YASKAWA

ANTAIOS
User Manual

ANT1000/1001 | Revision 1.23

Liability Exclusion

We have tested the contents of this document regarding agreement with the hardware and software described. Nevertheless, there may be deviations and we do not guarantee complete agreement. The data in the document is tested periodically, however. Required corrections are included in subsequent versions. We gratefully accept suggestions for improvements.

This document is subject to changes without prior notice.

List of Figures

List of Tables

1 Introduction

1.1 ARM Cortex-A5 CPU Core

- 288 MHz core speed
- 64 bit AXI
- 32 kByte instruction cache
- 32 kByte data cache
- JTAG debug interface
- ETM, **E**mbedded **T**race **M**acro Cell for real time tracing
- ITM Instrumentation Trace Macrocell
- ETB Embedded Trace Buffer
- AHB DAP (Access internal memory when ARM is running)
- Little endian byte ordering
- 64 bit Floating Point Unit (No NEON)

1.2 Advanced Real-Time Ethernet Switch

- 3 port switch system to connect two external Ethernet ports with one internal port
- Flexible architecture based on a micro-coded 5-core Protocol Processor Unit (PPU) cluster
- Primary focus on dedicated support of Profinet IRT IO, EtherCAT and Mechatrolink-III
- Secondary focus on providing a fast and flexible hardware platform for later (micro-code and software stack based) implementation of other Ethernet based protocols

- **Profinet IO IRT Device (software in development)**
  Profinet IO IRT specification v2.3
  Conformance Class C; Real Time Class 3
  Cycle times down to 31,25µs
  Stack partner: Molex

- **Mechatrolink-III (software in development)**
  Mechatrolink-III Master implementation
  Mechatrolink-III Slave implementation

- **EtherCAT Slave (software in development)**
  EtherCAT Slave Stack by ETG/Beckhoff

- **other Ethernet protocols with special HW requirements**
  General IEEE1588V2 support for
  EtherNet/IP CIP Sync
  Ethernet Powerlink
  Sercos III

- **other Ethernet protocols without special requirements**
  Modbus TCP
  EtherNet/IP

1.3 Integrated 100BaseTX Ethernet PHYs (2x)

- 2 integrated 100BaseTX Ethernet PHYs
- 100BaseFX support
- Special adaptions made to address profichip's real-time requirements

1.4 SNAP+ (SliceBus) Master

(To be used in combination with SNAP+ ASIC)

Basic SliceBus information:

- Single master system
- Up to 64 slaves (SNAP+ modules)
- Asynchronous, serial data transmission with 48 MBit/s via point-to-point LVDS physics

Error detection mechanism:

- CRC code with Hamming distance 4 for every telegram (all 3 bit errors will be detected)
- Watchdog function inside every SNAP+ module for SNAP+ Master observation
- "Auto shut down" in case of SNAP+ Master malfunction
- Retry statistic for early detection of possible transmission issues

Time-Synchronisation:

- Every SNAP+ module has its own clock with 1µs resolution
- All SNAP+ module clocks are synchronized with the SNAP+ Master (accuracy <100ns)
- Option for clock synchronization from SNAP+ Master to Fieldbus

SNAP+ Features (SliceBus Slave ASIC):

Technological functions in SNAP+ ASIC:

- Standard I/O function: 8 DI/DO or 16DI or 16DO with shift register
- Integrated digital input filter function
- Asynchronous event signaling with µs time stamping for advanced SNAP+ modules
- Two advanced counters with AB oversampling, latch, reset, output, hysteresis, compare value, repetitive/endless counting and additional time stamp information
- SSI function with time stamp information (speed calculations: counter difference/time)
- Pulse Width Modulation with 20ns resolution
- Frequency measurement mode
- Special digital I/O time stamp modules (ETS: **E**dge **T**ime **S**tamp System) for input edge and output control with 1µs resolution (independent from fieldbus cycle!)

SPI interface in SNAP+ for analog I/O, Safety I/O or serial CP with external MCU:

- 2.6 MBit/s SPI interface for external microcontroller

- up to 16 bytes IN / 16 bytes OUT data for external microcontroller
- up to 192 bytes of parameter data for external microcontroller
- Alarm function and watchdog function

1.5 Gigabit Ethernet MAC

- Supports 10/100/1000Mbps mode
- GMII interface
- DMA engine for transmitting and receiving packets with scatter gather list
- Supports IP, TCP, and UDP checksum offloads
- IEEE 802.1Q VLAN tag insertion for packet transmission, VLAN tag detection and removal for packet reception

1.6 DDR2 16 bit Controller

- 800 MByte/s maximum bandwidth
- 200 MHz clock rate (400 MHz data rate)
- 256 MByte maximum addressable[1)]
- 1 chip select

1.7 Asynchronous External Interface (AEI)

- Configurable 8 bit/16 bit master and slave interface (FIFO / CI)
- Setup/Hold/Accesstime/Pause time configurable
- 2 chip selects with 2 MB address range each and independent timings
- 1 dedicated external IRQ for ARM
- 1 dedicated external IRQ for SNAP+ Master synchronization
- Optional WAIT signal
- Slave Mode: Access time: 70ns in fastest mode

1.8 FIFO Interface

- FIFO Interface connected to Real-Time Ethernet Switch, internal ARM Processor and AEI Slave
- 48kByte total memory, divided into 256 FIFOs
- 255 IRQ Flags

1.9 Consistency Interface (CI)

- Direct connection to the Realtime switch and the AEI Slave
- 8 kByte input + 8 kByte output with consistency control
- Byte reorder function
  Examples:
  Unaligned endianness change with knowledge of data structure
  Separate PROFINET IOPS/IOCS from I/O Data if required

[1)] For memory configuration see chapter 4.9 in document *ANT1000/1001 Data Sheet*

Generate Data areas with different application update cycles
(e.g. 1ms and 250µs for IO Data of one device)
- 8 Process image partitions

1.10 PROFIBUS-DP Master (2x)

- 2 independent PROFIBUS-DP Master Controller
- Compliant with PROFIBUS standard IEC 61158
- Supported protocols: DP-V0, DP-V1, DP-V2 (DxB, IsoM, ClockSync)
- PROFIBUS Master Stack available from Profichip/Candeo

1.11 VPC3+ PROFIBUS-DP Slave

- PROFIBUS slave interface with data rates up to 12 Mbit/s
- Compliant with PROFIBUS standard IEC 61158
- 4 kByte communication RAM
- Supported protocols: DP-V0, DP-V1, DP-V2 (DxB, IsoM, ClockSync)
- Hardware-PLL for DP-V2 IsoM
- Hardware synchronization signal to SNAP+ Master

1.12 CAN Interface (2x)

- FullCAN controller for data rates up to 1 Mbit/s
- Complies with CAN standard ISO 11898
- Up to 15 messages simultaneously (each with maximum data length)
- Different message buffers can be combined as FIFO
- Listen only mode (monitoring of the CAN-bus, no acknowledge, no error flags)
- support of clock synchronization between ANTAIOS based stations

1.13 NAND Flash Controller

- 8-bit NAND Flash Controller
- DMA capable in conjunction with main DMA controller
- ECC: 16 bit correctable for 512 bytes

1.14 QuadSPI Interface

- Maximum 96 MHz per 4-line (max. 384 Mbit/s)
- DMA mode
- Programmable serial bit clock polarity, phase, and frequency
- SPI serial mode, dual mode and quad mode
- additional optional 4th address byte (extend address space up to 4096 M)
- 2 chip select lines

1.15 SD/MMC Card Controller

- Supports the MMC bus protocol, version 4.3

- Compliant with the SD memory card protocol version 3.0
- Write Protect pin
- Card Detect pin
- integrated DMA controller
- Built-in generation and check for 7-bit and 16-bit CRC data
- 1 kByte FIFO buffer
- 4-bit mode
- Up to 25 MByte data transfer

1.16 USB 2.0 Device Controller

- USB 2.0 high speed device controller (480Mbit/s)
- 8 endpoints
- Integrated USB PHY

1.17 Advanced IRQ Controller

- 8 priority levels
- Round-Robin option for IRQs with the same priority
- Throttling option for every IRQ channel
- All IRQs can be masked
- 32-bit ISR vector for each IRQ
- Configurable input filters for external IRQs
- IRQ/FIQ selectable for each IRQ channel

1.18 Main DMA Controller

- Scatter/Gather capable with chained transfer (linked list)
- 8 DMA channels
- Configurable arbitration schemes
- Support for fixed source address (read from auto-increment-register) to memory
- Support for reading from 8 bit device and copy to 32 bit device

1.19 AHB/APB Bridge (2x)

- 4 DMA channels

1.20 SPI Interface

- Master Mode with up to 80 Mbit/s
- Slave Mode with up to 24 Mbit/s
- DMA mode in conjunction with APB-Bridge
- Programmable frame/sync. polarity
- Programmable serial bit clock polarity, phase, and frequency
- Programmable serial bit data sequence (MSB or LSB first)
- 2 chip select lines

1.21 UART (2x)

Standard features (compatible to 16C550):

- 5/6/7/8 data bits
- 1/1.5/2 stop bits
- None/odd/even/stick parity
- Register/FIFO mode
- Line break generation & detection
- Programmable baud rate generator
- Fully prioritized interrupt system controls
- Status reporting capabilities
- Modem control functions
- Loopback mode

Enhanced Features:

- High speed mode for higher baud rates up to 12 Mbit/s
- Module controlled activation/deactivation for RTS
- 32-byte FIFO with 16C650 DMA behavior
- DMA mode in conjunction with APB-Bridge
- Enable/disable receiver
- IRQ generation by extended timeout control/detection
- IRQ generation by two configurable ETX characters
- IRQ generation by receive byte counter
- IRQ generation by transmitter with selectable “THR empty” or “TSR empty”

1.22 I²C Interface

- Master or slave for the I²C bus
- Data is transmitted to and received from the I²C bus via a buffered interface
- Supports the standard and fast modes
- Supports the 7-bit, 10-bit, and general-call addressing modes
- Glitch suppression by debounce circuit
- Programmable slave address
- Supports the master-transmit, master-receive, slave-transmit, and slave-receive modes
- Supports the multi-master mode
- General-call address detection in the slave mode

1.23 Timer and Watchdog Module

Timer

- Six independent 32-bit timer with pre-scaler (10ns - 80ns selectable)
- Interrupt can be issued upon overflow and time-up
- Each timer has two compare registers
- Supports increment and decrement modes
- Six interrupt sources, one for each counter/timer
- Supports single-shot and free running mode

- Automatically reloaded when reaching zero

Watchdog

- 32-bit down counter with pre-scaler
- Access protection
- Mode 1: system reset or IRQ at watchdog event
- Mode 2: watchdog IRQ at first watchdog event, system reset at next watchdog event (can be used for debugging)
- Option to pass information through the system reset: two registers with POWER-ON-RESET only (not affected by watchdog-reset)

1.24 Boot Code

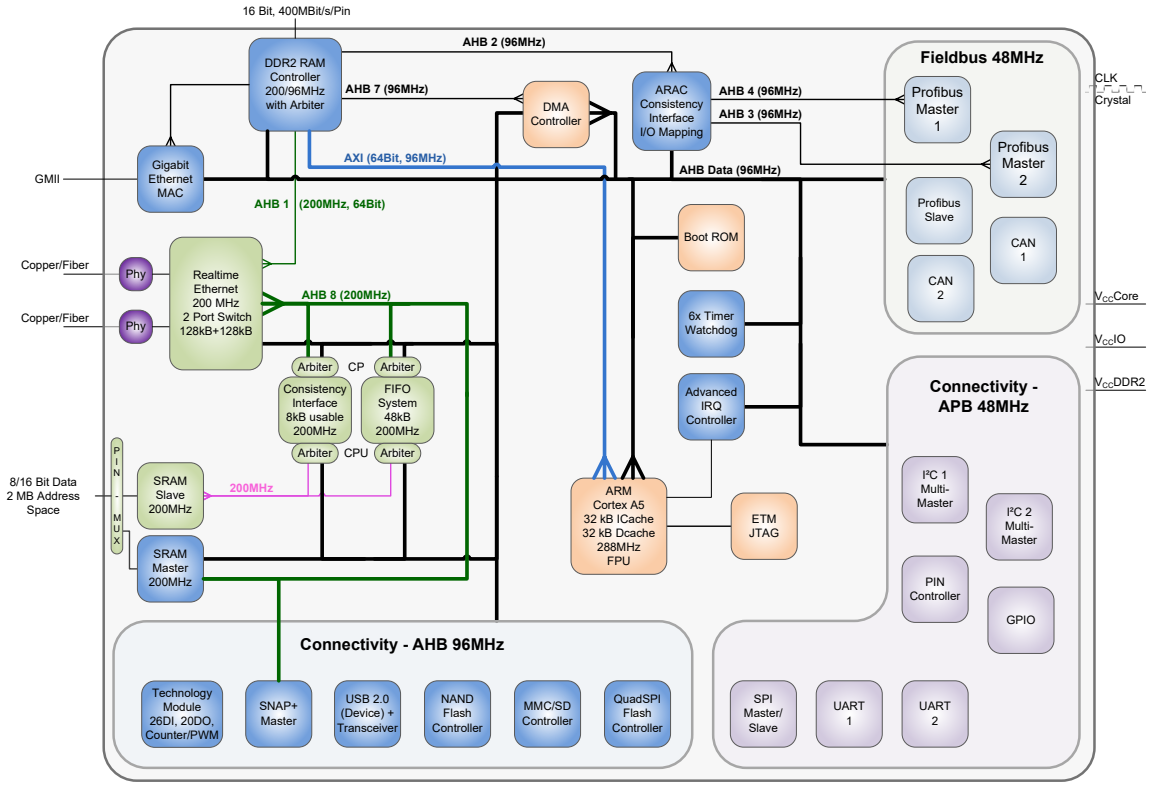
- Boot option selectable by GPIOs
- Boot from QuadSPI NOR Flash
- Boot from NAND Flash
- Boot from UART 1
- Boot from parallel NOR Flash

1.25 Technology Function Module (TechIO)

- Maximum 26 bits input and 20 bits output (shared with other interfaces)
- Configurable digital input low pass filter
- Up to 4 counter channels with quadruple evaluation for incremental encoders
- Up to 4 PWM channels (Pulse Width Modulation)
- Up to 2 SSI encoder interfaces

2 Block Diagram

Figure 2-1 ANTAIOS Block Diagram



3 AHB/APB Memory Mapping

Table 3-1 AHB/APB Memory Mapping

AHB Slave ID	Component	Notes		Address
0	AHB Controller Configuration			0xB8000000
1	DDR RAM	16Bit Interface		0x00000000
5	Profibus Master 1			0x48000000
6	Profibus Master 2			0x50000000
7	Gigabit MAC	10/100/1000Mbps		0x58000000
9	Profibus/CAN	VPC3+		0x68000000
		CAN 1		0x6A000000
		CAN 2		0x6C000000
10	AHB to APB Bridge 1	UART 2	(APB1 ID0)	0x70000000
		I^2C 1	(APB1 ID2)	0x71000000
		I^2C 2	(APB1 ID3)	0x71800000
		SPI	(APB1 ID4)	0x72000000
		Bus Monitor	(APB1 ID5)	0x72800000
		Pin Controller	(APB1 ID6)	0x73000000
		GPIO	(APB1 ID7)	0x73800000
11	AHB to APB Bridge 2	CoreSight APB-AP	(APB2 ID5)	0x78000000
		UART 1	(APB2 ID1)	0x78800000
		AXI Inter-Connect Config	(APB2 ID2)	0x79000000
		AXI to AHB Bridge Config	(APB2 ID3)	0x79800000
		DDR Controller Configuration	(APB2 ID6)	0x7B000000
12	Advanced IRQ Controller			0x80000000
14	USB 2.0 Device Controller			0x90000000
15	AEI Master	CS0 Data Area		0x98000000
		CS1 Data Area		0x99000000
		CS0 Configuration		0x9C000000
		CS1 Configuration		0x9C000004
16	Advanced Real-Timer Ethernet Switch (RTE)	Ethernet switch		0xA0000000
17	SNAP+ Master			0xA8000000
18	TimerWatchdog			0xB0000000
19	TechIO	DIO, Counter, PWM, SSI		0xC0000000
20	NAND Flash Controller			0xC8000000
21	DMA Controller			0xD0000000
22	MMC/SD Card Controller			0xD8000000
23	CI / FIFO Interface	FIFO (CPU side, shared with AEI Slave@0x000000)		0xE0000000
		CI (CPU side, shared with AEI Slave@0x100000)		0xE0100000
		FIFO (CP side, shared with Advanced Real-Time		0xE0200000

AHB Slave ID	Component	Notes	Address
		Ethernet Switch)	
		CI (CP side, shared with Advanced Real-Time Ethernet Switch)	0xE0300000
24	AHB to APB Bridge 1 Configuration		0xE8000000
25	AHB to APB Bridge 2 Configuration		0xF0000000
26	NAND Flash data		0xF1000000
27	QuadSPI Flash		0xF2000000
31	Boot Rom		0xFFFF0000

4 Pin Controller

Offset: 0x73000000

The Pin Controller is used to enable some of the external interfaces.
Additionally the user can control the chip select pins for SPI_cs0n/ SPI_cs1n and setup the physics and activate the DLL for the DDR2 RAM.

Table 4-1 Pin Controller register mapping

Offset	Size	Description / Option		Value	Access	Reset Value
0x00	Byte	PortA:	Unused	0x00	RW	0x0
			V1	0x01		
			V2	0x02		
0x04	Byte	PortB:	Unused	0x00	RW	0x0
			V1	0x01		
			V2	0x02		
			V3	0x03		
			V4	0x04		
			V5	0x05		
0x08	Byte	PortC:	Unused	0x00	RW	0x0
			V1	0x01		
			V2	0x02		
			V3	0x03		
			V4	0x04		
			V5	0x05		
0x0C	Byte	PortD:	Unused	0x00	RW	0x0
			V1	0x01		
			V2	0x02		
			V3	0x03		
			V4	0x04		
0x10	Byte	PortE:	Unused	0x00	RW	0x0
			V1	0x01		
			V2	0x02		
			V3	0x03		
0x14	Byte	PortF:	Unused	0x00	RW	0x1
			GPIO4/5	0x01		
			I2C_1	0x02		
0x20	Byte		SPI cs0n activated	0x00	RW	0x1
			SPI cs0n deactivated	0x01		
0x24	Byte		SPI cs1n activated	0x00	RW	0x1
			SPI cs1n deactivated	0x01		
0x2C	Byte	Bit 1:	Pleg DDR2 address phy	0/1	RW	0x0
		Bit 0:	Pleg DDR2 data phy	0/1		
0x30	Byte	PortE: debug select disabled (0x0000)		all others	RW	0x0
			ETM (Tracedata 31..16)	00000		
			PBM1	00001		
			PBM2	00010		

<table>
<tr><th>Offset</th><th>Size</th><th>Description / Option</th><th>Value</th><th>Access</th><th>Reset Value</th></tr>
<tr><td></td><td></td><td>PPU</td><td>000011</td><td></td><td></td></tr>
<tr><td></td><td></td><td>ARAC_L</td><td>000101</td><td></td><td></td></tr>
<tr><td></td><td></td><td>ARAC_H</td><td>000110</td><td></td><td></td></tr>
<tr><td>0x34</td><td>Byte</td><td>Bit 0: DDR2-Ctrl DLL power down, low active</td><td>0/1</td><td>RW</td><td>0x0</td></tr>
<tr><td>0x38</td><td>Byte</td><td>Bit 0: DDR2-Phy reset, low active</td><td>0/1</td><td>RW</td><td>0x0</td></tr>
<tr><td>0x3C</td><td>Byte</td><td>USB PHY control:
Bit 10..8: set reference voltage in the squelch circuit
76 mV
114 mv
143 mV
Bit 7: fix
Bit 6..4: set high-speed output slew rate
Fastest
Default
Slowest
Bit 3: fix
Bit 2..0: iref_st Reference DC current setting
for default setting, IREF_ST[2:0] = [011]
set from -4.2% to +4.2% at the TT corner
set from -4.8% to +4.8% at the FF corner
set from -3.8% to +3.8% at the SS corner</td><td>

000
100
111
0

000
001
111
0</td><td>RW</td><td>0x0416</td></tr>
<tr><td>0x40</td><td>Byte</td><td>route global control to SNAP+-Master</td><td>0x00</td><td>RW</td><td>0x0</td></tr>
<tr><td></td><td></td><td>route ext. AEI irq to SNAP+-Master</td><td>0x01</td><td></td><td></td></tr>
<tr><td>0x50</td><td>Byte</td><td>Bit 1: state of pin TEST2
Bit 0: state of pin TEST1
Boot source decoding
<table><tr><th>Bit 1</th><th>Bit 0</th><th>Boot from</th></tr><tr><td>0</td><td>0</td><td>I2C</td></tr><tr><td>0</td><td>1</td><td>AEI</td></tr><tr><td>1</td><td>0</td><td>NAND</td></tr><tr><td>1</td><td>1</td><td>QuadSPI</td></tr></table></td><td></td><td>read only</td><td>-</td></tr>
<tr><td>0x5c</td><td>Byte</td><td>Enable USB VBUS voltage if a USB device was plugged into a host.</td><td>0/1</td><td>RW</td><td>0x0</td></tr>
<tr><td>all others</td><td>-</td><td>Reserved</td><td>-</td><td>-</td><td>-</td></tr>
</table>

Table 4-2 Port options

Port	V1	V2	V3	V4	V5
A	Nand	SD/MMC QuadSPI			
B	MII1 MII2 GPIO7..6	GMII TechIO_out3..0 TechIO_in11..0 GPIO15..6	GMII TechIO_out3..0 TechIO_in5..0 GPIO14..6 Pbus	TechIO_out15..0 TechIO_in19..0	Debug - Txd1 - Rxd1 - Txd2 - Rxd2 - GPT
C	VPC I²C2 GPIO16	UART2 I²C2 GPIO17..16	CAN1 CAN2 I²C2 GPIO17..16	PBM1 PBM2	GPIO23..16
D	AEI Master	AEI Slave	CAN1 CAN2 VPC UART2 GPIO31-16	TechIO_out19..0 TechIO_in24..0	
E	PPUGPIO15..0	Debug: - ETM31..16 - PBM1 - PBM2 - Trace PPU - ARAC_L - ARAC_H	TechIO_out19..16 TechIO_in25..20 SSI1 SSI2		

5 SNAP+ (SliceBus) Master

Offset: 0xA8000000

See SNAP+MasterSpec document (subject to special agreement with profichip).

Note:
The SNAP+Master within ANTAIOS has BigEndian (Motorola) mapping.

6 Gigabit Ethernet Controller

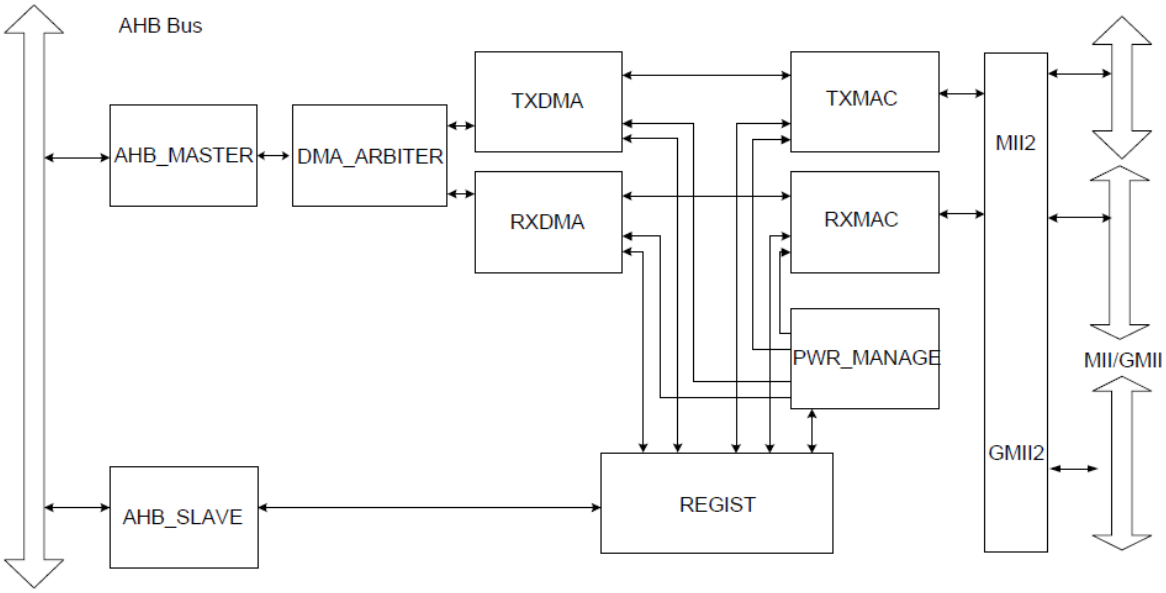
Offset: 0x58000000

ANTAIOS contains an 10/100/1000Mbit/s Ethernet controller with DMA function.
It includes the AHB wrapper, DMA engine, on-chip memories (TX FIFO and RX FIFO), MAC, and (G)MII interfaces. In order the uses the 1000Mbit/s mode the user must configure the Pin Controller, see Table 4-2.
The Ethernet controller is fully compliant with the IEEE 802.3 specification for the 10/100 Mbps Ethernet and the IEEE 802.3z specification for the 1000 Mbps Ethernet. The Ethernet controller with the DMA function handles all data transfers between the system memory and the on-chip memories. With the DMA engine, this controller reduces the CPU loading, maximizes the performance, and minimizes the FIFO size. The Ethernet controller has the on-chip memories for buffering so that the external local-buffer memory will not be needed.
The MII interface supports two specific data rates, 10 Mbps and 100 Mbps. The GMII interface supports the data rate at 1000 Mbps.
To reduce the processing load of the host CPU, the Ethernet controller implements the TCP, UDP, and IP V4 checksum generation and validation and supports the VLAN tagging. For the QoS and CoS requirements, the Ethernet controller supports the high-priority queues to reduce the processing load of the host CPU for transmitting packets.
The Ethernet controller provides a Wake-On-LAN function. It supports three wake-up events: Link status change, magic packet, and wake-up frame. This function allows a system containing the Ethernet controller to be woken up from the remote side.

- Supports DMA engine for transmitting and receiving packets
- Programmable DMA burst size
- Supports transmit and receive interrupt mitigation
- Supports zero-copy data transfer
- Supports IP, TCP and UDP checksum offloads
- Supports IEEE 802.1Q VLAN tag insertion for packet transmission, VLAN tag detection and removal for packet reception
- Supports High Priority Transmit Priority Queue for QoS and CoS applications
- Supports Wake-On-LAN function and three wake-up events:
    - Link status change
    - Magic packet
    - Wake-up frame
- Independent TX/RX FIFOs
- Supports half and full duplexes
- Supports flow control for full duplex and backpressure for half duplex
- Supports (G)MII interfaces
- Supports Jumbo packets (9K bytes)

6.1 Overview

Figure 6-1 Gigabit Ethernet Controller



6.1.1 AHB Master

AHB_MASTER implements the AHB master function of the Ethernet controller. When TXDMA moves the transmit packets from the transmit buffer to the TX FIFO, TXDMA will request DMA_ARBITER to use the DMA channel. After DMA_ARBITER gives the DMA channel to TXDMA, it initiates a burst transfer to AHB_MASTER through DMA_ARBITER and AHB_MASTER in turn to initiate a read operation to the AHB bus. Then AHB_MASTER passes the transmit packet data from the transmit buffer to TXDMA. When RXDMA moves the receive packets from the RX FIFO to the receive buffer, RXDMA will request DMA_ARBITER to use the DMA channel. After DMA_ARBITER gives the DMA channel to RXDMA, it initiates a burst transfer to AHB_MASTER through DMA_ARBITER and the AHB_MASTER in turn to initiate a write operation to the AHB bus. Then AHB_MASTER passes the receive packet data from RXDMA to the receive buffer.

6.1.2 AHB Slave

AHB_SLAVE implements the AHB slave function of the Ethernet controller. When other AHB master writes data to the registers in the Ethernet controller, it initiates a write operation to the AHB bus, which will prompt AHB_SLAVE to respond to the operation by forwarding the write data to REGIST. If other AHB master reads data from the registers in the Ethernet controller, the ADM master initiates a read operation to the AHB bus, which prompts AHB_SLAVE to respond to the operation by obtaining the read data from REGIST and passes information back to the AHB bus to complete a read operation.

6.1.3 DMA Arbiter

DMA_ARBITER acts as the bridge between two sets of control signals from TXDMA and RXDMA, and the control signal to AHB_MASTER. DMA_ARBITER acts as an arbiter to decide if TXDMA or RXDMA is entitled to use AHB_MASTER.

6.1.4 TXDMA

TXDMA performs four main functions:

1. Read the transmit descriptor;
2. Move the transmit packet data from the transmit buffer to TX FIFO;
3. Control the read/write actions of TX FIFO;
4. Do the TCP/UDP/IP checksum calculation.

When TXDMA transmits packet to Ethernet, it will fetch the descriptor information and transmit the buffer base address and size. Then TXDMA moves the transmit packet data from the corresponding transmit buffer to TX FIFO and requests TXMAC to read the transmit packet data with the help of TXDMA and sends it to the network. When the packet has been transmitted, TXMAC sends the transmit status to TXDMA and writes the transmit status back to the transmit descriptor.

6.1.5 RXDMA

RXDMA performs three main functions:

1. Read the receive descriptor and write the receive status to the receive descriptor
2. Move the receive packet data from the RX FIFO to the receive buffer
3. Control the read/write actions of the RX FIFO

When the Ethernet controller sends a packet, it will save the received packet into the RX FIFO. Then RXDMA moves the received packet from RX FIFO to the receiving buffer and writes the receive status to the receive descriptor.

6.1.6 TXMAC

TXMAC transmits packets from TX FIFO to Ethernet with CRC, preamble, jam generator, and transmit state machine included.
When TXMAC transmits a packet, it detects the Ethernet status and halts the transmission until Ethernet is free. Then TXMAC adds preamble and CRC to this packet and sends the packet to Ethernet. If TXMAC detects the collision during a transmission, it sends the jam to Ethernet and determines whether the collision is excessive. If not, it waits for backing to off-time and transmits the packet again.
TXMAC also performs the VLAN tag insertion.

6.1.7 RXMAC

The RXMAC receives packets from Ethernet to RX FIFO, with address recognition circuit, CRC check circuit, and receive state machine included.

When a packet is incoming, RXDMA will pass the received packet data to RX FIFO from RXMAC. RXDMA will save the received packet in RX FIFO if both the CRC result and packet address are correct; otherwise, the packet will be discarded.
RXMAC also performs the TCP/UDP/IP checksum verification and VLAN tag removal.

6.1.8 Regist

REGIST stores the registers of the Ethernet controller. Other AHB master reads/writes these registers by using AHB_SLAVE.

6.1.9 PWR_Manage

PWR_MANAGE manages the power control logic of the Ethernet controller. In the normal mode, PWR_MANAGE turns on the AHB bus clock, transmit clock, and receive clock. If the software forces the Ethernet controller into the power-down mode, PWR_MANAGE will turn off the AHB bus clock, transmit the clock and receive the clock to some modules to reduce the power consumption.

6.2 Memory Map and Register Definition

Table 6-1 lists and describes the control registers of ANTAIOS Ethernet controller.

Table 6-1 Summary of Ethernet controller

Address Offset	Type	Size (Byte)	Description	Default Value
0x00	RW1C	4	ISR	
0x04	RW	4	IME	0
0x08	RW	4	MAC_MADR	0
0x0C	RW	4	MAC_LADR	0
0x10	RW	4	MAHT0	0
0x14	RW	4	MAHT1	0
0x18	W	4	NPTXPD	0
0x1C	W	4	RXPD	0
0x20	RW	4	NPTXR_BADR	0
0x24	RW	4	RXR_BADR	0
0x28	W	4	HPTXPD	0
0x2C	RW	4	HPTXR_BADR	0
0x30	RW	4	ITC	0
0x34	RW	4	APTC	0
0x38	RW	4	DBLAC	0x00022F00
0x3C	R	4	DMAFIFOS	0x0C000000
0x40	R	8	Reserved	-
0x48	RW	4	TPAFCR	0x000000F1
0x4C	RW	4	RBSR	0x00000640
0x50	RW	4	MACCR	0
0x54	RW1C	4	MACSR	0
0x58	RW	4	TM	0
0x60	RW	4	PHYCR	0

Address Offset	Type	Size (Byte)	Description	Default Value
0x64	RW	4	PHYDATA	0
0x68	RW	4	FCR	0x00000400
0x6C	RW	4	BPR	0x00000400
0x70	RW	4	WOLCR	0
0x74	RW1c	4	WOLSR	0
0x78	RW	4	WFCRC	0
0x80	RW	4	WFBM1	0
0x84	RW	4	WFBM2	0
0x88	RW	4	WFBM3	0
0x8C	RW	4	WFBM4	0
0x90	R	4	NPTXR_PTR	0
0x94	R	4	HPTXR_PTR	0
0x98	R	4	RXR_PTR	0
0xA0	R	4	TPKT_CNT	0
0xA4	R	2	TXSCOL_CNT	0
0xA6	R	2	TXMCOL_CNT	0
0xA8	R	2	TXECOL_CNT	0
0xAA	R	2	TXFAIL_CNT	0
0xAC	R	2	TXLCOL_CNT	0
0xAE	R	2	TXUNDERUN_CNT	0
0xB0	R	4	RPKT_CNT	0
0xB4	R	4	BROPKT_CNT	0
0xB8	R	4	MULPKT_CNT	0
0xBC	R	2	AEP_CNT	0
0xBE	R	2	RPF_CNT	0
0xC0	R	2	RUNT_CNT	0
0xC4	R	2	FTL_CNT	0
0xC6	R	2	CRCER_CNT	0
0xC8	R	2	RCOL_CNT	0
0xCA	R	2	RLOST_CNT	0
0xE0	RW	4	ITC	0

6.2.1 Interrupt Status Register

Offset: 0x00

Table 6-2 Interrupt Status Register

Bit	Name	Type	Reset	Description
31..18	Reserved	R	0	-
17..16	MII_IF_SEL	R	0x0	MAC interface selection 00 = 10/100/1000 MII/GMII interface Others = reserved
15..11	Reserved	R	0	-
10	HPTXBUF_UNAVA	RW1C	0	High-priority transmit buffer unavailable
9	PHYSTS_CHG	RW1C	0	PHY link status change

Bit	Name	Type	Reset	Description
8	AHB_ERR	RW1C	0	AHB bus err
7	TPKT_LOST	RW1C	0	Packets transmitted to Ethernet are lost due to late collision, excessive collision, or under-run.
6	NPTXBUF_UNAVA	RW1C	0	Normal-priority transmit buffer unavailable
5	TPKT2F	RW1C	0	TXDMA has moved data into TX FIFO.
4	TPKT2E	RW1C	0	Packets successfully transmitted to Ethernet
3	RPKT_LOST	RW1C	0	Received packet lost due to RX FIFO full
2	RXBUF_UNAVA	RW1C	0	Receiving buffer unavailable
1	RPKT2F	RW1C	0	Packets successfully received into RX FIFO
0	RPKT2B	RW1C	0	RXDMA has successfully received packets to the RX buffer.

6.2.2 Interrupt Enable Register

Offset: 0x04

Table 6-3 Interrupt Enable Register

Bit	Name	Type	Reset	Description
31..11	Reserved	R	0	-
10	HPTXBUF_ UNAVA_EN	RW	0	Interrupt enable of ISR [10]
9	PHYSTS_CHG_EN	RW	0	Interrupt enable of ISR [9]
8	AHB_ERR_EN	RW	0	Interrupt enable of ISR [8]
7	TPKT_LOST_EN	RW	0	Interrupt enable of ISR [7]
6	NPTXBUF_ UNAVA_EN	RW	0	Interrupt enable of ISR [6]
5	TPKT2F_EN	RW	0	Interrupt enable of ISR [5]
4	TPKT2E_EN	RW	0	Interrupt enable of ISR [4]
3	RPKT_LOST_EN	RW	0	Interrupt enable of ISR [3]
2	RXBUF_ UNAVA_EN	RW	0	Interrupt enable of ISR [2]
1	RPKT2F_EN	RW	0	Interrupt enable of ISR [1]
0	RPKT2B_EN	RW	0	Interrupt enable of ISR [0]

6.2.3 MAC Most Significant Address Register

Offset: 0x08

Table 6-4 MAC Most Significant Address Register

Bit	Name	Type	Reset	Description
31..16	Reserved	R	0	-
15..0	MAC_MADR	RW	0	The most significant Two Bytes of the MAC address

6.2.4 MAC Least Significant Address Register

Offset: 0x0C

Table 6-5 MAC Least Significant Address Register

Bit	Name	Type	Reset	Description
31..0	MAC_MADR	RW	0	The least significant four bytes of the MAC address

6.2.5 Multicast Address Hash Table0 Register

Offset: 0x10

Table 6-6 Multicast Address Hash Table0 Register

Bit	Name	Type	Reset	Description
31..0	MAHT0	RW	0	Multicast address hash table bytes 3-0 (Hash table 31..0)

6.2.6 Multicast Address Hash Table1 Register

Offset: 0x14

Table 6-7 Multicast Address Hash Table1 Register

Bit	Name	Type	Reset	Description
31..0	MAHT1	RW	0	Multicast address hash table bytes 3-0 (Hash table 63..32)

6.2.7 Normal Priority Transmit Poll Demand Register

Offset: 0x18

Table 6-8 Normal Priority Transmit Poll Demand Register

Bit	Name	Type	Reset	Description
31..0	TXPD	RW	0	When writing a value to this register, Ethernet controller reads the normal-priority transmit descriptor and checks the txdma_own bit. If txdma_own = 1, it will move the transmit buffer data to the TX FIFO. The read value of the register is always 0.

6.2.1 Receive Poll Demand Register

Offset: 0x1C

Table 6-9 Receive Poll Demand Register

Bit	Name	Type	Reset	Description
31..0	RXPD	RW	0	When writing a value to this register, Ethernet controller reads the receive descriptor and checks the rxdma_own bit. If

Bit	Name	Type	Reset	Description
				rxdma_own = 1, it will move the receive packet data from the RX FIFO to the receiving buffer in the system memory. The read value of the register is always 0.

6.2.2 Normal Priority Transmit Ring Base Address Register

Offset: 0x20

Table 6-10 Normal Priority Transmit Ring Base Address Register

Bit	Name	Type	Reset	Description
31..0	TXR_BADR	RW	0	Base address of the normal priority transmit ring. The base address must be 16-byte aligned. Ethernet controller treats base address bits 3-0 as 0 when reading descriptors if bits 3-0 are not zero.

6.2.3 Receive Ring Base Address Register

Offset: 0x24

Table 6-11 Receive Ring Base Address Register

Bit	Name	Type	Reset	Description
31..0	RXR_BADR	RW	0	Base address of the receive ring. The base address must be 16-byte aligned. Ethernet controller treats base address bits 3-0 as 0 when reading descriptors if bits 3-0 are not zero.

6.2.4 High Priority Transmit Poll Demand Register

Offset: 0x28

Table 6-12 High Priority Transmit Poll Demand Register

Bit	Name	Type	Reset	Description
31..0	HPTXPD	W	0	When writing a value to the register, Ethernet controller reads the high-priority transmit descriptor process and checks the txdma_own bit, if txdma_own = 1, it will move the transmit buffer data into the TX FIFO. The read value of the register is always 0.

6.2.5 High Priority Transmit Ring Base Address Regitser

Offset: 0x2C

Table 6-13 High Priority Transmit Ring Base Address Register

Bit	Name	Type	Reset	Description
31..0	HPTXR_BADR	RW	0	Base address of the high priority transmit ring. The base address must be 16-byte aligned. Ethernet controller treats base address bits 3-0 as 0 when reading descriptors if bits 3-0 are not zero.

6.2.6 Interrupt Timer Control Register

Offset: 0x30

Table 6-14 Interrupt Timer Control Register

Bit	Name	Type	Reset	Description
31..30	Reserved	R	0	-
29..28	RXINT_THR_UNIT	RW	0	This field defines the unit of RXINT_THR. 00: Unit is 1 packet. 01: Unit is 4 packets. 10: Unit is 16 packets. 11: Unit is 64 packets.
27..25	Reserved	R	0	-
24..16	RXINT_CNT_H	RW	0	RXINT_CNT higher bits This field and RXINT_CNT_L defines the maximum wait time to issue receive interrupt after a packet has been received by Ethernet controller. The time unit is 1 RX cycle time. When RXINT_CNT = 0, the function is disabled. If RXINT_THR = 0 and RXINT_CNT = 0, a receive interrupt will be issued when a packet is received by the Ethernet controller.
15	TXINT_TIME_SEL	RW	0	This field defines the period of TX cycle time. When set, the TX cycle times are: 1000 Mbps mode: 16.384 µs 100 Mbps mode: 81.92 µs 10 Mbps mode: 819.2 µs When cleared, the TX cycle times are: 1000 Mbps mode: 1.024 µs 100 Mbps mode: 5.12 µs 10 Mbps mode: 51.2 µs
14..12	TXINT_THR	RW	0	This field defines the maximum number of transmit interrupts that are pending before an interrupt is generated. When TXINT_THR is not equal to 0, Ethernet controller will issue a transmit interrupt if the transmit packet number transmitted by the Ethernet controller reaches TXINT_THR. When TXINT_THR = 0 and TXINT_CNT = 0, Ethernet controller will issue a transmit interrupt or will not depend on TXIC in TXDES1.
11..8	TXINT_CNT	RW	0	This field defines the maximum wait time to issue transmit interrupt after a packet has been transmitted by the Ethernet controller. The time unit is 1 TX cycle time. When TXINT_CNT = 0, the function will be disabled. When TXINT_THR = 0 and TXINT_CNT = 0, Ethernet controller will issue a transmit interrupt or will not depend on TXIC in TXDES1.
7	RXINT_TIME_SEL	RW	0	This field defines the period of RX cycle time. When set, the RX cycle times are: 1000 Mbps mode: 16.38 µs 100 Mbps mode 81.92 µs 10 Mbps mode: 819.2 µs When cleared, the RX cycle times are: 1000 Mbps mode: 1.024 µs 100 Mbps mode: 5.12 µs 10 Mbps mode: 51.2 µs
6..4	RXINT_THR	RW	0	This field defines the maximum number of receive interrupts that are pending before an interrupt is generated. When

Bit	Name	Type	Reset	Description
				RXINT_THR is not equal to 0, the Ethernet controller will issue a receive interrupt if the receive packet number received by the Ethernet controller reaches RXINT_THR. If RXINT_THR = 0 and RXINT_CNT = 0, a receive interrupt will be issued when Ethernet controller finishes receiving a receive packet.
3..0	RXINT_CNT_L	RW	0	RXINT_CNT lower bits This field and RXINT_CNT_H defines the maximum wait time to issue receive interrupt after a packet has been received by the Ethernet controller. The time unit is 1 RX cycle time. When RXINT_CNT = 0, the function is disabled. If RXINT_THR = 0 and RXINT_CNT = 0, a receive interrupt will be issued when a packet is received by the Ethernet controller.

The Interrupt Timer Control Register allows the software driver to reduce the number of transmit interrupts (ISR[4]) and receive interrupts (ISR[0]) by setting the register. This lowers the CPU utilization for handling a large number of interrupts.
The register defines two threshold values for the receive packet number and transmit packet number and two associated timers. The threshold value defines the maximum number of receive or transmit interrupts that can be pending before an interrupt is generated. The timer defines the maximum wait time to issue the transmit/receive interrupt after a packet has been transmitted/received by the Ethernet controller. The threshold value and timer combination allow batching of several packets into a single interrupt with a limit for how long it will be pending. The combination prevents throughput from being impeded in heavy traffic and the time limit prevents resources from being held for too long in the low traffic.
The mitigation mechanism is similar for both receive and transmit interrupts. There is a counter (TXPKT_CNT) in the Ethernet controller to count the packets transmitted by the Ethernet controller. When the counter reaches TXINT_THR and TXINT_THR is not equal to 0, Ethernet controller will issue the transmit interrupt. There is also a counter (RXPKT_CNT) in the Ethernet controller to count the packets received by Ethernet controller. When the counter reaches RXINT_THR and RXINT_THR is not equal to 0, Ethernet controller will issue a receive interrupt. TXPKT_CNT will be cleared when the transmit interrupt is issued. RXPKT_CNT will be cleared when the receive interrupt is issued.
The following is the condition for the Ethernet controller to issue a transmit interrupt.

Table 6-15 Issue a Transmit Interrupt

TXINT_THR=0	TXINT_CNT=0	FTGMAC100_S Action
True	True	- The transmit interrupt will be issued after a packet is transmitted and TXIC of the packet is set. - Clear TXPKT_CNT
True	False	- The transmit interrupt will be issued after a packet is transmitted and timer reaches the value of TXINT_CNT. - Clear TXPKT_CNT
False	True	- The transmit interrupt will be issued if TXPKT_CNT = TXINT_THR. - Clear TXPKT_CNT

False	False	- The transmit interrupt will be issued if the following condition holds:
		  - TXPKT_CNT = TXINT_THR
		  - TXPKT_CNT = 1 and timer reaches the value of TXINT_CNT.
		- Clear TXPKT_CNT

Table 6-16 Issue a Receive Interrupt

RXINT_THR=0	RXINT_CNT_H/_L=0	FTGMAC100_S Action
True	True	- The receive interrupt will be issued after a packet is received by the Ethernet controller. - Clear RXPKT_CNT
True	False	- The receive interrupt will be issued after a packet is received by the Ethernet controller and timer reaches the value of RXINT_CNT. - Clear RXPKT_CNT
False	True	- The receive interrupt will be issued if RXPKT_CNT = RXINT_THR. - Clear RXPKT_CNT
False	False	- The receive interrupt will be issued if the following condition holds: - RXPKT_CNT = RXINT_THR - RXPKT_CNT = 1 and timer reaches the value of RXINT_CNT. - Clear RXPKT_CNT

6.2.7 Automatic Polling Timer Control Register

Offset: 0x34

This register allows the Ethernet controller to automatically poll the descriptors. This lowers the CPU utilization. If the transmit automatic poll function is enabled, Ethernet controller will automatically poll the transmit descriptor until transmit automatic poll timer expires. If the function is disabled, software will write the Transmit Poll Demand register (Offset: 0x18) to trigger the Ethernet controller for reading the transmit descriptors after software has prepared the transmit packets in the transmit buffers.
If the receive automatic poll function is enabled, Ethernet controller will automatically poll the receive descriptor until receive automatic poll timer expires. If the function is disabled, software will write the Receive Poll Demand register (RXPD, 6.2.1) to trigger the Ethernet controller for reading the receive descriptors after software has released the receive descriptors to the Ethernet controller.

Table 6-17 Automatic Polling Timer Control Register

Bit	Name	Type	Reset	Description
31..13	Reserved	R	0	-
12	TXPOLL_TIME_SEL	RW	0	This field defines the period of TX poll time. When set, the TX cycle times are: 1000 Mbps mode: 16.38 µs 100 Mbps mode 81.92 µs 10 Mbps mode: 819.2 µs When cleared, the TX poll times are: 1000 Mbps mode: 1.024 µs 100 Mbps mode: 5.12 µs 10 Mbps mode: 51.2 µs
11..8	TXPOLL_CNT	RW	0	This field defines the period of transmit automatic poll time. The unit is 1 TX poll time. When TXPOLL_CNT is not equal to 0, Ethernet controller will poll the transmit descriptor automatically. If TXPOLL_CNT = 0, Ethernet controller will not automatically poll the transmit descriptor.
7..5	Reserved	R	0	-

Bit	Name	Type	Reset	Description
4	RXPOLL_TIME_SEL	RW	0	This field defines the period of RX poll time. When set, the RX poll times are: 1000 Mbps mode: 16.38 µs 100 Mbps mode 81.92 µs 10 Mbps mode: 819.2 µs When cleared, the RX poll times are: 1000 Mbps mode: 1.024 µs 100 Mbps mode: 5.12 µs 10 Mbps mode: 51.2 µs
3..0	RXPOLL_CNT	RW	0	This field defines the period of receive automatic poll time. The unit is 1 RX poll time. When RXPOLL_CNT is not equal to 0, Ethernet controller will poll the receive descriptor automatically. If RXPOLL_CNT = 0, Ethernet controller will not automatically poll the receive descriptor.

6.2.8 DMA Burst Length and Arbitration Control Register

Offset: 0x38

Table 6-18 DMA Burst Length and Arbitration Control Register

Bit	Name	Type	Reset	Description
31..24	Reserved	R	0	-
23	IFG_INC	RW	0	IFG (Inter-Frame Gap) increase The field defines the increase or decrease of IFG in Ethernet. IFG_INC = 1 IFG will increase. IFG_INC = 0 IFG will decrease.
22.20	IFG_CNT	RW	0	IFG (Inter-Frame Gap) count The field defines the increase or decrease number of IFGin Ethernet. IFG_INC = 1: IFG increase IFG_INC = 0: IFG decrease The unit is 1 transmit clock in Ethernet 1000 Mbps mode: 8 ns 100 Mbps mode: 40 ns 10 Mbps mode: 400 ns When in the 1000-Mbps mode and IFG_INC = 0, the value in the field will not be set to be more than 2. For example: When IFG_CNT = 1 and IFG_INC = 1 in 1000 Mbps mode IFG = 96 + 1 x 8 = 104 ns. When IFG_CNT = 1 and IFG_INC = 0 in 1000-Mbps mode, IFG = 96 – 1 x 8 = 88 ns.
19..16	TXDES_SIZE	RW	2	Transmit descriptor size This field defines the transmit descriptor size. Writing 0 to this field is illegal. The unit is 8 bytes.
15..12	RXDES_SIZE	RW	2	Receive descriptor size This field defines the receive descriptor size. Writing 0 to this field is illegal. The unit is 8 bytes.
11..10	RXBXT_SIZE	RW	3	TXDMA maximum burst size per TXDMA burst This field sets the maximum size of the TXDMA burst. The burst sizes are as follows: 00: 64 bytes 01: 128 bytes 10: 256 bytes

Bit	Name	Type	Reset	Description
				11: 512 bytes
9..8	RXBST_SIZE	RW	3	RXDMA maximum burst size per RXDMA burst This field sets the maximum size of RXDMA burst. The burst sizes are as follows: 00: 64 bytes 01: 128 bytes 10: 256 bytes 11: 512 bytes
7	Reserved	R	0	-
6	RX_THR_EN	RW	0	Enable the RX FIFO threshold arbitration
5..3	RXFIFO_HTHR	RW	0	RX FIFO high threshold value for arbitration When the used space in RX FIFO is larger than or equal to the RX FIFO high threshold value, RXDMA will be higher priority over TXDMA by using the DMA channel. RXDMA keeps the higher priority until the used space in RX FIFO is less than or equal to the RX FIFO low threshold value. Then, TXDMA gets higher priority over RXDMA. Consequently, software must set RXFIFO_HTHR to be larger than RXFIFO_LTHR to keep the Ethernat controller work correctly. 0: Threshold = 0 1: Threshold = 1/8 space of RX FIFO 2: Threshold = 2/8 space of RX FIFO 3: Threshold = 3/8 space of RX FIFO 4: Threshold = 4/8 space of RX FIFO 5: Threshold = 5/8 space of RX FIFO 6: Threshold = 6/8 space of RX FIFO 7: Threshold = 7/8 space of RX FIFO
2..0	RXFIFO_LTHR	RW	0	RX FIFO low threshold value for arbitration When the used space in RX FIFO is less than or equal to the RX FIFO low threshold value, TXDMA will have higher priority over RXDMA by using the DMA channel. 0: Threshold = 0 1: Threshold = 1/8 space of RX FIFO 2: Threshold = 2/8 space of RX FIFO 3: Threshold = 3/8 space of RX FIFO 4: Threshold = 4/8 space of RX FIFO 5: Threshold = 5/8 space of RX FIFO 6: Threshold = 6/8 space of RX FIFO 7: Threshold = 7/8 space of RX FIFO

6.2.9 DMA/FIFO State Register

Offset: 0x3C

Table 6-19 DMA/FIFO State Register

Bit	Name	Type	Reset	Description
31	TXD_REQ	R	0	TXDMA request
30	RXD_REQ	R	0	RXDMA request
29	DARB_TXGNT	R	0	RXDMA grant
28	DARB_RXGNT	R	0	RXDMA grant
27	TXFIFO_EMPTY	R	1	TX FIFO is empty
26	RXFIFO_EMPTY	R	1	RX FIFO is empty
25..22	Reserved	R	0	-

Bit	Name	Type	Reset	Description
21..18	TXDMA3_SM	R	0	TXDMA 3 state machine The state machine is in charge of the read data flow from TX FIFO to TX pre-buffer.
17..16	TXDMA2_SM	R	0	TXDMA 2 state machine The state machine is in charge of the read data flow from TX FIFO to TX pre-buffer.
15..12	TXDMA1_SM	R	0	TXDMA 1 state machine The state machine is in charge of the read data flow from TX FIFO to TX pre-buffer.
11	Reserved	R	0	-
10..8	RXDMA3_SM	R	0	RXDMA 3 state machine The state machine is in charge of RXDMA PVCI interface read/write.
7..4	RXDMA2_SM	R	0	RXDMA 2 state machine The state machine is in charge of RXDMA PVCI interface read/write.
3..0	RXDMA1_SM	R	0	RXDMA 1 state machine The state machine is in charge of RXDMA PVCI interface read/write.

6.2.10 Transmit Priority Arbitration and FIFO Control Register

Offset 0x48

Table 6-20 DMA/FIFO State Register

Bit	Name	Type	Reset	Description
31..30	Reserved	R	0	-
29..27	TFIFO_SIZE	RW	0	TX FIFO real size Software can program this field to decide the TX FIFO size. Before programming this field, software must read the actual TX FIFO size in the FEATURE register. It is not allowed to program a value larger than the actual TX FIFO size. The FIFO sizes are as follows: 0: 2K 1: 4K 2: 8K 3: 16K 4: 32K 5-7: Reserved
26..24	RFIFO_SIZE	RW	0	RX FIFO Size Software can program this field to decide the RX FIFO size. Before programming this field, software must read the actual RX FIFO size in the FEATURE register. It is not allowed to program a value of larger than the actual RX FIFO size. The FIFO sizes are as follows: 0: 2K 1: 4K 2: 8K 3: 16K 4: 32K 5-7: Reserved
23..16	EARLY_TXTHR	RW	0	Early Transmit Threshold This field specifies the threshold level in TX FIFO to begin transmission. When the byte count of the data in TX FIFO reaches the threshold or there is at least one packet in TX

Bit	Name	Type	Reset	Description
				FIFO, hardware would begin to transmit the packet to network. Writing 0 to this field indicates that hardware should begin to transmit the packet after one whole packet has been saved in TX FIFO. The value software programs in this field should be less than TX FIFO size. The unit is 64 bytes.
15..8	EARLY_RXTHR	RW	0	Early Receive Threshold This field specifies the threshold level in RX FIFO to move packet data to system memory. When the byte count of the data in the RX FIFO reaches the threshold or there is at least one packet in RX FIFO, hardware would begin to move the packet from RX FIFO to system memory. Writing 0 to this field indicates that hardware should begin to move the packet after one whole packet has been stored in RX FIFO. The value software programs in this field should be less than RX FIFO size. The unit is 64 bytes.
7..4	HPKT_THR	RW	0xF	High Priority Transmit Packet Threshold When the packet number of TXDMA moving from the high priority transmit ring to TX FIFO is less than the threshold and the high priority packet is still available, TXDMA would switch to service the high priority transmit ring if TXDMA is servicing the normal priority transmit ring. If TXDMA is servicing the high priority transmit ring at that time, it would continue to service the high priority transmit ring. When the packet number of TXDMA moving from the high priority transmit ring to TX FIFO is equal to or greater than the threshold and the normal priority packet is still available, TXDMA would switch to service the high priority transmit ring. The hardware behavior is the same whether writing 0 or 1 to this field.
3..0	NPKT_THR	RW	1	Normal Priority Transmit Packet Threshold Under the following conditions, TXDMA will switch to serve the high priority transmit ring from the normal priority transmit ring: 1. When the high priority packet number is equal to or less than HPKT_THR and the high priority packet is available, TXDMA will switch to serve the high priority transmit ring after it finishes serving a normal priority transmit packet. 2. When the packet number of TXDMA moving from the normal priority transmit ring to TX FIFO is equal to or greater than the threshold and the high priority packet number is available, TXDMA will switch to serve the high priority transmit ring. 3. When the high priority packet is available and the normal priority packet is not available, TXDMA will switch to serve the high priority transmit ring. The hardware behavior is the same whether writing 0 or 1 to this field.

6.2.11 Receive Buffer Size Register

Offset: 0x4C

Table 6-21 Receive Buffer Size Register

Bit	Name	Type	Reset	Description
31..14	Reserved	R	0	-
13..0	RXBUF_SIZE	RW	0x640	Receive buffer size The unit is 1 byte. The receive buffer size must be 8-byte alignment

6.2.12 MAC Control Register

Offset: 0x50

Table 6-22 Mac Control Register

Bit	Name	Type	Reset	Description
31	SW_RST	RW	0	Software reset Writing 1 to this bit enables software reset. The software reset will last 175 96MHz bus clocks, and then be auto-cleared.
30	AITC_EN	RW	0	Advanced INT Time Control enable Writing 1 to this bit enables advanced INT time control (ITC, 6.2.14).
29..20	Reserved	R	0	-
19	SPEED_100	RW	0	Speed mode 1:100 Mbps 0:10 Mbps The field and GMAC_MODE (Bit 9) are used to determine the speed mode of the Ethernet controller. GMAC_MODE SPEED_100 0 1 100 Mbps mode 0 0 10 Mbps mode 1 0 1000 Mbps mode 1 1 1000 Mbps mode This field cannot be software reset.
18	DISCARD_CRCERR	RW	0	Discards the CRC error packet if there is CRC error status in the transmit packet.
17	RX_BROADOKT_EN	RW	0	Receives broadcast packets
16	RX_MULTIPKT_EN	RW	0	Receives all multicast packets
15	RX_HT_EN	RW	0	Enables storing incoming packet if the packet passes hash table address filtering and is a multicast packet.
14	RX_ALLADDR	RW	0	Destination address of the incoming packet is not checked
13	JUMBO_LF	RW	0	Jumbo Long Frame When set, packets with length more than 9216 (9220 for packets with VLAN tag) are treated as long frames. When cleared, packets with length more than 1518 (1522 for packets with VLAN tag) are treated as long frames.
12	RX_RUNT	RW	0	Receive the incoming packet even if its length is less than 64 bytes. The incoming packet length must be longer than or equal to 10 bytes.
11	Reserved	R	0	-
10	CRC_APD	RW	0	Append CRC to transmitted packets
9	GMAC_MODE	RW	0	GMAC mode If GMAC_MODE = 1, Ethernet controller will be in the 1000-Mbps mode; otherwise, Ethernet controller will be in the 10/100-Mbps mode. This field cannot be software reset.
8	FULLDUP	RW	0	Full duplex If FULLDUP = 1, Ethernet controller will be in the full-duplex mode; otherwise, Ethernet controller will be in the half-duplex mode.
7	ENRX_IN_HALFTX	RW	0	Enable packet reception when transmitting packets in the half-duplex mode
6	LOOP_EN	RW	0	Internal loopback enable This field cannot be software reset.
5	HPTXR_EN	RW	0	High priority transmit ring enable 1: software will use the high-priority transmit ring

Bit	Name	Type	Reset	Description
				0: software will not use the high-priority transmit ring.
4	REMOVE_VLAN	RW	0	Removes the VLAN tag from the packets received with the VLAN tag.
3	RXMAC_EN	RW	0	RXMAC enable When set, enable RXMAC to receive packets.
2	TXMAC_EN	RW	0	TXMAC enable When set, enable TXMAC to transmit packets.
1	RXDMA_EN	RW	0	Enables receive DMA channel If this bit is zero, reception will be stopped immediately.
0	TXDMA_EN	RW	0	Enables transmit DMA channel If this bit is zero, transmission will be stopped immediately.

6.2.13 MAC Status Register

Offset: 0x54

Table 6-23 Mac Control Register

Bit	Name	Type	Reset	Description
31..12	Reserved	R	0	-
11	COL_EXCEDD	RW1C	0	Collision amount exceeds 16
10	LATE_COL	RW1C	0	Transmitter detects late collision
9	XPKT_LOST	RW1C	0	Packets transmitted to Ethernet lost due to late collision or excessive collision
8	XPKT_OK	RW1C	0	Packets transmitted to Ethernet successfully
7	RUNT	RW1C	0	Receiver detects a runt packet
6	FTL	RW1C	0	Receiver detects a frame that is too long
5	CRC_ERR	RW1C	0	Incoming packet's CRC check result is invalid, unless the CRC_DIS bit is set
4	RPKT_LOST	RW1C	0	Received packets lost due to RX FIFO full
3	RPKT_SAVE	RW1C	0	Packets received to RX FIFO successfully
2	COL	RW1C	0	Incoming packet dropped due to collision
1	BROADCAST	RW1C	0	Incoming packet for broadcast address
0	MULTICAST	RW1C	0	Incoming packet for multicast address

6.2.14 Test Mode Register

Offset: 0x58

Table 6-24 Test Mode Register

Bit	Name	Type	Reset	Description
31..21	Reserved	R	0	-
20	PTIMER_TEST	RW	0	Automatic polling timer test mode
19	ITIMER_TEST	RW	0	Interrupt timer test mode
18..16	Reserved	R	0	-
15	TEST_COL	RW	0	Transmit collision test mode
14..5	TEST_BKOFF	RW	0	Back-off value in the transmission collision test mode

Bit	Name	Type	Reset	Description
4..0	TEST_EXSTHR	RW	0	Retry upper limit in the transmit collision test mode

6.2.15 PHY Data Register

Offset: 0x64

Table 6-25 PHY Data Register

Bit	Name	Type	Reset	Description
31..16	MIIRDATA	R	0	Read data from PHY
15..0	MIIWDATA	RW	0	Write data to PHY

6.2.16 Flow Control Register

Offset: 0x68

Table 6-26 Flow Control Register

Bit	Name	Type	Reset	Description
31..16	PAUSE_TIME	RW	0	Pause time in the pause frame The unit is 1 slot time.
15..9	FC_HIGH/LOW	RW	2	RX FIFO free space high threshold A pause frame is sent with pause time = 0 when the RX FIFO free space is larger than the high threshold. The unit is 256 bytes, and the default value is 5. RX FIFO free space low threshold A pause frame is sent with the pause time set in bits 31-16 when the RX FIFO free space is less than the low threshold. The unit is 256 bytes, and the default value is 2. When FC_HTHR_SEL = 1, the RX FIFO free space high threshold is selected. When FC_HTHR_SEL = 0, RX FIFO free space low threshold is selected. The value software programs in this field should be less than the RX FIFO size.
8	FC_HTHR_SEL	RW	0	RX FIFO free space high threshold select When set, the RX FIFO free space high threshold is selected. When cleared, the RX FIFO free space low threshold is selected.
7..5	Reserved	R	0	-
4	RX_PAUSE	RW1C	0	Receive pause frame
3	TXPAUSED	R	0	Packet transmission paused due to the receive pause frame
2	FCTHR_EN	RW	0	Enable flow control threshold mode This bit enables the transmit pause frame for high/low threshold
1	TX_PAUSE	RW	0	Transmit pause frame Software can set this bit to send the pause frames. This bit is auto-cleared after the pause frame has been transmitted.
0	FC_EN	RW	0	Flow control mode enable

6.2.17 Back Pressure Register

Offset: 0x6C

Table 6-27 Back Pressure Register

Bit	Name	Type	Reset	Description
31..15	Reserved	R	0	-
14..8	BK_LOW	RW	4	RX FIFO free space low threshold MAC generates a jam pattern if RX FIFO free space is less than the low threshold when packets are incoming. The unit is 256 bytes, and the default value is 2.
7..4	BKJAM_LEN	RW	0	Back pressure jam length 0: 4 bytes 1: 8 bytes 2: 16 bytes 3: 32 bytes 4: 64 bytes 5: 128 bytes 6: 256 bytes 7: 512 bytes 8: 1024 bytes 9: 1518 bytes 10: 2048 bytes Others: 4 bytes
3..2	Reserved	RW	0	-
1	BKADR_MODE	RW	0	Back pressure address mode 1: Generate jam pattern when packet address matches 0: Generate jam pattern when any packet is incoming
0	BK_EN	RW	0	Back pressure mode enable

6.2.18 Wake-On-LAN Control Register

Offset: 0x70

Table 6-28 Wake-On-LAN Control Register

Bit	Name	Type	Reset	Description
31..26	Reserved	R	0	-
25..24	WOL_TYPE	RW	0	WOL output signal type 00: Active high 01: Active low 10: Positive pulse 11: Negative pulse
23..19	Reserved	R	0	-
18	SW_PDNPHY	RW	0	Software power down PHY
17..16	WAKEUP_SEL	RW	0	Wake-up frame select This field is used to select the wake-up frame when the driver accesses the wake-up frame CRC or wake-up frame byte mask register. 00: Select wake-up frame 1 01: Select wake-up frame 2 10: Select wake-up frame 3 11: Select wake-up frame 4
15	PWRSAV	RW	0	Power saving mode This field is used to determine the current power state. When set, Ethernet controller enters the power-saving mode. When cleared, Ethernet controller is in the normal mode.

Bit	Name	Type	Reset	Description
14..17	Reserved	R	0	-
6	WAKEUP4_EN	RW	0	Enable wake-up frame 4 event
5	WAKEUP3_EN	RW	0	Enable wake-up frame 3 event
4	WAKEUP2_EN	RW	0	Enable wake-up frame 2 event
3	WAKEUP1_EN	RW	0	Enable wake-up frame 1 event
2	MAGICPKT_EN	RW	0	Enable magic packet event
1	LINKCHG1_EN	RW	0	Enable link change to 1 event
0	LINKCHG0_EN	RW	0	Enable link change to 0 event

6.2.19 Wake-On-LAN Status Register

Offset: 0x74

Table 6-29 Wake-On-LAN Status Register

Bit	Name	Type	Reset	Description
31..7	Reserved	R	0	-
6	WAKEUP4_STS	RW1C	0	Wake-up frame 4 event status
5	WAKEUP3_STS	RW1C	0	Wake-up frame 3 event status
4	WAKEUP2_STS	RW1C	0	Wake-up frame 2 event status
3	WAKEUP1_STS	RW1C	0	Wake-up frame 1 event status
2	MAGICPKT_STS	RW1C	0	Magic packet event status
1	LINKCHG1_STS	RW1C	0	Link change to 1 event status
0	LINKCHG0_STS	RW1C	0	Link change to 0 event status

6.2.20 Wake-up Frame CRC Register

Offset: 0x78

Table 6-30 Wake-On-LAN Status Register

Bit	Name	Type	Reset	Description
31..0	WFCRC	RW	0	Wake-up frame CRC value When WAKEUP_SEL (WOLCR, 6.2.18) is 00: The write value is to be stored in wake-up frame 1 CRC register. The read value is from the wake-up frame 1 CRC register. 01: The write value is to be stored in wake-up frame 2 CRC register. The read value is from the wake-up frame 2 CRC register. 10: The write value is to be stored in wake-up frame 3 CRC register. The read value is from the wake-up frame 3 CRC register. 11: The write value is to be stored in wake-up frame 4 CRC register. The read value is from the wake-up frame 4 CRC register.

6.2.21 Wake-up Frame Byte Mask 1st Double-word Register

Offset: 0x80

Table 6-31 Wake-up Frame Byte Mask 1st Double-word Register

Bit	Name	Type	Reset	Description
31..0	WFBM1	RW	0	Wake-up frame byte mask 1st double-word This field stores the byte mask of the wake-up frame from byte1 to byte32. Bit0 is the byte1 mask and bit31 is the byte32 mask. When WAKEUP_SEL (WOLCR, 6.2.18) is 00: The write value is to be stored in the wake-up frame 1 byte mask 1st double-word register. The read value is from wake-up frame 1 byte mask 1st double-word register. 01: The write value is to be stored in the wake-up frame 2 byte mask 1st double-word register. The read value is from byte mask 1st double-word register of wake-up frame 2. 10: The write value is to be stored in the wake-up frame 3 byte mask 1st double-word register. The read value is from byte mask 1st double-word register of wake-up frame 3. 11: The write value is to be stored in the wake-up frame 4 byte mask 1st double-word register. The read value is from byte mask 1st double-word register of wake-up frame 4.

6.2.22 Wake-up Frame Byte Mask 2nd Double-word Register

Offset: 0x84

Table 6-32 Wake-up Frame Byte Mask 2nd Double-word Register

Bit	Name	Type	Reset	Description
31..0	WFBM2	RW	0	Wake-up frame byte mask 2nd double-word This field stores the byte mask of the wake-up frame from byte33 to byte64. Bit0 is the byte33 mask and bit31 is the byte64 mask. When WAKEUP_SEL (WOLCR, 6.2.18) is 00 The write value is to be stored in the wake-up frame 1 byte mask 2nd double-word register. The read value is from byte mask 2nd double-word register of wake-up frame 1. 01 The write value is to be stored in the wake-up frame 2 byte mask 2nd double-word register. The read value is from byte mask 2nd double-word register of wake-up frame 2. 10: The write value is to be stored in the wake-up frame 3 byte mask 2nd double-word register. The read value is from byte mask 2nd double-word register of wake-up frame 3. 11 The write value is to be stored in the wake-up frame 4 byte mask 2nd double-word register. The read value is from byte mask 2nd double-word register of wake-up frame 4.

6.2.23 Wake-up Frame Byte Mask 3rd Double-word Register

Offset: 0x88

Table 6-33 Wake-up Frame Byte Mask 3rd Double-word Register

Bit	Name	Type	Reset	Description
31..0	WFBM3	RW	0	Wake-up frame byte mask 3rd double-word This field stores the byte mask of the wake-up frame from byte65 to byte96. Bit0 is the byte65 mask and bit31 is the byte96 mask. When WAKEUP_SEL (WOLCR, 6.2.18) is 00 The write value is to be stored in the wake-up frame 1 byte mask 3rd double-word register. The read value is from byte mask 3rd double-word register of wake-up frame 1. 01 The write value is to be stored in the wake-up frame 2 byte mask 3rd double-word register. The read value is from byte mask 3rd double-word register of wake-up frame 2. 10: The write value is to be stored in the wake-up frame 3 byte mask 3rd double-word register. The read value is from byte mask 3rd double-word register of wake-up frame 3. 11 The write value is to be stored in the wake-up frame 4 byte mask 3rd double-word register. The read value is from byte mask 3rd double-word register of wake-up frame 4.

6.2.24 Wake-up Frame Byte Mask 4th Double-word Register

Offset: 0x8C

Table 6-34 Wake-up Frame Byte Mask 4th Double-word Register

Bit	Name	Type	Reset	Description
31..0	WFBM3	RW	0	Wake-up frame byte mask 4th double-word This field stores the byte mask of the wake-up frame from byte97 to byte128. Bit0 is the byte97 mask and bit31 is the byte128 mask. When WAKEUP_SEL (WOLCR, 6.2.18) is 00 The write value is to be stored in the wake-up frame 1 byte mask 4th double-word register. The read value is from byte mask 4th double-word register of wake-up frame 1. 01 The write value is to be stored in the wake-up frame 2 byte mask 4th double-word register. The read value is from byte mask 4th double-word register of wake-up frame 2. 10: The write value is to be stored in the wake-up frame 3 byte mask 4th double-word register. The read value is from byte mask 4th double-word register of wake-up frame 3. 11 The write value is to be stored in the wake-up frame 4 byte mask 4th double-word register. The read value is from byte mask 4th double-word register of wake-up frame 4.

6.2.25 Normal Priority Transmit Ring Pointer Register

Offset: 0x90

Table 6-35 Normal Priority Transmit Ring Pointer Register

Bit	Name	Type	Reset	Description
31..0	NPTXR_PTR	R	0	Normal Priority Transmit Ring Pointer Register This register indicates the current value of the transmit descriptor pointer for the normal priority transmit ring pointer register

6.2.26 High Priority Transmit Ring Pointer Register

Offset: 0x94

Table 6-36 High Priority Transmit Ring Pointer Register

Bit	Name	Type	Reset	Description
31..0	HPTXR_PTR	R	0	High Priority Transmit Ring Pointer Register This register indicates the current value of the transmit descriptor pointer for the high priority transmit ring pointer register.

6.2.27 Receive Ring Pointer Register

Offset: 0x98

Table 6-37 Receive Ring Pointer Register

Bit	Name	Type	Reset	Description
31..0	RXR_PTR	R	0	Receive Ring Pointer Register This register indicates the current value of the transmit descriptor pointer for the receive ring pointer register.

6.2.28 TPKT_CNT Counter Register

Offset: 0xA0

Table 6-38 TPKT_CNT Counter Register

Bit	Name	Type	Reset	Description
31..0	TPKT_CNT	R	0	Counter for counting packets transmitted successfully

6.2.29 TXMCOL_CNT and TXSCOL_CNT Counter Register

Offset: 0xA4

Table 6-39 TXMCOL_CNT and TXSCOL_CNT Counter Register

Bit	Name	Type	Reset	Description
31..16	TXMCOL_CNT	R	0	Counter for counting packets transmitted OK with 2-15 collisions
15..0	TXSCOL_CNT	R	0	Counter for counting packets transmitted OK with single collision

6.2.30 TXECOL_CNT and TXFAIL_CNT Counter Register

Offset: 0xA8

Table 6-40 TXECOL_CNT and TXFAIL_CNT Counter Register

Bit	Name	Type	Reset	Description
31..16	TXECOL_CNT	R	0	Counter for counting packets failed in transmission (Due to the late collision or collision count ≥ 16 or transmit underrun)
15..0	TXFAIL_CNT	R	0	Counter for counting packets failed in transmission (Due to the collision count ≥ 16)

6.2.31 TXLCOL_CNT and TXUNDERRUN_CNT Counter Register

Offset: 0xAC

Table 6-41 TXLCOL_CNT and TXUNDERRUN_CNT Counter Register

Bit	Name	Type	Reset	Description
31..16	TXUNDERRUN_CNT	R	0	Counter for counting the packets failed in transmission (Due to transmit under-run)
15..0	TXLCOL_CNT	R	0	Counter for counting the packets failed in transmission (Due to late collision)

6.2.32 RPKT_CNT Counter Register

Offset: 0xB0

Table 6-42 RPKT_CNT Counter Register

Bit	Name	Type	Reset	Description
31..0	RPKT_CNT	R	0	Counter for counting the packets received successfully

6.2.33 BROPKT_CNT Counter Register

Offset: 0xB4

Table 6-43 BROPKT_CNT Counter Register

Bit	Name	Type	Reset	Description
31..0	BROPKT_CNT	R	0	Counter for counting the received broadcast packets

6.2.34 MULPKT_CNT Counter Register

Offset: 0xB8

Table 6-44 MULPKT_CNT Counter Register

Bit	Name	Type	Reset	Description
31..0	MULPKT_CNT	R	0	Counter for counting the received multicast packets

6.2.35 RPF_CNT and AEP_CNT Counter Register

Offset: 0xBC

Table 6-45 RPF_CNT and AEP_CNT Counter Register

Bit	Name	Type	Reset	Description
31..16	RPF_CNT	R	0	Receive pause frame counter
15..0	AEP_CNT	R	0	Counter for counting the packets with alignment error The counter is to count packets with CRC error and no octet-boundary discarded by the Ethernet controller

6.2.36 RUNT_CNT Counter Register

Offset: 0xC0

Table 6-46 RUNT_CNT Counter Register

Bit	Name	Type	Reset	Description
31..16	-	Reserved	0	-
15..0	RUNT_CNT	R	0	Counter for counting the received runt packets

6.2.37 CRCER_CNT and FTL_CNT Counter Register

Offset: 0xC4

Table 6-47 CRCER_CNT and FTL_CNT Counter Register

Bit	Name	Type	Reset	Description
31..16	CRCER_CNT	R	0	CRC error packet counter The counter counts the number of the octet-boundary frames discarded due to the CRC error.
15..0	FTL_CNT	R	0	Counter for counting received FTL packets

6.2.38 RCOL_CNT and RLOST_CNT Counter Register

Offset: 0xC8

Table 6-48 RCOL_CNT and RLOST_CNT Counter Register

Bit	Name	Type	Reset	Description
31..16	RLOST_CNT	R	0	Counter for counting the loss of the received packets (Due to RX FIFO full)
15..0	RCOL_CNT	R	0	Receive collision counter

6.2.39 Advance Interrupt Timer Control Register

Offset: 0xE0

Table 6-49 Advance Interrupt Timer Control Register

Bit	Name	Type	Reset	Description
31..10	Reserved	R	0	-
9..0	RXINT_RST	RW	0	This field defines the refresh time. The reset counter will be cleared once one packet is received by the Ethernet controller. The time unit is 1 RX cycle time. When RXINT_CNT = 0, the function will be disabled. If RXINT_THR = 0 and RXINT_CNT = 0, a receive interrupt will be issued when a packet is received by the Ethernet controller

6.3 Function Description

6.3.1 xMII Interface Connection

Ethernet controller supports the MII and GMII Interfaces. The following diagrams show the interface connections.

Figure 6-2 MII Connection

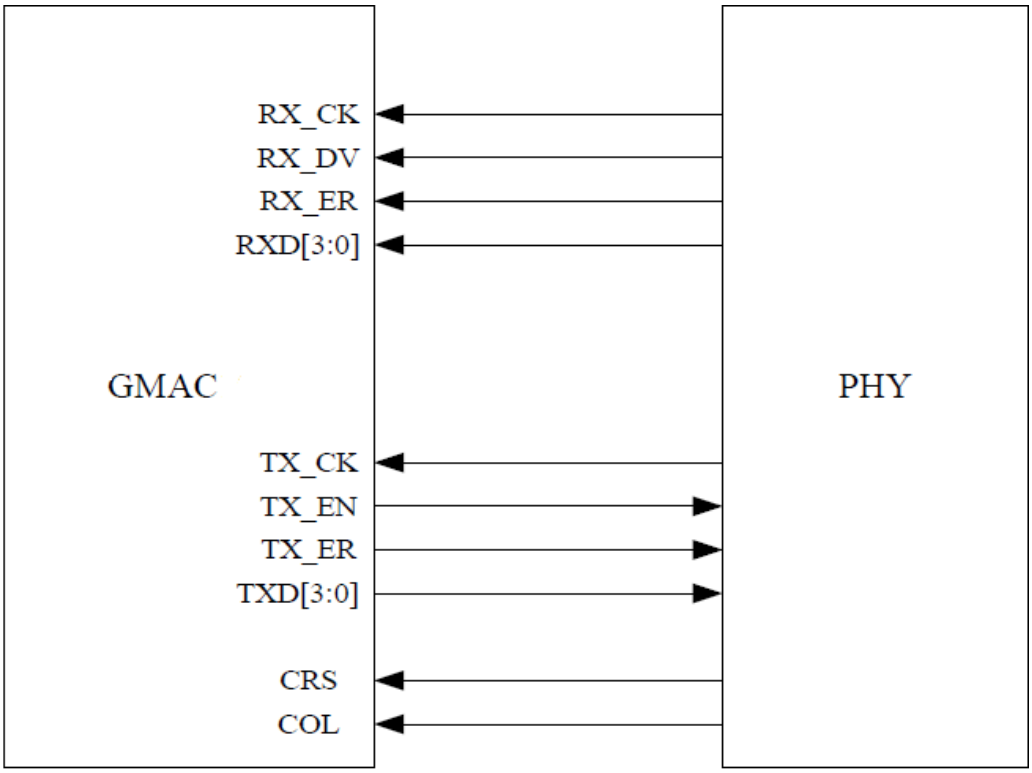
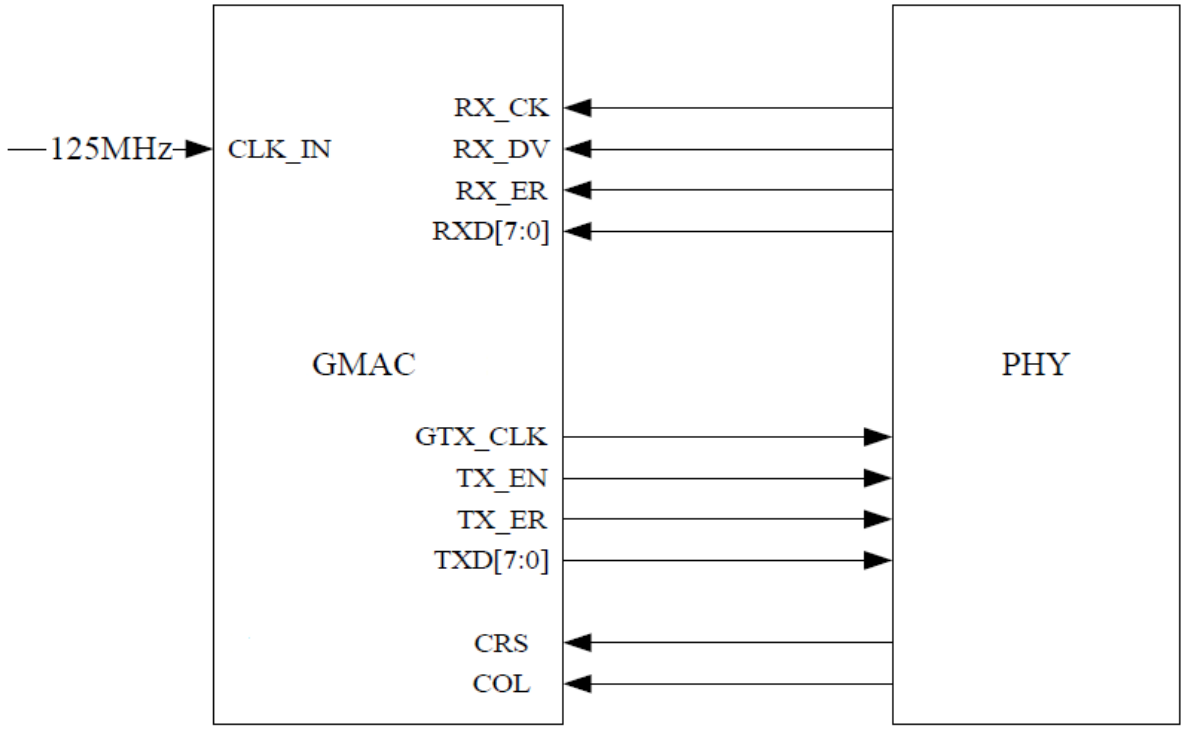
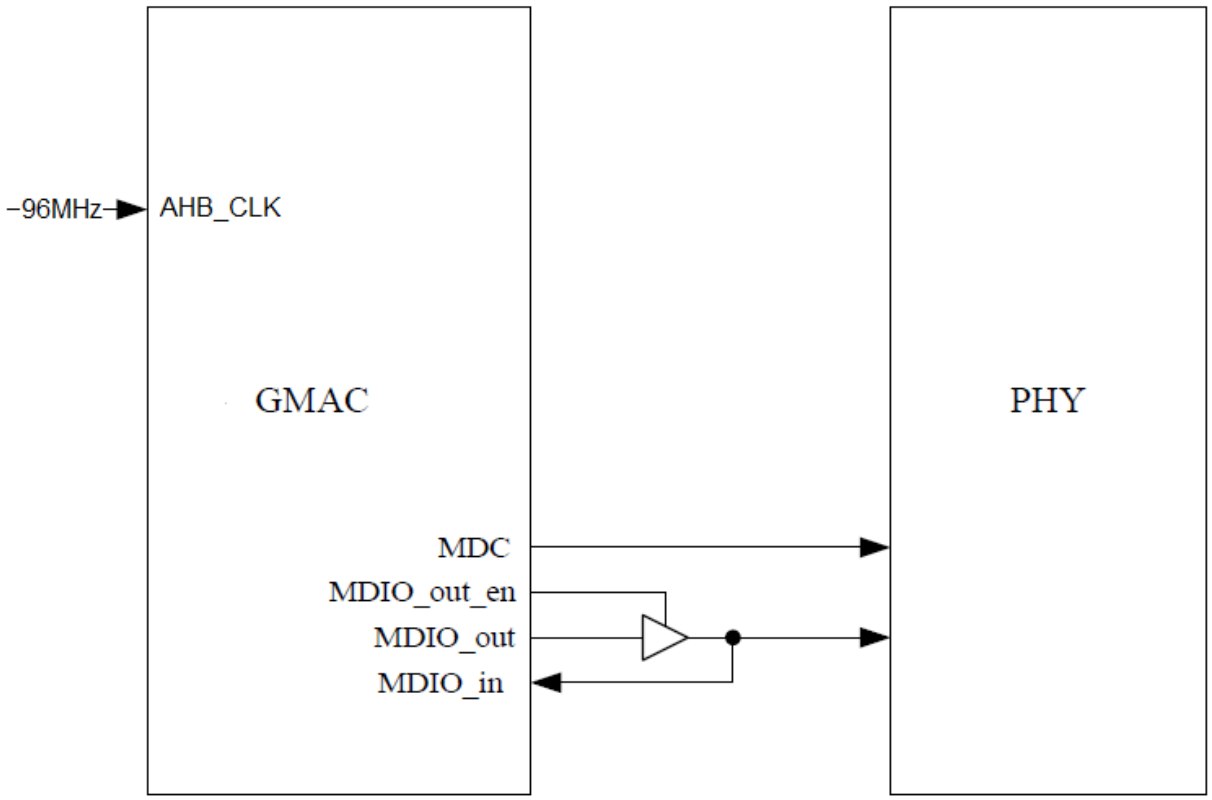


Figure 6-3 GMII Connection



Besides, the PHY access interfaces with the MDIO connection shown as the following:

Figure 6-4 MDIO Connection

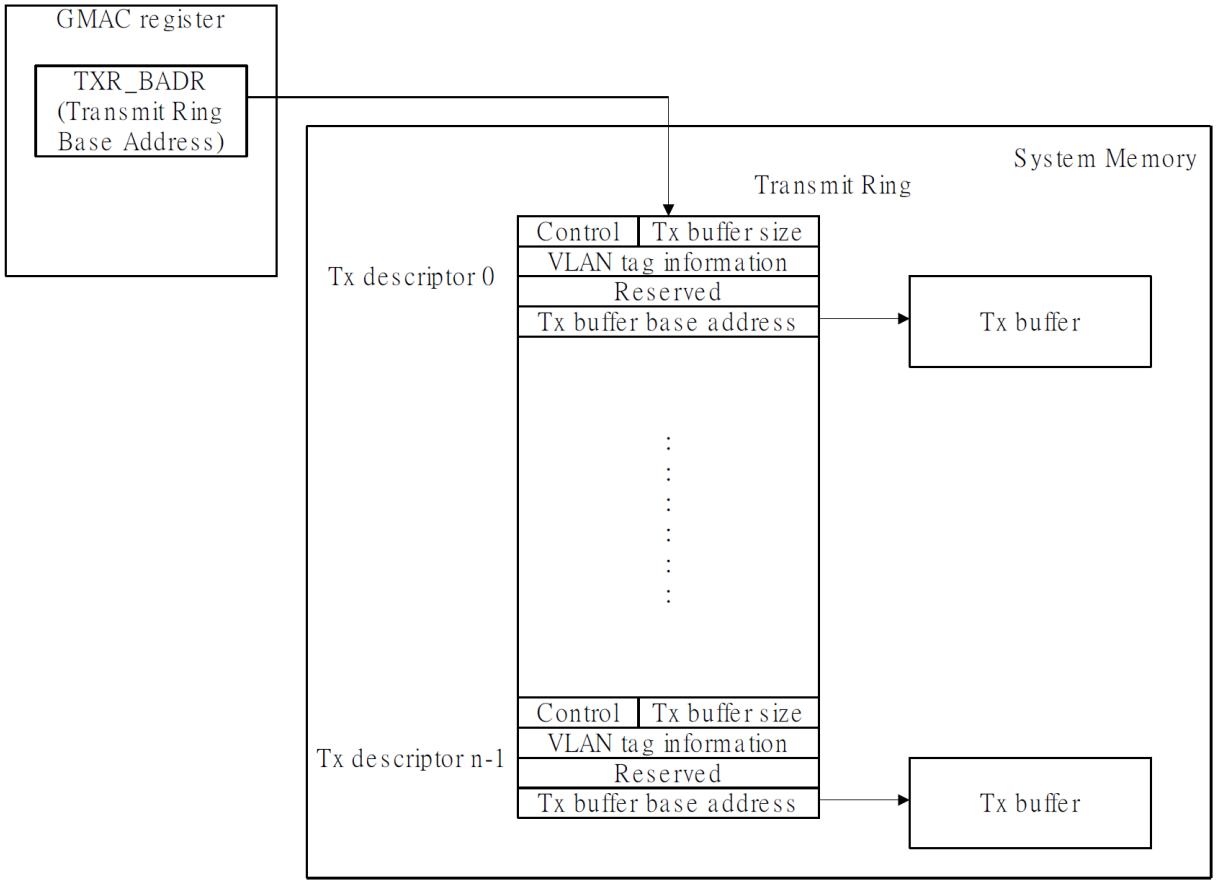


The clock domain of the MDIO function is the 96MHz AHB_CLK, AHB_CLK cannot be disabled or gated.

6.3.2 Transmit Descriptors and Data Buffers

The Ethernet controller uses a descriptor ring to manage the transmit buffers. The transmit descriptors and data buffers are all located in the system memory. The Ethernet controller moves the transmit packet data from the transmit buffers in system memory to the TX FIFO inside the Ethernet controller and then transmits the packet to Ethernet. The transmit descriptors reside in the system memory act as pointers to the transmit buffers.
Each transmit descriptor contains a transmit buffer. A transmit buffer consists of either an entire frame or part of a frame, but not multiple frames. The transmit descriptor contains the transmit buffer status and the transmit buffer that can only contain the transmit data. The Ethernet controller supports two descriptor rings for transmission. These descriptor rings are the normal-priority transmit ring and high-priority transmit ring. The normal priority transmit ring is for normal packet transmission; the high priority transmit ring is for high priority packet transmission. Higher priority packets can be put into the high priority transmit ring for quicker transmission.

Figure 6-5 Transmit Ring Descriptor Structure



The transmit descriptor structure is as follows.

Notes:

- The start address of each transmit descriptor must be 16-byte aligned.
- The maximum transmit packet size including CRC is 9216 bytes (9220 bytes if the VLAN tag is inserted).
- Ethernet controller only supports the IPV4 checksum offload. Software must be certain that the transmit packet is an IPV4 packet when software requests the Ethernet controller to do checksum offload.
- LLC packet is IEEE 802.3/802.2/SNAP format packet.
- Ethernet controller does not support the following two packets for the checksum offload:
  - IEEE 802.3 with IEEE 802.2 packet
  - IEEE 802.3 with 802.1Q and 802.2 packet.

TXDES0	TXDMA_OWN	CONTROL1	TX BUFFER SIZE
TXDES1	VLAN CONTROL	VLAN TAG CONTROL INFORMATION	
TXDES2	Reserved		
TXDES3	TX BUFFER BASE ADDRESS		

TXDES0 contains the control bits and transmit buffer size and descriptor ownership information.

Table 6-50 TXDES0

Bit	Name	Description
31	TXDMA_OWN	TXDMA_OWN – TXDMA ownership bit When set, it indicates that the descriptor is owned by the Ethernet controller. When reset, it indicates that the software owns the descriptor. The Ethernet controller clears this bit when it completes the frame transmission.
30	Reserved	-
29	FTS	FTS - First Transmit Segment descriptor When set, it indicates that this is the first descriptor of a TX packet.
28	LTS	LTS - Last Transmit Segment descriptor. When set, it indicates that this is the last descriptor of a TX packet.
27..20	Reserved	-
19	CRC_ERR	CRC_ERR - CRC error When CRC_ERR = 1 and DISCARD_CRCERR, bit 18 of MAC Control Register (WOLCR,6.2.18) = 1, TXDMA will discard the transmit packet and will not send it to Ethernet.
18..16	Reserved	-
15	EDOTR	EDOTR – End Descriptor of Transmit Ring When set, it indicates that the descriptor is the last descriptor of the transmit ring.
14	Reserved	-
13..0	TXBUF_SIZE	Transmit buffer size in byte The transmit buffer size cannot be zero.

TXDES1 contains the VLAN control bits and VLAN Control Information

Table 6-51 TXDES1

Bit	Name	Description
31	TXIC	TXIC - Transmit Interrupt on Completion When set, Ethernet controller will assert the transmit interrupt after the present frame has been transmitted. It is valid only when FTS=1 (TXDES0, Table 6-50) and bits14-8 (TXINT_THR, TXINT_CNT) of Interrupt Timer Control Register (ITC, 6.2.6) =0.
30	TX2FIC	TX2FIC - Transmit to FIFO Interrupt on Completion When set, Ethernet controller will assert the transmit interrupt after the present frame has been moved into TX FIFO. It is valid only when FTS=1.
29..23	Reserved	-
22	LLC_PKT	LLC_PKT – LLC packet When set, Ethernet controller will treat the packet as the LLC packet. It is valid only when FTS=1.
21..20	Reserved	-
19	IPCS_EN	IPCS_EN – IP checksum offload enable When set, Ethernet controller will offload the IP checksum. It is valid only when FTS=1.
18	UDPCS_EN	UDPCS_EN – UDP checksum offload enable When set, Ethernet controller will offload the UDP checksum. It is valid only when FTS=1.
17	TCPCS_EN	TCPCS_EN – TCP checksum offload enable. When set, Ethernet controller will offload the TCP checksum. It is valid only when FTS=1.
16	INS_VLAN	Insert VLAN Tag When set, 0x8100 (IEEE 802.1Q VLAN Tag Type) will be inserted after the source address, and two bytes VLAN_TAGC will be inserted after the IEEE 802.1Q VLAN tag type. When clear, the packet content will not be changed when transmitting to network. It is valid only when FTS=1.
15..0	VLAN_TAGC	VLAN Tag Control Information The 2-byte VLAN Tag Control Information contains information, from the upper layer, of user priority, canonical format indicator and VLAN ID. Please refer to IEEE

Bit	Name	Description
		802.1Q for more VLAN tag information. Bits[15:13]: User priority Bit 12: CFI (Canonical Format Indicator) Bits[11:0]: VID (VLAN Identifier) It is valid only when FTS=1.

Table 6-52 TXDES2

Bit	Name	Description
31..0	-	Reserved

Table 6-53 TXDES3

Bit	Name	Description
31..0	TXBUF_BADR	Transmit buffer base address

6.3.3 Receive Descriptors and Data Buffers

The Ethernet controller uses a descriptor ring to manage the receive buffers. The receive descriptors and data buffers are all located in the system memory. The Ethernet controller first stores the packet received from the network in the RX FIFO and then moves the received packet data to the receive buffers in system memory. The receive descriptors reside in the system memory act as pointers to the receive buffers.

There is a descriptor ring for reception. The base address of the receive ring is in the Receive Ring Base Address Register (RXR_BADR, 6.2.3). Each receive descriptor contains a receive buffer. A receive buffer consists of either an entire frame or part of a frame, but not multiple frames. The receive descriptor contains the receive buffer status and the receive buffer only contains the receive packet data.

The Ethernet controller supports the receive buffer base address as 2-byte alignment for the zero-copy feature. But there is a limitation when software program the receive buffer base address as 2-byte alignment. The limitation is the receive packet can only occupy one receive buffer. This means that the receive buffer size must be greater than the receive packet length. For example, if the length of the incoming packet is always less than 1600 bytes, software can program the receive buffer size as 1600 bytes. Then the limitation sustained when the receive buffer base address is 2-byte alignment. If software program the receive buffer base address as 8-byte alignment, then the limitation does not hold.

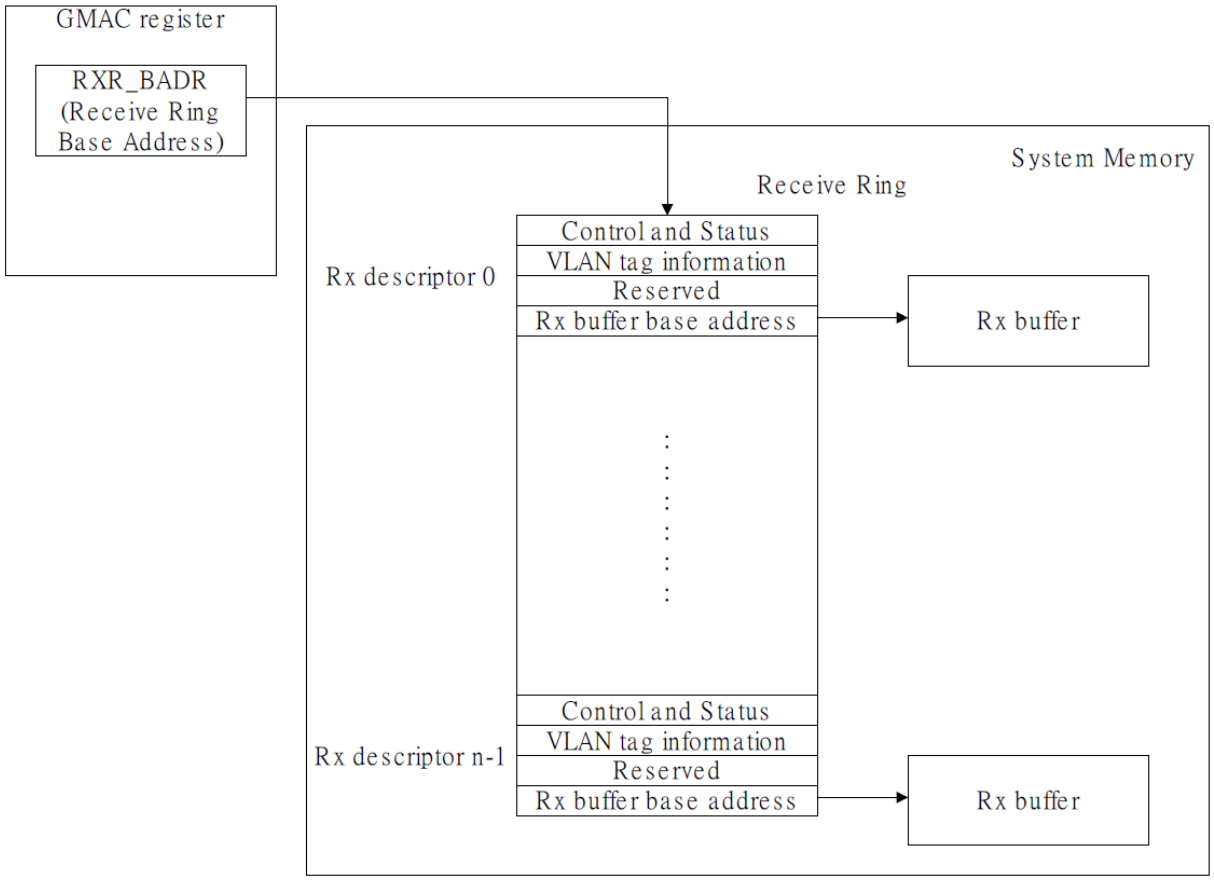


Figure 6-6 Receive Ring Descriptor Structure

The receive descriptor structure is as follows:

Notes:

- The start address of each receive descriptor must be 16-byte aligned.
- The maximum receive packet size is 9216 bytes (9220 bytes for packets with the VLAN tag).
- The Ethernet controller only supports the IPV4 checksum offload. If the incoming packet is not an IPV4 packet, the Ethernet controller will not perform the checksum verification. The IPCS_FAIL, UDPCS_FAIL, and TCPCS_FAIL fields are always zeros.
- LLC packet is IEEE 802.3/802.2/SNAP format packet.
- The Ethernet controller does not support the following two packets for the checksum offload. They are IEEE 802.3 with IEEE 802.2 packet and IEEE 802.3 with 802.1Q and 802.2 packets.
- The receive buffer size must be greater than the receive packet length when software programs the receive buffer base address as 2-byte alignment.

RXDES0	RXPKT_RDY	STATUS AND CONTROL
RXDES1	VLAN STATUS	VLAN TAG CONTROL INFORMATION
RXDES2	Reserved	
RXDES3	RX BUFFER BASE ADDRESS	

RXDES0 contains the receive frame status and descriptor ownership information.

Table 6-54 RXDES0

Bit	Name	Description
31	RXPKT_RDY	RXPKT_RDY – RX packet ready When clear, it indicates that the descriptor is owned by the Ethernet controller. When set, it indicates that the descriptor is owned by software. The Ethernet controller sets this bit when it completes the frame reception or when the receive buffer of the receive descriptor is full.
30	-	Reserved
29	FRS	FRS - First Receive Segment descriptor When set, it indicates that this is the first descriptor of a received packet. Bits25-0 are valid only when FRS=1.
28	LRS	LRS - Last Receive Segment descriptor When set, it indicates that this is the last descriptor of a received packet.
27..26	Reserved	-
25	PAUSE_FRAME	PAUSE_FRAME – Pause frame When set, it indicates that the receive packet is a pause frame.
24	PAUSE_OPCODE	PAUSE_OPCODE – Pause frame OP code When set, it indicates that there is pause frame OP code in the receive packet.
23	FIFO_FULL	FIFO_FULL – FIFO full When set, it indicates that RX FIFO is full when the packet is received.
22	RX_ODD_NB	RX_ODD_NB - Receive Odd Nibbles When set, it indicates that a packet is received with odd nibbles.
21	RUNT	RUNT - Runt packet When set, it indicates that the received packet length is less than 64 bytes.
20	FTL	FTL - Frame Too Long When set, it indicates that the received packet length exceeds the long frame length. If JUMBO_LF = 0, the long frame length will be 1518 (1522 for the receive packet with the VLAN tag) bytes. If JUMBO_LF = 1, the long frame length will be 9216 (9220 for the receive packet with the VLAN tag) bytes.
19	CRC_ERR	CRC_ERR – CRC error When set, it indicates that the CRC error occurs on the received packet.
18	RX_ERR	RX_ERR - Receive error When set, it indicates that receive error happens when receiving a packet.
17	BROADCAST	BROADCAST - Broadcast frame. When set, it indicates that the received packet is a broadcast frame.
16	MULTICAST	MULTICAST - Multicast frame When set, it indicates that the received packet is a multicast frame.
15	EDORR	EDORR - End Descriptor of Receive Ring When set, it indicates that the descriptor is the last descriptor of the receive ring.
14	Reserved	-
13..0	VDBC	VDBC – valid data byte count The field indicates the valid data in the receive buffer. The unit is 1 byte.

RXDES1 contains the VLAN status bits and VLAN Tag Control Information.

Table 6-55 RXDES1

Bit	Name	Description
31..28	Reserved	-
27	IPCS_FAIL	IPCS_FAIL – IP checksum failure When set, Ethernet controller detects IP checksum failure. The field is valid only when the FRS=1.
26	UDPCS_FAIL	UDPCS_FAIL – UDP checksum failure When set, Ethernet controller detects UDP checksum failure. The field is valid only when the FRS=1.
25	TCPCS_FAIL	TCPCS_FAIL – TCP checksum failure When set, Ethernet controller detects TCP checksum failure. The field is valid only when the FRS=1.
24	VLAN_AVA	VLAN_AVA – VLAN Tag Available When set, the receive packet is an IEEE 802.1Q VLAN Tag Type. The field is valid only when the FRS=1.

Bit	Name	Description
23	DF	Datagram Fragment When set, the IP packet is not fragment. When clear, the IP packet may fragment. Checksum status is valid only when DF=1.
22	LLC_PKT	LLC_PKT – LLC packet When set, it indicates that the receive packet is the LLC packet. The field is valid only when FRS=1.
21..20	PROTL_TYPE	PROTL_TYPE – Protocol Type These two bits indicate which protocol in the receive packet. 00: Not IP protocol 01: IP protocol 10: TCP/IP protocol 11: UDP/IP protocol The field is valid only when FRS=1.
19..16	Reserved	-
15..0	VLAN_TAGC	VLAN Tag Control Information If the receive packet contains the VLAN tag, Ethernet controller will extract four bytes from the receive packet. The four bytes data contains 0x8100 and two bytes VLAN Tag Control Information. FTGMAC100_S will move the two bytes VLAN Tag Control Information to this field. The 2-byte VLAN Tag Control Information contains information, from the upper layer, of user priority, canonical format indicator, and VLAN ID. Please refer to IEEE 802.1Q for more VLAN tag information. Bits[15:13]: User priority Bit 12: CFI (Canonical Format Indicator) Bits[11:0]: VID (VLAN Identifier) The field is valid only when FRS = 1.

Table 6-56 RXDES2

Bit	Name	Description
31..0	Reserved	-

Table 6-57 RXDES3

Bit	Name	Description
31..0	RXBUF_BADR	Receive buffer base address Receive buffer base address must be at least 2-byte alignment. This means that RXBUF_BADR[0] must be zero.

6.3.4 Transmitting Packets

When software wants to transmit a packet to Ethernet, it moves the packet data into the transmit buffer first. Then software writes the packet length and position into the transmit descriptor and triggers the Ethernet controller to send the packet. After the entire packet has been moved into the TX FIFO, Ethernet controller begins to transmit it to Ethernet. When the packet has been transmitted, Ethernet controller asserts interrupt to notify software that the packet has been transmitted successfully. Higher priority packets can be put into the high priority descriptor for quicker transmission.

6.3.5 Receiving Packets

When there is an incoming packet, the Ethernet controller first saves the received packet in the RX FIFO if the address check result is correct. After the incoming packet is successfully

saved in RX FIFO, the Ethernet controller initiates Direct Memory Access (DMA) function to move the received packet data from the RX FIFO to the system memory. Then Ethernet controller asserts interrupt to notify software that the packet has been received successfully.

6.3.6 Ethernet Address Filtering

The Ethernet controller can be set to recognize any one of the Ethernet receive address groups described in the following table.

- RX_BROADPKT: Bit 17 of MAC Control Register (6.2.12)
- RX_MULTIPKT: Bit 16 of MAC Control Register (6.2.12)
- RX_HT: Bit 15 of MAC Control Register (6.2.12)
- RX_ALLADR: Bit 14 of MAC Control Register (6.2.12)

Table 6-58 Ethernet Address Filtering

RX_ ALLADR	RX_ MULTIPKT	RX_ BROADPKT	RX_HT	GROUP	DESCRIPTION
0	0	0	0	A	Ethernet controller receives a frame of which the destination address exactly matches the MAC_ADR (6.2.3) of the Ethernet controller
0	0	0	1	B	Ethernet controller receives a frame of which the destination address exactly matches the MAC_ADR (6.2.3) of the Ethernet controller or a frame of which the destination address is a multicast address, or a frame which passes the address filtering of the multicast address hash table in the Ethernet controller
0	0	1	0	C	Ethernet controller receives a frame of which the destination address exactly matches the MAC_ADR (6.2.3) of the Ethernet controller, or a frame of which the destination address is a broadcast address.
0	0	1	1	D	Ethernet controller receives a frame of which the destination address exactly matches the MAC_ADR (6.2.3) of the Ethernet controller, or a frame of which the destination address is a multicast address, or a frame which passes the address filtering of the multicast address hash table in the Ethernet controller, or a frame of which the destination address is a broadcast address.
0	1	X	X	E	Ethernet controller receives a frame of which the destination address exactly matches the MAC_ADR (6.2.3) of the Ethernet controller, or a frame of which the destination address is a multicast address.
1	X	X	X	F	Ethernet controller supports reception of all frames on the network regardless of their destination address.

6.3.7 DMA Arbitration Scheme

The DMA arbitration scheme is decided by RX_THR_EN (Bit 6 of DBLAC, 6.2.8). When RX_THR_EN = 0, the DMA arbitration scheme does a fair arbitration between TXDMA and RXDMA. If both TXDMA and RXDMA request the DMA channel at the same time, the one last using the DMA channel has lower priority to get the DMA channel.
When RX_THR_EN is set, if the used space in the RX FIFO is larger than or equal to the RXFIFO_HTHR (Bits 5-3 of of DBLAC, 6.2.8), the RXDMA has higher priority over the TXDMA by using the DMA channel. The RXDMA keeps the higher priority until the used space in the RX FIFO is less than or equal to the RXFIFO_LTHR (Bits 2-0 of of DBLAC, 6.2.8). Then the TXDMA gets higher priority over the RXDMA. So software must set RXFIFO_HTHR to be larger than RXFIFO_LTHR to keep the Ethernet controller working correctly.

6.3.8 Wake-On-LAN

The Ethernet controller supports the Wake-On-LAN function. The Wake-On-LAN function supports three wake-up events: Link status change, magic packet, and wake-up frame.

6.3.9 Link Status Change

Link status change refers to the event where the link state to Ethernet changes. PHY offers a phy_linksts signal. If the link state to Ethernet changes, the state of phy_linksts will also change.
If is put into the power-saving mode with the enabled link status change mode, the link status change will be treated as a wake-up event.

6.3.10 Magic Packet

A magic packet contains a specific sequence consisting of 16 duplications of network the adaptor node address without breaks. The specific sequence must be preceded by 6 bytes of 0xFF. The format of a magic packet goes as follows:

DA+SA +…+ 6 * (0xFFh) + …+16 * (network adaptor’s node address)+…

If the Ethernet controller is put into the power-saving mode with the enabled magic packet mode, a magic packet will be treated as a wake-up event.

6.3.11 Wake-up Frame

The purpose of wake-up frame is to wake up a system when another machine on the network needs to communicate with the system. It does not require the application running on the remote machine to send a special wake-up frame pattern. Instead, when the Ethernet controller is in wake-up frame mode, it tries to identify certain interesting frames that are sent by existing network protocols. Some examples are NETBIOS name lookups and ARP requests.
Before putting the Ethernet controller into the wake-up frame mode, the system should pass to the driver a list of wake-up frames that will wake up the system and the corresponding

byte masks. Each byte mask defines which bytes of the incoming frames should be compared with the corresponding wake-up frame in order to determine whether or not to accept the incoming frame as a wake-up event.

Table 6-59 Wake-up Frame Format

Wake-up frame format								
Byte content	0x00	0x25	0x67	0x90	0x44	0xA3	0x6C	..
Byte mask	0	0	0	1	1	0	1	…

There are two ways to identify if a received packet is a wake-up frame. Signature matching is used in the Ethernet controller:

- Exact matching: The Ethernet controller needs many registers to store the entire byte content and byte mask for each wake-up frame. When a packet arrives from the network, Ethernet controller checks those bytes of the incoming frame that correspond to bits set to 1 in the byte mask for each wake-up frame. If the check result is ok for any wake-up frame and if the incoming frame passes the standard CRC check, Ethernet controller would treat it as a wake-up event.
- Signature matching: The Ethernet controller needs a CRC generation circuit and registers to save the entire byte mask and 4-byte CRC register for each wake-up frame. The driver calculates a CRC value based on those bytes of any wake-up frame that correspond to bits set to 1 in the byte mask. The driver stores the resulting value and corresponding byte mask into the Ethernet controller. When wake-up frame mode is enabled as a frame arrives from the network, each CRC generator calculates a CRC value based on those bytes of the incoming frame that correspond to bits set to 1 in that CRC generator byte mask. If the calculated value matches the stored value for any wake-up frame and the incoming frame passes the standard CRC check, the Ethernet controller would treat it as a wake-up event.

6.3.12 Power-down Mode

The Ethernet controller provides a power-down mode that significantly reduces power dissipation when its power state is not programmed into D0 power state.
The following is a brief account of the features of the power-down mode:

- Ethernet controller does not assert interrupt when in the power-down mode.
- Ethernet controller does not transmit packets to Ethernet.
- Ethernet controller does not save received packets in the RX FIFO.
- Ethernet controller asserts WOL when a wake-up event happens.

6.3.13 Flow Control

The Ethernet controller implements flow control function. It supports IEEE802.3x flow control for full-duplex mode and backpressure for half-duplex mode.
The IEEE802.3x flow control is used in full-duplex mode. When A and B are reciprocally transmitting and receiving packets in the full-duplex mode, if the RX FIFO in B is nearly full, B sends a pause frame to A in order to avoid loss of received packet. Then A is inhibited from

transmitting packets for a specified period of time. B consumes the received data during the specified period of time. A continues to send packets to B after the pause time has lapsed. Here is a brief account of the features of the flow control in full-duplex mode:

- Software configures the pause time of the pause frame.
- Ethernet controller sends the pause frame according to the low/high threshold of RX FIFO.
- Software sends the pause frame by writing the register.

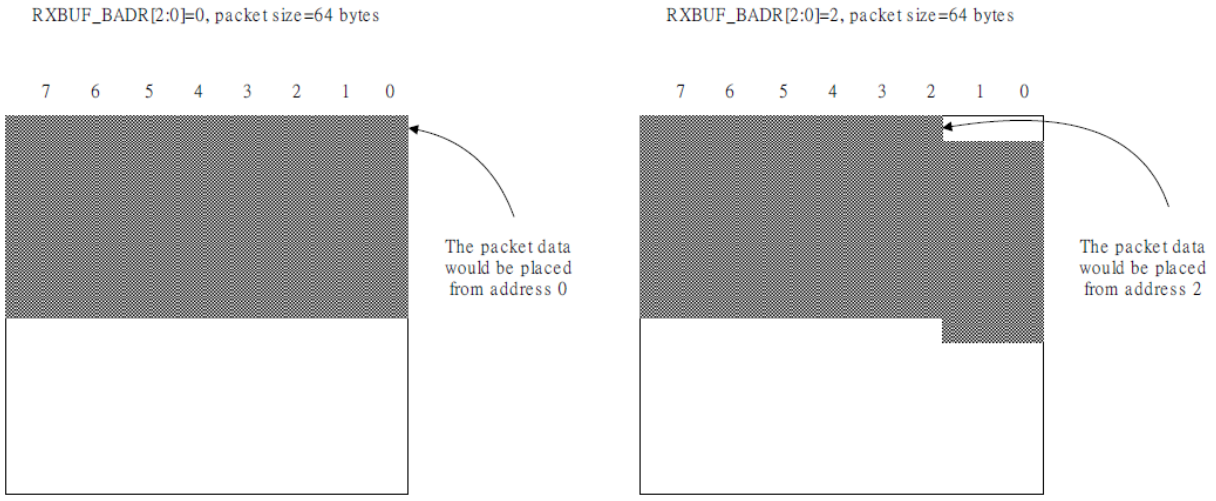
The back pressure mode is used in the half-duplex operation. When A is transmitting and receiving packets in the half-duplex mode, if the RX FIFO in A is nearly full, A will send a jam pattern to generate collisions to avoid incoming packets from being saved into the RX FIFO. A consumes the received data as soon as possible during the period of time. A does not send a jam pattern to receive packets again when the RX FIFO is not nearly full. Here is a brief account of the features of the back pressure mode:

- Software configures the length of the jam.
- Ethernet controller sends jam according to the low/high threshold of RX FIFO

6.3.14 Zero-copy

With the Ethernet controller zero-copy function, system does not need to perform data movement for packet header alignment. The Ethernet controller DMA will place the incoming packet data from 2-byte-aligned address if the receive buffer is programmed as 2-byte-aligned address. Figure 5-3 shows an example of packet placement in a little-endian system. The Ethernet controller supports the receive buffer base address is 2-byte alignment for the zero-copy function. But there is a limitation when software programs the receive buffer base address as 2-byte alignment. The limitation is the receive packet can only occupy one receive buffer. This means that the receive buffer size must be greater than the receive packet length. For example, if the length of the incoming packet is always less than 1600 bytes, software can program the receive buffer size as 1600 bytes. Then the limitation can be achieved when the receive buffer base address is 2-byte alignment. If software programs the receive buffer base address as 8-byte alignment, then the limitation does not hold.

Figure 6-7 Packet Data Placement for Different Receive Buffer Address



6.3.15 Supported Ethernet Frame Type for Checksum Offload

Two Ethernet frame formats are supported in order to correctly identify the IP and TCP/UDP headers. The Ethernet controller supports Ethernet Type II format and IEEE 802.3/802.2/SNAP format as shown below. VLAN tagging is also supported on top of these two frame formats.

Table 6-60 Ethernet Type II

Destination Addr	Source Address	Type	Data	CRC
6-byte	6-byte	2-byte	n-byte	4-byte

Table 6-61 IEEE802.3/802.2/SNAP

Destination Address	Source Address	Length	DSAP	SSAP	Control	OUI	Type	Data	CRC
6-byte	6-byte	2-byte	1-byte	1-byte	1-byte	3-byte	2-byte	n-byte	4-byte
where DSAP=0xAA, SSAP=0xAA, Control=0x03, OUI=0x000000									

6.4 Initialization Information

This chapter discusses the procedure for transmitting and receiving frames, as well as the procedure for entering and exiting power down mode.

6.4.1 Frame Transmitting Procedure

The frame transmitting procedure is as follows:
Initialization:

- Set LOOP_EN (MACCR bit6, 6.2.12), GMAC_MODE (MACCR bit9, 6.2.12) and SPEED_100 (MACCR bit19, 6.2.12) to proper setting

- Set SW_RST (MACCR bit31, 6.2.12) = 1 to do software reset. It takes about 200 system clocks for hardware to finish the software reset
- The TX/RX FIFO size is 2kByte.Allocate the system memory for the transmit descriptor ring and transmit buffer
- Initialize the transmit descriptor ring
- Set the Normal Priority Transmit Ring Base Address Register (NPTXR_BADR, 6.2.2) to the base address of the normal priority transmit descriptor ring in the system memory
- Set the High Priority Transmit Ring Base Address Register (HPTXR_BADR, 6.2.5) to the base address of the high priority transmit descriptor ring in the system memory if necessary
- Set the Interrupt Enable Register (IME, 6.2.2)
- Set the MAC Address Register (MAC_MADR, 6.2.3)
- Set the Multicast Address Hash Table Register (MAHT0, 6.2.5)
- Set the Interrupt Timer Control Register (ITC, 6.2.6) to select the manner of the transmit interrupt.
- Set the Automatic Polling Timer Control Register (APTC, 6.2.7) to select the manner of transmit poll
- Set the Transmit Priority Arbitration and FIFO Control Register (TPAFCR, 6.2.10) to set transmit priority arbitration and proper TX/RX FIFO size in use
- Set the DMA Burst Length and Arbitration Control Register (DBLAC, 6.2.8) to set proper TX/RX descriptor size and DMA burst length
- Set the MAC Control Register (MACCR, 6.2.12) to set valid configuration for the Ethernet controller and to enable the transmit channel

Transmit procedure:

- Software checks if the remained normal-priority transmit descriptors is sufficient for the next packet transmission. If not, software needs to wait until the transmit descriptors are sufficient.
- Prepare the transmit packet data to the transmit buffer.
- Set the normal priority transmit descriptor.
- Write the Normal Priority Transmit Poll Demand Register (NPTXPD, 6.2.7) to trigger the Ethernet controller to poll the transmit descriptor if necessary when the packet is put in the normal priority transmit ring.
- Wait for interrupt.
- When interrupt occurs, software checks at Register ISR (6.2.1) if it is a transmit interrupt. If BIT4 = 1, it means that the packet has been transmitted to network successfully. If Bit7 = 1, it means that the packet has been aborted during transmission due to late collision or excessive collision or under-run.
- Steps 1 through 6 are for the normal packets in the normal-priority transmit ring. If software wants to transmit the high-priority packets, repeat these steps for the high-priority transmit ring.

Notes:

- When setting the transmit descriptor, TXDES0 must be set last. Thus, the setting procedure should be as follows:
  - Set TXDES3
  - Set TXDES2
  - Set TXDES1
  - Set TXDES0
- When preparing a transmit packet which contains more than one transmit descriptors, the first transmit descriptor must be the last set descriptor of the transmit packet.

6.4.2 Frame Receiving Procedure

The frame receiving procedure is as follows:

Initialization:

- Set LOOP_EN (MACCR bit6, 6.2.12), GMAC_MODE (MACCR bit9, 6.2.12) and SPEED_100 (MACCR bit19, 6.2.12) to proper setting
- Set SW_RST (MACCR bit31, 6.2.12) = 1 to perform software reset. It takes about 200 system clocks for hardware to finish the software reset
- The TX/RX FIFO size is 2kByte.Allocate the system memory for the receive descriptor ring and receive buffer
- Initialize the receive descriptor ring
- Set the Receive Ring Base Address Register (RXR_BADR, 6.2.3) to the base address of the receive descriptor ring in the system memory
- Set the Interrupt Enable Register (IME, 6.2.2)
- Set the MAC Address Register (MAC_MADR, 6.2.3)
- Set the Multicast Address Hash Table Register (MAHT0, 6.2.5)
- Set the Interrupt Timer Control Register (ITC, 6.2.6) to select the manner of the receive interrupt
- Set the Automatic Polling Timer Control Register (APTC, 6.2.7) to select the manner of receive poll
- Set the Transmit Priority Arbitration and FIFO Control Register (TPAFCR, 6.2.10) for the transmit priority arbitration and proper TX/RX FIFO size in use
- Set the DMA Burst Length and Arbitration Control Register (DBLAC, 6.2.8) for the proper TX/RX descriptor size and DMA burst length
- Set the MAC Control Register (MACCR, 6.2.12) to set valid configuration for FTGMAC100_S and to enable receive channel
- Write the Receive Poll Demand Register (RXPD, 6.2.1) to trigger the Ethernet controller for polling the receive descriptor

Receive procedures:

- Wait for interrupt.
- When interrupt occurs, software checks at Register ISR (6.2.1) if it is a receive interrupt. If Bit0=1, it means the packet has been moved to the receive buffer

successfully. Then software needs to fetch the receive descriptor to get the receive packet until the owner bit of the next receive descriptor does not belong to software.

- Software releases the receive descriptors to the Ethernet controller after accessing the received packet.
- If the receive automatic poll function is disabled, software needs to write Receive Poll Demand Register (RXPD, 6.2.1) to trigger the Ethernet controller to poll the receive descriptor.

6.4.3 Procedures of Entering and Exiting Power-down Mode

The procedure for entering the power-down mode is as follows:

- Set TXDMA_EN (MACCR, 6.2.12) bit0=0
- Poll DMA/FIFO State Register (DMAFIFOS, 6.2.9) to wait for empty TX FIFO
- Set TXMAC_EN (MACCR, 6.2.12) bit2=0 to stop transmission
- Set RXMAC_EN (MACCR, 6.2.12) bit3=0
- Poll the DMA/FIFO State Register (DMAFIFOS, 6.2.9) to wait for empty RX FIFO
- Set RXDMA_EN (MACCR, 6.2.12) bit1=0 to stop reception
- Program Wake-up Frame CRC Register (WFCRC, 6.2.20), Wake-up Frame Byte Mask 1st Double-word Register (WFBM1, 6.2.21), Wake-up Frame Byte Mask 2nd Double-word Register (WFBM2, 6.2.22), Wake-up Frame Byte Mask 3rd Double-word Register (WFBM3, 6.2.23), and Wake-up Frame Byte Mask 4th Double-word Register (WFBM4, 6.2.24), if software wants to support wake-up frame event in power saving mode
- Write 0xFFFF_FFFF to clear the Wake-On-LAN Status Register (WOLSR, 6.2.19)
- Program the requested wake-up events and power state into the Wake-On-LAN Register (WOLCR, 6.2.18) to let the Ethernet controller enter the power-saving mode
- Set RXMAC_EN (MACCR, 6.2.12) bit3=1 to enable the reception

The procedure for exiting the power-down mode is as follows:

- Wait for occurrence of wake-up events
- Set RXMAC_EN (MACCR, 6.2.12) bit3=0 to stop the reception
- Read the Wake-On-LAN Status Register (WOLSR, 6.2.19) to check which wake-up event happened
- Program the Wake-On-LAN Register (WOLCR, 6.2.18) to let Ethernet controller exit the power-saving mode
- Set SW_RST (MACCR, 6.2.12) bit31=1 to reset Ethernet controller
- Check if SW_RST (MACCR, 6.2.12) bit31 = 0 to make sure that Ethernet controller has finished reset
- Re-initialize Ethernet controller to transmit and receive packets

6.4.4 MII Management Interface

Ethernet controller contains an MII Management Interface for an MII compliant PHY device. This allows control and status parameters to be passed between the Ethernet controller and PHY by MDIO and MDC; thereby, reducing the number of control pins required for PHY mode control. The protocol consists of the bit stream that is sampled at the rising edge of MDC; the bit stream format is described below.

Table 6-62 PHY Bit Stream Format

	PRE	ST	OP	PHYAD	REGAD	TA	DATA	IDLE
READ	1..1	01	10	AAAAA	RRRRR	Z0	D..D(16)	Z
WRITE	1..1	01	01	AAAAA	RRRRR	10	D..D(16)	Z

- PRE (Preamble) - At the beginning of each transaction, the Ethernet Controller sends a sequence of 32 contiguous logic bits on MDIO with 32 corresponding cycles on MDC to establish synchronization
- ST (Start of Frame) - A start of frame with 01
- OP (Operation Code) - 10 denotes read; 01 denotes write.
- PHYAD (PHY Address) - A 5-bit address of PHY device. The first bit transmitted is MSB
- REGAD (Register Address) - A 5-bit address of PHY register. The first bit transmitted is MSB
- TA (Turn Around) - The turn-around is a 2-bit time to avoid contention during a transaction
- DATA (Data) - The data field is 16-bit.
- IDLE (Idle) - The condition on MDIO is high impedance state.

6.4.5 Multicast Address Hash Table Filtering

There is a 64-bit multicast address hash table in the Ethernet controller. It is used to filter the incoming packet with a multicast destination address. For an incoming frame with a multicast destination address, Ethernet controller uses the packet destination address (The first six bytes of the received packet) and the standard Ethernet Cyclic Redundancy Check (CRC) function to calculate a 32-bit CRC value. Then, Ethernet controller uses the most significant six bits of the CRC value as the bit address index into the multicast address hash table. If the indexed bit in the table is set, the frame will pass the address filtering of the multicast address hash table. If the indexed bit in the table is cleared, the frame will fail to pass the address filtering of the multicast address hash table. If RX_HT_EN = ‘1’ (MACCR, 6.2.12) and the frame passes the address filtering of the multicast address hash table, the frame is accepted; otherwise the frame is rejected.

7 DDR2 Memory Controller

Offset: 0x7B00_0000

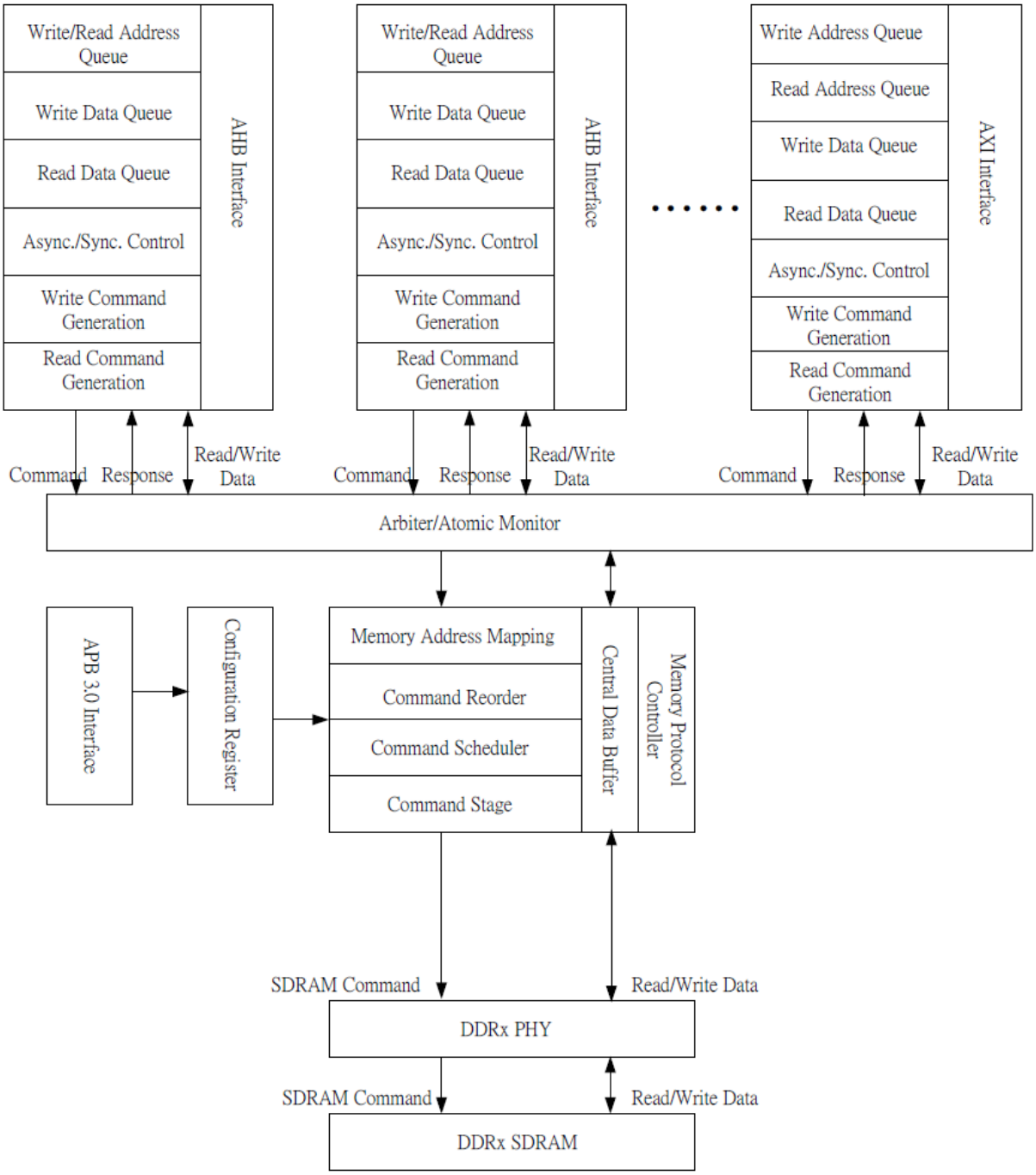
The DDR2 memory controller supports DDR2 SDRAMs. This controller uses a burst length of eight for the memories to accelerate the read and write speeds. The memory controller prefetches the sequential read data for the burst read commands to enhance the data transfer rate.

