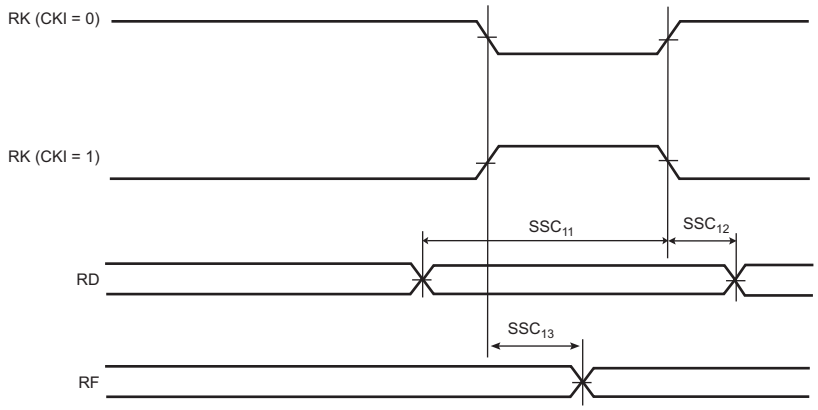


Figure 55-20. SSC Receiver, RK in Output and RF in Input

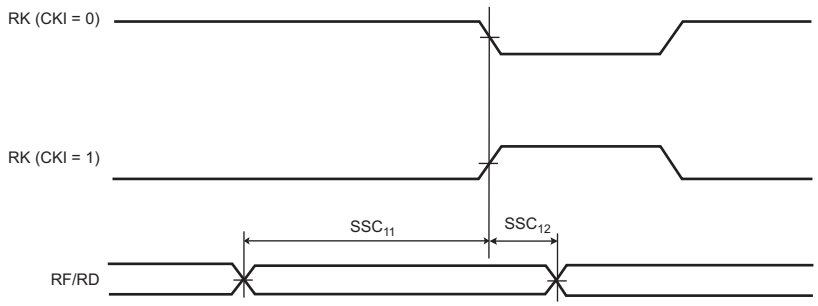


Table 55-43. SSC Timings with 3.3V Peripheral Supply

Symbol	Parameter	Conditions	Min	Max	Unit
Transmitter					
SSC_0	TK edge to TF/TD (TK output, TF output)	—	0(1)	6.1(1)	ns
SSC_1	TK edge to TF/TD (TK input, TF output)	—	2.5(1)	10.9(1)	ns
SSC_2	TF setup time before TK edge (TK output)	—	8.2	—	ns
SSC_3	TF hold time after TK edge (TK output)	—	5.4	—	ns
SSC_4	TK edge to TF/TD (TK output, TF input)	—	0(1)	4.1(1)	
		STTDLY = 0 START = 4, 5 or 7	$2 \times t_{CPMCK}$(1)	$4.1 + (2 \times t_{CPMCK})$(1)	ns
SSC_5	TF setup time before TK edge (TK input)	—	0	—	ns
SSC_6	TF hold time after TK edge (TK input)	—	t_{CPMCK}	—	ns
SSC_7	TK edge to TF/TD (TK input, TF input)	—	2.5(1)	9.7(1)	
		STTDLY = 0 START = 4, 5 or 7	$2.5 + (3 \times t_{CPMCK})$(1)	$9.7 + (3 \times t_{CPMCK})$(1)	ns
Receiver					
SSC_8	RF/RD setup time before RK edge (RK input)	—	0	—	ns
SSC_9	RF/RD hold time after RK edge (RK input)	—	t_{CPMCK}	—	ns
SSC_{10}	RK edge to RF (RK input)	—	2.6(1)	10.5(1)	ns
SSC_{11}	RF/RD setup time before RK edge (RK output)	—	$7.9 - t_{CPMCK}$	—	ns
SSC_{12}	RF/RD hold time after RK edge (RK output)	—	$t_{CPMCK} - 5.0$	—	ns
SSC_{13}	RK edge to RF (RK output)	—	0(1)	5.8(1)	ns

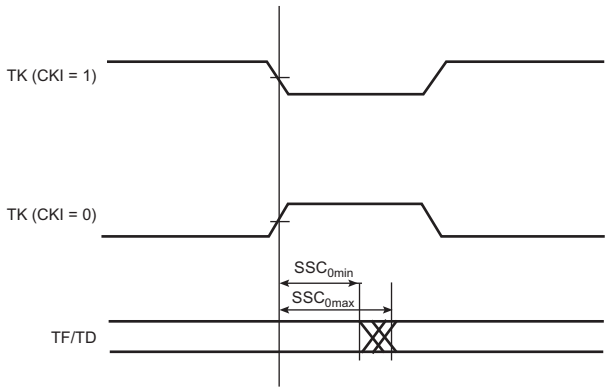
Note: 1. For output signals (TF, TD, RF), minimum and maximum access times are defined. The minimum access time is the time between the TK (or RK) edge and the signal change. The maximum access timing is the time between the TK edge and the signal stabilization. Figure 55-21 illustrates minimum and maximum accesses for SSC_0. The same applies for SSC_1, SSC_4, SSC_7, SSC_{10} and SSC_{13}.

Table 55-44. SSC Timings with 1.8V Peripheral Supply (SSC1 only)

Symbol	Parameter	Conditions	Min	Max	Unit
Transmitter					
SSC_0	TK edge to TF/TD (TK output, TF output)	—	0[1]	3.9 [1]	ns
SSC_1	TK edge to TF/TD (TK input, TF output)	—	3.3[1]	11.7[1]	ns
SSC_2	TF setup time before TK edge (TK output)	—	10.2	—	ns
SSC_3	TF hold time after TK edge (TK output)	—	7.3	—	ns
SSC_4	TK edge to TF/TD (TK output, TF input)	—	0[1]	3.9[1]	
		STTDLY = 0 START = 4, 5 or 7	$2 \times t_{CPMCK}$[1]	$3.9 + (2 \times t_{CPMCK})$[1]	ns
SSC_5	TF setup time before TK edge (TK input)	—	0	—	ns
SSC_6	TF hold time after TK edge (TK input)	—	t_{CPMCK}	—	ns
SSC_7	TK edge to TF/TD (TK input, TF input)	—	3.4[1]	11.6[1]	
		STTDLY = 0 START = 4, 5 or 7	$3.4 + (3 \times t_{CPMCK})$[1]	$11.6 + (3 \times t_{CPMCK})$[1]	ns
Receiver					
SSC_8	RF/RD setup time before RK edge (RK input)	—	0	—	ns
SSC_9	RF/RD hold time after RK edge (RK input)	—	t_{CPMCK}	—	ns
SSC_{10}	RK edge to RF (RK input)	—	3.1[1]	11.1[1]	ns
SSC_{11}	RF/RD setup time before RK edge (RK output)	—	$9.9 - t_{CPMCK}$	—	ns
SSC_{12}	RF/RD hold time after RK edge (RK output)	—	$t_{CPMCK} - 6.8$	—	ns
SSC_{13}	RK edge to RF (RK output)	—	0[1]	3.9[1]	ns

Note: 1. For output signals (TF, TD, RF), minimum and maximum access times are defined. The minimum access time is the time between the TK (or RK) edge and the signal change. The maximum access timing is the time between the TK edge and the signal stabilization. Figure 55-21 illustrates minimum and maximum accesses for SSC_0. The same applies for SSC_1, SSC_4, SSC_7, SSC_{10} and SSC_{13}.

Figure 55-21. Minimum and Maximum Access Time of Output Signals



55.17 ISI Timings

55.17.1 Timing Conditions

Timings assuming capacitance loads are given in Table 55-45.

Table 55-45. Capacitance Load

Supply	Corner	
	Max	Min
3.3V	30 pF	30 pF

55.17.2 Timing Extraction

Figure 55-22. ISI Timing Diagram

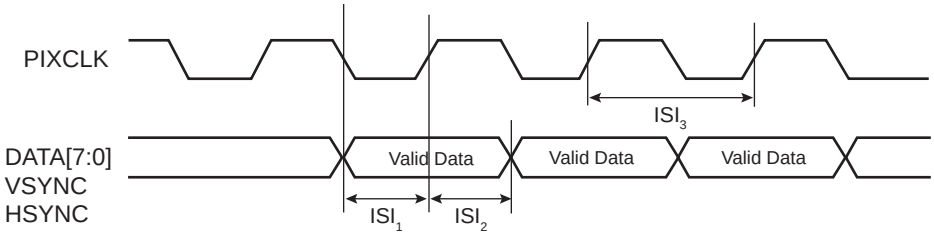


Table 55-46. ISI Timings with Peripheral Supply 3.3V

Symbol	Parameter	Min	Max	Unit
ISI_1	DATA/VSYNC/HSYNC setup time	3.2	—	ns
ISI_2	DATA/VSYNC/HSYNC hold time	0.7	—	ns
ISI_3	PIXCLK frequency	—	75	MHz

55.18 MCI Timings

The High Speed MultiMedia Card Interface (HSMCI) supports the MultiMedia Card (MMC) Specification V4.3, the SD Memory Card Specification V2.0, the SDIO V2.0 specification, and CE-ATA V1.1.

55.19 Ethernet MAC (GMAC) Timings

55.19.1 Timing Conditions

Timings assuming a capacitance load on data and clock are given in Table 55-47.

Table 55-47. Capacitance Load on Data, Clock Pads

Supply	Corner	
	Max	Min
3.3V	20 pF	0 pF

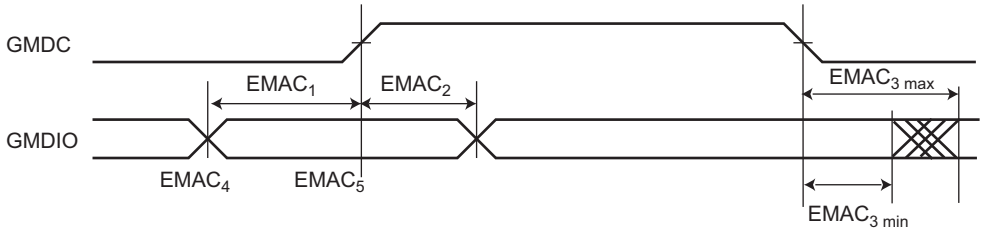
55.19.2 Timing Constraints

Table 55-48. Ethernet MAC Signals Relative to GMDC

Symbol	Parameter	Min	Max	Unit
$EMAC_1$	Setup for GMDIO from GMDC rising	10	—	ns
$EMAC_2$	Hold for GMDIO from GMDC rising	10	—	ns
$EMAC_3$	GMDIO toggling from GMDC rising	0(1)	300(1)	ns

Note: 1. For Ethernet MAC output signals, minimum and maximum access time are defined. The minimum access time is the time between the GMDC rising edge and the signal change. The maximum access timing is the time between the GMDC rising edge and the signal stabilizes. Figure 55-23 illustrates minimum and maximum accesses for $EMAC_3$.

Figure 55-23. Minimum and Maximum Access Time of Ethernet MAC Output Signals

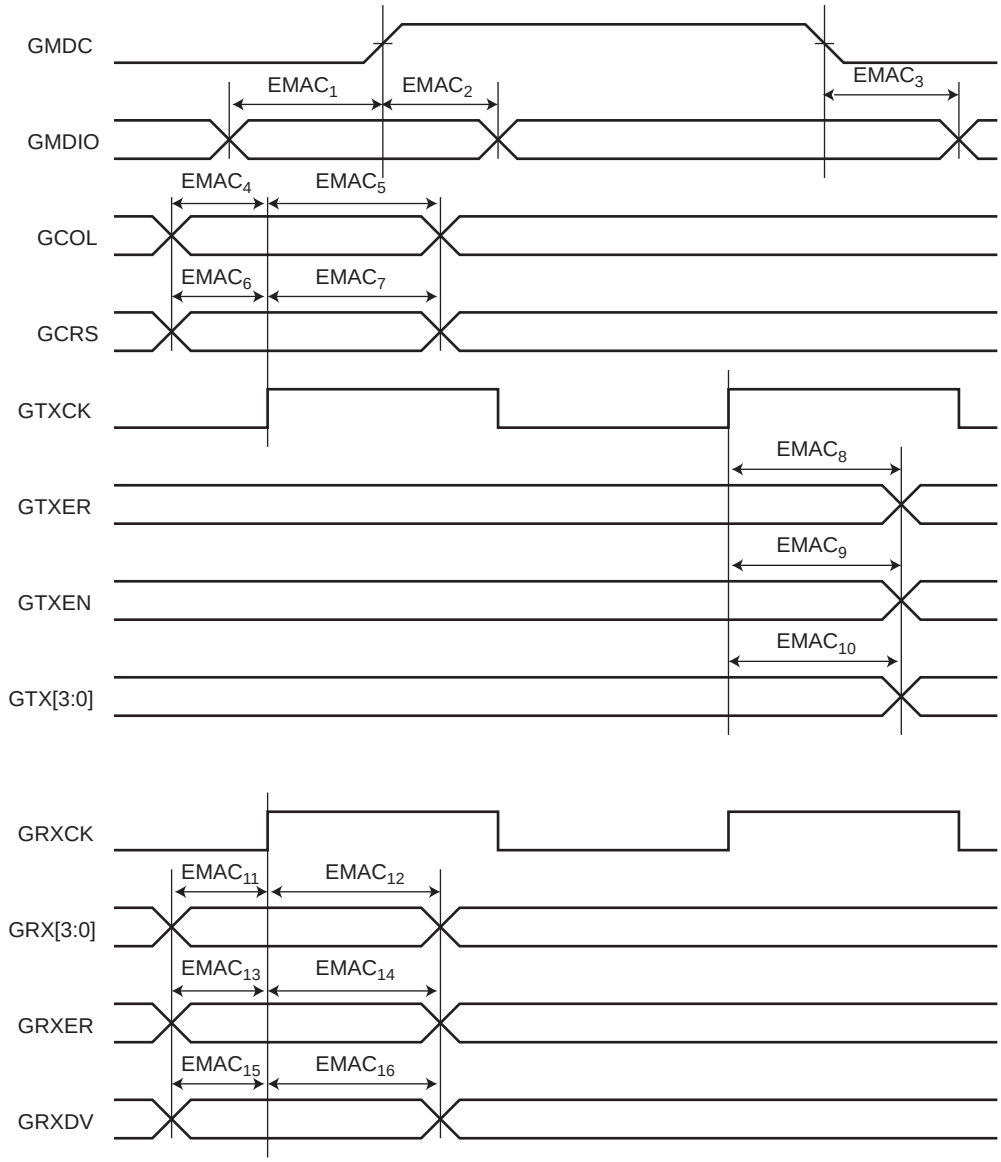


55.19.2.1 Ethernet MAC MII Mode

Table 55-49. Ethernet MAC MII Specific Signals

Symbol	Parameter	Min	Max	Unit
$EMAC_4$	Setup for GCOL from GTXCK rising	10	—	ns
$EMAC_5$	Hold for GCOL from GTXCK rising	10	—	ns
$EMAC_6$	Setup for GCRS from GTXCK rising	10	—	ns
$EMAC_7$	Hold for GCRS from GTXCK rising	10	—	ns
$EMAC_8$	GTXER toggling from GTXCK rising	10	25	ns
$EMAC_9$	GTXEN toggling from GTXCK rising	10	25	ns
$EMAC_{10}$	GTX toggling from GTXCK rising	10	25	ns
$EMAC_{11}$	Setup for GRX from GRXCK	10	—	ns
$EMAC_{12}$	Hold for GRX from GRXCK	10	—	ns
$EMAC_{13}$	Setup for GRXER from GRXCK	10	—	ns
$EMAC_{14}$	Hold for GRXER from GRXCK	10	—	ns
$EMAC_{15}$	Setup for GRXDV from GRXCK	10	—	ns
$EMAC_{16}$	Hold for GRXDV from GRXCK	10	—	ns

Figure 55-24. Ethernet MAC MII Mode

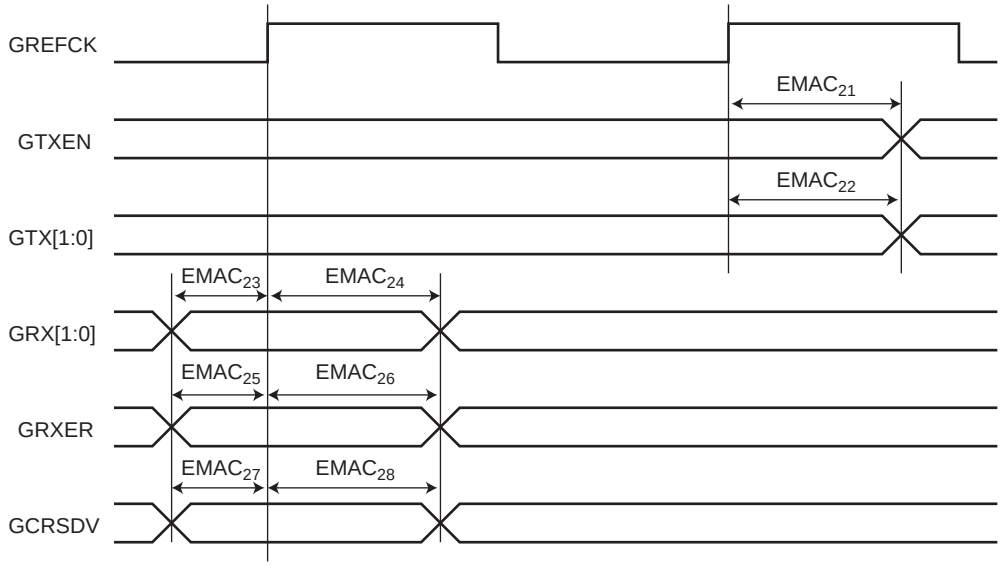


55.19.2.2 Ethernet MAC RMII Mode

Table 55-50. Ethernet MAC RMII Mode

Symbol	Parameter	Min	Max	Unit
$EMAC_{21}$	GTXEN toggling from GREFCK rising	2	16	ns
$EMAC_{22}$	GTX toggling from GREFCK rising	2	16	ns
$EMAC_{23}$	Setup for GRX from GREFCK rising	4	—	ns
$EMAC_{24}$	Hold for GRX from GREFCK rising	2	—	ns
$EMAC_{25}$	Setup for GRXER from GREFCK rising	4	—	ns
$EMAC_{26}$	Hold for GRXER from GREFCK rising	2	—	ns
$EMAC_{27}$	Setup for GCRSDV from GREFCK rising	4	—	ns
$EMAC_{28}$	Hold for GCRSDV from GREFCK rising	2	—	ns

Figure 55-25. Ethernet MAC RMII Timings



55.20 USART in Asynchronous Modes

In Asynchronous modes, the maximum baud rate that can be achieved is MCK2 / 8, if the bit USART_MR.OVER=1.

