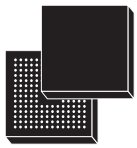
SPEAR-07-NC03

Ethernet Communication Controller with USB-Host

Features

- Based on ARM720T (8K Caches and MMU included)
- Support a 10/100 Mbits/s Ethernet connection (IEEE802.3)
- Full-Speed USB Host Controller, supports 12Mbit/s Full Speed Devices
- UART Interface: 115KBaud
- I^2C interface: Fast and Slow.
- IEEE1284 Host Controller
- Real Time Clock
- Timers and Watchdog peripherals
- Integrated PLL (25MHz Input, 48MHz Output)
- Up to 12 GPIOs (including IEEE1284 port)
- 8K SRAM shared with an External Microprocessor
- Static Memory Controller (up to 2 Banks, Max 16M each)



LFBGA180 (12x12x1.7mm)

- DRAM Controller SDRAM/EDO (up to 4 Banks, Max 32M each)
- External I/O Banks: 2 x 16KB.
- Package LFBGA 180 (12x12mm x1.7mm)

Description

SPEAR-07-NC03 is a smart Communication Controller for USB and Ethernet Communication.

SPEAR-07-NC03 allows the sharing of a Full-Speed USB or IEEE1284 or a UART Peripherals inside an Ethernet System.

SPEAR-07-NC03 is supported by several Operation Systems such as eCOS.

Order codes

Part number	Op. Temp. range, °C	Package	Packing
SPEAR-07-NC03	-40 to +105	LFBGA180	Tray

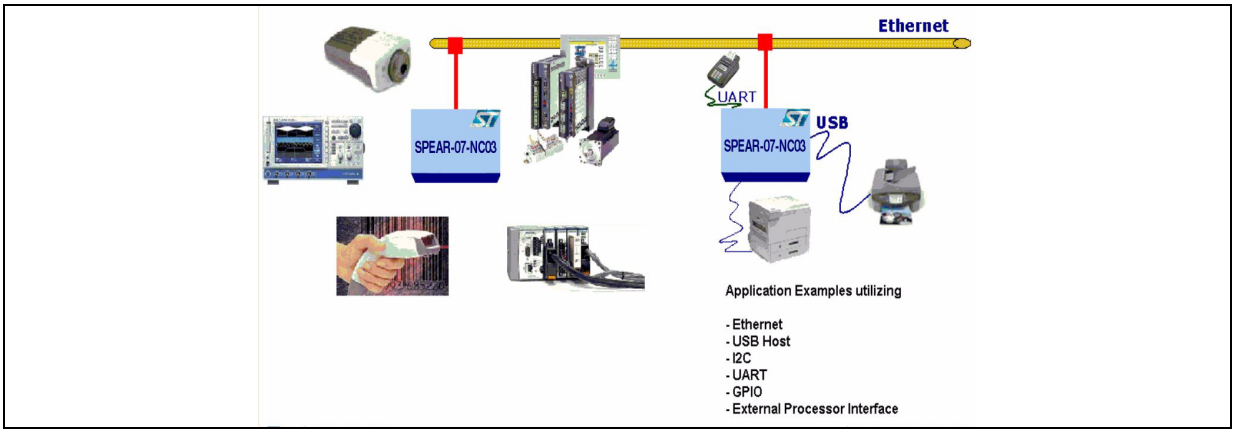


Table of Contents

List of tables

List of figures

1 Product overview

1.1 Overview

The SPEAR-07-NC03 is based on ARM720T RISC core, cache and MMU. It provide a bridge between four different I/F :

1. IEEE802.3/Ethernet MAC core for network interface. Its base interface with PHY (physical layer) chip is capable of 10/100 Mbps MII (Medium Independent Interface) and 7-wire interface.
2. USB host controller with both interrupt-based and DMA-based data handling method.
3. IEEE1284 host controller offering Compatibility mode, Nibble mode and ECP mode.
4. Shared RAM (Mail box method) for communication with other processors.
5. I^2C master controller.

2 Features

2.1 Architecture

- Integrated System For Ethernet Application
- 48MHz, 3.3V I/O and 1.8V Internal Core Voltage
- ARM720T RISC Processor Core
- AMBA Rev. 2.0 System Bus Architecture
- IEEE802.3/Ethernet Compliant MAC Core
- USB Interface Solution (Host)
- IEEE1284 Interface Solution (Master)

2.2 ARM720T RISC Processor

- ARM7TDMI RISC Core
- 8KB Unified Instruction/Data Cache
- Enlarged Write Buffer (8 words and 4 different addresses)
- Virtual Address Support with MMU

2.3 External Memory Interface

- 16bit/8bit Memory Bus Support
- ROM/SRAM/Flash Static Memory Controller
- SDRAM/EDO DRAM Controller
- External I/O Bank Controller
- Independent Configurable Memory and I/O Banks
- Replaceable Memory and I/O Bank Addresses

2.4 IEEE802.3/Ethernet MAC

- 10/100 MAC, MAC Host block, Station Management block, Address
- Compliant with IEEE 802.3 and 802.3u specifications
- Supports 10/100 Mb/s data transfer rates
- IEEE 802.3 Media Independent Interface (MII)
- Supports full and half duplex operations
- Check block and the Control Status Register (CSR) block

2.5 DMA Controller

- Dedicated Channels for MAC core, USB Host and IEEE1284 interface
- 2 Channels general purpose DMA (Memory To Memory)

- 2 Channels general purpose DMA dedicated to the external requests (I/O to Memory and Memory to I/O)
- Increments/Decrements of Source/Destination Address In 16/8bit(external), 32/16/8(internal) Data Transfers
- Burst Transfer Mode

2.6 UART

- Support For 8-Bit Serial Data Tx And Rx
- Selectable 2/1 Stop Bits
- Selectable Even, Odd and No Parity
- Parity, Overrun And Framing Error Detection
- Max Transfer rate:115KBaude

2.7 Timers

- Channel Programmable 16-Bit Timers with 8 bit pre-scaler

2.8 Watchdog Timer

- For Recovery from Unexpected System Hang-up
- One Programmable 16-Bit Watchdog Timer With Reset Output Signal (more than 200 system clock period to initial peripheral devices)
- Programmable period 1 ~ 10 sec

2.9 GPIO (Programmable I/O)

- 4 Dedicated Programmable I/O Ports (Pins)
- 2 Multiplexed Pins with I2C Bus Signals
- Pins Individually Configurable To Input, Output Or I/O Mode

2.10 Interrupt Controller

- 2 External Interrupt Sources Support
- 9 Internal interrupt sources.
- All channels can be individually rerouted to the Fast (nFIQ) or to the Normal (nIRQ) processor lines
- Level and edge (rise, fall and both) selectable
- Software Controlled Priority

2.11 IEEE1284 Host Controller

- Compatibility/Nibble/ECP/EPP Mode Host Support

- DMA-based Data Transfer Capability for ECP
- Fully Software Controllable Operation Mode

2.12 USB Host Controller

- Full-Speed USB compliant
- Supports Low Speed and Full Speed Devices
- Configuration data stored in Port Configurable Block
- Single 48 MHz input clock
- Integrated Digital PLL

2.13 Shared SRAM

- External Processor Communication Purpose
- Shared SRAM Bus Arbiter
- Same address can be accessed at the same time
- Separated from AHB Bus for Bus Traffic Reduce
- Interrupt Output Generation for Transfer Notification

2.14 Real Time Clock

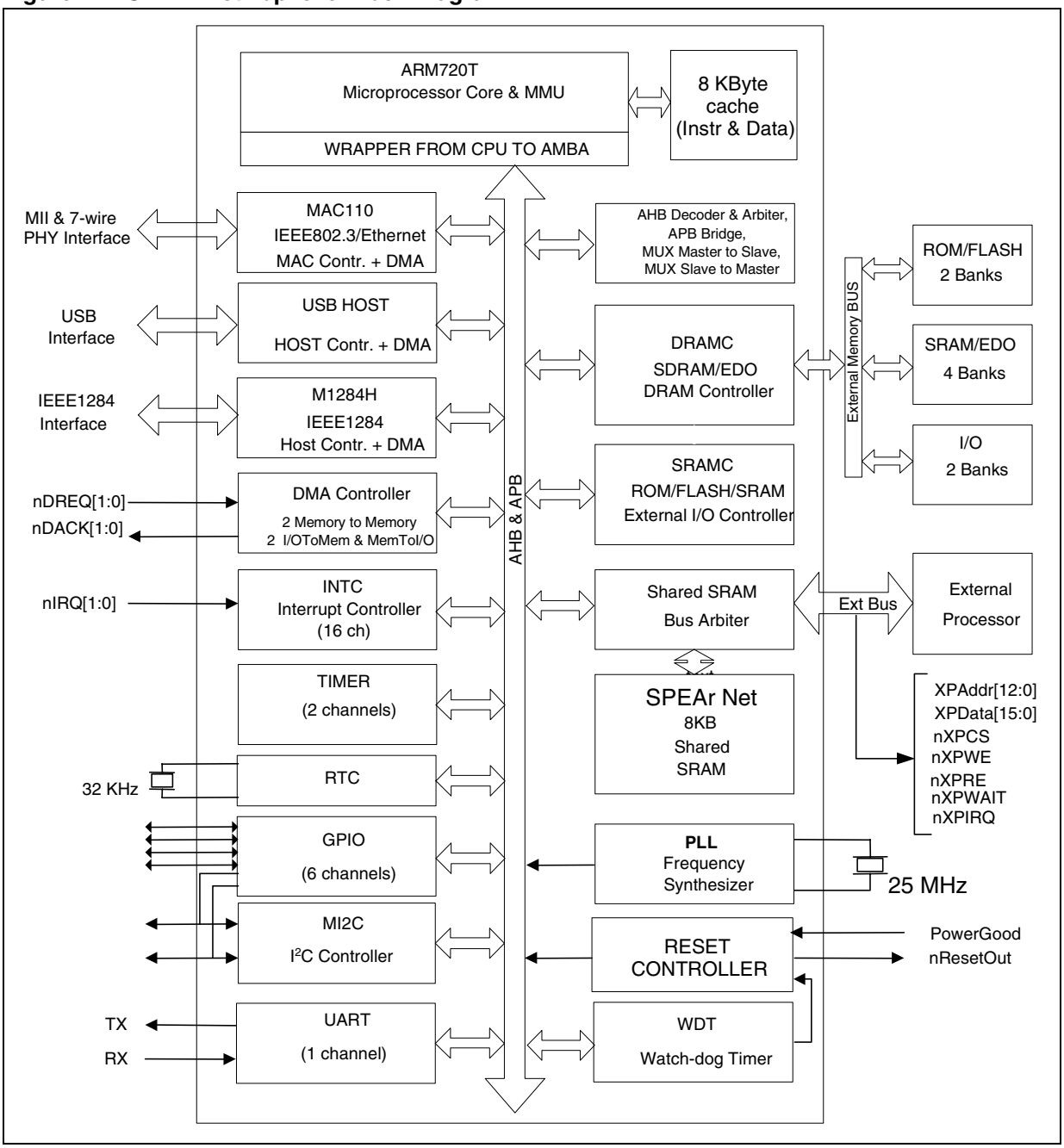
- Real time clock-calendar (RTC)
- Clocked by 32.768MHz low power clock input
- Separated power supply (1.8 V)
- 14 digits (YYYY MM DD hh mm ss) precision

2.15 Frequency Synthesizer

- On-chip Frequency Synthesizer Provided
  - Fin: 25 MHz.
  - Fout: 48 MHz

3 Top-level Block Diagram

Figure 1. SPEAr Net Top level Bock Diagram



4 Pin Descriptions

4.1 Functional Pin Groups

Note: Note: symbol / means multiplexing modes

Table 1. Pin Descriptions by Functional Groups

Group	Pin Name	Function	Remark
Clocks, Reset and configuration.	MCLKI	Oscillator Clock Input	25 MHz
	MCLKO	Oscillator Clock Output	25 MHz
	PowerGood	System Reset Input	
	nResetOut	Peripheral Chips Driving Reset Output	Synchronous with System Clock
	TMODE0	Test Mode Selector	"1" → Test
	PCLK	Output Clock (25 MHz)	
Memory Interface	Add[22:11], Add[10]/AP, Add[9:0]	Address Bus Add[10]/AP is the auto pre-charge control pin. The auto pre-charge command is issued at the same time as burst read or burst write by asserting high on Add[10]/ AP	ROM Banks: 16MB Max. DRAM Banks: 32MB Max.
	Data[15:0]	External, Bi-directional, 16 bit Data Bus	
	nRAS[3:0]/ nSDCS[3:0]	Row Address Strobe for EDO DRAM/ nSDCS is chip select pin for SDRAM	
	nSDRAS	Row Address Strobe for SDRAM	
	nSDCAS	Column Address Strobe for SDRAM	
	SDCLK	System Clock for SDRAM	48 MHz
	CKE	Clock enable signal for SDRAM	
	nWE	Write Enable for external devices.	
	nECS[1:0]	Chip Select for external I/O Bank	Banks size: 16KB
	nRCS[1:0]	Not ROM/SRAM/FLASH Chip Select	
	nOE	Not Output Enable for memory I/O	
	nCAS[1:0]/ DQM[1:0]	nCAS is the Column Address Strobe for EDO DQM is data mask signal for SDRAM.	

Table 1. Pin Descriptions by Functional Groups (continued)

Group	Pin Name	Function	Remark
Ethernet Controller	MDC	Management Data Clock	
	MDIO	Management Data I/O	
	COL/ COL_10M	Collision detected/collision detected for 10Mbps	
	TX_CLK/ TXCLK_10M	Transmit clock/ transmit clock for 10Mbps	
	TXD[3:0]/ TXD_10M/ LOOP_10M	Transmit data/ transmit data for 10Mbps	
	TX_EN/ TXEN_10M	Transmit enable/transmit enable for 10Mbps	
	CRS/ CRS_10M	Carrier sense/carrier sense for 10Mbps	
	RX_CLK/ RXCLK_10M	Receive clock/receive clock for 10Mbps	
	RXD[3:0]/ RXD_10M	Receive data/receive data for 10Mbps	
	RX_DV/ LINK_10M	Receive data valid	
	RX_ERR	Receive error	
Group	Pin Name	Function	Remark
UART	RXData	UART receive data	
	TXData	UART transmit data	
JTAG Interface	TCK	Test Clock Input	
	TDI	Test Data Input	
	TDO	Test Data Output	
	TMS	Test Mode select Input	
	TRST	Test Reset Line	
GPI/O	GPI/O[5:0]	General Purpose I/O #5 and #4/ I^2C Bus Controller SDA, SCL	
I2C Interface	SCL	Serial Clock line	
	SDA	Serial Data line	

Table 1. Pin Descriptions by Functional Groups (continued)

Group	Pin Name	Function	Remark
External Processor Interface	XPData[15:0]	External Processor Data Bus High Byte	
	XPAddr[12:0]	External Processor Address Bus	
	nXPCS	Shared SRAM CS from External Processor	
	nXPWE	Shared SRAM Write Strobe from Ext. Proc.	
	nXPRE	Shared SRAM Read Strobe from Ext. Proc.	
	nXPWAIT	Wait Signal to External Processor	
	nXPIRQ	Interrupt Request to External Processor	
USB	UHD+	USB Host differential Data signal high side	
	UHD-	USB Host differential Data signal low side	
IEEE1284	PpData[7:0]	Data Lines to/from peripheral	
	nSTROBE	Strobe Line to peripheral	
	nACK	Ack line from peripheral	
	Busy	Busy line from peripheral	
	PError	Paper Error line from peripheral	
	Select	Selection feedback from peripheral	
	nAutoFD	Auto Feed signal to peripheral	
	nFault	Generic error line from peripheral	
	nInit	Init (Reset) line to peripheral	
	nSelectIn	Select signal to peripheral	
	PpDataDir	Direction control for external transceiver	
RTC	RTCXI	RTC Oscillator Input line	32.768 KHz
	RTCXO	RTC Oscillator Output line	32.768 KHz
MISC.	nXIRQ[1:0]	External Interrupt request lines	
	nXDRQ[1:0]	External request lines for DMA	
	nXDACK[1:0]	External Acknowledge lines from DMA	
Configuration	Sdram/EDO	DRAM type selection	Pull-Up → SDRAMM
	USBEnable	USB Transceiver Enable or Disable	Pull-up → Enabled
	IEEE1284/ XProcessor	IEEE1284 or External Processor selection	Pull-up → IEEE1284
	BootRomBusWidth	Bus Width Selection for Rom 0	Pull-up → 16 bits
	UART/JTAG	Debug interface selection	Pull-up → UART
	GenConf[2:0]	General purpose Configuration lines	
Power	VDD_Core	Internal Logic Core VDD (1.8V 5%)	
	VDD_I/O	I/O Pad VDD (3.3V 5%)	
	VDD_RTC	RTC VDD (1.8V 5%)	
	GND	Ground	

4.2 PAD Types

Table 2. Pin Description by PAD Types (*LH)

Ref.	Ball	Name	Type	Dir.
1	B1	MCLKI	ANA	-
2	C1	MCLKO	OSCI27B	I
3	F2	PowerGood	SCHMITT_TC	I
4	E4	nRESETOut	B4TR_TC	O
5	D1	TMODE0	SCHMITT_TC	I
6	E3	PCLK	BD4TARP_TC	O
7	F5	Add0	B8TR_TC	O
8	G4	Add1	B8TR_TC	O
9	G2	Add2	B8TR_TC	O
10	G1	Add3	B8TR_TC	O
11	G5	Add4	B8TR_TC	O
12	H4	Add5	B8TR_TC	O
13	H5	Add6	B8TR_TC	O
14	H1	Add7	B8TR_TC	O
15	H3	Add8	B8TR_TC	O
16	J4	Add9	B8TR_TC	O
17	J5	Add10_AP	B8TR_TC	O
18	J2	Add11	B8TR_TC	O
19	J3	Add12	B8TR_TC	O
20	K5	Add13	B8TR_TC	O
21	K2	Add14	B8TR_TC	O
22	L1	Add15_GenConf0	BD8STRP_TC	I/O
23	K4	Add16_GenConf1	BD8STRP_TC	I/O
24	L2	Add17_GenConf2	BD8STRP_TC	I/O
25	M1	Add18_UART/JTAG	BD8STRP_TC	I/O
26	N1	Add19_BootRomBusWidth	BD8STRP_TC	I/O
27	M3	Add20_IEEE1284/XProcessor	BD8STRP_TC	I/O
28	N2	Add21_USBEnable	BD8STRP_TC	I/O
29	P1	Add22_Sdram/EDO	BD8STRP_TC	I/O
30	P3	Data0	BD8STRUQP_TC	I/O
31	N4	Data1	BD8STRUQP_TC	I/O
32	P4	Data2	BD8STRUQP_TC	I/O
33	N5	Data3	BD8STRUQP_TC	I/O

Table 2. Pin Description by PAD Types (*LH) (continued)

Ref.	Ball	Name	Type	Dir.
34	P5	Data4	BD8STRUQP_TC	I/O
35	M6	Data5	BD8STRUQP_TC	I/O
36	N6	Data6	BD8STRUQP_TC	I/O
37	P6	Data7	BD8STRUQP_TC	I/O
38	L6	Data8	BD8STRUQP_TC	I/O
39	M7	Data9	BD8STRUQP_TC	I/O
40	N7	Data10	BD8STRUQP_TC	I/O
41	K7	Data11	BD8STRUQP_TC	I/O
42	L7	Data12	BD8STRUQP_TC	I/O
Ref.	Ball	Name	Type	Dir.
43	M8	Data13	BD8STRUQP_TC	I/O
44	K8	Data14	BD8STRUQP_TC	I/O
45	P8	Data15	BD8STRUQP_TC	I/O
46	N8	nRAS0_nSDCS0	B4TR_TC	O
47	M9	nRAS1_nSDCS1_TDI	BD4STRUQP_TC	I/O
48	K9	nRAS2_nSDCS2_TDO	B4TR_TC	O
49	P9	nRAS3_nSDCS3_nTRST	BD4STRUQP_TC	I/O
50	N9	nSDRAS	B4TR_TC	O
51	M10	nSDCAS	B4TR_TC	O
52	P2	SDCLK	BD4TARP_TC	O
53	N10	CKE	B2TR_TC	O
Ref.	Ball	Name	Type	Dir.
54	P11	nWE	B8TR_TC	O
55	L10	nECS0	B4TR_TC	O
56	P12	nECS1	B4TR_TC	O
57	N11	nRCS0	B2TR_TC	O
58	P14	nRCS1	B2TR_TC	O
59	N12	nOE	B8TR_TC	O
60	N13	nCAS0_DQM0	B4TR_TC	O
61	M12	nCAS1_DQM1	B4TR_TC	O
62	M13	MDC	B4TR_TC	O
63	N14	MDIO	BD4STRUQP_TC	I/O
64	M14	COL/COL10M	SCHMITT_TC	I
65	K12	TXClk/TXClk10M	SCHMITT_TC	I
66	L14	TXD0/TXD010M	B4TR_TC	O

Table 2. Pin Description by PAD Types (*LH) (continued)

Ref.	Ball	Name	Type	Dir.
67	K13	TXD1/TXD110M	B4TR_TC	O
68	K14	TXD2/TXD210M	B4TR_TC	O
69	J12	TXD3/TXD310M	B4TR_TC	O
70	K10	TXEN/TXEN10M	B4TR_TC	O
71	J13	CRS/CRS10M	SCHMITT_TC	I
72	J14	RXClk/RXClk10M	SCHMITT_TC	I
73	J10	RXD0/RXD010M	SCHMITT_TC	I
74	H13	RXD1/RXD110M	SCHMITT_TC	I
75	H11	RXD2/RXD210M	SCHMITT_TC	I
76	H14	RXD3/RXD310M	SCHMITT_TC	I
77	H10	RxDV/LINK10M	SCHMITT_TC	I
78	G11	RXERR	SCHMITT_TC	I
79	G10	RXData_TCK	SCHMITT_TC	I
80	G14	TXData_TMS	BD4STRUQP_TC	I/O
81	A3	GPIO0	BD4STRUQP_TC	I/O
82	B4	GPIO1	BD4STRUQP_TC	I/O
83	A2	GPIO2	BD4STRUQP_TC	I/O
84	A1	GPIO3	BD4STRUQP_TC	I/O
Ref.	Ball	Name	Type	Dir.
85	B3	GPIO4_SCL	BD4STRUQP_TC	I/O
86	C3	GPIO5_SDA	BD4STRUQP_TC	I/O
87	F10	nXIRQ0	SCHMITT_TC	I
88	F14	nXIRQ1	SCHMITT_TC	I
89	E10	nXDRQ0	SCHMITT_TC	I
90	F12	nXDRQ1	SCHMITT_TC	I
91	E14	nXDACK0	B4TR_TC	O
92	E13	nXDACK1	B4TR_TC	O
93	E1	RTCXO	ANA	-
94	E2	RTCXI	OSCI32B	I
95	D14	XPData0_PpData0	BD8STRUQP_TC	I/O
96	E11	XPData1_PpData1	BD8STRUQP_TC	I/O
97	D13	XPData2_PpData2	BD8STRUQP_TC	I/O
98	B14	XPData3_PpData3	BD8STRUQP_TC	I/O
99	C13	XPData4_PpData4	BD8STRUQP_TC	I/O
100	C12	XPData5_PpData5	BD8STRUQP_TC	I/O

Table 2. Pin Description by PAD Types (*LH) (continued)

Ref.	Ball	Name	Type	Dir.
101	B13	XPData6_PpData6	BD8STRUQP_TC	I/O
102	B12	XPData7_PpData7	BD8STRUQP_TC	I/O
103	A13	XPData8	BD8STRUQP_TC	I/O
104	B11	XPData9	BD8STRUQP_TC	I/O
105	C10	XPData10	BD8STRUQP_TC	I/O
106	A11	XPData11	BD8STRUQP_TC	I/O
107	B10	XPData12	BD8STRUQP_TC	I/O
108	D10	XPData13	BD8STRUQP_TC	I/O
109	B9	XPData14	BD8STRUQP_TC	I/O
110	A9	XPData15	BD8STRUQP_TC	I/O
111	D9	XPAddr0_nSTROBE	BD2STRUQP_TC	I/O
112	C9	XPAddr1_nACK	SCHMITT_TC	I
113	A8	XPAddr2_Busy	SCHMITT_TC	I
114	E8	XPAddr3_PError	SCHMITT_TC	I
115	D8	XPAddr4_Select	SCHMITT_TC	I
116	C8	XPAddr5_nAutoFd	BD2STRUQP_TC	I/O
117	E7	XPAddr6_nFault	SCHMITT_TC	I
118	D7	XPAddr7_nInit	BD2STRUQP_TC	I/O
119	B7	XPAddr8_SelectIn	BD2STRUQP_TC	I/O
120	C7	XPAddr9_PpDataDir	BD2STRUQP_TC	I/O
121	A6	XPAddr10	SCHMITT_TC	I
122	B6	XPAddr11	SCHMITT_TC	I
123	D6	XPAddr12	SCHMITT_TC	I
124	C6	nXPCS	SCHMITT_TC	I
125	D5	nXPWE	SCHMITT_TC	I
126	A5	nXPRE	SCHMITT_TC	I
127	A4	nXPWAIT	B4TR_TC	O
Ref.	Ball	Name	Type	Dir.
128	C5	nXPIRQ	B4TR_TC	O
129	F11	UHD+	USB_PAD	I/O
130	G13	UHD-	USB_PAD	I/O
131	E6, A12, E12, G12, L13, P10, L8, K6, M5, M2, K1, H2, F4	VDD3I/O	Power	-

Table 2. Pin Description by PAD Types (*LH) (continued)

Ref.	Ball	Name	Type	Dir.
144	D2, G3, J1, K3, N3, L5, P7, L9, P13, K11, H12, C14, F13, A14, A10	VSS	Power	-
459	F3	VDDRTC	Power	-
160	E5	VSSRTC	Power	
161	B2	VDD3PLL	Power	-
162	C2	VSSPLL	Power	
163	F1, J11, E9	VDD	Power	-
166	B8, A7, B5	VSS	Power	-
169	C4, C11, D3, D4, D11, D12, L3, L4, L11, L12, M4, M11	NC	Not Connected	-

Table 3. PAD Description

PAD Name	Description
ANA	Analog PAD Buffer
B2TR_TC	TTL Output Pad Buffer, 3 V capable, 2mA drive capability
B4TR_TC	TTL Output Pad Buffer, 3 V capable, 4mA drive capability
B8TR_TC	TTL Output Pad Buffer, 3 V capable, 8mA drive capability
BD2STRUQP_TC	TTL Schmitt Trigger Bidirectional Pad Buffer, 2mA drive capability, 3 V Capable, with Pull-Up
BD4STRUQP_TC	TTL Schmitt Trigger Bidirectional Pad Buffer, 4mA drive capability, 3 V Capable with Pull-Up
BD4TARP_TC	TTL Bidirectional Pad Buffer, 3 V capable, 4mA drive capability, Active Slew Rate
BD8STRP_TC	TTL Schmitt Trigger Bidirectional Pad Buffer, 8mA drive capability, 3 V Capable
BD8STRUQP_TC	TTL Schmitt Trigger Bidirectional Pad Buffer, 8mA drive capability, 3 V Capable with Pull-Up
OSCI27B	Oscillator (Max Frequency 27 MHz)
OSCI32B	Oscillator (32 KHz)
SCHMITT_TC	TTL Schmitt Trigger Input Pad Buffer 3 V tolerant
USB_PAD	USB Transceiver
V_{DD3}IOCO	Power Pad - Internal supply for 3.3 V level
V_{DD}CO	Power Pad - Internal supply for 1.8 V level
V_{SS}CO	Power Pad - Internal ground, for Core only
V_{SS}IOCO	Power Pad - Internal ground

5 Memory Map

5.1 Global MAP (AHB)

Table 4. AHB Memory Map

Starting Address	End Address	Description
0x000.0000	0x01FF.FFFF	Two ROM/FLASH/SRAM Banks. – Bank size granularity: 64 KB – Max Bank size: 16 MB – The two banks are adjacent.
0x1000.0000	0x17FF.FFFF	Four SDRAM/EDO Banks. – Max Bank size: 32 MB – Bank size granularity: 64 KB – The four banks are adjacent.
0x2000.0000	0x2000.7FFF	Two External I/O Banks Bank size: 16 KB
0x2100.0000	0x2100.1FFF	8 KB Shared SRAM
0x2200.0000	0x2200.0BFF	IEEE1284 Interface & FIFOs
0x2300.0000	0x2300.03FF	ARM Slave Test Interface
0x3000.0000	0x3000.37FF	APB Bridge

Note: The decoder will ignore the ADD31 lines. In this way will be possible to access all the devices trough cache in range 0x000.0000 - 0x7FFF.FFFF and without cache in range 0x8000.0000 - 0xFFFF.FFFF

5.2 I/O MAP (APB)

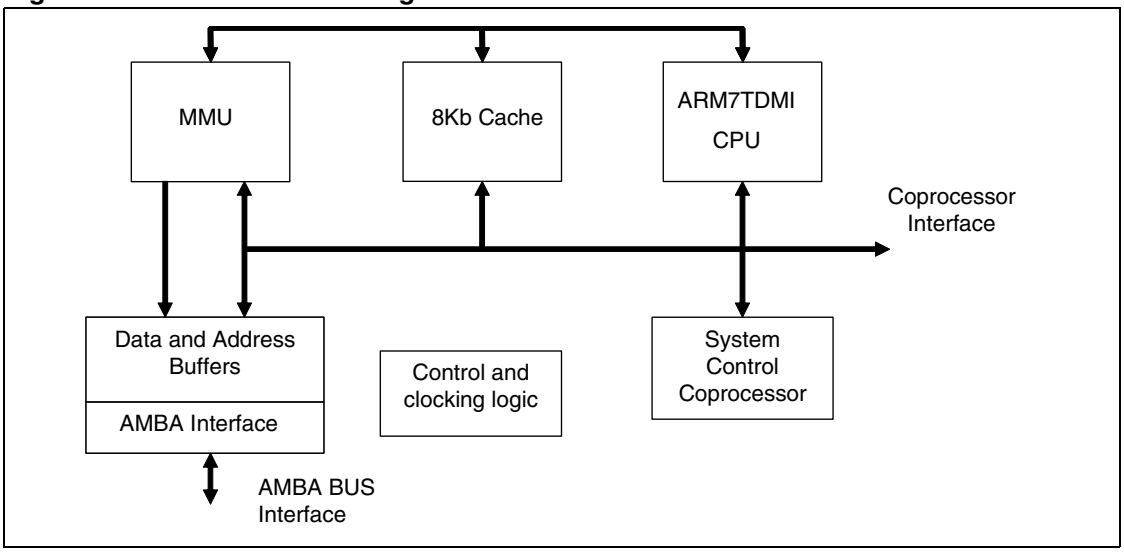
Table 5. APB Memory Map

Starting Address	End Address	Description
0x3000.0000	0x3000.03FF	Interrupt Controller
0x3000.0400	0x3000.07FF	General Purpose Timers
0x3000.0800	0x3000.0BFF	Watch Dog Timer
0x3000.0C00	0x3000.0FFF	Real Time Clock
0x3000.1000	0x3000.13FF	General Purpose I/O
0x3000.1400	0x3000.17FF	I^2C Interface
0x3000.1800	0x3000.1BFF	UART
0x3000.1C00	0x3000.1FFF	Configuration Registers
0x3000.2000	0x3000.23FF	DMA Controller General Purpose
0x3000.2400	0x3000.27FF	Static Memory Controller
0x3000.2800	0x3000.2BFF	Dynamic Memory Controller
0x3000.2C00	0x3000.2FFF	USB Host Controller
0x3000.3000	0x3000.33FF	DMA MAC
0x3000.3400	0x3000.37FF	MAC Ethernet Controller

6 Blocks description

6.1 CPU SUBSYSTEM & AMBA BUS

Figure 2. ARM720T Block Diagram



6.1.1 ARM720 Processor

The ARM720T is a general purpose 32-bit RISC microprocessor with 8KB cache, enlarged write buffer and Memory Management Unit (MMU) combined in a single chip.

The CPU within ARM720T is the ARM7TDMI.

The on-chip mixed data and instruction cache, together with the write buffer, substantially raise the average execution speed and reduce the average amount of memory bandwidth required by the processor.

The MMU supports a conventional two-level, page-table structure and the memory interface has been designed to allow the performance potential to be realized without incurring high costs in the memory system.

6.1.2 MMU Overview

The Memory Management MMU performs two primary functions. It:

- translates virtual addresses into physical addresses
- controls memory access permissions

The MMU hardware required to perform these functions consists of:

- a Translation Look-aside Buffer (TLB)
- access control logic
- translation-table-walking logic

When the MMU is turned off (as happens on reset), the virtual address is output directly onto the physical address bus.

6.1.3 Instruction and Data Cache overview

ARM720T contains an 8KB mixed instruction and data cache (IDC).

The cache only operates on a write-through basis with a read-miss allocation policy and a random replacement algorithm.

The IDC has 512 lines of 16 bytes (four words), arranged as a 4-way set-associative cache, and uses the virtual addresses generated by the processor core after relocation by the Process Identifier as appropriate.

The IDC is always reloaded a line at a time (4 words). It may be enabled or disabled via the ARM720T Control Register and is disabled immediately after the Power-On Reset.

The operation of the cache is further controlled by the Cacheable (C bit) stored in the Memory Management Page Table.

For this reason, the MMU must be enabled in order to use the IDC.

However, the two functions may be enabled simultaneously, with a single write to the Control Register.

6.1.4 Write Buffer Overview

The ARM720T write buffer is provided to improve system performance.

It can buffer up to eight words of data, and four independent addresses and may be enabled or disabled via the W bit (bit 3) in the ARM720T Control Register.

The buffer is disabled and flushed on reset.

The write buffer operation is further controlled by the Bufferable (B) bit, which is stored in the Memory Management Page Tables. For this reason, the MMU must be enabled so you can use the write buffer. The two functions may however be enabled simultaneously, with a single write to the Control Register.

6.1.5 Configuration

The operation and configuration of ARM720T is controlled:

- directly via coprocessor instructions
- indirectly via the Memory Management Page tables

The coprocessor instructions manipulate a number of on-chip registers which control the configuration of the following:

- Cache
- Write buffer
- MMU
- A number of other configuration options

6.1.6 Coprocessor Registers Programming

The ARM720T instruction set allows specialized additional instruction to be implemented using coprocessor.

The Memory Unit in the ARM720T core is referred as Coprocessor 15 (CP15).

Important: CP15 registers can only be accessed with **MRC** and **MCR** instructions in a **Privileged mode**. CDP, LDC and STC instructions, as well as unprivileged MRC and MCR instructions to CP15 cause the undefined instruction trap to be taken.

The bit fields of the instruction are shown in the following table:

Table 6. MRC and MCR (CP15) bit pattern

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Cond.				**1**	**1**	**1**	**0**	**Opcode_1**			**L**	**CRn**				**Rd**				**1**	**1**	**1**	**1**	**Opcode_2**			**1**	**CRm**			

Symbol Description

Cond: Condition Code Field

L: Direction

0 Store to Coprocessor (MCR)

1 Load from Coprocessor (MRC)

Rd: ARM register

CRn: Coprocessor Register

CRm: Should be zero except when accessing register 7, 8 and 13.

opcode_1: Should be zero

opcode_2: Should be zero except when accessing register 7, 8 and 13.

Note that the CPID field, bit 11:8, is set to 15 (MMU Coprocessor).

The assembler syntax is:

<MCR|MRC>{cond} p15, opcode_1, Rd, CRn, CRm, opcode_2

6.1.6.1 Registers

Register 0, ID (RO)

It is a read-only register. CRm and opcode_2 should be zero.

Reading from this register return always 0x41807203. Last nibble is the revision number.

Register 1, Control (R/W)

CRm and opcode_2 should be zero.

The control bits pattern is shown in

Table 7. Control Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNP/SBZ																		**V**	**UNP/SBZ**			**R**	**S**	**B**	**L**	**D**	**P**	**W**	**C**	**A**	**M**

M	MMU enable/disable bit (0= disable, 1 = enable)
A	Alignment fault Enable/disable bit (0 = disable, 1 = enable)
C	Cache enable/disable bit (0 = disable, 1 = enable)
W	Write Buffer enable/disable bit (0 = disable, 1 = enable)
P	When read return always 1. When written is ignored.
D	When read return always 1. When written is ignored.
L	When read return always 1. When written is ignored.
B	Endianess bit (0 = Little Endian, 1 = Big Endian)
S	System Protection (See Access Permission AP Bits)
R	ROM Protection (See Access Permission Bits)
V	Location of exception vectors (Windows CE)
UNP/SBZ	**Unp**redictable when read, **S**hould **B**e **Z**ero when written.

Example:

```
ldr r0, =0x0F                ; Enable MMU with cache, write buffer and
MCR p15, 0, r0, 1, 0, 0      ; alignment fault
```

Register 2, Translation Table Base (R/W)

CRm and opcode_2 should be zero.

This is the currently active first-level translation table.

Only bit 31:14 are valid. The others are unpredictable when read, should be zero if written.

Table 8. TTB Register

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

Register 3, Domain Access Control (R/W)

CRm and opcode_2 should be zero.

The Domain Access Control Register consists of 16 2-bit fields, each of which defines the access permissions for one of the 16 Domains (D15-D0).

The meaning of this bit is described in the MMU translation mechanism.

Table 9. DAC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15		**D14**		**D13**		**D12**		**D11**		**D10**		**D9**		**D8**		**D7**		**D6**		**D5**		**D4**		**D3**		**D2**		**D1**		**D0**	

Register 4 (Reserved)

Register 5, Fault Status Register (FSR)

Register 6, Fault Address Register (FAR)

Register 7, Cache Operations (WO)

For this operation opcode_2 must be 0x0b000 and CRm must be 0x0b0111.

This operation invalidates all cache data. Use with caution. Reading from it is undefined.

Example:

```
MCR p15, 0, r0, c7, c7, 0 ; invalidate all the data inside cache
```

Register 8, TLB Operations

Two operation are defined as showed on the following table:

Table 10. TLB Operation

Function	Opcode_2	CRm	(Rd)	Assembler Syntax
Invalidate entire TLB	0b000	0b0111	0	MCR p15, 0, Rd, C8, C7, 0
Invalidate TLB (Single entry)	0b001	0b0111	Virtual Address	MCR p15, 0, Rd, C8, C7, 1

The Invalidate TLB invalidates **all** of the unlocked entries in the TLB.
The invalidate TLB single entry invalidates any TLB entry corresponding to the Virtual Address in Rd.

Register 9 to 12 are Reserved

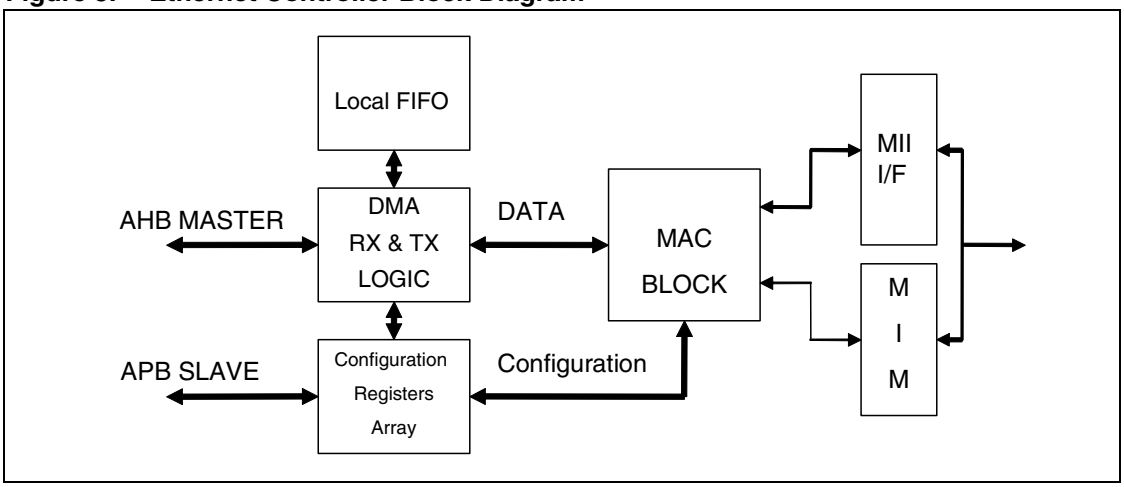
Register 13, Oricess Identifier

Not Used

Register 14 to 15 are Reserved

6.2 MAC Ethernet Controller

Figure 3. Ethernet Controller Block Diagram



6.2.1 Overview

The Ethernet Media Access Controller (MAC110) core incorporates the essential protocol requirements for operation of an Ethernet/IEEE 802.3 compliant node, and provides interface between the host subsystem and the Media Independent Interface (MII). The MAC110 core can

operate either in 100Mbps mode or the 10Mbps mode based on the clock provided on the MII interface (25/2.5 MHz).

The MAC110 core operates both in half-duplex mode and full-duplex modes. When operating in the half- duplex mode, the MAC110 core is fully compliant to Section 4 of ISO/IEC 8802-3 (ANSI/IEEE Standard) and ANSI/IEEE 802.3. When operating in the full-duplex mode, the MAC110 core is compliant to the IEEE 802.3x standard for full-duplex operations. It is also compatible with Home PNA 1.1.

The MAC110 core provides programmable enhanced features designed to minimize host supervision, bus utilization, and pre- or post-message processing. These features include ability to disable retires after a collision, dynamic FCS generation on a frame-by-frame basis, automatic pad field insertion and deletion to enforce minimum frame size attributes, automatic retransmission and detection of collision frames.

The MAC110 core can sustain transmission or reception of minimal-sized back -to-back packets at full line speed with an inter-packet gap (IPG) of 90.6 us for 10-Mb/s and 0.96 us for 100-Mb/s.

The five primary attributes of the MAC block are:

1. Transmit and receive message data encapsulation
   - Framing (frame boundary delimitation, frame synchronization)
   - Error detection (physical medium transmission errors)
2. Media access management
   - Medium allocation (collision detection, except in full-duplex operation)
   - Contention resolution (collision handling, except in full-duplex operation)
3. Flow Control during Full Duplex mode
   - Decoding of Control frames (PAUSE Command) and disabling the transmitter
   - Generation of Control Frames
4. Interface to the PHY
   - Support of MII protocol to interface with a MII based PHY.
5. Management Interface support on MII
   - Generation of PHY Management frames on the MDC/MDI/MDO.

To minimize the CPU load during the data transfer is available a local DMA with FIFO capable to fetch itself the descriptors for the data blocks and to manage the data according to the instruction included on the descriptor.

6.2.2 Transfer Logic

6.2.2.1 RX LOGIC

The receive (RX) DMA block includes all the logic required to manage data transfers from the RX port of the MAC110 wrapper to an external AHB memory mapped device.

It includes:

- RX wrapper interface
- RX FIFO
- RX DMA master SM
- DMA descriptor SM

6.2.2.2 RX WRAPPER INTERFACE

The wrapper interface is a simple synchronous interface with RX_nREQ, RX_nACK, RX_DATA signals for data handshake, plus some sideband signals for the MAC protocol support (see later). The data path (RX_DATA) is 32 bit wide.

When the RX DMA logic has been enabled, after a valid descriptor fetch, the RX interface control logic starts driving the RX_nACK signal, de-asserting it when

- the internal FIFO becomes full or
- the DMA transfer completes.

The wrapper logic will drive the RX_nREQ signal when it has data valid to be transferred: the transfer is done, and the data can be updated, if RX_nREQ and RX_nACK are both asserted on the same clock.

RX FIFO

The FIFO depth can be 2/4/8/16/32 entries, 32 bit each.

The RX FIFO is loaded by the RX wrapper interface logic and read by the RX DMA master SM.

The FIFO download is done with 32 bits operations (possibly burst type to optimize the bus bandwidth).

If there are some incomplete words coming from the MAC core (this con occurs only at the end of the frame) the DMA adds some dummy bytes in order to complete the word and increase the performance. Added bytes have an undefined value.

RX DMA MASTER SM

The RX DMA block has a State Machine (SM) dedicated to the DMA master operation. When enabled via the RX configuration registers, it's able to manage the RX data transfer without further processor intervention.