- Supports a maximum of 256M bytes[12)]
- Uses a burst length of eight for memories to speed-up read and write operations
- Enters DDR2 SDRAM self-refresh mode by using hardware handshake signal or software configuration
- Prefetches internal sequential read data for AHB burst read commands
- Supports 16 entries data FIFO depths
- Supports MA tables corresponding to different sizes and types
- Supports 16-bit memory device width by 200MHz/800MBit
- Provides two arbitration strategies for channel arbitrations:
    - Two-level round-robin arbitration with channel grant counts
    - Read-write grouping arbitration with RW commands grant counts
- Automatically enters SDRAM power-down mode when controller idles exceeds user-defined clock cycles
- Supports x16 DDR2 interface (one x16 or two x8 SDRAM chips possible)

2) For memory configuration see chapter 4.9 in document *ANT1000/1001 Data Sheet*

7.1 Overview

Figure 7-1 Block diagram of DDR2 Memory Controller



The DDR2 memory controller is separated into several blocks, including the AHB interface, AXI interface, arbiter and memory protocol controller. These blocks are described below.

7.1.1 Channel allocation

Channels are numbered from CH0 to CH5. The following listing shows the mapping of the indvidual channels.

Channel0 = Data/Instruction ARM CA5 (AXI, 96MHz, 64Bit)
Channel1 = ARM CA5 Peripherie bus (AHB, 96MHz, 32Bit)

Channel2 = Advanced RAM access controller for the PBM0/1 (AHB, 96MHz, 32Bit)
Channel3 = 2nd channel of main FTDMAC (AHB, 96MHz, 32Bit)
Channel4 = Realtime Ethernet Switch (AHB, 200MHz, 64Bit)
Channel5 = FTGMAC (AHB, 96MHz, 32Bit)

7.1.2 AHB Interface

The AHB interface comprises an AHB slave interface, which performs the following functions:

- Asynchronous FIFO depth control. The supported FIFO depths include:
  - Command FIFO: Depth of 4
  - Read data FIFO: Depth of 16
  - Write data FIFO: Depth of 16
- Read command generation to prefetch data for the AHB read bursts
  - A programmable register is provided to define the prefetch length for the AHB unspecified-length (INCR) read bursts.
- Provides the command flushing mechanism for handling the cross-channel data coherence problem

7.1.3 AXI Interface

The AXI interface comprises an AXI slave interface, which performs the following functions:

- Write command generation
- Read command generation
- AXI write response, read response, and exclusive access response generations. AXI write response will be issued after the command going into the arbiter.
- Merge write/read data for transfer efficiency

7.1.4 Arbiter

The two-level round-robin arbiter is used to arbitrate the requests from the AHB/AXI command queue. The commands from the highest priority channel will be issued to the memory protocol controller. An advanced arbitration mechanism of the read-write grouping is used to reduce the “Write-to-Read” wait cycles of the DDR2 SDRAMs.

7.1.5 Memory Protocol Controller

The memory protocol controller comprises the memory address mapping, page table maintenances, command reorder and scheduler and inter-command DDR2 timing parameter check.

- Memory address can map to different DDR2 SDRAM types and memory interface widths.
- Command reorder can choose the proper DDR command to reduce the bank-conflict or RW turn-around performance impact
- Command scheduler can check the inter-command timing parameters to meet the DDR2 SDRAM specifications and perform a command scheduling, which can choose the proper commands to maximize the DDR2 bus efficiency.

- Command stage is the actual DDR2 SDRAM command stage, which contains the pre_command register output of the command scheduler.
- Elastic fifo can separate MClk and POSTCTLCK domains clock tree to reduce the clock tree length of POSTCTLCLK clock domain.
- DDR2 Memory Controller accesses DDR2 SDRAM per byte basis, users can do bit swaps within the same byte for the PCB trace routing flexibilities.

7.1.6 Self-refresh Mode

DDR2 memory controller provides two operation modes: Normal mode and self-refresh mode. When the system is powered on and a hardware reset is terminated, DDR2 memory controller will be initialized by programming the control register in the controller to enter the normal mode. In the normal mode, all operations run at the full speed. In order to save power, the self-refresh mode is implemented to minimize the power dissipation of the memory module. The self-refresh mode can be entered by software programming. To properly operate the self-refresh mode, users should follow the steps below. The AHB/AXI commands should be stopped before issuing the self-refresh request to ensure proper operation.

7.1.6.1 Software Self-refresh Request

When bit 2 of the controller register 0x04 (7.2.2 MemCmd_srf) is set to ‘1’, the controller will automatically enter the self-refresh mode after all outstanding commands are terminated. Consequently, the external memory modules will also enter the self-refresh mode.

7.1.6.2 Exit from Self-refresh Mode

- To exit from the self-refresh mode by software (by programming bit 3 of 7.2.2 MemCmd_srf), users need to ensure that DDR2 SDRAM is in the self-refresh mode (Read bit 10 of 7.2.2 MemCmd_srf) before issuing an existing self-refresh command by software.
- To wake up the memory module by hardware, users can choose one of the AHB/AXI masters to issue a read/write command to the memory controller. Consequently, the memory controller will automatically wake up, exit from the self-refresh mode, and then enter the normal operation mode.
- After exiting the self-refresh mode, DDR2 memory controller will assert all bytes of phy_update_en_byten if phy_con_update (bit 13 of 7.1.5 PHYCR0) is set to ‘0’.
- After exiting the self-refresh mode, DDR2 memory controller will add one post-refresh command in the auto-refresh controller.

7.1.6.3 Warm-start Function

The warm-start function can be used when DDR is in the self-refresh mode while the DDR2 memory controller is powered off. After DDR2 memory controller entering the self-refresh mode, users can power off the DDR2 memory controller while keeping CKE low by powering on the domain circuit to keep the content of DDR. When DDR2 memory controller is powered on and reset is done, users can fill the controller register as the first initial sequence and then

set the warm-start bit (bit[24] of 7.2.2 MemCmd_srf) instead of setting the initial bit (bit[0] of 7.2.2 MemCmd_srf). DDR2 memory controller will skip the DDR initial sequence and assert initial ok=1 (bit[8] of 7.2.2 MemCmd_srf) soon. Then DDR2 memory controller is ready to accept any AHB/AXI command.

7.1.7 Refresh Function

The auto-refresh command prevents the data loss in the DDR2 SDRAM. The period of a refresh command can be programmed based on the speed of the system clock and the DDR2 SDRAM specification. This value is usually documented in the DDR2 SDRAM specification as a specified timing interval (T_{ref}), in which all rows (row_num) should be completely refreshed to prevent data loss. The average refresh intervals are 7.8μs (commercial) and 3.9 μs (industrial). The DDR2 memory controller allows the posted refresh commands until the threshold is achieved. Bits[15:13] of 0 MCCR are the threshold. The recommended value is 4.

7.1.8 Automatic Power-down

Users can program the automatic power-down enable (bit[12] of 7.2.10 AODCR) to save power at the system idle state. The controller can detect the empty AHB/AXI command queue, wait until the counter is timed out, and force DDR2 SDRAM to enter the active power-down mode (CKE pin is pulled low). In the power-down mode, when the auto-refresh timer is up, DDR2 memory controller will automatically pull the CKE pin high and issue an auto-refresh command. Once the auto-refresh command is issued and if there is no other command to be issued, the controller will re-enter the automatic power-down mode until the master issues a valid AHB/AXI command. The automatic power-down timer will stop counting down when DDR2 SDRAM is at the self-refresh state.

7.1.9 Channel Arbitration

For the multi-channel configurations, the channel arbiter is responsible for selecting a granted channel that issues the commands to the memory controller. The arbitration policies are described in the following subsections. There are two arbitration modes in the arbiter:

1. Two-level round-robin arbitration with channel grant counts
2. Read-Write grouping arbitration with RW group grant counts and channel grant counts

7.1.9.1 Two-level Round-Robin Arbitration with Channel Grant Counts

All channels are divided into two levels: High-priority level and low-priority level. Users can specify the high-priority channels by setting the CHARBR 7.2.10. For example, when five channels are enabled; channel2 and channel 4 will be specified as the high-priority channels; other channels (0, 1 and 3) will be specified as the low-priority channels. The low-priority group will be a member of the high-priority group.
The grant sequences will be:
CH2 →CH4 →CH0 →CH2 →CH4 →CH1 →CH2 →CH4 →CH3 →CH2 →CH4 →CH0 ……
Each channel has one 8-bit grant window count (CHGNTRA Table 7-16 and CHGNTRB Table 7-17). When a channel gains the grant and issues one command to the memory

controller, its grant window count will be decreased by one. Once the grant window count of the granted channel is decreased to zero, the arbiter will perform the re-arbitration. However, if the granted channel de-asserts its request before the grant window count becomes zero, the re-arbitration will be performed.

7.1.9.2 Read-Write Grouping Arbitration with RW Group counts and Channel Grant Counts

Besides the two-level round-robin algorithm, an advanced mechanism, “Read-Write group”, is used to reduce the SDRAM “Write-to-Read” (WtR) wait cycles. The key concept of “Read-Write group” is to merge the same command type (Read with read or write with write) to reduce the occurrence of the Write-to-Read turnaround probabilities.
The “Read-Write group” arbitration can be enabled by programming the Channel Arbitration Setup Register (bit 31 of CHARBR Table 7-16). If the Read-Write group mechanism is enabled, all channels will be classified into the Read or Write arbitration group. Please note that if the “Read-Write group” arbitration is enabled, all channels will be set as the high-priority level group re-arbitration.

Group Re-arbitration

A programmable counter, “group_grant_count” (CHARBR Table 7-16), defines the maximum commands to be issued for a read or write group. Each time a channel issues a command to the memory controller, the allowed command count will be decreased by one. When “group_grant_count” becomes zero, the group-level re-arbitration will be performed. Consequently, the arbiter will select an arbitration group as the new active group, reload the setting of “group_grant_count”, and start counting.

Channel Re-arbitration

Similar to the two-level round-robin arbitration, when the grant window count of a channel becomes zero, the channel re-arbitration will be performed. The arbiter will select a channel that belongs to the same Read group or Write group if “group_grant_count” is not decreased to zero.
For example, if channel0 and channel2 belong to the read group, channel1 and channel3 belong to the write group. group_grant_count = 16, ch0, ch1, ch2, ch3 grant_count = 3, the arbitration will be as follows:

Ch0 rd →	Ch2 rd →	Ch0 rd →	Ch2 rd →	Ch1 wr →	Ch3 wr →	Ch1 wr →	Ch3 wr
4 read	4 read	4 read	4 read	4 write	4 write	4 write	4 write

7.1.9.3 Burst Oriented Arbitration (Only for AHB Channels)

An AHB/AXI burst transaction can be divided into many DDR commands according to the burst length and burst type. For example, in a DDR2 system with the system bus being 32 bits and the memory bus being 32 bits, a WRAP16 burst transaction with word size may be divided into four DDR2 commands (BL4). For some applications, it is better not to break the DDR2 commands that belong to the same burst transaction when the arbiter performs the arbitration.

Users can set the BrtOriArb register defined at the CHARBR (Table 7-16) to enable the burst oriented arbitration function.
When the burst oriented arbitration functions enabled, the AHB INCR (Undefined length command) burst transaction will be as follows:

- For the INCR write burst, the burst oriented arbitration will have no effect on the INCR write since there is no burst boundary information for arbitration. However, users can set the grant count register (CHGNTRA Table 7-16 and CHGNTRB Table 7-17) to limit the INCR write bandwidth.
- For the INCR read burst, users can set the prefetch mechanism as the limited prefetching mode (AHBRPRER1 Table 7-32 and AHBRPRER2 Table 7-33). The AHB interface module will generate the prefetch commands for the INCR read burst transaction. The number of the prefetch commands is defined at the Register (AHBRPRER1 Table 7-32 and AHBRPRER2 Table 7-33). The arbiter will treat all prefetch commands as a whole burst and will not break these prefetch commands during arbitration.

If the prefetch mechanism is defined as the unlimited prefetching mode, the burst oriented arbitration will have no effect on the INCR read burst transaction.

7.1.10 Cross-channel Data Coherence

In the multi-channel environment, there may be data coherence issues between different channels. For example, channel1 writes a data to address “A”, and channel0 wants to read the data written by channel1 in address “A”. Because DDR2 memory controller provides the command FIFO and the opportunity for channel switch, it is possible that the read command of channel0 will exceed the write command of channel1. Thus, channel0 will read the wrong data.
For the scenario “Channel1writes and channel0 reads”, the data hazard situations can be solved by using the following three approaches with DDR2 memory controller:

1. The first approach is to terminate the write transaction with the AHB channel 1 “non-bufferable” flag (Set HProt[2] to ‘0’ for the write transaction). Once the arbiter receives the non-bufferable write command of CH1, the hready of CH1 will be pulled low until CH1 is serviced by the arbiter and the grant will be parked in CH1 until the write commands of this channel is finished. Thus, it is guaranteed that the CH0 read command will be behind the CH1 write command.
2. The second approach is to use the “command flushing mechanism” of the AHB slave interfaces. The AXI interfaces do not need to support the command-flushing function because the AXI interfaces will always reply the write response in the “non-bufferable write” manner. The command-flushing function of channel1 can be enabled after the write command is received by the DDR2 memory controller. When the corresponding bit (Table 7-21) is set, it indicates that the write command has been popped to the memory controller. In other words, channel0 can start to perform the read access, which will never exceed the write command of channel1. The data coherence

between channel0 and channel1 will be guaranteed. For the programming information about the command-flushing, please refer to 7.2.17 and 7.2.18 for more details.

7.2 Memory Map and Register Definition

R → Read
W → Write
P → Precharge
A → Active

Table 7-1 lists and describes the control registers of ANTAIOS DDR2 memory controller. The detailed information of the DDR2 memory controller registers will be described in the following subsection.

Table 7-1 Summary DDR2 Memory Controller Registers

Address Offset	Type	Size (Byte)	Description	Default Value
0x000	RW	4	MCCR	0x08030000
0x004	RW	4	MCSR	0x0000C000
0x008	RW	4	MRSVR0	0x0
0x00C	RW	4	MRSVR1	0x0
0x010	RW	4	EXRANKR	0x0
0x014	RW	4	TMPR0	0x0
0x018	RW	4	TMPR1	0x0
0x01C	RW	4	TMPR2	0x0
0x020	RW	4	PHYCR0	0x00002F41
0x024	RW	4	RHYRDTR	0x00007777
0x028	RW	4	COMPBLKCR	0x00000FBF
0x02C	RW	4	AODCR	0x0
0x030	RW	4	CHARBRA	0x10000000
0x034	RW	4	CHGNTRA	0xFFFFFFFF
0x038	RW	4	CHGNTRB	0xFFFFFFFF
0x03C	RW	4	PHYWRTMR	0x0
0x040	RW	4	FLUSHCR	0x0
0x044	RWC	4	FLUSHSR	0x0
0x048	R	4	RES	0x0
0x04C	RW	4	UPDCR	0x0
0x05C	RW	4	UDEFR	0x0
0x060	R	12	RES	0x0
0x06C	RW	4	PHYMISCR1	0x0
0x070	R	4	RES	0x0
0x074	R	4	MSDLYCR	0x00001111
0x078	R	4	RES	0x0
0x07C	RW	4	TRAFMR	0x0
0x080	R	4	CMDCNTR0	0x0
0x084	R	4	CMDCNTR1	0x0
0x088	R	4	CMDCNTR2	0x0

Address Offset	Type	Size (Byte)	Description	Default Value
0x08C	R	4	CMDCNTR3	0x0
0x090	R	4	CMDCNTR4	0x0
0x094	R	4	CMDCNTR5	0x0
0x098	R	8	RES	0x0
0x0A0	RW	4	AHBRPRER1	0x03030303
0x0A4	RW	4	AHBRPRER2	0x03030303
0x0A8	RW	4	INITWCR1	0x0001A0AB
0x0AC	R	4	RES	0x00061A80
0x0B0	RW	4	QOSCR	0x0
0x0B4	RW	4	QOSCNTRA	0x0
0x0B8	RW	4	QOSCNTRB	0x0
0x0BC	RW	4	QOSCNTRC	0x0
0x0C0	RW	4	QOSCNTRD	0x0
0x0C4	RW	4	CHARBRB	0x10000000
0x0C8	RW	4	CHARBRC	0xFFFFFFFF
0x0CC	RW	4	CHARBRB	0xFFFFFFFF
0x0D0	R	96	RES	0x0
0x130	RW	4	PHYRDTFR	0x77777777
0x134	RW	4	PHYMISCR2	0x00F84C77
0x138	RW	4	EFIFOCR	0x0
0x13C	RW	4	CMDCKTR	0x00000300
0x140	R	100	RES	-

7.2.1 Memory Controller Configuration Register

Offset: 0x00

Table 7-2 lists and describes the configuration register of the DDR2 memory controller. GDS is used to adjust the read data timing.

Table 7-2 Memory Controller Configuration Register

Bit	Name	Type	Reset	Description
31	SHARE_SLV	RW	0	When share_en is set to 1, share_slv is validated. 0 = Controller plays master role in share mode 1 = Controller plays slave role in share mode
30	SHARE_EN	RW	0	Share function means controller can share command/address PHY with another controller. 0 = Disable 1 = Enable
29..28	Reserved	-	0	-
27	DIS_APB_ERR_RESP	RW	1	This bit disables the APB error response register The APB register interface will respond to PSIVERR under one of the following conditions: • Write/Read to the non-existed registers • Write to the read-only registers

Bit	Name	Type	Reset	Description
				0: Enable the APB error response 1: Disable the APB error response The default value is '1'.
26..23	WAIT_CYCLE	RW	0	When one command is pushed into the reorder engine, it starts to count the cycles until the command is popped out to scheduler. If the cycle count of the command exceeds the wait cycle, this command will have the highest priority for popped out to scheduler. 0: Wait cycle equals to 64 cycles. 1: Wait cycle equals to 2 * 64 cycles. 2: Wait cycle equals to 3 * 64 cycles. … 15: Wait cycle equals to 16 * 64 cycles.
22..21	CONT_CMD_LIMIT	RW	0	Continuous Command Limitation in scheduler If there is a pending read command in scheduler, there will be at most n write commands that can be issued. If there is a pending write command in scheduler, there will be at most n read commands that can be issued. If Cont_Cmd_Limit = 0, n = 32 If Cont_Cmd_Limit = 1, n = 64 If Cont_Cmd_Limit = 2, n = 128 If Cont_Cmd_Limit = 3, n = 256
20	DIX_REORDER	RW	0	When the DDR command reorder engine checks the bank, the row information used to reduce the bank conflict performance will be impacted. 0 = Enable the DDR command reorder 1 = Disable the DDR command reorder
19..16	Reserved	R	0x7	-
15..13	POST-REFRESH COMMAND COUNTS THRESHOLD	RW	0	Post-refresh command count threshold The DDR controller will post a refresh command until the count of the posted refresh commands exceed the threshold. 000 = One refresh command when the TREFI counter is timed out. (7.2.3) 001 = One refresh command when the TREFI counter is timed out. (7.2.3) 010 = Two posted refresh command thresholds 011 = Three posted refresh command thresholds 100 = Four posted refresh command thresholds 101 = Five posted refresh command thresholds 110 = Six posted refresh command thresholds 111 = Seven posted refresh command thresholds
12..10	AUTO-REFRESH COMMANDS IN THE INITIAL SDRAM SEQUENCE	RW	0	Only for the DDR2 SDRAM initialization sequence 000 = Reserved 001 = Two auto-refresh commands 010 = Three auto-refresh commands 011 = Four auto-refresh commands 100 = Five auto-refresh commands 101 = Six auto-refresh commands 110 = Seven auto-refresh commands 111 = Eight auto-refresh commands
9..8	MEMORY WIDTH	RW	0	Memory width 00 = reserved **01 = 16-bit memory** 10 = reserved 11 = reserved
7..6	Reserved	R	0	-

Bit	Name	Type	Reset	Description
5..4	DDR2 MEMORY ADDRESS MAPPING TABLE SELECT (AMTSEL)	RW	0	00 = The mapping method of the AHB/AXI address to the DDR2 SDRAM memory address is RA, BA, and CA. 01 = The mapping method of the AHB/AXI address to the DDR2 SDRAM memory address is BA, RA, and CA. 10 = Reserved 11 = Reserved
3	Reserved	R	0	-
2..0	GENERATE DQS SAMPLING WINDOW (GDS)	RW	0	Before accessing the DDR memory, this field must be scanned from 0 to 7 until the read data are correctly captured. GDS depends on the PCB package trace length between the ASIC chip and DDR2 SDRAM. Longer trace length will induce larger GDS value. Note: The GDS value should be programmable at any time. If the GDS value is set in the ROM code, it should have the possibility to modify the GDS value by latching from jumper setting or other mechanism if booting from DDR SDRAM is failed.

7.2.2 Memory Controller State Control Register

Offset: 0x04

Note: The memory command can be issued once at the same time.
For example, set both bit[0] and bit[1] = 1 at the same time is not allowed. User should set the required command one by one separately.

Table 7-3 Memory Controller State Control Register

Bit	Name	Type	Reset	Description
31..25	Reserved	R	0	-
24	WARM START	RW	0	Users can set this bit to '1' to the warm-start controller without initial DDR2 SDRAM. This mode can be used when ASIC chip is recovered from the power-down state and DDR2 SDRAM is in the self-refresh mode. Please note that when ASIC chip is at the power down state, CKE of DDR2 SDRAM should be kept low from the power-on domain of ASIC chip. When the command is completed, this bit will be cleared to zero. 1 = Warm start 0 = No effect
23..16	Reserved	R	0	-
15	AHB/AXI COMMAND QUEUE EMPTY	R	1	1 = All AHB/AXI command queue is empty 0 = All AHB/AXI command queue is not empty
14	MEMORY CONTROLLER COMMAND QUEUE EMPTY	R	1	1 = Memory controller command queue is empty. 0 = Memory controller command queue is not empty.
13..11	Reserved	R	0	-
10	SELF-REFRESH STATE	R	0	1 = DDR2 SDRAM is in the self-refresh mode. 0 = DDR2 SDRAM is at other states.

Bit	Name	Type	Reset	Description
9	INITIAL STATE	R	0	1 = DDR2 SDRAM is at the initial state. 0 = DDR2 SDRAM is not at the initial state.
8	INITIAL OK	R	0	1 = DDR2 SDRAM initial is completed. 0 = DDR2 SDRAM initial is not completed.
7..6	Reserved	R	0	-
5..4	REGISTER MODE	RW	0	00 = Mode register 01 = Extended mode register 1 10 = Extended mode register 2 11 = Extended mode register 3
3	EXIT SELF-REFRESH COMMAND	RW	0	This bit is programmed to move the controller to enter the targeted state. When the command is completed, this bit will be cleared to zero. 1 = Move the memory controller from the self-refresh state 0 = No effect
2	MEMCMD_SRF	RW	0	This bit is programmed to move the controller to enter the targeted state. When the command is completed, this bit will be cleared to zero. 1 = Move the memory controller to the self-refresh state 0 = No effect
1	MRS COMMAND	RW	0	This bit is programmed to move the controller to enter the targeted state. When the command is completed, this bit will be cleared to zero. Users should program MR/EMR/EMRS2/EMRS3 in the Register (7.2.3, 7.2.4) bits[5:4] register mode. Before starting the MRS command: 1 = Move the memory controller to mode register state 0 = No effect
0	INITIAL COMMAND	RW	0	This bit is programmed to move the controller to enter the targeted state. When the command is completed, this bit will be cleared to zero. Setting this bit to '1' will start to perform the DDR2 SDRAM initialization and DDR PHY compensation block calibration. After these tasks are completed, the signal bit[8] of the initial_ok will be set to '1'. 1 = Move the memory controller to the DDR2 SDRAM initial state 0 = No effect

7.2.3 Mode Register Set Value Register of MR and EMR

Offset: 0x08

This register defines the values of the mode registers and the extended mode registers to be used in the initial sequence. During the initialization of DDR2 SDRAM, DDR2 memory controller uses the values in this register for the MRS command. DDR2 memory controller only supports AL (Additive Latency) = '0' in the DDR2 mode.
If users want to modify the values of MR and EMR after initialization, this Register should be programmed, and then bits [5:0] of MCSR (7.2.2) should be set to:

- 000010 = Send MRS command to modify the MR value of DDR2 SDRAM
- 010010 = Send MRS command to modify the EMR value of DDR2 SDRAM

Notes:
1. The mode register should be set as sequential burst length 4 in the DDR2

2. The mode register content write recovery should be set to the same value of the Timing Parameter1, Register TMPR1 (7.2.3), bits[23:20], t_{WR}.

3. AL should be set to '0'for the DDR2 applications to improve the Write-to-Read or Read-to-Write performance.

4. Bit26 of the Register MRSVR0 (7.2.3) DQS_B and bit3 of the Register PHYCR0 (7.2.9) should be set to the same value. For example, the DDR2 SDRAM and DDR PHY should operate in the same DQS mode, either in the differential mode or single-ended mode.

5. The "DLL_RESET" bit of mode register will be automatically controlled by DDR2 memory controller in the DDR initialization sequence.

Table 7-4 Mode Register Set Value Register of MR and EMR

Bit	Name	Type	Reset	Description
31..30	Reserved	R	0	-
29..16	EXTENDED MODE REGISTER	RW	0	Extended mode register
15..14	Reserved	R	0	-
13..0	MODE REGISTER	RW	0	Mode register

7.2.4 Mode Register Set Value Register of EMR2 and EMR3

Offset: 0x0C

This register defines the mode register value and the extended mode register value to be used in the MRS command after initiating the DDR2 SDRAM.
If users want to modify the values of EMR2 and EMR3 after initialization, Register MRSVR1 (7.2.4) should be programmed first, and then bits[5:0] of MCSR (7.2.2) should be set to:

- 100010 = Send MRS command to modify the EMR2 value of DDR2 SDRAM.
- 110010 = Send MRS command to modify the EMR3 value of DDR2 SDRAM.

Table 7-5 Mode Register Set Value Register of EMR 2 and EMR 3

Bit	Name	Type	Reset	Description
31..30	Reserved	R	0	-
29..16	EXTENDED MODE REGISTER 3	RW	0	Extended mode register
15..14	Reserved	R	0	-
13..0	EXTENDED MODE REGISTER2	RW	0	Mode register

7.2.5 External Rank0/Rank1 Register

Offset: 0x10

Table 7-6 External Rank0/Rank1 Register

Bit	Name	Type	Reset	Description
31..24	RNK0_BASE	RW	0	This field is the 8-bit base address of the external rank0. RNK0 (Chip select 0) address range: RNK0_BASE & 0x00_0000 to RNK0_BASE & 0x00_0000 + RNK0_SIZE-1

Bit	Name	Type	Reset	Description
23..7	Reserved	R	0	-
6..4	RNK0_TYPE	RW	0	Select the DDR2 type based on the MA table for rank0 RAxCAxBA, (Row Address)x(Column Address)x(Bank Address), is denoted as 13x10x2 to indicate that the row address is 13 bits, the column address is 10 bits, and the bank address is 2 bits. 000 = 13x9x2 001 = 13x10x2 010 = 14x10x2 Others = Reserved
3	Reserved	R	0	-
2..0	RNK0_SIZE	RW	0	Memory size of rank0 000: Reserved 001: 32M bytes 010: 64M bytes 011: 128M bytes 100: 256M bytes 101: 512M bytes 110: 1G bytes 111: 2G bytes

Fehler! Verweisquelle konnte nicht gefunden werden. lists the available combinations of R NK_TYPE and RNK_SIZE for different DDR2 memory configurations.

Table 7-7 DDR2 Memory Width = 16 bits

RNK_SIZE	RNK_TYPE	DDR CHIP	DDR CHIP NUMBER
000 (16Mbytes)	-	-	-
001 (32Mbytes)	000 (13x9x2)	16Mb x16 (256 Mb)	1
010 (64Mbytes)	001 (13x10x2)	32Mb x16 (512 Mb)	1
		32Mb x8 (256Mb)	2
011 (128 Mbytes)	010 (14x10x2)	64Mb x8 (512Mb)	2
100 (256 Mbytes)	101 (14x10x3) (use only half of 256Mbytes)	128 Mb x16 (2 Gb)	1
		128 Mb x8 (1 GB)	2
101 (512 Mbytes)	-	-	-
110 (1 Gbytes)	-	-	-
111 (2 Gbytes)	-	-	-

7.2.6 Timing Parameter0 Register

Offset: 0x14

Table 7-8 External Rank0/Rank1 Register

Bit	Name	Type	Reset	Description
31..24	TRFC	RW	0	Refresh-to-Active/Refresh command period 0x0 = Not used 0x1 = 2 200MHz cycles

Bit	Name	Type	Reset	Description
				0x2 = 3 200MHz cycles ... 0xff = 256 200MHz cycles
23..21	Reserved	R	0	-
20..16	TFAW	RW	0	Four bank activate window 0x00 = Not used 0x01 = 2 200MHz cycles 0x02 = 3 200MHz cycles ... 0x1F = 32 200MHz cycles Notes: The 5th Active command next to four consecutive active commands will be constrained by the greater value of TRC, TRAS, and TFAW. The minimum Active-to-Active command period for different banks will also be constrained by the priority of command scheduler selecting Write/Read/Active commands. The Write/Read commands have higher priority than the Active command.
15..14	Reserved	R	0	-
13..8	TRC	RW	0	Active-to-Active command period for the same bank 0x00 = Not used 0x01 = 2 200MHz cycles 0x02 = 3 200MHz cycles ... 0x3F = 64 200MHz cycles
7..5	Reserved	R	0	-
4..0	TRAS	RW	0	Active-to-Precharge period This parameter specifies the minimum period of an opened row. 0x0 = Not used 0x1 = 2 200MHz cycles 0x2 = 3 200MHz cycles … 0x1F = 32 200MHz cycles

7.2.7 Timing Parameter1 Register

Offset: 0x18

For DDR2, t_{WL}= AL (Additive Latency) + CL (CAS Latency) -1
DDR2 memory controller only supports AL = ‘0’

Table 7-9 Timing Parameter 1 Register

Bit	Name	Type	Reset	Description
31	Reserved	R	0	-
30..28	TWTR	RW	0	Internal write-to-read delay READ command after the last write data delay cycle 000 = Not used 001 = 2 200MHz cycles 010 = 3 200MHz cycles 011 = 4 200MHz cycles

Bit	Name	Type	Reset	Description
				100 = 5 200MHz cycles 101 = 6 200MHz cycles 110 = 7 200MHz cycles 111 = 8 200MHz cycles
27	Reserved	R	0	-
26..24	TRTP	RW	0	Internal read-to-precharge delay 000 = Not used 001 = 2 200MHz cycles 010 = 3 200MHz cycles ... 111 = 8 200MHz cycles
23..20	TWR	RW	0	Write recovery time The last non-write data cycle at the MCLK rising edge to the precharge command cycle. 0x0 = Not used 0x1 =2 200MHz cycles 0x2 = 3 200MHz cycles … 0xF = 16 200MHz cycles
19..16	TMOD	RW	0	The mode register sets the command update delay. 0x0 = Not used 0x1 = 2 200MHz cycles 0x2 = 3 200MHz cycles ... 0xF = 16 200MHz cycles
15..12	TMRD	RW	0	Cycle time of the load mode register command 0x0 = Not used 0x1 = 2 200MHz cycles 0x2 = 3 200MHz cycles 0x3 = 4 200MHz cycles ... 0xF = 16 200MHz cycles
11..8	TRP	RW	0	Precharge period 0x0 = Not used 0x1 = 2 200MHz cycles 0x2 = 3 200MHz cycles ... 0xF = 16 200MHz cycles
7..4	TRRD	RW	0	Active-to-active command period for different banks 0x0 = Not used 0x1 = 2 200MHz cycles 0x2 = 3 200MHz cycles ... 0xF = 16 200MHz cycles Note: The minimum Active-to-Active command period for different banks will be constrained by the priority of Write/Read/Active commands selected by the command scheduler. The Write/Read commands have higher priority than the Active command.
3..0	TRCD	RW	0	The minimum delay between the active and read/write commands 0x0 = Not used 0x1 = 2 200MHz cycles 0x2 = 3 200MHz cycles ... 0xF = 16 200MHz cycles

7.2.8 Timing Parameter 2 Register

Offset: 0x1C

t_{WL}= Write latency in terms of the DRAM clock cycles
t_{RL}= Read latency in terms of the DRAM clock cycles

t_{WL} = AL (Additive Latency) + CL (CAS Latency)-1
t_{RL} = AL (Additive Latency) + CL (CAS Latency)

Users can refer to 7.2.30 for the interval counting examples of the Write-to-Read, Write-to-Write, Read-to-Read, and Read-to-Write commands.

Table 7-10 Timing Parameter 1 Register

Bit	Name	Type	Reset	Description
31..30	TW_TO_R_CTRL	RW	0	Controller specific **Additional delay cycles** from the write command to the read command for accessing **different ranks** The programmed value compensates the unbalanced trace or I/O delay between different ranks of DDR2 SDRAM to avoid the DQS driving contention. 00 = No additional delay clock cycles 01 = Add additional 1 200MHz cycles 10 = Add additional 2 200MHz cycles 11 = Add additional 3 200MHz cycles
29..28	TW_TO_W_CTRL	RW	0	Controller specific **Additional delay cycles** from the write command to the write command. 00 = No additional delay clock cycles 01 = Add additional 1 200MHz cycles 10 = Add additional 2 200MHz cycles 11 = Add additional 3 200MHz cycles
27..26	TR_TO_R_CTRL	RW	0	Controller specific **Additional delay cycles** from the read command to the read command. 00 = No additional delay clock cycles 01 = Add additional 1 200MHz cycles 10 = Add additional 2 200MHz cycles 11 = Add additional 3 200MHz cycles
25..24	TR_TO_W_CTRL	RW	0	Controller specific **Additional delay cycles** from the read command to the write command The DQS will be driven by different devices in the read and write operations. The programmed value will compensate the round-trip delay between the chip and DDR2 SDRAM to avoid the DQS driving contention. Users should adjust this value to avoid the Read DQS/DQ and the Write DQS/DQ driving contention. 00 = No additional delay clock cycles 01 = Add additional 1 400MHz cycles 10 = Add additional 2 400MHz cycles 11 = Add additional 3 400MHz cycles
23..16	Reserved	R	0	-
15..8	TXSR	RW	0	Exit from the self-refresh mode to a command period This value should be programmed to the maximum of t_{XSNR} and

Bit	Name	Type	Reset	Description
				t_{XSRD} in the DDR2 mode. 0x0 = Not used 0x1 = 1 X 8 200MHz cycles 0x2 = 2 X 8 200MHz cycles ... 0xff = 255 X 8 200MHz cycles Notes: (1) There will be three additional MClk guard time to compensate the internal exiting self-refresh command to the DDR PHY I/O delay (2) The first active command after exiting the self-refresh state should conform to all the following timing constraints: TXSR
7..0	TREFI	RW	0	Average periodic refresh interval One refresh command should be issued if the refresh counter equals to the refresh interval. Please note that bit 7 is only for selecting between x32T and x8T. 0_000_0000 = Not used 0_000_0001 = 1x32 200MHz cycles 0_000_0010 = 2x32 200MHz cycles ... 0_111_1111 = 127x32 200MHz cycles 1_000_0000 = Not used 1_000_0001 = 1x8 200MHz cycles 1_000_0010 = 2x8 200MHz cycles ... 1_111_1111 = 127x8 200MHz cycles

7.2.9 DDR2 PHY Command and Data Block Control Register

Offset: 0x20

Bit 26 DQS_B (EMRS) of Register MRSVR0 (7.2.3) and bit 3 of current Register PHYCR0 should be set to the same value in the DDR2 mode. When both bit15, “auto_io_deep_pdn” and bit7, “auto_io_ctrl_pdn” are set to ‘1’ at the same time, the I/O enable will follow the setting of bit15,“auto_io_deep_pdn”.
In order to prevent the I/O enable switch too frequently, if “auto_io_deep_pdn” or “auto_ioctrl_pdn” is set to ‘1’
(1) The dqie, cmdaddroen will be disabled when power down period is larger than 10 200MHz cycles.
(2) The clkoen will be disabled when self-refresh period is larger than 15 MClk cycles.

Note: fldo_byte0, fldo_byte2, fldo_byte4, and fldo_byte6 are used in the 16-bit DDR2 PHY.

Table 7-11 DDR2 PHY Command and Data Block Control Register

Bit	Name	Type	Reset	Description
31..28	Reserved	R	0	-
27	FLDO_BYTE3	R	0	Byte3 DLL lock flag indicates that DLL is at the lock state. 1 = DLL is at the lock state. 0 = DLL is not at the lock state.

Bit	Name	Type	Reset	Description
26	FLDO_BYTE2	R	0	Byte2 DLL lock flag indicates that DLL is at the lock state. 1 = DLL is at the lock state. 0 = DLL is not at the lock state.
25	FLDO_BYTE1	R	0	Byte1 DLL lock flag indicates that DLL is at the lock state. 1 = DLL is at the lock state. 0 = DLL is not at the lock state.
24	FLDO_BYTE0	R	0	Byte0 DLL lock flag indicates that DLL is at the lock state. 1 = DLL is at the lock state. 0 = DLL is not at the lock state.
23..18	Reserved	R	0	-
17	PHY_ODD	RW	0	tRL_odd and tWL_odd signals usage for DDR2 PHY 1 = DDR PHY Data block is controlled by rdcmd_byten_in0, tRL_odd, dqsoe_byten_in0 and tWL_odd 0 = Data clock controlled by rdcmd_byten_in0, rdcmd_byten_in1, dqsoe_byten_in0 and dqsoe_byten_in1. tRL_odd and tWL_odd are not used by DDR2 PHY
16	Reserved	R	0	-
15	AUTO_IO_DEEP_PDN	RW	0	Automatic control of the I/O input buffer at the power-down state 1 = Enable 0 = Disable When auto_io_deep_pdn is enabled, the cmdaddroen and dqie will be disabled at the automatic power-down state; and clkoen, cmdaddroen, dqie, and odtoen will be disabled at the self-refresh state.
14	SELF_BIAS	RW	0	VREF selection 1 = DDR PHY uses the internal generated VREF. 0 = DDR PHY uses the external VREF.
13	PHY_CON_UPDATE	RW	1	DDR PHY digital DLL update control code 1 = Slave DLL continuously updates the control code regardless of the “update_en” signal. 0 = Slave DLL updates the control code when the “update_en” signal is asserted.
12	CLK_PHASESHIFT	RW	0	DDR PHY input clock for the DDR SDRAM phase 0 = Clock position is in the 1/2 clock cycle phase of the DDR command. 1 = Clock position is in the 3/4 clock cycle phase of the DDR command.
11	DQIE	RW	1	DQ and DQS receivers enable 1 = Enable 0 = Disable Note: In the automatic power-down mode, the dqie bit will be controlled by hardware if auto_ioctrl_pdn or auto_io_deep_pdn is set to ‘1’. PHY should be reset before normal operation if dqie is set to ‘0’.
10	CLKOEN	RW	1	The input controls of the DDR PHY CK and CKB 1 = Enable 0 = Disable Note: In the self-refresh mode, the clkoen bit will be controlled by hardware if auto_ioctrl_pdn or auto_io_deep_pdn is set to ‘1’.
9	CMDADDROEN	RW	1	The input controls of the DDR PHY ADDR, BA, RAS, CAS, WE and CS drivers 1 = Enable 0 = Disable

Bit	Name	Type	Reset	Description
				Note: In the automatic power-down mode, the cmdaddroen bit will be controlled by hardware if auto_ioctrl_pdn or auto_io_deep_pdn is set to '1'.
8	ODTOEN	RW	1	Enable input drivers of ODT 1 = Enable 0 = Disable Note: In the self_refresh mode, the odtoen bit will be controlled by hardware if auto_io_deep_pdn is set to '1'.
7	AUTO_IO_CTRL_PDN	RW	0	Automatic control of the I/O Input buffer at power-down 1 = Enable 0 = Disable When auto_io_ctrl_pdn is enabled, cmdaddroen and dqie will be disabled at the automatic power-down state, and clkoen will be disabled at the self-refresh state.
6	Reserved	R	1	-
5	DQSLOWFEN	RW	0	Enable the DQS bypass DLL This bit is used for the low-frequency operation to control the DDR PHY bypass DLL.
4	DMYODTEN	RW	0	On-Die Termination (ODT) enable for the DDR PHY DUMMY pad of the data block
3	SIO	RW	0	This bit should be compatible with the DQS_B enable bit in the extended mode register. 0 = DQS is in the differential mode. 1 = DQS is in the single-ended mode.
2..0	ODTMD	RW	1	Set the ODT value, only odtmd[1:0] is used for the DDR2 PHY. 010: ODT = 150 Ω 001: ODT = 75 Ω 011: ODT = 50 Ω 000: ODT is disabled.

Figure 7-2 illustrates the DDR PHY I/O control timing based on the 200MHz MClk when entering the self-refresh mode if "auto_io_deep_pdn" is set to '1'.

Figure 7-2 DDR PHY I/O Input Enable Timing When Entering Self-refresh Mode

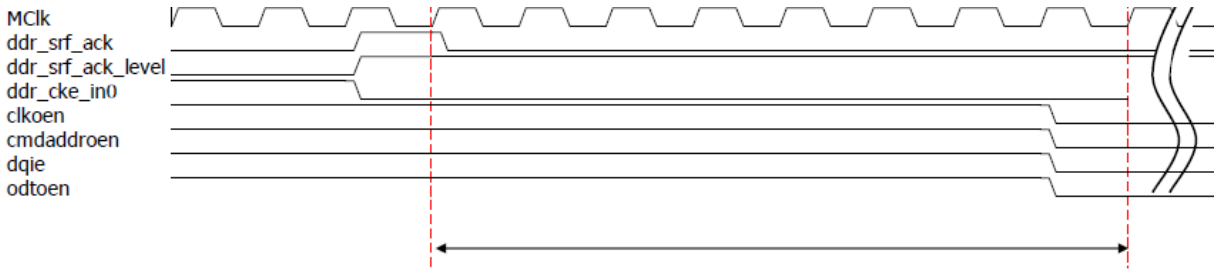
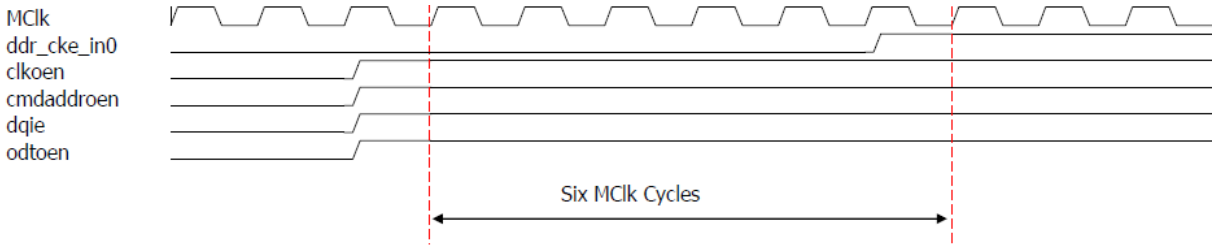


Figure 7-3 illustrates the DDR PHY I/O control timing based on MClk when exiting the self-refresh mode if "auto_io_deep_pdn"is set to '1'.

Figure 7-3 DDR PHY I/O Input Enable Timing When Exiting Self-refresh Mode



7.2.10 DDR2 PHY Read Path DLL Delay Tuning Register

Offset: 0x24

Dllsel is used to adjust the read DQ and read DQS. The read DQS should be at the center of the DQ data eye for more timing margin.
Please refer to Register PHYMISCR1 (7.2.21) for more details.

Table 7-12 DDR2 PHY Command and Data Block Control Register

Bit	Name	Type	Reset	Description
31..4	Reserved	R	0x0000777	-
3..0	DLLSEL_BYTE0	RW	0x7	Delay value control to add the delay into the read DQS or read DQ Please refer to the Register PHYMISCR1 (7.2.21) for details.

7.2.11 DDR2 PHY Command and Data Block Control Register

Offset: 0x28

Table 7-13 COMPBLK Control Register

Bit	Name	Type	Reset	Description
31..25	Reserved	R	0	-
24..19	DON	R	0	Value of the COMPBLK pull-down NMOS impedance setting from the input “DON” of compensation block of DDR2 PHY
18..13	DOP	R	0	Value of the COMPBLK pull-up PMOS impedance setting from the input “DOP” of compensation block of DDR2 PHY
12..7	DIN	RW	0x1F	Preset value for the COMPBLK pull-down NMOS impedance
6..1	DIP	RW	0x1F	Preset value for the COMPBLK pull-up PMOS impedance
0	COMP_SEL	RW	1	Compensation method selection 1 = Enable the COMPBLK auto-calibration before initiating DDR2 SDRAM 0 = Use the preset values of DIP and DIN

7.2.12 Automatic Power-down/Self-refresh Control Register

Offset: 0x2C

Table 7-14 COMPBLK Control Register

Bit	Name	Type	Reset	Description
31..29	Reserved	R	0	-
28	AUTO_SRF_EN	RW	0	Automatic self-refresh mode control 1 = Enable the automatic self-refresh 0 = Disable the automatic self-refresh
27..16	AUTO_SRF_TIMER	RW	0	Automatic self-refresh timer When the command queue is empty, the automatic self-refresh timer will start counting down. When the timer reaches zero, self-refresh request will be triggered and DDR2 SDRAM will enter the self-refresh mode. 0x000 = Not used 0x001 = 1 x 4 = 4 200MHz cycles 0x002 = 2 x 4 = 8 200MHz cycles … 0xfff = 4095 x 4 = 16380 200MHz cycles
15..13	Reserved	R	0	-
12	AUTO_PDN_EN	RW	0	Automatic power-down mode control 1 = Enable the automatic power-down mode 0 = Disable the automatic power-down mode
11..0	AUTO_PDN_TIMER	RW	0	Automatic power-down timer When the command queue is empty, the automatic power-down timer will start counting down. When the timer reaches zero, CKE will be automatically pulled low. The DDRx SDRAM will enter either active power down or precharge power down mode. 0x000 = Not used 0x001 = 1 x 4 = 4 200MHz cycles 0x002 = 2 x 4 = 8 200MHz cycles … 0xfff = 4095 x 4 = 16380 200MHz cycles

7.2.13 Channel Arbitration Setup RegisterA

Offset: 0x30

Table 7-16 lists and describes the parameter setup for the channel arbitration. There are two arbitration modes:

1. Two-level round-robin with the channel grant count
2. Read-Write group with grant_count_group and the channel grant count

When the Read-Write group arbitration is enabled, whether the priority, burst-orient, and grant count value for the read and write operations can be identical or not will be decided by the independent read-write setting enable bit, “Indepent_RW”.

Table 7-15 Setting for Read/Write Operation

Read-Write Group	Indepent_RW Enable	Priority, Burst-orient	Grant Count for Group	Grant Count for Channel
Enable	Enable	Register CHARBRA (7.2.13) defines the read setting. Register CHARBRB	Register CHARBRA (7.2.13) defines the read group grant count and the read grant count	Registers CHGNTRA (7.2.14) and PHYWRTMR (7.2.16) define the read grant

Read-Write Group	Indepent_RW Enable	Priority, Burst-orient	Grant Count for Group	Grant Count for Channel
		(7.2.37) defines the write setting	constraints the read operations in group. Register CHARBRB (7.2.37) defines the write group grant count and the write group grant count constraints the write operations in group.	count for each channel and the read grant count constraints the read operations. Registers CHGNTRC (7.2.38) and CHGNTRD (7.2.39) define the write grant count for each channel and the write grant count constraints the write operations.
	Disable	Register CHARBRA (7.2.13) defines both the read and write settings.	Register CHARBRA (7.2.13) defines the read group grant count and the read grant count constraints the read operations in group. Register CHARBRA (7.2.13) defines the write group grant count and the write grant count constraints the write operations in group.	Registers CHGNTRA (7.2.14) and PHYWRTMR (7.2.16) define the read grant count for each channel and the read grant count constraints the read operations. Registers CHGNTRA (7.2.14) and PHYWRTMR (7.2.16) define the write grant count for each channel and the write grant count constraints the write operations.
Disable	Enable or Disable	Register CHARBRA (7.2.13) defines both the read and write settings.	Register CHARBRA (7.2.13) defines both the read and write group grant counts.	Registers CHGNTRA (7.2.14) and PHYWRTMR (7.2.16) define the grant count for each channel and the grant count constraints both the read and write operations.

If Register CHARBRA (9.2.13) bit[30] “Indep_RW_EN”=‘1’, Register Offset CHARBRA (9.2.13) [28:0] will be used for the read commands. If Register Offset CHARBRA (9.2.13) bit[30] “Indep_RW_EN”=‘0’, Register Offset CHARBRA (9.2.13) [28:0] will be used for the read and write commands.

Table 7-16 Channel Arbitration Setup Register

Bit	Name	Type	Reset	Description
31	RW_GRP_EN	RW	0	Enable the Read-Write group arbitration Set this bit to ‘1’ to enable the read-write-group arbitration. Otherwise, the normal two-level round-robin arbitration will be used. 1 = Read-Write group arbitration is enabled. 0 = Read-Write group arbitration is disabled.
30	INDEP_RW_EN	RW	0	Independent RW control 1 = Independent RW is enabled. 0 = Independent RW is disabled.
29	Reserved	R	0	-

Bit	Name	Type	Reset	Description
28..24	GROUP_GRANT_ COUNT_LOW	RW	0x10	Total group_grant_count[7:0] = {group_grant_count_high, group_grant_count_low } CHARBRB (7.2.37) [18:16] indicates “group_grant_count_high” CHARBRA (7.2.13) [28:24] indicates “group_grant_count_low” For the Read-Write group arbitration, this field sets the initial value of the group grant window count. This field will not be used if the Read-Write group arbitration is disabled. 0x00 = Not used 0x01 = Issue a maximum of 1 command when each group is granted. 0x02 = Issue a maximum of 2 commands when each group is granted. … 0x1f = Issue a maximum of 31 commands when each group is granted.
23..14	Reserved	R	0	-
15..8	BST_ORI_ARB	RW	0	Burst oriented arbitration enable bit (Only for AHB slave ports) When this bit is set, the burst transaction will not be broken during arbitration. Bit 13 = 1 indicates that the channel 5 burst oriented arbitration is enabled. Bit 13 = 0 indicates that the channel 5 burst oriented arbitration is disabled. … Bit 8 = 1 indicates that the channel 0 burst oriented arbitration is enabled. Bit 8 = 0 indicates that the channel 0 burst oriented arbitration is disabled.
7..6	Reserved	R	0	-
5..0	CH_HI_PRIOR	RW	0	All channels are divided into two priority levels: The high-priority level and low-priority level. These bits indicate which channels belong to the high-priority level. Setting each bit to ‘1’ indicates that the corresponding channels belong to the high-priority level. Bit 5 = 1 indicates that channel 5 belongs to the high-priority group. Bit 5 = 0 indicates that channel 5 belongs to the low-priority group. .. Bit 0 = 1 indicates that channel 0 belongs to the high-priority group. Bit 0 = 0 indicates that channel 0 belongs to the low-priority group.

7.2.14 Channel Arbiter Grant Count RegisterA

Offset: 0x34

Each channel has its own grant window counter. When the arbiter perform the re-arbitration, the value set in this register will be loaded into the grant window counters of channel0, channel1, channel2 and channel3. Once the granted channel pops a command to the memory controller, the grant window counter of this channel will be decreased by one. If the burst oriented function is disabled and the grant window counter of the current granted channel becomes ‘0’, the arbiter will switch the grant to another channel. If the burst oriented

function is enabled, the arbiter will switch grant when the grant window counter becomes ‘0’ and the popped command will be the EOB (End of burst) command of a burst transaction, which is the burst transaction that will not be broken by the arbiter even when the grant window counter becomes ‘0’. The grant window counter of each channel defines the maximum service command counts allowed for each channel when that channel is granted by the arbiter. Please note that a command indicates a DDR2 WRAP4 RW command.
If Register CHARBRA (9.2.13) bit[31] “RW_Group_En”= ‘1’ and bit[30] “Indep_RW_EN”=‘1’, Register CHGNTRA (7.2.14) will be used for the read commands or Register CHGNTRA (9.2.14) will be used for the read and write commands.

Table 7-17 Channel Arbitration Grant Count RegisterA

Bit	Name	Type	Reset	Description
31..24	ARB_CNT3	RW	0xFF	Once channel3 is granted, the maximum allowed commands will be issued. 0 = One command 1 = Two commands … 255 = 256 commands
23..16	ARB_CNT2	RW	0xFF	Once channel2 is granted, the maximum allowed commands will be issued. 0 = One command 1 = Two commands … 255 = 256 commands
15..8	ARB_CNT1	RW	0xFF	Once channel1 is granted, the maximum allowed commands will be issued. 0 = One command 1 = Two commands … 255 = 256 commands
7..0	ARB_CNT0	RW	0xFF	Once channel0 is granted, the maximum allowed commands will be issued. 0 = One command 1 = Two commands … 255 = 256 commands

7.2.15 Channel Arbiter Grant Count RegisterB

Offset: 0x38

Similar to the Register CHGNTRA (7.2.14), this register defines the maximum service command counts of channel4 and channel5.
If Register CHARBRA (9.2.13) bit[31] “RW_Group_En”= ‘1’ and bit[30] “Indep_RW_EN”=‘1’, Register CHGNTRB (7.2.15) will be used for the read commands or Register CHGNTRB (7.2.15) will be used for the read and write commands.

Table 7-18 Channel Arbitration Grant Count RegisterB

Bit	Name	Type	Reset	Description
31..16	Reserved	R	0xFFFF	-

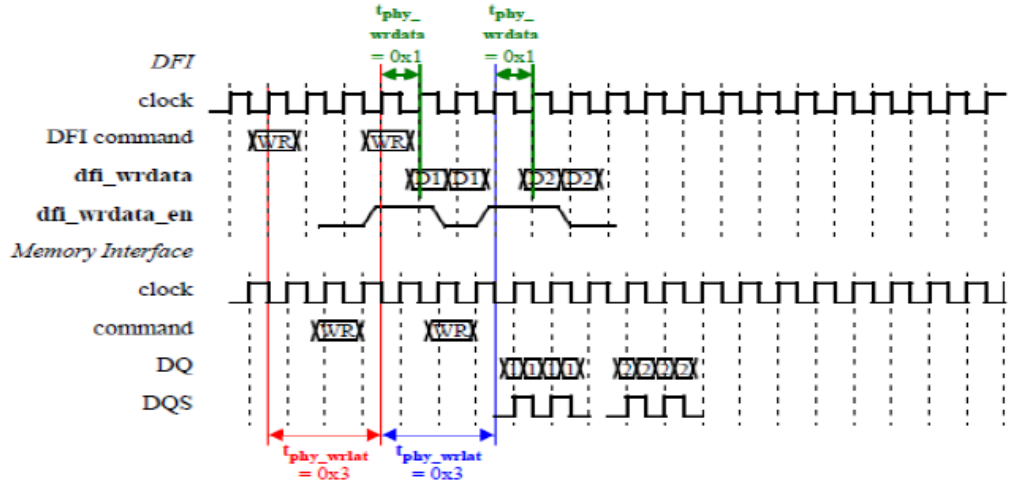
Bit	Name	Type	Reset	Description
15..8	ARB_CNT5	RW	0xFF	Once channel 5 is granted, the maximum allowed commands will be issued. 0 = One command 1 = Two commands … 255 = 256 commands
7..0	ARB_CNT4	RW	0xFF	Once channel 4 is granted, the maximum allowed commands will be issued. 0 = One command 1 = Two commands … 255 = 256 commands

7.2.16 DDR2 PHY Write/Read Data Timing Control Register

Offset: 0x3C

Tphy_wrlat specifies the number of 200MHz clock cycles between a write command sent on the PHY control interface and the asserted dfi_wrdata_en (dqsoe) signal. **Tphy_wrdata** specifies the number of 200MHz clock cycles between the asserted dfi_wrdata_en (dqsoe) signal and the associated write data driven on the dfi_wrdata (ddr_wdata) signal.

Figure 7-4 tphy_wrlat and tphy_wrdata Timing in DFI Specification 2.1



The **dfi_rddata_en (rdcmd)** signal must be asserted for **trddata_en** cycles after the assertion of a read command on the PHY control interface and remains valid for the duration of the contiguous read data expected on the dfi_rddata (ddr_rdata) bus.
The read data are expected to be received at DDR2 memory controller within **tphy_rdlat** cycles after the **dfi_rddata_en** signal is asserted.

Figure 7-5 trddata_en and tphy_rdlat in DFI Specification 2.1

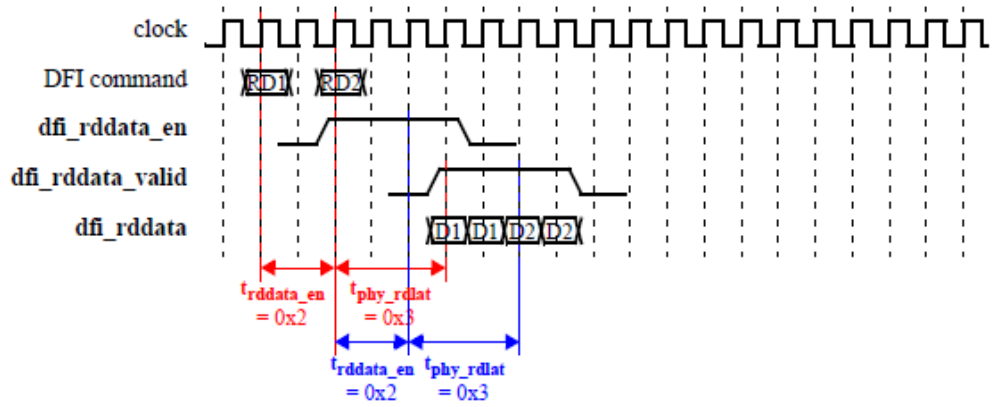


Table 7-19 DDR2 PHY Write/Read Data Timing Control Register

Bit	Name	Type	Reset	Description
31..24	Reserved	R	0	-
23..20	TPHY_RDLAT	RW	0	The read data are expected to be received at DDR2 memory controller within **tphy_rdlat** cycles after the **rdcmd** signal is asserted. The valid values are between 2 and 11. 2 = ddr_rdata is expected to receive two 200MHz cycles after rdcmd is asserted. ... 11 = ddr_rdata is expected to receive 11 200MHz cycles after rdcmd is asserted.
19..16	TRDATA_EN	RW	0	dfi_rddata_en (rdcmd) signal must be asserted for the **trddata_en** cycles after the assertion of a read command on the PHY control interface and remains valid for the duration of contiguous read data expected on the dfi_rddata (ddr_rdata) bus. The valid values are between 0 and **10**. 0 = rdcmd asserts at the same cycle as the read command asserts. 1 = rdcmd asserts one 200MHz cycle after the read command asserts. … 10 = rdcmd asserts 10 200MHz cycles after the read command asserts. Others = Reserved
15..6	Reserved	R	0	-
5..4	TPHY_WRDATA	RW	0	**tphy_wrdata** specifies the number of 200MHz clock cycles between when the dfi_wrdata_en (dqsoe) signal is asserted and when the associated write data are driven on the ddr_wdata signal. 0 = ddr_wdata will be asserted at the same time when dqsoe is asserted. 1 = ddr_wdata will assert one 200MHz cycle after dqsoe is asserted. 2 = ddr_wdata will assert two 200MHz cycles after dqsoe is asserted. 3 = ddr_wdata will assert three 200MHz cycles after dqsoe is asserted.
3..0	TPHY_WRLAT	RW	0	**tphy_wrlat** specifies the number of 200MHz clock cycles between when a write command is sent on the DFI control interface and when the dfi_wrdata_en (dqsoe) signal is asserted. The valid values are between 0 and 9. 0 = dqsoe is asserted at the same time with the write command

Bit	Name	Type	Reset	Description
				is asserted. 1 = dqsoe will assert one 200MHz cycle after the write command is asserted. … 9 = dqsoe will assert 10 200MHz cycles after the write command is asserted. Others = Reserved

t_{WL}= Write latency in terms of the DRAM clock cycles
t_{RL}= Read latency in terms of the DRAM clock cycles

DDR2 memory controller only supports AL = '0'
tWL= AL (Additive Latency) + CL (CAS Latency) -1
tRL= AL (Additive Latency) + CL (CAS Latency)

tphy_wrlat = tWL-1
tphy_wrdata = 1
trddata_en= tRL-2
tphy_rdlat= GDS + 6
For example,
GDS = 0, tWL= 4, tRL= 5, tphy_wrlat = 3, tphy_wrdata = 1, trddata_en = 3, tphy_rdlat = 0x6
GDS = 1, tWL= 4, tRL= 5, tphy_wrlat = 3, tphy_wrdata = 1, trddata_en = 3, tphy_rdlat = 0x7
GDS = 2, tWL= 4, tRL= 5, tphy_wrlat = 3, tphy_wrdata = 1, trddata_en = 3, tphy_rdlat = 0x8
GDS = 3, tWL= 4, tRL= 5, tphy_wrlat = 3, tphy_wrdata = 1, trddata_en = 3, tphy_rdlat = 0x9
GDS = 4, tWL= 4, tRL= 5, tphy_wrlat = 3, tphy_wrdata = 1, trddata_en = 3, tphy_rdlat = 0xA
GDS = 5, tWL= 4, tRL= 5, tphy_wrlat = 3, tphy_wrdata = 1, trddata_en = 3, tphy_rdlat = 0xB
GDS = 6, tWL= 4, tRL= 5, tphy_wrlat = 3, tphy_wrdata = 1, trddata_en = 3, tphy_rdlat = 0xC
GDS = 7, tWL= 4, tRL= 5, tphy_wrlat = 3, tphy_wrdata = 1, trddata_en = 3, tphy_rdlat = 0xD

7.2.17 Command Flush Control Register

Offset: 0x40

Table 7-20 Command Flush Control Register

Bit	Name	Type	Reset	Description
31..6	Reserved	R	0	-
5..0	CMD_FLUSH_EN	RW	0	Enable of command flushing Set the corresponding bit of this field to '1' to start the command flushing Bit 13 for channel 5 Bit 12 for channel 4 .. Bit 8 for channel 0 When the command flushing is completed or there are no commands in the command queue of the AHB channel, the corresponding bit of this field will be automatically cleared to '0' by DDR2 memory controller. **Please note that the AXI interface does not support the**

Bit	Name	Type	Reset	Description
				command flushing, writing '1' of the corresponding AXI channel will not be automatically cleared to '0'.

7.2.18 Command Flush Status Register

Offset: 0x44

Table 7-21 Command Flush Status Register

Bit	Name	Type	Reset	Description
31..6	Reserved	R	0	-
5..0	FLUSH_DONE	RW1C	0	Status of the command flushing This field records the status of the command flushing for each channel. When a channel terminates the command flushing, the corresponding bit of this field will be set to '1' by DDR2 memory controller. Bit 13 for channel 5 Bit 12 for channel 4 .. Bit 8 for channel 0 This field should be cleared by writing '1' to the corresponding bit.

7.2.19 DDR2 PHY Update Control Register

Offset: 0x4C

Table 7-22 Command Flush Status Register

Bit	Name	Type	Reset	Description
31..16	Reserved	R	0	-
15..8	TWLEVEL_UP	RW	0	Write-leveling calibration timing interval 0 = Disable the write-leveling calibration interval 1 = 1x1024 200MHz cycles 2 = 2x024 200MHz cycles … 255 = 255x1024 200MHz cycles Note: This value should be reasonably set; otherwise, it may impact the performance.
7..0	TDLL_UP	RW	0	Digital DLL update delay control code timing interval 0 = Disable the digital DLL update 1 = 1x1024 200MHz cycles 2 = 2x1024 200MHz cycles … 255 = 255x1024 200MHz cycles Note: This value should be reasonably set; otherwise, it may impact the performance.

7.2.20 User Defined Register

Offset: 0x5C

Table 7-23 User Defined Register

Bit	Name	Type	Reset	Description
31..0	USER DEF REG	RW	0	User-defined register for specific usage

7.2.21 DDR2 PHY MISC Control Register1

Offset: 0x6C

Table 7-24 DDR2 PHY MISC Control Register1

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	DQS_SEL_BYTE0	RW	0	Add the delay in DQ or DQS, the delay value is controlled by dllsel_byte0 (PHYRDTR, 7.2.10). 1 = Read DQ is fixed, delay DQS. 0 = Read DQS is fixed, delay DQ.

7.2.22 Traffic Monitor Clock Cycle Register

Offset: 0x7C

Table 7-25 Traffic Monitor Clock Cycle Register

Bit	Name	Type	Reset	Description
31..0	TM_CLK_CYCLE_REG	RW	0	This register works only when the traffic monitor is enabled. The value of this register will be decreased by 1 for every clock cycle (5ns). When the value of this register is not zero, the traffic monitor will count the issued command number for each channel.

7.2.23 Command Count Register for Channel0

Offset: 0x80

Table 7-26 Command Count Register for Channel0

Bit	Name	Type	Reset	Description
31..0	CMD_CNT_REG0	R	0	This register records the number of the issued command from channel0, and will be cleared automatically when the register, TM_clk_cycle_reg, is written by users.

7.2.24 Command Count Register for Channel1

Offset: 0x84

Table 7-27 Command Count Register for Channel1

Bit	Name	Type	Reset	Description
31..0	CMD_CNT_REG1	R	0	This register records the number of the issued command from channel1, and will be cleared automatically when the register, TM_clk_cycle_reg, is written by users.

7.2.25 Command Count Register for Channel2

Offset: 0x88

Table 7-28 Command Count Register for Channel2

Bit	Name	Type	Reset	Description
31..0	CMD_CNT_REG2	R	0	This register records the number of the issued command from channel2, and will be cleared automatically when the register, TM_clk_cycle_reg, is written by users.

7.2.26 Command Count Register for Channel3

Offset: 0x8C

Table 7-29 Command Count Register for Channel3

Bit	Name	Type	Reset	Description
31..0	CMD_CNT_REG3	R	0	This register records the number of the issued command from channel3, and will be cleared automatically when the register, TM_clk_cycle_reg, is written by users.

7.2.27 Command Count Register for Channel4

Offset: 0x90

Table 7-30 Command Count Register for Channel4

Bit	Name	Type	Reset	Description
31..0	CMD_CNT_REG4	R	0	This register records the number of the issued command from channel4, and will be cleared automatically when the register, TM_clk_cycle_reg, is written by users.

7.2.28 Command Count Register for Channel5

Offset: 0x94

Table 7-31 Command Count Register for Channel5

Bit	Name	Type	Reset	Description
31..0	CMD_CNT_REG5	R	0	This register records the number of the issued command from channel5, and will be cleared automatically when the register, TM_clk_cycle_reg, is written by users.

7.2.29 AHB INCR Read Prefetch Length1

Offset: 0xA0

DDR2 memory controller provides two mechanisms to prefetch the read data for the AHB unspecified-length (INCR) read bursts: Limited prefetching and unlimited prefetching.
For limited prefetching, users should define a value used as “total number” of the read commands. The DDR2 memory controller will generate the DDR read commands when receiving an AHB INCR read burst. For example, if the user-defined value is *N*, DDR2 memory controller will generate three *N*+1 DDR read commands to prefetch the read data for any AHB INCR bursts.
For unlimited prefetching, users should define a value used as “speed factor” of the read commands. In this mode, DDR2 memory controller will keep generating the read commands until it receives an AHB early termination condition. The user-defined value determines the maximum number of the prefetch read commands that are on-going in pipeline. For example, if the user-defined value is *N*, DDR2 memory controller will first generate the *N*+1 DDR read commands and will then suspend the command generation until the on-going command count reaches *N*+1. When the data belonged to the first read command arrives the read FIFO, DDR2 memory controller will generate another read command if the on-going command count is less than *N*+1. This mechanism will be repeatedly performed until the AHB early termination occurs.

Table 7-32 AHB INCR Read Prefetch Length1

Bit	Name	Type	Reset	Description
31	CH3_LIMITED_PREF	RW	0	Prefetch mechanism of channel3 1: Limited prefetching 0: Unlimited prefetching
30..29	Reserved	R	0	-
28..24	CH3_PREF_VALUE	RW	3	User-defined value of channel3 for the INCR read prefetching 0: One command 1: Two commands … 31: Thirty two commands
23	CH2_LIMITED_PREF	RW	0	Prefetch mechanism of channel2 1: Limited prefetching 0: Unlimited prefetching
22..21	Reserved	R	0	-
20..16	CH2_PREF_VALUE	RW	3	User-defined value of channel2 for the INCR read prefetching 0: One command 1: Two commands … 31: Thirty two commands
15	CH1_LIMITED_PREF	RW	0	Prefetch mechanism of channel1 1: Limited prefetching 0: Unlimited prefetching
14..13	Reserved	R	0	-

Bit	Name	Type	Reset	Description
12..8	CH1_PREF_VALUE	RW	3	User-defined value of channel1 for the INCR read prefetching 0: One command 1: Two commands … 31: Thirty two commands
7	CH0_LIMITED_PREF	RW	0	Prefetch mechanism of channel0 1: Limited prefetching 0: Unlimited prefetching
6..5	Reserved	R	0	-
4..0	CH0_PREF_VALUE	RW	3	User-defined value of channel0 for the INCR read prefetching 0: One command 1: Two commands … 31: Thirty two commands

7.2.30 AHB INCR Read Prefetch Length2

Offset: 0xA4

Table 7-33 AHB INCR Read Prefetch Length2

Bit	Name	Type	Reset	Description
31..13	Reserved	R	0	-
12..8	CH5_PREF_VALUE	RW	3	User-defined value of channel5 for the INCR read prefetching 0: One command 1: Two commands … 31: Thirty two commands
7	CH4_LIMITED_PREF	RW	0	Prefetch mechanism of channel4 1: Limited prefetching 0: Unlimited prefetching
6..5	Reserved	R	0	-
4..0	CH4_PREF_VALUE	RW	3	User-defined value of channel4 for the INCR read prefetching 0: One command 1: Two commands … 31: Thirty two commands

7.2.31 Initialization of Waiting Cycle Count1

Offset: 0xA8

Table 7-34 Initialization of Waiting Cycle Count1

Bit	Name	Type	Reset	Description
31..20	Reserved	R	0	-
19..0	WAIT_CYCLE_200US	RW	0x0001A0AB	After powering up the ramp of SDRAM, a waiting time of 200 µs is required before the RESET signal becomes inactive. This register defines the waiting clock cycles of the DDR2 memory controller to satisfy the requirement of 200 µs.

7.2.32 QoS Control Register

Offset: 0xB0

Table 7-35 Setting for QoS Command Count

Read-Write Group	QoS Indepent_RW Enable	QoS Command Count
Enable	Enable	Registers QOSCNTRA (7.2.33) and QOSCNTRB (7.2.34) define the QoS command count for the read operation, and the command count constraints the number of read operations. Registers QOSCNTRC (7.2.35) and QOSCNTRD (7.2.36) define the QoS command count for the write operation, and the command count constraints the number of write operations.
	Disable	Registers QOSCNTRA (7.2.33) and QOSCNTRB (7.2.34) define the QoS command count for the read operation, and the command count constraints the number of read operations. Registers QOSCNTRA (7.2.33) and QOSCNTRB (7.2.34) define the QoS command count for the write operation, and the command count constraints the number of write operations.
Disable	Enable or Disable	Registers QOSCNTRA (7.2.33) and QOSCNTRB (7.2.34) define the QoS command count, and the command count constraints the number of read and write operations.

Table 7-36 QoS Control Register

Bit	Name	Type	Reset	Description
31..4	Reserved	R	0	-
3	QoS_INDEP_RW_EN	RW	0	QoS independent RW enable If this bit is set to '1', the setting of QoS_CmdCnt in QOSCNTRA (7.2.33) and QOSCNTRB (7.2.34) will be the limitation only for the read command. If this bit is set to '0', the setting of QoS CmdCnt in QOSCNTRA (7.2.33) and QOSCNTRB (7.2.34) will be the limitation for both the read and write commands.
2..1	QoS_PERIOD_MODE	RW	0	QoS period mode 00: 512 clock cycles 01: 1024 clock cycles 10: 2048 clock cycles 11: 4096 clock cycles
0	QoS_EN	RW	0	QoS enable

7.2.33 QoS Command Count RegisterA

Offset: 0xB4

Table 7-37 QoS Command Count RegisterA

Bit	Name	Type	Reset	Description
31..24	QoS_CH3CMD_CNT	RW	0	Command Count Limitation for Channel 3 0x00: No limit 0x01: 1 x 8 commands ...

Bit	Name	Type	Reset	Description
				0xff: 255 x 8 commands
23..16	QoS_CH2CMD_CNT	RW	0	Command Count Limitation for Channel 2 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands
15..8	QoS_CH1CMD_CNT	RW	0	Command Count Limitation for Channel 1 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands
7..0	QoS_CH0CMD_CNT	RW	0	Command Count Limitation for Channel 0 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands

7.2.34 QoS Command Count RegisterB

Offset: 0xB8

Table 7-38 QoS Command Count RegisterB

Bit	Name	Type	Reset	Description
31..16	Reserved	R	0	-
15..8	QoS_CH5CMD_CNT	RW	0	Command Count Limitation for Channel 5 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands
7..0	QoS_CH4CMD_CNT	RW	0	Command Count Limitation for Channel 4 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands

7.2.35 QoS Command Count RegisterC

Offset: 0xBC

If QoS_Indep_RW_en is set to ‘1’ (QOSCR, 7.2.32), Register QOSCNTRC (7.2.35) will be used for the write command.
If QoS_Indep_RW_en is set to ‘0’ (QOSCR, 7.2.32), Register QOSCNTRC (7.2.35) will be meaningless.

Table 7-39 QoS Command Count RegisterC

Bit	Name	Type	Reset	Description
31..24	QoS_CH3CMD_CNT_W	RW	0	Write Command Count Limitation for Channel 3 0x00: No limit 0x01: 1 x 8 commands

Bit	Name	Type	Reset	Description
				... 0xff: 255 x 8 commands
23..16	QoS_CH2CMD_CNT_W	RW	0	Write Command Count Limitation for Channel 2 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands
15..8	QoS_CH1CMD_CNT_W	RW	0	Write Command Count Limitation for Channel 1 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands
7..0	QoS_CH0CMD_CNT_W	RW	0	Write Command Count Limitation for Channel 0 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands

7.2.36 QoS Command Count RegisterD

Offset: 0xC0

If QoS_Indep_RW_en is set to ‘1’ (QOSCR, 7.2.32), Register QOSCNTRD (7.2.36) will be used for the write command.
If QoS_Indep_RW_en is set to ‘0’ (QOSCR, 7.2.32), Register QOSCNTRD (7.2.36) will be meaningless.

Table 7-40 QoS Command Count RegisterD

Bit	Name	Type	Reset	Description
31..24	Reserved	R	0	-
15..8	QoS_CH5CMD_CNT_W	RW	0	Write Command Count Limitation for Channel 5 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands
7..0	QoS_CH4CMD_CNT_W	RW	0	Write Command Count Limitation for Channel 4 0x00: No limit 0x01: 1 x 8 commands ... 0xff: 255 x 8 commands

7.2.37 Channel Arbitration Setup RegisterB

Offset: 0xC4
If Register CHARBRA (7.2.13) bit[31] “RW_Group_En”= ‘1’ and bit[30] “Indep_RW_EN”= ‘1’, Register CHARBRB (7.2.37) will be used for the write command or Register CHARBRB (7.2.37) will be meaningless.

Table 7-41 Channel Arbitration Setup RegisterB

Bit	Name	Type	Reset	Description
31..24	GROUP_GRANT_ COUNT_W	RW	0x10	This field will not be used if the Read-Write group arbitration is disabled. 0x00 = Issue a maximum of 1 command when the write group is granted. 0x01 = Issue a maximum of 2 commands when the write group is granted. … 0xff = Issue a maximum of 256 commands when the write group is granted.
23..19	Reserved	R	0	-
18..16	GROUP_GRANT_ COUNT_HIGH	RW	0	Please refer to the descriptions at Register CHARBRA (7.2.137.2.37) Total group_grant_count[7:0] = {group_grant_count_high, group_grant_count_low } CHARBRB (9.2.40) [18:16] indicates “group_grant_count_high” . CHARBRA (7.2.137.2.37) [28:24] indicates “group_grant_count_low”,
15..14	Reserved	R	0	-
13..8	BST_ORI_ARB_W	RW	0	Burst oriented arbitration enable bit When these bits are set, the burst transaction will not be broken during arbitration. Bit13 =1 indicates that the burst oriented arbitration of channel5 is enabled for the write commands. 0 indicates that the burst oriented arbitration of channel5 is disabled for the write commands. .. Bit 8 = 1 indicates that the burst oriented arbitration of channel0 is enabled for the write commands. 0 indicates that the burst oriented arbitration of channel0 is disabled for the write commands.
7..6	Reserved	R	0	-
5..0	CH_HI_PRIOR_W	RW	0	All channels are divided into two priority levels: High-priority level and low-priority level. These bits indicate which channels belong to the high-priority level. Setting each bit to ‘1’ indicates that the corresponding channels belong to the high-priority level. Bit 5 = 1 indicates that the write commands of channel 5 belong to the high-priority group. Bit 5 = 0 indicates that the write commands of channel 5 belong to the low-priority group. … Bit 0 = 1 indicates that the write commands of channel 0 belong to the high-priority group. Bit 0 = 0 indicates that the write commands of channel 0 belong to the low-priority group.

7.2.38 Channel Arbiter Grant Count Register C

Offset: 0xC8

Each channel has its own grant window counter. When the arbiter performs there-arbitration, the value set in this register will be loaded into the grant window counters of channel0,

channel1, channel2 and channel3. Once the granted channel pops a command to the memory controller, the grant window counter of this channel will be decreased by one. If the burst oriented function is disabled and the grant window counter of the current granted channel becomes '0', the arbiter will switch the grant to another channel. If the burst oriented function is enabled, the arbiter will switch grant when the grant window counter becomes '0' and the popped command will be the EOB (End of burst) command of a burst transaction, which is the burst transaction that will not be broken by the arbiter even when the grant window counter becomes '0'. The grant window counter of each channel defines the maximum service command counts allowed for each channel when that channel is granted by the arbiter. Please note that a command indicates a DDR2 WRAP4 RW command.
If Register CHARBRA (9.2.13) bit[31] "RW_Group_En"= '1'and bit[30] "Indep_RW_EN"= '1', Register CHARBRC (7.2.38) will be used for the write command or Register CHARBRC (7.2.38) will be meaningless.

Table 7-42 Channel Arbitration Setup RegisterC

Bit	Name	Type	Reset	Description
31..24	ARB_CNT3_W	RW	0xFF	Once channel3 is granted, the maximum allowed write commands will be issued. 0 = 1 write command 1 = 2 write commands … 255 = 256 write commands
23..16	ARB_CNT2_W	RW	0xFF	Once channel2 is granted, the maximum allowed write commands will be issued. 0 = 1 write command 1 = 2 write commands … 255 = 256 write commands
15..8	ARB_CNT1_W	RW	0xFF	Once channel1 is granted, the maximum allowed write commands will be issued. 0 = 1 write command 1 = 2 write commands … 255 = 256 write commands
7..0	ARB_CNT0_W	RW	0xFF	Once channel0 is granted, the maximum allowed write commands will be issued. 0 = 1 write command 1 = 2 write commands … 255 = 256 write commands

7.2.39 Channel Arbiter Grant Count Register D

Offset: 0xCC

Similar to Register CHGNTRA (7.2.14), this register defines the maximum service command counts of channel4 and channel5.

If Register CHARBRA (7.2.137.2.37) bit[30] “Indep_RW_EN”= ‘1’, Register CHGNTRD (7.2.39) will be used for the write commands. If Register CHARBRA (9.2.13) bit[30] “Indep_RW_EN”= ‘0’, Register CHGNTRD (7.2.39) will be meaningless.

Table 7-43 Channel Arbitration Setup RegisterD

Bit	Name	Type	Reset	Description
31..16	Reserved	R	0xFFFF	-
15..8	ARB_CNT5_W	RW	0xFF	Once channel5 is granted, the maximum allowed write commands will be issued. 0 = 1 write command 1 = 2 write commands … 255 = 256 write commands
7..0	ARB_CNT4_W	RW	0xFF	Once channel4 is granted, the maximum allowed write commands will be issued. 0 = 1 write command 1 = 2 write commands … 255 = 256 write commands

7.2.40 DDR2 PHY Read Path DLL Tuning for Falling Edge Register

Offset: 0x130

DLSEL_RD_FALL_byten is used to adjust the timing relation between read DQ and read DQS at the falling edge, Register RHYRDTR (7.2.10) is used for the rising edge.

Table 7-44 DDR Phy Read Path DLL Tuning for Falling Edge Register

Bit	Name	Type	Reset	Description
31..28	DLSEL_RD_FALL_BYTE7	RW	7	Delay setting of byte7 to adjust the falling-edge delay of DQS or DQ for read
27..24	DLSEL_RD_FALL_BYTE6	RW	7	Delay setting of byte6 to adjust the falling-edge delay of DQS or DQ for read
23..20	DLSEL_RD_FALL_BYTE5	RW	7	Delay setting of byte5 to adjust the falling-edge delay of DQS or DQ for read
19..16	DLSEL_RD_FALL_BYTE4	RW	7	Delay setting of byte4 to adjust the falling-edge delay of DQS or DQ for read
15..12	DLSEL_RD_FALL_BYTE3	RW	7	Delay setting of byte3 to adjust the falling-edge delay of DQS or DQ for read
11..8	DLSEL_RD_FALL_BYTE2	RW	7	Delay setting of byte2 to adjust the falling-edge delay of DQS or DQ for read
7..4	DLSEL_RD_FALL_BYTE1	RW	7	Delay setting of byte1 to adjust the falling-edge delay of DQS or DQ for read
3..0	DLSEL_RD_FALL_BYTE0	RW	7	Delay setting of byte0 to adjust the falling-edge delay of DQS or DQ for read

7.2.41 DDR PHY MISC2 Control Register

Offset: 0x134

Table 7-45 DDR Phy Misc2 Register

Bit	Name	Type	Reset	Description
31..24	Reserved	R	0	-
23	CLOCK_EN	RW	1	Enable DDR PHY internal clock 1 = Normal mode, all clock signals in DDR PHY are toggling. 0 = Low power mode, the unused clock signals are gated.
22..20	CMD_DLSEL	RW	7	DRAM Command/Address versus CK position adjustment
19..16	DUTYSEL	RW	8	DRAM clock CK/CKB duty adjusting bits
15	DSRONB	RW	1	Power control signal for keeping CKE and RESET_N_DRAM state 1 = Normal operation 0 = Maintain DDR PHY output value, CKE to “L” and RESET_N_DRAM to “H” for DRAM in the self-refresh mode.
14	IO15V	RW	1	IO voltage selection 1 = When the Combo PHY is operated at 1.5 V or 1.8 V. 0 = When the Combo PHY is operated at 1.2 V or 1.35 V.
13	RDLVL_GATE_EN	RW	0	MSDLY margin check 1 = Enable 0 = Disable
12	EYE_RDLVL_EN	RW	0	Eye margin check of DQ for DQS latch 1 = Enable 0 = Disable
11	Reserved	R	0	-
10..8	VREF_SELECT	RW	4	Bias voltage by PHY for VREF when 0x20 bit[15] self_bias = 1 000: 0.55 * VCC15O_DDR 001: 0.525 * VCC15O_DDR 010: 0.475 * VCC15O_DDR 011: 0.45 * VCC15O_DDR 100 ~ 111: 0.5 * VCC15O_DDR
7	Reserved	R	0	-
6..4	RONMD_DATA	RW	7	The driving impedance setting of DDR I/O (Data block) 111: 34 Ω 110: 40 Ω 101: 48 Ω 100: 60 Ω 011: 60 Ω 010: 80 Ω 001: 120 Ω 000: 240 Ω
3	Reserved	R	0	-
2..0	RONMD_CMDADDR	RW	7	The driving impedance setting of DDR I/O (CMDADDR block) 111: 34 Ω 110: 40 Ω 101: 48 Ω 100: 60 Ω 011: 60 Ω 010: 80 Ω 001: 120 Ω 000: 240 Ω

7.2.42 DDR ELASTIC FIFO Control Register

Offset: 0x138

Write this register before the initial sequence triggers the elastic fifo reset, and the elastic fifo should be reset before the initial sequence when DDR2 PHY is ready.
Users should write this register before register MCSR (7.2.2) is set to '1'.

Table 7-46 DDR ELASTIC FIFO Control Register

Bit	Name	Type	Reset	Description
31..9	Reserved	R	0	-
8	EFIFO_RST	RW	0	Write to this bit will trigger efifo reset, this bit is self-clear. 1: Reset elastic fifo, user can reset elastic only when there are no DDR command in the whole controller pipe line stage 0: No effect
7..5	Reserved	R	0	-
4	AUTO_RST_SRF	RW	0	Automatically reset elastic fifo when self-refresh state is entered 1: Memory Controller will reset elastic fifo automatically when self-refresh mode is entered. 0: Memory Controller will not reset elastic fifo automatically when self-refresh mode is entered.
3..2	READ_SYNC_POS	RW	0	Read data elastic fifo synchronization position for POSTCTLCLK to MClk The valid values are 0, 1, 2. Set to 1 can work in most cases, while user can use post-simulation or STA report to decide the proper read synchronization position setting to reduce latency
1..0	CMD_SYNC_POS	RW	0	Command elastic fifo synchronization position for MClk to POSTCTLCLK The valid values are 0, 1, 2. Set to 1 can work in most cases, while user can use post-simulation or STA report to decide the proper command synchronization position setting to reduce latency

7.2.43 DDR PHY CMDADDR Block Clock Tree Control Register

Offset: 0x13C

Table 7-47 CMDADDR Block Clock Tree Control Register

Bit	Name	Type	Reset	Description
31..11	Reserved	R	0	-
10..8	CMD_CKTREE_SKEW	RW	3	The control bits are used to adjust the skew between POSTCTLCK and POSTPHYCLK2X at command block of DDR PHY.
7..5	Reserved	R	0	-
4..0	CMD_CKTREE_DELAY	RW	0	Programmable delay control bits which are used to balance the clock tree of command block of DDR PHY and DDR2 memory controller

7.3 Memory Address Table (MA Table)

The DDR2 memory controller supports various types of DDR2 SDRAMs. Users must refer to the MA table for the external rank0 configuration register, RNK_TYPE.
Each rank must choose the correct type of the MA tables. DDR2 memory controller supports two types of MA tables, which can be configured through the AMTSEL bit of the configuration register MCCR (7.2.1).

- When AMTSEL = 00, the system address will map with RA, BA, and CA.
- When AMTSEL = 01, the system address will map with BA, RA, and CA.

7.3.1 MA Table (AMTSEL = 00) for DDR2 Mode

Table 7-48 MA Table (AMTSEL=00) in 16-bit DDR2 Mode

MATABLE	**Row Address x Column Address x Bank Address**		
TYPE	13x9x2	13x10x2	14x10x2
SELECT	000	001	010
Column	**CA**	**CA**	**CA**
MA0	0	0	0
MA1	1	1	1
MA2	2	2	2
MA3	3	3	3
MA4	4	4	4
MA5	5	5	5
MA6	6	6	6
MA7	7	7	7
MA8	8	8	8
MA9	9	9	9
MA10	-	10	10
BANK	**BA**	**BA**	**BA**
BA0	10	11	11
BA1	11	12	12
ROW	**RA**	**RA**	**RA**
MA0	12	13	13
MA1	13	14	14
MA2	14	15	15
MA3	15	16	16
MA4	16	17	17
MA5	17	18	18
MA6	18	19	19
MA7	19	20	20
MA8	20	21	21
MA9	21	22	22
MA10	22	23	23
MA11	23	24	24
MA12	24	25	25
MA13	-	-	26

7.3.2 MA Table (AMTSEL = 01) for DDR2 Mode

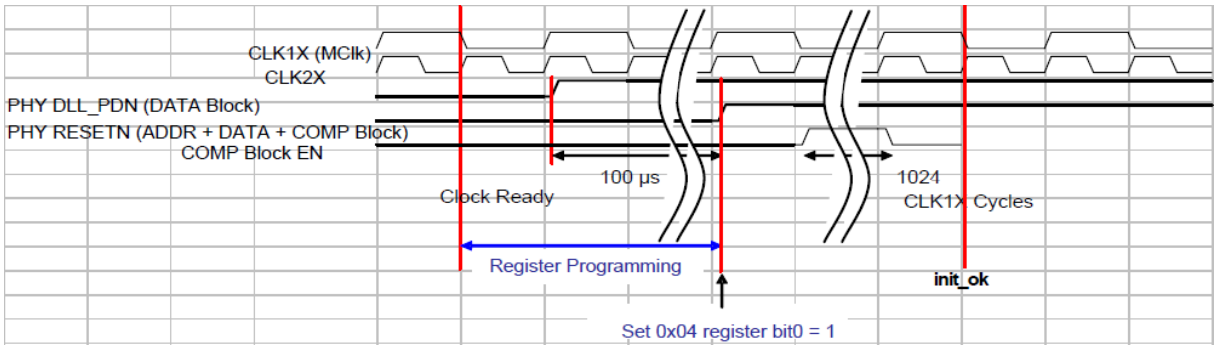
Table 7-49 MA Table (AMTSEL=01) in 16-bit DDR2 Mode

MATABLE	**Row Address x Column Address x Bank Address**		
TYPE	13x9x2	13x10x2	14x10x2
SELECT	000	001	010
Column	**CA**	**CA**	**CA**
MA0	0	0	0
MA1	1	1	1
MA2	2	2	2
MA3	3	3	3
MA4	4	4	4
MA5	5	5	5
MA6	6	6	6
MA7	7	7	7
MA8	8	8	8
MA9	9	9	9
MA10	-	10	10
BANK	**BA**	**BA**	**BA**
BA0	23	24	25
BA1	24	25	26
ROW	**RA**	**RA**	**RA**
MA0	10	11	11
MA1	11	12	12
MA2	12	13	13
MA3	13	14	14
MA4	14	15	15
MA5	15	16	16
MA6	16	17	17
MA7	17	18	18
MA8	18	19	19
MA9	19	20	20
MA10	20	21	21
MA11	21	22	22
MA12	22	23	23
MA13	-	-	24

7.4 Miscellaneous

7.4.1 Reset Sequence of DDR2 PHY

Figure 7-6 Reset Sequence of DDR2 PHY



Notes:

1. The programming sequence of the DDR2 memory controller register must be terminated before the de-assertion of PHY RESETN, except for Register MCSR (7.2.1).
2. The initial command (MCSR (7.2.1), bit 0) of the DDR2 memory controller must be programmed after the de-assertion of PHY RESETN.

7.4.2 Initialization Sequence of DDR2 Mode

Please follow the sequence below to initialize the DDR2 memory controller with the DDR2 PHY.

1. Wait until all the power supplies, reference voltages, and clocks become stable.
   a. Wait 200 µs
   b. Release the DDR2 PHY DLLPDN
   c. Set the Configuration Register (Offset = 0x00)
      Set the Mode Register and Extended Mode Value Register (Offset = 0x08)
      Set the Extended Mode Register2 and Extended Mode Register 3 (Offset = 0x0C)
      Set the Rank0/1 Register (Offset = 0x10)
      Set the DDRxTiming Parameter 0 Register (Offset = 0x14)
      Set the DDRxTiming Parameter 1 Register (Offset = 0x18)
      Set the DDRxTiming Parameter 2 Register (Offset = 0x1C)
      Set the PHY Command andData Block Control Register (Offset = 0x20)
      Set the PHY Read Path DLL Delay Tuning Control Register (Offset = 0x24)
      Set the PHY Compensation Block control Register (Offset = 0x28)
      Set the Arbitration Setup Register (Offset = 0x30)
      Set the Channel Arbiter Grant Count –A Register (Offset = 0x34)
      Set the Channel Arbiter Grant Count –B Register (Offset = 0x38)
      Set the DDRx PHY Write/Read Timing Control Register (Offset = 0x3C)
      Set the msdly byte control Register (Offset = 0x74)
   d. Release the DDR PHY RESETN

e. Set the EFIFO Control Register (Offset = 0x138)
f. Set the DDR2 Controller State Control Register (0x04)
   Bit0 = 1 (Initial Command) to initiate the DDR2 SDRAM
g. Read bit8 of the DDR2 Controller State Control Register (0x04). When this bit (Initial OK) is set to one, it iondicates that the DDR2 SDRAM is well initialized and ready for use.

2. Please refer to the Initial Code in the DDR2 200MHz 16bit system for a Micron_MT47H64M16-25E DDR2 memory device with ODT set to 75Ohm.
   (DDR2 Controller register system address is at 0x7B000000, DDR2 Memory Space is at 0x00000000.)

```
mem.write    0x7B000000  0x00078501
mem.write    0x7B000008  0x20040432
mem.write    0x7B00000C  0x00000000
mem.write    0x7B000010  0x00000013
mem.write    0x7B000014  0x19080A07
mem.write    0x7B000018  0x11221212
mem.write    0x7B00001C  0x0000192D
mem.write    0x7B000020  0x00004F41
mem.write    0x7B00003C  0x00710011
mem.write    0x7B000048  0x00555500
mem.write    0x7B0000A8  0x00004E20
```

   WAIT FOR DLL LOCK

```
mem.write    0x7B000038  0x00000001
mem.write    0x7B000004  0x00000001
```

   WAIT UNTIL INITIAL OK (BIT8) IS SET

3. Please refer to the typical DDR2 SDRAM timing parameter.
   TREFI = 7800 ns is for the commercial requirement and 3900 ns is for the industrial requirement.
   TRFC maybe different depending on different chip sizes.

7.4.3 Write/Read Command to Write/Read command Interval Counting Example

Write Command to Read Command

For examples:
TWTR = Register TMPR1 (7.2.7) [30:28] = 2
TWtoR_ctrl = Register TMPR2 (7.2.8) [31:30]
t_{WL} = Write Latency = 5
t_{RL} = Read Latency = 6

Write to read same rank
tWtoR_same_rank = tWL + 2 + TWTR = 5 + 2 + 2 = 9 →10 / 200MHz

Read Command to Write Command

For examples:
t_{WL}= Write Latency = 5
t_{RL}= Read Latency = 6
TRtoW_ctrl = Register TMPR2 (7.2.8) [25:24]

tRtoW = 4 + TRtoW_ctrl
TRtoW_ctrl = 0, tRtoW = 4 + 0 = 4 → 5 / 200MHz
TRtoW_ctrl = 1, tRtoW = 4 + 1 = 5 → 6 / 200MHz
TRtoW_ctrl = 2, tRtoW = 4 + 2 = 6 → 7 / 200MHz
TRtoW_ctrl = 3, tRtoW = 4 + 3 = 7 → 8 / 200MHz

Write Command to Write Command
tWtoW_ctrl = Register TMPR2 (7.2.8) [29:28]

Same Rank
tWtoW = 1 → 2 / 200MHz

Read Command to Read Command
TRtoR_ctrl = Register TMPR2 (7.2.8) [27:26]

Same Rank
tRtoR= 1 → 2 / 200MHz

8 Asynchronous External Interface (AEI)

Chip select 0: data area: 0x98000000; configuration register: 0x9C000000
Chip select 1: data area: 0x99000000; configuration register: 0x9C000004

The asynchronous transfers are divided into the following four phases: SETUP, DATA, HOLD and RECOVER. These names are used within the following pages of this section.

8.1 AEI configuration registers

There is one dedicated 32Bit wide configuration register for every AEI chip select to setup the AEI and adjust the timing of the asynchronous transfer. The configuration registers must only be accessed using 32Bit wide accesses. The user can adjust the duration of the four AEI phases by setting SETUP_WS, DATA_WS, HOLD_WS and RECOVER_WS.
Legal values for the wait state configuration are 1 to 127 clock cycles based on a 200MHz clock.

For example: a DATA_WS setting of 3 will result in a 15.00ns lasting DATA phase (1/200MHz * 3)

The configuration register for AEI_CS0_N is mapped to 0x9C000000 and the configuration register for AEI_CS1_N is mapped to 0x9C000004. The reset value for both configuration registers is 0x04281030.

The following table shows the bit assignment of the AEI configuration register:

Table 8-1 Bit assignment of the AEI configuration register

Bit	Name	Description Reset value: 0x04281030
31 …25	SETUP_WS	Number of clock cycles for SETUP phase. Only address, chip select and byte enable are driven. Possible values: 1 to 127
24 … 18	DATA_WS	Number of clock cycles for DATA phase. Additionally data and output- or write-enable are driven. Read data is sampled after last data wait state. Possible values: 1 to 127
17 … 11	HOLD_WS	Number of clock cycles for HOLD phase. Output- or write-enable is retracted by AEI, write data stays on bus. Possible values: 1 to 127
10 … 4	RECOVER_WS	Number of clock cycles for RECOVER phase. No outputs are driven. Possible values: 1 to 127
3	USE_WAIT	Use AEI_WAIT to extend DATA phase of AEI transfer 0: Data phase ends after configured DATA_WS time, regardless of AEI_WAIT 1: Data phase is extended **after DATA_WS time** until AEI_WAIT is low
2	HWORD_BUS	Width of the AEI data bus 0: AEI data bus width is 8Bit 1: AEI data bus width is 16Bit

Bit	Name	Description	Reset value: 0x04281030
0 ... 1	reserved	Read as zero, writes have no effect.	

See chapter 8.5 for timing considerations.

8.2 AEI read transfer

The following two figures give a brief overview of the AEI timing for read transfers.

Figure 8-1 AEI read transfer

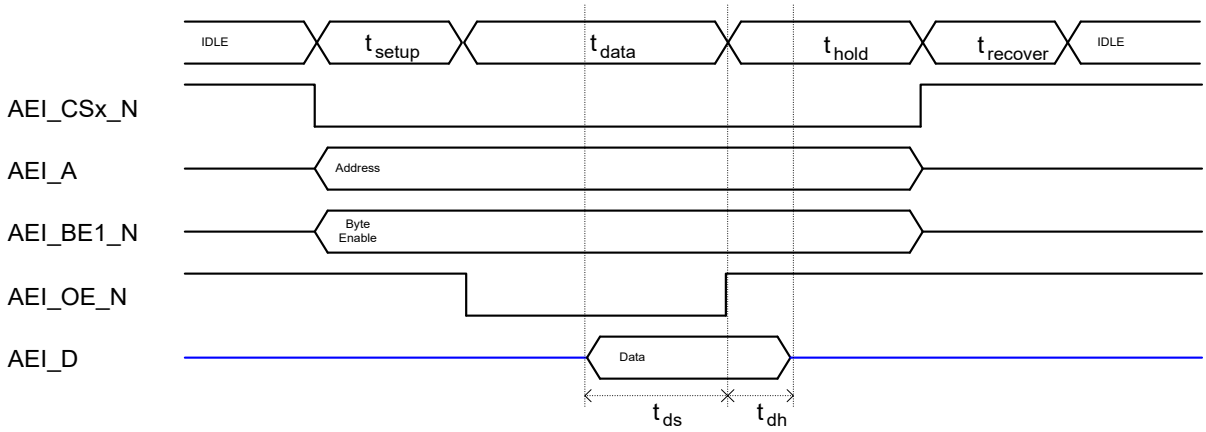


Table 8-2 Timing values for basic AEI read access

Symbol	Parameter	Min	Max	Comment
t_{setup} t_{data} t_{hold} $t_{recover}$	Duration of AEI phases	$1*t_{ck}$	$127*t_{ck}$	e.g.: t_{data} = DATA_WS * t_{ck} with t_{ck} = 1/200MHz
t_{ds}	Data setup time	10ns		
t_{dh}	Data hold time	10ns		

The DATA phase of the read transfer can be extended by using the high active input AEI_WAIT and setting USE_WAIT in the configuration register to one.
If USE_WAIT is one the AEI will extend the DATA phase of the read access as long as the input AEI_WAIT is sampled high after the configured DATA_WS cycles and change to the HOLD phase as soon as AEI_WAIT gets low. Please note that the asynchronous input AEI_WAIT is internally synchronized to the 200MHz AEI system clock and therefore it takes a maximum of 20ns to detect the high level of AEI_WAIT. Therefore the DATA phase must be configured to last for at least four cycles if AEI_WAIT is used!

Figure 8-2 AEI read transfer using AEI_WAIT

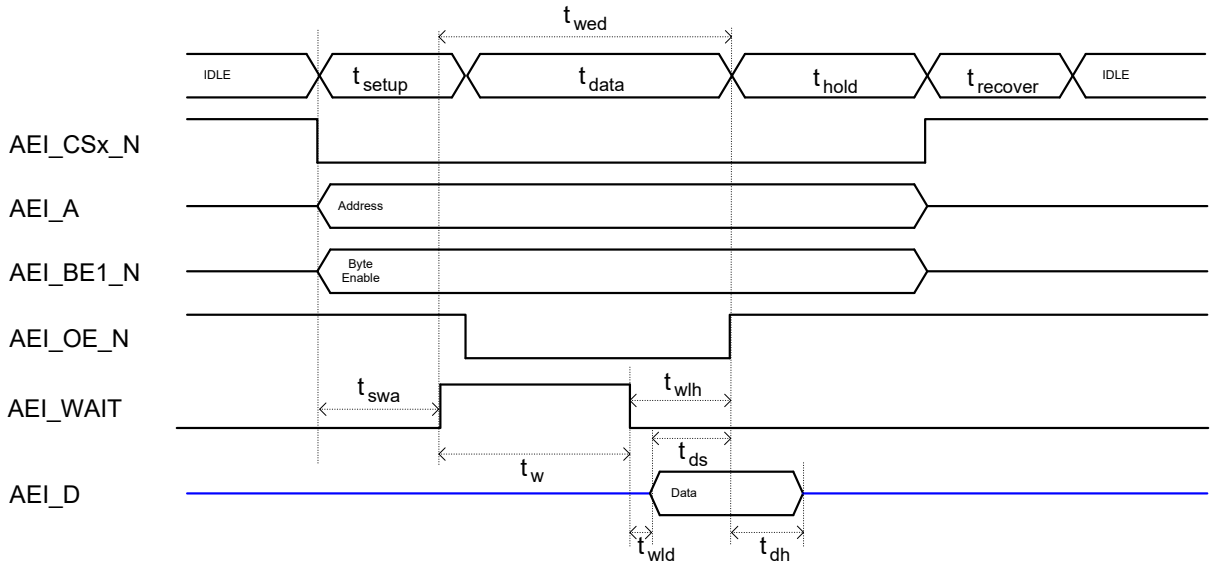


Table 8-3 Timing values of AEI read access using AEI_WAIT

Symbol	Parameter	Min	Max	Comment
t_{setup} t_{data} t_{hold} $t_{recover}$	Duration of AEI phases	$1*t_{ck}$	$127*t_{ck}$	e.g.: t_{data} = DATA_WS * t_{ck} with t_{ck} = 1/200MHz
t_{ds}	Data setup time	10ns	-	
t_{dh}	Data hold time	10ns	-	
t_{wed}	AEI_WAIT enable to end of DATA phase	20ns	-	To allow detection of AEI_WAIT=1
t_w	AEI_WAIT active width	7ns	-	
t_{wld}	AEI_WAIT low before data valid	-	10ns	
t_{swa}	Start of access to AEI_WAIT active	0ns	t_{setup} + t_{data} – 20ns	
t_{wlh}	AEI_WAIT low to change to HOLD phase	10ns	20ns	

8.3 AEI write transfer

The following two figures give a brief overview of the AEI timing for write transfers.

Figure 8-3 AEI write transfer

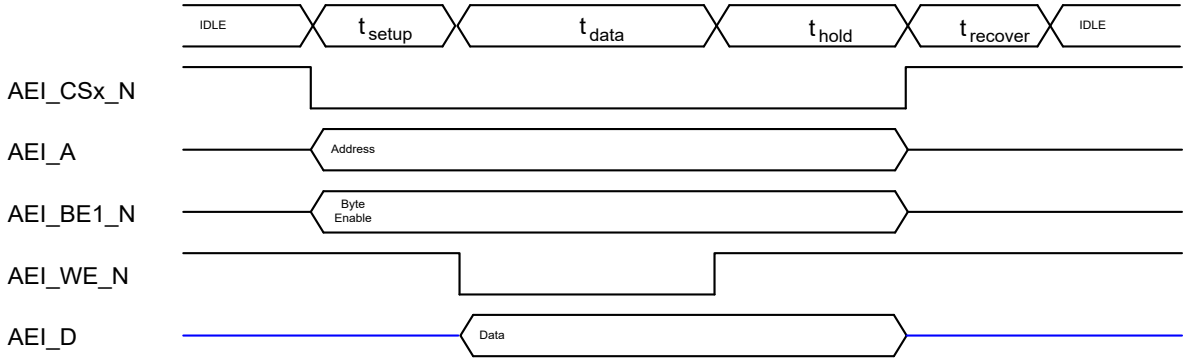


Table 8-4 Timing values of AEI write access

Symbol	Parameter	Min	Max	Comment
t_{setup} t_{data} t_{hold} $t_{recover}$	Duration of AEI phases	$1*t_{ck}$	$127*t_{ck}$	e.g.: t_{data} = DATA_WS * t_{ck} with t_{ck} = 1/200MHz

The DATA phase of the write transfers can be extended by using the high active input AEI_WAIT and setting USE_WAIT in the configuration register to one. If USE_WAIT is one the AEI will extend the DATA phase of the current access as long as the input AEI_WAIT is sampled high after the configured DATA_WS cycles and change to the HOLD phase as soon as AEI_WAIT gets low. Please note that the asynchronous input AEI_WAIT is internally synchronized to the 200MHz AEI system clock and therefore it takes a maximum of 20ns to detect the high level of AEI_WAIT. Therefore the DATA phase must be configured to last for at least four cycles if AEI_WAIT is used!

Figure 8-4 AEI write transfer using AEI_WAIT

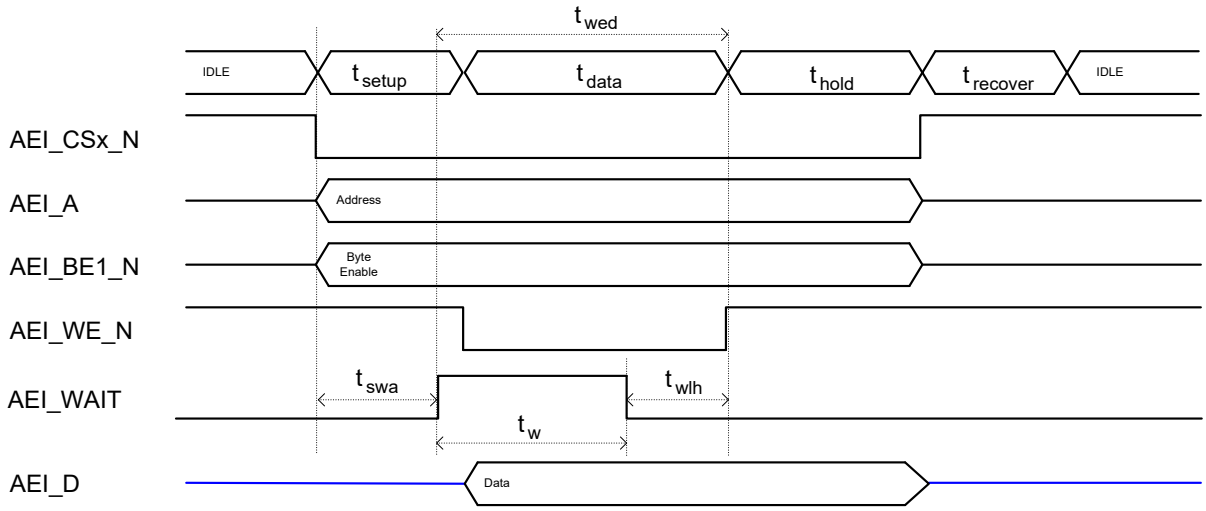


Table 8-5 Timing values of AEI write access using AEI_WAIT

Symbol	Parameter	Min	Max	Comment
t_{setup} t_{data} t_{hold} $t_{recover}$	Duration of AEI phases	1*t_{ck}	127*t_{ck}	e.g.: t_{data} = DATA_WS * t_{ck} with t_{ck} = 1/200MHz
t_{wed}	AEI_WAIT enable to end of DATA phase	20ns	-	To allow detection of AEI_WAIT=1
t_w	AEI_WAIT active width	7ns	-	
t_{swa}	Start of access to AEI_WAIT active	0ns	t_{setup} + t_{data} – 20ns	
t_{wlh}	AEI_WAIT low to change to HOLD phase	10ns	20ns	

8.4 AEI data areas

The both 2MB AEI data areas are mapped to the following locations in the ANTAIOS address space:

AEI data area chip select 0: 0x98000000-0x981FFFFF
AEI data area chip select 1: 0x99000000-0x991FFFFF

Read or write transfers to the data areas can be performed 8, 16 or 32Bit wide and will be automatically transformed to 8 or 16Bit wide accesses on the AEI interface depending on the width configuration of the AEI data bus.
E.g.: If AEI_CS0 is configured 8Bit wide a 32Bit write access to address 0x98000000 with 0x44332211 as data will result in the following four 8Bit wide consecutive write transfers with activated AEI_CS0_N on the AEI bus:

1. 0x11 to 0x000000
2. 0x22 to 0x000001
3. 0x33 to 0x000002

4. 0x44 to 0x000003

8.5 Timing considerations

As described in the previous section transfers wider as the AEI data bus width are automatically transformed to a series of transfers with the AEI data bus width.

The maximum length of such a series of transfers must not exceed a total time of 10.2µs.

The following table gives a formula to calculate the total length of the AEI transfer depending on transfer width and AEI data bus width:

Table 8-6 Calculation of AEI transfer length

Transfer width	AEI data bus width	Length of the complete AEI transfer
8Bit	8Bit	1 x (t_{setup}+t_{data}+t_{hold}) + ∑wait_delay[1)]
16Bit	8Bit	2 x (t_{setup}+t_{data}+t_{hold}) + 1 x $t_{recover}$ + ∑wait_delay[1)]
32Bit	8Bit	4 x (t_{setup}+t_{data}+t_{hold}) + 3 x $t_{recover}$ + ∑wait_delay[1)]
8Bit	16Bit	1 x (t_{setup}+t_{data}+t_{hold}) + ∑wait_delay[1)]
16Bit	16Bit	1 x (t_{setup}+t_{data}+t_{hold}) + ∑wait_delay[1)]
32Bit	16Bit	2 x (t_{setup}+t_{data}+t_{hold}) + 1 x $t_{recover}$ + ∑wait_delay[1)]

1) "∑wait_delay" denotes the total sum of extra time added to the DATA phase by an active AEI_WAIT during the whole series of the AEI transfer. If AEI_WAIT is not used ∑wait_delay = 0.

9 VPC3+ Profibus DP V2 Slave

Offset: 0x68000000

See VPC3+S User Manual for details.

Attention:
Access the VPC3+ only with Byte-Size(8Bit) not 16Bit or 32 Bit.

10 CAN

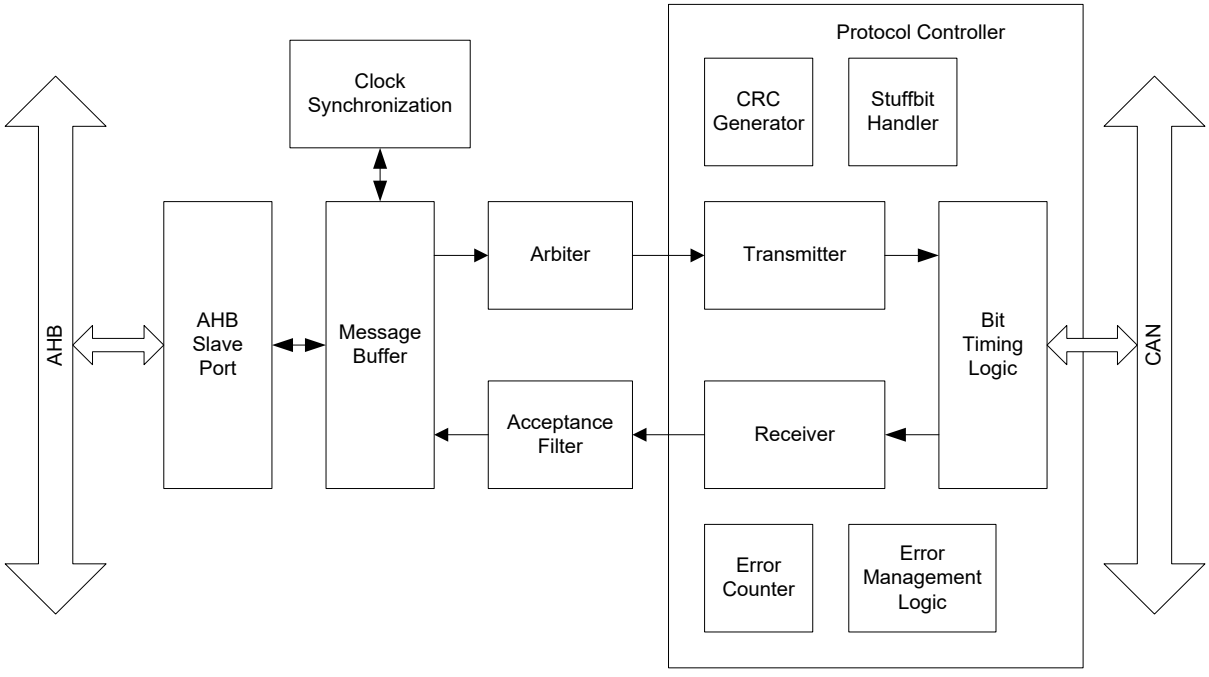
Offsets: CAN1: 0x6A000000
CAN2: 0x6C000000

The ANATIOS incorporates two identical CAN modules (CAN1 and CAN2). These modules use the serial communications protocol conforming to CAN version 2.0 A and B. Sending and receiving is available in standard and extended frame format.

- conform to CAN (Controller Area Network) versions 2.0 A and B
- sending and receiving is available in standard and extended frame format
- support transfer rates 10 kbit/s to 1 Mbit/s
- supplies 15 send/receive message objects
- support of two global masks for acceptance filtering
- support of media identification
- support of FIFO structures
- support of synchronization mechanism
- support monitoring mode

10.1 Overview

Figure 10-1 CAN Module



10.1.1 AHB Slave Port

The CAN module is connected to AHB via an AHB Slave, which provides only word data access.

10.1.2 Message Buffer

The message buffer provides storage for up to 15 message objects of maximum 8 data byte length. Each of these objects has a unique identifier and its own set of control and status bits. After the initial configuration, the message buffer can handle the reception and transmission of data without further user actions. Several message objects can be combined to FIFO structures.

10.1.3 Arbiter

The arbiter selects the message object with the highest priority and a transmission request, and makes it available to the protocol controller for sending.

10.1.4 Acceptance Filter

The acceptance filter checks if received frames can be stored in a message object. For this purpose, two global mask registers (applied to identifier) and the media ID registers (applied to the first and second data bytes) are available.

10.1.5 Protocol Controller

Serial data transfer and protocol handling is done by the protocol controller. In particular it performs the following tasks:

- data encapsulation
- frame coding
- error detection
- error signalling
- acknowledgement
- bit encoding
- synchronization

10.1.6 Clock Synchronization

The clock synchronization provides some timer functions for implementing a synchronization mechanism.

10.2 Memory Map and Register Definition

Table 10-1 lists and describes the registers of ANTAIOS CAN module. The address is given in the hexadecimal format. The second column indicates the access type of the register. Only 32-bit access supported.

- R: read only
- RW: read and write access

Table 10-1 Summary of CAN Registers

Address Offset	Type	Size (Byte)	Description	Reset Value
0x000	RW	1	Control Register	0x41
0x001	RW	1	Status Register	0x87

Address Offset	Type	Size (Byte)	Description		Reset Value
0x002	RW	1	Transmit Error Counter		0
0x003	RW	1	Receive Error Counter		0
0x004	RW	1	Interrupt Register		0
0x005	RW	3	Bit Timing Register		0x12_00_00
0x008	RW	4	Global Mask 0		0
0x00C	RW	4	Global Mask 1		0
0x010	RW	4	Message Object 0	Arbitration Register	
0x014	RW	8x1		Data Byte 7 .. 0	
0x01C	RW	1		Message Control Register	0
0x01D	RW	1		Message Configuration	0
0x01E	RW	2	Media ID Mask Register		0
0x020	RW	4	Message Object 1	Arbitration Register	
0x024	RW	8x1		Data Byte 7 .. 0	
0x02C	RW	1		Message Control Register	0
0x02D	RW	1		Message Configuration	0
0x02E	RW	2	Media Arbitration Register		0
0x030	RW	4	Message Object 2	Arbitration Register	
0x034	RW	8x1		Data Byte 7 .. 0	
0x03C	RW	1		Message Control Register	0
0x03D	RW	1		Message Configuration	0
0x03E	RW	2	Send Interval Time		0
0x040	RW	4	Message Object 3	Arbitration Register	
0x044	RW	8x1		Data Byte 7 .. 0	
0x04C	RW	1		Message Control Register	0
0x04D	RW	1		Message Configuration	0
0x04E	R	2	Release Number		0x00_0A
0x050	RW	4	Message Object 4	Arbitration Register	
0x054	RW	8x1		Data Byte 7 .. 0	
0x05C	RW	1		Message Control Register	0
0x05D	RW	1		Message Configuration	0
0x05E	RW	2	reserved		
0x060	RW	4	Message Object 5	Arbitration Register	
0x064	RW	8x1		Data Byte 7 .. 0	
0x06C	RW	1		Message Control Register	0
0x06D	RW	1		Message Configuration	0
0x06E	RW	2	reserved		
0x070	RW	4	Message Object 6	Arbitration Register	
0x074	RW	8x1		Data Byte 7 .. 0	
0x07C	RW	1		Message Control Register	0
0x07D	RW	1		Message Configuration	0
0x07E	RW	2	reserved		
0x080	RW	4	Message Object 7	Arbitration Register	
0x084	RW	8x1		Data Byte 7 .. 0	
0x08C	RW	1		Message Control Register	0
0x08D	RW	1		Message Configuration	0
0x08E	RW	2	reserved		
0x090	RW	4	Message Object 8	Arbitration Register	

Address Offset	Type	Size (Byte)	Description		Reset Value
0x094	RW	8x1		Data Byte 7 .. 0	
0x09C	RW	1		Message Control Register	0
0x09D	RW	1		Message Configuration	0
0x09E	RW	2	reserved		
0x0A0	RW	4	Message Object 9	Arbitration Register	
0x0A4	RW	8x1		Data Byte 7 .. 0	
0x0AC	RW	1		Message Control Register	0
0x0AD	RW	1		Message Configuration	0
0x0AE	RW	2	reserved		
0x0B0	RW	4	Message Object 10	Arbitration Register	
0x0B4	RW	8x1		Data Byte 7 .. 0	
0x0BC	RW	1		Message Control Register	0
0x0BD	RW	1		Message Configuration	0
0x0BE	RW	2	reserved		
0x0C0	RW	4	Message Object 11	Arbitration Register	
0x0C4	RW	8x1		Data Byte 7 .. 0	
0x0CC	RW	1		Message Control Register	0
0x0CD	RW	1		Message Configuration	0
0x0CE	RW	2	reserved		
0x0D0	RW	4	Message Object 12	Arbitration Register	
0x0D4	RW	8x1		Data Byte 7 .. 0	
0x0DC	RW	1		Message Control Register	0
0x0DD	RW	1		Message Configuration	0
0x0DE	RW	2	reserved		
0x0E0	RW	4	Message Object 13	Arbitration Register	
0x0E4	RW	8x1		Data Byte 7 .. 0	
0x0EC	RW	1		Message Control Register	0
0x0ED	RW	1		Message Configuration	0
0x0EE	RW	2	reserved		
0x0F0	RW	4	Message Object 14	Arbitration Register	
0x0F4	RW	8x1		Data Byte 7 .. 0	
0x0FC	RW	1		Message Control Register	0
0x0FD	RW	1		Message Configuration	0
0x0FE	RW	2	reserved		
0x100	RW	4	System Clock Value		
0x104	RW	4	Receive Delay Timer		

10.2.1 General Register

10.2.1.1 Control Register

Offset: 0x000

Table 10-2 Control Register

Bit	Name	Type	Reset	Description
7	MONITOR	RW	0	Monitoring Enable Switch to monitoring mode, in which no Acknowledge and frames are generated.

Bit	Name	Type	Reset	Description
6	CCE	RW	0	Configuration Change Enable Allows or inhibits user access to the bit timing register.
5	RXDLY	RW	0	Receive Delay Timer Enables the receive delay timer (time base: 1 µs).
4	START_TIME	RW	0	Start System Clock Enables the timer for the system clock.
3	EIE	RW	0	Error Interrupt Enable Enables or disables interrupt generation by a status change (BOFF, EPAS or EWRN) in the status register.
2	SIE	RW	0	Status Change Interrupt Enable Enables or disables interrupt generation when a message transfer (reception or transmission) is successfully completed or a CAN bus error is detected (and registered in the status register).
1	IE	RW	0	Interrupt Enable Enables or disables interrupt generation from the CAN module to the advanced IRQ controller of the ANTAIOS. Bit does not affect status updates.
0	INIT	RW	1	Initialization Starts the initialization of the CAN module, when set. No frames are transferred during initialization. If INIT is set, the CPU can change the message objects without previous locking with the MSGVAL bit. The user finishes the initialization by clearing this bit.

10.2.1.2 Status Register

Offset: 0x001

Table 10-3 Status Register

Bit	Name	Type	Reset	Description
7	BOFF	R	1	Bus Off Status Indicates when the CAN module is in Bus Off state (TEC ≥ 256).
6	EPAS	R	0	Error Passive Status Indicates when the CAN module is in Error Passive state (REC ≥ 128 ‖ 255 > TEC ≥ 128).
5	EWRN	R	0	Error Warning Status Indicates that at least one of the error counters has reached the error warning limit of 96.
4	RXOK	RW	0	Received Message Successfully Indicates that a message has been received successfully, since this bit was last reset by the CPU (the CAN module does not reset this bit!).
3	TXOK	RW	0	Transmitted Message Successfully Indicates that a message has been transmitted successfully (error free and acknowledged by at least one other node), since this bit was last reset by the CPU (the CAN module does not reset this bit!).
2..0	LEC	RW	7	Last Error Code This field holds a code which indicates the type of the last error occurred on the CAN bus. If a message has been transferred (reception or transmission) without error, this bit will be cleared. The value seven is unused and may be written by the user to check for updates. 0 = No Error 1 = Stuff Error: More than 5 equal bits in sequence have occurred in a part of a received message where this is not allowed. 2 = Form Error: A fixed format part of a received frame has the wrong format. 3 = Ack Error: The massage this CAN module transmitted was not acknowledged by another node. 4 = Bit1 Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (logical 1), but the monitored bus value was dominant. 5 = Bit0 Error: During the transmission of a message (or acknowledge bit, active error flag or overload flag), the device wanted to send a dominant level (logical 0), but the monitored bus value was recessive. During bus-off recovery this status is set each time a sequence of 11 recessive bits has been monitored. This enables the user to monitor the proceeding of the bus-off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). 6 = CRC Error: The CRC check sum was incorrect in the message received.

10.2.1.3 Interrupt Register

Offset: 0x004

Table 10-4 Interrupt Register

Bit	Name	Type	Reset	Description
7..0	INTID	R	0	Interrupt Identifier This number indicates the cause of the interrupt. When no interrupt is pending, the value will be 0. 0 = Idle — No interrupt request pending. 1 = Status change — The CAN module has updated the status register. To update the INITID value the status partition of the control register has to be read. (2+N) = Message N — Bit INTPND in the message control register of message object N has been set.(N = 0..14)

The interrupt line remains active until either INTID gets zero (i.e. the interrupt requester has been serviced) or until IE is reset (i.e. interrupts are disabled). The interrupt with the lowest number has the highest priority. If higher priorities interrupt (lower number) occurs before the current interrupt is processed, INTID is updated and the new interrupt overrides the last one. On the other hand a lower priority interrupt (higher number) is not able to override the last one.
Reading the status register when an interrupt is pending, resets the pending interrupt request.

10.2.1.4 CAN Bit Timing Register

Offset: 0x005

Table 10-5 CAN Bit Timing Register

Bit	Name	Type	Reset	Description
23	SAM	RW	0	Sampling Mode 0 = one time sampling 1 = three time sampling with majority decision
22..20	TSEG2	RW	1	Time segment after sample point There are (TSEG2+1) time quanta before the sample point. Valid values for TSEG2 are 1..7
19..16	TSEG1	RW	2	Time segment before sample point There are (TSEG1+1) time quanta before the sample point. Valid values for TSEG1 are 2..15
15..8	BRP	RW	0	Baud rate prescaler For generating the bit time quanta the oscillator frequency (24 MHz) is divided by (BRP+1). Valid values for BRP are 1..255
7..6	SJW	RW	0	(Re)Synchronization jump width Adjust the bit time by (SJW+1) time quanta for resynchronization.
5..0	reserved	R	0	-

The bit timing register can only be written, if the configuration change enable bit (CCE) is set. To compensate phase shifts between clock oscillators of different CAN controllers, any CAN controller has to synchronize on any edge from recessive to dominant bus level. The edge is expected within the SyncSeg. If the actual position differs, the bit length is adjusted so that the edge is shifted back into the SyncSeg.