Example: if MCK2 = 90 MHz, the baud rate is 11.25 MBit/s.

55.21 USART in SPI Mode Timings

55.21.1 Timing Conditions

Timings assuming a capacitance load on MISO, SPCK and MOSI are given in Table 55-51.

Table 55-51. Capacitance Load for MISO, SPCK and MOSI

Supply	Corner	
	Max	Min
3.3V	40 pF	40 pF
1.8V (USART 3 and 4 only)	20 pF	40 pF

55.21.2 Timing Extraction

Figure 55-26. USART SPI Master Mode

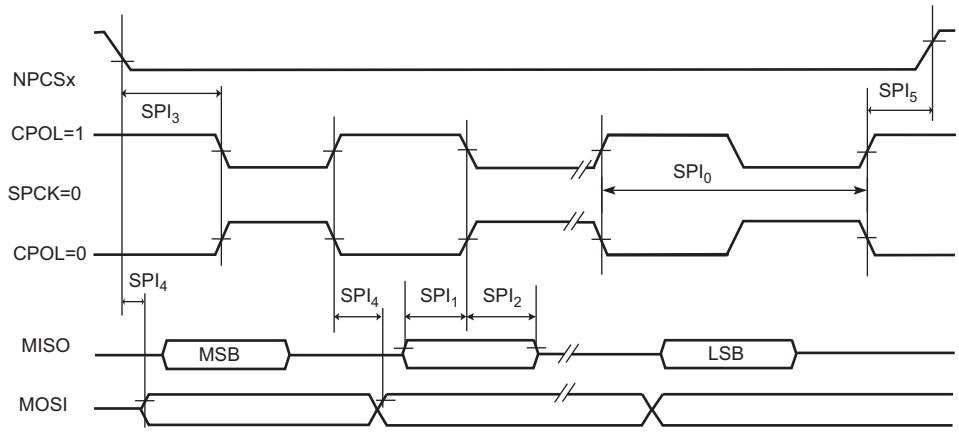


Figure 55-27. USART SPI Slave Mode 1 or 2

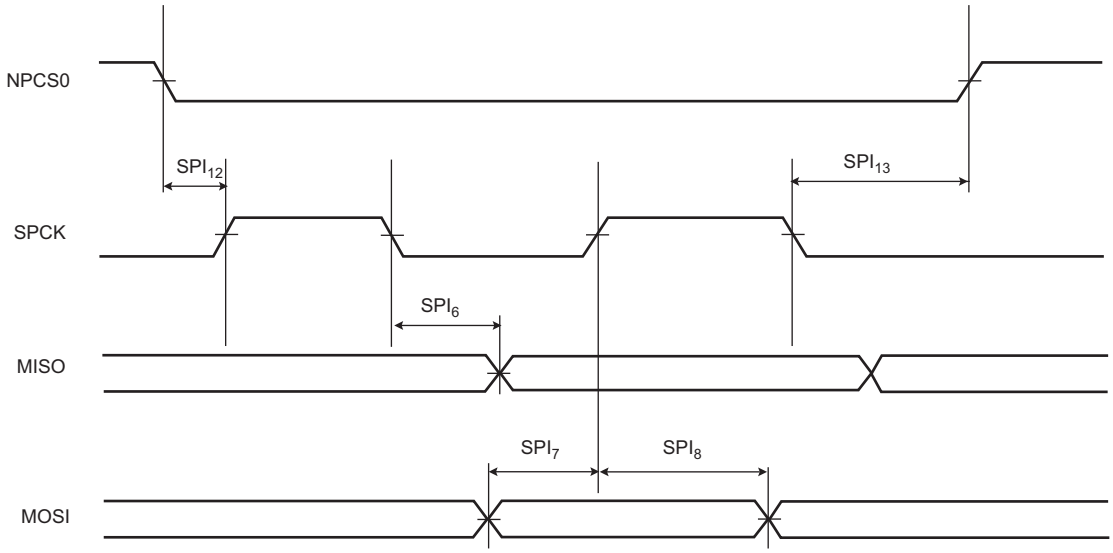


Figure 55-28. USART SPI Slave Mode 0 or 3

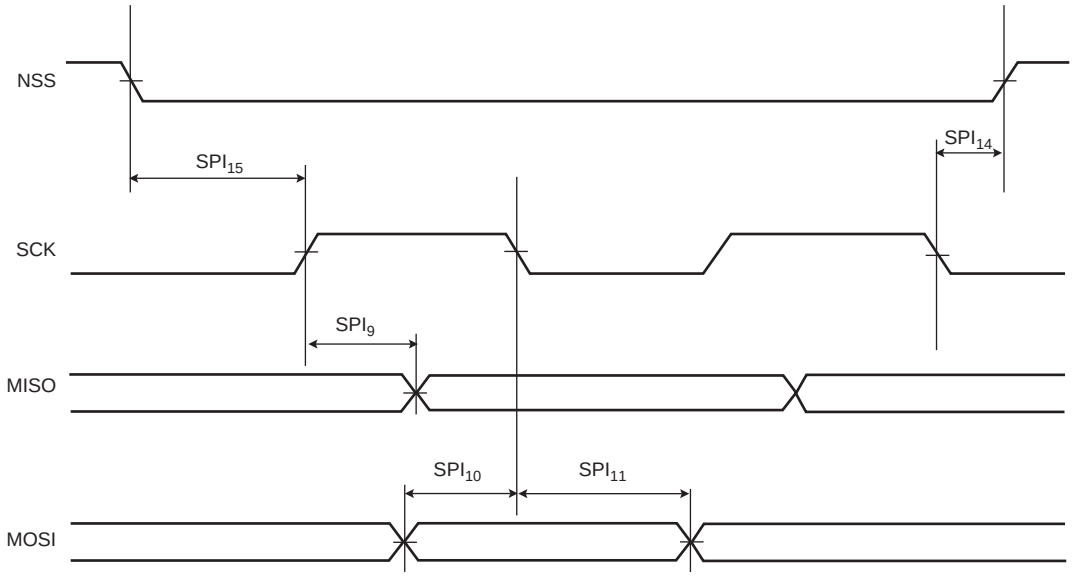


Table 55-52. USART SPI Timings 3.3V Peripheral Supply

Symbol	Parameter	Conditions	Min	Max	Unit
Master Mode					
SPI_0	SPCK frequency	—	MCK/6	—	Hz
SPI_1	Input Data Setup Time	—	$0.5 \times t_{CPMCK} + 3.7$	—	ns
SPI_2	Input Data Hold Time	—	$1.5 \times t_{CPMCK} + 1.4$	—	ns
SPI_3	Chip Select Active to Serial Clock	—	—	$1.5 \times t_{SPCK} - 1.9$	ns
SPI_4	Output Data Setup Time	—	0	12.4	ns
SPI_5	Serial Clock to Chip Select Inactive	—	—	$t_{SPCK} + 2.2$	ns
Slave Mode					
SPI_6	SPCK falling to MISO	—	2.4(1)	12.6(1)	ns
SPI_7	MOSI Setup time before SPCK rises	—	—	$2 \times t_{CPMCK} + 2.8$	ns
SPI_8	MOSI Hold time after SPCK rises	—	1.1	—	ns
SPI_9	SPCK rising to MISO	—	2.3(1)	12.1(1)	ns
SPI_{10}	MOSI Setup time before SPCK falls	—	2.7	—	ns
SPI_{11}	MOSI Hold time after SPCK falls	—	0.6	—	ns
SPI_{12}	NPCS0 setup to SPCK rising	—	$2.5 \times t_{CPMCK} + 1.7$	—	ns
SPI_{13}	NPCS0 hold after SPCK falling	—	$1.5 \times t_{CPMCK} + 1.0$	—	ns
SPI_{14}	NPCS0 setup to SPCK falling	—	$2.5 \times t_{CPMCK} + 1.5$	—	ns
SPI_{15}	NPCS0 hold after SPCK rising	—	$1.5 \times t_{CPMCK} + 0.5$	—	ns

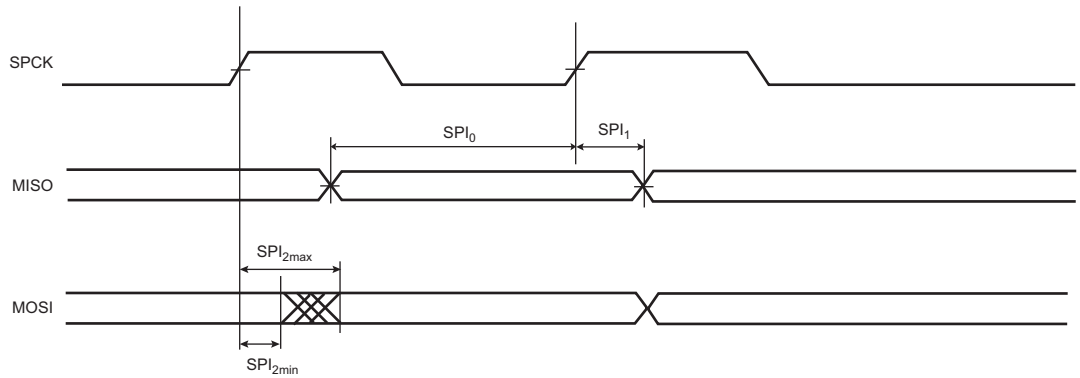
Note: 1. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access timing is the time between the SPCK rising or falling edge and the signal stabilizes. Figure 55-29 illustrates minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 55-53. USART SPI Timings 1.8V Peripheral Supply (USART3 and USART4 only)

Symbol	Parameter	Conditions	Min	Max	Unit
Master Mode					
SPI_0	SPCK frequency	—	MCK/6	—	Hz
SPI_1	Input Data Setup Time	—	$0.5 \times t_{CPMCK} + 3.9$	—	ns
SPI_2	Input Data Hold Time	—	$1.5 \times t_{CPMCK} + 1.4$	—	ns
SPI_3	Chip Select Active to Serial Clock	—	—	$1.5 \times t_{SPCK} - 1.7$	ns
SPI_4	Output Data Setup Time	—	0	11.7	ns
SPI_5	Serial Clock to Chip Select Inactive	—	—	$t_{SPCK} + 2.3$	ns
Slave Mode					
SPI_6	SPCK falling to MISO	—	3.0(1)	13.0(1)	ns
SPI_7	MOSI Setup time before SPCK rises	—	—	$2 \times t_{CPMCK} + 3.1$	ns
SPI_8	MOSI Hold time after SPCK rises	—	1.1	—	ns
SPI_9	SPCK rising to MISO	—	2.8(1)	12.5(1)	ns
SPI_{10}	MOSI Setup time before SPCK falls	—	2.9	—	ns
SPI_{11}	MOSI Hold time after SPCK falls	—	0.7	—	ns
SPI_{12}	NPCS0 setup to SPCK rising	—	$2.5 \times t_{CPMCK} + 1.5$	—	ns
SPI_{13}	NPCS0 hold after SPCK falling	—	$1.5 \times t_{CPMCK} + 0.9$	—	ns
SPI_{14}	NPCS0 setup to SPCK falling	—	$2.5 \times t_{CPMCK} + 1.3$	—	ns
SPI_{15}	NPCS0 hold after SPCK rising	—	$1.5 \times t_{CPMCK} + 0.4$	—	ns

Note: 1. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access timing is the time between the SPCK rising or falling edge and the signal stabilizes. Figure 55-29 illustrates minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Figure 55-29. Minimum and Maximum Access Time for USART SPI Output Signal



55.22 Two-wire Interface Characteristics

Figure 55-30. Two-wire Serial Bus Timing

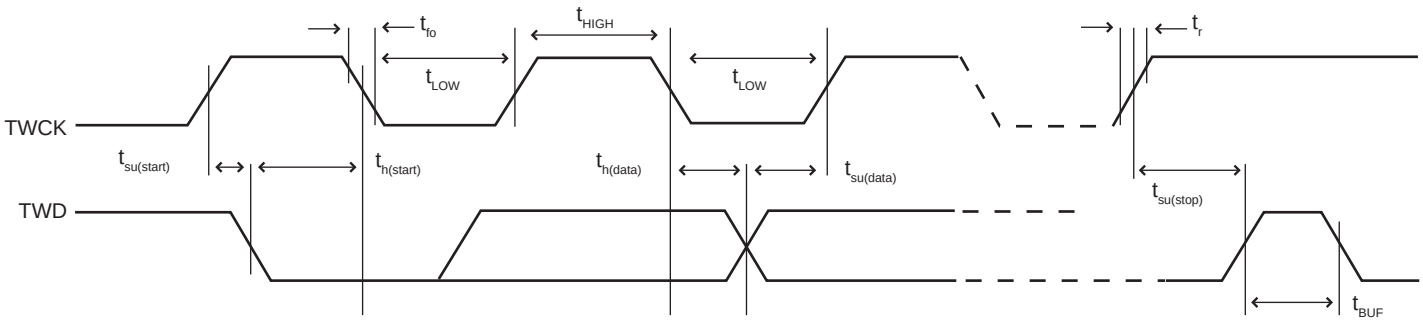


Table 55-54 describes the requirements for devices connected to the Two-wire Serial Bus.

Table 55-54. Two-wire Serial Bus Requirements

Symbol	Parameter	Conditions	Min	Max	Unit
V_{IL}	Input Low-voltage	—	-0.3	$0.3 \times V_{DDIO}$	V
V_{IH}	Input High-voltage	—	$0.7 \times V_{DDIO}$	V_{CC} + 0.3	V
V_{hys}	Hysteresis of Schmitt Trigger Inputs	—	0.150	—	V
V_{OL}	Output Low-voltage	3 mA sink current	—	0.4	V
t_r	Rise Time for both TWD and TWCK		$20 + 0.1C_b$ (2)	300	ns
t_{fo}	Output Fall Time from V_{IHmin} to V_{ILmax}	10 pF < C_b < 400 pF Figure 55-30	$20 + 0.1C_b$ (2)	250	ns
C_i (1)	Capacitance for each I/O Pin	—	—	10	pF
f_{TWCK}	TWCK Clock Frequency	—	0	400	kHz
Rp	Value of Pullup Resistor	$f_{TWCK} \le 100$ kHz	$(V_{DDIO}$ - 0.4V) ÷ 3mA	1000ns ÷ C_b	Ω
		$f_{TWCK} > 100$ kHz	$(V_{DDIO}$ - 0.4V) ÷ 3mA	300ns ÷ C_b	Ω
t_{LOW}	Low Period of the TWCK Clock	$f_{TWCK} \le 100$ kHz	(3)	—	µs
		$f_{TWCK} > 100$ kHz	(3)	—	µs
t_{HIGH}	High Period of the TWCK Clock	$f_{TWCK} \le 100$ kHz	(4)	—	µs
		$f_{TWCK} > 100$ kHz	(4)	—	µs
$t_{h(start)}$	Hold Time (repeated) START condition	$f_{TWCK} \le 100$ kHz	t_{HIGH}	—	µs
		$f_{TWCK} > 100$ kHz	t_{HIGH}	—	µs
$t_{su(start)}$	Setup Time for a Repeated START condition	$f_{TWCK} \le 100$ kHz	t_{HIGH}	—	µs
		$f_{TWCK} > 100$ kHz	t_{HIGH}	—	µs
$t_{h(data)}$	Data Hold Time	$f_{TWCK} \le 100$ kHz	0	(HOLD + 3) × $t_{peripheral\ clock}$	µs
		$f_{TWCK} > 100$ kHz	0	(HOLD + 3) × $t_{peripheral\ clock}$	µs
$t_{su(data)}$	Data Setup Time	$f_{TWCK} \le 100$ kHz	t_{LOW} - (HOLD + 3) × $t_{peripheral\ clock}$	—	ns
		$f_{TWCK} > 100$ kHz	t_{LOW} - (HOLD + 3) × $t_{peripheral\ clock}$	—	ns

Table 55-54. Two-wire Serial Bus Requirements (Continued)

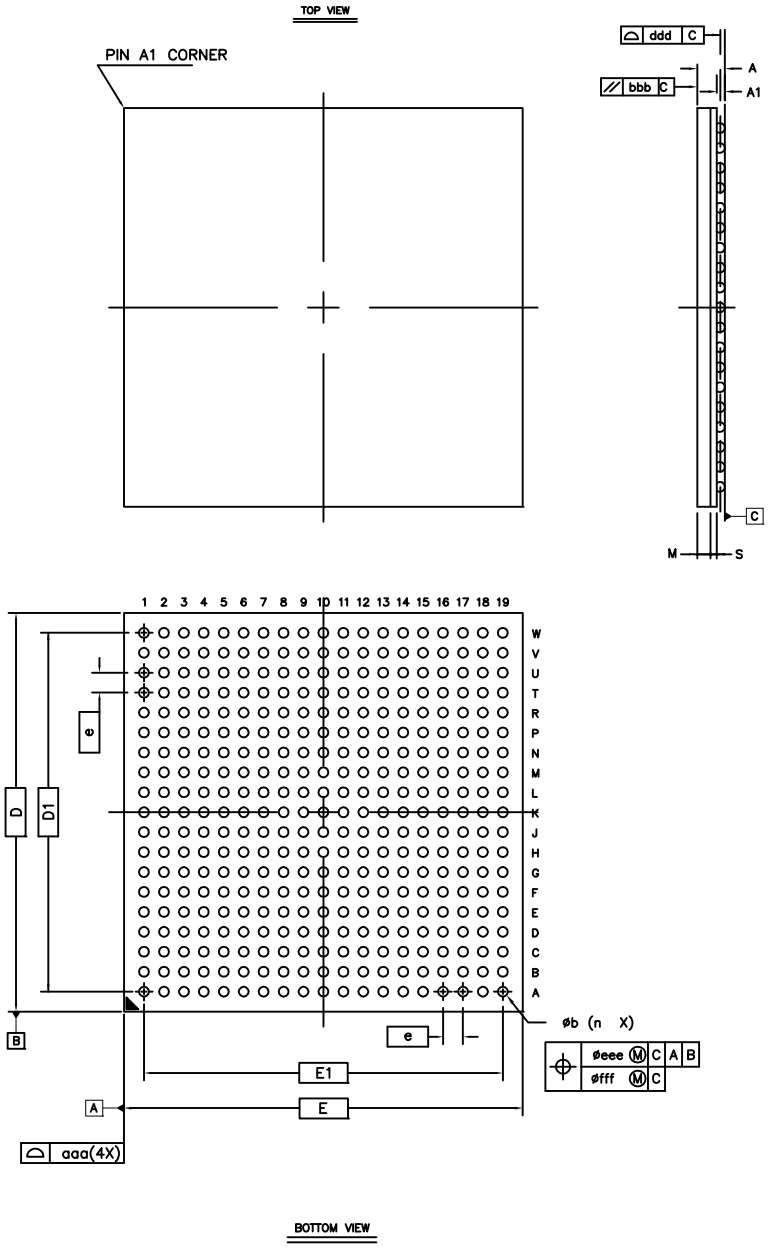
Symbol	Parameter	Conditions	Min	Max	Unit
$t_{su(stop)}$	Setup time for STOP condition	$f_{TWCK} \leq 100$ kHz	t_{HIGH}	—	µs
		$f_{TWCK} > 100$ kHz	t_{HIGH}	—	µs
t_{BUF}	Bus free time between a STOP and START condition	$f_{TWCK} \leq 100$ kHz	t_{LOW}	—	µs
		$f_{TWCK} > 100$ kHz	t_{LOW}	—	µs

