NET+Works™ for NET+ARM Hardware Reference Guide

This guide provides a description of the NET+ARM 15/40 hardware.

Part Number/Version: 8833198D

Release Date: September, 2000

NETsilicon, Inc.
411 Waverley Oaks Road Suite 227
Waltham, MA 02452
Tel: (781) 647-1234 or (800) 243-2333
Home Page: www.netsilicon.com

Important

No title to or ownership of the software described in this document or any of its parts, including patents, copyrights and trade secrets, is transferred to customers. It is against the law to decipher, decompile, or develop source code for the software described in this document, or knowingly allow others to do so.

NETsilicon makes no representations or warranties regarding the contents of this document. Information in this document is subject to change without notice and does not represent a commitment on the part of NETsilicon. This manual is protected by United States Copyright Law, and may not be copied, reproduced, transmitted or distributed, in whole or part, without the express prior written permission of NETsilicon.

Copyright Notice

© 2000 NETsilicon, Inc.
Printed in the United States of America. All rights reserved.

Trademarks

ARM is a registered trademark of Advanced RISC Machines, Limited.

Green Hills Software and the Green Hills logo are trademarks, and MULTI is a registered trademark of Green Hills Software, Inc.

NETsilicon, the NETsilicon logo and NET+Works are trademarks of NETsilicon, Inc.

NET+ARM is a trademark of ARM Limited and is exclusively sublicensed to NETsilicon, Inc.

pSOS+ and pRISM+ are trademarks of Wind River Systems, Inc.

VxWorks is a registered trademark and Tornado is a trademark of Wind River Systems, Inc.

All other brand and product names are trademarks, service marks, registered trademarks, or registered service marks of their respective companies.

Important Notice

NETsilicon reserves the right to make changes to its products without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

Certain applications using semiconductor products may involve potential risks of death, injury, or severe property or environmental damage (“Critical Applications”).

NETSILICON PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS, OR OTHER CRITICAL APPLICATIONS.

Inclusion of NETsilicon products in such applications is understood to be fully at the risk of the customer. Use of NETsilicon products in such applications requires the written approval of an appropriate NETsilicon officer.

NETsilicon assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does NETsilicon warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of NETsilicon covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

Conventions

Conventions used in this guide are described as follows.

Convention	Description
Italic type	Used for emphasis and for book and manual titles, and reference documents.
Bold type	Used to indicate specific choices within instructions in procedures. For example, Click on **Software Installation**.
signal, signal*	All signals are active high unless suffixed with an asterisk (*) in which case they are active low.

Note:

- All configuration register settings are reset to their inactive state unless otherwise noted.
- All addresses are defined in hexadecimal unless otherwise noted.

Structure of this Guide

- Chapter 1, *Introduction*, contains information about the NET+ARM chip, its features, the applications in which it can be used, and the pins that support the ENI interface.
- Chapters 2 through 11 contain information on the CPU core and the various modules.
- Chapter 12, *Test Support*, contains information on the various tests that NET+ARM supports, including ATPG, PLL, BIST, ATE, and ARM debugging.
- Chapter 13, *Electrical Specifications*, contains information on DC and AC characteristics, output rise and fall timing, and crystal oscillator specifications.
- Chapter 14, *NET+ARM Package Guide*, contains information on the NET+ARM chip package and dimensions.

Related Documentation

- *NET+Works Getting Started Guide*
- *NET+Works Software Reference Guide* provides a description of the Application Programming Interfaces (APIs) for developing HTML pages.
- *NET+Works User's Guide* explains how to use the NET+Works toolkit to develop programs for your application and hardware.

Customer Support

We invite your comments and recommendations about both our products and our documentation. In the event that you have a question or technical problem with this product, you can contact us as follows:

Documentation Comments	techpubs@netsilicon.com
Technical Support	**Phone:** (800) 243-2333 / (781) 647-1234 **Fax:** (781) 893-1338 **Email:** tech_support@netsilicon.com

NETsilicon Home Page	www.netsilicon.com
Online Problem Reporting	www.netsilicon.com/5supp_set/bug_report.htm

You are encouraged to visit the Support section of the NETsilicon web site for documentation and software updates, FAQs, and technical information.

If you want to address a specific technical issue with the product, use the bug reporting system indicated as the online problem reporting. After the information entered into this system is analyzed, an engineer will call you regarding the problem.

Table of Contents

Chapter 1
Introduction

The NETsilicon NET+ARM chip is a single chip 32-bit RISC processor containing an integrated 10/100 Mbit Ethernet MAC and all the peripherals (other than RAM or ROM) required to complete an embedded networking peripheral application.

1.1 NET+ARM Chip Overview

CPU Core

- 32-bit RISC Processor
- 3rd Party Software Support
- 32-bit Internal Bus
- 2 Programmable Timers
- 2 Async Serial Ports
- 4K Cache (NET+40 only)

Bus Interface

- 8-bit, 16-bit, and 32-bit peripherals
- 28-bit External Address Bus
- Multi-master Support
- Normal and Burst Cycles
- 5 Programmable Chip-Selects
- Glueless Interface Flash & DRAM
- Configurable Endian Support

Integrated Ethernet Support

- 10/100 Mbit Media Access Controller MII Interface to External Ethernet PHY
- Bi-directional Capability
- Address Filtering
- Dedicated DMA Support

Integrated ENI Interface

- 4-Port IEEE 1284 Host Interface
- ENI Host Interface
- Register-Mode Interface
- DMA Support

Low Power 3.3V Operation

1.2 NET+ARM Chip Key Features

32-bit ARM7TDMI RISC Processor

- Full 32-bit ARM Mode
- Extremely Cost-efficient 16-bit Thumb Mode
- 15 General Purpose 32-bit Registers
- 32-bit Program Counter and Status Register
- 5 Supervisor Modes, One User Mode
- 15 MIPS Peak Performance - NET+ARM-15 chip and 40 MIPS Peak Performance - NET+ARM-40 chip
- 3rd-Party Software Support

Integrated 10/100 Ethernet MAC

- 10/100 MII-based PHY Interface
- 10Mbit ENDEC Interface
- Supports TP-PMD and Fiber-PMD Devices
- Full Duplex
- Optional 4B/5B Coding
- Full Statistics Gathering (SNMP & RMON)
- Station, Broadcast, Multicast Address Detection & Filtering
- 128 Byte Transmit FIFO
- 2K Receive FIFO
- Intelligent Receive Side Buffer Selection

10 Channel DMA Controller

- 2 Dedicated to Ethernet Transmit and Receive
- 4 Dedicated to Serial Transmit and Receive
- 4 Dedicated to P1284/ENI Interface
- Flexible Buffer Management
- 2 Channels Configurable for External Peripherals

P1284/ENI Interface

- 4 IEEE 1284 Parallel Ports
- 64K Shared RAM ENI Interface (8 or 16 bit)
- Full Duplex FIFO Mode Interface (8 or 16 bit)
- 32 Byte Transmit/Receive FIFO Mode FIFOs

Serial Ports

- 2 Fully Independent Serial Ports (UART, HDLC, SPI)
- 32-Byte Transmit/Receive FIFOs
- Internal Programmable Bit-rate Generators
- Bit Rates from 75 to 518400 in 16X Mode
- Bit Rates from 1200 to 4Mbps in 1X Mode
- Odd, Even, or No Parity
- 5, 6, 7, or 8 Bits
- 1 or 2 Stop Bits
- Both Internal and External Clock Support
- Receive Side Character and Buffer Gap Timers
- 4 Receive Side Data Match Detectors

Bus Interface

- 5 Independent Programmable Chip Selects
- Supports 8-, 16-, and 32-bit Peripherals
- Supports External Address Decoding and Cycle Termination (32-bit peripherals only)
- Supports Dynamic Bus Sizing
- Supports ASYNC and SYNC Peripheral Timing

- All Chip Selects Supports SRAM, FD/EDO DRAM, SDRAM, Flash, EEPROM Without External Glue
- Internal DRAM Address Multiplexing
- Internal Refresh Controller (CAS before RAS)
- 256M Byte Addressing per Chip Select
- Burst-mode Support
- 0-15 Wait States per Chip Select
- Bootstrap Support
- External Bus Master Support
- Supports Internal or External Bus Arbiters

Timers

- Two Independent Programmable Timers (200 us to 500 ms)
- Programmable Watch-Dog Timer (Interrupt or Reset on Expiration)
- Programmable Bus Timer

General Purpose I/O

- 24 Programmable I/O Interface Pins
- 4 Pins with Programmable Interrupt

Clock Generator

- Requires Only a Simple External Crystal
- On-board Programmable Phase Lock Loop
- Supports Direct External Clock Input as well

Package

- 208-pin Plastic Quad Flat Pack (0.020 inch; 0.5 mm pitch)

Power

- 750mW Maximum (outputs switching)

Operating Voltage

- 3.0 - 3.6 Volts

Figure 1-1 provides an overview of the modules that make up the NET+ARM chip.

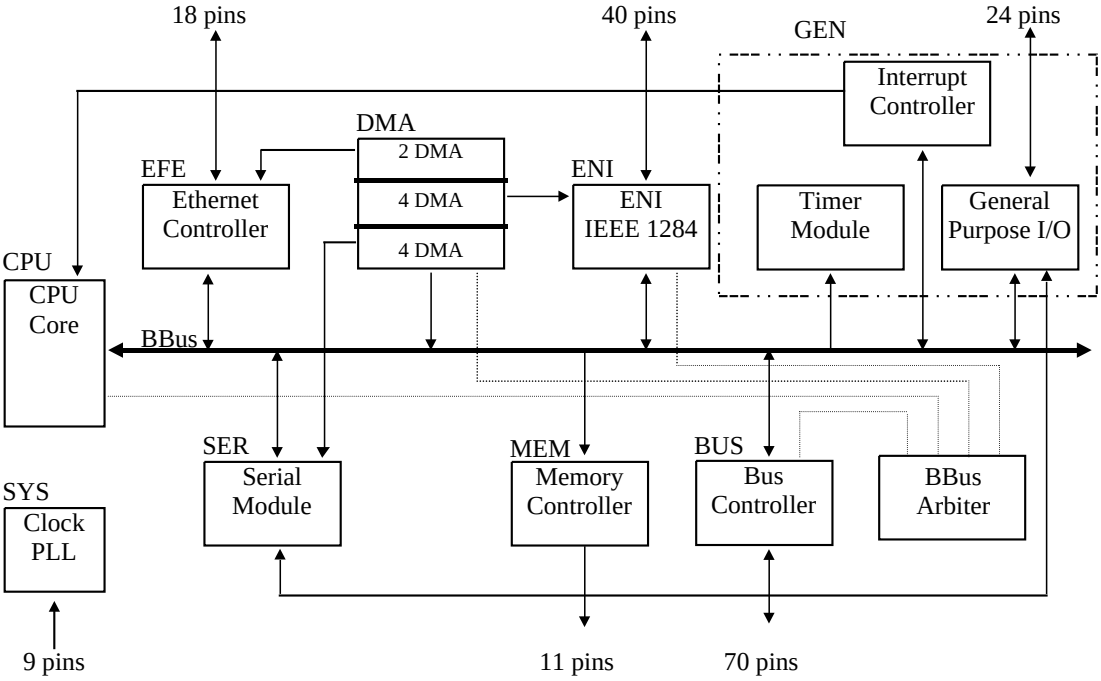


Figure 1-1: NET+ARM Chip Module Block Diagram

1.3 NET+ARM Chip Applications

The NET+ARM chip can be used in any embedded environment requiring networking services in an Ethernet LAN. The NET+ARM chip contains an integrated ARM RISC processor, 10/100 Ethernet MAC, serial ports, IEEE 1284 parallel ports, memory controllers, and parallel I/O. The NET+ARM chip can interface with another processor using a register or shared RAM interface. The NET+ARM chip provides all the tools required for any embedded networking application.

Figure 1-2 shows a typical hardware design used in an embedded networking application. The NET+ARM chip can attach to another processor system using the 1284, ENI shared RAM, or ENI FIFO interfaces. Also, additional application-specific hardware can be attached to the NET+ARM chip system bus for custom applications that make use of the NET+ARM chip internal RISC processor.

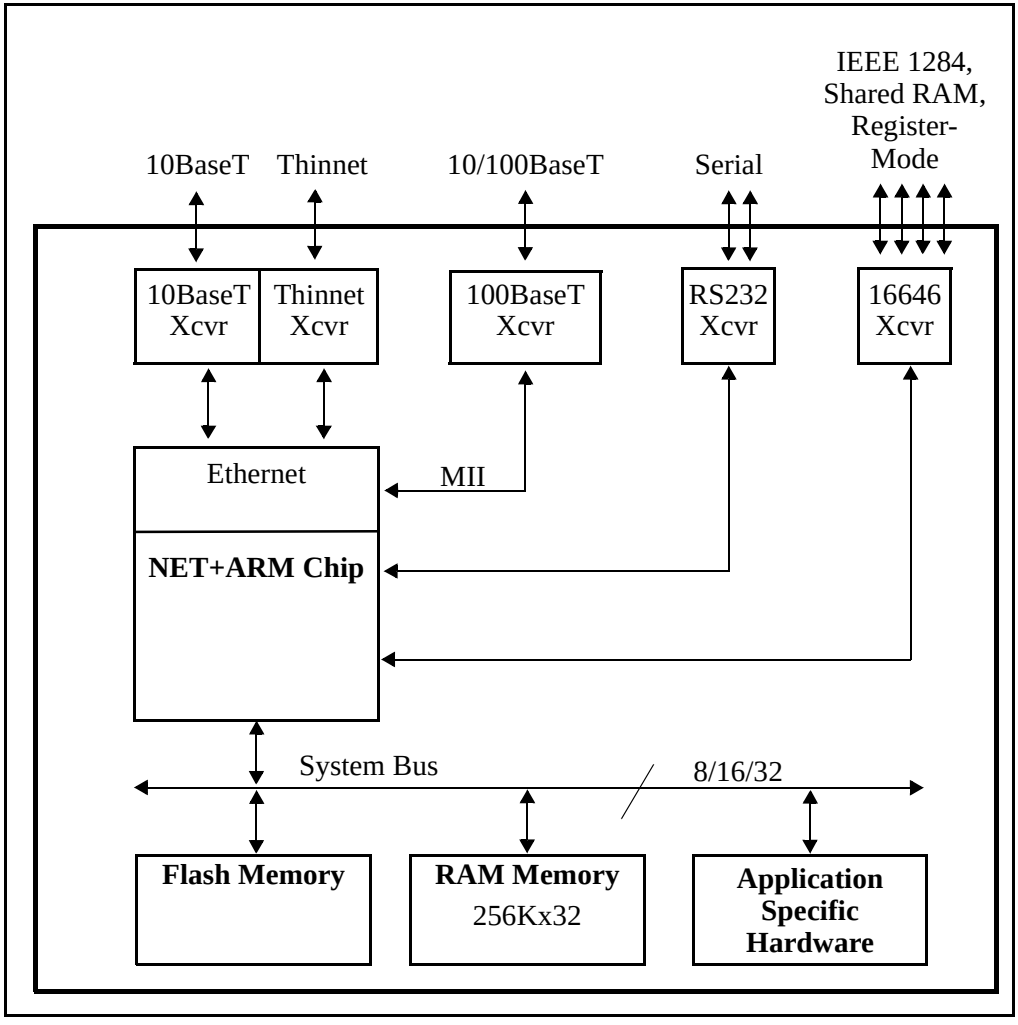


Figure 1-2: NET+ARM Chip Hardware Block Diagram

1.4 NET+ARM Chip Pinout

Refer to Table 1-1 for details concerning the pinout of the NET+ARM chip.

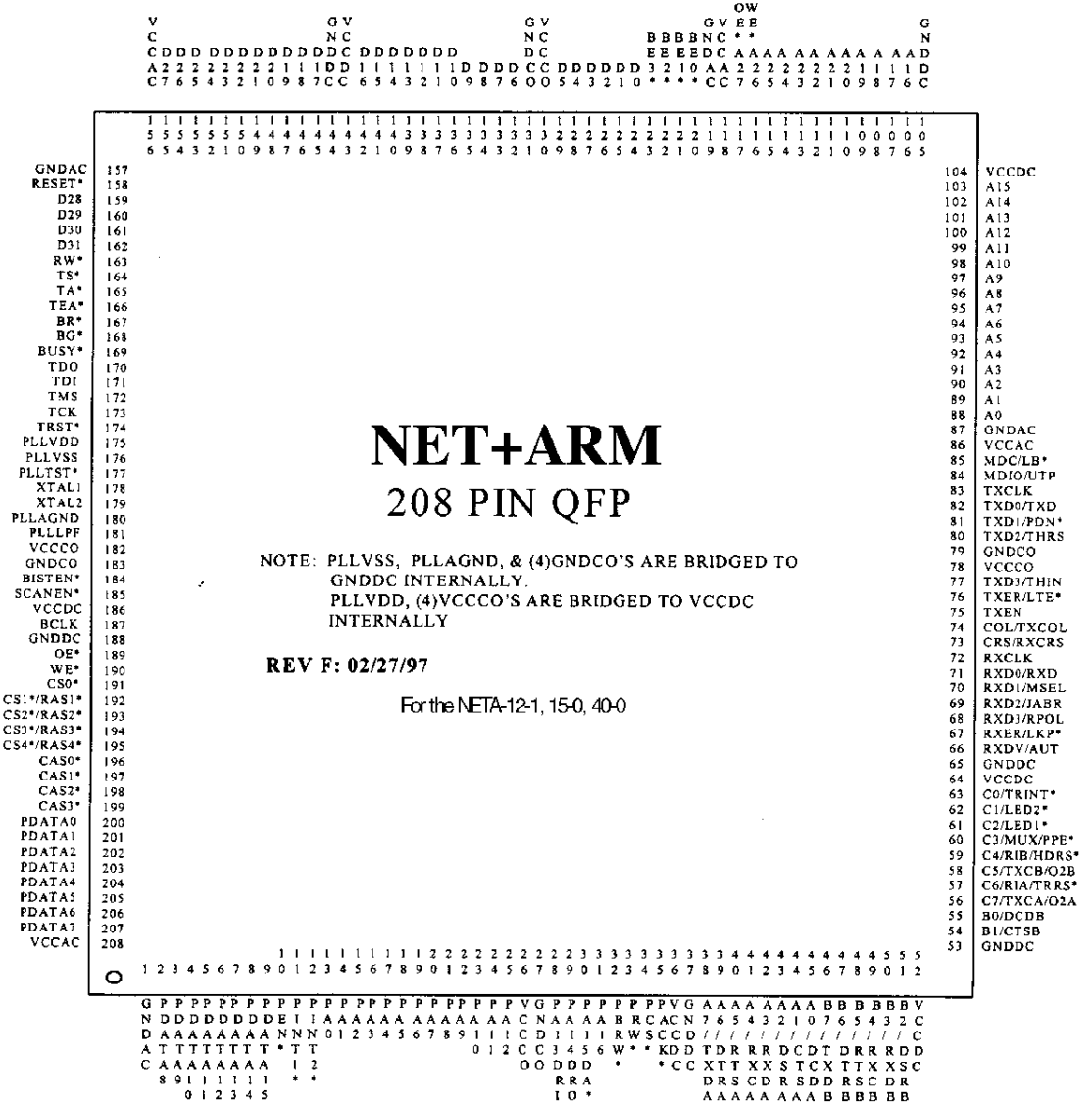


Figure 1-3: NET+ARM Chip Pinout

The *Signal* column identifies the pin name for each I/O signal. Note that some signals have dual modes and are identified accordingly. The mode is configured via firmware using a configuration register (some modes may require hardware configuration during a RESET condition).

The *208QFP* column identifies the pin number assignment for a specific I/O signal.

The *I/O* column identifies whether the signal is input, output, or input/output.

The *Drive* column identifies the drive strength of an output buffer. The NET+ARM chip is being designed using one of three types of drivers; **(2mA), ** (4mA), and **(8mA).

The *Pad Cell* column indicates the type of I/O cell driver selected. All Pad Cells with a suffix of **u** provide an internal pullup resistor. All Pad Cells with a suffix of **d** provide an internal pulldown resistor. The pullup and pulldown resistors can effectively terminate floating inputs, however, these resistors are not strong enough to provide a sharp rising edge for outputs transitioning from Vol to tri-state.

Signal		208 QFP Pin	Pad Cell	I/O	Drive	Description
System Bus Interface						
BCLK		187	pt3t03	O	8	Synchronous Bus Clock
ADDR27	CS0OE*	117	pt3b02u	I/O	4	Address Bus
ADDR26	CS0WE*	116	pt3b02u	I/O	4	
ADDR25		115	pt3b02u	I/O	4	
ADDR24		114	pt3b02u	I/O	4	
ADDR23		113	pt3b02u	I/O	4	
ADDR22		112	pt3b02u	I/O	4	
ADDR21		111	pt3b02u	I/O	4	
ADDR20		110	pt3b02u	I/O	4	
ADDR19		109	pt3b02u	I/O	4	
ADDR18		108	pt3b02u	I/O	4	
ADDR17		107	pt3b02u	I/O	4	
ADDR16		106	pt3b02u	I/O	4	
ADDR15		103	pt3b02u	I/O	4	
ADDR14		102	pt3b02u	I/O	4	
ADDR13		101	pt3b02u	I/O	4	
ADDR12		100	pt3b02u	I/O	4	

Table 1-1: NET+ARM Chip Pinout

Signal	208 QFP Pin	Pad Cell	I/O	Drive	Description
ADDR11	99	pt3b02u	I/O	4	
ADDR10	98	pt3b02u	I/O	4	
ADDR9	97	pt3b02u	I/O	4	
ADDR8	96	pt3b02u	I/O	4	
ADDR7	95	pt3b02u	I/O	4	
ADDR6	94	pt3b02u	I/O	4	
ADDR5	93	pt3b02u	I/O	4	
ADDR4	92	pt3b02u	I/O	4	
ADDR3	91	pt3b02u	I/O	4	
ADDR2	90	pt3b02u	I/O	4	
ADDR1	89	pt3b02u	I/O	4	
ADDR0	88	pt3b02u	I/O	4	
DATA31	162	pt3b02	I/O	4	Data Bus
DATA30	161	pt3b02	I/O	4	
DATA29	160	pt3b02	I/O	4	
DATA28	159	pt3b02	I/O	4	
DATA27	155	pt3b02	I/O	4	
DATA26	154	pt3b02	I/O	4	
DATA25	153	pt3b02	I/O	4	
DATA24	152	pt3b02	I/O	4	
DATA23	151	pt3b02	I/O	4	
DATA22	150	pt3b02	I/O	4	
DATA21	149	pt3b02	I/O	4	
DATA20	148	pt3b02	I/O	4	
DATA19	147	pt3b02	I/O	4	
DATA18	146	pt3b02	I/O	4	
DATA17	145	pt3b02	I/O	4	
DATA16	142	pt3b02	I/O	4	
DATA15	141	pt3b02	I/O	4	
DATA14	140	pt3b02	I/O	4	
DATA13	139	pt3b02	I/O	4	
DATA12	138	pt3b02	I/O	4	

Table 1-1: NET+ARM Chip Pinout (Continued)

Signal	208 QFP Pin	Pad Cell	I/O	Drive	Description
DATA11	137	pt3b02	I/O	4	
DATA10	136	pt3b02	I/O	4	
DATA9	135	pt3b02	I/O	4	
DATA8	134	pt3b02	I/O	4	
DATA7	133	pt3b02	I/O	4	
DATA6	132	pt3b02	I/O	4	
DATA5	129	pt3b02	I/O	4	
DATA4	128	pt3b02	I/O	4	
DATA3	127	pt3b02	I/O	4	
DATA2	126	pt3b02	I/O	4	
DATA1	125	pt3b02	I/O	4	
DATA0	124	pt3b02	I/O	4	
TS*	164	pt3b01u	I/O	8	Transfer Start
BE3*	123	pt3b01	I/O	2	Byte Enable D31:D24
BE2*	122	pt3b01	I/O	2	Byte Enable D23:D16
BE1*	121	pt3b01	I/O	2	Byte Enable D15:D08
BE0*	120	pt3b01	I/O	2	Byte Enable D07:D00
RW*	163	pt3b01	I/O	2	Transfer Direction
TA*	165	pt3b03u	I/O	8	Data Transfer Acknowledge
TEA* TLAST*	166	pt3b01u	I/O	8	Data Transfer Error Acknowledge
BR*	167	pt3b02u	I/O	4	Bus Request
BG*	168	pt3b02u	I/O	4	Bus Grant
BUSY*	169	pt3b02u	I/O	4	Bus Busy
Chip-Select Controller					
CS0*	191	pt3t02	O	4	Chip Select (Boot Select)
CS1*	192	pt3t02	O	4	Chip Select / DRAM RAS*
CS2*	193	pt3t02	O	4	Chip Select / DRAM RAS*
CS3*	194	pt3t02	O	4	Chip Select / DRAM RAS*
CS4*	195	pt3t02	O	4	Chip Select / DRAM RAS*
CAS3*	199	pt3t02	O	4	DRAM Column Strobe D31:24
CAS2*	198	pt3t02	O	4	DRAM Column Strobe D23:16
CAS1*	197	pt3t02	O	4	DRAM Column Strobe D15:08

Table 1-1: NET+ARM Chip Pinout (Continued)

Signal		208 QFP Pin	Pad Cell	I/O	Drive	Description	
CAS0*		196	pt3t02	O	4	DRAM Column Strobe D07:00	
WE*		190	pt3t02	O	4	Write Enable	
OE*		189	pt3t02	O	4	Output Enable	
Ethernet Interface							
MII	**10BaseT**					**MII**	**10BaseT**
MDC	LB*	85	pt3t01	O	2	MII Clock	Loopback Enable
MDIO	UTPSTP*	84	pt3b01u	I/O	2	MII Data	Cable Type
TXCLK	TXCLK	83	pc3d01	I		TX Clock	TX Clock
TXD0	TXD	82	pt3t01	O	2	TX Data 0	TX Data
TXD1	PDN* (O.D.)	81	pt3t01	O	2	TX Data 1	Power Down
TXD2	NTHRES	80	pt3t01	O	2	TX Data 2	Norm Thresh
TXD3	THIN	77	pt3t01	O	2	TX Data 3	Enable Thinnet
TXER	LTE	76	pt3t01	O	2	TX Code Err	Link Test Enable
TXEN	TXEN	75	pt3t01	O	2	TX Enable	TX Enable
COL	TXCOL	74	pc3d01	I		Collision	Collision
CRS	RXCRS	73	pc3d01	I		Carrier Sense	Carrier Sense
RXCLK	RXCLK	72	pc3d01	I		RX Clock	RX Clock
RXD0	RXD	71	pc3d01	I		RX Data 0	RX Data
RXD1	MANSENSE	70	pc3d01	I		RX Data 1	Sense Jumper
RXD2	JABBER	69	pc3d01	I		RX Data 2	Jabber
RXD3	REVPOL	68	pc3d01	I		RX Data 3	Reverse Pol
RXER	LINKPUL*	67	pc3d01	I		RX Error	Link Pulse Det
RXDV	AUTOMAN	66	pc3d01	I		RX Data Valid	10B2 Selected
ENI Interface							
IEEE 1284	**ENI**						
PDATA0	PDATA0	200	pt3b01	I/O	2	Parallel Port Data	
PDATA1	PDATA1	201	pt3b01	I/O	2		
PDATA2	PDATA2	202	pt3b01	I/O	2		
PDATA3	PDATA3	203	pt3b01	I/O	2		
PDATA4	PDATA4	204	pt3b01	I/O	2		
PDATA5	PDATA5	205	pt3b01	I/O	2		