The DMA transfer can be:

- DMA continuous/fixed size: the DMA can be required to run indefinitely or to stop after a configured number of data bytes has been transferred
- fixed/incrementing address: the DMA address can be fixed (i.e. all the data are transferred to the same AHB word aligned address) or it can be updated after each data transfer
- linear incrementing or wrapping address: when the address is defined as incrementing, it can be required that, once reached a programmed value, the address counter wraps back to the initial address value (the address location, pointed by the wrapping address, is not modified)
- with FIFO entry threshold: the DMA SM starts transferring data on the AHB bus when a programmable number of 32 bit RX FIFO entries is valid

When the DMA is enabled, as soon as data appears in the FIFO, the DMA may either initiate an AHB transfer immediately, or be delayed until X data bytes are available in the FIFO (FIFO entry threshold).

The DMA can be configured to wrap-round the AHB address at some point to implement a circular buffer in CPU memory.

The DMA can be configured to run indefinitely or to stop after DMA_XFERCOUNT data have been transferred.

When the DMA completes, the master DMA SM can be required to assert an interrupt request to the processor and wait for new instruction, or to wake up the DMA descriptor SM to require a new DMA descriptor fetch.

To save gates, the implementation limits the maximum DMA transfer count to 4 Kbytes, hence the XFER_COUNT field in the DMA control registers is limited to 12 bits.

The DMA start address (DMA_ADDRESS) must be 32 bit word aligned.

The DMA wrapping address point must be 32 bit word aligned.

If an AHB error condition occurs, while the DMA is running, the SM activity is suspended, until the error interrupt bit (MERR_INT) is reset. When the error condition is removed the DMA makes the same request previously interrupted by the error response.

DMA descriptor SM

A dedicated SM has been implemented that, when required by the DMA master logic, starts some AHB master read operations to load from the external memory all the information (DMA descriptors) required to start the new DMA data transfer.

The DMA descriptor consists of a VALID bit plus 3 registers: the DMA control (DMA_CTL), the DMA base address (DMA_ADDR) and the DMA next descriptor address register (DMA_NXT).

The Host Processor must ensure that the descriptors are up to date in memory when the DMA descriptor SM loads them. The fetch order is: DMA_CTL, DMA_ADDR, DMA_NXT and VALID bit.

If a fetched descriptor is not valid (VALID=0), then the DMA engine can be programmed to stop the operation (reset the DMA_EN bit in RX_DMA_START) and raise an interrupt (RX_DONE), or to repeat the descriptor fetch operation, until a valid descriptors is found.

The interrupt register bit named RX_NEXT is always set when a not valid descriptor is loaded.

In the first case, the DMA will then wait for the Host Processor to re-enable the DMA operation (START_FETCH bit in the

RX_DMA_START register set to 1) before attempting anew descriptor fetch.

While, when in polling mode, the DMA will keep reloading the descriptor, with an access frequency determined by the DFETCH_DLY field in the RX_DMA_START register.

An AHB ERROR response suspends the descriptor SM activity and reset the DMA_EN bit in RX_DMA_START register. To help the error source understanding, the RX_DMA_CADDR register value is the address at which the error occurred.

After clearing the error bit, the SW needs to reprogram the DMA registers, to start again a new descriptor fetch.

6.2.2.3 TX LOGIC

The transmit (TX) DMA block includes all the logic required to manage data transfers from an external AHB memory mapped device to the TX port of the MAC110 wrapper.

It includes:

- TX wrapper interface
- TX FIFO
- TX DMA master SM
- DMA descriptor SM

TX WRAPPER INTERFACE

The wrapper interface is a simple synchronous interface with TX_nREQ, TX_nACK, TX_DATA signals for data handshake, plus some sideband signals for the MAC protocol support (see later). The data path (TX_DATA) is 32 bit wide.

When the TX DMA logic has been enabled, as soon as the internal FIFO is no more empty the TX interface control logic starts driving the TX_nACK signal, de-asserting it when the internal FIFO becomes empty again or the DMA transfer completes.

The wrapper has to drive the TX_nREQ signal when it accept valid data to be transferred: the transfer is done and the data can be updated if TX_nREQ and TX_nACK are both asserted on the same clock.

TX FIFO

The FIFO depth can be 2/4/8/16/32 entries, 32 bit each. The TX FIFO is loaded by the TX DMA master SM and read by the TX wrapper interface. The FIFO load is usually done with 32 bits operations (possibly burst type to optimize the bus bandwidth), unless the DMA end has been reached and the DMA buffer size is not a multiple of 32 bits.

TX DMA MASTER SM

The TX DMA block has a State Machine (SM) dedicated to the DMA master operation. When enabled via the TX configuration registers, it's able to manage the TX data transfers without further processor intervention.

The DMA transfer can be:

DMA continuous/fixed size: the DMA can be required to run indefinitely or to stop after a configured number of data bytes has been transferred

fixed/incrementing address: the DMA address can be fixed (i.e. all the data are transferred from the same AHB, word aligned, address) or it can be updated after each data transfer

linear incrementing or wrapping address: when the address is defined as incrementing, it can be required that, once reached a programmed value, the address counter wraps back to the initial address value (the address location, pointed by the wrapping address, is not accessed)

with FIFO entry threshold: the DMA SM starts transferring data on the AHB bus when a programmable number of 32 bit TX FIFO entries is empty

When the DMA is enabled, as soon as one free entry is available in the FIFO, the DMA may initiate AHB transfers immediately, or can be delayed. The DMA may be delayed until X data entries are available in the FIFO (FIFO entry threshold).

The DMA can be configured to wrap-round the AHB address at some point to implement a circular buffer in CPU memory.

The DMA can be configured to run indefinitely or to stop after DMA_XFERCOUNT data have been transferred.

When the DMA completes, the master DMA SM can be required to assert an interrupt request to the processor and wait for new instruction, or to wake up the DMA descriptor SM, to require a new DMA descriptor fetch.

To save gates, the implementation limits the maximum DMA transfer count to 4Kbytes, hence the XFER_COUNT field in the DMA control registers is limited to 12 bits.

The DMA start address (DMA_ADDRESS) must be 32 bit word aligned and the DMA wrapping address point must be 32 bit word aligned.

If an AHB error condition occurs, while the DMA is running, the SM activity is suspended, until the error interrupt bit (MERR_INT) is reset. When the error condition is removed the DMA makes the same request previously interrupted by the error response.

Special care must be taken when the FIFO entry to be read has 3 valid bytes: in this case, because the AHB protocol doesn't allow to 3 byte transfers, the AHB master splits the transfer in two single transfers (byte + half or half + byte) and sends an acknowledge signal to the FIFO only when the second one has been read. If the second read receives an error response then, when the error condition is removed, the DMA repeats even the first one (because the FIFO has not yet see the acknowledge).

DMA DESCRIPTOR SM

A dedicated SM has been implemented that, when required by the DMA master logic, starts some AHB master read operations to load from the external memory all the information (DMA descriptors) required to start the DMA data transfer.

The DMA descriptor consists of a VALID bit plus 3 registers: the DMA control (DMA_CTL), the DMA base address (DMA_ADDR) and the DMA next descriptor address register (DMA_NXT).

The Host Processor must ensure that the descriptors are up to date in memory when the DMA descriptor SM loads them. The fetch order is: DMA_CTL, DMA_ADDR, DMA_NXT and VALID bit.

If a fetched descriptor is not valid (VALID=0), then the DMA engine can be programmed to stop the operation (reset the DMA_EN bit in TX_DMA_START) and raise an interrupt (TX_DONE), or to repeat the descriptor fetch operation, until a valid descriptors is found. The interrupt register bit named TX_NEXT is always set when a not valid descriptor is loaded.

In the first case, the DMA will then wait for the HP to re-enable the DMA operation (START_FETCH bit in the TX_DMA_START register set to 1) before attempting a new descriptor fetch.

While, when in polling mode, the DMA will keep reloading the descriptor, with an access frequency determined by the DFETCH_DLY field in the TX_DMA_START register.

An AHB ERROR response suspends the descriptor SM activity and reset the DMA_EN bit in TX_DMA_START register. To help the error source understanding, the TX_DMA_CADDR register value is the address at which the error occurred.

After clearing the error bit, the SW needs to reprogram the DMA registers, to start again a new descriptor fetch.

6.2.3 Ethernet register map

Table 11. Ethernet register map

Address	Register Name	Description
0x3000_3000	DMA_STS_CNTL	Ethernet DMA, status and control register
0x3000_3004	DMA_INT_EN	Ethernet DMA, Interrupt sources enable register
0x3000_3008	DMA_INT_STS	Ethernet DMA, Interrupt status register
0x3000_300C	Reserved	-
0x3000_3010	RX_DMA_START	Ethernet DMA, RX start register

Table 11. Ethernet register map (continued)

Address	Register Name	Description
0x3000_3014	RX_DMA_CNTL	Ethernet DMA, RX control register
0x3000_3018	RX_DMA_ADDR	Ethernet DMA, RX base address register
0x3000_301C	RX_DMA_NXT	Ethernet DMA, RX next descriptor address register
0x3000_3020	RX_DMA_CADDR	Ethernet DMA, RX current address register
0x3000_3024	RX_DMA_CXFER	Ethernet DMA, RX current transfer count register
0x3000_3028	RX_DMA_TO	Ethernet DMA, RX time out register
0x3000_302C	RX_DMA_FIFO	Ethernet DMA, RX FIFO status register
0x3000_3030	TX_DMA_START	Ethernet DMA, TX start register
0x3000_3034	TX_DMA_CNTL	Ethernet DMA, TX control register
0x3000_3038	TX_DMA_ADDR	Ethernet DMA, TX base address register
0x3000_303C	TX_DMA_NXT	Ethernet DMA, TX next descriptor address register
0x3000_3040	TX_DMA_CADDR	Ethernet DMA, TX current address register
0x3000_3044	TX_DMA_CXFER	Ethernet DMA, TX current transfer count register
0x3000_3048	TX_DMA_TO	Ethernet DMA, TX time out register
0x300_304C	TX_DMA_FIFO	Ethernet DMA, TX FIFO status register
0x3000_3050 - 0x3000_30FC	reserved	-
0x3000_3100 - 0x3000_317C	RX_FIFO	RX local FIFO (32 double word = 128 byte)
0x3000_3180 - 0x3000_31FC	Reserved	-
0x3000_3200 - 0x3000_327C	TX_FIFO	TX local FIFO (32 double word = 128 byte)
0x3000_3280 - 0x3000_33FC	Reserved	-
0x3000_3400	MAC_CNTL	MAC control register
0x3000_3404	MAC_ADDH	MAC address high register
0x3000_3408	MAC_ADDL	MAC address low register
0x3000_340C	MAC_MCHTH	MAC, multi cast hash table high register
0x3000_3410	MAC_MCHTL	MAC, multi cast hash table low register
0x3000_3414	MII_ADDR	MII, address register
0x3000_3418	MII_DATA	MII, data register
0x3000_341C	FCR	Flow control register
0x3000_3420	VLAN1	VLAN1 tag register
0x3000_3424	VLAN2	VLAN2 tag register
0x3000_3428 - 0x3000_34FC	Reserved	-

Table 11. Ethernet register map (continued)

Address	Register Name	Description
0x3000_3500	MMC_CTRL_REG	MMC Statistic Control Register
0x3000_3504	MMC_INT_HI_REG	MMC Statistic Interrupt High Register
0x3000_3508	MMC_INT_LO_REG	MMC Statistic Interrupt Low Register
0x3000_350C	MMC_INT_MSK_HI_REG	MMC Statistic Interrupt Mask High Register
0x3000_3510	MMC_INT_MSK_LO_REG	MMC Statistic Interrupt Mask Low Register
0x3000_3600	RxNumFrmsAllCntr	All frames received counter. This includes good and bad frames. Counter is incremented each time a frame is received.
0x3000_3604	RxNumFrmsOkCntr	Good frame received counter. This includes good frames only, which are free from runt frame error, frame too long error, collision error, MII error, dribble bit error, CRC error, invalid length error and FIFO overflow error.
0x3000_3608	RxCntrlFrmsCntr	Control frames received counter. This includes control frames, which are free from runt frame error, frame too long error, collision error, MII error, dribble bit error, CRC error, invalid length error and FIFO overflow error.
0x3000_360C	RxUnsupCntrlCntr	Unsupported control frames received counter. This includes control frames, which are free from runt frame error, frame too long error, collision error, MII error, dribble bit error, CRC error, invalid length error and FIFO overflow error but whose length/type field is not supported.
0x3000_3610	RxNumBytsAllCntr	No of bytes received counter. This includes all frames frame length added. Excludes preamble.
0x3000_3614	RxNumBytsOkCntr	No of bytes received counter. This includes all frames frame length added which are free from runt frame error, frame too long error, collision error, MII error, dribble bit error, CRC error, invalid length error and FIFO overflow error.
0x3000_3618	RxLenEqual64Cntr	Frame with length equal to 64 bytes counter. Includes good and bad frames.
0x3000_361C	RxLen65_127Cntr	Frame with length from 65 to 127 bytes counter. Includes good and bad frames.
0x3000_3620	RxLen128_255Cntr	Frame with length from 128 to 255 bytes counter. Includes good and bad frames.
0x3000_3624	RxLen256_511Cntr	Frame with length from 256 to 511 bytes counter. Includes good and bad frames.
0x3000_3628	RxLen512_1023Cntr	Frame with length from 512 to 1023 bytes counter. Includes good and bad frames.
0x3000_362C	RxLen1024_MaxCntr	Frame with length from 1024 to MaxPktSize bytes counter. Includes good and bad frames.

Table 11. Ethernet register map (continued)

Address	Register Name	Description
0x3000_3630	RxUnicastCntr	Unicast frames received counter. This includes good frames only.
0x3000_3634	RxMulticastCntr	Multicast frames received counter. This includes good frames only.
0x3000_3638	RxBroadcastCntr	Broadcast frames received counter. This includes good frames only.
0x3000_363C	RxFifoOverflowCntr	Frames with FIFO error counter. Counter with Rx FIFO overflow error. This counter is incremented if the missed frame bit in receive status is set.
0x3000_3640	RxMinLenCntr	Runt frame counter. Counter for frames with a minimum frame length violation. This counter is incremented if the runt frame bit in receive status is set.
0x3000_3644	RxMaxLenCntr	Long frames counter. Counter for frames with a maximum frame length violation. This counter is incremented if the Frame Too Long bit in receive status is set.
0x3000_3648	RxCrcErrorCntr	Frame with a CRC error counter. This counter is incremented if the CRC error bit in receive status is set.
0x3000_364C	RxAlignErrorCntr	Frames with a dribble bit counter. This counter is incremented if the dribble bit in receive status is set.
0x3000_3650	RxLenghtErrCntr	Length error frames counter. This counter is incremented if the length error bit in receive status is set.
0x3000_3654	RxEthrTypFrmCntr	Ethernet frames counter. This counter is incremented when the frame type bit in receive status is set.
0x3000_3658 - 0x3000_36FF	Reserved	
0x3000_3700	TxNumFrmsAllCntr	No of frames transmitted counter. This includes good and bad frames but no retries. Counter is incremented each time the transmit status is received.
0x3000_3704	TxCntrlFrmsCntr	No of control frames transmitted counter. Counter for good control frames only. Counter is incremented each time transmit status is received and if the frame transmitted is a control frame. Does not include retries.
0x3000_3708	TxNumBytsAllCntr	Total number of transmitted bytes counter. Counter is incremented each time transmit status is received. Includes good and bad frames but no retries.
0x3000_370C	TxNumBytsOkCntr	Total number of (well) transmitted bytes counter. Counter for bytes of a good frame transmitted. Does not include retries. The good frames are the ones for which the frame abort and packet retry bits in transmit status are reset.

Table 11. Ethernet register map (continued)

Address	Register Name	Description
0x3000_3710	TxLenEqual64Cntr	Frames with length equal to 64 counter. Counter for frames with length equal to 64. Includes good and bad frames but no retries.
0x3000_3714	TxLen65_127Cntr	Frames with length from 65 to 127 counter. Counter for frames with length between 65 and 127 bytes. Includes good and bad frames but no retries.
0x3000_3718	TxLen128_255Cntr	Frames with length from 128 to 255 counter. Counter for frames with length between 128 and 255 bytes. Includes good and bad frames but no retries.
0x3000_371C	TxLen256_511Cntr	Frames with length from 256 to 511 counter. Counter for frames with length between 256 and 511 bytes. Includes good and bad frames but no retries.
0x3000_3720	TxLen512_1023Cntr	Frames with length from 512 to 1023 counter. Counter for frames with length between 512 and 1023 bytes. Includes good and bad frames but no retries.
0x3000_3724	TxLen1024_MaxCntr	Frames with length from 1024 to MaxPktSize counter. Counter for frames with length between 1024 and maximum frame lenght bytes. Includes good and bad frames but no retries.
0x3000_3728	TxUnicastCntr	No of Unicast frames transmitted counter. Includes good frames only, no retries.
0x3000_272C	TxMulticastCntr	No of multicast frames transmitted counter. Includes good frames only, no retries.
0x3000_3730	TxBroadcastCntr	No of broadcast frames transmitted counter. Includes good frames only, no retries.
0x3000_3734	TxFifoUndFloCntr	No of frames aborted due to FIFO error counter. This counter is incremented when the under run bit in transmit status is set.
0x3000_3738	TxNumBadFrmsCntr	No of frames aborted counter. This counter is incremented if either the frame aborted or the heart bit fail are set in transmit status.
0x3000_373C	TxSingleColCntr	No of frames with single collision counter. The counter is incremented when the collisions count = 1 and packet retry set in transmit status.
0x3000_3740	TxMultiColCntr	No of frames with multiple collisions counter. The counter is incremented when the collisions count > 1 and packet retry set in transmit status.
0x3000_3744	TxNumDeffredCntr	No of frames deferred counter. This counter is incremented when the deferred bit in transmit status is set.
0x3000_3748	TxLateColCntr	No of frames with late collision counter. This counter is incremented when the late collision bit in transmit status is set.

Table 11. Ethernet register map (continued)

Address	Register Name	Description
0x3000_374C	TxAbortedFrmsCntr	No of frames aborted counter. This counter is incremented when the frame aborted bit in transmit status is set.
0x3000_3750	TxNoCrsCntr	Number of frames with no carrier counter. This counter is incremented when either the no carrier or the loss of carrier bit in transmit status is set.
0x3000_3754	TxXsDeferalCntr	No of frames with excessive deferral counter. This counter is incremented when the excessive deferral bit in transmit status is set.
0x3000_3758 - 0x3000_37FC	Reserved	

6.2.4 Register description

All the registers are 32 bit wide.

6.2.4.1 Ethernet DMA, Status and Control Register

Mnemonic: DMA_STS_CNTL

Address: 0x3000_3000

Default value: 4A4A0101

Bit	Field name	Access
31 - 28	TX_FIFO_SIZE	RO
27 - 26	TX_IO_DATA_WIDTH	RO
25 - 24	TX_CHANNEL_STATUS	RO
23 - 20	RX_FIFO_SIZE	RO
19 - 18	RX_IO_DATA_WIDTH	RO
17 - 16	RX_CHANNEL_STATUS	RO
15 - 08	REVISION	RO
Bit	**Field name**	**Access**
07 - 06	TX_MAX_BURST_SIZE	RW
05 - 04	RX_MAX_BURST_SIZE	RW
03 - 02	Reserved	RO
01	LOOPB	RW
00	SRESET	RW

TX_FIFO_SIZE: Size of transmitter data path FIFO.

Value: 04 → 16 * 32 bit words.

TX_IO_DATA_WIDTH: Width of the I/O bus transmit data path.

Value: 2'b10 → 32 bit.

TX_IO_CHANNEL_STATUS: TX channel status structure.

Value: 2'b10 → High End TX channel capable of DMA descriptor fetch.

RX_FIFO_SIZE: Size of receiver data path FIFO.

Value: 04 → 16 * 32 bit words.

RX_IO_DATA_WIDTH: Width of the I/O bus receiver data path.

Value: 2'b10 → 32 bit.

RX_IO_CHANNEL_STATUS: RX channel status structure.

Value: 2'b10 → High End RX channel capable of DMA descriptor fetch.

REVISION: Revision of the DMA block.

Value: 0x01

TX_MAX_BURST_SIZE: Maximum value of defined length burst that the TX DMA_MAC logic will perform on the AHB bus to read data from the main memory.

2'b00 16 beat incrementing burst (INCR16)

2'b01 8 beat incrementing burst (INCR8)

2'b10 4 beat incrementing burst (INCR4)

2'b11 Single transfer only (SINGLE)

Descriptor fetch operation isn't affected by this field.

RX_MAX_BURST_SIZE: Maximum value of defined length burst that the RX DMA_MAC logic will perform on the AHB bus to write data to the main memory.

2'b00 16 beat incrementing burst (INCR16)

2'b01 8 beat incrementing burst (INCR8)

2'b10 4 beat incrementing burst (INCR4)

2'b11 Single transfer only (SINGLE)

Descriptor fetch operation isn't affected by this field.

LOOPB: Set to '1' to enable the DMA block loop_back mode. When set the RX DMA data are extracted by the TX FIFO and pushed in the RX one.

SRESET: DMA soft reset. Set to '1' to hold the whole DMA_MAC and MAC110 logic in reset condition. Write '0' to exit from the reset phase.

Note: After a HW reset, the DMA logic wakes up with the SRESET bit asserted ('1'), to keep all the DMA and MAC110 logic in the reset condition, until the SW is sure that clocks and the other MII signals, inputs to the MAC110 core, are stable.
When this condition is met, the SW is allowed to clear the SRESET bit (write '0') to start the normal operation.
Until the SRESET bit is set to '1', no operation is allowed on the DMA_MAC or MAC110 registers, except the SRESET bit clear.
This signal has no effect on the AHB interface so, when asserted runtime, the whole DMA will be reset only when the last AHB transfer, in the AHB master queue, has been completed.

6.2.4.2 Ethernet DMA, Interrupt Sources Enable Register

Mnemonic: DMA_INT_EN

Address: 0x3000_3004

Default value: 0x0000_0000

Bit	Field name	Access
31	TX_CURR_DONE_EN	RW
30 - 29	Reserved	RO
28	MAC110_INT_EN	RW
27 - 26	Reserved	RO
25	TX_MERR_INT_EN	RW
24	Reserved	RO
23	TX_DONE_EN	RW
22	TX_NEXT_EN	RW
21 - 20	Reserved	RO
19	TX_TO_EN	RW
18	TX_ENTRY_EN	RW
17	TX_FULL_EN	RW
16	TX_EMPTY_EN	RW
15	RX_CURR_DONE_EN	RW
14 - 10	Reserved	RO
09	RX_MERR_INT_EN	RW
08	Reserved	RO
07	RX_DONE_EN	RW
06	RX_NEXT_EN	RW
05	PACKET_LOST_EN	RW
04	Reserved	RO
03	RX_TO_EN	RW
02	RX_ENTRY_EN	RW
01	RX_FULL_EN	RW
00	RX_EMPTY_EN	RW

The DMA Interrupt enable register allows the various sources of interrupt to be individually enabled.

All the enabled sources will then be OR-ed to generate the global DMA interrupt.

Setting a bit in DMA_INT_EN allows the corresponding interrupt described in DMA_INT_STAT to influence the global DMA Interrupt.

If any bit position is set to '1' in **BOTH** DMA_INT_STAT and DMA_INT_EN, then the DMA Interrupt will be asserted.

Refer to the DMA_INT_STAT for a description of interrupt sources.

6.2.4.3 Ethernet DMA, Interrupt Status Register

Mnemonic: DMA_INT_STS

Address: 0x3000_3008

Default value: 0x0000_0000

Bit	Field name	Access
31	TX_CURR_DONE	RC
30 - 29	Reserved	RO
28	MAC110_INT	RC
27 - 26	Reserved	RO
25	TX_MERR_INT	RC
24	Reserved	RO
23	TX_DONE	RC
22	TX_NEXT	RC
21 - 20	Reserved	RO
19	TX_TO	RC
18	TX_ENTRY	RC
17	TX_FULL	RC
16	TX_EMPTY	RC
15	RX_CURR_DONE	RC
14 - 10	Reserved	RO
09	RX_MERR_INT	RC
08	Reserved	RO
07	RX_DONE	RC
06	RX_NEXT	RC
05	PACKET_LOST	RC
04	Reserved	RO
03	RX_TO	RC
02	RX_ENTRY	RC
01	RX_FULL	RC
00	RX_EMPTY	RC

DMA Interrupt Status Register reports the interrupt status of interrupts from the following sources:

DMA RX, DMA TX, MAC110.

All the register locations are read/clear (RC): they can be read, a write with '0' has no effect, while writing '1' reset the bit.

TX_CURR_DONE: Set when the TX master DMA has completed the CURRENT DMA transfers.

This bit differs from the TX_DONE because the TX_CURRENT_DONE will be set after a single DMA descriptor execution has been completed, the status register updated and the descriptor valid bit cleared, while the TX_DONE will be set only after all the descriptors in the descriptor chain have been fully executed.

Write '1' to clear flag.

MAC110_INT: Set when the external MAC110 device sets an interrupt request.

Write '1' to clear flag.

TX_MERR_INT: Set when the AHB master receives an error response from the selected slave and the internal arbiter is granting the TX FIFO.

Write '1' to clear flag.

TX_DONE: Set when the TX master DMA completes.

Write '1' to clear flag.

TX_NEXT: Set when a descriptor fetch operation loads an invalid entry.

Write '1' to clear flag.

TX_TO: Set when some data are stalled in the TX FIFO for too long.

Write '1' to clear flag.

TX_ENTRY: Set when the TX DMA is triggered by a number of empty TX FIFO entries bigger than the value set in the DMA_CNTL register.

Write '1' to clear flag.

TX_FULL: Set when the TX FIFO becomes full (< 4 byte entries available).

Write '1' to clear flag.

TX_EMPTY: Set when the TX FIFO becomes empty.

Write '1' to clear flag.

RX_CURR_DONE: Set when the RX master DMA has completed the CURRENT DMA transfers.

This bit differs from the RX_DONE because the RX_CURRENT_DONE will be set after a single DMA descriptor execution has been completed, the status register updated and the descriptor valid bit cleared, while the RX_DONE will be set only after all the descriptors in the descriptor chain have been fully executed.

Write '1' to clear flag.

RX_MERR_INT: Set when the AHB master receives an error response from the selected slave and the internal arbiter is granting the RX FIFO.

Write '1' to clear flag.

RX_DONE: Set when the RX master DMA completes.

Write '1' to clear flag.

RX_NEXT: Set when the descriptor fetch operation loads an invalid entry.

Write '1' to clear flag.

PACKET_LOST: Set by the RX wrapper when there is an incoming frame but the RX DMA logic cannot service it because:

the RX FIFO is not empty yet

or

the next descriptor fetch is still running

Write '1' to clear flag.

RX_TO: Set when some data are stalled in the RX FIFO for too long.

Write '1' to clear flag.

RX_ENTRY: Set when the RX DMA is triggered by a number of valid RX FIFO entries bigger than the value set in the DMA_CNTL register.

Write '1' to clear flag.

RX_FULL: Set when the RX FIFO becomes full and no more data can be accepted.

Write '1' to clear flag.

RX_EMPTY: Set when the RX FIFO becomes empty.

Write '1' to clear flag.

6.2.4.4 Ethernet DMA, RX Start Register

Mnemonic: RX_DMA_START

Address: 0x3000_3010

Default value: 0x0000_0000

Bit	Field name	Access
31 - 24	Reserved	RO
23 - 08	DFETCH_DLY	RW
07	CALL_SEEN	RW
06	RUNT_FRAME	RW
05	FILTER_FAIL	RW
04 - 03	Reserved	RO
02	START_FETCH	RS
01	Reserved	RO
00	DMA_EN	RC

DFETCH_DLY: Descriptor fetch delay. This field specifies, in a bus clock periods, the delay between two descriptor fetches, in the event that the descriptor in main memory is not valid.

When set to '0' it forces the DMA_MAC logic, in case of invalid descriptor, to wait for 2**16 system bus clocks before attempting a new fetch.

COLL_SEEN: When '1' the Late Collision seen condition, reported by the MAC110 in the RX packet status word, will make the received frame to be discharged by the DMA_MAC, without any report to the CPU.

When '0' no action will be taken by the DMA_MAC.

RUNT_FRAME: When '1' the Damaged Frame condition (e.g. normal collision, frame too short, etc.), reported by the MAC110 in the RX packet status word, will make the received frame to be discharged by the DMA_MAC, without any report to the CPU.

When '0' no action will be taken by the DMA_MAC.

FILTER_FAIL: When '1' the Address Filtering Failed condition, reported by the MAC110 during the RX packet transmission, will make the received frame to be discharged by the DMA_MAC, without any report to the CPU.

When '0' no action will be taken by the DMA_MAC.

If this bit is set the data of packets that don't match the address filtering process (inside the MAC core) are not moved to the memory reducing the AHB bus utilization.

START_FETCH: This bit is a Read/Set bit, that means it can be both read and written, but writing a '0' has no effect.

The SW has to set this bit to '1' when the RX DMA has to start fetching the first descriptor.

The DMA logic will reset to '0' this bit and set the DMA_EN to '1' as soon as the first fetch has been completed.

Note: Before starting the DMA, the DMA_NXT register has to be loaded with the starting address of the descriptor to be fetched.

DMA_EN: Read/Clear bit: a write with '1' reset to '0' the bit value, while a write with '0' has no effect.

This bit, set to '1' by the DMA after the first descriptor fetch, can be reset to '0' by the SW to force a DMA abort and stop as soon as possible the data transfer, before the DMA completion.

When all the DMA sequences complete normally, this bit is reset by the DMA_MAC logic and a new SW intervention is required to restart the DMA engine.

Note:

- The DMA_EN 0->1 transition resets the FIFO content and the RX interrupts (DMA_INT_STAT(15:0)).
- The DMA_EN 1->0 transition forces the DMA to close immediately the transfers toward AHB bus and MAC core. When the AHB transfer completes the DMA_INT_STAT.RX_DONE interrupt is set and the processor can reprogram and reactivate the RX logic.

6.2.4.5 Ethernet DMA, RX Control Register

Mnemonic: RX_DMA_CNTL

Address: 0x3000_3014

Default value: 0x0000_0000

Bit	Field name	Access
32 - 22	ADDR_WRAP	RW
21-17	ENTRY_TRIG	RW
16	Reserved	RO
15	DLY_EN	RW
14	NXT_EN	RW
13	Reserved	RO
12	CONT_EN	RW
11 - 00	DMA_XFERCOUNT	RW

ADDR_WRAP: Determines where the DMA address counter wraps by forcing the DMA address counter to retain the data originally written by the host in DMA_ADDR. As soon as the DMA has written the memory location prior to the value specified in ADD_WRAP the wrapping condition occurs.

This can be used to restrict the address counter within an address window (e.g. circular buffer).

The wrapping point MUST be 32 bit aligned, so the 10 bits of ADDR_WRAP are used to compare DMA address bits 11 to 2; if ADD_WRAP=DMA_ADDR(11:2) then a 4Kbyte buffer is defined.

ADDRWRAP is ignored unless WRAP_EN is set.

ENTRY_TRIG: Determines the amount of valid entries (in 32 BIT WORDs) required in the receive FIFO before the DMA is re-triggered.

If the value is set to 0, as soon as one valid entry is present, the DMA logic starts the data transfer.

DLY_EN: This bit enables (when '1') the DMA trigger delay feature: if a FIFO valid data resides in the FIFO more than a programmed period (DMA_TO), a time-out condition occurs that requires the DMA SM to empty the

FIFO even if the number of valid words doesn't exceed the threshold value.

NXT_EN: Next Descriptor Fetch Mode enable. Set to '1', this bit enables the next descriptor fetch mechanism.

Whenever a DMA transfer is completed, if this field is set, a new DMA descriptor is fetched. If this field is '0' then no descriptor is fetched and an interrupt is raised as normal.

Note when a descriptor is fetched RX_DMA_CTL is one of the registers updated

CONT_EN: Continuos Mode Enable. This bit enables the DMA to run in continuo mode. If set the DMA runs indefinitely ignoring DMA_ XFERCOUNT.

Note: "continuos mode" supersedes "next descriptor mode".

DMA_XFERCOUNT: Block size (in bytes) of DMA data transfer, up to 4 Kbytes.

If DMA_XFERCOUNT is set to '0', the DMA will transfer 4 Kbyte data.

Note: RX_DMA_CNTL.XFERCOUNT is used to provide an upper limit to the number of bytes the system can accept for each frame (it's usually equal to the dedicated frame buffer size in main memory).
When this limit is not exceeded, all the data received from the line, via the MAC110 core, are copied to the memory frame buffer, and the effective length of the transfer it's the real frame size.
On the other side, if the packet exceeds the XFERCOUNT value it will be truncated.
The XFERCOUNT value must be approximated to the next word aligned block size (i.e. if the desired transfer size is 123 bytes, then the programmed value should be 124).

6.2.4.6 Ethernet DMA, RX Base address Register

Mnemonic: RX_DMA_ADDR

Address: 0x3000_3018

Default value: 0x0000_0000

Bit	Field name	Access
31 - 02	DMA_ADDR	RW
01	FIX_ADDR	RW
00	WRAP_EN	RW

DMA_ADDR: Start address, 32 bit WORD ALIGNED, for master DMA transfer.

This register is read by the DMA SM only before starting the DMA operation and when the wrap condition is met, so further updates of this register will have unpredictable effects on the running DMA.

FIX_ADDR: Disables incrementing of DMA_ADDR: this means that all the DMA data transfer operation will be performed at the same AHB address, i.e. the DMA base address.

WRAP_EN: Enables wrap of the DMA transfer address to DMA_ADDR when the memory location, specified in ADDR_WRAP, is reached.

6.2.4.7 Ethernet DMA, RX Next Descriptor Address Register

Mnemonic: RX_DMA_NXT

Address: 0x3000_301C

Default value: 0x0000_0000

Bit	Field name	Access
31 - 02	DMA_DESCR_ADDR	RW
01	Reserved	RO
00	NPOL_EN	RW

DMA_DESCR_ADDR: When the DMA next descriptor fetch is enabled, this register points to the next descriptor starting address. The DMA descriptors are 32 bits, so the DMA_DESCR_ADDR MUST be 32 bit aligned.

This register allows different DMA descriptors to be located in different memory area, because part of the current DMA descriptors, provides information to point to the next one (descriptor chaining).

If the DMA descriptor fetch is not enabled, this register doesn't need to be updated.

NPOL_EN: Next Descriptor Polling Enable. When in 'Next Descriptor Fetch Mode', the descriptor fetch logic can load a not jet valid descriptor: if the NPOL is enabled ('1'), the logic is required to keep polling the DMA descriptor in main memory, until it's found to be valid.

Note: In case of not valid descriptor, DMA_MAC behavior will be different depending on the NPOL bit; we can have:

- NPOL=1 (polling enabled) -> the RX_NEXT bit will be set and a new descriptor fetch will be attempt after DFETCH_DLY clocks
- NPOL=0 (polling disabled) -> the RX_DONE bit will be set and the DMA_EN bit, in DMA_START register, will be cleared.

6.2.4.8 Ethernet DMA, RX Current Address Register

Mnemonic: RX_DMA_CADDR

Address: 0x3000_3020

Default value: 0x0000_0000

Bit	Field name	Access
31 - 00	DMA_CADDR	RO

DMA_CADDR: Current DMA address value, byte aligned.

The value of this register will change while the DMA is running, reflecting the value driven by the core on the AHB bus.

6.2.4.9 Ethernet DMA, RX Current Transfer Count Register

Mnemonic: RX_DMA_CXFER

Address: 0x3000_3024

Default value: 0x0000_0000

Bit	Field name	Access
31 - 12	Reserved	RO
11 - 00	DMA_CXFER	RO

DMA_CXFER: Current DMA transfer count value.

It's updated while the DMA is running, when one word data is moved from the MAC core to the DMA FIFO, reporting the number of bytes that can still be accepted.

6.2.4.10 Ethernet DMA, RX Time Out register

Mnemonic: RX_DMA_TO

Address: 0x3000_3028

Default value: 0x0000_0000

Bit	Field name	Access
31 - 16	Reserved	RO
15 - 00	TIME_OUT	RW

TIME_OUT: This value is used as initial value for the FIFO entry time_out counter (it's recommended not to use too low value, to avoid too frequent interrupts).

The time-out counter starts as soon as one valid entry is present in the FIFO and is reset every time a data is pop out of the FIFO.

The counter expires (FIFO time_out condition) if no FIFO data are pop for a period longer than the TIME_OUT register value; when this happens, depending on the control registers settings, an interrupt can be set.

6.2.4.11 Ethernet DMA, RX FIFO Status Register

Mnemonic: RX_DMA_FIFO

Address: 0x3000_342C

Default value: 0x0000_0000

Bit	Field name	Access
31-30	Reserved	RO
29-24	ENTRIES	RO
23-21	Reserved	RO
20-16	DMA_POINTER	RO
15-13	Reserved	RO
12-08	IO_POINTER	RO
07-04	Reserved	RO
03	DELAY_T	RO
02	ENTRY_T	RO
01	FULL	RO
00	EMPTY	RO

ENTRIES: full entries (in 32 bit words) in FIFO.

DMA_POINTER: FIFO DMA SM side pointer value.

IO_POINTER: FIFO IO side pointer value.

DELAY_T: Set to '1' when the DMA FIFO delay time_out is expired.

ENTRY_T: Set to '1' when the DMA FIFO entry trigger threshold has been reached.

FULL: Set to '1' when DMA FIFO is full.

EMPTY: Set to '1' when the DMA FIFO is empty.

6.2.4.12 Ethernet DMA, TX Start Register

Mnemonic: TX_DMA_START

Address: 0x3000_3030

Default value: 0x0000_0000

Bit	Field name	Access
31-24	Reserved	RO
23-08	DFETCH_DELAY	RW
07	ADD_CRC_DIS	RW
06	PADDING_DIS	RW
05	UNDERRUN	RW
04-03	Reserved	RO
02	START_FETCH	RS
01	Reserved	RO
00	DMA_EN	RC

DFETCH_DLY: Descriptor fetch delay. This field specifies, in a bus clock periods, the delay between two descriptor fetches, in the event that the descriptor in main memory is not valid.

When set to '0' it forces the DMA_MAC logic, in case of invalid descriptor, to wait for 2**16 system bus clocks before attempting a new fetch.

ADD_CRC_DIS: This bit drives the MAC110 input pin ADD_CRC_DISABLE to tell the MAC110 core not to add the CRC field at the end of the frame.

If its value is modified while the DMA is enabled, the results will be unpredictable.

PADDING_DIS: This bit drives the MAC110 input pin DISABLE_PADDING, to avoid the MAC110 add the padding bits for frames too short.

If its value is modified while the DMA is enabled, the results will be unpredictable.

UNDERRUN: When '1', the Under Run condition, reported by the MAC in the TX packet status word, will enable the DMA logic to retransmit the same packet to the MAC110 core, without reporting any error condition to the CPU.

START_FETCH: This bit is a Read/Set bit, that means it can be both read and written, but writing a '0' has no effect.

The SW has to set this bit to '1' when the TX DMA has to start fetching the first descriptor.

The DMA logic will reset to '0' this bit and set the DMA_EN to '1' as soon as the first fetch has been completed.

Note: Before starting the DMA, the DMA_NXT register has to be loaded with the starting address of the descriptor to be fetched.

DMA_EN: Read/Clear bit: a write with '1' reset to '0' the bit value, while a write with '0' has no effect.

This bit, set to '1' by the DMA after the first descriptor fetch, can be reset to '0' by the SW to force a DMA abort and stop as soon as possible the data transfer, before the DMA completion.

When all the DMA sequences complete normally, this bit is reset by the DMA_MAC logic and a new SW intervention is required to restart the DMA engine.

Note:

- The DMA_EN 0->1 transition resets the FIFO content and the TX interrupts (DMA_INT_STAT (31:16)).
- The DMA_EN 1->0 transition forces the DMA to close immediately the transfers toward AHB bus and MAC core. When the AHB transfer completes the DMA_INT_STAT.TX_DONE interrupt is set and the processor can reprogram and reactivate the TX logic.

6.2.4.13 Ethernet DMA, TX Control register

Mnemonic: TX_DMA_CNTL

Address: 0x3000_3034

Default value: 0x0000_0000

Bit	Field name	Access
31-22	ADDR_WRAP	RW
21-17	ENTRY_TRIG	RW
16	Reserved	RO
15	DLY_EN	RW
14	NXT_EN	RW
13	Reserved	RO
12	CONT_EN	RW
11-00	DMA_XFER_COUNT	RW

ADDR_WRAP: Determines where the DMA address counter wraps by forcing the DMA address counter to retain the data originally written by the host in DMA_ADDR. As soon as the DMA has read the memory location prior to the value specified in ADD_WRAP the wrapping condition occurs.

This can be used to restrict the address counter within an address window (e.g. circular buffer).

The wrapping point MUST be 32 bit aligned, so the 10 bits of ADDR_WRAP are used to compare DMA address bits 11 to 2; if ADD_WRAP=DMA_ADDR(11:2) then a 4Kbyte buffer is defined. ADDRWRAP is ignored unless WRAP_EN is set.

ENTRY_TRIG: Determines the amount of empty entries (in 32 BIT WORDs) required in the TX FIFO before the DMA is re-triggered.

If the value is set to 0, as soon as one empty entry is present, the DMA logic starts the data request.

DLY_EN: This bit enables (when '1') the DMA trigger delay feature: if a FIFO valid data resides in the FIFO more than a programmed period (DMA_TO), a time-out condition occurs and the related (TX_TO) interrupt will be set.

NXT_EN: Next Descriptor Fetch Mode enable. Set to '1', this bit enables the next descriptor fetch mechanism.

Whenever a DMA transfer is completed, if this field is set, a new DMA descriptor is fetched. If this field is '0' then no descriptor is fetched and an interrupt is raised as normal.

Note: when a descriptor is fetched TX_DMA_CTL is one of the registers updated.

CONT_EN: Continuos Mode Enable. This bit enables the DMA to run in continuos mode. If set the DMA runs indefinitely ignoring DMA_ XFERCOUNT.

Note "continuos mode" supersedes "next descriptor mode".

DMA_XFERCOUNT: Block size (in bytes) of DMA, maximum 4 Kbytes. If DMA_XFERCOUNT is set to '0', the DMA will transfer 4 Kbyte data.

6.2.4.14 Ethernet DMA, TX Base Address Register

Mnemonic: TX_DMA_ADDR

Address: 0x3000_3038

Default value: 0x0000_0000

Bit	Field name	Access
31-02	DMA_ADDR	RW
01	FIX_ADDR	RW
00	WRAP_EN	RW

DMA_ADDR: Start address, 32 bit WORD ALIGNED, for master DMA transfer.

This register is read by the DMA SM only before starting the DMA operation and when the wrap condition is met, so further updates of this register will have unpredictable effects on the running DMA.

FIX_ADDR: Disables incrementing of DMA_ADDR: this means that all the DMA data transfer operation will be performed at the same AHB address, i.e. the DMA base address.

WRAP_EN: Enables wrap of the DMA transfer address to DMA_ADDR when the memory location, specified in ADDR_WRAP, is reached.

6.2.4.15 Ethernet DMA, TX Next Descriptor Address Register

Mnemonic: TX_DMA_NXT

Address: 0x3000_303C

Default value: 0x0000_0000

Bit	Field name	Access
31-02	DMA_DESCR_ADDR	RW
01	Reserved	RO
00	NPOL_EN	RW

DMA_DESCR_ADDR: When the DMA next descriptor fetch is enabled, this register points to the next descriptor starting address. The DMA descriptors are 32 bits, so the DMA_DESCR_ADDR MUST be 32 bit aligned.

This register allows different DMA descriptors to be located in different memory area, because part of the current DMA descriptors, provides information to point to the next one (descriptor chaining).

If the DMA descriptor fetch is not enabled, this register doesn't need to be updated.

NPOL_EN: Next Descriptor Polling Enable. When in 'Next Descriptor Fetch Mode', the descriptor fetch logic can load a not jet valid descriptor: if the NPOL is enabled ('1'), the logic is required to keep polling the DMA descriptor in main memory, until it's found to be valid.

Note: In case of not valid descriptor, DMA_MAC behavior will be different depending on the NPOL bit; we can have:
NPOL=1 (polling enabled) -> the TX_NEXT bit will be set and a new descriptor fetch will be attempt after DFETCH_DLY clocks
NPOL=0 (polling disabled) -> the TX_DONE bit will be set and the DMA_EN bit, in DMA_START register, will be cleared.