Notes:
1. Required only for $f_{TWCK} > 100$ kHz
2. C_b = capacitance of one bus line in pF. Per I2C Standard, C_b Max = 400 pF
3. The TWCK low period is defined as follows: $t_{LOW} = ((CLDIV \times 2^{CKDIV}) + 4) \times t_{MCK}$
4. The TWCK high period is defined as follows: $t_{HIGH} = ((CHDIV \times 2^{CKDIV}) + 4) \times t_{MCK}$

56. Mechanical Characteristics

56.1 361-ball TFBGA Mechanical Characteristics

Figure 56-1. 361-ball TFBGA Package Drawing



		Symbol	Common Dimensions		
			MIN.	NOM.	MAX.
Package :			TFBGA		
Body Size:	X	E	16.000		
	Y	D	16.000		
Ball Pitch :		e	0.800		
Total Thickness :		A			1.200
Mold Thickness :		M	0.530	Ref.	
Substrate Thickness :		S	0.260	Ref.	
Ball Diameter :			0.400		
Stand Off :		A1	0.270	–	0.370
Ball Width :		b	0.380	–	0.480
Package Edge Tolerance :		aaa	0.150		
Mold Flatness :		bbb	0.200		
Coplanarity:		ddd	0.150		
Ball Offset (Package) :		eee	0.150		
Ball Offset (Ball) :		fff	0.080		
Ball Count :		n	361		
Edge Ball Center to Center :	X	E1	14.400		
	Y	D1	14.400		

Atmel	TITLE	Thin Fine Pitch Ball Grid Array		GPC:	C E P
	Pins:	361		Drawing No.:	R–TFBGA361_A
	Body:	16x16x1.2	mm	REV.:	A
	Ball Pitch:	0.8	mm	Date:	4/3/2013
	Ball Diameter:	0.4	mm	Jedec Code:	MO–275–LLAC–1

Table 56-1. 361-ball TFBGA Package Characteristics

Moisture Sensitivity Level	3

Table 56-2. Device and 361-ball TFBGA Package Maximum Weight

490	mg

Table 56-3. Package Reference

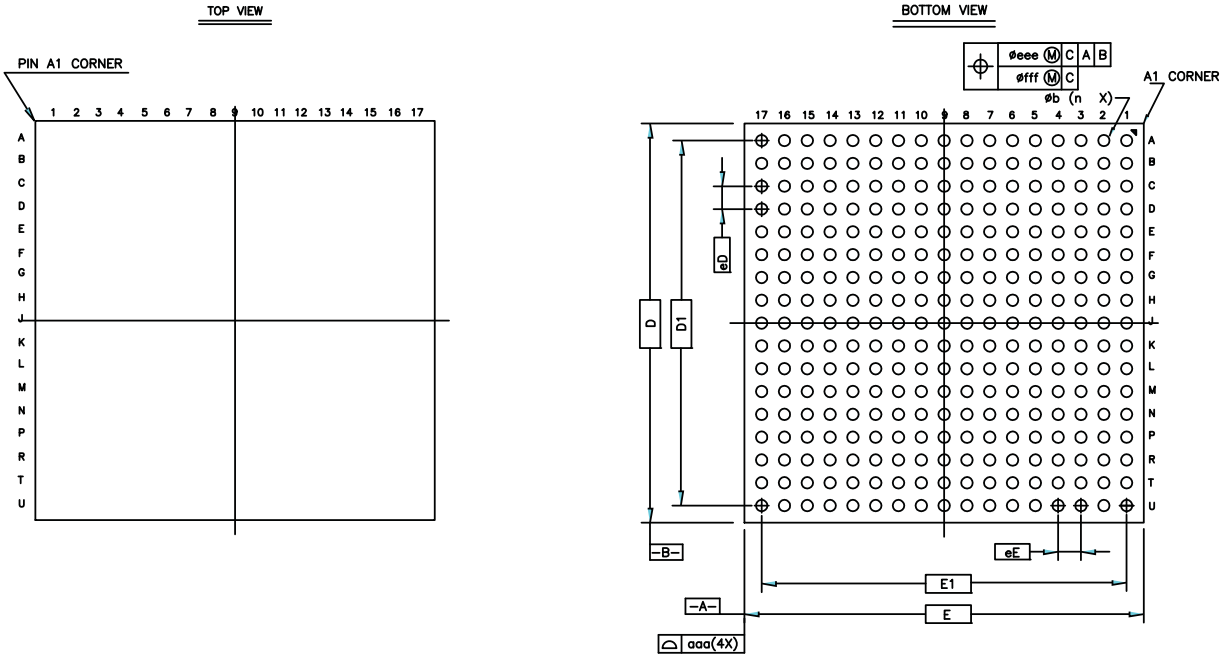
JEDEC Drawing Reference	MO-275-LLAC-1
J-STD-609 Classification	e8

Table 56-4. Package Information

Ball Land	0.45 mm ± 0.05
Nominal Ball Diameter	0.4 mm
Solder Mask Opening	0.35 mm ± 0.05
Solder Mask Definition	SMD
Solder	LF35

56.2 289-ball LFBGA Mechanical Characteristics

Figure 56-2. 289-ball LFBGA Package Drawing



		Symbol	Common Dimensions
Package :			LFBGA
Body Size:	X	E	14.000
	Y	D	14.000
Ball Pitch :	X	eE	0.800
	Y	eD	0.800
Total Thickness :		A	1.400 MAX.
Mold Thickness :		M	0.530 Ref.
Substrate Thickness :		S	0.360 Ref.
Ball Diameter :			0.400
Stand Off :		A1	0.270~0.370
Width :		b	0.380~0.480
Package Edge Tolerance :		aaa	0.150
Mold Flatness :		bbb	0.200
Coplanarity:		ddd	0.120
Ball Offset (Package) :		eee	0.150
Ball Offset (Ball) :		fff	0.080
Ball Count :		n	289
Edge Ball Center to Center :	X	E1	12.800
	Y	D1	12.800

11-Jan-2012

	TITLE	GPC	DRAWING NO.	REV
Atmel Package Drawing Contact: Packagedrawings@atmel.com	Low Profile Fine Pitch Ball Grid Array, 289 Balls Body : 14 x 14 x 1.4 mm Ball Pitch :0.8 mm Ball Diameter :0.4 mm	- - -	R-LFBGA289_A	A

Table 56-5. 289-ball LFBGA Package Characteristics

Moisture Sensitivity Level	3

Table 56-6. Device and 289-ball LFBGA Package Maximum Weight

450	mg

Table 56-7. Package Reference

JEDEC Drawing Reference	MO-275-JJAC-1
J-STD-609 Classification	e8

Table 56-8. Package Information

Ball Land	0.45 mm ± 0.05
Nominal Ball Diameter	0.4 mm
Solder Mask Opening	0.35 mm ± 0.05
Solder Mask Definition	SMD
Solder	LF35

57. Schematic Checklist

The schematic checklist provides the user with the requirements regarding the different pin connections that must be considered before starting any new board design. It also provides information on the minimum hardware resources required to quickly develop an application with the SAMA5D4. It does not consider PCB layout constraints.

It also provides recommendations regarding low-power design constraints to minimize power consumption.

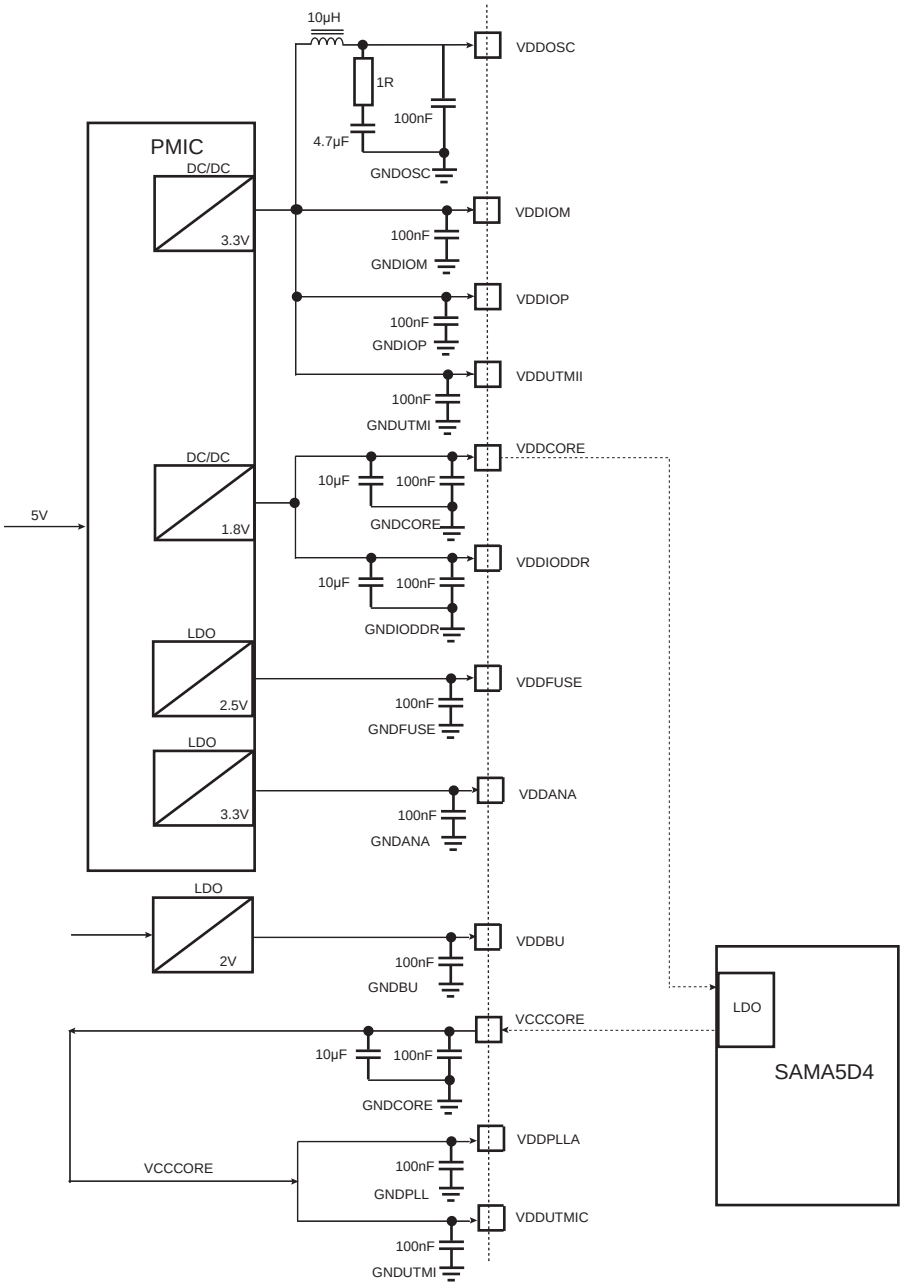
This information is not intended to be exhaustive. Its objective is to cover as many configurations of use as possible.

The checklist contains a column for use by designers, making it easy to track and verify each line item.

57.1 Power Supply

CAUTION: The board design must comply with the powerup and powerdown sequence guidelines provided in the datasheet to guarantee reliable operation of the device.

Figure 57-1. 1.2V, 1.8V, 2V, 2.5V, 3.3V Power Supplies Schematics (1)



Note: 1. These values are given only as a typical example.

Table 57-1. Power Supply Connections

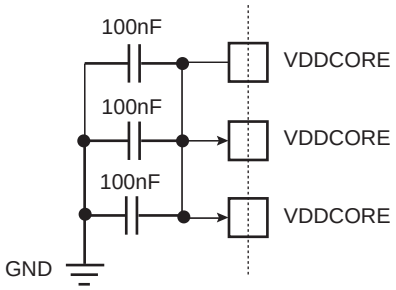
	Signal Name	Recommended Pin Connection	Description
	VDDCORE	1.62 V to 1.98 V Decoupling/Filtering capacitors (10 µF and 100 nF)[1][2]	Powers the regulator that generates core power supply on VCCCORE. Must be established after VDDIOP and VDDANA. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop. Supply ripple must not exceed 20 mVrms.
	VCCCORE	1.16 V to 1.32 V Decoupling/Filtering capacitors (10 µF and 100 nF)[1][2]	VCCCORE is supplied by the SAMA5D4 internal regulator and powers the internal logic.
	VDDIODDR	1.70 V to 1.90 V or 1.14 V to 1.30 V Decoupling/Filtering capacitors (10 µF and 100 nF)[1][2]	Powers the DDR2 Interface I/O lines. or Powers the LPDDR2 Interface I/O lines. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.
	VDDIOM	1.65 V to 1.95 V or 3.0 V to 3.6 V Decoupling capacitor (100 nF)[1][2]	Powers the NAND and HSMC Interface I/O lines. Dual voltage range is supported. The I/O drives are selected by programming the DRIVE0 and DRIVE1 fields in the SFR_EBICFG register. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.
	VDDIOP[3]	3.0 V to 3.6 V Decoupling capacitors (100 nF)[1][2]	Powers the peripherals I/O lines. Must be established prior to VDDCORE. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.
	VDDBU	1.88 V to 2.12 V Decoupling capacitor (100 nF)[1][2]	Powers the Slow Clock oscillator, the internal 64 kHz RC and a part of the System Controller. Must be established first. Supply ripple must not exceed 30 mVrms.
	VDDUTMIC	1.1 V to 1.32 V Decoupling capacitors (100 nF)[1][2]	Powers the USB device and host UTMI+ core and the UTMI PLL. Must be connected to VCCCORE. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.
	VDDUTMII	3.0 V to 3.6 V Decoupling capacitor (100 nF)[1][2]	Powers the USB device and host UTMI+ interface. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.
	VDDPLLA	1.1 V to 1.32 V Decoupling capacitor (100 nF)[1][2]	Powers the PLLA cell. Must be connected to VCCCORE. The VDDPLLA power supply pin draws small current, but it is noise sensitive. Care must be taken in VDDPLLA power supply routing, decoupling and also on bypass capacitors. Supply ripple must not exceed 10 mVrms.

Table 57-1. Power Supply Connections (Continued)

	Signal Name	Recommended Pin Connection	Description
	VDDOSC	3.0 V to 3.6 V Decoupling/Filtering RLC circuit[1]	Powers the main oscillator cell. The VDDOSC power supply pin is noise-sensitive. Care must be taken in VDDOSC power supply routing, decoupling and also on bypass capacitors. Supply ripple must not exceed 30 mVrms.
	VDDANA[3]	3.0 V to 3.6 V Decoupling capacitor (100 nF)[1][2] Application-dependent	Powers the analog parts. Must rise at same time as VDDIOP. Can be connected to VDDIOP with filtering.
	VDDFUSE	2.25 V to 2.75 V Decoupling capacitor (100 nF)[1][2]	Powers the fuse box for programming. VDDFUSE must not be left floating.
	GNDCORE	Core Chip ground	GNDCORE pins are common to VDDCORE and VCCCORE pins. GNDCORE pins should be connected as shortly as possible to the system ground plane.
	GNDIODDR	DDR2/LPDDR/LPDDR2 interface I/O lines ground	GNDIODDR pins should be connected as shortly as possible to the system ground plane.
	GNDIOM	NAND and HSMC Interface I/O lines ground	GNDIOM pins should be connected as shortly as possible to the system ground plane.
	GNDIOP	Peripherals and ISI I/O lines ground	GNDIOP pins are common to VDDIOP pins. GNDIOP pins should be connected as shortly as possible to the system ground plane.
	GNDBU	Backup ground	GNDBU pin is provided for VDDBU pins. GNDBU pin should be connected as shortly as possible to the system ground plane.
	GNDUTMI	UDPHS and UHPHS UTMI+ Core and interface ground	GNDUTMI pins are common to VDDUTMII and VDDUTMIC pins. GNDUTMI pins should be connected as shortly as possible to the system ground plane.
	GNDPLL	PLLA cell ground	GNDPLL pin is provided for VDDPLLA pins. GNDPLL pin should be connected as shortly as possible to the system ground plane.
	GNDOSC	PLLUTMI and Oscillator ground	GNDOSC pin is provided for VDDOSC pins. GNDOSC pin should be connected as shortly as possible to the system ground plane.
	GNDANA	Analog ground	GNDANA pins are common to VDDANA pins. GNDANA pins should be connected as shortly as possible to the system ground plane.
	GNDFUSE	Fuse box ground	GNDFUSE pins are common to VDDFUSE pins. GNDFUSE pins should be connected as shortly as possible to the system ground plane.

Note: 1. These values are given only as a typical example.