Table 1-1: NET+ARM Chip Pinout (Continued)

Signal		208 QFP Pin	Pad Cell	I/O	Drive	Description
PDATA6	PDATA6	206	pt3b01	I/O	2	
PDATA7	PDATA7	207	pt3b01	I/O	2	
POE1*	PDATA8	2	pt3b01	I/O	2	
POE2*	PDATA9	3	pt3b01	I/O	2	
POE3*	PDATA10	4	pt3b01	I/O	2	
POE4*	PDATA11	5	pt3b01	I/O	2	
PCLKC1	PDATA12	6	pt3b01	I/O	2	
PCLKC2	PDATA13	7	pt3b01	I/O	2	
PCLKC3	PDATA14	8	pt3b01	I/O	2	
PCLKC4	PDATA15	9	pt3b01	I/O	2	
PCLKD1	PACK*	35	pt3t03	O	8	
PCLKD2	PEN*	10	pt3t01	O	2	
PCLKD3	PINT1*	11	pt3t01	O	2	
PCLKD4	PINT2*	12	pt3b01	I/O	2	
ACK1*	PA0	13	pc3d01	I		
ACK2*	PA1	14	pc3d01	I		
ACK3*	PA2	15	pc3d01	I		
ACK4*	PA3	16	pc3d01	I		
BUSY1	PA4	17	pc3d01	I		
BUSY2	PA5	18	pc3d01	I		
BUSY3	PA6	19	pc3d01	I		
BUSY4	PA7	20	pc3d01	I		
PE1	PA8	21	pc3d01	I		
PE2	PA9	22	pc3d01	I		
PE3	PA10	23	pc3d01	I		
PE4	PA11	24	pc3d01	I		
PSELECT1	PA12	25	pc3d01	I		
PSELECT2	PA13 / PDRQI*	28	pt3b01	I/O	2	
PSELECT3	PA14 / PDRQO*	29	pt3b01	I/O	2	
PSELECT4	PA15 / PDACK*	30	pc3d01	I		
FAULT1*	PA16	31	pc3d01	I		
FAULT2*	PCS*	34	pc3d01	I		

Table 1-1: NET+ARM Chip Pinout (Continued)

Signal		208 QFP Pin	Pad Cell	I/O	Drive	Description
FAULT3*	PRW*	33	pc3d01	I		
FAULT4*	PBRW*	32	pt3b01	I/O	2	
General Purpose I/O						
PORTA7	TXDA	38	pt3b01u	I/O	2	PORT A
PORTA6	DTRA* / DREQ1*	39	pt3b01u	I/O	2	
PORTA5	RTSA*	40	pt3b01u	I/O	2	
PORTA4	OUT1A* / RxCA	41	pt3b01u	I/O	2	Also external Baud Clock Input
PORTA3	RXDA	42	pt3b01u	I/O	2	
PORTA2	DSRA* / DACK1*	43	pt3b01u	I/O	2	
PORTA1	CTSA*	44	pt3b01u	I/O	2	
PORTA0	DCDA* / DONE1*	45	pt3b01u	I/O	2	
PORTB7	TXDB	46	pt3b01u	I/O	2	PORT B
PORTB6	DTRB* / DREQ2*	47	pt3b01u	I/O	2	
PORTB5	RTSB* / REJECT*	48	pt3b01u	I/O	2	
PORTB4	OUT1B* / RxCB	49	pt3b01u	I/O	2	Also external Baud Clock Input
PORTB3	RXDB	50	pt3b01u	I/O	2	
PORTB2	DSRB* / DACK2*	51	pt3b01u	I/O	2	
PORTB1	CTSB* / RPSF*	54	pt3b01u	I/O	2	
PORTB0	DCDB* / DONE2*	55	pt3b01u	I/O	2	
PORTC7	OUT2A* / TxCA	56	pt3b01u	I/O	4	PORT C
PORTC6	RIA* / IRQ*	57	pt3b01u	I/O	4	
PORTC5	OUT2B* / TxCB	58	pt3b01u	I/O	4	
PORTC4	RIB* / RESET*	59	pt3b01u	I/O	4	
PORTC3	AMUX / CI3	60	pt3b03u	I/O	8	
PORTC2	CI2	61	pt3b03u	I/O	8	
PORTC1	CI1	62	pt3b03u	I/O	8	
PORTC0	CI0	63	pt3b03u	I/O	8	

Table 1-1: NET+ARM Chip Pinout (Continued)

Signal	208 QFP Pin	Pad Cell	I/O	Drive	Description
Clock Generation					
XTAL1	178		I		Crystal Oscillator Circuit
XTAL2	179		O		
PLLVDD	175				PLL Clean Power
PLLVSS	176				PLL Clean Ground
PLLAGND	180				PLL Loop Filter Return
PLLLPF	181				PLL Loop Filter Capacitor
PLLTST*	177	pc3d01u	I		PLL Test Mode
BISTEN*	184	pc3d01u	I		Enable Internal BIST operations
SCANEN*	185	pc3d01u	I		Enable Internal SCAN testing
System Reset					
RESET*	158	pc3d01u	I		System Reset
JTAG Test					
TDI	171	pc3d01u	I		Test Data In
TDO	170	pt3t01	O	2	Test Data Out
TMS	172	pc3d01u	I		Test Mode Select
TRST*	174	pc3d01d	I		Test Mode Reset
TCK	173	pc3d01	I		Test Mode Clock
Power Supply					
VCC	26,36,52,64,78,86,104,118,130,143,156,182,186,208				
GND	1,27,37,53,65,79,87,105,119,131,144,157,183,188				

Table 1-1: NET+ARM Chip Pinout (Continued)

1.5 Signal Description

1.5.1 System Bus Interface

The NET+ARM chip uses the system bus interface to interface with memory mapped peripheral devices such as Flash, SRAM, DRAM, and so on.

BCLK **Bus Clock**

The BCLK signal provides the bus clock. All system bus interface signals are referenced to the BCLK signal. The BCLK signal can be configured to operate at multiple speeds relative to the internal PLL-based system clock. The BCLK can be configured to operate in 1/4, 1/2,or full speed mode. A field in the system control register controls BCLK configuration. (See Section *8.2.1 System Control Register*.)

ADDR Address Bus

The address bus identifies the address of the peripheral that the current bus master is addressing. The NET+ARM chip supports a maximum of 28 address bits (256 Mbytes of addressing). The address bus is bi-directional. When the NET+ARM chip is the system bus master, the address bus operates as an output. When an external bus master owns the system bus, the address bus is an input to the NET+ARM chip.

The upper three address bits, A27, A26, and A25 can be configured for alternate modes. The alternate modes are controlled by fields within the Memory Module Configuration Register (MMCR).

The A27 bit can be alternately configured for CS0OE*. The CS0OE* signal provides an internal gating of the CS0* and OE* signals. The CS0OE* signal is the logical “OR” of CS0* and OE*. The CS0OE* signal can maximize access timing from Flash ROM devices. The CS0OE* signals allow the CE* pin for the external CS0 device to connect to logic 0 causing the CS0 peripheral access timing to be referenced from address instead of CE*. The CS0OE* is only driven while the NET+ARM chip is a bus master. CS0OE* is not driven when the NET+ARM chip is a bus slave.

The A26 bit can be alternately configured for CS0WE*. The CS0WE* signal provides an internal gating of the CS0* and WE* signals. The CS0WE* signal is the logical “OR” of CS0* and WE*. The CS0WE* signal can be used to maximize access timing from Flash ROM devices. The CS0WE* signals allow the CE* pin for the external CS0 device to be connected to logic 0 causing the CS0 peripheral access timing to be referenced from address instead of CE*. The CS0WE* is only driven while the NET+ARM chip is a bus master. CS0WE* is not driven when the NET+ARM chip is a bus slave.

The A25 bit can be alternatively configured for BLAST*. The BLAST* signal is driven by the current system bus master to signal the end of a burst memory cycle. The system bus master drives the A25 signal high to indicate it wishes to continue a burst cycle. The system bus master drives the A25 signal low to indicate it wishes to end the current burst cycle. Refer to section *9.9 External Bus Master Support* for a detailed description of the operation of the A25/BLAST* signal.

DATA Data Bus

The 32-bit data bus provides the data transfer path between the NET+ARM chip and

external peripheral devices. The data bus is bi-directional. When the NET+ARM chip is the system bus master, the data bus is an output during write cycles and input during read cycles. When an external bus master is the system bus master, the data bus is an input during write cycles and an output during read cycles.

Eight-bit peripheral devices must always attach to the NET+ARM chip using D31:D24. Sixteen-bit peripheral devices must always attach to the NET+ARM chip using D31:D16. Thirty-two-bit peripheral devices attach to the NET+ARM chip using D31:D00.

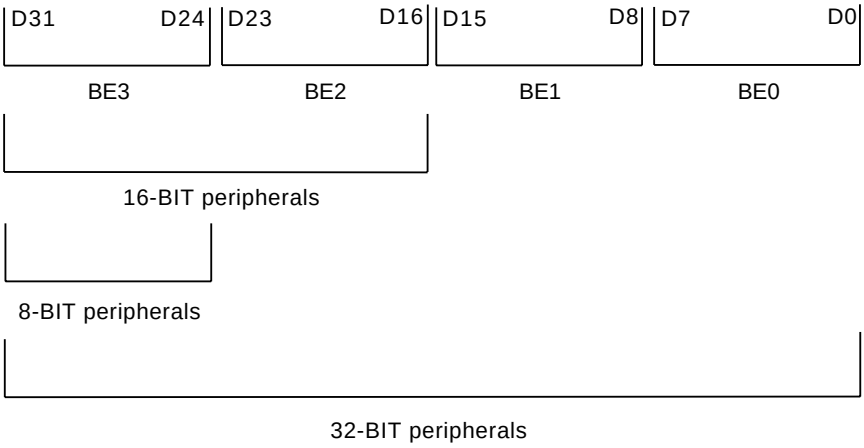


Figure 1-4: Data Bus Mapping

NET+ARM Chip Active Data Bus Lane	NET+ARM Chip Byte Enable	NET+ARM Chip DRAM CAS	External 32-bit Peripheral	External 16-bit Peripheral	External 8-bit Peripheral
D31:D24	BE3*	CAS3*	•	•	•
D23:D16	BE2*	CAS2*	•	•	
D15:D08	BE1*	CAS1*	•		
D07:D00	BE0*	CAS0*	•		

Table 1-2: Data Bus / Enable Cross Reference

TS* **Transfer Start**

The transfer start signal identifies the beginning of a system bus memory cycle. The TS* signal is active low for one BCLK cycle at the beginning of the memory cycle.

The transfer start signal is bi-directional. When the NET+ARM chip is the system bus master, TS* operates as an output. When an external bus master owns the system bus, TS* is an input to the NET+ARM chip.

BE* **Byte Enable**

The byte enable signals identify which 8-bit bytes of the 32-bit data bus are active during any given system bus memory cycle. The byte enable signals are active low. The following chart identifies the relationship between the byte enable signals and the data bus lanes.

Byte Enable	Active Data Bus Lane	Little Endian Byte Address	Big Endian Byte Address	Little Endian Word Address	Big Endian Word Address
BE3*	D31:D24	3	0	1	0
BE2*	D23:D16	2	1		
BE1*	D15:D08	1	2	0	1
BE0*	D07:D00	0	3		

Table 1-3: BE* Lane Configuration for 32-bit Peripherals

Byte Enable	Active Data Bus Lane	Little Endian Byte Address	Big Endian Byte Address
BE3*	D31:D24	1	0
BE2*	D23:D16	0	1

Table 1-4: BE* Lane Configuration for 16-bit Peripherals

Byte Enable	Active Data Bus Lane
BE3*	D31:D24

Table 1-5: BE* Lane Configuration for 8-bit Peripherals

The BE* signals are bi-directional. When the NET+ARM chip is the system bus master, the BE* signals operate as outputs. When an external bus master owns the system bus, the BE* signals operate as inputs to the NET+ARM chip.

RW* Read Write Indicator

The read/write signal indicates the direction of the system bus memory cycle. A read operation is identified when RW* is high. A write operation is identified when RW* is low. The RW* signal is bi-directional. When the NET+ARM chip is the system bus master, RW* operates as an output. When an external bus master owns the system bus, RW* is an input to the NET+ARM chip.

TA* Transfer Acknowledge

The transfer acknowledge signal indicates the end of the current system bus memory cycle. The TA* signal is sampled on the rising edge of BCLK. The TA* signal is bi-directional. When the NET+ARM chip memory controller is configured to control the timing of the system bus memory cycle, the NET+ARM chip drives TA*. Otherwise, the external peripheral is responsible for driving TA*.

The TA* signal is also driven by the NET+ARM chip when the external bus master addresses the internal NET+ARM chip registers. TA* asserted active low in conjunction with TEA* asserted active low indicates a burst cycle completion. TA* asserted active low in conjunction with TEA* asserted inactive high indicates a normal memory cycle completion.

TEA* Transfer Error Acknowledge

The TEA* signal indicates an error termination or a burst cycle termination. This signal is bi-directional. The NET+ARM chip or the external peripheral can be used to drive the TEA* signal. TEA* asserted active low in conjunction with TA* asserted inactive high indicates an error termination. The ARM processor takes the data abort exception trap when encountering an error termination during a memory access cycle.

The TEA* signal is asserted by the NET+ARM chip when the bus monitor timer expires in the GEN module. The bus monitor timer can be used to trap address cycles

that are not terminated by a peripheral. TEA* asserted active low in conjunction with TA* asserted active low indicates a burst cycle completion.

TA*	TEA*	Bus State
0	0	Burst Termination
0	1	Normal Termination
1	0	Error Termination
1	1	Waiting

Table 1-6: Transfer Error Acknowledge

BR* Bus Request

The bus request signal is used by a potential bus master to indicate it wishes to acquire the system bus to perform memory cycles. The bus request signal is active low and bi-directional. The direction for BR* is a function of the IARB field in the system control register. The NET+ARM chip can be configured to operate using either an internal or external bus arbiter. When configured to use the internal bus arbiter, the bus request signal is an active low input. When configured to use an external bus arbiter, the bus request signal is an output.

The IARB configuration bit can be automatically initialized to its proper state upon reset by including or not including a pulldown resistor on the A25 signal (refer to section *8.2 GEN Module Hardware Initialization*).

Upon receiving the bus request signal, the bus arbiter is responsible for asserting the bus grant (BG*) signal active low. The BG* signal indicates to the requesting bus master that it can assume ownership of the system bus one BCLK cycle after the current bus master asserts the BUSY* signal inactive high.

BG* Bus Grant

The bus grant signal is asserted active low by the bus arbiter to identify to the bus requestor that it may assume ownership of the system bus one BCLK cycle after the current bus master asserts the BUSY* signal inactive high.

This signal is active low and bi-directional. The direction for BG* depends on the IARB field in the system control register. The NET+ARM chip can use an internal or external bus arbiter. With the internal bus arbiter, the bus grant signal is an active low output. With an external bus arbiter, the bus grant signal is an input.

BUSY* **Bus Busy**

The bus busy signal is asserted active low by the current bus master to indicate it has assumed ownership of the system bus. The current bus master relinquishes ownership of the system bus by driving the BUSY* signal inactive high. The BUSY* signal is a bi-directional signal for the NET+ARM chip.

CS0* **Chip Select 0**

CS1* **Chip Select 1**

CS2* **Chip Select 2**

CS3* **Chip Select 3**

CS4* **Chip Select 4**

The NET+ARM chip supports five unique chip select outputs. Each chip select can be configured to decode a portion of the available address space and can address a maximum of 256M bytes of address space. The chip selects are configured using registers in the memory module.

CS0* is unique in that it can be configured via bootstrap configurations to be enabled at powerup. This allows the ARM processor to boot from a Flash or EPROM device attached to CS0*.

CAS0* **Column Address Strobe**

CAS1*

CAS2*

CAS3*

The CAS* signals are activated when an address is decoded by a chip select module that is configured for DRAM mode. The CAS* signals are active low and provide the column address strobe function for DRAM devices.

The CAS* signals also identify which 8-bit bytes of the 32-bit data bus are active during any given system bus memory cycle.

The following chart identifies the relationship between the CAS* signals and the data bus lanes.

CAS*	Active Data Bus Lane	Little Endian Byte Address	Big Endian Byte Address	Little Endian Word Address	Big Endian Word Address
CAS3*	D31:D24	3	0	1	0
CAS2*	D23:D16	2	1		
CAS1*	D15:D08	1	2	0	1
CAS0*	D07:D00	0	3		

Table 1-7: CAS* Lane Configuration for 32-bit Peripherals

CAS*	Active Data Bus Lane	Little Endian Byte Address	Big Endian Byte Address
CAS3*	D31:D24	1	0
CAS2*	D23:D16	0	1

Table 1-8: CAS* Lane Configuration for 16-bit Peripherals

CAS*	Active Data Bus Lane
CAS3*	D31:D24

Table 1-9: CAS* Lane Configuration for 8-bit Peripherals

WE* Write Enable

The WE* signal is an active low signal indicating a memory write cycle is in progress. This signal is only activated during write cycles to peripherals controlled by one of the five chip selects in the memory module. The WE* signal can have asynchronous timing or synchronous timing. Asynchronous timing differs in that the WE* pulse is inside the low time for the chip select. Asynchronous timing typically supports “Intel Timing” while synchronous mode timing typically supports “Motorola Timing.”

OE* **Output Enable**

The OE* signal is an active low signal indicating a memory read cycle is in progress. This signal is only activated during read cycles from peripherals controlled by one of the five chip selects in the memory module. The OE* signal can have asynchronous timing or synchronous timing. Asynchronous timing differs in that the OE* pulse is inside the low time for the chip select. Asynchronous timing typically supports “Intel Timing” while synchronous mode timing typically supports “Motorola Timing.”

1.5.2 Ethernet MII Interface

MDC **Management Data Clock**

The management data clock provides the clock for the MDIO serial data channel. The MDC signal is a NET+ARM chip output. The maximum frequency is 2.5 MHz.

MDIO **Management Data IO**

The management data IO signal is bi-directional signal providing a serial data channel between the NET+ARM chip and the external Ethernet PHY module.

TXCLK **Transmit Clock**

The transmit clock is an input to the NET+ARM chip from the external PHY module. It provides the synchronous data clock for transmit data. The transmit clock operates at 25Mhz for 100Mbit operation and 2.5Mhz for 10Mbit operation. The transmit clock operates at 10Mhz when operating in ENDEC mode.

TXD3 **Transmit Data**

TXD2 **Transmit Data**

TXD1 **Transmit Data**

TXD0 **Transmit Data**

The NET+ARM chip drives data to the external Ethernet PHY using this nibble bus. All transmit data signals are synchronized to TXCLK. When operating in ENDEC mode, only TXD0 is used for transmit data.

TXER **Transmit Coding Error**

The NET+ARM chip asserts this output when an error has occurred in the transmit data stream. While operating at 100Mbps the PHY responds to this signal by sending “invalid code symbols” on the line.

TXEN **Transmit Enable**

The NET+ARM chip asserts this signal when it drives valid data on the TXD outputs. This signal is synchronized to TXCLK.

COL Transmit Collision

The external Ethernet PHY asserts this input signal when a collision is detected. This input must remain high for the duration of the collision.

CRS Receive Carrier Sense

The external Ethernet PHY asserts this signal whenever the receive medium is non-idle. During half duplex operation, this can also coincide when the transmitter is active.

RXCLK Receive Clock

The receive clock is an input to the NET+ARM chip from the external PHY module. The receive clock provides the synchronous data clock for receive data. It operates at 25Mhz for 100Mbit operation and 2.5Mhz for 10Mbit operation. It operates at 10Mhz when in ENDEC mode. The receive clock must remain active a minimum of 25 bit times after the end of each received Ethernet frame.

RXD3 Receive Data

RXD2 Receive Data

RXD1 Receive Data

RXD0 Receive Data

The NET+ARM chip inputs receive data from the external Ethernet PHY using this nibble bus. All receive data signals are synchronized to RXCLK. When operating in ENDEC mode, only RXD0 is used for receive data.

RXER Receive Error

The external Ethernet PHY asserts this input when it encounters invalid symbols from the network. This signal must be synchronous to RXCLK.

RXDV Receive Data Valid

The external Ethernet PHY asserts this input when it drives valid data on the RXD inputs. This signal must be synchronous to RXCLK.

1.5.3 ENI Interface

The NET+ARM chip uses the ENI interface to interface with other embedded computer systems. The ENI interface provides various models to facilitate this interface, IEEE 1284, shared RAM, and FIFOs.

The ENI interface can only be configured to operate in either IEEE 1284 mode or ENI host mode, but never both at the same time. The ENI host mode supports shared RAM and FIFO interfaces.

The ENI interface mode is configured using a field in the ENI module general control register. This field can be automatically configured at powerup using bootstrap configuration.

1.5.4 ENI Interface Configured for IEEE 1284 Mode

PDATA 1284 Data Bus

The 1284 data bus is a bi-directional data bus that interfaces the NET+ARM chip with the external 1284 port data and control transceivers. The NET+ARM chip only interfaces with a single 1284 port at one moment in time.

POE1* 1284 Channel 1 Output Enable

POE2* 1284 Channel 2 Output Enable

POE3* 1284 Channel 3 Output Enable

POE4* 1284 Channel 4 Output Enable

The channel output enable signals are used to control the output enable for each specific port data register. When a port is configured in output mode, then the output enable signal is driven active low by the NET+ARM chip. When a port is configured in input mode, then the output enable signal for a given port is only driven active low when the NET+ARM chip is reading the inbound data for the specific port.

PCLKC1 1284 Channel 1 Control Clock

PCLKC2 1284 Channel 2 Control Clock

PCLKC3 1284 Channel 3 Control Clock

PCLKC4 1284 Channel 4 Control Clock

The channel control clock signals are driven by the NET+ARM chip to load the 1284 data bus outputs into the various 1284 control transceivers. Only one channel control clock is active at any moment in time. The channel control clocks are never active at the same time as any channel data clocks.

PCLKD1 1284 Channel 1 Data Clock

PCLKD2 1284 Channel 2 Data Clock

PCLKD3 1284 Channel 3 Data Clock

PCLKD4 1284 Channel 4 Data Clock

The channel control clock signals are driven by the NET+ARM chip to load the 1284 data bus outputs into the various 1284 data transceivers. Only one channel data clock is active at any moment in time. The channel control clocks are never active at the same time as any channel data clocks.

ACK1* **1284 Channel 1 1284 Acknowledge**

ACK2* **1284 Channel 2 1284 Acknowledge**

ACK3* **1284 Channel 3 1284 Acknowledge**

ACK4* **1284 Channel 4 1284 Acknowledge**

The channel ACK inputs are an input to the NET+ARM chip from the external 1284 port. The meaning of the ACK input changes as the 1284 operational mode changes. The ACK input can be configured to generate an interrupt on its falling edge.

BUSY1* **1284 Channel 1 1284 BUSY**

BUSY2* **1284 Channel 2 1284 BUSY**

BUSY3* **1284 Channel 3 1284 BUSY**

BUSY4* **1284 Channel 4 1284 BUSY**

The channel BUSY inputs are an input to the NET+ARM chip from the external 1284 port. The meaning of the BUSY input changes as the 1284 operational mode changes.

PE1* **1284 Channel 1 1284 ERROR**

PE2* **1284 Channel 2 1284 ERROR**

PE3* **1284 Channel 3 1284 ERROR**

PE4* **1284 Channel 4 1284 ERROR**

The channel ERROR inputs are an input to the NET+ARM chip from the external 1284 port. The meaning of the ERROR input changes as the 1284 operational mode changes.

PSELECT1* **1284 Channel 1 1284 Peripheral Select**

PSELECT2* **1284 Channel 2 1284 Peripheral Select**

PSELECT3* **1284 Channel 3 1284 Peripheral Select**

PSELECT4* **1284 Channel 4 1284 Peripheral Select**

The channel peripheral select inputs are an input to the NET+ARM chip from the external 1284 port. The meaning of the peripheral input changes as the 1284 operational mode changes.

FAULT1*	**1284 Channel 1 1284 FAULT**
FAULT2*	**1284 Channel 1 1284 FAULT**
FAULT3*	**1284 Channel 1 1284 FAULT**
FAULT4*	**1284 Channel 1 1284 FAULT**

The channel FAULT inputs are an input to the NET+ARM chip from the external 1284 port. The meaning of the FAULT input changes as the 1284 operational mode changes.

1.5.5 ENI Interface Configured for ENI Host Mode

PCS* ENI Chip Select

The PCS* signal provides the chip enable for the ENI interface from the external processor system. When PCS* is driven low, the ENI interface uses the PA address bus to determine which resource is being addressed. After the ENI interface cycle is complete, the ENI interface asserts the PACK* signal to complete the cycle. After PACK* is asserted active, the external processor must de-assert PCS* to complete the ENI interface cycle.

PRW* ENI Read/Write

The PRW* is driven by the external processor system during an ENI access cycle to indicate the data transfer direction. During normal PCS* cycles, PRW* high indicates a READ from the ENI interface; PRW* low indicates a WRITE to the ENI interface. During DMA cycles, PRW* high indicates a WRITE to the ENI interface; PRW* low indicates a READ from the ENI interface.

PA[16:0] ENI Address Bus

The ENI interface requires 17 address signals to identify the resource being accessed. The PA bus must be valid while PCS* is low. During DMA cycles, the PA address bus is ignored. During DMA cycles, only the FIFO data register is accessible.

PA16 low identifies an access to shared RAM. PA16 high identifies access to one of the ENI registers. Refer to section *6.6 ENI Host Interface* for a detailed description of the registers available in the ENI Interface.

When DMA FIFO operations are enabled, the PA15 input is used for the DMA acknowledge input PDACK*. DMA FIFO operations are enabled when the DMAE* bit in the ENI control register is set to 0.

When DMA FIFO operations are enabled, the PA14 output is used for the active low outbound FIFO DMA ready output PDRQO*. The PDRQO* signal is only routed through PA14 when both DMAE* and DMAE2 in the ENI control register are both set to 0. When DMAE* is set low and DMAE2 is set high, an active high version of

PDRQO is routed out the active high PINT2 pin instead. The PDRQO and PDRQI signals are internally “ORed” together before driving the PINT2 pin.

When DMA FIFO operations are enabled, the PA13 output is used for the active low inbound FIFO DMA ready output PDRQI*. The PDRQI* signal is only routed through PA13 when both DMAE* and DMAE2 in the ENI control register are set to 0. When DMAE* is set low and DMAE2 is set high, an active high version of PDRQI is routed out the active high PINT2 pin instead. The PDRQO and PDRQI signals are internally “ORed” together before driving the PINT2 pin.