6.2.4.16 Ethernet DMA, TX Current Address Register

Mnemonic: TX_DMA_CADDR

Address: 0x3000_3040

Default value: 0x0000_0000

Bit	Field name	Access
31-02	DMA_CADDR	RO

DMA_CADDR: Current DMA address value, byte aligned. The value of this register will change while the DMA is running, reflecting the value driven by the core on the AHB bus.

6.2.4.17 Ethernet DMA, TX Current Transfer Count Register

Mnemonic: TX_DMA_CXFER

Address: 0x3000_3044

Default value: 0x0000_0000

Bit	Field name	Access
31-12	Reserved	RO
11-00	DMA_CXFER	RO

DMA_CXFER: Current DMA transfer count value. It's updated while the DMA is running, when one data is moved from the main memory to the DMA FIFO, reflecting the number of bytes that must be still read.

6.2.4.18 Ethernet DMA, TX Time Out Register

Mnemonic: TX_DMA_TO

Address: 0x3000_3048

Default value: 0x0000_0000

Bit	Field name	Access
31-16	Reserved	RO
15-00	TIME_OUT	RW

TIME_OUT: This value is used as initial value for the FIFO entry time_out counter (it's recommended not to use too low value, to avoid too frequent interrupts).

This counter starts as soon as one valid entry is present in the FIFO and is reset every time a FIFO data is pop out of the FIFO.

The counter expires (FIFO time_out condition) if no FIFO data are pop for a period longer than the TIME_OUT register value; when this happens, depending on the control registers settings, an interrupt can be set.

6.2.4.19 Ethernet DMA, TX FIFO Status Register

Mnemonic: TX_DMA_FIFO

Address: 0x3000_304C

Default value: 0x0000_0000

Bit	Field name	Access
31-30	Reserved	RO
29-24	ENTRIES	RO
23-21	Reserved	RO
20-16	DMA_POINTER	RO
15-13	Reserved	RO
12-08	IO_POINTER	RO
07-04	Reserved	RO
03	DELAY_T	RO
02	ENTRY_T	RO
01	FULL	RO
00	EMPTY	RO

ENTRIES: free entries (in 32 bit words) in FIFO.

DMA_POINTER: FIFO DMA SM side pointer value.

IO_POINTER: FIFO IO side pointer value.

DELAY_T: Set to '1' when the DMA FIFO delay time_out is expired.

ENTRY_T: Set to '1' when the DMA FIFO entry trigger threshold has been reached.

FULL: Set to '1' when DMA FIFO is full.

EMPTY: Set to '1' when the DMA FIFO is empty.

6.2.4.20 MAC Control Register

Mnemonic: MAC_CNTL

Address: 0x3000_3400

Default value: 32'b00000000_00000100_00000000_00000000

Bit	Field name	Access
31	RA	RW
30	BLE	RW
29	Reserved	RO
28	HBD	RW
27	PS	RW
26 - 24	Reserved	RO
23	DRO	RW
22 - 21	OM	RW
20	FDM	RW
19	PAM	RW
18	PRM	RW
17	IF	RW
16	PBF	RW
15	HOFM	RW
14	Reserved	RO
13	HPFM	RW
12	LCC	RW
11	DBF	RW
10	DRT	RW
09	Reserved	RO
08	ASTP	RW
07 - 06	BOLMT	RW
05	DC	RW
04	Reserved	RO
03	TXE	RW
02	RXE	RW
01 - 00	Reserved	RO

The MAC Control Register establishes the RX and TX operating modes and controls for address filtering and packet filtering. Table 5 describes the bit fields of the register.

RA: Receive All. When set, all incoming packets will be received, regardless of the destination address. The address match checked according to Table 22, and is reported in Transmit Status.

BLE: Endian mode. When BLE is set, the MAC operates in the big Endian mode. When BLE is reset, the MAC operates in the little Endian mode. The Endian mode is only for the data buffers.

HBD: Heart Beat Disable. When set, the heartbeat signal quality (SQE) generator function is disabled. This bit should be set in the MII Mode.

PS: Port Select. When reset, the MII port is selected and when set, the SRL (ENDEC) port is selected for transmit/receive operations on the Ethernet side.

DRO: DRO-Disable Receive Own. When DRO is set, the MAC110 disables the reception of frames when the TXEN is asserted. The MAC110 will block the transmitted frame on the receive path. When DSO is reset, the MAC110 receives all the packets that are given by the PHY including those transmitted by the MAC100. This bit should be reset when the Full Duplex Mode bit is set or the Operating Mode is not set to 'Normal Mode'.

OM: OM-Loop-Back Operating Mode. This bit selects the Loop-Back operation modes for the MAC110. This setting is only for Full Duplex Mode.

OM Type	
2'b00	Normal. No feedback
2'b01	Internal. Through MII
2'b10	External. Through PHY
2'b11	Reserved.

In the Internal Loop-Back mode, the TX frame is received by the MII, and turned around back to the MAC110. In the External mode however, the TX frame is sent up to the PHY. The PHY will then turn that TX frame back to be received by the MAC110.

Note: that in the External mode the application has to set the PHY in Loop-Back mode by setting bit-14 in the register space of the PHY. (IEEE802.3 sec.22.2.4.1)

FDM: Full Duplex Mode. When Set, the MAC operates in a full-duplex mode where it can transmit and receive simultaneously. While in full-duplex mode: heartbeat check is disabled, heartbeat fail status should be ignored, and internal loop back is not allowed.

PAM: Pass All Multicast. When set, indicates that all the incoming frames with a multicast destination address (first bit in the destination address field is '1' are received. Incoming frames with physical address destinations are filtered only if the address matches with the MAC Address.

PRM: Promiscuous Mode. When set, indicates that any incoming valid frame is received regardless of its destination address.

IF: IF-Inverse filtering. When IF is set, Address Check block operates in the inverse filtering mode. This is valid only during perfect filtering mode.

PBF: Pass Bad Frames. When set, all incoming frames that passed the address filtering are received, including runt frames, collided frames, or truncated frames caused by Buffer underflow.

HPFM: Hash/Perfect Filtering Mode. When reset, the Address Check block does a perfect address filter of incoming frames according the address specified in the MAC Address register. When set, the Address Check block does imperfect address filtering of multicast incoming frames according to the hash table specified in the multicast Hash Table Register. If the Hash

Only (HO) is set, then physical addresses are imperfect filtered too. If Hash Only bit(HO) is reset, then physical addresses are perfect address filtered according to the MAC Address Register.

LCC: Late Collision Control. When set, enables the retransmission of the collided frame even after the collision period (late collision). When LC is reset, the MAC110 core disables the frame transmission on a late collision. In any case the Late Collision Status is appropriately updated in the Transmit Packet Status.

DBF: Disable Broadcast frames. When set, disables the reception of broadcast frames. When reset, forwards all the broadcast frames to the memory.

DRTY: Disable Retry. When set, the MAC will attempt only one transmission. When a collision is seen on the bus, the MAC will ignore the current frame and goes to the next frame and a retry error is reported in the Transmit Status. When DRTY is reset, the MAC will attempt 16 transmissions before signaling a retry error.

ASTP: Automatic Pad Stripping. When set the MAC will strip the pad field on all the incoming frames if the length field is less than 46 bytes. The FCS field will also be stripped since it is computed at the transmitting station based on the data and pad field characters, and will be invalid for a received frame that has had the pad characters stripped. Receive frames which have a length field of 46 bytes or greater will be passed to the host unmodified (FCS is not stripped). When reset, the MAC will pass all the incoming frames to the host unmodified.

BOLMT: BackOff Limit. The BOLMT bits allow the user to set its Back Off limit in a relaxed or aggressive mode. According to IEEE 802.3, the MAC110 has to wait for a random number [r] of Slot-Times** after it detects a collision, where:

$0 < r < 2K$

The number K is dependent on how many times the current Frame to be transmitted have been retried, as follows:

$$K = \min(n,10) \qquad \text{Eq. 1}$$

where n is the current number of retries.

If a frame has been retried for 3 times, then K = 3 and r= 8 Slot-Times maximum

If it has been retried for 12 times, then K = 10, and r = 1024 Slot-Times maximum.

A LFSR (linear feedback shift register) 20-bit counter is used to emulate a 20bit random number generator from which r is obtained. Once a collision is detected, the number of the current retry of the current frame is used to obtain K (eq.2). This value of K translates into the number of bits to use from the LFSR counter. If the value of k is 3, the MAC100 will take the value in the first 3 bits of the LFSR counter, and use it to count down till Zero on every Slot-Time. This will effectively causes the MAC110 to wait 8 Slot-Times.

To add more flexibility to the user the Value of the BOLMT will force the number of bits to be used from the LFSR counter to a predetermined value as in the table below.

BOLMT Value	# Bits Used from LFSR counter
2'b00	10
2'b01	8
2'b10	4
2'b11	1

Thus if the value of k = 10, then the MAC110 will look at the BOLMT if it is 00 then it will use the lower 10bits of the LFSR counter for the wait countdown. If BOLMT is 10 then it will only use the value in the first 4bits for the wait countdown, and so on…

**Slot-Time = 512 bit times.

DC: Deferral Check. When DC is set, the deferral check is enabled in the MAC. The MAC will abort the transmission attempt if it has deferred for more than 24,288 bit times. Deferring starts when the transmitter is ready to transmit, but is prevented from doing so because CRS is active. Defer time is not cumulative. If the transmitter defers for 10,000 bit times, then transmits, collides, backs off, and then has to defer again after completion of BackOff, the deferral timer resets to 0 and restarts.

When reset, the deferral check is disabled in the MAC and the MAC defers indefinitely.

TE: Transmitter Enable. When set, the MAC's transmitter is enabled and it will transmit frames from the buffer on to the cable.

When reset, the MAC's transmitter is disabled and will not transmit any frames.

RE: Receiver Enable. When set, the MAC's receiver is enabled and will receive frames from the MII interface.

When reset, the MAC's receiver is disabled and will not receive any frames from the MII interface.

6.2.4.21 MAC Address High Register

Mnemonic: MAC_ADDH

Address: 0x3000_3404

Default value: 0x0000_FFFF

Bit	Field name	Access
31 - 16	Reserved	RO
15 - 00	PADDR[47:32]	RW

The MAC Address Hi Register contains the upper 16 bits of the physical address of the MAC. The contents of this register are normally loaded from the EEPROM at power on through the EEPROM Controller.

PADDR[47:32]: Upper 16 bits (47:32) of the Physical Address of this MAC device.

6.2.4.22 MAC Address Low Register

Mnemonic: MAC_ADDL

Address: 0x3000_3408

Default value: 0xFFFF_FFFF

Bit	Field name	Access
31 - 00	PADDR[31:00]	RW

The MAC Address Low Register contains the lower 32 bits of the physical address of the MAC. The contents of this register are normally loaded from the EEPROM at power on through the EEPROM Controller.

PADDR[31:00]: Lower 32 bits (31:00) of the Physical Address of this MAC device.

6.2.4.23 MAC Multi Cast Hash Table High Register

Mnemonic: MAC_MCHTH

Address: 0x3000_340C

Default value: 0x0000_0000

Bit	Field name	Access
31 - 00	HTABLE[63:32]	RW

6.2.4.24 MAC Multi Cast Hash Table Low Register

Mnemonic: MAC_MCHTL

Address: 0x3000_3410

Default value: 0x0000_0000

Bit	Field name	Access
31 - 00	HTABLE[31:00]	RW

The 64-bit multicast table is used for group address filtering. For hash filtering, the contents of the destination address in the incoming frame is passed through the CRC logic and the upper 6 bits of the CRC register are used to index the contents of the Hash table. The most significant bit determines the register to be used (Hi/Low), while the other five bits determine the bit with in the register. A value of '00000' selects the bit 0 of the selected register and a value of '11111' selects the bit 31 of the selected register. If the corresponding bit is '1', then the multicast frame is accepted else it is rejected. If the Pass All Multicast is set, then all multi-cast frames are accepted regardless of the multi-cast hash values.

6.2.4.25 MII Address Register

Mnemonic: MAC_ADDR

Address: 0x3000_3414

Default value: 0x0000_0000

Bit	Field name	Access
31 - 16	Reserved	RO
15 – 11	PHY_ADD	RW
10 - 06	MII_REG	RW
05 - 02	Reserved	RO
01	MII_WR	RW
00	MII_BUSY	RW

The MII Address Register is used to control the Management cycles to the External PHY Controller chip.

PHY_ADD: Phy Address. These bits tell which of the 32 possible PHY devices are being accessed

MII_REG: MII Register. These bits select the desired MII register in the selected PHY device.

MII_WR: MII Write. Setting this bit tells the PHY that this will be a write operation using the MII data register. If this bit is not set, this will be a read operation, placing the data in the MII data register.

MII_BUSY: MII Busy. This bit should read a logic 0 before writing to the MII address and MII data registers. This bit must also be set to 0 during write to the MII address register. During a MII register access, this bit will be then set to signify that a read or write access is in progress. The MII data register should be kept valid until the MAC clears this bit during a PHY write operation. The MII data register is invalid until the MAC has cleared it during a PHY read operation. The MII address register should not be written to until this bit is cleared.

6.2.4.26 MII Data Register

Mnemonic: MAC_DATA

Address: 0x3000_3418

Default value: 0x0000_0000

Bit	Field name	Access
31 - 16	Reserved	RO
15 – 00	MII_DATA	RW

The MII Data Register contains the data to be written to the PHY register specified in the MII address register, or it contains the read data from the PHY register whose address is specified in the MII address register.

MII_DATA: This contains the 16-bit value read from the PHY after a MII read operation or the 16-bit data value to be written to the PHY before a MII write operation.

6.2.4.27 Flow Control register

Mnemonic: FCR

Address: 0x3000_341C

Default value: 0x0000_0000

Bit	Field name	Access
31 - 16	PTIME	RW
15 - 03	Reserved	RO
02	PCF	RW
01	FCE	RW
00	FCB	RW

This register is used to control the generation and reception of the Control (PAUSE Command) frames by the MAC's Flow control block. A write to register with busy bit set to '1' triggers the Flow Control block to generate a Control frame. The fields of the control frame are selected as specified in the 802.3x specification and PauseTime value from this register is used in the "Pause Time" field of the control frame. The Busy bit is set until the control frame is transferred onto the cable. The Host has to make sure that the Busy bit is cleared before writing the register. The Pass Control Frames bit indicates the MAC whether to pass the control frame to the Host or not and Flow Control Enable bit enables the receive portion of the Flow Control block.

PTIME: Pause Time. This field tells the value that is to be used in the PAUSE TIME field in the control frame.

PCF: Pass Control Frames. When set, the control frames are passed to the Host. The MAC110 core will decode the control frame (PAUSE), disables the transmitter for the specified amount of time. The Control Frame bit in the Receive Status (bit 25) is set and Transmitter Pause Mode signal indicates the current state of the MAC Transmitter.

When reset, the MAC110 core will decode the control frames but will not pass the frames to the Host. The Control Frame bit in the Receive Status (bit 25) will be set and the Transmitter Pause Mode signal gives the current status of the Transmitter, but the Packet Filter bit in the Receive Status is reset indication the application to flush the frame.

FCE: Flow Control Enable. When set, the MAC is enabled for operation and it will decode all the incoming frames for control frames. When the MAC receives a valid control frame (PAUSE command), it will disable the transmitter for the specified time.

When reset, the operation in the MAC is disabled and the MAC does not decode the frames for control frames.

Note: Flow Control is applicable when the MAC110 is set in Full Duplex Mode. In Half Duplex mode, this bit will enable using Backpressure to control flow of transmitted frames to the MAC110.

FCB: Flow Control Busy. This bit should read a logic 0 before writing to the Flow Control register. To initiate a PAUSE control frame the host must set this bit to '1'. During a transfer of Control Frame, this bit will continue to be set to signify that a frame transmission is in progress. After the completion of the transmission of the PAUSE control frame, the MAC will reset to '0'.

The Flow Control register should not be written to until this bit is cleared.

6.2.4.28 VLAN1 Tag Register

Mnemonic: VLAN1

Address: 0x3000_3420

Default value: 0x0000_FFFF

Bit	Field name	Access
31 - 16	Reserved	RO
15 - 00	VLAN1_TAG	RW

This register contains the VLAN Tag field to identify the VLAN1 frames. The MAC compares the 13th and 14th bytes of the incoming frame field and if a match is found, it sets the VLAN1 bit in the Rx-Status (bit 22) register. The legal length of the frame is increased from 1518 bytes to 1522 bytes.

VLAN1_TAG: This contains the VLAN Tag field to identify the VLAN1 frames. This field is compared with the 13th and 14th bytes of the incoming frames for VLAN1 frame detection.

6.2.4.29 VLAN2 Tag Register

Mnemonic: VLAN2

Address: 0x3000_3424

Default value: 0xFFFF_FFFF

Bit	Field name	Access
31 - 16	Reserved	RO
15 - 00	VLAN2_TAG	RW

This register contains the VLAN Tag field to identify the VLAN2 frames. The MAC compares the 13th and 14th bytes of the incoming frame field and if a match is found, it sets the VLAN2 bit in the Rx-Status register (bit 23). The legal length of the frame is increased from 1518 bytes to 1522 bytes.

VLAN2_TAG: This contains the VLAN Tag field to identify the VLAN2 frames. This field is compared to the 13th and 14th bytes of the incoming frames for VLAN2 frame detection.

6.2.4.30 MMC Control Register

Mnemonic: MMC_CTRL_REG

Address: 0x3000_3500

Default value: 0x0000_2F72

Bit	Field name	Access
31 – 14	Reserved	RO
13 – 03	MAX_FRM_SIZE	RW
02	RESET_ON_READ	RW
01	CNTR_ROLL_OVER	RW
00	CNTR_RESET	WO

This register establishes the operating mode of the management counters.

MAX_FRM_SIZE: These bits indicate the value of the Maximum Packet Size for the transmitted frames to be counted as long frames.

RESET_ON_READ: When set the counter will be reset to 0 after read.

CNTR_ROLL_OVER: When set, counters after reaching the maximum value start again from 0.

CNTR_RESET: When set, all counters will be reset to 0.

6.2.4.31 MMC Interrupt High Register

Mnemonic: MMC_INT_HI_REG

Address: 0x3000_33504

Default value: 0x0000_0000

Bit	Field name	Access
31 – 12	Reserved	RO
11	TX_EXC_DEFER_FRMS	RW
10	TX_CRS_ERROR_FRMS	RW
09	TX_ABORTED_FRMS	RW
08	TX_LATE_COL_FRMS	RW
07	TX_DEFERRED_FRMS	RW
06	TX_MUL_COL_FRMS	RW
05	TX_SINGLE_COL_FRMS	RW
04	TX_BAD_FRMS	RW
03	TX_FIFO_UND_FRMS	RW
02	TX_BROADCAST_FRMS	RW
01	TX_MULTICAST_FRMS	RW
00	TX_UNICAST_FRMS	RW

The MMC interrupt register maintains the interrupt generated due to the counters reaching half of their maximum value.

6.2.4.32 MMC Interrupt Low Register

Mnemonic: MMC_INT_LO_REG

Address: 0x3000_3508

Default value: 0x0000_0000

Bit	Field name	Access
31	TX_1024_TO_MAX_FRMS	RW
30	TX_512_TO_1023_FRMS	RW
29	TX_256_TO_511_FRMS	RW
28	TX_128_TO_255_FRMS	RW
27	TX_65_TO_127_FRMS	RW
26	TX_64_BYTES_FRMS	RW
25	TX_GOOD_TRANSM_BYTES	RW
24	TX_TRANSM_BYTES	RW
23	TX_CNTRL_FRMS	RW
22	TX_TRANSM_FRMS	RW
21	RX_ETH_FRMS	RW
20	RX_LEN_ERR_FRMS	RW
19	RX_DRIBBLE_ERR_FRMS	RW
18	RX_CRC_ERR_FRMS	RW
17	RX_LONG_FRMS	RW
16	RX_RUNT_FRMS	RW
15	RX_FIFO_ERR_FRMS	RW
14	RX_BROADCAST_FRMS	RW
13	RX_MULTICAST_FRMS	RW
12	RX_UNICAST_FRMS	RW
11	RX_1024_TO_MAX_FRMS	RW
10	RX_512_TO_1023_FRMS	RW
09	RX_256_TO_511_FRMS	RW
08	RX_128_TO_255_FRMS	RW
07	RX_65_TO_127_FRMS	RW
06	RX_64_BYTES_FRMS	RW
05	RX_GOOD_NUM_BYTES	RW
04	RX_NUM_BYTES	RW
03	RX_UNSUP_CNTRL_FRMS	RW

Bit	Field name	Access
02	RX_CNTRL_FRMS	RW
01	RX_GOOD_FRMS	RW
00	RX_NUM_FRMS	RW

The MMC interrupt register maintains the interrupt generated due to the counters reaching half of their maximum value.

6.2.4.33 MMC Interrupt Mask Registers

Mnemonic: MMC_INT_MSK_HI_REG, MMC_INT_MSK_LO_REG

Address: 0x3000_350C, 0x3000_3510

Default value: 0x0000_0000, 0x0000_0000

The MMC interrupt mask registers maintain the masks for the interrupt generated due to the counters reaching half of their maximum value. (MSB of the counter is set).

6.2.5 Programming the DMA MAC

6.2.5.1 The Ethernet Frame format

Figure 4. Ethernet Frame Format

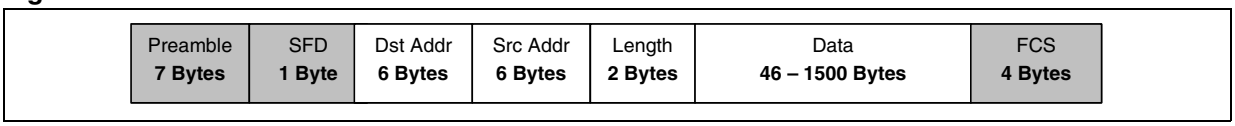
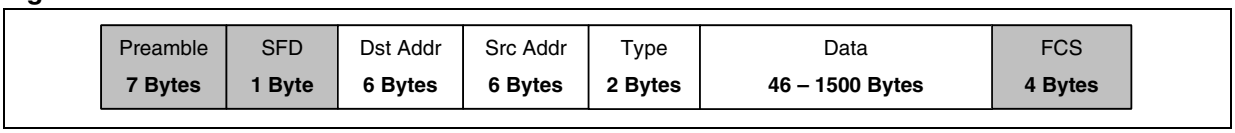


Figure 5. IEEE802.3 Frame Format



Automatically added by MAC layer

Input to MAC layer

The following is a description of what information the DMA MAC layer needs to be provided when sending frames on the LAN cable (See *Figure 4* and *Figure 5*)

- Destination Address (6 bytes):
  This field is the 48-bit standard format address of the device for whish this frame is intended.
- Source Address (6 bytes):
  This field is the 48-bit standard format address of the device which is sending this frame.
- Data (46-1500 Bytes):
  This field is the real content of the frame. It contains headers of the protocols above (like IEEE802.2 LLC/SNAP and TCP/IP) and data.
  Note that it must be at least 46 bytes long to have as a minimum of 64 bytes complete

frames on the cable. This is done to guarantee the collision condition detected before the packet transmission ends, even with a cable as long as the maximum cable length allowed by the IEEE specification.
However, since a zero-pad can be (depending on DIS_PADDING bit in TX_DMA_START register) automatically added by the DMA MAC when the minimum length constrain is not reached, the user doesn't needs to pay attention to it.

The following fields are automatically added by the DMA MAC layer before sending the frame (see *Figure 4* and *Figure 5*)

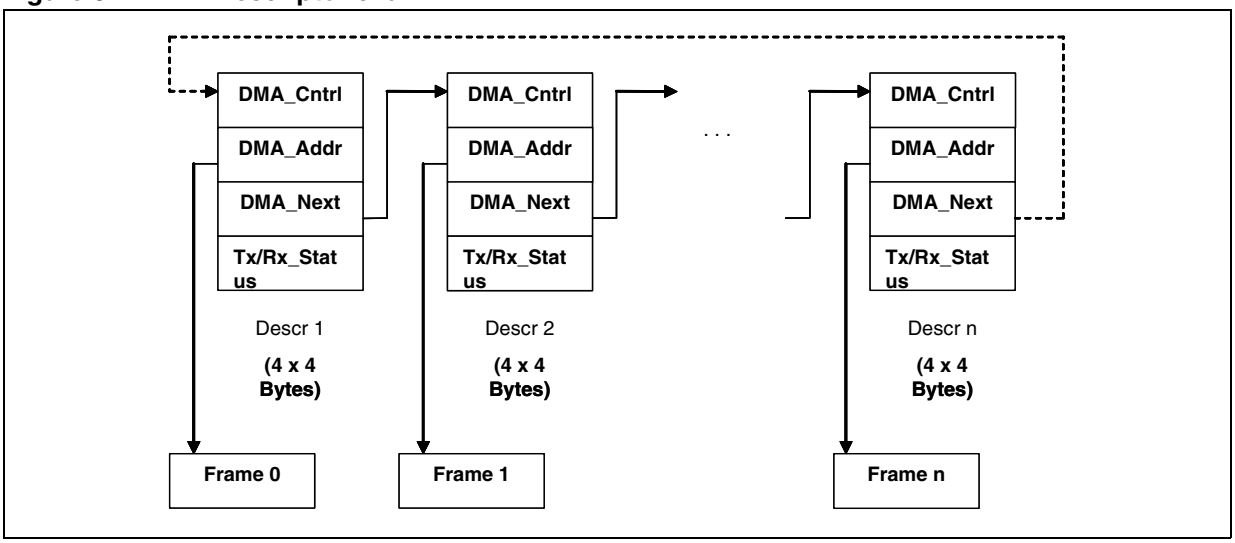
- Preamble (7 Bytes):
  This field is used to synchronize the receiver with the frame timing.
- Start Frame Delimiter (1 Byte):
  This field indicates the start of a frame.
- Frame Check Sequence (4 Bytes):
  This field represents the CRC32 value of all the data provided by the DMA MAC user. It is used by the receiver to check if the frame has been corrupted during the transmission on the line. The FCS field can be provided by the SW, together with the data: in this case the MAC logic will be requested, via the ADD_CRC_DIS bit in the TX_DMA_Start register, to do not generate it again.

On the other side, when the DMA MAC is receiving the frame from the LAN, it will download to the main memory the following field contents:

- Destination address (6 Bytes)
- Source Address (6 Bytes)
- Type or Length (2 Bytes)
- Data (46 - 1500 Bytes)
- FCS (4 Bytes)

6.2.5.2 The DMA Descriptor Chain

Figure 6. DMA Descriptor chain



The descriptor list is the mean the CPU and the DMA MAC use to communicate each other in order to transmit and receive frames on the cable. This list must be properly prepared before

initiating any transfer activity to or from the cable (see *Figure 6*). The descriptor is produced by the CPU and consumed by the DMA MAC.

A descriptor is a 16-bytes element which provides the DMA MAC with information about how to transmit or receive a single frame and how to report the transfer status back to the CPU.

A Descriptor can be stored in any main memory location with a 32 bit aligned address.

The first 3 words stored in a Descriptor are expected to be values of the 3 DMA MAC registers describing a DMA transfer (DMA_CNTL, DMA_ADDR and DMA_NEXT), while the fourth, related to the transmit/receive packet status, has the Descriptor Valid Bit as bit #16.

When the DMA MAC fetches a Descriptor it loads this three values into its own corresponding registers and checks the VALID bit value.

All the bits (except #16 - VALID bit) of the last word are to be used by the DMA MAC to report the transfer status. Its format should match the specification of the Transmit Packet Status and Receive Packet Status of the MAC110 core user manual with the minor changes reported in the following.

The following is the Descriptor format in C language notation:

```
{
   int DMA_CNTL;    //input-output
   int DMA_ADDR;    //input
   int DMA_NEXT;    //input
   int TxRx_STATUS; //output
};
```

6.2.5.3 The Descriptor Control Bits

The Descriptor keeps information about a single frame transfer and how to access to the next Descriptor. The following discussion is related to 3 bits of the Descriptor: The VALID bit, the NXT_EN bit and the NPOL_EN bit.

The Descriptor can be accessed simultaneously by the CPU and the DMA MAC. This concurrent access is synchronized by the VALID bit in the Receive/Transmit status register.

When the VALID bit is equal to 0 then the CPU is the owner of the Descriptor. Otherwise the owner is the DMA MAC. Since the Descriptor can be accessed in write mode by the owner at any time, race conditions are guaranteed to never happen.

The NXT_EN bit enables the fetch of the Next Descriptor. When the DMA MAC finds this bit set to 0 then the activity is considered to be completed as soon as the current Descriptor DMA transfers have been completed.

The NPOL_EN bit enables the DMA MAC to keep polling for a non valid Descriptor until its VALID bit become true (Set to 1). When the DMA MAC finds both the NPOL_EN bit and the VALD bit set to 0 then its activity is considered to be completed.

6.2.5.4 Transfer Status

The transfer status returned by the DMA MAC is based on the Tx/Rx Packet Status defined by the MAC110 core. Nevertheless the bits definition has been a slightly changed: the 16th bit is now the VALID bit for both the status fields, and the Frame To Long bit has been moved from the 16th to the 13th position.

Transfer Interrupts

The DMA MAC can interrupt the CPU with three different levels of information about transfer completion. The CPU can choose which interrupt needs to be enabled. They do not exclude each other though; they can be all three enabled at the same time.

The TX_CURR_DONE (RX_CURR_DONE) interrupt bit reports the CPU when a single Descriptor (i.e. one frame) has been completely treated by the DMA MAC and the CPU is again the owner (VALID bit is set to 0).

The TX_NEXT (RX_NEXT) interrupt bit is set when next descriptor fetch is enabled (NXT_EN set to 1 in the current Descriptor) but the next Descriptor is not valid (Valid bit is set to 0).

The TX_DONE (RX_DONE) interrupt bit is set when a whole DMA transfer is complete. This can happens either when the current is the last Descriptor in the chain (NXT_EN is set to 0) or when the next Descriptor is not valid yet (VALID bit set to 0) and the polling bit is disabled (NPOL_EN set to 0).

6.2.5.5 Frame Transmission (Tx)

When the CPU wants to transmit a set of frames on the cable, it needs to provide the DMA MAC with a Descriptor list. The CPU is expected to allocate a Descriptor for each frame it wants to send, to fill it with the DMA control information and the pointer to the frame and to link the Descriptor in the chain (see *Figure 6*). The frames will be sent on the cable in the same order they are found on the chain.

6.2.5.6 Open list approach

The simplest way to construct a Descriptor chain is the open list approach. Every Descriptor but the last one will have the DMA_NEXT field pointing to the next descriptor in the chain, the NXT_EN bit and the VALID bit on, the NPOL_EN bit on or off. The last Descriptor will e set in the same way except for the NXT_EN bit (off) and the DMA_NEXT field (NULL).

The CPU starts the DMA activity loading the physical location of the first Descriptor into the DMA_NEXT Register of the DMA MAC and then set the DMA_START resister enable bit to on.

The DMA MAC will then keep fetching the Descriptors one by one until it finds the NXT_EN bit set to off (last Descriptor in the chain). Every time it completes a descriptor (frame) it saves the transfer status into TxRx_STATUS, it turns the Descriptor VALID bit to off and rises the TX_CURR_DONE interrupt bit.

When the NXT_EN bit is found to be off, that means the DMA MAC has fetched the last Descriptor in the chain. When it completes also this Descriptor (the end of the DMA transfer) it raises both the TX_CURR_DONE and the TX_DONE interrupt bits.

Closed list approach

The approach above is easy since it doesn't require the DMA MAC and the CPU to synchronize their access to the descriptor chain. The problem is that requires the CPU to build the list every time it needs a transfer.

A faster way to operate is building a closed Descriptor list only the first time and using the VALID bit to mark the end of the transfer. Even more the polling facility could be used to save the CPU from the activity of programming the DMA_START register every time it needs to start the DMA Transfer. Instead, the DMA_START register will be activated only once and the DMA MAC will keep polling the invalid Descriptor, raising each time the TX_NEXT interrupt bit (if

enabled), until the CPU finally sets its VALID bit to on. Since the DMA transfer practically never ends, note that in this case the TX_DONE interrupt bit is never raised.

With this approach every Descriptor will have the DMA_NEXT field pointing to the next Descriptor in the chain (the last one will point the first one), the NXT_EN bit, the VALID bit and the NPOL_EN bit on.

The DMA MAC will keep fetching the Descriptor one by one until it finds one with its VALID bit set to 0. Every time the DMA MAC completes a Descriptor (frame) it saves the Transfer Status into the TxRx_STATUS, ot turns its VALID bit to off and raises the TX_CURR_DONE interrupt bit.

6.2.5.7 Frame Reception (Rx)

The frame reception process is something that needs to be activated at the beginning and kept always running. For this reason the closed Descriptor list (see above) is much more useful than the open list approach.

Again, with this approach every Descriptor will have the DMA_NEXT field pointing to the next Descriptor in the chain (the last one will point to the first one), the NXT_EN bit, the VALID bit and the NPOL_EN bit on.

The CPU starts the transfer activity loading the DMA Next register of the DMA MAC with the physical location of the first Descriptor and sets the DMA_START register enable bit to on. The DMA MAC will start fetching the Descriptors one by one, driven by the frame reception from the line. Every time the DMA MAC completes a Descriptor (frame) it saves the transfer status into the TxRx_STATUS, it turns its VALD bit to off and raises the RX_CURR_DONE interrupt bit.

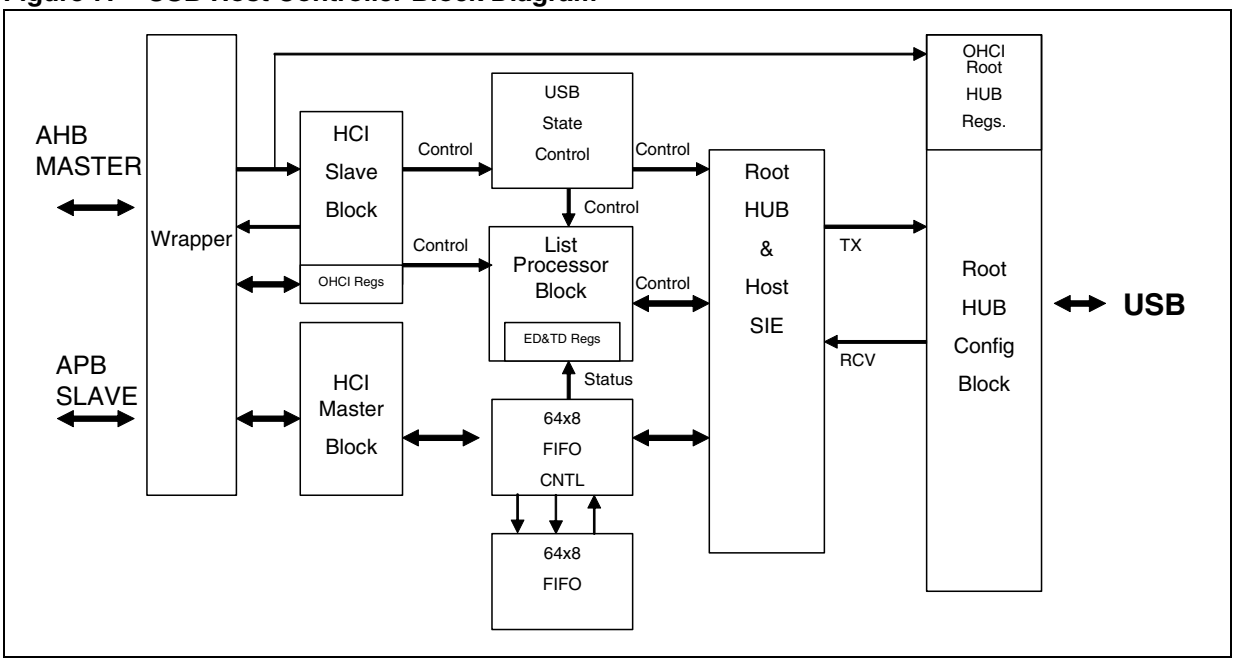
Eventually, if the DMA MAC will be faster then the CPU, it will wrap around the Descriptor chain finding a Descriptor still invalid.

Then the DMA MAC keeps polling the invalid Descriptor, raising each time the RX_NEXT interrupt bit (if enabled), until some Descriptors gets available (note that in this case some frame could be lost). In the meantime the CPU should consume the frames received and set the VALID bit to on of all the Descriptor released.

As soon as the DMA finds the Descriptor valid again, it will be able to complete the transfer and fetch the next Descriptor.

6.3 Full-Speed USB Host Controller

Figure 7. USB Host Controller Block Diagram



6.3.1 Overview

The USB interface integrated into the SPEAr Net device is an Full-Speed USB controller and is compliant with the USB1.1 standard and OpenHCI (Open Host Controller Interface) Rev.1.0 compatible.

SPEAr Net supports both, low and full speed USB devices.

The Open Host Controller Interface (OpenHCI) Specification for the Universal Serial Bus is a register-level description of a Host Controller for the Universal Serial Bus (USB) which in turn is described by the Universal Serial Bus Specification.

The purpose of OpenHCI is to accelerate the acceptance of USB in the marketplace by promoting the use of a common industry software/hardware interface. OpenHCI allows multiple Host Controller vendors to design and sell Host Controllers with a common software interface, freeing them from the burden of writing and distributing software drivers.

Figure 8. USB Focus Areas

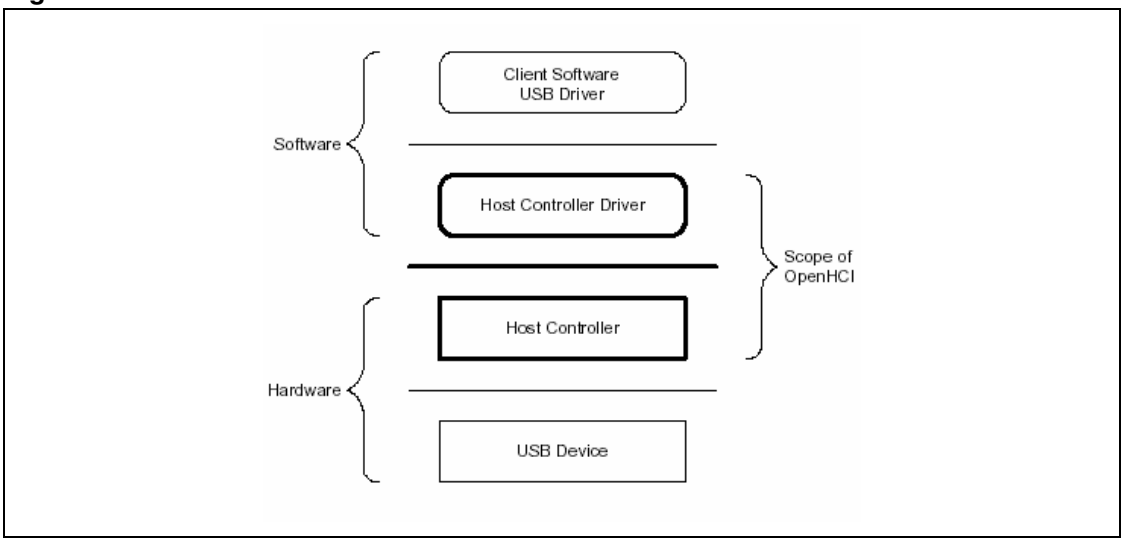


Figure 8 shows four main focus areas of a Universal Serial Bus (USB) system. These areas are the Client Software/USB Driver, Host Controller Driver (HCD), Host Controller (HC), and USB Device. The Client Software/USB Device and Host Controller Driver are implemented in software. The Host Controller and USB Device are implemented in hardware. OpenHCI specifies the interface between the Host Controller Driver and the Host Controller and the fundamental operation of each.

The Host Controller Driver and Host Controller work in tandem to transfer data between client software and a USB device. Data is translated from shared-memory data structures at the clientsoftware end to USB signal protocols at the USB device end, and vice-versa.

Figure 7 shows the Block Diagram of the protocols at the USB the SPEAr Net USB (HC) contains a set USB Host Controller.

The integrated USB Host Controller (HC) contains a set of on-chip operational registers which are mapped into the noncacheable portion of the system addressable space.

These registers are used by the Host Controller Driver (HCD).

According to the function of these registers, they are divided into four partitions, specifically for Control and Status, Memory Pointer, Frame Counter and Root Hub. All of the registers should be read and written as Dwords. Reserved bits may be allocated in future releases of this specification. To ensure interoperability, the Host Controller Driver that does not use a reserved field should not assume that the reserved field contains 0. Furthermore, the Host Controller Driver should always preserve the value(s) of the reserved field. When a R/W register is modified, the Host Controller Driver should first read the register, modify the bits desired, then write the register with the reserved bits still containing the read value. Alternatively, the Host Controller Driver can maintain an in-memory copy of previously written values that can be modified and then written to the Host Controller register. When a write to set/clear register is written, bits written to reserved fields should be 0.

6.3.2 Host Controller Management

The Host Controller (HC) is first managed through a set of Operational Registers. These registers exist in the Host Controller and are accessed using memory references via a noncached virtual pointer. All Host Controller Operational Registers start with the prefix ***Hc***.

The *HcHCCA* is filled in by software and points the Host Controller at the block of shared RAM called the Host Controller Communication Area (HCCA). All fields within the HCCA start with the prefix ***Hcca***.

6.3.3 Initialization of the HCI

There are a number of steps necessary for an OS to bring its Host Controller Driver to an operational state:

- Load Host Controller Driver and locate the HC
- Verify the HC and allocate system resources
- Take control of HC (support for an optional System Management Mode driver)
- Set up HC registers and HC Communications Area
- Begin sending SOF tokens on the USB

Note: Due to some devices on the USB that may take a long time to reset, it is desirable that the Host Controller Driver start-up process not transition to the USBRESET state if at all possible.

6.3.4 Operational States

The operational states of the Host Controller are defined by their effect on the USB:

- USBOPERATIONAL
- USBRESET
- USBRESUME
- USBSUSPEND

6.3.4.1 USBRESET

When the Host Controller enters this state, most of the operational registers are ignored by the Host Controller and need not contain any meaningful values; however, the contents of the registers (except Root Hub registers) are preserved by the HC. The obvious exception is that the Host Controller uses the *HcControl* register which contains the **HostControllerFunctionalState**.

While in this state, the Root Hub is being reset, which causes the Root Hub's downstream ports to be reset and possibly powered off. This state must be maintained for the minimum time specified in the USB Specification for the assertion of reset on the USB. Only the following interrupts are possible while the Host Controller is in the USBRESET state: **OwnershipChange**.

6.3.4.2 USBOPERATIONAL

This is the normal state of the HC. In this state, the Host Controller is generating SOF tokens on the USB and processing the various lists that are enabled in the *HcControl* register. This allows the clients of the Host Controller Driver, USBD and above, to communicate with devices on the USB. The Host Controller generates the first SOF token within one ms of the time that the USBOPERATIONAL state is entered (if the Host Controller Driver wants to know when this occurs, it may enable the **StartOfFrame** interrupt). All interrupts are possible in the USBOPERATIONAL state, except **ResumeDetected**.

6.3.4.3 USBSUSPEND

In this state, the Host Controller is not generating SOF tokens on the USB; nor is it processing any lists that may be enabled in the *HcControl* register. In fact, the Host Controller ignores most of the operational registers which need not contain any meaningful values; however, the Host Controller does preserve their values. While in this state, the Host Controller monitors the USB for resume signalling, and if detected, changes the state to USBRESUME. Because of this, there is a restriction on how the Host Controller Driver may modify the contents of *HcControl* while in the USBSUSPEND state: Host Controller Driver may only write to *HcControl* with the **HostControllerFunctionalState** field set to either USBRESET or USBRESUME (see exception).

OpenHCI - Open Host Controller Interface Specification for USB

After a certain length of time without SOF tokens, devices on the USB enter the suspend state. Normally, the Host Controller Driver must ensure that the Host Controller stays in this state for at least 5 ms and then exits this state to either the USBRESUME or the USBRESET state. An exception is when this state is entered due to a software reset and the previous state was not USBSUSPEND, in which case, if the Host Controller remains in the USBSUSPEND state for less than 1 ms, it may exit directly to USBOPERATIONAL (the timing of less than 1 ms ensures that no device on USB attempts to initiate resume signalling and thus the Host Controller does not attempt to modify *HcControl*). The only interrupts possible in the USBSUSPEND state are **ResumeDetected** (the Host Controller will have changed the **HostControllerFunctionalState** to the USBRESUME state) and **OwnershipChange**.