2. Decoupling capacitors must be connected as close as possible to the microprocessor and on each corresponding pin.



3. Must rise at the same time as VDDIOP. This specific power sequence ensures a reliable operation of the device.

57.2 Clock, Oscillator and PLL

Table 57-2. Clock, Oscillator and PLL Connections

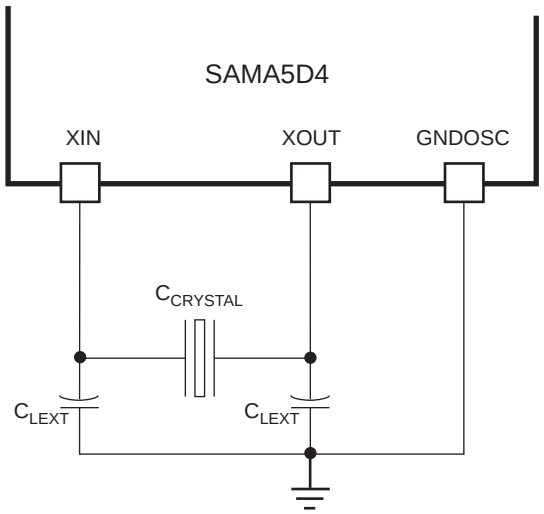
	Signal Name	Recommended Pin Connection	Description
	XIN XOUT 12 MHz Main Oscillator in Normal Mode	Crystals between 8 and 16 MHz USB High Speed (not Full Speed) Host and Device peripherals need a 12 MHz clock. Capacitors on XIN and XOUT (Crystal Load Capacitance-dependent)	Crystal Load Capacitance to check ($C_{CRYSTAL}$).  Example: for a 12 MHz crystal with a load capacitance of $C_{CRYSTAL}$ = 15 pF, external capacitors are required: C_{LEXT} = 22 pF. Refer to Section 55. “Electrical Characteristics”.
	XIN XOUT 12 MHz Main Oscillator in Bypass Mode	XIN: external clock source XOUT: can be left unconnected USB High speed (not Full Speed) Host and Device peripherals need a 12 MHz clock.	VDDOSC square wave signal External clock source up to 50 MHz Duty Cycle: 40 to 60% Refer to Section 55. “Electrical Characteristics”.

Table 57-2. Clock, Oscillator and PLL Connections (Continued)

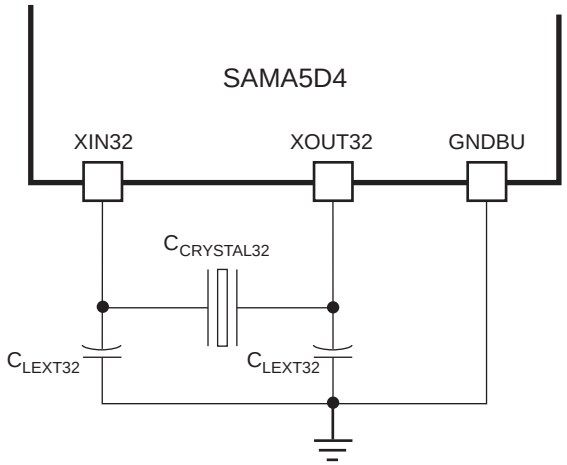
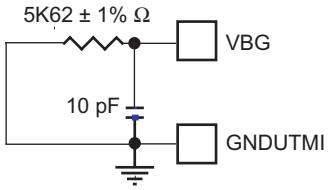
	Signal Name	Recommended Pin Connection	Description
	XIN XOUT 12 MHz Main Oscillator Disabled	XIN: can be left unconnected XOUT: can be left unconnected USB High Speed (not Full Speed) Host and Device peripherals need a 12 MHz clock.	 Typical nominal frequency 12 MHz (Internal 12 MHz RC Oscillator) Duty Cycle: 45 to 55% Refer to Section 55. “Electrical Characteristics”.
	XIN32 XOUT32 Slow Clock Oscillator	32.768 kHz Crystal Capacitors on XIN32 and XOUT32 (Crystal Load Capacitance-dependent)	Crystal load capacitance to check ($C_{CRYSTAL32}$).  Example: for a 32.768 kHz crystal with a load capacitance of $C_{CRYSTAL32}$ = 12.5 pF, external capacitors are required: C_{LEXT32} = 19 pF. Refer to Section 55. “Electrical Characteristics”.
	XIN32 XOUT32 Slow Clock Oscillator in Bypass Mode	XIN32: external clock source XOUT32: can be left unconnected	VDDBU square wave signal External clock source up to 44 kHz Duty Cycle: 40 to 60% Refer to Section 55. “Electrical Characteristics”.
	XIN32 XOUT32 Slow Clock Oscillator Disabled	XIN32: can be left unconnected XOUT32: can be left unconnected	Typical nominal frequency 32 kHz (internal 32 kHz RC oscillator) Duty Cycle: 45 to 55% Refer to Section 55. “Electrical Characteristics”.

Table 57-2. Clock, Oscillator and PLL Connections (Continued)

	Signal Name	Recommended Pin Connection	Description
	VBG	0.9–1.1V[2]	Bias Voltage Reference for USB To minimize noise on the VBG pin, it is recommended to configure the following layout: - VBG path as short as possible - ground connection to GNDUTMI  VBG can be left unconnected if USB is not used. Refer to Section 2. “Signal Description”.

57.3 ICE and JTAG

Table 57-3. ICE and JTAG Connections[1]

	Signal Name	Recommended Pin Connection	Description
	TCK	Pullup (100 kΩ)[2]	This pin is a Schmitt trigger input. Internal pullup resistor to V_{DDIOP} (100 kΩ).
	TMS	Pullup (100 kΩ)[2]	This pin is a Schmitt trigger input. Internal pullup resistor to V_{DDIOP} (100 kΩ).
	TDI	Pullup (100 kΩ)[2]	This pin is a Schmitt trigger input. Internal pullup resistor to V_{DDIOP} (100 kΩ).
	TDO	Floating	Output driven at up to V_{DDIOP}.
	NTRST	Refer to the pin description section.	This pin is a Schmitt trigger input. Internal pullup resistor to V_{DDIOP} (100 kΩ).
	JTAGSEL	In harsh environments[3], it is strongly recommended to tie this pin to GNDBU if not used or to add an external low-value resistor (such as 1 kΩ).	Internal pulldown resistor to GNDBU (15 kΩ). Must be tied to V_{DDBU} to enter JTAG Boundary Scan.

Notes:
1. It is recommended to establish accessibility to a JTAG connector for debug in any case.
2. These values are given only as a typical example.
3. In a well-shielded environment subject to low magnetic and electric field interference, the pin may be left unconnected. In noisy environments, a connection to ground is recommended.

57.4 Reset and Test

Table 57-4. Reset and Test Connections

	Signal Name	Recommended Pin Connection	Description
	NRST	Application-dependent. Can be connected to a push button for hardware reset.	NRST pin is a Schmitt trigger input. No internal pullup resistor.
	TST	In harsh environments[1], it is strongly recommended to tie this pin to GNDBU to add an external low-value resistor (such as 10 kΩ).	This pin is a Schmitt trigger input. Internal pulldown resistor to GNDBU (15 kΩ).

Note:
1. In a well-shielded environment subject to low magnetic and electric field interference, the pin may be left unconnected. In noisy environments, a connection to ground is recommended.

57.5 Shutdown/Wakeup Logic

Table 57-5. Shutdown/Wakeup Logic Connections

	Signal Name	Recommended Pin Connection	Description
	SHDN	Application-dependent. A typical application connects the pin SHDN to the shutdown input of the DC/DC Converter providing the main power supplies.	This pin is a push-pull output. SHDN pin is driven low to GNDBU by the Shutdown Controller (SHDWC).
	WKUP	0 V to V_{DDBU}	This pin is an input-only. WKUP behavior can be configured through the Shutdown Controller (SHDWC).

57.6 Parallel Input/Output (PIO)

Table 57-6. PIO Connections

	Signal Name	Recommended Pin Connection	Description
	PAx PBx PCx PDx PEx	Application-dependent.	All PIOs are pulled-up inputs (100 kΩ) at reset except those which are multiplexed with the Address Bus signals that require to be enabled as peripherals: In Section 3. “Package and Pinout”, refer to the column ‘Reset State’ of the Pin Description table. Schmitt trigger on all inputs. To reduce power consumption if not used, the concerned PIO can be configured as an output, driven at ‘0’ with internal pullup disabled.

57.7 Analog-to-Digital Converter (ADC)

Table 57-7. ADC Connections

	Signal Name	Recommended Pin Connection	Description
	ADVREF	3.3 V to VDDANA Decoupling/filtering capacitors. Application-dependent.	ADVREF is a pure analog input. To reduce power consumption if the ADC is not used, connect ADVREF to GNDANA.

57.8 External Bus Interface (EBI)

Table 57-8. EBI Connections

	Signal Name	Recommended Pin Connection	Description
	D0–D15	Application-dependent.	Data Bus (D0 to D15) All data lines are pulled-up inputs to V_{DDIOM} at reset.
	A0–A25	Application-dependent.	Address Bus (A0 to A25) All address lines are driven to ‘0’ at reset.

Table 57-9 and Table 57-10 detail the connections to be applied between the EBI pins and the external devices for each Memory Controller.

Table 57-9. EBI Pins and External Static Devices Connections

Signals: EBI_	Pins of the Interfaced Device		
	8-bit Static Device	2 x 8-bit Static Devices	16-bit Static Device
Controller	**SMC (Static Memory Controller)**		
D0–D7	D0–D7	D0–D7	D0–D7
D8–D15	–	D8–D15	D8–D15
A0/NBS0	A0	–	NLB
A1	A1	A0	A0

Table 57-9. EBI Pins and External Static Devices Connections

Signals: EBI_	Pins of the Interfaced Device		
	8-bit Static Device	2 x 8-bit Static Devices	16-bit Static Device
Controller	**SMC (Static Memory Controller)**		
A2–A22	A[2:22]	A[1:21]	A[1:21]
A23–A25	A[23:25]	A[22:24]	A[22:24]
NCS0	CS	CS	CS
NCS1	CS	CS	CS
NCS2	CS	CS	CS
NCS3/NANDCS	CS	CS	CS
NRD/NANDOE	OE	OE	OE
NWE/NWR0/NANDWE	WE	WE(1)	WE
NWR1/NBS1	–	WE(1)	NUB

Note: 1. NWR0 enables lower byte writes. NWR1 enables upper byte writes.

Table 57-10. EBI Pins and NAND Flash Device Connections

Signals: EBI_	Pins of the Interfaced Device	
	8-bit NAND Flash	16-bit NAND Flash
Controller	**NFC (NAND Flash Controller)**	
D0–D7	NFD0–NFD7	NFD0–NFD7
D8–D15	–	NFD8–NFD15
A21/NANDALE	ALE	ALE
A22/NANDCLE	CLE	CLE
NRD/NANDOE	RE	RE
NWE/NWR0/NANDWE	WE	WE
NCS3/NANDCS	CE	CE
NANDRDY	R/$\overline{B}$	R/$\overline{B}$
A0/NBS0	–	–
A1–A20	–	–
A23–A25	–	–
NWR1/NBS1	–	–
NCS0	–	–
NCS1	–	–
NCS2	–	–
NWAIT	–	–

57.9 DDR2 Bus Interface

Table 57-11. DDR2 I/O Lines Usage vs Operating Modes

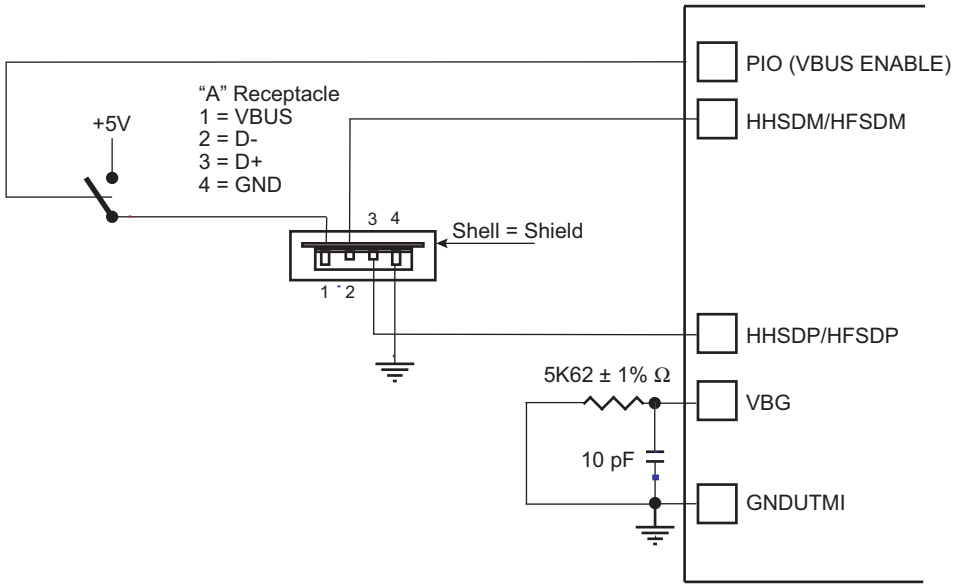
Signal Name	DDR2 Mode	LPDDR2 Mode	LPDDR
Controller	**MPDDRC (Multi-port DDR-SDRAM Controller)**		
DDR_VREF	VDDIODDR/2	VDDIODDR/2	VDDIODDR/2
DDR_CALP	GND via 200Ω resistor	GND via 240Ω resistor	GND via 200Ω resistor
DDR_CALN	VDDIODDR via 200Ω resistor	VDDIODDR via 240Ω resistor	VDDIODDR via 200Ω resistor
DDR_CK, DDR_CKN	CLK and CLKN	CLK and CLKN	CLK and CLKN
DDR_CKE	CLKE	CLKE	CLKE
DDR_CS	CS	CS	CS
DDR_BA[2..0]	BA[2..0]	BA[2..0]	BA[2..0]
DDR_WE	WE	CA2	WE
DDR_RAS–DDR_CAS	RAS, CAS	CA0, CA1	RAS, CAS
DDR_A[13..0]	A[13:0]	CAx, with x>2	A[13:0]
DDR_D[31..0]	D[31:0]	D[31:0]	D[31:0]
DQS[3..0], DQSN[3..0]	DQS[3:0] DQSN[3:0]	DQS[3:0] DQSN[3:0]	DQS[3:0] DQSN connected to DDR_VREF
DQM[3..0]	DQM[3..0]	DQM[3..0]	DQM[3..0]

57.10 USB High-Speed Host Port (UHPHS)/USB High-Speed Device Port (UDPHS)

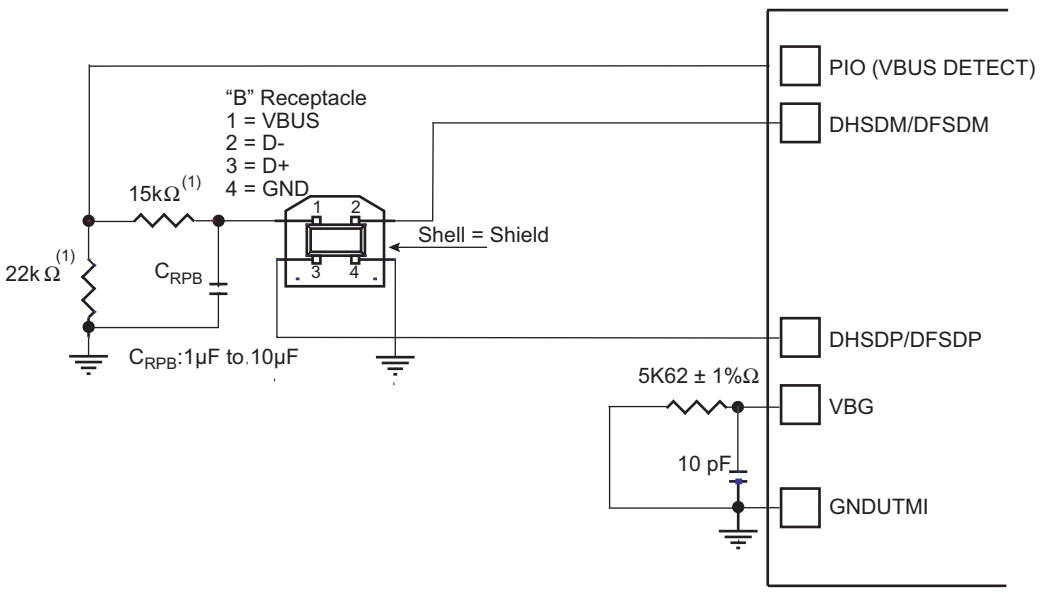
Table 57-12. UHPHS/UDPHS Connections

	Signal Name	Recommended Pin Connection	Description
	HHSDPA/DHSDP(1) HHSDMA/DHSDM(1)	Application-dependent(2)(3)	Pulldown output at reset.
	HHSDPB/HHSDMB	Application-dependent(2)	Pulldown output at reset.
	HHSDPC/HHSDMC	Application-dependent(2)	Pulldown output at reset.

Notes:
1. UDPHS shares Port A with UHPHS.
2. Example of USB High Speed Host connection:
   Refer to Section 35. “USB Host High Speed Port (UHPHS)”.



3. Typical USB High Speed Device connection:
   Refer to Section 34. “USB High Speed Device Port (UDPHS)”.



(1) The values shown on the 22 kΩ and 15 kΩ resistors are only valid with 3.3V supplied PIOs.

57.11 Boot Program Hardware Constraints

Refer to Section 12. “Standard Boot Strategies” for more details on the boot program.

57.11.1 Boot Program Supported Crystals (MHz)

A 12 MHz crystal or external clock (in bypass mode) is mandatory in order to generate USB and PLL clocks correctly for the following boots.

57.11.2 NAND Flash Boot

Boot is possible if the first page contains a valid header or if it is ONFI compliant. For more details, refer to Section 12.4.4.1 “NAND Flash Boot: NAND Flash Detection”.

Table 57-13. Pins Driven during NAND Flash Boot Program Execution

Peripheral	Pin	PIO Line
EBI CS3 SMC	NANDOE	PC13
EBI CS3 SMC	NANDWE	PC14
EBI CS3 SMC	NANDCS	PC15
EBI CS3 SMC	NANDRDY	PC16
EBI CS3 SMC	NANDALE	PC17
EBI CS3 SMC	NANDCLE	PC18
EBI CS3 SMC	Cmd/Addr/Data	–

57.11.3 SD Card Boot

SD card boot supports all SD card memories compliant with SD Memory Card Specification V2.0. This includes SDHC cards.