Figure 10-2 Bit Timing (shown for 1 Mbit/s)

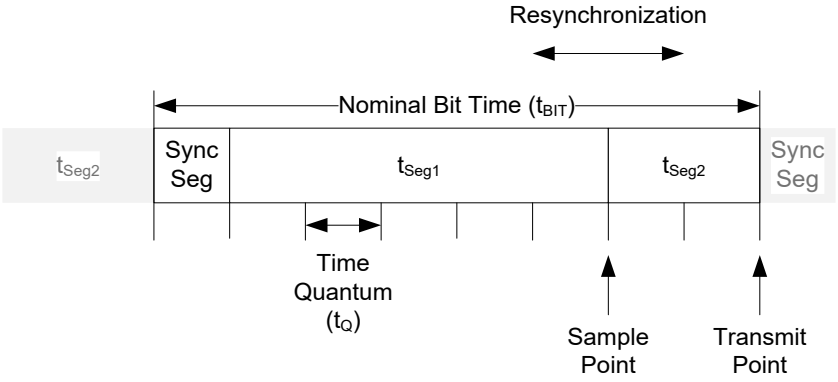


Table 10-6 Standard Bit Timing Parameter

Bit Rate [kbit/s]	Nominal Bit Time [µs]	Number of Time Quanta per Bit	Time Quantum Length [ns]	Location of Sample Point [%]	SJW	BRP	TSEG1	TSEG2	SAM
1000	1	8	125	75	1	2	4	1	0
800	1.25	10	125	80	1	2	6	1	0
500	2	16	125	87.5	1	2	12	1	0
400	2.5	10	250	80	1	5	6	1	0
250	4	16	250	87.5	1	5	12	1	0
125	8	16	500	87.5	1	11	12	1	0
100	10	16	625	87.5	1	14	12	1	0
50	20	16	1250	87.5	1	29	12	1	0
20	50	16	3125	87.5	1	74	12	1	0
10	100	16	6250	87.5	1	149	12	1	0

10.2.1.5 Global Mask Register

Offset: 0x008 (Global Mask 0)

Offset: 0x00C (Global Mask 1)

Table 10-7 Global Mask Register

Bit	Name	Type	Reset	Description
31..29	reserved	R	0	-
28..11	ID28..ID11	RW	0	Upper Global Mask Upper part of mask to filter incoming messages with extended identifier (29-bit).
10..0	ID10..ID0	RW	0	Lower Global Mask Lower part of mask to filter incoming messages with extended identifier (29-bit) or mask to filter incoming messages with standard identifier (11-bit).

Bits containing a ‘1’ in the mask register cause a comparison of the message identifier bits with the bits at arbitration register. On the other hand, zero entries mean "do not care", i.e. the bit at this position in the identifier of the frame is not compared with the corresponding bit in the arbitration register of the message object.

10.2.2 Message Object

Table 10-8 Message Object (overview)

Address	Byte Position			
	Byte 3	Byte 2	Byte 1	Byte 0
+0x0	Arbitration Register [31:0]			
+0x4	Data Byte 3 [31:24]	Data Byte 2 [23:16]	Data Byte 1 [15:8]	Data Byte 0 [7:0]
+0x8	Data Byte 7 [31:24]	Data Byte 6 [23:16]	Data Byte 5 [15:8]	Data Byte 4 [7:0]
+0xC	-		Message Configuration [15:8]	Message Control Register [7:0]

The message object is the primary means of communication between application and CAN module. Each of the 15 message objects consists of arbitration register (contains frame identifier), 8 data bytes, message control register and message configuration register. They can be either used as transmit or receive message objects. All objects have to be configured in front of clearing INIT.

The smaller the number of a message object, the higher its priority is. When assigning the identifiers to the different message objects, make sure that the first object always has the highest priority and the following are arranged in decreasing priority.

If a received message matches to more than one valid message object, it is stored in the first free message object with the lowest number (highest priority). If the CAN module stores a data frame, not only the data bytes but also the entire identifier and the data length code are stored. When storing a remote frame, only the Data Length Code is stored in the corresponding message object. The identifier remains unchanged.

For receiving messages with low priority, or objects with more than eight bytes, it is possible to concatenate several message objects to form a FIFO. The following settings must be configured in the message objects:

- all arbitration registers are set to the same values
- the same mask are used for the combined message objects
- RXIE of all objects which should be able to trigger an interrupt has to be set

The objects are filled in ascending order, i.e. the object with the lowest number is filled first.

A reconfiguration of message object during operation is possible. This is achieved by deactivation via MSGVAL. Also message objects combined to a FIFO can be created or reconfigured in this way. Therefore the following things have to be considered:

Different FIFO structures with the same configuration (identifier and message configuration) are not allowed. This could be happen after the reconfiguration of a message object in the centre of a FIFO structure. Since the acceptance filtering always takes place from the higher to the lower priority, objects of the second FIFO structure would never reached.

If a FIFO structure is modified, message objects may only be added or removed individually.

10.2.2.1 Arbitration Register

Offset: +0x0

Table 10-9 Arbitration Register

Bit	Name	Type	Reset	Description
31	MID_MASK	RW	0	Data byte mask select with Media ID Mask 0 = comparison w/o Media ID Mask 1 = acceptance filtering with use of Media ID Mask
30	GM1_MASK	RW	0	Message ID mask select with Global Mask 0 0 = comparison w/o Global Mask 0 1 = acceptance filtering with use of Global Mask 0
29	GMO_MASK	RW	0	Message ID mask select with Global Mask 1 0 = comparison w/o Global Mask 1 1 = acceptance filtering with use of Global Mask 1
28..11	ID28..ID11	RW	0	Upper Mask Upper part of mask to filter incoming messages with extended identifier (29-bit).
10..0	ID10..ID0	RW	0	Lower Mask Lower part of mask to filter incoming messages with extended identifier (29-bit) or mask to filter incoming messages with standard identifier (11-bit).

The arbitration registers are used for acceptance filtering of incoming messages and to define the identifier of outgoing messages. A received message is stored into the valid message object with a matching identifier and DIR=0 (data frame) or DIR=1 (remote frame). Extended frames can be stored only in message objects with XTD=1, standard frames only in message objects with XTD=0. If a received message matches with more than one valid message object, it is stored into the with the lowest message number.
For matching, the corresponding global mask and media identifier register are to be considered.

10.2.2.2 Message Control Register

Offset: +0xC

Table 10-10 Message Control Register

Bit	Name	Type	Reset	Description
7	RMTPND	RW	0	Remote pending (used for transmit-objects) Indicates that the transmission of this message object has been requested by a remote node, but the data has not yet been transmitted. When RMTPND is set, the CAN module also sets TXRQ. Both are cleared, when the message object has been successfully transmitted.
6	TXRQ	RW	0	Transmit request Indicates that the transmission of this message object is requested by the CPU or via remote frame and is not yet done. TXRQ can be disabled by CPUUPD.
5	MSGLST	RW	0	Message lost (this bit applies to receive objects only!) Indicates that the CAN module has stored a new message into this object, while NEWDAT was still set, i.e. the previously stored message is lost.
	CPUUPD			CPU update (this bit applies to transmit objects only!) Indicates that the corresponding message object may not be transmitted now. The user sets this bit in order to inhibit the transmission of a message that is currently updated, or to control the automatic response to remote requests.
4	NEWDAT	RW	0	New data Indicates, if new data has been written into the data portion of the message object by user (transmit objects) or CAN module (receive objects) since this bit was last reset, or not.
3	MSGVAL	RW	0	Message valid Indicates, if the corresponding message object is valid or not. The CAN module only operates on valid objects. Message objects can be tagged invalid, while they are changed, or if they are not used at all.
2	RXIE	RW	0	Receive interrupt enable Defines, if bit INTPND is set after successful reception of a frame.
1	TXIE	RW	0	Transmit interrupt enable Defines, if bit INTPND is set after successful transmission of a frame.
0	INTPND	RW	1	Interrupt pending Indicates, if this message object has generated an interrupt request (see TXIE and RXIE), since this bit was last reset by the user, or not.

When the user sets TXRQ of a receive message object, a remote frame will be generated instead of a data frame to request a remote node to send the corresponding data frame.

If several valid objects exist with pending transmission request, the message object with the lowest number is transmitted first.

To change the configuration of a message object during normal operation, the user first clears bit MSGVAL, which defines it as not valid. When the configuration is completed, MSGVAL is set again. But in case the message object is part of a FIFO structure, it can only be set to invalid at start or end of the FIFO structure. Regardless if it is the first or last object, the internal management of the FIFO structure is automatically adapted. If the MSGVAL bit is changed, the CAN module checks whether new FIFO structures have been set up.

10.2.2.3 Message Configuration Register

Offset: +0xD

Table 10-11 Message Configuration Register

Bit	Name	Type	Reset	Description
7..4	DLC	RW	0	Data length code Valid values for the data length are 0..8.
3	DIR	RW	0	Message direction 0 = receive 1 = transmit
2	XTD	RW	0	Extended Identifier 0 = standard 11-bit identifier 1 = extended 29-bit identifier
1	WOE	RW	0	Write-over Enable controls the ability of a new data frame to overwrite an existing message in the corresponding message object with DIR=receive 0 = overwrite not allowed 1 = overwrite allowed
0	MSGUPD	R	0	Message update 0 = message object isn't processed by CAN module 1 = CAN module has started to fill the message object with received data

10.2.3 Media Byte Arbitration

Media arbitration is an optional second arbitration performed by the following registers. Media arbitration compares the first two bytes of data field against media arbitration register. If media arbitration is configured, the media ID mask register is always used. Media arbitration takes place if the bit MID_MASK in the arbitration register is set.

10.2.3.1 Media ID Mask Register

Offset: 0x01E

Table 10-12 Media ID Mask Register

Bit	Name	Type	Reset	Description
15..8		RW	0	Media ID Mask Register 1 Mask to filter incoming messages.
7..0		RW	0	Media ID Mask Register 0 Mask to filter incoming messages.

10.2.3.2 Media Arbitration Register

Offset: 0x02E

Table 10-13 Media Arbitration Register

Bit	Name	Type	Reset	Description
15..8		RW	0	Media Arbitration Register 1 Compare value to filter incoming messages.
7..0		RW	0	Media Arbitration Register 0 Compare value to filter incoming messages.

10.2.4 Clock Synchronization

10.2.4.1 Send Interval Time

Offset: 0x03E

Table 10-14 Send Interval Time

Bit	Name	Type	Reset	Description
15..0		RW	0	Send interval in ms for message object 0

If the user initializes the Send Interval Time, this value is multiplied by the time base 1 ms to generate time intervals after which the message object 0 is sent automatically. The CAN module automatically sets TXRQ and NEWDAT.

10.2.4.2 System Clock Value

Offset: 0x100

Table 10-15 System Clock Value

Bit	Name	Type	Reset	Description
31..0		RW	0	Timestamp of system clock timer by receipt of a System Event object. Coded in µs.

10.2.4.3 Receive Delay Timer

Offset: 0x104

Table 10-16 Receive Delay Timer

Bit	Name	Type	Reset	Description
31..0		RW	0	Time in µs expired between receipt of the Time_Event and the invocation of that service primitve

10.3 Functional Description

10.3.1 Initialization

The software initialization is enabled by setting bit INIT in the control register. This can be done by the user via software, or automatically by the CAN module on a hardware reset. While INIT is set

- all message transfer from and to the CAN bus is stopped
- the CAN bus output CAN_TX is '1' (recessive)
- the counters of the EML are left unchanged

Setting bit CCE in addition, allows changing the configuration in the bit timing register. For initialization of the CAN module, the following actions are required:

- configuration of the Bit Timing Register (CCE required)
- setting of the Global Mask Register
- initialization of the Message Objects

If a message object is not needed, it is sufficient to clear its message valid bit (MSGVAL), i.e. to define it as not valid. Otherwise, the whole message object must be initialized.

After the initialization sequence has been completed, the user clears the INIT bit. Now the BSP synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (i.e. bus idle) before it can take part in bus activities and start message transfers.

The initialization of the message objects is independent of the state of INIT and can be done on the fly, the message objects should all be configured to particular identifiers or set to not valid before the module starts the message transfer, however.

To change the configuration of a message object during normal operation, the user first clears bit MSGVAL, which defines it as not valid. When the configuration is completed, MSGVAL is set again.

10.3.2 Handling of Message Objects

The following pictures shows the different communication cycles used by CAN.

Figure 10-3 Data Frame

Figure 10-4 Remote Frame with automatic Reply

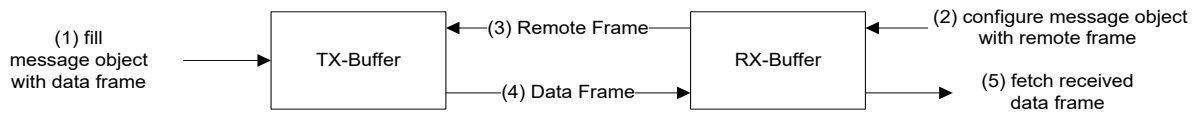
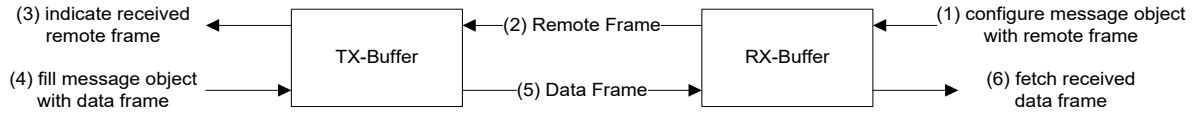


Figure 10-5 Remote Frame w/o automatic Reply



10.3.2.1 Transmit Message Objects

Transmit message objects used to transmit data frames, either generated by user or by receiving of a remote frame. The reply to a remote frame can be configured during initialization which makes possible an automatic reply. On the other hand it is also possible to fill the message object with the reply data after the receipt of a remote frame.

Table 10-17 Data Frame initiated by User (Requestor)

CAN Module	User
	check on TXRQ=0 and RMTPND=0 lock message object (CPUUPD:=1) indicate new data (NEWDAT:=1) write message object (data, length) set transmission request (TXRQ:=1) unlock message object (CPUUPD:=0)
check CAN on bus idle check on transmission request (TXRQ=1) check if message object unlocked (CPUUPD=0) reset indication of new data (NEWDAT:=0) generate data frame reset transmission request (TXRQ:=0) set interrupt flag (INTPND:=1)	
	reset interrupt (INTPND:=0)

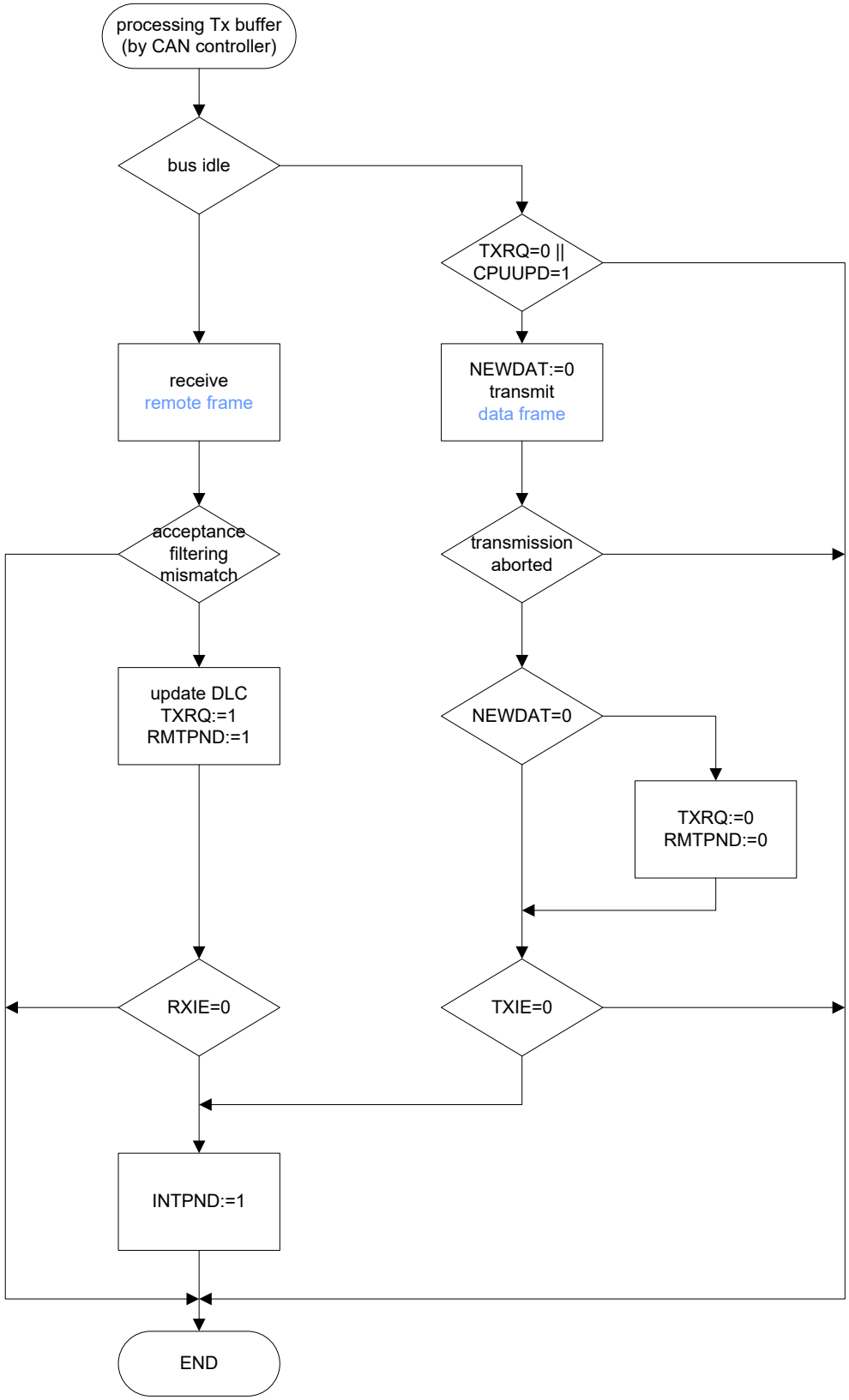
Table 10-18 Automatic Data Frame triggered by Remote Frame (Responder)

CAN Module	User
	check on TXRQ=0 and RMTPND=0 lock message object (CPUUPD:=1) indicate new data (NEWDAT:=1) write message object (data, length) unlock message object (CPUUPD:=0)
receive remote frame and acceptance filtering update DLC set transmission request (TXRQ:=1) indicate remote frame (RMTPND:=1) set interrupt flag (INTPND:=1)	
check on CAN bus idle check on transmission request (TXRQ=1) check if message object unlocked (CPUUPD=0) reset indication of new data (NEWDAT:=0) generate data frame reset transmission request (TXRQ:=0) reset remote frame indication (RMTPND:=0) set interrupt flag (INTPND:=1)	
	reset interrupt (INTPND:=0)

Table 10-19 Manual Data Frame triggered by Remote Frame (Responder)

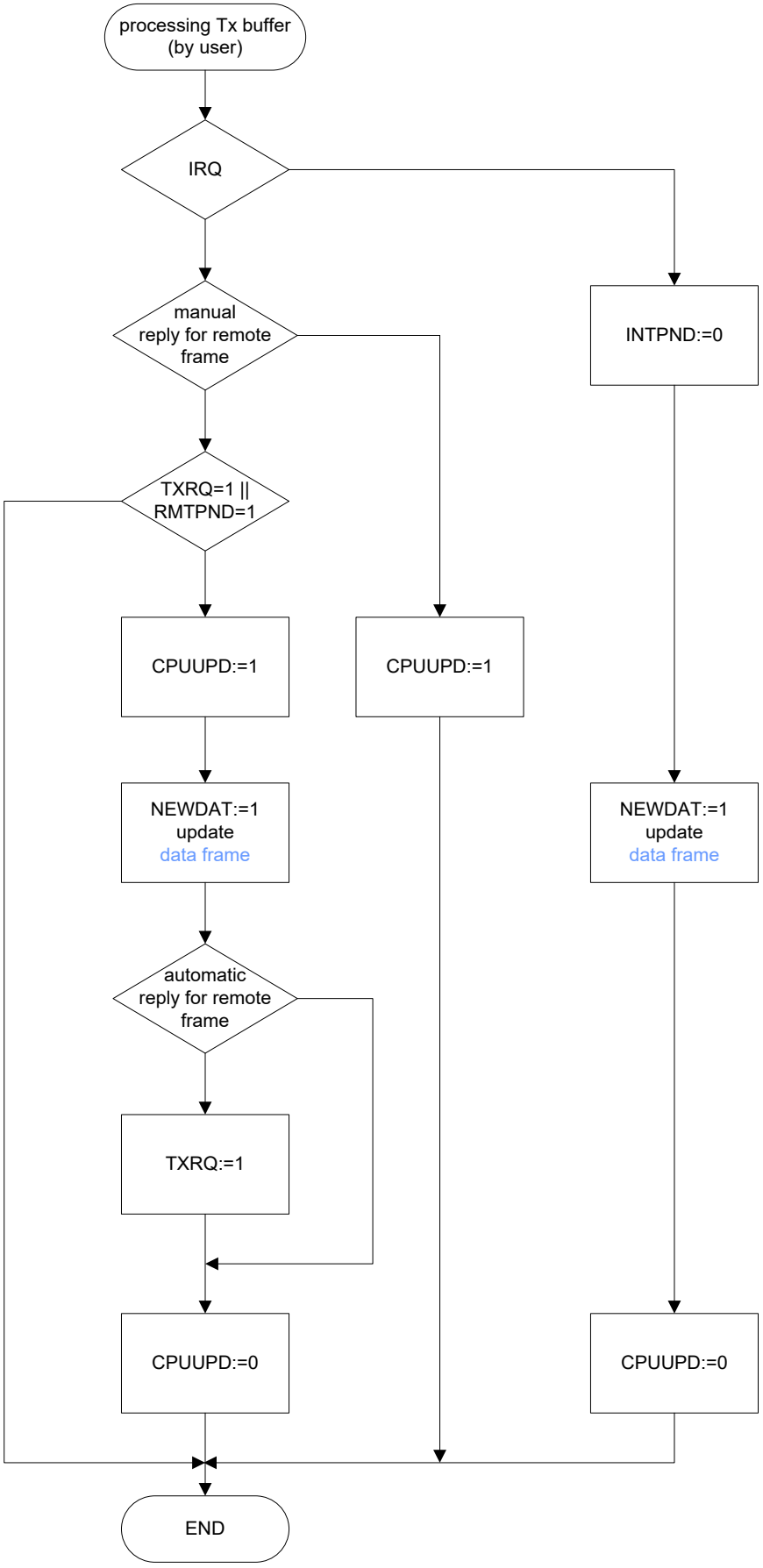
CAN Module	User
	prevent automatic reply: lock message object (CPUUPD:=1)
receive remote frame and acceptance filtering update DLC set transmission request (TXRQ:=1) indicate remote frame (RMTPND:=1) set interrupt flag (INTPND:=1)	
	reset interrupt (INTPND:=0) indicate new data (NEWDAT:=1) write message object (data, length) unlock message object (CPUUPD:=0)
check on CAN bus idle check on transmission request (TXRQ=1) check if message object is unlocked (CPUUPD=0) reset indication of new data (NEWDAT:=0) generate data frame reset transmission request (TXRQ:=0) reset remote frame indication (RMTPND:=0) set interrupt flag (INTPND:=1)	
	reset interrupt (INTPND:=0)

Figure 10-6 CAN Module Handling of Transmit Message Object



When the message has been successfully transmitted the CAN module clears the request bit (TXRQ) along with RMTPND, in case of an answer to a remote frame, if NEWDAT is not set again.

Figure 10-7 User Handling of Transmit Message Object



10.3.2.2 Receive Message Object

Receive Message Objects used to transmit remote frames and to receive data frames.

Table 10-20 Receive Data Frame (Responder)

CAN Module	User
receive data frame and acceptance filtering store frame in message object indicate new data (NEWDAT:=1) set interrupt flag (INTPND:=1)	
	reset interrupt (INTPND:=0) check on new data (NEWDAT=1) reset indication of new data (NEWDAT:=0) read message object double check on new data (NEWDAT=0) NEWDAT=1: read message object again

Table 10-21 Remote Frame (Requestor)

CAN Module	User
	set transmit request (TXRQ:=1)
wait on bus idle check on transmission request (TXRQ=1) check if message object is unlocked (CPUUPD=0) reset indication of new data (NEWDAT:=0) generate remote frame reset transmission request (TXRQ:=0) set interrupt flag (INTPND:=1)	
	reset interrupt (INTPND:=0)
receive data frame and acceptance filtering store frame in message object indicate new date (NEWDAT:=1) set interrupt flag (INTPND:=1)	
	reset interrupt (INTPND:=0) check on new data (NEWDAT=1) reset indication of new data (NEWDAT:=0) read message object double check on new data (NEWDAT=0) NEWDAT=1: read message object again

Figure 10-8 CAN Module Handling of Receive Message Object

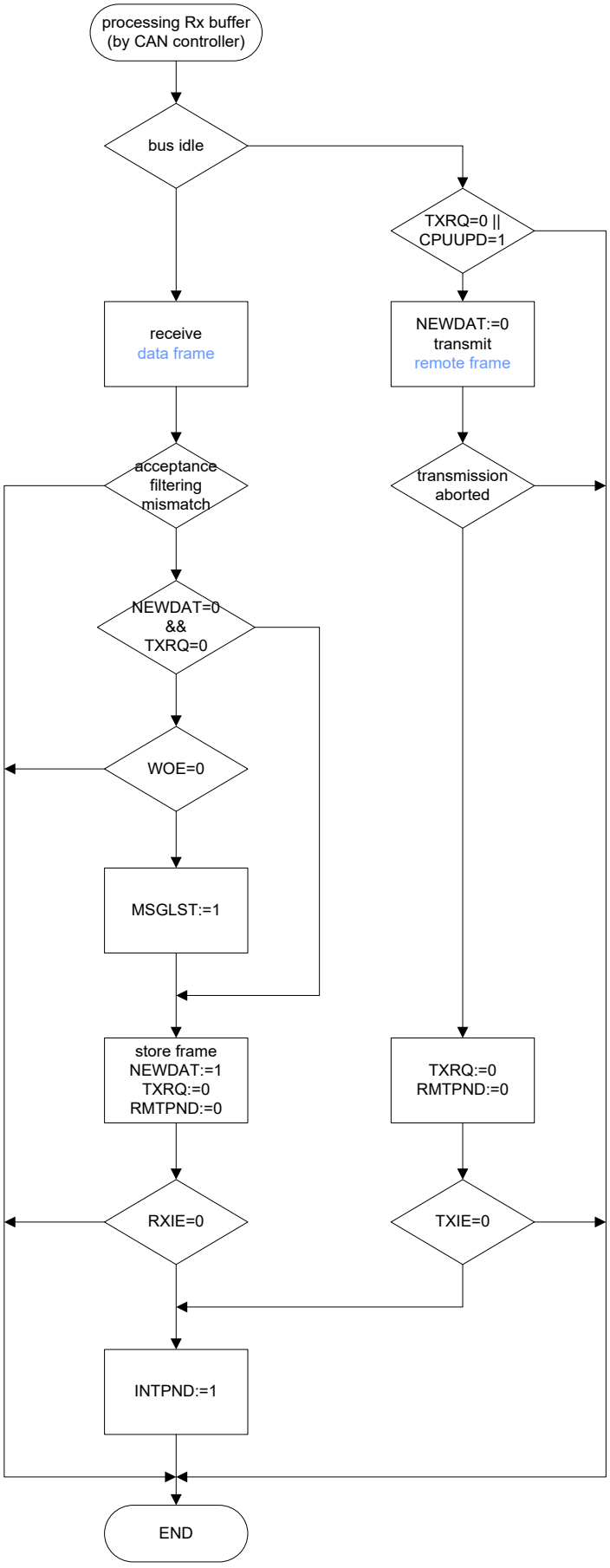
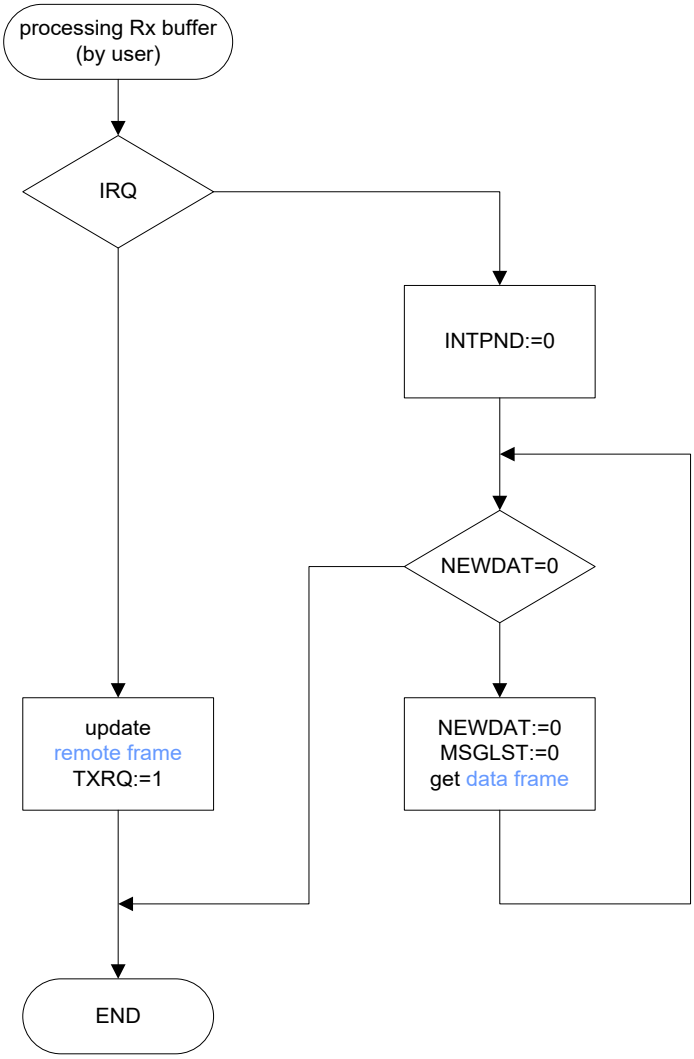


Figure 10-9 User Handling of Receive Message Object



When the CAN module writes new data into the message object, NEWDAT will be set. The user will clear this bit before the data is read, and verify that the bit is still cleared once it has finished working. This ensures that consistent data is being worked on, and are not parts of different frames.

10.3.3 Interrupt Handling

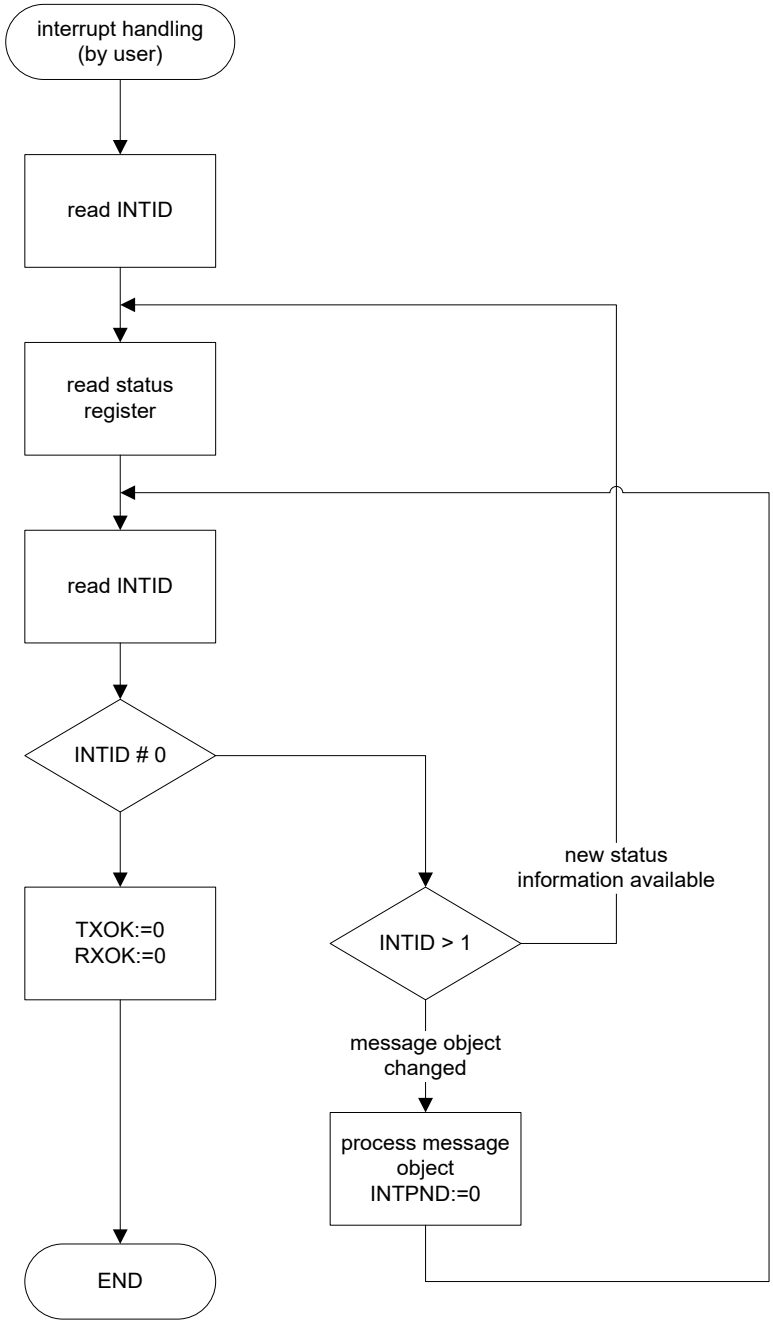
Interrupts are triggered by transmission or reception of a frame, and also if a status change occurs.

Message objects of the same FIFO structure always have the same priority, i.e. the prioritization by message object order is cancelled. The user is responsible for the management of the receive data in FIFO structures himself, and therefore have to use NEWDAT to determine in which message object the oldest data is stored. By means of RXIE it is possible to configure which message object triggers an interrupt.

Table 10-22 Interrupt handling

CAN Module	User
receive frame in message object 0 set interrupt flag (INTPND:=1) update status register (RXOK:=1) SIE=1: set interrupt ID (INTID:=1) and interrupt signal	
	read interrupt register read status register
update interrupt ID (INTID:=2) by read access	
	read interrupt register again read message object 0
receive frame in message object 1 set interrupt flag (INTPND:=1) update status register (RXOK:=1) SIE=1: set interrupt ID (INTID:=1) and interrupt signal	
	reset interrupt of message object 0 (INTPND:=0) read interrupt register read status register
update interrupt ID (INTID:=3) by read access	
	read interrupt register again read message object 1 reset interrupt of message object 1 (INTPND:=1)
update interrupt ID (INTID:=0)	
	read interrupt register reset status register (RXOK:=0)

Figure 10-10 User Handling of interrupt



10.3.4 Clock Synchronization

The clock synchronization is used to set a uniform system time for a CAN network. This system time can be used to order various events.

Figure 10-11 Clock Synchronization Mechanism

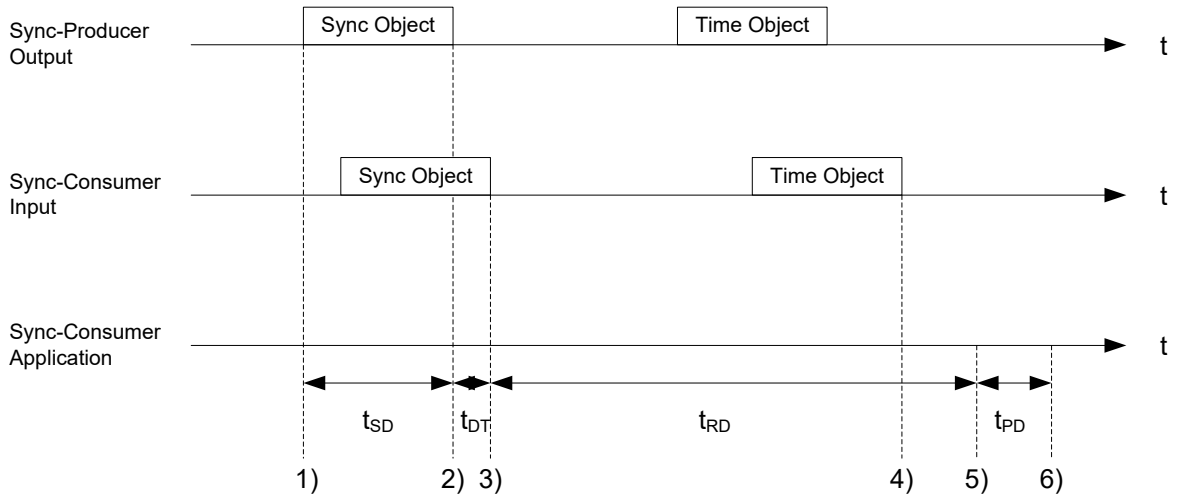


Table 10-23 Clock Synchronization Events and Intervals

Parameter		Symbol	Description
producer	Time_Event	1)	event caused transmission of Sync_Object frame
	sync interrupt	2)	interrupt caused by transmission of a Sync_Object frame
consumer	sync interrupt	3)	interrupt caused by reception of a Sync_Object frame
	Time_Object interrupt	4)	interrupt caused by reception of a Time_Object frame
	read access to timer	5)	receive delay timer is read out
	update system clock	6)	system clock (application) is updated
Send Delay Time		t_{SD}	length of Sync Object frame
Sync Delay Time		t_{DT}	time delay caused by transmission
Receive Delay Time		t_{RD}	interval between start of Receive Delay Timer and read access to this timer
Process Delay Time		t_{PD}	process time between read access to Receive Delay Timer and update of system clock

The synchronization mechanism consists of two frames. The first frame (Sync_Object) informs the consumer of the time measurement in the producer. With receipt of this frame, the CAN module of consumer starts a timer to measure the elapsed time (t_{RD}: Receive Delay Time) since this event. The second frame (Time_Object) contains the time at which the event occurs (determined by the send interval timer), plus the time to transmit the Sync_Object (t_{SD}: Send Delay Time). The added delay also includes a possible waiting time if the CAN bus is not idle when the time event occurs. Another interrupt at consumer side is triggered by reception of the Time_Object. However, the receive delay timer continues to run until read access to the counter value. This allows the synchronization of the time as precisely as possible. The transmission delay on the bus (t_{DT}: Sync Delay Time) is previously communicated to the sync consumer (the signal propagation delay is 5 ns/m on the CAN bus).

The CAN module at the consumer accomplishes only the task of frame synchronization. The system time is managed by the user application, which also has to handle the delay between read out of receive delay timer and update of system time. The system time is composed as follows:

$$system\ time = Time\_Event + t_{DT} + t_{RD} + t_{PD}$$

The used frames for clock synchronization are:

- Sync_Object: standard identifier 0x80 and no data contained
- Time_Object: data part contains current time (time event + send delay time) and a consecutive number for each time event

Table 10-24 Coding of Time Object Frame

Byte	Description	Value Range
0 .. 3	Time_Event + t_{SD}	0 .. 2^{31} µs
4	Time_Object_Number	0 .. 255

Table 10-25 Handling Sync Producer

CAN Module	User
	set system clock value set send interval time start send interval timer (START_TIME:=1)
send interval timer expires transmit Sync_Object frame update system clock value set interrupt (INTPND:=1)	
	reset interrupt (INTPND:=0) read system clock value update message object for Time_Object set transmission request
transmit Time_Object frame set interrupt (INTPND:=1)	
	reset interrupt (INTPND:=0)

Table 10-26 Handling Sync Consumer

CAN Module	User
	enable receive delay timer (RXDLY:=1)
receive Sync_Object frame set interrupt (INTPND:=1) start receive delay timer	
	reset interrupt (INTPND:=0)
receive Time_Object frame set interrupt (INTPND:=1)	
	reset interrupt (INTPND:=0) read Time_Object message object read receive delay timer
stop receive delay timer	

11 NAND Flash Controller

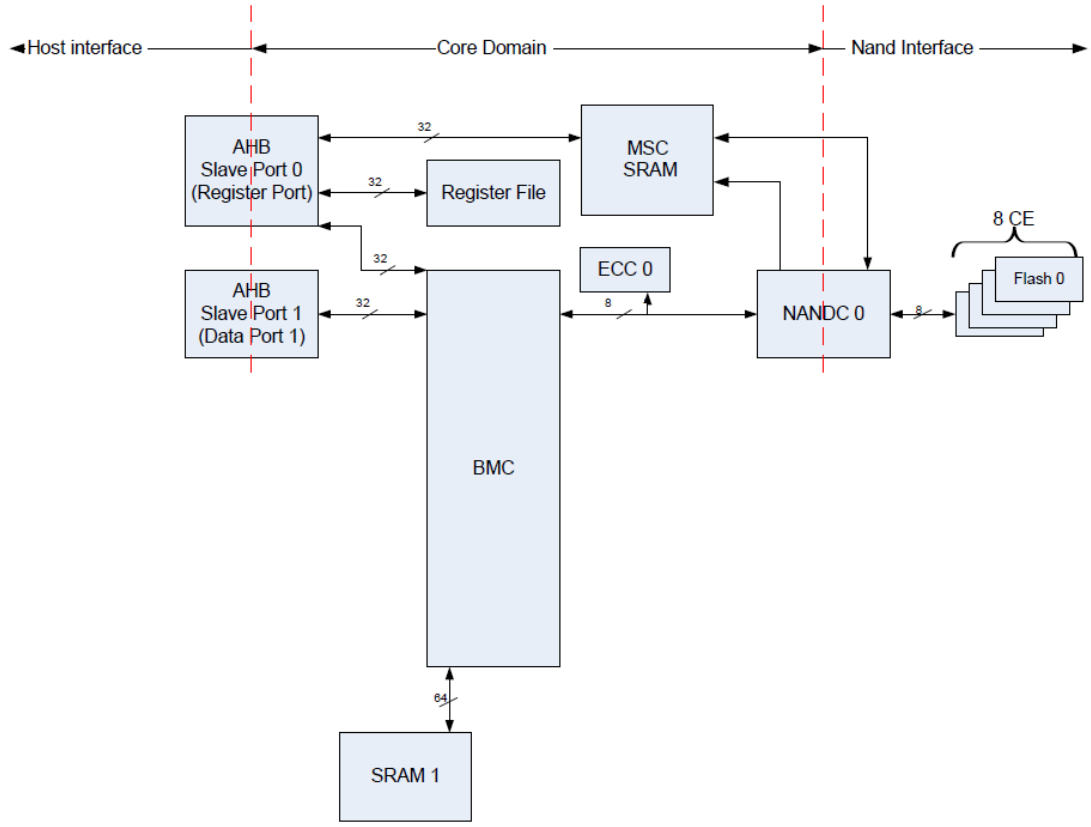
Offsets: Register Port: 0xB8000000
Data Port: 0xF1000000

The ANTAIOS includes a NAND Flash controller with AHB interface. This controller supports one NAND channel and the data bus of each NAND channel is 8–bit wide. This controller also supports an AHB data slave ports for accessing the data buffer. The registers can be accessed through a standalone AHB slave. The Buffer Management Unit (BMC) is used to control the data storage space for data from the AHB data slave port to NANDC or from NANDC to the AHB data slave port. If an error occurs after the parity check, the ECC engine will be responsible for the ECC parity generation, ECC parity check, and data correction. The MicroCode SRAM Controller (MSC) module provides a dedicated space for storing the programming flow, which can be read from NANDC to execute flash flow or written from register port.

- Supports page sizes of 512, 2K, 4K, 8K, and 16K bytes for NAND Flash
- Supports 8-bit data bus for NAND Flash
- Supports BI Reserved function
- Supports programmable timing parameters for NAND Flash
- Supports programmable command flow for different NAND Flash flow
- Supports 16 bits of ECC correction capability for spare region
- Supports AHB 32 bits data width
- Supports 16 bits for ECC correction capability of a 512-byte or 1K-byte sector
- Supports DMA handshake mode
- Supports scramble function
- Supports data inverse function
- Supports valid page of block function
- Provides AHB data slave port to support RETRY response without exploiting DMA handshake mode
- Supports flexible user-defined data length
- Supports warm-up function
- Supports ONFi 2.2
- Supports SLC, MLC, and TLC Flash

11.1 Overview

Figure 11-1 NAND Controller



11.1.1 AHB Data Port

The NAND-Controller comprises one data port with a memory space ranging from 512 bytes to 64k.

Note: The BMC region acts as FIFO in the DMA mode and there is no correlation between the address on the AHB bus and the data location in the BMC region.

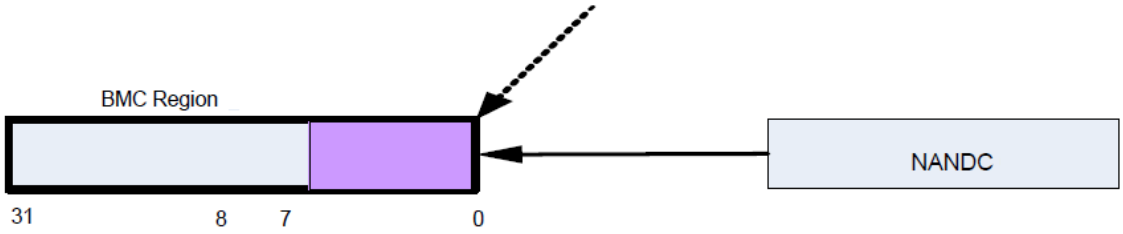
The AHB data port supports the AHB retry protocol. When “ahb_retry_en” is set, the AHB data port will retry if the selected BMC region is not available for accessing.

11.1.2 Buffer Management Controller

The Buffer Management Controller (BMC) is in charge of controlling data accessed from NANDC or the AHB data slave port.
The “bmc_region_sel” field in the 4th word of a NANDC command decides the region to be accessed during processing a NANDC command.
Setting the “bmc_ignore” bit in the 4th word of a NANDC command to ‘1’ indicates that this command acts in the user mode. Read/Write from NANDC will not affect the region status

used in the normal mode when operating in the user mode. The user-mode pointer can be adjusted by writing “User Mode Pointer Adjustment Register”.
Figure 11-2 shows a user-mode read operation from the NANDC channel to region. Please note that the data will be ready in the region only after the NAND command is completed when ECC is enabled.

Figure 11-2 User Mode Operation



Step1: Set the User Mode Pointer Adjustment Register (0x4040 as 0)
Step2: Push the command into the Command Queue0 (Set the ignore bit as 1 and select region as target)

After the above two steps, NANDC reads data from the flash and writes data to region from the first grid

When operating in the normal mode (That is, the NANDC command with the “bmc_ignore” bit is set to ‘0’), the region will act as FIFO. For a Flash write operation, the data will be pushed from the AHB data slave port and NANDC will pop the data from the designated region. For a Flash read operation, the data will be pushed from NANDC and will be popped by the AHB data slave port after the ECC checking and correction.
When a host issues two commands, one is in the normal mode another is in the ignore mode, the host must make sure that the normal mode command complete includes the data transfer. Afterwards, the host can issue the ignore command. Otherwise, the data in the normal mode will be destroyed when the host will execute the ignore command.
The DMA handshake protocol is supported in the NAND-Controller. When a NANDC command is issued with “cmd_hsk_mode” (In the 6th word of a NANDC command) as ‘1’, the dma_req signal will be asserted under the following conditions:

- For a write operation, the dma_req signal will be asserted when at least a space of 512/1024 bytes is available in a region.
- For a read operation, the dma_req signal will be asserted when at least one sector can be read by the AHB data slave port. The BMC region has one dma_req/dma_ack I/O. The transfer length of one pair of dma_req/dma_ack (DMA burst size) is 512/1024 bytes of data.

11.1.3 AHB Register Port

NAND-Controller comprises one AHB register port. This port can be used to access the following items:

- Register of the NAND-Controller
- Command queue

- Spare register
- Programmable OPCODE register
- MSC SRAM
- Data SRAM

Please note that the command queue can only be accessed with WORD as HSIZE.

11.1.4 ECC Correction Error Handling

When an ECC correction fail event is encountered, the ECC correction fail interrupt status can be checked by reading the register offset 0x0024. Consequently, the ECC correction failed channel will be halted, that is, the NAND channel controller will stop further operations. However, the BMC region may still contain the correct sectors that can be read out. Once the correct sectors in the BMC region are all read out, the “BMC region halt” and “region_buf_empty” statuses will be set to ‘1’ and can be checked by reading the offset 0x0400.
Please keep the following procedure in mind to deal with the ECC correction fail event:

1. Check the “ECC Interrupt Status Register” (Offset = 0x0024) to decide the channel in which the ECC correction fails.
2. Poll “BMC Region Status Register” (Offset = 0x0400) to make sure that the correct data in the designated region are all read out.
3. Check “Region x Remaining Sector Count of Read Data” (Offset = 0x0430). With the values obtained in this register and row address in the command queue, the precise page location of the ECC un-correctable error can be calculated.
4. Reset the designated BMC region (The external DMA must be aborted before reset the BMC region reset)
5. Reset the AHB Slave (If the AHB Slave port is asynchronous.) and wait for the AHB reset to be cleared
6. Flush the corresponded command queue (This step is optional. If the command queue is not flushed, NANDC will repeat the current command after reset.)
7. Poll “0x100” to check if the target channel is at the ready state.
8. Reset the NANDC (This step can be skipped if Step 6 is executed.)
9. Poll the NANDC software reset register until it returns to ‘0’.

11.1.5 Auto Compare Error Handling

When the auto compare fail event occurs at manipulating blanking check, command complete will not be asserted, and F/W read register (0x0154) will find out auto compare fail. The host controller will hang on and F/W must perform the following sequence.

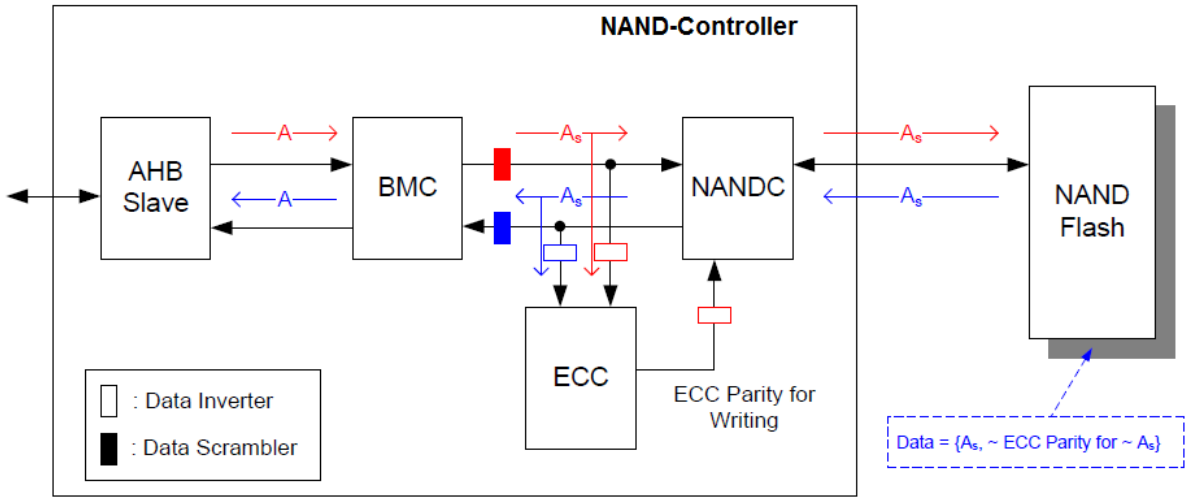
1. Check if the auto_cmp_fail (0x0154) status occurred
2. Flush the command queue (Including the reset controller and ECC engine)
3. Poll the NANDC software reset register until it returns to ‘0’

11.1.6 Auto Compare Check Mechanism

If the ECC correction bit is 6, the ECC parity bit will be 6 * 14/8 = 10.5 bytes and the auto-compare check mechanism will check total 11 bytes, not 10.5 bytes only. This mechanism is suitable for both the spare and data region ECC protection.

11.1.7 Scramble and Data Inverter

Figure 11-3 Scramble and data inverter



When reading data from the erase block region at scramble and data inverter enable, ECC will not fail and the BMC read data will not be FF. The returned data will be 0xFF through the descrambler. Both data region and spare region can be applied to the scrambler and data inverter function shown in Figure 11-3.

11.1.8 Performance

BMC can only offer a throughput of 384MB/s for NANDC channels at 96 MHz AHB_CLK (96MHz).

11.1.9 Abort Sequence

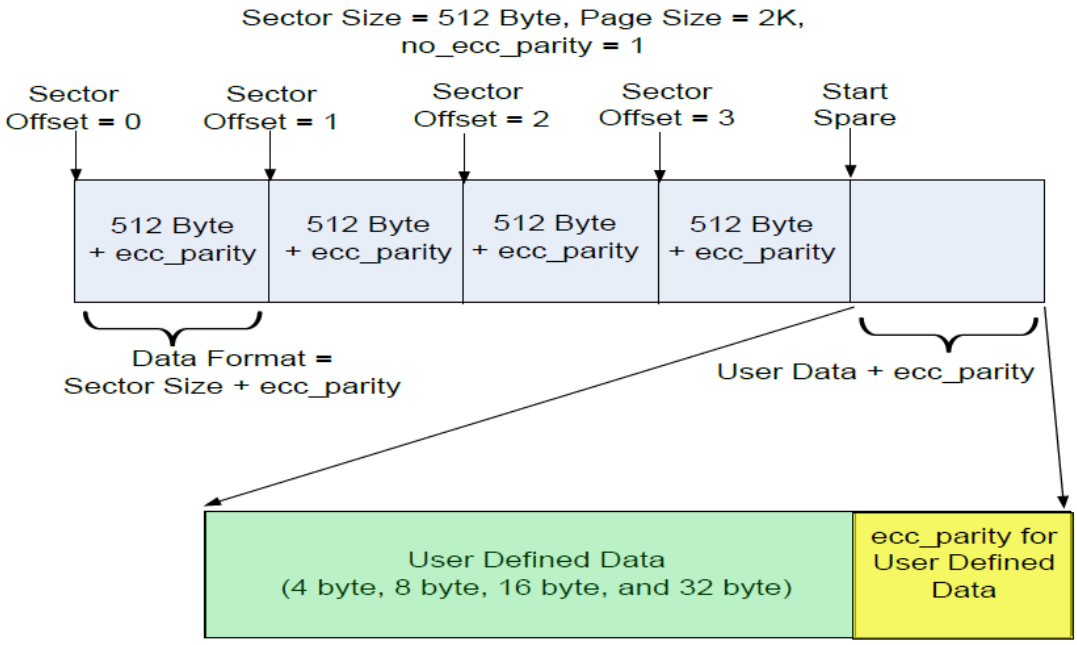
When the host controller wants to abort this command, please follow the sequence below to guarantee that the flow is correct.

1. Flush the command queue
2. Poll the NANDC software to ‘0’
3. Reset the BMC region
4. Issue the “RESET FLASH” command before issuing a new command

11.1.10 Host Controller Data Format

Figure 11-4 depicts the data format of the host controller.

Figure 11-4 Data Format at no_ecc_parity = '0'



Legacy Flash Operation

The ECC parity is combined with the Flash data and the spare data and ecc_parity are placed after the data format to reduce the random data access time in order to improve the performance.

High-speed Flash Operation

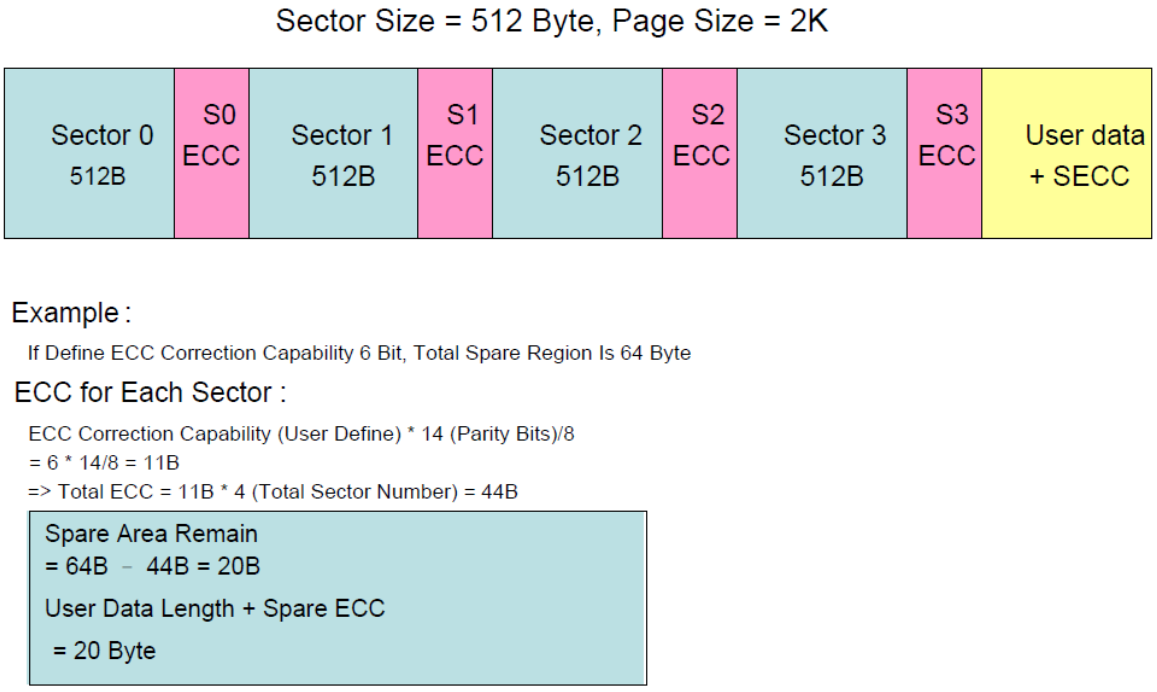
If the number of ecc_parity is programmed by F/W to be an odd number, the data strobe must be even, as specified in the high-speed Flash specification. Therefore, ecc_parity will be automatically added by '1' by using the host controller that meets the even number of the high-speed Flash. Thus, the spare data used will be compressed. For example, if the page size is 4K, sector size is 512 bytes, spare data size is 128 bytes, and ECC correct is 6 bits (ecc_parity is 11 bytes); 4K/512 byte = 8 sectors, the remaining spare data size will be 128 - (8 * (11 + 1)) = 32 bytes at the high-speed Flash operation. However, for the legacy Flash operation, the remaining spare data size will be 128 - (11 * 8) = 40 bytes.

11.1.10.1 Spare Register Organization

There are 32 bytes spare data available.

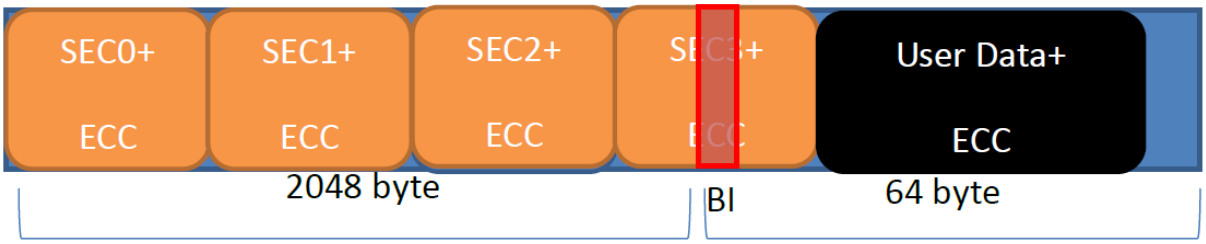
11.1.10.2 User Data and ECC usages

Figure 11-5 User data and ECC usages



11.1.10.3 BI Reserved Function

Figure 11-6 BI area



For example, if page size = 2K and spare size = 64 byte, the BI address will be 2048.
NAND controller will skip BI address (1 byte for legacy), (2 bytes for the ddr mode) to avoid BI information overwriting.
Users can use byte read/write to get the BI information (No ECC).
Users can use spare read/write to access the user data (W/ ECC).
Users can copy the BI information and write to the user data region to ECC protect.

11.1.11 Description of NANDC MicroCode

Table 11-1 Summary of MicroCode

MicroCode	Bit[7:0]	Operation comment
Opcode	“00?_?????”	bit[5:0]means different opcode command, at most 64 opcode
Zero_addr	“010_00000”	issue address with all zero value, cycle number is setting by col_cyc
1st_row_addr	“010_00001”	1st_row address cycles, cycle number is setting by row_cyc
1st_col_addr	“010_00010”	1st_col address cycles is setting by col_cyc, and column address is generated by hardware base on 1st_sec_offset

MicroCode	Bit[7:0]	Operation comment
2nd_row_addr	“010_00011”	2nd_row address cycles is setting by row_cyc
2nd_col_addr	“010_00100”	2nd_col address cycles is setting by col_cyc, and column address is generated by hardware base on 2nd_sec_offset
Tog_1st_addr	“010_00101”	toggle 1st_row_addr, and toggle address cycles is setting by row_cyc. toggle bit depends on user setting
Tog_2nd_addr	“010_00110”	toggle 2nd_row_addr, and toggle address cycles is setting by row_cyc. toggle bit depends on user setting
Sp_col_addr	“010_00111”	spare column address for flash random opcode depends on 1st_sec_offset. col_cyc means source sp_col_addr cycles.
Fix_1st_addr	“010_01000”	fix start 1st_row_addr, row_cyc is fix addr cycles.
Fix_2nd_addr	“010_01001”	fix start 2nd_row_addr row_cyc is fix addr cycles.
Buffer1	“011_00000”	buffer1: max{tADL,tCCS}~60ns
Buffer2	“011_00001”	buffer2: max{tAR,tRR,tCLR}~20ns
Buffer3	“011_00010”	buffer3: max{tRHW,tRHZ}~100ns
Buffer4	“011_00011”	buffer4: max{tWHR}~60ns
Write data	“011_00100”	write data state including data+ecc parity
Read data	“011_00101”	read data state including data+ecc parity
Busy	“011_00110”	busy state means wait busy ready by hardware
End	“011_00111”	control flow end state
Write spare	“011_01000”	write spare state
Read spare	“011_01001”	read spare state
Read status	“011_01010”	read status once
Multi read status	“011_01011”	read status until busy/ready OK
Blank write data	“011_01100”	write whole page data for blanking check using auto compare pattern
Blank read data	“011_01101”	read whole page data for blanking check using auto compare pattern
Inc_ce	“011_01110”	inc ce state means increase ce
Dec_ce	“011_01111”	dec ce state means decrease ce
Return	“011_10001”	Return to Main Routine from subroutine
Chk_rdy	“011_10010”	only check flash ready bit
Link	“011_10011”	Link 2 or more commands become one entire flash command
Goto_fix	“100_?????”	Goto fix flow subroutine. Following a microcode [7:0] means goto subroutine index
Goto_pro	“101_?????”	Goto programming flow subroutine. Following a microcode [7:0] means goto subroutine index
goback	“11?_?????”	goback state means go back which index of control flow table. bit[5:0] means go back index number

11.1.11.1 MicroCode Description

- OPCODE
  Users can use the OPCODE index to choose the proper opcode based on Table 11-2.

Table 11-2 OPCODE

Index	OPCODE	Index	OPCODE	Index	OPCODE	Index	OPCODE
0	0x00	16	0x3A	32	0x8A	48	0xEE

Index	OPCODE	Index	OPCODE	Index	OPCODE	Index	OPCODE	
1	0x01	17	0x3C	33	0x8B	49	0xEF	
2	0x02	18	0x3F	34	0x8C	50	0xF1	
3	0x03	19	0x50	35	0x90	51	0xF2	
4	0x05	20	0x55	36	0xA2	52	0xFA	
5	0x06	21	0x5C	37	0xC0	53	0xFC	
6	0x09	22	0x5D	38	0xC5	54	0xFF	
7	0x0D	23	0x60	39	0xD0	55	Programming Opcode	Byte0
8	0x10	24	0x69	40	0xD1	56		Byte1
9	0x11	25	0x70	41	0xDA	57		Byte2
10	0x15	26	0x71	42	0xDF	58		Byte3
11	0x1A	27	0x78	43	0xE0	59		Byte4
12	0x30	28	0x7B	44	0xE1	60		Byte5
13	0x31	29	0x80	45	0xE2	61		Byte6
14	0x32	30	0x81	46	0xEC	62		Byte7
15	0x35	31	0x85	47	0xED	63		

For example, if users want to issue the opcode of 0x10, which the index is 8, to fill “000_01000”.

- zero_addr:
  When users fill zero address in a flow, it indicates that the zero address will be issued to Flash.
- 1st_row_addr:
  When users fill the first row address in a flow, it indicates that the host will issue the first-row address from the command table. After the “goback”-state, 1st_row_addr will be increased using the page or block offset by programming the setting.
- 1st_col_addr:
  When it is not in the byte mode, the host will issue the first column address according to the first sector offset in the command table. For example, if the first sector offset is ‘1’, the sector size will be 512 bytes and the ECC parity will be 3 bytes. Then, the first column address can be calculated as (512 + 3). When it is in the byte mode, the first column address will be concatenated with the second offset and the first offset will become a 2-byte address, of which the address can be set to any value by users. If the host operates longer than the second time of 1st_col_addr in a flow, 1st_col_addr will become ‘0’.
- 2nd_row_addr:
  When users fill the second row address in a flow, it indicates that the host will issue the second row address from the command table. After the “goback”-state, 2nd_row_addr will be increased using a page or block offset by programming the settings.
- toggle_1st_addr:
  The host will issue an address to toggle the specified addresses. When 1st_row_addr = 0x0000_0000_0000, and toggle bit is 7, toggle_1st_addr will be 0x0000_0000_0080.
- toggle_2nd_addr:

The host will issue an address to toggle the specified addresses. When 2nd_row_addr = 0x0000_0000_0000, and toggle bit is 6, toggle_2nd_addr will be 0x0000_0000_0040.

- sp_col_addr:
  sp_col_addr = (sector_size + ecc_parity) * (page_size/sector_size).
- fix_1st_addr:
  The host will issue the fixed specific 1st_row_addr.
- fix_2nd_addr:
  The host will issue the fixed specific 2nd_row_addr.
- var_addr1, var_addr2, var_addr3, var_addr4:
  Variable byte addresses. It indicates variable address register (0x1D0) var_B1, var_B2, var_B3 and var_B4.
- Buffer1:
  The buffer1 state issued by the host indicates that Flash will delay the AC timing by using the AC timing programming register.
- Buffer2:
  The buffer2 state issued by the host indicates that Flash will delay the AC timing by using the AC timing programming register.
- Buffer3:
  The buffer3 state issued by the host indicates that Flash will delay the AC timing by using the AC timing programming register.
- Buffer4:
  The buffer4 state issued by the host indicates that Flash will delay the C timing by using the AC timing programming register.
- write data:
  The host will perform the write data strobe and ECC parity strobe until the page boundary or sector counter is reached.
- read data:
  The host will perform the read data strobe and ECC parity strobe until the page boundary or sector counter is reached.
- busy:
  The host will wait for the Flash busy/ready until busy/ready is set to high.
- end:
  The host issues the end state to perform the end command of the Flash and the controller will finish this command.
- write spare:
  The host will issue proper write spare length according to the sector counter until the page boundary or counter is reached.
- read spare:
  The host will issue proper read spare length according to the sector counter until the page boundary or counter is reached.
- multi-read status:

The host will issue re_n to poll the Flash status until this status is ready and the check status passes/fails and report its status.

- read status :
  The host will issue one re_n and check the status regardless Flash is ready or busy and check status fail/pass and report status.
- blank write data:
  The host will issue the blank write data strobe in a blanking flow and its length is the total data format including the data ECC parity and spare ECC parity.
- blank read data:
  The host will issue the blank read data strobe in a blanking flow and its length is the total data format including the data ECC parity and spare ECC parity.
- inc_ce:
  This state simply increases one CE.
- dec_ce:
  This state simply decreases one CE.
- Return:
  From subroutine return to the main flow
- Chk_rdy:
  Only continuous checks Flash ready bit till flash is ready and not check command fail/pass and not report status.
- LINK:
  Link 2 or more command flows become one entire Flash command. Link will be similar to the end state only when ce_n is active low at the link state and ce_n is active high at the end state. One flash entire command must be finished by end not by link. As a result, the valid command flow will be LINK-END, LINK-LINK-END, or LINK-LINK-...-END.
- goto_fix:
  Go to fixed flow sub-routine from main flow. The following microcode [7:0] is sub-routine index.
- goto_pro:
  Go to programming flow sub-routine from main flow. The following microcode [7:0] is sub-routine index.
- goback:
  The “goback”-state will continually perform a flow if the total data performed exceed one page. Bit 5 to Bit 0 are the go back index numbers. The maximum go back index is 63.

Note:

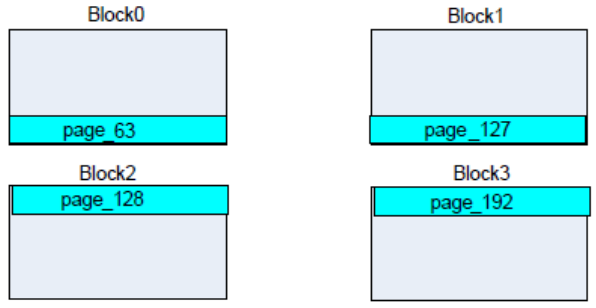
The “goback”-state is “11_??????”. Bit 5 to Bit 0 represent the go back index number. If one flow includes a 10-microcode state and the first state is opcode, the second state will be the 1st_col address and the 9th state will be go back. The state will go to the “goback”-state with index = 8 and the current go back index = 9. Consequently, go back to 9 - 8 = 1, the next state is the opcode state

11.1.11.2 Flow Rules

The fixed flow and programming flow are comprised by Microcode, as indicated in Table 11-1. The following rules must be followed.

- In the byte mode, only the read/write spare state can be used. The “goback”-state cannot be used in a flow and the sector counter can be ignored. spare_num is programmed from 0 to 31 (1 byte .. 32 bytes).
- The row/column address is decided by the flow state. When the state goes to a row/column state, the row/column address will be issued.
- In the byte mode, the first column address is decided by concatenating 1st_sec_offset and 2nd_sec_offset. 2nd_sec_offset is the high byte. When it is not in the byte mode, 1st_sec_offset will be the sector offsets. They can be programmed from 0 to 31. The first column address will be decided.
- The sector format of the read/write data state is (Data + ecc_parity); however, ecc_parity will always appear regardless ecc_en is enabled and no_ecc_parity is disabled. If no_ecc_parity is ‘1’ and ecc_en is ‘0’, ecc_parity will disappear in the data format.
- “intr_en” means that users must clear the command complete interrupt when a command finishes. The controller will then issue the next command if the command queue is not empty.
- The address will not be automatically generated across the chip boundary. This operation should be performed by users.
- The command queue will pop when a command finishes normally. However, when auto-compare fails or ECC fails, the command queue will not pop.
- The 2-plane commands only operate in one block. For the next block, users must re-issue a new command.
    - Example: As shown in Figure 11-7, block size = 64 pages. If users issue a 2-plane page write command to write four pages from page 63, this command will not be finished correctly by the host controller. Users must issue two commands; the 2-plane page writes to write two pages from page 63. Another command is from page 128.

Figure 11-7 2-plane write 4 pages cross block



- The row address can be increased by page or block after the “goback”-state.
- The cache page read/write must not cross the blocks.
- One flow does not include two “goback”-states.

- Each flow must have an end/link state. If a flow is finished with link, which is not the entire flash command, it must be finished with the end state.
- The programming flow cannot continuously include two busy states.
- The return state is returned to the main flow and the return state must be the last state of the subroutine.
- The go to state goes to the subroutine flow. The index of the go to state has at most 63. The subroutine must not be programmed more than 63 indexes. Subroutine is only allocated with fixed flow.
- The first microcode of flow must not be read data or write data.
- The microcode of the flow next to the read data or write data must not be read data or write data.
- The microcode of the flow next to the read data or write data must not be read spare or write spare.

11.1.11.3 Command Register Setting for Fixed Flow Command1

Before executing a command, it is recommended referring to the command register setting (As shown in Table 11-3) to set the counter.

Table 11-3 Command Register Setting

Command Register Setting	cmd index	bmc ignore	byte mode	cmd_ hsk mode	1st offest	2nd offset	counter unit	update spare
PAGE READ	0x1C	V	0	V	V	dc	sector	X
PAGE WRITE WITH SPARE	0x26	V	0	V	V	dc	sector	V
SPARE WRITE	0x33	dc	0	0	V	dc	page	V
SPARE READ	0x3E	dc	0	0	V	dc	page	X
PAGE READ WITH SPARE	0x48	V	0	V	V	dc	sector	X
PAGE WRITE	0x54	V	0	V	V	dc	sector	X
READ ID (byte mode)	0x5F	dc	1	0	V	V	dc	X
RESET	0x65	dc	0	0	dc	dc	dc	X
BLOCK ERASE	0x68	dc	0	0	dc	dc	block	X
INTERNAL COPY BACK	0x70	dc	0	0	V	dc	page	V
BYTE WRITE (byte mode)	0x80	dc	1	0	V	V	dc	X
BYTE READ (byte mode)	0x8A	dc	1	0	V	V	dc	X
MULTI READ STATUS	0x93	dc	0	0	dc	dc	dc	X
READ STATUS	0x96	dc	0	0	dc	dc	times	X
2P WRITE/MICRON	0xB0	V	0	V	V		sector	V
2P ERASE/SAM	0xC6	dc	0	0	dc	dc	2 blocks	X
2P PAGE READ	0xD0	V	0	V	V	dc	sector	X
2P WRITE/TOG1	0xE6	V	0	V	V	dc	sector	V
2P ERASE/TOG1	0xFC	dc	0	0	dc	dc	2 blocks	X
2P COPY BACK/TOG!	0x106	dc	0	0	dc	dc	2 pages	V
I2 PAGE WRITE	0x11A	V	0	V	V	dc	sector	V
I2 BLOCK ERASE	0x135	dc	0	0	dc	dc	2 blocks	X
2 LUN PAGE WRITE/MICRON	0x146	V	0	V	V	dc	sector	V

Command Register Setting	cmd index	bmc ignore	byte mode	cmd_ hsk mode	1st offest	2nd offset	counter unit	update spare
2 LUN PAGE WRITE/SAMSUNG	0x167	V	0	V	V	dc	sector	V
CACHE READ_1/TOSHIBA	0x185	V	0	V	0	dc	sector	X
CACHE READ_2/TOSHIBA	0x194	V	0	V	0	dc	sector	X
CACHE WRITE_1/TOSHIBA	0x19C	V	0	V	0	dc	sector	V
CACHE WRITE_2/TOSHIBA	0x1A9	V	0	V	0	dc	sector	V
COPY BACK WITH CACHE_1/TOSHIBA	0x1B5	dc	0	0	0	dc	dc	X
COPY BACK WITH CACHE_2/TOSHIBA	0x1C2	dc	0	0	0	dc	2pages + (page/ counter)	X
COPY BACK WITH CACHE_3/TOSHIBA	0x1D0	dc	0	0	0	dc	dc	X
HW COPY BACK	0x1DF	1	0	0	0	dc	sector	V
BLANKING WRITE	0x1F4	dc	0	0	dc	dc	page	X
BLANKING READ	0x1FF	dc	0	0	dc	dc	page	X
BLANKING CHECK	0x209	dc	0	0	dc	dc	page	X
SYN RESET	0x21A	dc	0	0	dc	dc	dc	X
RAD PARAMETER PAGE	0x21D	V	0	V	dc	dc	sector	X
READ UNIQUE ID	0x224	V	0	V	dc	dc	sector	X
GET FEATURE	0x22B	dc	1	0	dc	dc	dc	X
SET FEATURE	0x232	dc	1	0	dc	dc	dc	X
SELECT LUN WITH STATUS	0x238	dc	0	0	dc	dc	dc	X
SMALL PAGE COMMAND								
SMALL PAGE READ	0x23E	V	0	V	0	dc	sector	X
SMALL SPARE READ	0x249	dc	0	0	dc	dc	sector	X
SMALL BYTE READ 50	0x253	dc	1	0	V	dc	dc	X
SMALL BYTE READ 00	0x25c	dc	1	0	V	dc	dc	X
SMALL BYTE READ 01	0x264	dc	1	0	V	dc	dc	X
SMALL PAGE WRITE	0x26c	V	0	V	0	dc	sector	V
SMALL SPARE WRITE	0x278	dc	0	0	dc	dc	sector	V
SMALL BYTE WRITE 50	0x283	dc	1	0	V	dc	sector	X
SMALL BYTE WRITE 00	0x28d	dc	1	0	V	dc	sector	X
SMALL BYTE WRITE 01	0x297	dc	1	0	V	dc	sector	X
SMALL COPY BACK	0x2A1	dc	0	0	dc	dc	sector	X
SMALL HW COPY BACK	0x2AE	1	0	0	0	dc	sector	V
SMALL BLOCK ERASE	0x2C1	dc	0	0	dc	dc	block	X
SMALL BLANKING CHECK	0x2C8	dc	0	0	dc	dc	page	X

Notes:

1. “don't care” (dc) indicates that users can fill with anything.
2. “V” indicates the value-affect behavior.
3. “2 block/loop” indicates that “loop” is the counter number if the counter is set to ‘2’, which will be totally 2 * 2 = 4 blocks.
4. Update spare is the spare data written to Flash from the spare register.
5. The number behind the command name is the command index. Users can fill in specified value in the command table to perform a flow.

6. Cache read operation: The cache_read_1 counter must be filled with multiplier of page by sector and cache_read_2 counter can be filled with sector unit. Example if one page has four sectors and users want to perform cache read with nine sectors, then the cache_read_1 counter must be eight sectors and the cache_read_2 counter will be 9 – 8 = 1 sector. If users want to read three pages, the cache_read_1 counter will be eight sectors and the cache_read_2 counter will be four sectors. Consequently, the cache_read_2 counter will always be less than or equal to one page and the cache_read_1 counter will always be the multiplier of page
7. Cache write operation: The cache_write_1 counter must fill multiple pages by sector and the cache_write_2 counter can be filled with the sector unit. For example, if one page has eight sectors and users want to perform the cache write with 19 sectors, then the cache_write_1 counter must be 16 sectors and the cache_write_2 counter will be 19 – 16 = 3 sectors. If users want to write five pages, the cache_write_1 counter will be four pages (4 * 8 = 32 sectors) and the cache_write_2 counter will be one page (8 sectors). Consequently, the cache_write_2 counter will always be less than or equal to one page and cache_write_1 counter will always be the multiplier of page.
8. Cache write address: If users want perform cache write M page, then row1_addr of cache_write_1 will be N, and row1_addr of cache_write_2 will be (N + M - 1)
9. Cache Copy_back_2: if counter = 1, total copy back 3 pages and if counter = 3, total copy 3 + 2 = 5 pages. If users want to copy eight pages, the counter must be filled as 8 – 2 = 6.
10. Copy back with cache address: Please refer to the following Table 11-4 to decide the row address.

Table 11-4 Copy back with cache

copy M pages	row_1st_addr	row_2nd_addr
COPY BACK WITH CACHE_1/TOSHIBA	X	Y
COPY BACK WITH CACHE_2/TOSHIBA	X+1	Y+1
COPY BACK WITH CACHE_3/TOSHIBA	X+M-1	Y+M-1

11.1.11.4 Fixed Flow Command1 (CTD) and Usage

Before users issue a command, please refer to the following fixed flow, which contains some microcode and is filled with the corresponding register to execute this command. For example, the first row address and second row address can act as a counter. For other settings, please refer to Table 11-3.

Table 11-5 Subroutine Command Index

cmd_index	0	5	10	16	23
offset	FIX_RDST	FIX_MRDST	FIX_RDI	FIX_RDO	FIX_CRDY

cmd_index	0	5	10	16	23
0	OPCODE_70	OPCODE_70	OPCODE_85	OPCODE_05	OPCODE_70
1	BUFFER4	BUFFER4	BUFFER4	SP_COL_ADDR	BUFFER4
2	RD_ST	MRD_ST	SP_COL_ADDR	OPCODE_E0	CHK_RDY
3	BUFFER3	BUFFER3	BUFFER1	BUFFER4	BUFFER3
4	FLW_RETURN	FLW_RETURN	WR_SP	RD_SP	FLW_RETURN
5			FLW_RETURN	BUFFER3	
6				FLW_RETURN	

Subroutine usage: When the main flow executes FIX_MRDST subroutine, it will issue multiple read status subroutines and finally returns to the main flow.

Table 11-6 Basic I

offset	PAGE READ	PAGE WRITE WITH SPACE	SPARE WRITE	SPARE READ	PAGE READ WITH SPACE	PAGE WRITE
0	OPCODE_00	OPCODE_80	OPCODE_80	OPCODE_00	OPCODE_00	OPCODE_80
1	COL_1st ADDR	COL_1st ADDR	COL_1st ADDR	COL_1st ADDR	COL_1st ADDR	COL_1st ADDR
2	ROW_1st ADDR	ROW_1st ADDR	ROW_1st ADDR	ROW_1st ADDR	ROW_1st ADDR	ROW_1st ADDR
3	OPCODE_30	BUFFER1	BUFFER1	OPCODE_30	OPCODE_30	BUFFER1
4	BUSY	WDATA	WR_SP	BUSY	BUSY	WDATA
5	BUFFER2	GOTO_FIX	OPCODE_10	BUFFER2	BUFFER2	OPCODE_10
6	RDATA	FIX_RDI	BUSY	RD_SP	RDATA	BUSY
7	BUFFER3	OPCODE_10	GOTO_FIX	BUFFER3	BUFFER3	GOTO_FIX
8	GOBACK_8	BUSY	FIX_RDST	GOBACK_8	GOTO_FIX	FIX_RDST
9	FLW_END	GOTO_FIX	GOBACK_9	FLW_END	RIX_RDO	GOBACK_9
10		FIS_RDST	FLW_END		GOBACK_10	FLW_END
11		GOBACK_11			FLW_END	
12		FLW_END				

Note: The offset number is the offset of a flow. For example, if users want to perform a page read flow and issue 1st_row_addr instead of starting from '0', only (Index + offset) should be issued, which is equivalent to 0 + 2 = 2. Consequently, users can fill two command indexes in the command table.

Table 11-7 Basic II

offset	READ ID (Byte Mode)	RESET FLASH	BLOCK ERASE	COPY BACK
0	OPCODE_90	OPCODE_FF	OPCODE_60	OPCODE_00
1	COL_1st ADDR	COL_1st ADDR	COL_1st ADDR	ZERO_ADDR
2	BUFFER4	BUSY	OPCODE_D0	ROW_1st_ADDR
3	RD_SP	FLW_END	BUSY	OPCODE_35
4	BUFFER3		GOTO_FIX	BUSY
5	FLW_END		FIX_RDST	OPCODE_85
6			GOBACK_6	SP_COL_ADDR
7			FLW_END	ROW_2nd_ADDR
8				BUFFER1
9				WR_SP
10				OPCODE_10

offset	READ ID (Byte Mode)	RESET FLASH	BLOCK ERASE	COPY BACK
11				BUSY
12				GOTO_FIX
13				FIX_RDST
14				GOBACK_14
15				FLW_END

Table 11-8 Basic III

offset	BYTE WRITE	BYTE READ	MULTI READ STATUS	READ STATUS
0	OPCODE_80	OPCODE_00	GOTO_FIX	GOTO_FIX
1	COL_1st_ADDR	COL_1st_ADDR	FIX_MRDST	FIX_RDST
2	ROW_1st_ADDR	ROW_1st_ADDR	FLW_END	GOBACK_2
3	BUFFER1	OPCODE_30		FLW_END
4	WR_SP	BUSY		
5	OPCODE_10	BUFFER2		
6	BUSY	RD_SP		
7	GOTO_FIX	BUFFER3		
8	FIX_RDST	FLW_END		
9	FLW_END			

Table 11-9 2Plane I

offset	2P WRITE/MICRON	2P ERASE/SAM	2P PAGE READ
0	OPCODE_80	OPCODE _60	OPCODE_00
1	COL_1st_ADDR	FIX_2nd_ADDR	COL_1st_ADDR
2	ROW_1st_ADDR	OPCODE_60	ROW_1st_ADDR
3	BUFFER1	TOG_1st_ADDR	OPCODE_30
4	WDATA	OPCODE_D0	BUSY
5	GOTO_FIX	BUSY	BUFFER2
6	FIX_RDI	GOTO_FIX	RDATA
7	OPCODE _11	FIX_RDST	BUFFER3
8	BUSY	GOBACK_8	GOTO_FIX
9	OPCODE_80	FLW_END	FIX_RDO
10	ZERO_ADDR		OPCODE_00
11	ROW_2nd_ADDR		ZERO_ADDR
12	BUFFER1		TOG_1st_ADDR
13	WDATA		OPCODE_30
14	GOTO_FIX		BUSY
15	FIX_RDI		BUFFER2
16	OPCODE_10		RDATA
17	BUSY		BUFFER3
18	GOTO_FIX		GOTO_FIX
19	FIX_RDST		FIX_RDO
20	GOBACK_20		GOTO_RDO
21	FLW_END		FLW_END

Note: “2P PAGE WRITE” has two different fix flows, one is for Samsung and the other is for Micron. “2P PAGE WRITE/Samsung” is for Samsung and “2P PAGE WRITE/Micron”

is for Micron. “2P PAGE WRITE” depends on the Samsung Flash specifications to generate an easy and simple fixed flow. If users want to use this flow, please check if the flow matches the specifications. Users should treat “2P PAGE WRITE/Micron” as the one for Samsung.

Table 11-10 2Plane II

offset	2P WRITE/TOG1	2P ERASE/TOG1	2P COPY BACK/TOG1
0	OPCODE_80	OPCODE _60	OPCODE_60
1	COL_1st_ADDR	FIX_2nd_ADDR	ROW_1st_ADDR
2	ROW_1st_ADDR	OPCODE_60	OPCODE_60
3	BUFFER1	ROW_2nd_ADDR	TOG_1st_ADDR
4	WDATA	OPCODE_D0	OPCODE_35
5	GOTO_FIX	BUSY	BUSY
6	FIX_RDI	GOTO_FIX	OPCODE_85
7	OPCODE _11	FIX_RDST	ZERO_ADDR
8	BUSY	GOBACK_8	ROW_2nd_ADDR
9	OPCODE_81	FLW_END	OPCODE_11
10	ZERO_ADDR		BUSY
11	ROW_2nd_ADDR		OPCODE_81
12	BUFFER1		ZERO_ADDR
13	WDATA		TOG_2nd_ADDR
14	GOTO_FIX		OPCODE_10
15	FIX_RDI		BUSY
16	OPCODE_10		GOTO_FIX
17	BUSY		FIX_RDST
18	GOTO_FIX		GOBACK_18
19	FIX_RDST		FLW_END
20	GOBACK_20		
21	FLW_END		

Table 11-11 Interleaving

offset	I2 PAGE WRITE	I2 BLOCK ERASE	2 LUN PAGE WRITE/MICRON	2 LUN PAGE WRITE/SAMSUNG
0	OPCODE_80	OPCODE _60	OPCODE_80	OPCODE_80
1	COL_1st_ADDR	ROW_1st_ADDR	COL_1st_ADDR	COL_1st_ADDR
2	ROW_1st_ADDR	OPCODE_D0	ROW_1st_ADDR	ROW_1st_ADDR
3	BUFFER1	INC_CE	BUFFER1	BUFFER1
4	WDATA	GOTO_FIX	WDATA	WDATA
5	GOTO_FIX	FIX_CRDY	GOTO_FIX	GOTO_FIX
6	FIX_RDI	OPCODE_60	FIX_RDI	FIX_RDI
7	OPCODE _10	ROW_2nd_ADDR	OPCODE_10	OPCODE_10
8	INC_CE	OPCODE_DO	OPCODE_78	OPCODE_F2
9	GOTO_FIX	DEC_CE	FIX_2nd_ADDR	BUFFER4
10	FIX_CRDY	GOTO_FIX	BUFFER4	CHK_RDY
11	OPCODE_80	FIX_MRDST	CHK_RDY	BUFFER3
12	ZERO_ADDR	GOBACK_12	BUFFER3	OPCPDE_80
13	ROW_2nd_ADDR	INC_CE	OPCODE_80	ZERO_ADDR

offset	I2 PAGE WRITE	I2 BLOCK ERASE	2 LUN PAGE WRITE/MICRON	2 LUN PAGE WRITE/SAMSUNG
14	BUFFER1	GOTO_FIX	ZERO_ADDR	ROW_2nd_ADDR
15	WDATA	FIX_MRDST	ROW_2nd_ADDR	BUFFER1
16	GOTO_FIX	FLW_END	BUFFER1	WDATA
17	FIX_RDI		WDATA	GOTO_FIX
18	OPCODE_10		GOTO_FIX	FIX_RDI
19	DEC_CE		FIX_RDI	OPCODE_10
20	GOTO_FIX		OPCODE_10	OPCODE_F1
21	FIX_MRDST		OPCODE_78	BUFFER4
22	GOBACK_22		FIX_1st_ADDR	MRD_ST
23	INC_CE		BUFFER4	BUFFER3
24	GOTO_FIX		MRD_ST	GOBACK_24
25	FIX_MRDST		BUFFER3	OPCODE_F2
26	FLW_END		GOBACK_26	BUFFER4
27			OPCODE_78	MRD_ST
28			FIX_2nd_ADDR	BUFFER3
29			BUFFER4	FLW_END
30			MRD_ST	
31			BUFFER3	
32			FLW_END	

Table 11-12 Cache

offset	CACHE READ TOSHIBA/MICRON	CACHE WRITE TOSHIBA/MICRON	CACHE COPY BACK
0	OPCODE_00 (cache read_1)	OPCODE _80 (cache write_1)	OPCODE_00 (cache copy_1)
1	COL_1st_ADDR	COL_1st_ADDR	COL_1st_ADDR
2	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR
3	OPCODE_30	BUFFER1	OPCODE_30
4	BUSY	WDATA	BUSY
5	BUFFER2	GOTO_FIX	BUFFER2
6	OPCPDE_31	FIX_RDI	OPCODE_8C
7	BUSY	OPCODE_15	ZERO_ADDR
8	BUFFER2	BUSY	ROW_2nd_ADDR
9	RDATA	GOTO_FIX	BUFFER1
10	BUFFER3	FIX_RDST	OPCODE_15
11	GOTO_FIX	GOBACK_11	BUSY
12	FIX_RDO	LINK	LINK
13	GOBACK_7	OPCODE_80 (cache write_2)	OPCODE_00 (cache copy_2)
14	LINK	ZERO_ADDR	ZERO_ADDR
15	OPCODE_3F (cache read_2)	ROW_1st_ADDR	ROW_1st_ADDR
16	BUSY	BUFFER1	OPCODE_3A
17	BUFFER2	WDATA	BUSY
18	RDATA	GOTO_FIX	BUFFER3
19	BUFFER3	FIX_RDI	OPCPDE_8C
20	GOTO_FIX	OPCODE_10	ZERO_ADDR
21	RIX_RDO	BUSY	ROW_2nd_ADDR
22	FLW_END	GOTO_FIX	BUFFER1

offset	CACHE READ TOSHIBA/MICRON	CACHE WRITE TOSHIBA/MICRON	CACHE COPY BACK
23		FIX_RDST	OPCODE_15
24		FLW_END	BUSY
25			GOBACK_12
26			LINK
27			OPCODE_00 (cache copy_3)
28			ZERO_ADDR
29			ROW_1st_ADDR
30			OPCODE_3A
31			BUSY
32			BUFFER2
33			OPCPDE_8C
34			ZERO_ADDR
35			ROW_2nd_ADDR
36			BUFFER1
37			OPCODE_10
38			BUSY
39			GOTO_FIX
40			FIX_RDST
41			FLW_END

Table 11-13 Misc

offset	HW COPY BACK	BLANKING WRITE	BLANKING READ	BLANKING CHECK
0	OPCODE_00	OPCODE _80	OPCODE_00	OPCODE_80
1	ZERO_ADDR	COL_1st_ADDR	ZERO_ADDR	ZERO_ADDR
2	ROW_1st_ADDR	ZERO_ADDR	ROW_1st_ADDR	ROW_1st_ADDR
3	OPCODE_30	ROW_1st_ADDR	OPCODE_30	BUFFER1
4	BUSY	BUFFER1	BUSY	BLK_WR
5	BUFFER2	BLK_WR	BUFFER2	OPCODE_10
6	RDATA	OPCODE_10	BLK_RD	BUSY
7	BUFFER3	BUSY	BUFFER3	OPCODE_00
8	OPCODE_80	GOTO_FIX	GOBACK_8	ZERO_ADDR
9	ZERO_ADDR	FIX_RDST	FLW_END	ROW_1st_ADDR
10	ROW_2nd_ADDR	GOBACK_9		OPCODE_30
11	BUFFER1	FLW_END		BUSY
12	WDATA			BUFFER2
13	GOTO_FIX			BLK_RD
14	FIX_RDI			BUFFER3
15	OPCODE_10			GOBACK_15
16	BUSY			FLW_END
17	GOTO_FIX			
18	FIX_RDST			
19	GOBACK_19			
20	FLW_END			

Table 11-14 DDR

offset	SYN RESET	READ PARAMETER PAGE	READ UNIQE ID	GET FEATURE	SET FEATURE	SELECT LUN WITH STATUS
0	OPCODE_FC	OPCODE_EC	OPCODE_ED	OPCODE_EE	OPCODE_EF	OPCODE_78
1	BUSY	ROW_1st_ ADDR	ROW_1st_ ADDR	ROW_1st_ ADDR	ROW_1st_ ADDR	ROW_1st_ ADDR
2	FLW_END	BUSY	BUSY	BUSY	BUFFER1	BUFFER4
3		BUFFER2	BUFFER2	BUFFER2	WR_SP	RD_ST
4		RDATA	RDATA	RS_SP	BUSY	BUFFER3
5		BUFFER3	BUFFER3	BUFFER3	FLW_END	FLW_END
6		FLW_END	FLW_END	FLW_END		

Note: Users must set no_ecc_parity and the sector counter according to the specification when executing READ_PARAMSTER_PAGE/READ UNIQUE ID.

Table 11-15 Small Page Read

offset	SMALL PAGE READ	SMALL SPARE READ	SMALL BYTE READ_50	SMALL BYTE READ_00	SMALL BYTE READ_01
0	OPCODE_00	OPCODE_50	OPCODE_50	OPCODE_00	OPCODE_01
1	COL_1st_ADDR	SP_COL_ADDR	COL_1st_ADDR	COL_1st_ADDR	COL_1st_ADDR
2	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR
3	BUSY	BUSY	BUSY	BUSY	BUSY
4	BUFFER2	BUFFER2	BUFFER2	BUFFER2	BUFFER2
5	RDATA	RD_SP	RD_SP	RD_SP	RD_SP
6	RD_SP	BUFFER3	BUFFER3	BUFFER3	BUFFER3
7	BUFFER3	BUSY	BUSY	FLW_END	FLW_END
8	BUSY	GOBACK_8	FLW_END		
9	GOBACK_9	FLW_END			
10	FLW_END				

Table 11-16 Small Page Write

offset	SMALL PAGE WRITE	SMALL SPARE WRITE	SMALL BYTE WRITE_50	SMALL BYTE WRITE_00	SMALL BYTE WRITE_01
0	OPCODE_00	OPCODE_50	OPCODE_50	OPCODE_00	OPCODE_01
1	OPCODE_80	OPCODE_80	OPCODE_80	OPCODE_80	OPCODE_80
2	COL_1st_ADDR	SP_COL_ADDR	COL_1st_ADDR	COL_1st_ADDR	COL_1st_ADDR
3	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR
4	BUFFER1	BUFFER1	BUFFER1	BUFFER1	BUFFER1
5	WDATA	WR_SP	WR_SP	WR_SP	WR_SP
6	WR_SP	OPCODE_10	OPCODE_10	OPCODE_10	OPCODE_10
7	OPCODE_10	GOTO_FIX	GOTO_FIX	GOTO_FIX	GOTO_FIX
8	GOTO_FIX	FIX_MRDST	FIX_MRDST	FIX_MRDST	FIX_MRDST
9	FIX_MRDST	GOBACK_8	FLW_END	FLW_END	FLW_END
10	GOBACK_9	FLW_END			
11	FLW_END				

Table 11-17 Small Page Misc

offset	SMALL COPY BACK	SMALL HW COPY BACK	SMALL BLOCK ERASE	SMALL BLANKING CHECK
0	OPCODE_00	OPCODE_00	OPCODE_60	OPCODE_00
1	COL_1st_ADDR	COL_1st_ADDR	ROW_1st_ADDR	OPCODE_80
2	ROW_1st_ADDR	ROW_1st_ADDR	OPCODE_D0	ZERO_ADDR
3	BUSY	BUSY	GOTO_FIX	ROW_1st_ADDR
4	BUFFER2	BUFFER2	FIX_MRDST	BUFFER1
5	OPCODE_8A	RDATA	GOBACK_5	BLK_WR
6	COL_2nd_ADDR	BUFFER3	FLW_END	OPCODE_10
7	ROW_2nd_ADDR	BUSY		GOTO_FIX
8	BUSY	OPCODE_80		FIX_MRDST
9	GOTO_FIX	COL_2nd_ADDR		OPCODE_00
10	FIX_MRDST	ROW_2nd_ADDR		ZERO_ADDR
11	GOBACK_11	BUFFER1		ROW_1st_ADDR
12	FLW_END	WDATA		BUSY
13		WR_SP		BUFFER2
14		OPCPDE_10		BLK_RD
15		GOTO_FIX		BUFFER3
16		FIX_MRDST		BUSY
17		GOBACK_17		GOBACK_17
18		FLW_END		FLW_END

11.1.11.5 Command Register Setting for Fixed Flow Command 2

Table 11-18 Command Register Setting for Fixed Flow Command 2

Command Register Setting Large Page	cmd index	bmc ignore	byte mode	cmd_ hsk mode	1st offest	2nd offset	counter unit	update spare
PAGE READ	0x79	V	0	V	V	dc	sector	X
PAGE WRITE WITH SPARE	0x85	V	0	V	V	dc	sector	V
SPARE WRITE	0x91	dc	0	0	V	dc	page	V
SPARE READ	0x9B	dc	0	0	V	dc	page	X
PAGE READ WITH SPARE	0xA7	V	0	V	V	dc	sector	X
READ ID (byte mode)	0xB5	dc	1	0	V	V	dc	X
RESET	0xBB	dc	0	0	dc	dc	dc	X
BLOCK ERASE	0xBF	dc	0	0	dc	dc	block	X
BLANKING CHECK	0xC6	dc	0	0	dc	dc	page	X
BYTE WRITE (byte mode)	0xDA	dc	1	0	V	V	dc	X
BYTE READ (byte mode)	0xE3	dc	1	0	V	V	dc	X
MULTI READ STATUS	0xEE	dc	0	0	dc	Dc	dc	X
READ STATUS	0xF1	dc	0	0	dc	dc	times	X
SMALL PAGE COMMAND								
SMALL PAGE READ	0xF5	V	0	V	0	dc	sector	X
SMALL SPARE READ	0x100	dc	0	0	dc	dc	sector	X
SMALL BYTE READ 50	0x10A	dc	1	0	V	dc	dc	X
SMALL BYTE READ 00	0x113	dc	1	0	V	dc	dc	X
SMALL BYTE READ 01	0x11B	dc	1	0	V	dc	dc	X
SMALL PAGE WRITE	0x123	V	0	V	0	dc	sector	V

Command Register Setting Large Page	cmd index	bmc ignore	byte mode	cmd_ hsk mode	1st offest	2nd offset	counter unit	update spare
SMALL SPARE WRITE	0x12F	dc	0	0	dc	dc	sector	V
SMALL BYTE WRITE 50	0x13A	dc	1	0	V	dc	sector	X
SMALL BYTE WRITE 00	0x144	dc	1	0	V	dc	sector	X
SMALL BYTE WRITE 01	0x14E	dc	1	0	V	dc	sector	X
SMALL HW COPY BACK	0x158	1	0	0	0	dc	sector	V
SMALL BLOCK ERASE	0x16B	dc	0	0	dc	dc	block	X
SMALL BLANKING CHECK	0x172	dc	0	0	dc	dc	page	X
GET FEATURE	0x185	dc	1	0	dc	dc	dc	X
SET FEATURE	0x18E	dc	1	0	dc	dc	dc	X

11.1.11.6 Fixed Flow Command 2 (MTD) and Usage

Table 11-19 Subroutine Command Index

cmd_index	0	5	10	16	23
offset	FIX_RDST	FIX_MRDST	FIX_RDI	FIX_RDO	FIX_CRDY
0	OPCODE_70	OPCODE_70	OPCODE_85	OPCODE_05	OPCODE_70
1	BUFFER4	BUFFER4	BUFFER4	SP_COL_ADDR	BUFFER4
2	RD_ST	MRD_ST	SP_COL_ADDR	OPCODE_E0	CHK_RDY
3	BUFFER3	BUFFER3	BUFFER1	BUFFER4	BUFFER3
4	FLW_RETURN	FLW_RETURN	WR_SP	RD_SP	FLW_RETURN
5			FLW_RETURN	BUFFER3	
6				FLW_RETURN	

Table 11-20 Subroutine Command Index

cmd_index	subroutine
0	FIX_RDST
5	FIX_MRDST
10	FIX_RDI
16	FIX_RDO
23	FIX_CRDY
133	FIX_PWS
155	FIX_SR
167	FIX_PRS
191	FIX_ER
218	FIX_BW
227	FIX_BR

FIX_PWS is page write with spare fixed flow.
FIX_SR is spare write fixed flow.
FIX_PRS is page read with spare fixed flow.
FIX_ER is block erase fixed flow.
FIX_BW is byte write fixed flow.
FIX_BR is byte read fixed flow.

Table 11-21 Large page I

offset	PAGE READ	PAGE WRITE WITH SPARE	SPARE WRITE	SPARE READ	PAGE READ WITH SPARE
0	OPCODE_00	OPCODE_80	OPCODE_80	OPCODE_00	OPCODE_00
1	COL_1st_ADDR	COL_1st_ADDR	SP_COL_ADDR	SP_COL_ADDR	COL_1st_ADDR
2	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR
3	OPCODE_30	BUFFER1	BUFFER1	OPCODE_30	OPCODE_30
4	GOTO_FIX	WDATA	WR_SP	GOTO_FIX	GOTO_FIX
5	FIX_MRDST	GOTO_FIX	OPCODE_10	FIX_MRDST	FIX_MRDST
6	OPCODE_00	FIX_RDI	GOTO_FIX	OPCODE_00	OPCODE_00
7	BUFFER4	OPCODE_10	FIX_MRDST	BUFFER4	BUFFER4
8	RDATA	GOTO_FIX	GOBACK_8	RD_SP	RDATA
9	BUFFER3	FIX_MRDST	FLW_END	BUFFER3	BUFFER3
10	GOBACK_10	GOBACK_10		GOBACK_10	GOTO_FIX
11	FLW_END	FLW_END		FLW_END	FIX_RDO
12					GOBACK_12
13					FLW_END

Table 11-22 Large page II

offset	PAGE ID (Byte mode)	RESET FLASH	BLOCK ERASE	BLANKING CHECK
0	OPCODE_90	OPCODE_FF	OPCODE_60	OPCODE_80
1	COL_1st_ADDR	GOTO_FIX	ROW_1st_ADDR	ZERO_ADDR
2	BUFFER4	FIX_MRDST	OPCODE_D0	ROW_1st_ADDR
3	RD_SP	FLW_END	GOTO_FIX	BUFFER1
4	BUFFER3		FIX_MRDST	BLK_WR
5	FLW_END		GOBACK_5	OPCODE_10
6			FLW_END	GOTO_FIX
7				FIX_MRDST
8				OPCODE_00
9				ZERO_ADDR
10				ROW_1st_ADDR
11				OPCODE_30
12				GOTO_FIX
13				FIX_MRDST
14				OPCODE_00
15				BUFFER4
16				BLK_RD
17				BUFFER3
18				GOBACK_18
19				FLW_END

Table 11-23 Large page III

offset	PAGE WRITE	BYTE READ	MULTI READ STATUS	READ STATUS
0	OPCODE_80	OPCODE_00	GOTO_FIX	GOTO_FIX
1	COL_1st_ADDR	COL_1st_ADDR	FIX_MRDST	FIX_RDST
2	ROW_1st_ADDR	ROW_1st_ADDR	FLW_END	GOBACK_2
3	BUFFER1	OPCODE_30		FLW_END
4	WR_SP	GOTO_FIX		

offset	PAGE WRITE	BYTE READ	MULTI READ STATUS	READ STATUS
5	OPCODE_10	FIX_MRDST		
6	GOTO_FIX	OPCODE_00		
7	FIX_MRDST	BUFFER4		
8	FLW_END	RD_SP		
9		BUFFER3		
10		FLW_END		

Table 11-24 Small page I

offset	SMALL PAGE READ	SMALL SPARE READ	SMALL BYTE READ_50	SMALL BYTE READ_00	SMALL BYTE READ_01
0	OPCODE_00	OPCODE_50	OPCODE_50	OPCODE_00	OPCODE_01
1	COL_1st_ADDR	SP_COL_ADDR	COL_1st_ADDR	COL_1st_ADDR	COL_1st_ADDR
2	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR
3	BUSY	BUSY	BUSY	BUSY	BUSY
4	BUFFER2	BUFFER2	BUFFER2	BUFFER2	BUFFER2
5	RDATA	RD_SP	RD_SP	RD_SP	RD_SP
6	RD_SP	BUFFER3	BUFFER3	BUFFER3	BUFFER3
7	BUFFER3	BUSY	BUSY	FLW_END	FLW_END
8	BUSY	GOBACK_8	FLW_END		
9	GOBACK_9	FLW_END			
10	FLW_END				

Table 11-25 Small page II

offset	SMALL PAGE WRITE	SMALL SPARE WRITE	SMALL BYTE WRITE_50	SMALL BYTE WRITE_00	SMALL BYTE WRITE_01
0	OPCODE_00	OPCODE_50	OPCODE_50	OPCODE_00	OPCODE_01
1	OPCODE_80	OPCODE_80	OPCODE_80	OPCODE_80	OPCODE_80
2	COL_1st_ADDR	SP_COL_ADDR	COL_1st_ADDR	COL_1st_ADDR	COL_1st_ADDR
3	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR	ROW_1st_ADDR
4	BUFFER1	BUFFER1	BUFFER1	BUFFER1	BUFFER1
5	WDATA	WR_SP	WR_SP	WR_SP	WR_SP
6	WR_SP	OPCODE_10	OPCODE_10	OPCODE_10	OPCODE_10
7	OPCODE_10	GOTO_FIX	GOTO_FIX	GOTO_FIX	GOTO_FIX
8	GOTO_FIX	FIX_MRDST	FIX_MRDST	FIX_MRDST	FIX_MRDST
9	FIX_MRDST	GOBACK_8	FLW_END	FLW_END	FLW_END
10	GOBACK_9	FLW_END			
11	FLW_END				

Table 11-26 Small page III

offset	SMALL COPY BACK	SMALL BLOCK ERASE	SMALL BLANKING CHECK	GET FEATURE	SET FEATURE
0	OPCODE_00	OPCODE_60	OPCODE_00	OPCODE_EE	OPCODE_EF
1	COL_1st_ADDR	ROW_1st_ADDR	OPCODE_80	ROW_1st_ADDR	ROW_1st_ADDR
2	ROW_1st_ADDR	OPCODE_D0	ZERO_ADDR	GOTO_FIX	BUFFER1
3	BUSY	GOTO_FIX	ROW_1st_ADDR	FIX_MRDST	WR_SP
4	BUFFER2	FIX_MRDST	BUFFER1	OPCODE_00	GOTO_FIX

offset	SMALL COPY BACK	SMALL BLOCK ERASE	SMALL BLANKING CHECK	GET FEATURE	SET FEATURE
5	RDATA	GOBACK_5	BLK_WR	BUFFER4	FIX_MRDST
6	BUFFER3	FLW_END	OPCODE_10	RD_SP	FLW_END
7	BUSY		GOTO_FIX	BUFFER3	
8	OPCODE_80		FIX_MRDST	FLW_END	
9	COL_2nd_ADDR		OPCODE_00		
10	ROW_2nd_ADDR		ZERO_ADDR		
11	BUFFER1		ROW_1st_ADDR		
12	WDATA		BUSY		
13	WR_SP		BUFFER2		
14	OPCODE_10		BLK_RD		
15	GOTO_FIX		BUFFER3		
16	FIX_MRDST		BUSY		
17	GOBACK_17		GOBACK_17		
18	FLW_END		FLW_END		

11.2 Memory Map and Register Definition

Table 11-27 Summary of NAND-Controller Registers

Address Offset	Type	Size (Byte)	Description	Default Value
ECC Control				
0x0000	R	4	ECCSR	0x0000_0000
0x0004	Res	-	-	-
0x0008	RW	4	ECCCR	0x0000_0000
0x000C	Res	-	-	-
0x0010	RW	4	TNECCEBR0	0x0000_0000
0x0014	Res	-	-	-
0x0018	RW	4	NECCCCR0	0x0000_0000
0x001C	Res	-	-	-
0x0020	RW	4	ECCIER	0x0000_0000
0x0024	RW	4	ECCISR	0x0000_0000
0x0028	W	4	ECCSCR	0x0000_0000
0x002C	R	4	ECCSSRR0	0x0000_0000
0x0030	R	4	ECCSSRR1	0x0000_0000
0x0034	R	4	SRECCCR0	0x0000_0000
0x0038	Res	-	-	-
0x003C	R	4	SRECCCR1	0x0000_0000
0x0040	Res	-	-	-
NANDC Control				
0x0100	R	4	DBRSR	0x0000_00FF
0x0104	RW	4	NANDCGSR	0x0200_6000
0x0108	RW	4	MASR1	0x0000_0000
0x010C	RW	4	MASR2	0x00FF_8000

Address Offset	Type	Size (Byte)	Description	Default Value
0x0110	RW	4	ACTR0N0	0x0F1F_0F1F
0x0114	RW	4	ACTR1N0	0x0000_7F7F
0x014C..0x0118	Res	-	-	-
0x0150	RW	4	NANDCIER	0x0000_0000
0x0154	RW	4	NANDCISR	0x0000_0000
0x0158	R	4	CARACH0	0x0000_0000
0x0174..0x015C	Res	-	-	-
0x0178	R	4	RSR0	0x0000_0000
0x017C	R	4	RSR1	0x0000_0000
0x0180	RW	4	ATBLR	0x0000_0000
0x0184	W	4	NANDCSRR	0x0000_0000
0x0188	Res	-	-	-
0x018C	RW	4	NANDCACPR	0x0000_0000
0x0190	RW	4	ACTR2N0	0x0F1F_0F1F
0x0194	Res	-	-	-
0x01CC..0x0198	Res	-	-	-
0x01D0	RW	4	VAR	0x0000_0000
Command Queue Control				
0x0200	R	4	CQSR	0x0000_00FF
0x0204	W	4	CQFR	0x0000_0000
0x0208	R	4	CCC	0x0000_0000
0x020C	W	4	CCCRR	0x0000_0000
0x028C.. 0x0280	W	4	GCQA	0x0000_0000
NANDC Command Queue				
0x0300	RW	4	CQR1C	0x0000_0000
0x0304	RW	4	CQR2C	0x0000_0000
0x0308	RW	4	CQR3C	0x0000_0000
0x030C	RW	4	CQR4C	0x0000_0000
0x03F4..0x0310	Res	-	-	-
BMC Control				
0x0400	R	4	BMCRSR	0xFF00_00FF
0x0404	W	4	RUMPAR	0x0000_0000
0x0420..0x0408	Res	-	-	-
0x0424	W	4	DMAMWDRDPR	0x0000_0000
0x0428	W	4	RSRR	0x0000_0000
0x042C	RW	4	FRFRDR	0x0000_0000
0x0430	R	4	RRSCR	0x0000_0000
0x044C..0x0434	Res	-	-	-
Miscellaneous				
0x0508	RW	4	AHBSMSRR	0x0280_FF01
0x050C	W	4	GSRR	0x0000_0000
0x0510	Res	-	-	-
0x0514	R	4	ECCCCR1	0x0000_0000
0x0518	R	4	ECCCCR2	0x0000_0000

Address Offset	Type	Size (Byte)	Description	Default Value
0x051C	R	4	ECCCCR3	0x0000_0000
0x0520	Res	-	-	-
Programmable OPCODE				
0x0707..0x0700	RW	4	POPCR	0x0000_0000
Spare SRAM Access Port				
0x10FF..0x1000	RW	4	SAR	0x0000_0000
Programmable Flow Control				
0x20FF..0x2000	RW	4	PFCR	0x0000_0000
Data SRAM Access Port Register				
0x2_3FFF..0x2_0000	RW	4	DSAP	0xXXXX_XXXX

The registers between offsets 0x0 to 0x204 and offsets from 0x400 to 0x1FFFF can be accessed with BYTE, HWORD or WORD as HSIZE. The command queue input entry can only accept WORD as HSIZE.

11.2.1 ECC Status Register

Offset: 0x0000

The ECC status register reflects the number of ECC error bits. Only the maximum error bits value is recorded. To clear this register, write '1' to the ECC status clear register (Offset = 0x0028, 0). When ECC fails, the ECC status register will not be valid and the ECC status information may not be correct.

Table 11-28 ECC Status Register

Bit	Name	Type	Reset	Description
31..7	Reserved	-	0	-
6..0	ECC_ERR_NO	R	0	Number of the ECC error bits

11.2.2 ECC Control and Threshold Register

Offset: 0x0008, 0x0010, 0x0018

The ECC control register provides the following features:

- Use threshold of the ECC error bits to generate the ECC interrupt
- ECC correction capability bits. Users must set an accepted ECC correction capability at the data region and spare region whenever the ECC engine is enabled or disabled. For example, if the user hardware configuration can only be 4-bit, 8-bit, or 16-bit capability, user must set 4-bit, 8-bit, or 16-bit ECC correction capability at both data region and spare region.
- ECC base size (512 bytes or 1K bytes)
- ECC function enable/disable switch

- ECC error data blocking function mask (Sectors containing the uncorrectable data that can be read from the AHB data slave port without blocking.)

Table 11-29 ECC Control Register (0x0008)

Bit	Name	Type	Reset	Description
31..18	Reserved	-	0	-
17	NO_ECC_PARITY	RW	0	No ECC parity 1: No spacing between the data sectors to accommodate ECC parity 0: Keep spacing between the data sectors to accommodate ECC parity This bit is set when using the Flash device with the embedded ECC circuit. Note: no_ecc_parity is enabled; ecc_en is not valid.
16	ECC_BASE	RW	0	ECC base size 1: ECC parity is generated and checked based on the data size of 1K bytes. 0: ECC parity is generated and checked based on the data size of 512 bytes.
15..9	Reserved	-	0	-
8	ECC_EN	RW	0	ECC function enable/disable switch 1: Enable the ECC function 0: Disable the ECC function
7..1	Reserved	-	0	-
0	ECC_ERR_MASK	RW	0	ECC error data blocking mask 1: Mask the ECC error data blocking function 0: Keep the ECC error data blocking function Note: The ECC-related interrupts will remain asserting when this bit is set. It is only for the data region, which does not include the spare region.

Table 11-30 ECC Threshold Register (0x0010)

Bit	Name	Type	Reset	Description
31..7	Reserved	-	0	-
6..0	ECC_THRES_BITS	RW	0	Threshold number of the ECC error bits 0..15: 1bit to 16bits

Table 11-31 ECC Correction Register (0x0018)

Bit	Name	Type	Reset	Description
31..7	Reserved	-	0	-
6..0	ECC_CORR_BITS	RW	0	This field sets the number of the ECC correction capability bits 0..15: 1bit to 16bits

11.2.3 ECC Interrupt Enable Register

Offset: 0x0020

The ECC function may results in the following two kinds of interrupts:

- Number of the ECC error bits hit the threshold which is set in the ECC control register.
- Failure of the ECC correction

Table 11-32 ECC Interrupt Enable Register

Bit	Name	Type	Reset	Description
31..4	Reserved	-	0	-
3	ECC_ERR_HIT_THRES_EN_SP	RW	0	ECC error bits hit the threshold number in the spare region interrupt enable. 1: Enable 0: Disable
2	ECC_CORR_FAIL_EN_SP	RW	0	ECC correction fail of the spare region interrupt enable. 1: Enable 0: Disable
1	ECC_ERR_HIT_THRES_EN	RW	0	ECC error bits hit the threshold number interrupt enable. 1: Enable 0: Disable
0	ECC_CORR_FAIL_EN	RW	0	ECC correction fail interrupt enable 1: Enable 0: Disable

11.2.4 ECC Interrupt Status Register

Offset: 0x0024

The ECC interrupt status register indicates the failure of ECC correction fail or the error bits hitting the threshold of each channel. Write ‘1’ to the specific bit to clear the status.

Table 11-33 ECC Interrupt Status Register

Bit	Name	Type	Reset	Description
31..25	Reserved	-	0	-
24	ECC_ERR_HIT_THRES_SP	RW	0	ECC error bits hit the threshold spare region.
23..17	Reserved	-	0	-
16	ECC_ERR_FAIL_SP	RW	0	ECC correction fail spare region
15..9	Reserved	-	0	-
8	ECC_ERR_HIT_THRES	RW	0	ECC error bits hit the threshold.
7..1	Reserved	-	0	-
0	ECC_ERR_FAIL	RW	0	ECC correction fail.

11.2.5 ECC Status Clear Register

Offset: 0x0028

Table 11-34 ECC Status Clear Register

Bit	Name	Type	Reset	Description

Bit	Name	Type	Reset	Description
31..9	Reserved	-	0	-
8	ECC_ERR_CNT_SP_CLR	W	0	ECC error bits clear spare region 1: Clear the error bits recorded in the ECC status register of the spare region 0: No effect
7..1	Reserved	-	0	-
0	ECC_ERR_CNT_CLR	W	0	ECC error bits clear register 1: Clear the error bits recorded in the ECC status register 0: No effect

11.2.6 ECC Status of Spare Region Registers

Offset: 0x002C

The ECC status of the spare region register reflects the number of the ECC error bits happened in the spare region. Only the values of the maximum error bits are recorded. To clear this register, write '1' to the ECC status clear register (Offset = 0x0028 11.2.5). When ECC for spare fails, the ECC status of the spare register will not be valid and the status of the spare region register may be incorrect.

Table 11-35 ECC Status of Spare Region Register

Bit	Name	Type	Reset	Description
31..7	Reserved	-	0	-
6..0	ECC_ERR_NO_SP	R	0	Number of the ECC error bits on spare region

11.2.7 Spare Region ECC Control Registers0/1

Offset: 0x0034 and 0x003C

Table 11-36 Spare Region ECC Control Register0 (0x0034)

Bit	Name	Type	Reset	Description
31..7	Reserved	-	0	-
6..0	ECC_THRES_BITS_SP	RW	0	Threshold number of the ECC error bits on spare region 0..15: 1bit to 16bits

Table 11-37 Spare Region ECC Control Register1 (0x003C)

Bit	Name	Type	Reset	Description
31..7	Reserved	-	0	-
6..0	ECC_CORR_BITS_SP	RW	0	This field sets the number of the ECC correction capability bits on spare region. 0..15: 1bit to 16bits

11.2.8 Device Busy/Ready Status Register

Offset: 0x0100

The busy/ready pins of a Flash device can be monitored by reading this register.

Table 11-38 Device Busy/Ready Status Register

Bit	Name	Type	Reset	Description
31..1	Reserved	-	0	-
0	DEV_BUSY_RDY	R	1	The busy/ready (busy_n) status of the Flash device 1: Flash is ready. 0: Flash is busy.

11.2.9 NANDC General Setting Register

Offset: 0x0104

This register provides the general settings for the NANDC module. This register can also be referenced by the controller that will not be changed when executing command.

Table 11-39 NANDC General Setting Register

Bit	Name	Type	Reset	Description
31..26	Reserved	-	0	-
25..24	CE_NUM	RW	2	Number of CEs connected with Flash in a channel 00: 1 CE 01: 2 CEs 10: 4 CEs 11: 8 CEs Notes: • The number of Flashes used in a channel must be power of 2. • If two CEs are at the CE0 and CE2 positions, ce_num must be set to four CEs.
23..15	Reserved	-	0	-
14..12	BUSY_RDY_LOC	RW	6	Device busy/ready status bit location on the Flash data bus “000“ - “111”: Bit0 - Bit7
11	Reserved	-	0	-
10..8	CMD_STS_LOC	RW	0	Command pass/fail status bit location on the Flash data bus “000“ - “111”: Bit0 - Bit7
7..3	Reserved	-	0	-
2	WR_PROT_EN	RW	0	Flash write protect pin control 1: Enable the write protect 0: Disable the write protect
1	DATA_INV_EN	RW	0	Data inverse control The inverted targets include the ECC parity and data. 1: Enable 0: Disable

Bit	Name	Type	Reset	Description
				The inverse function is useful when reading data from an erased page in Flash. If the inverse function is not enabled, all the read data bits will be '1' and the ECC parity bits will also be '1', which will induce an ECC parity error. If the inverse function is enabled, all data will be '0' and the ECC parity bits will be '0'. All '0' data and parity can pass the ECC parity check without errors.
0	Scrambler_en	RW	0	Data scrambler The scrambling operation only affects data. 1: Enable 0: Disable The scrambler adopts a formula to inverse data in random pseudo. The scramble seed depends on the sector number and row address executing to Flash.

11.2.10 Memory Attribute Setting Register1

Offset: 0x0108

This register can also be referenced by the controller that will not be changed when executing command.

Table 11-40 Memory Attribute Setting Register1

Bit	Name	Type	Reset	Description
31..22	Reserved	-	0	-
21..19	BI_BYTE	RW	0	Reserved BI byte from spare location 0: disable BI function 1: reserved 1st byte of spare location 2: reserved 2nd byte of spare location 3: reserved 3rd byte of spare location 4: reserved 4th byte of spare location 5: reserved 5th byte of spare location 6: reserved 6th byte of spare location 7: Reserved Note1: if BI_byte is enable, no_ecc_parity must be set 0 Note2: BI_byte is only valid at data/spare read/write flow
18..16	PAGE_SIZE	RW	0	Page size of Flash "000": 512 byte (Small page) "001": 2K bytes "010": 4K bytes "011": 8K bytes "100": 16K byte Other : Reserved Note: The small page fixed flow is only used for the small page setting.
15	Reserved	-	0	-
14..13	ROW_CYC	RW	0	Flash row address cycles "00": One cycle "01": Two cycles

Bit	Name	Type	Reset	Description
				”10”: Three cycles ”11”: Reserved
12	COL_CYC	RW	0	Flash column address cycles ‘0’: One cycle ‘1’: Two cycles
11..2	BLOCK_SIZE	RW	0	Block size (Row address occupied by one block) 0x0-0x3FF: 1page - 1024pages. Note: block_size must be power of 2 order and the block size must be more than or equal to valid_page_num (0x10C, 0). The block size can only be 32, 64, 128, 256, 512, or 1024 pages.
1..0	Reserved	-	0	-

11.2.11 Memory Attribute Setting Register2

Offset: 0x010C

This register can also be referenced by the controller that will not be changed when executing command.

Table 11-41 Memory Attribute Setting Register2

Bit	Name	Type	Reset	Description
31..26	Reserved	-	0	-
25..16	VALID_PAGE_NUM	RW	0x0FF	Number of the valid pages in a block 0x0 – 0x3FF: 1 page - 1024 pages For some TLC Flash, the number of the valid pages in a block will be 192 and the address occupied by one block will be 256 pages. The last 64 pages are blank and cannot be accessed. This register must be less than or equal to block_size (0x108, 11.2.10). If valid_page_num is less than the block size (0x108), the start row address should not be more than valid_page_num. At SLC mode and TLC mode, this register may be different, user must care this. For example, 64 valid page at Samsung SLC mode but 192 valid page at Samsung TLC mode. If user execute SLC command and TLC command, must set this register before executing command. If user issue 3SLC copy to TLC mode command, this register is based on SLC mode.
15	DQS_CLK_OUT_EN	RW	1	DLL reference clock enable bit 1: Enable (Default) 0: Disable
14	VREF	RW	0	External voltage enable 1: Enable 0: Disable If this bit is set to ‘1’, the external VREFQ will be used as a reference for the input and I/O signals. If this bit is set to ‘0’, the internal VREFQ will be used as a reference for the input and I/O signals.
13	DQS_C	RW	0	DQS complementary signal(DQS_c) enable 1: Enable 0: Disable

Bit	Name	Type	Reset	Description
				If this bit is set to '1', the complementary DQS (DQS_c) signal will be enabled. If this bit is cleared to '0', the complementary DQS (DQS_c) signal will not be used.
12	RE_C	RW	0	RE complementary signal (RE_c) enable 1: Enable 0: Disable If this bit is set to '1', the complementary RE_n (RE_c) signal will be enabled. If this bit is cleared to '0', the complementary RE_n (RE_c) signal will not be used.
11..0	Reserved	-	0	-

11.2.12 AC Timing Register0 of NANDC

Offset: 0x0110

The AC timing registers 0 and 1 are used to decide the clock cycles that will be issued by a specific signal pulse. The reference clock is mem_clk (192MHz) as shown in Figure 11-8 through Figure 11-10. This register can also be referenced by the controller that will not be changed when executing command.

Table 11-42 AC Timing Register0 of NANDC

Bit	Name	Type	Reset	Description
31..28	Reserved	-	0	-
27..24	t_{WH}	RW	0xF	Reference summary of AC timing usage
23..21	Reserved	-	0	-
20..16	t_{WP}	RW	0xF	Reference summary of AC timing usage
15..12	Reserved	-	0	-
11..8	T_{REH}	RW	0xF	Reference summary of AC timing usage
7..5	Reserved	-	0	-
4..0	T_{RES}	RW	0xF	Reference summary of AC timing usage

Figure 11-8 Timing Diagram of Address State

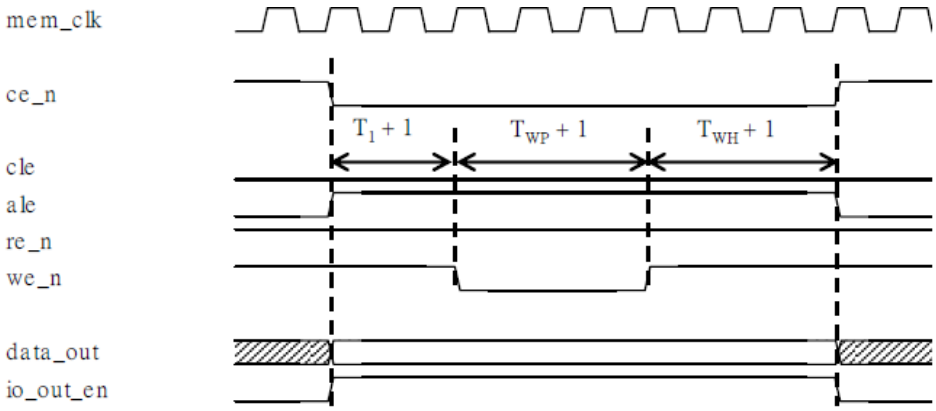


Figure 11-9 Timing Diagram of Command State

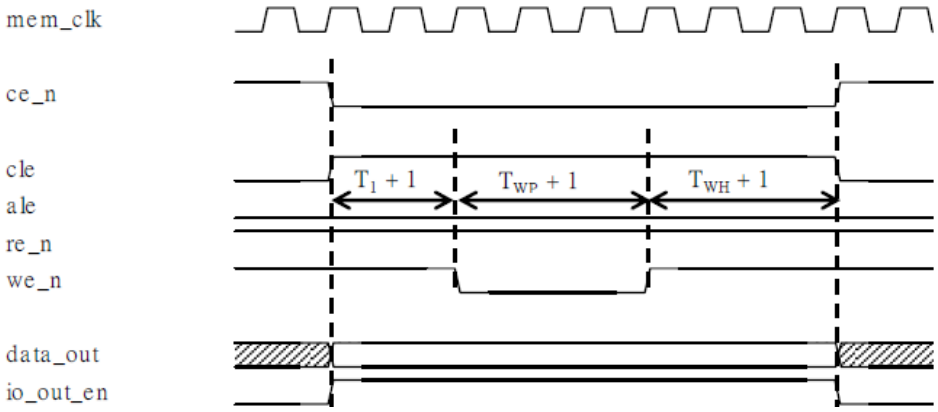
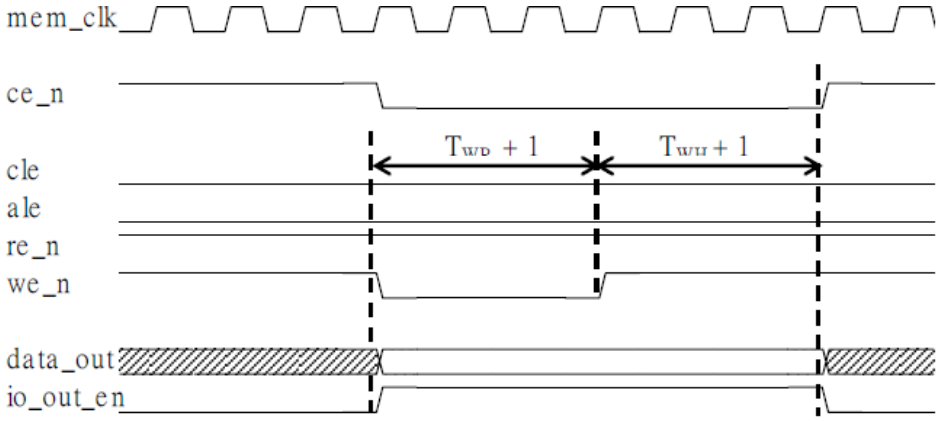


Figure 11-10 Timing Diagram of Write Data State



11.2.13 AC Timing Register1 of NANDC

Offset: 0x0114

This register can also be referenced by the controller that will not be changed when executing command.