6.3.4.4 USBRESUME

While the Host Controller is in the USBRESUME state, it is asserting resume signalling on the USB; as a result, no tokens are generated and the Host Controller does not process any lists that may be enabled in the *HcControl* register. In fact, most of the operational registers are ignored and need not contain any meaningful values; however, the Host Controller does preserve their values. This state must be maintained for the minimum time specified in the USB Specification for the assertion of resume on the USB. The only interrupt possible in the USBRESUME state is **OwnershipChange**.

For more details please refer to the OpenHCI Interface Specification for USB in Chapter 8) Reference Documents

6.3.5 Operational Registers Mapping

Table 12. USB Host Controller Operational Register Map

Address	Register Name	Description
0x3000_2C00	HcRevision	Revision field Read only
0x3000_2C04	HcControl	USB Host Controller Operating mode
0x3000_2C08	HcCommandStatus	Used by the Host Controller to receive commands issused by the Host Controller Driver
0x3000_2C0C	HcInterruptStatus	Register provides status on various events that cause hardware interrupts
0x3000_2C10	HcInterruptEnable	Register is used to control which events generate a hardware interrupt
0x3000_2C14	HcInterruptDisable	To clear the corresponding bit in the HcInterruptEnable register
0x3000_2C18	HcHCCA	Register contains the physical address of the Host Controller Communication Area
0x3000_2C1C	HcPeriodCurrentED	Register contains the physical address of the current isochronous or Interrupt Endpoint Descriptor
0x3000_2C20	HcControlHeadED	Register contains the physical address of the first Endpoint Descriptor of the Control list
0x3000_2C24	HcControlCurrentED	Register contains the physical address of the current Endpont Descriptor of the control list
0x3000_2C28	HcBulkHeadED	Register contains the physical address of the first Endpoint Descriptor of the Bulk list
0x3000_2C2C	HcBulkCurrentED	Register contains the physical address of the current endpoint of the Bulk list. As the bulk list will be served in a round-robin fashion, the endpoints will be ordered according to their insertion to the list.
0x3000_2C30	HcDoneHead	Contains the physical address of the last completed transfer descriptor that was added to the Done queue.
0x3000_2C34	HcFmInterval	Register contains a 14-bit value which indicates the bit time interval in a frame, and a 15-bit value indicating the full speed maximum packet size that the host controller may carry out.
0x3000_2C38	HcFmRemaining	Register is a 14-bit down counter showing the bit time remaining in the current frame
0x3000_2C3C	HcFmNumber	Register is a 16-bit counter providein a reference among events happening in the host controller and the host controller driver
0x3000_2C40	HcPeriodicStart	Register has a 14-bit programmable value which determines when is the earliest time HC should start processing the periodic list.
0x3000_2C44	HcLSThreshold	Register contains an 11-bit value used by the host controller to determine whether to commit to the transfer of a maximum of 8-byte LS packet before EOF.
0x3000_2C48	HcRhDescriptorA	Register (first of 2) describes the characteristics of the root hub
0x3000_2C4C	HcRhDescriptorB	Register (second of 2) describes the characteristic of the root hub
0x3000_2C50	HcRhStatus	Register represents the Hub status field (lower word of Dword) and the Hub Status Change field (upper word of Dword)
0x3000_2C54	HcRhPortStatus[1: NDP]	Register [1:NDP] is used to control and report port events on a per-port basis.
…		“
0x3000_2C54+4*NDP	HcRhPortStatus[NDP]	“

6.3.6 Register description

All the registers are 32 bit wide.

6.3.6.1 HcRevision Register

Mnemonic:

Address: 0x3000_2C00

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31 - 08	REV	10	R	R	**Revision** This read-only field contains the BCD representation of the version of the HCI specification that is implemented by this HC. For example, a value of 11h corresponds to version 1.1. All of the HC implementations that are compliant with this specification will have a value of 10h.
07 - 00	reserved	I			

6.3.6.2 HcControl Register

Address: 0x3000_2C04

Bit	Field name	Root Hub Reset	Read/Write		Description
			HCD	HC	
31 - 11	reserved				
10	RWE	0b	RW	R	**RemoteWakeupEnable** This bit is used by HCD to enable or disable the remote wakeup feature upon the detection of upstream resume signaling. When this bit is set and the **ResumeDetected** bit in *HcInterruptStatus* is set, a remote wakeup is signaled to the host system. Setting this bit has no impact on the generation of hardware interrupt.
09	RWC	0b	RW	RW	**RemoteWakeupConnected** This bit indicates whether HC supports remote wakeup signaling. If remote wakeup is supported and used by the system it is the responsibility of system firmware to set this bit during POST. HC clears the bit upon a hardware reset but does not alter it upon a software reset. Remote wakeup signaling of the host system is host-bus-specific and is not described in this specification.

Bit	Field name	Root Hub Reset	Read/Write		Description
			HCD	HC	
08	IR	0b	RW	R	**InterruptRouting** This bit determines the routing of interrupts generated by events registered in *HcInterruptStatus*. If clear, all interrupts are routed to the normal host bus interrupt mechanism. If set, interrupts are routed to the System Management Interrupt. HCD clears this bit upon a hardware reset, but it does not alter this bit upon a software reset. HCD uses this bit as a tag to indicate the ownership of HC.
07 - 06	HCFS	00b	RW	RW	**HostControllerFunctionalState** for USB 00b: USBRESET 01b: USBRESUME 10b: USBOPERATIONAL 11b: USBSUSPEND A transition to USBOPERATIONAL from another state causes SOF generation to begin 1 ms later. HCD may determine whether HC has begun sending SOFs by reading the **StartofFrame** field of *HcInterruptStatus*. This field may be changed by HC only when in the USBSUSPEND state. HC may move from the USBSUSPEND state to the USBRESUME state after detecting the resume signaling from a downstream port. HC enters USBSUSPEND after a software reset, whereas it enters USBRESET after a hardware reset. The latter also resets the Root Hub and asserts subsequent reset signaling to downstream ports.
05	BLE	0b	RW	R	**BulkListEnable** This bit is set to enable the processing of the Bulk list in the next Frame. If cleared by HCD, processing of the Bulk list does not occur after the next SOF. HC checks this bit whenever it determines to process the list. When disabled, HCD may modify the list. If *HcBulkCurrentED* is pointing to an ED to be removed, HCD must advance the pointer by updating *HcBulkCurrentED* before re-enabling processing of the list.

<table>
<tr><th rowspan="2">Bit</th><th rowspan="2">Field name</th><th rowspan="2">Root Hub Reset</th><th colspan="2">Read/Write</th><th rowspan="2">Description</th></tr>
<tr><th>HCD</th><th>HC</th></tr>
<tr><td>04</td><td>CLE</td><td>0b</td><td>RW</td><td>R</td><td>ControlListEnable
This bit is set to enable the processing of the Control list in the next Frame. If cleared by HCD, processing of the Control list does not occur after the next SOF. HC must check this bit whenever it determines to process the list. When disabled, HCD may modify the list. If HcControlCurrentED is pointing to an ED to be removed, HCD must advance the pointer by updating HcControlCurrentED before re-enabling processing of the list.</td></tr>
<tr><td>03</td><td>IE</td><td>0b</td><td>RW</td><td>R</td><td>IsochronousEnable
This bit is used by HCD to enable/disable processing of isochronous EDs. While processing the periodic list in a Frame, HC checks the status of this bit when it finds an Isochronous ED (F=1). If set (enabled), HC continues processing the EDs. If cleared (disabled), HC halts processing of the periodic list (which now contains only isochronous EDs) and begins processing the Bulk/Control lists. Setting this bit is guaranteed to take effect in the next Frame (not the current Frame).</td></tr>
<tr><td>02</td><td>PLE</td><td>0b</td><td>RW</td><td>R</td><td>PeriodicListEnable
This bit is set to enable the processing of the periodic list in the next Frame. If cleared by HCD, processing of the periodic list does not occur after the next SOF. HC must check this bit before it starts processing the list.</td></tr>
<tr><td>01 - 00</td><td>CBSR</td><td>00b</td><td>RW</td><td>R</td><td>ControlBulkServiceRatio
This specifies the service ratio between Control and Bulk EDs.
Before processing any of the nonperiodic lists, HC must compare the ratio specified with its internal count on how many nonempty Control EDs have been processed, in determining whether to continue serving another Control ED or switching to Bulk EDs.
The internal count will be retained when crossing the frame boundary. In case of reset, HCD is responsible for restoring this value..
<table>
<tr><th>CBSR</th><th>No. of Control EDs Over Bulk EDs Served</th></tr>
<tr><td>0</td><td>1 : 1</td></tr>
<tr><td>1</td><td>2 : 1</td></tr>
<tr><td>2</td><td>3 : 1</td></tr>
<tr><td>3</td><td>4 : 1</td></tr>
</table>
</td></tr>
</table>

The HcControl register defines the opertating modes for the Host Controller. Most of the fields in this register are modified only by the Host Controller Driver, **except HostControllerFunctionalState and RemoteWakeupConnected**.

6.3.6.3 HcCommandStatus Register

Address: 0x3000_2C08

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31 - 18	reserved				
17 - 16	SOC	00b	R	RW	**SchedulingOverrunCount** These bits are incremented on each scheduling overrun error. It is initialized to 00b and wraps around at 11b. This will be incremented when a scheduling overrun is detected even if **SchedulingOverrun** in *HcInterruptStatus* has already been set. This is used by HCD to monitor any persistent scheduling problems.
15 - 04	reserved				
03	OCR	0b	RW	RW	**OwnershipChangeRequest** This bit is set by an OS HCD to request a change of control of the HC. When set HC will set the **OwnershipChange** field in *HcInterruptStatus*. After the changeover, this bit is cleared and remains so until the next request from OS HCD.
02	BLF	0b	RW	RW	**BulkListFilled** This bit is used to indicate whether there are any TDs on the Bulk list. It is set by HCD whenever it adds a TD to an ED in the Bulk list. When HC begins to process the head of the Bulk list, it checks BF. As long as **BulkListFilled** is 0, HC will not start processing the Bulk list. If **BulkListFilled** is 1, HC will start processing the Bulk list and will set BF to 0. If HC finds a TD on the list, then HC will set **BulkListFilled** to 1 causing the Bulk list processing to continue. If no TD is found on the Bulk list, and if HCD does not set **BulkListFilled**, then **BulkListFilled** will still be 0 when HC completes processing the Bulk list and Bulk list processing will stop.

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
01	CLF	0b	RW	RW	**ControlListFilled** This bit is used to indicate whether there are any TDs on the Control list. It is set by HCD whenever it adds a TD to an ED in the Control list. When HC begins to process the head of the Control list, it checks CLF. As long as **ControlListFilled** is 0, HC will not start processing the Control list. If CF is 1, HC will start processing the Control list and will set **ControlListFilled** to 0. If HC finds a TD on the list, then HC will set **ControlListFilled** to 1 causing the Control list processing to continue. If no TD is found on the Control list, and if the HCD does not set **ControlListFilled**, then **ControlListFilled** will still be 0 when HC completes processing the Control list and Control list processing will stop.
00	HCR	0b	RW	RW	**HostControllerReset** This bit is set by HCD to initiate a software reset of HC. Regardless of the functional state of HC, it moves to the USBSUSPEND state in which most of the operational registers are reset except those stated otherwise; e.g., the **InterruptRouting** field of *HcControl*, and no Host bus accesses are allowed. This bit is cleared by HC upon the completion of the reset operation. The reset operation must be completed within 10 s. This bit, when set, should not cause a reset to the Root Hub and no subsequent reset signaling should be asserted to its downstream ports.

The *HcCommandStatus* register is used by the Host Controller to receive commands issued by the Host Controller Driver, as well as reflecting the current status of the Host Controller. To the Host Controller Driver, it appears to be a "write to set" register. The Host Controller must ensure that bits written as '1' become set in the register while bits written as '0' remain unchanged in the register. The Host Controller Driver may issue multiple distinct commands to the Host Controller without concern for corrupting previously issued commands. The Host Controller Driver has normal read access to all bits.

The SchedulingOverrunCount field indicates the number of frames with which the Host Controller has detected the scheduling overrun error. This occurs when the Periodic list does not complete before EOF. When a scheduling overrun error is detected, the Host Controller increments the counter and sets the SchedulingOverrun field in the *HcInterruptStatus* register.

6.3.6.4 HcInterruptStatus Register

Address: 0x3000_2C0C

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31	0				
30	OC	0b	RW	RW	**OwnershipChange** This bit is set by HC when HCD sets the **OwnershipChangeRequest** field in *HcCommandStatus*. This event, when unmasked, will always generate an System Management Interrupt (SMI) immediately. This bit is tied to 0b when the SMI pin is not implemented.
29 - 07	reserved				
06	RHSC	0b	RW	RW	**RootHubStatusChange** This bit is set when the content of *HcRhStatus* or the content of any of *HcRhPortStatus*[**NumberofDownstreamPort**] has changed.
05	FNO	0b	RW	RW	**FrameNumberOverflow** This bit is set when the MSb of *HcFmNumber* (bit 15) changes value, from 0 to 1 or from 1 to 0, and after *HccaFrameNumber* has been updated.
04	UE	0b	RW	RW	**UnrecoverableError** This bit is set when HC detects a system error not related to USB. HC should not proceed with any processing not signalling before the system error has been corrected. HCD clears this bit after HC has been reset.
03	RD	0b	RW	RW	**ResumeDetected** This bit is set when HC detects that a device on the USB is asserting resume signalling. It is the transition from no resume signalling to resume signalling causing this bit to be set. This bit is not set when HCD sets the USBRESUME state.
02	SF	0b	RW	RW	**StartofFrame** This bit is set by HC at each start of a frame and after the update of *HccaFrameNumber*. HC also generates a SOF token at the same time.

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
01	WDH	0b	RW	RW	**WritebackDoneHead** This bit is set immediately after HC has written *HcDoneHead* to HccaDoneHead. Further updates of the HccaDoneHead will not occur until this bit has been cleared. HCD should only clear this bit after it has saved the content of *HccaDoneHead*.
00	SO	0b	RW	RW	**SchedulingOverrun** This bit is set when the USB schedule for the current Frame overruns and after the update of *HccaFrameNumber*. A scheduling overrun will also cause the **SchedulingOverrunCount** of HcCommandStatus to be incremented.

This register provides status on various events that cause hardware interrupts. When an event occurs, Host Controller sets the corresponding bit in this register. When a bit becomes set, a hardware interrupt is generated if the interrupt is enabled in the HcInterruptEnable register (see Chapter 6.3.6.5) and the MasterInterruptEnable bit is set. The Host Controller Driver may clear specific bits in this register by writing '1' to bit positions to be cleared. The Host Controller Driver may not set any of these bits. The Host Controller will never clear the bit.

6.3.6.5 HcInterruptEnable Register

Address: 0x3000_2C10

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31	MIE	0b	RW	R	A ‘0’ written to this field is ignored by HC. A '1' written to this field enables interrupt generation due to events specified in the other bits of this register. This is used by HCD as a Master Interrupt Enable.
30	OC	0b	RW	R	0 - Ignore 1 - Enable interrupt generation due to Ownership Change.
29 - 07	reserved				
06	RHSC	0b	RW	R	0 - Ignore 1 - Enable interrupt generation due to Root Hub Status Change.
05	FNO	0b	RW	R	0 - Ignore 1 - Enable interrupt generation due to Frame Number Overflow.
04	UE	0b	RW	R	0 - Ignore 1 - Enable interrupt generation due to Unrecoverable Error.

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
03	RD	0b	RW	R	0 - Ignore 1 - Enable interrupt generation due to Resume Detect.
02	SF	0b	RW	R	0 - Ignore 1 - Enable interrupt generation due to Start of Frame.
01	WDH	0b	RW	R	0 - Ignore 1 - Enable interrupt generation due to HcDoneHead Writeback.
00	SO	0b	RW	R	0 - Ignore 1 - Enable interrupt generation due to Scheduling Overrun.

Each enable bit in the *HcInterruptEnable* register corresponds to an associated interrupt bit in the *HcInterruptStatus* register. The *HcInterruptEnable* register is used to control which events generate a hardware interrupt. When a bit is set in the *HcInterruptStatus* register AND the corresponding bit in the *HcInterruptEnable* register is set AND the MasterInterruptEnable bit is set, then a hardware interrupt is requested on the host bus.

Writing a '1' to a bit in this register sets the corresponding bit, whereas writing a '0' to a bit in this register leaves the corresponding bit unchanged. On read, the current value of this register is returned.

6.3.6.6 HcInterruptDisable Register

Address: 0x3000_2C14

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31	MIE	0b	RW	R	A '0' written to this field is ignored by HC. A '1' written to this field disables interrupt generation due to events specified in the other bits of this register. This field is set after a hardware or software reset.
30	OC	0b	RW	R	0 - Ignore 1 - Disable interrupt generation due to Ownership Change.
29 - 07	reserved				
06	RHSC	0b	RW	R	0 - Ignore 1 - Disable interrupt generation due to Root Hub Status Change.
05	FNO	0b	RW	R	0 - Ignore 1 - Disable interrupt generation due to Frame Number Overflow.

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
04	UE	0b	RW	R	0 - Ignore 1 - Disable interrupt generation due to Unrecoverable Error.
03	RD	0b	RW	R	0 - Ignore 1 - Disable interrupt generation due to Resume Detect.
02	SF	0b	RW	R	0 - Ignore 1 - Disable interrupt generation due to Start of Frame.
01	WDH	0b	RW	R	0 - Ignore 1 - Disable interrupt generation due to HcDoneHead Writeback.
00	SO	0b	RW	R	0 - Ignore 1 - Disable interrupt generation due to Scheduling Overrun.

Each disable bit in the *HcInterruptDisable* register corresponds to an associated interrupt bit in the *HcInterruptStatus* register. The *HcInterruptDisable* register is coupled with the *HcInterruptEnable* register. Thus, writing a '1' to a bit in this register clears the corresponding bit in the *HcInterruptEnable* register, whereas writing a '0' to a bit in this register leaves the corresponding bit in the *HcInterruptEnable* register unchanged. On read, the current value of the *HcInterruptEnable* register is returned.

6.3.6.7 HcHCCA Register

Mnemonic:

Address: 0x3000_2C18

Default value:

Bit	Field name	Read/Write		Description
		HCD	HC	
31 - 08	HCCA	R/W	R	Base Address of the Host Controller Communication Area
07 - 00	0			

The *HcHCCA* register contains the physical address of the Host Controller Communication Area.

The Host Controller Driver determines the alignment restrictions by writing all 1s to *HcHCCA* and reading the content of *HcHCCA*. The alignment is evaluated by examining the number of zeroes in the lower order bits. The minimum alignment is 256 bytes; therefore, bits 0 through 7 must always return '0' when read. Detailed description can be found in Chapter 4. This area is used to hold the control structures and the Interrupt table that are accessed by both the Host Controller and the Host Controller Driver.

6.3.6.8 HcPeriodCurrentED Register

Mnemonic:

Address: 0x3000_2C1C

Default value:

Bit	Field name	Read/Write		Description
		HCD	HC	
31 – 04	PCED	R	RW	PeriodCurrentED This is used by HC to point to the head of one of the Periodic of lists which will be processed in the current Frame. The content of this register is updated by HC after a periodic ED has been processed. HCD may read the content in determining which ED is currently being processed at the time of reading.
03 – 00	0			

The *HcPeriodCurrentED* register contains the physical address of the current Isochronous or Interrupt Endpoint Descriptor.

6.3.6.9 HcControlHeadED Register

Mnemonic:

Address: 0x3000_2C20

Default value:

Bit	Field name	Read/Write		Description
		HCD	HC	
31 – 04	CCED	R	RW	**ControlCurrentED** This pointer is advanced to the next ED after serving the present one. HC will continue processing the list from where it left off in the last Frame. When it reaches the end of the Control list, HC checks the **ControlListFilled** of in *HcCommandStatus*. If set, it copies the content of *HcControlHeadED* to *HcControlCurrentED* and clears the bit. If not set, it does nothing. HCD is allowed to modify this register only when the **ControlListEnable** of *HcControl* is cleared. When set, HCD only reads the instantaneous value of this register. Initially, this is set to zero to indicate the end of the Control list.
03 – 00	0			

The *HcControlCurrentED* register contains the physical address of the current Endpoint Descriptor of the Control list.

6.3.6.10 HcBulkHeadED Register

Mnemonic:

Address: 0x3000_2C28

Default value:

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31 – 04	BHED	0h	R/W	R	BulkHeadED HC traverses the Bulk list starting with the HcBulkHeadED pointer. The content is loaded from HCCA during the initialization of HC.
03 – 00	0				

The *HcBulkHeadtED* register contains the physical address of the first Endpoint Descriptor of the Bulk list.

6.3.6.11 HcBulkCurrentED Register

Address: 0x3000_2C2C

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31 – 04	BCED	0h	R/W	R/W	**BulkCurrentED** This is advanced to the next ED after the HC has served thepresent one. HC continues processing the list from where it left off in the last Frame. When it reaches the end of the Bulk list, HC checks the **ControlListFilled** of HcControl. If set, it copies the content of *HcBulkHeadED* to *HcBulkCurrentED* and clears the bit. If it is not set, it does nothing. HCD is only allowed to modify this register when the **BulkListEnable** of *HcControl* is cleared. When set, the HCD only reads the instantaneous value of this register. This is initially set to zero to indicate the end of the Bulk list.
03 – 00	0				

The *HcBulkHeadtED* register contains the physical address of the current endpoint of the Bulk list. As the Bulk list will be served in a round-robin fashion, the endpoints will be ordered according to their insertion of the list.

6.3.6.12 HcDoneHead Register

Address: 0x3000_2C30

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31 – 04	DH	0h	R	R/W	**DoneHead** When a TD is completed, HC writes the content of *HcDoneHead* to the NextTD field of the TD. HC then overwrites the content of *HcDoneHead* with the address of this TD. This is set to zero whenever HC writes the content of this register to HCCA. It also sets the **WritebackDoneHead** of *HcInterruptStatus*
03 – 00	0				

The *HcDoneHead* register contains the physical address of the last completed Transfer Descriptor that was added to the Done queue. In normal operation, the Host Controller Driver should not need to read the register as its content is periodically written to the HCCA.

6.3.6.13 HcFmInterval Register

Address: 0x3000_2C34

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31	FIT	0b	RW	R	**Frame Interval Toggle** HCD toggles this bit whenever it loads a new value to FrameInterval
30 - 16	FSMPS	TBD	RW	R	**FSLargestDataPacket** This field specifies a value which is loaded into the Largest Data Packet Counter at the beginning of each frame. The counter value represents the largest amount of data in bits which can be sent or received by the HC in a single transaction at any given time without causing scheduling overrun. The field value is calculated by the HCD.
15 – 14	Reserved				
13 - 00	FI	2EDFh	RW	R	**FrameInterval** This specifies the interval between two consecutive SOFs in bit times. The nominal value is set to be 11,999. HCD should store the current value of this field before resetting HC. By setting the **HostControllerReset** field of *HcCommandStatus* as this will cause the HC to reset this field to its nominal value. HCD may choose to restore the stored value upon the completion of the Reset sequence.

The *HcFmInterval* register contains a 14-bit value which indicates the bit time interval in a Frame, (i.e., between two consecutive SOFs), and a 15-bit value indicating the Full Speed maximum packet size that the Host Controller may transmit or receive without causing scheduling overrun. The Host Controller Driver may carry out minor adjustment on the **FrameInterval** by writing a new value over the present one at each SOF. This provides the programmability necessary for the Host Controller to synchronize with an external clocking resource and to adjust any unknown local clock offset.

6.3.6.14 HcFmRemaining Register

Address: 0x3000_2C38

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31	FRT	0b	R	RW	**FrameRemainingToggle** This bit is loaded from the **FrameIntervalToggle** field of *HcFmInterval* whenever **FrameRemaining** reaches 0. This bit is used by HCD for the synchronization between **FrameInterval** and **FrameRemaining**.
30 - 14	reserved				.
FR		0h	R	RW	**FrameRemaining** This counter is decremented at each bit time. When it reaches zero, it is reset by loading the **FrameInterval** value specified in *HcFmInterval* at the next bit time boundary. When entering the USBOPERATIONAL state, HC re-loads the content with the **FrameInterval** of *HcFmInterval* and uses the updated value from the next SOF.

The HcFmRemaining register is a 14-bit down counter showing the bit time remaining in the current Frame.

6.3.6.15 HcFmNumber Register

Address: 0x3000_2C3C

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31 - 16	reserved				
15 - 00	FN	0h	R	RW	**FrameNumber** This is incremented when *HcFmRemaining* is re-loaded. It will be rolled over to 0h after ffffh. When entering the USBOPERATIONAL state, this will be incremented automatically. The content will be written to HCCA after HC has incremented the **FrameNumber** at each frame boundary and sent a SOF but before HC reads the first ED in that Frame. After writing to HCCA, HC will set the **StartofFrame** in *HcInterruptStatus.*

The *HcFmNumber* register is a 16-bit counter. It provides a timing reference among events happening in the Host Controller and the Host Controller Driver. The Host Controller Driver may use the 16-bit value specified in this register and generate a 32-bit frame number without requiring frequent access to the register.

6.3.6.16 HcPeriodicStart Register

Address: 0x3000_2C40

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31 - 16	reserved				
15 - 00	PS	0h	RW	R	**PeriodicStart** After a hardware reset, this field is cleared. This is then set by HCD during the HC initialization. The value is calculated roughly as 10% off from *HcFmInterval*.. A typical value will be 3E67h. When *HcFmRemaining* reaches the value specified, processing of the periodic lists will have priority over Control/Bulk processing. HC will therefore start processing the Interrupt list after completing the current Control or Bulk transaction that is in progress.

The *HcPeriodicStart* register has a 14-bit programmable value which determines when is the earliest time HC should start processing the periodic list.

6.3.6.17 HcLSThreshold Register

Address: 0x3000_2C44

Bit	Field name	Reset	Read/Write		Description
			HCD	HC	
31 - 12	Reserved				
11 - 00	LST	0628h	RW	R	**LSThreshold** This field contains a value which is compared to the **FrameRemaining** field prior to initiating a Low Speed transaction. The transaction is started only if **FrameRemaining** ≥ this field. The value is calculated by HCD with the consideration of transmission and setup overhead.

6.3.6.18 HcRhDescriptorA Register

Address: 0x3000_2C48

Bit	Field name	Power on Reset	Read/Write		Description
			HCD	HC	
31 - 24	POTPGT	IS	RW	R	**PowerOnToPowerGoodTime** This byte specifies the duration HCD has to wait before accessing a powered-on port of the Root Hub. It is implementation-specific. The unit of time is 2 ms. The duration is calculated as **POTPGT** * 2 ms.
23 - 13	Reserved				
12	NOCP	IS	RW	R	**NoOverCurrentProtection** This bit describes how the overcurrent status for the Root Hub ports are reported. When this bit is cleared, the **OverCurrentProtectionMode** field specifies global or per-port reporting. 0: Over-current status is reported collectively for all downstream ports 1: No overcurrent protection supported
11	OCPM	IS	RW	R	**OverCurrentProtectionMode** This bit describes how the overcurrent status for the Root Hub ports are reported. At reset, this fields should reflect the same mode as **PowerSwitchingMode**. This field is valid only if the **NoOverCurrentProtection** field is cleared. 0: over-current status is reported collectively for all downstream ports 1: over-current status is reported on a per-port basis
10	DT	0b	R	R	**DeviceType** This bit specifies that the Root Hub is not a compound device. The Root Hub is not permitted to be a compound device. This field should always read/write 0.
09	NPS	IS	RW	R	**NoPowerSwitching** These bits are used to specify whether power switching is supported or port are always powered. It is implementationspecific. When this bit is cleared, the **PowerSwitchingMode** specifies global or per-port switching. 0: Ports are power switched 1: Ports are always powered on when the HC is powered on

Bit	Field name	Power on Reset	Read/Write		Description
			HCD	HC	
08	PSM	IS	RW	R	**PowerSwitchingMode** This bit is used to specify how the power switching of the Root Hub ports is controlled. It is implementation-specific. This field is only valid if the **NoPowerSwitching** field is cleared. 0: all ports are powered at the same time. 1: each port is powered individually. This mode allows port power to be controlled by either the global switch or perport switching. If the **PortPowerControlMask** bit is set, the port responds only to port power commands (**Set/ ClearPortPower**). If the port mask is cleared, then the port is controlled only by the global power switch (**Set/ClearGlobalPower**).
07 - 00	NDP	IS	R	R	**NumberDownstreamPorts** These bits specify the number of downstream ports supported by the Root Hub. It is implementation-specific. The minimum number of ports is 1. The maximum number of ports supported by OpenHCI is 15.

The *HcRhDescriptorA* register is the first register of two describing the characteristics of the Root Hub. Reset values are implementation-specific. The descriptor length (11), descriptor type (TBD), and hub controller current (0) fields of the hub Class Descriptor are emulated by the HCD. All other fields are located in the *HcRhDescriptorA* and *HcRhDescriptorB* registers.

6.3.6.19 HcRhDescriptorB Register

Address: 0x3000_2C4C

Bit	Field name	Power on Reset	Read/Write		Description
			HCD	HC	
31 - 16	PPCM	IS	RW	R	PortPowerControlMask Each bit indicates if a port is affected by a global power control command when **PowerSwitchingMode** is set. When set, the port's power state is only affected by per-port power control (**Set/ClearPortPower**). When cleared, the port is controlled by the global power switch (**Set/ClearGlobalPower**). If the device is configured to global switching mode (**PowerSwitchingMode**=0), this field is not valid. bit 0: Reserved bit 1: Ganged-power mask on Port #1 bit 2: Ganged-power mask on Port #2 ... bit15: Ganged-power mask on Port #15
15 - 00	DR	IS	RW	R	DeviceRemovable Each bit is dedicated to a port of the Root Hub. When cleared, the attached device is removable. When set, the attached device is not removable. bit 0: Reserved bit 1: Device attached to Port #1 bit 2: Device attached to Port #2 ... bit15: Device attached to Port #15

The *HcRhDescriptorB* register is the second register of two describing the characteristics of the Root Hub. These fields are written during initialization to correspond with the system implementation. Reset values are implementation-specific.

6.3.6.20 HcRhStatus Register

Address: 0x3000_2C50

Bit	Field name	Root Hub Reset	Read/Write		Description
			HCD	HC	
31	CRWE	-	W	R	ClearRemoteWakeupEnable Writing a '1' clears **DeviceRemoveWakeupEnable**. Writing a '0' has no effect.
30 - 18	Reserved				
17	OCIC	0b	RW	RW	OverCurrentIndicatorChange This bit is set by hardware when a change has occurred to the OCI field of this register. The HCD clears this bit by writing a ‘1’. Writing a ‘0’ has no effect.
16	LPSC	0b	RW	R	(read) **LocalPowerStatusChange** The Root Hub does not support the local power status feature; thus, this bit is always read as ‘0’. (write) **SetGlobalPower** In global power mode (**PowerSwitchingMode**=0), This bit is written to ‘1’ to turn on power to all ports (clear **PortPowerStatus**). In per-port power mode, it sets **PortPowerStatus** only on ports whose **PortPowerControlMask** bit is not set. Writing a ‘0’ has no effect.
15	DRWE	0b	RW	R	(read) **DeviceRemoteWakeupEnable** This bit enables a **ConnectStatusChange** bit as a resume event, causing a USBSUSPEND to USBRESUME state transition and setting the **ResumeDetected** interrupt. 0 = **ConnectStatusChange** is not a remote wakeup event. 1 = **ConnectStatusChange** is a remote wakeup event. (write) **SetRemoteWakeupEnable** Writing a '1' sets **DeviceRemoveWakeupEnable**. Writing a '0' has no effect.
14 - 02	reserved				

Bit	Field name	Root Hub Reset	Read/Write		Description
			HCD	HC	
01	OCI	0b	R	RW	OverCurrentIndicator This bit reports overcurrent conditions when the global reporting is implemented. When set, an overcurrent condition exists. When cleared, all power operations are normal. If per-port overcurrent protection is implemented this bit is always '0'
00	LPS	0b	RW	R	(read) **LocalPowerStatus** The Root Hub does not support the local power status feature; thus, this bit is always read as '0'. (write) **ClearGlobalPower** In global power mode (**PowerSwitchingMode**=0), This bit is written to '1' to turn off power to all ports (clear **PortPowerStatus**). In per-port power mode, it clears **PortPowerStatus** only on ports whose **PortPowerControlMask** bit is not set. Writing a '0' has no effect.

The *HcRhStatus* register is divided into two parts. The lower word of a Dword represents the **Hub Status** field and the upper word represents the **Hub Status Change** field. Reserved bits should always be written '0'.

6.3.6.21 HcRhPortStatus[1:NDP] Register

Address: 0x3000_2C54

Bit	Field name	Root Hub Reset	Read/Write		Description
			HCD	HC	
31 - 21	Reserved				
20	PRSC	0b	RW	RW	**PortResetStatusChange** This bit is set at the end of the 10-ms port reset signal. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. 0 = port reset is not complete 1 = port reset is complete
19	OCIC	0b	RW	RW	**PortOverCurrentIndicatorChange** This bit is valid only if overcurrent conditions are reported on a per-port basis. This bit is set when Root Hub changes the **PortOverCurrentIndicator** bit. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. 0 = no change in **PortOverCurrentIndicator** 1 = **PortOverCurrentIndicator** has changed

Bit	Field name	Root Hub Reset	Read/Write		Description
			HCD	HC	
18	PSSC	0b	RW	RW	**PortSuspendStatusChange** This bit is set when the full resume sequence has been completed. This sequence includes the 20-s resume pulse, LS EOP, and 3-ms resychronization delay. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. This bit is also cleared when **ResetStatusChange** is set. 0 = resume is not completed 1 = resume completed
17	PESC	0b	RW	RW	**PortEnableStatusChange** This bit is set when hardware events cause the **PortEnableStatus** bit to be cleared. Changes from HCD writes do not set this bit. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. 0 = no change in **PortEnableStatus** 1 = change in **PortEnableStatus**
16	CSC				**ConnectStatusChange** This bit is set whenever a connect or disconnect event occurs. The HCD writes a '1' to clear this bit. Writing a '0' has no effect. If **CurrentConnectStatus** is cleared when a **SetPortReset, SetPortEnable**, or **SetPortSuspend** write occurs, this bit is set to force the driver to re-evaluate the connection status since these writes should not occur if the port is disconnected. 0 = no change in **CurrentConnectStatus** 1 = change in **CurrentConnectStatus** Note: If the **DeviceRemovable[NDP]** bit is set, this bit is set only after a Root Hub reset to inform the system that the device is attached.
15 - 10	reserved				
09	LSDA	Xb	RW	RW	(read) **LowSpeedDeviceAttached** This bit indicates the speed of the device attached to this port. When set, a Low Speed device is attached to this port. When clear, a Full Speed device is attached to this port. This field is valid only when the **CurrentConnectStatus** is set. 0 = full speed device attached 1 = low speed device attached (write) **ClearPortPower** The HCD clears the **PortPowerStatus** bit by writing a '1' to this bit. Writing a '0' has no effect.

Bit	Field name	Root Hub Reset	Read/Write		Description
			HCD	HC	
08	PPS	0b	RW	RW	(read) **PortPowerStatus** This bit reflects the port's power status, regardless of the type of power switching implemented. This bit is cleared if an overcurrent condition is detected. HCD sets this bit by writing **SetPortPower** or **SetGlobalPower**. HCD clears this bit by writing **ClearPortPower** or **ClearGlobalPower**. Which power control switches are enabled is determined by **PowerSwitchingMode** and **PortPortControlMask[NDP]**. In global switching mode (**PowerSwitchingMode**=0), only **Set/ ClearGlobalPower** controls this bit. In per-port power switching (**PowerSwitchingMode**=1), if the **PortPowerControlMask[NDP]** bit for the port is set, only **Set/ClearPortPower** commands are enabled. If the mask is not set, only **Set/ ClearGlobalPower** commands are enabled. When port power is disabled, **CurrentConnectStatus**, **PortEnableStatus**, **PortSuspendStatus**, and **PortResetStatus** should be reset. 0 = port power is off 1 = port power is on (write) **SetPortPower** The HCD writes a '1' to set the **PortPowerStatus** bit. Writing a '0' has no effect. Note: This bit is always reads '1b' if power switching is not supported.
07 - 05	reserved				
04	PRS	0b	RW	RW	(read) **PortResetStatus** When this bit is set by a write to **SetPortReset**, port reset signaling is asserted. When reset is completed, this bit is cleared when **PortResetStatusChange** is set. This bit cannot be set if **CurrentConnectStatus** is cleared. 0 = port reset signal is not active 1 = port reset signal is active (write) **SetPortReset** The HCD sets the port reset signaling by writing a '1' to this bit. Writing a '0' has no effect. If **CurrentConnectStatus** is cleared, this write does not set **PortResetStatus**, but instead sets **ConnectStatusChange**. This informs the driver that it attempted to reset a disconnected port.

Bit	Field name	Root Hub Reset	Read/Write		Description
			HCD	HC	
03	POCI	0b	RW	RW	(read) **PortOverCurrentIndicator** This bit is only valid when the Root Hub is configured in such a way that overcurrent conditions are reported on a per-port basis. If per-port overcurrent reporting is not supported, this bit is set to 0. If cleared, all power operations are normal for this port. If set, an overcurrent condition exists on this port. This bit always reflects the overcurrent input signal 0 = no overcurrent condition. 1 = overcurrent condition detected. (write) **ClearSuspendStatus** The HCD writes a '1' to initiate a resume. Writing a '0' has no effect. A resume is initiated only if **PortSuspendStatus** is set.
02	PSS	0b	RW	RW	(read) **PortSuspendStatus** This bit indicates the port is suspended or in the resume sequence. It is set by a **SetSuspendState** write and cleared when **PortSuspendStatusChange** is set at the end of the resume interval. This bit cannot be set if **CurrentConnectStatus** is cleared. This bit is also cleared when **PortResetStatusChange** is set at the end of the port reset or when the HC is placed in the USBRESUME state. If an upstream resume is in progress, it should propagate to the HC. 0 = port is not suspended 1 = port is suspended (write) **SetPortSuspend** The HCD sets the **PortSuspendStatus** bit by writing a '1' to this bit. Writing a '0' has no effect. If **CurrentConnectStatus** is cleared, this write does not set **PortSuspendStatus**; instead it sets **ConnectStatusChange**. This informs the driver that it attempted to suspend a disconnected port.

Bit	Field name	Root Hub Reset	Read/Write		Description
			HCD	HC	
01	PES	0b	RW	RW	(read) **PortEnableStatus** This bit indicates whether the port is enabled or disabled. The Root Hub may clear this bit when an overcurrent condition, disconnect event, switched-off power, or operational bus error such as babble is detected. This change also causes **PortEnabledStatusChange** to be set. HCD sets this bit by writing **SetPortEnable** and clears it by writing **ClearPortEnable**. This bit cannot be set when **CurrentConnectStatus** is cleared. This bit is also set, if not already, at the completion of a port reset when **ResetStatusChange** is set or port suspend when **SuspendStatusChange** is set. 0 = port is disabled 1 = port is enabled (write) **SetPortEnable** The HCD sets **PortEnableStatus** by writing a '1'. Writing a '0' has no effect. If **CurrentConnectStatus** is cleared, this write does not set **PortEnableStatus**, but instead sets **ConnectStatusChange**. This informs the driver that it attempted to enable a disconnected port.
00	CCS	0b	RW	RW	(read) **CurrentConnectStatus** This bit reflects the current state of the downstream port. 0 = no device connected 1 = device connected (write) **ClearPortEnable** The HCD writes a '1' to this bit to clear the **PortEnableStatus** bit. Writing a '0' has no effect. The **CurrentConnectStatus** is not affected by any write. Note: This bit is always read '1b' when the attached device is nonremovable (**DeviceRemoveable[NDP]**).

The *HcRhPortStatus*[1:NDP] register is used to control and report port events on a per-port basis. **NumberDownstreamPorts** represents the number of *HcRhPortStatus* registers that are implemented in hardware. The lower word is used to reflect the port status, whereas the upper word reflects the status change bits. Some status bits are implemented with special write behavior (see below). If a transaction (token through handshake) is in progress when a write to change port status occurs, the resulting port status change must be postponed until the transaction completes. Reserved bits should always be written '0'.

6.4 IEEE1284 Host Controller

Figure 9. IEEE1284 Block Diagram

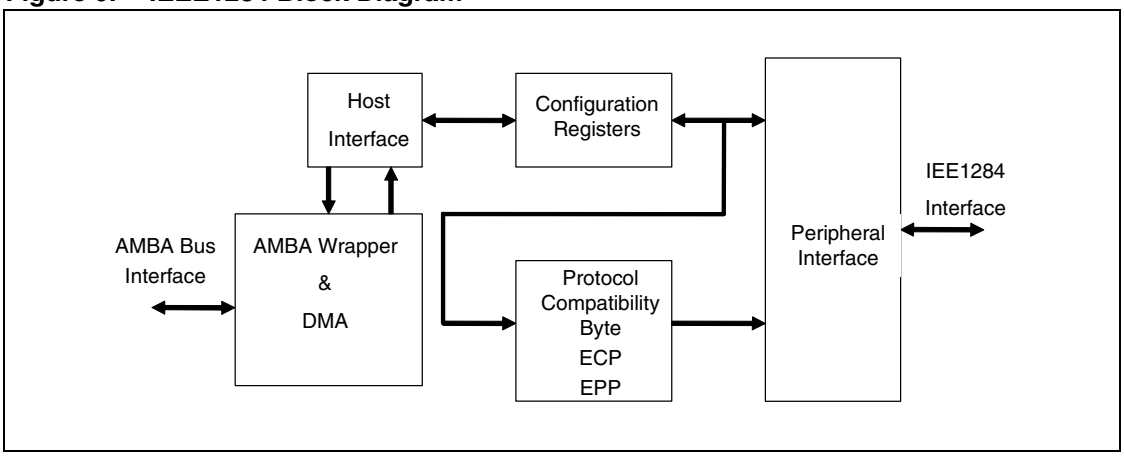
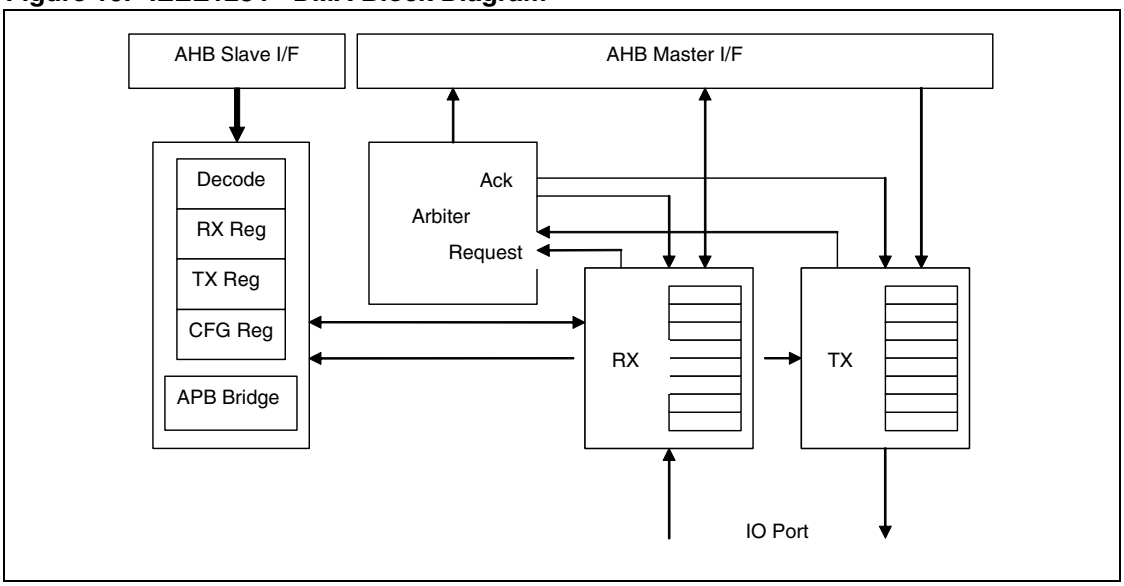


Figure 10. IEEE1284 - DMA Block Diagram



6.4.1 Overview

The IEEE1284 is a host-based multi-function parallel port that may be used to transfer data between a host PC and a peripheral such as a printer. It is designed to attach to the PC's ISA bus on one side and to the parallel port connector on the other.