Table 57-14. Pins Driven During SD Card Boot Program Execution

Peripheral	Pin	PIO Line
MCI1	MCI1_CK	PE18
MCI1	MCI1_CDA	PE19
MCI1	MCI1_D0	PE20
MCI1	MCI1_D1	PE21
MCI1	MCI1_D2	PE22
MCI1	MCI1_D3	PE23

Note: The MCI1 pins are pulldown by default (typical value 70 kΩ). To function correctly, the signals require an external pullup resistor and the internal pulldown resistor must be disabled.

57.11.4 Serial and DataFlash Boot

Two kinds of SPI Flash are supported: SPI Serial Flash and SPI DataFlash.

The SPI Flash bootloader tries to boot on SPI0 Chip Select 0, first looking for SPI Serial Flash, and then for SPI DataFlash.

The SPI Flash Boot program supports:

- All SPI Serial Flash devices
- All DataFlash devices

Table 57-15. Pins Driven During Serial or DataFlash Boot Program Execution

Peripheral	Pin	PIO Line
SPI0	MOSI	PC1
SPI0	MISO	PC0
SPI0	SPCK	PC2
SPI0	NPCS0	PC3
SPI0	NPCS1	PC4

57.11.5 TWI EEPROM Boot

The TWI EEPROM Flash boot program searches for a valid application in an EEPROM memory.

TWI EEPROM boot supports all I^2C-compatible EEPROM memories using 7-bit device (address 0x50).

Table 57-16. Pins Driven During TWI EEPROM Boot Program Execution

Peripheral	Pin	PIO Line
TWI0	TWD0	PA30
TWI0	TWCK0	PA31

57.11.6 SAM-BA Boot

The SAM-BA Boot Assistant supports serial communication via the DBGU or the USB Device Port.

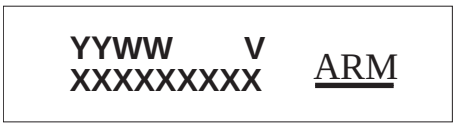
Table 57-17. Pins Driven During SAM-BA Boot Program Execution

Peripheral	Pin	PIO Line
DBGU	DRXD	PB24
DBGU	DTXD	PB25

58. Marking

All devices are marked with the Atmel logo and the ordering code.

Additional marking may be in one of the following formats:



where

- “YY”: manufactory year
- “WW”: manufactory week
- “V”: revision
- “XXXXXXXXX”: lot number

59. Ordering Information

Table 59-1. Ordering Information

Ordering Code	Package	Carrier Type	Package Type	Operating Temperature Range
ATSAMA5D44A-CU	TFBGA361	Tray	Green	Industrial -40°C to 85°C
ATSAMA5D44A-CUR		Tape and Reel		
ATSAMA5D43A-CU	LFBGA289	Tray		
ATSAMA5D43A-CUR		Tape and Reel		
ATSAMA5D42A-CU	TFBGA361	Tray		
ATSAMA5D42A-CUR		Tape and Reel		
ATSAMA5D41A-CU	LFBGA289	Tray		
ATSAMA5D41A-CUR		Tape and Reel		

60. Errata

These errata provide information on SAMA5D4 series Rev. A parts.

60.1 Standard Boot Strategies

60.1.1 Boot ROM: Xmodem is Not Working

Due to a bug in the ROM code, Xmodem is not working.

Problem Fix/Workaround

None.

60.1.2 Boot ROM: Boot on MCI0 is Not Working

Boot on MCI0 is not working.

Problem Fix/Workaround

Use a different boot media.

60.2 LCD Controller (LCDC)

60.2.1 LCDC: LCDC PWM Is Not Usable with DIV_1

When field PWMPS in register LCDC_LCDCFG6 is set to DIV_1 value, the PWM output is stuck.

The LCDC PWM works correctly for the dividers DIV_2 up to DIV_64.

Problem Fix/Workaround

None.

60.3 ADC (Analog-to-Digital Controller)

60.3.1 ADC: Compare Window limitations in user sequence

When USEQ = 1 in ADC_MR and CMPALL = 0 in ADC_EMR, the compare window function does not work for all sequences programmed into ADC_SEQR1 and ADC_SEQR2.

Problem Fix/Workaround

To have a correct compare window function on a given channel X selected by means of CMPSEL = X in ADC_EMR, the ADC_CHSR[X] must be set.

For example, if ADC_CHSR = 0x0000_000F and ADC_SEQ1R = 0x0000_3300, both channel index 0 and 3 can be used for comparison (i.e., CMPSEL = 0 and CMPSEL = 3 can be used, COMPE flag will behave correctly in ADC_ISR).

If ADC_CHSR = 0x0000_000F and ADC_SEQ1R = 0x0000_1100, both channel index 0 and 1 can be used for comparison (i.e., CMPSEL = 0 and CMPSEL = 1 can be used, COMPE flag will behave correctly in ADC_ISR).

If ADC_CHSR = 0x0000_000F and ADC_SEQ1R = 0x0000_4400, only channel index 0 can be used for comparison (i.e., CMPSEL = 0 can be used, COMPE flag will behave correctly in ADC_ISR).

Channel 4 cannot be used for comparison because ADC_CHSR[4] = 0.

60.3.2 ADC: Differential mode limitations in user sequence

If Differential mode is used with user sequence, the analog pad configuration is not the one expected.

For instance, if USCH4 is set to channel 2, the user must configure DIFF2 to enable differential mode on channel 2 (this is the expected behavior). However, due to this bug the ADC Controller will look at the DIFF4 state instead the DIFF2 state.

Problem Fix/Workaround

Set the differential mode on the DIFFx bit corresponding to the related sequence number instead of setting the differential mode on the DIFFx bit of the related channel number.

60.3.3 ADC: Calibration issue after Sleep Mode

Calibration does not work after exiting sleep state.

The sleep state is entered when the SLEEP bit in ADC_MR is set or if all channels are disabled after a first series of conversion.

Problem Fix/Workaround

1. If the SLEEP bit is set in ADC_MR and a new calibration is required at anytime, the SLEEP bit must first be disabled after which a dummy conversion must be performed. At the end of the conversion (EOC) a new calibration can be started. At the end of the calibration, the SLEEP bit can be set again to 1 and normal conversions can again be performed.
2. If the SLEEP bit is not used but all channels are disabled at any time, the equivalent of the “sleep” state is reached by the internal finite state machine of the ADC controller. Therefore a dummy conversion must be performed, after which the calibration can be performed.
3. The Soft reset command can be performed prior to start a calibration but the configuration of all channels is cleared and must be reprogrammed.

60.4 Multi-port DDR-SDRAM Controller (MPDDRC)

60.4.1 LPDDR2 Refresh Per Bank Not Functional

The refresh per bank feature for 8-bank LPDDR2 memories is not functional.

The REF_PB bit in the MPDDRC_RTR must be set to 0 (reset value).

Problem Fix/Workaround

None.

60.4.2 Scrambling/Unscrambling Feature Not Supported On All Devices

The external data bus scrambling/unscrambling feature is only available on 32-bit memory devices connected on the external bus interface.

Problem Fix/Workaround

None.

60.4.3 MPDDRC_TPR1 Incoherency Between Write and Read Value

The value read in register MPDDRC_TRP1 does not correspond to the written value.

Problem Fix/Workaround

If MPDDRC_TRP1 is written with a ‘Value’, the read can be computed using the formula below:

Value = (MPDDRC_TRP1 & 0x000000FF) | ((MPDDRC_TRP1 & 0xFFFFFF00) << 2)

TXP[27:24] becomes TXP[27:26] ==> 2 bits lost

TXSRD[23:16] becomes TXSRD[25:18]

TXSNR[15:8] becomes TXSNR[17:10]

Bits [9:8] unused

TRFC[6:0] unchanged

60.5 Static Memory Controller (SMC)

60.5.1 Scrambling/Unscrambling Feature Not Supported On All Devices

The external data bus scrambling/unscrambling feature is only available on 32-bit memory devices connected on the external bus interface.

Problem Fix/Workaround

None.

60.6 PIO Controller

60.6.1 PE0 and PE1 Reset State

The reset of the I/Os PE0 and PE1 is output, high level with pulldown enable instead of output low level.

Problem Fix/Workaround

None

60.7 DMA Controller

60.7.1 DMA Request Overflow Error

When a DMA memory-to-memory transfer is performed, if the hardware request line selected by the field XDMAC_CCx.PERID toggles when the copy is enabled, the XDMAC_CISx.ROIS flag is set incorrectly. The memory copy proceeds normally and the data area is correctly transferred.

Problem Fix/Workaround

Configure the field XDMAC_CCx.PERID to an unused peripheral ID (refer to Table 8-1 "Peripheral Identifiers").

60.8 Two-wire Interface (TWI)

60.8.1 The TWI Clear Command Does Not Work

Bus reset using the “CLEAR” bit of the TWI control register does not work correctly during a bus busy state.

Problem Fix/Workaround

When the TWI master detects the SDA line stuck in low state the procedure to recover is:

1. Reconfigure the SDA/SCL lines as PIO.
2. Try to assert a Logic 1 on the SDA line (PIO output = 1).
3. Read the SDA line state. If the PIO state is a Logic 0, then generate a clock pulse on SCL (1-0-1 transition).
4. Read the SDA line state. If the SDA line = 0, go to Step 3; if SDA = 1, go to Step 5.
5. Generate a STOP condition.
6. Reconfigure SDA/SCL PIOs as peripheral.

60.9 Serial Synchronous Controller (SSC)

60.9.1 Inverted Left/Right Channels

When the SSC is in Slave mode, the TF signal is derived from the codec and not controlled by the SSC. The SSC transmits the data when detecting the falling edge on the TF signal after the SSC transmission is enabled. In some cases of overflow, a left/right channel inversion may occur. In this case, the SSC must be re-initialized.

Problem Fix/Workaround

Using the SSC in Master mode will ensure that TF is controlled by the SSC. No error occurs. If the SSC must be used in TF Slave mode, the SSC must be started by writing TXEN and RXEN synchronously with TXSYN flag rising in the SSC_SR.

60.9.2 Unexpected Delay on TD Output

When SSC is configured with the following conditions:

- RCMR.START = Start on falling edge/Start on Rising edge/Start on any edge
- RFMR.FSOS = None (input)
- TCMR.START = Receive Start

an unexpected delay of 2 or 3 system clock cycles is added to TD output.

Problem Fix/Workaround

None.

Revision History

In the tables that follow, the most recent version of the document appears first.

Table 60-1. SAMA5D4 Datasheet Revision History

Doc. Rev.	Date	Changes
C	12-Jul-16	“Features” Replaced: - “NEON Multimedia Architecture” with “NEON Media Processing Engine” - “Low-power 32 kHz RC Oscillator” with “Internal low-power 64 kHz (typical) RC Oscillator”
		Section 1. “Block Diagram” Updated Figure 1-1 “Block Diagram”
		Section 3. “Package and Pinout” Table 3-1 ”TFBGA361 Pin Description” and Table 3-2 ”LFBGA289 Pin Description”: updated reset state for signals PA0, PA9, PA16, PB24, PB25, PD12, PE0-PE5, PE7-PE23, PE24-PE28
		Section 4. “Power Considerations” Updated Section 4.7.3 “External Bus Interface” Table 4-1 "Power Supplies": corrected VDDBU maximum value to 2.6V
		Section 5. “Memories” Figure 5-1 “Memory Mapping”: changed color code of block “SMC” to Programmable Secured (PS)
		Section 8. “Peripherals” Updated Table 8-1 ”Peripheral Identifiers” Reworked Section 8.4 “Peripheral Clock Type”
		Section 13. “L2 Cache Controller (L2CC)” Table 13-2 “Register Mapping”: updated reset value for L2CC_IDR, L2CC_TRCR and L2CC_DRCR Section 13.5.1 “L2CC Cache ID Register”: changed cache ID value from ‘0x410000C8’ to ‘0x410000C9'
		Section 14. “AXI Matrix (AXIMX)” Table 14-1 “Register Mapping”: in both rows, removed 0x00000000 reset value
		Section 15. “Matrix (H64MX/H32MX)” Removed all information regarding compliance of Bus Matrix user interface with the ARM Advanced Peripheral Bus (APB) Section 15.2 “Embedded Characteristics”: removed bullet “AMBA Advanced High-performance Bus (AHB-Lite) compliant interface”; modified last item to read “ARM TrustZone technology” Table 15-3 "Master to Slave Access on H64MX": replaced “H64MX APB - User interfaces” with “H64MX Peripheral Bridge” Table 15-9 “Peripheral Identifiers”: modified row 22 (HSMC now Programmable Secure); in row 60, removed CTB content (changed to Reserved); in row 63, removed CATB content (changed to Reserved)
		Section 16. “Special Function Registers (SFR)” Section 16.3.1 “DDR Configuration Register”: added note Updated Section 16.3.8 “AIC Interrupt Redirection Register” Removed EBI Configuration Register
		Section 17. “Advanced Interrupt Controller (AIC)” Updated Section 17.8.3.3 “Interrupt Handlers” and Section 17.8.4.3 “Fast Interrupt Handlers” Removed sections “Interrupt Vectoring” and “Fast Interrupt Vectoring”

Table 60-1. SAMA5D4 Datasheet Revision History

Doc. Rev.	Date	Changes
C (cont'd)	12-Jul-16	Section 18. "Watchdog Timer (WDT)" Replaced "Idle mode" with "Sleep mode (idle mode)" in Section 18.1 "Description" and with "Sleep mode" in Section 18.4 "Functional Description" Added "When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified." in Section 18.4 "Functional Description" and Section 18.5.2 "Watchdog Timer Mode Register" (WDDIS bit description) Modified paragraph starting with "The reload of the WDT must occur..." in Section 18.4 "Functional Description"
		Section 19. "Reset Controller (RSTC)" Renamed 'proc_nreset' to 'Processor Reset', 'periph_nreset' to 'Peripheral Reset', 'backup_neset' to 'Backup Reset', 'rstc_irq' to 'Reset Controller Interrupt', 'wd_fault' to 'Watchdog Fault', 'user_reset' to User Reset
		Section 20. "Shutdown Controller (SHDWC)" Updated Section 20.2 "Embedded Characteristics"
		Section 22. "Real-time Clock (RTC)" Reworked Section 22.5.6 "Updating Time/Calendar" Reworked Figure 22-5. "Calibration Circuitry Waveforms" Section 22.6.1 "RTC Control Register": updated CALEVSEL bit description Updated Section 22.6.16 "RTC TimeStamp Source Register"
		Section 24. "Secure Fuse Controller (SFC)" Table 24-1 "Register Mapping": access "Read/Write" corrected to "Write-only" for SFC_IER and SFC_IDR Section 24.4.4.3 "Fuse Masking": corrected data register names Section 24.5.2 "SFC Mode Register": updated MSK field description
		Section 25. "Clock Generator" Renamed "UTMI Phase Lock Loop Programming" to "UTMI PLL Clock" Updated Figure 25-1. "Clock Generator Block Diagram" Added Figure 25-4. "Main Frequency Counter Block Diagram" Section 25.4 "Slow Clock": removed "This allows the slow clock to be valid in a short time (about 100 µs)" Section 25.4.2 "32.768 kHz Crystal Oscillator": deleted Figure 2-5. Typical 32.768 kHz Crystal Oscillator Connection Section 25.5.3 "Main Clock Source Selection": added detail on advantages of different oscillators Section 25.5.5 "Main Frequency Counter": updated speed of counter incrementation
		Section 26. "Power Management Controller (PMC)" Reorganized order of sections within the chapter Updated Section 26.2 "Embedded Characteristics" for Matrix Clocks, Peripheral Clocks and Generic Clock Updated Figure 26-1. "General Clock Block Diagram" and Figure 26-3. "H32MX 32-bit Matrix Clock Configuration" Figure 26-6. "Clock Failure Detection (Example)": corrected CDFEV to CFDEV and CDFS to CFDS Added warning under Figure 26-5. "Fast Startup from Ultra-low-power Mode" Updated Section 26.6 "Matrix Clock Controller" Section 26.16 "Programming Sequence", sub-section "Selecting Master Clock and Processor Clock": updated sequence following "If a new value for CSS field corresponds to PLL Clock"

Table 60-1. SAMA5D4 Datasheet Revision History

Doc. Rev.	Date	Changes
C (cont'd)	12-Jul-16	Section 26. "Power Management Controller (PMC)" (cont'd) Section 26.19.10 "PMC Clock Generator PLLA Register": changed DIVA description for value '0' Section 26.19.11 "PMC Master Clock Register": updated H32MXDIV field description Section 26.19.17 "PMC Status Register": added APLLCKRDY bit (bit 22)
		Section 27. "Parallel Input/Output Controller (PIO)" Corrected register name from "MATRIX_PSELRx" to "MATRIX_SPSELRx" throughout the section Table 27-4 "Register Mapping": "Peripheral Select Register 1" corrected to "Peripheral ABCD Select Register 1"; "Peripheral Select Register 2" corrected to "Peripheral ABCD Select Register 2"
		Section 28. "Multiport DDR-SDRAM Controller (MPDDRC)" Section 28.4.3 "Low-power DDR2-SDRAM Initialization": inserted the sentence "Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read" in steps 3, 5, 7, 9–14 Section 28.7.7 "MPDDRC Low-power Register": updated DS, TIMEOUT and UPD_MR field descriptions Section 28.7.9 "MPDDRC Low-power DDR2 Low-power Register": updated DS field description; deleted SR field description Section 28.7.12 "MPDDRC I/O Calibration Register": updated RDIV field description Section 28.7.16 "MPDDRC Configuration Arbiter Register": updated ARB field description
		Section 29. "Static Memory Controller (SMC)" Section 29.2 "Embedded Characteristics": added NFC SRAM bullet Figure 29-1. "Block Diagram": deleted "(8 Kbytes)" from NFC Internal SRAM block Removed NFCCMD field and modified Section 29.17.2.1 "Building NFC Address Command Example" and Section 29.17.2.2 "NFC Address Command" accordingly Section 29.17.3 "NFC Initialization": changed instances of "rbn" to "Ready/Busy" Section 29.17.4.1 "NFC SRAM Mapping": in second paragraph, deleted sentence "The NFC SRAM size is 8 Kbytes" Table 29-20 "Register Mapping": removed reset value from HSMC_CTRL (register is write-only) Removed acronym 'HSMC' from full, written-out register names in Table 29-20 "Register Mapping" and in corresponding register description sections (however, kept 'HSMC_' in short register names) Section 29.20.3 "NFC Status Register": updated RB_RISE and RB_FALL bit descriptions Section 29.20.36 "Timings Register": removed RBNSEL field
		Section 30. "DMA Controller (XDMAC)" Updated Figure 30-1 "DMA Controller (XDMAC) Block Diagram" Section 30.5.4.1 "Single Block With Single Microblock Transfer": added information on memory-to-memory transfer Section 30.8 "XDMAC Software Requirements": added information on memory-to-memory transfer and XDMAC_CC.INITD Table 30-4 "Register Mapping" and corresponding register description sections: corrected access of XDMAC_GTYPE, XDMAC_GCFG, XDMAC_GWAC, XDMAC_CIM Section 30.9.3 "XDMAC Global Weighted Arbiter Configuration Register": replaced "XDMAC scheduler" with "DMAC scheduler" throughout Section 30.9.6 "XDMAC Global Interrupt Mask Register": corrected access to Read-only Section 30.9.28 "XDMAC Channel x [x = 0..15] Configuration Register": updated PERID field description; corrected INITD bit description