Table 11-43 AC Timing Register1 of NANDC

Bit	Name	Type	Reset	Description
31..22	Reserved	-	0	-
21..16	t_{RLAT}	RW	0x0	Reference summary of AC timing usage
15	Reserved	-	0	-
14..8	t_{BSY}	RW	0x7F	Reference summary of AC timing usage
7	Reserved	-	0	-
6..0	t1	RW	0x7F	Reference summary of AC timing usage

Figure 11-11 Timing Diagram of Read Data State

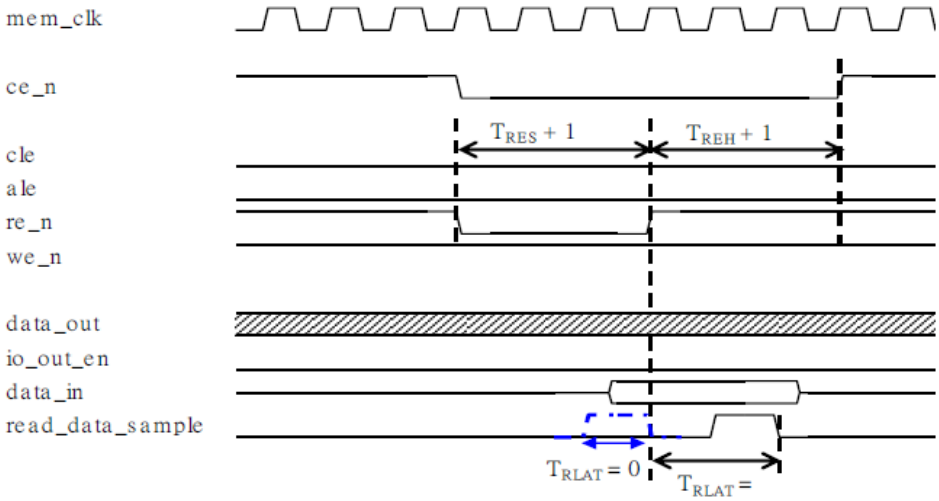
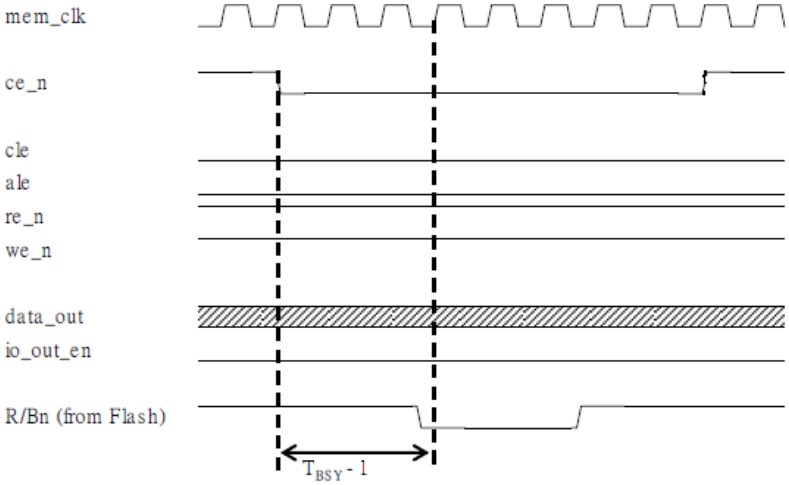


Figure 11-12 Timing Diagram of Busy State



11.2.14 AC Timing Register2 of NANDC

Offset: 0x190

This register can also be referenced by the controller that will not be changed when executing command.

Table 11-44 AC Timing Register2 of NANDC

Bit	Name	Type	Reset	Description
31	Reserved	-	0	-
30..24	t_{BUF4}	RW	0x7F	Reference summary of AC timing usage
23	Reserved	-	0	-
22..16	t_{BUF3}	RW	0x7F	Reference summary of AC timing usage
15	Reserved	-	0	-
14..8	t_{BUF2}	RW	0x7F	Reference summary of AC timing usage
7	Reserved	-	0	-
6..0	t_{BUF1}	RW	0x7F	Reference summary of AC timing usage

Notes:

- t_{BUF4}, t_{BUF3}, t_{BUF2}, t_{BUF1}at the fixed flow ACtiming are described as below:
- t_{BUF4} is WE high to RE low timing.
  Max. (tWHR, tWHR2)
- t_{BUF3} is Max. (RE high to WE low, RE high to output Hi-Z, W/Rn high to DQS/DQ tri-state by device)
  Max. (tRHW, tRHZ, tDQSHZ)
- t_{BUF2} is busy high to RE low timing.
  Max. (tAR, tRR, tCLR, tCDQSS, tCRES, tCALS, tCALS2, tDBS)
- t_{BUF1} is command phase to WE high phase.
  Max. (tADL, tCCS, tCWAW)

Table 11-45 Summary of AC Timing Usage

Item	Asynchronous IF
t_{WH}	max(t_{WH}, t_{CH}, t_{CLH}, t_{ALH})
t_{WP}	t_{WP}
t_{REH}	t_{REH}
t_{RES}	max(t_{REA}, t_{RP}, t_{RSTO}, t_{REAID}) t_{RP} when EDO mode
t_{BSY}	t_{WB}, t_{RB} The min value is 1
t_{BUF1}[2]	max(t_{WHR}, t_{WHR2}, t_{RHW}, t_{RR}, t_{AR}, t_{DBS}, t_{CALS}, t_{CALS2}, t_{ADL}, t_{RHZ}, t_{CDQSS}, t_{CCS}, t_{CRES}) The min value is 1
t_{BUF2}	
t_{BUF3}	
t_{BUF4}	
t_{RLAT}	Internal latch timing
t_1	max(t_{CS}, t_{CLS}, t_{ALS}) - t_{WP}

Notes:
1. The ONFi 2.2 protocol includes the asynchronous IF.
2. The "legacy protocol" means "asynchronous IF"

11.2.15 NANDC Interrupt Enable Register

Offset: 0x0150

The interrupt of the NAND-Controller will be asserted when the interrupt enable register is set and corresponding event happens. The event status can be checked by reading the NANDC Interrupt Status Register (Offset = 0x0154, 11.2.16)

Table 11-46 NANDC Interrupt Enable Register

Bit	Name	Type	Reset	Description
31..1	Reserved	-	0	-
0	STS_FAIL_INT_EN	RW	0	Flash status check fail interrupt enable 1: Enable 0: Disable

11.2.16 NANDC Interrupt Status Register

Offset: 0x0154

The NAND-Controller interrupt can be asserted by using one of the following three NANDC events:

[2] When the error handling/abort occurs, t_{BUF1} will be used to idle I/Fs to avoid the signal conflict.

- Auto-pattern compare failed
- NANDC command completed
- Status check failed

Table 11-47 NANDC Interrupt Enable Register

Bit	Name	Type	Reset	Description
31..25	Reserved	-	0	-
24	AUTO_CMP_PAT_FAIL	RW	0	Auto pattern compare fail
23..17	Reserved	-	0	-
16	NANDC_CMD_CMPLT	RW	0	NANDC command complete
15..1	Reserved	-	0	-
0	STS_FAIL	RW	0	Status check fail

Notes:
1. The failure of the auto pattern compare will always trigger the global interrupt.
2. The NANDC command complete interrupt enable is set in the command queue.
3. When auto_cmp_pat_fail occurs, nandc_cmd_cmplt will not be asserted.

11.2.17 Current Access Row Address Channel

Offset: 0x0158

The current access row address represents the row address that is issued by NANDC. These registers will be updated once NANDC issues a new row address.

Table 11-48 Current Access Row Address Channel

Bit	Name	Type	Reset	Description
31..0	ROW_ADDR_CH	R	0	Current access of the row address

11.2.18 Read Status Register

Offset 0x0178

NANDC stores the status returned from Flash in the register

Table 11-49 Read Status Register

Bit	Name	Type	Reset	Description
31..8	Reserved	-	0	-
7..0	READ_STS_CH0	R	0	Flash status

11.2.19 Address Toggle Bit Location Register

Offset 0x0180

The row address may need to be toggled for one-bit in order to perform the Flash two-plane access. This register can be set to determine which bit is toggled during the MicroCode of tog_1st_addr, tog_2nd_addr, tog_3rd_addr, and tog_targ_1st_addr.

Table 11-50 Address Toggle Bit Location Register

Bit	Name	Type	Reset	Description
31..5	Reserved	-	0	-
4..0	TOGGLE_BIT_LOC	RW	0	Toggle bit location 0-23: Bit0-Bit23 of the row addresses

11.2.20 NANDC Software Reset Register

Offset 0x0184

The NANDC and ECC can be reset by setting the NANDC software reset register. NANDC will return to the idle state when performing reset. The corresponding command queue will not pop and will be re-executed immediately after reset. If users issue this register to reset NANDC, the operation of flash will be suddenly stopped by the controller and may result unexpected behavior to flash. Therefore, to abort command, please follow the abort sequence (Please refer to 11.1.9) to complete the abort flow. No register value will be reset while issuing the software reset to NANDC.

Table 11-51 NANDC Software Reset Register

Bit	Name	Type	Reset	Description
31..1	Reserved	-	0	-
0	NANDC_SW_RST	RW	0	NANDC software reset register This reset will be cleared when the corresponding NANDC finishes the reset operation. This field must be checked to be '0' before issuing the new NAND commands. Write 1: Reset Write 0: No effect

11.2.21 NANDC Auto-compare Pattern Register

Offset: 0x018C

NANDC can perform a write command with specific patterns and read commands to be compared with the specific pattern recorded in this register. The data region threshold for the blanking check located offset (0x0010, 11.2.2) for different channels. The spare region threshold for the blanking check located offset (0x0034, 11.2.7) for the channel.
The data after scrambler is the blanking write data when scrambler enable (Offset 0x104, 11.2.9), and the scrambled seed is the row address [13:0]. When the seed is zero, the seed will be 0x2AAA.

Table 11-52 NANDC Auto-compare Pattern Register

Bit	Name	Type	Reset	Description
31..0	AUTO_CMP_PAT	RW	0	Auto-compare pattern Currently, only used bit[7:0], other bits can access but don't effect the function

11.2.22 Variable Address Register

Offset: 0x01D0

This register is used for microcode var_addr1, var_addr2, var_addr3 and var_addr4, respectively.

Table 11-53 Variable Address Register

Bit	Name	Type	Reset	Description
31..24	VAR_B4	RW	0	Variable 4th byte
23..16	VAR_B3	RW	0	Variable 3rd byte
15..8	VAR_B2	RW	0	Variable 2nd byte
7..0	VAR_B1	RW	0	Variable 1st byte

11.2.23 Command Queue Status Register

Offset: 0x0200

The command queue status includes:

- Command queue is full.
- Command queue is empty.

Before pushing a command into queue, please check the queue to make sure that it is not full.

Table 11-54 Command Queue Status Register

Bit	Name	Type	Reset	Description
31..9	Reserved	-	0	-
8	CMDQ_FULL	R	0xF	Command queue full status 1: Full 0: Not full
7..1	Reserved	-	0	-
0	CMDQ_EMPTY	R	0xF	Command queue empty status 1: Empty 0: Not empty

11.2.24 Command Queue Flush Register

Offset: 0x0204

The command queue can be flushed by writing ‘1’ to the specific flush register. The corresponded NANDC and ECC are also reset during setting the command queue flush.

Table 11-55 Command Queue Flush Register

Bit	Name	Type	Reset	Description
31..1	Reserved	-	0	-
0	CMDQ_FLUSH	W	0	Command queue flush 1: Write 1 to flush 0: Write 0 has no effect.

11.2.25 Command Complete Counter

Offset: 0x0208

When one command completes, the counter will increase by 1. By checking this register, the number of the command completion can be known. Please note that cmd_cmplt_cnt contains a maximum of seven cmd_cmplt. If cmd_cmplt is more than seven cmd_cmplt and cmd_cmplt is not cleared, cmd_cmplt_cnt will overflow.

Table 11-56 Address Toggle Bit Location Register

Bit	Name	Type	Reset	Description
31..3	Reserved	-	0	-
2..0	TOGGLE_BIT_LOC	R	0	Command complete counter

11.2.26 Command Complete Counter Reset Register

Offset: 0x020C

The command completion counter can be reset by writing ‘1’ to this register

Table 11-57 NANDC Software Reset Register

Bit	Name	Type	Reset	Description
31..1	Reserved	-	0	-
0	CMD_CMPLT_CNT_RST	W	0	Command complete counter reset Write 1: Clear the counter Write 0: No effect

11.2.27 General Command Queue Register Access

Offset: 0x0280..0x294

Writing to this command register will push the command to all the command queues. Reading from this command register will get all zero returns. Please note that this register cannot be set if one or more command queues are full. The general command queue can

only be pushed with a command that contains no data transfer from or to BMC (For example: When block erase, reset, and spare read/write are acceptable).

11.2.28 Command Queue1-4 Register

Offset: 0x0300..0x030C

The operation of NANDC is decided by the command pushed into the command queue. One command contains four words for basic flow. The command queue may have one, two or four entries of space for keeping the command. A command will be pushed into the command queue by writing the forth word of a command and will pop from the command queue when it is completely executed by the NANDC controller.
During reading the command queue register, the returned data will be the commands that are currently processed By NANDC but will not be the current command to be written to the command queue.

Table 11-58 Command Queue1 Register

Bit	Name	Type	Reset	Description
31..24	Reserved	-	0	-
23..0	ROW_ADDR_1st	RW	0	The first row address/source row address (Page index) The first row address decides the source address in a copy-back or a blanking check flow.

Table 11-59 Command Queue2 Register

Bit	Name	Type	Reset	Description
31..24	Reserved	-	0	-
23..0	ROW_ADDR_2nd	RW	0	The second row address/source row address (Page index) The second row address decides the second chip row address in an interleaving flow or decides the destination address in a copy-back or blanking check flow.

Table 11-60 Command Queue3 Register

Bit	Name	Type	Reset	Description
31..16	COUNTER	RW	0	Counter (For repeated operations) If the counter is used in a data transfer flow, the scale will show as below: If the ECC base is set to 1 kB, each count will represent 1 kB of data. If the ECC base is set to 512 bytes, each count will represent 512 bytes of data. The counter unit can be sector, page or block. Please refer to the “Command Register Setting” (Table 11-3). (0 is inhibited).
15..8	SEC_OFFSET_2nd	RW	0	The 2nd sector offset In the byte mode, this register decides the second column address (Issued with the first row address). Note: This value is only effective in the byte mode.

Bit	Name	Type	Reset	Description
7..0	SEC_OFFSET_1st	RW	0	The 1st sector offset This register decides the sector offset and hardware decode of this register to decide the first column address. In the byte mode, this register decides the first column address (Issued with the first row address).

Table 11-61 Command Queue4 Register

Bit	Name	Type	Reset	Description
31..29	START_CE	RW	0	Command starting CE This register decides the Flash from which the current command starts. 3'b000: CE0 3'b001: CE1 3'b010: CE2 3'b011: CE3 3'b100: CE4 3'b101: CE5 3'b110: CE6 3'b111: CE7
28	BYTE_MODE	RW	0	Byte mode In the byte mode, the spare value is updated from the spare register by channel. 1: Enable 0: Disable When executing the byte mode flow, data will not be protected by the ECC engine. The scramble and data inverse will not affect the byte mode flow.
27	BMC_IGNORE	RW	0	Ignore the BMC region status of full/empty (User mode) 1: Ignore 0: Do not ignore Please refer to 11.1.2 for more detailed information.
26..24	BMC_REGION_SEL	RW	0	BMC region selection Note: Only region0 is supported; 0 is the only valid value for this variable.
23..19	SPARE_NUM0-31	RW	0	Number of spare date byte Constraints: • In the byte mode: Users can program spare_num0-31, which is 1byte-32byte, respectively. • Not in the byte mode: Users can only program spare_num 3, 7, 15, 31, which is 4, 8, 16, 32 bytes for the ECC engine protect spare data.
18..8	CMD_INDEX	RW	0	Command index selection NANDC execute programming flow when bit[18] is set to '1'; otherwise, execute the fixed flow Bits[17:8] decide the starting position of the control flow. Please refer to the "Command Register Setting" (Table 11-3) for more detailed information.
7..5	FLASH_TYPE	RW	0	Support flash operation type "000": NV-SDR (Legacy Flash)
4	CMD_HSK_MODE	RW	0	Command handshake mode (Use the DMA handshake mode with the source DMA) This bit must be set '0' in the user mode.

Bit	Name	Type	Reset	Description
				1: Enable 0: Disable
3..2	CMD_SCALE	RW	0	Command incremental scale "11": Reserved "10": By two block "01": By one block "00": By page Notes: • cmd_scale must set to "00" in the Cache operation. • In the block erase flow, inc_by_page/blk must be set to in_by_blk. If the 2P block is erased or if the I2 block is erased, inc by_page/blk must be set to two block. • In a blanking check flow, in_by_page/blk must be set to in_by_page.
1	Reserved	-	0	-
0	CMPLT_INTR_EN	RW	0	Command complete interrupt and status enable 1: Enable 0: Disable Note: When this bit is set, users must clear the command complete interrupt and the host controller will execute the next command at the command queue. If the cmplt_intr_en bit is disabled and no command complete interrupt occurs, the host will automatically execute the next command. To know which command to be executed, please poll the command complete counter.

11.2.29 BMC Region Status Register

Offset: 0x0400

This register provides the full or empty status of the region. The region halt status can also be read from this register (Region halt occurs when an ECC uncorrectable error is encountered and this region is not used in the user mode). Once a region is halted, the region software reset will be required for solving the halt state. Except the region FIFO, the region contains a small buffer to temporarily keep the write data. Please note that the region buffer empty status must be '1' and the region full status must be '0' before setting the DMA Mode Write Data Fill Register (Offset = 0x0424, 11.2.30).

Table 11-62 BMC Region Status Register

Bit	Name	Type	Reset	Description
31..25	Reserved	-	0x7F	-
24	REGION_BUF_EMPTY	R	1	Region buffer empty status 1: Region buffer is empty. 0: Region buffer is not empty.
23..17	Reserved	-	0	-
16	REGION_HALT	R	0	Region halt status 1: Region is halted.

Bit	Name	Type	Reset	Description
				0: Region is not halted.
15..9	Reserved	-	0	-
8	REGION_FULL	R	0	Region full status 1: Region is full. 0: Region is not full.
7..1	Reserved	-	0x7F	-
0	REGION_EMPTY	R	1	Region empty status 1: Region is empty. 0: Region is not empty.

11.2.30 Region User Mode Pointer Adjustment Register

Offset: 0x0404

When the ignore bit is set in a command, the operation will enter the user mode. The BMC region will be used as a memory but not as a FIFO. By setting this register, the read and write positions can be adjusted.
In the user mode, one BMC region contains 32 grids. The size of each grid is 512 bytes. The pointers should be adjusted to the correct position every time before issuing a user-mode command. When processing a user-mode command with 1 kB per sector, the pointer must be set with an even value (e.g. 0, 2, 4, …, and so on).
Because each region only contains 16K bytes space, the length of one user mode read/write operation should be keep less than 16K bytes (Pointer offset + read/write sector count * n ≤ 32, n = 1 for 512 byte per sector; n = 2 for 1 kB per sector; ..).

Table 11-63 Region User Mode Pointer Adjustment Register

Bit	Name	Type	Reset	Description
31..13	Reserved	-	0	-
12..8	REGION_UM_WRPTR	W	0	Region user mode write pointer (For the NAND Flash read command)
7..5	Reserved	-	0	-
4..0	REGION_UM_RDPTR	W	0	Region user mode read pointer (For the NAND Flash write command)

11.2.31 DMA Mode Write Data Fill/Read Data Pop Register

Offset: 0x0424

By writing to this register once (Write ‘1’ and cleared by itself), one 512 bytes data (Dummy data) will be filled into or popped from the region. This is useful in Flash with 1 kB sector and only the 512 bytes data is required to be written to or read from the Flash. The pop or fill operation must be executed only once before the movement of the read/write data of the DMA mode or after all read/write data are transferred to/from AHB.

region_buf_empty should be ‘1’ and region_full should be ‘0’ before setting the fill_data register. The fill_data register is not allowed to be set during executing the NAND Flash read command.
region_empty should be ‘0’ before setting the pop_data register. The pop_data register is not allowed to be set during executing the NAND Flash write command.

Table 11-64 Command Queue1 Register

Bit	Name	Type	Reset	Description
31..9	Reserved	-	0	-
8	POP_DATA	W	0	This signal pops 512 bytes of data from the region (Used only in the DMA mode). Write 1: Pops 512 bytes of data Write 0: No effect
7..1	Reserved	-	0	-
0	FILL_DATA	W	0	This signal fills 512 bytes of data into the region (Used only in the DMA mode). Write 1: Fills 512 bytes of data Write 0: No effect

11.2.32 Region Software Reset Register

Offset: 0x0428

The region can be reset by writing to this register.

Table 11-65 Region Software Reset Register

Bit	Name	Type	Reset	Description
31..1	Reserved	-	0	-
0	REGION_SW_RST	W	0	Region software reset (This reset is cleared by itself.) Write 1: Reset Write 0: No effect

11.2.33 Force Region Fill Read Data Register

Offset: 0x042C

By setting this register to ‘1’, the region will enter a non-empty state and return meaningless data to a requested AHB master. This register can only be set when the region is not in the DMA handshake mode. (A region in the DMA handshake mode means that the command processed by a NANDC is with the DMA handshake mode enabled).
This register is useful in solving the condition that a DMA master reads data from a region in which an ECC uncorrectable event happens. Because the uncorrectable data cannot be read from this region, this bit must be set to prevent the DMA master from hanging.

Table 11-66 Force Region Fill Data Register

Bit	Name	Type	Reset	Description
31..1	Reserved	-	0	-
0	FORCE_FILL_RD	RW	0	Force region fill read data 1: Enable 0: Disable

11.2.34 Region Remaining Sector Count of Read Data Register

Offset: 0x0430

This register represents the remaining data in one region. The counting unit is in Sector (512 bytes or 1K bytes). Once an ECC uncorrectable error happens, the corresponding region will be halted in the uncorrected sector. The remaining sector counts can be read from this register.

Table 11-67 Region Remaining Sector Count of Read Data Register

Bit	Name	Type	Reset	Description
31..17	Reserved	-	0	-
16..0	REGION_RD_CNT	R	0	Remaining sector count of the read data in region

11.2.35 AHB Slave Memory Space Range Register

Offset: 0x0508

The range of the memory space of the region can be set from 512 bytes to 64K bytes in this register.

Table 11-68 AHB Slave Memory Space Range Register

Bit	Name	Type	Reset	Description
31..12	Reserved	-	0x0280F	-
11..9	Reserved	-	111	-
8	AHB_RETRY_EN	RW	1	AHB bus retry protocol enable 1: Enable 0: Disable
7..0	AHB_MEM_RANGE	RW	0x01	AHB slave memory space range 8'b0000_0001: 512 bytes 8'b0000_0010: 1K bytes 8'b0000_0100: 2K bytes 8'b0000_1000: 4K bytes 8'b0001_0000: 8K bytes 8'b0010_0000: 16K bytes 8'b0100_0000: 32K bytes 8'b1000_0000: 64K bytes

11.2.36 Global Software Reset Register

Offset: 0x050C

Writing '1' to this register resets all the modules in NAND-Controller, except for the asynchronous AHB data slave and the following registers:

- ECC control related register
- NANDC control-related register ("NANDC Auto Compare Pattern Register" is cleared.)
- Spare register
- Programmable flow control register
- Programmable OPCODE register

Global software reset will reset the following registers:

- ECC interrupt status of data and spare region
- ECC threshold interrupt status of data and spare region
- ECC error count of data and spare region
- NAND controller reset
- ECC engine reset
- BMC reset
- Programming/erase status
- Auto compare patterns and auto compare fail status
- Fill RD register

After the above registers are set, the NANDC software reset register must be '0' before further operations.

Table 11-69 AHB Slave Memory Space Range Register

Bit	Name	Type	Reset	Description
31..1	Reserved	-	0	-
0	GLB_SW_RST	W	0	Global software reset Write1: Reset Write0: No effect

11.2.37 ECC Correction Capability Register1

Offset: 0x0514

Table 11-70 ECC Correction Capability Register1

Bit	Name	Type	Reset	Description
31	ECC_CORR_32BIT_EN	R	0	ECC correction 32 bit
..	..	..	..	..
1	ECC_CORR_2BIT_EN	R	0	ECC correction 2 bit
0	ECC_CORR_1BIT_EN	R	0	ECC correction 1 bit

11.2.38 ECC Correction Capability Register2

Offset: 0x0518

Table 11-71 ECC Correction Capability Register2

Bit	Name	Type	Reset	Description
31	ECC_CORR_64BIT_EN	R	0	ECC correction 64 bit
..	..	..	..	..
1	ECC_CORR_34BIT_EN	R	0	ECC correction 34 bit
0	ECC_CORR_33BIT_EN	R	0	ECC correction 33 bit

11.2.39 ECC Correction Capability Register3

Offset: 0x051C

Table 11-72 ECC Correction Capability Register3

Bit	Name	Type	Reset	Description
31..10	Reserved	-	0	-
9	ECC_CORR_74BIT_EN	R	0	ECC correction 74 bit
..	..	..	..	..
1	ECC_CORR_66BIT_EN	R	0	ECC correction 66 bit
0	ECC_CORR_65BIT_EN	R	0	ECC correction 65 bit

11.2.40 Programmable OPCODE Register

Offset: 0x0700..0x0704

The OPCODE can be set in this register.

Table 11-73 Programmable OPCODE Register

Bit	Name	Type	Reset	Description
31..24	OPCODE_3	RW	0	Programmable OP-code byte 3
23..16	OPCODE_2	RW	0	Programmable OP-code byte 2
15..8	OPCODE_1	RW	0	Programmable OP-code byte 1
7..0	OPCODE_0	RW	0	Programmable OP-code byte 0

Table 11-74 Programmable OPCODE Register

Bit	Name	Type	Reset	Description
31..24	OPCODE_7	RW	0	Programmable OP-code byte 7
23..16	OPCODE_6	RW	0	Programmable OP-code byte 6
15..8	OPCODE_5	RW	0	Programmable OP-code byte 5

Bit	Name	Type	Reset	Description
7..0	OPCODE_4	RW	0	Programmable OP-code byte 4

11.2.41 Spare Access Register

Offset: 0x1000..0x10FF

The spare data can be read or written by accessing this register. The Flash ID is also read from this register. The channel contains 32 bytes spare data.

Spare register offset is from 0x1000-0x101F.

11.2.42 Programmable Flow Control Register

Offset: 0x2000..0x20FF

Users can program the Programmable Flow Control Register for flash flows that are not supported by the fixed flow. If the command index (Offset 30C bits[18:8], Table 11-61) bit[18] is set to '1', the programming flow by Programmable Flow Control Register will be executed, and bits[17:8] will be the address of the Programmable Flow Control Register address. Programmable Flow Control Register will be 128 bytes or 256 bytes by hardware configuration.

11.2.43 Data SRAM Access Port Register

Offset: 0x2_0000..0x2_3FFF

The data in BMC FIFO can be read or written by accessing this register.

11.3 Initial Steps

Although the NAND-Controller does not require specific initialization flow, a number of points are required to be checked before accessing Flash:

- Enable ECC
- Enable the data inverting and scramble functions
- The AC timing must be correctly set based on the current NAND clock speed and requirement of Flash.
- The memory attribute must be correctly set based on the specification of Flash.

11.4 Command Queue Access Method

The NAND-Controller provides a command queue. Each Flash can be accessed by issuing a command into the command queue. The 1st word to the 3rd word and 5th, 6th word can be arbitrarily changed before writing the 4th word. After writing the 4th word, a command will be pushed into the command queue.

Please note the following considerations:

- Always check the command status (should not be full) before preparing a new command (preparing a new command that includes writing from the 1st word to the 3rd word and 5th, 6th word)
- By using the general command queue, the command content will be the same to be written to all the command queues. With the possibility of leading to different NAND channels mapped to the same BMC region, the general command queue should not be used by a data read/write command that needed to be pushed into the command queue.

12 QuadSPI Channel

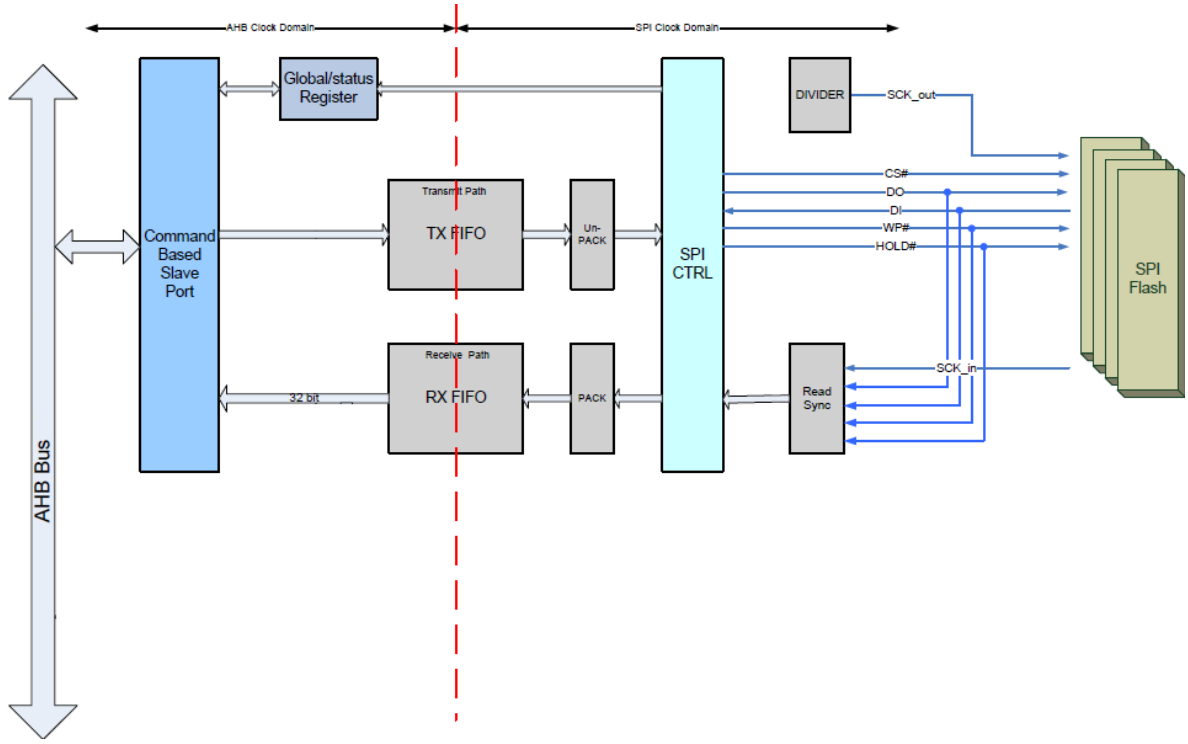
Offset: 0xF2000000

The ANTAIOS includes a QuadSPI-Controller. The maximum clk-frequency is 96 MHz and could be divided into lower values. It supports SPI serial, dual and quad mode. Supported SPI-commands are:

- Write enable
- Read status
- Write status
- Page program
- Quad page program
- Sector erase
- Block erase
- Chip erase
- Read data
- Fast read
- Fast read dual output
- Fast read quad output
- Fast read dual IO
- Fast read quad IO
- Read ID
- Fast DT read
- Dual IO DT read
- Enable write status
- Continuous read

12.1 Overview

Figure 12-1 QuadSPI Controller



The QuadSPI-Controller contains one command-based slave port and one SPI interface controller to execute the SPI Flash command. The command-based slave port is used to access the register and data port by a standard AMBA AHB bus. Moreover, the QuadSPI-Controller also provides the PIO mode to access the data from the AHB data port.

12.1.1 Command-based Slave Port

The QuadSPI-Controller has one command-based slave port, including the register and data port. Users can access the register or data port through this slave port. Command queue must be accessed by the WORD size; however, other registers, including the data port, can be access by the BYTE, HALFWORD, and WORD size. Users must issue the fixed address when accessing the data port.

12.1.2 PIO Mode

Users can access the data port via the PIO mode or in DMA mode over an external DMA. In the PIO mode, users must poll the RXFIFO/TXFIFO ready register (0x18).When RXFIFO is ready, users can read the whole RXFIFO data or remain data of RXFIFO out. When TXFIFO is ready, users can write the whole data into TXFIFO. Whenever operating in the PIO mode, users have to make sure to poll the status.

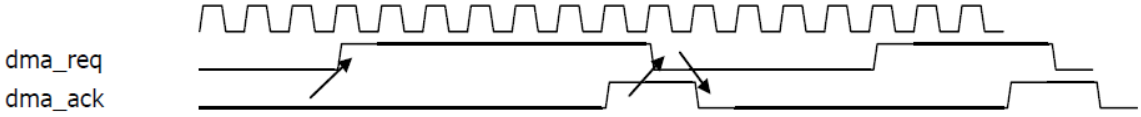
12.1.3 DMA Mode

The DMA mode is another option to access the data port. Users must set the external DMA controller register to the DMA handshake mode and set the burst size of one handshake.

The burst size of DMA depends on the QuadSPI-Controller RX/TX threshold trigger level (0x20).

If the RX/TX threshold trigger level is ‘8’(Unit: WORD), one DMA burst size must be 8 * 4 = 32 bytes for the DMA transfer width. If the RX/TX threshold trigger level is ‘8’(Unit: WORD), one DMA burst size must be 8 * 1 = 8 words for the DMA transfer width. If the RX/TX threshold trigger level is ‘8’(Unit: WORD), one DMA burst size must be 8 * 2 = 16 half word for the DMA transfer width. The RX/TX threshold trigger unit is WORD. The host controller will issue a DMA request by reaching the threshold trigger level of TX/RX FIFO.

Figure 12-2 DMA Handshake Mode

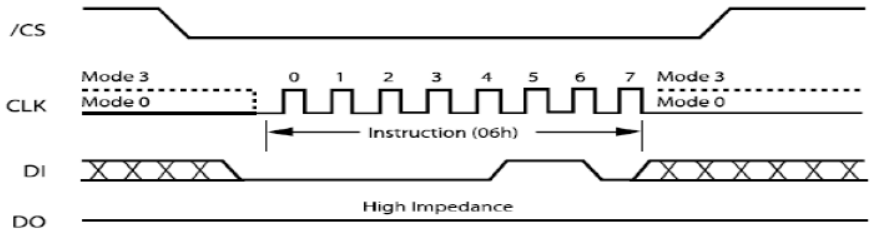


12.1.4 SPI Flash Operation

12.1.4.1 Serial Mode

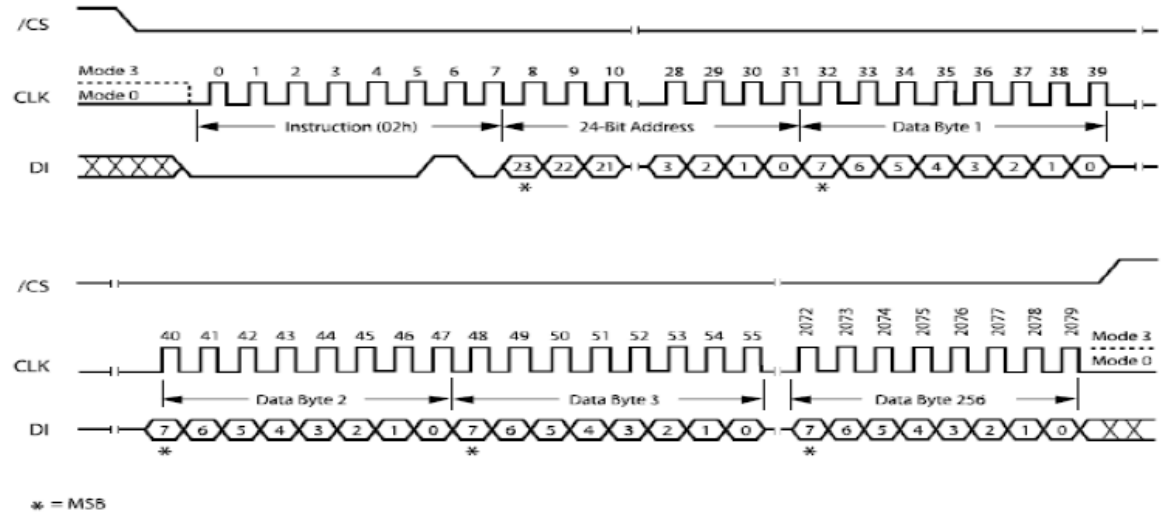
The host controller supports the SPI protocols, mode 0 and mode 3. Users must set the command queue register based on the specification of the SPI Flash. Please refer to the waveforms shown in Figure 12-3 through Figure 12-11 to set the command queue.
If users want to execute the write enable command, please set the instruction code to 0x06, write enable, and ‘1’ for the instruction length. Please refer to Figure 12-3 for more details.

Figure 12-3 Write Enable



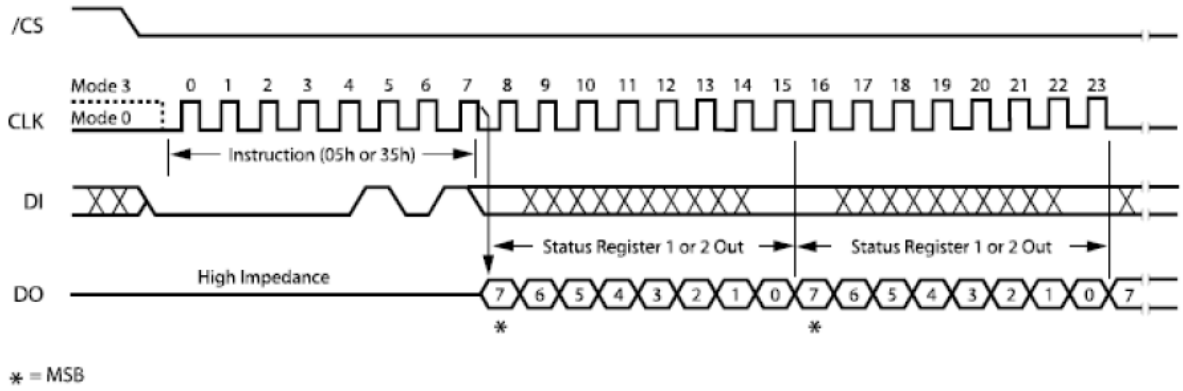
If users want to execute the page programming command, please set the instruction code to 0x02, address register, address length, write enable, and ‘1’ for the instruction length.

Figure 12-4 Page Program



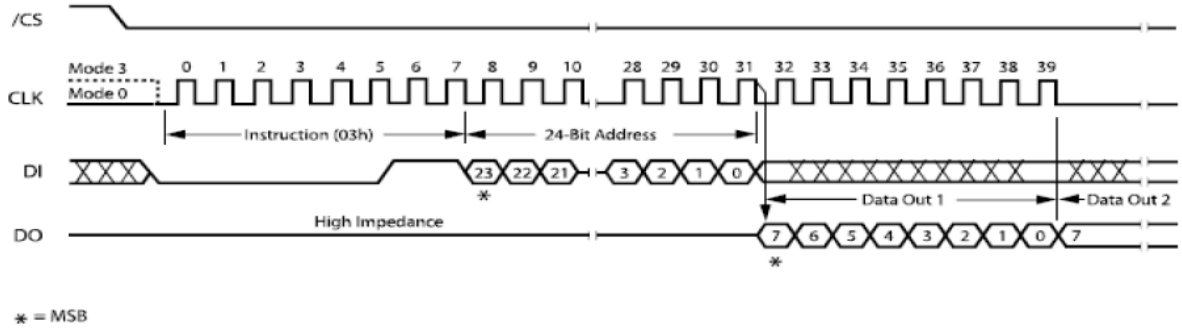
If users want to execute the read status command, please set the instruction code to 0x05/0x35, read status enable, optional read status by hw/sw, and '1' for the instruction length. Please refer to Figure 12-5 for more details.

Figure 12-5 Read Status



If users want to execute the read data command, please set the instruction code/length to 1 byte, address/address length to 3 bytes, and write enable to '0'. Please refer to Figure 12-6 for more details.

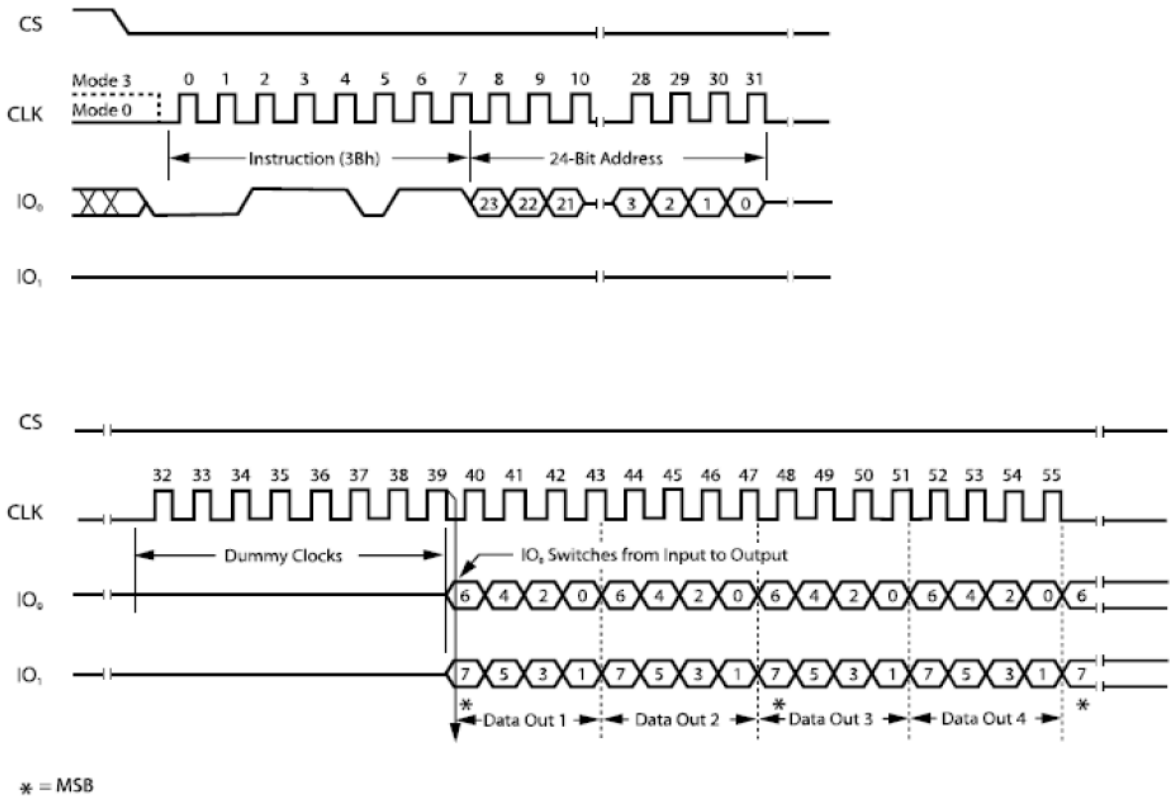
Figure 12-6 Read Data



12.1.4.2 Dual Mode/Dual_IO Mode

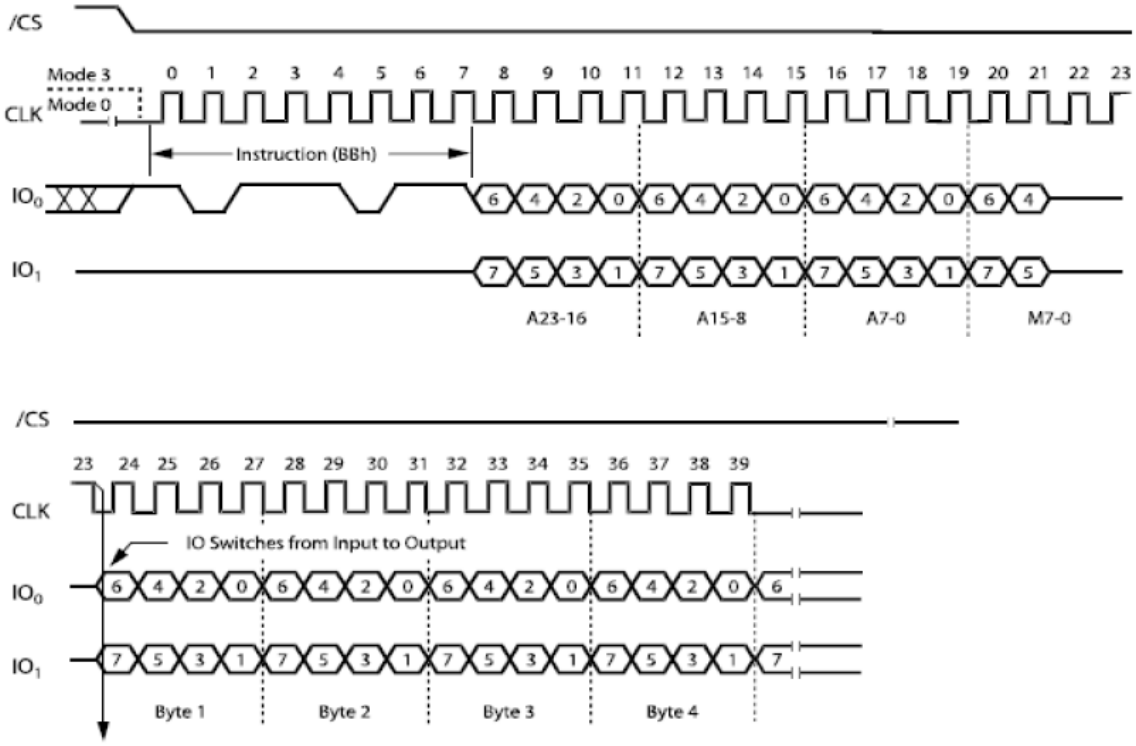
If users want to execute the fast read dual output command, please set the instruction code/length to ‘1’, address/address length to 3 bytes, dummy second cycle to 8, and operation mode to dual mode. Please refer to Figure 12-7 for more details.

Figure 12-7 Fast Read Dual Output Command



If users want to execute the fast read dual I/O, please set the instruction of code/length, address/address length, continuous read mode to enable, continuous read mode code, and operation to dual_io mode. Please refer to Figure 12-8 for more details.

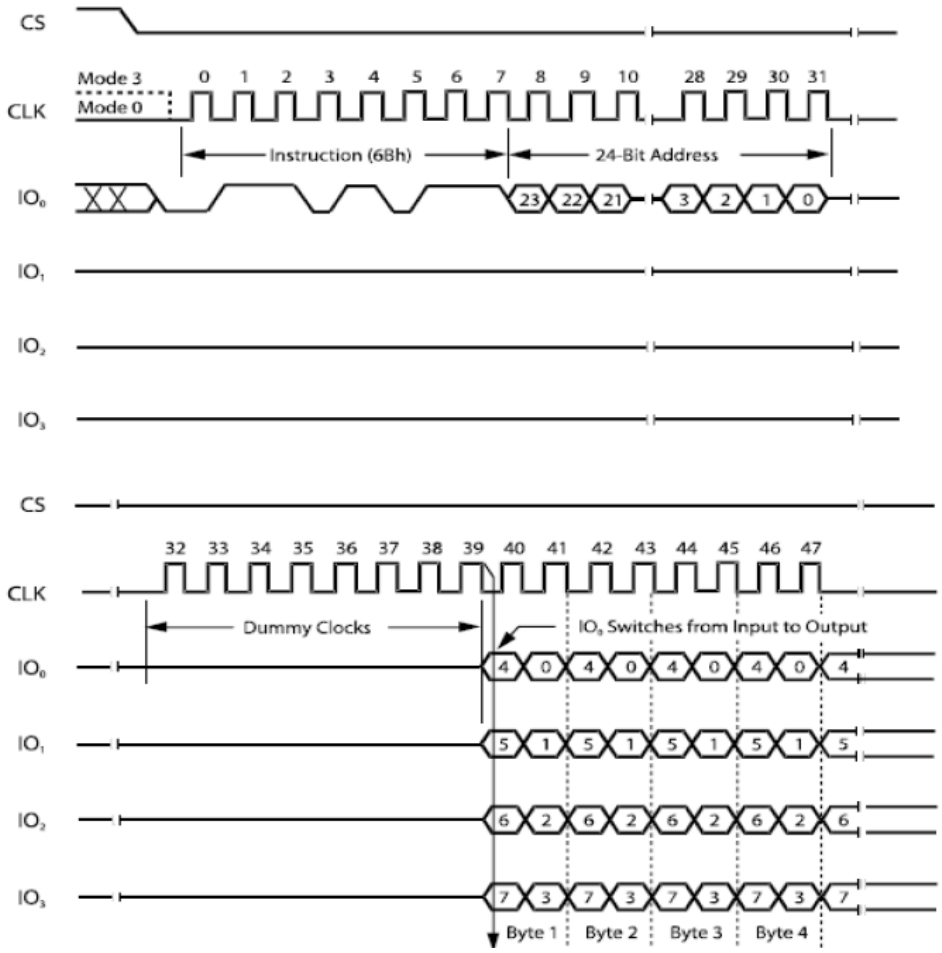
Figure 12-8 Fast Read Dual I/O



12.1.4.3 Quad Mode/Quad_IO Mode

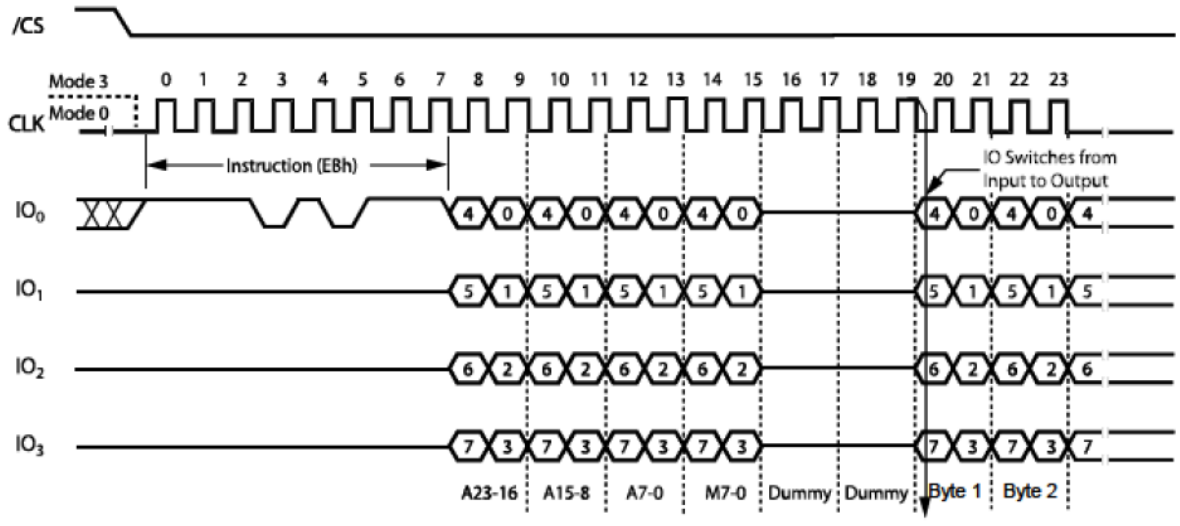
If users want to execute the fast read quad output, please set the instruction code/length, address/address length, dummy second cycle, and operation mode to the quad mode. Please refer to Figure 12-9 for more details.

Figure 12-9 Fast Read Quad Output



If users want to execute the fast read quad I/O command, please set the instruction code/length, address/address length, 4-cycle dummy second, continuous read mode/code, and operation mode to the quad_io mode. Please refer to Figure 12-11 for more details.

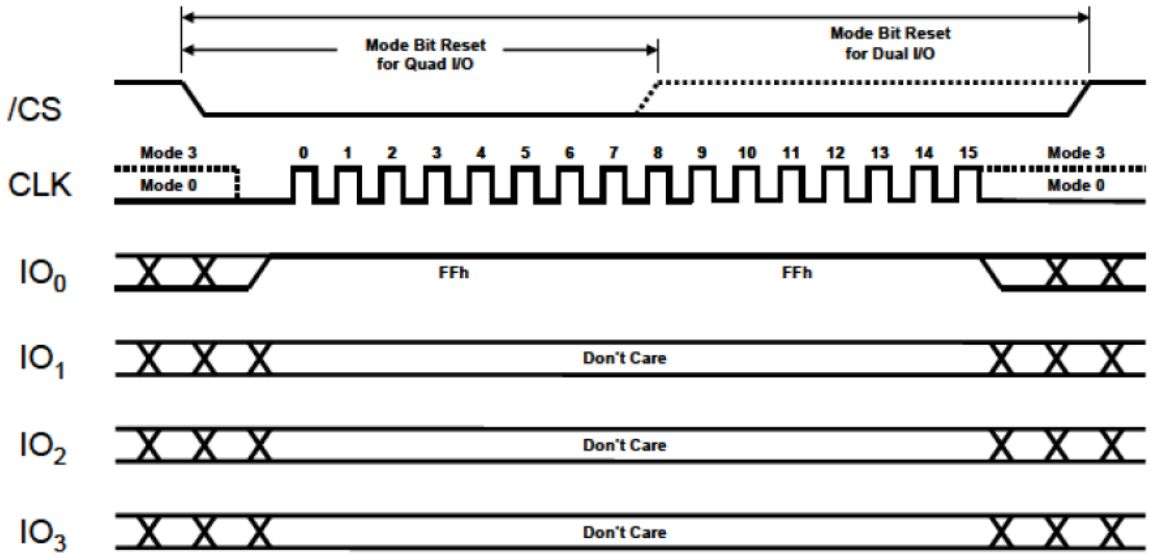
Figure 12-10 Fast Read quad_io Command



Note: If entering the continuous read mode, command can only be accepted by the continuous read command, except the reset continuous read mode.

If users want to reset the continuous read mode, please set the instruction length to ‘1’ for resetting to the quad I/O mode, or set instruction length to ‘2’ for resetting to the dual I/O mode. Please refer to Figure 12-11 for more details.

Figure 12-11 Continuous Read Mode Reset



12.2 Memory Map and Register Definition

Table 12-1 summarizes the control registers of QuadSPI-Controller. The register from offsets 0x10 to 0x54 and 0x100 can be accessed with BYTE, HWORD or WORD as SIZE. The command queue (Offsets 0x00..0x0C) input entry can only accept WORD as SIZE.

Table 12-1 Summary of QaudSPI Controller Registers

Address Offset	Type	Size (Byte)	Description	Default Value
0x000	RW	4	CMD_W0	0x0000_0000
0x004	RW	4	CMD_W1	0x0100_0003
0x008	RW	4	CMD_W2	0x0000_0000
0x00C	RW	4	CMD_W3	0x0000_0000
0x010	RW	4	CR	0x0000_0003
0x014	RW	4	ACTR	0x0000_000F
0x018	R	4	SR	0x0000_0001
0x020	RW	4	ICR	0x0000_0000
0x024	RW1C	4	ISR	0x0000_0000
0x028	R	4	SPISR	0x0000_0000
0x02C	RW	4	SPIAMR	0x0000_0000
0x100	RW	4	DR	0x0000_0000

12.2.1 Command Word0-3

One command contains four words. A command will be executed by writing the fourth word of a command and completely executed by the host QuadSPI-Controller. Use the commands like following:

- For the write command, users must check and clear the command complete interrupt before writing the next command.
- For the read status command, users must check and clear the command complete interrupt before writing the next command.
- For the read data command, users must finish reading the data from RXFIFO; and check and clear the command complete interrupt before writing the next command.

Notes:
2) The command must be executed before reading/writing the data into FIFO.
3) The reserved region of the command queue is not allowed to be filled to '1'.

12.2.1.1 Command Queue First Word

Offset: 0x000

Table 12-2 Command Queue First Word

Bit	Name	Type	Reset	Description
31..0	SPI_FLASH_ADDRESS	RW	0x0	SPI Flash address This register decides the values of the SPI Flash address and issues this address to the SPI Flash. The byte of the address is executed by the address length when offset = 0x04.

12.2.1.2 Command Queue Second Word

Offset: 0x004

Table 12-3 Command Queue Second Word

Bit	Name	Type	Reset	Description
31..29	Reserved	-	0	-
28	CONTINUOUS_READ_ MODE_ENABLE	RW	0	The continuous read mode state is located between the address and second dummy state. It can eliminate the instruction state after the second read command in the continuous read mode. '0': Disable the continuous read mode '1': Enable the 1-byte continuous read mode
27..26	Reserved	-	0	-
25..24	INSTRUCTION_LENGTH	RW	0x1	Instruction code length When users want to execute a SPI Flash command, the instruction code must be included. Different SPI Flash venders have different instruction lengths; therefore, users can set this register to meet different behaviors. Instruction code is normally 1 byte; however, if users set the 2-byte instruction length, the host controller will

Bit	Name	Type	Reset	Description
				issue this instruction code twice. "00": No instruction code, can only be used after the continuous read mode command has been finished. "01": 1-byte instruction code "10": 2-byte instruction code (Repeat the instruction code) "11": Reserved
23..16	DUM_2nd_CYC	RW	0x0	Second dummy state cycle Second dummy state is located between the address and the data state that exclude the continuous read mode state. Users can check whether the dummy state exists between the address and the data state or not in the SPI Flash specification. The host controller will issue logic 1 in the dummy cycle. 0: No second dummy cycle 1-32: 1 - 32 dummy second cycles
15..3	Reserved	-	0	
2..0	ADDRESS_LENGTH	RW	0x3	This register decides the SPI Flash address byte number. Users can set this register to decide the address byte ranging from 1 byte to 4 bytes. 000: No address state 001: 1-byte address 010: 2-byte address 011: 3-byte address 100: 4-byte address Others: Reserved

12.2.1.3 Command Queue third Word

Offset: 0x008

Table 12-4 Command Queue Third Word

Bit	Name	Type	Reset	Description
31..0	DATA COUNTER	RW	0x0	Read/Write data counter This register must be set to '0' when performing the read status command. 0x0: No read/write data 0x1-0xFFFF_FFFF: 1-byte data to 0xFFFF_FFFF data Please note that no matter it is a data read or data write, this register is not allowed to be filled as '0'. However, for the read status, this register must set to '0'.

12.2.1.4 Command Queue Fourth Word

Offset: 0x00C

Table 12-5 Command Queue Fourth Word

Bit	Name	Type	Reset	Description
31..24	INSTRUCTION CODE	RW	0x0	Instruction code Users can set this code to execute the SPI Flash

Bit	Name	Type	Reset	Description
				command.
23..16	CONTINUOUS READ MODE CODE	RW	0x0	Continuous read mode operate code Users can fill this code to execute the continuous read mode. To do the continuous read mode with this Flash, users have to issue 0xA0 as the mode bit because it does not need the instruction code for the next read command.
15..10	Reserved	-	0	-
9..8	START CS	RW	0x0	QuadSPI-Controller can connect four SPI Flashes (max.) and this bit is used to select cs. “00”: cs0 “01”: cs1 “10”: cs2 “11“: cs3
7..5	SPI OPERATE MODE	RW	0x0	SPI operate mode “000”: Serial mode “001”: Dual mode “010”: Quad mode “011“: dual_io mode “100“: quad_io mode Others: Reserved
4	DTR MODE	RW	0	n.a. must be set zero!!
3	READ STATUS	RW	0	Read the SPI Flash status by using software or hardware It is only available when the read status is enabled and the write enable = ‘0’. Users must issue the SPI Read Status command. ‘0’: Read status by hw, controller will poll the status until the status is ready (Not busy) and report the status register. ‘1’: Read status by sw, read status once and report the status register until users can read it.
2	READ STATUS ENABLE	RW	0	Enable the Read SPI status It is available at write_enable = ‘0’ and users must issue the SPI Read Status command. ‘0‘: Disable ‘1‘: Enable
1	WRITE ENABLE	RW	0	Enable the SPI write data, except for the read data or read status (Read the data return path); users must set the write enable = ‘1’ for other SPI commands. Please note that in the write data or erase Flash command, write enable is set to ‘1’. Only when in the read data or read status command, write enable must be set to ‘0’.
0	Reserved	-	0	-

12.2.2 Control Register

Offset: 0x010

This register must be set before executing the command, and it is prohibited to set this register when performing the command. Reserved region is not allowed to be filled to ‘1’.

Table 12-6 Control Register

Bit	Name	Type	Reset	Description
31..19	Reserved	-	0	-
18..16	RDY_LOC	RW	0x0	Busy bit of the SPI status Host polls this busy bit and be ready at the HW read status. “000“-“111“: Bit0-Bit7
15..9	Reserved	-	0	-
8	Abort	RW	0	Flush all commands/FIFOs and reset the state machine. When abort occurs, users must fill commands again. This bit will be automatically cleared to zero.
7..5	Reserved	-	0x0	-
4	SPI_CLK_MODE	RW	0	spi clk mode at the IDLE state 0: mode0, sck_out will be low at the IDLE state. 1: mode3, sck_out will be high at the IDLE state.
3..2	Reserved	-	0	-
1..0	SPI_CLK_DIVIDER	RW	0x0	By setting spi_clk divider sck_out can be set as follows: ”00”: 96MHz ”01”: 48MHz ”10”: 32MHz ”11”: 24MHz

12.2.3 AC Timing Register

Offset: 0x014

Table 12-7 AC Timing Register

Bit	Name	Type	Reset	Description
31..4	Reserved	-	0	-
3..0	CS_DELAY	RW	0xF	cs delay is the time from inactive cs to active cs, which means the timing from high to low. Please follow the specification for details. The unit is the sck_out period. “0000“-“1111“: 1 cycle - 16 cycles

12.2.4 Status Register

Offset: 0x018

Table 12-8 Status Register

Bit	Name	Type	Reset	Description
31..2	Reserved	-	0	-
1	RXFIFO READY	R	0	RXFIFO ready status When RXFIFO is ready, it indicates that RXFIFO is full, and the remained data in RXFIFO will be less than the RXFIFO depth and this is the last data.
0	TXFIFO READY	R	1	TXFIFO ready status When TXFIFO is ready, it indicates that TXFIFO will be empty, and users can transfer the data into

Bit	Name	Type	Reset	Description
				TXFIFO to full.

12.2.5 Interrupt Control Register

Offset: 0x020

Table 12-9 Interrupt Control Register

Bit	Name	Type	Reset	Description
31..14	Reserved	-	0	-
13..12	RXFIFO THOD	RW	0x0	This signal is used to set the trigger level for the RXFIFO threshold interrupt. “00”: 8 entries “01”: 16 entries “10”: 24 entries “11”: reserved The trigger level value is the data in RXFIFO. The unit is WORD.
11..10	Reserved	-	0	-
9..8	TXFIFO THOD		0x0	This signal is used to set the trigger level for the TXFIFO threshold interrupt. “00”: 8 entries “01”: 16 entries “10”: 24 entries “11”: reserved The trigger level value is the available data entry in TXFIFO. For example, if this register is “00”, it means that at least 8 entries are available, and DMA can fill 8 entry data to TXFIFO and will not cause data overflow. The unit is WORD.
7..2	Reserved	-	0	-
1	CMD_CMPLT_INTR_EN	RW	0	Command complete interrupt enable ‘0’: No interrupt ‘1’: Enable command complete interrupt
0	DMA_EN	RW	0	Enable the DMA handshake

12.2.6 Interrupt Status Register

Offset: 0x024

Table 12-10 Interrupt Status Register

Bit	Name	Type	Reset	Description
31..1	Reserved	-	0	-
0	CMD_CMPLT_STS	RW	0	Command complete status will be set when the command is complete

12.2.7 SPI Read Status Register

Offset: 0x028

Table 12-11 SPI Read Status Register

Bit	Name	Type	Reset	Description
31..8	Reserved	-	0	-
7..0	SPI READ STATUS	R	0x0	Host issues the read SPI Flash status command and stores the return data at this register. Users can read this register to check the SPI Flash status.

12.2.8 SPI Address Masking Register

Offset: 0x02C

Table 12-12 SPI Address Masking Register

Bit	Name	Type	Reset	Description
31..0	SPI ADDRESS MASKING	RW	0	For the direct address mapped function, the address of system is always larger than the address of the SPI Flash. Users need to mask the higher bits of the system address to fit the SPI Flash size.

12.2.9 Data Port Register

Offset: 0x100

Table 12-13 Data Port Register

Bit	Name	Type	Reset	Description
31..0	DATA PORT	RW	0	Data port register Users can read/write data from the data port.

13 SD/MMC Card Controller

Offset: 0xD8000000

The ANTAIOS includes a SD/MMC host controller used to access the Secure Digital (SD) card that conforms to the standard specifications, which includes the SDIO specification and SD memory card physical layer specification. The SD/MMC host controller provides the programmable I/O and DMA for data transfers. DMA supports the data transfers between the memory and the SD/MMC card through the AHB Master interface without being interrupted by CPU. DMA also supports the single-blocktransfer, multi-block transfer and infinite transfer. The system address register points to the address of the descriptor table and sequentially accesses the data until the end of a transfer.

SD/MMC Specification

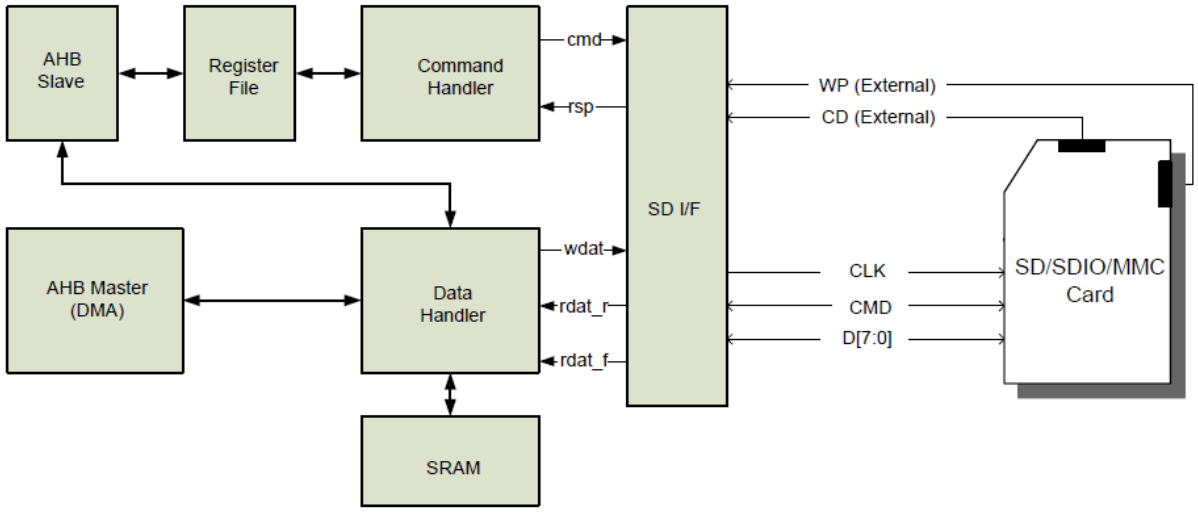
- Compliant with SD/MMC host controller standard specification, version 3.0 (only SDXC, no 1.8VC, no UHS-Mode)
- Supports both DMA and non-DMA data transfers
- Compliant with SD physical layer specification, version 3.0
- Supports 4-bit mode SD bus modes
- Compliant with SDIO card specification, version 2.0
- Compliant with MMC card specification, version 4.3
- Supports configurable 1K SRAM for data FIFO
- Supports configurable 1-bit/4-bit SD card bus and 1-bit/4-bit MMC card bus
- Configurable CPRM function for security
- Built-in generation and check for 7-bit and 16-bit CRC data

Card detection (Insertion/Removal)

- Supports Read Wait mechanism for SDIO function
- CPRM Function
- Compliant with SD security specification in SD specifications (Part 3)
- Supports encryption using C2 cipher in Electronic CodeBook (ECB) mode, C2_E (k,d)
- Supports decryption using C2 cipher in ECB mode, C2_D (k,d)
- Supports encryption using C2 cipher in Converted-Cipher Block Chaining (C-CBC) mode, C2_ECBC (k,d)
- Supports decryption using C2 cipher in C-CBC mode, C2_DCBC (k,d)
- Supports C2 one-way function, C2_G (d1,d2)
- Supports C2 random number generator, RNGC2_G_EN (k,d)
- Supports automatic encryption/decryption in C-CBC mode

13.1 Overview

Figure 13-1 Block diagram of SD Card Controller



The ANTAIOS SD Card controller includes the AHB master interface, AHB slave interface, Command Handler, DATA Handler, Register File, Configurable FIFO, Clock Divider and configurable CPRM function.

13.1.1 AHB Master

When the DMA transfer is used, the AHB Master will initiate a read/write transfer with the memory and the minimum data size will be limited to four bytes. The DMA algorithms defined in the SD Card controller standard specification include SDMA and ADMA (Advanced DMA). SDMA has a disadvantage that the DMA interrupt generated at the every page boundary will disturb CPU in reprogramming the new system address. The new ADMA uses the link-list mechanism to provide a higher data transfer speed.

13.1.2 AHB Slave

The AHB slave is implemented to read/write the host control registers by the processor using the slave interface. The AHB slave provides the byte, half-word, or word data access. Please note that the address must be 0x20 no matter the HSIZE is BYTE, HWORD, or WORD when accessing data port through the register.
The DMA handshake protocol is supported for the flow control when using the external DMA. Please note that the setting of the burst size in the DMA should match the setting of the block size if the FIFO size is in 256-word.
For example, when using the SD memory card with the 512 bytes block size, the burst size should be 128 (The data width is 32 bits). If the FIFO size is 8-word or 16-word, no matter the block size setting, the burst size should be 8 or 16, respectively (The data width is 32 bits).

13.1.3 Command Handler

The command handler is implemented to transmit command to the SD card and receive the response from the SD card. The built-in 7-bit CRC check and generation are involved in the command handler.

13.1.4 Data Handler

The data handler is implemented to transmit data to the SD card and receive data from the SD card. The built-in 16-bit CRC check and generation are involved in the data handler.

13.1.5 Control Register

The control register can be divided into two groups. Command register and status register. To activate a command or data transfer with the card, the command register should be programmed properly. The failure, success or received command/data CRC status will be reported in the status register under any condition. The control register conforms to the SD Host Standard Specification described in Chapter 4.

13.1.6 Data FIFO

The data FIFO is in the full-duplex mode use a 32-bit data width and is implemented as a 256-word depth SRAM.

13.1.7 Clock Divider

In the card identification mode, the maximum clock frequency is 400 kHz. During the data transfer, the maximum clock frequency rate should not exceed 25 MHz in the default speed mode, 50 MHz in the high-speed mode for the SD card and for the MMC card. When the card clock is stopped, the activated command, the received response or the transfer data may be temporarily suspended in the command unit the sufficient clock cycles re-start for the card operation.

13.1.8 Configurable CPRM Function

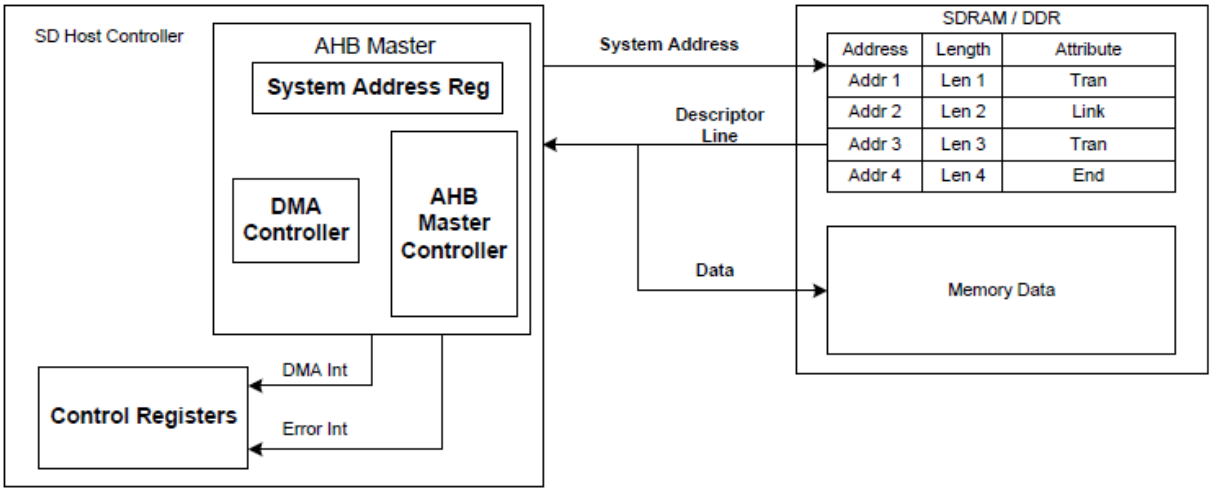
The content protection for the Recordable Media Specification (CPRM) defines a robust method for protecting the content stored on the physical media types. The CPRM function can also be used to protect the content stored on the SD memory card and the other protected storage with an interface equivalent to the SD memory card. The CPRM function includes the Electronic Codebook (ECB) mode cipher, Converted Cipher Block Chaining (C-CBC) mode cipher, C2 one-way function and C2 random number generator.

13.2 DMA Transaction

The ANTAIOS SD Card controller is compliant with the SD/MMC host controller standard specification, Version 2.0. The DMA algorithms (Also called SDAM and ADMA) are implemented in this host controller. SDMA has a disadvantage that the DMA interrupt generated at every page boundary will disturb CPU in reprogramming the new system addresses. Since the host driver is required to reprogram the next system address to restart

the transfer, the performance of the DMA transfer will be affected. The new ADMA implements the link-list mechanism to provide the higher data transfer speed. The host driver can program a list of descriptor line to transfer data without interrupting the host driver. The block diagram of ADMA is shown in Figure 13-2. The descriptor table is created in the system memory by the host driver.

Figure 13-2 Block diagram of ADMA



13.2.1 Data Address and Data Length

The minimum unit of an address is four bytes and the data address should be word-aligned. The maximum data length of each descriptor line should be less than 64K bytes. The total data length should be equal to the block count multiplied by the block size. The block count register is defined as a 16-bit register, which will limit the maximum transfer of 65535 blocks.

1. The address should be word-aligned (4-byte).
2. The data length of each descriptor line should be word-aligned except for the last descriptor.
3. The data length of each descriptor line is less than 64K bytes.
4. Total data length = Len 1 + Len 2 + …… + Len n = Multiple block size
5. If ADMA is used in the CPRM auto mode, the data length of the descriptor line should be the multiple of the block sizes.

13.2.2 Descriptor Table

Table 13-1 shows the 32-bit address descriptor table. Each descriptor line requires the 64-bit memory space. The address field stores the 32-bit data address. The length field defines the data length of each descriptor line.

Attribute

The data in the attribute field is used to control each descriptor line.

Symbol

Three action symbols are specified:

- The “Nop” operation indicates that the current descriptor line is not in operation and fetches the next descriptor line
- The “Tran” operation indicates that the current descriptor line is the data transfer operation, which is designated by the data address and data length.

- The “Link” operation indicates the connection to the other descriptor lines, including the next system address and the other system addresses. The next descriptor line of the system address is generated by the data address of the current descriptor line.

Table 13-1 32-bit Descriptor Table

Bit	Name	Description
63..32	ADDR	32-bit data address
31..16	LENGTH	16-bit data length 0xFFFF: 65535 bytes …. 0x0002: 2 bytes 0x0001: 1 byte 0x0000: 65536 bytes
15..6	Reserved	Reserved as “0000_0000_00”
5..4	ACT	“00”: Nop Current descriptor is not executed and go to the next descriptor line “01”: Res Reserved “10”: Tran Current descriptor line is executed to transfer data “11”: Link Link to the other descriptor line
3	Reserved	Reserved as ‘0’
2	INT	A ‘1’ will generate dma_interrupt_r (NISR, 13.5.16) interrupt when the current descriptor line is done
1	END	End indicates that the current descriptor line is the end of descriptor. When the current descriptor line is done, the tran_complete_r interrupt (NISR, 13.5.16) is generated
0	VALID	Valid indicates that the current descriptor line is valid. If the bit is set to 0, that will generatea dma_err_r (EISR, 13.5.17) interrupt and stop transaction

13.2.3 ADMA States

Figure 13-3 depicts the state diagram of the ADMA transfer. Four states are addressed in this state diagram: Stop transfer state, fetch descriptor state, change address state and transfer data state. The state operations are defined in Table 13-2.
The ADMA transfer can neither be stopped at the block gap nor continue the available requests. If an error occurs during the ADMA transfer, the ADMA operation may stop and generate the **adma_err_r** (EISR, 13.5.17) interrupt. The **adma_err_st_**r (AESR, 13.5.27) hold the ADMA state and **adma_lo_addr_r** (SAR, 13.5.28) holds the system address around the error descriptor. The host driver can read the **adma_err_st_r** (ADMAESR, 13.5.27) to check the state of the ADMA operation.

Figure 13-3 State Diagram ADMA

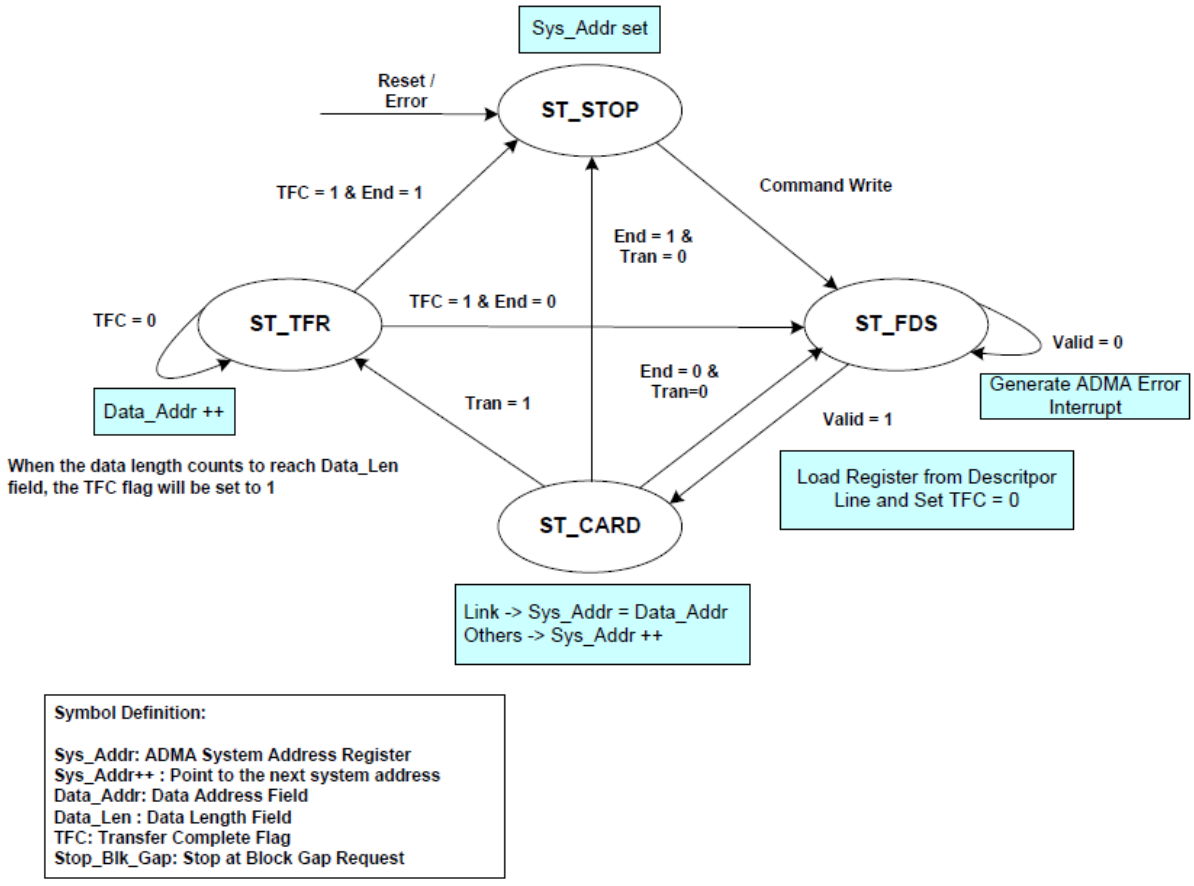


Table 13-2 ADMA States

State	Description
ST_STOP (Stop DMA)	ADMA will stay at this state under the following conditions: 1. When Reset or Error occurs. 2. When all descriptor line data transfers are completed. If the command register is written, the state will go to the ST_FDS state.
ST_FDS (Fetch Descriptor)	In this state, a descriptor line will be fetched from the system memory to the host controller. After fetching a descriptor line, go to the ST_CADR state.
ST_CADR (Change Address)	In this state, the line system of the next descriptor address will be acquired. In the link operation, the next system address will be loaded by the data address of current descriptor line and the state will change to the ST_FDS state to fetch the next descriptor line. In the other operations, the current system address will be incremented to point to the next descriptor line and the state will change to the ST_TFR state.
ST_TFR (Transfer Data)	The data transfer of one descriptor line will be executed between the system memory and the SD card. If the data transfer is not completed (End = '0'), it will go to the ST_FDS state to fetch the next descriptor line. If the data transfer is completed, it will go to the ST_STOP state.

Note: When an ADMA interrupt occurs, the asynchronous abort sequence should be issued.

13.3 SD Command and Data Input/Output Timing

For the SD interface timing, users can adjust the input/output timing to meet the timing margin.

13.3.1 Command and Data Output

Table 13-3 lists different cases of the timing adjustment for the SD interface output timing.

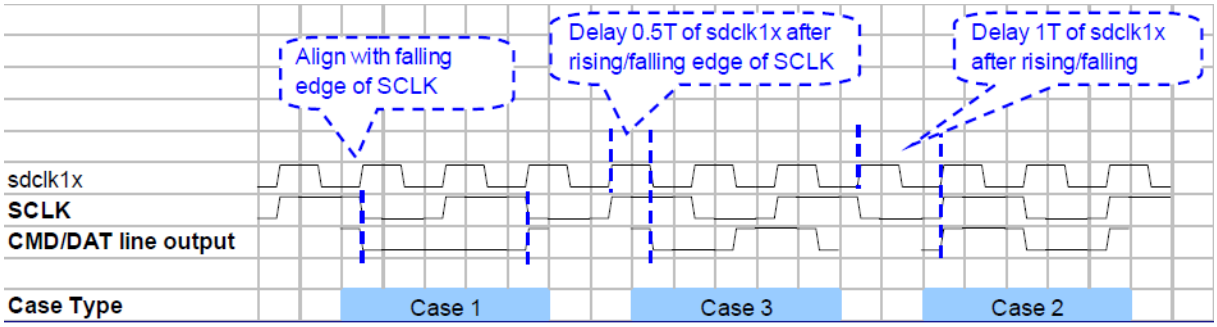
Table 13-3 Timing Adjustment of SD Interface Output

Clock Divided is Zero	Clock Divided is One	Int. Edge Enable	Output Timing Case
No	No	No	Case1
No	No	Yes	Case2
No	Yes	No	Case1
No	Yes	Yes	Case2
Yes	No	No	Case3
Yes	No	Yes	Case2
Others	Inhibit		

Where “clock divided is zero” means that the value of the 10-bit clock divider is zero (CCR, 13.5.13); “clock divided is one” means that the value of the 10-bit clock divider is one (CCR, 13.5.13); “Int. Edge Enable” means that bit[16] of the vendor register0 (VR0, 13.5.31) is set to ‘1’.

The output timing adjustment has three cases. Please refer to Figure 13-4 for details.

Figure 13-4 SD Output Timing



13.3.2 Command and Data Input Latching

For the SD input timing, users need to select the latching mechanism to avoid incorrect latching points. SD Card controller provides pulse latching mechanisms.

The pulse latching mechanism doesn’t require extra multiphase DLL in the system. In general, the system interface has the round-trip effect between the chip and device. To eliminate the round-trip latency, SD Card controller designs one adjustable latching pointer to avoid the issue. Users can set bits[13..8] of the vendor register0 (VR0, 13.5.31) to select the latching pointer. The default value is zero and the latching pointer is at the rising edge of the SD clock (SCLK). The latching pointer delays one sdclk1x (100MHz) period once the value increases by one.

13.4 CPRM Operation

The CPRM operation defines a renewable method for protecting the contents recorded in a number of physical media types.
The CPRM operation is designed to satisfy the following applications:

- It is applicable for both the audio and video contents.
- It is suitable for implementing the PCs devices.
- It is applicable to different media types.

Figure 13-5 depicts the block diagram of the ANTAOIS SD/MMC Card controller with the CPRM function. Table 13-4 lists the basic characteristics of the C2 cipher. Each operation needs the 64-bit data to encrypt or decrypt. There are two basic operational modes: Electronic Codebook (ECB) mode and Converted Cipher Block Chaining (C-CBC) mode.

Figure 13-5 Host Controller with CPRM Function

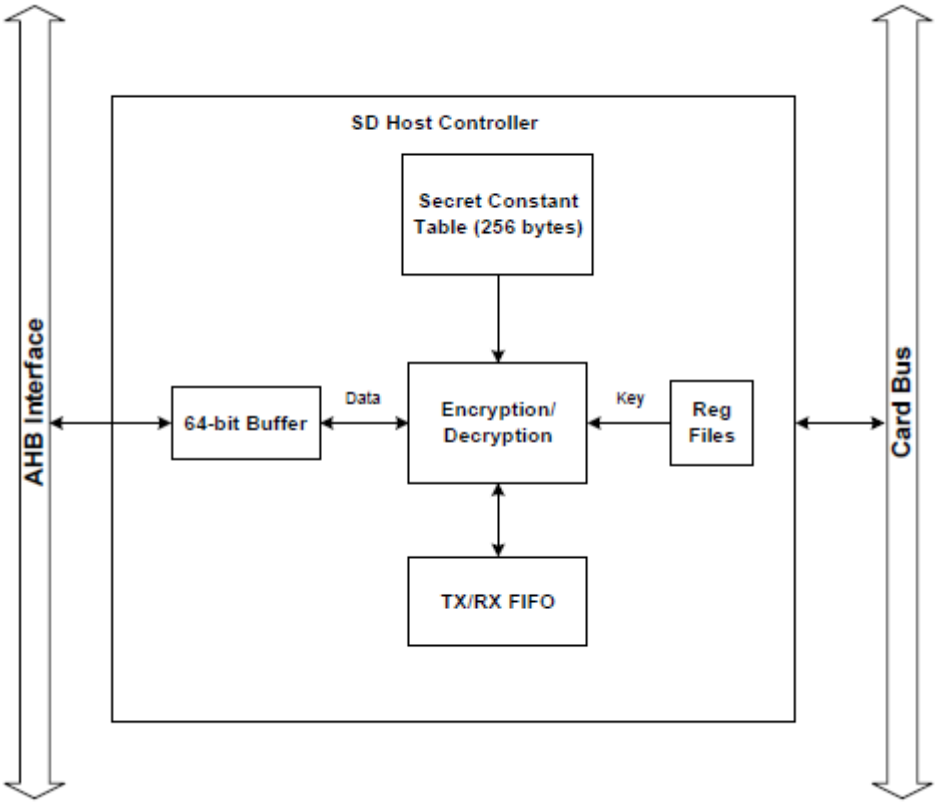


Table 13-4 C2 Cipher Characteristics

Input block size	Output block size	Input key size	Number of rounds
64 bits	64 bits	56 bits	10 rounds

13.4.1 C2 Block Cipher in Electronic Code Book (ECB) Mode

The encryption with the C2 cipher in the Electronic CodeBook (ECB) mode is represented by using the following function:

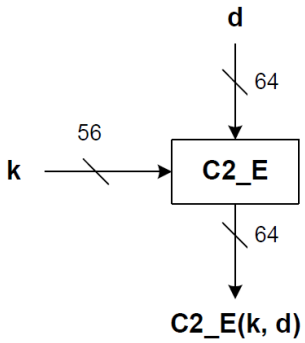
- **C2_E (k,d)**
  where *k* is a 56-bit key, *d* is the 64-bit data to be encrypted and C2_E returns the 64-bit result;

The decryption with the C2 cipher in the ECB mode is represented by using the following function:

- **C2_D (k,d)**
  where *k* is a 56-bit key, *d* is the 64-bit data to be decrypted and C2_D returns the 64-bit result.

Figure 13-6 depicts the encryption operation in the ECB mode. The decryption operation is similar to the one for encryption.

Figure 13-6 Encryption in ECB Mode



13.4.2 C2 Block Cipher in Converted Cipher Block Chaining (C-CBC) Mode

The encryption operation with the C2 cipher in the C-CBC mode is represented by using the following function:

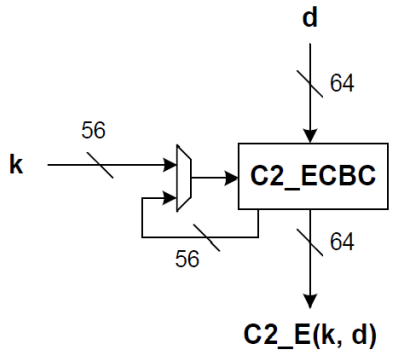
- **C2_ECBC (k,d)**
  where *k* is a 56-bit key, *d* is a frame of data to be encrypted and C2_ECBC returns the encrypted frame.

The decryption operation in the C-CBC mode is represented by using the following function:

- **C2_DCBC (k,d)**
  where *k* is a 56-bit key, *d* is a frame of data to be decrypted and C2_DCBC returns the decrypted frame.

Figure 13-7 depicts the encryption of the C-CBC mode. The decryption operation is similar to the encryption.

Figure 13-7 Encryption in C-CBC Mode



13.5 Memory Map and Register Definition

Table 13-5 SD Card Controller Registers

Address Offset	Type	Size (Byte)	Description	Default Value
0x000	RW	4	SDMADR	0
0x004	RW	2	BSR	0
0x006	RW	2	BCR	0
0x008	RW	4	ARG1	0
0x00C	RW	2	TMR	0
0x00E	RW	2	CR	0
0x010	R	4	RESP0	0
0x014	R	4	RESP1	0
0x018	R	4	RESP2	0
0x01C	R	4	RESP3	0
0x020	RW	4	BDP	0
0x024	R	4	PSR	0x00020000
0x028	RW	1	HC1	0
0x029	RW	1	PCR	0
0x02A	RWAC	1	BGCR	0
0x02C	RW	2	CCR	0
0x02E	RW	1	TCR	0x0E
0x02F	RWAC	1	SRR	0
0x030	RW1C	2	NISR	0
0x032	RW1C	2	EISR	0
0x034	RW	2	NISER	0
0x036	RW	2	EISER	0
0x038	RW	2	NISEN	0
0x03A	RW	2	EISEN	0
0x03C	R	2	AC12ES	0
0x03E	RW	2	HC2	0
0x040	RW	4	CAP0	0x01688480
0x044	RW	4	CAO1	0

Address Offset	Type	Size (Byte)	Description	Default Value
0x048	R	4	Reserved	0
0x04C	RW	4	MCC1	0
0x050	W	2	FACERR	0
0x052	W	2	FERR	0
0x054	R	1	AESR	0
0x058	RW	4	SAR	0
0x05C	R	4	RES	0
0x060	R	4	PV0	0x00080200
0x064	R	4	PV1	0x00080004
0x068	R	4	PV2	0x00020004
0x06C	R	4	PV3	0x00040001
0x0FE	RW	2	HCVR	0x0001
0x100	RW	4	VR0	0x00000001
0x104	RW	4	VR1	0
0x108	RWAC	4	VR2	0
0x10C	R	4	Reserved	0x1F000808
0x110	R	4	Reserved	0
0x114	RW	4	VR5	0
0x118	RW	4	VR6	0x00000001
0x11C	RW1C	4	VR7	0
0x120	RW	4	VR8	0
0x124	RW	4	VR9	0
0x128	RW	4	DHER	0
0x180	RW	4	CMCR	0
0x184	RW1C	4	CMSR	0
0x188	RW	4	CMSE	0
0x18C	RW	4	LWID	0
0x190	RW	4	HWID	0
0x194	RW	4	LWIK	0
0x198	RW	4	HWIK	0
0x19C	R	4	LWOD	0
0x1A0	R	4	HWOD	0
0x1A4	RW	1	SCTDP	0

13.5.1 SDMA System Address Register

Offset: 0x00

Table 13-6 lists the bit assignment of the SDMA system address register. This register is used to set the system memory address of an SDMA transfer or the second argument for Auto CMD23.

- **For the SDMA transfer**

The host driver sets the register before issuing a command to start the SDMA transfer. After the SDMA buffer boundary reaches and stops an SDMA transfer, the next system address of the next contiguous data address can be read from this register. The SDMA transfer waits at every boundary specified by **sdma_buf_bound** in the block size register. When the host driver sets the next system address to this register, the host controller will restart the SDMA transfer.

- **For the Auto CMD23 argument**

This register set a 32-bit block count to the argument of CMD23 while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block will be used. If Auto CMD23 is used without ADMA, the available block count will be limited by **blk_cnt_r**. In this case, 65535 blocks is the maximum value.

Table 13-6 SDMA System Address Register

Bit	Name	Type	Reset	Description
31..0	SDMA_SYS_ADDR_R	RW	0	SDMA system address register or Auto CMD23 argument 2

13.5.2 Block Size Register

Offset: 0x04

Table 13-7 lists the bit assignments of the block size register. This register is used to configure the number of bytes in a data block. The SDMA system address register is updated at each system memory boundary during the SDMA transfer. When the SDMA transfer reaches each boundary, the host controller will trigger the **dma_interrupt_r** interrupt to request the host driver to update the SDMA system address.

Table 13-7 Block Size Register

Bit	Name	Type	Reset	Description
15	Reserved	R	0	-
14..12	SDMA_BUF_BOUND	RW	0	Buffer boundary of the SDMA host 111: 512 kbytes 110: 256 kbytes 101: 128 kbytes 100: 64 kbytes 011: 32 kbytes 010: 16 kbytes 001: 8 kbytes 000: 4 kbytes
11..0	BLK_SIZE_R	RW	0	This register specifies the block size of data transfers for CMD17, CMD18, CMD24, CMD25 and CMD53 and can be set with values ranging from 1 up to the maximum buffer size. In the memory, this register should be set up to 512 bytes. It can be accessed only when no transaction is executed (i.e. After a transaction has stopped). The read operations during the transfers may return an invalid value and the write operations should be ignored.

Bit	Name	Type	Reset	Description
				0x800: 2048 bytes .. 0x200: 512 bytes 0x1FF: 511 bytes .. 0x002: 2 bytes 0x001: 1 bytes 0x000: No transfer

13.5.3 Block Count Register

Offset: 0x06

The block count register is set when the **blk_cnt_en** register is set to '1'. This register is used only for the multi-block transfers. The host controller will decrease the counting number during the data transfer and stop counting when it counts down to zero.

Table 13-8 Block Count Register

Bit	Name	Type	Reset	Description
15..0	BLK_CNT_R	RW	0	Block count of the current transfer 0xFFFF: 65535 blocks … … 0x0002: 2 blocks 0x0001: 1 block 0x0000: Stop counting

13.5.4 Argument 1 Register

Offset: 0x08

This register is assigned to bits[39:8] of the command field.

Table 13-9 Argument 1 Register

Bit	Name	Type	Reset	Description
31..0	ARG1_R	RW	0	Command argument

13.5.5 Transfer Mode Register

Offset: 0x0C

The host driver should set this register before issuing the data transfer command or resume command.

Table 13-10 Transfer Mode Register

Bit	Name	Type	Reset	Description
15..6	Reserved	R	0	-
5	MULTI_BLK_RW	RW	0	Single/Multi-block selection 1: Multiple blocks 0: Single block
4	TRAN_DIR_SEL	RW	0	Data transfer direction selection 1: Read from the card to host 0: Write from the host to card
3..2	AUTO_CMD_EN	RW	0	Auto CMD enable There are two methods to stop the read and write operations of multiple blocks: 1. Auto CMD12 Enable When this field is set to “01”, the Host Controller will automatically issue CMD12 when the last block transfer is completed. The Auto CMD12 error is indicated to the Auto CMD Error Status register. This bit should not be set by the Host Driver if CMD12 is not required by the command. User cannot issue ACMD before receiving transfer complete interrupt of Auto CMD12. 2. Auto CMD23 Enable When this bit field is set to “10”, the Host Controller will automatically issue a CMD23 before issuing a command specified in the Command Register. Auto CMD23 can be used with or without ADMA. By writing the Command Register, the Host Controller will issue CMD23 first and then issue a command specified by the **Command Index** in the Command Register. If the response errors of CMD23 are detected, the second command will not be issued. The Auto CMD23 error is indicated in the Auto CMD Error Status register. CMD23 with a block count value of 32-bit will be set to the SDMA System Address register. “11”: Reserved “10”: Auto CMD23 Enable “01”: Auto CMD12 Enable “00”: Auto Command Disable
1	BLK_CNT_EN	RW	0	Block count enable This bit is only valid for a multi-block transfer. When this bit is set to ‘0’, the blk_cnt_r register will be disabled. The multi-block transfer will be an infinite transfer. Table 13-12 shows the data transfer type setting.
0	DMA_EN	RW	0	DMA enable This bit can enable the DMA transfer. The DMA mode can be selected by **dma_type** in the host control register. When a data transfer command is issued, the DMA transfer will begin the operation. 1: DMA data transfer 0: No data transfer or Non-DMA data transfer

Table 13-11 Transfer Command Setting

Single/Multi-block select	Block Count Enable	Block Count	Transfer Type
0	Don’t care	Don’t care	Single transfer
1	0	Don’t care	Infinite block transfer
1	1	Non-zero	Multi-block transfer
1	1	0	Stop Multi-block transfer

13.5.6 Command Register

Offset: 0x0E

The host driver should check the **Command Inhibit (CMD)** and **Command Inhibit (DAT)** bits in the present state register to determine whether the SD bus is free to transfer.

Table 13-12 Transfer Mode Register

Bit	Name	Type	Reset	Description
15..14	Reserved	R	0	-
13..8	CMD_IDX	RW	0	Command Index These bits should be assigned to bits[45:40] of the command field.
7..6	CMD_TYPE	RW	0	Command Type There are two types of special commands: Normal and Abort. Abort Command If this command is set when executing a read transfer, the Host Controller should stop reading to the buffer. If this command is set when executing a write transfer, the Host Controller should stop driving the ***DAT*** line. After issuing the Abort command, the Host Driver should issue a software reset. These bits should be set to “00” for all other commands. “11”: Abort — CMD12, CMD52 for writing “I/O Abort” in Card Common Control Register (CCCR) “10”: Reserved — Reserved for SDIO resume “01”: Reserved — Reserved for SDIO suspend “00”: Nomal — Other commands
5	DATA_PRES_SEL	RW	0	Data Present Select This bit is set to ‘1’ to indicate that data is present and data transfer is enabled. This bit is set to ‘0’ under the following conditions: 1. Commands only using the ***CMD*** line (ex. CMD52) 2. Commands with no data transfer but using the busy signal on ***DAT[0]*** line (R1b or R5b ex. CMD38) 3. Resume command
4	CMD_IDX_CHK_EN	RW	0	Command Index Check Enable If this bit is set to ‘1’, the host controller will check the index field response to determine if the values are **cmd_idx**. If they are not the same, **cmd_idx_err** will be triggered.
3	CMD_CRC_CHK_EN	RW	0	Command CRC Check Enable If this bit is set to ‘1’, the Host Controller will check the CRC field response to determine whether CRC is correct. If an error is detected, **cmd_crc_err** will be triggered. Table 13-13 shows the relation between the response type and command check.
2	Reserved	R	0	-
1..0	RSP_TYPE_SEL	RW	0	Response Type Select “11”: Response length 48 with busy check “10”: Response length 48 “01”: Response length 136 “00”: No response

Table 13-13 Relationship between Parameter and Names of Response Types

Response Type	Command Index Check Enable	Command CRC Check Enable	Response
00	0	0	No response

Response Type	Command Index Check Enable	Command CRC Check Enable	Response
01	0	1	R2
10	0	0	R3, R4
10	1	1	R1, R5, R6, R7
11	1	1	R1b, R5b

13.5.7 Response Register0-3

Offset: 0x10..0x1C

Table 13-15 lists the mapping of the command responses of each response type. The Host Controller stores the Auto CMD12 response in the upper word of the Response Register to avoid the Auto CMD12 response, which tends to be overwritten by the other command.

Table 13-14 Response Register0-3

Bit	Name	Type	Reset	Description
127..0	RSP_R	R	0	Command Response

Table 13-15 Response Bit Definition of Each Response Type

Response Type	Meaning of Response	Response Field	Response Register
R1, R1b (Normal response)	Card status	R[39..8]	rsp_0_r (rsp_r[31..0])
R1b (Auto CMD12 response)	Card status for Auto CMD12	R[39..8]	rsp_3_r (rsp_r[127..96])
R1 (Auto CMD23 response)	Card status for Auto CMD23	R[39..8]	rsp_3_r (rsp_r[127..96])
R2 (CID, CSD register)	SID or CSD register	R[127..8]	rsp_0_r – rsp_3_r (rsp_r[119..0])
R3 (CID, CSD register)	OCR register for memory	R[39..8]	rsp_0_r (rsp_r[31..0])
R4 (OCR register)	OCR register for I/O	R[39..8]	rsp_0_r (rsp_r[31..0])
R5, R5b	SDIO response	R[39..8]	rsp_0_r (rsp_r[31..0])
R6 (RCA response)	RCA[31..16]	R[39..8]	rsp_0_r (rsp_r[31..0])

13.5.8 Buffer Data Port Register

Offset: 0x20

This register uses the 32-bit Data Port Register to access the internal buffer (Always access through the offset address 0x20 even when the transfer size is byte or half-word.)

Table 13-16 Response Register0-3

Bit	Name	Type	Reset	Description
31..0	DATA_PORT_R	RW	0	Buffer Data Port Register

13.5.9 Present State Register

Offset: 0x24

The host driver can access the status from the 32-bit read-only register.

Table 13-17 Present Status Register

Bit	Name	Type	Reset	Description
31..25	Reserved	R	0	-
24	CMD_LIN_LV	R	0	Command Line Signal Level This status bit is used to check the CMD line level.
23..20	DATA_LIN_LV	R	0	DATA[3:0] Line Signal Level This status bit is used to check the DATA[3:0] line level
19	WR_PROT_LV	R	0	Write Protect Pin Level This status bit is used to reflect the write protect pin level. 1: Write enabled 0: Write protected
18	CD_PIN_LV	R	0	Card Detect Pin Level This status bit is used to reflect the inverse value of the card detect pin. It is stable when sys_card_stable is set to '1'. 1: Card is detected. 0: Card is not detected.
17	SYS_CARD_STABLE	R	1	Card State Stable This status bit is used to reflect whether the card detect pin is stable. 1: No card or card is inserted. 0: Reset or de-bounce
16	SYS_CARD_INSERT	R	0	Card Inserted This status bit indicates whether a card has been inserted. After de-bounced by the Host Controller, the host driver can check whether the card is inserted. The de-bouncing time can be determined by setting db_timeout in the Vendor-defined Register 5.This bit can be changed from 0 to 1 to generate a card_insert_r interrupt in the normal interrupt status register. This bit can be changed from 1 to 0 to generate a card_remove_r interrupt in the normal interrupt status register. 1: Card inserted 0: Reset, de-bouncing, or no card is detected.
15..12	Reserved	R	0	-
11	BUF_REN_R	RC	0	Buffer Read Enable This status bit is used for the non-DMA read transfer. This bit indicates that there are valid data exist in the Rx buffer and is ready for read. This bit can be changed from 1 to 0 to indicate that the Rx buffer data have been read from the buffer. This bit can be changed from 0 to 1 when all block data are ready in the buffer for read or when the Rx buffer is full and generates a buf_r_rdy_r interrupt in the normal interrupt status register. 1: Read enable 0: Read disable
10	BUF_WEN_R	RC	0	Buffer Write Enable This bit is used for the non-DMA write transfer. This bit indicates that the TX buffer is ready to receive data. If the bit is '1', data can be written to the TX buffer. This bit changes from 1 to 0 when all block data are written to the TX buffer or when the Tx buffer is full. This bit changes from 0 to 1 when all block data are written to the Tx buffer. 1: Write enable

Bit	Name	Type	Reset	Description
				0: Write disable Note: This bit is still work when stop at block gap is set. But user should not base on this bit to write data to TX buffer when stop at block gap is set.
9	RD_TRAN_ACT_R	RC	0	Read Transfer Active This status bit is used to detect the completion of a read transfer. This bit is set to '1' under the following conditions: 1. After the end bit of a read command 2. When cont_req in the block gap control register is set to restart a transfer. This bit is set to '0' under the following conditions: 1. When all data blocks specified by the block length are transferred to the system. 2. When sp_blk_gap_req in the block gap control register is set to '1' and the host controller has transferred all the valid data blocks to the system. The tran_complete_r interrupt is generated when this bit changes from '1' to '0'.
8	WR_TRAN_ACT_R	RC	0	Write Transfer Active This status bit indicates that a write transfer is active. This bit is set to '1' under the following conditions: 1. After the end bit of a write command 2. When cont_req in the block gap control register is set to restart a transfer. This bit is set to '0' under the following conditions: 1. After getting the CRC status of the last data block specified by the transfer count. 2. After getting the CRC status of any block where data transmission is stopped by sp_blk_gap_req. A blk_gap_evt_r interrupt will be generated when sp_blk_gap_req is set to '1' and this bit changes to '0.' This bit is useful in the command with a busy data line.
7..3	Reserved	R	0	-
2	DATA_LIN_ACT_R	RC	0	Data Line Active This bit is used to determine whether the Data Line is in use. In a read transfer, this status bit is used to check whether a read transfer is executing on the bus. Changing this bit from 1 to 0 will generate a blk_gap_evt_r interrupt when sp_blk_gap_req is set to '1'. This bit is set to '1' under the following conditions: 1. After the end bit of a read command 2. When cont_req in the block gap control register is set to restart a transfer. This bit is set to '0' under the following conditions: 1. When the end bit of the last data block is sent from the SD bus to the host controller. 2. When sp_blk_gap_req is set to '1' and a read transfer is stopped at the block gap. In a write transfer, this status bit is used to check whether a write transfer is executing on the bus. Changing this bit from 1 to 0 will generate a tran_complete_r interrupt in the normal interrupt status register. This bit is set to '1' under the following conditions: 1. After the end bit of a read command 2. When cont_req in the block gap control register is set to restart a transfer. This bit is set to '0' under the following conditions: 1. When the card release the busy signal of the last data block. 2. When sp_blk_gap_req is set to '1' and the card releases the write busy at the block gap. In the command with busy data line, this bit indicates whether a command with busy is executing on the bus. This bit will be

Bit	Name	Type	Reset	Description
				set after the end bit of the command with busy and will be cleared when busy is de-asserted or busy is not detected after the end of a response.
1	CMD_INHIBIT_D	RC	0	Command Inhibit (DAT) This bit will be generated if data_lin_act_r or rd_tran_act_r is set to '1'. When this bit is set to '0', the host driver can issue a new command by using the data line. Command with busy data line belongs to cmd_inhibit_d. 1: Cannot issue new commands to use the data line 0: Issue new commands to use the data line
0	CMD_INHIBIT_C	RC	0	Command Inhibit (CMD) When this bit is set to '0', the host driver will issue a new command by using the command line. When this bit is set to '1', the command register will be written. This bit is cleared when the command response is received. Even if cmd_inhibit_d is set to '1', the commands using the command line can be issued if this bit is set to '0'. CMD0, CMD12, CMD13, and CMD52 can be issued when the data lines are in use during a data transfer. The cmd_complete_r interrupt in the normal interrupt register is issued when this bit is changed from 1 to 0. However, if the present command suffer some errors (CRC error, command index error and so on), this bit will be remained '1'. Note: This bit will not be set by Auto CMD12. 1: Cannot issue command 0: Issue command only with the command line

13.5.10 Host Control1 Register

Offset: 0x28

Table 13-18 Host Control1 Register

Bit	Name	Type	Reset	Description
7	CD_SEL	RW	0	Card Detect Signal Selection This bit is used to select the source of card detection. 1: The test level for the card detection 0: The card detect pin is selected.
6	CD_TEST_LV	RW	0	Card Detect Test Level This bit is used to set the test level when cd_sel is set to '1'. 1: Card is inserted. 0: Card cannot be found.
5	EXT_DATA_WIDTH	RW	0	Extended Data Transfer Width This bit controls the 8-bit bus width mode for the embedded device, which is indicated in the 8-bit Support for the Embedded Device in the Capabilities register. When this bit is set to '1', it indicates that the 8-bit bus mode is supported by the device. When this bit is set to '0', it indicates that the bus width is controlled by the Data Transfer Width in the Host Control 1 register. This bit will not be effective if multiple devices are installed on a bus slot (Slot Type is set to '10b' in the Capabilities register). In this case, each device bus width will be controlled by the Bus Width Preset field in the Shared Bus Control register. 1: 8-bit bus width 0: Bus width is selected by the data transfer width.
4..3	DMA_TYPE	RW	0	DMA Type Select 11: Reserved

Bit	Name	Type	Reset	Description
				10: 32-bit address ADMA2 is selected 01: Reserved 00: SCMA is selected
2	HI_SPEED	RW	0	High Speed Enable If this bit is set to '0', the host controller can output a command and data at the falling edge of io_sd_clk at a speed of up to 25 MHz. If this bit is set to '1', the host controller can output a command and data at the rising edge of io_sd_clk at a speed of up to 50 MHz.
1	DATA_WIDTH	RW	0	Data Width This bit can select the data width of the host controller. The data width should match the SD card bus width during the data transfer. 1: 4-bit mode 0: 1-bit mode
0	LED_CTRL	RW	0	LED Control This bit is used to notify users that the SD card has been accessed. 1: LED on 0: LED off

13.5.11 Power Control Register

Offset: 0x29

Table 13-19 Power Control Register

Bit	Name	Type	Reset	Description
7..4	Reserved	R	0	-
3..1	SD_BUS_VOL	RW	0	SD Bus Voltage Select By setting these bits, the Host Driver will select the voltage level for the SD card. Before setting this register, the Host Driver should check the Voltage Support bits in the Capabilities Register. If an unsupported voltage is selected, the Host System should not supply the SD bus voltage. 111: 3.3V (Typ.) Others: Reserved
0	SD_BUS_POW	RW	0	SD Bus Power This bit is used to enable the SD bus power. If no card is detected by the Host Controller, this bit will be set to '0'. The io_sd_clk signal cannot be driven to '0' by setting this bit.

13.5.12 Block Gap Control Register

Offset: 0x2A

Table 13-20 Block Gap Control Register

Bit	Name	Type	Reset	Description
7..4	Reserved	R	0	-
3	INT_AT_BLK_GAP	RW	0	Interrupt at Block Gap This bit is only useful in the 4-bit mode of the SDIO card. Setting this bit to '1' will enable the interrupt detection at the block gap for the multiple block transfers.

Bit	Name	Type	Reset	Description
				This bit should be set to '0' if the SDIO card cannot issue an interrupt during the block gap. 1: Check the interrupt at block gap enable 0: Check the interrupt at block gap disable
2	READ_WAIT	RW	0	Read Wait Control This bit is useful for the SDIO card. When the SDIO card supports the read wait function, the host controller can use the read wait protocol to control the SDIO bus. When the card does not support the read wait function, the host controller will stop io_sd_clk to hold the read transfer. 1: Enable the read wait function 0: Disable the read wait function
1	CONT_REQ	RW	0	Continue Request This bit is used to restart a transaction, which can be stopped by using sp_blk_gap_req. To restart a transaction, this bit should be set to '1' and sp_blk_gap_req should be set to '0'. The host controller will automatically clear this bit under the following conditions: 1. In a read transfer, data_lin_act_r will be changed from 0 to 1 to start a read transfer. 2. In a write transfer, wr_tran_act_r will be changed from 0 to 1 to start a write transfer. 1: Restart 0: No effect
0	SP_BLK_GAP_REQ	RW	0	Stop at Block Gap Request This bit is used to stop executing the read or write transfer at the next block gap. This bit is useful for the non-DMA and SDMA, but is not useful for ADMA. The host driver should keep the setting of this bit to '1' until the tran_complete_r interrupt is set to '1'. If this bit is set to '1', the host controller will stop at the block gap by using read_wait or stop io_sd_clk in a read transaction. If this bit is set to '0', the host controller will not write data to data_port_r. 1: Stop 0: Transfer

13.5.13 Clock Control Register

Offset: 0x2C

Table 13-21 Clock Control Register

Bit	Name	Type	Reset	Description
15..8	LOW_BIT_SD_CLK_SEL	RW	0	SD clock frequency value [7:0] for the 10-bit divided clock mode This byte is used to select the frequency of the io_sd_clk pin. The base clock is specified by 100MHz 0x3FF: Base clock divided by 2046 0x3FE: Base clock divided by 2044 .. 0x002: Base clock divided by 4 0x001: Base clock divided by 2 0x000: Base clock (100MHz)
7..6	UPPER_BIT_SD_CLK_SEL	RW	0	SD clock frequency value [9:8] for the 10-bit divided clock mode
5..3	Reserved	R	0	-

Bit	Name	Type	Reset	Description
2	SD_CLK_EN	RW	0	SD Clock Enable When this bit is set to '1', io_sd_clk will be outputted to the SD card. When this bit is set to '0', io_sd_clk will be stopped at '0'.
1	CLK_STABLE	RC	0	Internal Clock Stable This bit is set to '1' when the internal clock is stable. After inter_clk_en is set to '1', the internal clock will begin oscillating.
0	INTER_CLK_EN	RW	0	Internal Clock Enable When the Host Controller is not use by the Host Driver, this bit can be set to '0' to stop the internal clock to be in the low-power state. When this bit is set to '1', the internal clock will begin oscillating.

13.5.14 Timeout Control Register

Offset: 0x2E

This host driver should set the timeout value according to the **cap_r** register. The value of data_timer indicates the data line timeout times.

Table 13-22 Timeout Control Register

Bit	Name	Type	Reset	Description
7..4	Reserved	R	0	-
3..0	DATA_TIMER	RW	0xE	Data Timeout counter Value 0xF: Reserved 0xE: 1/96MHz x 2^{27} 0xD: 1/96MHz x 2^{26} … 0x0: 1/96MHz x 2^{13}

13.5.15 Software Reset Register

Offset: 0x2F

A reset pulse will be generated when this bit is set to '1'. This bit will be automatically cleared once the reset pulse is issued.

Table 13-23 Software Reset Register

Bit	Name	Type	Reset	Description
7..3	Reserved	R	0	-
2	SOFT_RST_DAT	RWC	0	Software Reset for Data Line The following registers and bits will be cleared by this reset bit: 1. Buffer Data Port Register data_port_r 2. Present State Registers buf_ren_r, buf_wen_r, rd_tran_act_r, wr_tran_act_r, data_lin_act_r and cmd_inhibit_d 3. Block Gap Control Registers cont_req and sp_blk_gap_req 4. Normal Interrupt Status Registers buf_r_rdy_r, buf_w_rdy_r, dma_interrupt_r, blk_gap_evt_r and tran_complete_r
1	SOFT_RST_CMD	RWC	0	Software Reset for Command Line The following registers and bits will be cleared by this reset bit:

Bit	Name	Type	Reset	Description
				1. Present State Register cmd_inhibit_c 2. Normal Interrupt Status Register cmd_complete_r
0	SOFT_RST_ALL	RWC	0	Software Reset for All The reset signal resets all registers except for the card detection and capability register.

13.5.16 Normal Interrupt Status Register

Offset: 0x30

The interrupt status can be latched by setting the **nor_int_st_en_r** register to '1'.

Table 13-24 Normal Interrupt Status Register

Bit	Name	Type	Reset	Description
15	ERR_INTRRUPT_R	RC	0	Error Interrupt This bit will be set to '1' if any bit in the err_int_status_r register is set to '1'.
14..12	Reserved	R	0	-
11	INT_C_R	RC	0	INT_C This status bit will be set if INT_C is enabled and the INT_C# pin is at the low level. Writing this bit to '1' will not clear this bit. This bit can be cleared by resetting the INT_C interrupt factor.
10	INT_B_R	RC	0	INT_B This status bit will be set if INT_B is enabled and the INT_B# pin is at the low level. Writing this bit to '1' will not clear this bit. This bit can be cleared by resetting the INT_B interrupt factor.
9	INT_A_R	RC	0	INT_A This status will be set if INT_A is enabled and INT_A# pin is at the low level. Writing this bit to '1' will not clear this bit. This bit can be cleared by resetting the INT_A interrupt factor.
8	CARD_INT_R	RC	0	Card Interrupt In the 4-bit mode, the card interrupt is sampled during the interrupt cycle. When this bit is set to '1', the interrupt will need to be handled by the Host Driver. The host driver needs to set card_int_st_en to '0' and clear the card_int_r status to avoid driving the interrupt signal to the host system again. If the Host Driver has handled the card interrupt completely, the card_int_st_en should be set to '1' to re-start sampling.
7	CARD_REMOVE_R	RWC	0	Card Remove This status bit will be set if sys_card_inst in the Present State Register changes form '1' to '0'. When the host driver writes '1' to this bit, this status will be cleared.
6	CARD_INSERT_R	RWC	0	Card Insert This status bit will be set if sys_card_inst in the Present State Register changes from '0' to '1'. When the host driver writes '1' to this bit, this status will be cleared.
5	BUF_R_RDY_R	RWC	0	Buffer Read Ready This status bit will be set if buf_ren_r in the Present State Register changes from '0' to '1'.
4	BUF_W_RDY_R	RWC	0	Buffer Write Ready This status bit will be set if buf_wen_r in the Present State Register changes from '0' to '1'.

Bit	Name	Type	Reset	Description
3	DMA_INTERRUPT_R	RWC	0	DMA Interrupt This status bit will be set if the host controller detects that the SDMA buffer boundary is reached during a transfer, except in the last block is completed. In case of ADMA, the int field will be set in the descriptor line and the host controller will generate the interrupt once the descriptor line is done.
2	BLK_GAP_EVT_R	RWC	0	Block Gap Event By setting sp_blk_gap_req to '1', this bit will be set when a read or write transaction is stopped at the block gap. 1. Read transaction This bit will be set when data_lin_act_r changes from '1' to '0'. The read wait should be supported for this function in the SDIO card. 2. Write transaction This bit will be set when wr_tran_act_r changes from '1' to '0'.
1	TRAN_COMPLETE_R	RWC	0	Transfer Complete A data transfer will be completed or a data transfer will be stopped at the block gap during a transmission to generate a transfer complete interrupt. 1. Read transaction This bit will be set when rd_tran_act_r changes from '1' to '0'. 2. Write transaction This bit will be set when data_lin_act_r changes from '1' to '0'. 3. Command with busy This bit will be set when the busy data line is released.
0	CMD_COMPLETE_R	RWC	0	Command Complete This bit will be set when the command response is received and the CRC check is okay. Auto CMD12 and Auto CMD23 will not generate the command complete interrupt. The Command Timeout Error bit (EISR, 13.5.17) has higher priority than the Command Complete bit. When both bits are set to '1', it indicates that the response has not been correctly received.

Command Complete	Command Timeout Error	Meaning of Status
0	0	Interrupted by another factor
Don't care	1	Response not received within 64 SD-Card clocks
1	0	Response received

13.5.17 Error Interrupt Status Register

Offset: 0x32

The interrupt status can be latched when the **err_int_st_en_r** registers are set to '1'.

Table 13-25 Error Interrupt Status Register

Bit	Name	Type	Reset	Description
15..11	Reserved	R	0	-
10	TUNING_ERR_R	RWC	0	Tuning Error This bit is set when an unrecoverable error is detected in a tuning circuit except during the tuning pRCedure

Bit	Name	Type	Reset	Description
				(Occurrence of an error during the tuning pRCedure is indicated by the Sampling Select). By detecting the Tuning Error, the Host Driver needs to abort a command executing and perform tuning. To reset the tuning circuit, the Sampling Clock should be set to '0' before executing the tuning pRCedure. The Tuning Error has higher priority than the other error interrupts generated during the data transfer. By detecting the Tuning Error, the Host Driver should discard the data transferred by a current read/write command and retry data transfer after the Host Controller is retrieved from the tuning circuit error.
9	ADMA_ERR_R	RWC	0	ADMA Error This bit will be set to '1' when an error is detected in the ADMA error status register during the ADMA transfer. In addition, the Host Controller will set this interrupt when it detects the invalid descriptor data (Valid = 0).
8	AUTO_CMD12_ERR_R	RWC	0	Auto CMD12 Error This bit will be set to '1' when any error is detected in the auto CMD12 error status register.
7	CUR_LIM_ERR_R	RWC	0	Current Limit Error By setting the SD Bus Power bit in the Power Control register, the Host Controller will be requested to supply power for the SD Bus. If the Host Controller supports the Current Limit function, it can be protected from an illegal card by stopping supplying power to the card. In this case, this bit will indicate a failure status. Reading 1 means that the Host Controller is not supplying power to the SD card due to some failures. Reading 0 means that the Host Controller is supplying power and no error has occurred. The Host Controller may require some sampling time to detect the current limit. This bit should be always set to '0' if the Host Controller does not support this function.
6	DATA_END_BIT_ERR_R	RWC	0	Data End Bit Error This status is set when the host controller detects 0 at the end bit of the read data, which uses the data line or at the end bit of the write CRC status.
5	DATA_CRC_ERR_R	RWC	0	Data CRC Error This status is set when the host controller detects the CRC error during the read transfer or detects the write CRC status that is not of the value of "010".
4	DATA_TIMEOUT_ERR_R	RWC	0	Data Timeout Error This bit will be set under the following conditions: 1. Wait to read data timeout 2. Wait write CRC status timeout 3. Busy timeout after write CRC status 4. Busy timeout for the R1b and R5b types
3	CMD_IDX_ERR_R	RWC	0	Command Index Error This bit will be set when the command index in the command response is different from the command.
2	CMD_END_BIT_ERR_R	RWC	0	Command End Bit Error This bit will be set when the end bit of the command response is 0.
1	CMD_CRC_ERR_R	RWC	0	Command CRC Error This bit will be set when the command CRC error is detected.
0	CMD_TIMEOUT_ERR_R	RWC	0	Command Timeout Error This bit will be set when no response is sent from the card within 64 SD card clock.

13.5.18 Normal Interrupt Status Enable Register

Offset: 0x34

If the corresponding bit of the interrupt source in the normal interrupt status enable register is set to '1', the interrupt becomes active, which is latched and available for the host driver in the normal interrupt status register.

Table 13-26 Normal Interrupt Status Enable Register

Bit	Name	Type	Reset	Description
15..12	Reserved	R	0	-
11	INT_C_ST_EN	RW	0	INT_C Status Enable If this bit is set to '0', the Host Controller will clear the interrupt request to the system. The Host Driver may clear this bit before serving INT_C and may reset this bit after all interrupt requests to the INT_C pin are cleared to prevent the inadvertent interrupts.
10	INT_B_ST_EN	RW	0	INT_B Status Enable If this bit is set to '0', the Host Controller will clear the interrupt request to the system. The Host Driver may clear this bit before serving INT_B and may reset this bit after all interrupt requests to the INT_B pin are cleared to prevent the inadvertent interrupts.
9	INT_A_ST_EN	RW	0	INT_A Status Enable If this bit is set to '0', the Host Controller will clear the interrupt request to the system. The Host Driver may clear this bit before serving INT_A and may reset this bit after all interrupt requests to the INT_A pin are cleared to prevent the inadvertent interrupts.
8	CARD_INT_ST_EN	RW	0	Card Interrupt Status Enable If this bit is set to '1', card_int_r will be served by the Host Driver. The Host Driver should set this bit to '0' before serving card_int_r and should reset this bit after card_int_r completes requesting card. The detection of card_int_r will be stopped when this bit is set to '0' and will be restarted when this bit is set to '1'.
7	CARD_REMOVE_ST_EN	RW	0	Card Remove Status Enable
6	CARD_INSERT_ST_EN	RW	0	Card Insert Status Enable
5	BUF_R_RDY_ST_EN	RW	0	Buffer Read Ready Status Enable
4	BUF_W_RDY_ST_EN	RW	0	Buffer Write Ready Status Enable
3	DMA_INTERRUPT_ST_EN	RW	0	DMA Interrupt Status Enable
2	BLK_GAP_EVT_ST_EN	RW	0	Block Gap Event Status Enable
1	TRAN_COMPLETE_ST_EN	RW	0	Transfer Complete Status Enable
0	CMD_COMPLETE_ST_EN	RW	0	Command Complete Status Enable

13.5.19 Error Interrupt Status Enable Register

Offset: 0x36

If the corresponding bit of the interrupt source in the Error Interrupt Status Enable Register is set to '1' and if the interrupt becomes active, the active state will be latched and will be available for the host driver in this register.

Table 13-27 Error Interrupt Status Enable Register

Bit	Name	Type	Reset	Description
15..10	Reserved	R	0	-
9	ADMA_ERR_ST_EN	RW	0	ADMA Error Status Enable
8	AUTO_CMD12_ERR_ST_EN	RW	0	Auto CMD12 Error Status Enable
7	CUR_LIM_ERR_ST_EN	RW	0	Current Limit Error Status Enable
6	DATA_END_BIT_ERR_ST_EN	RW	0	Data End Bit Error Status Enable
5	DATA_CRC_ERR_ST_EN	RW	0	Data CRC Error Status Enable
4	DATA_TIMEOUT_ERR_ST_EN	RW	0	Data Timeout Error Status Enable
3	CMD_IDX_ERR_ST_EM	RW	0	Command Index Error Status Enable
2	CMD_END_BIT_ERR_ST_EN	RW	0	Command End Bit Error Status Enable
1	CMD_CRC_ERR_ST_EN	RW	0	Command CRC Error Status Enable
0	CMD_TIMEOUT_ERR_ST_EN	RW	0	Command Timeout Error Status Enable

13.5.20 Normal Interrupt Signal Enable Register

Offset: 0x38

This register is used to select the interrupt status that is notified to the host system as an interrupt. These interrupt statuses share the same interrupt line.

Table 13-28 Normal Interrupt Signal Enable Register

Bit	Name	Type	Reset	Description
15..12	Reserved	R	0	-
11	INT_C_SIG_EN	RW	0	INT_C Signal Enable
10	INT_B_SIG_EN	RW	0	INT_B Signal Enable
9	INT_A_SIG_EN	RW	0	INT_A Signal Enable
8	CARD_INT_SIG_EN	RW	0	Card Interrupt Signal Enable
7	CARD_REMOVE_SIG_EN	RW	0	Card Remove Signal Enable
6	CARD_INSERT_SIG_EN	RW	0	Card Insert Signal Enable
5	BUF_R_RDY_SIG_EN	RW	0	Buffer Read Ready Signal Enable
4	BUF_W_RDY_SIG_EN	RW	0	Buffer Write Ready Signal Enable
3	DMA_INTERRUPT_SIG_EN	RW	0	DMA Interrupt Signal Enable
2	BLK_GAP_EVT_SIG_EN	RW	0	Block Gap Event Signal Enable
1	TRAN_COMPLETE_SIG_EN	RW	0	Transfer Complete Signal Enable
0	CMD_COMPLETE_SIG_EN	RW	0	Command Complete Signal Enable

13.5.21 Error Interrupt Signal Enable Register

Offset: 0x3A

Table 13-29 Normal Interrupt Signal Enable Register

Bit	Name	Type	Reset	Description
15..10	Reserved	R	0	-
9	ADMA_ERR_SIG_EN	RW	0	ADMA Error Signal Enable

Bit	Name	Type	Reset	Description
8	AUTO_CMD12_ERR_SIG_EN	RW	0	Auto CMD12 Error Signal Enable
7	CUR_LIM_ERR_SIGN_EN	RW	0	Current Limit Error Signal Enable
6	DATA_END_BIT_ERR_SIG_EN	RW	0	Data End Bit Error Signal Enable
5	DATA_CRC_ERR_SIG_EN	RW	0	Data CRC Error Signal Enable
4	DATA_TIMEOUT_ERR_SIG_EN	RW	0	Data Timeout Error Signal Enable
3	CMD_IDX_ERR_SIG_EN	RW	0	Command Index Error Signal Enable
2	CMD_END_BIT_ERR_SIG_EN	RW	0	Command End Bit Error Signal Enable
1	CMD_CRC_ERR_SIG_EN	RW	0	Command CRC Error Signal Enable
0	CMD_TIMEOUT_ERR_SIG_EN	RW	0	Command Timeout Error Signal Enable

13.5.22 Auto CMD12 Error Status Register

Offset: 0x3C

When the auto_cmd12_en register is set to ‘1’ and the Auto CMD12 error status register is set, the host driver will check this register to identify what kind of error happens during executing AUTO CMD12. This register will be valid only when **auto_cmd12_err_r** is set to ‘1’.