The parallel port interface comprises nine control/status lines and an 8-bit bi-directional data bus.

It can be configured to operate in five modes, corresponding to the IEEE Standard 1284 parallel interface protocol standards.

6.4.2 Communication modes

6.4.2.1 COMPATIBILITY MODE

Compatibility Mode provides an asynchronous, byte wide, forward channel (host-to-peripheral), with the data and status lines used according to original definitions, as per the original Centronics port.

6.4.2.2 NIBBLE MODE

Nibble Mode provides an asynchronous, reverse channel (peripheral-to-host) under the control of the host. Data bytes are transmitted as two sequential, four-bit nibbles using four peripheral-to-host status lines.

When the host and/or peripheral do not support bi-directional use of the data lines, Nibble Mode may be used with Compatibility Mode to implement a bi-directional channel.

Note: The two modes cannot be active simultaneously.

6.4.2.3 PS2 OR BYTE MODE

Byte Mode provides an asynchronous, byte wide, reverse channel (peripheral-to-host) using the eight data lines of the interface for data and the control/status lines for handshaking. Byte Mode may be used to implement a bi-directional channel, with the transfer direction controlled by the host when both host and peripheral support bi-directional use of the data lines.

6.4.2.4 EPP MODE

Enhanced Parallel Port (EPP) Mode provides an asynchronous, byte wide, bi-directional channel controlled by the host device. This mode provides separate address and data cycles over the eight data lines of the interface.

6.4.2.5 ECP MODE

Extended Capabilities Port (ECP) Mode provides an asynchronous, byte wide, bi-directional channel. An interlocked handshake replaces the Compatibility Mode's minimum timing requirements. A control line is provided to distinguish between command and data transfers.

6.4.3 Matrix of Protocol Signal Names

SPEAr Net Signal names	Compatible Signal Names	Nibble Signal Names	Byte Signal Names	EPP Signal Names	ECP Signal Names
PD[7:0}	PD[7:0}	-	PD[7:0}	PD[7:0}	PD[7:0}
SLCT	Select	XFlag	XFlag	XFlag	XFlag
NACK	nACK	PtrClk	PtrClk	Intr	nPeriphClk
BUSY	Busy	PtrBusy	PtrBusy	nWait	PeriphAck
PE	Pe	AckDataReq	AckDataReq	AckDataReq	nAckReverse
NERR	nFault	nDataAvail	nDataAvail	nDataAvail	nPeriphRequest
NSLCTIN	nSelectIn	1284 Active	1284 Active	nAStrb	ECPMode
NINIT	nINIT	-	-	nINIT	nReverseRequest
NSTROBE	nSTROBE	HostClk	HostClk	nWrite	HostClk
NAUTOFD	nAutoFd	HostBusy	HostBusy	nDStrb	HostAck

6.4.4 Register MAP

Table 13. IEEE1284 Register Map

Address	Register Name	Description
0x2200_0000	DMA_CTRL_STAT	DMA Status and control register
0x2200_0004	DMA_INT_EN	DMA Interrupt source enable register
0x2200_0008	DMA_INT_STAT	DMA Interrupt status register
0x2200_0010	DMA_RX_START	DMA RX Start register
0x2200_0014	DMA_RX_CTRL	DMA RX Control register
0x2200_0018	DMA_RX_ADDR	DMA RX Address register
0x2200_0020	DMA_RX_CADDR	DMA RX Current Address register
0x2200_0024	DMA_RX_CXFER	DMA RX Current transfer count register
0x2200_0028	DMA_RX_TO	DMA RX FIFO Time out register
0x2200_002c	DMA_RX_FIFO_STS	DMA RX FIFO Status register
0x2200_0030	DMA_TX_START	DMA TX Start register
0x2200_0034	DMA_TX_CTRL	DMA TX Control register
0x2200_0038	DMA_TX_ADDR	DMA TX Address register
0x2200_0040	DMA_TX_CADDR	DMA TX Current Address register
0x2200_0044	DMA_TX_CXFER	DMA TX Current transfer count register
0x2200_0048	DMA_TX_TO	DMA TX FIFO Time out register
0x2200_004c	DMA_TX_FIFO_STS	DMA TX FIFO Status register
0x2200_0678 (1)	IEEE_CMP_DATA	In/Out Data register when used in compatibility and nibble mode
	IEEE_EPP_DATA	In/Out Data register when used in EPP mode

Table 13. IEEE1284 Register Map

Address	Register Name	Description
	IEEE_ECP_ADDR	Address register when used in ECP mode
0x2200_0679 (1)	IEEE_CMP_STAT	Status register in all modes
	IEEE_EPP_STAT	
	IEEE_ECP_STAT	
0x2200_067A (1)	IEEE_CMP_CTRL	Control register in all modes
	IEEE_EPP_CTRL	
	IEEE_ECP_CTRL	
0x2200_067B	IEEE_EPP_ADDRSTB	Address Strobe register in EPP mode
0X2200_067C	IEEE_EPP_DATASTB1	Data 1 Strobe register in EPP mode
0X2200_067D	IEEE_EPP_DATASTB2	Data 2 Strobe register in EPP mode
0X2200_067E	IEEE_EPP_DATASTB3	Data 3 Strobe register in EPP mode
0X2200_067F	IEEE_EPP_DATASTB4	Data 4 Strobe register in EPP mode
0X2200_07F0	IEEE_CSR	Main Configuration register
0x2200_07F1	IEEE_CFG_CR1	Configuration register CR1
	IEEE_CFG_CR4	Configuration register CR4
	IEEE_CFG_CRA	Configuration register CRA
0x2200_0800 (3)	IEEE_ECP_FIFO	Data FIFO register in ECP mode
	IEEE_ECP_TEST	Test register in ECP mode
	IEEE_ECP_CFGA	Configuration register A in ECP mode
0X2200_0801	IEEE_ECP_CFGB	Configuration register B in ECP mode
0X2200_0802	IEEE_ECP_ECR	Extended Control register in ECP mode

6.4.5 IEEE1284 Configuration

The IEEE1284 is configured by a set of three programmable registers accessed through a Configuration Select Register (CSR). These registers will be in default state after power-up and are unaffected by RESET.

6.4.5.1 Configuration Procedure

The following sequence is required to program the configuration registers:

Step	Action	Method
1	Enter Configuration Mode	This requires 55h to be written to port 0X2200_07F0 (CSR) twice in succession. **Note:** It is recommended that interrupts be disabled for the duration of the two writes. If a write to another address or port occurs between the two writes, the IEEE1284 will not enter Configuration Mode.
2	Configure Registers	The IEEE1284 contains three configuration registers CR1, CR4 and CRA. These registers are accessed by first writing the number of the desired register to port 0X2200_07F0 (CSR), then writing or reading the selected register through port 0X2200_07F1
3	Exit Configuration Mode	Configuration Mode is exited by writing an AAh to port 0x2200_07F0 (CSR).

6.4.5.2 Configuration Select Register

This Write-Only register can only be accessed when the IEEE1284 is in Configuration Mode. The CSR is located at port 0x2200_07F0 and must be initialized upon entering Configuration Mode before the three configuration registers can be accessed, after which it can be used to select which of the configuration registers is to be accessed at port 0x2200_07F1.

6.4.5.3 Configuration Register CR1

This register can only be accessed when the M1284H is in the Configuration Mode and after CSR has been initialized to 01h. The default value of this register after power-up is 9Fh. The bit definitions are shown below:

Bit	Function	Description
7	-	Not used
6	-	Not used
5	-	Not used
4	-	Not used
3	Parallel Port Mode	If 1, sets the Parallel Port for Compatibility Mode (Default). If 0, enables the Extended Parallel Port Mode. (See CR4)
2	-	Not used
1:0	Parallel Port Address	These bits are used to select the Parallel Port Address.

Bit 1	Bit 0	Description
0	0	Disabled
0	1	0x2200_0778
1	0	0x2200_07BC
1	1	0x2200_0678 (Default)

6.4.5.4 Configuration Register CR4

This register can only be accessed when the IEEE1284 is in Configuration Mode and after CSR has been initialized to 04h.

The default value of this register after power-up is 00h. The bit definitions are shown below:

Bit	Function	Description
7	Reserved	Must be always written with 0
6	-	Not used
5	-	Not used
4	-	Not used
3	-	Not used
2	-	Not used
1:0		

Bit 1	Bit 0	If CR1[3] = 0 then
0	0	Standard Parallel Port (SPP) operation (Default) (Note 1)
0	1	EPP Mode (also supports SPP operation)
1	0	ECP Mode (Note 2)
1	1	ECP & EPP Modes (Notes 2, 3)

If CR1 (3) = 1, the port is placed in Compatibility Mode.

Note: 1 Standard Parallel Port operation denotes the use of the Peripheral data bus in either Compatibility Mode (and/or Nibble Mode) or PS/2 (Byte) Mode.

2 SPP operation may be selected through the ECR register of ECP as mode 000.

3 EPP Mode is selected through the ECR register of ECP as mode 100.

6.4.5.5 Configuration Register CRA

This register can only be accessed when the IEEE1284 is in the Configuration Mode and after CSR has been initialized to 0Ah. The default value of this register after power-up is 00h. This register's byte defines the FIFO threshold for the ECP Mode parallel port.

6.4.6 DMA Registers

All The DMA registers are 32 bit wide.

6.4.6.1 DMA Status and Control Register

Address 0x2200_0000

Default value 0x

Bit	Field name	Access
31 - 28	TX_FIFO _SIZE	RO
27 - 26	TX_IO_DATA_WIDTH	RO
25 - 24	TX_CHAN_STATUS	RO
23 - 20	RX_FIFO_SIZE	RO
19 - 18	RX_IO_DATA_WIDTH	RO
17 - 16	RX_CHAN_STATUS	RO
15 - 08	REVISION	RO
07 - 04	Reserved	RO
03 - 02	Reserved	RO
01	LOOPB	RW
00	SRESET	RW

TX_FIFOSIZE: Size of transmitter data path FIFO.

- 0001: 2 * 32 BIT WORDS

TX_IO_DATA_WIDTH: Width of the I/O bus transmit data path

- 00: 8-bit

TX_CHAN_STATUS: provides information about the TX channel structure

- 01: Low End TX Channel (No DMA descriptor fetch)

RX_FIFOSIZE: Size of receiver data path FIFO.

- 0001: 2 * 32 BIT WORDS

RX_IO_DATA_WIDTH: Width of the I/O bus receive data path

- 00: 8-bit

RX_CHAN_STATUS: provides information about the RX channel structure

- 01: Low End RX Channel (No DMA descriptor fetch)

REVISION: Revision of the DMA

- ????????

LOOPB: Set to '1' to enable the loop-back mode. When set the RX DMA data are extracted by the TX FIFO and pushed in the RX one.

SRESET: DMA soft reset, set to '1' to put the whole DMA logic in reset condition.

This signal has no effect on the AHB interface so the whole DMA will be reset only when the last AHB transfer is finished.

6.4.6.2 DMA_INT_EN

Address: 0x2200_0004

Default value: 0x0000_0000

Bit	Field Name	Access
31	Reserved	RO
30:29	Reserved	RO
28	TX_IO_INT_EN	
27:26	Reserved	
25	TX_MERR_INT_EN	
24	TX_SERR_INT_EN	
23	TX_DONE_EN	
22	Reserved	
21	TX_IOREQ_EN	
20	TX_RTY_EN	
19	TX_TO_EN	
18	TX_ENTRY_EN	
17	TX_FULL_EN	
16	TX_EMPTY_EN	
15	Reserved	
14:13	Reserved	
12	RX_IO_INT_EN	
11:10	Reserved	
09	RX_MERR_INT_EN	
08	RX_SERR_INT_EN	
07	RX_DONE_EN	
06	Reserved	
05	RX_IORQ_EN	
04	RX_RTY_EN	
03	RX_TO_EN	
02	RX_ENTRY_EN	
01	RX_FULL_EN	
00	RX_EMPTY_EN	

The DMA Interrupt enable register allows the various sources of interrupt to be individually enabled.

All the enabled sources will then be OR-ed to generate the global DMA interrupt.

Setting a bit in DMA_INT_EN register allows the corresponding interrupt described in DMA_INT_STAT to influence the global DMA interrupt.

If any bit is set to '1' in both DMA_INT_STAT and DMA_INT_EN then the DMA interrupt will be asserted.

Refer to the DMA_INT_STAT for a description of the interrupt sources.

6.4.6.3 DMA Interrupt Sources Status Register

Mnemonic: DMA_INT_STAT

Address: 0x2200_0008

Default value: 0x0000_0000

Bit	Field Name	Access
31	Reserved	RO
30:29	Reserved	RO
28	TX_IO_INT	RW
27:26	Reserved	RO
25	TX_MERR_INT	RW
24	TX_SERR_INT	RW
23	TX_DONE	RW
22	Reserved	RO
21	TX_IOREQ	RW
20	TX_RTY	RW
19	TX_TO	RW
18	TX_ENTRY	RW
17	TX_FULL	RW
16	TX_EMPTY	RW
15:13	Reserved	RO
12	RX_IO_INT	RW
11:10	Reserved	RO
09	RX_MERR_INT	RW
08	RX_SERR_INT	RW
07	RX_DONE	RW
06	Reserved	RO
05	RX_IOREQ	RW
04	RX_RTY	RW
03	RX_TO	RW
02	RX_ENTRY	RW
01	RX_FULL	RW
00	RX_EMPTY	RW

Client interrupt status register reports the interrupt status of interrupts from the following sources:

DMA_RX, DMA_TX and IO_IP.

All the significant register bits are read/Clear (RC): they can be read, a write with '0' has no effect while writing '1' reset the bit value to '0'.

Each bit can bi cleared writing '1' (writing '0' will have no effect)

TX_IO_INT: Set when the external IO device (IEEE1284 block), connected to the TX DMA port, sets an interrupt request.

TX_MERR_INT: Set when AHB master receives an error response from the selected slave and the internal arbiter is granting the TX FIFO.

TX_SERR_INT: Set when the AHB slave drives an error on the AHB BUS as response to an TX_FIOFO_PUSH request. This condition is achieved when on of the following conditions is true:

i. TX_DMA.START_SERR_EN is true and retry counter expires.

ii. Slave access with size > 32 bit.

iii. Slave access with a read request.

iv. Slave access when the TX_DMA_START.DMA_EN is true (DMA Master Mode)

TX_DONE: Set when the TX master DMA completes.

TX_IOREQ: Set when the DMA TX is active (master or slave mode) and the IO interface request cannot be served because:

i. FIFO is empty

ii. Current DMA cycle is finished and the next one is not yet started.

TX_RTY: Set when the AHB slave retry counter expires (even if the TX_DMA_START.SERR_EN is false), that means that an AHB TX FIFO write has been attempt, with a wrong byte size attributes, more than allowed by the retry counter.

TX_TO: Set when some data are stalled inside the TX FIFO for too long time.

TX_ENTRY: Set when the TX DMA is triggered by a number of empty TX FIFO entries bigger than the value set in the DMA_CNTL register.

TX_FULL: Set when the TX FIFO becomes full (< 4 byte entries available).

TX_EMPTY: Set when the TX FIFO becomes empty.

RX_IO_INT: Set when the external IO device (IEEE1284 block), connected to the RX DMA port, sets an interrupt request.

RX_MERR_INT: Set when AHB master receives an error response from the selected slave and the internal arbiter is granting the RX FIFO.

RX_SERR_INT: Set when the AHB slave drives an error on the AHB BUS as response to a **RX_FIFO_POP** request. This condition is achieved when on of the following conditions is true:

v. RX_DMA.START_SERR_EN is true and retry counter expires.

vi. Slave access with size > 32 bit.

vii. Slave access with a write request.

viii. Slave access when the RX_DMA_START.DMA_EN is true (DMA Master Mode)

RX_DONE: Set when the RX master DMA completes.

RX_IOREQ: Set when the DMA RX is active (master or slave mode) and the IO interface request cannot be served because:

i. FIFO is full

ii. Current DMA cycle is finished and the next one is not yet started.

iii. There is a Time-out condition (acknowledge is de-asserted for one clock cycle)

RX_RTY: Set when the AHB slave retry counter expires (even if the RX_DMA_START.SERR_EN is false), that means that an AHB RX FIFO read has been attempt, with a wrong byte size attributes, more than allowed by the retry counter.

RX_ENTRY: Set when the RX DMA is triggered by a number of valid RX FIFO entries bigger than the value set in the DMA_CNTL register.

RX_FULL: Set when the RX FIFO becomes full and no more data can be accepted.

RX_EMPTY: Set when the RX FIFO becomes empty.

6.4.6.4 RX DMA Start Register

Mnemonic: RX_DMA_START

Address: 0x2200_0010

Default value: 0x0000_0000

Bit	Field Name	Access
31:24	Reserved	RO
23:08	Reserved	RO
07:04	Reserved	RO
03	SERR_EN	RW
02	Reserved	RO
01	IO_EN	RW
00	DMA_EN	RW

SERR_EN: Set to '1' to enable the RX DMA slave logic to respond with an error, instead of retry, when the retry count expires.

IO_EN: Set to '1' enable the RX DMA interface, in slave mode (DMA_EN must be '0'), to get data from the IO IP (IEEE1284) and write it into the RX FIFO.

DMA_EN: Writing '1' starts the RX DMA master SM running. When all the DMA sequences complete, this bit is reset by the DMA logic.

Note: The DMA_EN 0->1 transition or IO_EN 0->1(with DMA_EN = 0) transition resets the FIFO content and the RX interrupts (DMA_INT_STAT(15:0)).

Even if DMA_EN has an higher priority with respect IO_EN (when DMA_EN=1, the IO_EN bit is don't care), it's suggested to avoid the set of the two bit at the same time.
When the processor wants to start a new master DMA, first it has to fill the descriptor registers (DMA_CNTL, DMA_ADDR and, if required, DMA_NXT) and then it has to enable the DMA (write a '1' in DMA_EN).
When all the DMA sequences complete, the DMA SM resets to '0' the DMA_EN bit and waits for this field being enabled again.
If the DMA descriptor fetch logic has been enabled, more than one DMA can complete before the DMA_EN bit is reset; in this case it will be set to '0' only after the last DMA ends.

6.4.6.5 RX DMA CNTL Register

Mnemonic: RX_DMA_CNTL

Address: 0x200_0014

Default value: 0x0000_0000

Bit	Field Name	Access
31:22	ADDR_WRAP	RW
21:17	ENTRY_TRIG	RW
16	Reserved	RO
15	DLY_EN	RW
14	Reserved	RO
13	Reserved	RO
12	CONT_EN	RW
11:00	DMA_XFERCOUNT	RW

ADDR_WRAP: Determines where the DMA address counter wraps by forcing the DMA address counter to retain the data originally written by the host in DMA_ADDR. As soon as the DMA has written the memory location prior to the value specified in ADD_WRAP the wrapping condition occurs.

This can be used to restrict the address counter within an address window (e.g. circular buffer).

The wrapping point MUST be 32 bit aligned, so the 10 bits of ADDR_WRAP are used to compare DMA address bits 11 to 2; if ADD_WRAP=DMA_ADDR (11:2) then a 4Kbyte buffer is defined.

ADDRWRAP is ignored unless WRAP_EN is set.

ENTRY_TRIG: Determines the amount of valid entries (in 32 BIT words) required in the receive FIFO before the DMA is re-triggered.

If the value is set to 0, as soon as one valid entry is present, the DMA logic starts the data transfer.

DLY_EN: This bit enables (when '1') the DMA trigger delay feature: if a FIFO valid data resides in the FIFO more than a programmed period (DMA_TO), a time-out condition occurs that requires the DMA SM to empty the FIFO even if the number of valid words doesn't exceed the threshold value.

CONT_EN: Continuous Mode Enable, enables continuous DMA run. If set the DMA runs indefinitely ignoring DMA_ XFERCOUNT.

Note: "continuous mode" supersedes "next descriptor mode".

DMA_XFERCOUNT: Block size (in bytes) of DMA, maximum 4 Kbytes.

Note: The DMA_XFERCOUNT field MUST have a value multiple of the IO DMA bus size, i.e.

- IO DMA DATA bus 8 bit -> all DMA_XFERCOUNT values are allowed
- IO DMA DATA bus 16 bit -> DMA_XFERCOUNT(0) MUST be 0
- IO DMA DATA bus 32 bit -> DMA_XFERCOUNT(1:0) MUST be 00

If DMA_XFERCOUNT is set to '0', the DMA will transfer 4 Kbyte data.

6.4.6.6 RX DMA ADDR Register

Mnemonic: RX_DMA_ADDR

Address: 0x2200_0018

Default value: xxxx_xxxx

Bit	Field Name	Access
31:02	DMA_ADDR	RW
01	FIX_ADDR	RW
00	WRAP_EN	RW

DMA_ADDR: Start address, 32 bits WORD ALIGNED, for master DMA transfer.

DMA SM will read this register only before starting the DMA operation, so further updates of this register will have no effect on the running DMA.

FIX_ADDR: Disables incrementing of DMA_ADDR: this means that all the DMA data transfer operation will be performed at the same AHB address, i.e. the DMA base address.

WRAP_EN: Enables wrap of the DMA transfer address to DMA_ADDR when the memory location, specified in ADDR_WRAP, is reached.

6.4.6.7 RX DMA Current Address Register

Mnemonic: RX_DMA_CADDR

Address: 0x2200_0020

Default: xxxx_xxxx

Bit	Field Name	Access
31:00	DMA_CADDR	RW

DMA_CADDR: Current DMA address value, byte aligned.

The value of this register will change while the DMA is running, reflecting the value driven by the core on the AHB bus.

6.4.6.8 RX DMA Current Transfer Count Register

Mnemonic: RX_DMA_CXFER

Address: 0x2200_0024

Default value: xxxx_xxxx

Bit	Field Name	Access
31:12	Reserved	RO
11:00	DMA_CXFER	RW

DMA_CXFER: Current DMA address value, byte aligned.

The value of this register will change while the DMA is running, reflecting the value driven by the core on the AHB bus.

6.4.6.9 RX DMA FIFO Time Out Register

Mnemonic: RX_DMA_TO

Address: 0x2200_0028

Default value: 0x0000_0000

Bit	Field Name	Access
31:16	Reserved	RO
15:00	TIME_OUT	RW

TIME_OUT: This value is used as initial value for the FIFO entry time out counter in master mode and as initial value for the RETRY counter in slave mode. Register value must be not zero if the feature that use it are activated.

The time-out counter starts as soon as one valid entry is present in the FIFO and is reset every time a FIFO data is pop out of the FIFO.

The counter expires (FIFO time out condition) if no FIFO data are pop for a period longer than the TIME_OUT register value; when this happens, depending on the control registers settings, an interrupt can be set or the FIFO can be flushed.

Retry counter is incremented after each AHB slave RETRY response and is cleared after OKAY or ERROR response.

6.4.6.10 RX DMA FIFO Status Register

Mnemonic: RX_DMA_FIFO

Address: 0x2200_002C

Default value: 0x????_????

Bit	Field Name	Access
31:30	Reserved	RO
29:24	ENTRIES	RO
23:21	Reserved	RO
20:16	DMA_POINTER	RO
15:13	Reserved	RO
12:08	IO_POINTER	RO
07:04	Reserved	RO
03	DELAY_T	RO
02	ENTRY_T	RO
01	FULL	RO
00	EMPTY	RO

ENTRIES: Full entries (in 32 bits words) in FIFO.

DMA_POINTER: FIFO DMA SM side pointer value.

IO_POINTER: FIFO IO side pointer value.

DELAY_T: Set to '1' when DMA FIFO delay time out is expired.

ENTRY_T: Set to '1' when the DMA FIFO entry trigger threshold has been reached.

FULL: Set to '1' when the DMA FIFO is full.

EMPTY: Set to '1' when the DMA FIFO is empty

6.4.6.11 DMA Start register

Mnemonic: TX_DMA_START

Address: 0x2200_0030

Default value: 0x0000_0000

Bit	Field Name	Access
31:04	Reserved	RO
03	SERR_EN	RW
02	Reserved	RO
01	IO_EN	RW
00	DMA_EN	RW

SERR_EN: Set to '1' to enable the TX DMA slave logic to respond with an error instead of retry when the retry counter is expired.

IO_EN: Set to '1' to enable the TX DMA interface, in slave mode (DMA_EN must be set to '0'), to get data from the TX FIFO, as soon as they are valid, and pass them to the IO IP (IEEE1284) block, on its request.

DMA_EN: The TX DMA master SM starts when this bit is set to '1'. When all the DMA sequences are completed, this bit is reset to '0' by the DMA logic itself.

Note: The DMA_EN 0->1 transition or IO_EN 0->1(with DMA_EN = 0) transition resets the FIFO content and the TX interrupts (DMA_INT_STAT(15:0)).
Even if DMA_EN has an higher priority with respect IO_EN (when DMA_EN=1, the IO_EN bit is don't care), it's suggested to avoid the set of the two bit at the same time.
When the processor wants to start a new master DMA, first it has to fill the descriptor registers (DMA_CNTL, DMA_ADDR and, if required, DMA_NXT) and then it has to enable the DMA (write a '1' in DMA_EN).
When all the DMA sequences complete, the DMA SM resets to '0' the DMA_EN bit and waits for this field being enabled again.
f the DMA descriptor fetch logic has been enabled, more than one DMA can complete before the DMA_EN bit is reset; in this case it will be set to '0' only after the last DMA ends.

6.4.6.12 TX DMA Control Register

Mnemonic: TX_DMA_CNTL

Address: 0x2200_0034

Default value: 0x0000_0000

Bit	Field Name	Access
31:22	ADDR_WRAP	RW
21:17	ENTRY_TRIG	RW
16	Reserved	RO
15	DLY_EN	RW
14:13	Reserved	RO
12	CONT_EN	RW
11:00	DMA_XFERCOUNT	RW

ADDR_WRAP: Determines where the DMA address counter wraps by forcing the DMA address counter to retain the data originally written by the host in DMA_ADDR. As soon as the DMA has read the memory location prior to the value specified in ADD_WRAP the wrapping condition occurs.

This can be used to restrict the address counter within an address window (e.g. circular buffer).

The wrapping point MUST be 32 bit aligned, so the 10 bits of ADDR_WRAP are used to compare DMA address bits 11 to 2; if ADD_WRAP=DMA_ADDR(11:2) then a 4Kbyte buffer is defined.

ADDRWRAP is ignored unless WRAP_EN is set.

ENTRY_TRIG: Determines the amount of empty entries (in 32 BIT words) required in the TX FIFO before the DMA is re-triggered.

If the value is set to 0, as soon as one empty entry is present, the DMA logic starts the data request.

DLY_EN: This bit enables (when '1') the DMA trigger delay feature: if a FIFO valid data resides in the FIFO more than a programmed period (DMA_TO), a time-out condition occurs and the related (TX_TO) interrupt will be set.

CONT_EN: Continuous Mode Enable, enables continuous DMA run. If set the DMA runs indefinitely ignoring DMA_ XFERCOUNT.

Note: "continuous mode" supersedes "next descriptor mode".

DMA_XFERCOUNT: Block size (in bytes) of DMA, maximum 4 Kbytes.

Note: The DMA_XFERCOUNT field MUST have a value multiple of the IO DMA bus size, i.e.

- IO DMA DATA bus 8 bit -> all DMA_XFERCOUNT values are allowed
- IO DMA DATA bus 16 bit -> DMA_XFERCOUNT(0) MUST be 0
- IO DMA DATA bus 32 bit -> DMA_XFERCOUNT(1:0) MUST be 00

If DMA_XFERCOUNT is set to '0', the DMA will transfer 4 Kbytes data.

6.4.6.13 TX DMA Address Register

Mnemonic: TX_DMA_ADDR

Address: 0x2200_0038

Default value: xxxx_xxxx

Bit	Field Name	Access
31:02	DMA_ADDR	RW
01	FIX_ADDR	RW
00	WRAP_EN	RW

DMA_ADDR: Start address, 32 bit WORD ALIGNED, for master DMA transfer.

The DMA SM reads this register only before starting the DMA operation, so further updates of this register will have no effect on the running DMA.

While the DMA is in progress, a read operation to this register will return the DMA current address value.

FIX_ADDR: Disables incrementing of DMA_ADDR: this means that all the DMA data transfer operation will be performed at the same AHB address, i.e. the DMA base address.

WRAP_EN: Enables wrap of the DMA transfer address to DMA_ADDR when the memory location, specified in ADDR_WRAP, is reached.

6.4.6.14 TX DMA Current Address Register

Mnemonic: TX_DMA_CADDR

Address: 0x2200_0040

Default value: xxxx_xxxx

Bit	Field Name	Access
31:00	DMA_CADDR	RO

DMA_CADDR: Current DMA address value, byte aligned.

The value of this register will change while the DMA is running, reflecting the value driven by the core on the AHB bus.

6.4.6.15 TX DMA Current Transfer Register

Mnemonic: TX_DMA_CXFER

Address: 0x2200_0044

Default value: xxxx_xxxx

Bit	Field Name	Access
31:12	Reserved	RO
11:00	DMA_CXFER	RO

DMA_CXFER: Current DMA transfer counter value.

It's updated while the DMA is running.

6.4.6.16 TX DMA FIFO Time Out Register

Mnemonic: TX_DMA_TO

Address: 0x2200_0048

Default value: 0x0000_0000

Bit	Field Name	Access
31:16	Reserved	RO
15:00	TIME_OUT	RW

TIME_OUT: This value is used as initial value for the FIFO entry time out counter in master mode and as initial value for the RETRY counter in slave mode. Register value must be not zero if the features that use it are activated.

This counter starts as soon as one valid entry is present in the FIFO and is reset every time a FIFO data is pop out of the FIFO.

The counter expires (FIFO time out condition) if no FIFO data are pop for a period longer than the TIME_OUT register value; when this happens, depending on the control registers settings, an interrupt can be set.

Retry counter is incremented after each AHB slave RETRY response and is cleared after OKAY or ERROR response.

6.4.6.17 TX DMA FIFO Status Register

Mnemonic: TX_DMA_FIFO

Address: 0x2200_004C

Default value: 0x????_????

Bit	Field Name	Access
31:30	Reserved	RO
29:24	ENTRIES	RO
23:21	Reserved	RO
20:16	DMA_POINTER	RO
15:13	Reserved	RO
12:08	IO_POINTER	RO
07:04	Reserved	RO
03	DELAY_T	RO
02	ENTRY_T	RO
01	FULL	RO
00	EMPTY	RO

ENTRIES: Free entries (in 32 bits word) in FIFO.

DMA_POINTER: FIFO DMA SM side pointer value.

IO_POINTER: FIFO IO side pointer value.

DELAY_T: Set to '1' when the DMA FIFO delay time out is expired.

ENTRY_T: Set to '1' when the DMA FIFO entry trigger threshold has been reached.

FULL: Set to '1' when DMA FIFO is full.

EMPTY: Set to '1' when the DMA FIFO is empty.

6.4.7 Parallel Port register

This section is split into three sub-sections: Compatibility and Byte Modes; EPP Mode; and ECP mode. Each register set description gives the I/O address assignments and a description of the relevant registers and its bits. It is worth noting that the STAT and CTRL registers (described under Compatibility and Byte Modes) are common to all modes.

The base address for the parallel port is determined at power-up. This can be changed by software as described in Section

All registers are accessed as byte quantities.

Some of the registers described contain reserved bits. These will have a hard value associated with them, defined in the register description: this value will not change even if these bits are written to. A read from a register that contains reserved bits will return the hard values associated with those bits.

6.4.7.1 Compatibility and Byte modes

The port consists of three registers and can be programmed to operate at three different base addresses - 0x2200_0678, 0x2200_0378 and 0x2200_03BC.

The write locations are:

i. Write data to output port (DATA) - Base Address + 0h
ii. Write command to output port (CTRL)- Base Address + 2h

The read locations are:

iii. Read peripheral data (DATA) - Base Address + 0h
iv. Read peripheral status data (STAT) - Base Address + 1h
v. Read back control register (CTRL) - Base Address + 2h

STAT Register (RO)

This Read-Only register has a port address of Base Address + 01h. The bit definitions are shown below:

Bit	Name	Comment
D7	NBUSY	If asserted, indicates that the peripheral is busy.
D6	NACK	If asserted, indicates that the peripheral has received a data byte and is ready for another.
D5	PE	If asserted, indicates that the peripheral is out of paper.
D4	SLCT	If asserted, indicates that the peripheral is selected.
D3	NERR	If asserted, indicates that the peripheral has a fault.
D2	Reserved	Always returns 0.
D1	Reserved	Always returns 0.
D0	TIMEOUT	Asserted high when timeout occurs (only in EPP mode).

Note: ***Note****: You may only read from the STAT register: writes to it have no effect.*

CTRL Register (RW)

This Read/Write register has a port address of Base Address + 02h. The bit definitions are shown below:

Bit	Name	Comment
D7	Reserved	Always returns 0.
D6	Reserved	Always returns 0.
D5	PDIR	Direction bit. (Read/Write only in EPP and ECP Modes.)
D4	INTEN	If asserted, allows the peripheral to interrupt the CPU.
D3	SLCTIN	If asserted, it means the host has selected the peripheral.
D2	NINIT	If asserted, the peripheral is initialized.
D1	AUTOFD	This tells the printer to advance the paper by one line each time a carriage return is received.
D0	STROBE	If asserted, this instructs the peripheral to accept the data on the data bus

Note: SLCTIN, AUTOFD and STROBE are inverted when read back from the CTRL register.
In PS/2 (Byte) Mode, bit 5 is used to control the direction of the data transfer on the parallel port data bus. Also in Byte Mode, when PDIR = 0 (forward direction), PDOUT[7:0] is enabled; when PDIR = 1 (reverse direction), PDIN[7:0] is enabled.
At reset, CTRL is set to 00h.

6.4.7.2 EPP Mode

The following table shows the I/O assignments for the registers used in EPP Mode:

Parallel Port Register Address Base Address +	Abbreviation	Register Name	Access
0h	DATA	Data Register	RW
1h	STAT	Status Register	RO
2h	CTRL	Control Register	RW
3h	ADDSTR	Address Strobe Register	RW
4h – 7h	DATASTR	Data Strobe Registers	RW

The DATA, STAT and CTRL registers are as described above for the Compatibility and Byte Modes.

The ADDSTR register provides a peripheral address to the peripheral via PDOUT[7:0] during a host write, and to the host via PDIN[7:0] during a host address read operation. An automatic address strobe is generated on the parallel port interface when data is read or written to this register.

The DATASTR registers provide data from the host to the peripheral via PDOUT[7:0] during a write operation, and data from the peripheral to the host during a read operation. An automatic data strobe is generated on the parallel port interface when data is read or written to these registers.

If no EPP Read, Write or Address cycle is currently being executed, the Peripheral data bus may be used in either Compatibility Mode (and/or Nibble Mode) or PS/2 (Byte) Mode. In this condition, all output signals (NSTROBE, NAUTOFD and NINIT) are set by the CTRL register and direction is controlled by the PDIR bit of the CTRL register.

Before an EPP cycle is executed, the control register PDIR bit must be set to 0 (by writing 04h or 05h to the CTRL register). If PDIR is left set to 1, the IEEE1284 will not be able to perform a write and will appear instead to perform an EPP read on the parallel bus without any error being indicated.

If an EPP bus cycle does not terminate within 10ms, the EPP timeout flag will be set and all following EPP bus cycles will be aborted until this flag is cleared writing to the STAT register.

6.4.7.3 ECP Mode

This section describes the registers used in ECP Mode. It is worth noting that the Extended Control Register ECR (which can only be entered from ECP Mode) allows various modes of operation. In particular, ECR[7:5] = 000 selects Compatibility Mode and ECR[7:5] = 001 selects Byte Mode. These have been discussed above.

The I/O assignments in ECP mode are shown below:

Port Address Base Address +	Abbreviation	Register Name	ECR[7:5]	Access
0h	ECPAFIFO	ECP Address Register	011	RW
1h	STAT	Status Register	All	R
2h	CTRL	Control Register	All	RW
400h	SDFIFO	Standard Parallel Port Data FIFO	010	RW
400h	ECPDFIFO	ECP Data FIFO	011	RW
400h	TFIFO	Test FIFO	110	RW
400h	CFGA	ECP Configuration A Register	111	RW
401h	CFGB	ECP Configuration B Register	111	RW
402h	ECR	Extended Control Register	All	RW

The functions of the above registers are detailed in the 'Extended Capabilities Port Protocol and ISA Interface Standard Rev 1.12', which is available from Microsoft. The bit map of the Extended Parallel Port Registers is shown below:

	D7	D6	D5	D4	D3	D2	D1	D0	Note
Data	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0	1
ECPAFIFO	1	Address							2
STAT	NBUSY	NACK	PE	SLCT	NERR	0	0	0	1
CTRL	0	0	PDIR	INTEN	SLCTIN	NINT	FD	STROBE	1
SDFIFO	Parallel Port Data FIFO								2
ECPDFIFO	ECP Data FIFO								2
TFIFO	Test FIFO								2
CFGA	0	0	0	1	0	0	0	0	
CFGB	0	PINTR1	0	0	0	0	0	0	
ECR	MODE			INTERR	DMAEN	SERVINT	FIFOF	FIFOE	

Note: 1 These Registers are available in all modes.

2 All FIFOs use one common 16 byte FIFO.

ECP Address FIFO Register (ECPAFIFO)

The ECPAFIFO register provides a channel address to the peripheral depending on the state of bit 7. This I/O address location is only used in ECP Mode (ECR bits [7:5]=011). In this mode, bytes written to this register are placed in the parallel port FIFO and transmitted via PDOUT[7:0] using the ECP protocol. Bit 7 should always be set to 1.

Bits [7:0] ECP Address

The register contents is passed to the peripheral by PDOUT[7:0]. The peripheral should interpret bits [6:0] as a channel address.

Note: that the SPEAr Net asserts NAUTOFD to indicate that information on the PD[7:0] is an ECP address. The SPEAr Net negates NAUTOFD when PD[7:0] is transferring data.

Status Register (STAT)

Already discussed in Section 5.4.5.1.1.

Control register (CTRL)

Already discussed in Section 5.4.5.1.2.

Standard Parallel Port Data FIFO Register (SDFIFO)

SDFIFO is used to transfer data from the host to the peripheral when the ECR register is set for compatible FIFO mode (Bits [7:5] = 010). Data bytes written or DMAed from the system to this FIFO are transmitted by a hardware handshake to the peripheral using the standard Compatibility protocol.

For this register, bytes are placed in the parallel port FIFO using DATAIN[7:0] and transmitted via PDOUT[7:0].

Note: that bit 5 in the CTRL register must be set to 0 for forward transfer.

ECP Data FIFO Register (ECPDFIFO)

ECPDFIFO is used to transfer data from the host to the peripheral when the ECR register is set for ECP mode (Bits [7:5] = 011). Data bytes written or DMAed from the system to this FIFO are transmitted by a hardware handshake to the peripheral using the ECP protocol.

Note: that bit 5 in the CTRL register must be set to 0 for forward transfer or to 1 for a reverse transfer.

Test FIFO Register (TFIFO)

The Test FIFO provides a test mechanism for the ECP Mode FIFO by allowing data to be read, written or DMAed in either direction between the system and this FIFO. This Test Mode is selected by setting ECR[7:5] = 110.

The data is transferred purely through the microprocessor interface and is therefore transferred at the maximum ISA rate.

It may appear on the parallel port data lines, but without any hardware handshake.

The Test FIFO does not stall when overwritten or under-run. Data is simply ignored or re-read. The full and empty bits of the ECR register (bits 1 and 0) can however be used to ascertain the correct state of the FIFO.

ECP Configuration Register A (CFGA)

The CFGA register provides information about the ECP Mode implementation. It is a Read Only register. Access to this register is enabled by programming the ECR register (ECR[7:5] = 111). The bit definitions in this register are shown below:

0b00010000

At reset CFGA is set to 10h.

10h indicates an 8-bit implementation.

ECP Configuration Register B (CFGB)

The CFGB register checks the PINTR1 line to determine possible conflicts. It is a Read Only register. The bit definitions are shown below:

Bit	Name	Description
D7	-	Always return 0
D6	PINTR1	Returns the value of the PINTR1 line
D5	-	Always return 0
D4	-	Always return 0
D3	-	Always return 0
D2	-	Always return 0
D1	-	Always return 0
D0	-	Always return 0

Extended Control Register (ECR)

This Read/Write register selects ECP mode, enables service and error interrupts and provides interrupt status. The ECR also enables and disables DMA operations and provides FIFO empty and FIFO full status. The bit definitions are shown below:

Bit	Name	Description
D7	MODE3	Mode bit.
D6	MODE2	Mode bit.
D5	MODE1	Mode bit.
D4	INTERR	Error interrupt enable.
D3	DMAEN	DMA enable.
D2	SERVINT	Service interrupt.
D1	FIFOF	This bit is set when the FIFO is full.
D0	FIFOE	This bit is set when the FIFO is empty.

Bits [7:5] - ECP Mode Select

This field selects one of the following modes:

Mode 000 This puts the parallel port into Compatibility Mode and resets the pointers to the FIFO (but not its contents). Setting the direction bit in the CTRL register does not affect the parallel port interface in this mode. For register descriptions in this mode, refer to *Section 4.1 'Functional Pin Groups'* on page 13.

Mode 001 This puts the parallel port into Byte Mode and resets the pointers to the FIFO (but not its contents). The outcome is similar to above except that the direction bit selects forward or reverse transfers.

Mode 010 This puts the parallel port into ISA Compatible FIFO mode, which is the same as mode 000 except that PWords are written or DMAed to the FIFO. FIFO data is automatically transmitted using the standard parallel port protocol. Note, this mode should only be used when PDIR = 0.

Mode 011 This puts the port into ECP Mode. In the forward direction, bytes written to the ECPDFIFO and ECPAFIFO locations are placed in the ECP FIFO and transmitted automatically to the peripheral using ECP protocol. In the reverse direction, bytes are transferred from PDIN [7:0] to the ECP FIFO.

Mode 100 Reserved.

Mode 101 Reserved.

Mode 110 This selects an ECP Test Mode in which the FIFO is read and written purely through the microprocessor interface.

Mode 111 This is places the interface into Configuration Mode. In this mode, the CFGA and CFGB registers are accessible at base address + 400h and at base address + 401h, respectively.

Bit [4] - INTERR

When 0, this bit enables error interrupts to the host when a high to low transition occurs on the NERR signal.

Bit [3] - DMAEN

DMA is enabled when DMAEN = 1, and disabled when DMAEN =0.

Bit [2] - Service Interrupt (SERVINT)

If SERVINT = '1', DMA is disabled and all service interrupts are disabled.

SERVINT = '0' enables one of three interrupts:

i. If DMAEN = '1' during DMA, the SERVINT bit will be set to a 1 when terminal count is reached.

ii. If DMAEN = '0' and PDIR = 0, the SERVINT bit will be set to '1' whenever there are 'WriteIntrThreshold' or more bytes free in the FIFO (see Section 4.3.6.1).

iii. If DMAEN = '0' and PDIR = 1, the SERVINT bit will be set to '1' whenever there are 'ReadIntrThreshold' or more valid bytes to be read from the FIFO (see Section 4.3.6.2).

Bit [1] - FIFO Full Status (FIFOF)

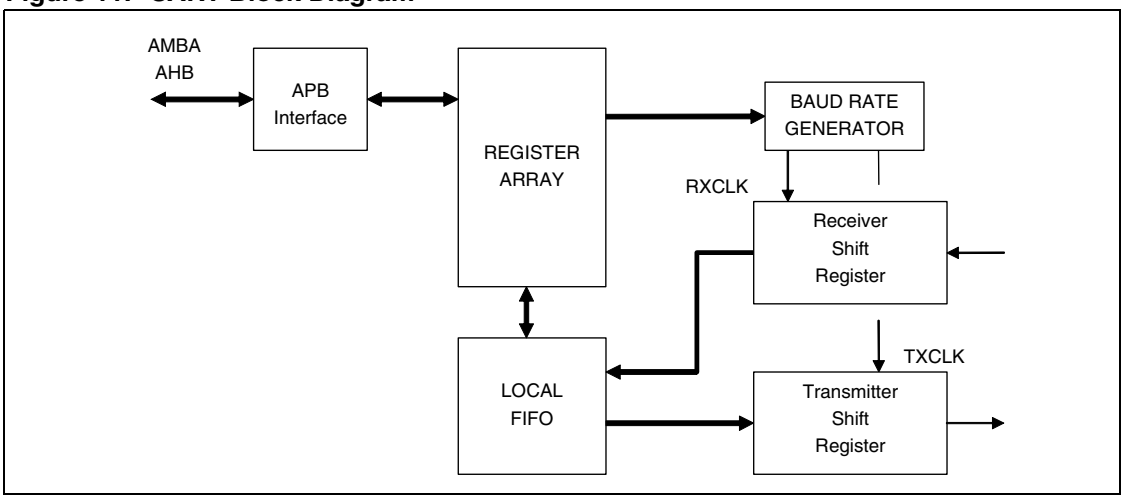
This bit indicates when the FIFO is full.