PDACK*/PA15 ENI DMA Acknowledge

When DMA FIFO operations are enabled, the PA15 input is used for the DMA acknowledge input PDACK*. DMA FIFO operations are enabled when the DMAE* bit in the ENI control register is set to 0.

The PDACK* input indicates a DMA cycle is in progress. Only the FIFO data register is accessed during a DMA cycle. During DMA cycles, the PCS* signal must not be asserted. During DMA cycles, the PA bus (other than PA15) is ignored. During DMA cycles, the PRW* defines the data transfer direction. PRW* high indicates a FIFO write operation while PRW* low indicates a FIFO read operation.

PDRQO*/PDRQO/PA14 ENI Outbound FIFO DMA Request

When DMA FIFO operations are enabled, the PA14 output is used for the active low outbound FIFO DMA ready output PDRQO*. The PDRQO* signal is only routed through PA14 when both DMAE* and DMAE2 in the ENI control register are both set to 0. When DMAE* is set low and DMAE2 is set high, an active high version of PDRQO is routed out the active high PINT2 pin instead. The PDRQO and PDRQI signals are internally “ORed” together before driving the PINT2 pin.

The PDRQO*/PDRQO signal is driven active to indicate the outbound FIFO has data available for reading. The PDRQO*/PDRQI signal is only driven active when the outbound FIFO is not empty and the EDRDBUFRDY bit is set active high in the ENI FIFO mode mask/status register. The FIFO mode mask/status register is available in the ENI address space.

PDRQI*/PDRQI/PA13 ENI Inbound FIFO DMA Request

When DMA FIFO operations are enabled, the PA13 output is used for the active low inbound FIFO DMA ready output PDRQI*. The PDRQI* signal is only routed through PA13 when both DMAE* and DMAE2 in the ENI control register are both set to 0. When DMAE* is set low and DMAE2 is set high, an active high version of PDRQI is routed out the active high PINT2 pin instead. The PDRQO and PDRQI signals are internally “ORed” together before driving the PINT2 pin.

The PDRQI*/PDRQI signal is driven active to indicate the inbound FIFO has room available for writing. The PDRQI*/PDRQI signal is only driven active when the inbound FIFO is not full and the EDWRBUFEMP bit is set active high in the ENI FIFO mode mask/status register. The FIFO mode mask/status register is available in the ENI address space.

PDATA[15:0] ENI Data Bus

The ENI interface supports a 16-bit data bus. The ENI interface can be configured to operate in either 16-bit data mode or 8-bit data mode. When operating in 16-bit mode, all data bits are used, however, when operating in 8-bit mode only PDATA[15:8] are used.

PACK* ENI Acknowledge

PACK* signal is driven by the ENI interface in response to an active low PCS* signal. The PACK* signal indicates the requested ENI bus cycle is complete.

The PACK* signal can be configured to operate as either an active low acknowledge (ACK) indicator or an active high ready (RDY) indicator. This control is provided by the EPACK* bit in the ENI control register. The EPACK* bit can be automatically initialized to its proper state upon reset by including or not including a pulldown resistor on the A1 signal (refer to Section *6.5.1 ENI Module Hardware Initialization*). Figure 13-22: ENI Shared RAM & Register Cycle Timing shows the difference between the ACK and RDY configuration for the PACK* signal.

The PACK* signal can be configured to operate as either a TTL or open-drain signal. The TTL configuration for the PACK* signal is provided by the WR_OC bit in the ENI control register. The WR_OC bit can be automatically initialized to its proper state upon reset by including or not including a pulldown resistor on the A6 signal (refer to Section *6.5.1 ENI Module Hardware Initialization*).

PINT1* ENI Interrupt 1

The PINT1* signal is driven active low to indicate an interrupt condition to the external ENI processor. An interrupt condition is established when the ARM processor sets the STSINT or VDAINT bits in the ENI shared register. An interrupt condition is also established based upon the FIFO status and the FIFO interrupt enable bits in the FIFO mode mask/status register.

The PINT1* signal is further qualified by the EHWINT and SINTP2 bits in the ENI shared register. The ENI shared register is available in the ENI address space. The EHWINT provides global interrupt enable/disable control. The SINTP2 bit controls whether the interrupt is driven out the PINT1* pin or the PINT2* pin. The PINT1* is only driven active low when an interrupt condition is pending, the EHWINT bit is set to 1, and the SINTP2 bit is set to 0.

PINT2* ENI Interrupt 2

The PINT2* signal is driven active low to indicate an interrupt condition to the external ENI processor. An interrupt condition is established when the ARM processor sets the STSINT or VDAINT bits in the ENI shared register. An interrupt condition is also established based upon the FIFO status and the FIFO interrupt enable bits in the FIFO mode mask/status register.

The PINT2* signal is further qualified by the EHWINT and SINTP2 bits in the ENI shared register. The ENI shared register is available in the ENI address space. The EHWINT provides global interrupt enable/disable control. The SINTP2 bit controls whether the interrupt is driven out the PINT1* pin or the PINT2* pin. The PINT2* is only driven active low when an interrupt condition is pending, the EHWINT bit is set to 1, and the SINTP2 bit is set to 1.

The PINT2* signal can also generate an interrupt from the external ENI processor to the ARM processor. The PINT2* pin becomes an input when the DINT2* bit is set to 0 in the ENI control register. The DINT2* bit can be automatically initialized to its proper state upon reset by including or not including a pulldown resistor on the A5 signal (refer to Section *6.5.1 ENI Module Hardware Initialization*). When DINT2* is configured as in the pulsed interrupt input, the low-to-high transition on the PINT2* input causes an interrupt condition to be established to the ARM processor. The low-to-high transition on the PINT2* input causes the INTIOF bit to be set in the shared register for the ARM processor.

The PINT2 signal can also provide an active high DMA request when the ENI interface is configured to operate in FIFO mode and the DMAE2 bit is set in the ENI control register. In this condition, the inbound and outbound DMA ready signals are logically “ORed” together and routed through the PINT2 pin as an active high DMA Request. When the DMAE2 bit is set, the PA14 and PA13 signals are no longer used as DMA request outputs and return to their shared RAM address function extending the amount of addressable shared RAM to 32K while operating in FIFO DMA mode.

PBRW* ENI Data Buffer Read/Write

The PBRW* signal is an output used to control the data direction pin for an external data bus transceiver. Some applications require a data bus buffer between the NET+ARM chip and the external ENI processor system. PBRW* high indicates a data transfer from the NET+ARM chip to the ENI processor; while PBRW* low indicates a data transfer from the ENI processor to the NET+ARM chip.

PEN* ENI Data Buffer Enable

The PEN* signal is an output used to control the output enable pin for an external data bus transceiver. Some applications require a data bus buffer between the NET+ARM

chip and the external ENI processor system. PEN* low indicates a data transfer between the NET+ARM chip and the external ENI processor is in progress.

1.5.6 General Purpose I/O

The NET+ARM chip provides 24 bits of general purpose I/O. Each pin can be separately configured for input, output, or special function. The configuration for the general purpose I/O signals is found in the GEN module configuration.

PORTA7 Input / Output / TXDA

PORTA7 can be configured for general purpose input or output. It can be configured as the special function TXDA output. The TXDA output provides the transmit data output function for serial channel A.

PORTA6 Input / Output / DREQ1* / DTRA*

PORTA6 can be configured for general purpose input or output. It can be configured as the special function DREQ1* input. The DREQ1* input provides the DMA request input for DMA channel 3 when DMA channel 3 is configured for external channel sourcing. PORTA6 can be configured as the special function DTRA. The DTRA output provides the data terminal ready control signal output for serial channel A.

PORTA5 Input / Output / RTSA*

PORTA5 can be configured for general purpose input or output. It can be configured for special function RTSA output. The RTSA output provides the request to send control signal output for serial channel A.

PORTA4 Input / Output / RxCA / OUT1A*

PORTA4 can be configured for general purpose input or output. It can be configured for special function RxCA input. The RxCA input provides the external receive clock input for serial channel A. It can be configured for special function OUT1A output. The OUT1A output provides the miscellaneous control 1 signal output for serial channel A.

PORTA3 Input / Output / RXDA

PORTA3 can be configured for general purpose input or output. It can be configured for special function RXDA input. The RXDA input provides the receive data input signal for serial channel A.

PORTA2 Input / Output / DSRA* / DACK1*

PORTA2 can be configured for general purpose input or output. It can be configured for special function DSRA input. The DSRA input provides the data set ready input for serial channel A. It can be configured for special function DACK1* output. The DACK1* output provides the DMA acknowledge indication for DMA channel 3

when DMA channel 3 is configured for external channel sourcing.

PORTA1 Input / Output / CTSA*

PORTA1 can be configured for general purpose input or output. It can be configured for special function CTSA input. The CTSA input provides the clear to send control signal input for serial channel A.

PORTA0 Input / Output / DCDA* / DONE1*

PORTA0 can be configured for general purpose input or output. It can be configured for special function DCDA input. The DCDA input provides the data carrier detect control signal input for serial channel A.

It can be configured for special function DONE1* input or output. The DONE1* signal provides the DMA done indication for DMA channel 3 when DMA channel 3 is configured for external channel sourcing.

PORTB7 Input / Output / TXDB

PORTB7 can be configured for general purpose input or output. It can be configured as the special function TXDB output. The TXDB output provides the transmit data output function for serial channel B.

PORTB6 Input / Output / DREQ2* / DTRB*

PORTB6 can be configured for general purpose input or output. It can be configured as the special function DREQ1* input. The DREQ1* input provides the DMA request input for DMA channel 4 when DMA channel 4 is configured for external channel sourcing. PORTB6 can be configured as the special function DTRB. The DTRB output provides the data terminal ready control signal output for serial channel B.

PORTB5 Input / Output / REJECT* / RTSB*

PORTB5 can be configured for general purpose input or output. It can be configured for special function REJECT* input. The REJECT* input is used by an external Ethernet address filtering block to force the internal NET+ARM chip Ethernet MAC to REJECT the current incoming data packet. It can be configured for special function RTSB output. The RTSB output provides the request to send control signal output for serial channel B.

PORTB4 Input / Output / RxCB / OUT1B*

PORTB4 can be configured for general purpose input or output. It can be configured for special function RxCB input. The RxCB input provides the external receive clock input for serial channel B. It can be configured for special function OUT1B output. The OUT1B output provides the miscellaneous control 1 signal output for serial channel B.

PORTB3 Input / Output / RXDB

PORTB3 can be configured for general purpose input or output. It can be configured for special function RXDB input. The RXDB input provides the receive data input signal for serial channel B.

PORTB2 Input / Output / DSRB* / DACK2*

PORTB2 can be configured for general purpose input or output. It can be configured for special function DSRB input. The DSRB input provides the data set ready input for serial channel B. It can be configured for special function DACK1* output. The DACK1* output provides the DMA acknowledge indication for DMA channel 4 when DMA channel 4 is configured for external channel sourcing.

PORTB1 Input / Output / CTSB* / RPSF*

PORTB1 can be configured for general purpose input or output. It can be configured for special function RPSF* output. The RPSF* output is used by external Ethernet address filtering logic to identify to receive packet start of frame boundary. The filtering logic can use this signal to determine when the destination address can be found in the NET+ARM/PHY MII interface.

PORTB1 can be configured for special function CTSB input. The CTSB input provides the clear to send control signal input for serial channel B.

PORTB0 Input / Output / DCDB* / DONE2*

PORTB0 can be configured for general purpose input or output. It can be configured for special function DCDB input. The DCDB input provides the data carrier detect control signal input for serial channel B. It can be configured for special function DONE1* input or output. The DONE1* signal provides the DMA done indication for DMA channel 4 when DMA channel 4 is configured for external channel sourcing.

PORTC7 Input / Output / TxCA / OUT2A*

PORTC7 can be configured for general purpose input or output. It can be configured for special function TxCA input. The TxCA input provides the external transmit clock input for serial channel A. It can be configured for special function OUT2A output. The OUT2A output provides the miscellaneous control 2 signal output for serial channel A.

PORTC6 Input / Output / RIA* / IRQ*

PORTC6 can be configured for general purpose input or output. It can be configured for special function RIA input. The RIA input provides the ring indicator control signal input for serial channel A.

PORTC6 can be configured for special function IRQ* output. The IRQ* output is driven low when a pending interrupt is active from the GEN module interrupt

controller. The IRQ* output only identifies pending normal interrupts. The IRQ* output does not go active for fast interrupts.

PORTC5 Input / Output / TxCB / OUT2B*

PORTC5 can be configured for general purpose input or output. It can be configured for special function TxCB input. The TxCB input provides the external transmit clock input for serial channel B.

It can be configured for special function OUT2B output. The OUT2B output provides the miscellaneous control 1 signal output for serial channel B.

PORTC4 Input / Output / RIB* / RESET*

PORTC4 can be configured for general purpose input or output. It can be configured for special function RIB input. The RIB input provides the ring indicator control signal input for serial channel B.

It can be configured for special function RESET* output. The RESET* output goes active low when the ARM processor executes a soft reset command or the RSTIO bit is set in the ENI shared register.

PORTC3 Input / Output / CI3 / AMUX

PORTC3 can be configured for general purpose input or output. It can be configured for special function C3 interrupt input. The C3 interrupt input can be configured to generate a processor interrupt on either the low-to-high or high-to-low transition on PORTC3.

PORTC3 can be configured to provide the DRAM address multiplexer function. This feature is controlled by the AMUX and AMUX2 bits in the MEM module configuration register.

PORTC2 Input / Output / CI2

PORTC2 can be configured for general purpose input or output. It can be configured for special function C2 interrupt input. The C2 interrupt input can be configured to generate a processor interrupt on either the low-to-high or high-to-low transition on PORTC2.

PORTC1 Input / Output / CI1

PORTC1 can be configured for general purpose input or output. It can be configured for special function C1 interrupt input. The C1 interrupt input can be configured to generate a processor interrupt on either the low-to-high or high-to-low transition on PORTC1.

PORTC0 Input / Output / CI0

PORTC0 can be configured for general purpose input or output. It can be configured

for special function C0 interrupt input. The C0 interrupt input can be configured to generate a processor interrupt on either the low-to-high or high-to-low transition on PORTC0.

1.5.7 Clock Generation

XTAL1 Crystal Oscillator Input

XTAL2 Crystal Oscillator Output

A standard parallel-resonant crystal can be attached to the XTAL1 and XTAL2 pins to provide the main input clock to the NET+ARM chip.

Quartz crystals can be chosen from a wide range of manufacturers and it is difficult to specify only one rule. The crystals' characteristics also vary significantly depending on their resonance frequency. The choice must always be parallel resonance for the NET+ARM chip with a load capacitance CL of 8pF. If the crystal-recommended load capacitance is greater, then add capacitors external to XTAL1 and XTAL2.

The crystal oscillator is a low-power oscillator; do not use no external capacitors or resistors to reduce the current driven in the crystal. If however, the crystal used has a drive level smaller than the drive level specified, then add a series resistor between XTAL2 and the crystal connection.

Instead of using the XTAL1 / XTAL2 crystal oscillator circuit, the NET+ARM chip can be driven using an external TTL oscillator. The TTL oscillator provides a single TTL clock input on the XTAL1 pin. The XTAL2 pin is left unconnected in this configuration. To use this configuration, the PLLTST* pin must be driven low. Also note that when this configuration is used, the internal PLL is bypassed and disabled.

PLLVDD Clean PLL Power

The PLLVDD power pin powers the internal PLL circuit. It is recommended this pin be provided with clean power.

PLLVSS Clean PLL Ground

The PLLVSS ground pin grounds the internal PLL circuit. It is recommended this pin be provided with clean ground.

PLLAGND Clean PLL Analog Ground

The PLLAGND ground pin grounds the internal PLL circuit. It is recommended this pin be provided with clean ground.

PLLPF PLL Loop Filter

The PLLPF pin provides the PLL loop filter circuit. An external RC circuit (100 ohms, 0.01uF) must be attached between PLLPF and ground.

RESET* System Reset

The RESET* input resets the NET+ARM chip hardware. A valid RESET* input must be issued for a minimum of 40ms after power reaches 3.0 volts. A RESET* input can be issued at any time to reset the NET+ARM chip.

1.5.8 Test Support

PLLTST* PLL Test and Disable

When driven active low, the PLLTST* input disables the internal PLL. The operating clock for the NET+ARM chip is based upon a TTL clock input attached to the XTAL1 pin. The PLLTST* pin must be left unconnected or driven high to enable the crystal oscillator and internal PLL.

BISTEN* BIST Enable

The BISTEN* pin is used for testing purposes only. It should always be left unconnected or driven high.

SCANEN* SCAN Enable

The SCANEN* pin is used for testing purposes only. It should always be left unconnected or driven high.

1.5.9 ARM Debugger

There are five pins that provide a dedicated connection to the internal ARM processor core. These five signals only connect to the ARM processor and are used for software development using the ARM Embedded ICE module (or similar device). These five signals should not be confused with 1149.1 JTAG Testing.

TDI

TDO

TMS

TRST*

TCK

Note: TRST* must be pulsed active low after power up.

1.5.10 Power

VCCAC Power A/C Switching

The VCCAC pins provide the 3.3V power for the switching component of the I/O output drivers. These pins can be filtered to lower any potential EMI.

VCCDC Power DC Drive

The VCCDC pins provide the 3.3V power for the DC component of the I/O output drivers. These pins should not require any filtering other than standard decoupling. practices.

VCCCO Power Core

The VCCCO pins provide the 3.3V power for the internal NET+ARM chip core. These pins should not require any filtering other than standard decoupling practices.

GNDAC Ground A/C Switching

The GNDAC pins provide the ground for the switching component of the I/O output drivers. These pins can be filtered to lower any potential EMI.

GNDDC Ground DC Drive

The GNDDC pins provide the ground for the DC component of the I/O output drivers. These pins should not require any filtering other than standard decoupling practices.

GNDCO Ground Core

The GNDCO pins provide the ground for the internal NET+ARM chip core. These pins should not require any filtering other than standard decoupling practices.

Chapter 2
BBUS Module

The BBUS module provides the data path between NET+ARM chip internal modules. The BBUS module provides the address and data multiplexing logic that supports the data flow through the NET+ARM chip.

The BBUS module provides the central arbiter for all the NET+ARM chip bus masters. The CPU, DMA, ENI, and BUS modules are all potential masters of the BBus.

Each bus master is given the opportunity to control the bus in a round-robin fashion. If a bus master does not require the resources of the bus when its turn comes around, it is skipped until the next round robin sequence. Exceptions to this rule are described in section *2.2 BBus Arbiter*.

The BBUS module is also responsible for providing decoding of the various slave modules. Each slave module is given a small portion of the system address MAP for configuration and status.

2.1 Address Decoding

Table 2-1 defines how the CPU address map is divided to allow access to the various internal modules and external resources. All internal resources are addressed using the upper memory addresses. Each internal module is given 1M byte of address space for its own internal decoding. The description for the respective module defines its specific register map.

Address Range	Module
0000 0000 - FF7F FFFF	BUS Module
FF80 0000 - FF8F FFFF	EFE Module
FF90 0000 - FF9F FFFF	DMA Module
FFA0 0000 - FFAF FFFF	ENI Module
FFB0 0000 - FFBF FFFF	GEN Module
FFC0 0000 - FFCF FFFF	MEM Module
FFD0 0000 - FFDF FFFF	SER Module
FFF0 0000 - FFFF FFFF	Cache RAM (NETA40 only)

Table 2-1: BBus Address Decoding

The BBUS module does not allow access to any internal registers unless the CPU_SUPV signal is active, which indicates the firmware is executing in supervisor mode. The system control register provides an override signal (the USER bit) to allow access to internal registers in user mode.

The BBUS module does not allow an external bus master access to any internal registers unless the BUSER bit is set in the system control register. When an external master is accessing internal register, the most significant four address bits (those bits that are not assigned an I/O pin) are removed from the address comparison logic.

2.2 BBus Arbiter

The BBus arbiter has been designed to not allow any one bus master to obtain consecutive back-to-back cycles while another bus master is waiting. The only two exceptions to this rule are burst and read-modify-write transactions. A bus master must be allowed to complete any burst transaction is already in progress. During read-modify-write cycles, the arbiter guarantees that the bus master retains consecutive cycles.

Chapter 3
CPU Module

The heart of the CPU module is provided by the ARM7TDMI from Advanced RISC Machines (ARM) Ltd. For details on the ARM7TDMI, refer to the ARM7TMDI Data Sheet.

The ARM7TDMI is a member of the Advanced RISC Machines (ARM) family of general purpose 32-bit microprocessors, which offer high performance for very low power consumption and size.

The ARM architecture is based on Reduced Instruction Set Computer (RISC) principles. The instruction set and related decode mechanism are much simpler than those of Complex Instruction Set Computers (CISC). This simplicity results in a high instruction throughput and impressive real-time interrupt response for a small cost-effective circuit.

Pipelining is employed such that all parts of the processing and memory systems can operate continuously. Typically, while one instruction is being executed, its successor is being decoded, and a third instruction is being fetched from memory.

3.1 Thumb Concept

The ARM7TDMI processor employs a unique architectural strategy known as Thumb, which makes it ideally suited to high-volume applications with memory restrictions or applications where code density is an issue.

The key idea behind the Thumb is a super-reduced instruction set. Essentially, the ARM7TDMI processor has two instruction sets:

- Standard 32-bit ARM Set
- 16-bit Thumb Set

The Thumb's 16-bit instruction length allows it to approach twice the density of standard ARM code while retaining most of the ARM's performance advantage over a traditional 16-bit processor using 16-bit registers. This is possible because Thumb code operates on the same 32-bit register set as the ARM code. Thumb code consumes only 65% of the same code compiled in ARM mode.

Thumb instructions operate with the standard ARM register configuration allowing excellent interoperability between ARM and Thumb states. Each 16-bit Thumb instruction has a corresponding 32-bit ARM instruction with the same effect on the processor model.

The Thumb architecture provides a Thumb instruction decoder in front of the standard 32-bit ARM processor. The Thumb instruction decoder basically remaps each 16-bit Thumb instruction into a 32-bit standard ARM instruction. The Thumb instruction set typically requires 30% more instructions to perform the same task as 32-bit instructions, however, the Thumb instruction set can fit twice as many instructions in the same code space. The next result is a 35% decrease in overall code density.

3.2 Performance

The ARM7TDMI core does not contain a cache. It runs as fast as instructions can be fetched. The performance rating for the ARM RISC depends on the system bus speed and cycle time. The performance is also affected by the size of the system bus and the type of code being executed (ARM vs. Thumb).

The ARM instruction set yields a 0.9 Dhrystone (2.1) rating per MHz of instruction execution, whereas, the Thumb instruction set yields 0.75 Dhrystones per MHz. The MHz rating reflects the rate at which instructions can be fetched from external Flash memory.

System Bus Size	Code Style	RISC Speed	System Bus Speed	Wait States	Instruction Cycle Time	Dhrystone Rating
Thumb Mode						
16-bit	Thumb	25 MHz	25 MHz	1	120 ns	6.25 MIPS
16-bit	Thumb	25 MHz	25 MHz	0	80 ns	9.30 MIPS
16-bit	Thumb	33 MHz	33 MHz	2	120 ns	6.25 MIPS
16-bit	Thumb	33 MHz	33 MHz	1	90 ns	8.33 MIPS
16-bit	Thumb	33 MHz	33 MHz	0	60 ns	12.50 MIPS
16-bit	Thumb	33 Mhz	33 Mhz	Cache	30 ns	25 MIPS

Table 3-1: ARM Performance

System Bus Size	Code Style	RISC Speed	System Bus Speed	Wait States	Instruction Cycle Time	Dhrystone Rating
ARM Mode						
32-bit	ARM	25 MHz	25 MHz	1	120 ns	7.50 MIPS
32-bit	ARM	25 MHz	25 MHz	0	80 ns	11.25 MIPS
32-bit	ARM	33 MHz	33 MHz	0	60 ns	15.00 MIPS
32-bit	ARM	33 Mhz	33 Mhz	Cache	30 ns	30 MIPS

Table 3-1: ARM Performance (Continued)

3.3 ARM Exceptions

Exceptions arise whenever the normal flow of a program has to be halted temporarily, for example, to service an interrupt from a peripheral. The ARM processor can be interrupted by seven basic exceptions. Each exception causes the ARM processor to save some state information, then jump to a location in low memory. Low memory is referred to as the Vector Table. The Vector Table always exists starting at address 0.

Before an exception can be handled, the current processor state must be preserved so that the original program can resume when the handler routine has finished.

It is possible for several exceptions to arise at the same time. If this happens, they are dealt with in a fixed order, refer to Section *3.3.12 Exception Priorities*.

3.3.1 Exception Vector Table

All exceptions result in the ARM processor vectoring to an address in low memory using the Exception Vector Table. The Exception Vector Table always starts at base address 0.

Vector Address	Vector	Description
0x0	RESET	Reset Vector; for initialization and startup
0x4	Undefined	Undefined Instruction Encountered
0x8	SWI	Software Interrupt; used for entry point into the kernel
0xC	Abort (Prefetch)	Bus Error (no response or error) fetching instructions
0x10	Abort (Data)	Bus Error (no response or error) fetching data
0x14	Reserved	Reserved
0x18	IRQ	Interrupt from NET+ARM Interrupt Controller
0x1C	FIRQ	Fast Interrupt from NET+ARM Interrupt Controller

Table 3-2: Exception Vector Table

All internal NET+ARM peripheral interrupts are presented to the CPU using the IRQ or FIRQ interrupt inputs. The ARM has the ability to mask various NET+ARM peripheral interrupts at the global level using the NET+ARM Interrupt Controller. Furthermore, the ARM has the ability to mask interrupts at the micro-level using configuration features with the peripheral modules.

All IRQ interrupts are disabled when the “I” bit is set in the ARM CPSR register. When the “I” bit is cleared, those interrupts enabled in the NET+ARM Interrupt Controller are allowed to assert the IRQ input to the ARM processor.

Upon entering an interrupt, the “I” bit is automatically set by the ARM processor. This disables recursive interrupts. The first task of the Interrupt Service Routine (ISR) is to read the interrupt status register. This register identifies all active sources for the IRQ interrupt. Firmware can prioritize which interrupt should be serviced by using the bits defined in the interrupt status register.

3.3.2 Action on Entering an Exception

When handling an exception, the ARM7TDMI:

1. Preserves the address of the next instruction in the appropriate Link Register. If the exception has been entered from ARM state, then the address of the next instruction is copied into the Link Register (that is, current PC + 4 or PC + 8 depending upon the exception. Please refer to Table 3-3: Exception Entry / Exit). If the exception has been entered from THUMB state, then the value written into the Link Register is the current PC offset by a value such that the program resumes from the correct place on return from the exception. This means that the exception handler need not determine which state the exception was entered from. For example, in the case of SWI, `MOVS PC, R14_svc` will always return to the next instruction regardless of whether the SWI was executed in ARM or THUMB state.
2. Copies the CPSR into the appropriate SPSR.
3. Forces the CPSR mode bits to a value that depends on the exception.
4. Forces the PC to fetch the next instruction from the relevant exception vector.
5. For IRQ or FIRQ interrupts, the “I” and “F” bits respectively are set to disable interrupts to prevent unmanageable nesting of exceptions.
6. If the processor is in THUMB state when an exception occurs, it will automatically switch into ARM state when the PC is loaded with the exception vector address.