Table 13-30 Auto CMD12 Error Status Register

Bit	Name	Type	Reset	Description
15..8	Reserved	R	0	-
7	CMD_NO_EX_BY_CMD12_R	RC	0	Command Not Executed by Auto CMD12 Error This bit indicates that the command followed Auto CMD12 is not executed due to an Auto CMD12 error (Bit 1 to bit 4) in this register. The bit will be set to ‘0’ when Auto CMD Error is generated by Auto CMD23.
6..5	Reserved	R	0	-
4	AUTO_CMD_IDX_ERR_R	RC	0	Auto CMD Index Error
3	AUTO_CMD_END_BIT_ERR_R	RC	0	Auto CMD End Bit Error
2	AUTO_CMD_CRC_ERR_R	RC	0	Auto CMD CRC Error
1	AUTO_CMD_TIMEOUT_ERR_R	RC	0	Auto CMD Timeout Error
0	AUTO_CMD12_NO_EX_R	RC	0	Auto CMD12 Not Executed If the memory multiple block data transfer is not started due to the command error, this bit will not be set because it is not necessary to issue Auto CMD12. Setting this bit to ‘1’ means that the Host Controller cannot issue Auto CMD12 to stop the memory multiple block data transfer due to some errors. If this bit is set to ‘1’, other error status bits [4..1] are meaningless. This bit is set to ‘0’ when Auto CMD Error is generated by Auto CMD23.

The relationship between Auto CMD CRC Error and Auto CMD Timeout Error is shown in Table 13-31

Table 13-31 Relationship between CRC Error and Timeout Error for Auto CMD

Auto CMDCRC Error	Auto CMD Timeout Error	Error Type
0	0	No Error
0	1	Response Timeout Error
1	0	Response CRC Error
1	1	CMD line conflict

13.5.23 Host Control 2 Register

Offset: 0x3E

Table 13-32 Normal Interrupt Signal Enable Register

Bit	Name	Type	Reset	Description
15	PRESET_VAL_EN	RW	0	Preset Value Enable Since the operating SDCLK frequency and I/O driving strength are dependent on the Host System implementation, it is difficult to determine these parameters in the Standard Host Driver. When this bit is set, the SDCLK frequency will be automatically generated and the driver strength will be selected without considering the specific system conditions. This bit enables the functions defined in the Preset Value registers. 1: Automatic selection by the preset value 0: SDCLK and driver strength are controlled by the Host driver.
14	ASYN_INT_EN	RW	0	Asynchronous Interrupt Enable This bit can be set to '1' if a card supports the asynchronous interrupts and the Asynchronous Interrupt Support is set to '1' in the Capabilities register. If this bit is set to '1', the Host driver will stop SDCLK during the asynchronous interrupt period to save power. During this period, the Host controller will continue delivering the Card Interrupt to the host when it is asserted by the card.
13..0	Reserved	R	0	-

13.5.24 Capabilities 0/1 Register

Offset: 0x40

The host controller may implement these values during initialization.

Table 13-33 Capabilities 0/1 Register

Bit	Name	Type	Reset	Description
63..44	Reserved	R	0	-
39..35	Reserved	R	0	-
34	DDR50_SUPPORT	RW	0	DDR50 Support '1': enable (not supported) '0': disable
33	DDR104_SUPPORT	RW	0	DDR104 Support '1': enable (not supported) '0': disable
32	SDR50_SUPPORT	RW	0	SDR50 Support

Bit	Name	Type	Reset	Description
				‘1‘: enable (not supported) ‘0‘: disable
31..30	SLOT_TYPE	RW	0	Slot Type This field indicates the usage of a slot by a specific Host System (A host controller register set is defined per slot). “00”: Removeable Card slot “01”: Embedded Slot for One Device indicates that only one non-removable device will be connected to a SD bus slot. Others: Reserved
29..27	Reserved	R	0	-
26	VOLTAGE_1_8_SUPPORT	RW	0	Voltage supports 1.8 V. (not supported)
25	VOLTAGE_3_0_SUPPORT	RW	0	Voltage supports 3.0 V. (not supported)
24	VOLTAGE_3_3_SUPPORT	RW	1	Voltage supports 3.3 V.
23	SUSPEND_RESUME _SUPPORT	RW	0	Suspend/Resume Support This bit suspends/resumes the SDIO function of the host controller.
22	SDMA_SUPPORT	RW	1	SDMA Support This bit indicates whether the host controller can use SDMA to transfer data.
21	HI_SPEED_SUPPORT	RW	1	High Speed Support This bit indicates whether the host controller can support the high-speed mode and can supply io_sd_clk from 25 MHz to 50 MHz.
20	Reserved	R	0	-
19	ADMA2_SUPPORT	RW	1	ADMA2 Support This bit indicates whether the host controller is capable of using ADMA2.
18	8BIT_SUPPORT	RW	0	8-bit Support for Embedded Device This bit indicates whether the host controller is capable of using the 8-bit bus width mode. (not supported)
17..16	MAX_BLK_LEN	RW	0	Maximum Block Length This value indicates the maximum block size that can be read and written by the host driver to the buffer in the host controller. The buffer should transfer this block size without wait cycles. “10”: 2048 bytes (not supported) “01”: 1024 bytes (not supported) “00”: 512 bytes
15..8	BASE_CLK_FOR_SD_CLK	RW	0x64	Base Clock Frequency For SD Clock This value indicates the base (Maximum) clock frequency for the SD Clock. 0xFF: 255 MHz … 0x64: 100 MHz … 0x02: 2 MHz 0x01: 1 MHz 0x00: Get information via another method
7	TIMEOUT_CLK_UNIT	RW	1	Timeout Clock Unit This bit shows the unit of the base clock frequency used to detect data_timeout_err_r. ‘1’: MHz ‘0’: kHz
6	Reserved	R	0	-
5..0	TIMEOUT_CLK_FREQ	RW	0	Timeout Clock Frequency This bit shows the base clock for detecting

Bit	Name	Type	Reset	Description
				data_timeout_err_r. The base clock is dominated by the system clock. Users should obtain the system clock frequency.

13.5.25 Force Event Register for Auto CMD Error Status

Offset: 0x50

The Force Event register is not a physical register. It is an address to which the Auto CMD error status register can be written. The force event register is only for debugging.

Table 13-34 Force Event Register for Auto CMD Error Status

Bit	Name	Type	Reset	Description
15..8	Reserved	R	0	-
7	F_CMD_NO_EX_BY_CMD12	W	0	Force event for the Command Not Executed by Auto CMD12 Error
6..5	Reserved	R	0	.
4	F_CMD_IDX_ERR	W	0	Force event for the Auto CMD Index Error
3	F_CMD_END_BIT_ERR	W	0	Force event for the Auto CMD End Bit Error
2	F_CMD_CRC_ERR	W	0	Force event for the Auto CMD CRC Error
1	F_CMD_TIMEOUT_ERR	W	0	Force event for the Auto CMD Timeout Error
0	F_CMD12_NO_EX	W	0	Force event for the Auto CMD12 Not Executed

13.5.26 Force Event Register for Error Interrupt Status

Offset: 0x52

The Force Event register is not a physical register. It is an address to which the error interrupt status register can be written. This Force Event register is for debugging only. The effect of writing to this address will be reflected in the error interrupt status register if the corresponding bit of the error interrupt status enable register is set.

Table 13-35 Force Event Register for Error Interrupt Register

Bit	Name	Type	Reset	Description
15..13	Reserved	R	0	-
12	F_AHB_RESP_ERR	W	0	Force Event for the AHB response Error
11..10	Reserved	R	0	-
9	F_ADMA_ERR	W	0	Force Event for the ADMA Error
8	F_AUTO_CMD_ERR	W	0	Force Event for the Auto CMD Error
7	F_CUR_LIM_ERR	W	0	Force Event for the Current Limit Error
6	F_DATA_END_BIT_ERR	W	0	Force Event for the Data End Bit Error
5	F_DATA_CRC_ERR	W	0	Force Event for the Data CRC Error
4	F_DATA_TIMEOUT_ERR	W	0	Force Event for the Data Timeout Error

Bit	Name	Type	Reset	Description
3	F_CMD_IDX_ERR	W	0	Force Event for the Command Index Error
2	F_CMD_END_BIT_ERR	W	0	Force Event for the Command End Bit Error
1	F_CMD_CRC_ERR	W	0	Force Event for the Command CRC Error
0	F_CMD_TIMEOUT_ERR	W	0	Force Event for the Command Timeout Error

13.5.27 ADMA Error Status Register

Offset: 0x54

When the ADMA error interrupt is occurred, adma_err_st_r holds the ADMA state and the ADMA system address register holds the address around the error descriptor. When an error occurs, the host driver should watch the ADMA state to identify the error descriptor address.

Table 13-36 ADMA Error Status Register

Bit	Name	Type	Reset	Description
7..3	Reserved	R	0	-
2	Adma_len_err	RC	0	ADMA Length Mismatch Error This error occurs under the following two conditions: 1. When Block Count Enable is set, the total data length specified by the Descriptor table will be different from the one specified by the Block Count and Block Length. 2. The total data length cannot be divided by the block length.
1..0	ADMA_ERR_ST_R	RC	0	ADMA Error State This field indicates the state of ADMA when an error occurs during an ADMA transfer "11": ST_TFR (transfer data), SDR (Table 13-2) points next of the error descriptor "10": Never used "01": ST_FDR (fetch descriptor), SDR (Table 13-2) points the error descriptor "00": ST_STOP (stop DMA), SDR (Table 13-2) points next of the error descriptor

13.5.28 ADMA System Address Register

Offset: 0x58

Table 13-37 ADMA System Address Register

Bit	Name	Type	Reset	Description
31..0	ADMA_LO_ADDR_R	RW	0	32-bit ADMA System Address The 32-bit address descriptor uses the lower 32-bit of the ADMA system address register. ADMA ignores the lower 2-bit of this register and assumes it to be 00.

13.5.29 Preset Value Register

Offset: 0x60..0x6F

Table 13-38 Preset Value Register

Offset	Reset	Preset Value Register	Signal Voltage
0x60	0x0200	Preset value for initialization	3.3V
0x62	0x0008	Preset value for the default speed	3.3V
0x64	0x0004	Preset value for the high speed	3.3V
0x6E..0x66		Reserved	-

When Preset Value Enable in the Host Control 2 register (HC2, 13.5.23) is set to '1', SDCLK Frequency Select in the Clock Generator Select of the Clock Control register (CCR, 13.5.13) will be automatically set based on the selected bus speed mode. This means that the Host driver does not need to set these fields when preset is enabled. A Preset Value for Initialization (PV0, 13.5.29) will not be selected by the bus speed mode. Before starting the initialization sequence, the Host driver needs to set a clock preset value to SDCLK Frequency Select in the Clock Control register. Preset Value Enable can be set after the initialization procedure is complete.
Table 13-39 lists the speed mode selections used for the Preset Value register.

Table 13-39 Speed Mode Selection

Selected Bus Speed Mode	High Speed Enable (Host Control1)
Default speed	0
High speed	1

Table 13-40 Preset Value Register Based on Speed Mode

Bit	Name	Type	Reset	Description
15..10	Reserved	R	0	-
9..0	SDCLK_FREQ_SEL	R	**Bus speed** / **value** Init / 0x200 Default / 0x008 High / 0x004	SDCLK Frequency Select Value The 10-bit preset value for setting the SDCLK Frequency Select in the Clock Control The register is described by a host system.

13.5.30 Host Controller Version Register

Offset: 0xFE

Table 13-41 Host Controller Version Register

Bit	Name	Type	Reset	Description
15..8	VENDOR_VER_NUM	R	0x00	Vendor Version Number This status is reserved for the vendor version number. The host driver should not use this status.
7..0	SPEC_VER_NUM	RW	0x01	Specification Version Number This field shows the specification version of the supportable host controller. 0x00: SD Host Specification Version 1.00 0x01: SD Host Specification Version 2.00 0x02: SD Host Specification Version 3.00 (only SDXC

Bit	Name	Type	Reset	Description
				supported) Others: Reserved

13.5.31 Vendor-defined Register0

Offset: 0x100

Table 13-42 Vendor-defined Register0

Bit	Name	Type	Reset	Description
31..28	Reserved	R	0	-
27..24	NCRC	RW	0	Write CRC Status Wait Cycle The host controller is used to set five SCLK clock cycles for the specifications and round-chip effect. Users can add the wait cycle if the SD card has no meet the specification.
23..17	Reserved	R	0	-
16	INT_EDGE_SEL	RW	0	Internal SCLK Edge Selection 1: The CMD and DAT line output at the rising edge of SCLK. 0: The CMD and DAT line output at the falling edge of SCLK.
15..14	Reserved	R	0	-
13..8	P_LAT_OFF	RW	0	Pulse Latch Offset When the Host Controller uses the pulse latch to sample the read data and response, users need to set the latch offset to correctly sample the value. The value set should be small than the SDCLK Frequency Select. 0x3F: Latch value at the 63rd sdclk1x (100MHz) rising edge after the SCLK edge … … 0x01: Latch value at the 1st sdclk1x (100MHz) rising edge after the SCLK edge 0x00: Latch value at the SCLK edge
7..1	Reserved	R	0	-
0	P_LAT_EN	RW	1	Pulse Latch Enable This bit is used to select the read data and respond the sampling function. Pulse Latching The host controller use the functions required for adjusting p_lat_off to correctly sample the values. Users should take this value into conditions with the round-chip effect to set p_lat_off. 1: Use the pulse latching function for the read data and response 0: Use the multiphase DLL latching for the read data and response (**not supported**)

13.5.32 Vendor-defined Register1

Offset: 0x104

Table 13-43 Vendor-defined Register1

Bit	Name	Type	Reset	Description

Bit	Name	Type	Reset	Description
31..25	Reserved	R	0	-
24	CMD_CONFLICT_EN	RW	0	Command Conflict Checker Enable The host controller can check the CMD line conflict error. Users can set this bit to '1' to enable the checker.
23..19	Reserved	R	0	-
18..16	NSB	RW	0	N_{SB} Timing of SD Specification The host controller is set to five SCLK clock cycles for the specification. Users can add the busy wait cycle if the SD card does not meet the specification.
15..12	Reserved	R	0	-
11..8	NCR	RW	0	N_{CR} Timing of SD Specification The host controller is set to 64 SCLK clock cycles for the specification. Users can add the response wait cycle if the SD card does not meet the specification.
7..3	Reserved	R	0	-
2	MMC_BOOT_ACK_EN	RW	0	MMC Booting Mode Acknowledge Enable The MMC specification defines the Booting Acknowledge that can be disabled. Users can enable the function.
1..0	MMC_BOOT	RW	0	MMC Booting Mode Selection The MMC specification defines two Booting modes and Bus Test modes. Users need to set the function to achieve these modes. The read data in the boot mode can be read from the data port or transferred by SDMA. To exit from the boot mode, the "software reset for all" must be set after clearing this register to '0'. (When in the alternative boot mode, the CMD0 for terminating the boot mode should be issued after setting "software reset for all".) The MMC version 4.3 does not support ultra-speed in boot modes, including SDR50, SDR104 and DDR50. "11": MMC Bus Test mode "10": MMC Alternative Boot mode "01": MMC Boot mode "00": Normal mode

13.5.33 Vendor-defined Register2

Offset: 0x108

Table 13-44 Vendor-defined Register2

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	CLK_CTRL_SW_RST	RWC	0	Clock Control Software Reset Users can reset the clock control of the host controller. When this bit is set to '1', the clock control will reset to the default value.

13.5.34 Vendor-defined Register3

Offset: 0x10C

Table 13-45 Vendor-defined Register3

Bit	Name	Type	Reset	Description
31..0	Reserved	R	0x1F000808	-

13.5.35 Vendor-defined Register5

Offset: 0x114

Table 13-46 Vendor-defined Register5

Bit	Name	Type	Reset	Description
31..4	Reserved	R	0	-
3..0	DB_TIMEOUT	RW	0	Card Insertion De-bounce Cycle 0: 2^9 /96MHz 1: 2^{10}/96MHz … 15: 2^{24}/96MHz

13.5.36 Vendor-defined Register6

Offset: 0x118

Table 13-47 Vendor-defined Register6

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	HBURST_INCR	RW	1	AHB Bus Burst Type 1: AHB master uses INCR as the AHB burst type. 0: AHB master uses SINGLE and INCR4 as the AHB burst type.

13.5.37 Vendor-defined Register7

Offset: 0x11C

Table 13-48 Vendor-defined Register7

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	AHB_RESP_ERR_STS	RWC	0	AHB Master Response Error Status This bit is set when the AHB master receives an error type response.

13.5.38 Vendor-defined Register8

Offset: 0x120

Table 13-49 Vendor-defined Register8

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	AHB_RESP_ERR_STS_EN	RW	0	AHB Master Response Error Status Enable 1: Enable the AHB master response error status 0: Disable the AHB master response error status

13.5.39 Vendor-defined Register9

Offset: 0x124

Table 13-50 Vendor-defined Register9

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	AHB_RESP_ERR_STS	RW	0	AHB Master Response Error Signal Enable 1: Enable the interrupt generation when the AHB master response status is set. 0: Disable the interrupt generation when the AHB master response status is set.

13.5.40 DMA Handshake Enable Register

Offset: 0x128

Table 13-51 DMA Handshake Enable Register

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	DMA_HSK_EN	RW	0	DMA Handshake Enable 1: Enable the DMA handshake protocol for the data port access 0: Disable the DMA handshake protocol for the data port access Note: When this register is set to '1', buf_r_rdy_r and buf_w_rdy_r (NISR, 13.5.16) will not be set and become ineffective. Data will be transferred by the external DMA and the flow control will be controlled by the DMA handshake protocol.

13.5.41 Cipher Mode Control Register

Offset: 0x180

This register is the configurable register for the CPRM function. When the CPRM function is used, this register will be used to select the mode of cipher function to encrypt or decrypt.

Table 13-52 Cipher Mode Control Register

Bit	Name	Type	Reset	Description
31..11	Reserved	R	0	-
10	SWAP_HL	RW	0	This bit is effective only in the auto_c2_ecbc_en mode and

Bit	Name	Type	Reset	Description
				auto_c2_dcbc_en mode. When this bit is set in the auto_c2_ecbc_en mode: The high-word and low-word of the encrypted data will be swapped before being written to TX FIFO. When this bit is set in the auto_c2_dcbc_en mode: The high-word and low-word of the encrypted data will be swapped before decryption.
9	CH_ENDIAN	RW	0	Change Endianness When this bit is set to the auto_c2_ecbc_en mode: The endianness of the encrypted data will be changed before being written to TX FIFO. When this bit is set to the auto_c2_dcbc_en mode: The endianness of data from RX FIFO will be changed before decryption.
8	SEC_ACCESS_EN	RW	0	Secret Constant Table Access Enable This bit must be enabled before writing or reading the secret constant table. Once this bit is enabled, the firmware will always access from the very beginning of the secret constant table.
7	AUTO_C2_DCBC_EN	RW	0	Auto C2 Decryption with C-CBC Mode Enable In this mode, data will be automatically decrypted and sent to the 64-bit line buffer. SD Card Controller only supports the data lengths that are multiples of 8 bytes in this mode.
6	AUTO_C2_ECBC_EN	RW	0	Auto C2 Encryption with C-CBC Mode Enable In this mode, data written to the 64-bit buffer will be automatically encrypted and sent to TX FIFO. SD Card Controller only supports data lengths which are the multiples of 8 bytes in this mode.
5	RNGC2_G_EN	RW	0	C2 Random Number Generator Enable
4	C2_DCBC_EN	RW	0	C2 Decryption with C-CBC Mode Enable
3	C2_D_EN	RW	0	C2 Decryption with ECB Mode Enable
2	C2ECBC_EN	RW	0	C2 Encryption with C-CBC Mode Enable
1	C2_E_EN	RW	0	C2 Encryption with ECB Mode Enable
0	C2_G_EN	RW	0	C2 One Way Function Enable

13.5.42 Cipher Mode Status Register

Offset: 0x184

Table 13-53 Cipher Mode Status Register

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	CP_RDY	RWC	0	Cipher is ready. When this bit is set to '1', reading LWOD (13.5.48) and HWOD (13.5.49) will get cipher or plain text.

13.5.43 Cipher Mode Signal Enable Register

Offset: 0x188

This register is used to select which interrupt status is notified to the host system as the interrupt. These interrupt statuses share the same interrupt line.

Table 13-54 Cipher Mode Status Register

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	CP_RDY_SIG_EN	RW	0	Cipher ready signal enable.

13.5.44 Low Word of Input Data Register

Offset: 0x18C

Table 13-55 Low Word of Input Data Register

Bit	Name	Type	Reset	Description
31..0	R_R	RW	0	Input port for the input data bits[31..0]

13.5.45 High Word of Input Data Register

Offset: 0x190

Table 13-56 High Word of Input Data Register

Bit	Name	Type	Reset	Description
31..0	L_R	RW	0	Input port for the input data bits[63..32]

13.5.46 Low Word of Input Key Register

Offset: 0x194

Table 13-57 Low Word of Input Key Register

Bit	Name	Type	Reset	Description
31..0	KEY_B_R	RW	0	Input port for the input key bits[31..0]

13.5.47 High Word of Input Key Register

Offset: 0x198

Table 13-58 High Word of Input Key Register

Bit	Name	Type	Reset	Description
31..0	KEY_A_R	RW	0	Input port for the input key bits[63..32]

13.5.48 Low Word of Output Data Register

Offset: 0x19C

Table 13-59 Low Word of Output Data Register

Bit	Name	Type	Reset	Description
31..0	CP_OUT_LOW_R	R	0	Output port for the output data bits[31..0]

13.5.49 High Word of Output Data Register

Offset: 0x1A0

Table 13-60 High Word of Output Data Register

Bit	Name	Type	Reset	Description
31..0	CP_OUT_HI_R	R	0	Output port for the output data bits[63..32]

13.5.50 Secret Constant Table Data Port

Offset: 0x1A4

CPRM requires the 256-byte secret constant released from 4C entity. To initialize the secret constant table, the firmware should set **sec_access_en** (CMCR, 13.5.41, bit 8) first and then send continual 256-byte data by writing SCTDP (13.5.50) 256 times. Only byte access is allowed.

Table 13-61 Secret Constant table Data Port

Bit	Name	Type	Reset	Description
7..0	SEC_DATA_PORT	RW	0	Secret constant table data port

13.6 Initialization/Application Information

13.6.1 Program Sequence

This chapter defines the flow chart of the basic program sequence.

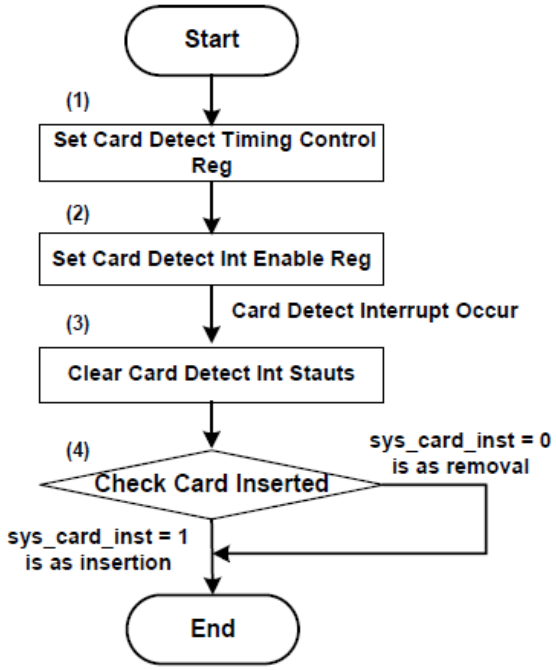
13.6.1.1 Card Detection

Figure 13-8 shows the flow of detecting a card. Each step is executed as follows:

1. Set the vendor-defined register5 (VR5, 13.5.35) to decide the debouncing time for card detection
2. To enable the normal interrupt status enable register and normal interrupt signal register for card detection, write ‘1’ to the following bits:
   card_insert_st_en and **card_remove_st_en** in the normal interrupt status enable register (NISER, 13.5.18)
   card_insert_sig_en and **card_remove_sig_en** in the normal interrupt signal enable register (NISEN, 13.5.20)

3. Write ‘1’ to clear the interrupt status (NISR, 13.5.16) when the host driver detects the card detection interrupt.
   If the **card_insert_r** interrupt is generated, write ‘1’ to clear this status register (NISR, 13.5.16).
   If the **card_remove_r** interrupt is generated, write ‘1’ to clear this status register (NISR, 13.5.16).
4. Check the present state register (PSR, 13.5.9) to check whether the card is inserted or removed. When **sys_card_ins** is set to ‘1’, the host driver can continue supplying the power and clock to the card.

Figure 13-8 Card Detect Sequence

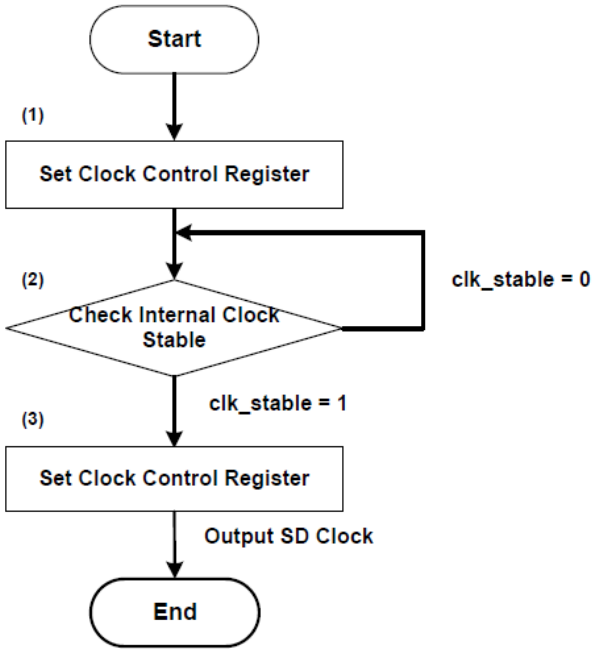


13.6.1.2 SD Clock Control

Figure 13-9 shows the flow of detecting an SD card. Each step is executed as follows:

1. Set **inter_clk_en** and **sd_clk_sel** in the clock control register (CCR, 13.5.13) to generate clock.
2. Check **clk_stable** in the clock control register (CCR, 13.5.13) to determine whether the clock is stable or not.
3. Set **sd_clk_en** in the clock control register (CCR, 13.5.13) to output the clock to the **io_sd_clk** pin. If the host driver has to stop the SD clock, **sd_clk_en** should be set to ‘0’. The host controller then stops the SD clock. When the host driver needs to change the SD clock, **sd_clk_en** should also be set to ‘0’.

Figure 13-9 SD Clock Control Sequence



13.6.1.3 Card Initialization and Identification

This section introduces the initialization and identification of the SD, SDIO, and MMC cards. Figure 13-10 and Figure 13-11 shows the flow of the card initialization and the flow of the card identification.

1. The SD bus mode is selected by resetting CMD0.
2. Issue CMD8 to check the high-capacity SD memory card
3. The legacy cards (Not the SD card) does not respond to CMD8. Set the F8 flag to ‘0’ (Used in step11), go to step5. Only cards with version 2.0 or higher can respond to CMD8. The host controller needs to check the validity of CRC of the response, verify VHS, check the pattern in an argument that equals to VCA and check the pattern in the response. If the card response is okay, then set the F8 flag to ‘1’ and go to step5. If the response check fails, go to step4.
4. If the initialization fails, the host driver should retry again.
5. Issue CMD5 to obtain SDIO OCR by setting the voltage window to ‘0’ in the argument.
6. If no response is detected, the card will not have the SDIO function. Set the SDIO flag to ‘0’ and go to step11. If the response is okay, go to step7. If the response has an error, set the SDIO flag to ‘0’, go to step10. The SDIO flag verifies the initialization of the SDIO functions.
7. Issue CMD5 to start the initialization by setting the voltage window. If the supplied voltage does not match with the voltage windows of the card, the card will enter an inactive state and will not return to the response.
8. If no response or error response is received, set the SDIO flag to ‘0’, go to step10. If the response is okay, go to step9.
9. Check the busy status bit in the response. If the busy status is released, set the SDIO flag to ‘1’, and go to step10. If the busy status is not released, repeat step7 until the

busy status is released. When the detection timeout of 1 second exits from a loop, set the SDIO flag to '0', go to step10.

10. When all responses at step6 and step8 are valid, the **MP** (Memory Present) flag in the response can be checked. If the response is okay and **MP**='0', go to step28. Otherwise, go to step11.
11. Check the F8 flag set in step3. If F8='1', go to step21; otherwise, go to step12.
12. Receiving OCR can issue an ACMD41 command with a voltage window (Bit 23..0) if the argument is set to '0'. CMD55 should be issued before the ACMD41 command.
13. If the ACMD41 response is not received, the card will not be a SD card, and go to step14. If the card responds to ACMD41, go to step17; otherwise, go to step29.
14. In this step, the card may be the MMC card, and the host driver can issue the CMD1 by setting the supply voltage to the voltage window.
15. If the CMD1 response indicates that the card is busy or the host omits the voltage range, the host driver will repeat to issue CMD1. If the OCR response is a non-compatible voltage range, go to step29.
16. The host recognizes that the card is a Multi Media Card (MMC) and quits the card initialization.
17. The memory portion starts the initialization by issuing ACMD41 by setting the supply voltage to the voltage window. If the supplied voltage does not match the voltage window of card, the card will enter the inactive state and will not return the response.
18. If no response or error response is received, go to step29. If a response is received, go to step19.
19. Check the busy status in a response. If busy is released, go to step20.
20. If the host recognizes that the card is Version 1.xx Standard Capacity SD Memory Card, go to step32.
21. OCR is available by issuing ACMD41 and setting the voltage window (Bit23..0) in the argument, and it is set to '0'. CMD55 should be issued before ACMD41.
22. If the card responds to CMD55, it may also respond to CMD41. If the response of ACMD41 is OK, go to step23. Otherwise, go to step29.
23. The memory portion starts the initialization by issuing ACMD41 and setting the supply voltage to the voltage window. If the supplied voltage does not match with the voltage window of card, the card will enter the inactive state and does not return the response. HCS in the argument is set to '1', which indicates that the High Capacity Memory Card is supported.
24. If no response or error response is received, go to step29. If a good response is received, go to step25.
25. Check the busy status in the response. If busy is released, go to step26. While the busy is indicated, repeat to issue ACMD41.
26. CCS in the response is valid after busy is released. If CCS='0', it indicates the Standard Capacity SD Memory Card and go to step27. If the CCS ='1', it indicates the High Capacity SD Memory Card and go to step28
27. If the host recognizes that the card is Standard Capacity SD Memory Card, go to step32.
28. If the host recognizes that the card is High Capacity SD Memory Card, go to step32.

29. Check the SDIO flag. If SDIO='1', go to step30.
30. The host recognizes that the card is the SDIO card and goes to step33.
31. The host recognizes that the card is unusable.
32. In case of memory card, CMD2 is issued to get CID, and go to step33.
33. CMD3 is issued to get RCA. If the RCA number is '0', the host should issue CMD3 again.

Figure 13-10 Card Initialization and Identification (PartA)

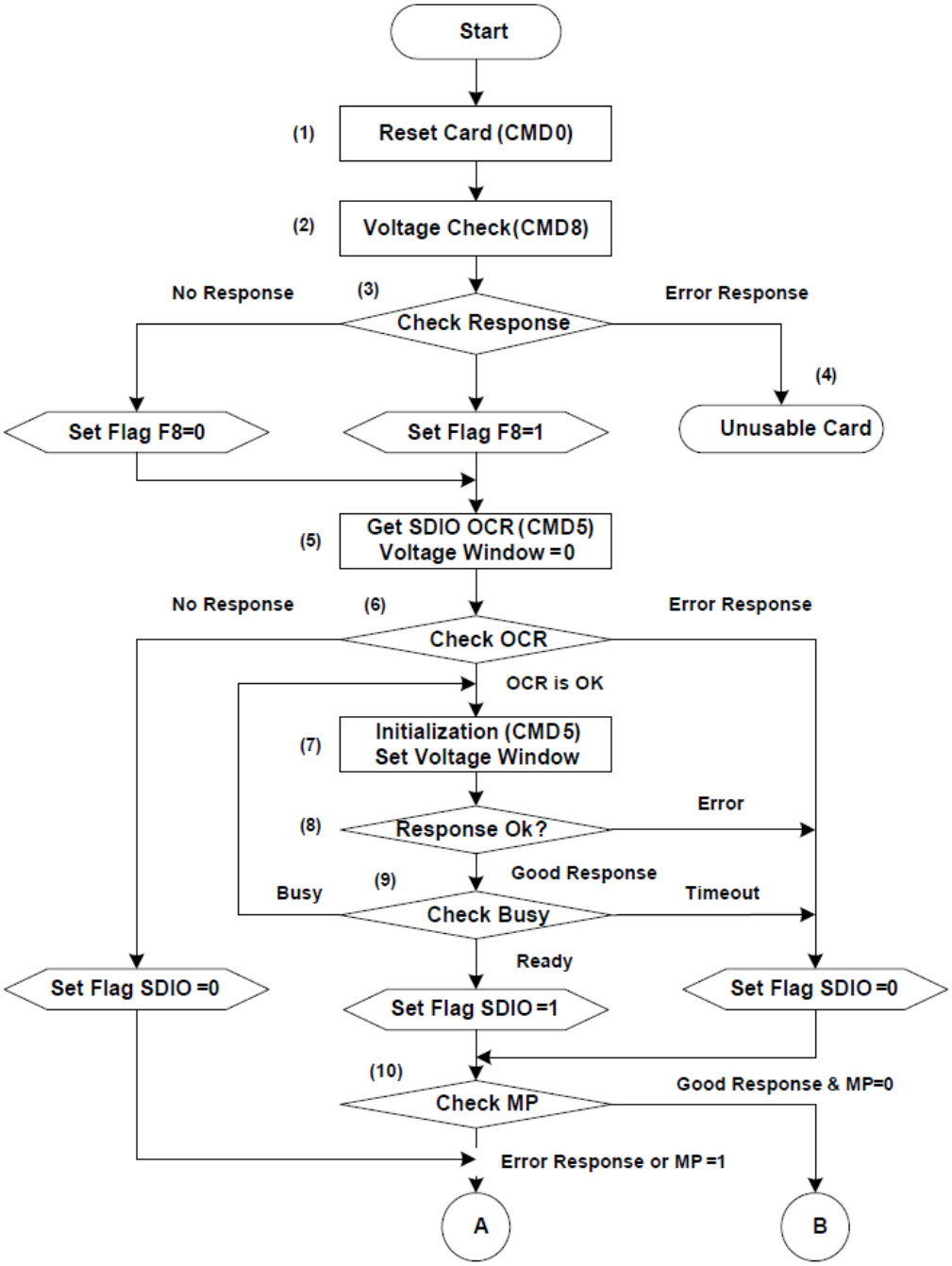
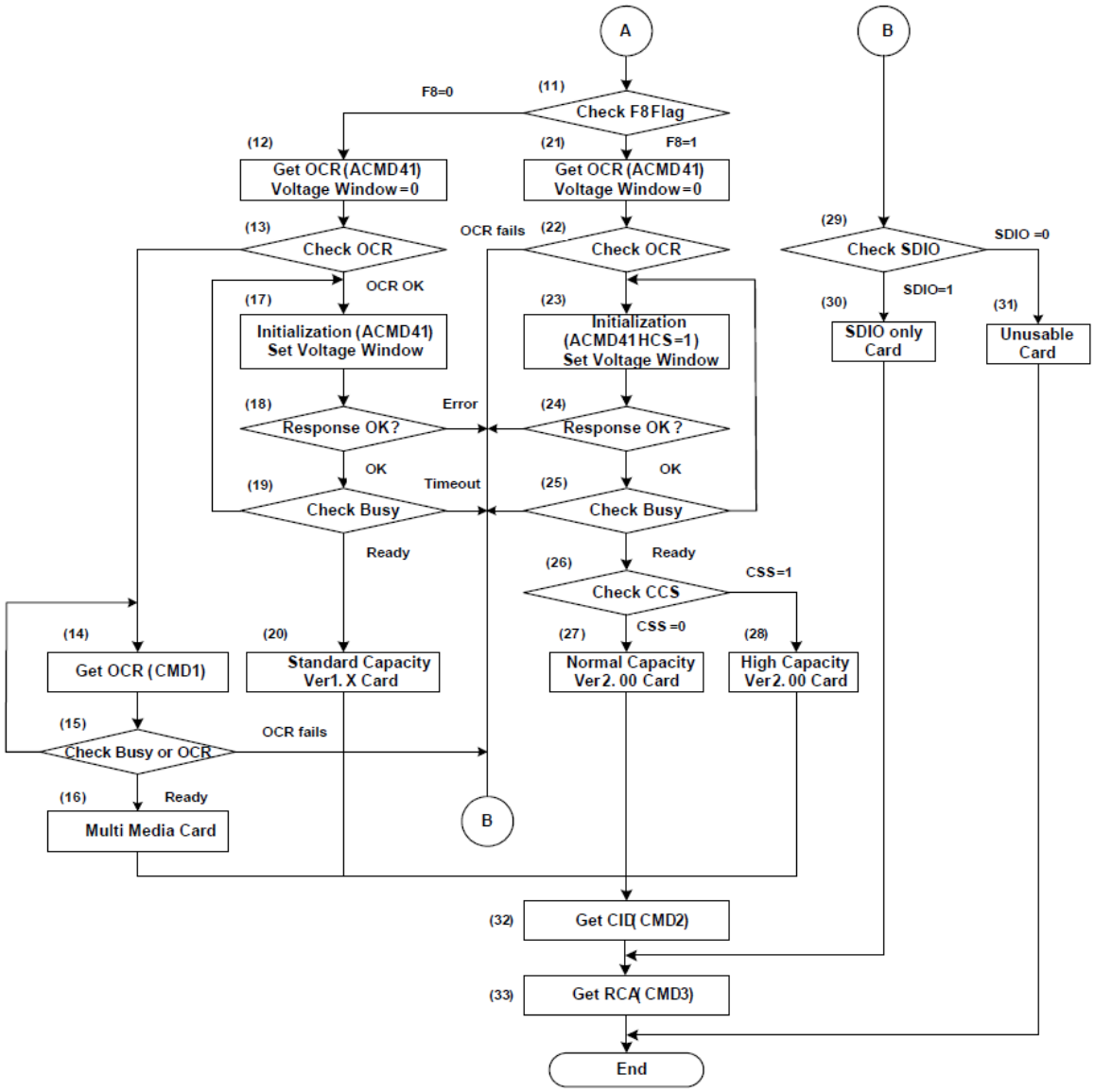


Figure 13-11 Card Initialization and Identification (PartB)



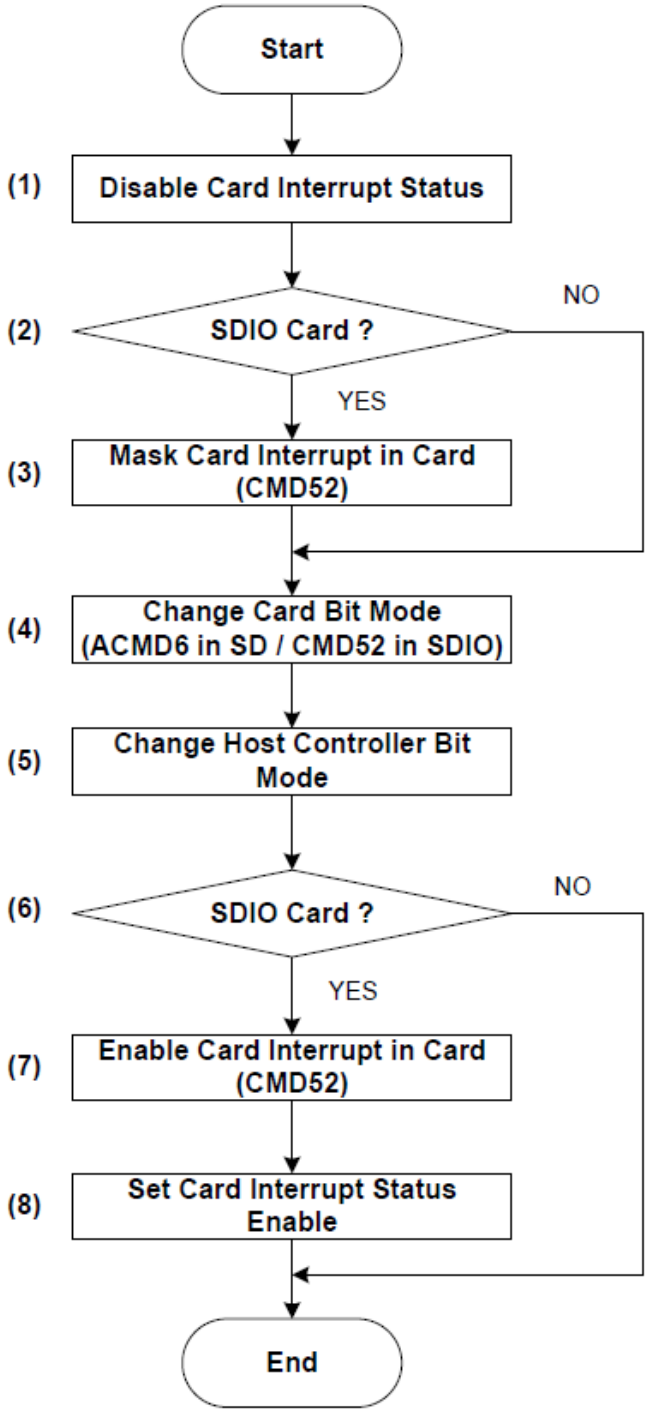
13.6.1.4 Change Bus Width

Figure 13-12 shows the flow of changing the bus width. Each step is executed as follows:

1. Set **card_int_st_en** to '0' in the normal interrupt status enable register (NISER, 13.5.18) to stop detecting the card interrupt when changing the bus width.
2. Detect the SDIO card. If the card is an SDIO card, go to step3. If the card is other card types, go to step4.
3. Set the Interrupt Enable Master (IENM) bit of Card Common Control Register (CCCR) in the SDIO card by using CMD52.
4. For the SD memory card, the bus width mode can be changed by using ACMD6. For the SDIO memory card, the bus width mode can be changed by using CMD52 to set the "**Bus Width**" bits of the Bus Interface Control Register in CCCR. For the MMC card, the bus width mode can be changed by using CMD6.

5. The host controller should set the **data_width** register in the host control register to ‘1’ to activate the 4-bit mode. The host controller should set the **data_width** register to ‘0’ to activate the 1-bit mode.
6. For the SDIO card, go to step7. For other cards, go to “End”.
7. Set the IENM bit of CCCR in the SDIO card to ‘1’ by using CMD52.
8. Set **card_int_st_en** to ‘1’ in the normal interrupt status enable register (NISER, 13.5.18)

Figure 13-12 Change Bus Width Sequence

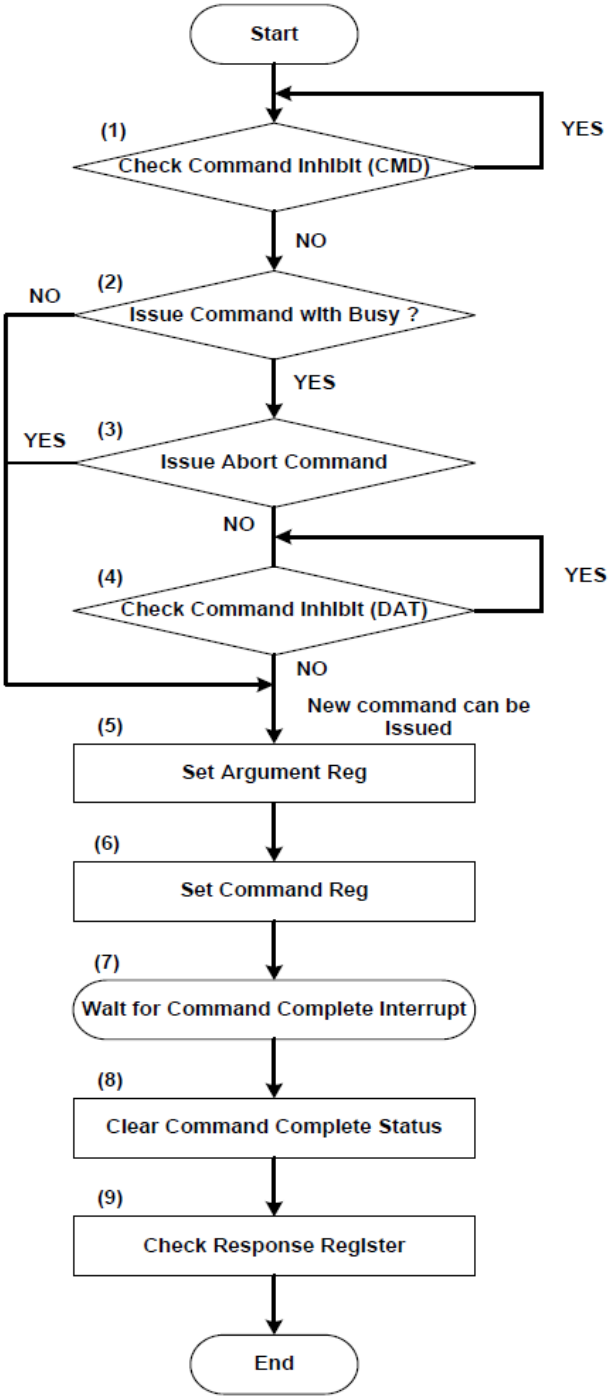


13.6.1.5 Command Transfer without Data Transfer

In this section, the sequence of a command transfer is addressed. Figure 13-13 shows the flow of a command transfer. Each step is executed as follows:

1. Check **cmd_inhibit_c** in the present state register (PSR, 13.5.9). Repeat this step until **cmd_inhibit_c** becomes '0'. When **cmd_inhibit_c** is set to '1', the SD command bus is used and the host driver cannot issue a SD command.
2. If the previous SD command uses the SD data bus with a busy signal, go to step3. If the previous SD command uses the SD data bus without a busy signal, go to step5.
3. If the host driver issues an abort command, go to step5. If the host driver does not issue an about command, go to step4.
4. Check **cmd_inhibit_d** in the present state register (PSR, 13.5.9). Repeat this step until **cmd_inhibit_d** becomes '0'.
5. Set the argument to the argument register
6. Set the command register.
7. Wait for the **cmd_complete_r** interrupt (NISR, 13.5.16). If the **cmd_complete_r** interrupt occurs, go to step8.
8. Write '1' to clear **cmd_complete_r** in the normal interrupt status register.
9. Check the response register to obtain information.

Figure 13-13 Command Sequence



13.6.1.6 Data Transfer without DMA

This section describes the sequence of a data transfer without DMA. Figure 13-14 shows the flow of the data transfer. Each step is executed as follows:

1. Set the length to **blk_size_r** in the block size register (BSR, 13.5.2).
2. Set the count to **blk_cnt_r** in the block count register (BCR, 13.5.3).
3. Set the argument to the argument register (ARG1, 13.5.4).
4. Set the transfer mode register (TMR, 13.5.5).
5. Set the command register (CR, 13.5.6).

6. Wait for the **cmd_complete_r** interrupt[3] (NISR, 13.5.16).
7. When an interrupt occurs, the host driver writes '1' to clear the **cmd_complete_r** status in the normal interrupt status register (NISR, 13.5.16).
8. Check the response register to obtain information (RR, 13.5.7).
9. In case of a write transfer, go to step10. In case of a read transfer, go to step14
10. Wait for the **buf_w_rdy_r** interrupt (NISR, 13.5.16).
11. When an interrupt occurs, the host driver writes '1' to clear the **buf_w_rdy_r** status in the normal interrupt status register (NISR, 13.5.16).
12. Write the block data to the TX buffer for the write transfer.
13. When all blocks are sent, go to step18. If the TX buffer is full (**buf_wen_r**=0) (PSR, 13.5.9) or the block transfer has not completed, go to step10 to repeat the operation .
14. Wait for the **buf_r_rdy_r** interrupt (NISR, 13.5.16).
15. When an interrupt occurs, the host driver writes '1' to clear the **buf_r_rdy_r** status in the normal interrupt status register (NISR, 13.5.16).
16. Read the block data from the RX buffer for the read transfer.
17. When all blocks are received, go to step18. If the RX buffer is empty (**buf_ren_r**=0) (PSR, 13.5.9) or the transfer has not completed, go to step10 to repeat the operation.
18. In the case of single or multiple block transfer, go to step19. In the case of infinite block transfer, go to step21.
19. Wait for **Buffer Write Ready** interrupt (NISR, 13.5.16).
20. When an interrupt occurs, the host driver writes '1' to clear the **tran_complete_r** status in the normal interrupt status register (NISR, 13.5.16).
21. For the infinite block transfer, the abort command is used to stop the transfer according to Section 13.6.1.9.

[3] In some cases, the "Buffer Read Ready" interrupt may assert before "Command Complete" interrupt. If the read data is not read through the data port in time, the data FIFO will be full and SDCLK will stop. This will lead to a command complete timeout because the response cannot be received if SDCLK stops.

Figure 13-14 Data Transfer without DMA

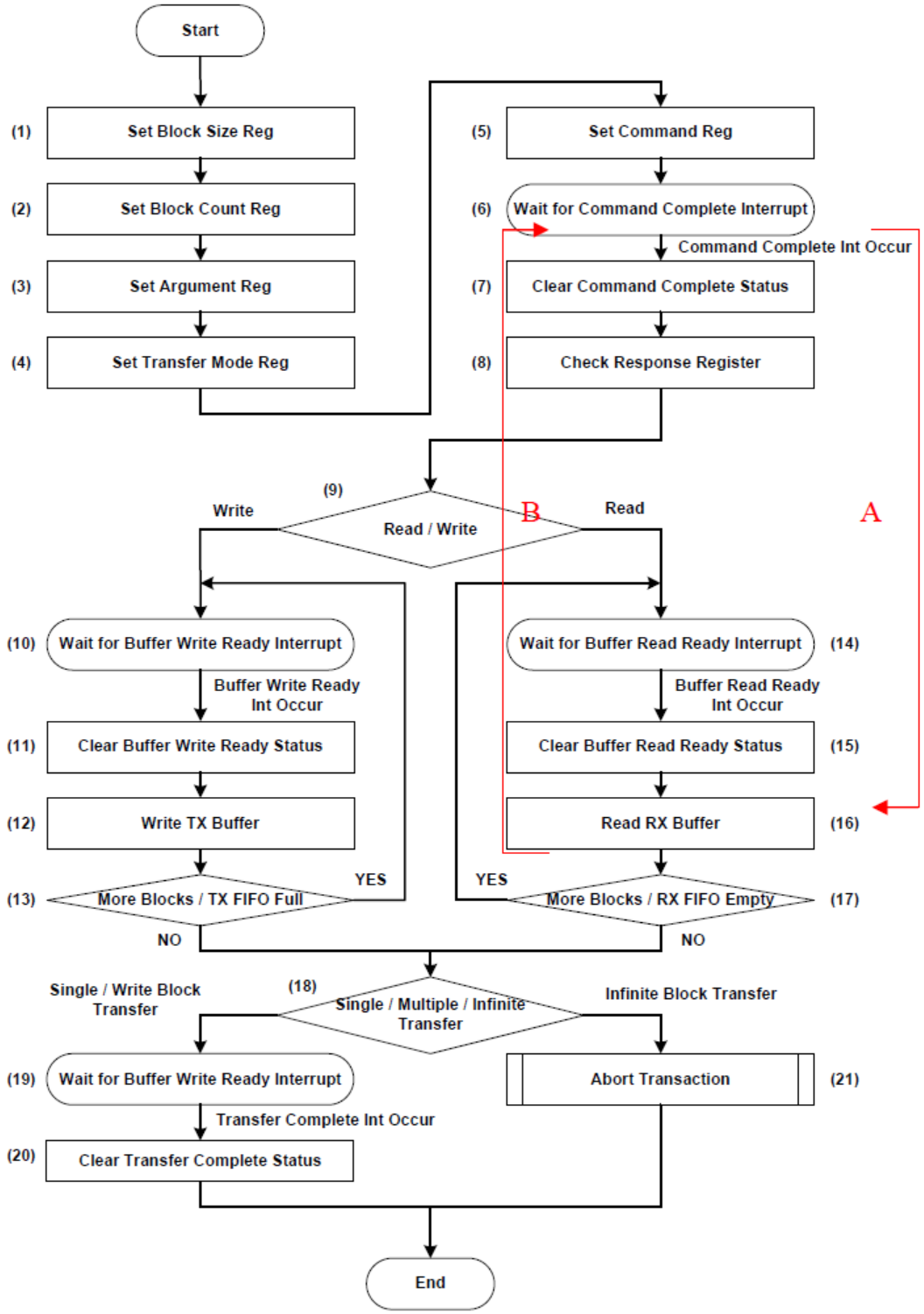
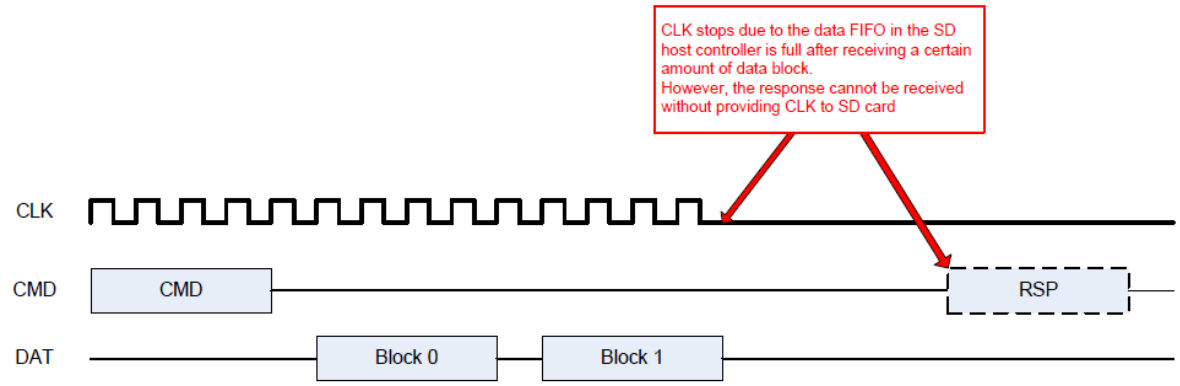


Figure 13-15 Command complete timeout, because of a full FIFO



The solution is to modify the “Not using DMA” sequence:
The buffer read ready interrupt should be serviced when it asserts even before the command complete asserts

For the read operation:

(a) At step6, if the buffer read ready interrupt asserts and command complete interrupt does not assert (NISR, 13.5.16), go to step15.
(b) After finishing step16, if command complete is not received go back to step6

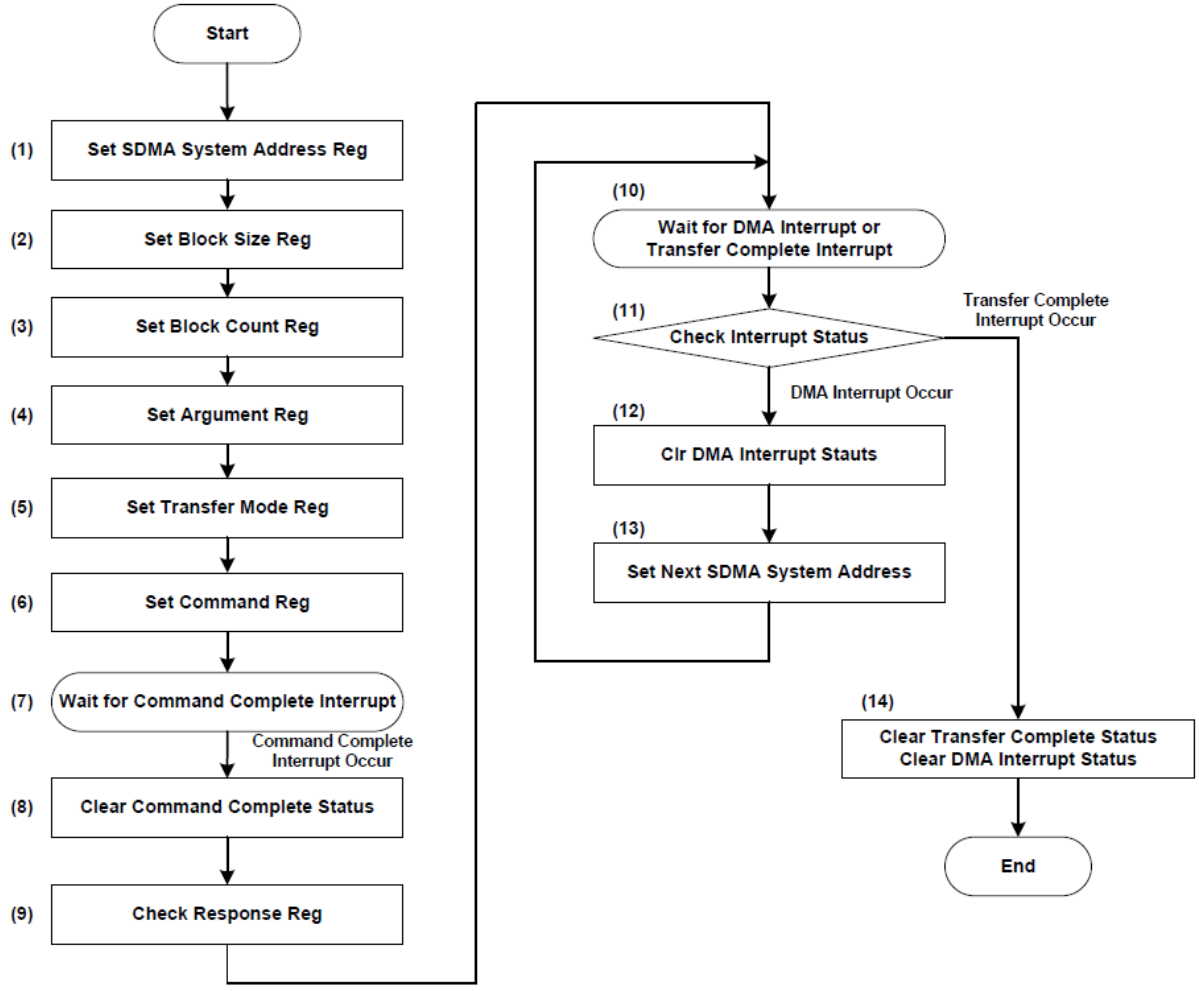
13.6.1.7 Data Transfer with SDMA

Figure 13-16 shows the flow of the data transfer with SDMA. The sequence for using SDMA is shown as follows:

1. Set the data address of the system memory to the SDMA system address register (SDMADR, 13.5.1).
2. Set the length to **blk_size_r** in the block size register (BSR, 13.5.2).
3. Set the count to **blk_cnt_r** in the block count register (BCR, 13.5.3).
4. Set the argument to the argument register (ARG1, 13.5.4).
5. Set the transfer mode register (TMR, 13.5.5).
6. Set the value to the command register (CR, 13.5.6).
7. Wait for the **cmd_complete_r** interrupt (NISR, 13.5.16).
8. When an interrupt occurs, the host driver writes ‘1’ to clear the **cmd_complete_r** status in the normal interrupt status register (NISR, 13.5.16).
9. Check the response register and obtain information (RR, 13.5.7).
10. Wait for the **dma_int_r** interrupt when reaching the SDMA transfer boundary or wait for the **tran_complete_r** interrupt when the data transfer has been finished (NISR, 13.5.16).
11. If the **trans_complete_r** interrupt occurs, go to step14. If the **dma_int_r** interrupt occurs, go to step12. The **trans_complete_r** interrupt has higher priority than the **dma_int_r** interrupt (NISR, 13.5.16).
12. When the **dma_int_r** interrupt occurs, write ‘1’ to clear **dma_int_r** in the normal interrupt status register (NISR, 13.5.16).

13. Set the next system address to the SDMA system address register (SDMADR, 13.5.1) and go to step10.
14. Write ‘1’ to clear **tran_complete_r** and **dma_int_r**in the normal interrupt status register (NISR, 13.5.16).

Figure 13-16 Data Transfer with SDMA



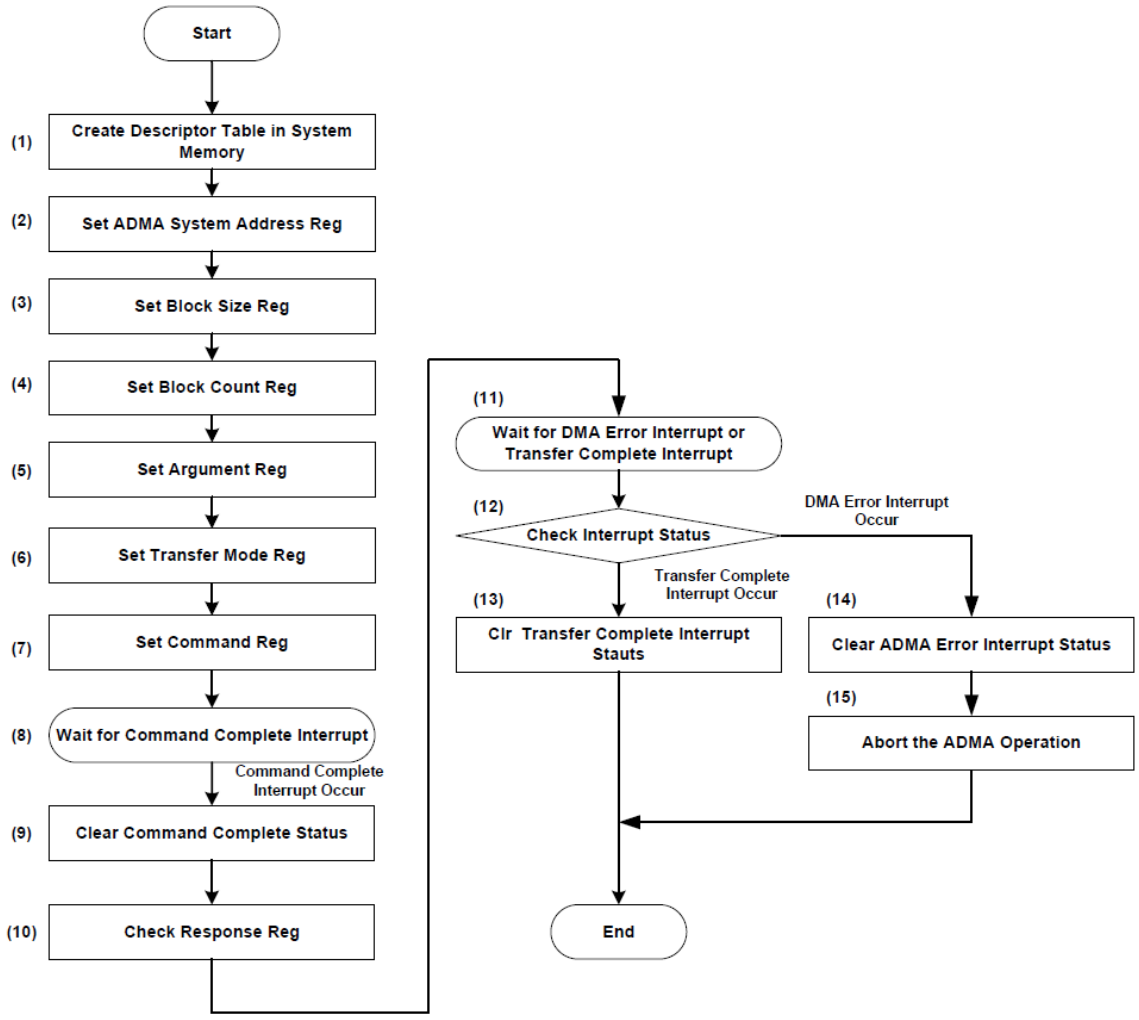
13.6.1.8 Data Transfer with ADMA

Figure 13-17 shows the flow of the data transfer with ADMA. The sequence for using ADMA is shown as follows:

1. Create a table for the ADMA descriptor in the system memory
1. Set the address of the descriptor table to the SDMA system address register (SDMADR, 13.5.1).
2. Set the length to **blk_size_r** in the block size register (BSR, 13.5.2).
1. Set the count to **blk_cnt_r** in the block count register (BCR, 13.5.3).
3. Set the value of argument to the argument register (AR, 13.5.4).
4. Set the value to the transfer mode register (TMR, 13.5.5).
5. Set the value to the command register (CR, 13.5.6).
6. Wait for the **cmd_complete_r** interrupt (NISR, 13.5.16).
7. When an interrupt occurs, the host driver writes ‘1’ to clear the **cmd_complete_r** status in the normal interrupt status register (NISR, 13.5.16).

2. Check the response register to obtain information (RR, 13.5.7).
8. Wait for the **dma_err_r** interrupt in the Error interrupt status register (EISR, 13.5.17) when the ADMA error interrupt occurs or wait for the **tran_complete_r** interrupt when the data transfer is finished (NISR, 13.5.16).
9. If the **tran_complete_r** interrupt occurs, go to step13. If the **dma_err_r** interrupt occurs, go to step14.
10. Write '1' to clear **tran_complete_r** in the normal interrupt status register (NISR, 13.5.16).
11. Write '1' to clear **dma_err_r** in the error interrupt status register (EISR, 13.5.17).
12. Abort the ADMA operation. The SD card should issue an abort command to stop the transfer. The host driver should check the ADMA error status register to understand how the ADMA error is generated.

Figure 13-17 Data Transfer with ADMA



13.6.1.9 Abort Transaction

Users can operate an abort transaction by issuing the CMD12 command for the SD card and the CMD52 command for the SDIO card. The host driver can operate in two different transaction stop types. The first type is to stop an infinite block transfer. The second type is

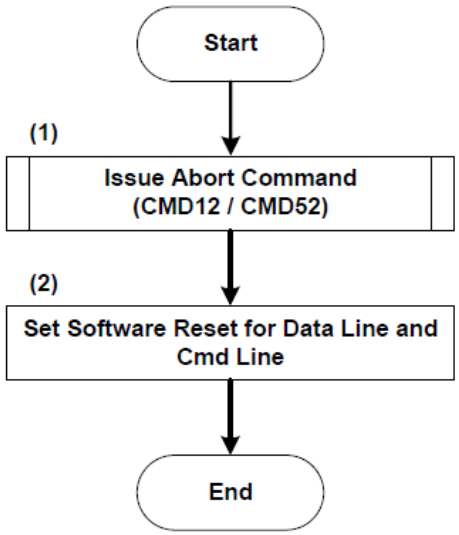
to stop a multi-block transfer. These two abort transactions are described in the following subsections.

Asynchronous Abort

In the asynchronous abort, the host driver will issue an abort command at any time except when **cmd_inhibit_c** is set to '1'. The sequence of the asynchronous abort is shown in Figure 13-18. Each step is executed as follows:

1. Issue an abort command according to section 13.6.1.5.
2. Set **soft_rst_dat** and **soft_rst_cmd** to '1' in the software reset register (SRR, 13.5.15) for resetting the software.

Figure 13-18 Asynchronous Abort Sequence



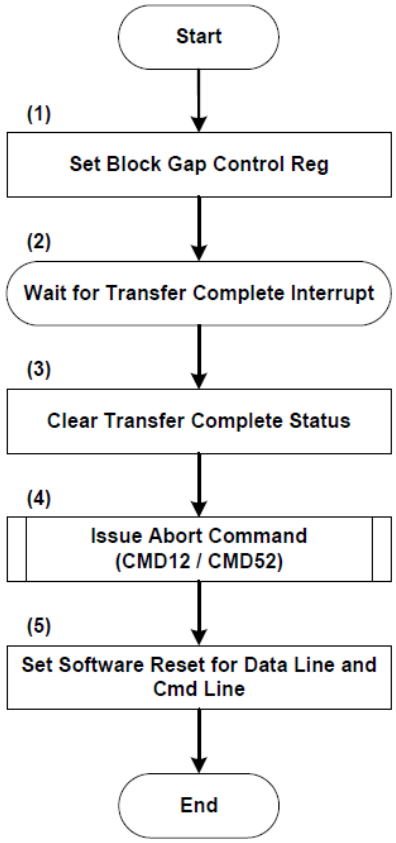
Synchronous Abort

In a synchronous abort, the host driver will issue an abort command after the data transfer, which is stopped by using **sp_blk_gap_req** in the block gap control register (BGCR, 13.5.12). The sequence of the synchronous abort is shown in Figure 13-19.

Each step is executed as follows.

1. Set **sp_blk_gap_req** to '1' to stop the SD transaction (BGCR, 13.5.12).
2. Wait the **tran_complete_r** interrupt (NISR, 13.5.16).
3. When an interrupt occurs, write '1' to clear **tran_complete_r** in the normal interrupt status register (NISR, 13.5.16).
4. Issue an abort command according to section 13.6.1.5.
5. Set **soft_rst_dat** and **soft_rst_cmd** to '1' in the software reset register (SRR, 13.5.15) for resetting the software.

Figure 13-19 Synchronous Abort Sequence



13.6.2 CPRM Sequence

This optional CPRM operation can be operated in the non-auto mode and the auto mode. The following sub-sections will describe the sequences of these modes.

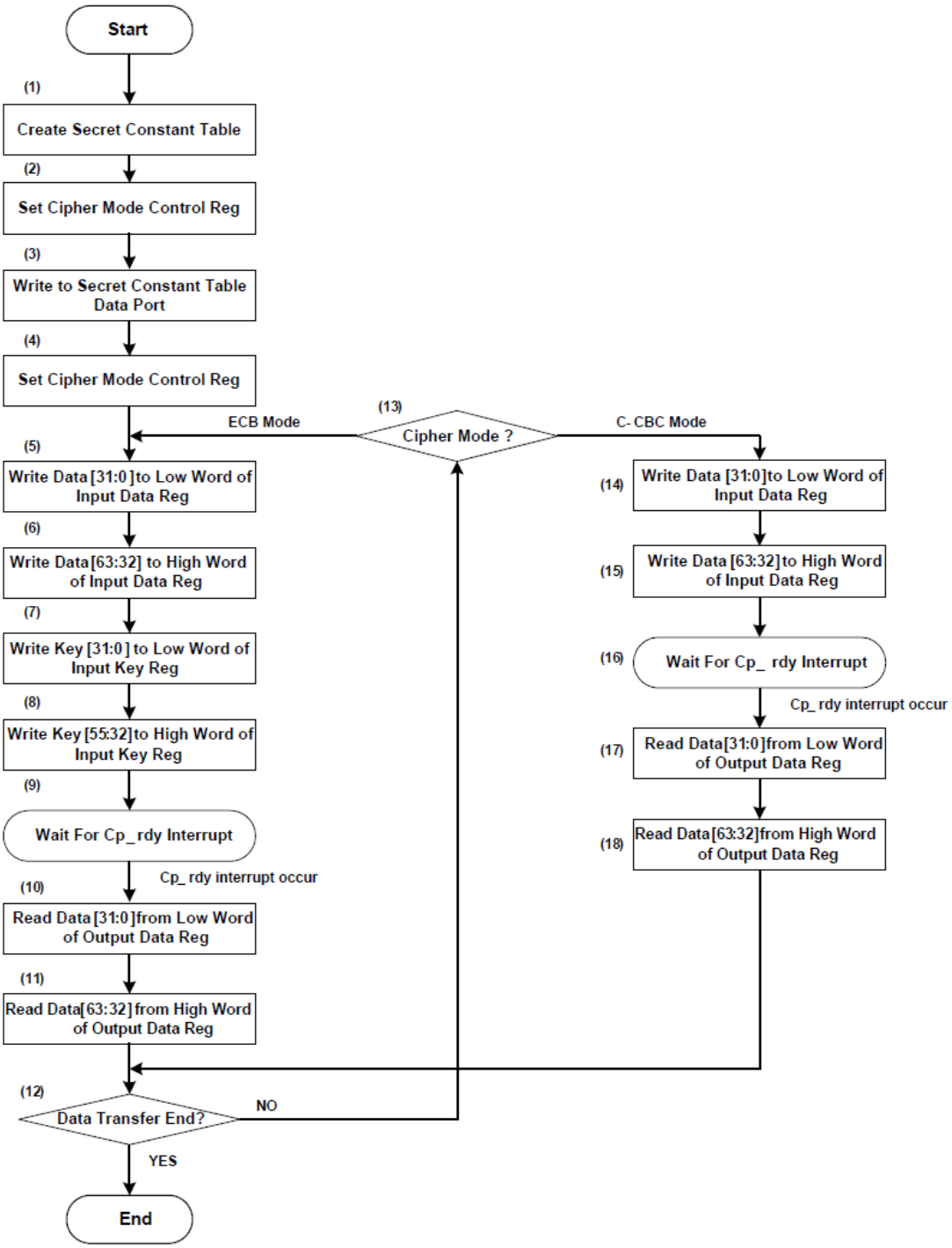
13.6.2.1 Non-auto Mode

The host driver should always follow the programming sequence to make encryption/decryption work correctly. The flow of the non-auto mode is shown in Figure 13-20.

1. Create the Secret Constant Table.
2. Set **sec_access_en** in the cipher mode control register to access the secret constant table.
3. Write the secret constant table to **sec_data_port** in the secret constant table data port.
4. Select the encryption/decryption mode in the cipher mode control register to transfer data. Only one bit is set at anytime.
5. Write data to **R_r** in the low word of the input data register.
6. Write data to **L_r** in the high word of the input data register.
7. Write a key to **key_b_r** in the low word of the input key register.
8. Write a key to **key_a_r** in the high word of the input key register.
9. Wait until **cp_rdy** in the cipher mode status register is ‘1’.
10. Get the output data from **cp_out_low_r** in the low word of the output data register.
11. Get bit output data from **cp_out_hi_r** in the high word of the output data register.

12. If the data transfer is completed, then go to “End”. If the data transfer has additional data, then go to step13.
13. When the cipher mode is the ECB mode, then go to step5. When the cipher mode is the C-CBC mode, go to step14.
14. Write data to **R_r** in the low word of the input data register.
15. Write data to **L_r** in the high word of the input data register.
16. Wait until **cp_rdy** in the cipher mode status register is ‘1’.
17. Get the output data from **cp_out_low_r** in the low word of the output data register.
18. Get bit output data from **cp_out_hi_r** in the high word of the output data register.

Figure 13-20 Non-auto Mode CPRM Sequence



13.6.2.2 Auto Mode

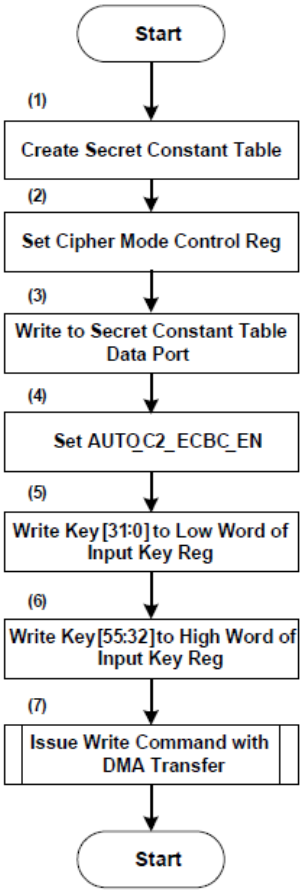
This section describes the auto mode transfer for CPRM. The auto mode can only be valid on the DMA transfer. The following subsections describe the DMA transfer sequence with the C-CBC mode.

Auto_C2_ECBC Mode with ADMA

The host driver should always follow the programming sequence so that the encryption will correctly work. Figure 13-21 shows the flow of the auto_c2_ecbc mode with the ADMA transfer.

1. Create the secret constant table (SCTDP, 13.5.50).
2. Set the **sec_access_en** in the cipher mode control register (CMCR, 13.5.41) to access the secret constant table.
3. Write the secret constant table to sec_data_port in the secret constant table data port (SCTDP, 13.5.50).
4. Set **auto_c2_ecbc_en** in the cipher mode control register (CMCR, 13.5.41) to '1'.
5. Write key to the **key_b_r** in the low word of the input key register (LWIK, 13.5.46).
6. Write key to the **key_a_r** in the high word of the input key register (HWIK, 13.5.47).
7. Send the write command with the ADMA transfer according to Section 13.6.1.8.

Figure 13-21 Auto_C2_ECBC Mode Transfer wieth ADMA



Auto_C2_DCBC Mode with DMA

The host driver should always follow the programming sequence so that the decryption will work correctly. The sequence of the AUTO_C2_DCBC mode with DMA is similar to the AUTO_C2_ECBC mode with DMA.

1. Create the Secret Constant Table (SCTDP, 13.5.50).
2. Set the **sec_access_en** in the cipher mode control register (CMCR, 13.5.41) to access the secret constant table.

3. Write the secret constant table to the **sec_data_port** in the secret constant table data port (SCTDP, 13.5.50).
4. Set **auto_c2_dcbc_en** in the cipher mode control register (CMCR, 13.5.41) to ‘1’.
5. Write key to the **key_b_r** in the low word of the input key register (LWIK, 13.5.46).
6. Write key to the **key_a_r** in the high word of the input key register (HWIK, 13.5.47).
7. Send the read command with the ADMA transfer according to Section 13.6.1.8.

14 USB Device Controller

Offset: 0x90000000

The ANTAIOS includes an USB device controller, which complies with the USB 2.0 specification. It can operate at the high-speed signaling rate of 480 Mbps, full-speed signaling rate of 12 Mbps and low-speed signaling rate of 1.5 Mbps. In addition, the USB controller supports three transfer types, which are bulk transfer, interrupt transfer, and isochronous transfer.

- Compliant with USB2.0 and USB 1.1 high-speed, full-speed and low–speed specifications
- 8 general configurable endpoints (EP1..EP8)
- 10 configurable FIFOs (FIFO0..FIFO7; FIFO14..FIFO15)
- Supports transfer types (bulk, interrupt and isochronous)
- Includes digital clock and data recovery function6
- Supports bit stuffing / bit un-stuffing, NRZI encoding/decoding and CRC generation/check)
- Supports chirp sequences
- Supports suspend mode, host resume, and device remote wake-up
- Integrated PHY to bridge between the digital circuit and analog electrical layer

14.1 Overview

Figure 14-1 USB Device Controller

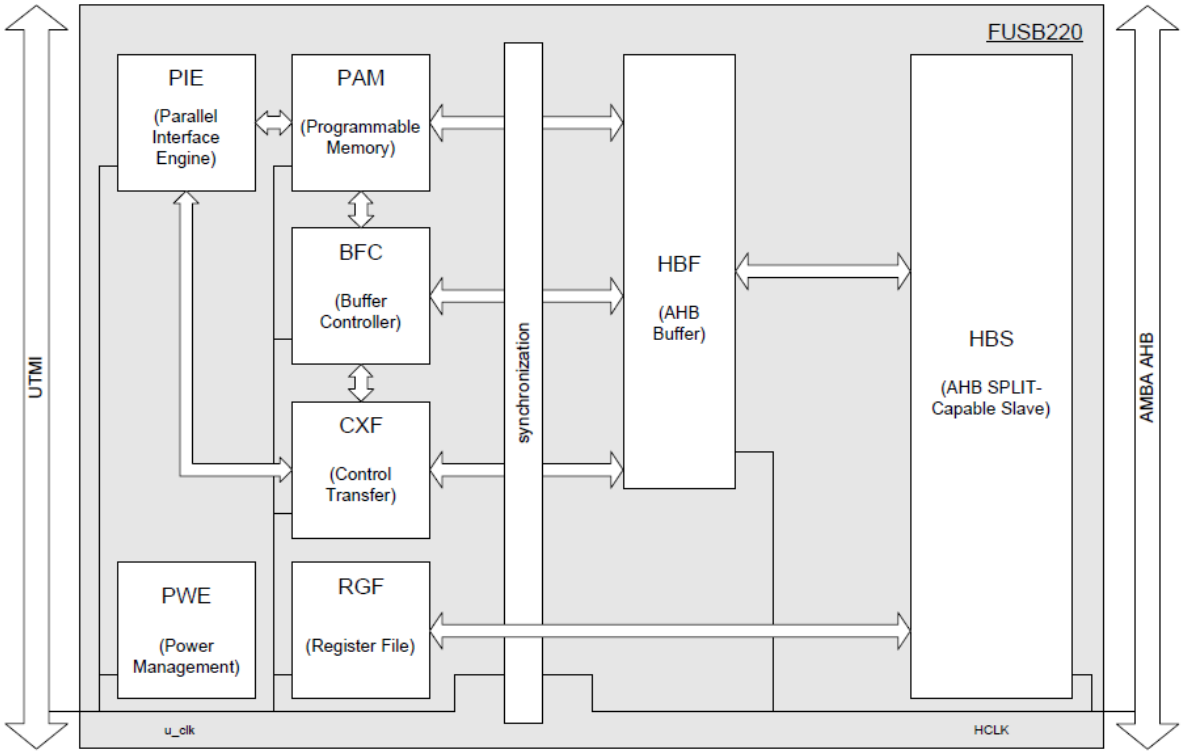


Figure 14-1 shows the eight major sub-blocks of the ANTAIOS USB Device Controller as a functional block diagram.

- Parallel Interface Engine (PIE)
- Control Transfer FIFO (CXF)
- Programmable Memory (PAM)
- Register Files (RGF)
- Power Management and Speed Emulation (PWE)
- AMBA AHB SPLIT-capable slave (HBS)
- AHB Buffer (HBF)
- AHB Buffer Controller (BFC)

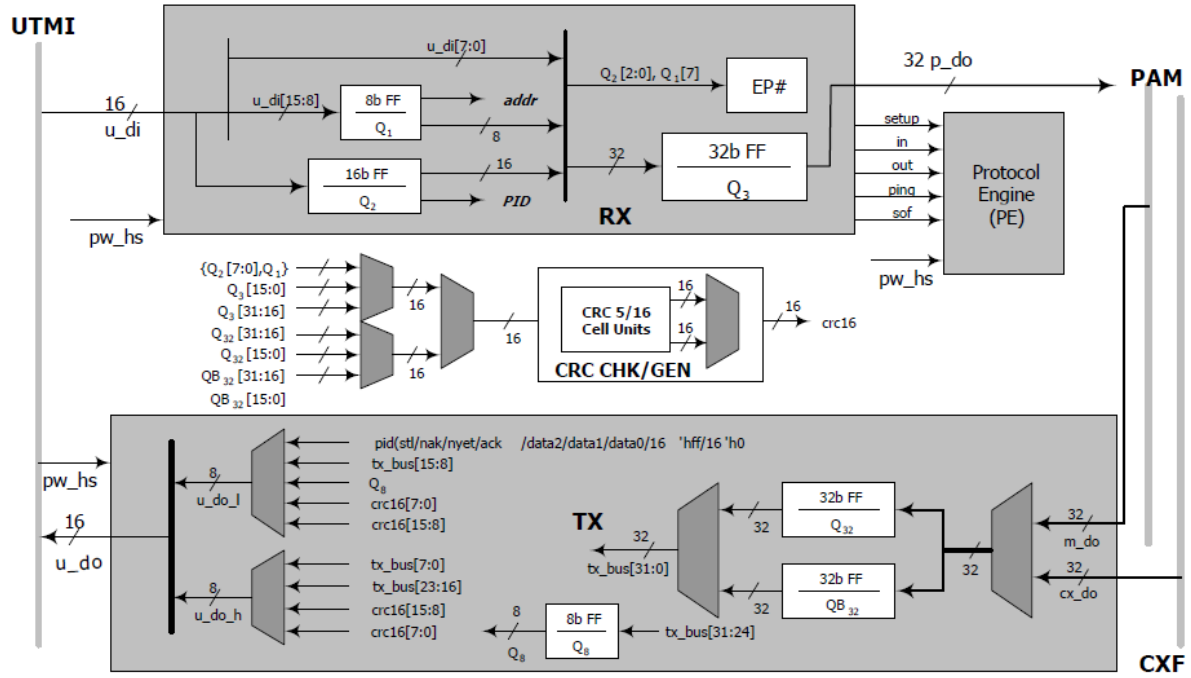
14.1.1 Parallel Interface Engine (PIE)

The block diagram of the PIE is illustrated in Figure 14-2. The PIE operates in the 16-bit parallel data streams with a Universal Transceiver Macro Interface (UTMI). The PIE runs on a 30-MHz clock provided by the transceiver. It decodes the token and the data packets issued by the USB host, and then generate the internal signals to other sub-modules based on the decoded packets. The PIE also handles the CRC5/CRC16 generation and checks for the packets transferred from or to the USB host.

After the chirp mode, the PIE operates in either the high-speed or full-speed mode informed by PWE. The communication between the PIE and CXF regards information about the control transfers. The information for the bulk, isochronous, and interrupt transfers are

passed on through the communication channel between PIE and PAM. The messages passed between the PIE and RGF control the PIE operations. For example, start-of-frame is sent to RGF by PIE for later processing by AP. Please note that only the interconnections among PIE, PAM, and CXF are shown in Figure 14-2. The communication paths among PIE, PWE, and RGF are not addressed.

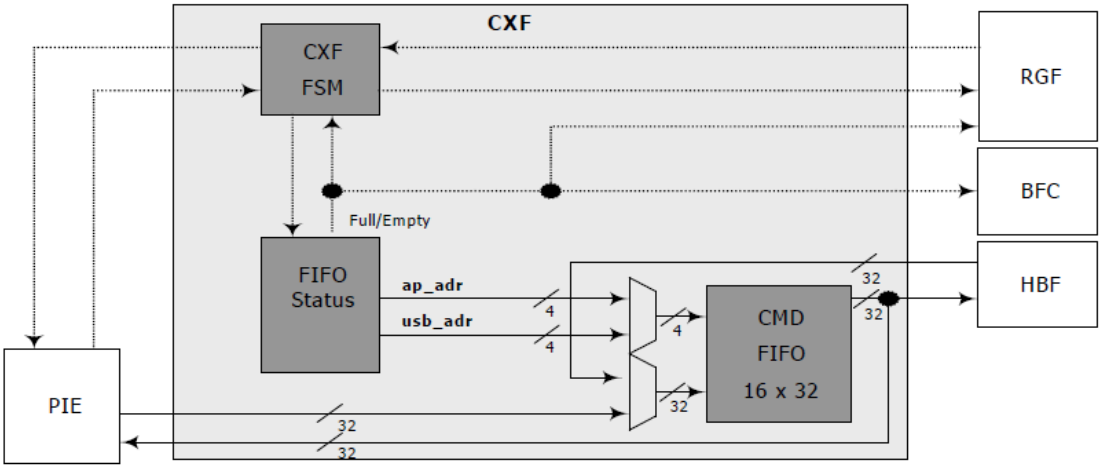
Figure 14-2 PIE Block Diagram



14.1.2 Control Transfer FIFO (CXF)

The block diagram of the Control Transfer FIFO (CXF) is shown in Figure 14-3. CXF contains three control-logic sub-blocks and one memory block. The control signals are shown in the dotted lines. The solid lines represent the data and address signals. The CXF sub-module uses a 32-bit data interface to talk with the Protocol Interface Engine (PIE) for transmitting the control-transfer data from/to the USB host. CXF also communicates with the AMBA interface to send/receive the control-transfer data to/from microprocessor. The channel between the CXF and RGF is used to pass information for the control transfer interrupts and status control of the control-transfer FIFO.

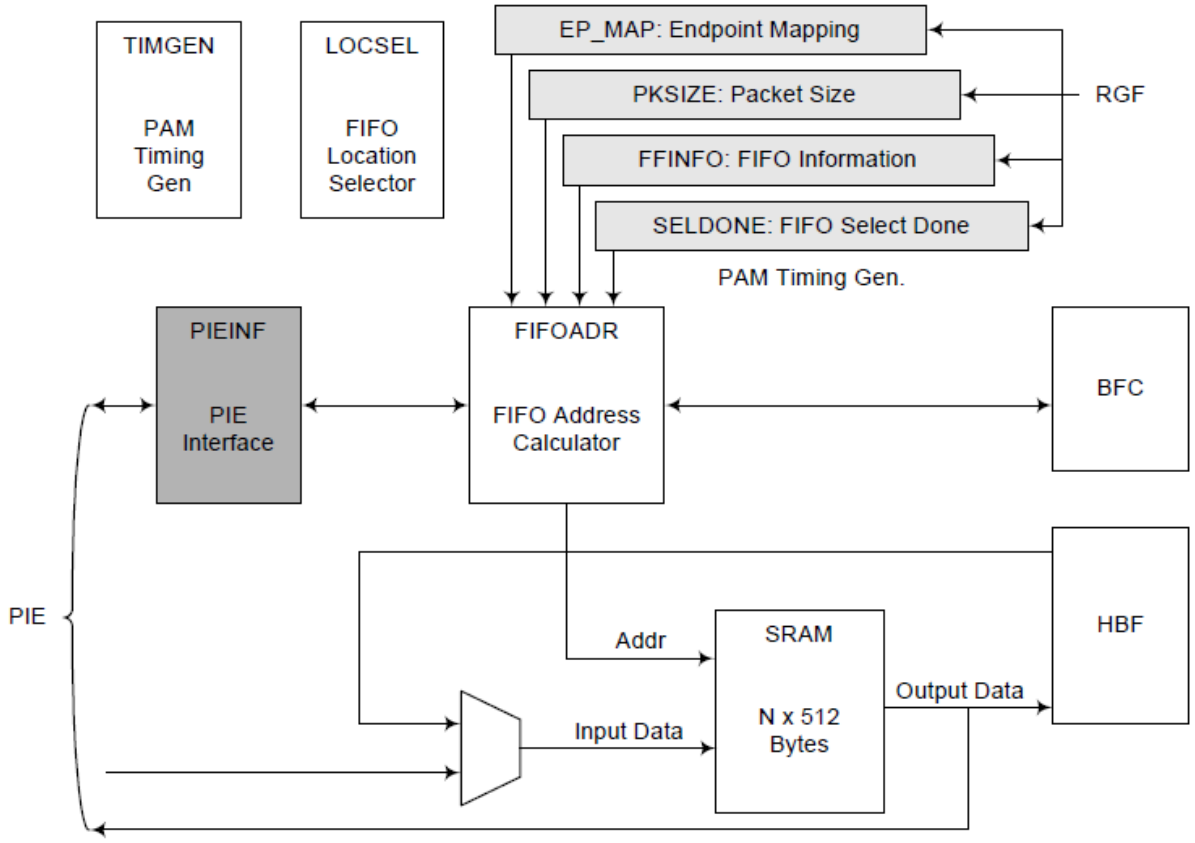
Figure 14-3 CXF Block Diagram



14.1.3 Programmable Memory (PAM)

The block diagram for PAM is shown in Figure 14-4. PAM consists of three major blocks. The first is an interface block, which includes the interfaces to the PIE and BFC modules. Both interfaces run on a 30-MHz clock and a 32-bit data bus. The second block is a timing generation block. These modules receive the control signals and data from PIE and BFC and then route the data to the corresponding FIFO based on the direction of programming. The built-in memory is a synchronous single-port SRAM macro. It is approximately the size of the SRAM, depending on the configuration of the design FIFO Controller.

Figure 14-4 PAM Block Diagram

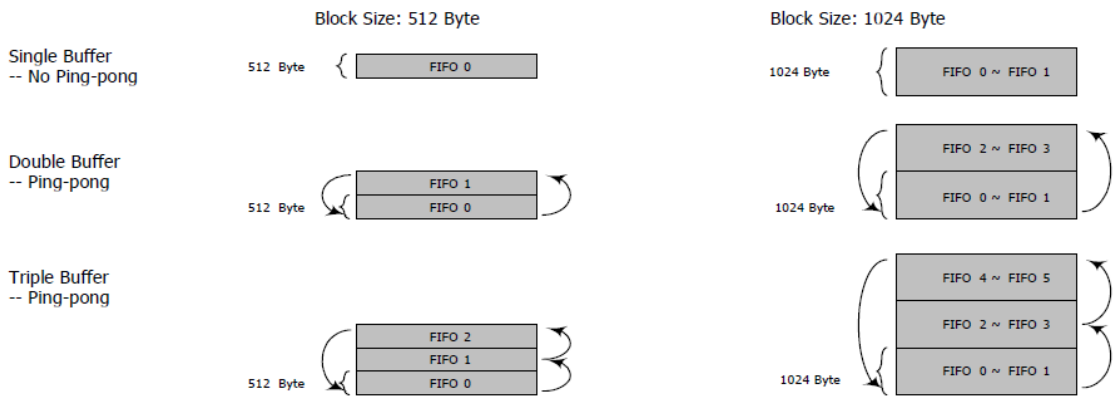


PAM consists of 10 FIFOs that are numbered from 0 to 7 and 14 to 15. FIFO0 to FIFO7 are 512-byte, while FIFO14 and FIFO15 are 64-byte.

A “block” can be composed of one or two FIFOs depending on the programming. The ping-pong FIFO mechanism is block-based.

For example, if FIFO0 and FIFO1 are programmed for the OUT endpoint1 and the block size is set to 512-byte, the initial value of the location counter will be ‘0’. When the host issues the first OUT transaction to endpoint1, FIFO0 (0 + 0 = 0) will be accessed, and the location counter will add to ‘1’. The next OUT transaction to endpoint1 will be automatically redirected to FIFO1 (0 + 1 = 1), and then the location counter is increased by 1 at the end of operation. Since the FIFO block number is programmed as 2, the location counter value is limited to between 0 and 1, and the next value of the location counter will be ‘0’ again.

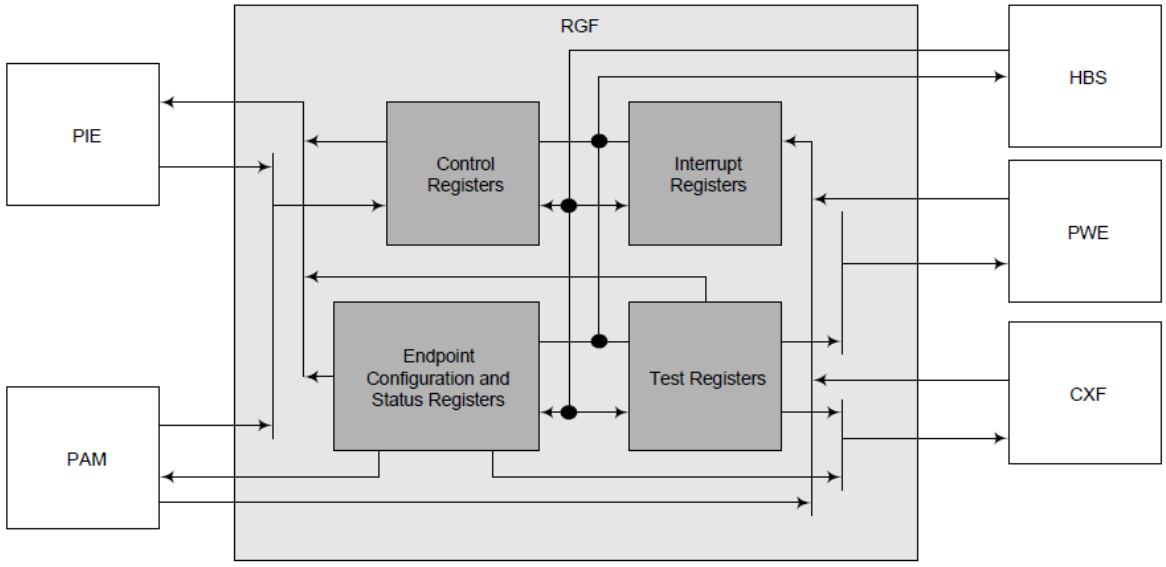
Figure 14-5 Ping-Pong FIFO mechanism with different block size and block number



14.1.4 Register File (RGF)

The block diagram of the RGF is shown in Figure 14-6. RGF contains the registers for all the system configurations and status controls between CPU and the internal blocks. RGF records the auto-configuration setting of each FIFO, the status of each IN/OUT endpoint, interrupt status, test setting, and so on. It also communicates with PIE for information about USB transfers. The interconnection between RGF and PAM provides PAM with configurations for the non-control transfer endpoints. PAM passes the information for each transfer from the non-control endpoints to RGF and vice versa to enable AP to read that information for further processing. The information for the control transfers is transmitted between CXF and RGF. RGF talks with PWE for the detection and support of the speed emulation, power-down, USB reset, and timeout. Finally, RGF adopts the AMBA protocol to enable AP to access the information recorded in the registers of USB Device Controller. The registers are not accessible when u_clk is stopped, because USB Device Controller cannot respond to the AHB master.

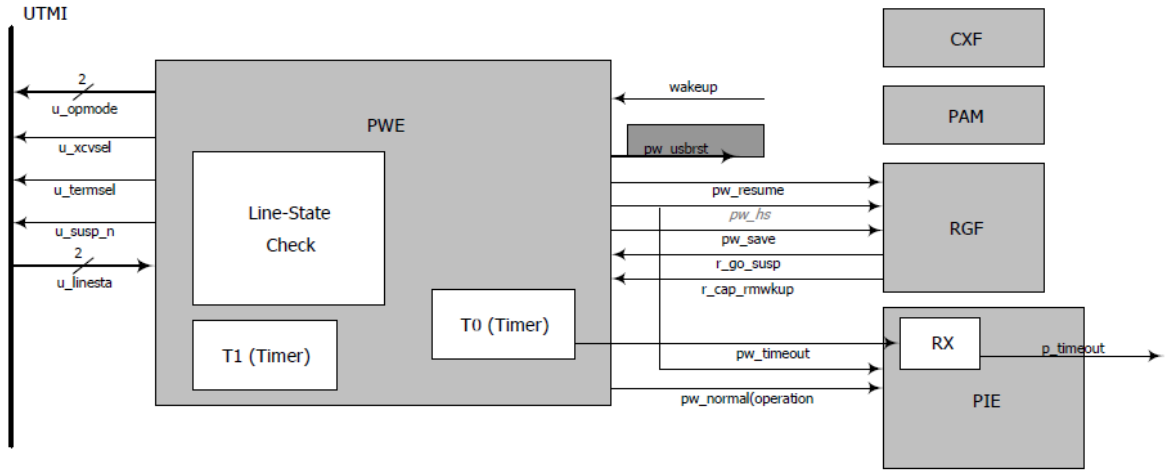
Figure 14-6 RGF Block Diagram



14.1.5 Power Management and Speed Emulation (PWE)

The block diagram of PWE is shown in Figure 14-7. PWE controls the switching of the USB Device Controller speed-operating modes, high-speed mode and full-speed mode. For the power management, PWE monitors u_linesta to detect the idle state of the transaction on USB. If there is no traffic for over 3 ms, PWE will assert a power-save signal to let AP know the idle event. PWE will assert the suspend signal to enable the transceiver to enter a suspend mode if the AP allows PWE to enter the suspend mode. PWE also monitors u_linesta to achieve the detection of the reset and resume of USB.

Figure 14-7 PWE Block Diagram



14.1.6 AHB SPLIT-capable Slave (HBS)

HSB of USB Device Controller is an AMBA 2.0 compliant slave that supports the multiple split transfers. It is the interface between the AHB buffer (HBF) or RGF and the external AHB masters. Before a USB packet is completely transferred to the destination, any AHB masters attempting to access another FIFO of endpoint will be split. The number of the split AHB masters will be recorded. After a USB packet is transferred, the HBS recalls all the split masters to re-attempt the transfer by asserting the proper HSPLIT[15:0] bit to the arbiter and so on.

14.1.7 AHB Buffer (HBF)

Data written to or read from CXF and PAM will be first stored in HBF. In the case of an AP write, data in HBF will be sent to CXF or PAM FIFOs only if the HBF is full or contains the last data payload of a USB packet. In the case of an AP read, HBS will not accept any cycle by the SPLIT response until the HBF is full or contains the last data payload of a USB packet.
In the case of an AP write, the FIFO done bit (0x0B bit0 for CXF, 0xA0 - 0xAF bit3 for PAM) should be set properly after sending a packet under the following circumstances:

- A short packet is sent.
- The packet length is not a multiple of the HBF size. The done bit is used to flush the HBF data and finish a packet.
- AP intends to read FIFO after sending the packet.
- The next packet is sent by another AHB master.

After setting the FIFO-done bit, the firmware should make sure that HBF is emptied by 0x28 bit0 before sending the next packet to the USB Device Controller.
Data written to or read from RGF will not be buffered. These are transferred directly between HBS and RGF.

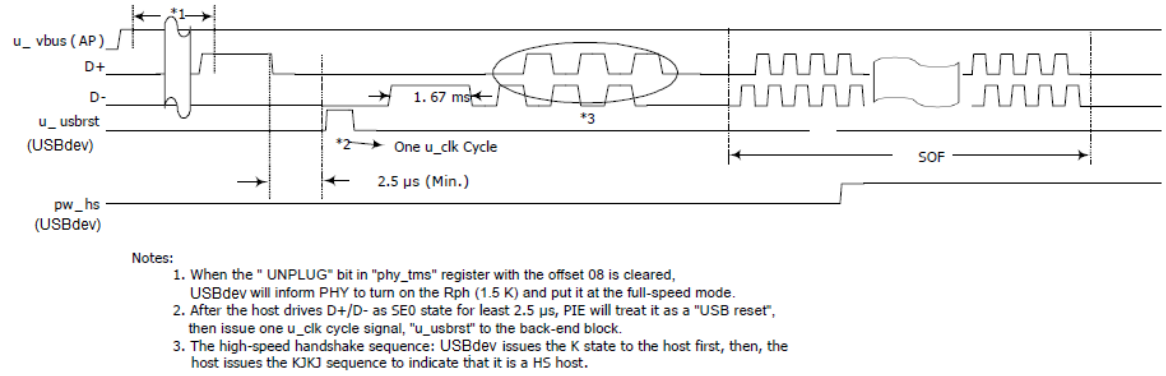
14.1.8 AHB Buffer Controller (BFC)

BFC assumes the responsibility for the data flows between HBF and PAM or CXF. In an AP write case, BFC moves HBF data to the target FIFO when HBF is full or when the FIFO-done bit is set. In the case of an AP read, BFC moves the data from FIFO to HBF when HBF is empty.

14.2 USB Rest and Power-saving Mode

14.2.1 USB Reset

Figure 14-8 Timing Diagram of USB Reset and High-speed Detection Handshake

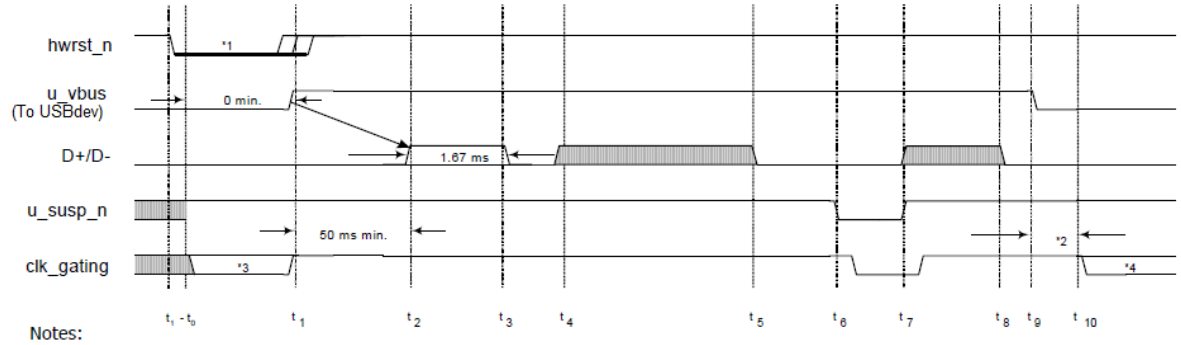


Notes:
1. When the " UNPLUG" bit in "phy_tms" register with the offset 08 is cleared, USBdev will inform PHY to turn on the Rph (1.5 K) and put it at the full-speed mode.
2. After the host drives D+/D- as SE0 state for least 2.5 µs, PIE will treat it as a "USB reset", then issue one u_clk cycle signal, "u_usbrst" to the back-end block.
3. The high-speed handshake sequence: USBdev issues the K state to the host first, then, the host issues the KJKJ sequence to indicate that it is a HS host.

When a device (AP) plugs into the host/hub (detect the plug-in by any gpio-pin), the device must assert Bit0, Byte 0x5C of Pin Controller (Table 4-1) to enable u_vbus. These enables internal u_clk, inform USB Device Controller that the device has attached to the downstream port of the hub, including the root hub on the host. When the “UNPLUG” bit in “phy_tms” (register offset 08, Table 14-7) is cleared, USB Device Controller will inform PHY to turn on Rpu (1.5K), then place the PHY in the full-speed mode, so the J state will be asserted on USB. USB Device Controller also spends at least **15 ms** to cover the debounce interval. After that, any time the device observes no bus activity, it must obey the rules for going into a suspend mode. If the SE0 state on USB has been detected for at least 2.5 µs at this moment, USB Device Controller will assert one clk cycle internal signal, “u_usbrst”, to the back-end blocks. For a while, USB Device Controller will assert Chirp K of 1.67 ms on the USB, and then the host will detect this event and respond to the device with the KJKJKJ sequence. However, the USB Device Controller will not assert the indication for operating in the HS mode, “pw_hs” (state is shown on bit6 HS_EN, Table 14-2), until the first SOF packet has been asserted on the USB. AP will be concern about the result of speed negotiation when AP works in ISO mode. AP could check the speed mode when it receives the first SOF from host. u_clk will be ready after 3 ms of u_vbus assertion. Therefore, the USB Device Controller setting and the UNPLUG bit clearing will be fine after detecting the assertion of u_vbus after approximately 3 ms.

14.2.2 Power-saving Mode

Figure 14-9 USB Device Controller Asserts internal u_susp_n to Turn Off the PLL in PHY



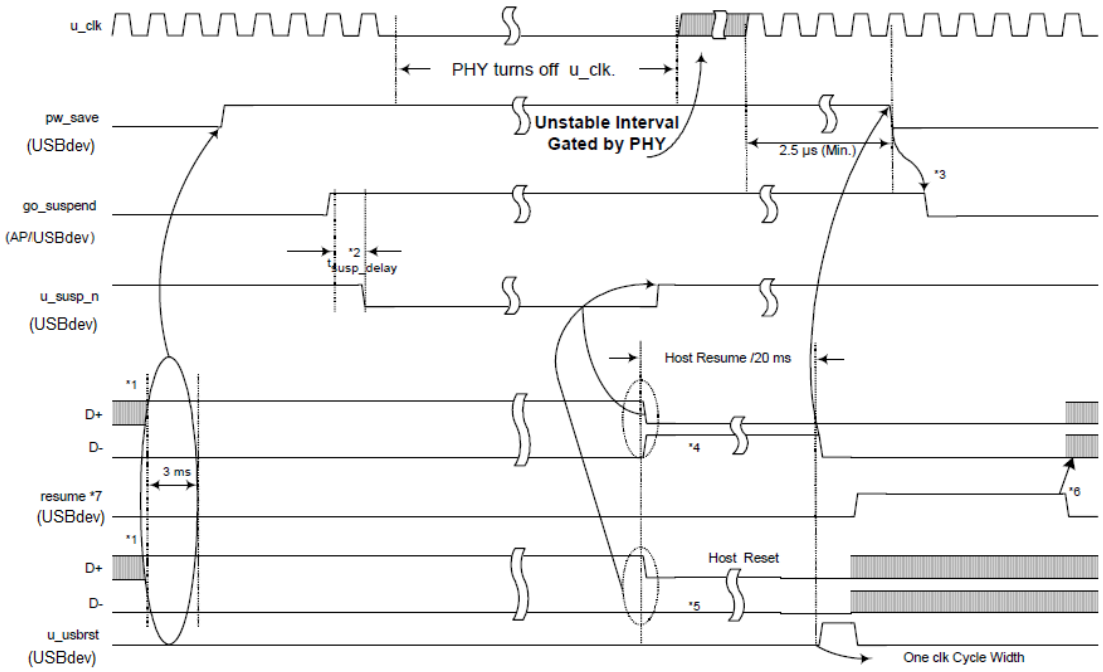
Notes:
1. This interval depends on the customer design.
2. The AP guarantees a minimum of 3 ms for the recovery time of the back-voltage for the assertion of clk_gating since t_9 to t_{10}.
3. The integrator can use come glue logic to combine both u_susp and the clock-masked control provided by AP to turn on or turn off the clock from PHY, such as the "clk_gating " timging.

USB Device Controller provides the internal u_susp_n signal to inform the PHY to turn on/off the internal clock, u_clk, from PHY.

Figure 14-9 shows the timing relationship of u_susp_n between u_vbus and D+/D-. Here is an explanation of what happens at each t_n:

- **t_{-1}** System reset, such as power on or hardware reset
- **t_0** USB Device Controller drives u_susp_n to high state since now.
- **t_1** The u_vbus assertion indicates that the device has been plugged into an upstream facing port, such as a root hub.
- **t_2** USB Device Controller is initially attached as a full-speed device, and the J state asserted on USB by Rpu (1.5K).
- **t_3** The host drives reset on USB.
- **t_4** USB Device Controller operates in the high-speed mode. Communication occurs on USB between the host and USB Device Controller.
- **t_5** There is no traffic since now.
- **t_6** USB Device Controller detects no traffic on USB for more than 3 ms, then asserts the u_susp_n to the low state to put PHY into the suspend mode.
- **t_7** The host resumes USB Device Controller, then the u_susp_n is de-asserted.
- **t_8** The device is unpluged from an upstream facing port.
- **t_9** u_vbus is de-asserted because of the detachment.
- **t_{10}** u_susp_n still remains at a high state. However, in order to reduce the battery power consumption of a self-powered device, the integrator should create another signal, clk_gating, to turn off the clock from PHY.

Figure 14-10 USB Device Controller Woken by Host Resume/Reset



Notes:

1. The idle state is SE0 in the HS mode and J state in the FS mode. If USBdev detects the idle for longer than 3 ms, USBdev will asset the "pw_save" to urge AP to enter the "SUSPEND" mode.
2. The value of t_{susp_delay} can be set by AP. Please refer to the idle_cnt register for details.
3. "o_suspend" is the internal signal of USBdev, which is not visible to AP. However, AP can be set by writing a 1/0 to the GOSUSP bit of the main control register. Then, USBdev will clear it if detecting the **RESUME/USB RESET** signals.
4. At the end of resuming asserted by the host, the USBdev will assert "resume".
5. If the host issues a reset in the suspend mode, USBdev will assert "u_usbrst" when seeing the SE0 state for at least 2.5 µs on USB.
6. When F/W sees the resume interrupt in offset 0x28, it must clear this interrupt bit as soon as possible before new packets are issued on USB, then the resume signaling will be de-asserted.
7. The rising edge of the resume signal is the trigger event for the "RESM_INT" bit in the offset 0x28.

USB Device Controller provides a suspend mode to save power. The sequence to enter the suspend mode is as follows:

1. If there is no transition on D+/D- for more than 3 ms, USB Device Controller will assert a suspend interrupt to inform the firmware that it wants to enter the suspend mode.
2. Once the firmware detects a suspend interrupt, it decides whether or not to accept the suspend request from USB Device Controller. If the firmware accepts the suspend request, it must finish the necessary operations (e.g. read some of the USB Device Controller registers before setting the “GOSUSP” bit in the “main_ctl” register with the USB Device Controller offset “00”). These operations are necessary because the USB Device Controller clock will be turned off after GOSUSP is set.
3. Firmware sets the “GOSUSP” bit in the register file of USB Device Controller.
4. Once the USB Device Controller “GOSUSP” is set, USB Device Controller will enter the suspend mode and assert the suspend output u_susp_n to transceiver. The clock input, u_clk, will then be turned off by the transceiver.
5. Finally, if USB Device Controller detects a resume or the USB reset signal, it will clear the “GOSUSP” bit.

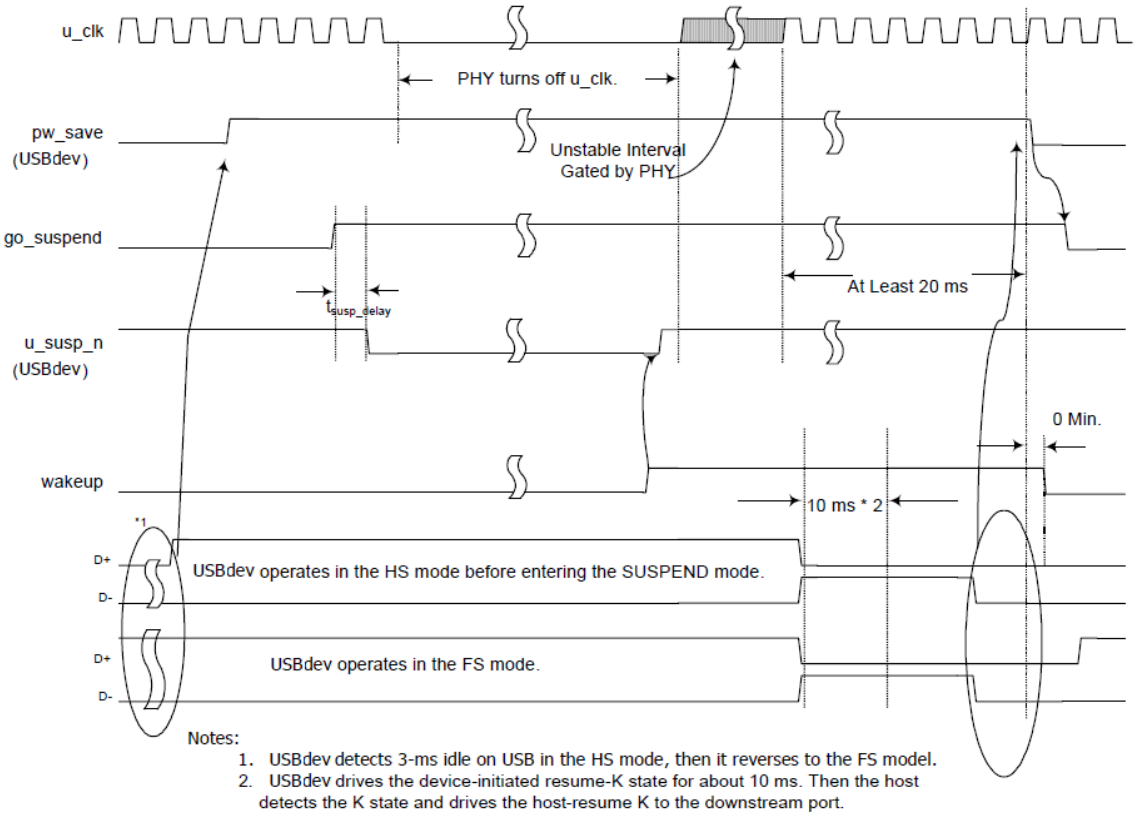
There are three events in which USB Device Controller can leave the suspend mode (Wake up):

1. A host resume, as shown in Figure 14-10. This occurs when the D+/D- toggles.
2. A host reset, as shown in Figure 14-10. This occurs when USB Device Controller detects the SE0 state on USB.
3. Remote wake-up, as shown in Figure 14-11. This takes place when the AP asserts a wake-up input. For the remote wake-ups, the AP should keep the wake-up asserted until USB Device Controller de-asserts pw_save.

In other words, the AP cannot de-assert a wake-up before USB Device Controller de-asserts pw_save. The wake-up procedure is:

1. The host sets the Device_Remote_Wakeup feature to wake-up USB Device Controller.
2. Under the suspend mode, either the host resumes or a remote wake-ups from the AP triggers the wake-up process and the USB Device Controller will be woken up first.
3. Once the USB Device Controller is woken up, it will de-assert the suspend output to PHY. PHY will then turn on the 30-MHz clock output after USB Device Controller de-asserts the suspend output.
4. The clock output of PHY is unstable for a short period of time after the suspended transceiver is woken. The unstable interval is approximately 2 ms for the transceiver.

Figure 14-11 USB Device Controller Woken by AP



14.3 Memory Map and Register Definition

Table 14-1 lists and describes the control registers of ANTAIOS USB Device Controller.

Table 14-1 Summary of USB Controller Registers

Address Offset	Type	Size (Byte)	Description	Default Value
0x00	R,RW	1	Main_ctl	0x00
0x01	RW	1	Dev_addr	0x00
0x02	RW	1	Tst_ep	0x00
0x03	-	-	Reserved	-
0x04	R	2	Frm_numb	0x00
0x06	RW	2	Sof_tmskb	0x00
0x08	RW	1	Phy_tms	0x01
0x09	RW	1	Vnd_ctl	0x00
0x0A	R	1	Vnd_sta	0x00
0x0B	RW	1	Cx_csr	0x20
0x0C	RW	-	Ep0_db	-
0x0D-0x0F	-	-	Reserved	-
0x10	RW	1	Int_mgrp	0x00
0x11	RW	1	Int_mskb0	0x20
0x12	RW	1	Int_mskb1	0xFF
0x13	RW	1	Int_mskb2	0xFF
0x14	-	-	Reserved	-
0x15	RW	1	Int_mskb4	0xFF
0x16	RW	1	Int_mskb5	0xFF
0x17	RW	1	Int_mskb5	0xFF
0x18	RW	1	Int_mskb7	0x00
0x19	RW	1	Rx0byte_epb0	0x00
0x1A	RW	1	Rx0byte_epb1	0x00
0x1B	-	-	Reserved	-
0x1C	R	1	Fempt_b0	0xFF
0x1D	R	1	Fempt_b1	0xFF
0x1E	RW	1	Tst_inf_ini	0x00
0x1F	RW	1	Tst_inf_cont	0xFF
0x20	R	1	Int_grp	0x80
0x21	R,RW	1	Int_srcb0	0x00
0x22	R	1	Int_srcb1	0x00
0x23	R	1	Int_srcb2	0x00
0x24	-	-	Reserved	-
0x25	R	1	Int_srcb4	0x00
0x26	R	1	Int_srcb5	0x00
0x27	R	1	Int_srcb6	0x00
0x28	R,RW	1	Int_srcb7	0x01
0x29	RW	1	Iso_seq_errb0	0x00
0x2A	RW	1	Iso_seq_errb1	0x00
0x2B	RW	1	Iso_seq_abtb0	0x00

Address Offset	Type	Size (Byte)	Description	Default Value
0x2C	RW	1	Iso_seq_abtb1	0x00
0x2D	RW	1	Tx0byteb0	0x00
0x2E	RW	1	Tx0byteb1	0x00
0x2F	RW	1	Idle_cnt	0x00
0x30	RW	1	Ep1_map	0xFF
0x31	RW	1	Ep2_map	0xFF
0x32	RW	1	Ep3_map	0xFF
0x33	RW	1	Ep4_map	0xFF
0x34	RW	1	Ep5_map	0xFF
0x35	RW	1	Ep6_map	0xFF
0x36	RW	1	Ep7_map	0xFF
0x37	RW	1	Ep8_map	0xFF
0x38-0x3E	-	-	Reserved	-
0x3F	R	1	Hbf_cnt	0x00
0x40	RW	2	Iep1_xpsz	0x0200
0x42	RW	2	Iep2_xpsz	0x0200
0x44	RW	2	Iep3_xpsz	0x0200
0x46	RW	2	Iep4_xpsz	0x0200
0x48	RW	2	Iep5_xpsz	0x0200
0x4A	RW	2	Iep6_xpsz	0x0200
0x4C	RW	2	Iep7_xpsz	0x0200
0x4E	RW	2	Iep8_xpsz	0x0200
0x50-0x5F	-	-	Reserved	-
0x60	RW	2	Oep1_xpsz	0x0200
0x62	RW	2	Oep2_xpsz	0x0200
0x64	RW	2	Oep3_xpsz	0x0200
0x66	RW	2	Oep4_xpsz	0x0200
0x68	RW	2	Oep5_xpsz	0x0200
0x6A	RW	2	Oep6_xpsz	0x0200
0x6C	RW	2	Oep7_xpsz	0x0200
0x6F	RW	2	Oep8_xpsz	0x0200
0x70-0x7C	-	-	Reserved	-
0x7E	RW	2	Fifo_dma_en	0x0000
0x80	RW	1	Fifo0_map	0x0F
0x81	RW	1	Fifo1_map	0x0F
0x82	RW	1	Fifo2_map	0x0F
0x83	RW	1	Fifo3_map	0x0F
0x84	RW	1	Fifo4_map	0x0F
0x85	RW	1	Fifo5_map	0x0F
0x86	RW	1	Fifo6_map	0x0F
0x87	RW	1	Fifo7_map	0x0F
0x88-0x8D	-	-	Reserved	-
0x8E	RW	1	Fifo14_map	0x0F
0x8F	RW	1	Fifo15_map	0x0F

Address Offset	Type	Size (Byte)	Description	Default Value
0x90	RW	1	Fifo0_config	0x00
0x91	RW	1	Fifo1_config	0x00
0x92	RW	1	Fifo2_config	0x00
0x93	RW	1	Fifo3_config	0x00
0x94	RW	1	Fifo4_config	0x00
0x95	RW	1	Fifo5_config	0x00
0x96	RW	1	Fifo6_config	0x00
0x97	RW	1	Fifo7_config	0x00
0x98-0x9D	-	-	Reserved	-
0x9E	RW	1	Fifo14_config	0x00
0x9F	RW	1	Fifo15_config	0x00
0xA0	RW	1	Fifo0_inst	0x00
0xA1	RW	1	Fifo1_inst	0x00
0xA2	RW	1	Fifo2_inst	0x00
0xA3	RW	1	Fifo3_inst	0x00
0xA4	RW	1	Fifo4_inst	0x00
0xA5	RW	1	Fifo5_inst	0x00
0xA6	RW	1	Fifo6_inst	0x00
0xA7	RW	1	Fifo7_inst	0x00
0xA8-0xAD	-	-	Reserved	-
0xAE	RW	1	Fifo14_inst	0x00
0xAF	RW	1	Fifo15_inst	0x00
0xB0	R	1	Fifo0_bc	0x01
0xB1	R	1	Fifo1_bc	0x01
0xB2	R	1	Fifo2_bc	0x01
0xB3	R	1	Fifo3_bc	0x01
0xB4	R	1	Fifo4_bc	0x01
0xB5	R	1	Fifo5_bc	0x01
0xB6	R	1	Fifo6_bc	0x01
0xB7	R	1	Fifo7_bc	0x01
0xB8-0xBD	-	-	Reserved	-
0xBE	R	1	Fifo14_bc	0x01
0xBF	R	1	Fifo15_bc	0x01
0xC0	RW	-	Fifo0_dp	-
0xC4	RW	-	Fifo1_dp	-
0xC8	RW	-	Fifo2_dp	-
0xCC	RW	-	Fifo3_dp	-
0xD0	RW	-	Fifo4_dp	-
0xD4	RW	-	Fifo5_dp	-
0xD8	RW	-	Fifo6_dp	-
0xDC	RW	-	Fifo7_dp	-
0xE0-0xF4	-	-	Reserved	-
0xF8	RW	-	Fifo14_dp	-
0xFC	RW	-	Fifo15_dp	-

14.3.1 Main Control Register

Offset: 0x00

This register records the device-oriented characteristics. Table 14-2 shows the bit assignment of this control register.

Table 14-2 Main Control Register

Bit	Name	Type	Reset	Bus Reset	Description
7	AHB_RST	RW	0	-	AHB software reset Write a '1' to reset HBS, HBF and BFC. This bit is automatically cleared by the hardware reset
6	HS_EN	R	0	-	High-speed enable 1: Device is in the high-speed mode. 0: Device is in the full-speed mode.
5	CHIP_EN	RW	0	-	Chip enable Write a '1' to enable the write cycle of PAM FIFO
4	SFRST	RW	0	-	Software reset Write a '1' to set a software-initiated reset to USB-Controller. This bit cannot be set when USB-Controller is in the suspend mode, since the u_clk is stopped. Setting this bit will cause the de-assertion of the pw_save output if it is asserted. • The chirp sequence will be terminated while this bit is set. • The command FIFO will also be cleared by setting this bit. • The Frame Number Register and SOF Timer Mask Register will be cleared too. • The micro-frame number in the RGF will be cleared. Note: The data FIFO status will not be cleared by this reset.
3	GOSUSP	RW	0	0	Go suspend [1)] Writing a '1' will activate the suspend mode.
2	GLINT_EN	RW	0	-	Global interrupt enable A '1' enables all interrupts. Individual interrupts can be masked by setting the corresponding bits in the interrupt mask register (Index 0x11-0x17).
1	FLUSH_HBF	RW	0	-	Flush HBF Write a '1' to flush HBF data to FIFO. This bit is automatically cleared by hardware.
0	CAP_RMWKUP	RW	0	-	Capability of remote wake-up A '1' indicates that USB-Controller can be woken by a "wakeup" signal.

[1)] If USB port is not used, set this bit to enable power saving.

14.3.2 Device Address Register

Offset: 0x01

The device address register stores the assigned address of a USB device and enables the USB device after executing the set-configuration command.

Table 14-3 Device Address Register

Bit	Name	Type	Reset	Bus Reset	Description
7	AFT_CONF	RW	0	0	After the set-configuration A '1' indicates that the device successfully executes the SET_CONFIGURATION command. Note: Before this bit is set, USB-Controller will not respond to (i.e. timeout) any non-control transfer sent by the host. It is the responsibility of AP to set this bit when the AP receives a SET_CONFIGURATION command. Otherwise, USB-Controller will timeout the non-control transfers following this command.
6..0	DEVADDR	RW	0x00	0x00	Device address This signal records the latest USB device address for each SET_ADDRESS.

14.3.3 Test Register

Offset: 0x02

The CXF loopback test procedure is described as follows:

1. Clear "TST_CLREA" and set "TST_LPCX" at the same time
2. Write data to control FIFO via C_BUS
3. Set the CX_DONE bit of CX configuration and status register (Address = 0x0B, bit 0)
4. Poll the CX_DONE bit until its value is 1.
5. Set "TST_CLREA" and clear "TST_LPCX" at the same time
6. Clear the CX_DONE bit
7. Read the data from control FIFO via C_BUS
8. Compare data

Table 14-4 Test Register

Bit	Name	Type	Reset	Bus Reset	Description
7	TST_HALF_SPEED	RW	0	-	Set the half-speed A '1' turns on the half-speed mode When this bit is set to '1', USB-Controller allows AP to access FIFO only at each even cycle of u_clk.
6	TST_MOD	RW	0	-	Test mode A '1' turns on the test_mode. When this bit is set to '1', USB-Controller will enter the test mode. In the normal mode, USB-Controller uses a counter for 10 ms detection of the USB reset. The count is a large number. In the test mode, USB-Controller will use a smaller count for the USB reset detection to save the test cycles on test machine.
5	TST_DISTO	RW	0	-	Disable the toggle sequence A '1' disables the toggle sequence.

Bit	Name	Type	Reset	Bus Reset	Description
4	TST_DISCRC	RW	0	-	Disable CRC When setting this bit as '1', USB-Controller will not append CRC for the upstream packets.
3	TST_DISGEMSOF	RW	0	-	Disable the self-generation of SOF It uses SOF sent from the host instead of generating a SOF by USB-Controller. (SOF pin output depends on the SOF generated by USB-Controller.)
2	TST_CLREA	RW	0	-	Clear the external side address Write '1' then a '0' to clear external side address for the loopback test.
1	TST_LPCX	RW	0	-	Loopback test for CX A '1' indicates the loop-back test is active for the control transfer (Endpoint 0).
0	TST_CLRFF	RW	0	-	Clear FIFO Write '1' to clear all the PAM FIFO counters and location counters.