Bit [0] - FIFO Empty Status (FIFOE)

This bit indicates when the FIFO is empty.

6.5 UART Controller

Figure 11. UART Block Diagram



6.5.1 Overview

The UART provides a standard serial data communication with transmit and receive channels that can operate concurrently to handle a full-duplex operation.

Two internal FIFO for transmitted and received data, deep 16 and wide 8 bits, are present; these FIFO can be enabled or disabled through a register.

Interrupts are provided to control reception and transmission of serial data.

The clock for both transmit and receive channels is provided by an internal baud rate generator that divides the AHB System clock by any divisor value from 1 to 255. The output clock frequency of baud generator is sixteen times the baud rate value.

The maximum speed achieved is 115.000 baud.

6.5.2 Registers Map

Address	Register Name	Access
0x3000_1800	UART_BAUD_RATE	RW
0x3000_1804	UART_TX_BUFFER	WO
0x3000_1808	UART_RX_BUFFER	RO
0x3000_180C	UART_CONTROL	RW
0x3000_1810	UART_INT_ENABLE	RW
0x3000_1814	UART_STATUS	RO
0x3000_1818	UART_GUARD_TIME	RW
0x3000_181C	UART_TIME_OUT	RW
0x3000_1820	UART_TX_RESET	WO
0x3000_1824	UART_RX_RESET	WO

6.5.3 Register description

All the UART registers are 16 bit wide.

6.5.3.1 Baud Rate Generator Register

Mnemonic: UART_BAUD_RATE

Address: 0x3000_1800

Default value: 0x0001

Bit	Field Name	Access
15:00	BAUD_RATE	RW

BAUDE_RATE - This register select the UART baud rate according to the following formula: HCLK Frequency (48MHZ) / (UART_BAUD_RATE * 16)

Baud	Reg. value	Error
110	0x6A88	0.003%
150	0x4E20	0
300	0x2710	0
600	0x1388	0
1200	0x09C4	0
2400	0x04E2	0
4800	0x01A0	0
9600	0x0138	0.16%
19200	0x009C	0.16%
38400	0x004e	0.16%
115200	0x001A	0.16%

UART TX Buffer Register

6.5.3.2 Mnemonic: UART_TX_BUFFER

Address: 0x3000_1804

Default value: 0x0000

Bit	Field Name	Access
15:09	Reserved	RO
08:00	TX_DATA	WO

TX_DATA - Values written to this register will fill the 16 word tx FIFO. When the FIFO is disabled values written to this register will directly fill the transmitter shift register.

6.5.3.3 UART RX Buffer Register

Mnemonic: UART_RX_BUFFER

Address: 0x3000_1808

Default value: 0x0000

Bit	Field Name	Access
15:10	Reserved	RO
09:00	RX_DATA	RO

RX_DATA - Values read to this register will un-fill the 16 word RX FIFO. When the FIFO is disabled value read to this register will directly empty the RX shift register.

6.5.3.4 UART Control Register

Mnemonic: UART_CONTROL

Address: 0x3000_180C

Default value: 0x0000

Bit	Field Name	Access
15:11	Reserved	RO
10	FIFO_EN	RW
09	Reserved	RW
08	RX_EN	RW
07	RUN	RW
06	LOOP_EN	RW
05	PARITY	RW
04:03	STOP_BIT	RW
02:00	MODE	RW

FIFO_EN - When set to '1' enable the FIFO.

Reserved - This bit should be always set to '0'.

RX_EN - When set enable the RX channel.

RUN - When set to '1' enable the Bud Rate Generator.

Note: If this bit is reset both TX and RX channels are inactive.

LOOP_EN - When set to '1' enable the internal loop and the external TX and RX lines become inactive.

PARITY - Parity selection bit. '1' means "odd" parity (parity bit set on even number of '1's in data)

STOP_BIT - Select the number of stop bit added.

b4	b3	Stop Bit
0	0	0.5 Stop Bit
0	1	1 Stop Bit
1	0	1.5 Stop Bit
1	1	2 Stop Bit

MODE - Mode Control Field

b2	b1	b0	Mode
0	0	0	Reserved
0	0	1	8 bit Data
0	1	0	Reserved
0	1	1	7 bit Data + parity
1	0	0	9 bit Data
1	0	1	8 bit Data + Wake Up
1	1	0	Reserved
1	1	1	8 Bit Data + parity

6.5.3.5 UART Interrupt Enable Register

Mnemonic: UART_INT_EN

Address: 0x3000_1810

Default value: 0x0000

Bit	Field Name	Access
15:09	Reserved	RO
08	RX_HALF_FULL	RW
07	TIME_OUT_IDLE	RW
06	TIME_OUT_NOT_EMPTY	RW
05	OVERRUN_ERROR	RW
04	FRAME_ERROR	RW
03	PARITY_ERROR	RW
02	TX_HALT_EMPTY	RW
01	TX_BUFFER_EMPTY	RW
00	RX_BUFFER_FULL	RW

Each bit, when set to '1' will enable the corresponding interrupt while, when reset to '0' will mask it.

6.5.3.6 UART Status Register

Mnemonic: UART_STATUS

Address: 0x3000_1814

Default value: 0x0006

Bit	Field Name	Access
15:10	Reserved	RO
09	TX_FULL	RO
08	RX_HALF_FULL	RO
07	TIME_OUT_IDLE	RO
06	TIME_OUT_NOT_EMPTY	RO
05	OVERRUN_ERROR	RO
04	FRAME_ERROR	RO
03	PARITY_ERROR	RO
02	TX_HALT_EMPTY	RO
01	TX_BUFFER_EMPTY	RO
00	RX_BUFFER_FULL	RO

6.5.3.7 UART Guard Time Register

Mnemonic: UART_GUARD_TIME

Address: 0x3000_1818

Default value: 0x0000

Bit	Field Name	Access
15:08	Reserved	RO
07:00	GUARD_TIME	RW

GUARD_TIME - This register define the delay, in term of bit time from the last character transmitted and the assertion of TX_BUFFER_EMPTY.

6.5.3.8 UART Time Out Register

Mnemonic: UART_TIME_OUT

Address: 0x3000_181C

Default value: 0x0000

Bit	Field Name	Access
15:08	Reserved	RO
07:00	TIME_OUT	RW

TIME_OUT - This register set the time out value in term of bit time between two consecutive characters received.

6.5.3.9 UART TX Reset Register

Mnemonic: UART_TX_RESET

Address: 0x3000_1820

Default value: NA

Any value written to this address will reset the TX FIFO. This register is WO.

6.5.3.10 UART RX Reset Register

Mnemonic: UART_TX_RESET

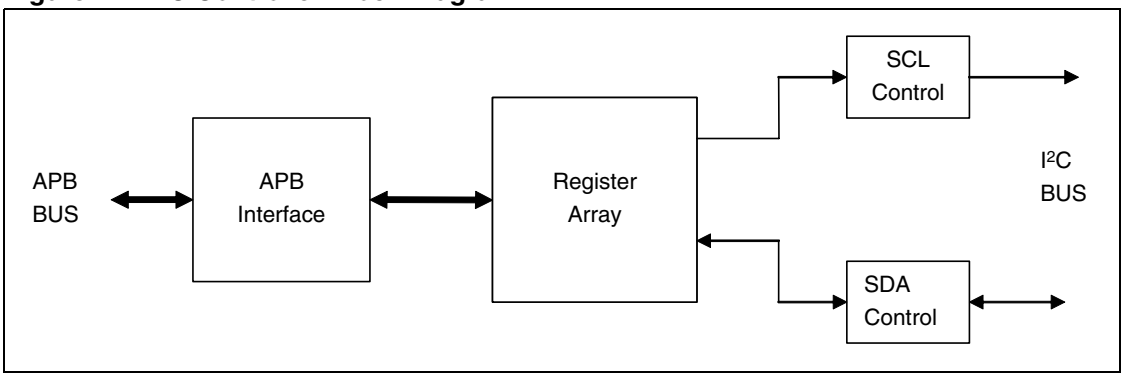
Address: 0x3000_1824

Default value: NA

Any value written to this address will reset the RX FIFO. This register is WO.

6.6 I²C Controller

Figure 12. I²C Controller Block Diagram



6.6.1 Overview

The controller serves as an interface between the ARM720 and the serial I²C bus. It provides multi-master functions, and controls all I2C bus-specific sequencing, protocol, arbitration and timing. It supports fast I²C mode (400 KHz).

Main Features:

- Parallel-bus/I²C protocol converter
- Multi-master capability
- 7-bit Addressing
- Transmitter/Receiver flag
- End-of-byte transmission flag
- Transfer problem detection
- Clock generation

- I^2C bus busy flag
- Arbitration Lost Flag
- End of byte transmission flag
- Transmitter/Receiver Flag
- Start bit detection flag
- Start and Stop generation

In addition to receiving and transmitting data, this interface converts it from serial to parallel format and vice versa, using either an interrupt or polled handshake. The interrupts can be enabled or disabled by software. The interface is connected to the I^2C bus by a data pin (SDAI) and by a clock pin (SCLI).

It can be connected both with a standard I^2C bus and a Fast I^2C bus. This selection is made by software.

Mode Selection

The interface can operate in the four following modes:

– Slave transmitter/receiver
– Master transmitter/receiver

By default, it operates in slave mode.

The interface automatically switches from slave to master after it generates a START condition and from master to slave in case of arbitration loss or a STOP generation, this allows Multi-Master capability.

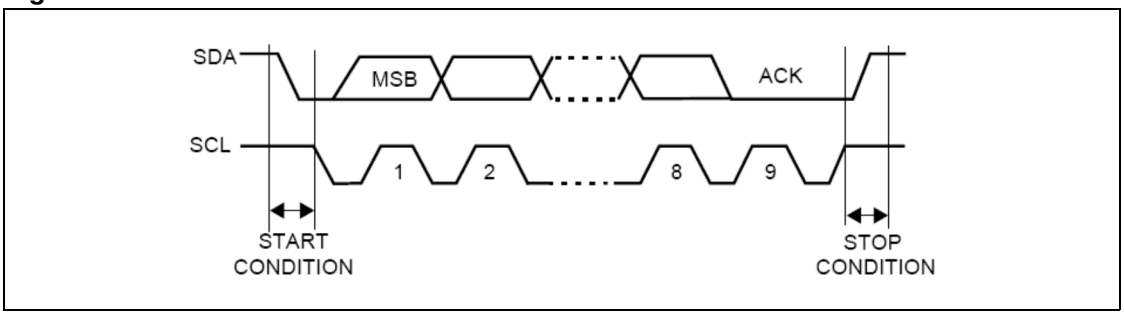
Communication Flow

In Master mode, it initiates a data transfer and generates the clock signal. A serial data transfer always begins with a start condition and ends with a stop condition. Both start and stop conditions are generated in master mode by software.

The first byte following the start condition is the address byte; it is always transmitted in Master mode.

A 9th clock pulse follows the 8 clock cycles of a byte transfer, during which the receiver must send an acknowledge bit to the transmitter.

Figure 13. I^2C Bus Protocol



Acknowledge may be enabled and disabled by software.

The I^2C interface address and/or general call address can be selected by software.

The speed of the I^2C interface may be selected between Standard (0 - 100 KHz) and Fast I^2C (100 - 400 KHz).

SDA/SCL Line Control

Transmitter mode: the interface holds the clock line low before transmission to wait for the microcontroller to write the byte in the Data Register.

Receiver mode: the interface holds the clock line low after reception to wait for the microcontroller to read the byte in the Data Register.

The SCL frequency (FSCL) is controlled by a programmable clock divider which depends on the I^2C bus mode.

When the I^2C cell is enabled, the SDA and SCL ports must be configured as floating open-drain output or floating input. In this case, the value of the external pull-up resistance used depends on the application.

6.6.2 Register Map

Address	Register Name	Access
0x3000_1560	CR	RW
0x3000_1564	SR1	RW
0x3000_1568	SR2	RW
0x3000_156C	CCR	RW
0x3000_1570	OAR	RW
0x3000_1574	Reserved	RO
0x3000_1578	DR	RW

6.6.3 Register Description

All the registers are 8 bit wide and aligned at 32 bit.

6.6.3.1 Control Register

Mnemonic: CR

Address: 0x3000_1560

Default value: 0x00

Bit	Field Name	Access
07:06	Reserved	RO
05	PE	RW
04	ENGC	RW
03	START	RW
02	ACK	RW
01	STOP	RW
00	ITE	RW

PE: Peripheral enable.

This bit is set and cleared by software.

0: Peripheral disabled

1: Master/Slave capability

Notes:

– When PE=0, all the bits of the CR register and the SR register except the Stop bit are reset. All outputs are released while PE=0

– When PE=1, the corresponding I/O pins are selected by hardware as alternate functions.

– To enable the I^2C interface, write the CR register TWICE with PE=1 as the first write only activates the interface (only PE is set).

ENGC: Enable general call.

This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0). The 00h General Call address is acknowledged (01h ignored).

0: General Call disabled

1: General Call enabled

START: Generation of a Start condition.

This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0) or when the Start condition is sent (with interrupt generation if ITE=1).

– In master mode:

0: No start generation

1: Repeated start generation

– In slave mode:

0: No start generation

1: Start generation when the bus is free

ACK: Acknowledge enable.

This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0).

0: No acknowledge returned.

1: Acknowledge returned after an address byte or a data byte is received.

STOP: Generation of a stop condition.

This bit is set and cleared by software. It is also cleared by hardware in master mode. Note: This bit is not cleared when the interface is disabled (PE=0).

– In master mode:

0: No stop generation.

1: Stop generation after the current byte transfer or after the current Start condition is sent. The STOP bit is cleared by hardware when the Stop condition is sent.

– In slave mode:

0: No stop generation.

1: Release the SCL and SDA lines after the current byte transfer (BTF=1). In this mode the STOP bit has to be cleared by software.

ITE: Interrupt enable.

This bit is set and cleared by software and cleared by hardware when the interface is disabled (PE=0).

0: Interrupts disabled

1: Interrupts enabled

6.6.3.2 Status Register 1

Mnemonic: SR1

Address: 0x3000_1564

Default value: 0x00

Bit	Field Name	Access
07	EVF	RO
06	Reserved	RO
05	TRA	RO
04	BUSY	RO
03	BTF	RO
02	ADSL	RO
01	M/SL	RO
00	SB	RO

EVF: Event flag.

This bit is set by hardware as soon as an event occurs.

It is cleared by software reading SR2 register in case of error event.

It is also cleared by hardware when the interface is disabled (PE=0).

0: No event

1: One of the following events has occurred:

- BTF=1 (Byte received or transmitted)
- ADSL=1 (Address matched in Slave mode while ACK=1)
- SB=1 (Start condition generated in Master mode)
- AF=1 (No acknowledge received after byte transmission)
- STOPF=1 (Stop condition detected in Slave mode)
- ARLO=1 (Arbitration lost in Master mode)
- BERR=1 (Bus error, misplaced Start or Stop condition detected)
- Address byte successfully transmitted in master mode.

TRA: Transmitter / Receiver

When BTF is set, TRA=1 if a data byte has been transmitted. It is cleared automatically when BTF is cleared. It is also cleared by hardware after detection of Stop condition (STOPF=1), loss of bus arbitration (ARLO=1) or when the interface is disabled (PE=0).

0: Data byte received (if BTF=1)

1: Data byte transmitted

BUSY: Bus busy.

This bit is set by hardware on detection of a Start condition and cleared by hardware on detection of a Stop condition. It indicates a communication in progress on the bus. This information is still updated when the interface is disabled (PE=0).

0: No communication on the bus

1: Communication ongoing on the bus

BTF: Byte transfer finished.

This bit is set by hardware as soon as a byte is correctly received or transmitted with interrupt generation if ITE=1. It is cleared by software reading SR1 register followed by a read or write of DR register. It is also cleared by hardware when the interface is disabled (PE=0).

- Following a byte transmission, this bit is set after reception of the acknowledge clock pulse. In case an address byte is sent, this bit is set only after the EV6 event. BTF is cleared by reading SR1 register followed by writing the next byte in DR register.
- Following a byte reception, this bit is set after transmission of the acknowledge clock pulse if ACK=1. BTF is cleared by reading SR1 register followed by reading the byte from DR register. The SCL line is held low while BTF=1.

0: Byte transfer not done

1: Byte transfer succeeded

ADSL: Address matched (Slave mode)

This bit is set by hardware as soon as the received slave address matched with the OAR register content or a general call is recognized. An interrupt is generated if ITE=1. It is cleared by software reading SR1 register or by hardware when the interface is disabled (PE=0). The SCL line is held low while ADSL=1.

0: Address mismatched or not received

1: Received address matched

M/SL: Master/Slave.

This bit is set by hardware as soon as the interface is in Master mode (writing START=1). It is cleared by hardware after detecting a Stop condition on the bus or a loss of arbitration (ARLO=1). It is also cleared when the interface is disabled (PE=0).

0: Slave mode

1: Master mode

SB: Start bit (Master mode).

This bit is set by hardware as soon as the Start condition is generated (following a write START=1). An interrupt is generated if ITE=1. It is cleared by software reading SR1 register followed by writing the address byte in DR register. It is also cleared by hardware when the interface is disabled (PE=0).

0: No Start condition

1: Start condition generated

6.6.3.3 Status Register 2

Mnemonic: SR2

Address: 0x3000_1568

Default value: 0x00

Bit	Field Name	Access
07:05	Reserved	RO
04	AF	RO
03	STOPF	RO
02	ARLO	RO
01	BERR	RO
00	GCAL	RO

AF: Acknowledge failure.

This bit is set by hardware when no acknowledge is returned. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0). The SCL line is not held low while AF=1.

0: No acknowledge failure

1: Acknowledge failure

STOPF: Stop detection (slave mode).

This bit is set by hardware when a Stop condition is detected on the bus after an acknowledge (if ACK=1). An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0). The SCL line is not held low while STOPF=1.

0: No Stop condition detected

1: Stop condition detected

ARLO: Arbitration lost.

This bit is set by hardware when the interface loses the arbitration of the bus to another master. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0). After an ARLO event the interface switches back automatically to Slave mode (M/SL=0). The SCL line is not held low while ARLO=1.

0: No arbitration lost detected

1: Arbitration lost detected

BERR: Bus error.

This bit is set by hardware when the interface detects a misplaced Start or Stop condition. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0). The SCL line is not held low while BERR=1.

0: No misplaced Start or Stop condition

1: Misplaced Start or Stop condition

GCAL: General Call (Slave mode).

This bit is set by hardware when a general call address is detected on the bus while ENGC=1. It is cleared by hardware detecting a Stop condition (STOPF=1) or when the interface is disabled (PE=0).

0: No general call address detected on bus

1: general call address detected on bus

6.6.3.4 Clock Control Register

Mnemonic: CCR

Address: 0x3000_156C

Default value: 0x00

Bit	Field Name	Access
07	SM/FM	RW
06:00	CC	RW

SM/FM: Fast/Standard I^2C mode.

This bit is set and cleared by software. It is not cleared when the interface is disabled (PE=0).

0: Standard I^2C mode

1: Fast I^2C mode

CC(6:0): 7-bit clock divider.

These bits select the speed of the bus (FSCL) depending on the I^2C mode. They are not cleared when the interface is disabled (PE=0).

Standard mode (FM/SM=0): FSCL <= 100kHz

FSCL = fCPU/(2x([CC6..CC0]+2))

Fast mode (FM/SM=1): FSCL > 100kHz

FSCL = fCPU/(3x([CC6..CC0]+2))

Note: The programmed FSCL assumes no load on SCL and SDA lines.

6.6.3.5 Own Address Register

Mnemonic: OAR

Address: 0x3000_1570

Default value: 0x00

Bit	Field Name	Access
07:01	ADD	RW
00	DIR	RW

ADD(7:1): Interface address.

These bits define the I^2C bus address of the interface. They are not cleared when the interface is disabled (PE=0).

DIR: Address direction bit.

This bit is don't care, the interface acknowledges either 0 or 1. It is not cleared when the interface is disabled (PE=0).

Note: Address 01h is always ignored.

6.6.3.6 Data Register.

Mnemonic: DR

Address: 0x3000_1578

Default value: 0x00

Bit	Field Name	Access
07:00	D	RW

D(7:0): 8 bit data register.

These bits contain the byte received or to be transmitted on the bus.

Transmitter mode: Byte transmission start automatically when the software writes in the DR register.

Receiver mode: the first data byte is received automatically in the DR register using the least significant bit of the address.

Then, the next data bytes are received one by one after reading the DR register.

6.6.4 I^2C Functional description

By default the I^2C interface operates in Slave mode (M/SL bit is cleared) except when it initiates a transmit or receive sequence.

6.6.4.1 Slave Mode

As soon as a start condition is detected, the address is received from the SDA line and sent to the shift register; then it is compared with the address of the interface or the General Call address (if selected by software).

Address not matched: the interface ignores it and waits for another Start condition.

Address matched: the interface generates in sequence:

- Acknowledge pulse if the ACK bit is set.
- EVF and ADSL bits are set with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register, holding the SCL line low (see Figure 8 Transfer sequencing EV1).

Next, read the DR register to determine from the least significant bit if the slave must enter Receiver or Transmitter mode.

Slave Receiver

Following the address reception and after SR1 register has been read, the slave receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

- Acknowledge pulse if the ACK bit is set
- EVF and BTF bits are set with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register followed by a read of the DR register, holding the SCL line low (see Figure 8) Transfer sequencing EV2).

Slave Transmitter

Following the address reception and after SR1 register has been read, the slave sends bytes from the DR register to the SDA line via the internal shift register.

The slave waits for a read of the SR1 register followed by a write in the DR register, holding the SCL line low (see Figure 8 Transfer sequencing EV3).

When the acknowledge pulse is received:

- The EVF and BTF bits are set by hardware with an interrupt if the ITE bit is set.

Closing slave communication

After the last data byte is transferred a Stop Condition is generated by the master. The interface detects this condition and sets:

– EVF and STOPF bits with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR2 register (see Figure 3 Transfer sequencing EV4).

Error Cases

- BERR: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and the BERR bits are set with an interrupt if the ITE bit is set.

If it is a Stop then the interface discards the data, released the lines and waits for another Start condition.

If it is a Start then the interface discards the data and waits for the next slave address on the bus.

- AF: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set with an interrupt if the ITE bit is set.

Note: In both cases, SCL line is not held low; however, SDA line can remain low due to possible "0" bits transmitted last. It is then necessary to release both lines by software.

How to release the SDA / SCL lines

Set and subsequently clear the STOP bit while BTF is set. The SDA/SCL lines are released after the transfer of the current byte.

6.6.4.2 Master Mode

To switch from default Slave mode to Master mode a Start condition generation is needed.

Start condition and Transmit Slave address

Setting the START bit while the BUSY bit is cleared causes the interface to switch to Master mode (M/SL bit set) and generates a Start condition.

Once the Start condition is sent:

– The EVF and SB bits are set by hardware with an interrupt if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the DR register with the Slave address byte, holding the SCL line low (see Figure 8 Transfer sequencing EV5).

Then the slave address byte is sent to the SDA line via the internal shift register.

After completion of this transfers (and acknowledge from the slave if the ACK bit is set):

– The EVF bit is set by hardware with interrupt generation if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the CR register (for example set PE bit), holding the SCL line low (see Figure 8 Transfer sequencing EV6).

Next the master must enter Receiver or Transmitter mode.

Master Receiver

Following the address transmission and after SR1 and CR registers have been accessed, the master receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

– Acknowledge pulse if if the ACK bit is set

– EVF and BTF bits are set by hardware with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register followed by a read of the DR register, holding the SCL line low (see Figure 8 Transfer sequencing EV7).

To close the communication: before reading the last byte from the DR register, set the STOP bit to generate the Stop condition. The interface goes automatically back to slave mode (M/SL bit cleared).

Note: In order to generate the non-acknowledge pulse after the last received data byte, the ACK bit must be cleared just before reading the second last data byte.

Master Transmitter

Following the address transmission and after SR1 register has been read, the master sends bytes from the DR register to the SDA line via the internal shift register.

The master waits for a read of the SR1 register followed by a write in the DR register, holding the SCL line low (see Figure 8 Transfer sequencing EV8).

When the acknowledge bit is received, the interface sets:

– EVF and BTF bits with an interrupt if the ITE bit is set.

To close the communication: after writing the last byte to the DR register, set the STOP bit to generate the Stop condition. The interface goes automatically back to slave mode (M/SL bit cleared).

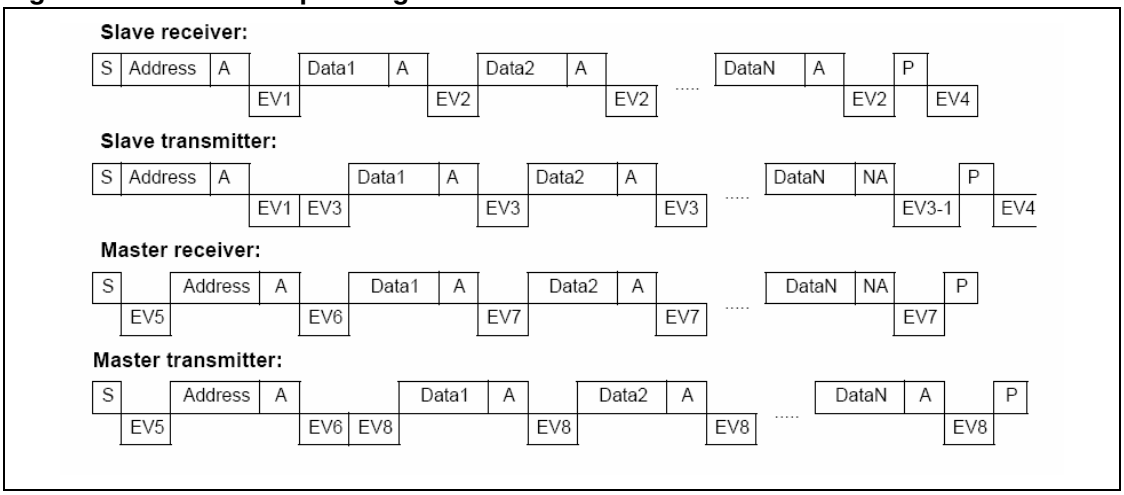
Error Cases

- BERR: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and BERR bits are set by hardware with an interrupt if ITE is set.
- AF: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set by hardware with an interrupt if the ITE bit is set. To resume, set the START or STOP bit.
- ARLO: Detection of an arbitration lost condition.

In this case the ARLO bit is set by hardware (with an interrupt if the ITE bit is set and the interface goes automatically back to slave mode (the M/SL bit is cleared).

Note: In all these cases, the SCL line is not held low; however, the SDA line can remain low due to possible "0" bits transmitted last. It is then necessary to release both lines by software.

Figure 14. Transfer sequencing



Legend:

S=Start, P=Stop, A=Acknowledge, NA=Non-acknowledge

EVx=Event (with interrupt if ITE=1)

EV1: EVF=1, ADSL=1, cleared by reading SR1 register.

EV2: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.

EV3: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.

EV3-1: EVF=1, AF=1, BTF=1; AF is cleared by reading SR1 register. BTF is cleared by releasing the ines (STOP=1, STOP=0) or by writing DR register (DR=FFh). Note: If lines are released by STOP=1, STOP=0, the subsequent EV4 is not seen.

EV4: EVF=1, STOPF=1, cleared by reading SR2 register.

EV5: EVF=1, SB=1, cleared by reading SR1 register followed by writing DR register.

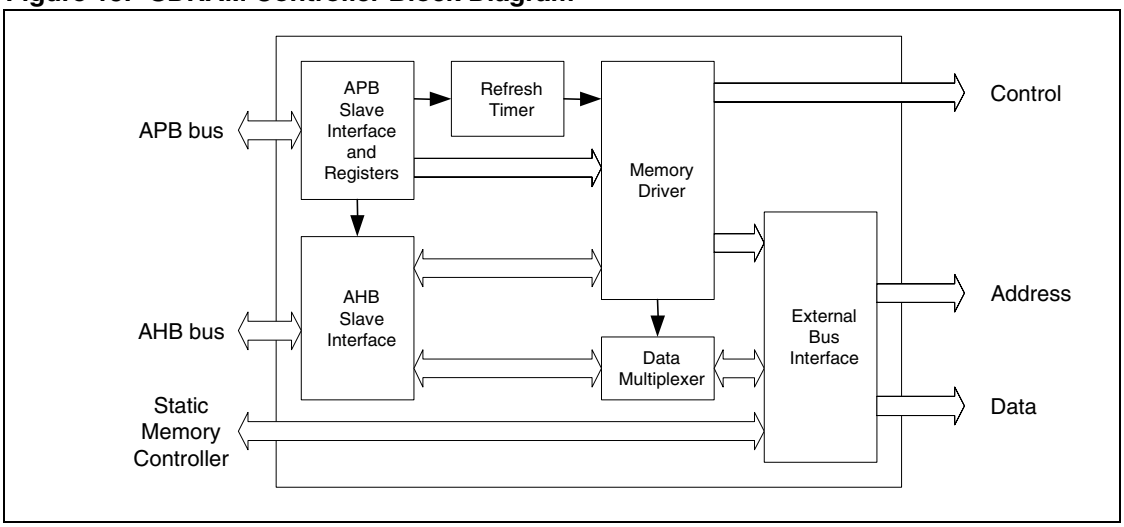
EV6: EVF=1, cleared by reading SR1 register followed by writing CR register (for example PE=1).

EV7: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.

EV8: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.

6.7 Dynamic Memory Controller

Figure 15. SDRAM Controller Block Diagram



6.7.1 Overview

The DRAM Controller block is an AHB slave used to provide an interface between the system bus and external memory device. The controller supports four external banks, containing either SDRAM or EDO memories. Memory components with 32, 16 and 8 bit data buses are supported.

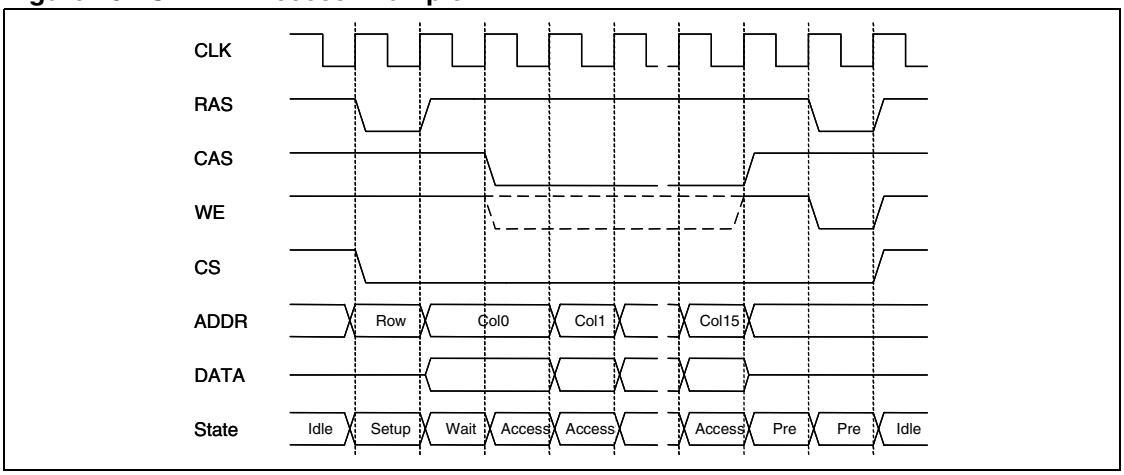
6.7.2 Memory Access

The DRAM Controller supports single and burst accesses for both memory types. A single state machine and shared control signals are used for a memory access.

6.7.2.1 SDRAM Access

SDRAM accesses are performed with the pre-charge command at the end of each access, i.e. banks are not kept open, as shown in *Figure 16*.

Figure 16. SDRAM Access Example



Initially the row address is output and the MISA (RAS) and MICS outputs are asserted. During the Access state the column address is output on the address bus and the MIAA (CAS) output is asserted. When the memory access has completed the MIAA and MIWE outputs are asserted to pre-charge the accessed SDRAM bank.

The SDRAM data accesses are not performed as a single burst. In each access cycle a new read/write command is issued to the SDRAM device, causing the column address to reload. To support this mode of operation, the SDRAM device must be configured to have a burst length of one. Using a number of individual accesses allows the same location to be accessed in a number of consecutive access cycles e.g. during a byte burst access to a word wide memory bank.

A SDRAM access is not permitted to cross a 512 byte boundary (the minimum supported SDRAM row size) without passing through the SETUP cycle to update the SDRAM row address. After the last access cycle of every transfer, the pre-charge command must be issued to the SDRAM.

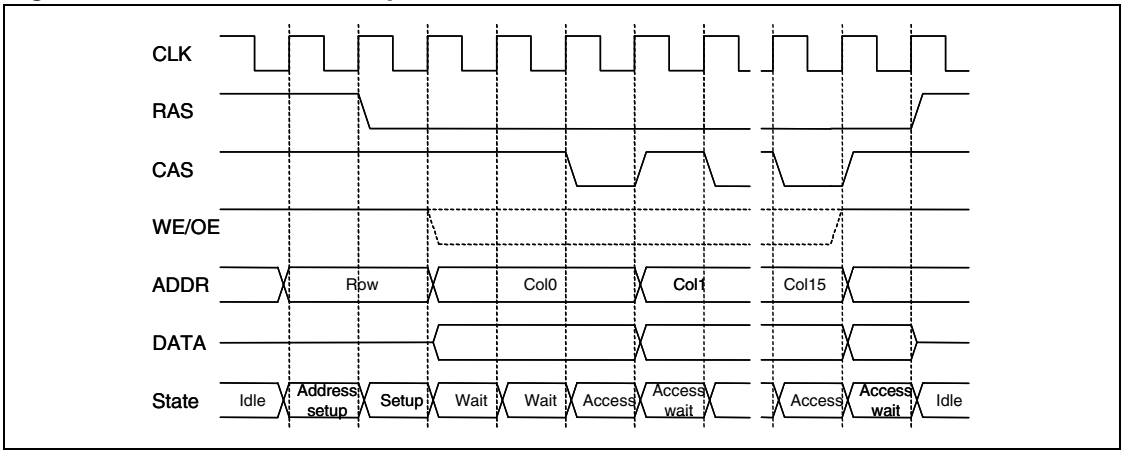
A feature of SDRAM read accesses is that there is some latency between the read access being initiated and the data being returned by the device. The data latency from the read command to the first valid data must be configured by the user.

The memory interface is compatible with the Intel "PC SDRAM Specification" and only uses commands defined in that specification.

6.7.2.2 EDO Access

An EDO access is shown in *Figure 17*.

Figure 17. EDO Access Example



To support EDO accesses an ADDRESS SETUP cycle has been added to provide a cycle of setup between the Row address output becoming valid and the RAS signal being activated. Also added is an ACCESS WAIT cycle, after each ACCESS cycle.

The MICS output is used as the EDO RAS signal.

The DRAM Controller must capture read data on the clock following the read access. The user must configure the single cycle data latency.

6.7.3 Address Mapping Table

Table 14. DRAM Address Bus

Memory Data Width	Address	Column Address Width	Memory Interface Output Address Bits														AHB bus address
			13	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
8 bits	Column		**ba**	**ba**	**ba**	0	9	8	7	6	5	4	3	2	1	0	
	Row	9	22	21	20	19	18	17	16	15	14	13	12	11	10	9	
		10	X	22	21	20	19	18	17	16	15	14	13	12	11	10	
16 bits	Column		**ba**	**ba**	**ba**	0	10	9	8	7	6	5	4	3	2	1	
	Row	8	22	21	20	19	18	17	16	15	14	13	12	11	10	9	
		9	X	22	21	20	19	18	17	16	15	14	13	12	11	10	
		10	x	x	22	21	20	19	18	17	16	15	14	13	12	11	
32 bits	Column		**ba**	**ba**	**ba**	0	11	10	9	8	7	6	5	4	3	2	
	Row	8	x	22	21	20	19	18	17	16	15	14	13	12	11	10	
		9	x	x	22	21	20	19	18	17	16	15	14	13	12	11	
		10	x	x	x	22	21	20	19	18	17	16	15	14	13	12	

6.7.4 External Bus Interface

The External Bus Interface is used to share the address and data pads (pins) between the DRAM Controller and the Static Memory Controller.

The handshaking between the EBI and the memory controller consists of a two-wire interface, BusReq and BusGnt, both signals are active high. BusReq is asserted by the memory controller indicating that it requires access to the bus to perform a transfer. When the EBI sees a BusReq it responds with a BusGnt active when the bus is available and can be granted to the memory controller.

The EBI will wait for the currently granted memory controller to finish the access (i.e. to de-assert BusReq) before it grants the next memory controller. Therefore the memory controller must keep the BusReq signal asserted as long as it requires the memory bus. The arbitration uses a fixed priority scheme.

6.7.5 Register MAP

Address	Register Name	Access
0x3000_2800	MB1Config	R/W
0x3000_2804	MB2Config	R/W
0x3000_2808	MB3Config	R/W
0x3000_280C	MB4Config	R/W
0x3000_2810	SDRAM1ConfigLo	RO
0x3000_2814	SDRAM1ConfigHi	RO
0x3000_2818	SDRAM2ConfigLo	RO
0x3000_281C	SDRAM2ConfigHi	RO
0x3000_2820	SDRAM3ConfigLo	RO
0x3000_2824	SDRAM3ConfigHi	RO
0x3000_2828	SDRAM4ConfigLo	RO
0x3000_282C	SDRAM4ConfigHi	RO
0x3000_2830	MemConfig	R/W
0x3000_2834	Bank 1 Size	R/W
0x3000_2838	Bank 2 Size	R/W
0x3000_283C	Bank 3 Size	R/W
0x3000_2840	Bank 4 Size	R/W

6.7.6 Register Description

All the registers are 16 bit wide.

6.7.6.1 Memory Bank Configuration Register

Mnemonic: MB1Config, MB2Config, MB3Config, MB4Config

Address: 0x3000_2800, 0x3000_2804, 0x3000_2808, 0x3000_280C

Default value: 0x0000

Bit	Field Name	Access
15:12	Reserved	RO
11:10	DEV_WID	RW
09:08	DATA_LAT	RW
07:05	SETUP_TIME	RW
04:02	IDLE_TIME	RW
01:00	SDRAM_COL	RW

DEV_WID - Define the data width of the external memory device:

00 Byte (8 bit)

01 Half Word (16 bit)

10 Word (32 bit)

11 Reserved

DATA_LAT - Defines the number of memory clock cycles between the start of a memory read access and the first valid data. The DATALAT value is valid between 0 and 3.

SETUP_TIME - Defines the number of memory clock cycles the memory drivers spends in the DECODE state before accessing the external memory.

The SETUPTIME value is valid between 0 and 7.

IDLE_TIME - Defines the minimum time the memory driver must spend in the IDLE state following memory accesses. The value defines the number of Memory Clock cycles.

The IDLETIME value is valid between 0 and 7.

SDRAM_COL - Specifies the width of the SDRAM column address:

00 8 bits

01 9 bits

10 10 bits

11 Reserved

6.7.6.2 SDRAM Configuration Registers

These registers are write only. A write access to the high registers will start the SDRAM configuration cycle, during which the value written to the register will be asserted on the memory bus for a one clock period.

After the power-up the CPU must configure each SDRAM device, i.e. perform precharge-refresh-mode register set procedure as specified in the SDRAM device data sheet.

Mnemonic: SDRAM1ConfigLo, SDRAM2ConfigLo, SDRAM3ConfigLo, SDRAM4ConfigLo,

Address: 0x3000_2810, 0x3000_2818, 0x3000_2820, 0x3000_2828

Default value: 0x0000

Bit	Field Name	Access
15:14	Reserved	RO
13:00	MIAB	WO

Mnemonic: SDRAM1ConfigHi, SDRAM2ConfigHi, SDRAM3ConfigHi, SDRAM4ConfigHi

Address: 0x3000_2814, 0x3000_281C, 0x3000_2824, 0x3000_282C

Default value: 0x0000

Bit	Field Name	Access
15:03	Reserved	RO
02	MIWE	WO
01	MIAA	WO
02	MISA	WO

MIAB - Memory Interface Address Bus

MIWE - Memory Interface Write Enable

MIAA - Memory Interface Access Active (nCAS)

MISA - Memory Interface Setup Active (nRAS)

6.7.6.3 Memory Configuration Register

Mnemonic: MEM_CONFIG

Address: 0x3000_2830

Default value: 0x0000

Bit	Field Name	Access
15:14	Reserved	RO
13	PWR_SAVE	RW
12	TYPE	RW
11	B3EN	RW
10	B2EN	RW
09	B1EN	RW
08	B0EN	RW
07:00	REFR	RW

PWR_SAVE - When set to '1', the following refresh cycle will put the memory in "Self Refresh" mode. The memory will exit the Self-refresh mode when this bit is reset to '0'.

Type - Define the memory type connected.

1 - SDRAM

0 - EDO

Note: The four banks must be populated with the same type of memory.

BxEN - The bank enable bits are used to enable each bank separately. If an AHB transfer is accessing a disabled bank, the DRAM Controller will return the error response to the AHB master.

1 - Enable

0 - Disable

REFR - This value is used to determine the refresh period. The period can be set in the 1 us steps.

REFR	Refresh Period
00000000	Refresh is disabled
00000001	Refresh period is 1us
00000010	Refresh period is 2us
…	
11111111	Refresh period is 255us

6.7.6.4 Bank Size Registers

Mnemonic: Bank Size 0, Bank Size 1, Bank Size 2, Bank Size 3

Address: 0x3000_2834, 0x3000_2838, ox3000_283C, 0x3000_2840

Default value: 0x0000

There are four registers of 16 bits each that specify, for each bank, its size. The size is defined parametrically as 2^{STEP_SIZE} in our case 64Kbytes steps, but it could be larger by increasing STEP_SIZE in the Dram Package, also smaller step size are possible however STEP_SIZE can not be smaller than 10.

For 64Kbytes size the Bank[3:0]Size registers have the organization below:

Table 15. Memory Bank Size Register

15 9	8 0
Unused	Size

Where (Size+1) represents the size of the bank in 64Kbytes steps, that is:

Table 16. Bank size field and its corresponding actual size

Size field (binary)	Bank actual size
0_0000_0000	1 x 64Kbytes = 64Kbytes
0_0000_0001	2 x 64Kbytes = 128Kbytes
0_0000_0010	3 x 64Kbytes = 192Kbytes
. . .	. . .
0_0000_1111	16 x 64Kbytes = 1Mbytes
. . .	. . .
0_0001_1111	32 x 64Kbytes = 2Mbytes
. . .	. . .
0_0011_1111	64 x 64Kbytes = 4Mbytes
. . .	. . .
0_0111_1111	128 x 64Kbytes = 8Mbytes
. . .	. . .
0_1111_1111	256 x 64Kbytes = 16Mbytes
. . .	. . .
1_1111_1111	512 x 64Kbytes = 32Mbytes

6.8 Static Memory Controller

6.8.1 SRAMC description

The block is designed to control the data flow from the internal AMBA (Advanced Micro-controller Bus Architecture) AHB (Advanced High-Performance Bus) bus to any static memory components (ROM, SRAM and FLASH).

Length, setup and hold timings required to properly transfer data to external memory components are controllable by programming of internal registers, dedicated to each external memory space as is the external bus width. Each internal register also contains a flag bit which informs the controller if a particular memory space is accessible or not. SRAMC receives the HSEL from The AHB decoder. The block contains 4 registers which are used to control 4 external regions:

- 2 (Region 0 and 1) with programmable contiguous size for static memory
- 2 (Region 2 and 3) for External I/O with fixed size

6.8.2 Registers Map and description

Address	Register Name	Access
0x3000_2400	REG0	R/W
0x3000_2404	REG1	R/W
0x3000_2408	REG2	R/W
0x3000_240C	REG3	R/W

All the registers are 16 bit wide.