3.3.3 Action on Leaving an Exception

On completion, the exception handler:

1. Moves the Link Register, minus the offset where appropriate, to the PC. (The offset will vary depending on the type of exception.)
2. Copies the SPSR back to the CPSR.
3. Clears the Interrupt disable flags, if they were set on entry.

Note: An explicit switch back to THUMB state is never needed, since restoring the CPSR from the SPSR automatically set the T bit to the value it held immediately prior to the exception.

3.3.4 Exception Entry / Exit Summary

Summarizes the PC value preserved in the relevant R14 on exception entry, and the recommended instruction for exiting the exception handler.

Return/ Exception	Return Instruction	Previous State ARM R14_x	Previous State THUMB R14_x	Notes
BL	MOV PC, R14	PC + 4	PC + 2	1
RESET	NA	—	—	4
UNDEF	MOVS PC, R14_und	PC + 4	PC + 2	1
SWI	MOVS PC, R14_svc	PC + 4	PC + 2	1
ABORT P	SUBS PC, R14_abt, #4	PC + 4	PC + 4	1
ABORT D	SUBS PC, R14_abt, #8	PC + 8	PC + 8	3
IRQ	SUBS PC, R14_irq, #4	PC + 4	PC + 4	2
FIRQ	SUBS PC, R14_firq, #4	PC + 4	PC + 4	2

Table 3-3: Exception Entry / Exit

Notes:

1. Where PC is the address of the BL/SWI/Undefined Instruction fetch that had the prefetch abort.
2. Where PC is the address of the instruction that did not get executed since the FIRQ or IRQ took priority.
3. Where PC is the address of the Load or Store instruction that generated the data abort.
4. The value saved in R14_svc upon reset is unpredictable.

3.3.5 Reset Exception

When the ARM7TDMI is held in reset, the ARM7TDMI abandons the executing instruction and then continues to fetch instructions from incrementing word addresses.

When the ARM7TDMI is removed from reset, the ARM7TDMI:

1. Overwrites R14_svc and SPSR_svc by copying the current values of the PC and CPSR into them. The value of the saved PC and SPSR is not defined.
2. Forces the CPSR M field to 10011 (Supervisor mode), sets the I and F bits in the CPSR, and clears the CPSR T bit (back to ARM mode).

3. Forces the PC to fetch the next instruction from address 0x00.
4. Execution resumes in ARM state.

3.3.6 Undefined Exception

When the ARM7TDMI comes across an instruction that it cannot handle, it takes the undefined instruction trap. This mechanism may be used to extend either the THUMB or ARM instruction set by software emulation.

After emulating the failed instruction, the trap handler should execute the following instruction irrespective of the state (ARM or THUMB):

```
MOVS PC, R14_und
```

This restores the PC and CPSR, and returns to the instruction following the undefined instruction.

3.3.7 SWI Exception

The software interrupt instruction (SWI) is used for entering Supervisor mode, usually to request a particular supervisor function. A SWI handler should return by executing the following instruction irrespective of the state (ARM or THUMB):

```
MOVS PC, R14_svc
```

This restores the PC and CPSR, and returns to the instruction following the SWI.

3.3.8 Abort Exception

An abort indicates that the current memory access cannot be completed. It can be signaled by the external ABORT input. The ARM7TDMI checks for the abort exception during memory access cycles.

There are two types of abort:

Prefetch Abort	occurs during an instruction prefetch
Data Abort	occurs during a data operand access

If a prefetch abort occurs, the prefetched instruction is marked as invalid, but the exception will not be taken until the instruction reaches the head of the pipeline. If the

instruction is not executed – for example because a branch occurs while it is in the pipeline – the abort does not take place.

If a data abort occurs, the action taken depends on the instruction type:

1. Single data transfer instructions (LDR, STR) write back modified base registers; the Abort handler must be aware of this.
2. The swap instruction (SWP) is aborted as though it had not been executed.
3. Block data transfer instructions (LDW, STM) complete. If write-back is set, the base is updated. If the instruction would have overwritten the base with data (i.e. it has the base in the transfer list), the overwriting is prevented. All register overwriting is prevented after an abort is indicated, which means in particular that R15 (always the last register to be transferred) is preserved in an aborted LDM instruction.

The abort mechanism allows the implementation of a demand paged virtual memory system. In such a system the processor is allowed to generate arbitrary addresses. When the data at an address is unavailable, the Memory Management Unit (MMU) signals an abort. The abort handler must then work out the cause of the abort, make the requested data available, and retry the aborted instruction. The application program needs no knowledge of the amount of memory available to it, nor is its state in any way affected by the abort.

After fixing the reason for the abort, the handler should execute the following instructions irrespective of the state (ARM or THUMB):

```
SUBSPC, R14_abt, #4for a prefetch abort, or
SUBSPC, R14_abt, #8for a data abort
```

3.3.10 IRQ Exception

The IRQ (Interrupt Request) exception is a normal interrupt sourced by the NET+ARM Interrupt Controller. IRQ has a lower priority than FIRQ and is masked out when a FIRQ sequence is entered. It may be disabled at any time by setting the I bit in the CPSR to 1, though this can only be done from a privileged (non-User) mode.

Irrespective of whether the exception was entered from ARM or THUMB state, a FIRQ handler should leave the interrupt by executing

```
SUBSPC, R14_irq, #4
```

3.3.11 FIRQ Exception

The FIRQ (Fast Interrupt Request) exception is designed to support a data transfer or channel process, and in ARM state has sufficient registers to remove the need for register saving (thus minimizing the overhead of context switching).

The only two internal peripherals that can generate an FIRQ interrupt are the GEN Module Timers and the GEN Module watchdog timer.

Irrespective of whether the exception was entered from ARM or THUMB state, a FIRQ handler should leave the interrupt by executing:

```
SUBSPC, R14_firq, #4
```

Setting the CPSR F flag to 1 can disable the FIRQ interrupt. (However, note that this is not possible from User mode). If the F flag is clear, the ARM7TDMI checks for a LOW level on the output of the FIRQ synchronizer at the end of each instruction.

3.3.12 Exception Priorities

When multiple exceptions arise at the same time, a fixed priority system determines the order in which they are handled:

Highest Priority

1. Reset
2. Data Abort
3. FIRQ
4. IRQ
5. Prefetch Abort
6. Undefined Instruction, SWI

Lowest Priority

Not all exceptions can occur at once:

Undefined Instruction and Software Interrupt are mutually exclusive, since they each correspond to particular (non-overlapping) decoding of the current instruction.

If a data abort occurs at the same time as a FIRQ, and FIRQ is enabled (i.e. the CPSR F flag is clear), the ARM7TDMI enters the data abort handler and then immediately process to the FIRQ vector. A normal return from FIRQ causes the data abort handler to resume execution. Placing data abort at a higher priority than FIRQ is necessary to

ensure that the transfer error does not escape detection. The time for this exception entry should be added to worst-case FIRQ latency calculations.

3.3.13 Overview of the Hardware Interrupts

There are two wires that go into the ARM7 CPU core used for interrupting the processor. These lines are:

- IRQ (normal interrupt)
- FIRQ (fast interrupt)

They are basically the same except for the fact that FIRQ can interrupt IRQ. The purpose of the FIRQ line is to add a simple two-tier priority scheme to the interrupt system. Most all sources of interrupts on the NET+ARM come from the IRQ line. The only potential sources for FIRQ interrupts on the NET+ARM are the two built in timers and the watchdog timer. The timers are controlled using the Timer Control registers in the GEN module (0xFFB0 0010 / 18). The watchdog timer is controlled using the System Control Register in the GEN module (0xFFB0 0000).

Interrupts may come from many different sources on the NET+ARM and are managed by the interrupt controller within the GEN module (see Figure 3-1). Interrupts can be enabled/disabled on a per-source basis using the Interrupt Enable Register (0xFFB0 0030). This register serves as a mask for the various interrupt sources and ultimately controls whether or not an interrupt from a NET+ARM module can reach the IRQ line.

There are two read-only registers in the interrupt controller:

- The first is the Interrupt Status Register Raw that indicates the source of a NET+ARM interrupt regardless of the Interrupt Enable Register's state. All interrupts that are active in their respective module will be visible in the Interrupt Status Register Raw (0xFFB0 0038).
- The second read-only register is the Interrupt Status Register Enabled (0xFFB0 0034). This register identifies the current state of all interrupt sources that are enabled and is defined by performing a logical *AND* of the Interrupt Status Register Raw and the Interrupt Enable Register. All of the bits in the Interrupt Status Register Enabled are then *OR-ed* together; the output of which is fed directly to the IRQ line that then interrupts the ARM.

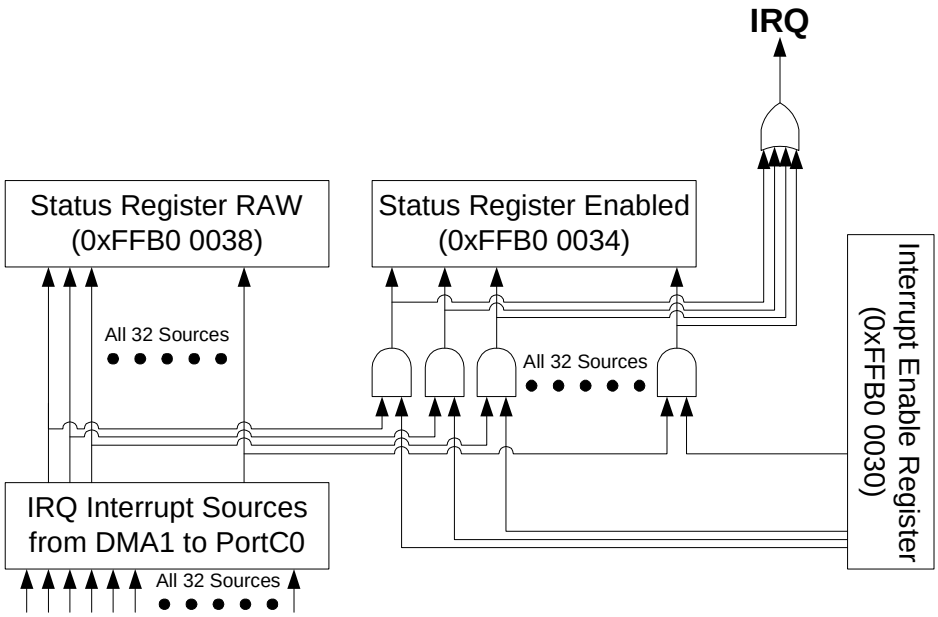


Figure 3-1: Interrupt Controller Diagram

NET+ARM Interrupt Enable Register (0xFFB0 0030)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA1	DMA2	DMA3	DMA4	DMA5	DMA6	DMA7	DMA	DMA9	DMA10	ENI PORT 1	ENI PORT 2	ENI PORT 3	ENI PORT 4	ENET RX	ENET TX
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SER 1 RX	SER 1 TX	SER2 RX	SER 2 TX	—	—	—	—	—	Watch Dog	Timer 1	Timer 2	PortC PC3	PortC PC2	PortC PC1	PortC PC0
RESET:															
0	0	0	0						0	0	0	0	0	0	0
R/W	R/W	R/W	R/W						R/W	R/W	R/W	R/W	R/W	R/W	R/W

Interrupt sources include DMA channels 1-10, ENI, Ethernet receive and transmit, serial port receive and transmit, watchdog timer, timers 1 and 2, and 4 pins on Port C. Each of these sources is enabled/disabled within their respective module (and sub-modules) within the NET+ARM ASIC, however, the interrupt controller in the GEN module does not latch any of the interrupt signals

Note: Causes of interrupts are latched in their respective sub-module until cleared.

The specific interrupts that each source is capable of generating are outlined below:

3.3.14 DMA Interrupts

All of the DMA channels, including the four sub-channels of the Ethernet receiver, have four possible sources of interrupts. These are found in the DMA status/interrupt enable register (starting 0xFF90 0014) in Section *4.4.3 DMA Status/Interrupt Enable Register*. Under normal operations, only the NCIP and NRIP interrupts are used; the other interrupts would indicate a catastrophic failure. The interrupts are acknowledged by writing a “1” to the register location. This resets the interrupt condition.

0xFF90 0014 / 34 / 54 / 74 / 94 / B4 / D4 / F4 / 114 / 134 / 154 / 174 / 194 DMA Channel Status Registers			
D31	R/C	NCIP	Normal Completion Interrupt Pending
D30	R/C	ECIP	Error Completion Interrupt Pending
D29	R/C	NRIP	Buffer Not Ready Interrupt Pending
D28	R/C	CAIP	Channel Abort Interrupt Pending

3.3.15 ENI Interrupts

The ENI can be operated in one of two modes, IEEE 1284, or ENI host mode. Interrupts are handled differently for each mode and are explained below.

3.3.15.1 IEEE 1284 Mode

There are three possible sources of interrupts when the ENI is operated in IEEE 1284 mode. These interrupts are in the IEEE 1284 Port Control register (starting at 0xFFA0 0010) in Section *6.5.5 IEEE 1284 Port Control Registers*. It should be noted that the Active Edge of ACK Interrupt (ACKI) is active when bit is set low.

0xFFA0 0010 / 14 / 18 / 1C IEEE 1284 Port Control Registers			
D06	R	FIFOE	FIFO Empty Interrupt
D05	R	OBE	Output Buffer Empty Interrupt
D04	R/C	ACKI	Active Edge of ACK Interrupt Pending

3.3.15.2 ENI Host Mode

The ENI host mode allows an external processor to access the NET+ARM as a slave memory device. The NET+ARM ENI is connected to the system bus of the external processor. The external processor may generate four different interrupts to the NET+ARM. The ENI interrupts to the NET+ARM share the same interrupt mask location bits as the IEEE 1284 port interrupts. This is okay, because the ENI can only

be configured in 1284 mode or ENI host mode. Table 3-4 describes the shared positions in the Interrupt Enable Register.

GEN interrupt	Bit position	IEEE 1284 Mode	ENI/FIFO mode
ENI port 1	D21	Port 1	FIFO Receiver
ENI port 2	D20	Port 2	FIFO Transmitter
ENI port 3	D19	Port 3	ENI Interrupt
ENI port 4	D18	Port 4	ENI Bus Error

Table 3-4: Shared Positions in the Interrupt Enable Register

When the NET+ARM is configured in the ENI host mode, it can be interrupted by the external processor through the ENI shared register. This is the ENI interrupt, which shares the 1284 port 3 interrupt bit. If the ENI interface uses the FIFOs, they will interrupt the NET+ARM using the same bit locations as 1284 port 1 and port 2. The ENI Bus Error interrupt occurs when the external processor attempts an access to a non-existent memory location on the NET+ARM's system bus. The NET+ARM also has the capability to interrupt the external processor using the PINT1 and PINT2 signals in the ENI. These signals are covered in much greater detail in the ENI host port application note available from NETsilicon.

3.3.16 Ethernet Receive/Transmit Interrupts

There are three interrupts for Ethernet receive and four interrupts for Ethernet transmit found in the Ethernet General Status register (0xFF80 0004), Section *5.3.2 Ethernet General Status Register*. These interrupts are *only* used when the Ethernet receiver/transmitter are in interrupt mode instead of DMA mode. In general, the DMA mode will be more desirable and these bits can be ignored. The NETsilicon Ethernet driver included with the BSP uses DMA mode on all parts except the NET+5&10, which uses interrupt mode for transmit only.

0xFF80 0004 Ethernet General Status Register			
D27	R	RXREGR	Receive Register Ready Interrupt
D26	R	RXFIFOH	Receive FIFO Half Empty Interrupt
D25	R/C	RXBR	Receive Buffer Ready Interrupt Pending
D19	R	TXREGE	Transmit Register Empty Interrupt
D18	R	TXFIFOH	Transmit FIFO Half Empty Interrupt
D17	R/C	TXBC	Transmit Buffer Complete Interrupt
D16	R	TXFIFOE	Transmit FIFO Empty Interrupt

3.3.17 Serial Interrupts

The serial channel status register has many sources for interrupts. They are listed here, but much more information is available in Section *7.5.3 Serial Channel Status Register A.*

0xFFD0 0008 / 48 Serial Channel Status Register A			
D15	R/C	RBRK	Receive Break Interrupt Pending
D14	R/C	RFE	Receive Framing Error Interrupt Pending
D13	R/C	RPE	Receive Parity Error Interrupt Pending
D12	R/C	ROVER	Receive Overrun Interrupt Pending
D11	R	RRDY	Receive Register Ready Interrupt
D10	R	RHALF	Receive FIFO Half Empty Interrupt
D09	R/C	RBC	Receive Buffer Closed Interrupt Pending
D08	R	RFULL	Receiver FIFO Full Interrupt
D07	R/C	RCDI	Change in DCD Interrupt Pending

D06	R/C	RII	Change in RI Interrupt Pending
D05	R/C	DSRI	Change in DSR Interrupt Pending
D04	R/C	CTSI	Change in CTS Interrupt Pending
D03	R	TRDY	Transmit Register Empty Interrupt
D02	R	THALF	Transmit FIFO Half Empty Interrupt
D01	R/C	TBC	Transmit Buffer Closed Interrupt Pending
D00	R	TEMPTY	Transmit FIFO Empty Interrupt

3.3.18 Watchdog Timer Interrupts

Upon expiration of the watchdog timer the system can generate either an IRQ interrupt, an FIRQ interrupt or a system reset. The type of interrupt produced and the length of the watchdog timer is configured via the System Control register (0xFFB0 0000), Section *8.2.1 System Control Register*. The watchdog is strobed using the software service register (0xFFB0 000C) in the GEN module.

0xFFB0 0000 System Control Register			
D23:D22	R/W	SWRI	Software Watch Dog Reset Interrupt Select 00 – IRQ Interrupt 01 – FIRQ Interrupt 10 – Reset 11 – Reserved

3.3.19 Timers 1 and 2 Interrupts

There are two different types of interrupts that can be generated by Timers 1 and 2. The type of interrupt is configured in the Timer Control register (0xFFB0 0010 / 18) in Section *8.2.5 Timer Control Register* while the interrupt itself is contained within the Timer Status register (0xFFB0 0014 / 1C), Section *8.2.6 Timer Status Register*.

0xFFB0 0010 / 18 Timer Control Register			
D29	R/W	TIRQ	Timer Interrupt Generation 0 – IRQ Interrupt 1 – FIRQ Interrupt

0xFFB0 0014 / 1C Timer Status Register			
D30	R/C	TIP	Timer Interrupt Pending

3.3.20 Port C Interrupts

The lower four pins of Port C on the NET+ARM may be used as interrupt sources (C3, C2, C1, C0). There is only one register for enabling, configuring and servicing the interrupts. This is done using the Port C register (0xFFB0 0028). To enable the interrupts, set the corresponding mode bit for special mode (1). The interrupts are *edge-sensitive*, and the transition is selectable using the DIR bit in the same register.

To generate an interrupt on a high to low transition, set the DIR bit to zero, and for a low to high transition, set the DIR bit to 1. So, for example to, enable C1 and C2 as interrupt inputs sensitive to low to high transitions, one would write 0xFFB00028 = 0x06060000. When an interrupt occurs, a “1” is stored in the corresponding data portion of the same register. Clear the interrupt by writing a “1” to the same data bit.

Writing a “0” to the least significant 4 bits in this register has no effect.

3.4 NET+40 With Cache

3.4.1 Block Diagram

Figure 3-2 shows a block diagram of the CPU module structure within the NET+-40 chip. The NET+40 CPU module uses the ARM7TDMI stand-alone core and adds an instruction/data cache, write buffer, and pre-fetch control. The CPU module contains its own internal bus referred to as the CBus.

The CBus and BBus system buses are isolated via the bus interface unit (BIU). This structure allows code execution to continue from the cache while the BBus is being

used for DMA data transfer operations.

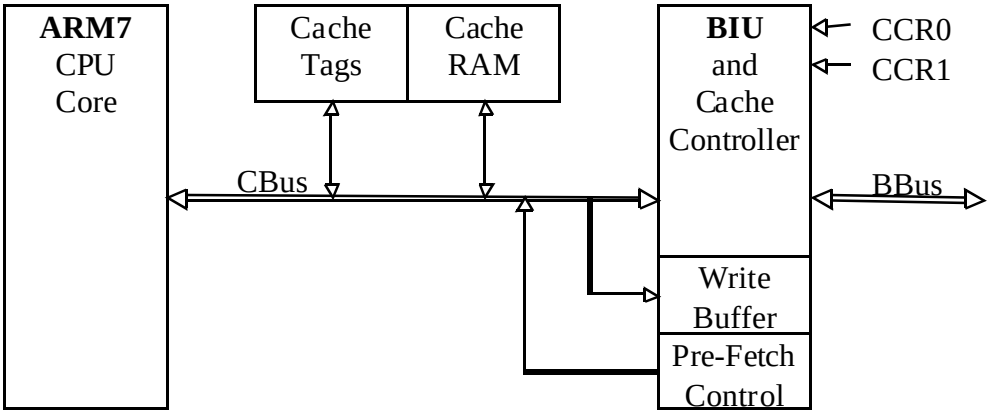


Figure 3-2: NET+40 CPU Core Block Diagram

3.4.2 BIU and Cache Controller

The bus interface unit (BIU) block provides the control functions necessary to decouple the CBus and BBus. The bus decoupling allows the CPU to execute instructions from the cache while the BBus is being used for DMA bus transactions.

The BIU uses the configuration information provided by the cache control registers (CCR) to determine when an external bus cycle is required instead of an internal cache cycle.

The BIU uses the write buffer to allow the ARM processor to continue executing instructions from the cache while write data is being written to external memory. The BIU allows the ARM to execute a single cycle write cycle to the write buffer. Note that write cycles to external memory can take multiple cycles to complete. The BIU can empty the write buffer to external memory while the processor is executing additional instructions from the cache.

The BIU uses the pre-fetch control when the ARM7TDMI is executing in Thumb mode and the external instruction storage peripheral is configured in 32-bit mode. The pre-fetch feature can squeeze some extra performance in this configuration.

3.4.3 Cache

The NET+40 chip contains a 4K-byte mixed instruction and data cache. The cache is arranged in a 4-way set associative configuration. Each cache-set can be configured to cache instructions, data, or both. When a cache-set is not enabled for caching, it can be

used as a simple block of RAM. Each cache-set can independently be configured as a 1K-byte block of cache or a 2K-byte block of RAM.

The NET+40 chip supports two cache control registers. Each control register identifies a configurable block of physical address space. The cache control register identifies all the cache and buffer features for the identified memory block. Typically, one cache control register identifies cacheable instructions while the other cache control register identifies cacheable data.

The cache tags are configured to identify individual 8-bit entries. This configuration maximizes the efficiency of a low-cost 16-bit bus running ARM Thumb code by minimizing the number of unused memory accesses. Traditional cache architectures require an entire cache-line (16 bytes per line) to be filled at a time. The traditional architecture approach can result in many unnecessary memory cycles when the program flow alters direction.

Each line in the cache RAM contains a 22-bit tag, 2 control bits, an 8-bit status field, and a 32-bit data field. Figure 3-3 provides a block diagram of the cache structure.

Refer to Section *3.4.14 Cache RAM (NET+40 only)* and Section *3.4.15 Cache TAG Format (NET+ 40 only)* for detailed information.

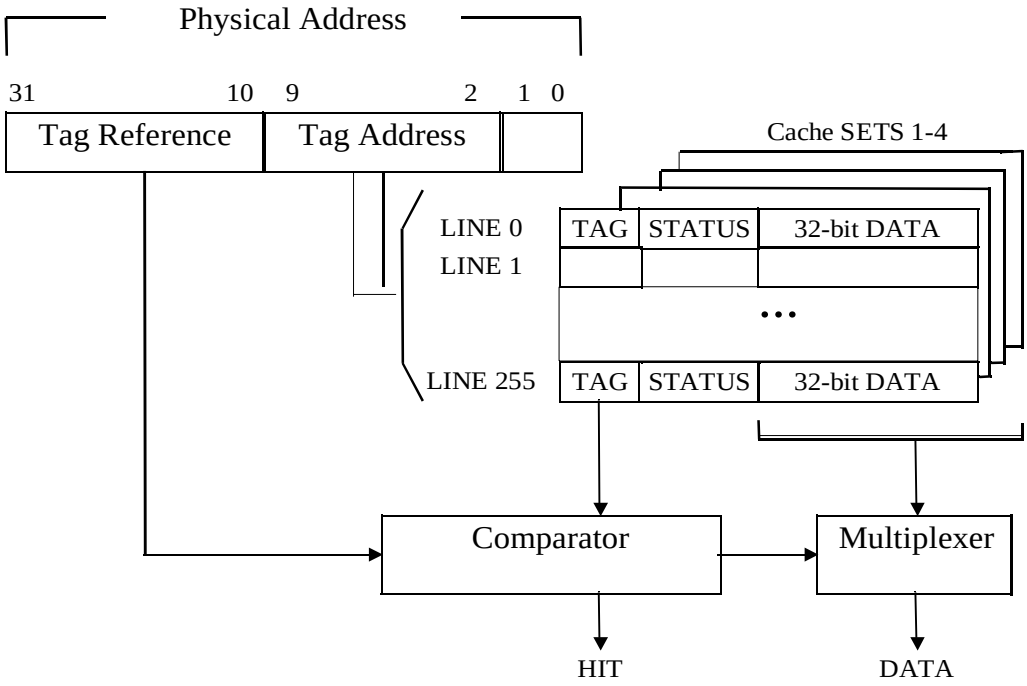


Figure 3-3: 4-Way Set Associated Cache

3.4.4 Cache Control Registers

The NET+40 chip supports two cache control registers (CCR0, CCR1). The CCR determines which physical addresses are cacheable. The CCR selects cacheable address blocks using an address mask and compare operation. Other CCR entries select the cache mode (write-through vs. copy-back), enable 32-bit prefetching, determine which cache sets can be used, and provide a write protection feature.

Refer to Section *3.4.13 Cache Control Registers (NET+40 only)* for detailed information.

3.4.5 Cache Operation

A read from a cacheable location results in data being taken from the cache if a cache-hit is found. Otherwise, data is read from memory and the cache is updated

accordingly. When a cache-miss occurs, the cache controller reads external memory and attempts to put the new line into an invalid (empty) location within the four possible cache-sets. When a cache-set needs to be flushed to make room for the new entry, a simple round-robin algorithm identifies which cache-set to flush. Cache flush operations (from a dirty location) are always written to the write buffer to update external memory.