14.3.4 SOF Frame Number Register0

Offset: 0x04

The frame number records the last successfully received Start-of-Frame (SOF) number. The register contains two bytes. The bit allocation is given in the following tables

Table 14-5 SOF Frame Number Register

Bit	Name	Type	Reset	Bus Reset	Description
15..14	Reserved	-	-	-	-
13..11	USOFN	R	0x0	0x0	SOF micro-frame number bits[2:0] Record the micro-frame number in the high speed mode
10..0	SOFN	R	0x0	0x0	SOF frame number bits [10:0] Record the frame number for the high speed and full speed

14.3.5 SOF Mask Timer Register

Offset: 0x06

This 2-byte register is used to mask the last SOF.
The main function of the SOF mask register is to avoid the SOF misalignment between SOFs from host and the self-generated SOF, but the SOF mask register is also used to decide the SOF period. It indicates the time period during which USB Device Controller cannot accept a new SOF after the present SOF. Users can set the appropriate period to ensure that AP's ISO mode transfer will not be interrupted/corrupted due to an unexpected SOF. It is recommended that the SOF mask register be set to as long as 80% of the micro-frame time in HS and 90% of the frame time in the FS mode.

USB-Controller will self-generate a SOF to make up the missing SOF from the host. The period will be decided by two sources, SOF from the host and the time-limit value, which is a fixed value as a (micro) frame time. In normal conditions, the self-generated SOF will be aligned with SOF from host. However, if USB-Controller does not wait for SOF (From the host) over the maximum time limit, then SOF may be corrupted and misaligned. USB-Controller will have to generate a self-generated SOF to recover. In certain cases, the host might send a SOF right after the USB-Controller self-generated SOF and USB-Controller will be confused by the two unaligned SOFs for the same frame period. The mask register is used to avoid this situation.

Table 14-6 Mask Timer Register

Bit	Name	Type	Reset	Soft Reset	Description
15..0	SOFTM	0x0	0x0	0x0	SOF mask timer Time since the last SOF in the 30 MHz clock bit.

14.3.6 Phy Test Mode Selector Register

Offset: 0x08

This one-byte register allows the firmware to set the D+/D- lines to predetermined states for the testing purpose. The bit allocation is given in the following table. Please note that only one bit can be set at a time.

Table 14-7 Phy Test Mode Selector Register

Bit	Name	Type	Reset	Bus Reset	Description
7	Reserved	-	-	-	-
6	TST_FAIL	RW	0	-	Test interface fail This bit will be asserted to inform users that there is a crc16_err or pid_err error during receiving data with “TST_INF” being active. This bit must be cleared manually by users.
5	TST_INF	RW	0	-	Test interface between PHY and the controller Writing a ‘1’ to this bit, USB-Controller will start automatically to send the random patterns to PHY in the high-speed mode. At the same time, USB-Controller also receives the data bypassing PHY. USB-Controller can verify if there is any timing violation at the interface between PHY and the controller by checking whether errors occurred or not. This bit will be cleared automatically when EOP is received. Note: Before writing a ‘1’ to this bit, the “Vendctrol” pins of PHY must be set to 3.
4	TST_PKT	RW	0	-	Test mode for packet Writing a ‘1’ to this bit, USB-Controller will repeatedly send the packet defined in the UTMI specification to the transceiver. After the set_feature command shows the test mode and the index Test_Packet is decoded, this bit will be asserted.
3	TST_SE0NAK	RW	0	-	Writing a ‘1’, the D+/D- lines will be set to HS, the quiescent state. The device only responds to a valid HS IN token and always responds to the IN token with NAK.
2	TST_KSTA	RW	0	-	Writing a ‘1’, the D+/D- will be set to the high-speed K state.

Bit	Name	Type	Reset	Bus Reset	Description
1	TST_JSTA	RW	0	-	Writing a '1', the D+/D- will be set to the high-speed J state.
0	UNPLUG	RW	0	-	When the UNPLUG is set to logic '1', the device controller will set PHY to the non-driving mode to emulate the detachment of a device even if it is really plugged. The USB host will not detect a plug of a device. Such event is called "soft-detachment". After a hardware reset, UNPLUG will be in logic '1'. The device is now soft-detached. To enable the USB host in order to detect an attachment of a device, PHY must drive D+ and D- in to the manner defined in the USB specifications. In order to enable PHY to drive D+ and D-, AP should clear the UNPLUG bit after a hardware reset. If AP does not clear the UNPLUG bit, the device will always be soft-detached, and the USB host will never detect the attachment.

14.3.7 Vendor Specific IO Control Register

Offset: 0x09

This register is provided for the vendor defined test control and status for PHY.

Table 14-8 Endpoint *n* Status Registers

Bit	Name	Type	Reset	Bus Reset	Description
7..5	Reserved	-			
4	VCTLAD_N	RW	0	-	Vendor-specific test mode control load This bit controls the active low output, u_vctload_n, to PHY. A '1' in this bit makes u_vctload_n output a '1'. When this bit is cleared, u_vctload_n outputs a '0'.
3..0	VCTL[3:0]	RW	0	-	Vendor-specific test mode control The programmed value is delivered to PHY via the output, "u_vctl".

14.3.8 Vendor-specific I/O Status Registers

Offset: 0x0A

This reset value depends on PHY.

Table 14-9 Vendor-specific I/O Status Registers

Bit	Name	Type	Reset	Bus Reset	Description
7..0	VSTA[7:0]	R			Vendor-specific test mode status

14.3.9 CX Configuration and Status Register

Offset: 0x0B

The CX configuration and the status register are used to control the FIFO management for endpoint 0. The maximum packet size for endpoint 0 is 64 bytes.

Table 14-10 CX Configuration and Status Register

Bit	Name	Type	Reset	Bus Reset	Description
7..6	Reserved	-	-	-	-
5	CX_EMP	R	1	-	CX FIFO is empty A '1' indicates that the endpoint 0 FIFO is empty.
4	CX_FUL	R	0	-	CX FIFO is full. A '1' indicates that the endpoint 0 FIFO is full.
3	CX_CLR	RW	0	-	Clear CX FIFO data Write a '1' to clear the data in endpoint 0 FIFO. Note: For endpoint 0, all the data in FIFO will be cleared no matter if the previous SETUP or IN or OUT transaction has completed or not.
2	CX_STL	RW	0	0	Stall CX Writing a '1' to this bit can stall Endpoint 0 and the endpoint 0 FIFO will be cleared at the same time. Once CX_STL is set, AP cannot access endpoint 0 FIFO until this bit is cleared. The stall status will be cleared by the next setup transaction. This bit will be cleared automatically when the endpoint 0 transaction has ended. Please note that when AP wants to set CX_STL, it should set CX_DONE bit at the same time, within one write operation to CX Configuration and Status Register. In other words, the AP cannot set CX_STL with one write operation and then set CX_DONE with another write operation; it must set both bits within one write operation.
1	TST_PKDONE	RW	0	-	Data transfer is done for test packet Firmware has completed sending the whole test patterns to the endpoint 0 FIFO for a PHY test by writing a '1' to this bit. This bit is cleared by a hardware reset.
0	CX_DONE	RW	0	0	Data transfer is done for CX Firmware has finished the whole packet transaction for endpoint 0 by writing a '1' to this bit. This bit is cleared by a hardware reset. This bit is cleared by the internal signal "p_endcx" or "p_comfail".

14.3.10 Endpoint = Data Port Register

Offset: 0x0C

The address 0x0C provides the direct access for a micro-controller to the FIFO for endpoint 0. In order to access the endpoint 0 FIFO, AP should only use the address 0x0C.
For example, in order to read a 31-byte packet from CXF FIFO, the AHB master should issue the following cycles in the following table.

Table 14-11 Sequence for reading a 31-byte packet from CXF FIFO

Cycle Number	HAddr	HTrans	HSize
1	0x0C	NONSEQ	WORD
2	0x0C	NONSEQ	WORD
3	0x0C	NONSEQ	WORD
4	0x0C	NONSEQ	WORD

Cycle Number	HAddr	HTrans	HSize
5	0x0C	NONSEQ	WORD
6	0x0C	NONSEQ	WORD
7	0x0C	NONSEQ	WORD
8	0x0C	NONSEQ	HALFWORD
9	0x0C	NONSEQ	BYTE

Alternatively, the AHB master may issue eight NONSEQUENTIAL read cycles with a word size and discard the invalid byte.

14.3.11 Interrupt Mask Register

This register masks the individual interrupt sources. The interrupts for each source group and source byte can be individually controlled via the corresponding mask bits. All interrupts can be globally disabled by setting the bit, GLINT_EN, as ‘0’ in the main control register.

14.3.11.1 Interrupt Group Mask Register

Offset: 0x10

To disable an interrupt, the micro-controller should set the corresponding bit as ‘1’.

Table 14-12 Interrupt Group Mask Register

Bit	Name	Type	Reset	Bus Reset	Description
7	MINT_SRC7	RW	0	-	Mask all the interrupt bits of interrupt source register byte 7 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
6	MINT_SRC6	RW	0	-	Mask all the interrupt bits of interrupt source register byte 6 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
5	MINT_SRC5	RW	0	-	Mask all the interrupt bits of interrupt source register byte 5 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
4	MINT_SRC4	RW	0	-	Mask all the interrupt bits of interrupt source register byte 4 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
3	MINT_SRC3	RW	0	-	Mask all the interrupt bits of interrupt source register byte 3 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
2	MINT_SRC2	RW	0	-	Mask all the interrupt bits of interrupt source register byte 2 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
1	MINT_SRC1	RW	0	-	Mask all the interrupt bits of interrupt source register byte 1 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt

Bit	Name	Type	Reset	Bus Reset	Description
0	MINT_SRC0	RW	0	-	Mask all the interrupt bits of interrupt source register byte 0 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt

14.3.11.2 Interrupt Mask Register Byte0

Offset: 0x11

Table 14-13 Interrupt Mask Register Byte0

Bit	Name	Type	Reset	Bus Reset	Description
7	MR_COM_ABORT	RW	0	-	Mask command abort interrupt 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
6	Reserved	-	-	-	-
5	MRBUF_ERR	RW	1	-	Mask the read HBF error interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
4	MCX_COMFAIL_INT	RW	0	-	Mask the interrupt caused by the emitting extra IN or OUT data of the host 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
3	MCX_COMEND_INT	RW	0	-	Mask the host end of command (Entering status stage) interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
2	MCX_OUT_INT	RW	0	-	Mask the interrupt bits of endpoint 0 for OUT 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
1	MCX_IN_INT	RW	0	-	Mask the interrupt bits of endpoint 0 for IN 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
0	MCX_SETUP_INT	RW	0	-	Mask endpoint 0 setup data received interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt

14.3.11.3 Interrupt Mask Register Byte1-4

Offset: 0x12..0x15

Table 14-14 Interrupt Mask Register Byte1-4

Bit	Name	Type	Reset	Bus Reset	Description
31	MF15_SPK_INT	RW	1	-	Mask the Short Packet Interrupt of FIFO15

Bit	Name	Type	Reset	Bus Reset	Description
					0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
30	MF15_OUT_INT	RW	1	-	Mask the OUT interrupt of FIFO15 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
29	MF14_SPK_INT	RW	1	-	Mask the Short Packet Interrupt of FIFO14 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
28	MF14_OUT_INT	RW	1	-	Mask the OUT interrupt of FIFO14 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
27..16	Reserved	-	-	-	-
15	MF7_SPK_INT	RW	1	-	Mask the Short Packet Interrupt of FIFO7 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
14	MF7_OUT_INT	RW	1	-	Mask the OUT interrupt of FIFO7 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
..	..	..	..	..	..
3	MF1_SPK_INT	RW	1	-	Mask the Short Packet Interrupt of FIFO1 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
2	MF1_OUT_INT	RW	1	-	Mask the OUT interrupt of FIFO1 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
1	MF0_SPK_INT	RW	1	-	Mask the Short Packet Interrupt of FIFO0 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
0	MF0_OUT_INT	RW	1	-	Mask the OUT interrupt of FIFO0 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt

14.3.11.4 Interrupt Mask Register Byte5-6

Offset: 0x16..0x17

Table 14-15 Interrupt Mask Register Byte5-6

Bit	Name	Type	Reset	Bus Reset	Description
15	MF15_IN_INT	RW	1	-	Mask the IN interrupt bits of FIFO15 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
14	MF14_IN_INT	RW	1	-	Mask the IN interrupt bits of FIFO15 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
13..8	Reserved	-	-	-	-

Bit	Name	Type	Reset	Bus Reset	Description
7	MF8_IN_INT	RW	1	-	Mask the IN interrupt bits of FIFO7 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
..	..	..	..	..	…
1	MF1_IN_INT	RW	1	-	Mask the IN interrupt bits of FIFO1 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
0	MF0_IN_INT	RW	1	-	Mask the IN interrupt bits of FIFO0 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt

14.3.11.5 Interrupt Mask Register Byte7

Offset: 0x18

Table 14-16 Interrupt Mask Register Byte7

Bit	Name	Type	Reset	Bus Reset	Description
7	MRY0BYTE_INT	RW	0	-	Mask received zero-length data packet interrupt Masks the active to the received zero-length data packet interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
6	MTX0BYTE_INT	RW	0	-	Mask transferred zero-length data packet interrupt Masks the active to transferred zero-length data packet interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
5	MSEQ_ABORT_INT	RW	0	-	Mask ISO sequential abort interrupt Masks the active to the ISO sequential abort interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
4	MSEQ_ERR_INT	RW	0	-	Mask ISO sequential error interrupt Masks the active to received ISO sequential error interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
3	MRESM_INT	RW	0	-	Mask resume interrupt Mask the active to the resume state change interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
2	MSUSP_INT	RW	0	-	Mask suspend interrupt Mask the active to suspend state change interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
1	MUSBRST_INT	RW	0	-	Mask USB reset interrupt Mask the bus reset interrupt bit

Bit	Name	Type	Reset	Bus Reset	Description
					0: Enable the corresponding interrupt 1: Disable the corresponding interrupt
0	MHBF_EMPTY_INT	RW	0	-	Mask HBF empty interrupt Mask the HBF empty interrupt bit 0: Enable the corresponding interrupt 1: Disable the corresponding interrupt

14.3.11.6 Receive Zero-length Data Packet Register Byte0-1

Offset: 0x19..0x1A

When a “Receive Zero-length Data Packet Interrupt” occurs (Bit 7 of register 0x28), the firmware will further check the registers, 0x19 and 0x1A, to determine which endpoint receives a zero-length data packet. This register will then be cleared by firmware.

Table 14-17 Receive Zero-length Data Packet Register Byte0

Bit	Name	Type	Reset	Bus Reset	Description
15..9	Reserved	-	-	-	-
8	RX0BYTE_EP8	RW	0	-	Endpoint8 receives a zero-length data packet
..	..	..	..	..	..
2	RX0BYTE_EP2	RW	0	-	Endpoint2 receives a zero-length data packet
1	RX0BYTE_EP1	RW	0	-	Endpoint1 receives a zero-length data packet
0	Reserved	-	-	-	-

14.3.11.7 FIFO Empty Byte0-1

Offset: 0x1C..0x1D

By polling this register, firmware can know whether the FIFO is fully empty.

Table 14-18 FIFO Empty Byte0

Bit	Name	Type	Reset	Bus Reset	Description
15	FEMPT_F7	R	1	-	1: FIFO15 is completely empty. 0: FIFO15 is not completely empty.
14	FEMPT_F6	R	1	-	1: FIFO14 is completely empty. 0: FIFO14 is not completely empty.
13..8	Reserved	-	-	-	-
7	FEMPT_F7	R	1	-	1: FIFO7 is completely empty. 0: FIFO7 is not completely empty.
..	..	..	..	..	…
1	FEMPT_F1	R	1	-	1: FIFO1 is completely empty. 0: FIFO1 is not completely empty.
0	FEMPT_F0	R	1	-	1: FIFO0 is completely empty.

Bit	Name	Type	Reset	Bus Reset	Description
					0: FIFO0 is not completely empty.

14.3.11.8 Initial Value of Random Pattern

Offset: 0x1E

Table 14-19 Initial Value of Random Pattern

Bit	Name	Type	Reset	Bus Reset	Description
7..0	TST_INF_INI	RW	0x0	-	USB-Controller can generate different test-patterns, of random sequence depending on the different initial value, to test the interface between PHY and USB-Controller when the bit, TST_INF, is active.

14.3.11.9 Byte Count of Random Pattern

Offset: 0x1F

Table 14-20 Byte Count of Random Pattern

Bit	Name	Type	Reset	Bus Reset	Description
7..0	TST_CONT	RW	0xFF	-	The byte count of the random test-patterns test the interface between PHY and USB-Controller is dependent on the value of TST_CONT.

14.3.12 Interrupt Source Register

14.3.12.1 Interrupt Group Mask Register

Offset: 0x20

The returned value is the raw status when micro-controller reads this register.

Table 14-21 Interrupt Group Mask Register

Bit	Name	Type	Reset	Bus Reset	Description
7	INT_SRC7	R	1	-	This bit indicates the occurrence of some interrupts in “Interrupt Source Register Byte7”
6	INT_SRC6	R	0	-	This bit indicates the occurrence of some interrupts in “Interrupt Source Register Byte6”
5	INT_SRC5	R	0	-	This bit indicates the occurrence of some interrupts in “Interrupt Source Register Byte5”
4	INT_SRC4	R	0	-	This bit indicates the occurrence of some interrupts in “Interrupt Source Register Byte4”
3	INT_SRC3	R	0	-	This bit indicates the occurrence of some interrupts in “Interrupt Source Register Byte3”
2	INT_SRC2	R	0	-	This bit indicates the occurrence of some interrupts in “Interrupt Source Register Byte2”
1	INT_SRC1	R	0	-	This bit indicates the occurrence of some interrupts in “Interrupt Source Register Byte1”

Bit	Name	Type	Reset	Bus Reset	Description
0	INT_SRC0	R	0	-	This bit indicates the occurrence of some interrupts in “Interrupt Source Register Byte0”

14.3.12.2 Interrupt Source Register Byte0

Offset: 0x21

The returned value is the raw status when micro-controller reads this register.

Table 14-22 Interrupt Source Register Byte0

Bit	Name	Type	Reset	Bus Reset	Description
7	CX_COMABT_INT	RW	0	-	This bit indicates that a command abort event has occurred. For the interrupts recoded in this source register, a command abort interrupt receives the highest priority. For a command abort interrupt, AP should only clear the CX_COMABT_INT bit, and all other operations are unnecessary and should be avoided. In general, the command abort interrupt will be accompanied by a CX_SETUP_INT. The AP should service the command abort interrupt first in order to clear the CX_COMABT_INT. This is done because the CXF FIFO is frozen for the AP access when the CX_COMABT_INT remains at ‘1’. In order to get the 8-byte for SETUP which leads to command an abort, AP should clear CX_COMABT_INT first.
6	Reserved	-	-	-	-
5	RBUF_ERR	RW	0	-	This bit Indicates that AP reads an empty FIFO. This bit should be cleared by firmware.
4	CX_COMFAIL_INT	R	0	-	This bit indicates that the control transfer has terminated abnormally. This bit will be asserted when USB-Controller receives an extra IN/OUT token, comparing to the wLength in packet and the real length of packet, at the data stage of the control transfer. Once this bit is asserted, it will be kept at ‘1’ before AP sets the CX_STL bit of the CX_Config_Status register. CX_COMFAIL_INT will be cleared automatically by HW and USB-Controller will also clear the rest redundant data stored in CX_FIFO after SW sets CX_STL. After setting the CX_STL bit of the AP should set the CX_DONE bit of the CX_Config_Status register.
3	CX_COMEND_INT	R	0	-	This bit indicates that the control transfer has entered the status stage. This bit will remain asserted before the firmware sets the CX Configuration and the CX_DONE bit of the Status Register (Address 0x0B, bit 0). This bit will remain unchanged after AP sets the CX_STL bit of the CX_Config_Status register.
2	CX_OUT_INT	R	0	-	This bit indicates that the control transfer contains valid data for transferring the control-write. This bit will remain asserted until the firmware starts to read the data from the control transfer FIFO (CXF) of USB-Controller. This bit will be cleared after AP sets the CX_STL bit of the CX_Config_Status register.

Bit	Name	Type	Reset	Bus Reset	Description
1	CX_IN_INT	R	0	-	It indicates that the firmware should write the data for the control-read transfer to the control transfer FIFO. For a control-read with length less than or equal to 64-byte, this bit will never be asserted. The firmware will decode the 8-byte data sent at the SETUP stage of the control transfer. The firmware should write the first data payload into the control transfer FIFO if the 8-byte represents the control-read transfer without assertion of this bit. This bit will be asserted only when the length of control-read transfer is longer than 64-byte and USB host has successfully received the data of previous packet. For example: For a 65-byte control-read transfer, firmware should automatically write the first 64-byte after decoding eight bytes of the SETUP data. Firmware will be interrupted to write the 65th byte when the USB host ACKs to the first 64-byte. This bit will remain asserted until the firmware starts to write data into the control transfer FIFO (CXF) of USB-Controller. This bit will be cleared after AP sets the CX_STL bit of the CX_Config_Status register.
0	CX_SETUP_INT	R	0	-	This bit will remain asserted until the firmware starts to read the data from the control transfer FIFO (CXF) of USB-Controller. This bit will remain unchanged after AP sets the CX_STL bit of the CX_Config_Status register.

14.3.12.3 Interrupt Source Register Byte1-4

Offset: 0x22..0x25

The returned value is the raw status when micro-controller reads this register. In the case of a ping-pong FIFO, the firmware only takes care of the status of the “first” FIFO. For example, if FIFO 0, FIFO 1, and FIFO 2 are ping-ponging for OUT in endpoint 1, the firmware will only need to take care of the status for FIFO 0.

Table 14-23 Interrupt Source Register Byte1-4

Bit	Name	Type	Reset	Bus Reset	Description
31	F15_SPK_INT	R	0	-	This bit becomes 1 when the short packet data are received in FIFO15. This bit is cleared once the AHB master reads FIFO15.
30	F15_OUT_INT	R	0	-	This bit becomes 1 when FIFO15 is ready to be read. This bit is cleared when all data in FIFO15 are read out.
29	F14_SPK_INT	R	0	-	This bit becomes 1 when the short packet data are received in FIFO14. This bit is cleared once the AHB master reads FIFO14.
28	F14_OUT_INT	R	0	-	This bit becomes 1 when FIFO14 is ready to be read. This bit is cleared when all data in FIFO14 are read out.
27..16	Reserved	-	-	-	-
15	F7_SPK_INT	R	0	-	This bit becomes 1 when the short packet data are received in FIFO7. This bit is cleared once the AHB master reads FIFO7.

Bit	Name	Type	Reset	Bus Reset	Description
14	F7_OUT_INT	R	0	-	This bit becomes 1 when FIFO7 is ready to be read. This bit is cleared when all data in FIFO7 are read out.
..	..	..	..	..	..
3	F1_SPK_INT	R	0	-	This bit becomes 1 when the short packet data are received in FIFO1. This bit is cleared once the AHB master reads FIFO1.
2	F1_OUT_INT	R	0	-	This bit becomes 1 when FIFO1 is ready to be read. This bit is cleared when all data in FIFO1 are read out.
1	F0_SPK_INT	R	0	-	This bit becomes 1 when the short packet data are received in FIFO0. This bit is cleared once the AHB master reads FIFO0.
0	F0_OUT_INT	R	0	-	This bit becomes 1 when FIFO0 is ready to be read. This bit is cleared when all data in FIFO0 are read out.

14.3.12.4 Interrupt Source Register Byte5-6

Offset: 0x26..0x27

The returned value is the raw status instead of the masked status when the micro-controller reads this register. In the case of a ping-pong FIFO, the firmware has only to take care of the status of the “first” FIFO. For example, if the FIFO 0, FIFO 1, and FIFO 2 are ping-ponging for the IN at endpoint 1, the firmware only needs to take care of the status for FIFO 0. Generally, these interrupt sources indicate whether the FIFO is ready to receive data from the AHB. However, if the FIFO is dedicated to an interrupt type endpoint, the interrupt source occurs only after the USB host has issued an IN transfer while the FIFO is empty.

Table 14-24 Interrupt Source Register Byte5-6

Bit	Name	Type	Reset	Bus Reset	Description
15	F15_IN_INT	R	0	-	This bit becomes 1 to indicate that FIFO15 is ready to be written. This bit can be cleared under the following three conditions: 1. A maximum-size packet is received in FIFO15. 2. AHB master ends the transfer with the signal dma_termwr. 3. CPU sets the done bit of register 0xAF.
14	F14_IN_INT	R	0	-	This bit becomes 1 to indicate that FIFO14 is ready to be written. This bit can be cleared under the following three conditions: 1. A maximum-size packet is received in FIFO14. 2. AHB master ends the transfer with the signal dma_termwr. 3. CPU sets the done bit of register 0xAE.
13..8	Reserved	-	-	-	-
7	F7_IN_INT	R	0	-	This bit becomes 1 to indicate that FIFO7 is ready to be written. This bit can be cleared under the following three conditions: 4. A maximum-size packet is received in FIFO7. 5. AHB master ends the transfer with the signal dma_termwr.

Bit	Name	Type	Reset	Bus Reset	Description
					6. CPU sets the done bit of register 0xA7.
..	..	..	..	..	..
2	F2_IN_INT	R	0	-	This bit becomes 1 to indicate that FIFO2 is ready to be written. This bit can be cleared under the following three conditions: 7. A maximum-size packet is received in FIFO2. 8. AHB master ends the transfer with the signal dma_termwr. 9. CPU sets the done bit of register 0xA2.
1	F1_IN_INT	R	0	-	This bit becomes 1 to indicate that FIFO1 is ready to be written. This bit can be cleared under the following three conditions: 1. A maximum-size packet is received in FIFO1. 2. AHB master ends the transfer with the signal dma_termwr. 3. CPU sets the done bit of register 0xA1.
0	F0_IN_INT	R	0	-	This bit becomes 1 to indicate that FIFO0 is ready to be written. This bit can be cleared under the following three conditions: 1. A maximum-size packet is received in FIFO0. 2. AHB master ends the transfer with the signal dma_termwr. 3. CPU sets the done bit of register 0xA0.

14.3.12.5 Interrupt Source Register Byte7

Offset: 0x28

When a bus reset occurs, bits[3:1] will be reset to “001”.
As to the condition that AP cannot catch up the FIFO access through AHB, USB-Controller will not issues interrupt. Instead, USB-Controller will blocks transfer data from host.
USB-Controller will discard the ISO OUT if there still is data in FIFO
USB-Controller will issue 0-byte ISO IN if the FIFO is empty.

This will cause the data lost in a (Micro) frame, but USB-Controller will work as usual and be ready at the next (Micro) frame.
Please note that when the TX0BYTE_IEPx bit (Please refer to 14.3.14.3 for details) is applied to an IN endpoint, AP should not send the next packet to the endpoint until the TX0BYTE_INT interrupt occurs. The returned value is the raw status when the micro-controller reads this register.

Table 14-25 Interrupt Source Register Byte7

Bit	Name	Type	Reset	Bus Reset	Description
7	RX0BTYE_INT	RW	0	-	Received zero-length data packet interrupt USB-Controller receives a zero-length data packet from the USB host. When USB-Controller receives a zero-length data packet from the USB host, this bit will be set. The firmware may further check the registers, 0x19 and 0x1A, to determine which endpoint receives a zero-length data packet from the USB host. When interrupt occurs, USB-Controller will NAK the next OUT

Bit	Name	Type	Reset	Bus Reset	Description
					transaction to the same endpoint until the corresponding bit (0x19 or 0x1A) is cleared by firmware.
6	TX0BYTE_INT	RW	0	-	Transferred zero-length data packet interrupt USB-Controller returned a zero-length data packet to the USB host. This bit will be set under the following two cases: 1. When the USB host issues an IN transaction to an isochronous endpoint while USB-Controller is not ready to return data, USB-Controller will transfer a zero-length data packet to the USB host. In such cases, this bit will be used. 2. When the TX0BYTE_IEPx bit is set (Please refer to Section 14.3.14.3), and after the endpoint's data in the FIFO are transferred, USB-Controller will return a zero-length data packet to the next IN transaction to the same endpoint. The firmware may further check the registers, 0x2D and 0x2E, to determine which endpoint returns a zero-length data packet to the USB host. After the AP has serviced the interrupt request, this bit must be cleared by firmware.
5	ISO_SEQ_ABORT_INT	RW	0	-	ISO sequential abort interrupt A high bandwidth isochronous sequential abort When USB-Controller detects an incomplete DATA PID sequence during a micro-frame, this bit will be set. For example, if USB-Controller detects an MDATA followed by an SOF, this is taken as a sequential abort. The firmware should further check the registers 0x2B and 0x2C to determine which endpoint received an isochronous sequential abort. After the AP has serviced the interrupt request, this bit must be cleared by firmware.
4	ISO_SEQ_ERR_INT	RW	0	-	ISO sequential error interrupt High bandwidth isochronous sequential error When USB-Controller detects a DATA PID sequence error in an isochronous transaction in high bandwidth, this bit will be set. Any out-of-order sequence will be taken as a sequence error. The firmware should further check the registers, 0x29 and 0x2A, to determine which endpoint received an isochronous sequential error. After AP has serviced the interrupt request, this bit must be cleared by firmware.
3	RESM_INT	RW	0	0	Resume interrupt Resume-state-change interrupt bit When USB-Controller detects a resume event from the host, this bit will be set. After AP has serviced the interrupt request, this bit must be cleared by firmware.
2	SUSP_INT	RW	0	0	Suspend interrupt Suspend-state-change interrupt bit When the USB bus remains at an idle state for over 3 ms, this bit will be set. This bit must be cleared before firmware sets the "GOSUSP" in register 0x00.
1	USBRST_INT	RW	0	1	USB reset interrupt Bus reset interrupt bit When USB-Controller detects a USB bus reset from the

Bit	Name	Type	Reset	Bus Reset	Description
					host, SE0 longer than 2.5 µs, the bit will be set. This bit is asserted once for the current USB reset before this interrupt cleared. When AP has serviced the interrupt request, this bit must be cleared by firmware.
0	HBF_EMPTY_INT	R	1	-	HBF empty interrupt Before sending data to USB-Controller, AP should make sure that the HBF is empty and the FIFO is not full. Please note that the value of this bit is not masked during CPU reads.

14.3.12.6 Isochronous Sequential Error Register Byte0-1

Offset: 0x29..0x2A

When an Isochronous Sequential Error Interrupt occurs (Bit 4 of register 0x28), the firmware should further check registers 0x29 and 0x2A to determine which endpoint received an isochronous sequential error. This bit should be cleared by firmware.

Table 14-26 Isochronous Sequential Error Register Byte0-1

Bit	Name	Type	Reset	Bus Reset	Description
15..9	Reserved	-	-	-	-
8	ISO_SEQ_ERR_EP8	RW	0	-	Endpoint8 receives an isochronous sequential error.
..	..	..	..	..	..
2	ISO_SEQ_ERR_EP2	RW	0	-	Endpoint2 receives an isochronous sequential error.
1	ISO_SEQ_ERR_EP1	RW	0	-	Endpoint1 receives an isochronous sequential error.
0	Reserved	-	-	-	-

14.3.12.7 Isochronous Sequential Abort Register Byte0-1

Offset: 0x2B..0x2C

Table 14-27 isochronous Sequential Abort Register Byte0-1

Bit	Name	Type	Reset	Bus Reset	Description
15..9	Reserved	-	-	-	-
8	ISO_SEQ_ABT_EP8	RW	0	-	Endpoint8 receives an isochronous sequential abort.
..	..	..	..	..	..
2	ISO_SEQ_ABT_EP2	RW	0	-	Endpoint2 receives an isochronous sequential abort.
1	ISO_SEQ_ABT_EP1	RW	0	-	Endpoint1 receives an isochronous sequential abort.
0	Reserved	-	-	-	-

14.3.12.8 Transferred zero-length Register Byte0-1

Offset: 0x2D..0x2E

Table 14-28 Transferred zero-length Register Byte0-1

Bit	Name	Type	Reset	Bus Reset	Description
15..9	Reserved	-	-	-	-
8	tx0byte_ep8	RW	0	-	Endpoint8 transfers a zero-length data packet.
..	..	..	..	..	..
2	tx0byte_ep2	RW	0	-	Endpoint2 transfers a zero-length data packet.
1	tx0byte_ep1	RW	0	-	Endpoint1 transfers a zero-length data packet.
0	Reserved	-	-	-	-

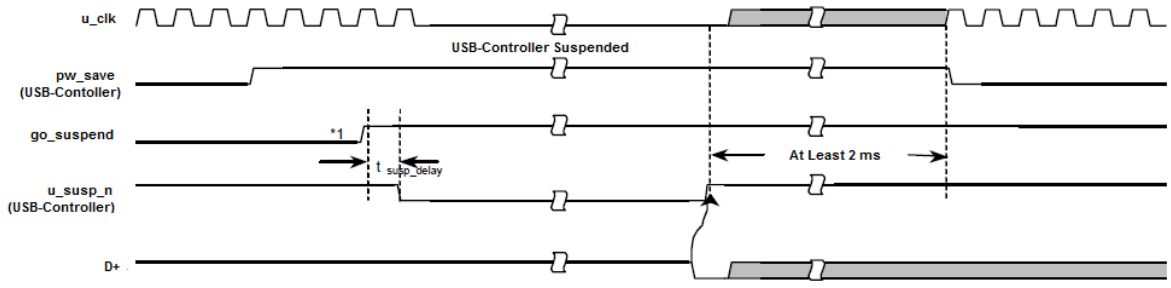
14.3.13 Idle Counter

Offset: 0x2F

Table 14-29 Idle Counter

Bit	Name	Type	Reset	Bus Reset	Description
7..3	Reserved	R	0	-	-
2..0	IDLE_CNT	RW	0	-	It controls the timing delay from the time indicated in the GOSUSP bit of the main control register to the time USB-Controller enters a suspend mode. For reference, the delay is denoted as t_{susp_delay} in the list below: 000: t_{susp_delay} = 0 ms 001: t_{susp_delay} = 1 ms 010: t_{susp_delay} = 2 ms 011: t_{susp_delay} = 3 ms 100: t_{susp_delay} = 4 ms 101: t_{susp_delay} = 5 ms 110: t_{susp_delay} = 6 ms 111: t_{susp_delay} = 7 ms Note: The USB 2.0 specification defines T_{SUSP} to mandate that the device should enter the suspend mode no later than 10 ms after the D+/D- is continuously at an idle state. The firmware programmer should be cautious when programming the value of t_{susp_delay}.

Figure 14-12 Timing Diagram of t_{susp_delay} Programing



Note: go_suspend is the internal signal of USB-Controller, which is not visible to AP. However, AP can set/clear it by writing 1/0 to GOSUSP of the main control register. The value of $t_{suspend\_delay}$ can be set by AP. Please refer to the idle_cnt register for details.

14.3.14 Endpoint Configuration and status Register

14.3.14.1 Endpoint1-8 Map Register

Offset: 0x30(Endpoint1)..0x38(Endpoint8)

Table 14-30 Endpoint1-8

Bit	Name	Type	Reset	Bus Reset	Description
7..4	FNO_OEP1/8	RW	0xF	-	FIFO Number for the OUT Endpoint1-8 This bit records the physical FIFO number for the logical-out Endpoint1-8.
3..0	FNO_IEP1/8	RW	0xF	-	FIFO Number for the IN Endpoint1-8 This bit records the physical FIFO number for the logical-in Endpoint1-8.

14.3.14.2 HBF Data Byte Count

Offset: 0x3F

This register contains the byte count information of HBF.

Table 14-31 HBF Data Byte Count

Bit	Name	Type	Reset	Bus Reset	Description
7..5	Reserved	-	-	-	-
4..0	HBF_CNT	R	0x0	-	HBF data byte count This bit records the data byte count in HBF.

14.3.14.3 IN Endpoint1-15 MaxPacketSize Register Word

Offset: 0x40(Endpoint1)..0x50(Endpoint8)

Table 14-32 Transferred zero-length Register Byte0-1

Bit	Name	Type	Reset	Bus Reset	Description
15	TX0BYTE_IEP1/8	RW	0	-	Transfer a zero-length data packet from the Endpoint1-8 to USB host This bit should be set after the last transaction packet is sent to FIFO by the AHB master. After the data of endpoint in FIFO are transferred, USB-Controller will return a zero-length data packet to the next IN transaction to the same endpoint. AP should not send the next packet to the same endpoint until the TX0BYTE_INT interrupt for the endpoint occurs. This bit is cleared by hardware automatically when TX0BYTE_INT occurs.
14..13	TX_NUM_HBW_IEP1/8	RW	0x0	-	Transaction number for high-bandwidth Endpoint1-

Bit	Name	Type	Reset	Bus Reset	Description
					8 TX_NUM_HBW: "0X" These indicate that the endpoint x is non-high-bandwidth. "10" This indicates that there are two transactions per microframe. "11" This indicates that there are three transactions per microframe. While using three transactions per microframe, the mapped FIFO must be set as triple blocks (Please refer to the FIFO configuration register for details). While using two transactions per microframe, the mapped FIFO must be set as double blocks.
12	RSTG_IEP1/8	RW	0	-	Reset toggle sequence for the IN Endpoint1-8 Firmware resets the toggle bit of the indexed endpoint1-8 by writing a '1' to this bit. This bit should also be cleared by firmware.
11	STL_IEP1/8	RW	0	0	Stall IN Endpoint1-8 The indexed endpoint1-8 can be stalled by writing a '1' to this bit. The stall status of the indexed endpoint1-8 can be cleared by writing a '0' to this bit.
10..0	MAXPS_IEP1/8	RW	0x200	-	Max packet size of the IN Endpoint1-8 The maximum packet size bits of endpoint1-8, which is capable of sending or receiving the data smaller than or equal to this size.

14.3.14.4 OUT Endpoint1-8 MaxPacketSize Register Word

Offset: 0x60(Endpoint1)..0x70(Endpoint8)

Table 14-33 Transferred zero-length Register Byte0-1

Bit	Name	Type	Reset	Bus Reset	Description
15..13	Reserved	-	-	-	-
12	RSTG_OEP1/8	RW	0	-	Reset toggle sequence for the OUT Endpoint1-8 Firmware resets the toggle bit of the indexed endpoint1-8 by writing a '1' to this bit. This bit should also be cleared by firmware.
11	STL_OEP1/8	RW	0	0	Stall OUT Endpoint1-8 Writing a '1' to STL_OEP will stall endpoint1-8. The stall status of the indexed endpoint1-8 can be cleared by writing a '0' to this bit. Note: After setting this bit to '1', AP still needs to monitor the OUT interrupt of this endpoint. If the OUT interrupt asserts, AP must read out the data in the OUT FIFO even AP does not need this data.
10..0	MAXPS_OEP1/8	RW	0x200	-	Max packet size of the OUT Endpoint1-8 The maximum packet size bits of endpoint1-8, which is capable of sending or receiving the data smaller than or equal to this size.

14.3.14.5 DMA Mode Enable Register Word

Offset: 0x7E

This register allows USB-Controller to assert USB-Controller_dma_req_r according to the enabled FIFO. Only one bit should be set as '1' between 0x7E and 0x7F at any time. This setting informs USB-Controller that FIFO is currently being serviced by the DMA controller. For example, assume that the FIFO0 and FIFO1 are ping-ponging for the IN endpoint1. Only bit0 of 0x7E should be set by firmware. USB-Contoller_dma_req_r is asserted when either the FIFO0 or FIFO1 is not full. If endpoint1 is for OUT, USB-Controller_dma_req_r will be asserted when either the FIFO0 or FIFO1 is not empty.

Table 14-34 DMA Mode Enable Register Word

Bit	Name	Type	Reset	Bus Reset	Description
15	FIFO15_DMA_EN	RW	0	-	FIFO15 is currently being serviced by the DMA controller
14	FIFO14_DMA_EN	RW	0	-	FIFO14 is currently being serviced by the DMA controller
13..8	Reserved	-	-	-	-
7	FIFO7_DMA_EN	RW	0	-	FIFO7 is currently being serviced by the DMA controller
..	..	..	..	..	..
1	FIFO1_DMA_EN	RW	0	-	FIFO1 is currently being serviced by the DMA controller
0	FIFO0_DMA_EN	RW	0	-	FIFO0 is currently being serviced by the DMA controller

14.3.15 FIFO Configuration and Status Register

14.3.15.1 FIFO0-7 and FIFO14-15 Map Register

Offset: 0x80(FIFO0)..0x87(FIFO7) and 0x8E(FIFO14)..0x8F(FIFO15)

Table 14-35 FIFO0-7 and FIFO14-15 Map Register

Bit	Name	Type	Reset	Bus Reset	Description
7..5	Reserved	-	-	-	-
4	DIR_F0/7+14/15	RW	0	-	FIFO0-7+14-15 direction: Data transfer direction 0 = OUT 1 = IN
3..0	EP_FIFO0/7+14/15	RW	0xF	-	Endpoint number for FIFO0-7+14-15 These bits record the physical endpoint number for physical FIFO0-7+14-15

14.3.15.2 FIFO0-7 and FIFO14.15 Configuration Register

Offset: 0x90(FIFO0)..0x97(FIFO7) and 0x9E(FIFO14)..0x9F(FIFO15)

The USB Device Controller contains 10 FIFOs that are numbered from 0 to 7 and 14 to 15. FIFO0 to FIFO7 are 512-byte, while FIFO14 and FIFO15 are 64-byte each.
A “block” can be composed of one FIFO or two FIFOs, depending on the setting of the bit, BLKSZ_Fx. When this bit is set to ‘1’, the block size is 1024-byte; when this bit is set as ‘0’, the block size is 512-byte. For example, if 0x90 bit4 is set to ‘1’, it means that FIFO0 and FIFO1 will be combined as a 1024-byte FIFO; otherwise it will known as a “block”.
The ping-pong FIFO mechanism is block-based. For example, if bit4 of the 0x90 is set to ‘1’ and bits[3:2] are set to “10”, when FIFO0 is accessed, three 1024-byte blocks (FIFO0 combined with FIFO1, FIFO2 combined with FIFO3, and FIFO4 combined with FIFO5) will be ping-pong in turn.

Table 14-36 FIFO0-15 Configuration Register

Bit	Name	Type	Reset	Bus Reset	Description
7	EN_F 0/7+14/15	RW	0	-	Enable FIFO 0-7+14-15 A ‘1’ indicates that the FIFO is enabled.
6..5	Reserved	R	0	-	-
4	BLKSZ_F 0/7+14/15	RW	0	-	Block Size of FIFO 0-7+14-15 BLKSIZE_Fx = 0 FIFO0-FIFO7: For transferring packets whose maximum packet size is smaller than or equal to 512 bytes. FIFO14/FIFO15: For transferring packets whose maximum packet size is smaller than or equal to 64 bytes. BLKSIZE_Fx = 1 FIFO0-FIFO7: For transferring packets whose maximum packet size is smaller than or equal to 1024 bytes and greater than 512 bytes. FIFO14/FIFO15: For transferring packets whose maximum packet size is smaller than or equal to 128 bytes and greater than 64 bytes.
3..2	BLKNUM_F 0/7+14/15 0/7+14/15	RW	0	-	Block number of FIFO0-7+14-15 BLKNUM_Fx = 00: Single block BLKNUM_Fx = 01: Double blocks BLKNUM_Fx = 10: Triple blocks BLKNUM_Fx = 11: Reserved
1..0	TYP_F 0/7+14/15	RW	0	-	Transfer type of FIFO0-7+14-15 These bits indicate the transfer type used for FIFO0-7+14-15 transfer. TYP_Fx = 00: Reserved TYP_Fx = 01: Isochronous type TYP_Fx = 10: Bulk type TYP_Fx = 11: Interrupt type

14.3.15.3 FIFO0-7 and FIFO14-15 Instruction Register

Offset: 0xA0(FIFO0)..0xA7(FIFO7) and 0xAE(FIFO14)..0xAF(FIFO15)

Table 14-37 DMA Mode Enable Register Word

Bit	Name	Type	Reset	Bus Reset	Description
7..5	Reserved	-	-	-	-
4	FFRST0/7+14/15	RW	0	-	FIFO0-7+14-15 reset FIFO can be reset by firmware through setting this bit. This bit must be cleared by firmware after resetting FIFO.
3	DONE_F0/7+14/15	RW	0	-	Data transfer is done for IN FIFO0-7+14-15. The firmware must set this bit to inform the HBS the whole transaction is completed. This bit can be automatically cleared by hardware.
2..0	BC_F0/7+14/15 [10..8]	RW	0	-	OUT FIFO 0-7+14-15 byte count BC_Fx [10:0] indicates the byte number of data stored in the FIFO for OUT EPx.

14.3.15.4 FIFO0-7 and FIFO14-15 Byte-Count Register

Offset: 0xB0(FIFO0)..0xB7(FIFO7) and 0xBE(FIFO14)..0xBF(FIFO15)

The value of the byte-count register is not valid after hardware reset. AP should check it only when the OUT interrupts are issued.

Table 14-38 DMA Mode Enable Register Word

Bit	Name	Type	Reset	Bus Reset	Description
7..0	BC_F0/7+14/15 [7:0]	R	1	-	OUT FIFO0-7+14-15 byte count BC_F0-7+14-15 [10:0] indicates the byte number of data stored in the FIFO for OUT EPx.

14.3.16 Data Port Register

Offset: 0xC0(FIFO0)..0xDC(FIFO7) and 0xF8(FIFO14)..0xFC(FIFO15)

This address provides a port for an AHB master to access the PAM FIFO. For example, if an AHB master intends to read the data from FIFO0, it should always issue the read cycle with the address, 0xC0. In the case of a ping-pong FIFO, the AHB master should access the data via the address of the “first” FIFO. For example, if FIFO2 and FIFO3 are ping-ponging for endpoint2, the AHB master will always access the data of endpoint2 via the address, 0xC8 only. It is invalid for the AHB master to access endpoint2 via the address, 0xC8, in the example.

14.4 Programming Guide

14.4.1 Init USB

```
read  0x00 temp       // read current value of main ctrl reg
write 0x00 temp|0x10  // enable soft reset
write 0x00 temp&~0x10 // disable soft reset
write 0x00 temp|0x24  // set globel interrupt and chip enable
write 0x5c[4] 0x01    // ANTAIOS_PINC_USB_VBUS_ON
wait 3ms
read  0x08 temp
write 0x08 temp&~0x01 // clear unplug bit
```

14.4.2 Interrupt Handling

```
Wait for interrupt
GRPINT = (ReadUSBdev(Address 0x20) & ~MGPRINT (Address 0x10))
if ((GRPINT[0] & ~MGPRINT[0] == 1)
  // Check Control transfer status
if ((GRPINT[7] & ~MGPRINT[7] == 1)
  // Special condition interrupt
if (GRPINT[3] & ~MGPRINT[3]== 1 or GRPINT[2] & ~MGPRINT[2] == 1 or
GRPINT[1] & ~MGPRINT[1] == 1)
  // OUT Endpoint has data in FIFO
  Start data transfer
if (GRPINT[6] & ~MGPRINT[6] == 1 or GRPINT[5] & ~MGPRINT[5] == 1)
// FIFO for IN Endpoint is not full
  if there is data available
  Start data transfer
Else
  Mask interrupt
```

14.4.3 IN Endpoint Data Transfer

```
if ReadUSBdev(Address 0x26) != 0
  calculate corresponding interrupt FIFO_number(n)
else
  ReadUSBdev(Address 0x27)
  calculate corresponding interrupt FIFO_number(n)
mask interrupt of FIFO_number
wait for buffer empty
write data to FIFO_number
if write data complete
  if write data number is less than max_packet
    // Setting Done_Fn, n=0, 1, 2, …, f
    _size write 1 to USBdev(Address 0xAn) : bit 3
unmask interrupt
```

[4] Pin Controller Offset (0x73000000)

14.4.4 OUT Endpoint Data Transfer

```
if ReadUSBdev(Address 0x22) != 0
  calculate interrupt FIFO_number(n)
else if ReadUSBdev(Address 0x23) != 0
  Calculate interrupt FIFO_number(n)
else
  ReadUSBdev(Address 0x24)
  Calculate interrupt FIFO_number(n)
mask interrupt of FIFO_number
if short_packet
  read ByteCount of FIFO_number
else
  ByteCount = MAX_PACKET_SIZE(FIFO_number)
Read data from FIFO_number
unmask interrupt of FIFO_number
```

14.4.5 Setting the Relationship between Internal FIFO and Endpoint

Two registers have to be set to establish the relationship between the internal FIFO and the endpoint. The user must make these two registers match. For example, if the user wants to use EP1 with FIFO1, users must set EP1 to map to FIFO1 and set FIFO1 to map to EP1 for USB Device Controller to work correctly.

See Table 14-30 and Table 14-35.

The two registers are in contrast. For example, if users want to have two endpoints, one being endpoint1 for the bulk-in direction and the other being endpoint3 for the isochronous-out direction. If users choose FIFO0 to store the endpoint1 data and FIFO5 to store the endpoint3 data, the setting will be:

```
write 0x30 0xF0 // Assign IN Endpoint1 to FIFO0
write 0x32 0x5F // Assign OUT Endpoint3 to FIFO5
write 0x80 0x11 // Dedicate FIFO0 to IN Endpoint1
write 0x85 0x03 // Dedicate FIFO5 to OUT Endpoint3
```

14.4.6 Setting the Endpoint Properties

For example, if users want to have two endpoints, one being endpoint1 for the bulk-in direction with the ping-pong buffers and max_packet_size = 512 bytes, while the other being endpoint3 for the isochronous-out direction with single buffer and max_packet_size = 1024 bytes, users can choose FIFO0 and FIFO1 to store the endpoint1 data, and FIFO5 combined with FIFO6 to store endpoint3 data, then the setting will be:

See Table 14-30, Table 14-32, Table 14-33, Table 14-35 and Table 14-36

```
write 0x30 0xF0 // Assign IN Endpoint1 to FIFO0
write 0x32 0x5F // Assign OUT Endpoint3 to FIFO5
write 0x80 0x11 // Dedicate FIFO0 to IN Endpoint1
write 0x85 0x03 // Dedicate FIFO5 to OUT Endpoint3
write 0x40 0x00 // Set max_Packet_size of IN Endpoint1
```

```
                    // to be 512 bytes
write 0x41 0x02
write 0x64 0x00 // Set max_Packet_size of OUT Endpoint3
                    // to be 1024 bytes
write 0x65 0x04
write 0x90 0x86 // Set FIFO0 to be bulk 512 bytes,
                    // double buffer
write 0x91 0x06 // Set FIFO1 to be bulk 512 bytes,
                    // double buffer, setting the FIFO enable
                    // bit is unnecessary
write 0x95 0x91 // Set FIFO5 to be isochronous 1024 bytes,
                    // single buffer
                    // Please note FIFO6 has also been used in
                    // conjunction with FIFO5 to become
                    // a 1024-byte FIFO
```

14.4.7 Endpoints Access through Coexisting DMA/PIO Modes

In an application-specified system, different USB endpoints may be defined to be accessed by different mechanisms, such as DMA and PIO. This system may consist of two or more AHB masters accessing non-controlled endpoints. Each master accesses a different endpoint. When this application requirement is raised, some key issues need to be taken care of in the USB Device Controller programming, because concurrent FIFO access through DMA and PIO is not supported in USB Device Controller.
The recommended solution for this kind of usage may be that, when an application is interrupted by an endpoint and this endpoint is serviced by DMA, software has to set the DMA mode enable register and the DMA control setup for the corresponding FIFO number. After the setup, the software can enable DMA and ask DMA to access the data FIFO. When DMA finishes accessing this FIFO, the software should clear the DMA mode enable register and disable DMA. Briefly speaking, DMA should only be enabled in the period of DMA accessing FIFO and be disabled during normal operations.

15 Advanced IRQ Controller

Offset: 0x80000000

15.1 Interrupt Sources

Table 15-1 Interrupt Sources

Number	Bit LO-Reg	Bit HI-Reg	Description
63		31	Asynchronous AHB2AHB bridge error Interrupt
62		30	FIFO-System Interrupt CPU side
61		29	FIFO-System Interrupt CP side
60		28	RTE IRQ source 7
59		27	RTE IRQ source 6
58		26	RTE IRQ source 5
57		25	RTE IRQ source 4
56		24	RTE IRQ source 3
55		23	RTE IRQ source 2
54		22	RTE IRQ source 1
53		21	RTE IRQ source 0
52		20	VPC+ Sync Interrupt
51		19	AHB to APB Bridge 2 DMA Controller Interrupt
50		18	EXT_INT; PBUS Alarm line, filtered[5]
49		17	DMA Controller ERR Interrupt
48		16	AHB to APB Bridge 1 DMA Controller Interrupt
47		15	DMA Controller Terminal Count Interrupt
46		14	AHB Controller Interrupt
45		13	SNAP+ Master Interrupt
44		12	MMC/SD Card Interrupt
43		11	Reserved
42		10	High Speed UART 2 Interrupt
41		9	High Speed UART 1 Interrupt
40		8	I2C Controller 2 Interrupt
39		7	I2C Controller 1 Interrupt
38		6	QuadSPI Controller Interrupt
37		5	SPI Controller Interrupt
36		4	GPIO Interrupt
35		3	Timer 1 Interrupt
34		2	Timer 2 Interrupt
33		1	Timer 3 Interrupt
32		0	Timer 4 Interrupt

[5] The user can configure the level (high active / low active) or edge (rising / falling) of the external interrupt sources and can choose a filter time for the external interrupts between 10.4ns and 2.6µs.

Number	Bit LO-Reg	Bit HI-Reg	Description
31	31		Timer 5 Interrupt
30	30		Timer 6 Interrupt
29	29		WatchDog Interrupt
28	28		reserved
27	27		PBUS Controller Interrupt
26	26		CAN 2 Interrupt
25	25		CAN 1 Interrupt
24	24		VPC+ Interrupt
23	23		USB 2.0 Device Controller Interrupt
22	22		NAND Flash Controller Interrupt
21	21		USB Resume Interrupt
20	20		Gigabit MAC Interrupt
19	19		Profibus Master 2 IRQ0
18	18		Profibus Master 1 IRQ0
17	17		reserved
16	16		Profibus Master 2 IRQ1
15	15		Profibus Master 1 IRQ1
14	14		reserved
13	13		DMA Controller Interrupt (Combines TC and ERR)
12	12		reserved
11	11		reserved
10	10		reserved
9	9		reserved
8	8		reserved
7	7		EXT_INT; PortD_44 (only available when using AEI; AEI_INT1)[5]
6	6		reserved
5	5		reserved
4	4		EXT_INT; PortD_43 (only available when using AEI; AEI_INT0)[5]
3	3		reserved
2	2		reserved
1	1		reserved
0	0		TechIO Interrupt

15.2 Register Mapping

Table 15-2 Register Mapping

Offset	Byte 3	Byte 2	Byte 1	Byte 0	Access	Reset Value
0x000	Status_LO				RO	0x0
0x004	Status_HI				RO	0x0
0x008	EnSet_LO_P0 (lowest Priolevel)				RW	0x0
0x00C	EnSet_HI_P0				RW	0x0
0x010	EnSet_LO_P1				RW	0x0
0x014	EnSet_HI_P1				RW	0x0
0x018	EnSet_LO_P2				RW	0x0
0x01C	EnSet_HI_P2				RW	0x0
0x020	EnSet_LO_P3				RW	0x0
0x024	EnSet_HI_P3				RW	0x0
0x028	EnSet_LO_P4				RW	0x0
0x02C	EnSet_HI_P4				RW	0x0
0x030	EnSet_LO_P5				RW	0x0
0x034	EnSet_HI_P5				RW	0x0
0x038	EnSet_LO_P6				RW	0x0
0x03C	EnSet_HI_P6				RW	0x0
0x040	EnSet_LO_P7 (highest Priolevel)				RW	0x0
0x044	EnSet_HI_P7				RW	0x0
0x048	EnClear_LO				WO	-
0x04C	EnClear_HI				WO	-
0x050	MaskSet_LO				RW	0xFFFFFFFF
0x054	MaskSet_HI				RW	0x7FFFFFFF
0x058	MaskClear_LO				WO	-
0x05C	MaskClear_HI				WO	-
0x060	Ack_LO				WO	-
0x064	Ack_HI				WO	-
0x068	IRQIsrVec				RO	0xFFFFFFFF
0x06C	RoundRobinCtrl		IRQPrioLevel	IRQNum	RW	0xFFFFFFFF
0x070	FIQIsrVec				RO	0xFFFFFFFF
0x074	RoundRobinCtrl		FIQPrioLevel	FIQNum	RW	0xFFFFFFFF
0x078	IntIsFIQ_LO				RW	0x0
0x07C	IntIsFIQ_HI				RW	0x0
0x080	IsrVec0				RW	0x0
⋮					⋮	⋮
0x17C	IsrVec63				RW	0x0
0x180	REQ_Status_LO				RO	0x0
0x184	REQ_Status_HI				RO	0x0
0x188	PBUS Alarm line (config)		AEI_INT1 (PortD_44) (config)		RW	0x01100112
0x18C	reserved				RW	0x01100112
0x190	reserved		AEI_INT0 (PortD_43) (config)		RW	0x00000112
0x194	reserved					0x01100110
0x198	reserved			Ext_INT_Status	RO	0x0

Offset	Byte 3	Byte 2	Byte 1	Byte 0	Access	Reset Value
0x19C	reserved		µs_Prescaler		RW	0x60
0x1A0	ThrottleMask_LO				RO	0x0
0x1A4	ThrottleMask_HI				RO	0x0
0x1A8	ThrottleTimeInt0				RW	0x0
⋮					⋮	⋮
0x2A4	ThrottleTimeInt63				RW	0x0

15.3 Register Description

15.3.1 Status_LO Register

- Address: 0x000…0x003
- Reset value: 0x0
- Access: Read-only

This register displays the status of the interrupt sources 31 to 0. If a source requests an interrupt the corresponding bit is set to ‘1’ if the interrupt is enabled and not masked.

Table 15-3 Status_LO Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x000 ... 0x003	Status of the interrupt sources 31..0				Status_LO register

15.3.2 Status_HI Register

- Address: 0x004…0x007
- Reset value: 0x0
- Access: Read-only

This register displays the status of the interrupt sources 63 to 32. If a source requests an interrupt the corresponding bit is set to ‘1’ if the interrupt is enabled and not masked.

Table 15-4 Status_HI Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x004 ... 0x007	Status of the interrupt sources 63..32				Status_HI register

15.3.3 EnSet_LO_Pn Register

- Address: 0x008+n*8…0x00B+n*8 (n = 0..7)
- Reset value: 0x0
- Access: Read/Write

The AIRQC supports up to eight priority levels with level 0 being the lowest and level 7 being the highest priority. Each of the interrupt sources can be programmed to one of these eight priority levels independently. The registers EnSetLO_Pn (with n from 0 to 7) are used to enable the interrupt sources 31 to 0 on a specific priority level by setting the corresponding bit to ‘1’. To disable an interrupt registers EnClear_XY must be used.
The following table shows the register EnSet_LO_P0.

Table 15-5 EnSet_LO_Pn Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x008 ... 0x00B	Enable interrupt source by setting the corresponding bit to ‘1’.				EnSet_LO_P0 register

15.3.4 EnSet_HI_Pn Register

- Address: 0x00C+n*8…0x00F+n*8 (n = 0..7)
- Reset value: 0x0
- Access: Read/Write

The AIRQC supports up to eight priority levels with level 0 being the lowest and level 7 being the highest priority. Each of the interrupt sources can be programmed to one of these eight priority levels independently. The registers EnSetHI_Pn (with n from 0 to 7) are used to enable the interrupt sources 63 to 32 on a specific priority level by setting the corresponding bit to ‘1’. To disable an interrupt registers EnClear_XY must be used.
The following table shows the register EnSet_HI_P0.

Table 15-6 EnSet_HI_Pn Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x00C ... 0x00F	Enable interrupt source by setting the corresponding bit to ‘1’.				EnSet_HI_P0 register

15.3.5 EnClear_LO Register

- Address: 0x048…0x04B
- Reset value: na
- Access: Write-only

To disable one or more of the interrupt sources 31..0 the user must set the corresponding bits in the EnClear_LO register to ‘1’. The interrupt source is disabled no matter on which priority level it was enabled.

Table 15-7 EnClear_LO Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x048 ... 0x04B	Disable interrupt source 31..0 by setting the corresponding bit to ‘1’.				EnClear_LO register

15.3.6 EnClear_HI Register

- Address: 0x04C…0x04F
- Reset value: na
- Access: Write-only

To disable one or more of the interrupt sources 63 to32 the user must set the corresponding bits in the EnClear_HI register to ‘1’. The interrupt source is disabled no matter on which priority level it was enabled.

Table 15-8 EnClear_HI Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x04C ... 0x04F	Disable interrupt source 63..32 by setting the corresponding bit to ‘1’.				EnClear_HI register

15.3.7 MaskSet_LO Register

- Address: 0x050…0x053
- Reset value: 0xFFFFFFFF
- Access: Read/Write

To mask one or more of the interrupt sources 31 to 0 the user must set the corresponding bits in the MaskSet_LO register to ‘1’. The interrupt source is mask no matter on which priority level it was enabled.

Table 15-9 MaskSet_LO Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x050 ... 0x053	Mask interrupt source 31..0 by setting the corresponding bit to ‘1’.				MaskSet_LO register

15.3.8 MaskSet_HI Register

- Address: 0x054…0x057
- Reset value: 0x001FFFFF
- Access: Read/Write

To mask one or more of the interrupt sources 63 to 32 the user must set the corresponding bits in the MaskSet_HI register to ‘1’. The interrupt source is masked no matter on which priority level it was enabled.

Table 15-10 MaskSet_HI Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x054 ... 0x057	Mask interrupt source 63..32 by setting the corresponding bit to ‘1’.				MaskSet_HI register

15.3.9 MaskClear_LO Register

- Address: 0x058…0x05B
- Reset value: na
- Access: Write-only

To unmask one or more of the interrupt sources 31 to 0 the user must set the corresponding bits in the MaskClear_LO register to ‘1’. The interrupt source is unmasked no matter on which priority level it was enabled.

Table 15-11 MaskClear_LO Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x058 ... 0x05B	Unmask interrupt source 31..0 by setting the corresponding bit to ‘1’.				MaskClear_LO register

15.3.10 MaskClear_HI Register

- Address: 0x05C…0x05F
- Reset value: na
- Access: Write-only

To unmask one or more of the interrupt sources 63 to 32 the user must set the corresponding bits in the MaskClear_HI register to ‘1’. The interrupt source is unmasked no matter on which priority level it was enabled.

Table 15-12 MaskClear_HI Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x05C ... 0x05F	Unmask interrupt source 63..32 by setting the corresponding bit to ‘1’.				MaskClear_HI register

15.3.11 Ack_LO Register

- Address: 0x060…0x063
- Reset value: na
- Access: Write-only

To acknowledge one or more of the interrupt sources 31 to 0 the user must set the corresponding bits in the Ack_LO register to ‘1’. The interrupt source is acknowledged no matter on which priority level it was enabled.

Table 15-13 Ack_LO Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x060 ... 0x063	Acknowledge interrupt source 31..0 by setting the corresponding bit to ‘1’.				Ack_LO register

15.3.12 Ack_HI Register

- Address: 0x064…0x067
- Reset value: na
- Access: Write-only

To acknowledge one or more of the interrupt sources 63 to 32 the user must set the corresponding bits in the Ack_HI register to ‘1’. The interrupt source is acknowledged no matter on which priority level it was enabled.

Table 15-14 Ack_HI Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x064 ... 0x067	Acknowledge interrupt source 63..32 by setting the corresponding bit to ‘1’.				Ack_HI register

15.3.13 IRQIsrVec Register

- Address: 0x068…0x06B
- Reset value: 0xFFFFFFFF
- Access: Read-only

For each of the 64 interrupt sources the user can store an individual start address of the Interrupt Service Routine in the registers IsrVec0 to IsrVec63. If an interrupt is asserted the address inside the corresponding IsrVecN register is displayed in register IRQIsrVec and can be used call the Interrupt Service Routine for the interrupt source directly.

Table 15-15 IRQIsrVec Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x068 ... 0x06B	32Bit start address of the ISR of the asserted interrupt.				IRQIsrVec register

15.3.14 IRQInfo Register

- Address: 0x06C…0x06F
- Reset value: 0xFFFFFFFF
- Access: Read/Write

The register IRQInfo displays the number and priority level of the currently asserted interrupt.

Table 15-16 IRQInfo Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x06C ... 0x06F	RoundRobinCtrl Bit 24	reserved	IRQPrioLevel Bits 10..8	IRQNum Bits 5..0	IRQInfo register

Bit	Name	Description
0..5	IRQNum	Displays the number of the currently asserted interrupt. This field can't be written.
6..7	reserved	
8..10	IRQPrioLevel	Displays the number of the currently asserted interrupt. This field can't be written.
11..23	reserved	
24	RoundRobinCtrl	Simultaneous interrupts on the same priority level are prioritized by the number of the interrupt source. The lower number is asserted first. If there are more than one interrupt on the same priority level the first acknowledged interrupt is internally masked until all other interrupts on this priority level are processed. Writing a '1' to Bit 24 of the IRQInfo register disables this round robin function of the AIRQ. The value of this bit is only valid if an interrupt is asserted otherwise a '1' is read. This setting is valid for standard and fast interrupts requests.
25..31	reserved	

15.3.15 FIQIsrVec Register

- Address: 0x070…0x073
- Reset value: 0xFFFFFFFF
- Access: Read-only

For each of the 64 interrupt sources the user can store an individual start address of the Interrupt Service Routine in the registers IsrVec0 to IsrVec63. If an interrupt is asserted and configured as FIQ the address inside the corresponding IsrVecN register is displayed in register FIQIsrVec and can be used call the Interrupt Service Routine for the interrupt source directly.

Table 15-17 FIQIsrVec Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x070 ... 0x073	32Bit start address of the ISR of the asserted interrupt.				FIQIsrVec register

15.3.16 FIQInfo Register

- Address: 0x074…0x077
- Reset value: 0xFFFFFFFF
- Access: Read/Write

The register FIQInfo displays the number and priority level of the currently asserted fast interrupt.

Table 15-18 FIQInfo Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x074 ... 0x077	reserved		FIQPrioLevel Bits 10..8	FIQNum Bits 5..0	FIQInfo register

Bit	Name	Description
0..5	FIQNum	Displays the number of the currently asserted interrupt. This field can't be written.
6..7	reserved	
8..10	FIQPrioLevel	Displays the number of the currently asserted interrupt. This field can't be written.
11..23	reserved	
24	RoundRobinCtrl	Simultaneous interrupts on the same priority level are prioritized by the number of the interrupt source. The lower number is asserted first. If there are more than one interrupt on the same priority level the first acknowledged interrupt is internally masked until all other interrupts on this priority level are processed. Writing a '1' to Bit 24 of the IRQInfo register disables this round robin function of the AIRQ. The value of this bit is only valid if an interrupt is asserted otherwise a '1' is read. This setting is valid for standard and fast interrupts requests.
25..31	reserved	

Note that the RoundRobinCtrl bit mentioned inside the IRQInfo register controls the behavior for standard and fast interrupt requests.

15.3.17 IntIsFIQ_LO Register

- Address: 0x078…0x07B
- Reset value: 0x0
- Access: Read/Write

Each of the 64 interrupt sources can be configured as standard or fast interrupt. By default all 64 interrupt sources are setup to assert a standard interrupt request. To configure one of the interrupt sources 31 to 0 to assert a fast interrupt request the user must set the corresponding bit in the IntIsFIQ_LO register to ‘1’.

Table 15-19 IntIsFIQ_LO Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x078 ... 0x07B	Configure interrupt source 31..0 as fast interrupt by setting the corresponding bit to ‘1’.				IntIsFIQ_LO register

15.3.18 IntIsFIQ_HI Register

- Address: 0x07C…0x07F
- Reset value: 0x0
- Access: Read/Write

Each of the 64 interrupt sources can be configured as standard or fast interrupt. By default all 64 interrupt sources are setup to assert a standard interrupt request. To configure one of the interrupt sources 63 to 32 to assert a fast interrupt request the user must set the corresponding bit in the IntIsFIQ_HI register to ‘1’.

Table 15-20 IntIsFIQ_HI Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x07C ... 0x07F	Configure interrupt source 63..32 as fast interrupt by setting the corresponding bit to ‘1’.				IntIsFIQ_HI register

15.3.19 IsrVecN Register

- Address: 0x080+N*4…0x083+N*4 (N = 0..63)
- Reset value: 0x0
- Access: Read/Write

For each of the 64 interrupt sources an individual start address of the corresponding interrupt service routine can be stored in the registers IsrVecN with N from 0 to 63. The user can read the corresponding ISR start address from register IRQIsrVec if a standard interrupt is asserted. For fast interrupts the ISR start address is displayed in register FIQIsrVec.
The following table shows the register IsrVec0.

Table 15-21 IsrVec0 Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x080 ... 0x083	ISR start address of interrupt source 0.				IsrVec0 register

15.3.20 REQ_Status_LO Register

- Address: 0x180…0x183
- Reset value: 0x0
- Access: Read-only

This register displays the status of the request line of the interrupt sources 31 to 0.
If a source requests an interrupt the corresponding bit is set to ‘1’ no matter if the interrupt is disabled or masked.

Table 15-22 REQ_Status_LO Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x180 ... 0x183	Status of the interrupt sources 31..0				REQ_Status_LO register

15.3.21 REQ_Status_HI Register

- Address: 0x184…0x187
- Reset value: 0x0
- Access: Read-only

This register displays the status of the request line of the interrupt sources 63 to 32.
If a source requests an interrupt the corresponding bit is set to ‘1’ no matter if the interrupt is disabled or masked.

Table 15-23 REQ_Status_HI Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x184 ... 0x187	reserved	Status of the interrupt sources 63..32			REQ_Status_HI register

15.3.22 EXT_INT Configuration Register

- Address: 0x188…0x191
- Reset value for PBUS Alarm line: 0x0110
- Reset value for AEI_INT0 and AEI_INT1: 0x0112
- Access: Read/Write

The three external interrupt lines into the ANTAIOS can be programmed to assert an interrupt dependent on the level or the edge of the input signal. The configuration of the external interrupt sources is done with a separate 16Bit register for each external interrupt. The offset of the configuration registers and its corresponding external interrupt is listed in the following table:

Table 15-24 EXT_INT Configuration Register

Address	Configuration register for
0x188	AEI_INT1 (PortD_44)
0x18A	PBUS Alarm line
0x190	AEI_INT0 (PortD_43)

Table 15-25 EXT_INTx_conf

Bit	Name	Description	
0	H	Level triggered ‘high’	Level/Edge configuration must be ONE-HOT encoded!
1	L	Level triggered ‘low’	
2	R	Edge triggered ‘rising’	
3	F	Edge triggered ‘falling’	
4..7	filt_cycles down	Defines the time the IRQ Input should be filtered first. User can program different filter times for falling and rising events. Reset value: 1 Range: 1..8 $filtertime = 10.4\,ns \cdot 2^{filt\_cycles}$ Spikes shorter than the filtertime will not assert an interrupt.	
8..11	filt_cycles up		
12..15	reserved		

15.3.23 EXT_INT_Status Register

- Address: 0x198
- Reset value: depends on external signals
- Access: Read-only

This register displays the status of the three external interrupt sources after the programmed filtering and level configuration. If an external interrupt source has the status '1' it will assert an interrupt if it is enabled and not masked. Note that the bits 1, 3 and 5 of this register are always read as zero.

Table 15-26 EXT_INT_Status Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x198 ... 0x19B	Status of the external interrupt sources				EXT_INT_Status register

Bit	Name	Description
0	AEI_INT1	AEI master: 2nd IRQ line
1	PBUS Alarm line	
2..3	reserved	
4	AEI_INT0	AEI master: 1st IRQ line
5..31	reserved	

15.3.24 µs_Prescaler Register

- Address: 0x19C…0x19D
- Reset value: 0x60
- Access: Read/Write

The advanced IRQ controller gives the user the possibility to automatically mask out asserted interrupts for a defined time individually for all 64 interrupt sources. This feature will be called interrupt throttling on the following pages and can be used to control the frequency of occurrence of interrupts. The throttle time can be set in the registers ThrottleTimeInt0 … ThrottleTimeInt63 and is given in microseconds. Therefore the µs_Prescaler register must be set accordingly to the AMBA AHB speed of the system. The reset value of 96 for this register is the correct setting for an ANTAIOS clocked by a 32MHz crystal oscillator, which results in a 96MHz module frequency for the AIRQC.

Table 15-27 µs_Prescaler Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x19C ... 0x19F	reserved		µs_Prescaler		µs_Prescaler register

15.3.25 ThrottleMask_LO Register

- Address: 0x1A0…0x1A3
- Reset value: 0x0
- Access: Read-only

The ThrottleMask_LO register shows if one or more of the interrupt sources 31 to 0 are currently “throttled”. To enable the throttling algorithm the user must program the corresponding ThrottleTimeInt register for the desired interrupt source to a value greater than zero. The throttle mask for this interrupt source will be set and the throttle timer will start to decrease as soon as the interrupt is acknowledged inside the AIRQC. If the throttle timer reaches zero the throttle mask for this interrupt will be cleared automatically.

Table 15-28 ThrottleMask_LO Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x1A0 ... 0x1A3	If the throttle mask for one of the interrupt sources 31..0 is active the corresponding bit is set to ‘1’.				ThrottleMask_LO register

15.3.26 ThrottleMask_HI Register

- Address: 0x1A4…0x1A7
- Reset value: 0x0
- Access: Read-only

The ThrottleMask_HI register shows if one or more of the interrupt sources 63 to 32 are currently “throttled”. To enable the throttling algorithm the user must program the corresponding ThrottleTimeInt register for the desired interrupt source to a value greater than zero. The throttle mask for this interrupt source will be set and the throttle timer will start to decrease as soon as the interrupt is acknowledged inside the AIRQC. If the throttle timer reaches zero the throttle mask for this interrupt will be cleared automatically.

Table 15-29 ThrottleMask_HI Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x1A4 ... 0x1A7	If the throttle mask for one of the interrupt sources 63..32 is active the corresponding bit is set to ‘1’.				ThrottleMask_HI register

15.3.27 ThrottleTimeIntN Register

- Address: 0x1A8+N*4…0x1AB+N*4 (N = 0..63)
- Reset value: 0x0
- Access: Write-only

The advanced IRQ controller gives the user the possibility to automatically mask out asserted interrupts for a defined time individually for all 64 interrupt sources. This interrupt throttling feature and can be used to control the frequency of occurrence of interrupts. For each of the 64 interrupt sources an individual throttle time in microseconds can be stored in the registers ThrottleTimeIntN with N from 0 to 63. To enable the throttling algorithm the user must program the corresponding ThrottleTimeInt register for the desired interrupt source to a value greater than zero. The throttle mask for this interrupt source will be set and the throttle timer will start to decrease as soon as the interrupt is acknowledged inside the AIRQC. If the throttle timer reaches zero the throttle mask for this interrupt will be cleared automatically. An active throttle masked can be cleared by writing zero to the corresponding ThrottleTimeIntN register. The following table shows the register ThrottleTimeInt0.

Table 15-30 ThrottleTimeInt0 Register

Address	Byte Position				Designation
	Byte 3	Byte 2	Byte 1	Byte 0	
0x1A8 ... 0x1AB	reserved		17Bit register for a maximum of 131071µs throttle time.		ThrottleTimeInt0 register

NOTE: **IRQ 62..53 and 26..24 are triggered by modules which are connected to ARM via asynchronous AHB bridges. This may lead to asynchronicity in the interrupt processing. In order to avoid inconsistencies caused by a retriggering of already cleared interrupts, one of the following actions should be followed.**

One option would be to read back the interrupt request register after the interrupt has been confirmed in the module. As the processing of these commands is always done sequential, this ensures that IRQ is reset before it is read. If the interrupt is no longer active, it can also be confirmed in the Advanced IRQ controller.

Another way to avoid the problem is an appropriate structure of the ISR and DSR routines. Processing should be start in the ISR by masking and acknowledging the interrupt in the module. The mask in the Advanced IRQ controller should be cleared at the very end of the processing inside the DSR, when it can be assumed that the interrupt is no longer pending.

16 DMA Controller

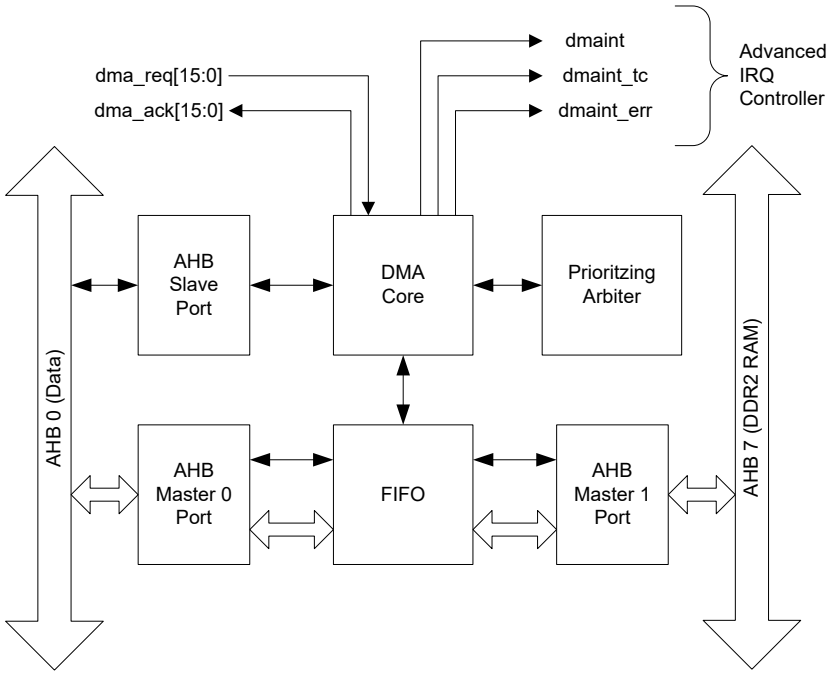
Offset: 0xD0000000

The Direct Memory Access controller is designed to enhance the system performance and reduce the interrupt generation for processor. The system efficiency is improved by employing the high-speed data transfers between the system and device. The DMA controller provides eight configurable channels for the memory-to-memory, memory-to-peripheral, peripheral-to-peripheral and peripheral-to-memory transfers with shared buffer.

- supports eight configurable DMA channels
- supports chain transfer
- supports four arbitration priority levels with round-robin scheme
- supports hardware handshake
- supports transactions with 8-bit, 16-bit and 32-bit data widths
- supports only little-endian transfer

16.1 Overview

Figure 16-1 Block diagram of DMA Controller



AHB Bus 0 Slave: Connected to the main internal Bus System, the ARM Processor can configure the DMA controller via this way.

AHB Bus 0 Master: The DMA controller can access all devices connected to this bus, like the ARM processor can do.
(address mapping as described in Table 3-1)

AHB Bus 7 Master: The DMA controller can access DDR2-RAM.
(same address mapping)

DMA FIFO size of the controller is 32 x 32 bit per channel.

16.2 Programming Model

The DMA controller consists of up to eight DMA channels, one DMA engine and one channel prioritizing arbiter.

16.2.1 General Description

Table 16-1 lists the hardware DMA requests (dma_req) for hardware handshake mode.

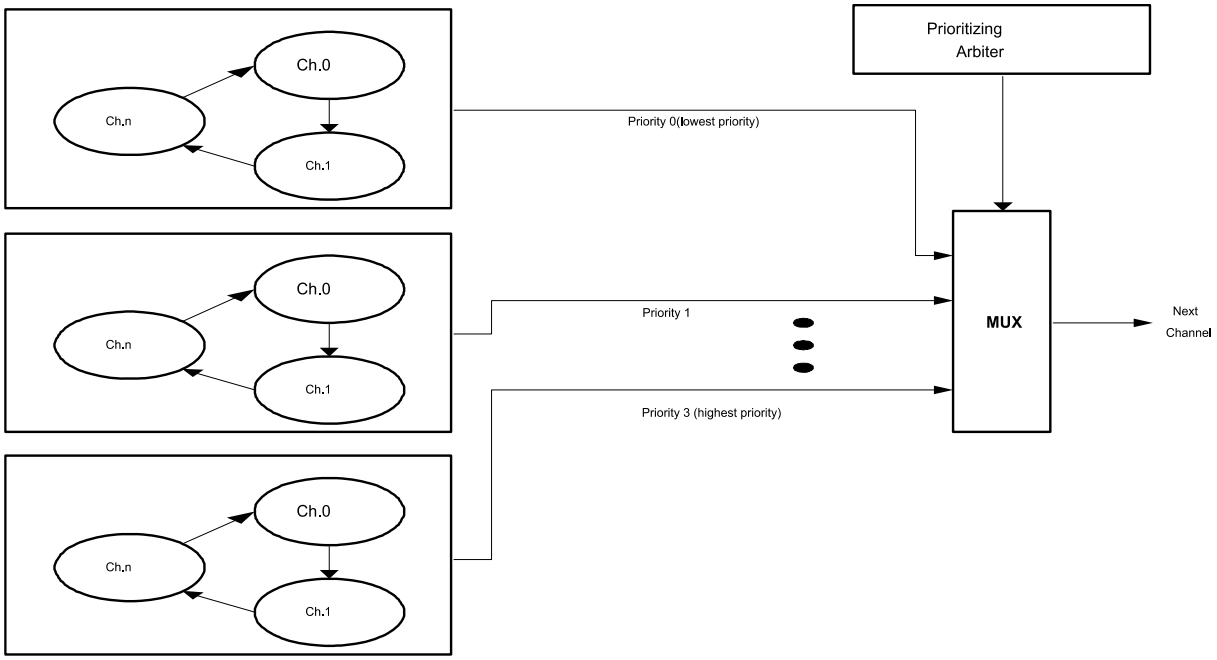
Table 16-1 DMA Hardware Handshake Requests

DMA_Request	Name	Description
15..7		reserved
6	usb_dma_req	USB Req
5	sdc_dma_req	SD/MMC Card Controller
4		reserved
3	qspi_dma	QuadSPI
2	spi_txdmareq	SPI Transmitter
1	spi_rxdmareq	SPI Receiver
0	nand_req	From NAND Flash Controller

16.2.2 Prioritizing Arbiter

The DMA controller uses the 4-group priority and the round-robin scheme to select which channel to serve. Arbitration is based on the priority level of the channels. If channels have the same priority level, the arbitration will then be based on the round-robin scheme. Each channel has a 2-bit priority value. A value of 3 indicates the highest priority level and a value of 0 indicates the lowest priority level. The arbitration scheme is shown in Figure 16-2.

Figure 16-2 Arbitration Scheme

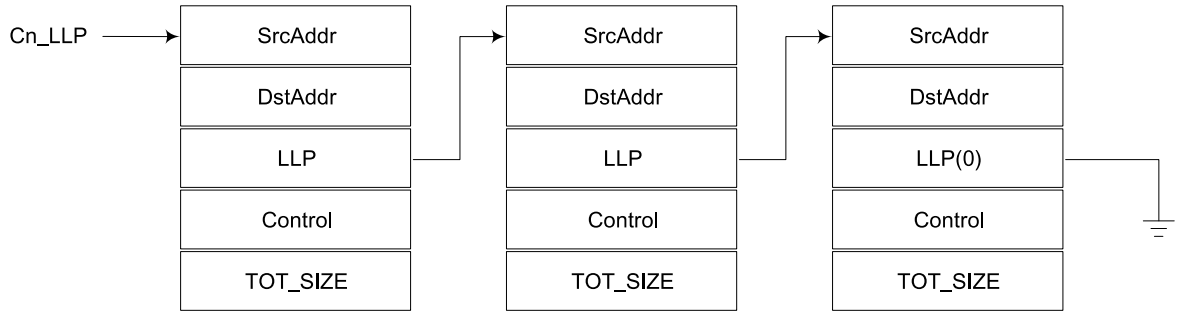


16.2.3 Chain Transfer

The DMA controller provides the chain transfer function. Multiple blocks can be transferred consecutively without CPU processing after each block transfer. Before starting the chain transfer, a linked list structure (Table 16-2) must be built in system memory for each channel. Then the linked list descriptor pointer register (Cn_LLP) for this channel must be programmed to point to the head of this linked list structure.

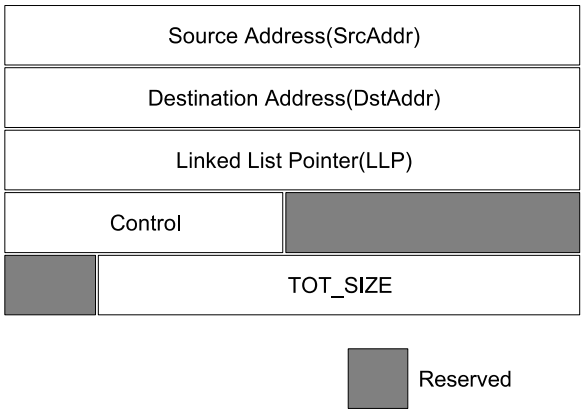
For channel *n*, after finishing the first block data transfer, the DMA engine will verify whether or not the linked list descriptor pointer address (bits [31:2]) of the linked list descriptor pointer register (Cn_LLP) is zero. If the value is not zero, the chain transfer function is enabled for this channel. In this case, the DMA controller will fetch the first channel linked list descriptor from the memory attached to the AHB Bus 0 or AHB Bus 7 depending on bit 0 of the linked list descriptor pointer register (Cn_LLP). After finishing the data transfer of the first channel linked list descriptor, the DMA engine will continue the data transfer of the next until the end of the linked list structure. Figure 16-3 shows a linked list structure with three linked list descriptors. Therefore, there can be four consecutive blocks to be transferred without CPU processing. The first block is specified in Cn_CSR, Cn_SrcAddr, Cn_DstrAddr and Cn_SIZE. The second block is specified in the first linked list descriptor. The third block is specified in the second linked list descriptor. The fourth block is specified in the third linked list descriptor. Please note that for the first block data transfer, the source, destination, total size and control are based on Cn_SrcAddr, Cn_DstAddr, Cn_Size, and Cn_CSR. For transferring the second and later blocks, the source, destination, total size, and control are based on the linked list descriptor.

Figure 16-3 Linked List Structure of Chain Transfer Operation



To configure the DMA controller for the chain transfer, please refer to subsection 16.4 Programming Guidelines for the detailed descriptions.

Figure 16-4 Linked List Descriptor



Each time the DMA controller fetches the linked list descriptor (32-bit word-aligned), SrcAddr is copied to the Cn_SrcAddr register, DstAddr is copied to the Cn_DstAddr register, LLP is copied to the Cn_LLP register, Total Transfer Size is copied to the TOT_SIZE field in the Cn_SIZE register, and Control is copied to the respective bits in the Cn_CSR register.

Table 16-2 Address Map for Linked List Descriptor (Base Address: Cn_LPP[31:2])

Address Offset	Name	Width (Bit)	Description
+0x0	SrcAddr	32	Source address
+0x4	DstAddr	32	Destination address
+0x8	LLP	32	Linked list pointer
+0xC	Control	32	Control and total transfer size (Table 16-3)
+0x10	Total size	32	Total transfer size (Table entries are same as in Cn_CSR, but they are arranged differently. Table 16-4)

Table 16-3 Control Field Definition in Linked List Descriptor

Bit	Name	Description
31..29	DMA_FF_TH	DMA FIFO threshold value
28	TC_MSK	Channel terminal count status mask
27..25	SRC_WIDTH	Source transfer width
24..22	DST_WIDTH	Destination transfer width
21..20	SRCAD_CTL	Source address control
19..18	DSTAD_CTL	Destination address control
17	SRC_SEL	Source selection
16	DST_SEL	Destination selection
15..0	reserved	-

Table entries are same as in Cn_CSR, but they are arranged differently.

Table 16-4 Total Transfer Size Definition in Linked List Descriptor

Bit	Name	Description
31..22	reserved	-
21..0	TOT_SIZE	Total transfer size

Table entries are same as in Cn_SIZE.

For each channel, when the chain transfer is enabled, after finishing transfer of one block, the DMA engine will assert the interrupt **dmaint** (interrupt number 13 at Advanced IRQ Controller) to inform CPU that the data transfer is complete. Then CPU should de-assert the interrupt in time by writing '1' to the respective bit of the Terminal Count Interrupt Status Clear Register before the data transfer of the next block is done. However, in some cases, CPU will not de-assert the interrupt in time before the data transfer of the next block is done. In such cases, CPU does not know whether the data transfer of two blocks is done. This will cause problem to CPU during the chain transfer operation.

For example, Figure 16-5 shows a chain transfer operation composed of transactions 1, 2, and 3, i.e., two linked list descriptors are fetched. At time t_1, transaction 1 is finished and the DMA sets INT_TC[n] to assert dmaint_tc and dmaint. At time t_2, the CPU reads INT_TC to know the interrupt source in the interrupt service routine. At time t_3, transaction 2 is finished and then the DMA sets INT_TC[n] again. At time t_4, the interrupt service routine for time t_2 clears INT_TC[n] by writing INT_TC_CLR[n]. At time t_5, transaction 3 is finished, and CPU does not know that transaction 2 had been finished (At t_4, the software clears INT_TC[n] of transactions 1 and 2). This causes an interrupt conflict. To avoid this conflict situation, the DMA is designed to provide the LLP counter to record the transaction number when the chain transfer is enabled.

The LLP counter (denoted as LLP_CNT, bit [19:16] of Channel Configuration Register) operation is shown in Figure 16-6. When setting Cn_CSR[CH_EN]=1, Cn_CFG[LLP_CNT] is reset to '0'. When each transaction is finished, LLP_CNT is increased by '1'. Please note that when the last transaction is finished, LLP_CNT is also increased by '1'. When an error or abort condition happens, the chain transfer is stopped and CH_EN is cleared to '0', and then LLP_CNT will also be increased by '1'.

Figure 16-5 Interrupt Conflict during Chain Transfer Operation

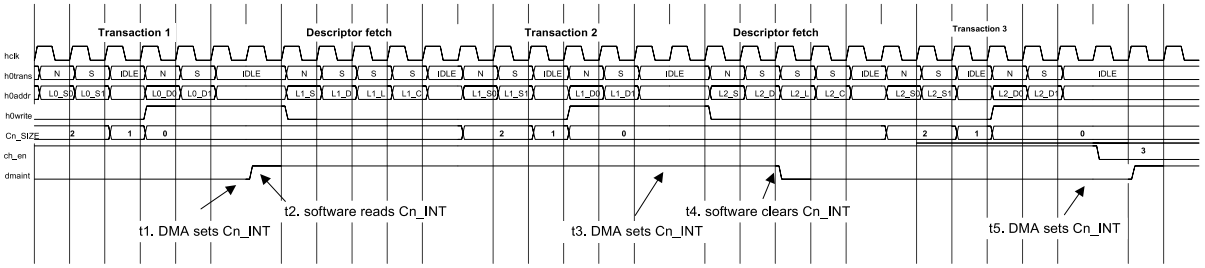
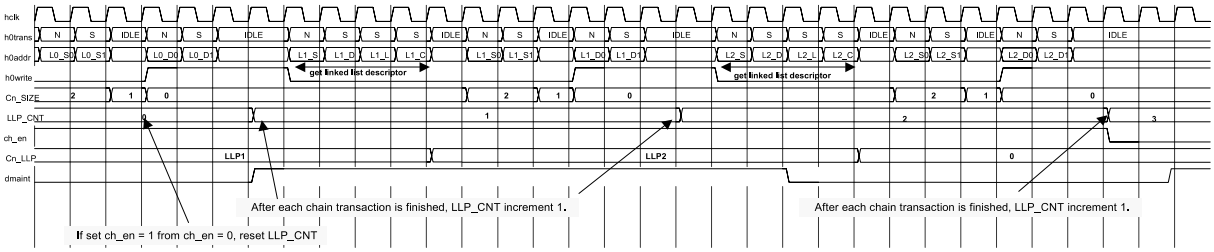


Figure 16-6 LLP_CNT Operation



16.2.4 DMA Hardware Handshake Mode

Each channel of the DMA controller must be programmed into either the DMA hardware handshake mode or DMA normal mode. The DMA hardware handshake mode will be explained in this section and DMA normal mode will be explained in the next section.

The DMA hardware handshake mode for a dedicated channel can be enabled by setting Cn_CSR[MODE]=1. For example, the DMA hardware handshake mode of channel 0 can be enabled by setting bit 7 of C0_CSR. If the DMA hardware handshake mode of channel 0 is enabled, after channel 0 wins the arbitration, the DMA controller will wait for the external DMA request to be asserted before starting the DMA transfer. Each time when the DMA request is asserted, the controller transfers units equal to SRC_BURST_SIZE (SRC_BURST_SIZE is set according to Cn_CSR[SRC_SIZE]). When the SRC_BURST_SIZE transfer is completed, the DMA controller will assert acknowledge and then re-arbitrate among all DMA requests. After detecting the assertion of acknowledge, the external device should de-assert the DMA request to let the DMA controller de-assert acknowledge. Please note that DMA request cannot be asserted again until the de-assertion is acknowledged. After the TOT_SIZE transfers are complete, the DMA controller will assert TC[0], INT_TC[0] and both **dmaint_tc** (interrupt number 47 at Advanced IRQ Controller**)** and **dmaint** interrupts (if not masked). Figure 16-7 illustrates an example of the DMA transfer between two AHB interfaces in the hardware handshake mode. Cn_CSR[SRC_WIDTH] = Cn_CSR[DST_WIDTH]

Figure 16-8 shows the hardware handshake protocol. Please note that the DMA controller will de-assert dma_ack when dma_req is de-asserted.

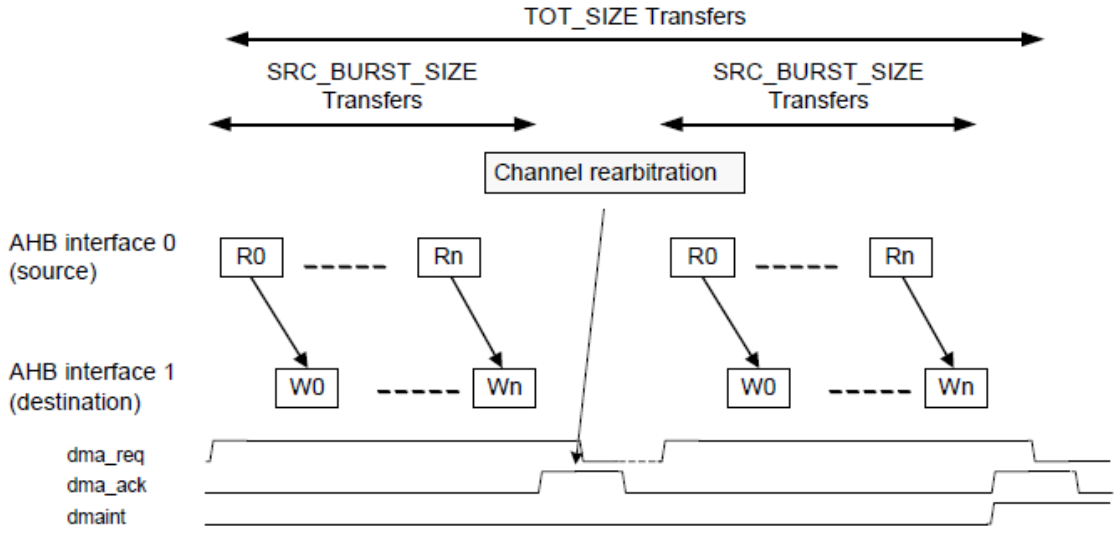
During the transfer, if the source or destination slave returns an ERROR response, the DMA will set ERR_ABT[n] and terminate the DMA transfer at once. (Please note that if the source and destination slaves are on different AHB buses and the other destination or source slave returns a RETRY response at the same time, the DMA controller will finish the RETRY cycle before setting the ERR bit). Then, if Cn_CFG[INT_ERR_MSK] is not set, the DMA controller will assert **dmaint_err** (interrupt number 49 at Advanced IRQ Controller**)** and **dmaint**. In this

case, the DMA controller will not assert acknowledge (dma_ack), and the device must de-assert dma_req by itself. For example, Figure 16-9 shows that an ERROR response happens when the DMA controller reads address S6 of source slave, and that a RETRY cycle happens when the DMA controller reads address D2 of destination slave; therefore, it will force the DMA controller to complete the RETRY cycle before asserting dmaint_err and dmaint interrupt. Please note that, in this case, dma_ack will not be asserted by the DMA controller.

During a transfer, if the software sets Cn_CSR[ABT], after finishing the SRC_BURST_SIZE transfers or the TOT_SIZE transfers, the DMA controller will set ABT[n] of the Error/Abort Satus Register (ERR/ABT) and terminate the DMA transfer at once. (Please note that if the source and destination slaves return a RETRY response at the same time, the DMA controller will finish the RETRY cycle before setting the ABT bit.) Then, if Cn_CFG[INT_ABT_MSK]=0, the DMA controller will assert **dmaint**. In such a case, the DMA controller will not assert acknowledge (dma_ack), and the device must de-assert dma_req by itself. For example, Figure 16-10 shows the abort operation when this channel is active. At t_1, the internal abort state is set, and after finishing the SRC_BURST_SIZE transfer or the TOT_SIZE transfer (Time t_2) and completing the RETRY cycle (Time t_3), the DMA controller asserts the dmaint interrupt without asserting dma_ack at time t_4.

Figure 16-11 shows the abort operation when this channel is inactive. At t_1, the internal abort state is set. Because the channel is inactive, the DMA controller asserts the **dmaint** interrupt but does not assert the dma_ack at time t_2.

Figure 16-7 Example of Hardware Handshake Mode Transfer



Cn_CSR[SRC_WIDTH] = Cn_CSR[DST_WIDTH]

Figure 16-8 DMA Hardware Handshake Protocol

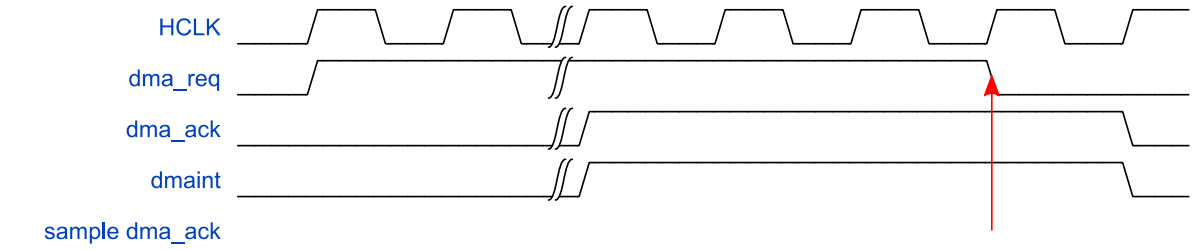


Figure 16-9 Error Response of Transfer in Hardware Handshake Mode

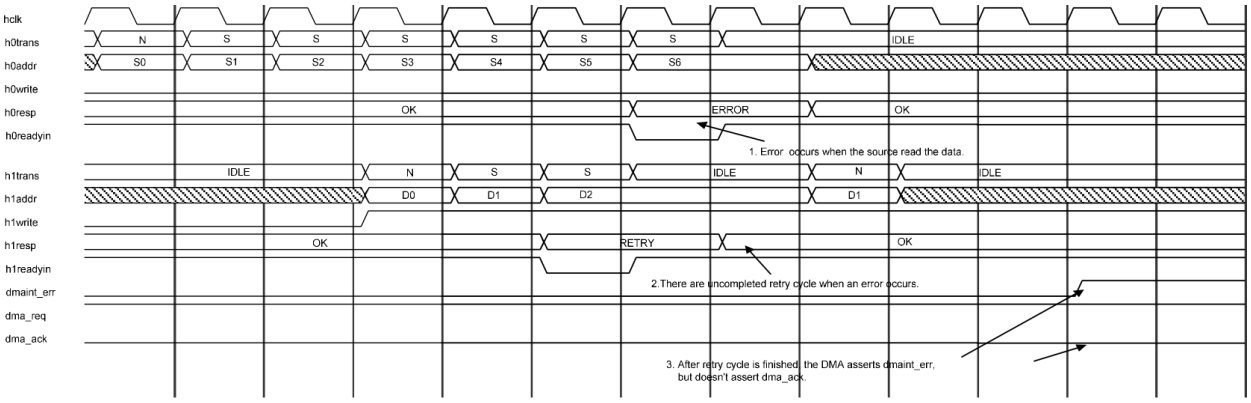


Figure 16-10 Abort Operation with Active Channel

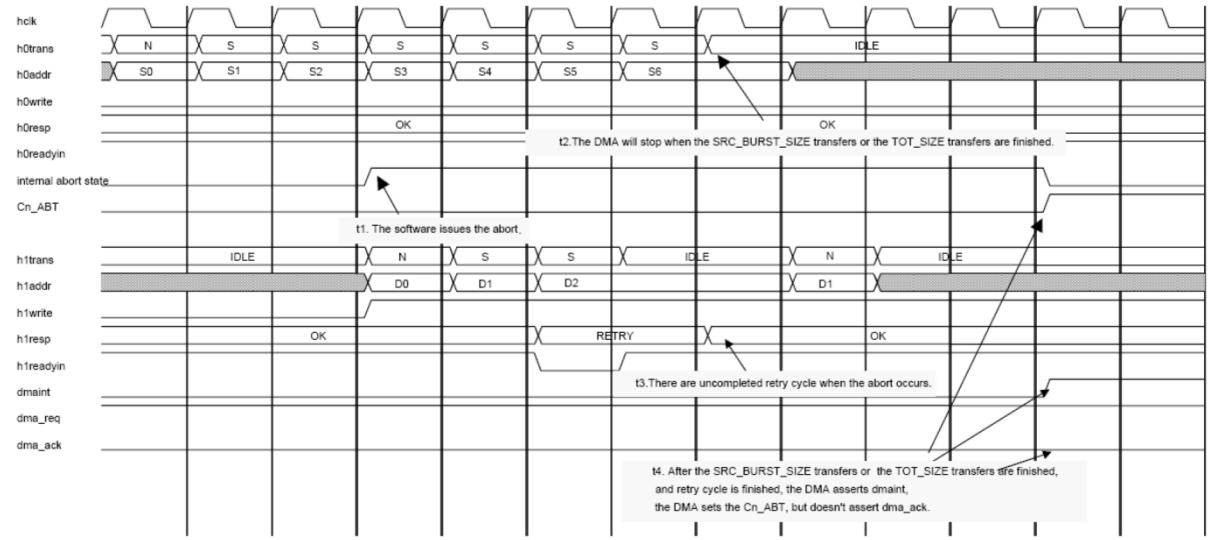
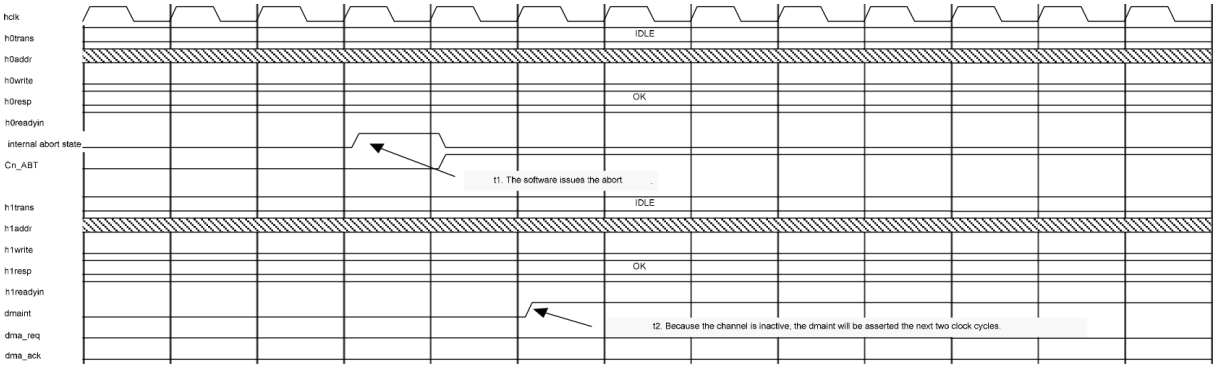


Figure 16-11 Abort Operation with Inactive Channel

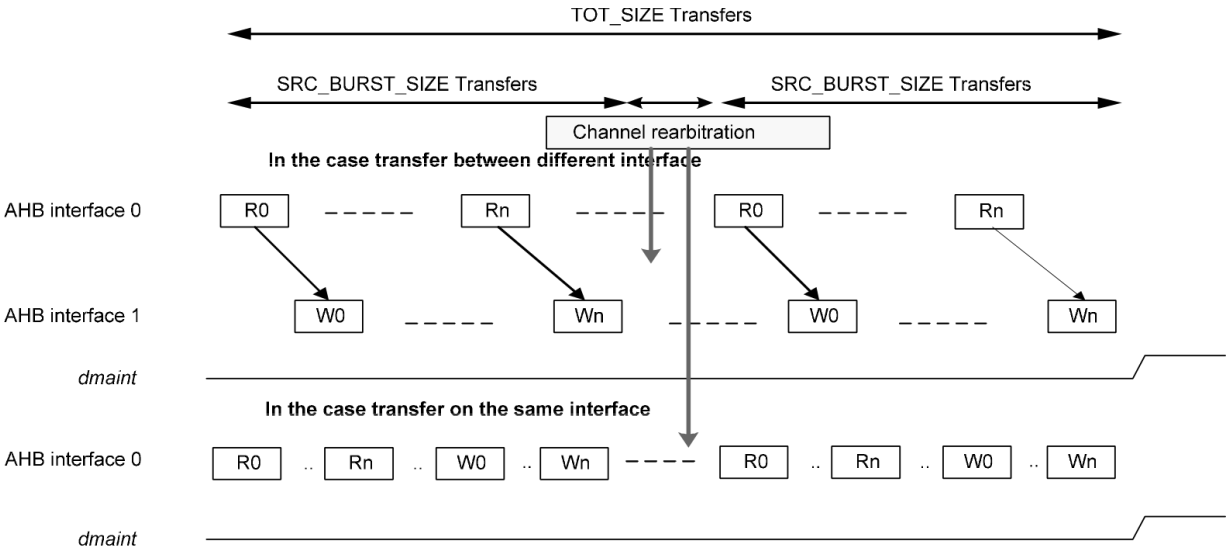


16.2.5 DMA Normal Mode

In this mode, no external DMA request is needed. The DMA controller will automatically generate a transfer request signal internally. Figure 16-12 illustrates an example of a transfer in this mode. The dmaint_tc and dmaint signals work in a similar way as described in Section 16.2.4.

Please note that dma_req[n:0] and dma_ack[n:0] are not used in this mode. The error and abort operations are the same as those in the DMA hardware handshake mode. Please refer to the section 16.2.4 for more information.

Figure 16-12 Transfer Example in DMA Normal Mode



16.3 Register Definition

16.3.1 Summary of DMA Controller Registers

Please note that all control registers should be programmed by 32-bit word accesses. Accesses with any other width may lead to unexpected errors.

This section describes all control and status registers in the DMA controller. The address column indicates the related address of the register in the hexadecimal format. The width column indicates the number of bits of the register. The type column indicates the access type of the register.

- RW: Read and write access
- RO: Read only
- WO: Write 1 to clear

In addition, all reserved bits should always be written as '0'. Reading these bits will return an undefined value.

Table 16-5 DMA Controller Registers

Address Offset	Name	Width (Bit)	Type	Reset Value	Description	
0x000	INT	8	RO	0	Interrupt status register	
0x004	INT_TC	8	RO	0	Interrupt terminal count status register	
0x008	INT_TC_CLR	8	WO	0	Interrupt terminal count clear register	
0x00C	INT_ERR/ABT	32	RO	0	Interrupt Error/Abort status register	
0x010	INT_ERR/ABT_CLR	32	WO	0	Interrupt Error/Abort clear register	
0x014	TC	8	RO	0	Terminal count status register	
0x018	ERR/ABT	32	RO	0	Error/Abort status register	
0x01C	CH_EN	8	RO	0	Channel enable status register	
0x020	CH_BUSY	8	RO	0	Channel busy status register	
0x024	CSR	8	RW	0	Main configuration status register	
0x028	SYNC	8	RW	0	Sync register	
0x100	C0_CSR	32	RW	0x0000_1200	Channel 0	Control Register
0x104	C0_CFG	32	RW	0x0000_2087		Configuration Register
0x108	C0_SrcAddr	32	RW	0		Source Address Register
0x10C	C0_DstAddr	32	RW	0		Destination Address Register
0x110	C0_LLP	32	RW	0		Linked List Descriptor Pointer
0x114	C0_SIZE	32	RW	0		Transfer Size Register
0x120	C1_CSR	32	RW	0x0000_1200	Channel 1	Control Register
0x124	C1_CFG	32	RW	0x0000_2087		Configuration Register
0x128	C1_SrcAddr	32	RW	0		Source Address Register
0x12C	C1_DstAddr	32	RW	0		Destination Address Register
0x130	C1_LLP	32	RW	0		Linked List Descriptor Pointer
0x134	C1_SIZE	32	RW	0		Transfer Size Register

Address Offset	Name	Width (Bit)	Type	Reset Value	Description	
0x140	C2_CSR	32	RW	0x0000_1200	Channel 2	Control Register
0x144	C2_CFG	32	RW	0x0000_2087		Configuration Register
0x148	C2_SrcAddr	32	RW	0		Source Address Register
0x14C	C2_DstAddr	32	RW	0		Destination Address Register
0x150	C2_LLP	32	RW	0		Linked List Descriptor Pointer
0x154	C2_SIZE	32	RW	0		Transfer Size Register
0x160	C3_CSR	32	RW	0x0000_1200	Channel 3	Control Register
0x164	C3_CFG	32	RW	0x0000_2087		Configuration Register
0x168	C3_SrcAddr	32	RW	0		Source Address Register
0x16C	C3_DstAddr	32	RW	0		Destination Address Register
0x170	C3_LLP	32	RW	0		Linked List Descriptor Pointer
0x174	C3_SIZE	32	RW	0		Transfer Size Register
0x180	C4_CSR	32	RW	0x0000_1200	Channel 4	Control Register
0x184	C4_CFG	32	RW	0x0000_2087		Configuration Register
0x188	C4_SrcAddr	32	RW	0		Source Address Register
0x18C	C4_DstAddr	32	RW	0		Destination Address Register
0x190	C4_LLP	32	RW	0		Linked List Descriptor Pointer
0x194	C4_SIZE	32	RW	0		Transfer Size Register
0x1A0	C5_CSR	32	RW	0x0000_1200	Channel 5	Control Register
0x1A4	C5_CFG	32	RW	0x0000_2087		Configuration Register
0x1A8	C5_SrcAddr	32	RW	0		Source Address Register
0x1AC	C5_DstAddr	32	RW	0		Destination Address Register
0x1B0	C5_LLP	32	RW	0		Linked List Descriptor Pointer
0x1B4	C5_SIZE	32	RW	0		Transfer Size Register
0x1C0	C6_CSR	32	RW	0x0000_1200	Channel 6	Control Register
0x1C4	C6_CFG	32	RW	0x0000_2087		Configuration Register
0x1C8	C6_SrcAddr	32	RW	0		Source Address Register
0x1CC	C6_DstAddr	32	RW	0		Destination Address Register
0x1D0	C6_LLP	32	RW	0		Linked List Descriptor Pointer
0x1D4	C6_SIZE	32	RW	0		Transfer Size Register
0x1E0	C7_CSR	32	RW	0x0000_1200	Channel 7	Control Register
0x1E4	C7_CFG	32	RW	0x0000_2087		Configuration Register
0x1E8	C7_SrcAddr	32	RW	0		Source Address Register
0x1EC	C7_DstAddr	32	RW	0		Destination Address Register
0x1F0	C7_LLP	32	RW	0		Linked List Descriptor Pointer
0x1F4	C7_SIZE	32	RW	0		Transfer Size Register

16.3.2 Interrupt Status Register

Abbr.: INT *Offset: 0x000*

This register is used to keep the result of the following equation:

$$INT[n] = INT\_ABT[n] \mid INT\_ERR[n] \mid INT\_TC[n]$$

Where n is the channel number, and '|' is the logic OR.

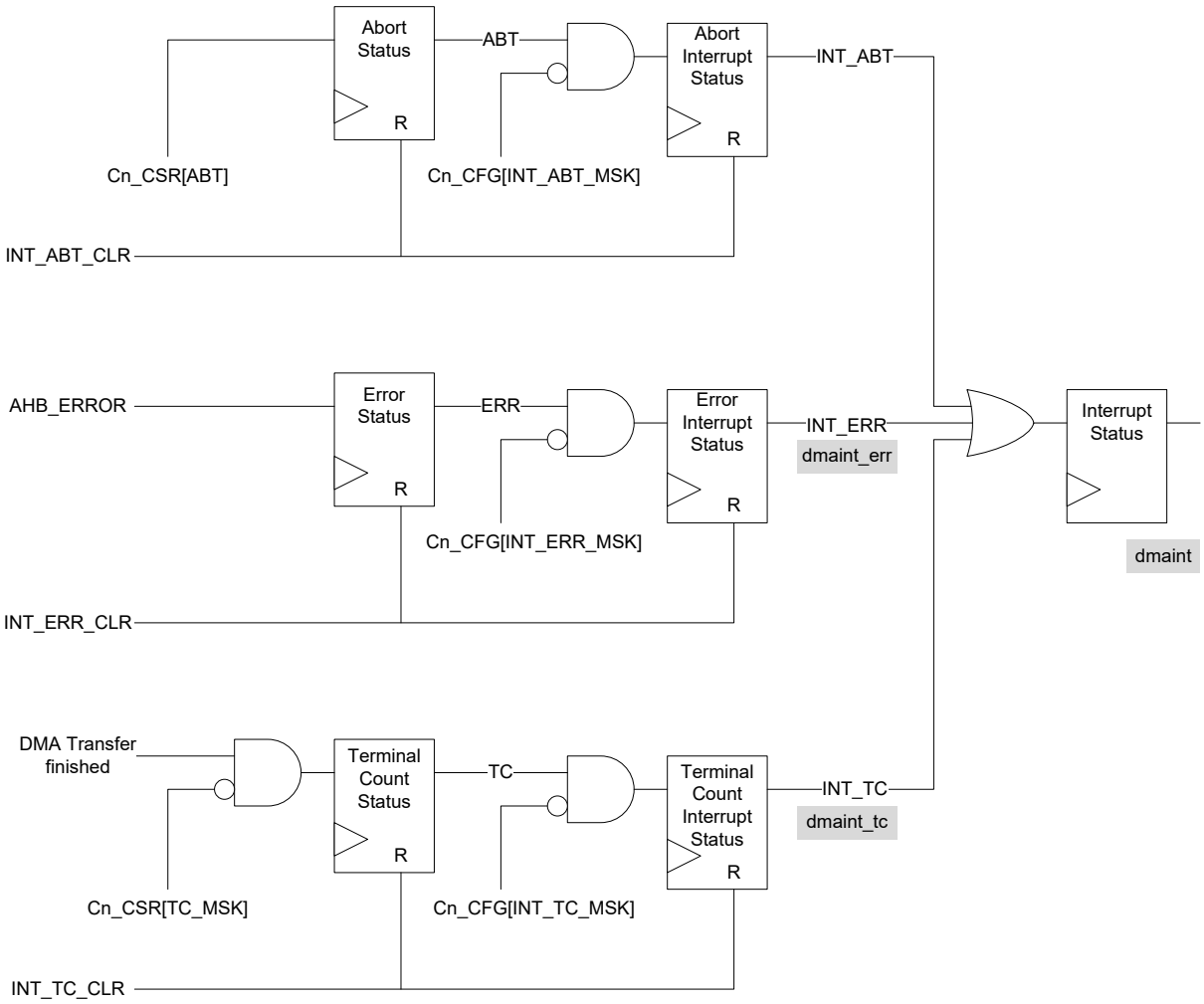
INT_ABT[n] means bit 23 to bit 16 of the error/abort interrupt status register (INT_ERR/ABT).
INT_ERR[n] means bit 7 to bit 0 of the error/abort interrupt status register (INT_ERR/ABT).
INT_TC[n] means bit 7 to bit 0 of the terminal count interrupt status register (INT_TC).
Signal **dmaint** is asserted by the DMA controller if at least one bit of this register is set; on the other hand, dmaint is de-asserted by the DMA controller if no bit of this register is set.

If the maximum channel number of DMA is configured as n, bit n to bit 7 of this register will be reserved.

Table 16-6 INT Register

Bit	Name	Type	Reset	Description
7	INT[7]	RO	0	The result of INT_ABT[7] | INT_ERR[7] | INT_TC[7] 0: Channel 7 has no pending interrupt 1: Channel 7 has a pending interrupt.
6	INT[6]	RO	0	The result of INT_ABT[6] | INT_ERR[6] | INT_TC[6] 0: Channel 6 has no pending interrupt 1: Channel 6 has a pending interrupt.
5	INT[5]	RO	0	The result of INT_ABT[5] | INT_ERR[5] | INT_TC[5] 0: Channel 5 has no pending interrupt. 1: Channel 5 has a pending interrupt.
4	INT[4]	RO	0	The result of INT_ABT[4] | INT_ERR[4] | INT_TC[4] 0: Channel 4 has no pending interrupt. 1: Channel 4 has a pending interrupt.
3	INT[3]	RO	0	The result of INT_ABT[3] | INT_ERR[3] | INT_TC[3] 0: Channel 3 has no pending interrupt. 1: Channel 3 has a pending interrupt.
2	INT[2]	RO	0	The result of INT_ABT[2] | INT_ERR[2] | INT_TC[2] 0: Channel 2 has no pending interrupt. 1: Channel 2 has a pending interrupt.
1	INT[1]	RO	0	The result of INT_ABT[1] | INT_ERR[1] | INT_TC[1] 0: Channel 1 has no pending interrupt. 1: Channel 1 has a pending interrupt.
0	INT[0]	RO	0	The result of INT_ABT[0] | INT_ERR[0] | INT_TC[0] 0: Channel 0 has no pending interrupt. 1: Channel 0 has a pending interrupt.

Figure 16-13 Overview of Interrupt Functionality



Advanced IRQ Controller - Interrupt Sources	
13: DMA Controller Interrupt	→ dmaint
47: DMA Controller Terminal Count Interrut	→ dmaint_tc
49: DMA Controller Error Interrupt	→ dmaint_err

16.3.3 Terminal Coun Interrupt Status Register

Abbr.: INT_TC *Offset: 0x004*

This register shows the status of the DMA terminal count interrupts after masking. The mask bit of these interrupts is bit 0 (INT_TC_MSK) of the Channel Configuration register (Cn_CFG). If this mask bit is set, the content of INT_TC[n] of this register is always '0' regardless of the presence of a pending DMA terminal count interrupt. Please refer to Section 16.2.4 and Section 16.2.5 for the detailed information on the generation of these interrupts. **dmaint_tc** is asserted by the DMA controller if at least one bit of this register is set; on the other hand, dmaint_tc is deasserted by the DMA controller if no bit of this register is set.

If the maximum channel number of DMA is configured as n, bit n to bit 7 of this register will be reserved.

Table 16-7 INT_TC Register

Bit	Name	Type	Reset	Description
7	INT_TC[7]	RO	0	Status of the DMA terminal count interrupts after masking 0: Channel 7 has no pending interrupt 1: Channel 7 has a pending interrupt.
6	INT_TC[6]	RO	0	Status of the DMA terminal count interrupts after masking 0: Channel 6 has no pending interrupt 1: Channel 6 has a pending interrupt.
5	INT_TC[5]	RO	0	Status of the DMA terminal count interrupts after masking 0: Channel 5 has no pending interrupt. 1: Channel 5 has a pending interrupt.
4	INT_TC[4]	RO	0	Status of the DMA terminal count interrupts after masking 0: Channel 4 has no pending interrupt. 1: Channel 4 has a pending interrupt.
3	INT_TC[3]	RO	0	Status of the DMA terminal count interrupts after masking 0: Channel 3 has no pending interrupt. 1: Channel 3 has a pending interrupt.
2	INT_TC[2]	RO	0	Status of the DMA terminal count interrupts after masking 0: Channel 2 has no pending interrupt. 1: Channel 2 has a pending interrupt.
1	INT_TC[1]	RO	0	Status of the DMA terminal count interrupts after masking 0: Channel 1 has no pending interrupt. 1: Channel 1 has a pending interrupt.
0	INT_TC[0]	RO	0	Status of the DMA terminal count interrupts after masking 0: Channel 0 has no pending interrupt. 1: Channel 0 has a pending interrupt.

16.3.4 Terminal Count Interrupt Status Clear Register

Abbr.: INT_TC_CLR *Offset: 0x008*

Writing '1' to bit n of this register clears both INT_TC[n] and TC[n]. INT_TC[n] means bit 7: to bit 0 of the terminal count interrupt status register (INT_TC) and TC[n] means bit 7 to bit 0 of the terminal count status register (TC).

If the maximum channel number of DMA is configured as n, bit n to bit 7 of this register will be reserved.

Table 16-8 INT_TC_CLR Register

Bit	Name	Type	Reset	Description
7	INT_TC_CLR[7]	WO	0	Write '1' to clear the INT_TC[7] and TC[7] statuses
6	INT_TC_CLR[6]	WO	0	Write '1' to clear the INT_TC[6] and TC[6] statuses
5	INT_TC_CLR[5]	WO	0	Write '1' to clear the INT_TC[5] and TC[5] statuses
4	INT_TC_CLR[4]	WO	0	Write '1' to clear the INT_TC[4] and TC[4] statuses
3	INT_TC_CLR[3]	WO	0	Write '1' to clear the INT_TC[3] and TC[3] statuses
2	INT_TC_CLR[2]	WO	0	Write '1' to clear the INT_TC[2] and TC[2] statuses
1	INT_TC_CLR[1]	WO	0	Write '1' to clear the INT_TC[1] and TC[1] statuses
0	INT_TC_CLR[0]	WO	0	Write '1' to clear the INT_TC[0] and TC[0] statuses

16.3.5 Error/Abort Interrupt Status Register

Abbr.: INT_ERR/ABT *Offset:* 0x00C

INT_ERR is the status of the DMA error interrupts after masking. The mask bit of these interrupts is bit 1 (INT_ERR_MSK) of the channel configuration register (Cn_CFG). If this mask bit is set, the content of INT_ERR[n] of this register is always '0' regardless of the presence of a pending DMA error interrupt. If an AHB ERROR response happens during the DMA transfer, the DMA controller will stop the current DMA transfer and will set ERR[n] (bit 7 to bit 0 of the error/abort status register (ERR/ABT)) to '1'. Then, if INT_ERR_MSK is not set, the DMA controller will set INT_ERR[n] to '1' and assert the **dmaint_err** and dmaint interrupts. Please note that dmaint_err is asserted by the DMA controller if at least one bit of the INT_ERR[7:0] register is set; on the other hand, dmaint_err is de-asserted by the DMA controller if no bit of the INT_ERR[7:0] register is set.

INT_ABT is the status of the DMA abort interrupts after masking. The mask bit of these interrupts is bit 2 (INT_ABT_MSK) of the channel configuration register (Cn_CFG). If this mask bit is set, the content of INT_ABT[n] of this register is always '0' regardless of the presence of a pending DMA abort interrupt. If the ABT bit (bit 15 of the channel control register (Cn_CSR)) is set, the DMA controller will stop the current DMA transfer and set ABT[n] (bit 23 to bit 16 of the error/abort status register (ERR/ABT)) to '1'. Then, if INT_ABT_MSK is not set, the DMA controller will set INT_ABT[n] to '1' and assert the **dmaint** interrupt.

If the maximum channel number of DMA is configured as n, bit n to bit 15 and bit n+16 to bit 31 of this register will be reserved.

Table 16-9 INT_ERR/ABT Register

Bit	Name	Type	Reset	Description
31..24	reserved	RO	0	
23	INT_ABT[7]	RO	0	Status of the DMA abort interrupts after masking 0: Channel 7 has no pending interrupt 1: Channel 7 has a pending interrupt.
22	INT_ABT[6]	RO	0	Status of the DMA abort interrupts after masking 0: Channel 6 has no pending interrupt 1: Channel 6 has a pending interrupt.
21	INT_ABT[5]	RO	0	Status of the DMA abort interrupts after masking 0: Channel 5 has no pending interrupt 1: Channel 5 has a pending interrupt.
20	INT_ABT[4]	RO	0	Status of the DMA abort interrupts after masking 0: Channel 4 has no pending interrupt 1: Channel 4 has a pending interrupt.
19	INT_ABT[3]	RO	0	Status of the DMA abort interrupts after masking 0: Channel 3 has no pending interrupt 1: Channel 3 has a pending interrupt.
18	INT_ABT[2]	RO	0	Status of the DMA abort interrupts after masking 0: Channel 2 has no pending interrupt 1: Channel 2 has a pending interrupt.
17	INT_ABT[1]	RO	0	Status of the DMA abort interrupts after masking 0: Channel 1 has no pending interrupt 1: Channel 1 has a pending interrupt.
16	INT_ABT[0]	RO	0	Status of the DMA abort interrupts after masking 0: Channel 0 has no pending interrupt 1: Channel 0 has a pending interrupt.
15..8	reserved	RO	0	
7	INT_ERR[7]	RO	0	Status of the DMA error interrupts after masking 0: Channel 7 has no pending interrupt. 1: Channel 7 has a pending interrupt.
6	INT_ERR[6]	RO	0	Status of the DMA error interrupts after masking 0: Channel 6 has no pending interrupt. 1: Channel 6 has a pending interrupt.
5	INT_ERR[5]	RO	0	Status of the DMA error interrupts after masking 0: Channel 5 has no pending interrupt. 1: Channel 5 has a pending interrupt.
4	INT_ERR[4]	RO	0	Status of the DMA error interrupts after masking 0: Channel 4 has no pending interrupt. 1: Channel 4 has a pending interrupt.
3	INT_ERR[3]	RO	0	Status of the DMA error interrupts after masking 0: Channel 3 has no pending interrupt. 1: Channel 3 has a pending interrupt.
2	INT_ERR[2]	RO	0	Status of the DMA error interrupts after masking 0: Channel 2 has no pending interrupt. 1: Channel 2 has a pending interrupt.
1	INT_ERR[1]	RO	0	Status of the DMA error interrupts after masking 0: Channel 1 has no pending interrupt. 1: Channel 1 has a pending interrupt.
0	INT_ERR[0]	RO	0	Status of the DMA error interrupts after masking 0: Channel 0 has no pending interrupt. 1: Channel 0 has a pending interrupt.

16.3.6 Error/Abort Interrupt Status Clear Register

Abbr.: INT_ERR/ABT_CLR *Offset: 0x010*

Writing ‘1’ from bit 7 to bit 0 of this register clears both INT_ERR[n] and ERR[n]. INT_ERR[n] means bit 7 to bit 0 of the error/abort interrupt status register (INT_ERR/ABT) and ERR[n] means bit 7 to bit 0 of the error/abort status register (ERR/ABT).

Writing ‘1’ from bit 23 to bit 16 of this register clears both INT_ABT[n] and ABT[n]. INT_ABT[n] means bit 23 to bit 16 of the error/abort interrupt status register (INT_ERR_ABT) and ABT[n] means bit 23 to bit 16 of the error/abort status register (ERR/ABT).

If the maximum channel number of DMA is configured as n, bit n to bit 15 and bit n+16 to bit 31 of this register will be reserved.