6.8.2.1 Region 0 Control Register

Mnemonic: REG0

Address: 0x3000_2400

Default value: 0b000_0000_0011_11xx

Bit	Field Name	Access
15:08	SIZE	RW
07:06	Reserved	RO
05	ENABLE	RW
04:02	LENGTH	RW
01:00	SIZE	R0

SIZE - Define the region size in steps of 64 Kb:

0x00 = 64 Kb

0x01 = 128 Kb

0FF = 16 Mb

ENABLE - This bit, when set to '1' and the CPU access this region, an external cycle is performed with the proper length as defined in the next field. If the CPU access the region when the ENABLE bit is reset the external cycle is not performed and an Error is sent to the CPU.

LENGTH - This field define the strobe pulse width in term of clock cycle.

SIZE - This field define the data bus size to be used for this region.

00 = 8 bit

01 = 16 bit

10 = 32 bit

11 = Reserved

This field latch its contents during the rising edge of nRESET through two external address line (See configuration register for more detail).

6.8.2.2 Region 1 Control Register

Mnemonic: REG1

Address: 0x3000_2404

Default value: 0x001D

The fields are exactly the same of the previous register. The unique difference is that in this register also the SIZE field is programmable by CPU.

Note: when B_SIZE differs from "00" then the external address bus shift right by1 bit: eadd[31:0]<="0" & int_eaddr[31:1].
When B_SIZE = "00" there isn't left shift: eadd[31:0]<= int_eaddr[31:0]

6.8.2.3 Region 2 and 3 Control Registers

Mnemonic: REG2, REG3

Address: 0x3000_2408, 0x3000_240C

Default value: 0x001D

Bit	Field Name	Access
15:12	Reserved	RO
11:09	WE_SETUP	RW
08:06	WE_HOLD	RW
05	ENABLE	RW
04:02	LENGTH	RW
01:00	SIZE	RW

WE_SETUP - This field define the setup time of Write Enable assertion with respect to Chip Select assertion in term of clock cycles.

Note: the whole data access cycle (both for read AND write) is:
CS_LENGTH =WE_SETUP+WE_LENGHT+WE_HOLD+2

WE_HOLD - These 3 bits control the hold time of Write Enable de-assertion with respect to Chip Select de-assertion. The real WE hold time is WE_HOLD +1 CLK

ENABLE - When set to '0' prevents assertion of the chip select signal for the regions controlled by these register.

LENGTH - These 3 bits control the length of time that an access to the external memory region controlled by these register will take.

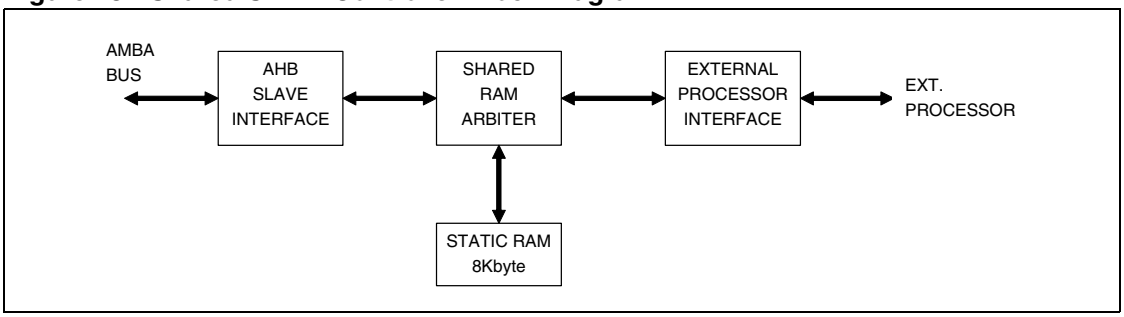
SIZE - Define the effective external bus size for an access to these memory regions.

- 00 = 8 bit
- 01 = 16 bit
- 10 = 32 bit
- 11 = NA

Note: ***Note****: "11" is not defined and no error condition exists for this case so responsibility lies with the programmer to ensure this never occurs.*

6.9 Shared SRAM Controller

Figure 18. Shared SRAM Controller Block Diagram



6.9.1 Overview

This block is based on 2 sub blocks:

- 8Kbyte static RAM
- Controller to manage access to the ram

Access to the ram is driven by a round robin arbiter, which receives requests from two agents:

- AHB master, i.e. internal CPU or DMA (by AHB bus)
- External processor (by asynchronous external bus).

The 2 agents access the shared ram as a standard register located on its own address space, belonging to its proper bus (AHB and external asynchronous bus).

Shared Ram access is synchronized to the 2 bus timings by means of the WAIT and nXPWAIT signals. If the access cannot be done immediately, these 2 signals put in wait state either the AHB bus agent or the external bus agent, respectively, until the action can be accomplished.

In other words, request of accessing the ram is done by simply accessing the addresses reserved to the shared ram itself, and the grant condition is indirectly flagged by either the nXPWAIT or WAIT signal de-assertion.

The interrupt signal is used to perform the handshake between PNCU and external processor.

The output of this register is compared to the byte: "55h"; if the content of the register is equal to 55h, the interrupt is set low.

AHB agents can assert or de-assert interrupt to the external CPU by reading or writing the interrupt register inside the shared ram while the external CPU can only read or clear this register.

Table 17. Register MAP and Description

Address	Register Name	Access
0x2100_0000 ~ 0x2100_1FFF	SHRAM	RW
0x2100_2000	IRR	R/W

6.9.1.1 8 KB Shared SRAM

Mnemonic: SHRAM

Address: 0x2100_0000 ~ 0x2100_1FFF

Default value: unpredictable

6.9.1.2 Interrupt Request Register

Mnemonic: IRR

Address: 0x2100_2000

Default value: 0xFFFF_FFFF

Bit	Field Name	AHB Agent Access	Ext. Processor Access
31:08	Reserved	RO	RO
07:00	IRR	RW	RC

Access by AHB Agent (CPU or DMA)

AHB agent can access this register for reading or writing. No interrupt versus AHB interrupt controller is generated.

Writing 55H will set nXPIRQ interrupt external request active (nXPIRQ='0'). External interrupt request become inactive (nXPIRQ='1') when IRR contents differs from 55H.

Access by external processor

XCPU can access this register either for reading or clear. Writing access to this address (A(13)='1', A(12:0) = don't care) resets the interrupt (IRR = FFH), regardless of the contents of the XPWDATA data bus.

6.9.2 External Processor Timings

Figure 19. SPEAr Net Write Timing Diagram

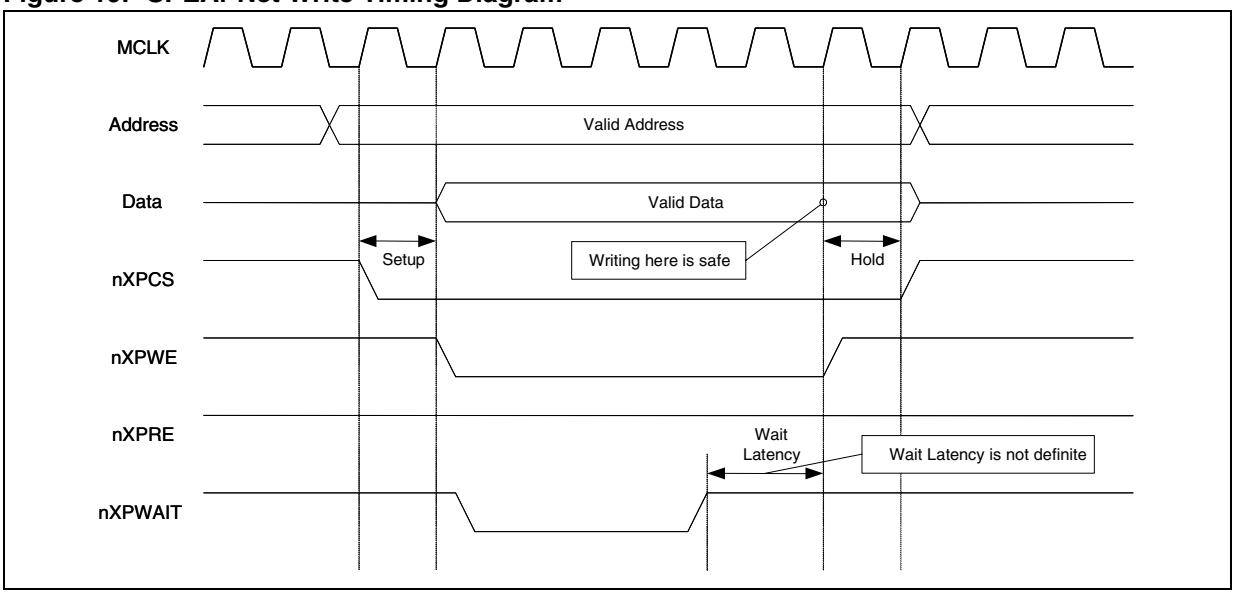
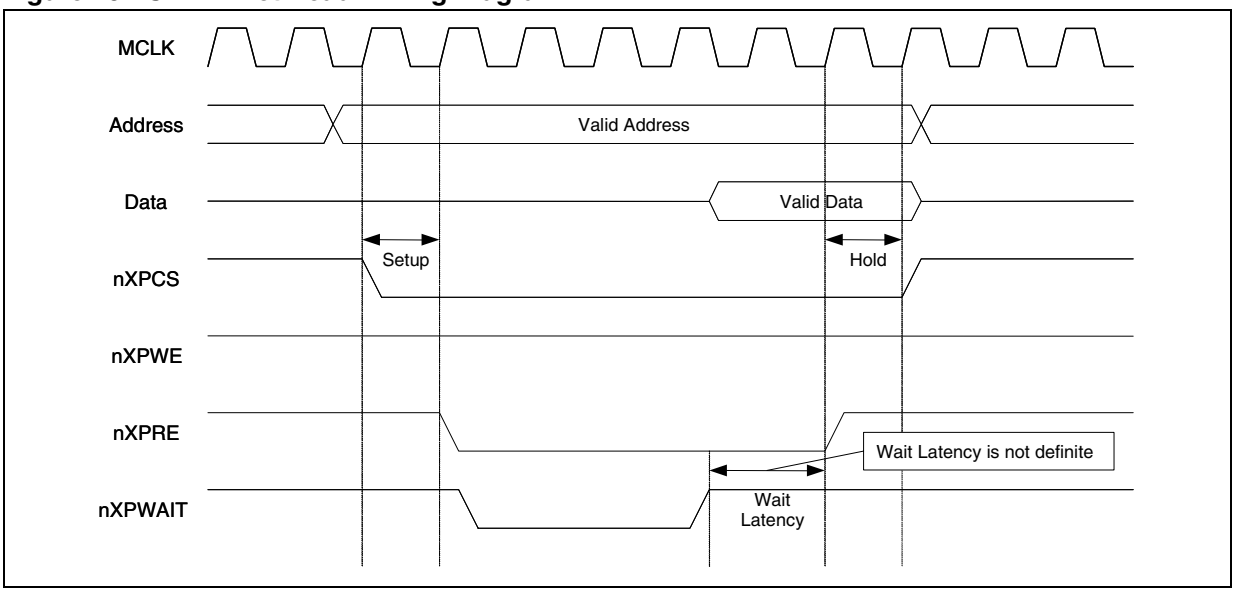


Figure 20. SPEAr Net Read Timing Diagram

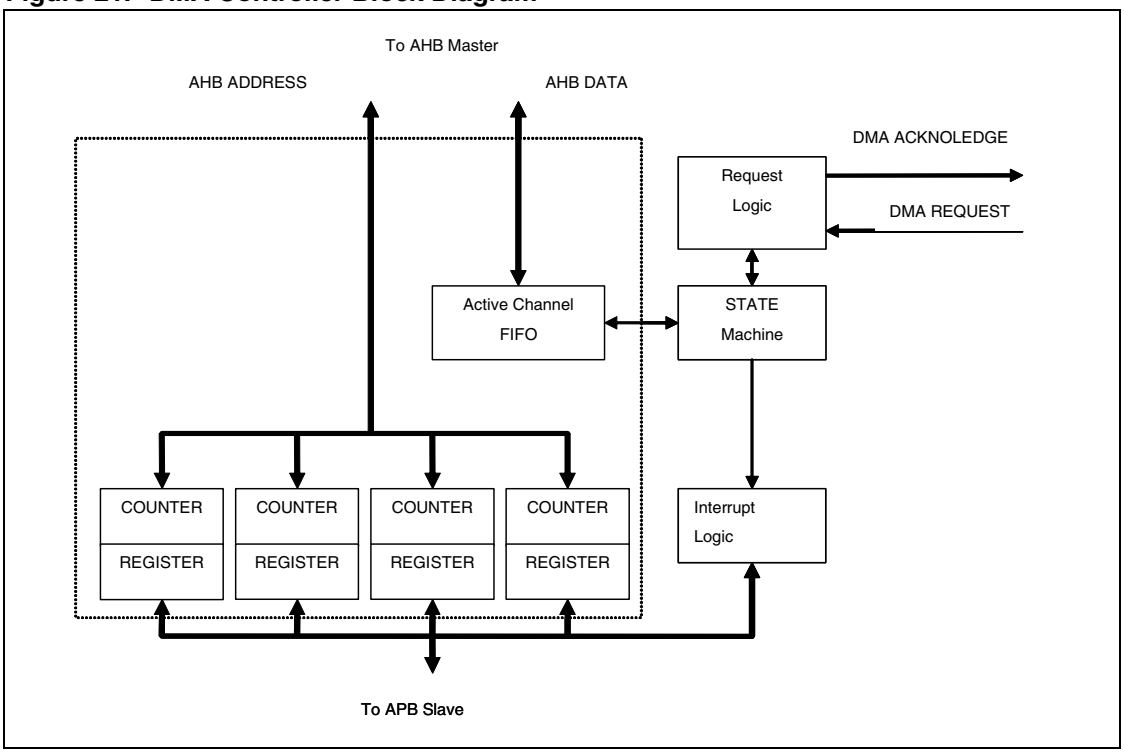


SPEAr Net specific requirements are as follows.

1. External processor and SPEAr Net are not synchronized so MCLK of timing diagram is not absolute.
2. For safe handshaking, nXPWAIT should be asserted immediately after nXPCS assertion.
3. The wait latency is not defined so when read access it is safe to hold data output until nXPCS will be de-asserted.

6.10 DMA Controller

Figure 21. DMA Controller Block Diagram



6.10.1 Overview

The DMA block has one DMA channel and it's capable of servicing up to four data stream.

Two channels are offering memory to memory transfer capability while the other two channels are only able to move data from I/O to memory or memory to I/O. The main differences between the two modes is that in case of memory to memory transfer both the addresses (source and destination) are incremented during the transfer while in case of I/O to memory only the destination address in incremented.

The three channels are:

- CH0: External request 0
- CH1: External request 1
- CH2: Memory to Memory
- CH3: Memory to Memory

The priority between channels is fixed (CH0 has the highest and CH3 the lowest).

6.10.2 Register Map

Channel	Address	Register Name	Access
CH0	0x3000_2000	DMASourceLow	RW
	0x3000_2004	DMASourceHigh	RW
	0x3000_2008	DMADestLo	RW
	0x3000_200C	DMADestHigh	RW
	0x3000_2010	DMAMax	RW
	0x3000_2014	DMACtrl	RW
	0x3000_2018	DMASoCurrLo	RO
	0x3000_201C	DMASoCurrHigh	RO
	0x3000_2020	DMADeCurrLo	RO
	0x3000_2024	DMADeCurrHigh	RO
	0x3000_2028	DMATCnt	RO
CH1	0x3000_2040	DMASourceLow	RW
	0x3000_2044	DMASourceHigh	RW
	0x3000_2048	DMADestLo	RW
	0x3000_204C	DMADestHigh	RW
	0x3000_2050	DMAMax	RW
	0x3000_2054	DMACtrl	RW
	0x3000_2058	DMASoCurrLo	RO
	0x3000_205C	DMASoCurrHigh	RO
	0x3000_2060	DMADeCurrLo	RO
	0x3000_2064	DMADeCurrHigh	RO
	0x3000_2068	DMATCnt	RO
CH2	0x3000_2080	DMASourceLow	RW
	0x3000_2084	DMASourceHigh	RW
	0x3000_2088	DMADestLo	RW
	0x3000_208C	DMADestHigh	RW
	0x3000_2090	DMAMax	RW
	0x3000_2094	DMACtrl	RW
	0x3000_2098	DMASoCurrLo	RO
	0x3000_209C	DMASoCurrHigh	RO
	0x3000_20A0	DMADeCurrLo	RO
	0x3000_20A4	DMADeCurrHigh	RO
	0x3000_20A8	DMATCnt	RO

Channel	Address	Register Name	Access
CH3	0x3000_20C0	DMASourceLow	RW
	0x3000_20C4	DMASourceHigh	RW
	0x3000_20C8	DMADestLo	RW
	0x3000_20CC	DMADestHigh	RW
	0x3000_20C0	DMAMax	RW
	0x3000_20C4	DMACtrl	RW
	0x3000_20C8	DMASoCurrLo	RO
	0x3000_20CC	DMASoCurrHigh	RO
	0x3000_20C0	DMADeCurrLo	RO
	0x3000_20C4	DMADeCurrHigh	RO
	0x3000_20C8	DMATCnt	RO
Common to all channels	0X3000_20F0	DMAMask	RW
	0X3000_20F4	DMAClr	WO
	0X3000_20F8	DMAStatus	RO

6.10.3 Registers Description

6.10.3.1 DMA Source Registers

Mnemonic: DMASourceLow, DMASourceHigh

These two 16 bit registers contain the 32 bit source base address for the next DMA transfer. When the DMA Controller is enabled, the content of the Source Base Address Registers are loaded in the Current Source Address Registers.

6.10.3.2 DMA Destination Registers

Mnemonic: DMADestLow, DMADestHigh

These two 16 bit registers contain the 32 bit destination base address for the next DMA transfer. When the DMA Controller is enabled, the content of the Destination Base Address Registers are loaded in the Current Destination Address Registers.

6.10.3.3 DMA Maximum Count Register

Mnemonic: DMAMax

This register is programmed with the maximum data unit count of the next DMA transfer. The data unit is equal to the source to DMA data width (byte, half-word or word).

When the DMA Controller is enabled, the content of the Maximum Count Register is loaded in the Terminal Count Register.

6.10.3.4 DMA Control Register

Mnemonic: DMAMax

Default value: 0x0000

Bit	Field Name	Access
15:14	Reserved	RO
13	Dir	RW
12	Reserved	RO
11	Mem2Mem	RW
10:09	Reserved	RO
08:07	DeSize	RW
06:05	SoBurst	RW
04:03	SoSize	RW
02	DeInc	RW
01	SoInc	RW
00	Enable	RW

Dir: This bit defines the direction for the next transfer

- 0: Peripheral to memory.
- 1: Memory to peripheral.

Mem2Mem: This bit is only used on CH2 and, in this channel, should be always set to '1'.

DeSize and SoSize: These fields define the bus width for the next transfer. Since that a FIFO is present inside DMA different bus width can be set.

- 00 - Byte (8 bit)
- 01 - Half word (16 bit)
- 10 - Word (32 bit)
- 11 - Reserved

SoBurst: This field defines the number of words in the peripheral burst. When the peripheral is the source, that is the number of (SoWidth) words read in to the FIFO before writing FIFO contents to destination. When the peripheral is the destination, the DMA interface will automatically read the correct number of source words to compile an SoBurst of DeWidth data.

When stream 3 is configured as a memory-memory transfer, SoBurst relates to the source side burst length.

Values are given in the following table..

SoBurst value	AHB Burst Type
00	Single
01	4 incrementing
10	8 incrementing
11	16 incrementing

DEInc: This bit is used to enable the Current Destination Register increment after each source to DMA data transfer. If the bit is set to '1', the Current Destination Register will be incremented.

SoInc: The SoInc bit is used to enable the Current Source Register increment after each source to DMA data transfer. If the bit is set to '1', the Current Source Register will be incremented.

Enable: This bit enables the channel when set to '1'.

- 1 - DMA enabled.
- 0 - DMA disabled.

6.10.3.5 DMA Source Current Register

Mnemonic: DMASoCurLow and DMASOCurHigh

The Current Source Registers hold the current value of the source address pointer. The registers are 16 bit read only registers. The value in the registers is used as an AHB address in a source to DMA data transfer over the AHB bus.

If the SoInc bit in the Control Register is set to '1', the value in the Current Source Registers will be incremented as data are transferred from a source to the DMA. The value will be incremented at the end of the address phase of the AHB bus transfer by the HSIZE value.

If the SoInc bit is '0', the Current Source Register will hold a same value during the whole DMA data transfer.

6.10.3.6 DMA Destination Current Register

Mnemonic: DMADeCurLow and DMADeCurHigh

The Current Destination Registers hold the current value of the destination address pointer. The registers are 16 bit read only registers. The value in the registers is used as an AHB address in a DMA to destination data transfer over the AHB bus.

If the DeInc bit in the Control Register is set to '1', the value in the Current Destination Registers will be incremented as data are transferred from the DMA to a destination. The value will be incremented at the end of the address phase of the AHB bus transfer by the HSIZE value.

If the DeInc bit is '0', the Current Destination Register will hold a same value during the whole DMA data transfer.

6.10.3.7 DMA Current Count Register

Mnemonic: DMACurTCnt

The Terminal Count Register is a 16 bit read-only register. The register contains the number of data units remaining in the current DMA transfer. The data unit is equal to the source to DMA data width (byte, half-word or word).

The register value is decremented every time data is transferred to the DMA FIFO. When the terminal count reaches zero, the FIFO content is transferred to the destination and a DMA transfer is finished.

6.10.3.8 Mask Register

Mnemonic: DMAMask

Default value: 0x0000

Bit	Field Name	Access
15:08	Reserved	RO
07	MaskE3	RW
06	MaskE2	RW
05	MaskE1	RW
04	MaskE0	RW
03	Mask3	RW
02	Mask2	RW
01	Mask1	RW
00	Mask0	RW

MaskEx: When set to '1' this bit enable the corresponding channel to generate interrupt when an error occurs during the transfer.

Maskx: When set to '1' this bit enable the corresponding channel to generate interrupt when the terminal count is reached.

6.10.3.9 DMA Clear Register

Mnemonic: DMAClear

Bit	Field Name	Access
15:08	Reserved	RO
07	ClearE3	WO
06	ClearE2	WO
05	ClearE1	WO
04	ClearE0	WO
03	Clear3	WO
02	Clear2	WO
01	Clear1	WO
00	Clear0	WO

ClearEx: Writing '1' to this bit will clear ErrorIntx flag in the status register and the interrupt will be de-asserted.

Clearx: Writing '1' to this bit will clear Intx flag in the status register and the interrupt will be de-asserted.

6.10.3.10 DMA Status Register

Mnemonic: DMAStatus

Default value: 0x0000

Bit	Field Name	Access
15:12	Reserved	RO
11	Active3	RO
10	Active2	RO
09	Active1	RO
08	Active0	RO
07	ErrorInt3	RO
06	ErrorInt2	RO
05	ErrorInt1	RO
04	ErrorInt0	RO
03	Int3	RO
02	Int2	RO
01	Int1	RO
00	Int0	RO

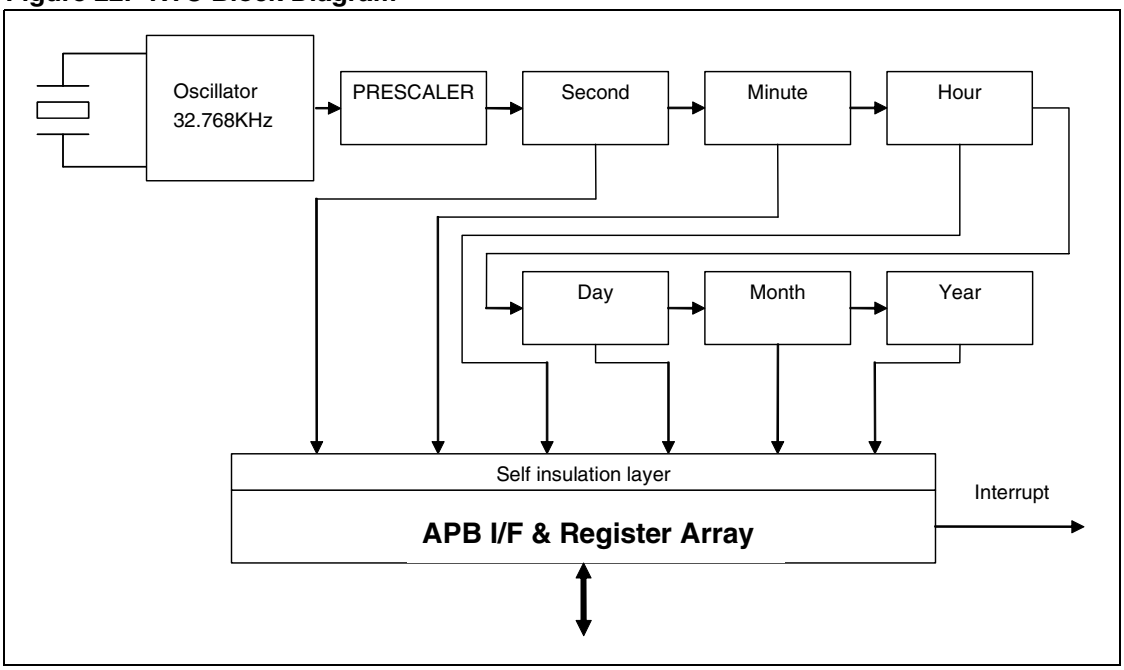
Activex: The Active3-0 flags are used to indicate if a data stream is transferring data. It is high if a data transfer is in progress. The Active flags have the same as the Enable bits in the data stream control register.

ErrorIntx: The ErrorInt3-0 bits are the Error interrupt flags. They are set when the data stream transfer was aborted by an ERROR response on HRESP by an AHB slave. When this occurs, the stream will be disabled until the Enable bit is again set by software.

Intx: The Int3-0 bits are the data stream interrupt flags. A Data stream will set the interrupt flag when a data transfer has finished (the whole packet has been transferred to destination).

6.11 RTC

Figure 22. RTC Block Diagram



6.11.1 Overview

The RTC block combines a complete time-of-day clock with an alarm and a 9999-year calendar. The time is in 24 hour mode, and time/calendar values are stored in binary-coded decimal format. The time-of-day, alarm and calendar, status and control registers can all be accessed via APB bus.

The RTC provides also a self-isolation mode that is activated during power down: this feature allows the RTC to continue working even if main power supply isn't supplied to the rest of the chip.

This is realized supplying separate power and clock connections.

6.11.2 Register Map

Address	Register Name	Access
0x3000_0C00	TIME	RW
0x3000_0C04	DATE	RW
0x3000_0C08	ALARM_TIME	RW
0x3000_0C0C	ALARM_DATE	RW
0x3000_0C10	CONTROL	RW
0x3000_0C14	STATUS	RC

6.11.3 Register Description

6.11.3.1 TIME Register

Mnemonic: TIME

Address: 0x3000_0C00

Default value: 0x0000_0000

Bit	Field Name	Access
31:22	Reserved	RO
21:20	HT	RW
19:16	HU	RW
15	Reserved	RO
14:12	MIT	RW
11:08	MIU	RW
07	Reserved	RO
06:04	ST	RW
03:00	SU	RW

SU: Current value of seconds units in BCD format.

ST: Current value of seconds tens in BCD format.

MIU: Current value of minutes units in BCD format.

MIT: Current value of minutes tens in BCD format.

HU: Current value of hours units in BCD format.

HT: Current value of hours tens in BCD format.

Note: If there is a pending write to this register (see status register), a further write will be lost.

6.11.3.2 DATE Register

Mnemonic: DATE

Address: 0x3000_0C04

Default value: 0x0000_0000

Bit	Field Name	Access
31:28	YM	RW
27:24	YH	RW
23:20	YT	RW
19:16	YU	RW
15:13	Reserved	RO
12	MT	RW
11:08	MU	RW
07:06	Reserved	RO
05:04	DT	RW
03:00	DU	RW

DU: Current value of day's units in BCD format.
DT: Current value of days tens in BCD format.
MU: Current value of month's units in BCD format.
MT: Current value of months tens in BCD format.
YU: Current value of year's units in BCD format.
YT: Current value of years tens in BCD format.
YH: Current value of years hundreds in BCD format.
YM: Current value of year's millennium in BCD format.

Note: If there is a pending write to this register (see status register), a further write will be lost.

6.11.3.3 ALARM TIME Register

Mnemonic: ALARM_TIME

Address: 0x3000_0C08

Default value: 0x0000_0000

Bit	Field Name	Access
31:22	Reserved	RO
21:20	AHT	RW
19:16	AHU	RW
15	Reserved	RO
14:12	AMIT	RW
11:08	AMIU	RW
07	Reserved	RO
06:04	AST	RW
03:00	ASU	RW

6.11.3.4 ALARM DATE Register

Mnemonic: ALARM_DATE

Address: 0x3000_0C0C

Default value: 0x0000_0000

Bit	Field Name	Access
31:28	AYM	RW
27:24	AYH	RW
23:20	AYT	RW
19:16	AYU	RW
15:13	Reserved	RO
12	AMT	RW
11:08	AMU	RW
07:06	Reserved	RO
05:04	ADT	RW
03:00	ADU	RW

6.11.3.5 CONTROL Register

Mnemonic: CONTROL

Address: 0x3000_0C10

Default value: 0x0000_0000

Bit	Field Name	Access
31	IE	RW
30:10	Reserved	RO
09	TB	RW
08	PB	RW
07:06	Reserved	RO
05	MASK5	RW
04	MASK4	RW
03	MASK3	RW
02	MASK2	RW
01	MASK1	RW
0	MASK0	RW

IE: Interrupt enable. If set to '1' an interrupt will be sent to the CPU when TIME and DATE registers are equal to the ALARM_TIME and ALARM_DATE.

TB: Time bypass (Test only). When this bit is set to '1' the date counter will be driven directly with the prescaler output bypassing the TIME counter.

PB: Prescaler bypass (Test only). When this bit is set to '1' the TIME counter will be directly driven with the 32 KHz coming from oscillator bypassing the prescaler.

MASK5: When set to '1' the years compare will be forced to true.

MASK4: When set to '1' the months compare will be forced to true.

MASK3: When set to '1' the days compare will be forced to true.

MASK2: When set to '1' the hours compare will be forced to true.

MASK1: When set to '1' the minutes compare will be forced to true.

MASK0: When set to '1' the seconds compare will be forced to true.

6.11.3.6 STATUS Register

Mnemonic: STATUS

Address: 0x3000:0C14

Default value: 0x0000_0000

Bit	Field Name	Access
31	INT	RC
30:06	Reserved	RO
05	LD	RO
04	LT	RO
03	PD	RO
02	PT	RO
01	Reserved	RO
00	RC	RO

INT: Interrupt request. Writing '1' will reset the flag.

LD: Write to date register Lost. If a second write is request to date register before the first is competed, this second request is aborted and LD bit is asserted. This bit is cleared when a write to date register is performed successfully.

LT: Write to time register Lost. If a second write is request to time register before the first is competed, this second request is aborted and LT bit is asserted. This bit is cleared when a write to time register is performed successfully.

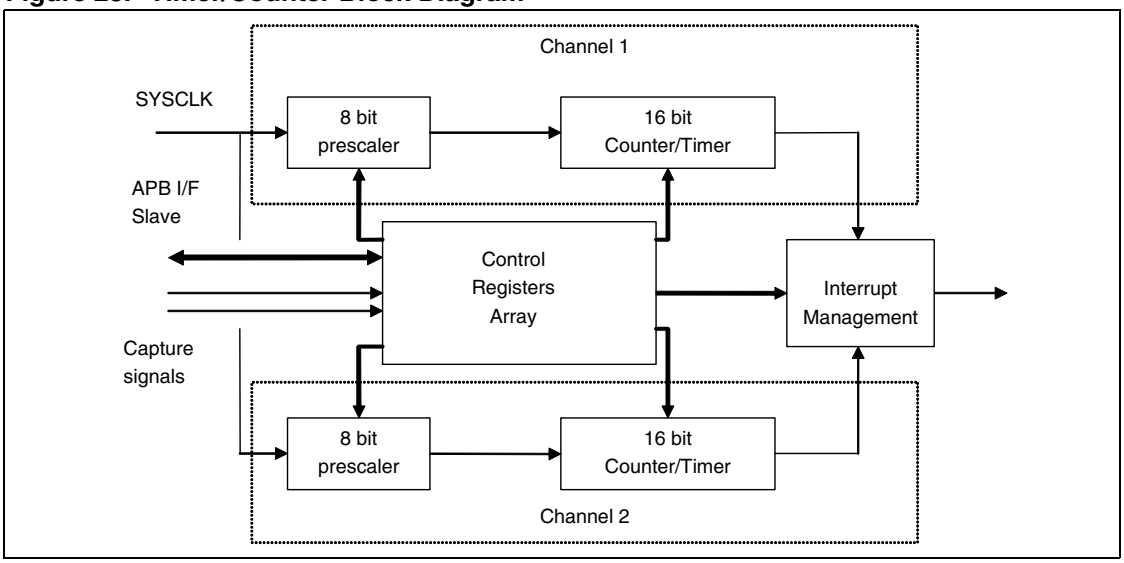
PD: Pending write to Date register. This bit indicates that a write request to date register is asserted from 48 MHz part to 32 KHz part. It is independent from PT. A new write can be successfully requested only when this bit is de-asserted.

PT: Pending write to Time register. This bit indicates that a write request to time register is asserted from 48 MHz part to 32 KHz part. A new write can be successfully requested only when this bit is de-asserted.

RC: RTC Connected. When reset to '0' the timer will be self-isolated from the other blocks. Reading and writing to time and date can be safely done only when this bit is set to '1'.

6.12 Timer/Counter

Figure 23. Timer/Counter Block Diagram



6.12.1 Overview

Two channels constitute the GPT and each one consists of a programmable 16 bits counter and a dedicated 8 bits timer clock prescaler. The programmable 8-bit prescaler unit performs a clock division by 1, 2, 4, 8, 16, 32, 64, 128, and 256.

Two different interrupts can be generated for each timer:

- Toggle
- Interval

Two modes of operations are available for each timer:

- Auto-reload mode
- Single-shot mode

6.12.1.1 Auto Reload Mode

When the timer is enabled, the counter is cleared and starts incrementing. When it reaches the compare register value, an interrupt source is activated the counter is automatically cleared and restarts incrementing. The process is repeated until the timer is disabled.

6.12.1.2 Single Shot Mode

When the timer is enabled, the counter is cleared and starts incrementing. When it reaches the compare register value, an interrupt source is activated, the counter stopped and the timer disabled.

6.12.2 Registers Map

Address	Register Name	Access
0x3000_0480	TIMER_CONTROL1	RW
0x3000_0484	TIMER_STATUS1	RO
0x3000_0484	TIMER_INT_ACK1	WO
0x3000_0488	TIMER_COMPARE1	RW
0x3000_048C	TIMER_COUNT1	RO
0x3000_0500	TIMER_CONTROL2	RW
0x3000_0504	TIMER_STATUS2	RO
0x3000_0504	TIMER_INT_ACK2	WO
0x3000_0508	TIMER_COMPARE2	RW
0x3000_050C	TIMER_COUNT2	RO

6.12.3 Registers Description

All registers are 16 bit wide and 32 bit aligned.

6.12.3.1 Timer Control Register

Mnemonic: TIMER_CONTROL1, TIMER_CONTROL2

Address: 0x3000_0480 (Timer 1) and 0x3000_0500 (Timer 2)

Default value: 0x0000

Bit	Field Name	Access
15:09	Reserved	RO
08	MATCH_INT	RW
07:06	Reserved	RO
05	ENABLE	RW
04	MODE	RW
03:00	PRESCALER	RW

MATCH_INT: If set enables the interruption when the comparator matches.

CAPTURE: Capture Configuration Bits:

MODE: When set single-shot mode is enabled. When reset auto-reload mode is enabled.

PRESCALER: Prescaler configuration (considered a clock frequency of 48MHz):

Value	Division	Frequency	Resolution	MAX Time
0000	/1	48 MHz	20.8 ns	1.365 ms
0001	/2	24 MHz	41.6 ns	2.73 ms
0010	/4	12 MHz	83.3 ns	5.46 ms
0011	/8	6 MHz	166.7 ns	10.922 ms
0100	/16	3 MHz	333.3 ns	21.845 ms
0101	/32	1.5 MHz	666.7 ns	43.69 ms
0110	/64	750 KHz	1333 ns	87.381 ms
0111	/128	375 KHz	2663 ns	174.529 ms
1000	/256	187.5 KHz	5333 ns	349.525 ms
1001:1111	Not Allowed	Not Allowed	Not Allowed	Not Allowed

Note: ***Enable and Disable***

After RESET timer is disabled and all interrupt sources are masked. When a timer is enabled, an initialization phase is performed before starting to count; during that initialization phase, the capture registers and the counter are cleared.

When a timer is disabled, the capture registers and the counter are frozen.

6.12.3.2 Timer Status Register

Mnemonic: TIMER_STATUS1 and TIMER_STATUS2

Address: 0x3000_0484 (Timer 1) and 0x3000_0504 (Timer 2)

Default value: 0x0000

Bit	Field Name	Access
15:03	Reserved	RO
02	REDGE	RO
01	FEDGE	RO
00	MATCH	RO

This register indicates the raw interrupt sources status, prior any mask setting.

REDGE: When set a rising edge has been detected on the capture input.

FEDGE: When set a falling edge has been detected on the capture input.

MATCH: When set a match has occurred in the compare unit.

6.12.3.3 Timer Interrupt Acknowledge Register

Mnemonic: TIMER_INT_ACK1 and TIMER_INT_ACK2

Address: 0x3000_0484 (Timer 1) and 0x3000_0504 (Timer 2)

Default value: 0x????

Bit	Field Name	Access
15:01	Reserved	RO
00	MATCH	RO

This register allows the software to clear the interrupt sources.

MATCH: Writing a '1' clears the bit. Writing a '0' hasn't effect.

Note: ***Pending Interrupts***

Independently by the TIMER activity, pending interruptions remain active until they have been acknowledged. They are not automatically deactivated when the timer is disabled or enabled. It is therefore strongly recommended to acknowledge all active interrupt sources before enabling a timer.

6.12.3.4 Compare Register

Mnemonic: TIMER_COMPARE1 and TIMER_COMPARE2

Address: 0x3000_0488 (Timer 1) and 0x3000_0508 (Timer 2)

Default value: 0xFFFF

Bit	Field Name	Access
15:00	COMPARE_VALUE	RW

This register allows the software to program the timer period.

In auto-reload mode, when the counter has reached the compare value, it is cleared and restarts incrementing:

TIMER_PERIOD = (COMPARE_VALUE - 1) x COUNTER_PERIOD + 2 TIMER_CLK periods

Min value: 0001H

Max value: FFFFH (in auto-reload mode this value means free running)

6.12.3.5 Timer Count Register

Mnemonic: TIMER_COUNT1 and TIMER_COUNT2

Address: 0x3000_048c (Timer 1) and 0x3000_050C (Timer 2)

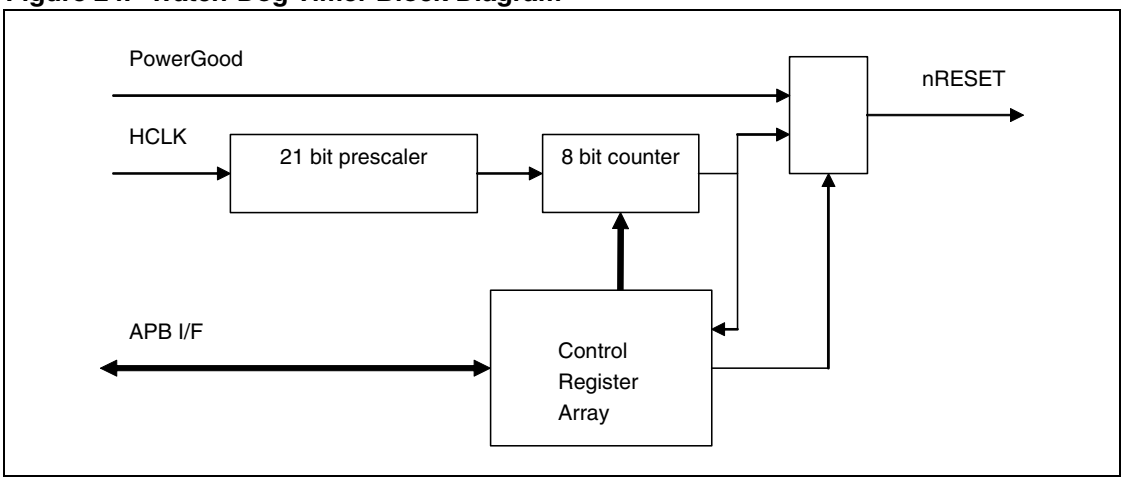
Default value: 0x0000

Bit	Field Name	Access
15:00	COUNT_VALUE	RO

This register indicates the current counter value.

6.13 Watch-Dog Timer

Figure 24. Watch-Dog Timer Block Diagram



6.13.1 Overview

The WT is based on a programmable 8 bit counter and generates an hot reset (single pulse) when it overflows. The counter is clocked by a signal coming from a 21 bit prescaler clocked by the system clock. The step for the counter is 43.69 ms (1/48000000) * 2^21).

When enabled the counter start and, as soon the limit is reached, a RESET will be sent to the system. To avoid it the CPU has to restart the counter in a given time (less than the value written inside register multiplied for 43.69 ms).

6.13.2 Register map

Address	Register Name	Access
0x3000_0800	WDOG_CONTROL	RW
0x3000_0804	WDOG_STATUS	RO
0x3000_0808	WDOG_MAX_CNT	RW
0x3000_080C	WDOG_COUNTER	RO

6.13.3 Register Description

All registers are 16 bit wide and 32 bit aligned.

6.13.3.1 Watch-Dog Control Register

Mnemonic: WDOG_CONTROL

Address: 0x3000_0800

Default Value: 0x0004

Bit	Field Name	Access
15:04	Reserved	RO

03	FAST	RW
02	DEBUG_FRZ	RW
01	ENAB	RW
00	CLEAR	RW

FAST: When set the elapsing time of the counter will by divided by 16. This mode is used only for testing purposes.

DEBUG_FRZ: When this bit is set and the CPU is in DEBUG mode the watch-dog timer will freeze its contents. This feature allow the user to stop the CPU activity using break avoiding unexpected RESET.

ENAB: When set the WT is enabled; when it is cleared the prescaler and the counter are cleared and they don't start to count until the ENAB bit is set again.

CLEAR: When set the internal counter and prescaler are cleared. So the WT will be restarted. The hardware automatically clears this bit after the software has set it.

6.13.3.2 Watch-Dog Status Register

Mnemonic: WDOG_STATUS

Address: 0x3000_0804

Default value: 0x0000

Bit	Field Name	Access
15:01	Reserved	RO
00	WD_RES	RO

This register allows the software reset handler to determine the reset source.

WD_RES: set when a WT reset is occurred. To clear this bit the software must write a "0". Writing a "1" has not effect.

6.13.3.3 Watch-Dog Maximum Count Register

Mnemonic: WDOG_MAX_CNT

Address: 0x3000_0808

Default value: 0x0000

Bit	Field Name	Access
15:08	Reserved	RO
07:00	MAXCNT_VALUE	RW

MAXCNT_VALUE: programmable value for 8-bit counter clocked with the 21 bit prescaler output.

When MAXCNT_VALUE + 1 is reached WT generates the hot reset.