The cache controller can distinguish between instructions and data fetches. All external instruction fetches can be loaded into the cache regardless of the state of the write buffer. However, any new external data fetch cannot occur until the write buffer is emptied. The empty write buffer requirement is imposed to ensure consistency between external RAM and whatever might be sitting in the write buffer.

The write-through cache mode implies any write accesses to areas defined as cacheable are written to both the cache (when data exists in the cache) and external memory. The cache line dirty bit is never set in write-through mode. The copy-back cache mode implies a cache location is only to be written to external memory when the cache line is flushed. In either case, the initial cache entry results from an external memory read operation and never from an internal processor write operation.

3.4.6 Cache Line Fill

A cache line can be filled using 8, 16, or 32-bit operations depending upon the ARM bus transactions. The NET+40 chip cache controller does not force the entire line to be filled at the same time. When a cacheable location is read that is not currently in the cache, the cache controller examines the four cache-sets attached to the associated address block. The cache controller inserts the new entry into any invalid set. If no empty sets are available, the cache controller flushes one of the current lines to make room for the new data. The algorithm for determining which set is flushed is a simple round-robin one.

When the ARM is fetching 16-bit Thumb instructions from an external 32-bit memory device, the cache line can be filled 32-bits at a time. This process is referred to as 32-bit pre-fetching. This feature is enabled using the FRC32 control bit in the cache control register. When FRC32 is set, the cache controller forces a 32-bit operand fetch whenever the ARM fetches any operand on a zero byte boundary. This feature can squeeze out some extra performance in a system executing Thumb code from a 32-bit instruction storage peripheral.

3.4.7 Cache Line Flush

A cache line is flushed when a new line is being fetched from external memory and an empty cache set is unavailable. Each time a cache line is flushed, the dirty bit is

checked. If the dirty bit is marked, the cache line is written to the write buffer. When a cache line is flushed, both the dirty bit and valid bits are cleared making the line empty and available. Any cache lines marked with the INVALID or LOCK bits set is not flushed by a cache line replacement operation.

3.4.8 Cache Coherency

Cache coherency must be carefully managed in any firmware design. If an external bus master and the ARM processor must share data, shared data cannot be stored in any area configured as cache. Any shared data must reside in a non-cacheable address block.

3.4.9 Cache Line Locking

Instructions and data can be locked within the cache by writing to the cache TAG and cache data registers. When the cache TAG is written, the LOCK bit must be set to 1. Once cache lines are locked, the lines remain in the cache until the LOCK bit is removed.

3.4.10 Write Buffer

The write buffer allows the CPU and the cache controller to execute write instructions without stalling to wait for the bus controller to complete the write cycles. The write buffer stores both the address and data values. The BIU attempts to empty the write buffer whenever possible. The current NET+40 chip cache architecture only supports a single long word in the write buffer. Future versions of the cache controller may provide a deeper write buffer.

To maintain coherency between external RAM and the write buffer, the BIU forces the write buffer to be emptied before the ARM processor can read a data operand from external memory. This may cause the ARM processor to temporarily stall while the write buffer is flushed. Since cached instructions are assumed to be static, an instruction read from external memory does not force the write buffer to first be flushed.

Writing a 0 to the WB ENABLE bit in the general control register stops additional entries into the FIFO. Existing entries are written to external memory in the normal way.

3.4.11 32-bit Pre-Fetch Feature

The 32-bit pre-fetch feature offers a performance improvement for those 16-bit Thumb applications using a 32-bit bus for instruction storage. The pre-fetch feature

causes the BUS module to fetch a 32-bit word each time the ARM processor fetches any cacheable operand on a zero byte boundary where the FRC32 bit is set in the associated cache control register. When the ARM processor fetches the next consecutive 16-bit instruction, the next instruction is most likely already available in the cache. This enhancement offers a performance improvement of at least 25%.

3.4.12 Cache Initialization

Before the cache can be enabled for operation, the cache TAG area must be initialized to zero. A simple software loop can accomplish this by writing zeros between long word addresses 0xFFF00000 and 0xFFF01FFC.

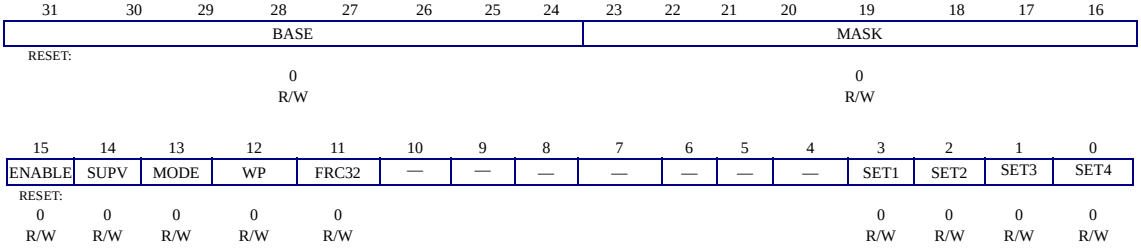
While writing zeros to the cache RAM block, the CINIT bit must also be set in the system control register. After the cache RAM block has been initialized to zero, the CINIT bit must be returned to zero.

3.4.13 Cache Control Registers (NET+40 only)

The CCR registers provide control for the instruction and data cache. Each register selects a specific block of physical address space and determines how the cache is used.

CCR0 / CCR1

Address = FFB0 0040 / 44



BASE **Physical Address Base**

The BASE field defines the base address of a physical block of memory that is to be cached using the CCR configuration parameters. The MASK field defines the size of the physical block of memory. This 8-bit field is processed with a logical AND of the MASK field and then compared with the upper eight bits of the CPU physical address (A31-A24) to determine which locations are to be cached with the CCR configuration.

The upper eight CPU physical address bits are also processed with a logical AND of the MASK field before the comparison with the BASE field is executed.

MASK Physical Address Mask

The MASK field is used to define the size of the cacheable region. The MASK field identifies those bits in the BASE field definition used in the address comparison operation with the CPU physical address. Only the bit positions with a 1 in the MASK field are to be used in the address comparison function.

ENABLE Enable

0 – Disable CCR configuration

1 – Enable CCR configuration

The ENABLE bit must be set to 1 to allow the address defined by BASE and MASK to be cacheable according to the CCR parameters.

SUPV Supervisor Only Mode

0 - Allow both User and Supervisor accesses to be cached.

1 - Allow only Supervisor accesses to be cached.

The SUPV bit can be used to cause this CCR configuration to cache only Supervisor Mode accesses and not User Mode accesses. This setting is useful for applications that only want to cache operating system kernel execution.

MODE Cache Mode

0 - Write-through mode

1 - Copy-back mode

The MODE field allows the CCR configuration to operate in either Write-through mode or Copy-back mode.

Write-through mode implies that all writes to a memory location that is currently in the cache causes a simultaneous update to both the cache entry and the external memory location. Write-through mode is useful when cache coherency between the cache and external memory is required.

Copy-back mode implies that all writes to a memory location that is currently in the cache results in an update to the cache memory entry but not the external memory. In copy-back mode, the external memory location will only be updated when the dirty cache entry is flushed from the cache. Copy-back mode causes cache incoherency, however, offers a performance advantage.

WP **Write Protect**

0 - Allow writes to this block to occur

1 - Disallow writes to this block

The WP field provides the ability to write-protect a cacheable region of memory. Writing to a cacheable region of memory when the WP bit is set to 1, causes the processor to receive a data abort exception. The WP feature is generally used to protect a cacheable region of memory used solely for instruction execution.

FRC32 **Force 32-bit Prefetches**

0 – Disable

1 – Force 32-bit fetches on 0 byte boundaries

The FRC32 field should always be set to 1 when the CCR region is used with a peripheral that is 32-bits wide. The FRC32 field must be set to 0 when the CCR region is used with any peripheral that is not 32-bit wide. The FRC32 bit forces 32-bit operands to always be fetched when address boundaries allow. This feature increases the system performance when THUMB mode instructions are fetched from a 32-bit peripheral.

SET1 **SET1 Enable**

0 – Disable SET1

1 – Enable SET1

The SET1 must be set to 1 in order to attach the SET1 cache segment to the CCR region. Each CCR can be attached to 1, 2, 3, or 4 of the cache sets. The ability to control how many sets are attached to each CCR provides the ability to utilize more cache sets for different memory regions. This feature is typically used to separate instruction and data regions and apply different amounts of cache to each individual region.

SET2 **SET2 Enable**

0 - Disable SET2

1 - Enable SET2

SET3 **SET3 Enable**

0 - Disable SET3

1 - Enable SET3

SET4 **SET4 Enable**

0 - Disable SET4

1 - Enable SET4

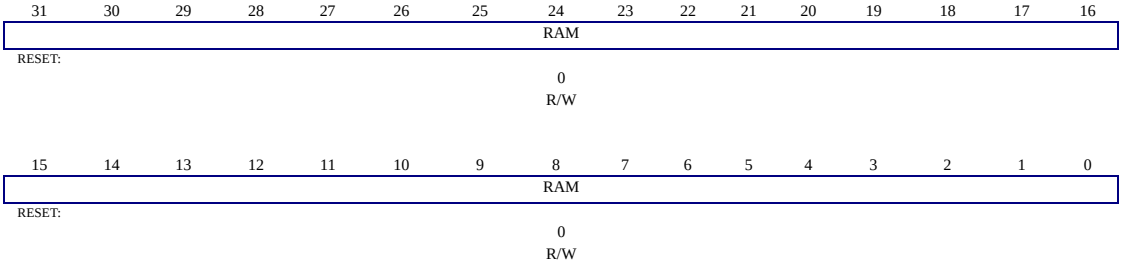
3.4.14 Cache RAM (NET+40 only)

The cache RAM memory is fully addressable by the processor. Accesses to the cache RAM memory block generates a data abort when accessed in non-supervisor mode while the USER bit in the system control register is set to 0. Accesses beyond the physical size of the cache RAM block also generates a data abort.

Table 3-5 identifies how the CACHE RAM memory block is addressed. The information contained in each set falls into contiguous memory locations. When not enabled in any CCR for cache, each set provides 2K bytes of contiguous random access memory. If 2 adjacent sets are not enabled as cache, then 4K bytes of contiguous are available.

When a set is enabled for cache RAM, each entry is 8 bytes wide. The first 4 bytes provide the cache TAG information while the second 4 bytes provide the cache data information. The cache TAG and data fields are modified when information needs to be locked into the cache.

Address = FFF0 000 / FFF0 1FFF



	FFF0 0000		FFF0 0800		FFF0 1000		FFF0 1800	
	+0	+4	+0	+4	+0	+4	+0	+4
+0	Set1 TAG	Set1 DATA	Set2 TAG	Set2 DATA	Set3 TAG	Set3 DATA	Set4 TAG	Set4 DATA
+8								
+10								
Etc.....								
+7F8								

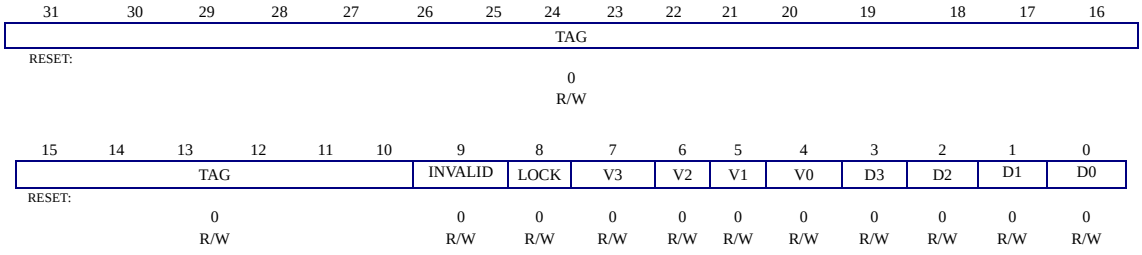
Table 3-5: Cache RAM Memory Addressing

3.4.15 Cache TAG Format (NET+ 40 only)

Each cache line stores 32 bits of data. A cache line is always in one of three states: valid, invalid, or dirty. Each cache line can contain 1 to 4 bytes of valid or dirty data. The cache line status bits identify the state of each cache line. When a line is invalid, the cache line is clear and lookups are ignored. Valid lines contain data consistent with external memory. Dirty lines contain data inconsistent with external memory.

Figure 3-3, 4-Way Set Associated Cache, provides a simplified view of how the cache mechanism works. The caches always use the physical address. To determine if the physical address is currently located in the cache, physical address bits A31-A2 are used to either reference the cache tag or update the cache tag field. If a match occurs, the selected cache line is used and flagged as a “hit.”

Cache TAG Format



TAG Tag Reference

The TAG field identifies which memory location is currently stored within this cache entry. The TAG field is only valid when one of the four "V" valid bits are set to 1. The following formula can be used to calculate the actual physical address of the information for this cache entry.

physicalAddress =

```
(TAG << 10) || ((cacheAddress & 0x000007F8) >> 1)
```

INVALID Invalidate

0 - Location is value for cacheable entries

1- Location is invalid for any cache entries

The INVALID bit is used to prevent specific cache lines within a SET from being used for cache entries. The INVALID field is typically used to reserve cache entries for code that will be locked down in the cache at a later time.

LOCK Locked State

0 - Entry not locked

1- Entry locked in cache

The LOCK field is used to cause an entry to remain static within the cache. The LOCK field can be set to 1 in order to fix the current entry to remain in the cache SET forever. The LOCK bit is typically used to preload software functions in the cache.

V3 Byte 3 Valid

0 – Byte 3 is invalid

1 – Byte 3 is valid

The V3 bit is set to 1 to indicate that Byte 3 in this cache entry is valid. Byte 3 refers to the least significant address byte of a 32-bit word when operating in Big Endian mode, the most significant byte when operating in Little Endian mode.

V2 Byte 2 Valid

0 – Byte 2 is invalid

1 – Byte 2 is valid

The V2 bit is set to 1 to indicate that Byte 2 in this cache entry is valid.

V1 Byte 1 Valid

0 – Byte 1 is invalid

1 – Byte 1 is valid

The V1 bit is set to 1 to indicate that Byte 1 in this cache entry is valid.

V0 Byte 0 Valid

0 – Byte 0 is invalid

1 – Byte 0 is valid

The V0 bit is set to 1 to indicate that Byte 0 in this cache entry is valid. Byte 0 refers to the most significant address byte of a 32-bit word when operating in Big Endian mode, the least significant byte when operating in Little Endian mode.

D3 Byte 3 Dirty

0 – Byte 3 is Clean

1 – Byte 3 is Dirty

The D3 bit is set to 1 to indicate that Byte 3 is dirty and inconsistent with external memory. Byte 3 will be updated in external memory when this cache entry is flushed to make room for a new entry. The dirty bit can only be set when the CCR is configured to operate in copy-back mode. Byte 3 refers to the least significant address byte of a 32-bit word when operating in Big Endian mode, the most significant byte when operating in Little Endian mode.

D2 Byte 2 Dirty

0 – Byte 2 is Clean

1 – Byte 2 is Dirty

The D2 bit is set to 1 to indicate that Byte 2 is dirty and inconsistent with external memory. Byte 3 will be updated in external memory when this cache entry is flushed to make room for a new entry. The dirty bit can only be set when the CCR is configured to operate in copy-back mode.

D1 Byte 1 Dirty

0 – Byte 1 is Clean

1 – Byte 1 is Dirty

The D1 bit is set to 1 to indicate that Byte 1 is dirty and inconsistent with external memory. Byte 3 will be updated in external memory when this cache entry is flushed to make room for a new entry. The dirty bit can only be set when the CCR is configured to operate in copy-back mode.

D0 Byte 0 Dirty

0 – Byte 0 is Clean

1 – Byte 0 is Dirty

The D0 bit is set to 1 to indicate that Byte 0 is dirty and inconsistent with external memory. Byte 3 will be updated in external memory when this cache entry is flushed to make room for a new entry. The dirty bit can only be set when the CCR is configured to operate in copy-back mode. Byte 0 refers to the most significant address byte of a 32-bit word when operating in Big Endian mode, the least significant byte when operating in Little Endian mode.

Chapter 4
DMA Controller Module

4.1 DMA Controller Description

The NET+ARM chip supports 10 DMA controllers that facilitate the movement of data operands between external memory and internal peripherals, minimizing CPU intervention. All DMA channels support both fly-by operations and memory-to-memory operations. When configured for fly-by operation, the DMA controller does not touch the data, rather it controls the flow of data through the BBus and provides the external address for a single data transfer operation. When configured for memory-to-memory operations, the DMA controller uses a temporary holding register between read and write operations; two memory cycles are executed.

Figure 4-1 shows DMA fly-by transfers; Figure 4-2 shows DMA memory-to-memory transfers.

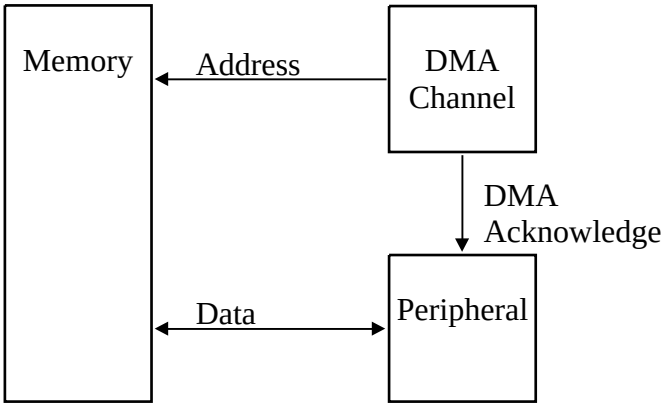


Figure 4-1: DMA Fly-By Transfers

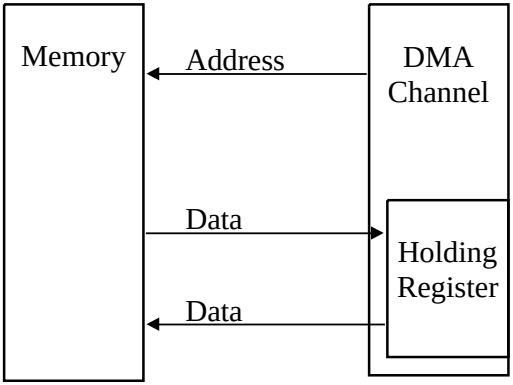


Figure 4-2: DMA Memory-to-Memory Transfers

Figure 4-3 provides a block diagram of the DMA Module. The DMA module has a single centralized state machine and a block of static RAM referred to as the context RAM. The context RAM contains the current state of each DMA channel. The single state machine supports all 10 DMA channels in parallel by context switching from channel to channel. The DMA Channel arbiter determines which channel the state machine is currently operating on.

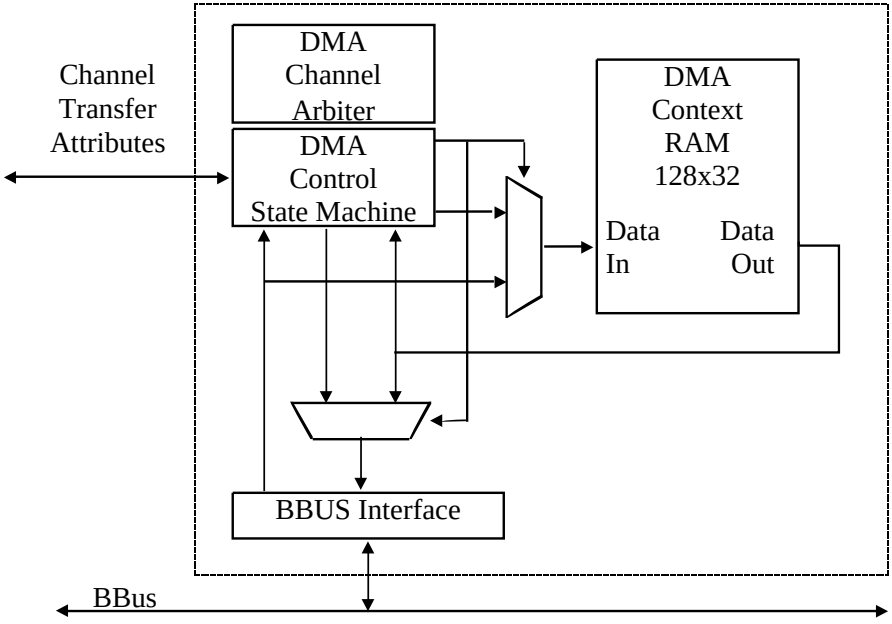


Figure 4-3: DMA Channel Block Diagram

4.2 DMA Buffer Descriptor

All DMA channels operate using a buffer descriptor. Each DMA channel remains idle until the DMA channel buffer descriptor is initialized and the DMA channel is enabled via the channel control register. Once the DMA channel is started, additional buffer descriptors are referenced by the original buffer descriptor.

Each DMA buffer descriptor requires two 32-bit words for fly-by mode and four 32-bit words for memory-to-memory operations. Multiple buffer descriptors are located in contiguous memory locations. The first buffer descriptor address is provided by the DMA channel's buffer descriptor pointer. Subsequent buffer descriptors are found adjacent to the first descriptor. The final buffer descriptor is defined with its "W" bit set. When the W bit is encountered by the DMA Channel, the channel wraps around to the first descriptor.

Each DMA channel can address a maximum of 128 fly-by buffer descriptors or 64 memory-to-memory buffer descriptors. A DMA channel configured for more than the maximum number of buffer descriptors operates in an unpredictable fashion.

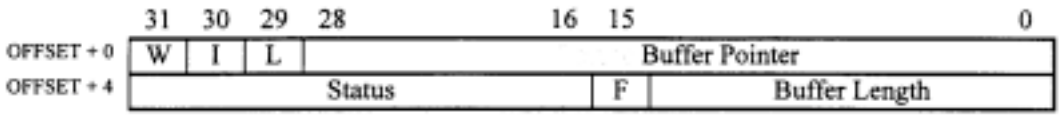


Table 4-1: DMA Buffer Descriptor (Fly-By Mode)

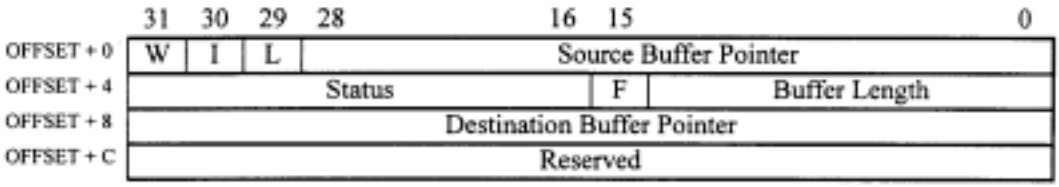


Table 4-2: DMA Buffer Descriptor (Memory-to-Memory Mode)

W The WRAP bit, when set, informs the DMA controller that this is the last buffer descriptor within the continuous list of descriptors. The next buffer descriptor is found using the initial DMA channel buffer descriptor pointer. When the WRAP bit is not set, the next buffer descriptor is found using an offset from the current buffer descriptor.

I The “I” bit, when set, informs the DMA controller to issue an interrupt to the CPU when the buffer is closed due to a normal channel completion. The interrupt occurs regardless of the normal completion interrupt enable configuration for the DMA channel.

L The “L” bit, when set, informs the DMA controller this buffer descriptor is the last descriptor that completes an entire message frame. The DMA controller uses this bit to signal the peripheral. This bit should be used when multiple descriptors are chained together to constitute a data frame.

F For fly-by operations, the “F” bit, when set, indicates the buffer is full. A DMA channel sets this bit after filling a buffer. A DMA channel clears this bit after emptying a buffer. A DMA channel will not attempt to empty a buffer with the “F” bit clear. Likewise, a DMA channel does not attempt to fill a buffer with the “F” bit set. Whenever the “F” bit is modified by firmware, the firmware driver must write to the DMA status/ interrupt enable register in order to activate an idle DMA channel.

For memory-to-memory operations, the “F” bit must be set to 1 in order to begin the DMA operations.

Source Buffer Pointer

The source buffer pointer field identifies the starting location of the data buffer. The source buffer pointer can start on any byte boundary for fly-by memory-to-peripheral operations.

The source buffer pointer must be aligned on 32-bit boundaries to support peripheral-to-memory operations.

Status

The Ethernet controller and serial controllers use the 16-bit Status field to store transmit and receive status words as a result of a completed transmit or receive data frame.

Buffer Length - Peripheral to Memory

The buffer length field is used in fly-by peripheral to memory operations to indicate the maximum number of bytes available in the receive buffer pointed to by the source buffer pointer. After filling a receive buffer with peripheral data, the DMA controller updates this field with the actual receive data byte count.

Buffer Length - Memory to Peripheral

The buffer length field is used in fly-by memory to peripheral operations to indicate the number of bytes to move from the source address pointer to the peripheral device.

After completing a transmit buffer descriptor, the DMA controller updates this field with the actual data byte count (useful during error conditions).

Buffer Length - Memory-to-Memory

The buffer length field is used in memory-to-memory operations to indicate the number of bytes to move between the source destination locations. After completing a memory-to-memory DMA copy, the DMA controller updates this field with the remaining data byte count (useful during error conditions).

Destination Address Pointer

The destination address pointer is only used when the DMA Channel is configured for memory-to-memory operations. The destination address pointer must start on the same byte boundary as the source address pointer. If the source and destination byte boundaries are different, then the data operand size must be set for 8-bit operations (refer to the SIZE field in the DMA control register).

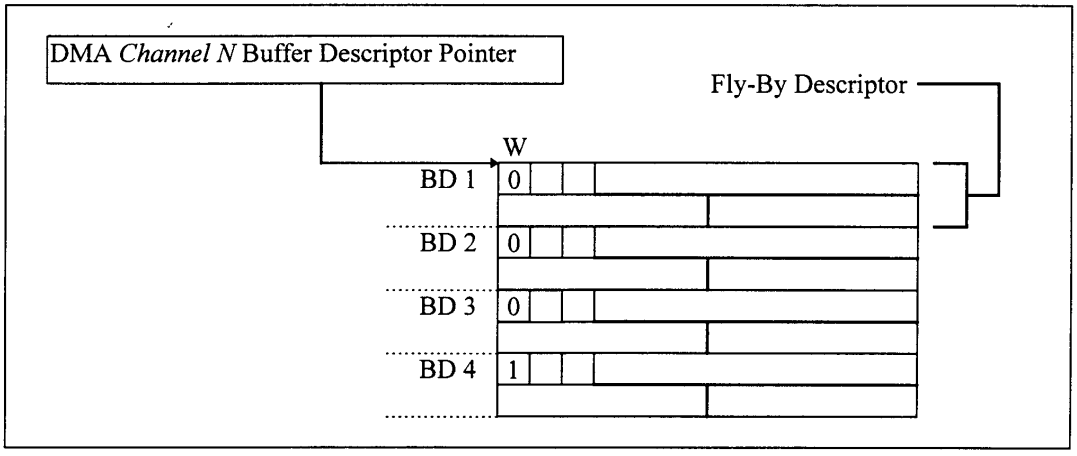


Figure 4-4: DMA Buffer Descriptor Structure

4.3 DMA Controller Assignments

Table 4-3 identifies how the 10 DMA channels are used in NET+ARM chip applications. There are two basic configurations that the NET+ARM chip can be used in. Configuration 1 is designed for IEEE 1284 applications that use 1 to 4 IEEE 1284 parallel ports and 2 serial ports. Configuration 2 is used in ENI applications that provide shared RAM, FIFO Moe, and 2 serial ports.