Table 60-1. SAMA5D4 Datasheet Revision History

Doc. Rev.	Date	Changes
C (cont'd)	12-Jul-16	Section 31. "LCD Controller (LCDC)" Updated Section 31.2 "Embedded Characteristics" Section 31.5.3 "Interrupt Sources": "Advanced Interrupt Controller" and "AIC" changed to "interrupt controller" Updated Section 31.6.1.1 "Pixel Clock Period Configuration" Section 31.7.2 "LCD Controller Configuration Register 1": width of fields HSPW and VSPW changed from 8-bit to 10-bit Section 31.7.3 "LCD Controller Configuration Register 2": width of fields VFPW and VBPW changed from 8-bit to 10-bit
		Section 33. "Image Sensor Interface (ISI)" Section 33.6.12 "ISI Interrupt Enable Register", Section 33.6.13 "ISI Interrupt Disable Register": changed access from "Read/Write" to "Write-only" Section 33.6.14 "ISI Interrupt Mask Register": changed access from "Read/Write" to "Read-only"
		Section 35. "USB Host High Speed Port (UHPHS)" Updated Section 35.5.2 "Power Management" Added Section 35.7 "USB Host High Speed Port (UHPHS) User Interface"
		Section 36. "Ethernet MAC (GMAC)" Updated Section 36.1 "Description" Table 36-1 "GMAC Connections in Different Modes": added table note on GTXCK Section 36.5.2 "Power Management": deleted reference to PMC_PCER Section 36.5.3 "Interrupt Sources": added information on interrupt sources and priority queues; deleted reference to 'Advanced Interrupt Controller'. Replaced by 'interrupt controller'. Section 36.6.14 "IEEE 1588 Support": removed reference to 'output pins' in 2nd paragraph; deleted reference to GMAC_TSSx Added Section 36.6.18 "Energy-efficient Ethernet Support" and Section 36.6.19 "LPI Operation in the GMAC" Section 36.7.1.2 "Receive Buffer List" and Section 36.7.1.3 "Transmit Buffer List": added note at end of sections on queue pointer initialization Table 36-17 "Register Mapping": added registers at offsets 0x270 to 0x27C Section 36.8.1 "GMAC Network Control Register": added bit 19: TXLPIEN: Enable LPI Transmission (was 'reserved'). Added bit description. Section 36.8.3 "GMAC Network Status Register": added bit 7: RXLPIS: LPI Indication (was'reserved'). Added bit description. Section 36.8.13 "GMAC Interrupt Mask Register": added bit 26: SRI, and bit 28: WOL Added bit 27: RXLPISBC (Receive LPI indication Status Bit Change) and bit 29: TSUTIMCOMP (TSU timer comparison interrupt) in: - Section 36.8.10 "GMAC Interrupt Status Register" - Section 36.8.11 "GMAC Interrupt Enable Register" - Section 36.8.12 "GMAC Interrupt Disable Register" - Section 36.8.13 "GMAC Interrupt Mask Register"

Table 60-1. SAMA5D4 Datasheet Revision History

Doc. Rev.	Date	Changes
C (cont'd)	12-Jul-16	Section 36. "Ethernet MAC (GMAC)" (cont'd) Added the following sections: - Section 36.8.104 "GMAC Received LPI Transitions" - Section 36.8.105 "GMAC Received LPI Time" - Section 36.8.106 "GMAC Transmit LPI Transitions" - Section 36.8.107 "GMAC Transmit LPI Time"
		Section 37. "High Speed Multimedia Card Interface (HSMCI)" Section 37.8.5 "WRITE_SINGLE_BLOCK/WRITE_MULTIPLE_BLOCK Operation using DMA Controller": instance of "HSMCI_ARG" corrected to "HSMCI_ARGR" Table 37-9 "Register Mapping": added reset value for HSMCI_WPMR and HSMCI_WPSR Section 37.14.2 "HSMCI Mode Register": modified CLKDIV field description
		Section 38. "Serial Peripheral Interface (SPI)" Section 38.7.3 "Master Mode Operations": modified transmission condition description Table 38-5 "Register Mapping": for Chip Select Register, replaced fixed offset with equation Section 38.8.1 "SPI Control Register": added bit REQCLR Section 38.8.9 "SPI Chip Select Register": updated description of fields CSNAAT, SCBR, DLYBS and DLYBCT
		Section 40. "Synchronous Serial Controller (SSC)" Figure 40-19. "Interrupt Block Diagram": renamed RXSYNC to RXSYN; renamed TXSYNC to TXSYN Section 40.8.10 "Register Write Protection": replaced "AIC behavior" with "SSC behavior"
		Section 41. "Debug Unit (DBGU)" Section 41-3 "Register Mapping": added reset value for Chip ID Register (DBGU_CIDR) Section 41.6.10 "Debug Unit Chip ID Register": modified "ARCH: Architecture Identifier" field description to show only relevant values
		Section 43. "Universal Synchronous Asynchronous Receiver Transceiver (USART)" Section 43.6.1 "Baud Rate Generator": corrected value in "The frequency of the signal provided on SCK must be at least..." Section 43.6.1.2 "Fractional Baud Rate in Asynchronous Mode": added warning "When the value of field FP is greater than 0..."; removed sentence "This feature is only available when using USART normal mode." Section 43.6.1.3 "Baud Rate in Synchronous Mode or SPI Mode": corrected calculation for SCK maximum frequency Section 43.6.3.4 "Manchester Decoder": corrected "MANE flag" with "MANERR" flag Added Figure 43-27. "RTS Line Software Control when US_MR.USART_MODE = 2" Section 43.6.4 "ISO7816 Mode": corrected USART_MODE value for prototcol T = 1 Section 43.6.7.5 "Character Transmission": added content "An additional condition...on the receiver side."; corrected bit names: RTSEN to RCS, RTSDIS to FCS Section 43.6.7.7 "Receiver Timeout": deleted redundant paragraphs on STTTO and RETTO Section 43.7.1 "USART Control Register": updated RTSDIS bit description Section 43.7.3 "USART Mode Register": updated description for row 0xE, SPI_MASTER Section 43.7.15 "USART Baud Rate Generator Register": added warning "When the value of field FP is greater than 0..." to FP field description

Table 60-1. SAMA5D4 Datasheet Revision History

Doc. Rev.	Date	Changes
C (cont'd)	12-Jul-16	Section 45. "Timer Counter (TC)" Throughout, replaced TIOA, TIOB, TCLK with TIOAx, TIOBx, TCLKx Section 45.2 "Embedded Characteristics": rephrased "Total number of TC channels" to read "Total number of TC channels implemented on this device" Reformatted and renamed Table 45-2 "Channel Signal Description" Section 45.6.3 "Clock Selection": updated notes (1) and (2) Section 45.6.9 "Transfer with DMAC in Capture Mode": added "in Capture mode" and updated Figure 7-4 "Example of Transfer with DMAC_PDC in Capture Mode" Section 45.6.16.4 "Position and Rotation Measurement": added sentence about internal counter clearing Added Section 45.6.16.6 "Detecting a Missing Index Pulse"
		Section 46. "Pulse Width Modulation Controller (PWM)" Removed all content related to DMA. Updated Figure 46-1. "Pulse Width Modulation Controller Block Diagram" Updated Section 46.6.2.2 "Comparator" Section 46.6.5.1 "Initialization": modified "Enable of the interrupts..." list item Updated Section 46.6.5.6 "Interrupt Sources" Corrected PWM period formulas in Section 46.7.42 "PWM Channel Period Register" and Section 46.7.43 "PWM Channel Period Update Register" In register descriptions, added reference to Section 46.5.4 "Fault Inputs"
		Section 47. "Analog-to-Digital Converter (ADC)" Renamed "Hold time" to "Transfer time" throughout Section 47.1 "Description": removed "Finally, the user can configure ADC timings, such as startup time and tracking time." Updated Section 47.2 "Embedded Characteristics" Section 47.5 "Product Dependencies": removed sections "Timer Triggers", "Conversion Performances" and "PWM Event Line" Added Section 47.5.4 "Hardware Triggers" Reworked Section 47.6.1 "Analog-to-Digital Conversion" Modified Section 47.6.4 "Conversion Resolution" and Section 47.6.6 "Conversion Triggers" Section 47.6.8 "Comparison Window": replaced ADC_SR with ADC_ISR Section 47.6.9 "ADC Timings": reworded first paragraph; "ADC_ADTRG" corrected to "ADTRG" Revised Section 47.6.10 "Enhanced Resolution Mode and Digital Averaging Function" Section 47.6.12.1 "Classic ADC Channels Only (Touchscreen Disabled)": changed title (was "Classical ADC Channels Only") Table 47-6 "Register Mapping": defined offsets 0x48 and 0x4C as reserved Modified information about USCHx fields in Section 47.7.3 "ADC Channel Sequence 1 Register" and Section 47.6.7 "Sleep Mode and Conversion Sequencer" Section 47.7.4 "ADC Channel Enable Register": updated CHx field description Section 47.7.13 "ADC Extended Mode Register": modified CMPMOD bit description Section 47.7.21 "ADC Trigger Register": added sentence about register write protection

Table 60-1. SAMA5D4 Datasheet Revision History

Doc. Rev.	Date	Changes
C (cont'd)	12-Jul-16	Section 49. "Advanced Encryption Standard (AES)" Section 49.1 "Description": corrected index of AES_KEYWR0 registers from 3 to 7. Section 49.2 "Embedded Characteristics": replaced "12/14/16 Clock Cycles Encryption/Decryption Processing Time" with "10/12/14 Clock Cycles Encryption/Decryption Inherent Processing Time" Section 49.4.2 "Operating Modes": removed text with restriction on CTR counter size
		Section 50. "Triple Data Encryption Standard (TDES)" Section 50.4.1 "Operating Modes": deleted sentence "The OFB and CFB modes of operation are only available if 2-key mode is selected (KEYMOD = 1 in TDES_MR)." Section 50.4.3 "Last Output Data Mode": deleted sentence "No more Output Data Register reads are necessary between consecutive encryptions/decryptions Section 50.5.2 "TDES Mode Register": in OPMOD field description, deleted sentence "The OFB and CFB modes of operation are only available if 2-key mode is selected (KEYMOD = 1)."
		Section 52. "Advanced Encryption Standard Bridge (AESB)" Updated Section 52.1 "Description" Section 52.2 "Embedded Characteristics": added "On-the-fly off-chip memory encryption/decryption"; replaced "12 clock cycles encryption/decryption processing time with a 128-bit cryptographic key" with "10 clock cycles encryption/decryption inherent processing time"
		Section 53. "Integrity Check Monitor (ICM)" Table 53-9 "Register Mapping": changed Status Register access type from Write-only to Read only Section 53.5.2.2 "ICM Region Configuration Structure Member": removed MRPROT field Section 53.6.1 "ICM Configuration Register": removed fields HAPROT and DAPROT; updated DUALBUFF field description
		Section 55. "Electrical Characteristics" Table 55-2 "DC Characteristics": corrected V_{DDOSC} minimum value Corrected Figure 55-16. "SSC Transmitter, TK and TF in Input", Figure 55-18. "SSC Receiver, RK in Input and RF in Output" and Figure 55-19. "SSC Receiver, RK and RF in Output" (swapped CKI = 1 and CKI = 0) Added Section 55.20 "USART in Asynchronous Modes"
		Section 56. "Mechanical Characteristics" Updated Table 56-2 "Device and 361-ball TFBGA Package Maximum Weight" and Table 56-6 "Device and 289-ball LFBGA Package Maximum Weight"
		Section 60. "Errata" Added Section 60.7 "DMA Controller", Section 60.8 "Two-wire Interface (TWI)" and Section 60.9 "Serial Synchronous Controller (SSC)"

Table 60-2. SAMA5D4_11238B Datasheet Revision History

Doc. Rev.	Date	Changes
B	24-Aug-15	Throughout: editorial and formatting changes; harmonized package naming (TFBGA and LFBGA)
		Section "Description" Updated maximum processor frequency from 528 MHz to 600 MHz; added paragraph relating to low-power modes; in fifth paragraph, changed "hashing function" to "SHA function"
		Section "Features" Updated core performance from "832 MIPS @ 528 MHz" to "945 MIPS @ 600 MHz"; bullet "System running up to 176 MHz..." changed to "System running up to 200 MHz..."; inserted bullet "Three Low-power Modes: Idle, Ultra Low-power, and Backup"; added "ICM" and "AESB" to "Cryptography" features
		Section 1. "Block Diagram" Figure 1-1 "Block Diagram": added block "Osc RC 12M"; added "Backup Area" caption; "32K OSC" block renamed to "32K XTAL Oscillator"; "SHDWC" corrected to "SHDC"; renumbered three timer counter blocks as TC0, TC1, and TC2 (were previously nine TC channels numbered TC0–TC8); renamed "Cyphering" to "AESB"; moved "Scrambling" from "Trustzone Secured Multi-Layer Matrix" into DDR controller block
		Section 3. "Package and Pinout" Table 3-1 "TFBGA361 Pin Description": changed I/O type 'ANAIN2' to 'GPIO' (signals PD18–PD27) Table 3-2 "LFBGA289 Pin Description": changed I/O type 'ANAIN2' to 'GPIO' (signals PD18–PD27); at end of table, added pin K10 (Not connected) Table 3-3 "SAMA5D4 I/O Type Description": removed I/O type 'ANAIN2' row Table 3-4 "SAMA5D4 I/O Type Assignment and Frequency": removed I/O type 'ANAIN2' row
		Section 4. "Power Considerations" Updated Section 4.2 "Powerup Considerations" and Section 4.5 "Powerdown Considerations"
		Section 5. "Memories" Figure 5-1 "Memory Mapping": corrected "SHDC" to "SHDWC"; changed NFC SRAM space from 4 Kbytes to 16 Kbytes
		Section 6. "Real-time Event Management" Updated Table 6-1 "Real-time Event Mapping List"
		Section 7. "System Controller" Figure 7-1 "System Controller Block Diagram": removed 'WDRPROC' caption between Watchdog Timer and Reset Controller
		Section 8. "Peripherals" Table 8-1 "Peripheral Identifiers": added 'SHDWC' to "Wired-OR interrupts" for System Controller Interrupt
		Section 11. "Boot Sequence Controller (BSC)" Corrected register name "Boot Sequence Configuration Register" to "Boot Sequence Controller Configuration Register"
		Section 12. "Standard Boot Strategies" Section 12.4.4.4 "SPI Flash Boot": updated section "Supported DataFlash Devices" Section 12.5.3.2 "USB Class": removed reference to "Windows 7" Updated Section 12.6 "Fuse Box Controller"

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 13. “L2 Cache Controller (L2CC)” Table 13-2 ”Register Mapping”: changed L2CC_IDR reset value from ‘0x4100_00C8’ to ‘0x4100_00C9’ Section 13.5.1 “L2CC Cache ID Register”: changed cache ID value from ‘0x410000C8’ to ‘0x410000C9’
		Section 14. “AXI Matrix (AXIMX)” Section 14.2 “Embedded Characteristics”: changed “2 Masters” to “1 Master” and deleted subbullet “AHB/AXI bridge from AHB Matrix” from masters
		Section 15. “Matrix (H64MX/H32MX)” Table 15-2 ”List of H64MX Slaves”: replaced “AESOTF” with “AESB” in description of slave 3 Modified description of WPVS flag in Section 15.11 “Register Write Protection” and Section 15.13.11 “Write Protection Status Register”
		Section 16. “Special Function Registers (SFR)” Table 16-1 “Register Mapping”: - at offset 0x44, added Analog Configuration Register (SFR_ANACFG) - added reset values for SFR_EBICFG and SFR_AICREDIR - added footnote “If an offset is not listed in the table it must be considered as reserved.” Added Section 16.3.6 “Analog Configuration Register”
		Section 17. “Advanced Interrupt Controller (AIC)” Table 17-3 ”Register Mapping”: added row for reserved offsets 0x70–0xE0 Deleted reset values from individual register description sections (register reset values are provided in Table 17-3 ”Register Mapping”)
		Section 18. “Watchdog Timer (WDT)” Section 18.4 “Functional Description”: in fifth paragraph, “WDT_MR can be written only once” changed to “WDT_MR can be written until a LOCKMR command is issued in WDT_CR” Section 18.5.1 “Watchdog Timer Control Register”: added note on modification of WDT_CR values; added LOCKMR bit Section 18.5.2 “Watchdog Timer Mode Register”: changed access from “Read/Write Once” to “Read/Write; updated note on write access; updated note on modification of WDT_MR values Section 18.5.3 “Watchdog Timer Status Register”: appended names of bits WDUNF and WDERR with “(cleared on read)”
		Section 19. “Reset Controller (RSTC)” Figure 19-1 “Reset Controller Block Diagram”: removed WDRPROC signal Section 19.3 “Functional Description”: deleted redundant section “Reset Controller Status Register” (register is described in Section 19.4.2 “Reset Controller Status Register”) Section 19.3.3.5 “Watchdog Reset”: deleted content referencing WDRPROC bit in WDT_MR Figure 19-7 “Watchdog Reset”: deleted caption “(only if WDRPROC = 0)” from periph_nreset waveform Section 19.4.2 “Reset Controller Status Register”: updated descriptions of bits URSTS and NRSTL
		Section 20. “Shutdown Controller (SHDWC)” Table 20-1 ”I/O Lines Description”: added line WKUP1 Section 20.6 “Functional Description”: inserted subsection heading “Wake-up Inputs”
		Section 22. “Real-time Clock (RTC)” Updated Section 22.1 “Description” Updated Section 22.2 “Embedded Characteristics” Figure 22-1 “Real-time Clock Block Diagram”: renamed “APB” to “System Bus”