6.13.3.4 Watch-Dog Counter Register

Mnemonic: WDOG_COUNTER

Address: 0x3000_080C

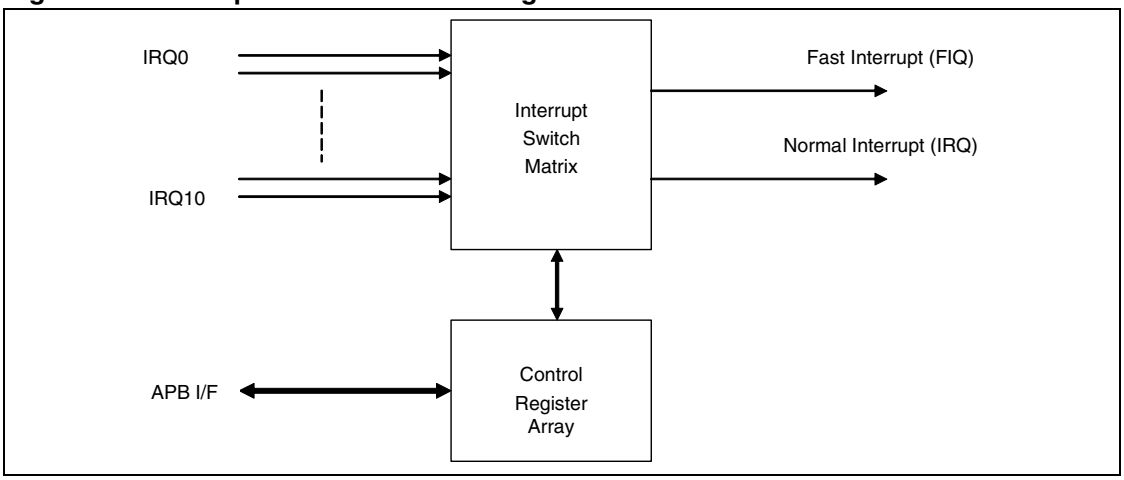
Default value: 0x0000

Bit	Field Name	Access
15:08	Reserved	RO
07:00	COUNTER_CVALUE	RO

COUNTER_CVALUE: Current 8 bit counter value.

6.14 Interrupt Controller

Figure 25. Interrupt Controller Block Diagram



6.14.1 Overview

The interrupt controller provides a simple software interface to the interrupt system.

In an ARM system, two levels of interrupt are available:

- nFIQ (Fast Interrupt Request) for fast, low latency interrupt handling
- nIRQ (Interrupt Request) for more general interrupts

Ideally, in an ARM system, only a single nFIQ source would be in use at any particular time. This provides a true low-latency interrupt, because a single source ensures that the interrupt service routine may be executed directly without the need to determine the source of the interrupt. It also reduces the interrupt latency because the extra banked registers, which are available for FIQ interrupts, may be used to maximum efficiency by preventing the need for a context save.

The interrupt controller manages 11 interrupt sources. For each source, is possible to select which event is to be considered as the active one: level (active low or high), rising edge, falling edge or both. Is also possible to choose if each request will be asserted to the ARM as FIQ or IRQ.

The interrupt requests are stored into the "pending_reg" register. The output of this register is combined by logical "or" to the software interrupts stored in the "soft_interrupt_reg" register but the software interrupts are not stored in the pending register.

Resulting requests can be masked by using the "enable_reg" register. Global enable/disable of both nIRQ and nFIQ is done by the "cntrl_reg" register.

Note: an event on input "i" (int_req(i)) causes an interrupt request on nIRQ (or nFIQ) if:

1. *The event at the input int_req(i) is a valid event (rising edge, falling edge, both or right level), as programmed in the related nibble of CONFIG_xx_xx_reg register.*
2. *Request (i) is enabled on ENABLE_REG register (enable_reg(i) = '1')*
3. *Request(i) is routed onto nIRQ (nFIQ) by its nibble in CONFIG_xx_xx_reg register.*
4. *nIRQ request is enabled by CNTRL register CNTRL(0)='1'.*

6.14.2 Register Map

Address	Register Name	Access
0x3000_0000	CONTROL	RW
0x3000_0004	IRQ_STATUS	RO
0x3000_0008	FIQ_STATUS	RO
0x3000_000C	PENDING	RC
0x3000_0010	CONFIG_1	RW
0x3000_0014	CONFIG_2	RW
0x3000_0018	Reserved	RO
0x3000_001C	Reserved	RO
0x3000_0020	ENABLE	RW
0x3000_0024	Reserved	RO
0x3000_0028	Reserved	RO
0x3000_002C	Reserved	RO
0x3000_0030	SOFT_INTERRUPT	RW

6.14.3 Register Description

All the registers are 32 bit wide.

6.14.3.1 Control Register

Mnemonic: CONTROL

Address: 0x3000_0000

Default value: 0x0000_0000

Bit	Field Name	Access
31:02	Reserved	RO
01	FIQ_ENABLE	RW
00	IRQ_ENABLE	RW

FIQ_EN: nFIQ global enable. It's an active high bit and when '1' enables the nFIQ ITC output.

IRQ_EN: nIRQ global enable. It's an active high bit and when '1' enables the nIRQ ITC output.

6.14.3.2 IRQ Status Register

Mnemonic: IRQ_STATUS

Address: 0x3000_0004

Default value: 0x0000_0000

Bit	Field Name	Access
31:11	Reserved	RO
10	INT_10	RO
09	INT_09	RO
08	INT_08	RO
07	INT_07	RO
06	INT_06	RO
05	INT_05	RO
04	INT_04	RO
03	INT_03	RO
02	INT_02	RO
01	INT_01	RO
00	INT_00	RO

This register allows the software interrupt handler to determine the nIRQ interrupt source. Each request is considered to be active high.

INT_i: Interrupt request number "i" status. If this bit is '0' it means that the request is NOT active.

If the bit is '1' it means that

1) either the input int_req(i)
   a) is pending AND
   b) has been routed to the nIRQ output AND
   c) has been enabled
2) or the software Interrupt i
   a) has been set
   b) has been routed to the nIRQ output AND
   c) has been enabled

6.14.3.3 FIQ Status Register

Mnemonic: FIQ_STATUS

Address: 0x3000_0008

Default value: 0x0000_0000

Bit	Field Name	Access
31:11	Reserved	RO
10	INT_10	RO
09	INT_09	RO
08	INT_08	RO
07	INT_07	RO
06	INT_06	RO
05	INT_05	RO
04	INT_04	RO
03	INT_03	RO
02	INT_02	RO
01	INT_01	RO
00	INT_00	RO

This register allows the software interrupt handler to determine the nFIQ interrupt source. Each request is considered to be active high.

INT_i: Interrupt request number "i" status. If this bit is '0' it means that the request is NOT active.

If the bit is '1' it means that

1) either the input int_req(i)
   a) is pending AND
   b) has been routed to the nFIQQ output AND
   c) has been enabled
2) or the software Interrupt i
   a) has been set AND
   b) has been routed to the nFIQ output AND
   c) has been enabled

6.14.3.4 Interrupt Pending Register

Mnemonic: PENDING

Address: 0x3000_000C

Default value: 0x0000_0000

Bit	Field Name	Access
31:11	Reserved	RO
10	INT_10	RC
09	INT_09	RC
08	INT_08	RC
07	INT_07	RC
06	INT_06	RC
05	INT_05	RC
04	INT_04	RC
03	INT_03	RC
02	INT_02	RC
01	INT_01	RC
00	INT_00	RC

This register stores the requests coming from input agents only. The request is stored even though the corresponding "enable_reg" bit is not active.

INT_i: Active "high": if the bit pending_reg(i) is '1', it means that 1 interrupt request was recognized on int_req(i). This register can be cleared bit by bit by writing '1' in the corresponding bit. Writing '0' doesn't change the bit value.

Note: A read operation doesn't change the value of the register. A write with '0' of bit "i" doesn't change the value of the bit.

6.14.3.5 Configuration Registers

Mnemonic: CONFIG_1

Address: 0x3000_0010

Default value: 0x0000_0000

Bit	Field Name	Access
31:28	CONF_7	RW
27:24	CONF_6	RW
23:20	CONF_5	RW
19:16	CONF_4	RW
15:12	CONF_3	RW
11:08	CONF_2	RW
07:04	CONF_1	RW
03:00	CONF_0	RW

Mnemonic: CONFIG_2

Address: 0x3000_0014

Default value: 0x0000_0000

Bit	Field Name	Access
31:12	Reserved	RO
11:08	CONF_10	RW
07:04	CONF_9	RW
03:00	CONF_8	RW

The registers are organized nibble-by-nibble: each nibble refers to the corresponding "int_req(i)" input.

- Bit(2:0) of each nibble set which event is recognized as interrupt request (see the following table).
- Bit(3) of each nibble sets which kind of interrupt will be stated to ARM ("0":IRQ ; "1":FIQ).

b2	b1	b0	Purpose
0	0	0	Int_req(i) completely masked. (default)
0	0	1	Int_req(i) is falling edge sensitive.
0	1	0	Int_req(i) is rising edge sensitive.
0	1	1	Int_req(i) is both edges sensitive.
1	-	0	Int_req(i) is level sensitive, active Low.
1	-	1	Int_req(i) is level sensitive, active High.

6.14.3.6 Enable Register

Mnemonic: ENABLE

Address: 0x3000_0020

Default value: 0x0000_0000

Bit	Field Name	Access
31:11	Reserved	RO
10	INT_10	RW
09	INT_09	RW
08	INT_08	RW
07	INT_07	RW
06	INT_06	RW
05	INT_05	RW
04	INT_04	RW
03	INT_03	RW
02	INT_02	RW
01	INT_01	RW
00	INT_00	RW

This register enables the input interrupt requests bit by bit.

EN(i): Active high enable bit for input int_req(i).

Bit "i" = 1 means that request "i" is enabled.

Bit "i" = 0 means that request "i" is masked

6.14.3.7 Software Interrupt Register

Mnemonic: SOFT_INTERRUPT

Address: 0x3000_0030

Default value: 0x0000_0000

Bit	Field Name	Access
31:11	Reserved	RO
10	SI_10	RW
09	SI_09	RW
08	SI_08	RW
07	SI_07	RW
06	SI_06	RW
05	SI_05	RW
04	SI_04	RW
03	SI_03	RW
02	SI_02	RW
01	SI_01	RW
00	SI_00	RW

This register allows to sets soft interrupts.

It's intended for debugging purposes and allows the user to simulate an interrupt request on each of the 11 interrupt channels.

SI_i: Active high, Soft Interrupt on channel i. When '0', no Soft Interrupt is set. If '1', the Soft Interrupt is active and the ITC logic will react as if the input int_req(i) was set. Each interrupt request can be cleared just writing "0" on the corresponding bit.

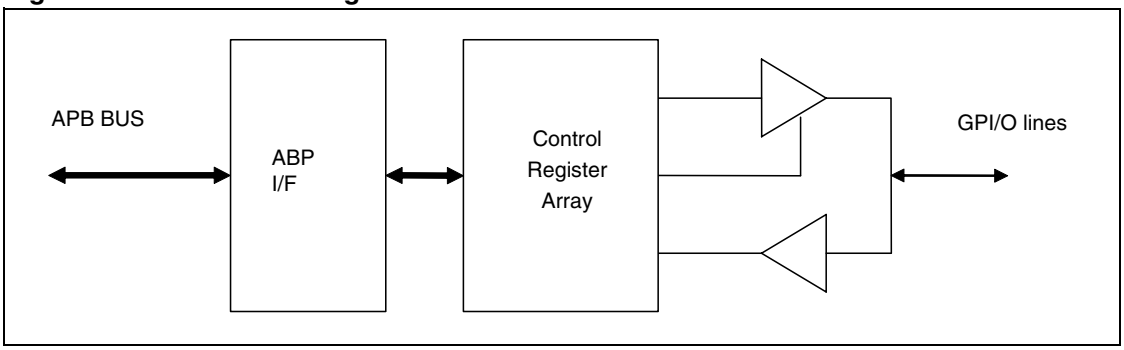
6.14.4 Interrupt Table

The following table show the connection of the 11 interrupt sources inside SPEAr Net and the proper setting that should be used for each particular interrupt.

Interrupt number	Agent	Interrupt name	Description
0	Ethernet MAC	MAC	Active High.
1	USB	USB	Active Low.
2	IEEE1284	IEEE1284	Active High.
3	I2C	I2C	Active High.
4	UART	UART	Active High.
5	RTC	RTC	Active High.
6	Timer1	TIMER1	Active Low.
7	Timer2	TIMER2	Active Low.
8	External	nXIRQ(0)	Active Low.
9	External	nXIRQ(1)	Active Low.
10	DMA	DMA	Active High.

6.15 GPIO

Figure 26. GPIO Block Diagram



6.15.1 Overview

The GPIO block consists of 6 general purpose IO's which are configured by means of a dedicated direction register. The GPIO block acts as a buffer between the IO Pads and the processor Core.

Data is stored in the GPIO block and can be written to and read from by the Processor via the APB bus.

6.15.2 Register Map

Address	Register Name	Access
0x3000_1000	GPP_DIR	RW
0x3000_1004	GPP_DIN	RO
0x3000_1010	GPP_DOUT0	WO
0x3000_1014	GPP_DOUT1	WO
0x3000_1018	GPP_DOUT2	WO
0x3000_101C	GPP_DOUT3	WO
0x3000_1020	GPP_DOUT4	WO
0x3000_1024	GPP_DOUT5	WO

6.15.3 Registers Description

6.15.3.1 General Purpose I/O Direction Register

Mnemonic: GPP_DIR

Address: 0x3000_1000

Default value: 0x3F

Bit	Field Name	Access
07:06	Reserved	RO
05	DIR5	RW
04	DIR4	RW
03	DIR3	RW
02	DIR2	RW
01	DIR1	RW
00	DIR0	RW

DIR(5:0) : When set to '1' the IO pin is configured as an Input. When set to '0' the IO pin is configured as an Output.

6.15.3.2 General Purpose I/O Input Data Register

Mnemonic: GPP_DIN

Address: 0x3000_1004

Default value: NA

Bit	Field Name	Access
07:06	Reserved	RO
05	DIN5	RO
04	DIN4	RO
03	DIN3	RO
02	DIN2	RO
01	DIN1	RO
00	DIN0	RO

DIN(5:0): value of the GPIO pins.

6.15.3.3 General Purpose I/Ox Output Data Register

Mnemonic: DOUT(5:0)

Address: 0x3000_1010 (GPIO0), 0x3000_1014 (GPIO1), 0x3000_1018 (GPIO2), 0x3000_101C (GPIO3), 0x3000_1020 (GPIO4), 0x3000_1024 (GPIO5)

Default value 0x00

Bit	Field Name	Access
07:01	Reserved	RO
00	DOUT	WO

DOUT: Its value will appear on the GPIO pin if the direction bit is set to '0'.

6.16 RESET and Clock Controller

6.16.1 Overview

The Reset Clock Generator provides the clock and reset signals for the SPEAr Net core. It also produces a refresh signal for DRAM.

The reference clock frequency is 25 MHz. This signal is used to generate the 48 MHz system clock by an internal PLL. Clock signals for USB and IEEE blocks are provided separately. They can be stopped by the control signals "ENABLE_USB_CLK" and "IEEE1XP0" in order to save power. USB 12 MHz clock is generated by dividing the 48 MHz USB clock by 4.

Refresh signal is generated by dividing 48 MHz by 48, and it is used by the Refresh Timer.

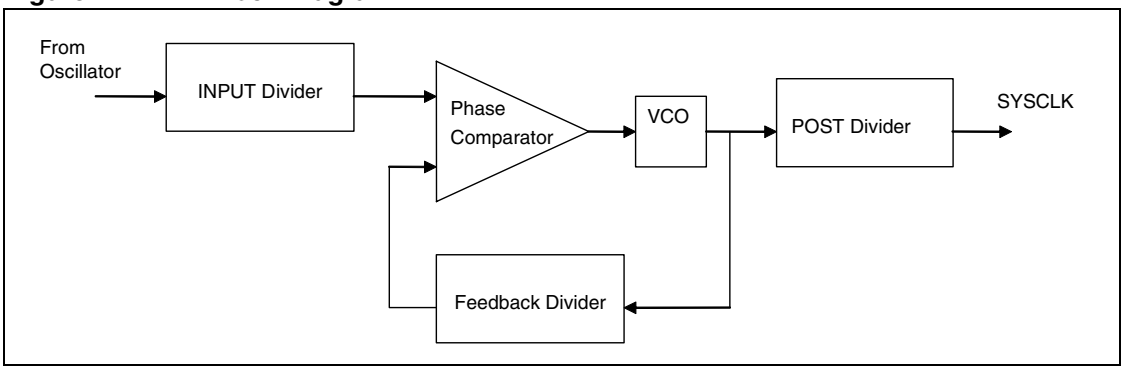
Input global reset is connected to POWERGOOD pin. It's supposed to be active low and asynchronous. It must be kept active until the PLL is locked (in example, for 50 us).

The second input reset signal comes from watch dog, and acts on all of the other blocks, causing an active pulse 200 clock cycles long (48 MHz).

Output reset signal is synchronized on 48 MHz clock. 12 MHz clock domain has its proper reset synchronization circuit (inside USB block).

6.17 PLL (Frequency synthesizer)

Figure 27. PLL Block Diagram



6.17.1 Overview

The PLL is based upon the charge pump principle. In consist of mainly 5 blocks:

1) Input (PRE) divider
2) Phase/ Frequency Comparator
3) Voltage Controller Oscillator
4) Feedback Divider
5) Post Divider.

Pre-Divider: In SPEAr Net this divider is set to 25.

Phase/Frequency Comparator: This comparator drives trough a low pass filter the VCO control imput.

VCO: In SPEAr Net the VCO runs at 48 MHz.

Feedback Divider: In SPEAr Net this divider is set to 48.

Post Divider: In SPEAr Net this divider is set to 1.

According to the setting on the three dividers the System Frequency, connecting a 25 MHz crystal will be:

$$F_{OUT} = 2 * \text{Feedback Divider} * F_{IN} / (\text{Pre Divider}) * 2^{\text{Post Divider}})$$

Then the system clock will be:

$$F_{OUT} = 2 * 48 * 25 / (25 * 21) = 48 \text{ MHz}$$

This means that the CPU, the system bus and also the external DRAM will run at this frequency.

6.17.2 Global Configuration Block

The global configuration block includes the system configuration registers, the system control/ status registers and the shared memory control/status registers.

The system configuration registers sample the value presented at the ADD lines during the power- on reset phase. This reset phase is caused by the POWERGOOD signal driven low. During this phase the ADD lines are configured as input; there should be resistances on the board to drive the ADD lines with a weak high or low signal that will be latched on the registers and will configure the system hardware and possibly the software. The PLL_BYPASS and the JTAG_ENABLE_N conditions are propagated to the system even before the POWERGOOD signal is asserted, to guarantee a proper setup in every case and to allow usage of JTAG before any clock cycle is completed.

When JTAG_ENABLE_N is driven low the pins in the first column of Table 1, "Pin mapping for JTAG interface", on page 2 change their function as defined in the second column.

6.17.3 Register Map

Address	Register Name	Access
0x3000_1C00	FW_CFG	
0x3000_1C04	HW_CFG	
0x3000_1C08	GLOBAL_CONTROL	
0x3000_1C0C	GLOBAL_STATUS	
0x3000_1C10	SHRAM_TEST_CTRL	
0x3000_1C14	SHRAM_TEST_STATUS	

6.17.4 Registers Description

All the register in this block are 8 bit wide and 32 bit aligned.

6.17.4.1 Firmware Configuration Register

Mnemonic: FW_CFG

Address: 0x3000_1C00

Default value: NA

Bit	Field Name	Access
07	FW_CFG7	RO
06	FW_CFG6	RO
05	FW_CFG5	RO
04	FW_CFG4	RO
03	FW_CFG3	RO
02	FW_CFG2	RO
01	FW_CFG1	RO
00	FW_CFG0	RO

This register holds the value of eight address lines after the PowerOn reset.

FW_CFG(7-0) ← ADD(14-7)

6.17.4.2 Hardware Configuration Register

Mnemonic: HW_CFG

Address: 0x3000_1C04

Default value: NA

Bit	Field Name	Access
07	SDRAM_TYPE	RO
06	USB_CLK_EN	RO
05	IEEE_XP	RO
04	ROM_BSIZE	RO
03	JTAG_ENABLE	RO
02	PLL_BYPASS	RO
01-00	Reserved	RO

SDRAM_TYPE: this bit holds the value of the onboard configuration pull-up/pull-down resistance connected to pad ADD[22]. When high dynamic memory controller is configured to drive an SDRAM, when low it drives an EDO DRAM.

USB_CLK_EN: this bit holds the value of the onboard configuration pull-up/pull-down resistance connected to pad ADD[21]. When low the USB clock is disabled for power consumption reduction

IEEE_XP: this bit holds the value of the on-board configuration pull-up/pull-down resistance connected to pad ADD[20]. When low the shared memory controller drives the corresponding device pads, the IEEE1284 block clock is disabled to reduce power consumption and the AHB decoder maps the IEEE memory area to default slave.

When high the IEEE1284 block is clocked and mapped and has the control of XPDATA[7:0] and XPADDR[7:0] pads as in following table.

Table 18. Pin mapping for IEEE1284 Interface

Ball	Function when IEEE_XP = '1'	Function when IEEE_XP = '0'
D14	XPDATA(0)	PPDATA(0)
E11	XPDATA(1)	PPDATA(1)
D13	XPDATA(2)	PPDATA(2)
B14	XPDATA(3)	PPDATA(3)
C13	XPDATA(4)	PPDATA(4)
C12	XPDATA(5)	PPDATA(5)
B13	XPDATA(6)	PPDATA(6)
B12	XPDATA(7)	PPDATA(7)
D9	XPADD(1)	nSTROBE
C9	XPADD(2)	nACK
A8	XPADD(3)	BUSY
E8	XPADD(4)	PERROR
D8	XPADD(5)	SELECT
C8	XPADD(6)	nAUTOFD
E7	XPADD(7)	nFAULT
D7	XPADD(8)	nINIT
B7	XPADD(9)	SELECTIN
C7	XPADD(10)	PDATADIR

ROM_BSIZE: this bit holds the value of the onboard configuration pull-up/pull-down resistor connected to pad ADD[19]. When high the static memory controller access to bank 0 expecting a 16 bit memory; when low accesses to bank 0 are performed considering an 8 bit memory chip.

JTAG_ENABLE: this bit holds the value of the on-board configuration pull-up/pull-down resistance connected to pad ADD[18]. When high the RCS[1] and ECS[1] pads are connected to the static memory controller block and the NRAS[3], NRAS[2] and NRAS[1] pads are driven by dynamic memory controller. When low these pads are connected to the ARM JTAG interface as described in the following table.

Table 19. Pin mapping for JTAG Interface

Ball	Function when JTAG_ENABLE = '1'	Function when JTAG_ENABLE = '0'
P12	nECS1	TCK
P14	nRCS1	TMS
M9	nRAS1	TDI
K9	nRAS2	TDO
P9	nRAS3	nTRST

PLL_BYPASS: This bit holds the value of the onboard configuration pull-up/pull-down resistance connected to pad ADD[17]. When high the internal PLL is bypassed and the MCLKI pad signal is used instead of PLL output.

6.17.4.3 Global Control Register

Mnemonic: GLOBAL_CONTROL

Address: 0x3000_1C08

Default value: 0x00

Bit	Field Name	Access
07:02	Reserved	RO
01	nUSB_ENABLE	RW
00	Ni2C_ENABLE	RW

USB_ENABLE: When low GPIO pads are driven by GPIO block and XPADDR[10] and XPADDR[11] pads drive the shared memory controller. When high GPIO pads are driven by the USB block and XPADDR[10] and XPADDR[11] drive the USB block (instead of the internal transceiver). This signal should be driven high only when SPEAr Net Global control register bit 0 (which has higher priority) is set.

Note: This bit is only intended for debug purpose.

The following table shows the ball mapping for this mode.

Table 20. Pin mapping for nUSB_ENABLE

Ball	Function when nUSB_ENABLE = '1'	Function when nUSB_ENABLE = '0'
A3	GPIO0	OE
B4	GPIO1	RCV
A2	GPIO2	SUSPEND
A1	GPIO3	SPEED
B3	GPIO4	VO
C3	GPIO5	FSE0
	XPADD11	VP
	XPADD12	VM

I²C_ENABLE: When low pad GPIO[4] and pad GPIO[5] are respectively I²C SCL and SDA open drain signals driven by I²C block. When high pad GPIO[4] and pad GPIO[5] are connected to GPIO block or to USB block depending on the value of PNCU control register bit 1. The following table shows the pin mapping for this mode.

Table 21. Pin mapping for nI²C_ENABLE

Ball	Function when nI²C_ENABLE = '1'	Function when nI²C_ENABLE = '0'
B3	GPIO4	SCL
C3	GPIO5	SDA

6.17.4.4 Global status Register

Mnemonic: GLOBAL_STATUS

Address: 0x3000_1C0C

Default value: 0x00

Bit	Field Name	Access
07:01	Reserved	RO
00	PLL_LOCK	RO

PLL_LOCK: When the internal PLL is locked this bit is high.

6.17.4.5 Shared Ram Test Control Register

Mnemonic: SHRAM_TEST_CTRL

Address: 0x3000_1C10

Default value: 0x00

Bit	Field Name	Access
07:02	Reserved	RO
01	EM_BIST_START	RW(*)
00	CSN_SHRAM	RW(*)

(*)To write to this register user should first disable the protection, i.e. write a signature value (45h) to the same address of SHRAM_TEST_CTRL. Writing in any location in the APB domain (SHRAMC_CTRL included) re-enables the protection.

EM_BIST_START: This signal drives the shared memory BIST (Built In Self Test) engine. When high the BIST starts testing the memory. User should not use the shared memory when the BIST is active.

CSN_SHRAM: This bit enables the shared memory chip select signal. When high the memory is disabled.

6.17.4.6 Shared Ram Test Status Register

Mnemonic: SHRAM_TEST_STATUS

Address: 0x3000_1C14

Default value: 0x00

Bit	Field Name	Access
07:03	Reserved	RO
02	EM_BIST_ERROR	RO
01	EM_BIST_GONOGO	RO
00	EM_BIST_DONE	RO

EM_BIST_ERROR: When the BIST is finished this bit tells whether the procedure was correct or not.

EM_BIST_GONOGO: When the BIST is finished this bit tells whether the memory passed or no the test.

EM_BIST_DONE: This bit goes high when the BIST is finished.

7 Electrical Characteristics

7.1 Absolute Maximum Rating

This product contains devices to protect the inputs against damage due to high static voltages, however it is advisable to take normal precaution to avoid application of any voltage higher than the specified maximum rated voltages.

Table 22. Absolute Maximum Ratings

Symbol	Parameter	Value	Unit
$V_{DD3I/O}$	Supply Voltage (I/O ring)	-0.3 to 3.9	V
V_{DD3PLL}	Supply voltage (PLL)	-0.3 to 3.9	V
V_{DD}	Supply Voltage (CORE logic)	-0.3 to 2.1	V
V_{DDRTC}	Supply Voltage (Real Time Clock)	-0.3 to 2.1	V
V_{I1}	Input Voltage (Except MCLKI, MCLKO, RTCXI, RTCXO, UHD+ and UHD-)	VSS – 0.3 to VDD3I/O + 0.3	V
V_{I2}	Input Voltage for MCLKI, MCLKO	VSS – 0.3 to VDD + 0.3	V
V_{I3}	Input Voltage for RTCXI and RTCXO		
V_{I4}	Input Voltage for UHD+ and UHD-	-0.5 to 5.5V	V
V_{O1}	Output Voltage (except UHD+ and UHD-)	VSS – 0.3 to VDD3I/O + 0.3	V
V_{O2}	Output Voltage for UHD+ and UHD-	-0.5 to 5.5	V
T_{STG}	Storage temperature	-60 to 150	°C

Warning: Stresses above those listed as "absolute maximum ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

7.2 Recommended Operating Conditions

Table 23. Recommended Operating Conditions

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
T_A	Operating temperature range (see note 1)		-40		105	°C
$V_{DD3I/O}$	Power supply for I/O ring		3.0	3.3	3.6	V
V_{DD3PLL}	Power supply of analog blocks inside PLL		3.0	3.3	3.6	V
V_{DD}	Power supply for core logic		1.62	1.8	1.98	V
V_{DDRTC}	Power supply for RTC block		1.62	1.8	1.98	V
f_{OSC1}	Main oscillator frequency			25		MHz
f_{OSC2}	Oscillator for RTC			32.768		KHz

Note: 1 The device is characterized between -40°C and 105°C and tested at 25°C and 85°C.

7.3 DC Electrical Characteristics

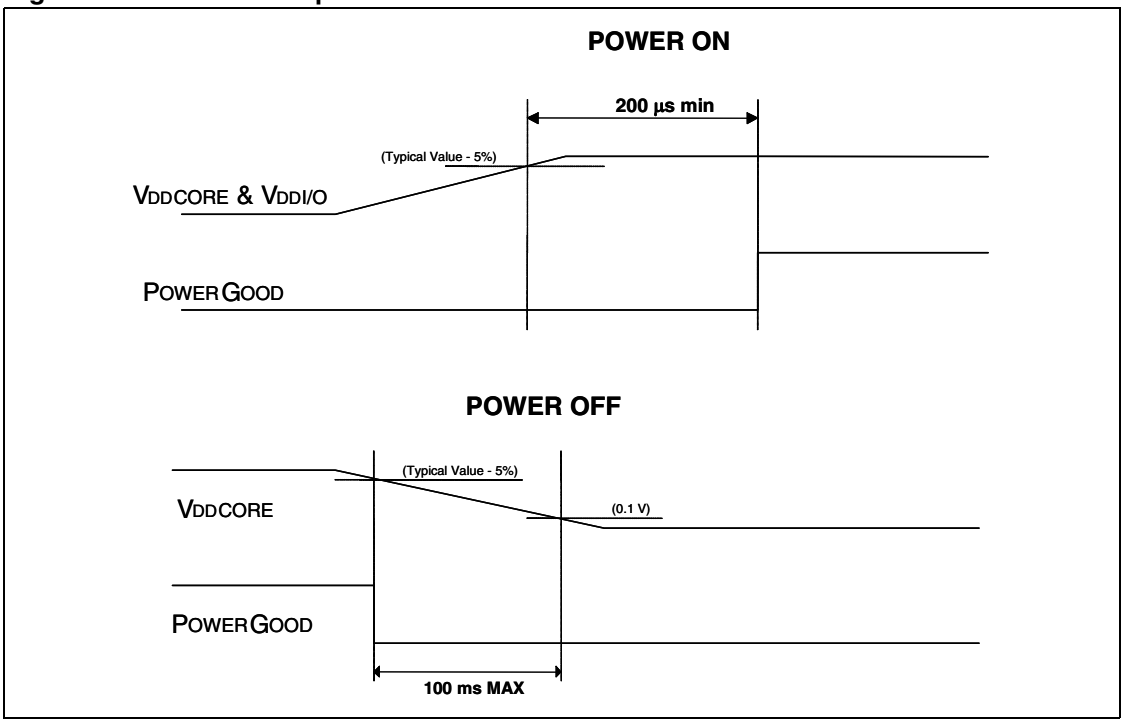
Table 24. DC Electrical Characteristics
(T_A = 0°C to +105°C and V_{DD} = 5V unless otherwise specified)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
V_{IL}	Input low level voltage All input pins except ……..				0.8	V
V_{IH}	Input High level voltage All input except …….		2			V
V_{HYS}	Hysteresis voltage (note 1)		0.4			V
V_{OL}	Low level output voltage	I_{OL} = X mA (note 2)			0.4	V
V_{OH}	High level output voltage	I_{OL} = X mA (note 2)	2.4			V
I_{IL} - I_{IH}	Input leackage current	$V_{SS} < V_{IN} < V_{DD3I/O}$	-10		+10	µA
R_{PU}	I/O weak pull up (note 3)		30	50	100	KΩ

Note: 1 See the Table 2 to determine which pins have hysteresis.
2 See the Table 2 to determine the drive capability of the pads.
3 See the Table 2 to determine which pins have pull-up.

7.3.1 PowerGood timing requirement

Figure 28. PowerGood requirement



7.4 AC Electrical characteristics

Table 25. Core power consumption (V_{DD} = 1.8V, T_A = 25°C)

Symbol	Parameter	Test Conditions	Value			Unit
			Min.	Typ.	Max.	
I_{CORE}	Core current consumption	System clock freq. 48MHz, running code in internal SRAM		47		mA
I_{CORE}	Core current consumption	System clock freq. 48MHz, running code in external Flash memory		40		mA
I_{CORE}	Core current consumption	System clock freq. 48MHz, running code in external SDRAM		44		mA

7.5 External Memory Bus Timing

7.5.1 Timings for External CPU writing access

This section deals with expected timings for external CPU writing access. Next diagram shows expected timing, as detailed in the following Table .

Figure 29. External CPU writing timings

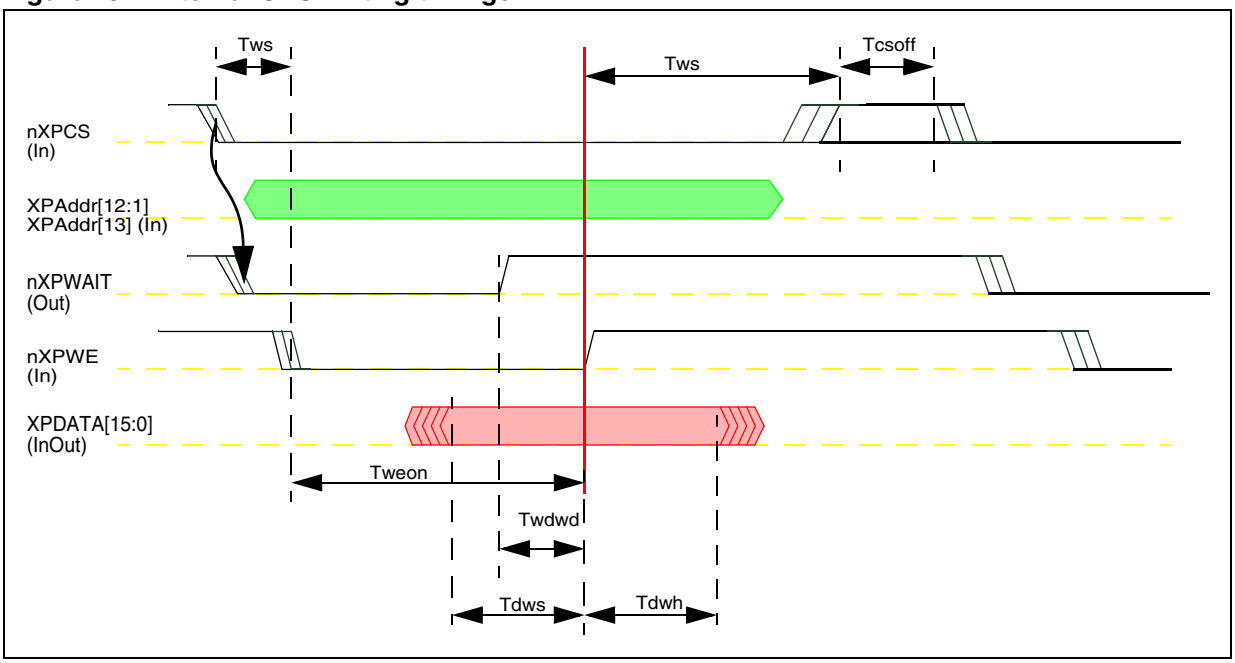


Table 26. Expected timings for external CPU writing access(1)

Symbol	Parameter	Test conditions	Min	Typ	Max	Unit
Tws	nXPWE asserted versus nXPCS asserted		0			ns
Tws	nXPWE deasserted versus nXPCS deasserted		0			ns
Tdws	Data setup vs nXPWE rising edge		2			ns
Tdwh	Data hold vs nXPWE rising edge		2			ns
Twdwd	nXPWAIT deasserted vs nXPWE deasserted		0			ns
Tcsoff	Chip select off pulse width		45			ns
Tweon	Write enable pulse width		45			ns

1. nXPCS is the first signal asserted at the beginning of the writing cycle, as well as the last signal deasserted at the end of the writing cycle.
nXPWE is asserted after, or at the same time, with respect to the assertion of nXPCS. nXPWE assertion must be insensitive to nXPWAIT asserted.
However, nXPWE deassertion MUST take place with nXPWAIT deasserted. In other words, even though nXPWAIT is asserted, nXPWE must assert, while deassertion of nXPWE has to wait until nXPWAIT deassertion.
Once a writing access has finished, nXPCS must be kept deasserted for at least 45 ns, before starting next reading/writing access.

7.5.2 Timings for External CPU reading access

Timings for reading are about the same with respect to the writing access. The main difference is due to the Data, which becomes valid at the rising edge of nXPWAIT, and it is kept until the end of the cycle.

Figure 30. External CPU reading timingss

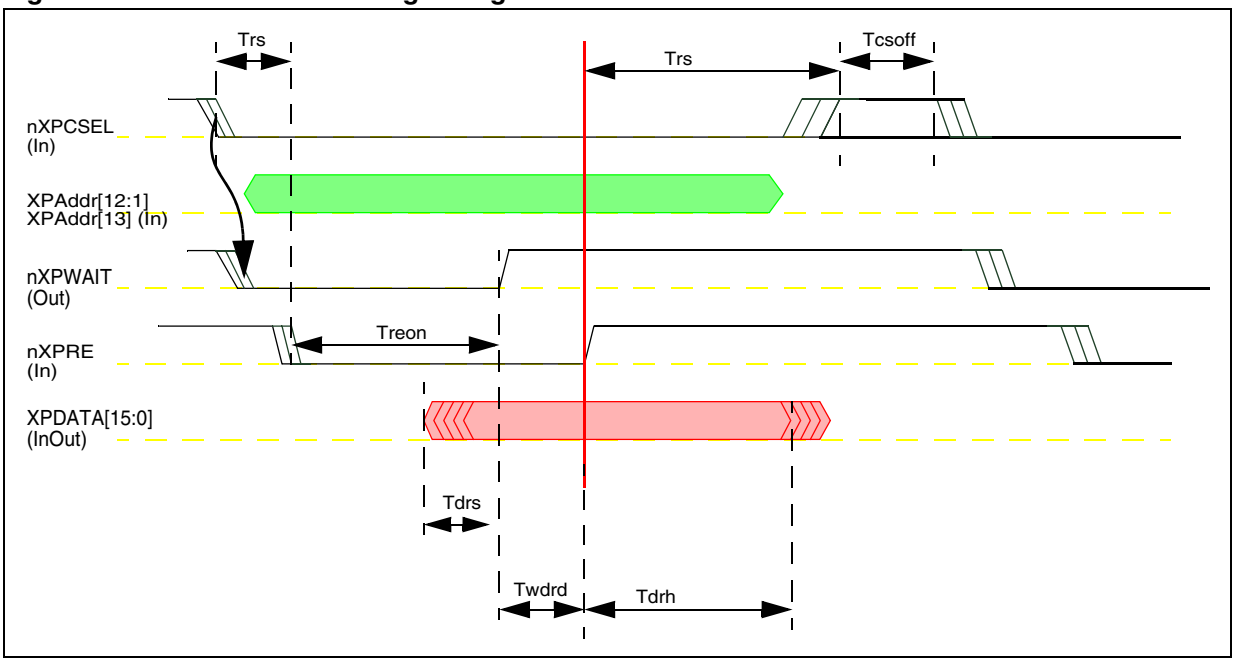


Table 27. Expected timings for external CPU reading access(1)

Symbol	Parameter	Test conditions	Min	Typ	Max	Unit
Trs	nXPRE asserted versus nXPCS asserted		0			ns
Trs	nXPRE deasserted versus nXPCS deasserted		0			ns
Tdrs	Data setup vs nXPWAIT rising edge		0			ns
Tdrh	Data hold vs nXPRE rising edge		20			ns
Twdrd	nXPWAIT deasserted vs nXPWE deasserted		0			ns
Tcsoff	Chip select off pulse width		45			ns
Treon	Read enable pulse width		45			ns

1. nXPCS is the first signal asserted at the beginning of the reading cycle, as well as the last signal deasserted at the end of the reading cycle.
nXPRE is asserted after, or at the same time, with respect to the assertion of nXPCS. nXPRE assertion must be insensitive to nXPWAIT asserted.
However, nXPRE deassertion MUST take place with nXPWAIT deasserted. In other words, even though nXPWAIT is asserted, nXPRE must assert, while deassertion of nXPRE has to wait until nXPWAIT deassertion.
Once a reading access has finished, nXPCS must be kept deasserted for at least 45 ns before starting a new reading/ writing access.

8 Reference Document

#	Name	ID	Description	Reference	Where
1	ARM720T	ARM720T	ARM720T Datasheet	ARM DDI 0087E	www.arm.com
2	ARM720T	ARM720T	ARM720 Technical Reference	ARM DDI0229A	www.arm.com
3	AMBA Bus	AMBA	AMBATM Specification Rev 2.0	ARM IHI 0011A	www.arm.com
4	IEEE1284 - ECP	‘Extended Capabilities Port Protocol and ISA Interface Standard Rev 1.12’			www.microsoft.com
5	OpenHCI	Open Host Controller Interface Specification for USB	Release 1.0a Compaq Microsoft National Semiconductor		http://h18000.www1.hp.com/productinfo/development/openhci.html

9 Package Information

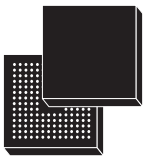
In order to meet environmental requirements, ST offers these devices in ECOPACK® packages. These packages have a Lead-free second level interconnect. The category of second Level Interconnect is marked on the package and on the inner box label, in compliance with JEDEC Standard JESD97. The maximum ratings related to soldering conditions are also marked on the inner box label.

ECOPACK is an ST trademark. ECOPACK specifications are available at: www.st.com.

Figure 31. LFBGA180 Mechanical Data & Package Dimensions

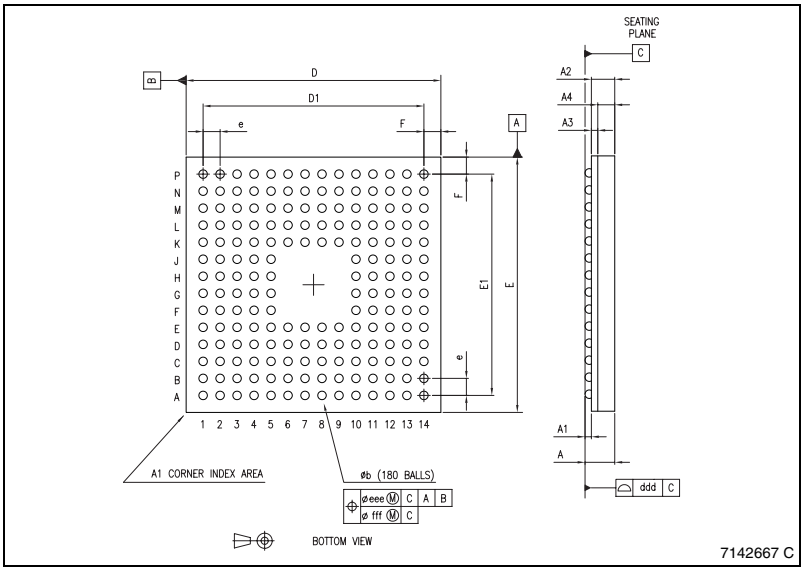
DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			1.7			0.0669
A1	0.21			0.0083		
A2		1.085			0.0427	
A3		0.3			0.0118	
A4			0.8			0.0315
b	0.35	0.4	0.45	0.0138	0.0157	0.0177
D	11.85	12	12.15	0.4665	0.4724	0.4783
D1		10.4			0.4094	
E	11.85	12	12.15	0.4665	0.4724	0.4783
E1		10.4			0.4094	
e		0.8			0.0315	
f		0.8			0.0315	
ddd			0.1			0.0039
eee			0.15			0.0059
fff			0.08			0.0031

OUTLINE AND MECHANICAL DATA



Body: 12 x 12 x 1.7mm

LFBGA180
Low Profile **F**ine Pitch **B**all **G**rid **A**rray



10 Revision history

Date	Revision	Changes
20-Sep-2005	1	Initial release.
30-Jan-2006	2	The staus is changed from “Preliminary data” to “Maturity”. Corrected a typing error in the title of the *Section 6.4 on page 96*. Updated the mechanical data in the “Package information” section.
28-Feb-2006	3	Modified *Section 6.3.1 on page 68*.
14-Apr-2006	4	Added new chapters *Section 7.4 on page 188* & *Section 7.5 on page 189*.
03-May-2006	5	Modified *Figure 11 on page 121*. Modified *Section 6.12 on page 162*. Modified *Table 22 on page 186*.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZE REPRESENTATIVE OF ST, ST PRODUCTS ARE NOT DESIGNED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS, WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com