DMA Channel	Configuration 1 IEEE 1284 Mode	DMA Direction
1	Ethernet Receiver	FB Write
2	Ethernet Transmitter	FB Read
3	IEEE 1284 Channel 1	FB Read/Write
4	IEEE 1284 Channel 2	FB Read/Write
5	IEEE 1284 Channel 3	FB Read/Write
6	IEEE 1284 Channel 4	FB Read/Write
7	SER Channel 1 Receiver	FB Write
8	SER Channel 1 Transmitter	FB Read
9	SER Channel 2 Receiver	FB Write
10	SER Channel 2 Transmitter	FB Read

DMA Channel	Configuration 2 ENI Mode	DMA Direction
1	Ethernet Receiver	FB Write
2	Ethernet Transmitter	FB Read
3	FIFO Mode Receiver	FB Write
4	FIFO Mode Transmitter	FB Read
5	Available for Memory-to-Memory	MM
6	Available for Memory-to-Memory	MM
7	SER Channel 1 Receiver	FB Write
8	SER Channel 1 Transmitter	FB Read
9	SER Channel 2 Receiver	FB Write
10	SER Channel 2 Transmitter	FB Read

Table 4-3: DMA Channel Assignment

DMA Direction Codes:

FB Write	Fly-by peripheral to memory
FB Read	Fly-by memory to peripheral
MM	Memory-to-memory copy from Source Pointer to Destination Pointer

4.4 DMA Channel Configuration

Each DMA channel has a block of configuration space that is mapped into the DMA module configuration space as defined in Table 2-1: Bus Address Decoding.

Address	Register
FF90 0000	DMA 1 “A” Buffer Descriptor Pointer
FF90 0010	DMA 1 “A” Control Register
FF90 0014	DMA 1 “A” Status Register
FF90 0020	DMA 1 “B” Buffer Descriptor Pointer
FF90 0030	DMA 1 “B” Control Register
FF90 0034	DMA 1 “B” Status Register
FF90 0040	DMA 1 “C” Buffer Descriptor Pointer
FF90 0050	DMA 1 “C” Control Register
FF90 0054	DMA 1 “C” Status Register
FF90 0060	DMA 1 “D” Buffer Descriptor Pointer
FF90 0070	DMA 1 “D” Control Register
FF90 0074	DMA 1 “D” Status Register
FF90 0080	DMA 2 Buffer Descriptor Pointer
FF90 0090	DMA 2 Control Register
FF90 0094	DMA 2 Status Register
FF90 00A0	DMA 3 Buffer Descriptor Pointer
FF90 00B0	DMA 3 Control Register
FF90 00B4	DMA 3 Status Register

Table 4-4: DMA Control Registers

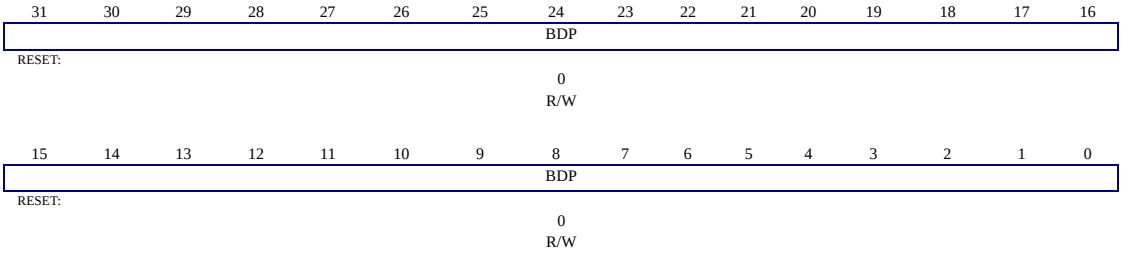
Address	Register
FF90 00C0	DMA 4 Buffer Descriptor Pointer
FF90 00D0	DMA 4 Control Register
FF90 00D4	DMA 4 Status Register
FF90 00E0	DMA 5 Buffer Descriptor Pointer
FF90 00F0	DMA 5 Control Register
FF90 00F4	DMA 5 Status Register
FF90 0100	DMA 6 Buffer Descriptor Pointer
FF90 0110	DMA 6 Control Register
FF90 0114	DMA 6 Status Register
FF90 0120	DMA 7 Buffer Descriptor Pointer
FF90 0130	DMA 7 Control Register
FF90 0134	DMA 7 Status Register
FF90 0140	DMA 8 Buffer Descriptor Pointer
FF90 0150	DMA 8 Control Register
FF90 0154	DMA 8 Status Register
FF90 0160	DMA 9 Buffer Descriptor Pointer
FF90 0170	DMA 9 Control Register
FF90 0174	DMA 9 Status Register
FF90 0180	DMA 10 Buffer Descriptor Pointer
FF90 0190	DMA 10 Control Register
FF90 0194	DMA 10 Status Register

Table 4-4: DMA Control Registers (Continued)

4.4.1 DMA Buffer Descriptor Pointer

The DMA Buffer Descriptor Pointer is a 32-bit register with the following bit definitions. All bits are set to 0 on reset.

Address = FF90 0000 / 20 / 40 / 60 / 80 / A0 / C0 / E0 / 100 / 120 / 140 / 160 / 180



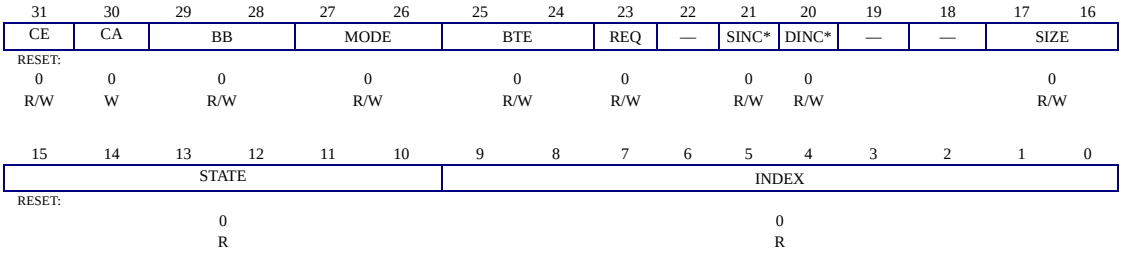
The DMA buffer descriptor pointer is a 32-bit register that provides a pointer to the first buffer descriptor in a contiguous list of descriptors. The last descriptor in the list is identified with the “W” bit set in the buffer descriptor header.

Note that DMA channel 1 is unique in that it supports four different buffer descriptors “A,” “B,” “C,” and “D.” Each buffer descriptor contains a separate ring of descriptors. Each buffer descriptor identifies a block of data buffers with a different size. This feature allows the Ethernet Receiver to choose the most optimum buffer size for the incoming packet.

4.4.2 DMA Control Register

The DMA Control Register is a 32-bit register with the following bit definitions. All bits are set to 0 on reset.

Address = FF90 0010 / 30 / 50 / 70 / 90 / B0 / D0 / F0 / 110 / 130 / 150 / 170 / 190



CE

The channel enable (CE) bit must be set only after the other channel mode bits are set. The DMA channel will begin reading the first buffer description when CE is set to 1.

CA

The channel abort (CA) bit, whenever set, causes the current DMA operation to complete and the buffer to be closed. The CA bit is not automatically cleared after the requested abort is complete; firmware must clear the CA bit after recognizing CAIP active.

BB

The bus bandwidth field (BB) determines how often the DMA channel can arbitrate for access to the bus. When set to 25%, the channel can only arbitrate 1 out of 4 times; 50% gets 2 out of four times; 75% gets 3 out of 4 times; 100% allows the DMA channel to arbitrate each time.

MODE

“00” - Fly-by Write (from Peripheral to Memory)
“01” - Fly-by Read (Memory to Peripheral)
“10” - Memory-to-Memory (Source to Destination)
“11” - Reserved

The mode field (MODE) identifies the data transfer mode. Each DMA channel can be configured to operate in either fly-by or memory-to-memory mode. Refer to Table 4-1 and Table 4-2.

BTE

“00” - No Burst

“01” - 8-byte burst
“10” - 16-byte burst
“11” - Reserved

The burst transfer enable (BTE) field determines if the DMA channel can use burst transfers through the bus. This configuration applies to both buffer descriptor and peripheral data access. For DMA channel 1, the BB, MODE, and BTE configuration must be the same in all four control registers, A, B, C, and D.

REQ

0 - Internal Request
1 - External Source

The REQ bit can be used to allow channels 3 and/or 4 to interface with external DMA connections. The external DMA connections are provided by I/O ports A and B.

SINC* and DINC*

SINC*

0 - Increment Source Address Pointer
1 - Do not increment Source Address Pointer

DINC*

0 - Increment Destination Address Pointer
1 - Do not increment Destination Address Pointer

The SINC* and DINC* bits can be used to control whether or not the source and/or destination address pointers are incremented after each DMA transfer. These bits are used by the DMA controller in all modes whenever referring to a memory address. These bits are not applicable when the source or destination is an internal peripheral resource.

SIZE

“00” - 32-bit
“01” - 16-bit
“10” - 8-bit
“11” - reserved

The SIZE configuration bits are used by the DMA controller whenever configured for external request mode (REQ bit; Channels 3 and 4 only), or memory-to-memory DMA mode. These bits define the size of each DMA transaction.

STATE

0 : IDLE
1 : Load source buffer address
4 : Load destination buffer address
8 : First operand

10: Memory-to-memory second operand

20: Update buffer description

The STATE field describes the current state of the DMA controller state machine.

INDEX

The INDEX field identifies the DMA controller's current byte offset pointer relative to the DMA buffer descriptor pointer.

4.4.3 DMA Status/Interrupt Enable Register

The DMA status register is a 32-bit register with the following bit definitions:

Address = FF90 0014 / 34 / 54 / 74 / 94 / B4 / D4 / F4 / 114 / 134 / 154 / 174 / 194

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NCIP	ECIP	NRIP	CAIP	—	—	—	—	NCIE	ECIE	NRIE	CAIE	WRAP	IDONE	LAST	FULL
RESET:															
0	0	0	0					0	0	0	0	0	0	0	0
R/C	R/C	R/C	R/C					R/W	R/W	R/W	R/W	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
—	BLEN														
RESET:															
								0							
								R							

The interrupt enable (IE) bits can be set to cause an interrupt to occur when the corresponding interrupt status bit is set in the status register. The interrupt pending (IP) bits are set to indicate begin active. These bits are cleared by writing a 1 to the same bit locations.

NCIP

The NCIP bit is set when a buffer descriptor has been closed (for normal conditions) and either the NCIE bit is set or the IDONE bit was found active in the current buffer descriptor. A normal DMA channel completion occurs when the BLEN count expires to zero or when a peripheral device signals completion.

ECIP

The ECIP bit is set when the DMA channel encounters either a bad buffer descriptor pointer or a bad data buffer pointer. When the ECIP bit is set, the DMA channel stops until the ECIP bit is cleared by firmware. The DMA channel does not advance to the next buffer descriptor. When ECIP is cleared by firmware the buffer descriptor is retried from where it left off. The CA bit in the control register can be used to abort the current buffer descriptor and advance to the next descriptor.

NRIP

The NRIP bit is set when the DMA channel encounters a buffer descriptor whose “F” bit is in the incorrect state. When the NRIP bit is set, the DMA channel stops until the NRIP bit is cleared by firmware. The DMA channel does not advance to the next buffer descriptor. When NRIP is cleared by firmware the buffer descriptor is retried.

CAIP

The CAIP bit is set when the DMA channel detects the CA bit set in the control register. When CAIP is set, the DMA channel will stop until the CAIP bit is cleared by firmware. The DMA channel automatically advances to the next buffer descriptor after CAIP is cleared. The CA bit in the control register must be cleared, via firmware, before the CAIP is cleared. Failure to reset the CA bit causes the subsequent buffer descriptor to also abort.

NCIE, ECIE, NRIE, and CAIE

The NCIE, ECIE, NRIE, and CAIE bits are used to enable interrupts to be generated when the associated IP bits are set. In general, the NCIE is used for inbound (WRITE DMA) operations. The DMA Buffer Descriptor “I” bit should be used for outbound (READ DMA) operations. The ECIE and CAIE bits should always be enabled. NCIE and DMA Buffer Descriptor “I” bit should not be used at the same time.

WRAP, IDONE, LAST, FULL, and BLEN

The WRAP, IDONE, LAST, FULL, and BLEN status bits are provided for debugging purposes. They serve no useful purpose to the firmware device driver.

4.5 Ethernet Receiver Considerations

When an Ethernet frame is received, DMA channel 1 hunts through the four buffer descriptors in search of the most optimum buffer size. The DMA channel searches the four descriptors starting with “A,” then “B,” “C,” and finally “D.” The search stops as soon as the DMA channel encounters an available buffer that is large enough to hold the entire receive frame. The search also stops when the DMA channel encounters a control register whose channel enable (CE) bit is zero.

Interrupts are set when the DMA channel encounters buffers that are “not ready.” As such, the device driver should be designed with the smallest buffers in the “A” pool and the largest buffers in the “D” pool. The number of available pools can be configured (from 1 to 4) with proper use of the CE bits.

4.6 External Peripheral DMA Support

NET+ARM DMA Channels 3 and 4 can be setup for external DMA transfers and use three signals (DREQ*, DACK*, and DONE*) to facilitate communications between themselves and an external device. It is up to the external device to source or react to these three signals. The NET+ARM can treat the external device as a data latch and do Fly-by transfers, or as a block of memory and do memory-to-memory transfers. Figure 4-6 shows how the signals are connected between the NET+ARM, an external device, and memory when in the Fly-by mode. Figure 4-7 shows how the signals are connected between the NET+ARM, an external device, and memory when in the memory-to-memory mode. Not shown are the CAS and RAS signals from the NET+ARM to memory that would be needed in either mode if the memory were DRAM.

4.6.1 Signal Description

Figure 4-5 is a rough timing diagram of the DREQ*, DACK*, and DONE* signals. A more detailed timing diagram is available in Chapter 13, *Electrical Specifications* (Figure 13-21).

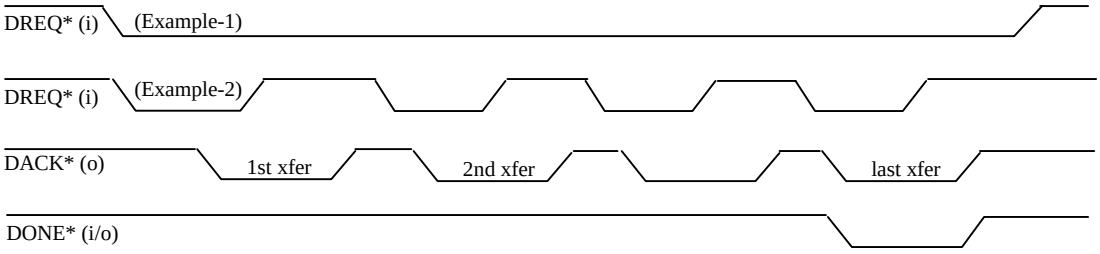


Figure 4-5: External DMA Timing

DREQ*

DREQ* is an input to the NET+ARM, sourced by the external device. All transfers are initiated when the external device asserts DREQ* low. When the external device wants a DMA transfer (either read or write) it will assert the DREQ* signal. The DREQ* can stay low until all data transfers are complete (see the first example of DREQ* in Figure 4-5) or it can be asserted once for each transfer (see the second example of DREQ* in Figure 4-5). Either method is treated the same by the NET+ARM.

DACK*

DACK* is an input to the external device, sourced by the NET+ARM. After the NET+ARM receives DREQ*, and it is ready for a data transfer, it will assert DACK* low at the same time that it asserts the data transfer signals (ADDR, R/W, CAS, RAS, etc). DACK* should be used by the external device as an enable for its memory. If DACK* and R/W* are low the external device should take data from its memory and put it on the data bus. If DACK* is low and R/W* is high, the external device should take the data that is on the data bus and put it in its memory.

DONE*

The DONE* signal is either an input or an output depending upon configuration of the direction bit in the GPIO configuration register.

When the transfer is a fly-by write (external device to memory), the external device can send DONE* to the NET+ARM to indicate that it has no more data to send. The external device should make sure that the DONE* signal it sends to the NET+ARM is asserted low while DACK* is asserted low (it should OR the DONE* signal it sends with the DACK* signal it receives). When the NET+ARM receives the DONE* signal, it will close the current buffer and set the NCIP (Normal Completion Interrupt

Pending) bit in the DMA Status Register. The number of bytes that were received will be in the Buffer Length field of the DMA buffer descriptor.

When the transfer is a fly-by read (memory to external device), the NET+ARM can send DONE* to the external device to indicate that the last DMA buffer has been emptied.

The NET+ARM will assert the DONE* output to the external device when the 'L'(Last) bit is set in the buffer descriptor and the buffer has been emptied. The NCIP (Normal Completion Interrupt Pending) bit in the DMA Status Register will be set.

4.6.2 External DMA Configuration

The REQ bit in the DMA Control Register must be set to 1 for External Source and the DMA signals must be chosen at I/O Ports 'A' or 'B.' Everything else is set the same as any other DMA transfer.

The NET+ARM pins that have the DREQ*, DACK*, and DONE* signals are multifunctional and are shared with GPIO ports A and B. Port-A has the DMA channel-3 signals and Port-B has the DMA channel-4 signals. Select the DMA-3 signals with the Port-A Register (0xFFB00020) and the DMA-4 signals with the Port-B Register (0xFFB00024). To set DONE* as an input, set Port-A or B bit 0 to Mode = 1 and Dir = 0. To set DONE* as an output, set Port-A or B bit 0 to Mode = 1 and Dir = 1.

DMA 3-Channel Signals				DMA 4-Channel Signals		
Pin #	**Signal Name**	**Port-A #**		**Pin #**	**Signal Name**	**Port-B #**
39	DREQ1*	6		47	DREQ2*	6
43	DACK1*	2		51	DACK2*	2
45	DONE1*	0		55	DONE2*	0

Table 4-5: DMA Channel Signals

Enable the Channel-3 signals with the Port-A register:

0xFFB00020 = 0x45040000 (enable DREQ1*, DACK1*, and DONE1* = input)

0xFFB00020 = 0x45050000 (enable DREQ1*, DACK1*, and DONE1* = output)

Enable the Channel-4 signals with the Port-B register:

0xFFB00024 = 0x45040000 (enable DREQ2*, DACK2*, and DONE2* = input)

0xFFB00024 = 0x45050000 (enable DREQ2*, DACK2*, and DONE2* = output)

4.6.3 Fly-by Mode

When in the Fly-by mode, the NET+ARM treats the external device as an I/O port that it can write to or read from. The NET+ARM provides a 32 bit data bus, a 28-bit address bus, a read/write signal and the DMA transfer signals. Figure 4-6 is a block diagram of the hardware needed for external Fly-by DMA transfers.

4.6.4 Fly-by Write

During a Fly-by Write cycle, it is up to the external device to provide data that will be written to memory. When the external device has data to write, it should assert DREQ* low. The NET+ARM will then assert DACK* and R/W* low and also provide the address of the memory location where the data will be written. The NET+ARM provides ADDR[27:0], R/W* and CSx* signals to the memory to accomplish data transfers. The external device should enable the data onto the data bus when R/W* and DACK* are low and tri-state the data when R/W* or DACK* is high. The DONE* signal should be configured as an input to the NET+ARM. When the external device has no more data to send, it should assert DONE* low.

4.6.5 Fly-by Read

During a Fly-by Read, cycle data is read from memory onto the data bus so the external device can receive it. When the external device wants a data transfer, it should assert DREQ* low. The NET+ARM will then assert DACK* low and also provide the address of the memory location where the data will be read. The R/W* signal will remain high. Data is valid after DACK* goes low; see Chapter 13, *Electrical Specifications* (Figure 13-21) for the exact timing. The DONE* signal should have been configured as an output from the NET+ARM. The NET+ARM will assert the DONE* output low to the external device when the ‘L’ (Last) bit is set in the buffer descriptor and the buffer has been emptied.

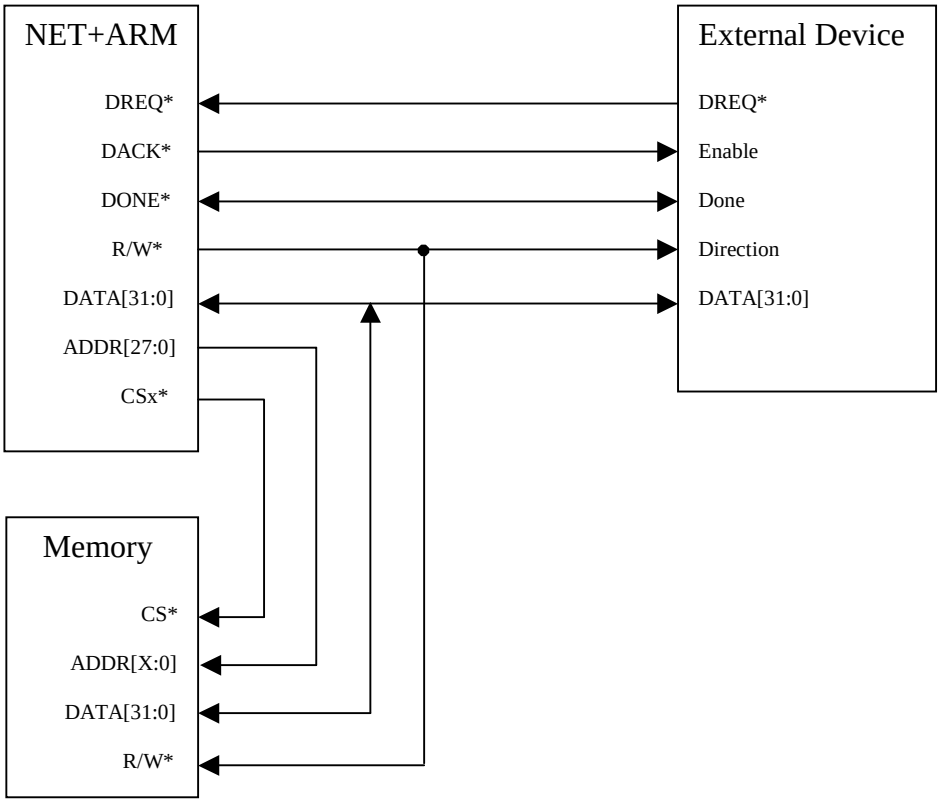


Figure 4-6: Hardware Needed for External Fly-by DMA Transfers

4.6.5 Memory-to-Memory Mode

When in the memory-to-memory mode the NET+ARM treats the external device as an addressable block of memory. The NET+ARM provides the external device the same signals as it would in Fly-by mode plus it also provides the address bus and a CSx* signal so it can access the external device's memory. Figure 4-7 is a block diagram of the hardware needed for external memory-to-memory DMA transfers.

The configuration of the buffer descriptor determines which side (NET+ARM memory or external device memory) will source the data. If the Source Buffer Pointer points to external device memory, then the Destination Buffer Pointer must point to NET+ARM memory. The transfer will then be similar to a Fly-by Write, the DONE*

must be configured as an input and DREQ* and DACK* will act the same as in Fly-by mode.

If the Source Buffer Pointer points to NET+ARM memory, then the Destination Buffer Pointer must point to external device memory. The transfer will then be similar to a Fly-by Read, the DONE* must be configured as an output and DREQ* and DACK* will act the same as in Fly-by mode. In either direction the external device initiates the transfer by asserting DREQ* low, the NET+ARM will then take over and control the memory cycles.

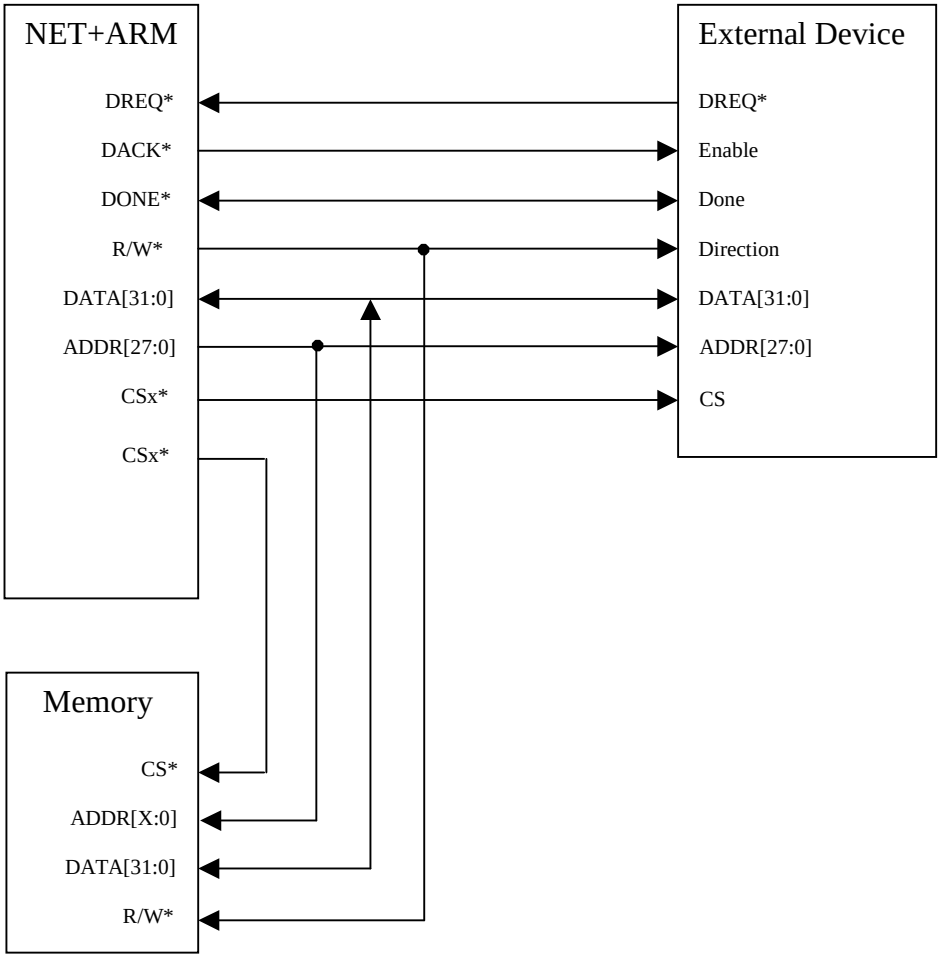


Figure 4-7: Hardware Needed for External Memory-to-Memory DMA Transfers

4.7 DMA Controller Reset

The entire DMA Controller Module can be reset without affecting any of the NET+ARM modules, by simply setting the DMA reset bit in the System Control Register to “1” and then back to “0.”

The DMA Controller Module is also reset by all forms of hardware and software resets.

Chapter 5
Ethernet Controller Module

The Ethernet controller module is divided into two modules: the EFE module, which provides the Ethernet front end interface, and the MAC module, which provides the Ethernet media access controller for both 10 and 100 Mbit applications.