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 22. “Real-time Clock (RTC)” (cont’d) Updated Section 22.5 “Functional Description” Section 22.5.5 “RTC Internal Free Running Counter Error Checking”: replaced “RTC status clear control register” with “Status Clear Command Register” Updated Section 22.5.7 “RTC Accurate Clock Calibration” Table 22-2 ”Register Mapping”: added reserved offset 0xCC Section 22.6.1 “RTC Control Register”: updated descriptions of value ‘0’ for bits UPDTIM and UPDCAL Section 22.6.3 “RTC Time Register”: deleted sentence “All non-significant bits read zero.” Section 22.6.4 “RTC Calendar Register”: deleted sentence “All non-significant bits read zero.” Section 22.6.11 “RTC Interrupt Mask Register”: added TDERR bit Section 22.6.13 “RTC TimeStamp Time Register 0”: added sentence “RTC_TSTR0 reports the timestamp of the first tamper event after reading RTC_TSSR0”; deleted sentence “All non-significant bits read zero.” Section 22.6.14 “RTC TimeStamp Time Register 1”: added sentence “RTC_TSTR1 reports the timestamp of the last tamper event”; deleted sentence “All non-significant bits read zero.” Section 22.6.15 “RTC TimeStamp Date Register”: added sentence “RTC_TSTR0 reports the timestamp of the first tamper event after reading RTC_TSSR0, and RTC_TSTR1 reports the timestamp of the last tamper event”; deleted sentence “All non-significant bits read zero.” Section 22.6.16 “RTC TimeStamp Source Register”: inserted addresses; removed bits DET15:DET8 (register bits 31:24 now reserved)
		Section 23. “Slow Clock Controller (SCKC)” Harmonized naming of oscillators Updated Section 23.1 “Description” Updated Section 23.2 “Embedded Characteristics” Updated Figure 23-1. "Block Diagram" Inserted heading Section 23.4 “Functional Description” and updated content
		Section 24. “Secure Fuse Controller (SFC)” Replaced instances of “Atmel area” with “Atmel reserved area” throughout Updated Section 24.1 “Description” Updated Section 24.2 “Embedded Characteristics” Updated Figure 24-1 “SFC Block Diagram” Section 24.4 “Functional Description”: removed section “Programming Lock Fuse” Added Section 24.4.1 “Accessing the SFC” Updated Section 24.4.2 “Fuse Partitioning” Updated Section 24.4.4.2 “Fuse Programming” Section 24.4.4.3 “Fuse Masking”: at end of section, added sentence “The MSK bit has no effect on the programming of masked fuses.” Updated Section 24.4.5 “Fuse Functions” (old title “Specific Fuse Function) Table 24-1 ”Register Mapping”: updated number of listed SFC Data Registers Section 24.5.3 “SFC Interrupt Enable Register”: added bit PGMF; removed APLE bit Section 24.5.4 “SFC Interrupt Disable Register”: added bit PGMF; removed APLE bit Section 24.5.5 “SFC Interrupt Mask Register”: added bit PGMF; removed APLE bit Section 24.5.6 “SFC Status Register”: updated bit descriptions; added bit PGMF Section 24.5.7 “SFC Data Register x”: in “Name” line, added register index range

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 25. “Clock Generator” Updated Section 25.2 “Embedded Characteristics” Updated Section 25.3 “Block Diagram” Replaced Section 26.4 “Main Clock Selection” with Section 25.4 “Slow Clock” Extensive revision of Section 25.5 “Main Clock” (including subsections) Updated Section 25.7 “UTMI Phase Lock Loop Programming”
		Section 26. “Power Management Controller (PMC)” Harmonized oscillator naming; changed instances of “Sleep mode” to “Idle mode” Updated Figure 26-1 “General Clock Block Diagram” Section 26.4 “Master Clock Controller”: in first paragraph, changed “MCK is the clock provided to all the peripherals and the memory controller” to “MCK is the source clock of the peripheral clocks”; added note concerning fields MDIV and CSS Updated Section 26.5 “Processor Clock Controller” Updated Section 26.7 “LCDC Clock Controller” Section 26.8 “USB Device and Host Clocks”: removed last paragraph referencing the FREQ field and the SFR UTMI Clock Trimming Register (this register is not present) Added Section 26.11 “Fast Startup from Ultra Low-power (ULP) Mode” Revised Section 26.12 “Peripheral Clock Controller” Section 26.13 “Programmable Clock Controller”: in first sentence, changed “The PMC controls two signals” to read “The PMC controls three signals” Section 26.15 “32.768 kHz Crystal Oscillator Frequency Monitor”: changed title (was “Slow Crystal Clock Frequency Monitor”) and updated content Section 26.16 “Programming Sequence”: updated step 6 and step 9 Section 26.18 “Register Write Protection”: added three registers (PMC Peripheral Clock Disable Register 0, PMC Clock Generator Main Oscillator Register, and PMC Peripheral Clock Enable Register 1) to list of protectable registers Table 26-3 ”Register Mapping”: added row for reserved offset 0x110 Section 26.19.1 “PMC System Clock Enable Register”: added sentence about register write protection; updated LCDCK bit description Section 26.19.2 “PMC System Clock Disable Register”: added sentence about register write protection; updated LCDCK bit description Section 26.19.3 “PMC System Clock Status Register”: updated LCDCK bit description Section 26.19.8 “PMC Clock Generator Main Oscillator Register”: added sentence about register write protection; updated descriptions of bits MOSCXTBY and CFDEN Section 26.19.9 “PMC Clock Generator Main Clock Frequency Register”: added sentence about register write protection; updated field descriptions Section 26.19.10 “PMC Clock Generator PLLA Register”: added sentence about register write protection; removed warning referencing bit 29 Section 26.19.11 “PMC Master Clock Register”: added sentence about register write protection; updated descriptions of fields CSS and MDIV Section 26.19.12 “PMC USB Clock Register”: added sentence about register write protection Section 26.19.14 “PMC Programmable Clock Register”: added sentence about register write protection Section 26.19.15 “PMC Interrupt Enable Register”: removed MOSCRCS bit Section 26.19.16 “PMC Interrupt Disable Register”: removed MOSCRCS bit Section 26.19.17 “PMC Status Register”: added OSCSELS bit; removed MOSCRCS bit

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 26. “Power Management Controller (PMC)” (cont’d) Section 26.19.18 “PMC Interrupt Mask Register”: removed MOSCRCS bit Section 26.19.20 “PLL Charge Pump Current Register”: added sentence about register write protection; updated description of field ICP_PLLA Section 26.19.26 “PMC Peripheral Control Register”: updated description of field GCKDI
		Section 27. “Parallel Input/Output Controller (PIO)” Section 27.1 “Description”: changed number of programmable I/O lines from 32 to 152 Section 27.2 “Embedded Characteristics”: changed number of programmable I/O lines and number of bits of data output from 32 to 152 Figure 27-1 “Block Diagram”: modified to replace maximum number of pins with “x” integer Added Table 27-1 ”Peripheral IDs” Section 27.5 “Functional Description”: changed number of programmable I/O lines and number of possible indexes per signal from 32 to 152 Section 27.5.3 “Peripheral A or B or C or D Selection”: corrected “corresponding bit at level zero in PIO_ABCDSR2 means peripheral D is selected” to read “corresponding bit at level one in PIO_ABCDSR2 means peripheral D is selected”
		Section 28. “Multi-port DDR-SDRAM Controller (MPDDRC)” Section 28.2 “Embedded Characteristics”: updated list of supported configurations Section 28.4.1 “Low-power DDR1-SDRAM Initialization”: modified Step 9 Section 28.5.1 “DDR-SDRAM Controller Write Cycle”: updated text and inserted Table 28-1 ”CAS Write Latency” Section 28.5.2 “DDR-SDRAM Controller Read Cycle”: updated text and inserted Table 28-2 ”CAS Read Latency” Updated Section 28.5.3.2 “Power-down Mode” Added Section 28.6.3 “DDR-SDRAM Address Mapping for Low-cost Memories” Table 28-31 ”Register Mapping”: removed four “MPDDRC Smart Adaptation Wrapper” registers (offset range 0x60–0x6C now reserved); reworded footnote to read “Values vary with the product implementation” Updated Section 28.7.1 “MPDDRC Mode Register” Updated Section 28.7.7 “MPDDRC Low-power Register” Updated Section 28.7.8 “MPDDRC Memory Device Register” Section 28.7.12 “MPDDRC I/O Calibration Register”: updated field descriptions of RDIV and EN_CALIB Removed section “MPDDRC Smart Adaptation Wrapper x Register” Section 28.7.26 “MPDDRC DLL Master Offset Register”: renamed “MPDDRC_DLL_MO” to “MPDDRC_DLL_MAO”; renamed field “MxOFF: Master x Delay Line Offset” to “MAOFF: Master Delay Line Offset”
		Section 29. “Static Memory Controller (SMC)” Section 29.2 “Embedded Characteristics”: added NFC SRAM bullet Figure 29-1 “Block Diagram”: deleted “(8 Kbytes)” from NFC Internal SRAM block Section 29.17.4.1 “NFC SRAM Mapping”: in second paragraph, deleted sentence “The NFC SRAM size is 8 Kbytes” Section 29.19.3.1 “Error Location”: in first paragraph, changed “32 fully programmable coefficients” to “24 fully programmable coefficients” Table 29-20 ”Register Mapping”: changed access of HSMC_SIGMA0 from “Read/Write” to “Read-only” Added Section 29.20.30 “PMECC Error Location SIGMA 0 Register” Section 29.20.31 “PMECC Error Location SIGMA x Register”: in ‘Name’ line, changed [x=0..24] to [x=1..24]

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 30. “DMA Controller (XDMAC)” Section 30.2 “Embedded Characteristics”: added bullet “1 Kbyte Embedded FIFO”
		Section 31. “LCD Controller (LCDC)” Figure 31-1 “Block Diagram”: added “OVRx: Overlay” to legend
		Section 32. “Video Decoder (VDEC)” Section 32.11.2 “Interrupt Sources”: replaced instance of “Advanced Interrupt Controller (AIC)” and “AIC” with “interrupt controller”
		Section 33. “Image Sensor Interface (ISI)” Section 33.2 “Embedded Characteristics”: updated bullet “Preview Path”; added bullet “Codec Path up to 2048 × 2048” Added Section 33.4 “Product Dependencies” Figure 33-3 “HSYNC and VSYNC Synchronization”: corrected bus name from DATA[7..0] to ISI_DATA[7..0]. Figure 33-4 “SAV and EAV Sequence Synchronization”: corrected bus name from DATA[7..0] to ISI_DATA[7..0] Section 33.5.4.3 “Memory Interface”: changed heading “Grayscale Mode” to “12-bit Grayscale Mode” and updated content; added section “8-bit Grayscale Mode” Removed ‘Reset’ line from register descriptions to avoid redundancy with Table 33-13 ”Register Mapping” Section 33.6.2 “ISI Configuration 2 Register”: updated IM_VSIZE and IM_HSIZE field descriptions Section 33.6.3 “ISI Preview Size Register”: updated PREV_VSIZE and PREV_HSIZE field descriptions Section 33.6.11 “ISI Status Register”: updated bit descriptions
		Section 34. “USB High Speed Device Port (UDPHS)” Figure 34-1 “Block Diagram”: updated signal line configuration between UTMI and DP and between UTMI and DM Figure 34-2 “Board Schematic”: modified diagram and added two footnotes Section 34.7.1 “UDPHS Control Register”: added “(cleared upon USB reset)” to relevant field descriptions Section 34.7.2 “UDPHS Frame Number Register”: added “(cleared upon USB reset)” to field descriptions Section 34.7.3 “UDPHS Interrupt Enable Register”: added bit DMA_7 to bitmap; added “(cleared upon USB reset)” to bit descriptions Section 34.7.4 “UDPHS Interrupt Status Register”: added bit DMA_7 to bitmap; added “(cleared upon USB reset)” to description of bits EPT_x Section 34.7.8 “UDPHS Endpoint Configuration Register”: added “(cleared upon USB reset)” to field descriptions Section 34.7.9 “UDPHS Endpoint Control Enable Register (Control, Bulk, Interrupt Endpoints)”: inserted register addresses Section 34.7.13 “UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints)”: inserted register addresses; added “(cleared upon USB reset)” to bit descriptions Section 34.7.14 “UDPHS Endpoint Control Register (Isochronous Endpoint)”: added “(cleared upon USB reset)” to bit descriptions Section 34.7.15 “UDPHS Endpoint Set Status Register (Control, Bulk, Interrupt Endpoints)”: inserted register addresses Section 34.7.17 “UDPHS Endpoint Clear Status Register (Control, Bulk, Interrupt Endpoints)”: inserted register addresses Section 34.7.19 “UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints)”: inserted register addresses; updated description of BUSY_BANK_STA field; added “(cleared upon USB reset)” to field descriptions

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 34. “USB High Speed Device Port (UDPHS)” (cont’d) Section 34.7.20 “UDPHS Endpoint Status Register (Isochronous Endpoint)”: updated description of BUSY_BANK_STA field; added “(cleared upon USB reset)” to field descriptions
		Section 35. “USB Host High Speed Port (UHPHS)” Section 35.5.3 “Interrupt Sources”: replaced instance of “Advanced Interrupt Controller (AIC)” and “AIC” with “interrupt controller”
		Section 36. “Ethernet MAC (GMAC)” Section 36.2 “Embedded Characteristics”: updated bullet “Interrupt generation ...” Added Section 36.5 “Product Dependencies” Updated Section 36.6.2 “1588 Time Stamp Unit” Reworded Section 36.6.3 “AHB Direct Memory Access Interface” Section 36.6.3.2 “Transmit AHB Buffers”: updated third and eleventh paragraphs Section 36.6.4 “MAC Transmit Block”: updated second paragraph Section 36.6.5 “MAC Receive Block”: updated fourth and sixth paragraphs Section 36.6.7 “MAC Filtering Block”: updated first and sixth paragraphs Section 36.6.14 “IEEE 1588 Support”: added paragraph beginning “IEEE 802.1AS is mostly a subset of 1588...” Updated Section 36.6.15 “Time Stamp Unit” Table 36-17 “Register Mapping”: - updated descriptions of GMAC_HRB, GMAC_HRT, GMAC_SAB1, GMAC_SAT1, GMAC_SAB2, GMAC_SAT2, GMAC_SAB3, GMAC_SAT3, GMAC_SAB4, GMAC_SAT4, GMAC_SAMB1, and GMAC_SAMT1, GMAC_ORLO, GMAC_ORHI, and GMAC_TSL - new registers GMAC_RJFML (offset 0x048); GMAC_NSC (offset 0x0DC); GMAC_SCL (offset 0x0E0); GMAC_SCH (offset 0x0E4); GMAC_EFTSH (offset 0x0E8); GMAC_EFRSH (offset 0x0EC); GMAC_PEFTSH (offset 0x0F0); GMAC_PEFRSH (offset 0x0F4); GMAC_TISUBN (offset 0x1BC); GMAC_TSH (offset 0x1C0) - removed GMAC_TSSSL (offset 0x1C8) and GMAC_TSSN (offset 0x1CC) - offset 0x1E0: PTP Event Frame Transmitted Seconds / GMAC_EFTS replaced by PTP Event Frame Transmitted Seconds Low Register / GMAC_EFTSL - offset 0x1E8: PTP Event Frame Received Seconds / GMAC_EFRS replaced by PTP Event Frame Received Seconds Low Register / GMAC_EFRSL - offset 0x1F0: PTP Peer Event Frame Transmitted Seconds / GMAC_PEFTS replaced by PTP Peer Event Frame Transmitted Seconds Low Register / GMAC_PEFTSL - offset 0x1F8: PTP Peer Event Frame Received Seconds / GMAC_PEFRS replaced by PTP Peer Event Frame Received Seconds Low Register / GMAC_PEFRSL - updated reserved space Section 36.8.6 “GMAC Transmit Status Register”: updated TXGO bit description Section 36.8.9 “GMAC Receive Status Register”: updated RXOVR bit description Section 36.8.11 “GMAC Interrupt Enable Register”: updated bit descriptions Section 36.8.12 “GMAC Interrupt Disable Register”: updated bit descriptions Section 36.8.13 “GMAC Interrupt Mask Register”: updated bit descriptions Section 36.8.14 “GMAC PHY Maintenance Register”: added content on Clause 22/Clause 45 PHYs read/write access; updated CLTTO bit description Added Section 36.8.17 “GMAC RX Jumbo Frame Max Length Register” Added Section 36.8.38 “GMAC 1588 Timer Nanosecond Comparison Register” Added Section 36.8.39 “GMAC 1588 Timer Second Comparison Low Register”