Table 16-10 INT_ERR/ABT_CLR Register

Bit	Name	Type	Reset	Description
31..24	reserved	WO	0	-
23	INT_ABT_CLR[7]	WO	0	Write ‘1’ to clear the INT_ABT[7] and ABT[7] statuses
22	INT_ABT_CLR[6]	WO	0	Write ‘1’ to clear the INT_ABT[6] and ABT[6] statuses
21	INT_ABT_CLR[5]	WO	0	Write ‘1’ to clear the INT_ABT[5] and ABT[5] statuses
20	INT_ABT_CLR[4]	WO	0	Write ‘1’ to clear the INT_ABT[4] and ABT[4] statuses
19	INT_ABT_CLR[3]	WO	0	Write ‘1’ to clear the INT_ABT[3] and ABT[3] statuses
18	INT_ABT_CLR[2]	WO	0	Write ‘1’ to clear the INT_ABT[2] and ABT[2] statuses
17	INT_ABT_CLR[1]	WO	0	Write ‘1’ to clear the INT_ABT[1] and ABT[1] statuses
16	INT_ABT_CLR[0]	WO	0	Write ‘1’ to clear the INT_ABT[0] and ABT[0] statuses
15..8	reserved	WO	0	-
7	INT_ERR_CLR[7]	WO	0	Write ‘1’ to clear the INT_ERR[7] and ERR[7] statuses
6	INT_ERR_CLR[6]	WO	0	Write ‘1’ to clear the INT_ERR[6] and ERR[6] statuses
5	INT_ERR_CLR[5]	WO	0	Write ‘1’ to clear the INT_ERR[5] and ERR[5] statuses
4	INT_ERR_CLR[4]	WO	0	Write ‘1’ to clear the INT_ERR[4] and ERR[4] statuses
3	INT_ERR_CLR[3]	WO	0	Write ‘1’ to clear the INT_ERR[3] and ERR[3] statuses
2	INT_ERR_CLR[2]	WO	0	Write ‘1’ to clear the INT_ERR[2] and ERR[2] statuses
1	INT_ERR_CLR[1]	WO	0	Write ‘1’ to clear the INT_ERR[1] and ERR[1] statuses
0	INT_ERR_CLR[0]	WO	0	Write ‘1’ to clear the INT_ERR[0] and ERR[0] statuses

16.3.7 Terminal Count Status Register

Abbr.: TC *Offset: 0x014*

This register shows the status of the DMA terminal count after masking. The mask bit for the DMA terminal count is bit 31 (TC_MSK) of the channel control register (Cn_CSR). If this mask bit of the Cn_CSR register is set, TC[n] of this register will always be '0' regardless of the presence of a DMA terminal count. Please refer to Section 16.2.4 and Section 16.2.5 for more information on the DMA terminal count.

If the maximum channel number of DMA is configured as n, bit n to bit 7 of this register will be reserved.

Table 8-10 TC Register

Bit	Name	Type	Reset	Description
7	TC[7]	RO	0	Status of the DMA terminal count 0: Channel 7 has no terminal count status. 1: Channel 7 has a terminal count status.
6	TC[6]	RO	0	Status of the DMA terminal count 0: Channel 6 has no terminal count status. 1: Channel 6 has a terminal count status.
5	TC[5]	RO	0	Status of the DMA terminal count 0: Channel 5 has no terminal count status. 1: Channel 5 has a terminal count status.
4	TC[4]	RO	0	Status of the DMA terminal count 0: Channel 4 has no terminal count status. 1: Channel 4 has a terminal count status.
3	TC[3]	RO	0	Status of the DMA terminal count 0: Channel 3 has no terminal count status. 1: Channel 3 has a terminal count status.
2	TC[2]	RO	0	Status of the DMA terminal count 0: Channel 2 has no terminal count status. 1: Channel 2 has a terminal count status.
1	TC[1]	RO	0	Status of the DMA terminal count 0: Channel 1 has no terminal count status. 1: Channel 1 has a terminal count status.
0	TC[0]	RO	0	Status of the DMA terminal count 0: Channel 0 has no terminal count status. 1: Channel 0 has a terminal count status.

16.3.8 Error/Abort Status Register

Abbr.: ERR/ABT *Offset: 0x018*

ERR is the status of the DMA error. If an AHB ERROR response happens during a DMA transfer, the DMA controller will stop the current DMA transfer and set ERR[n]=1. If INT_ERR_MSK is not set, the DMA controller will set INT_ERR[n]=1 and assert both the **dmaint_err** and **dmaint** interrupts.

ABT is the status of the DMA abort. If the ABT bit (bit 15 of the channel control register (Cn_CSR)) is set, the DMA controller will stop the current DMA transfer and set ABT[n]=1. If INT_ABT_MSK is not set, the DMA controller will set INT_ABT[n]=1 and assert the **dmaint** interrupt.

If the maximum channel number of DMA is configured as n, bit n to bit 15 and bit n+16 to bit 31 of this register will be reserved.

Table 8-11 ERR/ABT Register

Bit	Name	Type	Reset	Description
31..24	reserved	RO	0	-
23	ABT[7]	RO	0	Status of the DMA abort 0: Channel 7 has no abort status. 1: Channel 7 has an abort status.
22	ABT[6]	RO	0	Status of the DMA abort 0: Channel 6 has no abort status. 1: Channel 6 has an abort status.
21	ABT[5]	RO	0	Status of the DMA abort 0: Channel 5 has no abort status. 1: Channel 5 has an abort status.
20	ABT[4]	RO	0	Status of the DMA abort 0: Channel 4 has no abort status. 1: Channel 4 has an abort status.
19	ABT[3]	RO	0	Status of the DMA abort 0: Channel 3 has no abort status. 1: Channel 3 has an abort status.
18	ABT[2]	RO	0	Status of the DMA abort 0: Channel 2 has no abort status. 1: Channel 2 has an abort status.
17	ABT[1]	RO	0	Status of the DMA abort 0: Channel 1 has no abort status. 1: Channel 1 has an abort status.
16	ABT[0]	RO	0	Status of the DMA abort 0: Channel 0 has no abort status. 1: Channel 0 has an abort status.

Bit	Name	Type	Reset	Description
15..8	reserved	RO	0	-
7	ERR[7]	RO	0	Status of the DMA error 0: Channel 7 has no error status. 1: Channel 7 has an error status.
6	ERR[6]	RO	0	Status of the DMA error 0: Channel 6 has no error status. 1: Channel 6 has an error status.
5	ERR[5]	RO	0	Status of the DMA error 0: Channel 5 has no error status. 1: Channel 5 has an error status.
4	ERR[4]	RO	0	Status of the DMA error 0: Channel 4 has no error status. 1: Channel 4 has an error status.
3	ERR[3]	RO	0	Status of the DMA error 0: Channel 3 has no error status. 1: Channel 3 has an error status.
2	ERR[2]	RO	0	Status of the DMA error 0: Channel 2 has no error status. 1: Channel 2 has an error status.
1	ERR[1]	RO	0	Status of the DMA error 0: Channel 1 has no error status. 1: Channel 1 has an error status.
0	ERR[0]	RO	0	Status of the DMA error 0: Channel 0 has no error status. 1: Channel 0 has an error status.

16.3.9 Channel Enable Status Register

Abbr.: CH_EN *Offset: 0x01C*

This register shows the DMA channel enable status. It is a read-only register.

If the maximum channel number of DMA is configured as n, bit n to bit 7 of this register will be reserved.

Table 3-12 CH_EN Register

Bit	Name	Type	Reset	Description
7	CH_EN[7]	RO	0	Status of the channel 7 using the CH_EN bit of the C7_CSR register 0: CH_EN = 0 1: CH_EN = 1
6	CH_EN[6]	RO	0	Status of the channel 6 using the CH_EN bit of the C6_CSR register 0: CH_EN = 0 1: CH_EN = 1
5	CH_EN[5]	RO	0	Status of the channel 5 using the CH_EN bit of the C5_CSR register 0: CH_EN = 0 1: CH_EN = 1
4	CH_EN[4]	RO	0	Status of the channel 4 using the CH_EN bit of the C4_CSR register 0: CH_EN = 0 1: CH_EN = 1
3	CH_EN[3]	RO	0	Status of the channel 3 using the CH_EN bit of the C3_CSR register 0: CH_EN = 0 1: CH_EN = 1
2	CH_EN[2]	RO	0	Status of the channel 2 using the CH_EN bit of the C2_CSR register 0: CH_EN = 0 1: CH_EN = 1
1	CH_EN[1]	RO	0	Status of the channel 1 using the CH_EN bit of the C1_CSR register 0: CH_EN = 0 1: CH_EN = 1
0	CH_EN[0]	RO	0	Status of the channel 0 using the CH_EN bit of the C0_CSR register 0: CH_EN = 0 1: CH_EN = 1

16.3.10 Channel Busy Status Register

Abbr.: CH_BUSY *Offset: 0x020*

This register shows the DMA channel busy status. It is a read-only register.

If the maximum channel number of DMA is configured as n, bit n to bit 7 of this register will be reserved.

Table 8-13 CH_BUSY Register

Bit	Name	Type	Reset	Description
7	CH_BUSY[7]	RO	0	Status of the channel 7 using the BUSY bit of the C7_CFG register
6	CH_BUSY[6]	RO	0	Status of the channel 6 using the BUSY bit of the C6_CFG register
5	CH_BUSY[5]	RO	0	Status of the channel 5 using the BUSY bit of the C5_CFG register
4	CH_BUSY[4]	RO	0	Status of the channel 4 using the BUSY bit of the C4_CFG register
3	CH_BUSY[3]	RO	0	Status of the channel 3 using the BUSY bit of the C3_CFG register
2	CH_BUSY[2]	RO	0	Status of the channel 2 using the BUSY bit of the C2_CFG register
1	CH_BUSY[1]	RO	0	Status of the channel 1 using the BUSY bit of the C1_CFG register
0	CH_BUSY[0]	RO	0	Status of the channel 0 using the BUSY bit of the C0_CFG register

16.3.11 Main Configuration Status Register

Abbr.: CSR *Offset: 0x024*

Table 8-14 CSR Register

Bit	Name	Type	Reset	Description
7..3	reserved		0	-
2	M1ENDIAN	RW	0	Endian configuration of AHB Master 1 0: Little endian 1: reserved
1	M0ENDIAN	RW	0	Endian configuration of AHB Master 0 0: Little endian 1: reserved
0	DMACEN	RW	0	DMA controller enable 0: Disable 1: Enable

AHB Master 0 accesses AHB 0 (Data) and AHB Master 1 accesses to AHB 7 (Figure 16-1).

16.3.12 Synchronization Register

Abbr.: SYNC *Offset:* 0x028

The synchronization register is used to disable or enable DMA controller internal synchronization logic for the hardware DMA request (dma_req[n:0]). If the synchronization logic is enabled, it will synchronize the hardware DMA request. If the synchronization logic is disabled, the hardware DMA request bypasses synchronization circuit. It is necessary to enable the synchronization logic if the DMA controller and the module that drives **dma_req** run at different clock domains.

If the maximum channel number of DMA is configured as n, bit n to bit 7 of this register will be reserved.

Table 8-15 SYNC Register

Bit	Name	Type	Reset	Description
7	SYNC[7]	RW	0	Control the DMA synchronization logic for requesting channel 7 0: Disable 1: Enable
6	SYNC[6]	RW	0	Control the DMA synchronization logic for requesting channel 6 0: Disable 1: Enable
5	SYNC[5]	RW	0	Control the DMA synchronization logic for requesting channel 5 0: Disable 1: Enable
4	SYNC[4]	RW	0	Control the DMA synchronization logic for requesting channel 4 0: Disable 1: Enable
3	SYNC[3]	RW	0	Control the DMA synchronization logic for requesting channel 3 0: Disable 1: Enable
2	SYNC[2]	RW	0	Control the DMA synchronization logic for requesting channel 2 0: Disable 1: Enable
1	SYNC[1]	RW	0	Control the DMA synchronization logic for requesting channel 1 0: Disable 1: Enable
0	SYNC[0]	RW	0	Control the DMA synchronization logic for requesting channel 0 0: Disable 1: Enable

16.3.13 Channel Registers

16.3.13.1 Channel Control Register

Abbr.: Cn_CSR *Offset: 0x100 + 0x20 * n; n = 0..7*

Table 8-16 C*n*_CSR Register

Bit	Name	Type	Reset	Description
31	TC_MSK	RW	0	Terminal count status mask for the current transaction: 0: When terminal count happens, the TC status register will be set (Default). 1: When terminal count happens, the TC status register will not be set.
30..27	reserved	RW	0	-
26..24	DMA_FF_TH	RW	0	DMA FIFO threshold value: 000: Threshold value = 1 001: Threshold value = 2 010: Threshold value = 4 011: Threshold value = 8 100: Threshold value = 16 101..111: reserved When DMA FIFO space ≥ DMA_FF_TH, the DMA controller will start to transfer data from source to FIFO. When the number of valid data in DMA FIFO is greater than DMA_FF_TH, the DMA controller will start to pop out data from FIFO to the destination. Please not that DMA_FF_TH cannot be larger than the ½ DMA FIFO size.

Bit	Name	Type	Reset	Description
23..22	CHPRI	RW	0	Channel priority level: 3: Highest priority 2: Second highest priority 1: Third highest priority 0: Lowest priority (Default)
21	PROT3	RW	0	Protection information for cacheability Note: This bit controls the AHB HPROT[3]. tied fix to zero
20	PROT2	RW	0	Protection information for bufferablity Note: This bit controls the AHB HPROT[2]. tied fix to zero
19	PROT1	RW	0	Protection information for mode indication Note: This bit controls the AHB HPROT[1]. tied fix to zero
18..16	SRC_SIZE	RW	0	Source burst size selection 000: Burst size = 1 (Default) 001: Burst size = 4 010: Burst size = 8 011: Burst size = 16 100: Burst size = 32 101: Burst size = 64 110: Burst size = 128 111: Burst size = 256 Notes: Recommended sizes are 4, 8 or 16. 1. The source burst size is not related to HBURST (The AHB signals). It only indicates the number of existing transfers before DMA re-arbitrates among the enabled channels. The number of bytes to be transferred for one burst depends on the source burst size and the source transfer width. For example, if the source burst size is 64 (SRC_SIZE=101) and the source transfer width is 16 bits (SRC_WIDTH=001), the total number of bytes of this burst transfer will be 128 (64 * 2) 2. (SRC_SIZE * SRC_WIDTH) must be equal to or larger than DST_WIDTH. Therefore, the following settings are not allowed: • Burst size = 1, source width = 8, destination width = 16 • Burst size = 1, source width = 8, destination width = 32 • Burst size = 1, source width = 16, destination width = 32

Bit	Name	Type	Reset	Description
15	ABT	WO	0	Transaction abort Writing '1' to this bit causes DMA to stop the current transfer, then set the ABT[n] bit of the Error/Abort Status register and assert the dmaint interrupt if INT_ABT_MSK=0. Note: When writing ABT=1, all other bits of this register will remain the same. That is, users cannot program bit 15 to '1' and other bits of this register simultaneously.
14	reserved	RW	0	-
13..11	SRC_WIDTH	RW	2	Source transfer width Hardware will automatically pack and unpack data as required. 000: Transfer width is 8 bits. 001: Transfer width is 16 bits. 010: Transfer width is 32 bits (Default). others: reserved Notes: 1. If source transfer width < destination transfer width, DMA will pack the source input data. For example, if the source transfer width equals to 8 bits and the destination transfer width equals to 32 bits, then DMA will pack four sets of 8-bit source data and transfer one set of 32-bit data to destination. Limitation: Do not set SRCAD_CTL=01 (Decrement source address) when the pack function works; otherwise DMA will have a wrong action. 2. If source transfer width > destination transfer width, DMA will unpack the source input data. For example, if the source transfer width equals to 32 bits and the destination transfer width equals to 8 bits, then DMA will unpack source of 32-bit data and transfer four sets of 8-bit data to destination.
10..8	DST_WIDTH	RW	2	Destination transfer width. Hardware will automatically pack and unpack data as required. 000: Transfer width is 8 bits. 001: Transfer width is 16 bits. 010: Transfer width is 32 bits (Default). others: reserved Notes: 1. If source transfer width < destination transfer width, DMA will pack the source input data. For example, if the source transfer width equals to 8 bits and the destination transfer width equals to 32 bits, then DMA will pack four sets of 8-bit source data and transfer one set of 32-bit data to destination. Limitation: Do not set SRCAD_CTL=01 (Decrement source address) when the pack function works; otherwise DMA will have a wrong action. 2. If source transfer width > destination transfer width, DMA will unpack the source input data. For example, if the source transfer width equals to 32 bits and the destination transfer width equals to 8 bits, then DMA will unpack the source of 32-bit data and transfer four sets of 8-bit data to destination.

Bit	Name	Type	Reset	Description
7	MODE	RW	0	0: Normal mode (Default) 1: Hardware handshake mode
6..5	SRCAD_CTL	RW	0	Source address control 00: Increment source address (Default) 01: Decrement source address 10: Fixed source address 11: reserved Notes: 1. If source transfer width < destination transfer width, DMA will pack the source input data. For example, if the source transfer width equals to 8 bits and the destination transfer width equals to 32 bits, then DMA will pack four sets of 8-bit source data and transfer one set of 32-bit data to destination. Limitation: Do not set SRCAD_CTL=01 (Decrement source address) when the pack function works; otherwise DMA will have a wrong action. 2. If source transfer width > destination transfer width, DMA will unpack the source input data. For example, if the source transfer width equals to 32 bits and the destination transfer width equals to 8 bits, then DMA will unpack the source of 32-bit data and transfer four sets of 8-bit data to destination.
4..3	DSTAD_CTL	RW	0	Destination Address Control 00: Increment destination address (Default) 01: Decrement destination address 10: Fixed destination address 11: reserved Notes: 1. If source transfer width < destination transfer width, DMA will pack the source input data. For example, if the source transfer width equals to 8 bits and the destination transfer width equals to 32 bits, then DMA will pack four sets of 8-bit source data and transfer one set of 32-bit data to destination. Limitation: Do not set SRCAD_CTL=01 (Decrement source address) when the pack function works; otherwise DMA will have a wrong action. 2. If source transfer width > destination transfer width, DMA will unpack the source input data. For example, if the source transfer width equals to 32 bits and the destination transfer width equals to 8 bits, then DMA will unpack the source of 32-bit data and transfer four sets of 8-bit data to destination.
2	SRC_SEL	RW	0	0: AHB Master 0 is the source (access to all modules) (Default) 1: AHB Master 1 is the source (exclusive access to DDR2-RAM)
1	DST_SEL	RW	0	0: AHB Master 0 is the destination (access to all modules) (Default) 1: AHB Master 1 is the destination (exclusive access to DDR2-RAM)
0	CH_EN	RW	0	Channel Enable 0: Disable (Default) 1: Enable

16.3.13.2 Channel Configuration Register

Abbr.: Cn_CFG *Offset: 0x104 + 0x20 * n; n = 0..7*

Table 8-17 C*n*_CFG Register

Bit	Name	Type	Reset	Description
31..20	reserved	RO	0	-
19..16	LLP_CNT	RO	0	Chain transfer counter This counter is cleared when CH_EN changes from '0' to '1'. If the chain transfer is enabled after each block data transfer is finished, LLP_CNT will be increased by 1. In case of an error or abort conditions happens, LLP_CNT will also be increased by 1. Please refer to Section 16.2.3 for more information on this counter.
15..14	reserved	RO	0	-
13	DST_HE	RW	1	Destination hardware handshake mode enable 0: Disable 1: Enable (Default) When users disable the destination hardware handshake, DMA will start transferring data without waiting for the destination request. This bit is only valid when channel is in the hardware handshake mode
12..9	DST_RS	RW	0	Destination DMA request select Specifies dma_req for destination transfer, if hardware handshake mode is enabled. DMA request sources listed in Table 16-1.
8	BUSY	RO	0	The DMA channel is busy. Please note that only one channel is busy at a time.
7	SRC_HE	RW	1	Source Hardware Handshake Mode enable 0: Disable 1: Enable (Default) When users disable the source hardware handshake, DMA will start transferring data without waiting for the source request. This bit is only valid when DMAC is in the hardware handshake mode
6..3	SRC_RS	RW	0	Source DMA request select Specifies dma_req for source transfer, if hardware handshake mode is enabled. DMA request sources listed in Table 16-1.
2	INT_ABT_MSK	RW	1	Channel abort interrupt mask 0: No mask interrupt 1: Mask interrupt (Default)
1	INT_ERR_MSK	RW	1	Channel error interrupt mask 0: No mask interrupt 1: Mask interrupt (Default)
0	INT_TC_MSK	RW	1	Channel terminal count interrupt mask 0: No mask interrupt 1: Mask interrupt (Default)

16.3.13.3 Channel Source Address Register

Abbr.: Cn_SrcAddr *Offset: 0x108 + 0x20 * n; n = 0..7*

Table 8-18 Cn_SrcAddr Register

Bit	Name	Type	Reset	Description
31..0	Cn_SrcAddr	RW	0	Source starting address Please note that when the DMA transaction is complete, its value will be changed to the DMA source ending address.

16.3.13.4 Channel Destination Address Register

Abbr.: Cn_DstAddr *Offset: 0x10C + 0x20 * n; n = 0..7*

Table 8-19 Cn_DstAddr Register

Bit	Name	Type	Reset	Description
31..0	Cn_DstAddr	RW	0	Destination starting address Please note that when the DMA transaction is complete, its value will be changed to the DMA destination ending address.

16.3.13.5 Linked List Descriptor Pointer

Abbr.: Cn_LLP *Offset: 0x110 + 0x20 * n; n = 0..7*

Table 8-20 Cn_LLP Register

Bit	Name	Type	Reset	Description
31..2		RW	0	Linked list descriptor pointer address
1	reserved	RW	0	-
0		RW	0	Master for loading the next LLP 0: Load the next LLP from the AHB Master 0 (Default) 1: Load the next LLP from the AHB Master 1

AHB Master 0 accesses AHB 0 (Data) and AHB Master 1 accesses to AHB 7 (Figure 16-1).

16.3.13.6 Transfer Size Register

Abbr.: Cn_SIZE *Offset: 0x114 + 0x20 * n; n = 0..7*

Table 8-21 Cn_SIZE Register

Bit	Name	Type	Reset	Description
31..22	reserved	RW	0	-
21..0	TOT_SIZE	RW	0	Total transfer size The transfer unit depends on the source width. For example: SRC_WIDTH = 000, unit: 8-bit SRC_WIDTH = 001, unit: 16-bit SRC_WIDTH = 010, unit: 32-bit Note: When the DMA transaction is done, its value changes to 0.

16.3.13.7 DMA Source/Destination Data Transfer Mapping Rule

Rule 1: Push into FIFO from the Source Bus Byte Lane according to the following sequence if source width ≥ destination width.

Table 16-11 Storage in FIFO

Destination Address Control	Destination Transfer Width	push into FIFO			
		1. Entry	2. Entry	3. Entry	4. Entry
increment	8 bit	$\{B_0, B_0, B_0, B_0\}$	$\{B_1, B_1, B_1, B_1\}$	$\{B_2, B_2, B_2, B_2\}$	$\{B_3, B_3, B_3, B_3\}$
	16 bit	$\{B_1, B_0, B_1, B_0\}$	$\{B_3, B_2, B_3, B_2\}$		
	32 bit	$\{B_3, B_2, B_1, B_0\}$			
decrement	8 bit	$\{B_3, B_3, B_3, B_3\}$	$\{B_2, B_2, B_2, B_2\}$	$\{B_1, B_1, B_1, B_1\}$	$\{B_0, B_0, B_0, B_0\}$
	16 bit	$\{B_3, B_2, B_3, B_2\}$	$\{B_1, B_0, B_1, B_0\}$		
	32 bit	$\{B_3, B_2, B_1, B_0\}$			

Rule 2: Push into FIFO from the collected (Packed) Source Bus Byte Lane according to the following sequence if source width < destination width and the source address control is the increment type (pack function is not allowed for the decrement type).

Table 16-12 Source Transfers (Source Address Control = increment)

Source Transfer Width	Destination Transfer Width	Source Bus Byte Lane		push into FIFO (after 2. Transfer)	Source Bus Byte Lane		push into FIFO (after 4. Transfer)
		1. Transfer	2. Transfer		3. Transfer	4. Transfer	
8 bit	32 bit	B_{n+0}	B_{n+1}		B_{n+2}	B_{n+3}	$\{B_{n+3}, B_{n+2}, B_{n+1}, B_{n+0}\}$
	16 bit	B_{n+0}	B_{n+1}	$\{B_{n+1}, B_{n+0}, B_{n+1}, B_{n+0}\}$	B_{n+2}	B_{n+3}	$\{B_{n+3}, B_{n+2}, B_{n+3}, B_{n+2}\}$
16 bit	32 bit	$\{B_{n+1}, B_{n+0}\}$	$\{B_{n+3}, B_{n+2}\}$	$\{B_{n+3}, B_{n+2}, B_{n+1}, B_{n+0}\}$			

Rule 3: Pop from FIFO and unpack (source width > destination width) to Destination Bus Byte Lane according to the following sequences:

Table 16-13 Destination Transfers

Destination Address Control	Destination Transfer Width	Destination Bus Byte Lane			
		1. Transfer	2. Transfer	3. Transfer	4. Transfer
increment or fix	8 bit	$\{B_0, B_0, B_0, B_0\}$	$\{B_1, B_1, B_1, B_1\}$	$\{B_2, B_2, B_2, B_2\}$	$\{B_3, B_3, B_3, B_3\}$
	16 bit	$\{B_1, B_0, B_1, B_0\}$	$\{B_3, B_2, B_3, B_2\}$		
	32 bit	$\{B_3, B_2, B_1, B_0\}$			
decrement	8 bit	$\{B_3, B_3, B_3, B_3\}$	$\{B_2, B_2, B_2, B_2\}$	$\{B_1, B_1, B_1, B_1\}$	$\{B_0, B_0, B_0, B_0\}$
	16 bit	$\{B_3, B_2, B_3, B_2\}$	$\{B_1, B_0, B_1, B_0\}$		
	32 bit	$\{B_3, B_2, B_1, B_0\}$			

16.4 Programming Guidelines

16.4.1 Channel Initialization (Normal Mode, No Chain Transfer)

1. set CSR=0x01
   a. little endian configuration (M1ENDIAN=0, M0ENDIAN=0)
   b. enable the DMA controller (DMACEN=1)
2. set channel registers
   a. Cn_CFG:
      - interrupt mask (INT_ABT_MSK, INT_ERR_MSK, INT_TC_MSK)
   b. Cn_SrcAddr and Cn_DstAddr
   c. Cn_LLP=0 (no linked list)
   d. Set the transfer number, Cn_SIZE, to determine how many transfers are required in one DMA transaction
   e. Cn_CSR:
      - priority (CHIPRI)
      - transfer burst size (SRC_SIZE)
      - transfer width (SRC_WIDTH, DST_WIDTH)
      - increment or decrement address
      - source or destination interface (SRC_SEL, DST_SEL)
      - normal mode (MODE=0)
      - start transfer (CH_EN=1)

16.4.2 Channel Initialization (Normal Mode, Chain Transfer)

1. fill linked list descriptors according to Table 16-2, Table 16-3 and Table 16-4
2. set CSR=0x01
   a. little endian configuration (M1ENDIAN=0, M0ENDIAN=0)
   b. enable the DMA controller (DMACEN=1)
3. set channel registers (1st linked list transfer)
   a. Cn_CFG:
      - interrupt mask (INT_ABT_MSK, INT_ERR_MSK, INT_TC_MSK)
   b. Cn_SrcAddr and Cn_DstAddr
   c. Cn_LLP= starting address of next linked list descriptor (1st descriptor of list defined in system memory ≙ 2nd linked list transfer)
   d. Set the transfer number and Cn_SIZE. to determine how many transfers are required in one DMA transaction
   f. Cn_CSR:
      - priority (CHIPRI)
      - transfer burst size (SRC_SIZE)
      - transfer width (SRC_WIDTH, DST_WIDTH)
      - increment or decrement address
      - source or destination interface (SRC_SEL, DST_SEL)
      - normal mode (MODE=0)
      - start DMA transfer (CH_EN=1)

16.4.3 Channel Initialization (Handshake Mode, No Chain Transfer)

1. set CSR=0x01
    a. little endian configuration (M1ENDIAN=0, M0ENDIAN=0)
    b. enable the DMA controller (DMACEN=1)
2. set SYNC to control the channel synchronization logic
3. set channel registers
    a. Cn_CFG:
        - destination request enable (DST_HE)
        - destination request select (DST_RS)
        - source request enable (SRC_HE)
        - source request select (SRC_RS)
        - interrupt mask (INT_ABT_MSK, INT_ERR_MSK, INT_TC_MSK)
    b. Cn_SrcAddr and Cn_DstAddr
    c. Cn_LLP=0 (no linked list)
    d. Set the transfer number, Cn_SIZE, to determine how many transfers are required in one DMA transaction
    a. Cn_CSR:
        - priority (CHIPRI)
        - transfer burst size (SRC_SIZE)
        - transfer width (SRC_WIDTH, DST_WIDTH)
        - increment or decrement address
        - source or destination interface (SRC_SEL, DST_SEL)
        - hardware handshake mode (MODE=1)
        - start/enable transfer (CH_EN=1)

16.4.4 Channel Initialization (Handshake Mode, Chain Transfer)

1. fill linked list descriptors according to Table 16-2, Table 16-3 and Table 16-4
2. set CSR=0x01
    a. little endian configuration (M1ENDIAN=0, M0ENDIAN=0)
    b. enable the DMA controller (DMACEN=1)
3. set SYNC to control the channel synchronization logic
4. set channel registers(1st linked list transfer)
    e. Cn_CFG:
        - destination request enable (DST_HE)
        - destination request select (DST_RS)
        - source request enable (SRC_HE)
        - source request select (SRC_RS)
        - interrupt mask (INT_ABT_MSK, INT_ERR_MSK, INT_TC_MSK)
    f. Cn_SrcAddr and Cn_DstAddr
    g. Cn_LLP= starting address of next linked list descriptor (1st descriptor of list defined in system memory ≙ 2nd linked list transfer)
    h. Set the transfer number, Cn_SIZE, to determine how many transfers are required in one DMA transaction
    a. Cn_CSR:
        - priority (CHIPRI)
        - transfer burst size (SRC_SIZE)
        - transfer width (SRC_WIDTH, DST_WIDTH)
        - increment or decrement address
        - source or destination interface (SRC_SEL, DST_SEL)
        - hardware handshake mode (MODE=1)
        - start/enable transfer (CH_EN=1)

16.4.5 Channel Abort (During DMA Data Transfer)

1. set channel registers
    a. Cn_CFG:
        - clear mask (INT_ABT_MSK=0)
    b. Cn_CSR:
        - abort transfer (ABT=1)
2. wait for the assertion of abort status (ERR/ABR register) and interrupt (INT_ERR/ABR register), and read ERR/ABT to detect which channel has been aborted
3. clear ABT by setting the corresponding INT_ABT_CLR (INT_ERR/ABT_CLR register) to enable a new transfer
4. A new transfer can now be initiated.

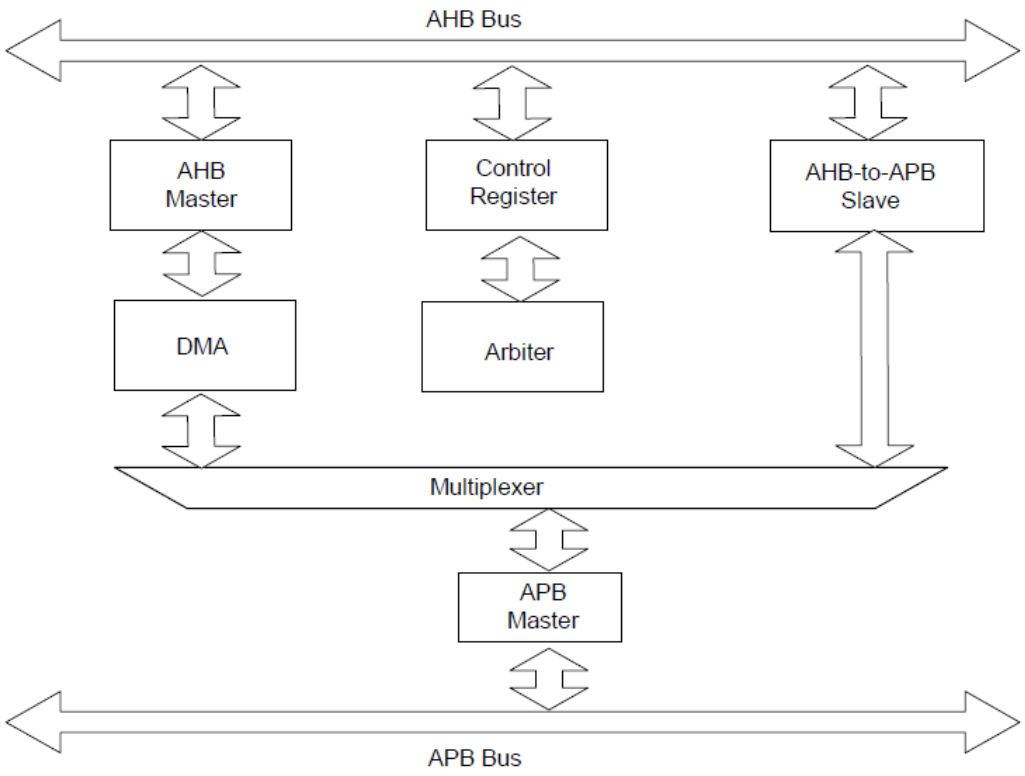
17 APB_Bridge

Offset: APB_Bridge1: 0xE8000000
APB_Bridge2: 0xF0000000

17.1 Overview

The main purpose of the APB Bridge is to convert the transactions between the APB and AHB buses. It provides the DMA function by using the hardware configurations. Up to 15 sets of DMA requests/grants and four configurable DMA channels are available to the end users.

Figure 17-1 Functional Block Diagram of APB Bridge



The APB Bridge contains a set of control registers, two AHB slave interfaces (One for accessing the control registers and the other for accessing the APB devices), one AHB master interface for the DMA functions, one DMA engine, and one arbiter. The following sections describe the functions of each block.

17.1.1 Control Register

This block controls the APB Bridge. The base address and the space size of all devices can be set by programming this block. Users can also program the control register to support DMA by using the APB Bridge.

17.1.2 AHB Slave for Accessing APB Devices

The AHB slave interface converts the AHB-bus transfers into the APB-bus transfers.

17.1.3 AHB Slave for Accessing APB Bridge Registers

The AHB slave interface accesses the control registers of the APB Bridge.

17.1.4 AHB Master

The AHB master performs the AHB bus transfers if DMA of the APB Bridge requests the data transfers of the AHB bus.

17.1.5 APB Master

The APB master is the only master on the APB bus that performs the APB transfers.

17.1.6 DMA Engine

The APB Bridge supports the DMA functions for the AHB-to-AHB, AHB-to-APB, APB-to-AHB, and APB-to-APB transactions. The DMA engine can support up to four channels and 15 request/grant pins. A channel granted by the arbiter block is the only channel that starts transfers. Each channel can be programmed to one of the 15 sets of request/grant pins in the hardware handshake mode to act as a source device or act as a destination device. The main function of the hardware handshake mode is to provide an indication of the device status. When the device is ready to transfer data, the request signal will be set. DMA will then reply with a grant signal. Users can also disable the hardware handshake mode by programming the register when it is not necessary for a DMA transfer to refer to the request/grant pins.
For the BYTE transfers, the engine puts data of the target byte in the least significant byte of data bus (Bits[7:0]). For the HALFWORD transfers, the engine puts data of the target halfword in the least significant halfword of data bus (Bits[15:0]).

17.1.7 DMA Hardware Handshake Mode

In this mode, there is a handshake protocol to be followed between DMA and other devices to complete the DMA transfer. Both the source and the destination devices can be programmed in the hardware handshake mode when users set the SRREQ_SEL/DRREQ_SEL bits in the command register (Table 17-8) to individually select the source and destination request signal, (see Table 17-1 and Table 17-2) for reference. After the ENbDis (Table 17-8)
bit is set, the DMA will start transfer when the request signal is active from both the source and the destination devices. After one DMA cycle transfer is completed, DMA will assert the grant signal for the source and the destination devices. When the device detects the grant signal, it will de-assert its request signal. After detecting the de-assertion of the request signal, DMA will de-assert the grant signal. One handshake communication will cause one DMA cycle transfer, and another handshake communication will cause another DMA cycle transfer. After the total count of the DMA cycle is completed, it will assert the interrupt. Figure 17-2 shows the timing diagram of the hardware handshake protocol.

Figure 17-2 Hardware Handshake Protocol

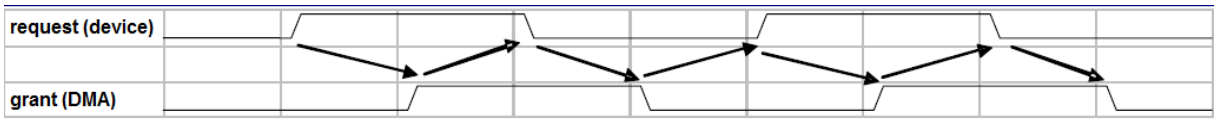


Table 17-1 Source and destination of requests – APB_Bridge1

Request/Grand Signal	Request Source/Grand Destination
15..7	Reserved
6	SPI_RX
5	SPI_TX
4..3	Reserved
2	UART2_TX
1	UART2_RX

Table 17-2 Source and destination of requests – APB_Bridge2

Request/Grand Signal	Request Source/Grand Destination
15..5	Reserved
4	UART1_TX
3	UART1_RX
2..1	Reserved

17.1.8 Arbiter

The arbitration mechanism is used to ensure that only one channel has the right to access the bus at one time. The arbiter performs this function by observing a number of different requests to use the bus and by deciding which channel will have the highest priority. The round robin algorithm is implemented to arbitrate the requests of all DMA channels. That is, the DMA channel, of which the request is currently being granted, will have the lowest priority in the next arbitration.

17.2 Memory Map and Register Definition

Table 17-3 Summary of APB Bridge Registers

Address Offset	Type	Size (Byte)	Description	Default Value
0x00	RW	4	APB slave0 base/size reg	0x00030000
0x04	RW	4	APB slave1 base/size reg	0x00830000
0x08	RW	4	APB slave2 base/size reg	0x01030000
0x0C	RW	4	APB slave3 base/size reg	0x01830000
0x10	RW	4	APB slave4 base/size reg	0x02030000
0x14	RW	4	APB slave5 base/size reg	0x02830000
0x18	RW	4	APB slave6 base/size reg	0x03030000
0x1C	RW	4	APB slave7 base/size reg	0x03830000
0x20..0x7F	RW	96	Reserved	0
0x80	RW	4	Source address DMA_A	0
0x84	RW	4	Destination address DMA_A	0

Address Offset	Type	Size (Byte)	Description	Default Value
0x88	RW	4	Cycles of DMA_A	0
0x8C	RW	4	Command of DMA_A	0
0x90	RW	4	Source address DMA_B	0
0x94	RW	4	Destination address DMA_B	0
0x98	RW	4	Cycles of DMA_B	0
0x9C	RW	4	Command of DMA_B	0
0xA0	RW	4	Source address DMA_C	0
0xA4	RW	4	Destination address DMA_C	0
0xA8	RW	4	Cycles of DMA_C	0
0xAC	RW	4	Command of DMA_C	0
0xB0	RW	4	Source address DMA_D	0
0xB4	RW	4	Destination address DMA_D	0
0xB8	RW	4	Cycles of DMA_D	0
0xBC	RW	4	Command of DMA_D	0
0xC0	RW	4	APB control register	0
0xC4	RWOC	4	APB status register	0

The following subsections describe the details of the APB Bridge register.

17.2.1 Base/Size Register of APB Slave0..7

Offset: 0x00..0x1C

This register configures the base address/space size of APB slave0..7. The values of the Base/Size register of all the slaves have the same format and same definition as the fields. Table 17-4 shows the bit assignments.

Table 17-4 Base/Size Register of APB Slave0..7

Bit	Name	Type	Reset	Description
31..30	Reserved	R	0	-
29..20	BASE	RW	see Table 17-3	Base addresses[29..20]
19..16	SIZE	RW	0x3	Size of the address space 0x0: 1M 0x1: 2M 0x2: 4M 0x3: 8M 0x4: 16M 0x5: 32M 0x6: 64M 0x7: 128M 0x8: 256M 0x9 ~ 0xF: Reserved
15..0	Reserved	R	0	-

17.2.2 Source Address of DMA Channel

Offset: Channel A: 0x80

Channel B: 0x90
Channel C: 0xA0
Channel D: 0xB0

This register configures the source address of DMA channels. Table 17-5 shows the bit assignments.

Table 17-5 Source Address of DMA Channel Register

Bit	Name	Type	Reset	Description
31..0	SRC_ADR	RW	0	Source address of the current DMA cycle

17.2.3 Destination Address of DMA Channel

Offset: Channel A: 0x84
Channel B: 0x94
Channel C: 0xA4
Channel D: 0xB4

This register configures the destination address of DMA channels. Table 17-6 shows the bit assignments.

Table 17-6 Destination Address of DMA Channel Register

Bit	Name	Type	Reset	Description
31..0	DES_ADR	RW	0	Destination address of the current DMA cycle

17.2.4 Cycle of DMA Channel

Offset: Channel A: 0x88
Channel B: 0x98
Channel C: 0xA8
Channel D: 0xB8

This register configures the cycles of DMA channels. Table 17-7 shows the bit assignments.

Table 17-7 Cycles of DMA Channel Register

Bit	Name	Type	Reset	Description
31..24	Reserved	R	0	-
23..0	CYC	RW	0	A DMA cycle consists of one or four bus data transfer cycles, depending on the DMA command burst bit (BUR_MOD, Table 17-8) in the DMA channel command register.

17.2.5 Command Address of DMA Channel

Offset: Channel A: 0x8C
Channel B: 0x9C
Channel C: 0xAC
Channel D: 0xBC

This register configures the commands of DMA channels. Table 17-8 shows the bit assignments.

Table 17-8 Commands of DMA Channels Register

Bit	Name	Type	Reset	Description
31..28	Reserved	R	0	-
27..24	SREQ_SEL	RW	0	Request/Grant signal to select the source addresses of the DMA hardware handshake mode 0x0: No request/grant signal 0x1: Select ps1_preq/ps1_pgnt as the source of the DMA hardware handshake signals 0x2: Select ps2_preq/ps2_pgnt as the source of the DMA hardware handshake signals : 0xF: Select ps15_preq/ps15_pgnt as the source of the DMA hardware handshake signals
23..22	Reserved	R	0	-
21..20	DATA_WIDTH	RW		Data width of the transfer (When the DMA source or the destination is APB device, the data width must be set to word. The APB device supports the word-width transfer only.) 00: Word 01: Half-word 10: Byte 11: Reserved
19..16	DREQ_SEL	RW	0	The request/grant signal selects the destination address in the DMA hardware handshake mode. 0: No request/grant signal 1: Select ps1_preq/ps1_pgnt as destination of the DMA hardware handshake signals 2: Select ps2_preq/ps2_pgnt as destination of the DMA hardware handshake signals : 15: Select ps15_preq/ps15_pgnt as destination of the DMA hardware handshake signals
15	Reserved	R	0	-
14..12	DES_ADR_INC	RW	0	Destination address incremental 000: No increment 001: +1 (Burst = 0), +4 (Burst = 1) 010: +2 (Burst = 0), +8 (Burst = 1) 011: +4 (Burst = 0), +16 (Burst = 1) 101: -1 (Burst = 0), no support in the burst mode (Burst = 1) 110: -2 (Burst = 0), no support in the burst mode (Burst = 1) 111: -4 (Burst = 0), no support in the burst mode (Burst = 1) Please note that when DMA is configured in the decreasing mode, the destination address cannot cross the 1K boundary within one DMA transfer period.
11	Reserved	R	0	-
10..8	SRC_ADR	RW	0	Source address incremental 000: No increment 001: +1 (Burst = 0), +4 (Burst = 1) 010: +2 (Burst = 0), +8 (Burst = 1) 011: +4 (Burst = 0), +16 (Burst = 1) 101: -1 (Burst = 0), no support in the burst mode (Burst = 1)

Bit	Name	Type	Reset	Description
				110: -2 (Burst = 0), no support in the burst mode (Burst = 1) 111: -4 (Burst = 0), no support in the burst mode (Burst = 1) Please note that when DMA is configured in the decreasing mode, the source address cannot cross the 1K boundary within one DMA transfer period.
7	DES_ADR_SEL	RW	0	Destination address control 0: APB 1: AHB
6	SRC_ADR_SEL	RW	0	Source address control 0: APB 1: AHB
5	ERR_INT_ENB	RW	0	Interrupt when the AHB slave returns an error response during a DMA transfer 0: Disable 1: Enable
4	ERR_INT_STS	RW	0	Error response interrupt flag 0: No interrupt occurs 1: Interrupt occurs This bit can be manually cleared by writing a '0' to it.
3	BUR_MOD	RW	0	Burst mode control 0: No burst; each DMA cycle consists of one single-bus transfer cycle 1: Burst; each DMA cycle consists of four bus transfer cycles
2	FIN_INT_STS	RW	0	Finished interrupt control 0: Disable 1: Enable
1	FIN_INT_STS	RW	0	Finishing interrupt flag 0: No interrupt occurs 1: Interrupt occurs This bit can be manually cleared by writing a '0' to it.
0	ENB_DIS	RW	0	Enable/Stop DMA transfer This bit can be automatically cleared when all DMA cycles are finished. 0: Stop the DMA channel 1: Enable the DMA channel Please note that CPU stops the DMA transfers by setting this bit to '0' before all DMA cycles are finished. Before CPU reconfigures another DMA transfer, CPU needs to monitor this bit to ensure that this bit is cleared by the DMA state machine.

17.2.6 APB Control Register

Offset: 0xC0

Table 17-9 APB Control Register

Bit	Name	Type	Reset	Description
31..3	Reserved	R	0	-
2	BUF_WR_ERR_INT_EN	RW	0	Interrupt enable of bufferable write 1: When the APB Bridge receives an error response of an AHB bufferable write command from an APB device, the

Bit	Name	Type	Reset	Description
				APB Bridge will assert the interrupt signal. 0: When the APB Bridge receives an error response of an AHB bufferable write command from an APB device, the APB Bridge will not assert the interrupt signal.
1..0	PROT_TYPE	RW	0	APB Bridge write buffer control 00: The AHB write data will be saved into the write buffer if the attribute of the AHB input port, hprot[1:0], is bufferable. 01: The AHB write data is always saved into the write buffer no matter what the attribute of an AHB command, hprot[1:0], will be. 10: The AHB write data is always not saved into the write buffer no matter what the attribute of an AHB command, hprot[1:0], will be.

17.2.7 APB Status Register

Offset: 0xC4

Table 17-10 APB Status Register

Bit	Name	Type	Reset	Description
31..1	Reserved	R	0	-
0	BUF_WR_ERR_STS	R/WOC	0	Error status flag for an AHB bufferable write command This flag is set when the APB Bridge receives an error of an AHB bufferable write command response from an APB device. This flag is cleared by CPU by writing a '1' to this register. (Write '1' to clear)

17.3 Initialization

Users must make sure that at least the following configuration is written to the APB slave base/size registers before accessing any of the APB slaves:

Configuration of APB_Bridge 1

0xE8000000 0x00000000
0xE8000004 0x00800000
0xE8000008 0x01000000
0xE800000C 0x01800000
0xE8000010 0x02000000
0xE8000014 0x02800000
0xE8000018 0x03000000
0xE800001C 0x03800000

Configuration of APB_Bridge 2

0xF0000000 0x02800000
0xF0000004 0x00800000
0xF0000008 0x01000000

0xF000000C 0x01800000
0xF0000010 0x02000000
0xF0000014 0x00000000
0xF0000018 0x03000000

The following sequence must be followed to turn on the DMA channel:

1. Program all registers except the ENB_DIS (Table 17-8) bit of command for DMA channel register
2. Set the ENB_DIS (Table 17-8) bit to turn on DMA, and DMA will then start to transfer
3. Clear the ENB_DIS (Table 17-8) bit if pause the DMA transfer is necessary. After DMA has been paused, users can restart DMA by setting the ENB_DIS (Table 17-8) bit.
4. When the DMA transfer is completed, the ENB_DIS (Table 17-8) bit will be cleared and the FIN_IN_STS (Table 17-8) bit will be automatically set.

18 Serial Peripheral Interface Controller (SPI)

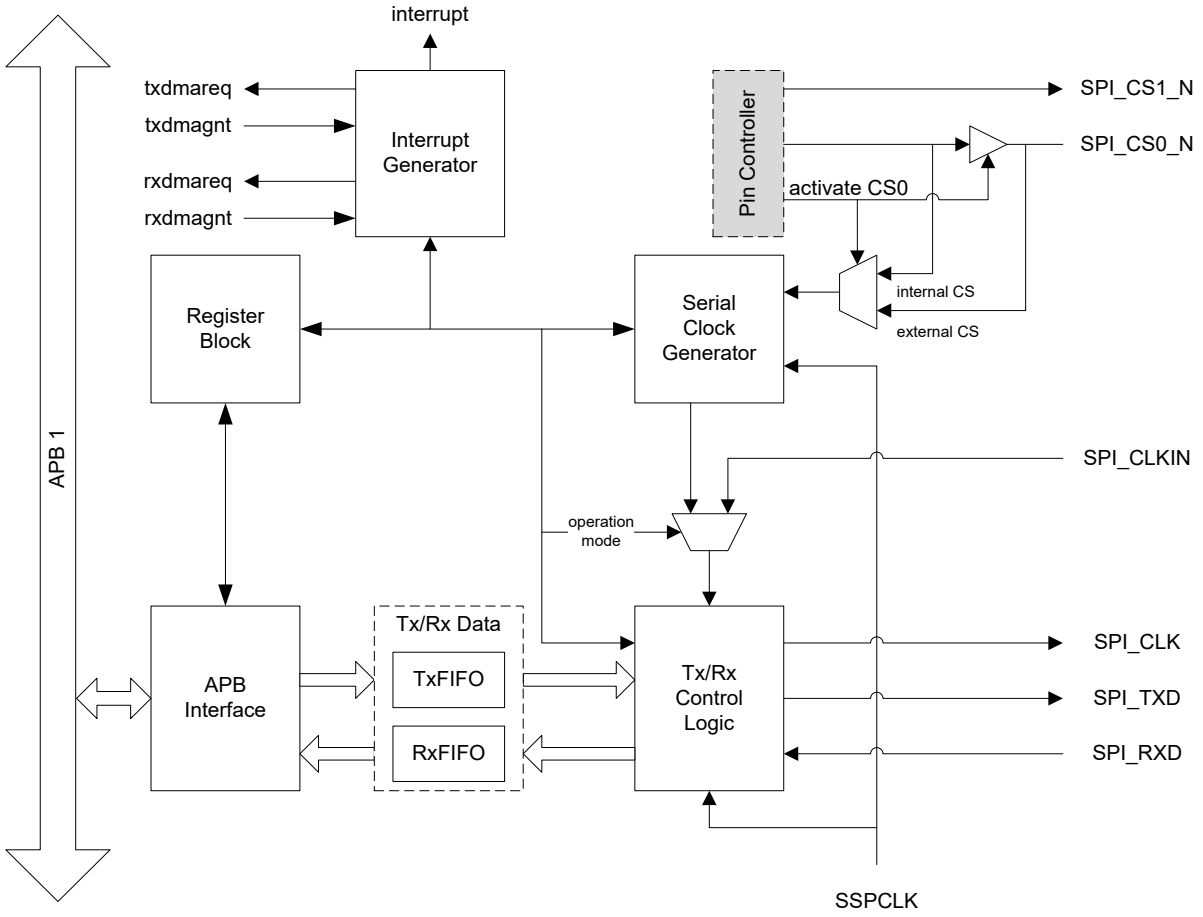
Offset: 0x72000000

The SPI controller is a synchronous serial port interface that allows the host processor to serve as a master or a slave. Various devices can be connected to this controller by using the serial protocol. SPI controller supports the Synchronous Serial Port (SSP) from Texas Instruments and the Serial Peripheral Interface (SPI) from Motorola. The serial data formats may range from 4 bits to 128 bits in length. The SPI controller can use the on-chip DMA to directly transfer data between the external serial device and the system memory without the intervention from the processor.

- Supports TI SSP (only Slave mode) and Motorola SPI frame formats
- Master Mode with up to 80 Mbit/s
- Slave Mode with up to 24 Mbit/s
- Supports independent SPI clock for easy bit clock generation
- Supports internally or externally controlled serial bit clock
- Programmable serial bit clock polarity, phase and frequency
- Programmable serial bit data sequence (MSB or LSB first)
- Programmable threshold interrupt of transmit/receive FIFO
- Independently programs interrupt enable/disable
- Independently configures transmit and receive data FIFO depths
- Receive and Transmit FIFOs size is 32 x 32 bit

18.1 Overview

Figure 18-1 Block diagram of SPI Controller



Signal feedback described in Table 18-4 is not drawn in the figure above.

18.1.1 APB Interface

The APB interface is used to decode signals from the APB to read/write signals to FIFO and register block.

18.1.2 Register Block

The register block accepts the register read/write from the APB interface. This register block also provides the property information to other blocks of the SPI controller. For more detailed descriptions of this register block, please refer to section 18.3.

18.1.3 Interrupt Generation Control

The interrupt generation control block collects information (FIFO underrun/overrun, FIFO threshold) from the transmit/receive control block and provides the value to the register block. If the interrupt condition matches, an interrupt signal will be asserted.
This block also generates the DMA request signals, if necessary. The detailed DMA operation is described in section 18.2.5.

18.1.4 Serial Clock Generator

This block generates a serial clock for communication. The format of the serial clock is defined through the control register 0 (Table 18-4). Generally, the clock will not start if the SPI controller is not enabled. Once the SPI controller is enabled and the clock running condition is matched, the serial clock will start running. The operating frequency of the serial clock depends on the setting of the SCLKDIV register. Please refer to section 18.2.1 for more details about the serial clock generation.
This block also sends the serial clock and chip select information to the data transmit/receive control block.

18.1.5 Transmit/Receive Control

The main function of this block is to perform the parallel-to-serial transmission or handle the serial-to-parallel reception from the external devices.
If the master mode is specified and the transmit FIFO is not empty, the data in the transmit FIFO will be read and shifted out via the SPI_TXD pin. If a whole word is completely shifted out and the transmit FIFO still contains valid data, the next data will be read and shifted out. The received data can be shifted in via the SPI_RXD pin. Once the reception is complete, the received word will be written into the receive FIFO, and the receive control logic will continue to receive the next word.
If the slave mode is specified, the SPI controller will start to transmit/receive data when SPI_CS0_N is activated. The transmission or reception will continue until the SPI controller is disabled or chip select is deactivated.

18.2 SPI Operation

This section describes the operations of the synchronous peripheral interface. Please refer to the programming guidelines in section 18.4 for details.

18.2.1 SPI Serial Clock Ratio and Bit Rate

The frequency of the internally generated serial clock is controlled by SCLKDIV in the SPI control register 1 (Table 18-6). The frequency of the internally generated serial clock is determined by the following formula:

Equation 18-1 Calculation of f_{SCLK}

$$f_{SCLK} = \frac{160MHz}{2 \times (SCLKDIV + 1)}$$

Table 18-1 Special f_{sclk} values

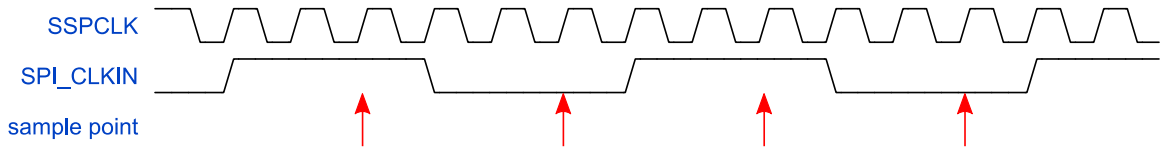
f_{sclk} (kHz)	SCLKDIV	f_{sclk} (kHz)	SCLKDIV	f_{sclk} (kHz)	SCLKDIV	f_{sclk} (kHz)	SCLKDIV	f_{sclk} (kHz)	SCLKDIV
80000	0	8000	9	800	99	80	999	8	9999
		5333.33	14	533.33	149	53.33	1499	5.33	14999
		5000	15	500	159	50	1599	5	15999
40000	1	4000	19	400	199	40	1999	4	19999
		3200	24	320	249	32	2499	3.2	24999
26666,67	2	2666.67	29	266.67	299	26.67	2999	2.67	29999
		2500	31	250	319	25	3199	2.5	31999
								2.44	32768
20000	3	2000	39	200	399	20	3999	2	39999
16000	4	1600	49	160	499	16	4999	1.6	49999
13333.33	5	1333.33	59	133.33	599	13.33	5999	1.33	59999
10000	7	1000	79	100	799	10	7999	1.22	65535

When data are transmitted or received according to the SCLK frequency, the bit rate of SPI controller will be equal to the frequency of SCLK. If the externally generated serial clock is specified (slave mode), SCLKDIV will be ignored and all the transfers will depend on the externally supplied SPI_CLKIN. Because SPI controller has to synchronize the externally generated SPI_CLKIN with internal 160 MHz clock (SSPCLK), the frequency of the externally generated SPI_CLKIN is expected to be slower than 24 MHz. Table 18-2 shows the settings of the SCLKDIV and the max. values of SCLK in the master and slave mode. Figure 18-2 shows the relationships among SSPCLK, SPI_CLKIN and the sample points. Each arrow indicates a sample point (where SPI controller transmits or receives data). When the externally generated SPI_CLKIN has a worse duty cycle, the ratio of SSPCLK/SPI_CLKIN must increase for synchronization. The high/low level of SPI_CLKIN should be at least three times of SSPCLK.

Table 18-2 SCLKDIV Settings and maximum values for SCLK

Mode Selection	max. SCLK (MHz)	SCLKDIV Setting
Master mode	80 (SPI)	by register setting
Slave mode	24	don't care

Figure 18-2 Relationships among SSPCLK, SPI_CLKIN and Sample Points



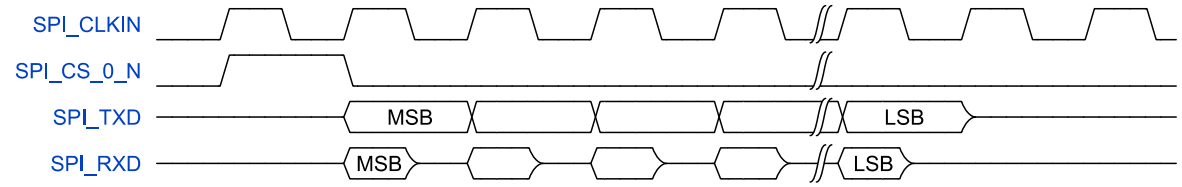
18.2.2 SPI Frame Format

The SPI-Controller supports the frame formats of TI SSP (only Slave mode) and Motorola SPI. The following subsections provide the detailed information about these frame formats.

18.2.2.1 Texas Instrument SSP Frame Format

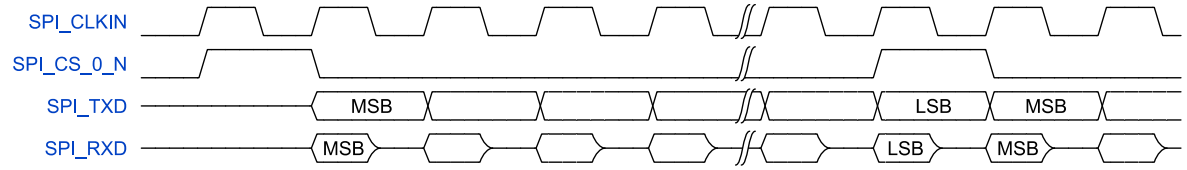
Figure 18-3 shows a single transfer by using the TI SSP frame format. In this format, each transfer begins with SPI_CS_0_N pulsed high for one SPI_CLKIN period. Then the master and slave drive data at rising edge of SPI_CLKIN and sample data at the falling edge.

Figure 18-3 Single Transfer Using TI SSP Frame Format



SPI_CS_0_N is forced to low when no data transfer exists. Figure 18-4 shows the continuous transfer waveform.

Figure 18-4 Continuous Transfer Using TI SSP Frame Format

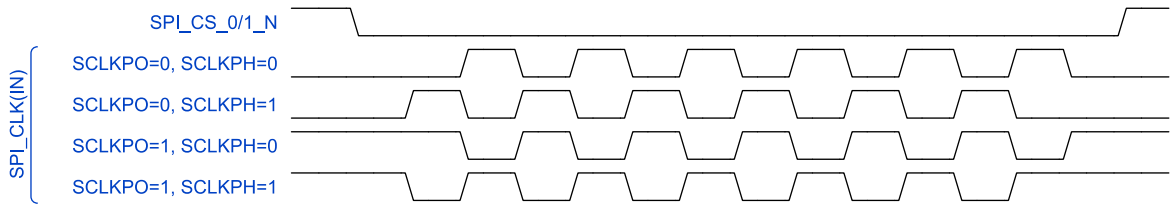


In the slave mode, when the SPI_CS_0_N is sampled high, the transmit logic will read data in the transmit FIFO and start shifting it out. If the transmit FIFO is empty, the transmit logic will try to read the last word of the transmit FIFO to shift it out. Under these circumstances, the transmit FIFO underrun interrupt will be issued if the underrun interrupt function is enabled. The receive logic will start receiving data at the same time. During transmission/reception, SPI_CS_0_N will not be qualified until the data length is reached.
If the TI SSP frame format is specified, the bit sequence will be from MSB to LSB. The setting of LSB in control register 0 (Table 18-4) will be ignored and SCLKPH and SCLKPO will be ignored as well.

18.2.2.2 Motorola SPI Frame Format

The Motorola SPI frame format is a 4-wire serial peripheral protocol. The frame format is completely controlled by SCLKPO and SCLKPH. SCLKPO controls the SCLK polarity at the idle state, and SCLKPH (SCLK phase) determines the relationship between SPI_CS_0/1_N and SCLK. Figure 18-5 shows the four modes of SPI with different combinations of SCLKPO and SCLKPH.

Figure 18-5 SPI Frame Format Relation to Different SCLKPH and SCLKPO



As shown in Figure 18-5, if SCLKPH is specified as logic 1 and SPI_CS_0/1_N is pulled active, the serial clock will start running after half of the SCLK cycle. If SCLKPH is specified

as logic 0 and SPI_CS_0/1_N is pulled active, the serial clock will start running after one SCLK cycle. If SCLKPO and SCLKPH have the same value, the transmit logic will transmit data at the falling edge of SCLK, and the receive logic will latch data at the rising edge of SCLK. If SCLKPO and SCLKPH have the opposite values, the transmit logic will transmit data at the rising edge of SCLK, and the receive logic will receive data at the falling edge of SCLK (see Table 8-10 ANTAIOS datasheet).

In the master mode, previously to a transmission the user has to activate the chip select signal via Pin Controller. If the transmit FIFO contains data and SPI controller is enabled, the transmit logic will read data from the transmit FIFO and shift it out at each transmit edge (depending on SCLKPO and SCLKPH). The receive logic will receive data at the same time. The received data will be written to the receive FIFO. If data are still available in the transmit FIFO, the transmit logic will restart to transmit data. After transmission of the last bit, SPI_CS_0/1_N stays active until the user changes the setting in the Pin Controller. Due to independent generation of chip select signal via Pin Controller the time periods between chip select and first/last clock edges are much bigger than half or full SCLK cycle.

In the slave mode, if SPI_CS_0_N is activated and SPI_CLKIN starts running, the transmit logic will try to shift the data from the transmit FIFO even if the transmit FIFO is empty. The transmit FIFO underrun interrupt will be issued if the transmit FIFO is enabled. The receive logic will start receiving data at the same time and the received data will be written to the receive FIFO.

The bit-shifting sequence depends on the LSB setting in SPI control register 0 (Table 18-4) is from MSB or LSB regardless of the controller is in the master or slave mode. The SPICSPO in control register 0 (Table 18-4) decides the polarity of SPI_CS_0_N signal in Slave Mode. For Master Mode the SPI_CS0/1_N are controlled by Pin Controller (Table 4-1 and Table 18-5).

Figure 18-6 shows the Motorola SPI frame format for a single transfer, and Figure 18-7 shows the Motorola SPI frame format for a continuous transfer.

Figure 18-6 Single Transfer Using Motorola SPI Frame Format

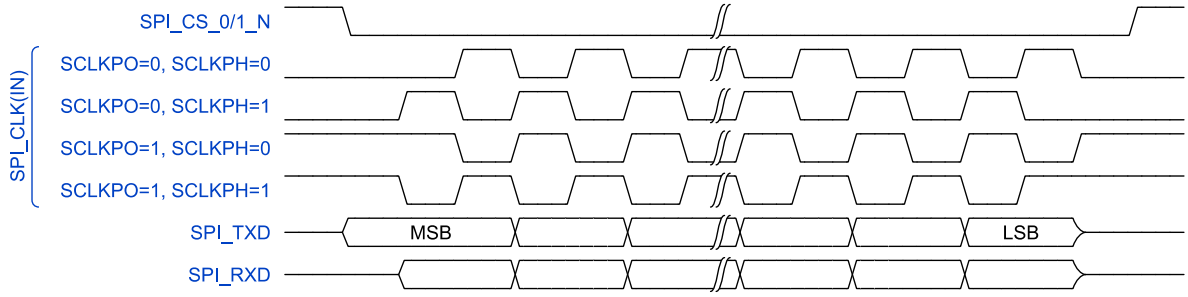
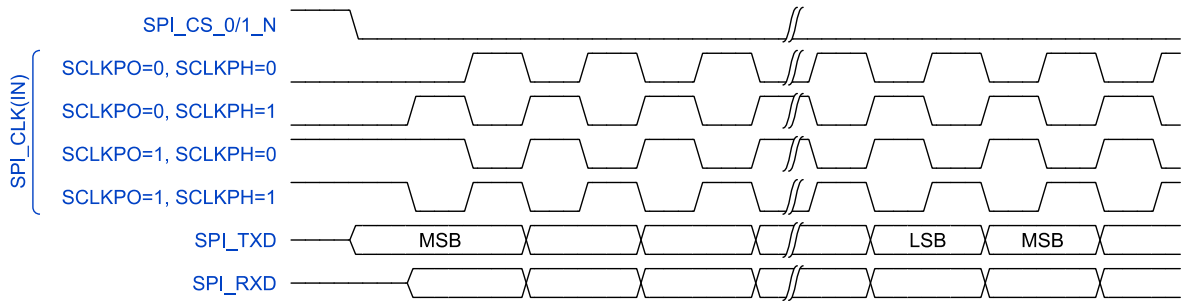


Figure 18-7 Continuous Transfer Using Motorola SPI Frame Format



In the SPI frame format, SCLK will stop toggling when it is idle.

18.2.3 Transmit/Receive FIFO Read/Write

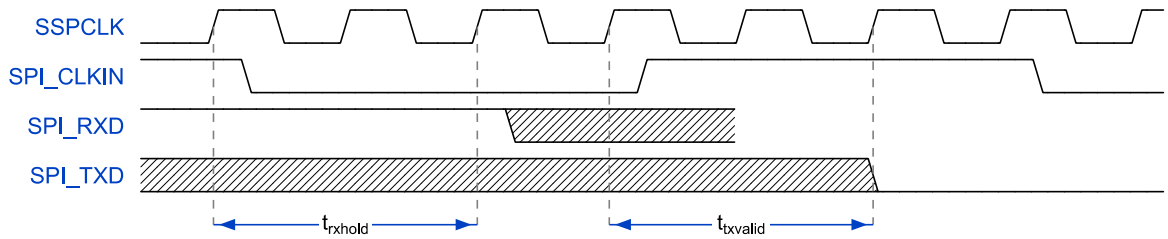
The transmit FIFO and receive FIFO occupy the same address space. When the data register is read, the data from the receive FIFO will be retrieved. When the data register is written, the written data will be stored in the transmit FIFO.

18.2.4 Setup/Hold Time

In the master mode, the SPI controller issues data at either the rising or falling edge of SPI_CLK without the hold time. That is, the output data will be changed whenever SPI_CLK changes the phase. Because the SPI controller generates the serial clock, the rising/falling edge can be predicted by the SPI controller. The received data will be latched by the SPI controller at the rising or falling edge of SPI_CLK and the setup/hold times will depend on the I/O cells.

In the slave mode, the serial clock and chip select are supplied externally. The SPI controller uses SSPCLK (internal 160 MHz clock) to synchronize the incoming signals, including SPI_CLKIN and SPI_CS_0_N. The SPI controller performs double synchronization so that the setup/hold times of incoming data are required. If data are latched at the falling edge of SPI_CLKIN and transmitted at the rising edge of SPI_CLKIN, with double synchronizations, the received data should have a hold time of at least two SSPCLK cycles (12,5ns) and the transmitted data should have an output valid time of at least two SSPCLK cycles, as shown in Figure 18-8.

Figure 18-8 Setup/Hold Time



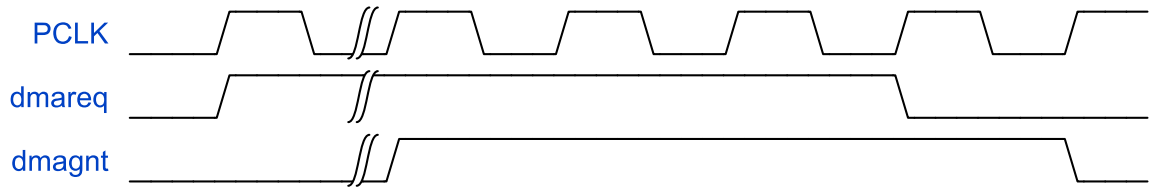
18.2.5 DMA Interface

The SPI controller provides separate DMA handshaking signals for transmit and receive FIFOs. If the transmitting DMA handshaking is enabled in the interrupt control register (Table 18-9) and the transmitting FIFO contains data equal to or less than the threshold (TFTHOD) set in the interrupt control register, then txdmareq signal will be asserted high to the DMA controller for the DMA transmission. The receiving DMA handshaking signal is completely independent of the DMA transmission handshaking signal. If the DMA reception is enabled in the interrupt control register and the receive FIFO contains data equal to or greater than the threshold (RFTHOD) set in the interrupt control register, then rxdmareq signal will be asserted high to the DMA controller for DMA reception.

The SPI controller employs two-stage synchronization. The DMA grant from the DMA controller will be synchronized by the SPI controller, and dmareq will be pulled low if the

DMA grant is sampled high. dmareq signal will not be reasserted high until the dmagnt signal is sampled low. The DMA handshaking protocol is the same for transmission and reception.

Figure 18-9 DMA Interface Timing



Peripherial Bus Clock (PCLK) from APB is 48 MHz.

18.2.6 Loopback Mode Testing

The loopback mode test (Table 18-4) takes effect only when the SPI controller is specified in the master mode. The received data from the external source is disconnected and the transmitted data are directly connected to the received data.

18.3 Register Definition

18.3.1 Summary of SPI Controller Registers

Please note that all registers should be read and written by 32-bit word accesses. Accesses with any other width may lead to unexpected errors.

This section describes all control and status registers in the SPI controller. The address column indicates the related address of the register in the hexadecimal format. The width column indicates the number of bits of the register. The type column indicates the access type of the register.

- RW: Read and write access
- RO: Read only
- RC: Read to clear

Table 18-3 SPI Controller Registers

Address Offset	Name	Width (Byte)	Type	Reset Value	Description
0x000	SPICR0	4	RW	0x0000_010C	SPI Control Register 0
0x004	SPICR1	4	RW	0x0007_8000	SPI Control Register 1
0x008	SPICR2	4	RW	0x0000_0002	SPI Control Register 2
0x00C	SPISTATUS	4	RO	0x0000_0002	SPI Status Register
0x010	INTR_CR	4	RW	0x0000_2100	Interrupt Control Register
0x014	INTR_STATUS	4	RO	0x0000_0008	Interrupt Status Register
0x018	TX_RX_DR	4	RW	-	Transmit/Receive Data Register

18.3.2 SPI Control Register 0

Abbr.: SPICR0 *Offset: 0x000*

This register controls the interface signals and frame generation.

Table 18-4 SPICR0 Register

Bit	Name	Type	Reset	Description
31..18	reserved	RO	0	-
17	CSFDBK	RW	0	feedback of chip select in master mode 0: chip select input is external signal SPI_CS_0_N 1: chip select input is fed back chip select output This chip select signal is an internal signal managed by the SPI controller, not the chip select signals generated via Pin Controller.
16	SCLKFDBK	RW	0	feedback of SPI clock in master mode 0: SPI clock input is external signal SPI_CLKIN 1: SPI clock input is fed back output signal SPI_CLK
15	SPICSPO	RW	0	chip select polarity for the SPI mode 0: active low chip select used 1: active high chip select used This bit will take effect only when the SPI frame format and Slave Mode is specified.
14..12	FFMT	RW	0	Frame format 000: Texas Instrument Synchronous Serial Port (SSP) 001: Motorola Serial Peripheral Interface (SPI) 010..111: reserved (unpredictable operation)
11	FLASH	RW	0	This bit indicates that the current application is SPI Flash. This bit will take effect only when the SPI frame format is specified.
10..8	reserved	RO	1	-
7	LBM	RW	0	Loopback mode 0: normal operation → SPI_TXD and SPI_RXD are unconnected 1: SPI_TXD connected to SPI_RXD internally (used for self-test)
6	LSB	RW	0	Bit sequence indicator 0: most significant bit (MSB) of data word transmitted/received first 1: least significant bit (LSB) of data word transmitted/received first This bit will take effect only when SPI frame format is specified.
5..4	reserved	RO	0	-
3..2	OPM	RW	3	Operation mode 00, 01: Slave mode (SSP, SPI) 10, 11: Master mode (SPI)
1	SCLKPO	RW	0	SCLK polarity (see section 18.2.2.2) 0: SCLK will remain low when SPI Controller is idle. 1: SCLK will remain high when SPI Controller is idle. This bit will take effect only when SPI frame format is specified.
0	SCLKPH	RW	0	SCLK phase (see section 18.2.2.2) 0: sampling data on odd numbered SCLK edges 1: sampling data on even numbered SCLK edges This bit will take effect only when SPI frame format is specified.

In the SPI master mode the chip select signals are generated via settings in the Pin Controller. The following table lists appropriate settings.

Table 18-5 settings for chip select polarity in SPI mode

Operation Mode	Signal	Direction	Description
Master	SPI_CS_0_N	output	only active low polarity is possible Pin Controller → offset 0x20: 0..active; 1..inactive
	SPI_CS_1_N		both polarities (active high/low) are possible Pin Controller → offset 0x21
Slave	SPI_CS_0_N	input	both polarities (active high/low) are possible SPICR0[SPICSPO] select polarity Pin Controller → offset 0x20: 1

18.3.3 SPI Control Register1

Abbr.: SPICR1 *Offset: 0x004*

This register defines the clock divider and the data length of transfer.

Table 18-6 SPICR1 Register

Bit	Name	Type	Reset	Description
31..23	reserved	RO	0	-
22..16	SDL	RW	7	Serial data length These bits define the bit length of a transmitted/received data word. The actual data length equals to these bits plus one.
15..0	SCLKDIV	RW	0x8000	SCLK divider These bits define the serial clock divider. They cannot be set as zeros except that SPI frame format is specified. Section 18.2.1 describes the details about the SCLK generation.

The minimum value of SDL should not be less than four, and the maximum value should not be greater than 31.

18.3.4 SPI Control Register2

Abbr.: SPICR2 *Offset: 0x008*

This register defines the SPI enable/disable and clear of FIFOs.

Table 18-7 SPICR2 Register

Bit	Name	Type	Reset	Description
31..9	reserved	RO	0	-
8	TXEN	RW	0	Transmit Function Enable When SPI frame format is specified, transmit and receive functions work independently. This bit controls the transmit function. 0: disable transmit function 1: enable transmit function For the SPI frame formats, this bit can only be changed when SPI controller is idle.
7	RXEN	RW	0	Receive Function Enable When SPI frame format is specified, transmit and receive functions will work independently. This bit controls the receive function. 0: disable receive function 1: enable receive function For the SPI frame formats, this bit can only be changed when SPI controller is idle.
6	SPIRST	RW1C	0	SPI Reset software reset of the SPI controller state machine 0: no effect 1: reset SPI controller This is a write only bit, reading always returns zero.
5..4	reserved	RO	0	-
3	TXFCLR	RW1C	0	Transmit FIFO Clear 0: no effect 1: clear data in TxFIFO This is a write only bit, reading always returns zero.
2	RXFCLR	RW1C	0	Receive FIFO Clear 0: no effect 1: clear data in RxFIFO This is a write only bit, reading always returns zero.
1	TXDOE	RW	1	Transmit Data Output Enable This bit is only valid when the SPI controller slave mode is specified. In the multi-slave system, it is possible for an SPI controller master to broadcast a message to all slaves within the system while ensuring that only one slave drives data onto its serial output line. In such a system, the rxd line from multiple slaves can be tied together. To operate in such a system, TXDOE may be cleared if the SPI slave is not supposed to drive the SPI_TXD line. 0: disable driver of SPI_TXD line 1: enable driver of SPI_TXD line
0	SPIEN	RW	0	SPI Enable 0: stop toggling serial data 1: enable transmitting/receiving data Note: If SPIEN is cleared during active operation, then SPI controller will stop transmitting or receiving after the current word is transmitted or received.

Writing SPIRST=1 will reset the SPI controller state machine even if the normal transmission or reception is in progress. Usually, this bit will be written when the software stops the SPI controller by setting SPIEN=0. For the slave mode of the SPI controller, the serial clock will be controlled by the external master and will be stopped under certain circumstances and will cause the SPI controller state machine to halt. Writing SPIRST=1 puts the SPI controller back to the idle state when carrying out the following operations.

18.3.5 SPI Status Register

Abbr.: SPISTATUS *Offset: 0x00C*

This register provides the SPI controller internal status for the user to take corresponding actions. It shows the raw status before interrupt is enabled.

Table 18-8 SPISTATUS Register

Bit	Name	Type	Reset	Description
31..18	reserved	RO	0	-
17..12	TFVE	RO	0	TxFIFO Valid Entry number of entries in Tx FIFO waiting for transmission
11..10	reserved	RO	0	-
9..4	FRVE	RO	0	RxFIFO Valid Entry number of entries in RxFIFO waiting for DMA or host processor read access
3	reserved	RO	0	-
2	BUSY	RO	0	Busy Indicator 0: SPI controller idle or disabled 1: SPI controller transmitting/receiving data
1	TFNF	RO	1	TxFIFO not full 0: TxFIFO full 1: possible to write data by DMA or host processor
0	RFF	RO	0	RxFIFO full 1. RxFIFO full 0: possible to receive data by SPI

18.3.6 Interrupt Control Register

Abbr.: INTR_CR *Offset: 0x010*

This register controls the interrupt and DMA request generation of the SPI controller.

Table 18-9 INTR_CR Register

Bit	Name	Type	Reset	Description
31..17	reserved	RO	0	-
16..12	TFTHOD	RW	2	TxFIFO Threshold 0: interrupt disabled 1..31: If the valid data in the TxFIFO is equal to or less than the actual threshold, the DMA request and/or the interrupt will be asserted.
11..7	RFTHOD	RW	2	RxFIFO Threshold 0: interrupt disabled 1..31: If the valid data in the RxFIFO is equal to or greater than the actual threshold, the DMA request and/or the interrupt will be asserted.
6	reserved	RW	0	-
5	TFDMAEN	RW	0	Transmit DMA Request Enable 0: Tx DMA request disabled 1: Tx DMA request is issued when TxFIFO threshold is hit
4	RFDMAEN	RW	0	Receive DMA Request Enable 0: Rx DMA request disabled 1: Rx DMA request is issued when RxFIFO threshold is hit
3	TFTHIEN	RW	0	Transmit FIFO Threshold Interrupt Enable 0: interrupt disabled 1: interrupt is issued when the valid data in the TxFIFO is equal to or less than the TxFIFO threshold value
2	RFTHIEN	RW	0	Receive FIFO Threshold Interrupt Enable 0: interrupt disabled 1: interrupt is issued when the valid data in the RxFIFO is equal to or greater than the RxFIFO threshold value
1	TFURIEN	RW	0	Transmit FIFO Underrun Interrupt Enable 0: interrupt masked, even when the TxFIFO underrun occurs 1: interrupt is issued when TxFIFO underrun occurs
0	RFORIEN	RW	0	Receive FIFO Overrun Interrupt Enable 0: interrupt masked, even when the RxFIFO overrun occurs 1: interrupt is issued if RxFIFO overrun occurs

18.3.7 Interrupt Status Register

Abbr.: INTR_STATUS *Offset: 0x014*

This register shows the current interrupt status.

Table 18-10 INTR_STATUS Register

Bit	Name	Type	Reset	Description
31..4	reserved	RO	0	-
3	TFTHI	RO	1	Transmit FIFO Threshold Interrupt 0: valid data in the TxFIFO is larger than TxFIFO threshold 1: valid data in the TxFIFO is equal to or less than the TxFIFO threshold This bit will be automatically cleared.
2	RFTHI	RO	0	Receive FIFO Threshold Interrupt 0: valid data in the RxFIFO is less than RxFIFO threshold 1: valid data in the RxFIFO is equal to or greater than the TxFIFO threshold This bit will be automatically cleared
1	TFURI	RC	0	Transmit FIFO Underrun Interrupt 0: bit is cleared after read access by user 1: transmit logic tries to fetch data from the empty TxFIFO
0	RFORI	RC	0	Receive FIFO Overrun Interrupt 0: bit is cleared after read access by user 1: receive logic tries to write data into the full RxFIFO

18.3.8 SPI Transmit/Receive Data Register

Abbr.: TX_RX_DR *Offset: 0x018*

The SPI controller separates transmit and receive FIFOs, which are 32-bit wide. The transmitted and received data occupy the same address space. The write operation writes data to the TxFIFO and the read operation reads data from the RxFIFO. If the size of a serial data length is less than 32 bits, data will be automatically right justified during reception or transmission. If the size of a serial data length is larger than 32 bits, the data will be processed from LSB to MSB based on the 32-bit data width during receiving or transmitting data.

18.4 Programming Guidelines

This section provides the initialization sequence of the SPI controller. First of all, the user has to determine whether to use the master or slave mode for the application. Once the mode is determined, please follow the appropriate SPI controller initialization sequence provided in the following sections.

18.4.1 Initialization in Master Mode

The following steps outline the initialization sequence of the data transmission/reception to/from slave in the master mode:

1. set SPICR0 (Table 18-4):
    a. frame format SPI (FFMT=001)
    b. bit sequence indicator (LSB)
    c. operation mode master (OPM=10)
    d. SPI_CLK polarity (SCLKPO)
    e. SPI_CLK phase (SCLKPH)
2. activate pin controller signals SPI_CS_0_N=0 or SPI_CS_1_N (Table 4-1 and Table 18-5)
3. set SPICR1 (Table 18-6):
    a. serial data length (SDL)
    b. clock divider (SCLKDIV)
4. fill transmit FIFO (TxRxDR)
5. config INTR_CR (Table 18-9), if required:
    a. FIFO thresholds (TFTHOD, RFTHOD)
    b. transmit/receive interrupt enable (TFTHIEN, RFTHIEN, TFURIEN, RFORIEN)
    c. DMA requests (TFDMAEN, TRFDMAEN)
6. set SPICR2 (Table 18-7):
    a. transmit/receive function enable (TXEN=1, RXEN=1)
    b. transmit data output enable (TXDOE=1)
    c. SPI controller enable (SPIEN=1)
7. if TxFIFO empty, and
    a. SDL > 31:
        - disable SPI controller (SPIEN=0)
        - go to Step 4 for the next transfer
    b. SDL ≤ 31:
        - SPI controller stops providing SPI_CLK until new transmit data available in transmit FIFO

After the initialization in the master mode, the SPI controller will start transmitting/receiving data without user intervention. The SPI controller will report the FIFO status or transmission/reception to/from the user if the corresponding interrupt is enabled. The SPI controller will request DMA to transfer data if the DMA is enabled. If all the data in the transmit FIFO are transmitted, the SPI controller will report the transmit data empty if the interrupt is enabled. Please note that the SPI controller will report that the transmit FIFO is empty before the transmission is complete and while the transmit logic is still busy transmitting the last data out. The user has to monitor the busy flag of the SPI controller to confirm the completion of the transmission.

18.4.2 Initialization in Slave Mode

The following steps show the initialization sequence for transmitting/receiving data to/from the master in the slave mode:

1. set SPICR0 (Table 18-4):
    a. frame format (FFMT)
    b. operation mode (OPM=00)
    c. if frame format SPI: SPI_CS_0_N polarity (SPICSPO)
    d. if frame format SPI: SPI_CLK polarity (SCLKPO)
    e. if frame format SPI: SPI_CLK phase (SCLKPH
2. deactivate pin controller signals SPI_CS_0_N=1 (Table 4-1 and Table 18-5)
3. set SPICR1 (Table 18-6):
    a. serial data length (SDL)
4. fill transmit FIFO (TxRxDR), if response data for SPI master necessary
5. config INTR_CR (Table 18-9), if required:
    a. FIFO thresholds (TFTHOD, RFTHOD)
    b. transmit/receive interrupt enable (TFTHIEN, RFTHIEN, TFURIEN, RFORIEN)
    c. DMA requests (TFDMAEN, TRFDMAEN)
6. set SPICR2 (Table 18-7):
    a. if frame format SPI: transmit/receive function enable (TXEN=1, RXEN=1)
    b. transmit data output enable (TXDOE=1), if allowed
    c. SPI controller enable (SPIEN=1)

In the slave mode, the external master controls the action of the SPI controller. The SPI controller will operate according to the SPI_CS_0_N and serial clock input.

18.4.3 Disable/Enable Program Sequence

When SPI controller is used in a disabling sequence and then enabling, please note that:

1. The transmit FIFO must be cleared before disabling SPI controller.
2. The receive FIFO must be cleared before enabling SPI controller.

18.4.4 SPI Flash Controller Write Command Program Sequence

1. set SPICR0 (Table 18-4):
    a. frame format SPI (FFMT=001)
    b. SPI Flash application (FLASH=1)
    c. operation mode master (OPM=10)
    d. SPI_CLK polarity (SCLKPO)
    e. SPI_CLK phase (SCLKPH)
2. set SPICR1 (Table 18-6):
    a. serial data length (SDL)
    b. clock divider (SCLKDIV)
3. set SPICR2 (Table 18-7):
    a. transmit function enable (TXEN=1)
    b. transmit data output enable (TXDOE=1)
    c. SPI controller enable (SPIEN=1)
4. activate pin controller signal SPI_CS_0_N=0 or SPI_CS_1_N=0 (Table 4-1 and Table 18-5)
5. config INTR_CR (Table 18-9), if required:
    a. FIFO thresholds (TFTHOD)
    b. transmit interrupt enable (TFTHIEN, TFURIEN)
    c. DMA requests (TFDMAEN)
6. fill transmit FIFO (TxRxDR)
7. check TFVE and BUSY in SPI status register (Table 18-8)
   if data transfer completed (TFVE=0 and BUSY=0):
    a. set SPICR2 (Table 18-7):
        - transmit function disable (TXEN=0)
        - SPI controller disable (SPIEN=0)
    b. deactivate pin controller signal SPI_CS_0_N=1 or SPI_CS_1_N=1 (Table 4-1 and Table 18-5) to stop the write command
8. check the SPI FLASH status register
   SPI controller cannot issue a new command unless the write command has completed.

18.4.5 SPI Flash Controller Read Command Program Sequence

1. set SPICR0 (Table 18-4):
    a. frame format SPI (FFMT=001)
    b. SPI Flash application (FLASH=1)
    c. operation mode master (OPM=10)
    d. SPI_CLK polarity (SCLKPO)
    e. SPI_CLK phase (SCLKPH)
2. set SPICR1 (Table 18-6):
    a. serial data length (SDL)
    b. clock divider (SCLKDIV)
3. set SPICR2 (Table 18-7):
    a. clear transmit and receive FIFO (TXFCLR=1, RXFCLR=1)
4. fill read command code into the transmit FIFO (TxRxDR)
5. set SPICR2 (Table 18-7):
    a. transmit and receive function enable (TXEN=1, RXEN=1)
    b. transmit data output enable (TXDOE=1)
    c. SPI controller enable (SPIEN=1)
6. activate pin controller signal SPI_CS_0_N=0 or SPI_CS_1_N=0 (Table 4-1 and Table 18-5)
7. config INTR_CR (Table 18-9), if required:
    a. FIFO thresholds (RFTHOD)
    b. receive interrupt enable (RFTHIEN, RFORIEN)
    c. DMA requests (RFDMAEN)
8. after all data received (BUSY=0):
    a. set SPICR2 (Table 18-7):
        - transmit and receive function disable (TXEN=0, RXEN=0)
        - SPI controller disable (SPIEN=0)
    b. deactivate pin controller signal SPI_CS_0_N=1 or SPI_CS_1_N=1 (Table 4-1 and Table 18-5) to stop the read command

19 UART

Offsets: UART1: 0x78800000 (APB2 ID1)
UART2: 0x70000000 (APB1 ID0)

19.1 Overview

The ANTAIOS contains 2 Universal Asynchronous Receiver Transmitter (UART). They are standards 16550 UARTs with enhanced features.

The main features are

- 5/6/7/8 data bits
- 1/1.5/2 stop bits
- none/odd/even/stick parity
- register/FIFO mode
- line break generation & detection
- programmable baud rate generator
- fully prioritized interrupt system controls
- status reporting capabilities
- Modem control functions
- Loopback mode
- DMA mode

The enhanced features are

- High speed mode for higher baud rates up to 12 Mbit/s
- ID field
- 32 Byte FIFO with 16C650 DMA behavior
- module controlled activation/deactivation for RTS
- high speed mode for higher baud rates
- selectable interrupt polarity
- extended timeout econtrol/detection
- up to 2 configurable ETX characters
- Receive Byte Counter
- enable/disable Receiver
- Transmitter interrupt can be “THR empty” or “TSR empty”

19.2 Memory Map and Register Definition

Table 19-1 describes all registers of a UART. The address column indicates the related address of the register in the hexadecimal format. The access column indicates the access type of the register.

- R: Read only
- W: Write access
- RW: Read and write access

Take care when reading registers, because reading of some registers cause an internal function.

Table 19-1 Summary of UART Registers

Name	Addr.	Access	Description
Standard 16550/16650 Registers			
RHR	+0	R	Revceive Hold Register
THR	+0	W	Transmit Hold Register
IER	+4	RW	Interrupt Enable Regiser
IIR	+8	R	Interrupt Identification Register
FCR	+8	W	FIDO Control Register
LCR	+C	RW	Line Control Register
MCR	+10	RW	Modem Control Register
LSR	+14	R	Line Status Register
XFR	+14	W	eXtended Feature Register
MSR	+18	RW	Modem Status Register
SPR	+1C	RW	Scratchpad Register
DIVLATL[6]	+0	RW	Divisor Latch Low Byte
DIVLATH[6]	+0	RW	Divisor Latch High Byte
Enhanced UART Registers			
IDENT	+20	R	Device Ident Code
EFR	+24	RW	Enhanced Feature Register
TCFG	+28	RW	Timeout Timer Configuration
TOT1L	+2C	RW	Timeout Timer 1, Lowbyte
TOT1H	+30	RW	Timeout Timer 2, Highbyte
TOT2L	+34	RW	Timeout Timer 2, Lowbyte
TOT2H	+38	RW	Timeout Timer 2, Highbyte
ETX1	+3C	RW	ETX Character 1
ETX2	+40	RW	ETX Character 2
BCC	+44	R	Byte Counter Current
BCL	+44	W	Byte Counter Length

[6] To access registers DIVLATH and/or DIVLATL set Bit 7 of register LCR!

Table 19-2 UART Quick Overview

Register	Bit Position							
	7	6	5	4	3	2	1	0
RHR	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0
THR	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0
IER	byte counter interrupt	etx char interrupt	t2 interrupt	t1 interrupt	modem status interrupt	line status interrupt	transmit holding register	receive holding register
IIR	FIFO en /BC irq	FIFO en /ETX irq	0 / T2 irq	0 / T1 irq	INT priority 2	INT priority 1	INT priority 0	INT status
FCR	RCVR trigger	RCVR trigger	XMIT trigger	XMIT trigger	DMA mode	XMIT FIFO reset	RCVR FIFO reset	FIFO enable
LCR	divisor latch enable	set break	set parity	even parity	parity enable	stop bits	word length 1	word length 0
MCR	0	0	0	loop back	/OP2	/OP1	/RTS	/DTR
LSR	data error	trans. empty	trans. holding empty	break indica-tion	framing error	parity error	overrun error	receive data ready
XFR	0	0	invertRTS	0	0	0	auto RTS	0
MSR	/DCD	/RI	/DSR	/CTS	delta /CD	delta /RI	delta /DSR	delta /CTS
SPR	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0
DIVLATL	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0
DIVLATH	bit-15	bit-14	bit-13	bit-12	bit-11	bit-10	bit-9	bit-8
IDENT	0	0	0	0	0	0	1	1
EFR	BCount mode	EXT mode 1	ETX mode 0	INT polarity	HS mode	THR is TSR	dis RX	650 mode
TCFG	0	0	0	rearm T1 / T2	TOT2 enable	TOT2 base	TOT1 enable	TOT1 base
TOT1L	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0
TOT1H	bit-15	bit-14	bit-13	bit-12	bit-11	bit-10	bit-9	bit-8
TOT2L	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0
TOT2H	bit-15	bit-14	bit-13	bit-12	bit-11	bit-10	bit-9	bit-8
ETX1	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0
ETX2	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0
BCC	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0
BCL	bit-7	bit-6	bit-5	bit-4	bit-3	bit-2	bit-1	bit-0

19.2.1 Standard Mode Register Description

19.2.1.1 Receive Hold Register

Abbr.: RHR Offset: 0x00

This register contains the received data byte. This register is read-only.

19.2.1.2 Enhanced Register Description

Abbr.: THR Offset: 0x00

The data byte which shall be transmitted is written here. This register is write-only.

19.2.1.3 Interrupt Enable Register

Abbr.: IER Offset: 0x04

Enables (1) or disables (0) the eight different interrupt sources.
If an interrupt is disables there will also be no entry in the IIR.

19.2.1.4 Interrupt Identify Register

Abbr.: IIR Offset: 0x08

For standard UART functions the interrupt level is stored if an interrupt has occurred. The priority of the interrupt levels are as in Table 3-1.

Table 19-3 Priority of interrupt levels

PRI	Bit 3	Bit 2	Bit 1	Bit 0	Source of Interrupt
-	0	0	0	1	No pending interrupt
1	0	1	1	0	LSR (Receiver Line Status Register)
2	1	1	0	0	RXRDY (Receive data timeout)
3	0	1	0	0	RXRDY (Received data ready)
4	0	0	1	0	TXRDY (Transmitter Hold Register empty)
5	0	0	0	0	MSR (Modem Status Register)

In standard UART mode , Bit 7 and Bit 6 are set to 1 when the FIFOs are enabled, otherwise they are set to 0. If any of the functions “bytecounter”, “etx character” or “extended timeout” is activated Bit 7 … 4 have the meaning as in Table 19-4.

Table 19-4 Source of Interrupts

Bit7	Bit6	Bit5	Bit4	Source of Interrupt
1	-	-	-	Byte Counter
-	1	-	-	ETX character
-	-	1	-	Timeout 2
-	-	-	1	Timeout 1

19.2.1.5 FIFO Control Regsiter

Abbr.: FCR Offset: 0x08

Contains the controls of the FIFOs.

Table 19-5 FIFO Control Register

Bit	Name	Description
0	fifo_enable	Enable the FIFO mode. 0: FIFOs are disabled. 1: FIFOs are enabled.the Size of the TX/RX FIFO is 16/32 bytes, depending on **mode650** bit in register **EFR**. This bit *must* be 1 when other FCR bits are written to or they will not be programmed. The contents of the FIFO will be erased when FIDO mode is disabled.
1	rcvr_fifo_reset	Reset the receiver FIFO. 0: No receiver FIFO reset. 1: The receiver FIFO is resetted.
2	xmit_fifo_reset	Reset the transmitter FIFO. 0: No transmitter FIFO reset. 1: The transmitter FIFO is resetted.
3	dma_mode	Selection of DMA mode. 0: RXRDY: : If there is at least 1 character in the RCVR FIFO or RCVR holding register, the RXRDY pin will be low active. Once it is activated the RXRDY pin will go inactive when there are no more characters in the FIFO or holding register. TXRDY: If there are no characters in the XMIT FIFO or XMIT holding register, the TXRDY pin will be low active. Once it is activated the TXRDY pin will go inactive after the first character is loaded into the XMIT FIFO or holding register. 1: RXRDY: When the trigger level or the timeout has been reached, the RXRDY pin will go low active. Once it is activated it will go inactive when there are no more characters in the FIFO or holding register. TXRDY: When there are no characters in the XMIT FIFO, the TXRDY pin will go low active. This pin will become inactive when the XMIT FIFO is completely full.
4, 5	xmit_fifo_trigger	Transmitters FIFO trigger level (only on **mode650**!) 00: Trigger level is one Byte. 01: Trigger level is 8 Bytes. 10: Trigger level is 16 Bytes. 11: Trigger level is 28 Bytes. If the trigger level is met a transmitter empty interrupt will be generated.
6, 7	rcvr_fifo_trigger	Receivers FIFO trigger level. 00: Trigger level is 1/8 Bytes. 01: Trigger level is 4/16 Bytes. 10: Trigger level is 8/24 Bytes. 11: Trigger level is 14/28 Bytes. If the trigger level is met a receiver interrupt will be generated. However the FIFO will continue loading until it is full.

Value of register after reset will be 0x00.

19.2.1.6 Line Control Register

Abbr.: LCR Offset: 0x0C

Contains the transmission parameter of the UART.

Table 19-6 Line Control Register

Bit	Name	Description
0, 1	word_length	Data size selection. 00: Data size is 5 Bit. 11: Data size is 6 Bit. 10: Data size is 7 Bit. 11: Data size is 8 Bit.
2	stop_bits	Stop Bit selection. 0: Stop Bit length is 1 Bit 1: Stop Bit length is 1.5 Bit if **word_length** is 5 Bit. Stop Bit length is 2 Bits if **word_length** is 6, 7 or 8 Bits.
3 … 5	set parity even_parity parity_enable	Parity selection. xx0: No parity 001: Odd parity 011: Even parity 101: Force parity 1 111: Force parity 0
6	set_break	Force Break Condition 0: No breal condition 1: The transmitter is forced to low level to indicate a breal condition at the receiver.
7	divisor_latch_ enable	Enable Divisor Latches 0: The divisor latches are disabled. 1: The divisor latches are enabled. A write address 0 or 1 will cause a write to a divisor latch.

Value of register after reset will be 0x00.

19.2.1.7 Line Status Register

Abbr.: LSR Offset: 0x14

Contains the status of the transmitter and receiver.

Table 19-7 Line Status Register

Bit	Name	Description
0	receive_data_ ready	Receive Data Ready. 0: No data in receive holding register of FIFO or SRAM. 1: Data has been received and is saved in the RHR, FIFO, or SRAM.
1	overrun_error	Overrun Error 0: No overrun error. 1: A data overrun error occurred in the receive shift register. In this case the previous data in the shift register is overwritten.
2	parity_error	Parity Error 0: Np parity error. 1: The received character does not have correct parity information. In the FIFO mode this error is associated with the character available for reading.
3	framing_error	Framing Error 0: No framing error. 1: The received character does not have a valid stop bit. In the FIFO mode this error is associated with the character available for reading.
4	break_indication	Break Indication 0: No break indication. 1: The receiver received a break signal (RxD was logic 0 for at least one character frame). In FIFO mode this condition is only stored once when detected.
5	transmitter_ hold_empty	Transmitter Hold Register Empty 0: Transmit hold register contains a data character. 1: Transmit hold register is empty, indicating that it is ready to accept a new data character for transmission.
6	transmitter_empty	Transmitter Empty 0: Transmit hold register OR transmit shift register contain a data character. 1: Transmit hold register AND transmit shift register are both empty.
7	data_error	FIFO Data Error 0: No error. 1: At least one parity error, framing error or break indication is in the current FIFO data. This bit is cleared when there are no remaining LSR errors in the receiver FIFO.

Value of register after reset will be 0x60.

19.2.1.8 eXtended Feature Register

Abbr.: XFR Offset: 0x14

Defines the extended features of the UART introduced by the C650 UART from EXAR.

Table 19-8 eXtended Feature Register

Bit	Name	Description
0		Reserved
1	invert_RTS_ polarity	RTS Polarity 0: Idle state of RTS is high level. 1: Idle state of RTS is low level.
2 … 4		Reserved
5	auto_RTS_ mode	Automatic RTS Generation 0: RTS is driven manually by setting/clearing Bit 1 in register MCR. 1: RTS is driven automatically when transmitting data.
6, 7		Reserved

Value of register after reset will be 0x00.

Attention: If auto_RTS_mode is used the RS485 interface part has to be designed carefully, because the RTS lead time is fixed to 83.3ns. This may direct to a timing problem when running on 12Mbaud and using a separate driver (e.g. ADM 2486 doesn't generate a timing problem).

19.2.1.9 Modem Control Register

Abbr.: MCR Offset: 0x10

This register controls the modem.

Table 19-9 Modem Control Register

Bit	Name	Description
0	/DTR	Data Terminal Ready 0: Force DTR to logic 1 1: Force DTR to logic 0
1	/RTS	Ready To Send (**Automatic RTS generation (XFR register) will override this Bit.**) 0: Force RTS to logic 1 1: Force RTS to logic 0
2	/OP1	User Output 1 0: Force OP1 to logic 1 1: Force OP1 to logic 0
3	/OP2	User Output 2 0: Force OP2 to logic 1 1: Force OP2 to logic 0
4	loopback	Loopback Mode 0: Loopback mode is deactivated. 1: Loopback mode is activated. The four modem control signals (DTR, RTS, OP1, OP2) are externally disconnected to their inactive state and internally connected to the modem status signals (DSR, CTS, RI, DCD). Moreover the same is done with RxD and TxD.
5 … 7		Reserved

Value of register after reset will be 0x00.

19.2.1.10 Modem Status Register

Abbr.: MSR Offset: 0x18

The register contains the status of the modem.

Table 19-10 Modem Status Register

Bit	Name	Description
7	/DCD	Data Carrier Detect This bit reflects the complement of the /DCD signal. However in loop-back mode this bit is equivalent to the /OP2 bit in the MCR register.
6	/RI	Ring Indication This bit reflects the complement of the /RI signal. However in loop-back mode this bit is equivalent to the /OP1 bit in the MCR register.
5	/DSR	Data Set Ready This bit reflects the complement of the /DSR signal. However in loop-back mode this bit is equivalent to the /CTS bit in the MCR register.
4	/CTS	Clear To Send This bit reflects the complement of the /CTS signal. However in loop-back mode this bit is equivalent to the /RTS bit in the MCR register.
3	delta_DCD	Change of signal /DCD If this bit is set the /DCD input has changed state since the last time it was read. A modem status interrupt will be generated.
2	delta_RI	Change of signal /RI If this bit is set the /RI input has changed state since the last time it was read. A modem status interrupt will be generated.
1	delta_DSR	Change of signal /DSR If this bit is set the /DSR input has changed state since the last time it was read. A modem status interrupt will be generated.
0	delta_CTS	Change of signal /CTS If this bit is set the /CTS input has changed state since the last time it was read. A modem status interrupt will be generated.

Value of register after reset will be 0xX0.

19.2.1.11 Scratchpad Register

Abbr.: SPR Offset: 0x1C

A register for user defined information, which will not be altered by the chip.

19.2.1.12 Divisor Latch Registers

Abbr.: DIVLATH *Offset: 0x00 (see note below!)*
Abbr.: DIVLATL *Offset: 0x04*

These registers adjust the Baud rate of the UART, whereas a value of 0 disables the Baud rate generation. The following equations are used to calculate the necessary value for a desired baud rate.

$$DIVLAT = \frac{48MHz}{baudrate * 16}$$ if hs_mode = 0 (register EFR, default)

$$DIVLAT = \frac{48MHz}{baudrate * 4}$$ if hs_mode = 1

To access registers DIVLATH and/or DIVLATL set Bit 7 of register LCR.
The following both tables show the configuration for different baud rates.
If the baud rate error is unequal zero then it is specified. It is calculated as follows:

$$Baud\ Rate\ Error = \frac{|desired\ baud\ rate - actual\ baud\ rate|}{desired\ baud\ rate} \cdot 100\%$$

Table 19-11 Baud Rate Configuration (Standard Baud Rates)

Baud Rate	EFR.hs_mode = 0		EFR.hs_mode = 1	
	DIVLAT	Baud Rate Error	DIVLAT	Baud Rate Error
50 bit/s	60000		240000	
110 bit/s	27273	0,001 %	109091	0,0001 %
150 bit/s	20000		80000	
300 bit/s	10000		40000	
1.2 kbit/s	2500		10000	
2.4 kbit/s	1250		5000	
4.8 kbit/s	625		2500	
9.6 kbit/s	313	0,1597 %	1250	
19.2 kbit/s	156	0,1603 %	625	
38.4 kbit/s	78	0,1603 %	313	0,1597 %
57.6 kbit/s	52	0,1603 %	208	0,1603 %
115.2 kbit/s	26	0,1603 %	104	0,1603 %
230.4 kbit/s	13	0,1603 %	52	0,1603 %
460.8 kbit/s			26	0,1603 %
921.6 kbit/s			13	0,1603 %

The last both baud rates aren't adjustable without High Speed Mode, because of a Baud Rate Error greater than three percent.

Table 19-12 Baud Rate Configuration (PROFIBUS Baud Rates)

Baud Rate	EFR.hs_mode = 0		EFR.hs_mode = 1	
	DIVLAT	Baud Rate Error	DIVLAT	Baud Rate Error
31.25 kbit/s	96		384	
45.45 kbit/s	66	0.01 %	264	0.01 %
93.75 kbit/s	32		128	
187.5 kbit/s	16		64	
500 kbit/s	6		24	
1.5 Mbit/s	2		8	
3 Mbit/s	1		4	
6 Mbit/s			2	
12 Mbit/s			1	

The last both baud rates aren’t adjustable without High Speed Mode, because of a Baud Rate Error greater than three percent.

19.2.2 Enhanced Mode Register Description

19.2.2.1 Enhanced Feature Register

Abbr.: EFR *Offset: 0x24*

This register defines the enhanced features of the UART.

Table 19-13 Enhanced Feature Register

Bit	Name	Description
7	bcount_ mode	Byte Counter Mode 0 Rx Byte Counter inactive 1 Rx Byte Counter active
6, 5	etx_mode	End of Text Mode 00 No End of Text Characters defined 10 ETX1 character defined 11 ETX1 and ETX2 character defined
4	int_pol	Interrupt Polarity 0 Interrupt is active low. 1 Interrupt is active high.
3	hs_mode	High Speed Mode 0 Oversampling factor is 16. 1 Oversampling factor is 4.
2	THR_is_ TSR	THR interrupt is TSR interrupt 0 Interrupt is generated when Transmitter Hold Register empty. 1 Interrupt is generated when Transmitter Shift Register empty.
1	dis_rx	Disable Receiver 0 Receiver is enabled. 1 Receiver is disabled.
0	mode650	16650 FIFO mode 0 16550 FIFO mode. FIFO size is 16 bytes, trigger level and DMA mode show the behaviour of a 16550 compatible device. 1 16650 FIFO mode. FIFO size is 32 bytes, trigger level and DMA mode show the behaviour of a 16650 compatible device.

Value of register after reset will be 0x10.

19.2.2.2 Timeout Timer Configuration

Abbr.: TCFG *Offset: 0x28*

This register contains the configuration for the advanced timeout.

Table 19-14 Timeout Timer Configuration Register

Bit	Name	Description
7 … 5		Reserved
4	rearmT1T2	Rearm Timeout Timers 1 and 2 A logical 1 will rearm both timers. After rearming the timer the bit is cleared by the device.
3	tot2en	Enable Timeout Timer 2 0 Timeout Timer 2 is disabled. 1 Timeout Timer 2 is enabled.
2	tot2base	Time Base of Timeout Timer 2 0 Timeout Timer 2 tick is TBIT. 1 Timeout Timer 2 tick is 1ms.
1	tot1en	Enable Timeout Timer 1 0 Timeout Timer 1 is disabled. 1 Timeout Timer 1 is enabled.
0	tot1base	Time Base of Timeout Timer 1 0 Timeout Timer 1 tick is TBIT. 1 Timeout Timer 1 tick is 1ms.

Value of register after reset will be 0x00.

19.2.2.3 Timeout Timer Compare Value Registers

Abbr.: TOT1L *Offset: 0x2C*
Abbr.: TOT1H *Offset: 0x30*
Abbr.: TOT2L *Offset: 0x34*
Abbr.: TOT2H *Offset: 0x38*

This registers contain the compare values for the timeout-timer 1/2. In order to prevent unpredictable timeout behavior a new value must only be written when the timer is disabled. A value of 0 is not permissible and will lead to no timeout (timer is inactive).

19.2.2.4 ETX Character Registers

Abbr.: ETX1 *Offset: 0x3C*
Abbr.: ETX2 *Offset: 0x40*

This registers contain the RX End-of-Text character(s). If ETX mode is active and the character(s) is /are detected an interrupt is generated.

19.2.2.5 Byte Counter Current Register

Abbr.: BCC *Offset: 0x44*

This register contains the current value of the RX byte counter. If BCC is read, BCC is cleared to zero. The BCC will also be cleared, if RX-FIFO is reset.

19.2.2.6 Byte Counter Length Register

Abbr.: BCL *Offset: 0x44*

This register contains the length for the RX byte counter. If the length is hit an interrupt is generated. If BCL is written (e.g. a new value) BCC is cleared to zero.

20 I²C Bus Interface Controller

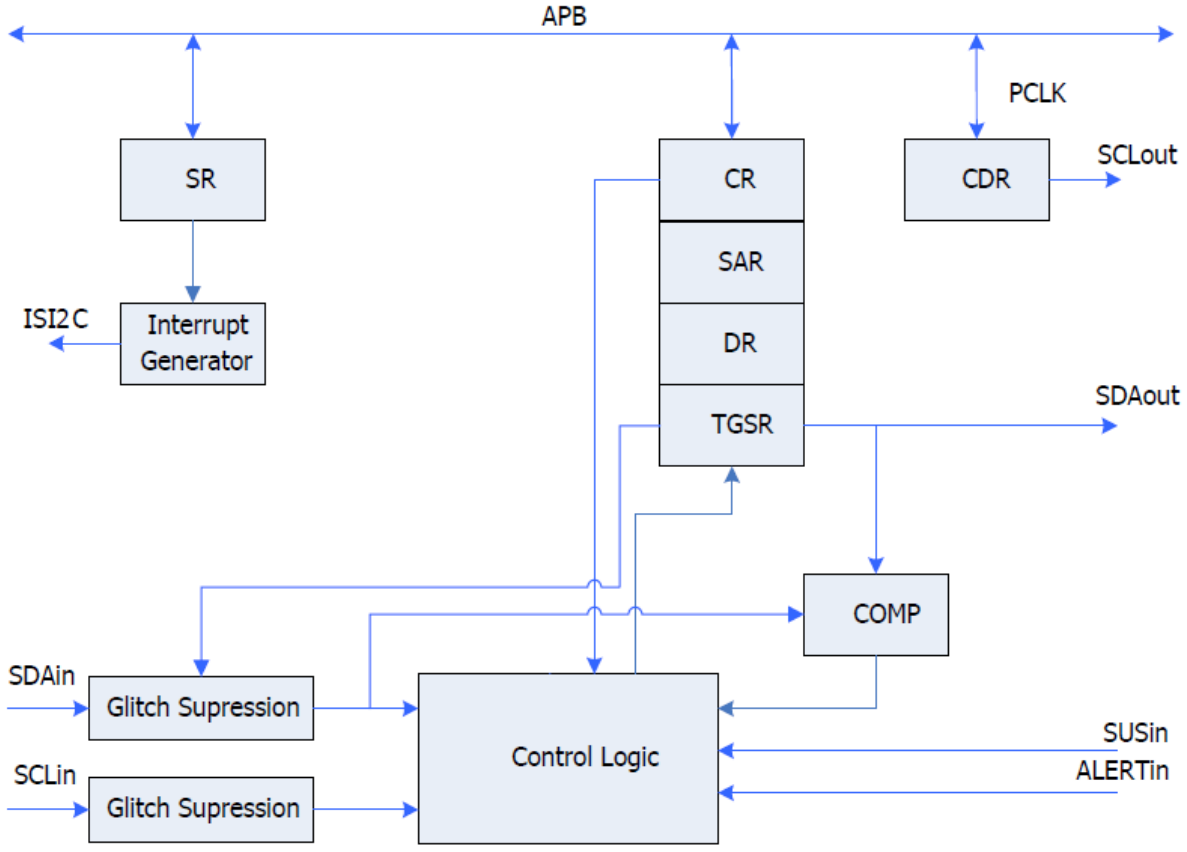
Offset: I²C1 0x71000000 (APB1 ID2)
I²C2 0x71800000 (APB2 ID3)

The ANTAIOS I²C bus interface controller allows the host processor to serve as a master or slave in the I²C bus. Data are transmitted to and received from the I²C bus via a buffered interface.

- Supports standard and fast modes by programming clock division register
- Supports 7-bit, 10-bit, and general-call addressing modes
- Glitch suppression throughout debounce circuit
- Programmable slave address
- Supports master-transmit, master-receive, slave-transmit, and slave-receive modes
- Supports multi-master mode (if the master losses the arbitration, it stops driving SDA but don't change into slave mode)
- Includes general-call address detection in slave mode
- START byte procedure: Not available

20.1 Overview

Figure 20-1 Block diagram of I²C Controller



20.1.1 Register Files

The register files, which contain the control register, slave address register, clock divider, status, data, setup/hold time, and glitch suppression, bus monitor registers, SM bus control register, maximum timeout register, minimum timeout register, master extend time register, and slave extend time register are the read/write or read-only or read/clear from the host processor throughout the AMBA 2.0 APB bus protocol.

20.1.2 Control Logic

The control logic is used to detect SCLin and SDAin in the I²C bus and decide the status of the bus.

20.1.3 SCLout Generator

The SCLout generator accepts the APB bus clock (48MHz) and divides the value in the clock division register and multiplies 2 to generate the SCLout on the I²C bus.

20.1.4 Debounce Circuit

The debounce circuit is used as the glitch suppression logic. Glitches are suppressed according to the GSR* internal bus clock period, where GSR is Bits13..10 of the register TGSR (Table 20-7).

20.2 Memory Map and Register Definition

Table 20-1lists and describes the registers associated with the I²C bus interface controller.

Table 20-1 I²C Controller Registers

Address Offset	Type	Size (Byte)	Description	Default Value
0x00	RW	4	CR	0
0x04	RC	4	SR	0
0x08	RW	4	CDR	0
0x0C	RW	4	DR	0
0x10	RW	4	SAR	0
0x14	RW	4	TGSR	0x00000401
0x18	R	4	BMR	0x00000003
0x1C	R	4	RES	0
0x20	R	4	RES	0x003FFFFF
0x24	R	4	RES	0x003FFFFF
0x28	R	4	RES	0x000FFFFF
0x2C	R	4	RES	0x001FFFFF
0x30	R	4	REVR	0x00011300
0x34	R	4	FEAR	0

The following subsections describe the details of the I²C controller register.

20.2.1 I²C Control Register

Offset: 0x00

The host processor uses the I²C Control Register (CR) to control the I²C interface controller for transmitting and receiving data from I²C bus.
Please note that SCL_EN must be set to '1' to enter the master mode, unless users want to release SCLout in a special case. Set SCL_EN to '0' to enter the slave mode. The START and STOP bits can only be set in the master mode.

Table 20-2 I²C Control Register

Bit	Name	Type	Reset	Description
31..18	Reserved	R	0	-
17	TEST_BIT	RW	0	Special test mode This bit must be set to '0'.
16	SDA_LOW	RW	0	If set, SDAout will be tied to '0'. For a normal case, it is suggested setting this bit to '0'.
15	SCL_LOW	RW	0	If set, SCLout will be tied to '0'. For a normal case, it is suggested setting this bit to '0'.
14	STARTI_EN	RW	0	If set, this bit will enable the I²C controller to interrupt the host processor when the I²C controller detects a start condition on the I²C bus.
13	ALI_EN	RW	0	If set, this bit will enable the I²C controller to interrupt the host processor when the I²C controller loses the arbitration in the master mode.
12	SAMI_EN	RW	0	If set, this bit will enable the I²C controller to interrupt the host processor when the I²C controller detects a slave address that matches the SAR register (Table 20-6) or a general call address (When GC_EN is set).
11	STOPI_EN	RW	0	If set, this bit will enable the I²C controller to interrupt the host processor when the I²C controller detects a stop condition happening on the I²C bus.
10	BERRI_EN	RW	0	If set, this bit will enable the I²C controller to interrupt the host processor when the I²C controller detects the non-ACK responses from the slave device after one byte of data has been sent in the master mode.
9	DRI_EN	RW	0	If set, this bit will enable the I²C controller to interrupt the host processor when the I²C controller DR register (Table 20-5)has received one data byte from the I²C bus.
8	DTI_EN	RW	0	If set, this bit will enable the I²C controller to interrupt the host processor when the I²C controller DR register has transmitted one data byte onto the I²C bus.
7	TB_EN	RW	0	When Transfer Byte Enable (TB_EN) is set, the I²C controller is ready to receive or transmit one byte. Otherwise, the I²C controller will insert the wait state by pulling SCLout low in the I²C bus. Please note that the TB_EN will be automatically cleared when the device receive or transmit one byte or detect the stop/start condition.
6	ACK/NACK	RW	0	The acknowledge signal sent by the I²C controller when the I²C controller is in the master-receive or slave-receive mode 0: ACK 1: NACK
5	STOP	RW	0	I²C controller initiates a stop condition after transferring the next data byte on the I²C bus when I²C is in the master mode.
4	START	RW	0	I²C controller initiates start condition when the I²C bus is idle or initiates a repeated start condition after transferring the next data byte on the I²C bus in the master mode.
3	GC_EN	RW	0	Enable the I²C controller to respond to a general call message as a slave
2	SCL_EN	RW	0	Enable the I²C controller clock output for the master mode operation

Bit	Name	Type	Reset	Description
1	I2C_EN	RW	0	Enable the I²C bus interface controller
0	I2C_RST	RW	0	Reset the I²C controller This bit will be automatically cleared after two APB_CLK cycles.

20.2.2 I²C Status Register

Offset: 0x04

I²C interrupts are signaled directly to the Advanced Interrupt Controller. The IRQ will be asserted when the interrupt enable bits in the I²C CR (20.2.1) and the corresponding status register are set. When the I²C controller interrupt is asserted, software reads the SR bits to check the status of the I²C controller.

SR is used to clear the following interrupts by reading the register status:

- Data Register (DR) receive data are completed.
- Data Register (DR) transmit data are completed.
- Slave address is detected.
- Bus error is detected.
- Start condition is detected.
- Stop condition is detected.
- Arbitration loss is detected.
- Leave suspend mode
- Receive alert response address
- Enter suspend mode
- Detect alert active
- Detect maximum timeout
- Detect minimum timeout
- Detect master extend time
- Detect slave extend time

Table 20-3 I²C Status Register

Bit	Name	Type	Reset	Description
31..21	Reserved	R	0	-
20	DDA	RC	0	This signal will be set when slave receive address is the device default address.
19..12	Reserved	R	0	-
11	START	RC	0	This signal will be set when the I²C controller detects start condition on the I²C bus.
10	AL	RC	0	This signal will be set when the I²C controller loses arbitration in the master mode.
9	GC	RC	0	This signal will be set when the I²C controller receives slave address that matches the general-call address and the I²C controller is in the slave mode.
8	SAM	RC	0	This signal will be set when the I²C controller receives slave address that matches the address in the Slave Address Register (SAR) when the I²C controller is in the slave mode.
7	STOP	RC	0	This signal will be set when the I²C controller detects stop condition in the I²C bus.

Bit	Name	Type	Reset	Description
6	BERR	RC	0	This signal will be set when the I²C controller detects non-ACK responses from the slave device after transmitting one byte of data when the I²C controller is in the master mode.
5	DR	RC	0	This signal will be set when Data Register (DR) receives one new data byte from the I²C bus.
4	DT	RC	0	This signal will be set when Data Register (DR) transmits one data byte to the I²C bus.
3	BB	R	0	This signal will be set when the I²C bus is busy but the I²C controller is not involved in the transaction.
2	I2CB	R	0	This signal will be set when the I²C controller is busy, i.e. during the time period between START and STOP.
1	ACK	R	0	This signal will be set when the I²C controller receives or sends the non-acknowledgements.
0	RW	R	0	This signal will be set when the I²C controller serves in a master-receive or slave-transmit mode.

20.2.3 I²C Clock Division Register

Offset: 0x08

The I²C Clock Division Register (CDR) defines the divided value used to generate the I²C SCL clock. This register is used with an internal 18-bit counter. When the SCL enable bit in the control register is set, this counter decrements from the programmable value to zero, and then reloads the programmable value and decrements again. Each time the counter reaches zero, the SCL line transaction starts from high to low or vice versa, depending on the current state. This register can be configured to select the transfer speed needed in the I²C bus.

Table 20-4 I²C Clock Division Register

Bit	Name	Type	Reset	Description
31..18	Reserved	R	0	-
17..0	COUNT	RW	0	Counter value is used to generate an I²C clock (SCLout) from the internal APB bus clock. The relation between APB_CLK and the I²C bus clock (SCLout) is shown in the following equation, where GSR is TGSR13-10 (Table 20-7): SCLout = APB_CLK/(2*(COUNT + 2) + GSR) APB_CLK = 48MHz

20.2.4 I²C Data Register

Offset: 0x0C

The I²C Data Register (DR) is used by the host processor to transmit and receive data from the I²C bus. DR is accessed by either the host processor or the I²C controller Shift Register (SHR). Data coming from the I²C bus interface are received by the DR register after a full data byte has been received. Data going out of the I²C bus interface are written into the DR register by the host processor and sent to the serial bus.
When the I²C controller is in the transmit mode, the host processor writes data into the DR register over the APB bus. This occurs when a master transition is initiated or when a data transmit interrupt is signaled. Data are moved from the DR register to the SHR register when the transfer byte enable (TB_EN, 20.2.2) in the control register is set. The data transmit interrupt is signaled when one byte of data has been transferred on the I²C bus. If DR is not

written by the host processor before the next byte package, the I²C controller will insert a wait state until the host processor writes to the DR and sets the Transfer Byte Enable bit.
When the I²C controller is in the received mode, the processor reads DR register data over the APB bus and this occurs when the DR received interrupt is signaled. When the I²C controller has received one new data byte, it will automatically clear the transfer byte enable bit (TB_EN) and issue ACK/NACK on the I²C bus. After issuing ACK/NACK, the I²C controller will insert the DR-received interrupt to processor. Users must set the Transfer Byte bit again for the next byte transfer on the I²C bus.

Table 20-5 I²C Data Register

Bit	Name	Type	Reset	Description
31..8	Reserved	R	0	-
7..0	DR	RW	0	Buffer for the I²C bus data transmission and reception

20.2.5 I²C Slave Address Register

Offset: 0x10

The I²C slave address register defines the 10-bit I²C controller or slave address to which the processor responds when the I²C controller operates in the slave mode. The host processor writes this register before enabling the I²C operation. The register is fully programmable to be set to a value other than the fixed slave peripheral address preexisted in the system.

Table 20-6 I²C Slave Address Register

Bit	Name	Type	Reset	Description
31	EN10	RW	0	10-bit addressing mode enable bit
30..10	Reserved	R	0	-
9..7	SAR_M	RW	0	The most significant 3-bit address to which the I²C controller responds when I²C operates in 10-bit addressing slave mode (EN10 = '1'). When EN10 = '0', the I²C controller ignores these three bits.
6..0	SAR_L	RW	0	The 7-bit address to which the I²C controller responds when the I²C operates in the 7-bit addressing slave mode (EN10 = '0') or the least significant 7-bit address to which the I²C controller responds when the I²C operates in the 10-bit addressing slave mode.

20.2.6 I²C Set/Hold Time and Glitch Suppression Setting Register

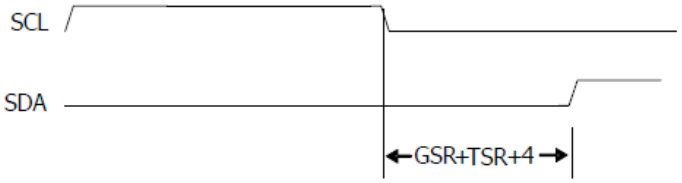
Offset: 0x14

The I²C TGSR13-10 defines the values of the APB_CLK clock period when the I²C bus interface has a built-in glitch suppression logic. Glitches are suppressed according to TGSR13-10 * APB_CLK clock period. For example, with the 48MHz (20.83ns period) APB_CLK clock, and TGSR13-10 = "0100", glitches of 83 ns or less are suppressed. With a 48MHz (20.83ns period) clock, and TGSR13-10 = "0011", glitches of 61.25 ns or less are suppressed. This is within the 50-ns glitch suppression specification. The only limitation is: CDR > 3 + GSR + TSR.

Table 20-7 I²C Set/Hold Time and Glitch Suppression Setting Register

Bit	Name	Type	Reset	Description
31..14	Reserved	R	0	-
13..10	GSR	RW	0	These bits define the values of the APB_CLK clock period when the I²C bus interface has the built-in glitch suppression logic. Glitch is suppressed according to the “GSR * APB_CLK” clock period.
9..0	TSR	RW	0	These bits define the delay values of the APB_CLK clock cycles that the data or acknowledgement will be driven into the I²C SDA bus after I²C SCL bus goes low. The actual delay value is GSR + TSR + 4. Please refer to Figure 20-2 for more details. Please note that the TSR cannot be set to ‘0’.

Figure 20-2 Relationship among TSR, SCL and SDA



20.2.7 I²C Bus Monitor Register

Offset: 0x18

Table 20-8 I²C Bus Monitor Register

Bit	Name	Type	Reset	Description
31..2	Reserved	R	0	-
1	SCL_IN	R	0	This bit continuously reflects the value of the SCLin pin.
0	SDA_IN	R	0	This bit continuously reflects the value of the SDAin pin.

20.3 Programming Guide

20.3.1 Data Write in Slave Mode

1. Write SAR (20.2.5): Set the slave address
2. Write CR (20.2.1): Enable all interrupts and I²C enable and disable the SCL enable
3. Wait for the data receive interrupt - Read SR (20.2.2):
   DR(1), SAM (1), I²CB (1), RW (1) and ACK (0)
4. Write DR (20.2.4): Load the data byte to DR for transfer
   Write CR (20.2.1): Set the transfer byte enable bit
5. Wait for the data transmit interrupt - Read SR (20.2.2):
   DT (1), ACK (1 or 0), RW (1) and STOP (1 or 0)
   If ACK or STOP is set, go to the initial step.
   If ACK and STOP are not set, go to Step 4 to load the second data byte.

Note: If the STOP interrupt occurs, go back to the initial step.

20.3.2 Data Read in Slave Mode

1. Write SAR (20.2.5): Set the slave address
2. Write CR (20.2.1): Enable all interrupts and I²C enable, and disable the SCL enable
3. Wait for the data receive interrupt - Read SR (20.2.2):
   DR (1), SAM (1), I²CB (1), RW (0) and ACK (0)
4. Write CR (20.2.1): Set the transfer byte enable bit (NACK if the last data byte will be sent.)
5. Wait for the data receive interrupt - Read SR (20.2.2):
   DR (1), ACK (1 or 0) and STOP (1 or 0)
6. Read DR (20.2.5): To get data
   If STOP is set, go to the initial step.
   If STOP is not set, go to Step 4 to get the next data.

Note: If the STOP interrupt occurs, go back to the initial step.