5.1 Ethernet Front End (EFE) Module

Figure 5-1 shows the hardware blocks required in the Ethernet front-end module. The Ethernet front end is responsible for providing the FIFOs and glue logic required to interface the MAC with the NET+ARM chip architecture.

The basic features of the EFE include:

- Control/Status Registers for MAC, Transmitter, and Receiver
- 128-Byte Transmit FIFO
- 2048-Byte Receive FIFO
- DMA Interface Logic

All control and status registers required by the Ethernet module are provided by the EFE logic. The transmitter and receiver each provide a 16-bit status word after processing each Ethernet frame. These status words can be provided to the CPU on an interrupt basis or automatically moved to the DMA buffer descriptor for the associated Ethernet frame.

The EFE contains a 128-byte transmit FIFO and a 2048-byte receive FIFO. The transmit FIFO allows the critical portion of the transmit buffer to sit in the FIFO while collisions occur on the Ethernet medium. This scheme removes the need for the transmitter to fetch the buffer multiple times from memory.

The large 2048-byte receive FIFO allows the entire Ethernet frame to be received and wait in the FIFO while the receive byte count is analyzed. The receive byte count is analyzed to determine the optimum buffer descriptor for DMA transfer. The DMA channel assigned to the Ethernet receiver has the option of using four different size receive buffers. Only successfully received frames, with acceptable destination addresses, are committed to external system memory.

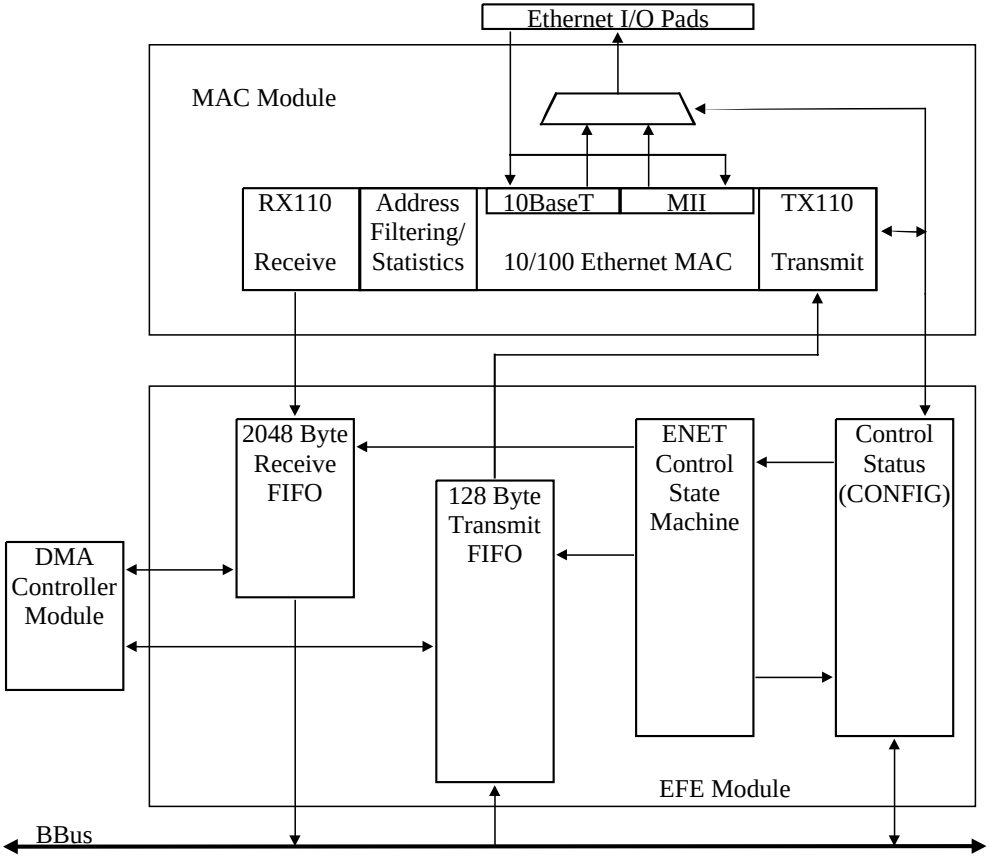


Figure 5-1: Ethernet Controller Module Block Diagram

5.2 Media Access Controller (MAC) Module

The MAC module interfaces with the EFE module on one side and the Ethernet I/O pins on the other. The MAC module supports both ENDEC and Media Independent Interfaces (MII) under firmware control.

Basic features in the MAC include:

- 100Mbit Ethernet MAC
- MII Interface
- Optional 100Mbit Physical Coding Sublayer
- 10Mbit ENDEC Interface
- MII Management Function
- Address Filtering
- Statistics Gathering

The MAC module provides a full-function 100Mbit Ethernet MAC that can be connected to the following devices through the MII: a 100Mb PHY, a 10Mb PHY, a 100Mbit TP-PMD or an ENDEC. The MAC module also provides the MII management feature.

The MAC module provides both the Station Address Logic (SAL) and Statistics Gathering (STAT) blocks. The SAL block filters the incoming receive addresses. The filter causes the receive packets to be accepted or rejected. The STAT block stores statistics for discarded transmit or receive packets.

Reading or writing the MAC configuration registers (address location 0xFF800400 through 0xFF8005DC) is unreliable without valid clocks available on the TXCLK and RXCLK input pins.

5.3 Ethernet Controller Configuration

The Ethernet controller has a block of configuration space that is mapped into the DMA module configuration space as defined in Table 2-1: BBus Address Decoding.

Address	Register
FF80 0000	Ethernet General Control Register
FF80 0004	Ethernet General Status Register
FF80 0008	Ethernet FIFO Data Register
FF80 000C	
FF80 0010	Ethernet Transmit Status Register
FF80 0014	Ethernet Receive Status Register
FF80 0400	MAC Configuration Register
FF80 0404	MAC Test Register
FF80 0408	PCS Configuration Register
FF80 040C	PCS Test Register
FF80 0410	STL Configuration Register
FF80 0414	STL Test Register
FF80 0440	Transmit Control Registers
FF80 0464	
FF80 0480	Receive Control Registers
FF80 0488	
FF80 04C0	Link Fail Counter
FF80 0500	10Mbit Jabber Counter
FF80 0504	10Mbit Loss of Carrier Counter
FF80 0540	MII Command Register
FF80 0544	MII Address Register
FF80 0548	MII Write Data Register
FF80 054C	MII Read Data Register
FF80 0550	MII Indicators Register
FF80 0580	CRC Error Counter

Table 5-1: Ethernet Configuration Registers

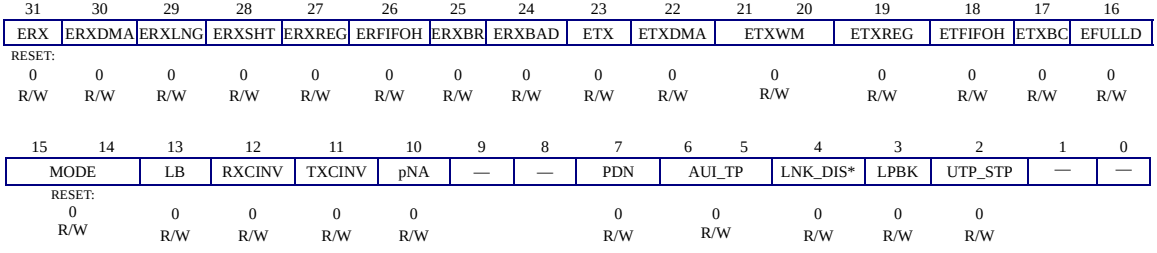
Address	Register
FF80 0584	Alignment Error Counter
FF80 0588	Code Error Counter
FF80 058C	Long Frame Counter
FF80 0590	Short Frame Counter
FF80 0594	Late Collision Counter
FF80 0598	Excessive Deferral Counter
FF80 059C	Maximum Collision Counter
FF80 05C0	SAL Address Filter Register
FF80 05C4	SAL Station Address
FF80 05C8	SAL Station Address
FF80 05CC	SAL Station Address
FF80 05D0	SAL Hash Table
FF80 05D4	SAL Hash Table
FF80 05D8	SAL Hash Table
FF80 05DC	SAL Hash Table

Table 5-1: Ethernet Configuration Registers (Continued)

5.3.1 Ethernet General Control Register

The Ethernet general control register is a 32-bit register. The register bit assignments are defined as follows:

Address = FF80 0000



ERX **Enable RX FIFO**

The ERX bit must be set to active high to allow data to be received from the MAC receiver. The ERX can be used to reset the receive side FIFO. In general, the ERX bit should be set once on device open.

ERXDMA **Enable Receive DMA**

The ERXDMA bit must be set to active high to allow the EFE module to issue receive data move requests to the DMA controller. This bit can be cleared to temporarily stall receive side Ethernet DMA. This bit should not be set when operating the Ethernet receiver in interrupt service mode. In general, the ERXDMA bit should be set once on device open.

ERXLNG **Accept Long (> 1518 bytes) Receive Packets**

When set, the ERXLNG bit allows packets that are larger than 1518 bytes to be accepted by the MAC. In general, the ERXLNG bit should be set once on device open. The ERXLNG bit is typically set only for debugging.

ERXSHT **Accept Short (< 64 bytes) Receive Packets**

When set, the ERXSHT bit allows packets that are smaller than 64 bytes to be accepted by the MAC. In general, the ERXSHT bit should be set once on device open. The ERXSNT bit is typically set only for debugging.

ERXREG **Enable Receive Data Register Ready Interrupt**

The ERXREG bit must be set to active high to generate an interrupt when data is available in the RX FIFO. The ERXREG bit should only be set when operating the Ethernet receive in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and ERXREG is never required to be set.

ERFIFOH **Enable Receive Data FIFO Half Full Interrupt**

The ERFIFOH bit must be set to active high to generate an interrupt when the RX FIFO is at least half full (1024 bytes). The ERFIFOH bit should only be set when operating the Ethernet receive in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and ERFIFOH is never required to be set.

ERXBR **Enable Receive Buffer Ready Interrupt**

The ERXBR bit must be set to active high to generate an interrupt when a new data packet is available in the RX FIFO. The ERXBR bit should only be set when operating the Ethernet receive in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and ERXBR is never required to be set.

ERXBAD **Accept Bad Receive Packets**

When set, the ERXBAD bit allows packets received in error to be accepted by the MAC. Bad receive packets include those packets with CRC errors, alignment errors, and dribble errors. In general, the ERXBAD bit should be set once on device open. The ERXBAD bit is typically set only for debugging.

ETX **Enable TX FIFO**

The ETX bit must be set to active high to allow data to be written to the TX FIFO. The ETX bit can be used to reset the transmit side FIFO. In general, the ETX bit should be set once on device open.

ETXDMA **Enable Transmit DMA**

The ETXDMA bit must be set to active high to allow the EFE module to issue transmit data move requests to the DMA controller. This bit can be cleared to temporarily stall transmit side Ethernet DMA. This bit should not be set when operating the Ethernet receiver in interrupt service mode. In general, the ETXDMA bit should be set once on device open.

ETXWM **Transmit FIFO Water Mark before Transmit Start**

The ETXWM field identifies the minimum number of bytes required in the transmit FIFO before the Ethernet transmitter attempts sending the packet. A larger watermark setting increases transmit packet latency, however, allowing for more slack in the memory system FIFO fill rate. In general, the ETXWM field should be set once on

device open. For NETA12 devices, an errata requires the ETXWM to always be set to a value of “11.”

ETXREG **Enable Transmit Data Register Ready Interrupt**

The ETXREG bit must be set to active high to generate an interrupt when the TX FIFO is ready to accept data. The ETXREG bit should only be set when operating the Ethernet transmitter in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and ETXREG is never required to be set.

ETFIFOH **Enable Transmit Data FIFO Half Empty Interrupt**

The ETFIFOH bit must be set to active high to generate an interrupt when the TX FIFO is at least half empty (< 64 bytes). The ETFIFOH bit should only be set when operating the Ethernet transmitter in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and ETFIFOH is never required to be set.

ETXBC **Enable Transmit Buffer Complete Interrupt**

The ETXBC bit must be set to active high to generate an interrupt when the transmit packet transmission is complete. The ETXBC bit should only be set when operating the Ethernet transmitter in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and ETXBC is never required to be set.

EFULLD **Enable Full Duplex Operation**

The EFULLD bit must be set to active high to allow the Ethernet TX and RX DMA operations to operate simultaneously. This bit must be set when the Ethernet link negotiation results in full duplex operation. In general, the EFULLD bit should be set once on device open.

MODE **Ethernet Interface Mode**

The MODE field identifies what type of Ethernet PHY is attached to the NET+ARM chip. The NET+ARM chip supports two basic styles of Ethernet PHY; the MII PHY or the ENDEC PHY.

The NET+ARM chip supports two styles of ENDEC interfaces; SEEQ and Level1. In SEEQ mode, the LPBK bit is inverted before driving the MDC pin. In Level1 mode, the PDN signal is driven using an open-drain output driver.

LB **MAC Configured in Internal Loopback**

The LB bit must be set active high whenever the MAC or PCS blocks are configured to operate in internal loopback mode. Failure to set the LB bit in these conditions results in erratic receive side behavior.

RXCINV **Invert the Receive Clock Input**

The RXCINV bit should only be set when the external Ethernet PHY generates a clock, which is inverted in phase relative to what the MAC is expecting. This bit is not required for most commercially available PHY devices.

TXCINV **Invert the Transmit Clock Input**

The TXCINV bit should only be set when the external Ethernet PHY generates a clock, which is inverted in phase relative to what the MAC is expecting. This bit is not required for most commercially available PHY devices.

pNA **pSOS pNA Buffer Descriptors**

The EFE module can be configured to allow the receiver to provide receive data, which is compatible with the pSOS pNA Ethernet header and data block formats. The pSOS pNA configuration forces the receiver to insert 2 bytes of data padding between the 14-byte header and the data block sections. This configuration aligns both the header and data blocks on a 32-bit long word boundary.

D7:D0 **ENDEC Media Control Bits**

The least significant 8 bits of the EGCR are used only when the MODE field is configured for ENDEC mode. In this configuration, various NET+ARM chip signals are reconfigured to provide simple control outputs. The control outputs can be used to manipulate control signals on the external ENDEC PHY device.

Table 5-2 shows the relationship between the lower bits in the EGCR and the NET+ARM chip pin, which they control. The NET+ARM chip pins can only be controlled when the MODE field is configured for ENDEC mode.

.

Data Bit	Signal Name	NET+ARM Chip Pin	State Inversion
D7	PDN	TXD1	SEEQ: Inverted TTL Level 1: Inverted Open Drain
D6	AUI_TP[1]	TXD2	No Inversion
D5	AUI_TP[0]	TXD3	No Inversion
D4	LNK_DIS*	TXER	No Inversion

Table 5-2: ENDEC Control Signal Cross-Reference

Data Bit	Signal Name	NET+ARM Chip Pin	State Inversion
D2	LPBK	MDC	Seeq: Inverted Level1: Inverted
D1	UTP_STP	MDO	No Inversion

Table 5-2: ENDEC Control Signal Cross-Reference

Media Control (ENDEC Mode Only)

PDN **Power Down Mode**

0 - Normal

1 - External PHY in low power mode

AUI_TP **AUI/TP Selection**

“00” - Auto Select (with LNK_DIS* = 1)

“01” - AUI Port

“10” - TP Port

“11” - Reserved

LNK_DIS* **10BaseT Link Pulse Disable**

0 - Disable Link Pulse Detection

1 - Enable Link Pulse Detection

LPBK **Enable Loopback Mode**

0 - Normal Mode

1 - External Loopback Mode

UTP_STP **Unshielded/Shielded Selection**

0 - STP wiring

1 - UTP wiring

5.3.2 Ethernet General Status Register

The Ethernet General Status Register is a 32-bit register. The register bit assignments are defined as follows:

Address = FF80 0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
—	—	RXFDB		RXREGR	RXFIFOH	RXBR	RXSKIP	—	—	—	—	TXREGE	TXFIFOH	TXBC	TXFIFOE
RESET:															
		0		0	0	0	0					0	0	0	0
		R		R	R	R/C	R/W					R	R	R/C	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FullDuplex	—	MANSENSE	AUD_TP	LINKPUL*	ATUOMAN	—	—	—	—	—	—	—	—	—	—
RESET:															
0		0	0	0	0										
R		R	R	R	R										

RXFDB **Receive FIFO Data Bytes Available**

This field must be used in conjunction with RXREGR. When RXREGR is set, this field identifies how many bytes are available in the receive data register. The RXFDB field is only used when operating the Ethernet receiver in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and the RXFDB can usually be ignored.

RXREGR **Receive Register Ready**

The RXREGR bit is set whenever data is available to be read from the FIFO data register. This bit, when active high, can cause an interrupt to occur when the ERXREGR bit is set. The RXREGR bit is never active while RXBR is set. The RXBR bit must be cleared to activate RXREGR. The RXREGR bit is only used when operating the Ethernet receiver in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and the RXREGR bit can usually be ignored.

RXFIFOH **Receive FIFO Half Full**

The RXFIFOH bit is set whenever the receive FIFO is at least half full (> 1024 bytes). This bit, when active high, can cause an interrupt to occur when the ERFIFOH bit is set. The RXFIFOH bit is only used when operating the Ethernet receiver in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and the RXFIFOH bit can usually be ignored.

RXBR **Receive Buffer Ready**

The RXBR bit is set whenever a new packet is available in the RX FIFO. This bit, when active high, can cause an interrupt to occur when the ERXBR bit is set. The RXBR bit is an indication to the interrupt service routine to read the receive status register. After the receive status register is read, the RXBR bit can be cleared by writing a 1 to the RXBR bit position in the EGSR. After the RXBR bit is cleared, the RXREGR and RXFIFOH bits become active. The RXBR bit is only used when operating the Ethernet receiver in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and the RXBR bit can usually be ignored. While operating in DMA mode, the RXBR is automatically cleared by the DMA controller.

RXSKIP **Receive Buffer Skip**

The RXSKIP bit can be written in lieu of clearing the RXBR bit. Writing to the RXSKIP bit clears the RXBR bit and flushes the next available packet from the receive FIFO. This RXSKIP bit might be used as a result of reading the receive status register. The interrupt service routine may wish to simply flush the packet from the FIFO instead of reading it out of the FIFO. The RXSKIP bit is a simple means of performing receive packet filtering in the interrupt service routine. The RXSKIP bit is only used when operating the Ethernet receiver in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and the RXSKIP bit can usually be ignored.

TXREGE **Transmit Register Empty**

The TXREGE bit is set whenever the transmit FIFO is ready to accept data. This bit, when active high, can cause an interrupt to occur when the ETXREGE bit is set. The TXREGE bit is never active while TXBC is set. The TXBC bit must be cleared to activate TXREGE. The TXREGE bit is only used when operating the Ethernet receiver in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and the TXREGE bit can usually be ignored.

TXFIFOH **Transmit FIFO Half Empty**

The TXFIFOH bit is set whenever the transmit FIFO is at least half empty (< 64 bytes). This bit, when active high, can cause an interrupt to occur when the ETFIFOH bit is set. The TXFIFOH bit is only used when operating the Ethernet receiver in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and the TXFIFOH bit can usually be ignored.

TXBC **Transmit Buffer Complete**

The TXBC bit is set whenever packet transmission has completed. This bit, when active high, can cause an interrupt to occur when the ETXBC bit is set. The TXBC bit is an indication to the interrupt service routine to read the transmit status register. After the transmit status register is read, the TXBC bit can be cleared by writing a 1 to the TXBC bit position in the EGSR. After the TXBC bit is cleared, the TXREGE and TXFIFOH bits become active. The TXBC bit is only used when operating the Ethernet receiver in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable and the TXBR bit can usually be ignored. While operating in DMA mode, the TXBC is automatically cleared by the DMA controller.

TXFIFOE **Transmit FIFO Empty**

The TXFIFOE bit is active whenever the Transmit FIFO is empty.

D15:D0 **ENDEC Media Status Bits**

The least significant 16 bits of the EGSR are only used when the MODE field is configured for ENDEC mode. In this configuration, various NET+ARM chip signals are reconfigured to provide simple status inputs. The status inputs can be used to monitor status signals on the external ENDEC PHY device.

Table 5-3 shows the relationship between the lower bits in the EGSR and the NET+ARM chip pin, which they monitor. The NET+ARM chip pins can only be monitored when the MODE field is configured for ENDEC mode.

Data Bit	Signal Name	NET+ARM Pin
D15	FullDuplex	RXD2
D13	Mansense	RXD1
D12	AUI_TP	RXD3
D11	LINKPUL*	RXER
D10	Automan	RXDV

Table 5-3: ENDEC Status Sign Cross-Reference

5.3.3 Ethernet FIFO Data Register

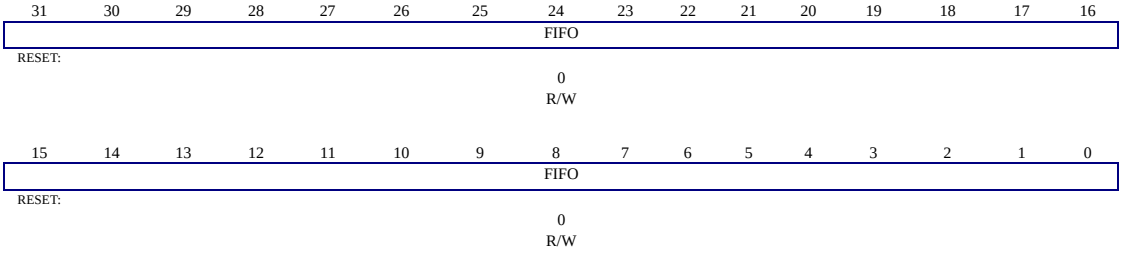
The FIFO data register is a 32-bit register used to manually interface with the Ethernet FIFO in lieu of using DMA support. This register is anticipated to be used primarily as a diagnostic tool.

Writing to this register loads the transmit FIFO. This register can only be written when the TX FIFO register empty bit is set in the general status register, which indicates space is available in the transmit FIFO.

The Transmit FIFO has a secondary address (FF80 000C) that signifies the last word of a transmit frame. The first and middle words must use the primary address (FF80 0008).

Reading from this register empties the receive FIFO. The general status register identifies how many bytes are available to be read. The receive FIFO is only available after the RXBC bit has been cleared in the general status register. The receiver status register should be read before clearing the RXBC bit. When operating in DMA mode, the reading of the receiver status register and clearing of the RXBC bit are performed automatically.

Address = FF80 000C



5.3.4 Ethernet Transmit Status Register

The transmit status register is a 32-bit register that contains the status for the last completed transmit buffer. The transmit buffer complete bit (TXBC) is set in the general status register whenever a transmit frame is completed and the transmit status register is loaded. The lower 16 bits of the transmit status register are also loaded into the StatusOrIndex field of the DMA buffer descriptor (when using DMA mode).

Address = FF80 0010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RESET:															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXOK	TXBR	TXMC	TXAL	TXAED	TXAEC	TXAUR	TXAJ	—	TXDEF	TXCRC	TXLCOL	TXCOLC			
RESET:															
0	0	0	0	0	0	0	0		0	0	0	0			
R	R	R	R	R	R	R	R		R	R	R	R			

TXOK **Transmit Packet OK**

The TXOK bit is set to 1 when the last Ethernet packet was transmitted without error. TXOK is set to 0 when the last Ethernet packet was not transmitted.

TXBR **Transmit Broadcast Packet**

The TXBR bit is set to 1 to indicate the last Ethernet packet transmitted was a broadcast packet.

TXMC **Transmit Multicast Packet**

The TXMC bit is set to 1 to indicate the last Ethernet packet transmitted was a multicast packet.

TXAL **Transmit Abort Late Collision**

The TXAL bit is set to 1 to indicate the last Ethernet packet was not transmitted successfully. Transmission of the packet was aborted due to a late collision problem.

TXAED **Transmit Abort Excessive Deferral**

The TXAED bit is set to 1 to indicate the last Ethernet packet was not transmitted successfully. Transmission of the packet was aborted due to excessive deferral. Excessive deferral indicates there was carrier on the Ethernet channel for a long period of time making it impossible for the transmitter to attempt transmission.

TXAEC **Transmit Abort Excessive Collisions**

The TXAEC bit is set to 1 to indicate the last Ethernet packet was not transmitted successfully. Transmission of the packet was aborted due to an excessive number of collisions. The maximum number of collisions allowed is determined by the RETRY field on the Collision Window / Collision Retry Register.

TXAUR **Transmit Abort Under Run**

The TXAUR bit is set to 1 to indicate the last Ethernet packet was not transmitted successfully. Transmission of the packet was aborted due to a FIFO under run condition. A FIFO under run condition indicates the DMA controller was unable to fill the FIFO at a fast enough rate as compared to the rate of transmission on the Ethernet medium. This typically means the DMA controller was not configured for Bursting or the memory peripheral device was not configured for bursting or the memory peripheral device is too slow to support the Ethernet interface.

TXAJ **Transmit Abort Jumbo**

The TXAJ bit is set to 1 to indicate the last Ethernet packet was not transmitted successfully. Transmission of the packet was aborted due to a jumbo condition. The jumbo condition means the packet was too large, larger than 1518 bytes. Packets larger than 1518 bytes will not be transmitted successfully unless the HUGEN bit is set in the MAC Configuration Register.

TXDEF **Transmit Packet Deferred**

The TXDEF bit is set to 1 to indicate the last Ethernet packet transmitted successfully encountered a deferral. TXDEF simply means the packet transmission was delayed because the Ethernet medium was busy at the time of first transmission attempt.

TXCRC **Transmit Packet CRC Error**

The TXCRC bit is set to 1 to indicate the last Ethernet packet transmitted successfully had an embedded CRC error. This condition only occurs when the CRCEN bit in the MAC Configuration Register is set to 0. When CRCEN is set to 0, the MAC does not insert a CRC. Instead, the MAC expects a precompiled CRC to be contained in the last four bytes of the Ethernet packet. If the precompiled CRC is found to be incorrect by the MAC, the MAC sets the TXCRC bit in the Transmit Status Register.

TXLCOL **Transmit Packet Late Collision**

The TXLCOL bit is set to 1 to indicate the MAC encountered a Late Collision while attempting to transmit the Ethernet packet. TXLCOL does not imply the packet was not transmitted, only that a late collision event had occurred. The TXAL bit will be set if the packet transmission was aborted. The MAC attempts to retransmit the packet when the RETRYL bit in the MAC Configuration Register is set to 1.