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 36. “Ethernet MAC (GMAC)” (cont’d) Added Section 36.8.40 “GMAC 1588 Timer Second Comparison High Register” Added Section 36.8.41 “GMAC PTP Event Frame Transmitted Seconds High Register” Added Section 36.8.42 “GMAC PTP Event Frame Received Seconds High Register” Added Section 36.8.43 “GMAC PTP Peer Event Frame Transmitted Seconds High Register” Added Section 36.8.44 “GMAC PTP Peer Event Frame Received Seconds High Register” Section 36.8.50 “GMAC Pause Frames Transmitted Register”: changed PFTX field description Removed sections “1588 Timer Sync Strobe Seconds [31:0] Register” and “1588 Timer Sync Strobe Nanoseconds Register” Added Section 36.8.90 “GMAC 1588 Timer Increment Sub-nanoseconds Register” Added Section 36.8.91 “GMAC 1588 Timer Seconds High Register” Section 36.8.96 “GMAC PTP Event Frame Transmitted Seconds Low Register” (was “PTP Event Frame Transmitted Seconds Register”): updated RUD field description Section 36.8.97 “GMAC PTP Event Frame Transmitted Nanoseconds Register”: updated RUD field description
		Section 37. “High Speed Multimedia Card Interface (HSMCI)” Section 37.14.12 “HSMCI Status Register”: reworded clearing descriptions in relevant bit descriptions Added Section 37.14.20 “HSMCI FIFOx Memory Aperture”
		Section 38. “Serial Peripheral Interface (SPI)” Section 38.7.3 “Master Mode Operations”: modified text describing behavior of TDRE and TXEMPTY flags; added note ”When the SPI is enabled, the TDRE and TXEMPTY flags are set.” Added Figure 38-5 “TDRE and TXEMPTY flag behavior” Revised Figure 38-7 “Master Mode Flow Diagram” Section 38.7.3.5 “Peripheral Selection”: in last paragraph, “command must be issued before writing the last character” replaced by “command must be issued after writing the last character” Section 38.7.3.8 “Peripheral Deselection without DMA”: in last sentence, “(LASTXFER) bit in the SPI_MR” replaced by “(LASTXFER) bit in SPI_CR” Section 38.8.1 “SPI Control Register”: updated description of bit SPIDIS Section 38.8.5 “SPI Status Register”: updated description of bits RDRF, TDRE, and TXEMPTY Section 38.8.9 “SPI Chip Select Register”: updated descriptions of fields SCBR, DLYBS, and DLYBCT
		Section 39. “Two-wire Interface (TWI)” Section 39.1 “Description”: removed sentence: "Arbitration of the bus is performed internally and puts the TWIHS in Slave mode automatically if the bus arbitration is lost." Replaced section “Application Block Diagram” with updated Section 39.5 “I/O Lines Description” Removed section “Application Block Diagram” from Section 39.7.3 “Master Mode” and Section 39.7.5 “Slave Mode” Section 39.7.3.2 “Programming Master Mode”: replaced prefixes "TWIHS_" with "TWI_" Section 39.7.3.3 “Master Transmitter Mode”: modified 3rd paragraph related to NACK; added note on clearing TXRDY flag Section 39.7.3.5 “Internal Address”: under “10-bit Slave Addressing”, removed reference to “Atmel AT24LC512 EEPROM” Section 39.7.3.6 “Using the DMA Controller”: replaced instances of “(Optional) Wait for the TXCOMP flag in TWI_SR before disabling the peripheral clock if required” with “(Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWI_SR” Section 39.7.5.3 “Receiving Data”: under “Read Sequence”, added note on clearing TXRDY flag

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 39. “Two-wire Interface (TWI)” (cont’d) Corrected Figure 39-22 “Read Access Ordered by a Master” and Figure 39-23 “Write Access Ordered by a Master” (replaced EOSVACC with EOSACC) Corrected Figure 39-24 “Master Performs a General Call” (replaced GCACC with GACC) Figure 39-26 “Clock Synchronization in Write Mode”: replaced “SCL is stretched” with “TWCK is stretched” Section 39.7.5.5 “Using the DMA Controller”: replaced instances of “(Optional) Wait for the TXCOMP flag in TWI_SR before disabling the peripheral clock if required” with “(Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWI_SR” Corrected Figure 39-29 “Read Write Flowchart in Slave Mode” (changed “SVREAD = 0” to “SVREAD = 1”; changed "RXRDY = 0" to "RXRDY = 1") Section 39.8.1 “TWI Control Register”: updated START bit description Section 39.8.5 “TWI Clock Waveform Generator Register”: updated descriptions of fields CLDIV, CHDIV, and CKDIV Section 39.8.6 “TWI Status Register”: updated bit descriptions
		Section 42. “Universal Asynchronous Receiver Transmitter (UART)” Section 42.2 “Embedded Characteristics”: deleted bullet “Baud rate can be driven by processor independent source clock” Added Table 42-3 ”Peripheral IDs” Revised Section 42.5.1 “Baud Rate Generator” Section 42.6.2 “UART Mode Register”: removed BRSRCCK bit Section 42.6.9 “UART Baud Rate Generator Register”: updated CD field description
		Section 43. “Universal Synchronous Asynchronous Receiver Transmitter (USART)” Section 43.5.1 “I/O Lines”: deleted paragraph “To prevent the TXD line ...” Figure 43-2 “Baud Rate Generator”: added label “Selected Clock” to USCLKS mux output Section “Baud Rate Calculation Example”: in baud rate calculation formula, replaced “$f_{peripheral\ clock}$” with “Selected Clock” Figure 43-3 “Fractional Baud Rate Generator”: added label “Selected Clock” to USCLKS mux output Section 43.6.1.3 “Baud Rate in Synchronous Mode or SPI Mode”: in second paragraph, replaced “$f_{peripheral\ clock}$” with “Selected Clock” Updated Section 43.6.3.15 “Hardware Handshaking” Section 43.6.7.5 “Character Transmission”: after first paragraph, inserted new paragraph “The chip select line is de-asserted for a period equivalent to three bits between the transmission of two data.” Section 43.7.1 “USART Control Register”: updated STTTO bit description Section 43.7.6 “USART Interrupt Enable Register (SPI_MODE)”: added bit NSSE (register bit 19) Section 43.7.8 “USART Interrupt Disable Register (SPI_MODE)”: added bit NSSE (register bit 19) Section 43.7.10 “USART Interrupt Mask Register (SPI_MODE)”: added bit NSSE (register bit 19) Section 43.7.11 “USART Channel Status Register”: updated bit descriptions; added bit NSSE (register bit 19) and bit NSS (register bit 23) Section 43.7.12 “USART Channel Status Register (SPI_MODE)”: updated bit descriptions Section 43.7.15 “USART Baud Rate Generator Register”: restructured equations in CD field description Section 43.7.16 “USART Receiver Time-out Register”: restructured equation in TO field description Section 43.7.17 “USART Transmitter Timeguard Register”: restructured equation in TG bit description

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 45. “Timer Counter (TC)” Updated Section 45.1 “Description” Section 45.2 “Embedded Characteristics”: updated to indicate “9” as total number of TC channels Section 45.3 “Block Diagram”: inserted Table 45-1 ”Timer Counter Clock Assignment” (table was previously in Section 45.1 “Description”) Added Table 45-5 ”Peripheral IDs” Section 45.6.3 “Clock Selection”: updated names of external clock signals Section 45.6.14 “Synchronization with PWM”: in second paragraph, added output reference “OCx”; in third paragraph, corrected field names TRGSRCA and TRGSRCB to TRIGSRCA and TRIGSRCB Section 45.6.16 “Quadrature Decoder”: removed subsection “Missing Pulse Detection and Auto-correction” Section 45.6.16.4 “Position and Rotation Measurement”: in second paragraph, described selection of External Trigger Edge and External Trigger Section 45.6.16.5 “Speed Measurement”: in fifth paragraph, replaced “EDGTRG can be set to 0x01” with “ETRGEDG must be set to 0x01” Section 45.6.19 “Register Write Protection”: moved to end of Section 45.6 “Functional Description”; inserted sentence “The Timer Counter clock of the first channel must be enabled to access TC_WPMR” Section 45.7.2 “TC Channel Mode Register: Capture Mode”: in ‘Name’ line, replaced “(WAVE = 0)” with “(CAPTURE_MODE)”; updated TCCLKS field description for values 0–4 Section 45.7.3 “TC Channel Mode Register: Waveform Mode”: in ‘Name’ line, replaced “(WAVE = 1)” with “(WAVEFORM_MODE)”; inserted addresses; updated TCCLKS field description for values 0–4 Section 45.7.10 “TC Status Register”: updated bit descriptions (CKLSTA description unchanged) Section 45.7.14 “TC Extended Mode Register”: updated TRIGSRCB bit value ‘1’ description Section 45.7.16 “TC Block Mode Register”: removed AUTOC bit and MAXCMP field Section 45.7.20 “TC QDEC Interrupt Status Register”: removed MPE bit Section 45.7.22 “TC Write Protection Mode Register”: updated WPEN bit description
		Section 46. “Pulse Width Modulation Controller (PWM)” Section 46.2 “Embedded Characteristics”: renamed bullet “Programmable Fault/Break Inputs ... Outputs” to “Programmable Fault Inputs ... Outputs” Section 46.5.3 “Interrupt Sources”: deleted sentence “Note that it is not recommended to use the PWM interrupt line in Edge-sensitive mode.” Section 46.6.2.8 “Synchronous Channels”: in fourth paragraph, corrected references to bits, fields and registers; updated step 13 under heading “Method 3: Automatic write of duty-cycle values and automatic trigger of the update” Added Figure 46-18 “Event Line Generation Waveform (Example)” Table 46-7 ”Register Mapping”: removed reset value from PWM_SCUPUPD (register is write-only) Section 46.7.1 “PWM Clock Register”: inserted individual descriptions for fields DIVA, DIVB, PREA, and PREB Section 46.7.34 “PWM Write Protection Control Register” and Section 46.7.35 “PWM Write Protection Status Register”: removed reset value line Section 46.7.40 “PWM Channel Mode Register”: updated CPRE field description

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 46. “Pulse Width Modulation Controller (PWM)” (cont’d) Corrected referenced register name PWM_CPRx to PWM_CPRDx in Section 46.7.41 “PWM Channel Duty Cycle Register”, Section 46.7.42 “PWM Channel Duty Cycle Update Register”, Section 46.7.46 “PWM Channel Dead Time Register”, and Section 46.7.47 “PWM Channel Dead Time Update Register”
		Section 47. “Analog-to-Digital Converter (ADC)” Section 47.5 “Product Dependencies”: removed section “Analog Inputs” Updated Section 47.5.3 “I/O Lines” Revised Section 47.6.1 “Analog-to-Digital Conversion” Added Section 47.6.2 “ADC Clock” Section 47.6.3 “ADC Reference Voltage”: changed title (was “Conversion Reference”) Section 47.6.8 “Comparison Window”: in second paragraph, added details on filtering option Section 47.7.2 “ADC Mode Register”: - added configuration ADC_TRG7 in TRGSEL field description table - modified equation in PRESCAL bit description Section 47.7.11 “ADC Interrupt Status Register”: updated bit descriptions (PENS description unchanged) Section 47.7.13 “ADC Extended Mode Register”: updated CMPFILTER field description
		Section 48. “True Random Number Generator (TRNG)” Section 48.1 “Description”: updated names of referenced test suites Section 48.2 “Embedded Characteristics”: updated names of referenced test suites Section 48.5 “Functional Description”: updated terminology in text and in Figure 48-2 “TRNG Data Generation Sequence” Table 48-2 “Register Mapping”: defined offset ranges 0x04–0x0C, 0x20–0x4C, and 0x54–0xFC as reserved Removed “Reset” line from Section 48.6.4 “TRNG Interrupt Mask Register”, Section 48.6.5 “TRNG Interrupt Status Register” and Section 48.6.6 “TRNG Output Data Register”
		Section 49. “Advanced Encryption Standard (AES)” Added Section 49.4.1 “AES Register Endianism” Section 49.4.5.2 “DMA Mode”: removed references to ‘BTC’ throughout and replaced with generic wording Section 49.5.6 “AES Interrupt Status Register”: updated descriptions of DATRDY URAD, and URAT Added register address in Section 49.5.11 “AES Additional Authenticated Data Length Register”, Section 49.5.12 “AES Plaintext/Ciphertext Length Register”, Section 49.5.13 “AES GCM Intermediate Hash Word Register x”, Section 49.5.14 “AES GCM Authentication Tag Word Register x”, Section 49.5.15 “AES GCM Encryption Counter Value Register” and Section 49.5.16 “AES GCM H Word Register x”
		Section 50. “Triple Data Encryption Standard (TDES)” Section 50.3.2 “Interrupt Sources”: replaced instance of “Advanced Interrupt Controller (AIC)” and “AIC” with “interrupt controller” Updated Section 50.4.3.2 “DMA Mode” Table 50-6 “Register Mapping”: - defined offset ranges 0x68–0x6C and 0x74–0xFC as “Reserved” - for TDES_XTEA_RNDR, defined access as “Read/Write” and added reset value 0x0 Section 50.5.6 “TDES Interrupt Status Register”: updated field descriptions

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 51. “Secure Hash Algorithm (SHA)” Register index position in register names and acronyms modifed: - from “SHA Input Data x Register” to “SHA Input Data Register x” - from “SHA Input/Output Data x Register” to “SHA Input/Output Data Register x” - from “SHA_IDATAxR” to “SHA_IDATARx” - from “SHA_IODATAxR” to “SHA_IODATARx” Updated Section 51.4.4 “Internal Registers for Initial Hash Value” Section 51.4.5.3 “DMA Mode”: changed “The FIRST bit of the control register should be set to 1 prior to start the DMA” to “The FIRST bit of the SHA_CR must be set before starting the DMA” Section 51.4.5.4 “SHA Register Endianism”: rephrased first part of section Table 51-2 “Register Mapping”: in last row, defined offset range 0xC0–0xFC as reserved Section 51.5.1 “SHA Control Register”: modified WUIHV bit description Section 51.5.2 “SHA Mode Register”: modified UIHV bit description; removed configurations HMAC_SHA384, HMAC_SHA512, and HMAC_SHA224 from ALGO field values table; deleted notation “7 words for SHA224, , 12 words for SHA384, 16 words for SHA512” from end of section Section 51.5.6 “SHA Interrupt Status Register”: updated field descriptions Section 51.5.8 “SHA Input/Output Data Register x”: in IODATA field description, changed instance of “SHA_IODATAR5 to the last one in SHA1 mode...” to “SHA_IODATAR4 to the last one in SHA1 mode...”
		Section 53. “Integrity Check Monitor (ICM)” Section 53.5.1 “Overview”: reorganized and updated content under this new heading Section 53.5.2.2 “ICM Region Configuration Structure Member”: updated descriptions of fields RHIEN, DMIEN, BEIEN, WCIEN, ECIEN, SUIEN, and MPROT Updated Section 53.5.4 “Using ICM as SHA Engine” Added Section 53.5.4.1 “Settings for Simple SHA Calculation” Moved Section 53.5.7 “Security Features” to end of Section 53.5 “Functional Description” Section 53.6.1 “ICM Configuration Register”: updated description of fields DAPROT and HAPROT Section 53.6.3 “ICM Status Register”: updated description of fields RAWRMDIS and RMDIS
		Section 55. “Electrical Characteristics” Section 55.3.2.3 “Ultra-low-power Mode”: in second step, deleted “If not used, 12 MHz RC Oscillator can be disabled. MOSCRCEN is set to 0 in CKGR_MOR.” Updated Table 55-3 ”Low-power Mode Configuration Summary” Section 55.3.3 “Power Consumption Versus Modes”: updated T_A description Updated Table 55-5 ”Power Consumption in Active Mode” Updated Table 55-6 ”Power Consumption in Idle Mode” Table 55-7 ”VDDCORE Power Consumption in Ultra-low-Power Mode (AMP2)”: removed “ULP 512 Hz” mode row Table 55-8 ”Processor Clock Waveform Parameters”: updated maximum value to 600 MHz Table 55-9 ”Master Clock Waveform Parameters”: updated maximum value to 200 MHz Section 55.5 “Crystal Oscillator Characteristics”: removed figure “Main Oscillator Schematics” (redundant with diagram provided in Table 57-2 ”Clock, Oscillator and PLL Connections”) Section 55.7 “32 kHz Crystal Oscillator Characteristics”: removed figure “32 kHz Oscillator Schematics” (redundant with diagram provided in Table 57-2 ”Clock, Oscillator and PLL Connections”) Updated Section 55.8 “64 kHz RC Oscillator Characteristics” (was previously “32 kHz RC Oscillator Characteristics”)

Table 60-2. SAMA5D4_11238B Datasheet Revision History (Continued)

Doc. Rev.	Date	Changes
B	24-Aug-15	Section 55. “Electrical Characteristics” (cont’d) Table 55-12 ”XIN Clock Electrical Characteristics”: updated conditions Table 55-23 ”Channel Conversion Time and ADC Clock”: updated parameter “ADC Clock Frequency” Section 55.14.1 “Maximum SPI Frequency”: updated content under headings “Master Write Mode” and “Master Read Mode” Section 55.15 “MPDDRC Timings”: updated “DDRCK cycle time” minimum values for DDR2-SDRAM, LPDDR1-SDRAM, and LPDDR2-SDRAM Section 55.15.1 “Board Design Constraints”: in first sentence, corrected instance of SAM2613 to SAMA5D4 Table 55-38 ”SPI Timings with 1.8V Peripheral Supply (SPI0 only)”: added missing footnote Table 55-43 ”SSC Timings with 3.3V Peripheral Supply”: updated values of SSC_4 and SSC_7; replaced two footnotes with single footnote Table 55-44 ”SSC Timings with 1.8V Peripheral Supply (SSC1 only)”: updated values of SSC_4 and SSC_7; added single footnote Table 55-49 ”Ethernet MAC MII Specific Signals”: deleted footnote “See Note (1) of Table 56-48.” Table 55-50 ”Ethernet MAC RMII Mode”: deleted footnote “See Note (1) of Table 56-48.” Table 55-53 ”USART SPI Timings 1.8V Peripheral Supply (USART3 and USART4 only)”: added missing footnote Added Figure 55-29 “Minimum and Maximum Access Time for USART SPI Output Signal” Updated Figure 55-30 “Two-wire Serial Bus Timing” Table 55-54 ”Two-wire Serial Bus Requirements”: in last row, corrected parameter symbol to ”t_{BUF}” and conditions to “t_{LOW}”
		Section 56. “Mechanical Characteristics” Changed instances of “LFBGA” to “TFBGA” for 361-ball package Updated Figure 56-1 “361-ball TFBGA Package Drawing”
		Section 57. “Schematic Checklist” Table 57-1 ”Power Supply Connections”: updated description of VDDCORE and VDDANA; deleted footnote 4. “For more information, see Table 56-29 “Core Power Supply POR Characteristics”.”
		Section 60. “Errata” Added Section 60.1.2 “Boot ROM: Boot on MCI0 is Not Working” Added Section 60.6 “PIO Controller”

Table 60-3. SAMA5D4_11238A Datasheet Revision History

Doc. Rev.	Date	Changes
A	30-Sep-14	First release

Table of Contents

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**