20.3.3 Data Write in Master Mode

1. Write CDR (20.2.3): Set the clock count
2. Write DR (20.2.4): Target slave address and RW bit for slave
3. Write CR(20.2.1): Enable all interrupts, set the SCL enable, set the I²C enable, set the START bit, clear the STOP bit and set the Transfer Byte enable
4. Wait for the data transmit interrupt - Read SR (20.2.2):
   DT (1), ACK (0), RW (0) and AL (0)
5. (a) If the master wants to stop after one byte of data has been transferred:
   - Write DR (20.2.4): First data will be sent.
   - Write CR (20.2.1): Clear the START bit, set the STOP bit and set the transfer byte bit
   - Wait for the data transmit interrupt - Read SR (20.2.2):
     DT (1), ACK (0), RW (0) and AL (0)
   - Go to the initial step

   (b) If the master wants to the send the second data byte transfer after the first data byte transfer:
   - Write DR (20.2.4): First data will be sent.
   - Write CR (20.2.1): Clear the START bit, clear the STOP bit and set the transfer byte bit
   - Wait for the data transmit interrupt - Read SR (20.2.2):
     DT (1), ACK (0), RW (0) and AL (0)
   - Go to Step 5 or Step 6
6. If the master wants to send data byte transfer on the same slave or other slave (i.e. restart):
   Go to Step 2

20.3.4 Read in Master Mode

1. Write CDR (20.2.3): Set the clock count

2. Write DR (20.2.4): Target slave address and RW bit for slave
3. Write CR(20.2.1): Enable all interrupts, set the SCL enable, set the I²C enable, set the START bit, clear the STOP bit and set the Transfer Byte enable
4. Wait for the data transmit interrupt - Read SR (20.2.2):
   DT (1), ACK (0), RW (1) and AL (0)
5. (a). If the master wants to stop after one byte of data has been received:
   - Write CR (20.2.1): Clear the START bit, set the STOP bit, set the NACK bit and set the transfer byte enable bit
   - Wait for the data receive interrupt - Read SR (20.2.2):
     DR (1), ACK (1), and RW (1)
   - Read DR (20.2.4): To get data
   - Go to the initial step

   (b). If the master wants to receive the second data byte transfer after the first data byte transfer:
   - Write CR (20.2.1): Clear the START bit, clear the STOP bit and set the transfer byte bit
   - Wait for the data receive interrupt - Read SR (20.2.2):
     DR (1), ACK (0) and RW (1)
   - Read DR (20.2.4): To get data
   - Go to Step 5 or Step 6
6. If the master wants to receive the data byte transfer from the same slave or other slave (i.e. restart):
   Go to Step 2

21 Timer and Watchdog Unit

Offset: 0xB0000000

21.1 Timer

21.1.1 Programming Model

21.1.1.1 Summary of Timer Registers

Table 21-1 shows the offset, type, width, reset value and name of each timer register. All registers must be accessed with 32Bit Word accesses.

Table 21-1 Timer Register Summary

Offset	Type	Reset	Width	Name	Description
0x00	RW	0x00	32	Timer1Counter	Timer 1 Counter
0x04	RW	0x00	32	Timer1Load	Timer 1 reload value
0x08	RW	0x00	32	Timer1Match1	Timer 1 match 1 value
0x0C	RW	0x00	32	Timer1Match2	Timer 1 match 2 value
0x10	RW	0x00	32	Timer2Counter	Timer 2 Counter
0x14	RW	0x00	32	Timer2Load	Timer 2 reload value
0x18	RW	0x00	32	Timer2Match1	Timer 2 match 1 value
0x1C	RW	0x00	32	Timer2Match2	Timer 2 match 2value
0x20	RW	0x00	32	Timer3Counter	Timer 3 Counter
0x24	RW	0x00	32	Timer3Load	Timer 3 reload value
0x28	RW	0x00	32	Timer3Match1	Timer 3 match 1 value
0x2C	RW	0x00	32	Timer3Match2	Timer 3 match 2 value
0x30	RW	0x00	32	Timer4Counter	Timer 4 Counter
0x34	RW	0x00	32	Timer4Load	Timer 4 reload value
0x38	RW	0x00	32	Timer4Match1	Timer 4 match 1 value
0x3C	RW	0x00	32	Timer4Match2	Timer 4 match 2 value
0x40	RW	0x00	32	Timer5Counter	Timer 5 Counter
0x44	RW	0x00	32	Timer5Load	Timer 5 reload value
0x48	RW	0x00	32	Timer5Match1	Timer 5 match 1 value
0x4C	RW	0x00	32	Timer5Match2	Timer 5 match 2 value
0x50	RW	0x00	32	Timer6Counter	Timer 6 Counter
0x54	RW	0x00	32	Timer6Load	Timer 6 reload value
0x58	RW	0x00	32	Timer6Match1	Timer 6 match 1 value
0x5C	RW	0x00	32	Timer6Match2	Timer 6 match 2 value
0x60	RW	0x00	32	InterruptST	Interrupt State of all Timer
0x64	RW	0x00777777	32	InterMaskSet	Interrupt Mask Set of Timer
0x68	W	0x00	32	InterMaskClear	Interrupt Mask Clear of Timer
0x6C	RW	0x00	32	InterEnableSet	Interrupt Enable Set of Timer
0x70	W	0x00	32	InterEnableClear	Interrupt Enable Clear of Timer
0x74	W	0x00	32	InterAck	Acknowledge Register of Interrupt
0x78	RW	0x00	32	Tm1_CR	Timer 1 Control Register

Offset	Type	Reset	Width	Name	Description
0x7c	RW	0x00	32	Tm2_CR	Timer 2 Control Register
0x80	RW	0x00	32	Tm3_CR	Timer 3 Control Register
0x84	RW	0x00	32	Tm4_CR	Timer 4 Control Register
0x88	RW	0x00	32	Tm5_CR	Timer 5 Control Register
0x8C	RW	0x00	32	Tm6_CR	Timer 6 Control Register
0x90	W	0x00	32	Tm_Dis_En	Timer Enable Disable

21.1.2 Register Description

21.1.2.1 Timer*n*Counter

Offset: Timer1Counter 0x00
Timer2Counter 0x10
Timer3Counter 0x20
Timer4Counter 0x30
Timer5Counter 0x40
Timer6Counter 0x50

Timer*n*Counter are the 32 Bit counter registers of Timer*n*, respectively. If the timer is disabled the Timer*n*Counter will hold the value.

21.1.2.2 Timer*n*Load

Offset: Timer1Load 0x04
Timer2Load 0x14
Timer3Load 0x24
Timer4Load 0x34
Timer5Load 0x44
Timer6Load 0x54

Timer*n*Load are the auto-reload registers for Timer*n*. When a Timer*n* overflow occurs, the value of Timer*n*Load is loaded into the counter*n* register. This value can be used to set the period between two overflows or underflows.
Note: An overflow indicates that the counter has been exceeded than what can be supported. That is, for counts up to 0xFFFF_FFFF or down to 0x0000_0000.

21.1.2.3 Timer*n*Match 1 or 2

Offset: Timer1Match1 0x08, Timer1Match2 0x0C,
Timer2Match1 0x18, Timer2Match2 0x1C,
Timer3Match1 0x28, Timer3Match2 0x2C,
Timer4Match1 0x38, Timer4Match2 0x3C,
Timer5Match1 0x48, Timer5Match2 0x4C,
Timer6Match1 0x58, Timer6Match2 0x5C,

Timer*n*Match1 and Timer*nMatch2* are the match registers of Timer*n*. When the values of the counter*n* equals the value of Timer*n*Match1 or Timer*nMatch*2 register and the EnableTm*n*Match1 and EnableTm*n*Match2 bit in the enable interrupt register is set and the TmEnable bit in the Tm*n*_CR is set, then tm*n*_intr will be triggered.
After the counter reaches the match value the counter will keep counting.

21.1.2.4 Tm*n*_CR Regsiter

Offset: Timer1CR 0x78
Timer2CR 0x7C
Timer3CR 0x80
Timer4CR 0x84
Timer5CR 0x88
Timer6CR 0x8C

Tm*n*_CR is the control register of every timer module. It controls the timer enable/disable function, clock prescale selection, up or down count and (ADC) auto disable counter. When TmEnable is disabled, tm*n*_intr will never be triggered and the value of counter*n* will be held. The tm*n*_intr will be triggered once the timer overflow occurs or when the value of the counter is equals the value of Tm*n*Match1 or Tm*n*Match2. This depends on which interrupt is enabled.

Table 21-2 Timer*n* Control Register Bit Description

Bit	Name	Description
31..9	Reserved	-
8	Special Function	If the value 0x100 (hex) is written into the Tmn_CR register the Tmn_CR is reset, and the corresponding interrupt (Match1, Match2, Overflow) will be masked. The interrupt status is set to zero and all interrupts are disabled. Bits7..0 must be zero.
7..6	Reserved	-
5..4	Prescale	Prescale field. Determines effective clock rate for the counter based on the AHB HCLK: 00 = divide by 1 (default) 01 = divide by 2 10 = divide by 4 11 = divide by 8
3	Reserved	-
2	Tm*n* Down Up	Timer*n* up or down count 0: Down count The Timer*n*Counter decreases to 0 from its initial value. When it underflows, Timer*n*Counter will be auto- reloaded with the value of Timer*n* Load. 1: Up count The Timer*n*Counter increases to 0xffff_ffff from its initial value. When it overflows, Timer*n*Counter will be auto-reloaded with the value of Timer*n* Load.
1	ADC	Timer*n* Auto Disable Counter (Free Running, single-shot) 0 :The Timer Counter is enabled after a interrupt occurs 1 :The Timer Counter is disabled after a interrupt occurs
0	TmEnable	Timer*n* Enable Bit 0 : Disable

Bit	Name	Description
		1 : Enable

When the ADC bit (Auto Disable Counter) is set to ’0’ and the counter rolls over when counting up or counting down and if only the overflow interrupt is enabled, an interrupt will be generated and the value in the load register will be automatically reloaded into the counter and the counter will continue to count. This is useful for generating a repetitive pulse train with a specified period.

When the ADC bit (Auto Disable Counter) is set to ’1’ and the counter rolls over when counting up or counting down and if only the overflow interrupt is enabled, an interrupt will be generated. The value in the load register will be automatically reloaded into the counter and the counter will be disabled.

If the Tm*n*Match1 or Tm*n*Match2 Interrupt is enabled, the ADC is set to ‘1’ and the value of the counter is equal to the value in the Timer*n*Match1 or Timer*nMatch*2 register an interrupt will be generated. The value in the load register will be not reloaded into the counter. The counter will not continue to count is disabled.

If Tm*n*Match1 or Tm*n*Match2 Interrupt is enabled, the ADC is set ‘0’ and the value of the counter is equal to the value in the Timer*n*Match1 or Timer*nMatch*2 register an interrupt will be generated. The value in the load register will be not reloaded into the counter. The counter is continued to count.

21.1.2.5 InterruptST Register

The InterruptST register provides the interrupt status of the timer module. When the tmr*n*_intr is asserted, the user identifies the interrupt by reading the InterruptST register. All the bits of InterruptST must be cleared by the firmware this is achieved by writing to the corresponding bit in the Acknowledge-Register.

Table 21-3 Interrupt Status register

Bit	Name	Description
0	Tm1Match1	Tm1Match1 interrupt 0: No effect 1: Timer1Counter value equals to the value in the Timer1Match1 register
1	Tm1Match2	Tm1Match2 interrupt 0: No effect 1: Timer1Counter value equals to the value in the Timer1Match2 register
2	Timer1Overflow	Timer 1Overflow interrupt 0: No effect 1: Timer1Counter overflow
3		Reserved
4	Tm2Match1	Tm2Match1 interrupt 0: No effect 1: Timer2Counter value equals to the value in the Timer2Match1 register

Bit	Name	Description
5	Tm2Match2	Tm2Match2 interrupt 0: No effect 1: Timer2Counter value equals to the value in the Timer2Match2 register
6	Timer2Overflow	Timer 2 Overflow interrupt 0: No effect 1: Timer2Counter overflow
7		Reserved
8	Tm3Match1	Tm3Match1 interrupt 0: No effect 1: Timer3Counter value equals to the value in the Timer3Match1 register
9	Tm3Match2	TmMatch2 interrupt 0: No effect 1: Timer3Counter value equals to the value in the Timer3Match2 register
10	Timer3Overflow	Timer 3 Overflow interrupt 0: No effect 1: Timer3Counter overflow
11		Reserved
12	Tm4Match1	Tm4Match1 interrupt 0: No effect 1: Timer4Counter value equals to the value in the Timer4Match1 register
13	Tm4Match2	Tm4Match2 interrupt 0: No effect 1: Timer4Counter value equals to the value in the Timer4Match2 register
14	Timer4Overflow	Timer 4 Overflow interrupt 0: No effect 1: Timer4Counter overflow
15		Reserved
16	Tm5Match1	Tm5Match1 interrupt 0: No effect 1: Timer5Counter value equals to the value in the Timer5Match1 register
17	Tm5Match2	Tm5Match2 interrupt 0: No effect 1: Timer5Counter value equals to the value in the Timer5Match2 register
18	Timer5Overflow	Timer 5 Overflow interrupt 0: No effect 1: Timer5Counter overflow
19		Reserved
20	Tm6Match1	Tm6Match1 interrupt 0: No effect 1: Timer6Counter value equals to the value in the Timer6Match1 register
21	Tm6Match2	TimerMatch1interrupt 0: No effect 1: Timer6Counter value equals to the value in the Timer6Match2 register
22	Timer6Overflow	Timer 6 Overflow interrupt 0: No effect 1: Timer6Counter overflow

21.1.2.6 InterMaskSet and InterMaskClear

Offsets: 0x64, 0x68

The InterMaskSet and InterMaskClear register provide the opportunity to the user to mask interrupts. If one bit of the InterMaskSet register is set ‘1’ then the corresponding Interrupt will be masked. If one bit of the InterMaskClear register is set ‘1’ then the mask of the corresponding Interrupt will be disabled.

Table 21-4 Interrupt Mask register

Bit	Name	Description
0	MaskTm1Match1	Mask Tm1Match1 interrupt 0: No effect **InterMaskSet** 1: Tm1Match1 interrupt will be masked **InterMaskClear** 1: Tm1Match1 interrupt will not be masked
1	MaskTm1Match2	Mask Tm1Match2 interrupt 0: No effect **InterMaskSet** 1: Tm1Match2 interrupt will be masked **InterMaskClear** 1: Tm1Match2 interrupt will not be masked
2	MaskTm1Overflow	Mask Timer1Overflow interrupt 0: No effect **InterMaskSet** 1: Timer1Overflow interrupt will be masked **InterMaskClear** 1: Timer1Overflow interrupt will not be masked
3		Reserved
4	MaskTm2Match1	Mask Tm2Match1 interrupt 0: No effect **InterMaskSet** 1: Tm2Match1 interrupt will be masked **InterMaskClear** 1: Tm2Match1 interrupt will not be masked
5	MaskTm2Match2	Mask Tm2Match2 interrupt 0: No effect **InterMaskSet** 1: Tm2Match2 interrupt will be masked **InterMaskClear** 1: Tm2Match2 interrupt will not be masked
6	MaskTm2Overflow	Mask Timer2Overflow interrupt 0: No effect **InterMaskSet** 1: Timer2Overflow interrupt will be masked **InterMaskClear** 1: Timer2Overflow interrupt will not be masked
7		Reserved
8	MaskTm3Match1	Mask Tm3Match1 interrupt 0: No effect **InterMaskSet** 1: Tm3Match1 interrupt will be masked **InterMaskClear** 1: Tm3Match1 interrupt will not be masked

Bit	Name	Description
9	MaskTm3Match2	Mask Tm3Match2 interrupt 0: No effect **InterMaskSet** 1: Tm3Match2 interrupt will be masked **InterMaskClear** 1: Tm3Match2 interrupt will not be masked
10	MaskTm3Overflow	Mask Timer3Overflow interrupt 0: No effect **InterMaskSet** 1: Timer3Overflow interrupt will be masked **InterMaskClear** 1: Timer3Overflow interrupt will not be masked
11		Reserved
12	MaskTm4Match1	Mask Tm4Match1 interrupt 0: No effect **InterMaskSet** 1: Tm4Match1 interrupt will be masked **InterMaskClear** 1: Tm4Match1 interrupt will not be masked
13	MaskTm4Match2	Mask Tm4Match2 interrupt 0: No effect **InterMaskSet** 1: Tm4Match2 interrupt will be masked **InterMaskClear** 1: Tm4Match2 interrupt will not be masked
14	MaskTm4Overflow	Mask Timer4Overflow interrupt 0: No effect **InterMaskSet** 1: Timer4Overflow interrupt will be masked **InterMaskClear** 1: Timer4Overflow interrupt will not be masked
15		Reserved
16	MaskTm5Match1	Mask Tm5Match1 interrupt 0: No effect **InterMaskSet** 1: Tm5Match1 interrupt will be masked **InterMaskClear** 1: Tm5Match1 interrupt will not be masked
17	MaskTm5Match2	Mask Tm5Match2 interrupt 0: No effect **InterMaskSet** 1: Tm5Match2 interrupt will be masked **InterMaskClear** 1: Tm5Match2 interrupt will not be masked
18	MaskTm5Overflow	Mask Timer5Overflow interrupt 0: No effect **InterMaskSet** 1: Timer5Overflow interrupt will be masked **InterMaskClear** 1: Tm5Match2 interrupt will not be masked
19		Reserved
20	MaskTm6Match1	Mask Tm6Match1 interrupt 0: No effect **InterMaskSet** 1: Tm6Match1 interrupt will be masked **InterMaskClear** 1: Tm6Match1 interrupt will not be masked

Bit	Name	Description
21	MaskTm6Match2	Mask Tm6Match2 interrupt 0: No effect **InterMaskSet** 1: Tm6Match2 interrupt will be masked **InterMaskClear** 1: Tm6Match2 interrupt will not be masked
22	MaskTm6Overflow6	Mask Timer6Overflow interrupt 0: No effect **InterMaskSet** 1: Timer6Overflow interrupt will be masked **InterMaskClear** 1: Timer6Overflow interrupt will not be masked

21.1.2.7 InterEnableSet and InterEnableClear

Offsets: 0x6C ,0x70

The InterEnableSet and InterEnableClear register provides the opportunity to the user to enable or disable the kinds of the interrupts. If one bit of the InterEnableSet register is set ‘1’ then the corresponding Interrupt kind will be enabled. If one bit of the InterEnableClear register is set ‘1’ then the corresponding Interrupt kind will be disabled.

Table 21-5 Interrupt Enable register

Bit	Name	Description
0	EnableTm1Match1	EnableTm1Match1 interrupt 0: No effect **InterEnableSet** 1: Tm1Match1 interrupt will be Enabled **InterEnableClear** 1: Tm1Match1 interrupt will be Disabled
1	EnableTm1Match2	Enable Tm1Match2 interrupt 0: No effect **InterEnableSet** 1: Tm1Match2 interrupt will be Enabled **InterEnableClear** 1: Tm1Match2 interrupt will be Disabled
2	EnableTmOverflow1	Enable TimerOverflow1 interrupt 0: No effect **InterEnableSet** 1: Timer1Overflow interrupt will be Enabled **InterEnableClear** 1: Timer1Overflow interrupt will be Disabled
3		**Reserved**
4	EnableTm2Match1	Mask Tm2Match1 interrupt 0: No effect **InterEnableSet** 1: Tm2Match1 interrupt will be Enabled **InterEnableClear** 1: Tm2Match1 interrupt will be Disabled
5	EnableTm2Match2	Enable Tm2Match2 interrupt 0: No effect **InterEnableSet** 1: Tm2Match2 interrupt will be Enabled **InterEnableClear** 1: Tm2Match2 interrupt will be Disabled

Bit	Name	Description
6	EnableTmOverflow2	Enable TimerOverflow2 interrupt 0: No effect **InterEnableSet** 1: Timer2Overflow interrupt will be Enabled **InterEnableClear** 1: Timer2Overflow interrupt will be Disabled
7		**Reserved**
8	EnableTm3Match1	Enable Tm3Match1 interrupt 0: No effect **InterEnableSet** 1: Tm3Match1 interrupt will be Enabled **InterEnableClear** 1: Tm3Match1 interrupt will be Disabled
9	EnableTm3Match2	Enable Tm3Match2 interrupt 0: No effect **InterEnableSet** 1: Tm3Match2 interrupt will be Enabled **InterEnableClear** 1: Tm3Match2 interrupt will be Disabled
10	EnableTmOverflow3	Enable TimerOverflow3 interrupt 0: No effect **InterEnableSet** 1: Timer3Overflow interrupt will be Enabled **InterEnableClear** 1: Timer3Overflow interrupt will be Disabled
11		**Reserved**
12	EnableTm4Match1	Enable Tm4Match1 interrupt 0: No effect **InterEnableSet** 1: Tm4Match1 interrupt will be Enabled **InterEnableClear** 1: Tm4Match1 interrupt will be Disabled
13	EnableTm4Match2	Enable Tm4Match2 interrupt 0: No effect **InterEnableSet** 1: Tm4Match2 interrupt will be Enabled **InterEnableClear** 1: Tm4Match2 interrupt will be Disabled
14	EnableTmOverflow4	Enable TimerOverflow4 interrupt 0: No effect **InterEnableSet** 1: Timer4Overflow interrupt will be Enabled **InterEnableClear** 1: Timer4Overflow interrupt will be Disabled
15		**Reserved**
16	EnableTm5Match1	Enable Tm5Match1 interrupt 0: No effect **InterEnableSet** 1: Tm5Match1 interrupt will be Enabled **InterEnableClear** 1: Tm5Match1 interrupt will be Disabled
17	EnableTm5Match2	Enable Tm5Match2 interrupt 0: No effect **InterEnableSet** 1: Tm5Match2 interrupt will be Enabled **InterEnableClear** 1: Tm5Match2 interrupt will be Disabled

Bit	Name	Description
18	EnableTmOverflow5	Enable TimerOverflow5 interrupt 0: No effect **InterEnableSet** 1: Timer5Overflow interrupt will be Enabled **InterEnableClear** 1: Tm5Match2 interrupt will be Disabled
19		**Reserved**
20	EnableTm6Match1	Enable Tm6Match1 interrupt 0: No effect **InterEnableSet** 1: Tm6Match1 interrupt will be Enabled **InterEnableClear** 1: Tm6Match1 interrupt will be Disabled
21	EnableTm6Match2	Enable Tm6Match2 interrupt 0: No effect **InterEnableSet** 1: Tm6Match2 interrupt will be Enabled **InterEnableClear** 1: Tm6Match2 interrupt will be Disabled
22	EnableTmOverflow6	Enable TimerOverflow6 interrupt 0: No effect **InterEnableSet** 1: Timer6Overflow interrupt will be Enabled **InterEnableClear** 1: Timer6Overflow interrupt will be Disabled

21.1.2.8 InterAck

Offset: 0x74

This register is used to acknowledge the asserted interrupts. When an interrupt is acknowledged the corresponding bit in the InterruptST will be cleared.

Table 21-6 Interrupt Acknowledge

Bit	Name	Description
0	IntAckTm1Match1	Acknowledge Tm1Match1 interrupt 0: No effect 1: Tm1Match1 in InterruptST will be cleared
1	IntAckTm1Match2	Enable Tm1Match2 interrupt 0: No effect 1: Tm1Match2 in InterruptST will be cleared
2	IntAckTmOverflow1	Enable TimerOverflow1 interrupt 0: No effect 1: Timer1Overflow in InterruptST will be cleared
3		**Reserved**
4	IntAckTm2Match1	Mask Tm2Match1 interrupt 0: No effect 1: Tm2Match1 in InterruptST will be cleared
5	IntAckTm2Match2	Enable Tm2Match2 interrupt 0: No effect 1: Tm2Match2 in InterruptST will be cleared
6	IntAckTmOverflow2	Enable TimerOverflow2 interrupt 0: No effect 1: Timer2Overflow in InterruptST will be cleared

Bit	Name	Description
7		**Reserved**
8	IntAckTm3Match1	Enable Tm3Match1 interrupt 0: No effect 1: Tm3Match1 in InterruptST will be cleared
9	IntAckTm3Match2	Enable Tm3Match2 interrupt 0: No effect 1: Tm3Match2 in InterruptST will be cleared
10	IntAckTmOverflow3	Enable TimerOverflow3 interrupt 0: No effect 1: Timer3Overflow in InterruptST will be cleared
11		**Reserved**
12	IntAckTm4Match1	Enable Tm4Match1 interrupt 0: No effect 1: Tm4Match1 in InterruptST will be cleared
13	IntAckTm4Match2	Enable Tm4Match2 interrupt 0: No effect 1: Tm4Match2 in InterruptST will be cleared
14	IntAckTmOverflow4	Enable TimerOverflow4 interrupt 0: No effect 1: Timer4Overflow in InterruptST will be cleared
15		**Reserved**
16	IntAckTm5Match1	Enable Tm5Match1 interrupt 0: No effect 1: Tm5Match1 in InterruptST will be cleared
17	IntAckTm5Match2	Enable Tm5Match2 interrupt 0: No effect 1: Tm5Match2 in InterruptST will be cleared
18	IntAckTmOverflow5	Enable TimerOverflow5 interrupt 0: No effect 1: Tm5Match2 in InterruptST will be cleared
19		**Reserved**
20	IntAckTm6Match1	Enable Tm6Match1 interrupt 0: No effect 1: Tm6Match1 in InterruptST will be cleared
21	IntAckTm6Match2	Enable Tm6Match2 interrupt 0: No effect 1: Tm6Match2 in InterruptST will be cleared
22	IntAckTmOverflow6	Enable TimerOverflow6 interrupt 0: No effect 1: Timer6Overflow in InterruptST will be cleared

21.1.2.9 Tm_Dis_En Register (offset 0x90)

This register is used to enable or disable the Timer*n*Counters. The Tm_Dis_En register has effect in the TmEnable bit in the Tm*n*_CR register. Each timer*n* can be individually enabled. This functionality is useful to enable or disable all the timers with one write access only to the timer module.

Table 21-7 Tm_Dis_En Register

Bit	Name	Description
0	ENTC1	0: No effect

Bit	Name	Description
		1: Enable Timer Counter 1
1	ENTC2	0: No effect 1: Enable Timer Counter 2
2	ENTC3	0: No effect 1: Enable Timer Counter 3
3	ENTC4	0: No effect 1: Enable Timer Counter 4
4	ENTC5	0: No effect 1: Enable Timer Counter 5
5	ENTC6	0: No effect 1: Enable Timer Counter 6
6~7		Reserve
8	DISTC1	0: No effect 1: Disable Timer Counter 1
9	DISTC2	0: No effect 1: Disable Timer Counter 2
10	DISTC3	0: No effect 1: Disable Timer Counter 3
11	DISTC4	0: No effect 1: Disable Timer Counter 4
12	DISTC5	0: No effect 1: Disable Timer Counter 5
13	DISTC6	0: No effect 1: Disable Timer Counter 6
14~31		Reserve

21.1.3 Programming Sequence

21.1.3.1 Basic Information

The user must first set the TmLoad, TmMatch1 and TmMatch2 registers and determine TmCR[Prescale] and the InterMaskSet and InterEnableSet register also. When the Tm*n*_CR bit 0 is set ‘1’ the counting starts. If the counter reaches the value of Timer*n*Match1 or Timer*nMatch*2 register or an overflow/underflow occurs, an interrupt will be generated and the event is recorded in the InterruptST register. The firmware can read the InterruptST to detect which type of interrupt was triggered.

Programming sequence:
1. Set TmLoad or TimerCounter
2. Set TmMatch1
3. Set TmMatch2
4. Use down or up counting, use TmCR bit ‘2’
5. Use (ADC) on or off, use TmCR bit ‘1’
6. Mask or unmask the Interrupt type for each Timer
7. Enable or disable the Interrupt type for each Timer
8. Enable the Timer, use TmCR bit ‘0’

21.2 Watchdog

Offset: 0xB0000000

21.2.1 Basic Information

A watchdog is used in systems to prevent system lockup due to software or hardware failures. In normal operation, the user restarts the Watchdog at regular intervals before the counter counts down to 0. If the counter reaches 0, the Watchdog generates one or a combination of events (system reset or system interrupt).
The counter is loaded with a value stored in a load register.
If the counter is enabled, it will count down towards zero. If the counter reaches zero, the watchdog reset is asserted and should be used to reset the system. With normal operation the watchdog is prevented from reaching zero by periodical reload operations executed by firmware.

21.2.2 Features

- During the timeout, the outputs are one or a combination of the following events:
  - System reset
  - System interrupt
- 32-bit down counter
- Access protection to prevent unexpected restart, the firmware needs to write 0x56495041 to the WdRestart register.

- Supports prescale features to determine effective clock rate based on AHB_CLK (96MHz).

- Two operation modes:

  Mode 1: If a counter reaches zero a Watchdog interrupt will be generated. The timer will reload the value from the WdLoad register and continue decrementing. If the WdStatus register is not cleared and the counter reaches the zero value again then a Watchdog reset will be generated.

  Mode 0: As soon as the counter reaches zero the watchdog will generate reset or the Watchdog interrupt immediately (depending on WdCR register - Wd_Rst_Enable, Wd_Inter_Enab bit).

21.2.3 Programming Model

21.2.3.1 Summary of Timer Registers

Table 11-8 shows the offset, type, width, reset value and name of each watchdog register. All registers must be accessed with 32Bit Word accesses.

Table 21-8 Watchdog Register Summary

Offset	Type	Power ON Reset	Watchdog Reset	Width	Name	Description

Offset	Type	Power ON Reset	Watchdog Reset	Width	Name	Description
0x94	R	0x05B8D800	0x05B8D800	32	WdCounter	watchdog counter
0x98	RW	0x05B8D800	0x05B8D800	32	WdLoad	watchdog reload value
0x9C	W	0	0	32	WdRestart	If writing 0x56495041 to this register, the watchdog timer will automatically reload the WdLoad to WdCounter and restart the counting.
0xA0	RW	0	0	32	WdCR	watchdog control register
0xA4	R	0	0	1	WdStatus	The watchdog timer status register This bit is set when the counter reaches 0. 0: Did not reach 0 1: Reached 0
0xA8	W	0	0	1	WdClear	The watchdog timer is cleared. Writing 1 or 0 to this register will clear the WdStatus
0xAC	RW	0	n/a	2	WdResetStatus	Bit0: • "1" indicates that a watchdog reset has been triggered • “0” if a power on reset is triggered. • Provide to the application the information that a watchdog reset has been triggered, after that the application can reset the bit0. Bit1: • "1" it is used as a flag to distinguish whether the watchdog reset was intentionally triggered by the software or not • “0” if a power on reset is triggered.
0xB0	RW	0x00FF	n/a	16	WdRestlen	Watchdog Reset Length Set width of reset pulse (value * 1/96MHz). The default value is 0xFF.

21.2.4 Register Description

21.2.4.1 WdCounter

The WdCounter register contains the current counter value. After Watchdog-/POWER ON-Reset the WdCounter register is set to the default value 0x05B8D800 (Watchdog time: 1sec, timebase: 1/96Mhz). If the programmer writes 0x56495041 to the WdRestart register, the value of the WdLoad register will be loaded into the WdCounter register. If the Wd_Enable (WdCR register) bit is set, the WdCounter starts to decrease the counter value. If the Wd_Enable bit is disabled, the WdCounter holds the value.

21.2.4.2 WdLoad

The WdLoad register contains the value which will be loaded automatically into the WdCounter register. The default value after reset is 0x05B8D800.

21.2.4.3 WdRestart

To prevent an unexpected restart, the programmer must write the code 0x56495041 to the WdRestart register as password to reload the WdLoad value.

21.2.4.4 WdCR

Table 21-9 Watchdog Control register

Bit	Name	Reset	Description
0	Wd_Enable	0	Watchdog timer enable 0: Disable 1: Enable
1	Wd_Rst_Enable	0	Watchdog timer reset enable 0: Disable 1: Enable
2	Wd_Intr_Enable	0	Watchdog timer interrupt enable 0: Disable 1: Enable
3	Mode	0	Mode 0: Mode 0 1: Mode 1
5..4	Prescale	00	Prescaler Determines effective clock rate for the Watchdog counter (based on 96MHz) 00: divide by 1 (default) 01: divide by 2 10: divide by 4 11: divide by 8

Mode:

Table 21-10 Watchdog modes

Mode	Settings	Description
0	Wd_Enable = 1 Wd_Rst_Enable = 1 Wd_Intr_Enable = 0	If the watchdog counter reaches zero, the watchdog will generate reset.
	Wd_Enable = 1 Wd_Rst_Enable = 0 Wd_Intr_Enable = 1	If the Watchdog counter reaches zero, the watchdog generates the Watchdog interrupt.
1	Wd_Enable = 1 Wd_Rst_Enable = 1 Wd_Intr_Enable = 1	If the Watchdog counter reaches zero, the watchdog generates the Watchdog interrupt. The WdCounter register is reloaded with WdLoad value and continue decrementing. A - WdStatus = 1: if the WdCounter reaches zero, reset will be generated B – WdStatus = 0 (write “1” to WdClear): if the WdCounter reaches zero a Watchdog interrupt will be generated

21.2.4.5 WdStatus

WdStatus shows the status of WdCounter. If the WdCounter register reaches the value 0, the WdStatus bit is set.

21.2.4.6 WdClear

The WdStatus will be cleared, if WdClear is set.

21.2.4.7 WdResetStatus

Provides to the application the information the source of a watchdog reset. These bits can be used during software startup to check who triggers the reset.
Bit0: Provide to the application the information that a watchdog reset has been triggered, after that the application can reset the bit0.
Bit1: It is used as a flag from the application to distinguish whether the watchdog reset was intentionally triggered by the software or by the hardware.

Table 21-11 Watchdog reset status

Bit	Value	Description
1..0	00	No watchdog reset occurred
	01	Watchdog reset was triggered by hardware
	10	Firmware set bit1 before a software reset will be triggered. In this case a reset event was released from a differnet source, because the reset wasn't triggered by the software (bit0 is zero)
	11	watchdog reset was triggered, watchdog reset was intentionally triggered by the software

21.2.4.8 WdRestlen

Set width of reset pulse (value * 1/96MHz). The default value is 0xFF.

21.2.5 Programming Sequence

Programming sequence:
1. Set the WdLoad register
2. Write 0x56495041 to the WdRestart register to load the WdLoad value
3. Set the Watchdog reset length
4. Initialize Watchdog control register WdCR
5. Enable the Watchdog timer

22 General Purpose I/O

Offset: 0x73800000

22.1 Programming Model

22.1.1 Summary of General Purpose I/O Registers

Table 2-1 shows the offset, type, width, reset value, name and configuration option of each GPIO programming register. The GPIOs 0 to 3 are always available, GPIOs 4 to 31 are shared with other functions and can only be used after configuring the Pin Controller. All these GPIOs can be freely configured with the following registers.
The GPIO block is driven by the APB_CLK running at 48MHz.

Table 22-1 General Purpose I/O Registers

Offset	Type	Width	Reset	Name	Description
0x00	RW	[31:0]	0x0	GpioDataOut	GPIO data output register
0x04	R	[31:0]	0x0	GpioDataIn	GPIO data input register
0x08	RW	[31:0]	0x0	PinDir	GPIO direction register 0 : Input 1 : Output
0x0C	RW	[31:0]	0x0	PinBypass	GPIO bypass register 0 : No bypass 1 : Bypass
0x10	W	[31:0]	0x0	GpioDataSet	GPIO data bit set register When writing to this register, the corresponding bits in the data register are set to 1, and the other bits remain unchanged.
0x14	W	[31:0]	0x0	GpioDataClear	GPIO data bit clear register When writing to this register, the corresponding bits in the data register are cleared to 0, and the other bits remain unchanged.
0x18			0x0	Reserved	Do not modify
0x1C			0x0	Reserved	Do not modify
0x20	RW	[31:0]	0x0	IntrEnable	GPIO interrupt enable register 0: Pin interrupt is disabled. 1: Pin interrupt is enabled.
0x24	R	[31:0]	0x0	IntrRawState	GPIO interrupt raw status register 0: Interrupt is not detected. 1: Interrupt is detected.
0x28	R	[31:0]	0x0	IntrMaskedState	GPIO interrupt masked status register 0: Interrupt is not detected or masked. 1: Interrupt is detected and not masked.
0x2C	RW	[31:0]	0x0	IntrMask	GPIO interrupt mask register 0: Mask is disabled. 1: Mask is enabled.
0x30	W	[31:0]	0x0	IntrClear	GPIO interrupt clear 0: No effect 1: Clear interrupt
0x34	RW	[31:0]	0x0	IntrTrigger	GPIO interrupt trigger method register 0: Edge trigger 1: Level trigger

Offset	Type	Width	Reset	Name	Description
0x38	RW	[31:0]	0x0	IntrBoth	GPIO edge-trigger interrupt by single or both edges 0: Single edge 1: Both edges
0x3C	RW	[31:0]	0x0	IntrRiseNeg	GPIO interrupt triggered at the rising or falling edge 0: Rising edge 1: Falling edge GPIO interrupt triggered by high or low level 0: High level 1: Low level
0x40	RW	[31:0]	0x0	BounceEnable	GPIO pre-scale clock enable When enabled, the APB_CLK (48MHz) will be divided by the BouncePreScale clocks. This signal is used to extend the clock cycle of detecting interrupt. 0: Disable 1: Enable
0x44	RW	[31:0]	0x7D0	BouncePreScale	GPIO Pre-scale, used to generate slower debounce clock frequencies based on the APB_CLK (48MHz). The allowable range is from 0x1 to 0xFFFFFF.

Register Descriptions

The following sections provide the details of the general purpose I/O registers.

22.1.1.1 GpioDataOut

GpioDataOut register is the GPIO data out register. When the PinDir indicates the pin is an output, the GpioDataOut is connected to io_out. When PinDir indicates the pin is an input, GpioDataOut can hold the data.

22.1.1.2 GpioDataIn

GpioDataIn register is the GPIO data in register. When the PinDir indicates the pin is an input, the GpioDataIn will latch gpio_in at the APB_CLK rising edge. When PinDir indicates the pin is an output, the GpioDataIn register is a “don't care” register.

22.1.1.3 PinDir

PinDir register controls the gpio_en. When the PinDir indicates the pin is an output, the related gpio_en is set to 1. Otherwise, the related gpio_en is set to 0.

22.1.1.4 PinBypass

PinBypass register controls the bypass mode. When the PinBypass indicates the pin is in the bypass mode, gpio_en is connected to gpio_bps_en, gpio_in is connected to gpio_bps_out, and gpio_bps_in is connected to gpio_out.

22.1.1.5 GpioDataSet

GpioDataSet is the bit operation logic. When writing to this address, if some bits of the GpioDataSet are 1, the related bits of GpioDataOut will be set. For example, if the GpioDataOut[7:0] = 0x23, and 0x47 is written to GpioDataSet, the result of GpioDataOut[7:0] will be 0x67.

22.1.1.6 GpioDataClear

GpioDataClear is the bit operation logic. When writing to this address, if some bits of the GpioDataClear are 1s, the related bits of the GpioDataOut will be cleared. For example, if the GpioDataOut[7:0] = 0x23, and 0x47 is written to GpioDataClear, the result of GpioDataOut[7:0] will be 0x20.

22.1.1.7 IntrEnable

IntrEnable register controls the enable or disable interrupt detection logic. It is a mask of the interrupt detection logic. When the pin direction is the input and the interrupt detection is enabled, the pin can accept interrupt from pad. The sensed state is stored in IntrMaskedState register (Masked by IntrEnable). Before turning on IntrEnable, the programmer can clear the masked state by writing a 1 to IntrClear to ensure the initial state.

22.1.1.8 IntrRawState

IntrRawState register is the raw result of the interrupt detection. If IntrEnable is enabled, IntrRawState register reflects the interrupt detection status. The programmer can poll this register to detect an interrupt.

22.1.1.9 IntrMaskedState

IntrMaskedState register is the masked result of the interrupt detection. The IntrMaskedState register is controlled by IntrEnable, IntrRawState, and IntrMask registers.

22.1.1.10 IntrMask

IntrMask register is the mask register of the interrupt detection. It masks the IntrRawState register. For example, if IntrEnable[0] = 1, IntrRawState[0] = 1, and IntrMask[0] = 1, then IntrMaskedState[0] will never change to 1.

22.1.1.11 IntrClear

IntrClear is the bit operation logic. If some bits of the write data are set, the related bits of IntrMaskedState and IntrRawState will be cleared.

22.1.1.12 IntrTrigger

IntrTrigger register indicates the interrupt trigger method of each pin. If the IntrTrigger is 0, the interrupt is triggered at the edge; otherwise, the interrupt is triggered at the level.

22.1.1.13 IntrBoth

IntrBoth register indicates the edge trigger is by both-edge or single-edge. If the IntrTrigger indicates the edge trigger and IntrBoth is 0, the interrupt edge trigger is done by the single edge. If the IntrTrigger indicates the edge trigger and IntrBoth is 1, the interrupt edge trigger is done by both edges.

22.1.1.14 IntrRiseNeg

IntrRiseNeg register indicates whether the edge trigger is at the rising edge or falling edge. If the IntrTrigger is the edge trigger, IntrBoth is the single edge and IntrRiseNeg is 0, interrupt edge trigger is done at the rising edge. If IntrTrigger is the edge trigger, IntrBoth is single and IntrRiseNeg is 1, the interrupt edge trigger is done at the falling edge.

22.1.1.15 BounceEnable

BounceEnable register controls the bounce function. If the BounceEnable is on, the interrupt detection is sampled by the extended clock. The extension number is controlled by the BouncePreScale register.

22.1.1.16 BouncePreScale

BouncePreScale register is an automatic register to indicate for the bounce timer. It can extend the APB_CLK to the BouncePreScale cycles. This can be used to adjust the interrupt sample clock period in different machine. The reset value is 0x7D0, with the APB clock frequency of 48 MHz the disbounce clock will be divided by (0x7D0+1) to 23.99 kHz.

23 TechIO

Offsets: 0xC0000000

All functionality in this chapter which is written in italic style is special function and is not completly tested.

23.1 IO-Pinning

Table 23-1 Input Mapping

Pin name	Digital	Counter
Tech_input(0)	Input0	Counter0 trackA
Tech_input(1)	Input1	Counter0 trackB
Tech_input(2)	Input2	Counter0 third input
Tech_input(3)	Input3	Counter1 trackA
Tech_input(4)	Input4	Counter1 trackB
Tech_input(5)	Input5	Counter1 third input
Tech_input(6)	Input6	Counter2 trackA
Tech_input(7)	Input7	Counter2 trackB
Tech_input(8)	Input8	Counter2 third input
Tech_input(9)	Input9	Counter3 trackA
Tech_input(10)	Input10	Counter3 trackB
Tech_input(11)	Input11	Counter3 third input
Tech_input(12)	Input12	Counter0 fourth input
Tech_input(13)	Input13	Counter1 fourth input
Tech_input(14)	Input14	Counter2 fourth input
Tech_input(15)	Input15	Counter3 fourth input
Tech_input(16)	Input16	Counter0 reset input
Tech_input(17)	Input17	Counter1 reset input
Tech_input(18)	Input18	Counter2 reset input
Tech_input(19)	Input19	Counter3 reset input
Tech_input(20)	Input20	-
Tech_input(21)	Input21	-
Tech_input(22)	Input22	-
Tech_input(23)	Input23	-
Tech_input(24)	Input24	-
Tech_input(25)	Input25	-

Table 23-2 Output Mapping

Pin name	Digital	Counter	PWM
Tech_output(0)	Output0	Counter0	PWM0
Tech_output(1)	Output1	Counter1	PWM1
Tech_output(2)	Output2	Counter2	PWM2
Tech_output(3)	Output3	Counter3	PWM3
Tech_output(4)	Output4	-	-
Tech_output(5)	Output5	-	-

Pin name	Digital	Counter	PWM
Tech_output(6)	Output6	-	-
Tech_output(7)	Output7	-	-
Tech_output(8)	Output8	-	-
Tech_output(9)	Output9	-	-
Tech_output(10)	Output10	-	-
Tech_output(11)	Output11	-	-
Tech_output(12)	Output12	-	-
Tech_output(13)	Output13	-	-
Tech_output(14)	Output14	-	-
Tech_output(15)	Output15	-	-
Tech_output(16)	Output16	-	-
Tech_output(17)	Output17	-	-
Tech_output(18)	Output18	-	-
Tech_output(19)	Output19	-	-

23.2 Digital input and output data

- Address: 0x00..0x03
- Reset value: 0x0
- Access: Read/Write

Table 23-3 digital input/output data

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	Digital input				R	0x0
	Res	Digital output			W	0x0

Table 23-4 digital input/output bits

Bit	Name	Type	Description
0	Input0	R	Digital input0
	Output0	W	Digital output0
1	Input1	R	Digital input1
	Output1	W	Digital output1
..	..	..	..
	..	..	..
19	Input19	R	Digital input19
	Output19	W	Digital output19
20	Input20	R	Digital input20
	Res	W	Reserved
..	..	..	..
	..	..	..
25	Input25	R	Digital input25
	Res	W	Reserved
26	Res	RW	Reserved

23.3 Filter

Offsets: 0x040

23.3.1 Parameter register

Table 23-5 Parameter Mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	Res	Filter input1	Res	Filter input0	RW	0x000D000D
0x04	Res	Filter input3	Res	Filter input2	RW	0x000D000D
0x08	Res	Filter input5	Res	Filter input4	RW	0x000D000D
0x0C	Res	Filter input7	Res	Filter input6	RW	0x000D000D
0x10	Res	Filter input9	Res	Filter input8	RW	0x000D000D
0x14	Res	Filter input11	Res	Filter input10	RW	0x000D000D
0x18	Res	Filter input13	Res	Filter input12	RW	0x000D000D
0x1C	Res	Filter input15	Res	Filter input14	RW	0x000D000D
0x20	Res	Filter input17	Res	Filter input16	RW	0x000D000D
0x24	Res	Filter input19	Res	Filter input18	RW	0x000D000D
0x28	Res	Filter input21	Res	Filter input20	RW	0x000D000D
0x2C	Res	Filter input23	Res	Filter input22	RW	0x000D000D
0x30	Res	Filter input25	Res	Filter input24	RW	0x000D000D

23.3.2 Register description

23.3.2.1 Filter input register

- Address: 0x00..0x23
- Reset value: 0x0D
- Access: Read/Write

The input filter should eliminate glitches/spikes on the input signal. The duration depends on the value of the filter input register.
The duration is calculated using following formula:

$t/us = 1/3*2^{filter\_input}$ e.g. filter_input = 4; t=5,333us

The filter_input value is valid from 0 to 18.

23.4 Alarm system digital input

Offsets: 0x080

23.4.1 Parameter and status register

Table 23-6 Parameter Mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	Alarm_rising_edge				RW	0x0
0x04	Alarm_falling_edge				RW	0x0

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x08	Reserved			Diag_en_dio	W	-
0x0C	Reserved		Alarm_typ_tech	Alarm_typ_dio	R	0x0
0x10	Reserved		Notify_tech	Nofify_dio	W	-
0x14	Reserved		Ack_tech	Ack_dio	W	-
0x18	Reserved				-	-
0x1C	Reserved				-	-
0x20	Palarm_dio0				R	undef.
0x24	Palarm_dio1				R	undef.
0x28	Palarm_dio2				R	undef.
0x2C	Dalarm_dio0				R	undef.
0x30	Dalarm_dio1				R	undef.
0x34	Dalarm_dio2				R	undef.
0x38	Dalarm_dio3				R	undef.
0x3C	Reserved (do not access)				-	-

23.4.2 Register description

23.4.2.1 Alarm_rising_edge

- Address: 0x00..0x03
- Reset value: 0x0
- Access: Read/Write

Table 23-7 alarm_rising_edge

Bit	Name	Type	Description
0	Input0	RW	if a rising edge on the input0 occurs 0 = no alarm 1 = alarm released
1	Input1	RW	if a rising edge on the input1 occurs 0 = no alarm 1 = alarm released
..	..	..	..
25	Input25	RW	if a rising edge on the input25 occurs 0 = no alarm 1 = alarm released
31..26	Res	R	0

23.4.2.2 Alarm_falling_edge

- Address: 0x04..0x07
- Reset value: 0x0
- Access: Read/Write

Table 23-8 alarm_falling_edge

Bit	Name	Type	Description

Bit	Name	Type	Description
0	Input0	RW	if a falling edge on the input0 occurs 0 = no alarm 1 = alarm released
1	Input1	RW	if a falling edge on the input1 occurs 0 = no alarm 1 = alarm released
..	..	..	..
25	Input25	RW	if a falling edge on the input25 occurs 0 = no alarm 1 = alarm released
31..26	Res	R	0

23.4.2.3 Diag_en_dio

- Address: 0x0B
- Reset value: 0x0
- Access: Write

Table 23-9 diag_en_dio

Bit	Name	Type	Description
0	Diag_en_dio	W	0 = digital diagnostic alarm disabled 1 = digital diagnostic alarm enabled
7..1	res		Reserved

23.4.2.4 Alarm_typ_tech

- Address: 0x0E
- Reset value: 0x0
- Access: Read

This register is mirrored from the counter area (23.8.2.2) to get the possibility of reading both alarm types with one read access.

Table 23-10 alarm_typ_tech

Byte	Name	Type	Description
0x0E	Alarm_typ_tech	R	Alarm type for counter/PWM and SSI 0 = no alarm pending 1 = process alarm 2 = diagnostic alarm pending No other values possible.

23.4.2.5 Alarm_typ_dio

- Address: 0x0F
- Reset value: 0x0

- Access: Read

Table 23-11 alarm_typ_dio

Byte	Name	Type	Description
0x0F	Alarm_typ_dio	R	Alarm type for digital inputs 0 = no alarm pending 1 = process alarm 2 = diagnostic alarm pending No other values possible.

23.4.2.6 Notify_tech

- Address: 0x12
- Reset value: 0x0
- Access: Write

This register is mirrored from the counter area (23.8.2.3) to get the possibility of writing both alarm notify with one write access.

Table 23-12 notify_tech

Byte	Name	Type	Description
0x12	notify_tech	W	After an alarm was set the alarm must be mask with a notify write. The alarm lines change back in idle mode after writing the notify register. 1 = process alarm notify 2 = diagnostic alarm notify No other values allowed

23.4.2.7 Notify_dio

- Address: 0x13
- Reset value: 0x0
- Access: Write

Table 23-13 notify_dio

Byte	Name	Type	Description
0x13	Notify_dio	W	After an alarm was set the alarm must be mask with a notify write. The alarm lines change back in idle mode after writing the notify register. 1 = process alarm notify 2 = diagnostic alarm notify No other values allowed

23.4.2.8 Ack_tech

- Address: 0x16
- Reset value: 0x0
- Access: Write

This register is mirrored from the counter area (23.8.2.4) to get the possibility of writing both alarm acknowledge with one write access.

Table 23-14 ack_tech

Byte	Name	Type	Description
0x16	ack_tech	W	After the alarm data were read the alarm must be acknowledge getting a new alarm from this module and this priority level. 1 = process alarm acknowledge 2 = diagnostic alarm acknowledge No other values allowed

23.4.2.9 Ack_dio

- Address: 0x17
- Reset value: 0x0
- Access: Write

Table 23-15 ack_dio

Byte	Name	Type	Description
0x17	ack_dio	W	After the alarm data were read the alarm must be acknowledge getting a new alarm from this module and this priority level. 1 = process alarm acknowledge 2 = diagnostic alarm acknowledge No other values allowed

23.4.2.10 Palarm_dio0

- Address: 0x20..0x23
- Reset value: undef.
- Access: Read

Table 23-16 Palarm_dio0

Bit	Name	Type	Description
0	Alarm on input0	R	0 = no alarm 1 = a rising or falling edge occurred
1	Alarm on input1	R	0 = no alarm 1 = a rising or falling edge occurred
..	..	..	..
25	Alarm on input25	R	0 = no alarm 1 = a rising or falling edge occurred
31..26	Res	R	0

23.4.2.11 Palarm_dio1

- Address: 0x24..0x27
- Reset value: undef.
- Access: Read

Table 23-17 Palarm_dio1

Bit	Name	Type	Description

Bit	Name	Type	Description
0	input0	R	0 = input0 was low after the alarm occurred 1 = input0 was high after the alarm occurred
1	input1	R	0 = input1 was low after the alarm occurred 1 = input1 was high after the alarm occurred
..	..	..	..
25	input25	R	0 = input25 was low after the alarm occurred 1 = input25 was high after the alarm occurred
31..26	Res	R	0

23.4.2.12 Palarm_dio2

- Address: 0x28..0x2B
- Reset value: undef.
- Access: Read

This register is always read as 0xF0000000.

23.4.2.13 Dalarm_dio0

- Address: 0x2C..0x2F
- Reset value: undef.
- Access: Read

Table 23-18 Dalarm_dio0

value	Type	Description
0x00001F00	R	Only the last going diagnostic alarm
0x40001F0B	R	Every diagnostic alarm; except the last going diagnostic alarm

23.4.2.14 Dalarm_dio1

- Address: 0x30..0x33
- Reset value: undef.
- Access: Read

Table 23-19 Dalarm_dio1

offset	Byte 0		Byte 1	Byte 2	Byte 3	Designation
0x30 ... 0x33	Bit	Description	0x08	0x08	0x70	Dalarm_dio1
	0	Diag alarm on input0, 1, 2 or 3 '0' = not occurred '1' = occurred				
	1	Diag alarm on input4, 5, 6 or 7 '0' = not occurred '1' = occurred				
	2	Diag alarm on input8,				

offset	Byte 0		Byte 1	Byte 2	Byte 3	Designation
		9, 10 or 11 '0' = not occurred '1' = occurred				
	3	Diag alarm on input12, 13, 14 or 15 '0' = not occurred '1' = occurred				
	4	Diag alarm on input16, 17, 18 or 19 '0' = not occurred '1' = occurred				
	5	Diag alarm on input20, 21,2 2 or 23 '0' = not occurred '1' = occurred				
	6	Diag alarm on input24 or 25 '0' = not occurred '1' = occurred				
	7	0				

23.4.2.15 Dalarm_dio2

- Address: 34..0x37
- Reset value: undef.
- Access: Read

Table 23-20 Dalarm_dio2

Bit	Name	Type	Description
0	Diagnostic alarm input0	R	0 = no alarm 1 = occurred
1	Res	R	Always read as zero
2	Diagnostic alarm input1	R	0 = no alarm 1 = occurred
3	Res	R	Always read as zero
..	..	..	..
30	Diagnostic alarm input15	R	0 = no alarm 1 = occurred
31	Res	R	Always read as zero

23.4.2.16 Dalarm_dio3

- Address: 0x38..0x3B
- Reset value: undef.
- Access: Read

Table 23-21 Dalarm_dio2

Bit	Name	Type	Description

Bit	Name	Type	Description
0	Diagnostic alarm input16	R	0 = no alarm 1 = occurred
1	Res	R	Always read as zero
2	Diagnostic alarm input17	R	0 = no alarm 1 = occurred
3	Res	R	Always read as zero
..	..	..	..
18	Diagnostic alarm input25	R	0 = no alarm 1 = occurred
31..19	Res	R	Always read as zero

23.5 Counter

The ANTAIOS contains 4 independent counter channels. The maximum count frequency is 1.5 MHz. There are different operating modes, four encounter modes, seven alarm sources and some optional function input and one function output.

23.5.1 Register mapping

23.5.1.1 Status and configuration register

Offsets: counter: 0x100

Table 23-22 Status and configuration Mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	Count/latch value0				R	0x0
	Config0				W	0x0
0x04	Status0				R	0x21000000
	Config1				W	0x0
0x08	Count/latch value1				R	0x0
	Config2				W	0x0
0x0C	Status1				R	0x21000000
	Config3				W	0x0
0x10	Count/latch value2				R	0x0
0x14	Status2				R	0x21000000
0x18	Count/latch value3				R	0x0
0x1C	Status3				R	0x21000000

23.5.1.2 Parameter register

Offsets: counter0: 0xC0000120

counter1: 0xC0000140
counter2: 0xC0000160
counter3: 0xC0000180

Table 23-23 Parameter Mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	mode				RW	0x0
0x04	compare value				RW	0x0
0x08	load value				RW	0x0
0x0C	end value				RW	0x0
0x10	count value				RW	0x0
0x14	res			hysteresis value	RW	0x0

23.5.2 Register description

23.5.2.1 CounterX Count/Latch value

- Address: counter0 0x00…0x03
  counter1 0x08…0x0B
  counter2 0x10…0x13
  counter3 0x18…0x1B
- Reset value: 0x0
- Access: Read

The register holds the current count value or the current latch value. When the count value is displayed bit16 of the counter status register (23.5.2.2) is read as zero. Otherwise the latch value is read at the register.
The count and latch value is a 32-bit width signed value.

Table 23-24 count/latch value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x00 … 0x03	Count/Latch value				Current count or latch value

23.5.2.2 CounterX Status register

- Address: counter0 0x04…0x07
  counter1 0x0C…0x0F
  counter2 0x14…0x17
  counter3 0x1C…0x1F
- Reset value: 0x21000000
- Access: Read

Table 23-25 Status register

Bit	Name	Type	Description
0..14	Res	R	Always read as zero
15	Sync	R	Sync bit is 0 = if no synchronization event occurred at the input or was reset with Res 1 = if a synchronization event occurred at the input
16	Count_latch	R	count_latch bit is 0 = when the count value is displayed at count/latch value 1 = when the latch value is displayed at count/latch value
17	Control_do	R	Control_do bit is 0 = if output is controlled by the digital function 1 = if output is controlled by the technological function
18	SW_gate	R	Status of the software gate 0 = gate closed 1 = gate open
19	Reset_input	R	Status of the reset input 0 = reset inactive 1 = reset active
20	HW_input	R	Status of the hardware input 0 = gate closed 1 = gate open
21	Gate	R	Status of the internal gate 0 = gate closed 1 = gate open
22	Do	R	Status of the digital ouput 0 = low 1 = high
23	Count_down	R	Indicates the last count direction was 0 = up or no counting occurred since start up 1 = down
24	Count_up	R	Indicates the last count direction was 0 = dwon or no counting occurred since start up 1 = up
25	Compare	R	Compare status indicates whether the comparison condition for the comparator 0 = wasn’t met or was reset again with Res 1 = was or is met.
26	End	R	Indicates that the end value 0 = wasn’t reached or was reset with Res 1 = is or was reached
27	Overflow	R	Indicates that 0 = no overflow occurred or was reset with Res 1 = an overflow occurred
28	Underflow	R	Indicates that 0 = no underflow occurred or was reset with Res 1 = an underflow occurred

Bit	Name	Type	Description
29	Zero_mark	R	Zero mark bit is 0 = no zero crossing occurred or was reset with Res 1 = if the count value is/was zero **ATTENTION:** This bit is only set when counting without a main count direction
30	Latch_input	R	Status of the latch input 0 = low 1 = high A new latch value is only stored with the rising edge 0 -> 1
31	New_latch	R	New latch indicates that 0 = no new latch value available 1 = a new latch value was stored The bit change from 1 to 0 if count_latch bit is 1 and the count/latch value is read.

23.5.2.3 CounterX Control register

- Address: counter0 0x00…0x01
  counter1 0x04…0x05
  counter2 0x08…0x09
  counter3 0x0C…0x0D
- Reset value: 0x0
- Access: Write

Table 23-26 control register

Bit	Name	Type	Description
0	Count _read	W	Count/latch register displays 0 = no effect 1 = count value
1	Do_set	W	Output is controlled by 0 = no effect 1 = counter function
2	SW_gate_set	W	Software gate is 0 = no effect 1 = open
3	Sync_set	W	Synchronization is 0 = no effect 1 = enabled
4	Res	W	Reserved
5	Count_set	W	Set current count value with the count value parameter (23.5.2.5)
6	Rel_sts	W	The release status bit reset the following status bits 0 = no effect 1 = Compare, Overflow, Underflow, Zero_mark and End
7	Res	W	Reserved
8	Latch_read	W	Count/latch register displays 0 = no effect

Bit	Name	Type	Description
			1 = latch value (preferred for bit0)
9	DO_reset	W	Output is controlled by 0 = no effect 1 = digital function (preferred for bit1)
10	SW_gate_reset	W	Software gate is 0 = no effect 1 = closed (preferred for bit2)
11	Sync_reset	W	Synchronization is 0 = no effect 1 = disabled (preferred for bit3)
15..12	Res	W	Reserved

23.5.2.4 Mode register

- Address: 0x00…0x03
- Reset value: 0x0
- Access: Read/Write

This register configures the main technology function and the special counter modes

Table 23-27 Configuration register

Bit	Name	Type	Description
2..0	Technology function	RW	Choose the technology function 000 = technology function disabled 001 = counter, rotary encoder single 010 = counter, rotary encoder double 011 = counter, rotary encoder quadruple 100 = counter, up/down 101 = PWM 110 = reserved 111 = reserved
6..3	Third input	RW	Choose a function of the third input pin (available only at counter) 0000 = no function 0001 = HW-Gate input *0010 = Monoflop HW-Gate input* 0100 = Latch input 1000 = Reset input
7	Gate function	RW	Function of the gate (available only at counter) 0 = abort 1 = interrupt
10..8	Comparison	RW	Comparison bit is set … (available only at counter) 000 = never 001 = counter value >= comparison value 010 = counter value <= comparison value 100 = counter value = comparison value
11	Inverted	RW	Track B count direction inverted (available only at counter)

Bit	Name	Type	Description
			0 = track B not inverted 1 = track B inverted
14..12	Reset counter	RW	counter value is reset … (available only at counter) 000 = never 010 = high level 011 = edge 0-1 periodic 110 = edge 0-1 once
15	Fix	RW	0
21..16	Counter function	RW	Counter function (available only at counter) 000000 = count endless 000001 = count once, Main count direction up 000010 = count once, Main count direction down 000100 = count once, Main count direction none 001000 = count periodic, Main count direction up 010000 = count periodic, Main count direction down 100000 = count periodic, Main count direction none
23..22	Fourth input	RW	Choose a function of the fourth input pin (available only at counter) 00 = no function 01 = Latch input 10 = Reset input
24	HW-Gate open alarm	RW	Release an alarm if the HW-Gate has been opened (available only at counter) 0 = disable 1 = enable
25	HW-Gate close alarm	RW	Release an alarm if the HW-Gate has been closed (available only at counter) 0 = disable 1 = enable
26	Overflow alarm	RW	Release an alarm if the counter value overflows (available only at counter) 0 = disable 1 = enable
27	Underflow alarm	RW	Release an alarm if the counter value underflows (available only at counter) 0 = disable 1 = enable
28	Compare alarm	RW	Release an alarm if the compare value reaches (available only at counter) 0 = disable 1 = enable
29	End value alarm	RW	Release an alarm if the end value reaches (available only at counter) 0 = disable 1 = enable
30	Latch alarm	RW	Release an alarm if the latch value has been set (available only at counter) 0 = disable 1 = enable
31	Reset latch	RW	Reset (set to zero) latch value with BASP (available only at counter)

Bit	Name	Type	Description
	value		0 = disable 1 = enable

23.5.2.5 Compare value

- Address: 0x04…0x07
- Reset value: 0x0
- Access: Read/Write

The register holds the current compare value.

Table 23-28 compare value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x04 … 0x07	Compare value				32 bit signed compare value

23.5.2.6 Load value

- Address: 0x08…0x0B
- Reset value: 0x0
- Access: Read/Write

The register holds the load value. The load value is applied as the new count value when specific events occur, depending on the set operating mode.

Table 23-29 load value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x08 … 0x0B	Load value				32 bit signed load value

23.5.2.7 End value

- Address: 0x0C…0x0F
- Reset value: 0x0
- Access: Read/Write

The register holds the current end value. The counter stops or start again from the load value when it reaches the end value, depending on the set operation mode.

Table 23-30 compare value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x0C … 0x0F	End value				32 bit signed end value

23.5.2.8 Count value

- Address: 0x10…0x13
- Reset value: 0x0
- Access: Read/Write

This register holds a 32 bit signed count value. The value could set as current value for the counter with bit5 of the counter control register (23.5.2.3)

Table 23-31 count value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x10 ... 0x13	Count value				32-bit signed count value

23.5.2.9 Hysteresis value

- Address: 0x17
- Reset value: 0x0
- Access: Read/Write

The hysteresis is used to avoid frequent output switching actions if the count value lies within the range of the comparison value. The hysteresis is used also for the zero_mark and the over-/underflow.

Table 23-32 hysteresis value

Address	Byte Position Byte 0	Designation
0x17	Hysteresis value	8-bit unsigned hysteresis

23.5.3 Counter functions

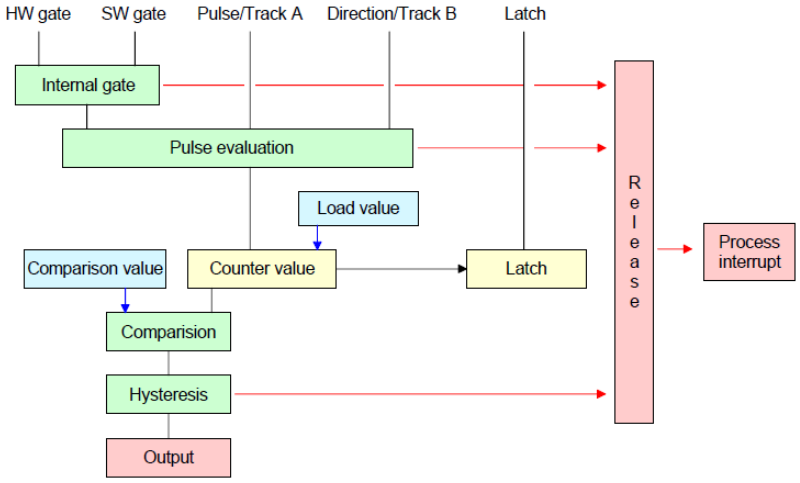
It is possible to count forward and backwards and choose between the following counter functions:

- Count endless, e.g. distance measuring with incremental encoder
- Count once, e.g. count to a maximum limit
- Count periodic, e.g. count with repeated counter process

In the operating modes "Count once" and "Count periodic" you may parametrize a counter range by defining a start and end value. For every counter are additional functions available. You can select gate function, comparison, hysteresis and process interrupt.

The illustration shows how the additional functions influence the counting behavior. The following pages describe these functions in detail:

Figure 23-1 Counter overview



23.5.3.1 Main counting direction

You have the opportunity to define a main counting direction separately for every counter. If "none" is chosen, the complete counting range is available:

Table 23-33 main counting direction limits

Limits	Valid value range
Lower count limit	- 2 147 483 648 (-2^{31})
Upper count limit	+ 2 147 483 647 (2^{31}-1)

Main counting direction upwards:
This is the upper limit of the count range. The counter starts from 0 or the load value in positive direction until the end value -1 and jumps with the next positive encoder pulse back to the load value.

Main counting direction downwards:
This is the lower limit of the count range. The counter starts from 0 or load value in negative direction until the end value +1 and jumps with the next negative encoder pulse back to the load value.

23.5.3.2 Gate function

You have the possibility to control the counter via two gates.

- A **software gate** which is controlled by the user program.
  To open the software gate set the control-bit SW_gate_set to one. To close the software gate set the control-bit SW_gate_reset to one.
- A **hardware gate** which can be parametrize on the third input via the “mode” register. (23.5.2.4)
  To open the gate, a positive edge at the digital input is necessary. The gate is closed with a negative edge.

The logical AND between the hardware and software gate represents the **internal gate** (I gate).

Cancel counting operation:
The count process starts after closing and restart of the gate with the load value.

Stop counting operation:
The count process is stopped when the gate is closed and resumes at the last actual value after the gate is re-opened again.

The internal Gate (I-gate) is influenced like following tables:

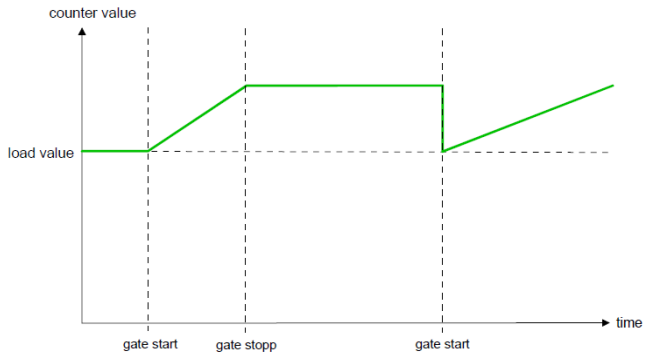
Table 23-34 gate function

SW gate	HW gate	Influences I gate
0	with edge 0-1	0
1	with edge 0-1	1
with edge 0-1	1	1
with edge 0-1	0	0
with edge 0-1	de-activated	1

Gate control canceling or interrupting
The parameterization defines if the gate interrupts or cancels the counter process.

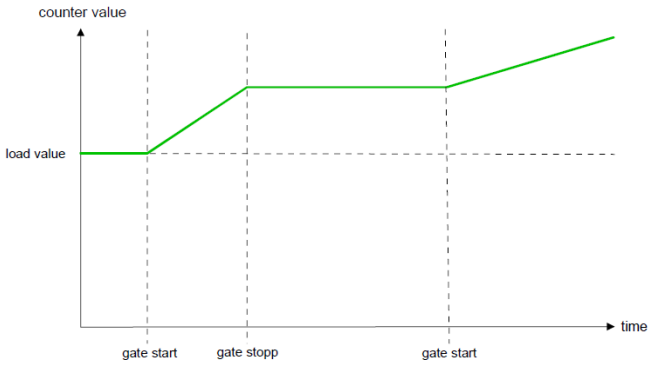
- When set gate canceled operation the counter starts counting with the load value after gate restarted.

Figure 23-2 gate canceled



- During set gate interrupted operation, the counter resumed counting at the last actual count value after the gate was closed.

Figure 23-3 gate interrupted



Gate control via SW gate, cancelling:

Table 23-35 software gate cancel

SW gate	HW gate	Reaction counter
edge 0-1	de-activated	restart with *load value*

Gate control via SW gate, interrupting:

Table 23-36 software gate interrupt

SW gate	HW gate	Reaction counter
edge 0-1	de-activated	continue

Gate control via SW/HW gate, cancelling:

Table 23-37 software/hardware gate cancel

SW gate	HW gate	Reaction counter
edge 0-1	1	continue
1	edge 0-1	restart with *load value*

Gate control via SW/HW gate, interrupting:

Table 23-38 software/hardware gate interrupt

SW gate	HW gate	Reaction counter
edge 0-1	1	continue
1	edge 0-1	continue

Gate control in “Single Count” mode

If the internal gate was closed automatically, it can be reopened by the following conditions:

Table 23-39 gate control “single count”

SW gate	HW gate	Reaction I gate
1	edge 0-1	1
edge 0-1 (after edge 0-1 at HW gate)	1	1

23.5.3.3 Comparison function

The compare value is to be pre-defined by the output area. The comparison bit may be found at the counter status at STS_COMP. Please consider that the bit STS_COMP may only be influenced when in the counter status the bit STS_CTRL_COMP is set. The following behavior for the comparison bit may be pre-defined via the parameterization:

- no comparison: Comparison bit is not influenced
- Counter value ≥ comparison value: comparison bit is set
- Counter value ≤ comparison value: comparison bit is set
- Counter value = comparison value: comparison bit is set

No comparison
The comparison bit is not influenced.

Comparison bit is set when counter value ≥ comparison value
The comparison bit remains set as long as the count value is higher or equal to the comparison value.

Comparison bit is set when counter value ≤ comparison value
The comparison bit remains set as long as the count value is lower or equal to the comparison value.

Comparison bit is set when counter value = comparison value
When the counter reaches the comparison value the comparison bit is set.
The comparison bit remains set as long as the comparison condition is met.
When you've set a main counting direction the comparison bit is only set at reaching the comparison value from the main counting direction.

Note!
The bit “compare” is set together with the bit “do” in the counter status. In comparison to the bit “do” the bit “compare” remains set as long as it is released by the bit “rel_sts” in the control word.

23.5.3.4 Latch function

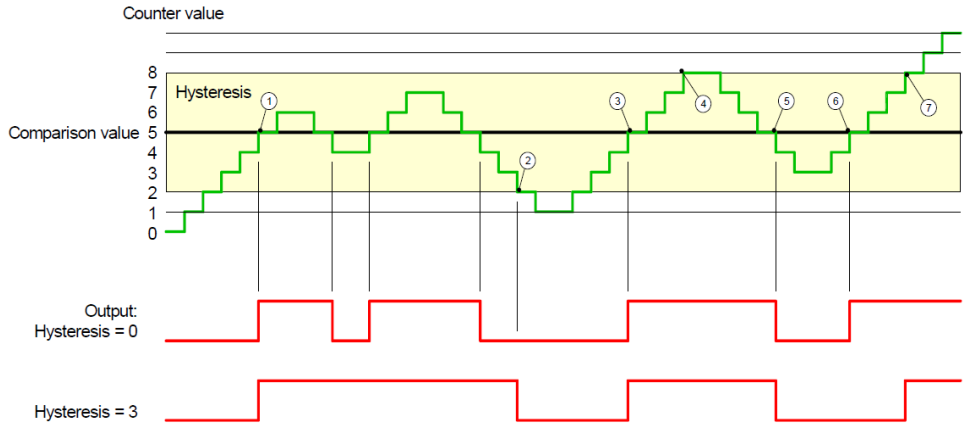
With a rising edge 0-1 on the "Latch" input of the counter, the actual counter value is stored in the latch register. You may access the latch register via the input area. After “basp” was set latch may be cleared or hold it last value. This is parameterized with bit31 of the “mode” register (23.6.2.2).
Latch function could be enabled either on input three or four. (See “mode” register 23.6.2.2)

23.5.3.5 Hysteresis

The hysteresis can be used to prevent the output and the interrupt from toggling, if the counter value fluctuates in a small range around the comparison value. You may set a range of 0 to 255. The settings 0 and 1 deactivate the hysteresis. The hysteresis also influences the zero crossing, over- and underflow. An activated hysteresis remains active after a change. The new hysteresis value is only used at the next hysteresis event. The following pictures illustrate the output behavior for hysteresis 0 and hysteresis 3 for the according conditions:

Output switch on if counter value ≥ comparison value

Figure 23-4 Counter value greater than compare value



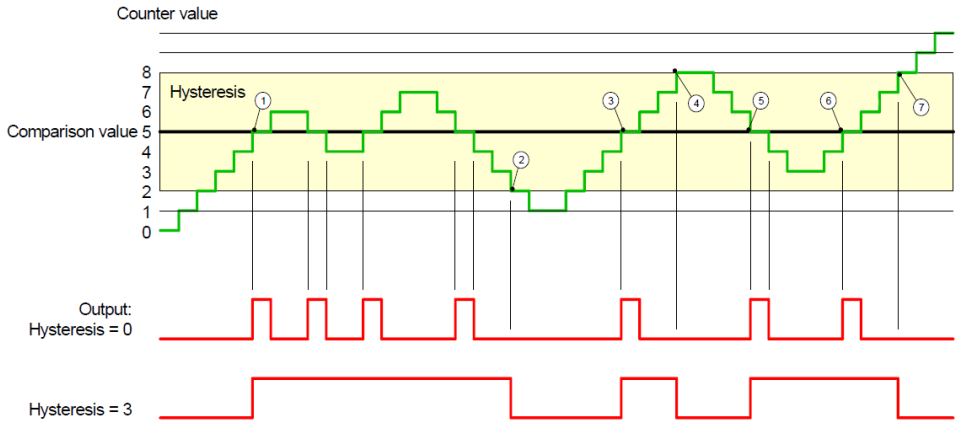
Description for Hysteresis = 3:

(1) Counter value ≥ comparison value → output is set and hysteresis is activated
(2) counter value < comparison value → output is reset and hysteresis is deactivated
(3) counter value ≥ comparison value → output is set and hysteresis is activated
(4) counter value ≥ comparison value → output remains set and hysteresis deactivated
(5) counter value < comparison value → output is reset and hysteresis is activated
(6) counter value ≥ comparison value → output isn't set and hysteresis is already active
(7) counter value ≥ comparison value → output is set and hysteresis is deactivated

The hysteresis is activated if the comparison condition is reached. During hysteresis is active comparison result remains unchanged. When the count value exits the hysteresis range the output then switches according to the comparison result.

Effect when "pulse at comparison value" select:

Figure 23-5 Pulse at comparison



Description for Hysteresis = 3:

(1) Counter value = comparison value → output is set and hysteresis activated
(2) Leave hysteresis range → output is reset
(3) Counter value = comparison value → output is set and hysteresis activated
(4) Leave hysteresis range → output is reset
(5) Counter value = comparison value → output is set and hysteresis activated
(6) Counter value = comparison value → output remains set because hysteresis is already active
(7) Leave hysteresis range → output is reset

The hysteresis is activated if the comparison condition is reached. During hysteresis is active comparison result remains unchanged and output is set. When the count value exits the hysteresis range the hysteresis is no longer active and the output is reset. The output is only set as long as the hysteresis is active.

23.5.3.6 Count continuously

In this operating mode, the counter counts from 0 or the load value.
When the counter counts upward and reaches the upper limit, it jumps to the lower limit, with the next positive count pulse and resumes the count from there.
When the counter counts downward and reaches the lower limit, it jumps to the upper limit, with the next negative count pulse and resumes the count from there.
The count limits are set to maximum range.

Table 23-40 count continuously

Limits	Valid value range
Lower count limit	- 2 147 483 648 (-2^{31})
Upper count limit	+ 2 147 483 647 (2^{31}-1)

When an overflow or underflow occurred the status bits “overflow” respectively “underflow” are set. These bits remain set until these are reset with RES_SET in the control word.
Additionally a process interrupt is triggered, when it is enabled.

Figure 23-6 Count continuously

23.5.3.7 Single cycle count

No main counting direction!

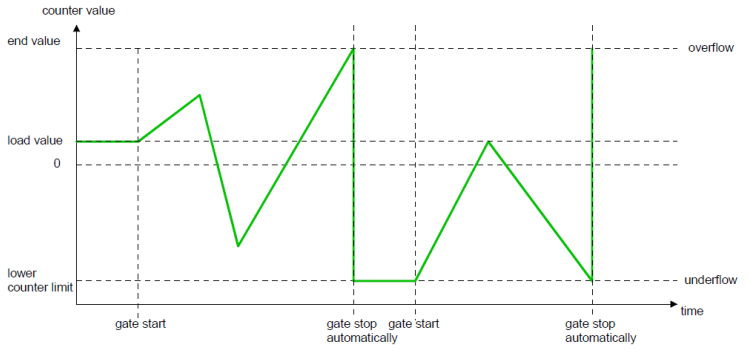
- The counter counts up or down starting with the load value.
- The count limits are set to the maximum range.
- At over- or underflow at the count limits, the counter jumps to the according other count limit and the internal gate is closed automatically. The status bits “overflow” respectively “underflow” are set and a process interrupt is triggered, when it is enabled.
- To restart the count process, you have to re-open the internal gate.
- At stopping gate control, the count resumes with the last recent count value.
- At canceling gate control, the count starts with the load value.

Table 23-41 count once without main direction

Limits	Valid value range
Lower count limit	- 2 147 483 648 (-2^{31})
Upper count limit	+ 2 147 483 647 (2^{31}-1)

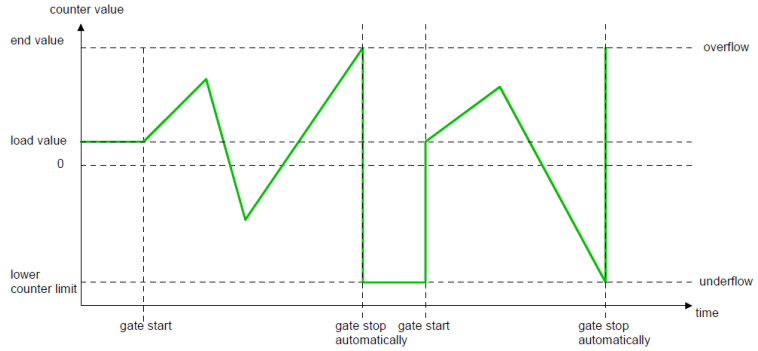
Stopping gate control:

Figure 23-7 Single cycle count: gate control stopped



Canceling gate control:

Figure 23-8 Single cycle count: gate control canceled



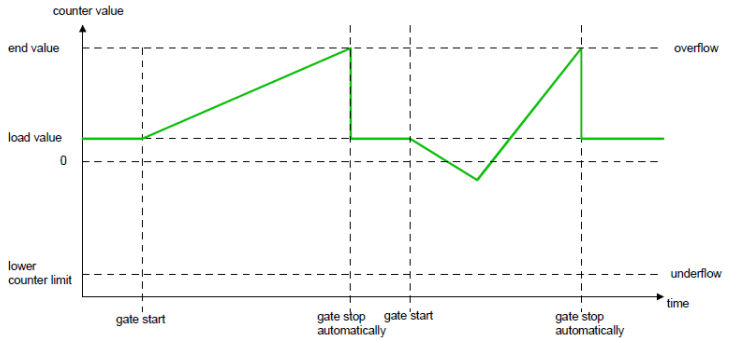
Main counting direction up

- The counter counts up or down starting with the load value.
- When the counter reaches the end value +1 in positive direction, it jumps to the load value at the next positive count pulse and the gate is automatically closed. If enabled additionally a process interrupt is triggered.
- To restart the count process, you must re-open the internal gate. The counter starts with the load value.
- You may count beyond the lower counter limit.

Table 23-42 Single cycle count upwards

Limits	Valid value range
Limit value	-2 147 483 647 (-2^{31}+1) to +2 147 483 647 (2^{31}-1)
Lower count limit	-2 147 483 648 (-2^{31})

Figure 23-9 Single cycle count: upwards



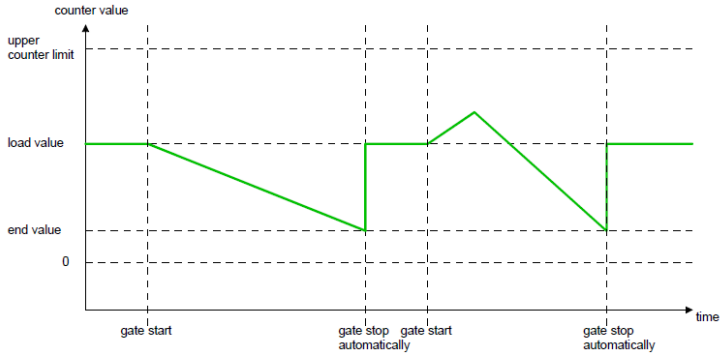
Main counting direction down:

- The counter counts up or down starting with the load value.
- When the counter reaches the end value +1 in negative direction, it jumps to the load value at the next negative count pulse and the gate is automatically closed. If enabled additionally a process interrupt is triggered.
- To restart the count process, you must re-open the internal gate. The counter starts with the load value.
- You may count beyond the upper counter limit.

Table 23-43 Single cycle count downwards

Limits	Valid value range
Limit value	-2 147 483 648 (-2^{31}) to +2 147 483 646 ($2^{31}-2$)
Upper count limit	+2 147 483 647 ($2^{31}-1$)

Figure 23-10 Single cycle count: downwards



23.5.3.8 Periodic count

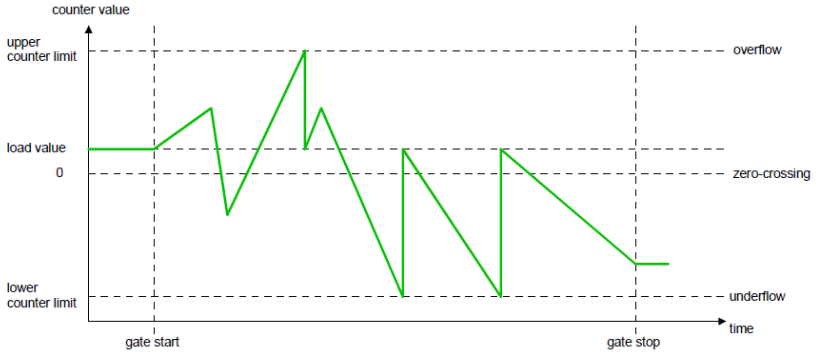
No main counting direction

- The counter counts up or down starting with the load value.
- At over- or underflow at the count limits, the counter jumps to the load value and resumes counting from there. If enabled additionally a process interrupt is triggered.
- The count limits are set to the maximum count range.

Table 23-44 Periodic count without main direction

Limits	Valid value range
Lower count limit	-2 147 483 648 (-2^{31})
Upper count limit	+2 147 483 647 ($2^{31}-1$)

Figure 23-11 Periodic count without main direction



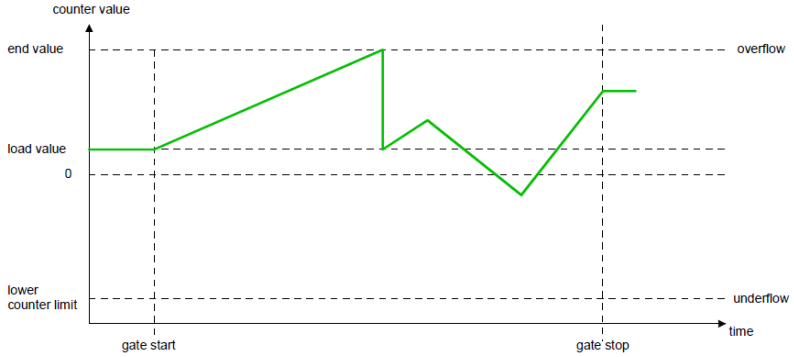
Main counting direction upwards

- The counter counts up or down starting with the load value.
- When the counter reaches the end value +1 in positive direction, it jumps to the load value at the next positive count pulse and resumes counting from there. If enabled additionally a process interrupt is triggered.
- You may count beyond the lower counter limit.

Table 23-45 Periodic count upwards

Limits	Valid value range
Limit value	-2 147 483 647 ($-2^{31}+1$) to +2 147 483 647 ($2^{31}-1$)
Lower count limit	-2 147 483 648 (-2^{31})

Figure 23-12 Periodic count: upwards



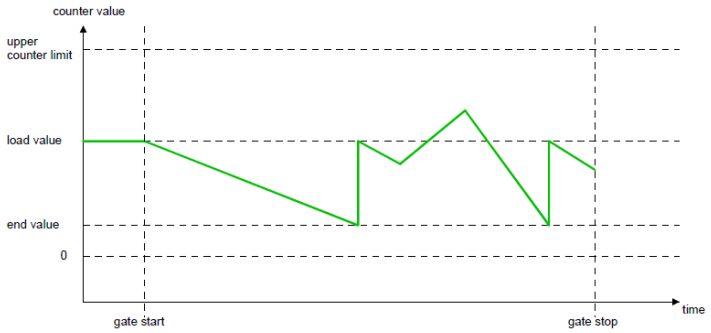
Main counting direction downwards

- The counter counts up or down starting with the load value.
- When the counter reaches the end value +1 in negative direction, it jumps to the load value at the next negative count pulse and resumes counting from there. If enabled additionally a process interrupt is triggered.
- You may count beyond the lower counter limit.

Table 23-46 Periodic count downwards

Limits	Valid value range
Limit value	-2 147 483 648 (-2^{31}) to +2 147 483 646 ($2^{31}-2$)
Lower count limit	+2 147 483 647 ($2^{31}-1$)

Figure 23-13 Periodic count: downwards



23.6 Pulse width modulation (PWM)

The ANTAIOS contains 4 independent PWM channels. The minimum pulse width is 10.41 ns and the minimum period duration is 41.66 ns. There are four possible operating modes for the PWM.

- Output of a pulse train; the output value corresponds to the ratio of the pulse duration to the period.
- *Output of a pulse with a specified pulse duration (max high time is ~1.3 seconds)*
- *Output of n pulses with a specifiable period and pulse duration.*
- *Output of n pulses with a specifiable period and pulse duration and write a second set of pulse duration and period into a shadow register. The output of the second value is used immediately after the first n pulses are finished.*

23.6.1 Register mapping

23.6.1.1 Status and configuration register

Offsets: PWM: 0x100

Table 23-47 Status and configuration Mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	reserved				R	0x0
0x04	reserved		PWM0 status		R	0x0
0x08	reserved				R	0x0
0x0C	reserved		PWM1 status		R	0x0
0x10	reserved				R	0x0
0x14	reserved		PWM2 status		R	0x0
0x18	reserved				R	0x0
0x1C	reserved		PWM3 status		R	0x0

23.6.1.2 Parameter register

Offsets: PWM0: 0xC0000120
PWM1: 0xC0000140
PWM2: 0xC0000160
PWM3: 0xC0000180

Table 23-48 Parameter Mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	Mode				RW	0x0
0x04	New period				W	0x0
	Current period				R	0x0
0x08	New burst				W	0x0
	Current burst				R	0x0

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x0C	New duration				W	0x0
	Current duration				R	0x0
0x10	Reserved		Configuration		W	0x0

23.6.2 Register description

23.6.2.1 PWMx status register

- Address: PWM0 0x06…0x07
  PWM1 0x0E…0x0F
  PWM2 0x16…0x17
  PWM3 0x1E…0x1F
- Reset value: 0x0
- Access: Read

Table 23-49 Status register

Bit	Name	Type	Description
0	Burst enable	R	Burst enable register This bit is set to '1' if a certain number of pulse trains is configured (burst mode) This bit is set to '0' if a continuous pulse train is configured (continuous mode)
1	PWM running	R	PWM running register This bit displays the status of the PWM module It is read as '1' if the PWM is running and '0' if the PWM is stopped
2	Burst_irq enable	R	burst_irq enable register This bit displays if an interrupt is enabled ('1') or not ('0')
7..3	Reserved	R	Reserved

23.6.2.2 Mode register

- Address: 0x00…0x03
- Reset value: 0x0
- Access: Read/Write

This register configures the functionality of the technology component. In this case PWM function is activated by writing the value 0x00000005 to the mode register. A detailed description of mode register is under 23.5.1

Table 23-50 mode register

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x00 ... 0x03	Configuration of technology function				Mode register

23.6.2.3 Period register

- Address: 0x04…0x07
- Width: 23 Bit
- Reset value: 0x0
- Access: Read/Write

A new period value of the PWM could be written to the period register. The value is multiplied by the PWM base time value (23.9.6). The minimum value of period register is 4.

PWM_period = period register * base time value

If the period register is read the current active period value is read, not the new period value. Do activate a new period value you must set the new_pwm_data bit at the config register (23.6.2.6)

- In continuous mode the new value is used immediately (burst enable bit at configuration register (23.6.2.6) is zero or burst register is zero)
- In burst mode the new value is used after the current running burst is finished (burst enable bit at configuration register (23.6.2.6) is one and burst register is not zero)

Take care that always the complete pwm parameter set is changed. One parameter set consists of period, duration, burst and the burst_enable bit.

Table 23-51 Period register

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x04 ... 0x07	period register of PWM				period register

23.6.2.4 Burst register

- *Address: 0x08…0x0B*
- *Width: 23 Bit*
- *Reset value: 0x0*
- *Access: Read/Write*

A new burst value of the PWM could be written to the burst register. When the number of burst reached the pulse output stops.
If the burst register is read the current active burst value is read, not the new burst value. To activate a new burst value you must set the new_pwm_data bit at the config register (23.6.2.6)

- In continuous mode the new value is used immediately (burst enable bit at configuration register (23.6.2.6) is zero or burst register is zero)
- In burst mode the new value is used after the current running burst is finished (burst enable bit at configuration register (23.6.2.6) is one and burst register is not zero)

Take care that always the complete pwm parameter set is changed. One parameter set consists of period, duration, burst and the burst_enable bit.

Table 23-52 Burst register

Address	*Byte Position*				*Designation*
	Byte 0	*Byte 1*	*Byte 2*	*Byte 3*	
0x08 ... *0x0B*	*number of pulses*				*Burst register*

23.6.2.5 Duration register

- Address: 0x0C…0x0F
- Width: 23 Bit
- Reset value: 0x0
- Access: Read/Write

A new duration value of the PWM could be written to the duration register. The value is multiplied by the PWM base time value (23.9.6). The duration determines the time the PWM output is high.

PWM_period = duration register * base time value

If the duration register is read the current active duration value is read, not the new duration value. To activate a new duration value you must set the new_pwm_data bit at the config register (23.6.2.6)

- In continuous mode the new value is used immediately (burst enable bit at configuration register (23.6.2.6) is zero or burst register is zero)
- In burst mode the new value is used after the current running burst is finished (burst enable bit at configuration register (23.6.2.6) is one and burst register is not zero)

Take care that always the complete pwm parameter set is changed. One parameter set consists of period, duration, burst and the burst_enable bit.

Table 23-53 Duration register

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x0C ... 0x0F	duration register of PWM				Duration register

23.6.2.6 Configuration register

- Address: 0x12…0x13
- Reset value: 0x0
- Access: Write

This register configures the PWM.

Table 23-54 Configuration register

Bit	Name	Type	Description
0	Burst enable	W	Burst enable This bit must be written to ‘1’ if a fix number of pulse trains should be

Bit	Name	Type	Description
			configured (burst mode) This bit must be written to ‘0’ if a continuous pulse train should be configured (continuous mode)
1	Burst_irq enable	W	Interrupt at the end of a burst This bit must be set to ‘1’ if an interrupt should occur after a burst is finished.
7..2	Reserved	W	Reserved
8	Str_pwm	W	Start pulse train output A rising edge (bit goes from ‘0’ to ‘1’) starts the pulse train output
9	Stp_pwm	W	Stop pulse train output A rising edge (bit goes from ‘0’ to ‘1’) stops the pulse train output immediately
10	New_pwm_data	W	New pulse train parameter set available This bit must be set to ‘1’ if new pwm parameters (period, duration, burst) were written and should be activated. One parameter set consists of period, duration, burst and the burst_enable bit
15..11	Reserved	W	Reserved

23.6.3 Burst mode sequence

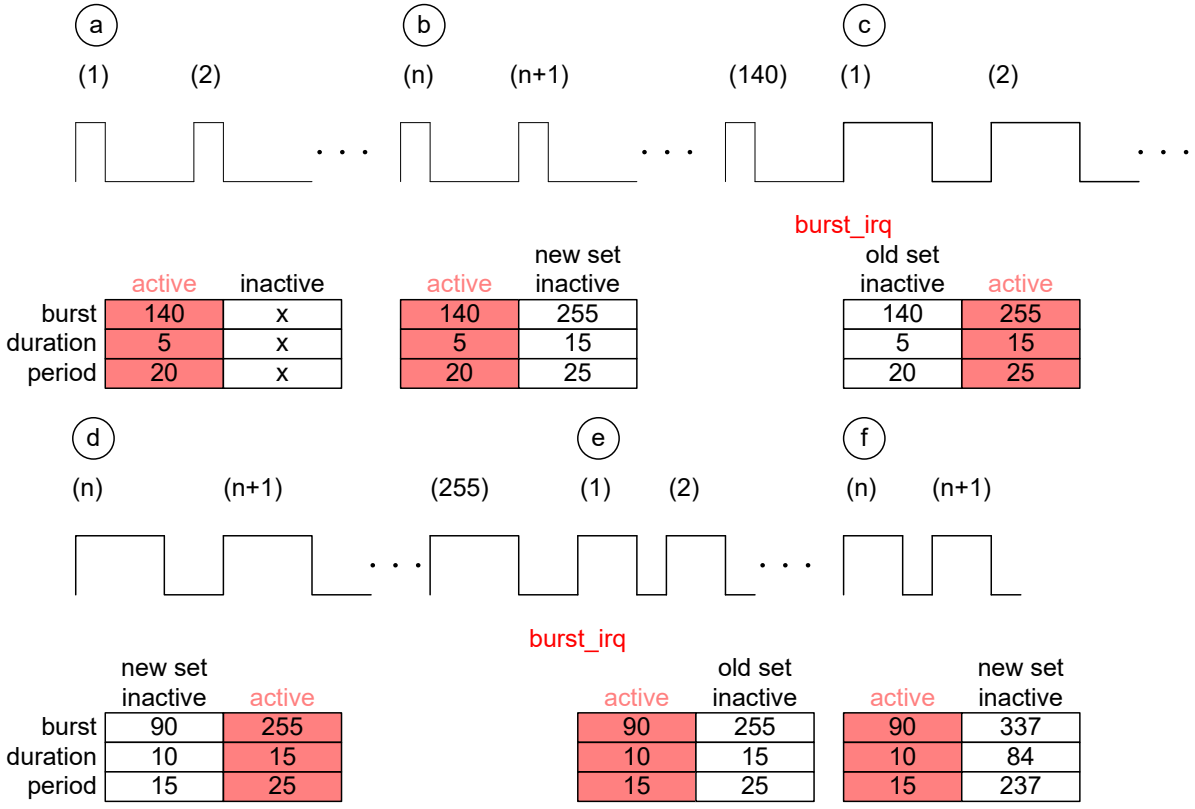
The burst mode enables the PWM module to output a continuous stream of pulse trains with different duration and/or period. Therefore a set of duration, period and the number of pulse trains must be written to the module. The next set must be written before the first set has been processed. When the number of pulse trains of the first set is reached the second set is automatically activated. An additional end of burst interrupt could be triggered after a set is completed.

Figure 23-14 *shows a burst sequence with four different sets:*

- *(a) Write the first set and start the PWM output by writing 0x0503 to the configuration register (23.6.2.6)*
- *(b) While the first set is running write a second set to the module and confirm it by writing 0x0403 to the configuration register (23.6.2.6)*
- *(c) After the number of burst is reached, the second set is automatically activated and an interrupt is triggered*
- *(d) While the second set is running write a third set to the module and confirm it by writing 0x0403 to the configuration register (23.6.2.6)*
- *(e) After the number of burst is reached, the third set is automatically activated and an interrupt is triggered*
- *(f) While the third set is running write a fourth set to the module and confirm it by writing 0x0403 to the configuration register (23.6.2.6)*

You must ensure that the next set was written before the currently running set is finished. Otherwise the output stops immediately after the number of pulse trains are reached.

Figure 23-14 burst mode sequence



23.7 Synchronous serial interface (SSI)

The ANTAIOS contains 2 independent SSI interfaces. It can be operate as master or slave, can be parametrized and can release interrupts.

23.7.1 Register mapping

23.7.1.1 Status and configuration register

Offsets: counter: 0x200

Table 23-55 Status and configuration Mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	SSI_value0				R	0x0
0x04			SSI_status0		R	0x0
0x08	SSI_value1				R	0x0
0x0C			SSI_status1		R	0x0

23.7.1.2 Parameter register

Offsets: SSI0: 0x240
SSI1: 0x260

Table 23-56 Parameter Mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	Delay_time		Send_clock		RW	0x18000300
0x04		config			RW	0x0000000D
0x08	Irq+/-	Irq_ena		enable	RW	0x02000000
0x0C	SIs compare value				RW	0x0
0x10	SIe compare value				RW	0x0
0x14	C0 compare value				RW	0x0
0x18	C1 compare value				RW	0x0
0x1C	C2 compare value				RW	0x0

23.7.2 Register description

23.7.2.1 SSI_valueX

- Address: SSI_value0 0x00…0x03
  SSI_value1 0x08…0x0B
- Reset value: 0x0
- Access: Read

The register holds the current SSI value.

Table 23-57 SSI_value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x00 … 0x03	SSI_Value				Current SSI value

23.7.2.2 SSI status register

- Address: SSI0 0x06…0x07
  SSI1 0x0E…0x0F
  Reset value: 0x0
- Access: Read

Table 23-58 Status register

Bit	Name	Type	Description
0	Sending	R	SSI Module is configured and working 0 = not sending 1 = sending clock or during TBS (time between send)
1	Listening	R	SSI Module is configured and 0 = not listening 1 = listening for SSI_clock
2	Data_valid	R	Received valid SSI telegram 0 = while receiving 1 = after received a valid telegram till the end of TBS

Bit	Name	Type	Description
3	Tbs	R	Time between two SSI telegrams 0 = while receiving data 1 = between two telegrams
7..4	Res	R	Reserved
8	SLB	R	Interrupt on slb (software limit switch begin) is just 0 = inactive 1 = active
9	SLE	R	Interrupt on sle (software limit switch end) is just 0 = inactive 1 = active
10	C0	R	Interrupt on c0 (compare value0) is just 0 = inactive 1 = active
11	C1	R	Interrupt on c1 (compare value1) is just 0 = inactive 1 = active
12	C2	R	Interrupt on c2 (compare value2) is just 0 = inactive 1 = active
15..13	Res	R	Reserved

23.7.2.3 Delay time register

- Address: 0x00…0x01
- Reset value: 0x1800
- Access: Read/Write

Table 23-59 delay time register

Address	Byte Position		Designation
	Byte 0	Byte 1	
0x00 ... 0x01	Delay_time		Delay time register delay=(delay_time register/96) us

23.7.2.4 Send clock register

- Address: 0x02…0x03
- Reset value: 0x0300
- Access: Read/Write

Table 23-60 send clock register

Address	Byte Position		Designation
	Byte 0	Byte 1	
0x02 ... 0x03	Send_clock		Send_clock frequency=(96/send_clock register) MHz

23.7.2.5 SSI configuration register

- Address: 0x04...0x07
- Reset value: 0x0000000D
- Access: Read/Write

Table 23-61 SSI configuration register

Bit	Name	Type	Description
0	Listening	RW	Listening register 0 = SSI module uses its own clock 1 = SSI module uses clock from another master
1	Master	RW	SSI Module is configured as 0 = slave 1 = master
2	Msb_first	RW	The first received bit is 0 = LSB-Bit 1 = MSB-Bit
3	Send_rising	RW	Data changes on the 0 = falling clock edge 1 = rising clock edge
4	Gray_coded	RW	Data are send 0 = binary coded 1= gray coded
5	Auto_dis_int	RW	Auto disable interrupts 0 = asserted interrupts are disabled automatically, but enabled again, if value leaves “hot area” (“hot area” means area in which interrupts asserted) 1 = asserted interrupts are disabled automatically and must be re-enabled by the user.
7..6	Res	-	Reserved
12..8	Ssi_length	RW	Length of SSI telegram – 1
15..13	Res	RW	Reserved
19..16	Data_pre_shift	RW	Number of bits, data will be shifted to LSB after telegram received. Mask out irrelevant bits
31..20	Res	-	Reserved

23.7.3 Irq+/-

- Address: 0x08
- Reset value: 0x02
- Access: Read/Write

Table 23-62 irq+/- register

Bit	Name	Type	Description
0	Irq_sls+/-	RW	Fix

Bit	Name	Type	Description
			0 = interrupt is asserted if actual value is smaller than sls value
1	Irq_sle+/-	RW	Fix 1 = interrupt is asserted if actual value is greater than sle value
2	Irq_c0+/-	RW	Interrupt is asserted if actual value is 0 = smaller than c0 value 1 = greater than c0 value
3	Irq_c1+/-	RW	Interrupt is asserted if actual value is 0 = smaller than c1 value 1 = greater than c1 value
4	Irq_c2+/-	RW	Interrupt is asserted if actual value is 0 = smaller than c2 value 1 = greater than c2 value
7..5	Res	RW	Reserved

23.7.4 Irq enable

- Address: 0x09
- Reset value: 0x0
- Access: Read/Write

Table 23-63 irq enable register

Bit	Name	Type	Description
0	Irq_sls_en	RW	Interrupt for sls value is 0 = disabled 1 = enabled
1	Irq_sle_en	RW	Interrupt for sle value is 0 = disabled 1 = enabled
2	Irq_c0_en	RW	Interrupt for c0 value is 0 = disabled 1 = enabled
3	Irq_c1_en	RW	Interrupt for c1 value is 0 = disabled 1 = enabled
4	Irq_c2_en	RW	Interrupt for c2 value is 0 = disabled 1 = enabled
7..5	Res	RW	Reserved

23.7.5 Ssi_enable

- Address: 0x0B
- Reset value: 0x0
- Access: Read/Write

Table 23-64 Status register

Bit	Name	Type	Description
0	Ssi_enable	RW	Ssi module is 0 = disabled 1 = enabled
7..1	Res	-	Reserved

23.7.5.1 Sls compare value

- Address: 0x0C…0x0F
- Reset value: 0x0
- Access: Read/Write

Table 23-65 sls compare value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x0C … 0x0F	Sls compare value				Current sls compare value

23.7.5.2 Sle compare value

- Address: 0x10…0x13
- Reset value: 0x0
- Access: Read/Write

Table 23-66 sle compare value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x10 … 0x13	Sle compare value				Current sle compare value

23.7.5.3 C0 compare value

- Address: 0x14…0x17
- Reset value: 0x0
- Access: Read/Write

Table 23-67 c0 compare value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x14 … 0x17	C0 compare value				Current c0 compare value

23.7.5.4 C1 compare value

- Address: 0x18…0x1B
- Reset value: 0x0
- Access: Read/Write

Table 23-68 c1 compare value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x18 ... 0x1B	C1 compare value				Current c1 compare value

23.7.5.5 C2 compare value

- Address: 0x1C…0x1F
- Reset value: 0x0
- Access: Read/Write

Table 23-69 c2 compare value

Address	Byte Position				Designation
	Byte 0	Byte 1	Byte 2	Byte 3	
0x1C ... 0x1F	C2 compare value				Current c2 compare value

23.8 Alarm system technology function

Offsets: 0x280

23.8.1 Parameter and status register

Table 23-70 Parameter mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x00	Reserved				-	-
0x04	Reserved				-	-
0x08	Reserved			Diag_en_tech	W	0x0
0x0C	Reserved			Alarm_typ_tech	R	0x0
0x10	Reserved			Nofify_tech	W	0x0
0x14	Reserved			Ack_tech	W	0x0
0x18	Reserved				-	-
0x1C	Reserved				-	-
0x20	Palarm_tech0				R	undef.
0x24	Palarm_tech1				R	undef.
0x28	Palarm_tech2				R	undef.
0x2C	Dalarm_tech0				R	undef.
0x30	Dalarm_tech1				R	undef.

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x34	Dalarm_tech2				R	undef.
0x38	Dalarm_tech3				R	undef.
0x3C	Reserved (do not access)				-	-

23.8.2 Register description

23.8.2.1 Diag_en_tech

- Address: 0x0B
- Reset value: 0x0
- Access: Write

Table 23-71 diag_en_tech

Bit	Name	Type	Description
0	Diag_en_dio	W	0 = technological diagnostic alarm disabled 1 = technological diagnostic alarm enabled
7..1	Res		Reserved

23.8.2.2 Alarm_typ_tech

- Address: 0x0F
- Reset value: 0x0
- Access: Read

This register is also mirrored to the digital area (23.4.2.4) to get the possibility of reading both alarm types with one read access.

Table 23-72 alarm_typ_tech

Byte	Name	Type	Description
0x0F	Alarm_typ_tech	R	Alarm type for counter/PWM and SSI 0 = no alarm pending 1 = process alarm 2 = diagnostic alarm pending No other values possible.

23.8.2.3 Notify_tech

- Address: 0x13
- Reset value: 0x0
- Access: Write

This register is mirrored from the counter area (23.4.2.6) to get the possibility of writing both alarm notifications with one write access.

Table 23-73 notify_tech

Byte	Name	Type	Description

Byte	Name	Type	Description
0x13	Notify_tech	W	After an alarm was set the alarm must be mask with a notify write. The alarm lines change back in idle mode after writing the notify register. 1 = process alarm notify 2 = diagnostic alarm notify No other values allowed

23.8.2.4 Ack_tech

- Address: 0x17
- Reset value: 0x0
- Access: Write

This register is mirrored from the counter area (23.4.2.8) to get the possibility of writing both alarm acknowledges with one write access.

Table 23-74 ack_tech

Byte	Name	Type	Description
0x17	Ack_tech	W	After the alarm data were read the alarm must be acknowledge getting a new alarm from this module and this priority level. 1 = process alarm acknowledge 2 = diagnostic alarm acknowledge No other values allowed

23.8.2.5 Palarm_tech0

- Address: 0x20..0x23
- Reset value: undef.
- Access: Read

Table 23-75 Palarm_tech0

Bit	Name	Type	Description
0	HW_gate0 rise	R	0 = no alarm 1 = the hw-gate0 was opened
1	HW_gate0 fall	R	0 = no alarm 1 = the hw-gate0 was closed
2	Eou0	R	0 = no alarm 1 = overflow, underflow occurred or end value was reached
3	Compare value0	R	0 = no alarm 1 = the compare value was reached
4	Latch value0 new	R	0 = no alarm 1 = new letch value available
5	HW_gate1 rise	R	0 = no alarm 1 = the hw-gate0 was opened
6	HW_gate1 fall	R	0 = no alarm 1 = the hw-gate0 was closed

Bit	Name	Type	Description
7	Eou1	R	0 = no alarm 1 = overflow, underflow occurred or end value was reached
8	Compare value1	R	0 = no alarm 1 = the compare value was reached
9	Latch value1 new	R	0 = no alarm 1 = new letch value available
10	HW_gate2 rise	R	0 = no alarm 1 = the hw-gate0 was opened
11	HW_gate2 fall	R	0 = no alarm 1 = the hw-gate0 was closed
12	Eou2	R	0 = no alarm 1 = overflow, underflow occurred or end value was reached
13	Compare value2	R	0 = no alarm 1 = the compare value was reached
14	Latch value2 new	R	0 = no alarm 1 = new letch value available
15	HW_gate3 rise	R	0 = no alarm 1 = the hw-gate0 was opened
16	HW_gate3 fall	R	0 = no alarm 1 = the hw-gate0 was closed
17	Eou3	R	0 = no alarm 1 = overflow, underflow occurred or end value was reached
18	Compare value3	R	0 = no alarm 1 = the compare value was reached
19	Latch value3 new	R	0 = no alarm 1 = new letch value available
20	SSI0_sls	R	0 = no alarm 1 = SSI0 compare value begin reached
21	SSI0_sle	R	0 = no alarm 1 = SSI0 compare value end reached
22	SSI0_c0	R	0 = no alarm 1 = SSI0 compare value 0 reached
23	SSI0_c1	R	0 = no alarm 1 = SSI0 compare value 1 reached
24	SSI0_c2	R	0 = no alarm 1 = SSI0 compare value 2 reached
25	SSI1_sls	R	0 = no alarm 1 = SSI1 compare value begin reached
26	SSI1_sls	R	0 = no alarm 1 = SSI1 compare value end reached
27	SSI1_c0	R	0 = no alarm 1 = SSI1 compare value 0 reached
28	SSI1_c1	R	0 = no alarm 1 = SSI1 compare value 1 reached

Bit	Name	Type	Description
29	SSI1_c2	R	0 = no alarm 1 = SSI1 compare value 2 reached
31..30	Res	R	Always 0

23.8.2.6 Palarm_tech1

- Address: 0x24..0x27
- Reset value: undef.
- Access: Read

Table 23-76 Palarm_tech1

Bit	Name	Type	Description
0	Input0	R	0 = input0 was low after the alarm occurred 1 = input0 was high after the alarm occurred
1	Input1	R	0 = input1 was low after the alarm occurred 1 = input1 was high after the alarm occurred
..	..	..	..
25	Input25	R	0 = input25 was low after the alarm occurred 1 = input25 was high after the alarm occurred
31..26	Res	R	0

23.8.2.7 Palarm_tech2

- Address: 0x28..0x2B
- Reset value: undef.
- Access: Read

This register is always read as 0x80000000.

23.8.2.8 Dalarm_tech0

- Address: 0x2C..0x2F
- Reset value: undef.
- Access: Read

Table 23-77 Dalarm_tech0

value	Type	Description
0x00001800	R	Only the last going diagnostic alarm
0x4000180B	R	Every diagnostic alarm; except the last going diagnostic alarm

23.8.2.9 Dalarm_tech1

- Address: 0x30..0x33
- Reset value: undef.
- Access: Read

Table 23-78 Dalarm_tech1

offset	Byte 0		Byte 1	Byte 2	Byte 3	Designation
0x30 ... 0x33	Bit	Description	0x06	0x08	0x76	Dalarm_tech1
	0	Diag alarm on counter0 '0' = not occurred '1' = occurred				
	1	Diag alarm on counter1 '0' = not occurred '1' = occurred				
	2	Diag alarm on counter2 '0' = not occurred '1' = occurred				
	3	Diag alarm on counter3 '0' = not occurred '1' = occurred				
	4	Diag alarm on SSI0 '0' = not occurred '1' = occurred				
	5	Diag alarm on SSI1 '0' = not occurred '1' = occurred				
	7..6	0				

23.8.2.10 Dalarm_tech2

- Address: 0x34..0x37
- Reset value: undef.
- Access: Read

Table 23-79 Dalarm_tech2

Bit	Name	Type	Description
0	Diagnostic alarm for hw_gate0 rise	R	0 = no alarm 1 = occurred
1	Diagnostic alarm for hw_gate0 fall	R	0 = no alarm 1 = occurred
2	Diagnostic alarm for eou0	R	0 = no alarm 1 = occurred
3	Diagnostic alarm for compare0	R	0 = no alarm 1 = occurred

Bit	Name	Type	Description
4	Diagnostic alarm for new latch0	R	0 = no alarm 1 = occurred
7..5	Res	R	Always 0
8	Diagnostic alarm for hw_gate1 rise	R	0 = no alarm 1 = occurred
9	Diagnostic alarm for hw_gate1 fall	R	0 = no alarm 1 = occurred
10	Diagnostic alarm for eou1	R	0 = no alarm 1 = occurred
11	Diagnostic alarm for compare1	R	0 = no alarm 1 = occurred
12	Diagnostic alarm for new latch1	R	0 = no alarm 1 = occurred
15..13	Res	R	Always 0
16	Diagnostic alarm for hw_gate2 rise	R	0 = no alarm 1 = occurred
17	Diagnostic alarm for hw_gate2 fall	R	0 = no alarm 1 = occurred
18	Diagnostic alarm for eou2	R	0 = no alarm 1 = occurred
19	Diagnostic alarm for compare2	R	0 = no alarm 1 = occurred
20	Diagnostic alarm for new latch2	R	0 = no alarm 1 = occurred
23..21	Res	R	Always 0
24	Diagnostic alarm for hw_gate3 rise	R	0 = no alarm 1 = occurred
25	Diagnostic alarm for hw_gate3 fall	R	0 = no alarm 1 = occurred
26	Diagnostic alarm for eou3	R	0 = no alarm 1 = occurred
27	Diagnostic alarm for compare3	R	0 = no alarm 1 = occurred
28	Diagnostic alarm for new latch3	R	0 = no alarm 1 = occurred

Bit	Name	Type	Description
31..29	Res	R	Always 0

23.8.2.11 Dalarm_tech3

- Address: 0x38..0x3B
- Reset value: undef.
- Access: Read

Table 23-80 Dalarm_tech3

Bit	Name	Type	Description
0	Diagnostic alarm for SSI0_sls	R	0 = no alarm 1 = occurred
1	Diagnostic alarm for SSI0_sle	R	0 = no alarm 1 = occurred
2	Diagnostic alarm for SSI0_c0	R	0 = no alarm 1 = occurred
3	Diagnostic alarm for SSI0_c1	R	0 = no alarm 1 = occurred
4	Diagnostic alarm for SSI0_c2	R	0 = no alarm 1 = occurred
7..5	Res	R	Always read as zero
8	Diagnostic alarm for SSI1_sls	R	0 = no alarm 1 = occurred
9	Diagnostic alarm for SSI1_sle	R	0 = no alarm 1 = occurred
10	Diagnostic alarm for SSI1_c0	R	0 = no alarm 1 = occurred
11	Diagnostic alarm for SSI1_c1	R	0 = no alarm 1 = occurred
12	Diagnostic alarm for SSI1_c2	R	0 = no alarm 1 = occurred
31..13	Res	R	Always read as zero

23.9 Miscellaneous

23.9.1 Invert digital input

- Address: 0x0C3
- Reset value: 0x0
- Access: Write

This register inverts the input value byte wise.

Table 23-81 invert digital input register

Bit	Name	Type	Description
0	Byte0_inv	W	Invert the digital input 25..24
1	Byte1_inv	W	Invert the digital input 23..16
2	Byte2_inv	W	Invert the digital input 15..8
3	Byte3_inv	W	Invert the digital input 7..0
7..4	Res	-	Reserved

23.9.2 Delay output byte2

- *Address: 0x0CA..0x0CB*
- *Reset value: 0x0*
- *Access: Write*

With this register the output on or off switching can be delayed. The range of the delay lies between 10.41ns to 2,656 ns.

Table 23-82 delay output byte2 register

Bit	*Name*	*Type*	*Description*
7..0	*Delay value*	*W*	*The base time of 1/96 ns is multiplied with the delay value and defines the output switching delays of outputs 23..16*
8	*Active edge*	*W*	*The bit defines the switching edge on which the delay is active* *0 = falling edge is delayed* *1 = rising edge is delayed*
15..9	*Res*	*-*	*Reserved*

23.9.3 Delay output byte1

- *Address: 0x0CE..0x0CF*
- *Reset value: 0x0*
- *Access: Write*

With this register the output on or off switching can be delayed. The range of the delay lies between 10.41ns to 2,656 ns.

Table 23-83 delay output byte1 register

Bit	Name	Type	Description
7..0	*Delay value*	*W*	*The base time of 1/96 ns is multiplied with the delay value and defines the output switching delays of outputs 15..8*
8	*Active edge*	*W*	*The bit defines the switching edge on which the delay is active* *0 = falling edge is delayed* *1 = rising edge is delayed*
15..9	*Res*	*-*	*Reserved*

23.9.4 Delay output byte0

- *Address: 0x0D2..0x0D3*
- *Reset value: 0x0*
- *Access: Write*

With this register the output on or off switching can be delayed. The range of the delay lies between 10.41ns to 2,656 ns.

Table 23-84 delay output byte0 register

Bit	Name	Type	Description
7..0	*Delay value*	*W*	*The base time of 1/96 ns is multiplied with the delay value and defines the output switching delays of outputs 7..0*
8	*Active edge*	*W*	*The bit defines the switching edge on which the delay is active* *0 = falling edge is delayed* *1 = rising edge is delayed*
15..9	*Res*	*-*	*Reserved*

23.9.5 Basp register

- Address: 0x0FB
- Reset value: 0x0
- Access: Read/Write

The basp register set the whole module in an active state. It is possible to do all parametrization but to set any output value or count something the basp register must be set to one.

Table 23-85 basp register

Bit	Name	Type	Description
0	Basp	RW	Set the module in an active state 0 = whole module inactive 1 = whole module active
7..1	Res	RW	read as zero

23.9.6 PWM base time value

- Address: 0x1FB
- Reset value: 0x20
- Access: Read/Write

This register configures the PWM.

Table 23-86 PWM base time register

Bit	Name	Type	Description
3..0	PWM base time	RW	Define the base time for the PWM 0000 = 1/96 us 0001 = 2/96 us … 1110 = 15/96 us 1111 = 16/96 us
7..4	Res	RW	Reserved

23.10 Entire techIO mapping

Table 23-87 Entire techIO mapping

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x000	Digital input				R	0x0
		Digital output			W	0x0
0x004 .. 0x03F	Reserved				R	0x0
0x040		Filter input1		Filter input0	RW	0x000D000D
0x044		Filter input3		Filter input2	RW	0x000D000D
0x048		Filter input5		Filter input4	RW	0x000D000D
0x04C		Filter input7		Filter input6	RW	0x000D000D
0x050		Filter input9		Filter input8	RW	0x000D000D
0x054		Filter input11		Filter input10	RW	0x000D000D
0x058		Filter input13		Filter input12	RW	0x000D000D
0x05C		Filter input15		Filter input14	RW	0x000D000D
0x060		Filter input17		Filter input16	RW	0x000D000D
0x064		Filter input19		Filter input18	RW	0x000D000D
0x068		Filter input21		Filter input20	RW	0x000D000D
0x06C		Filter input23		Filter input22	RW	0x000D000D
0x070		Filter input25		Filter input24	RW	0x000D000D
0x074 .. 0x07F	Reserved				RW	0x000D000D
0x080	Alarm_rising_edge				RW	0x0
0x084	Alarm_falling_edge				RW	0x0

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x088	Reserved			Diag_en_dio	W	0x0
0x08C	Reserved		Alarm_typ_tech	Alarm_typ_dio	R	0x0
0x090	Reserved		Notify_tech	Nofify_dio	W	0x0
0x094	Reserved		Ack_tech	Ack_dio	W	0x0
0x098	Reserved				R	0x0
0x09C	Reserved				R	0x0
0x0A0	Palarm_dio0				R	undef.
0x0A4	Palarm_dio1				R	undef.
0x0A8	Palarm_dio2				R	undef.
0x0AC	Dalarm_dio0				R	undef.
0x0B0	Dalarm_dio1				R	undef.
0x0B4	Dalarm_dio2				R	undef.
0x0B8	Dalarm_dio3				R	undef.
0x0BC	Reserved (do not access)				-	-
0x0C0				Invert digital input	W	0x0
0x0C4	Reserved				RW	0x0
0x0C8	Reserved		Delay_output_byte2		W	0x0
0x0CC	Reserved		Delay_output_byte1		W	0x0
0x0D0	Reserved		Delay_output_byte0		W	0x0
0x0D4 … 0x0F7	Reserved				R	0x0
0x0F8	Reserved			Basp	RW	0x0
0x0FC	Digital ID				R	0x00005a5a
0x100	Count/latch value0				R	0x0
	Config0				W	
0x104	Count status0/PWM status0				R	0x21000000
	Config1				W	
0x108	Count/latch value1				R	0x0
	Config2				W	
0x10C	Count status1/PWM status1				R	0x21000000
	Config3				W	
0x110	Count/latch value2				R	0x0
0x114	Count status2/PWM status2				R	0x21000000
0x118	Count/latch value3				R	0x0
0x11C	Count status3/PWM status3				R	0x21000000
0x120	Mode0				RW	0x0
0x124	Compare value0/Current period0				R	0x0
	Compare value0/New period0				W	0x0
0x128	Load value0/Current burst0				R	0x0
	Load value0/New burst0				W	0x0
0x12C	End value0/Current duration0				R	0x0
	End value0/New duration0				W	0x0

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x130	Count value0/Configuration0				RW	0x0
0x134	Reserved			Hysteresis value0	RW	0x0
0x138 .. 0x13F	Reserved				R	0x0
0x140	Mode1				RW	0x0
0x144	Compare value1/Current period1				R	0x0
	Compare value1/New period1				W	
0x148	Load value1/Current burst1				R	0x0
	Load value1/New burst1				W	
0x14C	End value1/Current duration1				R	0x0
	End value1/New duration1				W	
0x150	Count value1/Configuration1				RW	0x0
0x154	Reserved			Hysteresis value1	RW	0x0
0x158 .. 0x15F	Reserved				R	0x0
0x160	Mode2				RW	0x0
0x164	Compare value2/Current period2				R	0x0
	Compare value2/New period2				W	
0x168	Load value2/Current burst2				R	0x0
	Load value2/New burst2				W	
0x16C	End value2/Current duration2				R	0x0
	End value2/New duration2				W	
0x170	Count value2/Configuration2				RW	0x0
0x174	Reserved			Hysteresis value2	RW	0x0
0x178 .. 0x17F	Reserved				R	0x0
0x180	Mode3				RW	0x0
0x184	Compare value3/Current period3				R	0x0
	Compare value3/New period3				W	
0x188	Load value3/Current burst3				R	0x0
	Load value3/New burst3				W	
0x18C	End value3/Current duration3				R	0x0
	End value3/New duration3				W	
0x190	Count value3/Configuration3				RW	0x0
0x194	Reserved			Hysteresis value3	RW	0x0
0x198 .. 0x1F7	Reserved				R	0x0
0x1F8	Reserved			PWM_base_ time	RW	0x00000020

Offset	Byte 0	Byte 1	Byte 2	Byte 3	Access	Reset Value
0x1FC	Counter ID				R	0x0000a5a5
0x200	SSI_value0				R	0x0
0x204			SSI_status0		R	0x0
0x208	SSI_value1				R	0x0
0x20C			SSI_status1		R	0x0
0x210 .. 0x23F	Reserved				R	0x0
0x240	Delay_time0		Send_clock0		RW	0x14000280
0x244		Config0			RW	0x0000180E
0x248	Irq+/-0	Irq_ena0		Enable0	RW	0x02000000
0x24C	SIs compare value0				RW	0x0
0x250	SIe compare value0				RW	0x0
0x254	C0 compare value0				RW	0x0
0x258	C1 compare value0				RW	0x0
0x25C	C2 compare value0				RW	0x0
0x260	Delay_time1		Send_clock1		RW	0x14000280
0x264		Config1			RW	0x0000180E
0x268	Irq+/-1	Irq_ena1		Enable1	RW	0x02000000
0x26C	SIs compare value1				RW	0x0
0x270	SIe compare value1				RW	0x0
0x274	C0 compare value1				RW	0x0
0x278	C1 compare value1				RW	0x0
0x27C	C2 compare value1				RW	0x0
0x280 .. 0x287	Reserved				R	0x0
0x288	Reserved			Diag_en_tech	W	0x0
0x28C	Reserved			Alarm_typ_tech	R	0x0
0x290	Reserved			Nofify_tech	W	0x0
0x294	Reserved			Ack_tech	W	0x0
0x298	Reserved				R	0x0
0x29C	Reserved				R	0x0
0x2A0	Palarm_tech0				R	undef.
0x2A4	Palarm_tech1				R	undef.
0x2A8	Palarm_tech2				R	undef.
0x2AC	Dalarm_tech0				R	undef.
0x2B0	Dalarm_tech1				R	undef.
0x2B4	Dalarm_tech2				R	undef.
0x2B8	Dalarm_tech3				R	undef.
0x2BC	Reserved (do not access)				-	-
0x2C0 .. 0x2FB	Reserved				R	0x0
0x2FC	SSI ID				R	0x0000a55a

24 Revision History

Table 24-1 Revision history

Version	Date	Remarks
V0.10	18.05.2016	First draft Added TechIO capture
V0.11	08.06.2016	First complete TechIO description
V0.12	15.06.2016	Added USB 2.0 Device Controller
V0.13	12.08.2016	Added additional chapter
V0.14	14.10.2016	Added chapter SSP-Controller
V0.15	18.10.2016	Reworked external SRAM-IF, Timer and Watchdog, AIRQ-Controller, Pin-Controller, AHB/APB-Memory and SPP/SPI-Controller Add parts of chapter AHB/APB-Bridge
V0.16	19.10.2016	Rename SPI, I²C and UART Rework DMA-Controller Delete feature and revision register of ip-cores
V0.17	26.10.2016	Preliminary review result
V0.18	27.10.2016	Review with comment (beta version)
V0.19	18.11.2016	New CI
V0.20	24.11.2016	Correct SD-Card and some USB paragraph
V0.21	29.11.2016	Minor corrections on format and typos
V0.22	09.12.2016	Added some detail description in chapter DDR2 Memory Controller, NAND Flash Controller and USB Device Controller
V0.23	09.12.2016	Sub chapter “watchdog” moved from “General Purpose I/O” to “Timer and Watchdog”
V0.24	05.01.2017	Changed some reset values from TechIO (SSI) module
V0.25	12.01.2017	Added rising/falling edge support for external interrupts to AIRQC
V0.26	19.01.2017	Fix copy/paste error in TechIO, PWM chapter
V0.27	25.01.2017	Enabled ODT in DDR2 configuration
V0.28	02.02.2017	Delete synchronous mode form NAND-Flash Controller, because it is not supported
V1.00	22.02.2017	First official release
V1.01	17.03.2017	Fix some inaccuracy in NAND Flash Controller and SPI chapter
V1.02	23.03.2017	Deleted I²C IO-Connection chapter and remove description for SMC-Bus
V1.03	20.04.2017	Small changes in chapter I²C, SD/MMC according to some reviewer notes
V1.04	21.04.2017	Added chapter CAN
V1.05	02.05.2017	Fix some bugs in SD-Card-Controller chapter “Initialization/Application Information”
V1.06	09.05.2017	Corrected typos in the AIRQC chapter
V1.07	16.05.2017	Corrected some alarm address in chapter TechIO
V1.08	19.05.2017	Changed document title
V1.09	02.06.2017	Corrected typo in boot options
V1.10	07.06.2017	Changed format of AHB/APB memory mapping
V1.11	08.06.2017	Corrected timer enumeration in interrupt controller description and DDR interface options
V1.13	20.03.2018	Added hints for UART configuration
V1.14	26.03.2018	Change name of Ext. interrupt sources.
V1.15	12.04.2018	Added note to Main Control Register of USB controller
V1.16	10.08.2018	Added note to Advanced IRQ Controller
V1.17	01.10.2019	New document design
V1.18	12.05.2020	Change name from table 23-82, 23-83 and 23-84 to delay output byteX register
V1.19	30.11.2020	review chapter 16 DMA Controller and chapter 18 SPI Controller
V1.20	05.02.2021	minor changes chapter 16 DMA Controller
V1.21	08.02.2021	fix Figure 16-13
V1.22	12.04.2021	minor change in chapter 18 SPI controller

V1.23	13.09.2022	CAN: Modification on Arbitration Register and Global Mask Register

YASKAWA Europe GmbH
Ohmstr. 4, 91074 Herzogenaurach
Germany
Phone: +49 (0)9132 744-200
E-Mail: support.profichip@yaskawa.eu.com
www.profichip.com

YASKAWA