TXCOLC **Transmit Packet Collision Count**

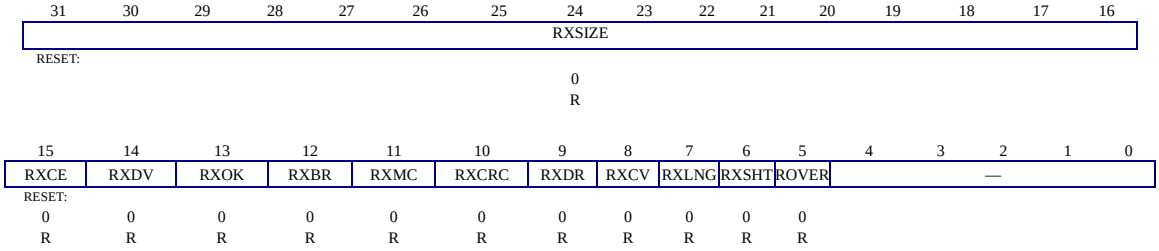
The TXCOLC field indicates how many collisions the MAC encountered while it was attempting to transmit the packet. TXCOLC does not imply the packet was not transmitted; only that collision events had occurred. The TXAEC bit is set if the packet transmission was aborted. The MAC attempts to retransmit the packet up to

the maximum number of collision retries defined by the RETRY field in the Collision Window/Collision Retry Register.

5.3.5 Ethernet Receive Status Register

The receive status register is a 32-bit register that contains the status for the last completed receive buffer. The receive buffer complete bit is set in the general status register whenever a receive frame is completed and the receive status register is loaded. The lower 16 bits of the transmit status register are also loaded into the StatusOrIndex field of the DMA buffer descriptor (when using DMA mode).

Address = FF80 0014



RXSIZE **Receive Packet Size**

The RXSIZE field provides the number of bytes contained in the next packet to be read from the Ethernet receive FIFO. When using DMA mode, the RXSIZE field is also copied to the Buffer Length field in the DMA Buffer Descriptor.

RXCE **Receive Carrier Event**

The RXCE bit is set to 1 to indicate the MAC received a carrier event from the Ethernet PHY since the last time a receive packet event was recorded in this status register. The carrier event is not associated with this packet. A carrier event is activity on the receive channel that does not result in a packet receive attempt being made.

RXDV **Receive Data Event**

The RXDV bit is set to 1 to indicate that at some time since the last recorded receive packet event, a receive data event was detected, noted and reported with this receive packet event. The RXDV event is not associated with this packet. An RXDV event is an RXDV occurring without a valid preamble and Start Frame Delimeter (SFD I) in the data stream.

RXOK **Receive Packet OK**

The RXOK bit is set to 1 to indicate the next packet in the receive FIFO has been successfully received without any errors.

RXBR **Receive Broadcast Packet**

The RXBR bit is set to 1 to indicate the next packet in the receive FIFO is a broadcast packet.

RXMC **Receive Multicast Packet**

The RXMC bit is set to 1 to indicate the next packet in the receive FIFO is a multicast packet.

RXCRC **Receive CRC Error**

The RXCRC bit is set to 1 to indicate the next packet in the receive FIFO was received with a CRC error.

RXDR **Receive Dribble Nibble**

The RXDR bit is set to 1 to indicate that at the end of the next packet in the receive FIFO an additional 1 to 7 bits of data were received. Even though this received packet has a dribble nibble, it is still considered valid provided the RXCRC bit is not set.

RXCV **Receive Code Violation**

The RXCV bit is set to 1 to indicate that the Ethernet PHY asserted the RXER signal during the data phase of this frame. RXCV indicates that the PHY encountered invalid receive codes while receiving the data.

RXLNG **Receive Long Packet**

The RXLNG bit is set to 1 to indicate that the next packet in the receive FIFO is longer than 1518 bits. This can only happen when the ERXLNG bit is set in the Ethernet General Control Register.

RXSHT **Receive Short Packet**

The RXSHT bit is set to 1 to indicate that the next packet in the receive FIFO is smaller than 64 bytes. This can only happen when the ERXSHT bit is set in the Ethernet General Control Register.

ROVER **Receive Over Run**

The ROVER bit is set to 1 to indicate that a receive FIFO over run condition has occurred. An over run condition happens when the FIFO becomes full while receiving an Ethernet packet. The FIFO over run condition typically indicates the DMA controller was unable to empty the FIFO at a fast enough rate as compared to the rate of receiving information from the Ethernet medium. This typically means the DMA controller was not configured for Bursting or the memory peripheral device was

not configured for bursting or the memory peripheral device is too slow to support the Ethernet interface.

5.3.6 MAC Configuration Register

The MAC configuration register is a 32-bit register that contains various MAC configuration options. All bits initialize to 0 on reset.

Address = FF80 0400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RESET:															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRST	INTLB	TSTMD		—	—	—	—	—	—	NOBO	RTRYL	CRCEN	PADEN	FULLD	HUGEN
RESET:															
0	0	0								0					
R/W	R/W	R/W								R/W					

SRST **MAC Soft Reset**

The SRST bit is set to 1 to “soft reset” the MAC module. The SRST bit has been know to be unreliable. The standard NET+ARM hardware and software resets all reset the MAC; as such, a soft reset of the MAC is not required.

INTLB **Internal Loop Back**

The INTLB bit is used to place the MAC module in internal loopback mode. This configuration is useful for diagnostic testing of the MAC module. Note that when the INTLB is set, the LB bit in the Ethernet General Control Register must also be set to 1.

TSTMD **Transmit Test Mode**

00: Normal Operation

01: Reserved

10: Transmitter Test Mode

11: Receive Test Mode

The TSTMD field is used for simulation of the MAC module. This field must be set to 00 for normal functional operation of the MAC Module.

NOBO **No Back-Off**

The NOBO bit can be set to 1 to disable the back-off feature. The back-off feature is used when a collision is detected. The back-off feature causes the MAC to wait a

random amount of time before attempting retransmission. When the NOBO bit is set, the MAC will immediately attempt a retransmission after the jamming period is complete.

RTRYL Retry Late Collision

The RTRYL bit can be set to 1 to enable the MAC to attempt a retransmission when a Late Collision is detected. When RTRYL is set to 0 and a Late Collision is detected, the transmission will be aborted with the TXAL bit set in the Transmit Status Register.

CRCEN Append CRC

The CRCEN bit can be set to 1 to enable the MAC to automatically append a calculated 4-byte CRC checksum at the end of the Ethernet Data Frame.

When CRCEN is set to 0, the MAC expects the last four bytes of the Ethernet Data Frame to contain a precompiled valid CRC value. The MAC will verify that the precompiled 4-byte CRC value is correct. If the MAC finds an incorrect CRC value, the MAC will report the condition by setting the TXCRC bit in the Transmit Status Register.

PADEN Auto Pad Enable

The PADEN bit can be set to 1 to force the MAC to automatically pad fill all Ethernet packets that are smaller than 64 bytes. The MAC will automatically insert zero byte padding at the end of the Ethernet data frame in order to make the Ethernet packet the minimum required 64 bytes in size.

FULLD Full Duplex Enable

The FULLD bit must be set to 1 to enable Full Duplex Ethernet operation. To operate in true Ethernet Full Duplex mode, firmware must set this bit to 1 after determining that the PHY is operating in Full Duplex mode. When the FULLD bit is set to 1, the MAC transmitter completely ignores any and all collision indication from the TXCOL input pin.

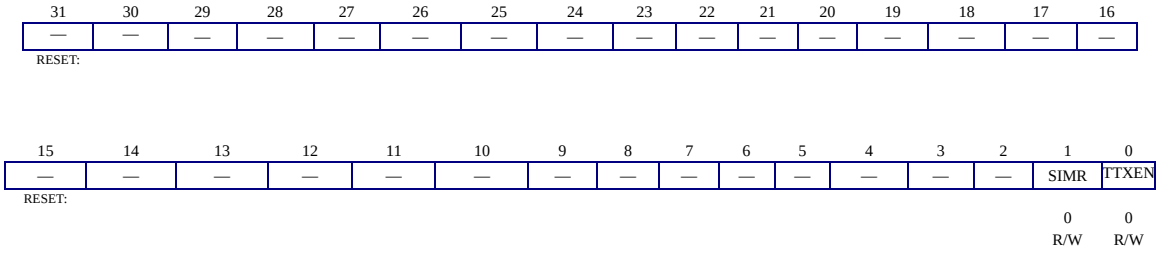
HUGEN Enable Huge Frames

The HUGEN bit can be set to 1 to allow the MAC to transmit Ethernet packets that are larger than 1518 bytes in size. When HUGEN is set to 1, the MAC will transmit unlimited Ethernet packets of any size. When HUGEN is set to 0, the MAC will abort transmission of all packets that are larger than 1518 bytes and will indicate this condition by setting the TXAJ bit in the Transmit Status Register.

5.3.7 MAC Test Register

This 32-bit register is only used during hardware simulation and never under normal operation conditions. All bits initialize to 0 on reset.

Address = FF80 040C



SIMR **Simulation Reset**

The SIMR field is only used for simulation of the MAC module. This field must be set to 0 for normal functional operation of the MAC Module.

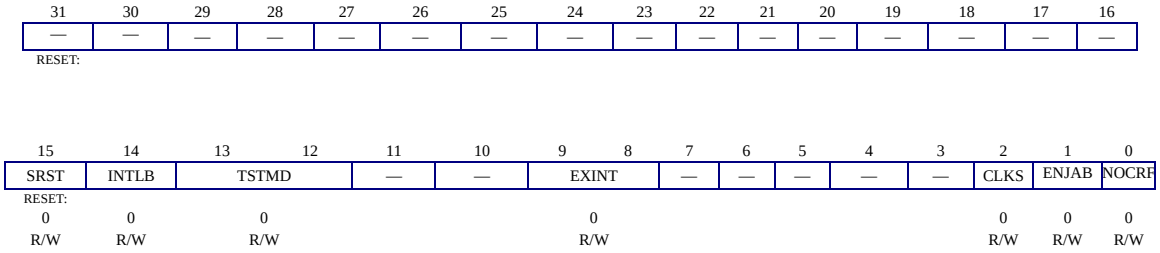
TTXEN **Test Transmit Enable**

The TTXEN field is only used for simulation of the MAC module. This field must be set to 0 for normal functional operation of the MAC Module.

5.3.8 PCS Configuration Register

The Physical Coding Sublayer (PCS) configuration register is a 32-bit register that contains various PCS configuration options. All bits initialize to 0 on reset.

Address = FF80 0408



SRST **Soft Reset**

The SRST bit is set to 1 to “soft reset” the PCS module. The SRST bit has been known to be unreliable. The standard NET+ARM hardware and software resets all reset the PCS; as such, a soft reset of the PCS is not required.

INTLB **Internal Loop Back**

The INTLB bit is used to place the PCS module in internal loopback mode. This configuration is useful for diagnostic testing of the PCS module. Note that when the INTLB is set, the LB bit in the Ethernet General Control Register must also be set to 1.

TSTMD **Test Mode**

00: Normal Operation

01: Reserved

10: PE100X Test Mode

11: PE10T Test Mode

The TSTMD field is used for simulation of the PCS module. This field must be set to 00 for normal functional operation of the PCS Module.

EXINT **External Interface**

00: MII Normal Operation

01: TP-PMD Mode

10: 10Mbit ENDEC Mode

The 00 MII Normal Operation mode is used for all modern MII style 10/100 PHY devices. The TP-PMD mode is used for older PHY that contain their own non standard Physical Coding Sub layer (PCS) circuitry. The ENDEC mode configuration is used for the older 10Mbit only PHY devices that pre-date the MII interface standard.

CLKS **Clock Select**

0: 25 MHz

1: 33 MHz

The CLKS field is used by the MAC to select the divide-down-ratio for the SYS_CLK system clock in order to produce the Management Data Clock (MDC). Simply choose a value for CLKS that is closest to the frequency of operation for SYS_CLK. The SYS_CLK frequency is described in *Chapter 11*.

ENJAB **Enable Jabber Protection**

Setting the ENJAB bit to 1 enables the Jabber Protection logic within the PCS when the PCS is operating in ENDEC mode (see EXINT field). Jabber is the condition where a transmitter is stuck on for longer than 50mS preventing other stations from transmitting.

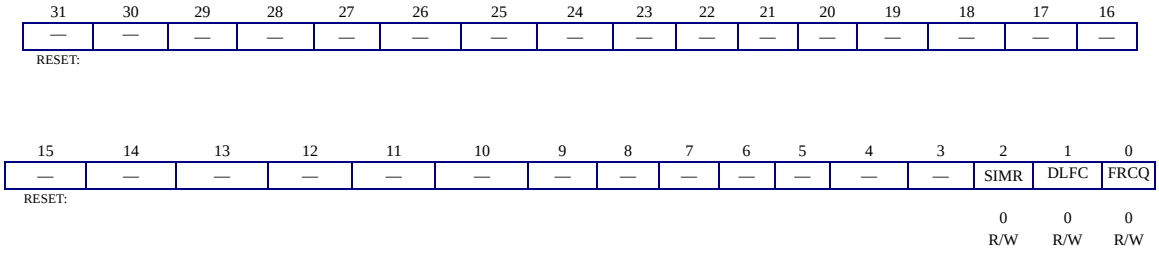
NOCFR **Disable Ciphering**

Setting the NOCFR bit to 1 disables the ciphering login within the PCS when the PCS is operating in TP-PMD mode (see EXINT field). The ciphering operation converts the 100Base-X Service Data Unit (SDU) into the Protocol Data Unit (PDU).

5.3.9 PCS Test Register

The PCS test register is a 32-bit register that is only used during hardware simulation and never under normal operation conditions. All bits initialize to 0 on reset.

Address = FF80 040C



SIMR **Simulation Reset**

The SIMR field is only used for simulation of the PCS module. This field must be set to 0 for normal functional operation of the PCS Module.

DLFC **Disable Link Fail Counter**

The DLFC field is only used for simulation of the PCS module. This field must be set to 0 for normal functional operation of the PCS Module.

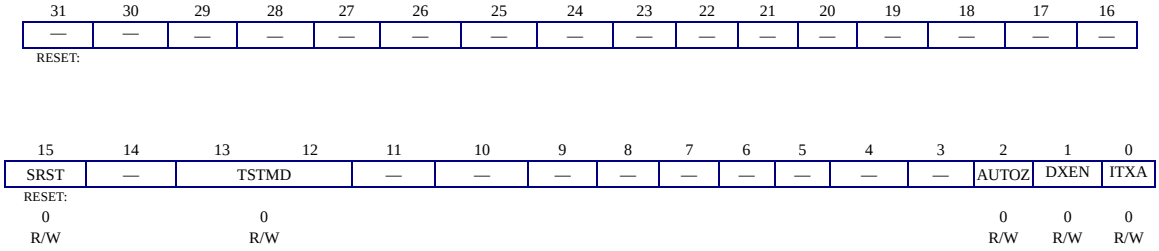
FRCQ **Force Quiets**

The FRCQ field is only used for simulation of the PCS module. This field must be set to 0 for normal functional operation of the PCS Module.

5.3.10 STL Configuration Register

The Station Logic (STL) configuration register is a 32-bit register that contains various STL configuration options. The STL contains both the Statistics (STAT) and Station Address Logic (SAL) blocks. All bits initialize to 0 on reset.

Address = FF80 0410



SRST **Soft Reset**

The SRST bit is set to 1 to “soft reset” the STL module. The SRST bit has been known to be unreliable. The standard NET+ARM hardware and software resets all reset the STL; as such, a soft reset of the STL is not required.

TSTMD **Test Mode**

00: Normal Operation

01: Reserved

10: Statistics Module Test Mode

11: Station Address Logic Test Mode

The TSTMD field is used for simulation of the STL module. This field must be set to 00 for normal functional operation of the STL Module.

AUTOZ **Auto Zero Statistics**

The AUTOZ bit can be set in order to cause the Statistics Registers (refer to Section *5.3.16 Statistics Monitoring*) to be automatically reset to 0 after reading. When the AUTOZ bit is set to 0, the Statistics Register must be manually reset to 0 by writing a 0 value to the register.

RXEN **Receiver Enable**

The RXEN bit must be set to 1 in order to Enable the Ethernet Receiver logic. The RXEN bit should be set to 1 after the fields in the Station Address Filter Register (refer to Section *5.3.12.7 Transmit Random Number Generator*) have been

configured. The Ethernet Receiver can be disabled at anytime by setting the RXEN bit to 0.

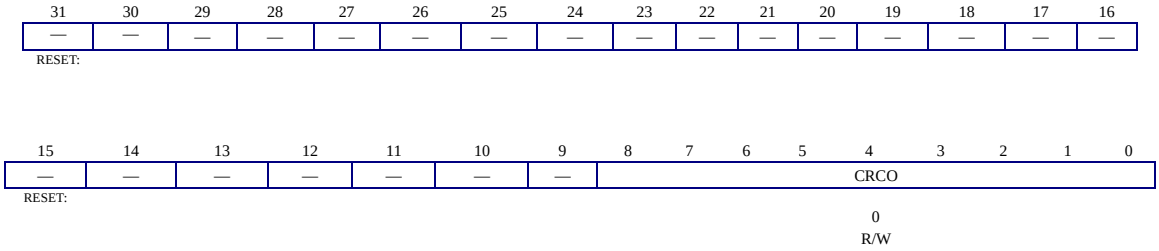
ITXA **Insert Transmit Source Address**

The ITXA bit can be set to 1 in order to force the MAC to automatically insert the Ethernet Source MAC Address into the Ethernet Transmit packet. The MAC Address information is provided by the data configured in the Station Address Register (refer to Section *5.3.12.8 Transmit Masked Random Number*). When the ITXA bit is set to 0, the 7^{th} through 12^{th} data bytes in the Ethernet packet will be ignored and replaced by the size bytes found in the MAC Address Register.

5.3.11 STL Test Register

The STL test register is a 32-bit register that is only used during hardware simulation and never under normal operation conditions. All bits initialize to 0 on reset.

Address = FF80 0414



CRCO

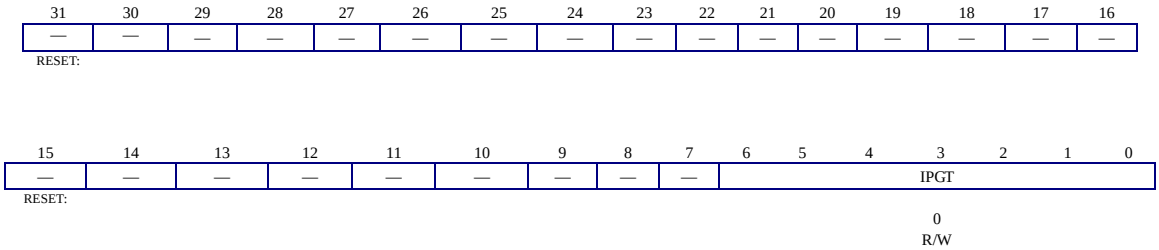
The CRCO field is only used for simulation of the STL module. This field must be set to 0 for normal functional operation of the STL Module.

5.3.12 Transmit Control Registers

There are 10 transmit control registers. Each register is a 32-bit register. All bits are initialized to 0 on reset (unless otherwise noted).

5.3.12.1 Back-to-Back IPG Gap Timer Register

Address = FF80 0440



The Inter-Packet Gap (IPG) is the measurement between the last nibble (or bit in ENDEC mode) of Frame Check Sequence (FCS) and the first nibble (or bit in ENDEC mode) of the preamble for the next packet.

For 100MB Ethernet, the IEEE 802.2u standard requires IPG to be a minimum of 0.96 microseconds.

For 10MB Ethernet, the IEEE 802.2u standard requires IPG to be a minimum of 9.6 microseconds.

In the Ethernet Transmit block, there are three fields that can be used to fine tune the IPG. The fields are IPGT, IPGR1, and IPGR2.

IPGT **Back-to-Back Transmit IPG**

The IPGT field is used to space back to back transmit packets. The Ethernet Transmit state machine has an intrinsic delay of 4 Ethernet clocks between packets. With the IPGT field set to 0, the IPG will be a minimum of 4 Ethernet clocks, or 0.16 uS when operating at 100BT.

The IEEE 802.3u standard requires an IPG value of 9.6 uS. The recommended value for IPGT when operating in MII mode is 0x14.

The following table provides examples of IPGT configuration for 100BT operating in MII Mode.

100MB IPGT	802.3 IPG
0x00	0.16 uS
0x03	0.28 uS
0x07	0.44 uS
0x0B	0.60 uS
0x0F	0.76 uS
0x14	0.96 uS
0x1F	1.40 uS

Table 5-4: MII Mode 100MB IPGT Values

The following table provides examples of IPGT configuration for 10BT operating in MII Mode.

100MB IPGT	802.3 IPG
0x00	1.6 uS
0x03	2.8 uS
0x07	4.4 uS
0x0B	6.0 uS
0x0F	7.6 uS
0x14	9.6 uS
0x1F	14.0 uS

Table 5-5: MII Mode 10MB IPGT Values

When operating in ENDEC mode, the IPGT values are different than MII mode since the ENDEC transmit clock is 10MHz and not 25MHz. As before, there is a 4 clock intrinsic delay in the transmitter state machine.

The IEEE 802.3u standard requires an IPG value of 9.6 uS. The recommended value for IPGT when operating in ENDEC mode is 0x5C.

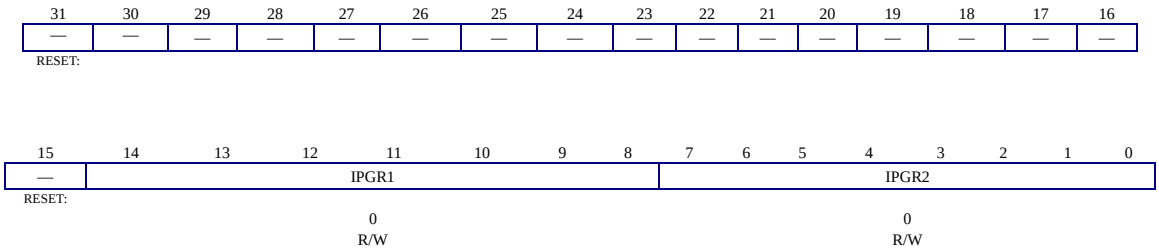
The following table provides examples of IPGT configuration for 10BT operating in ENDEC Mode.

100MB IPGT	802.3 IPG
0x07	1.1 uS
0x0F	1.9 uS
0x3F	6.7 uS
0x5C	9.6 uS
0x5F	9.9 uS

Table 5-6: ENDEC Mode 10MB IPGT Values

5.3.12.2 Non Back-to-Back IPG Gap Timer Register

Address = FF80 0444



IPGR1 **Non Back-to-Back IPG Length Part I**

The IPGR1 and IPGR2 timers begin their interval timing at the same time. IPGR1 is the interval in which the transmitter will defer if carrier sense goes high. It is usually set to 2/3 of the IPGR2 value. During the period of time after IPGR1 but before IPGR2 times out, the transmitter will not defer if it sees carrier sense and will cause a

collision if it has a packet to send resulting in activation of the Random Back off fairness algorithm.

The following table provides examples of IPGR1 configuration for 100BT operating in MII Mode.

100MB IPGR1	802.3 IPG
0x00	0.28 uS
0x06	0.52 uS
0x07	0.56 uS
0x08	0.60 uS
0x09	0.64 uS
0x1F	1.52 uS

Table 5-7: MII Mode 100 MB IPGR1 Values

The following table provides examples of IPGR1 configuration for 10BT operating in MII Mode.

100MB IPGR1	802.3 IPG
0x00	2.8 uS
0x06	5.2 uS
0x07	5.6 uS
0x08	6.0 uS
0x09	6.4 uS
0x1F	15.2 uS

Table 5-8: MII Mode 10MB IPGR1 Values

When operating in ENDEC mode, the IPGR1 values are different than MII mode since the ENDEC transmit clock is 10MHz and not 25MHz. As before, there is a 4 clock intrinsic delay in the transmitter state machine.

The IEEE 802.3u standard recommends an IPGR1 value of 6.4 uS. The recommended value for IPGR1 when operating in ENDEC mode is 0x5C.

The following table provides examples of IPGR1 configuration for 10BT operating in ENDEC Mode.

100MB IPGR1	802.3 IPG
0x07	1.4 uS
0x0F	2.2 uS
0x2F	5.4 uS
0x39	6.4 uS
0x5F	9.9 uS

Table 5-9: ENDEC Mode 10MB IPGR1 Values

IPGR2 Non Back-to-Back IPG Length Part II

The IPGR2 field is used to space non back-to-back transmit packets. For non back-to-back packets, the CRS input signal is sampled. The CRS sampling time plus the Ethernet Transmit state machine delay of 3 Ethernet clocks between packets results in a total intrinsic delay of 7 Ethernet clocks when calculating IPGR2. With the IPGR2 field set to 0, the IPG will be a minimum of 7 Ethernet clocks, or 0.24 uS when operating at 100BT.

The IEEE 802.3u standard requires an IPG value of 9.6 uS. The recommended value for IPGR2 when operating in MII mode is 0x10.

The following table provides examples of IPGR2 configuration for 100BT operating in MII Mode.

100MB IPGR2	802.3 IPG
0x00	0.28 uS
0x03	0.40 uS
0x07	0.56 uS
0x0B	0.72 uS
0x0F	0.88 uS
0x11	0.96 uS
0x1F	1.52 uS

Table 5-10: MII Mode 100MB IPGR2 Values

The following table provides examples of IPGR2 configuration for 10BT operating in MII Mode.

100MB IPGR2	802.3 IPG
0x00	2.8 uS
0x03	4.0 uS
0x07	5.6 uS
0x0B	7.2 uS
0x0F	8.8 uS
0x11	9.6 uS
0x1F	15.2 uS

Table 5-11: MII Mode 10MB IPGR2 Values

When operating in ENDEC mode, the IPGR2 values are different than MII mode since the ENDEC transmit clock is 10MHz and not 25MHz. As before, there is a 7 Ethernet clock intrinsic delay in the transmitter state machine.

The IEEE 802.3u standard requires an IPG value of 9.6 uS. The recommended value for IPGR2 when operating in ENDEC mode is 0x59.

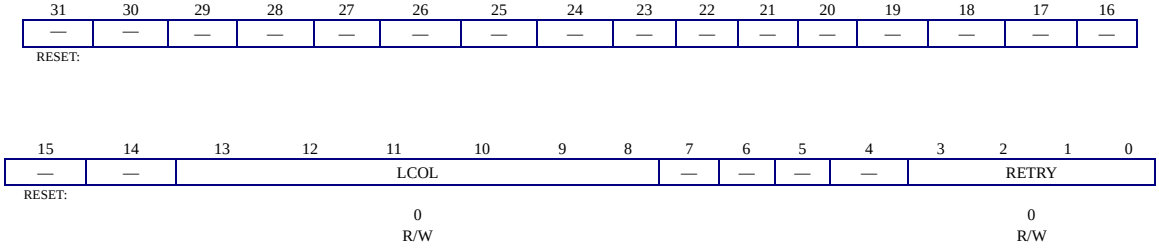
The following table provides examples of IPGR2 configuration for 10BT operating in ENDEC Mode.

100MB IPGR2	802.3 IPG
0x07	1.4 uS
0x0F	2.2 uS
0x3F	6.9 uS
0x59	9.6 uS
0x5F	10.2 uS

Table 5-12: ENDEC Mode 10MB IPGR2 Values

5.3.12.3 Collision Window / Collision Retry Register

Address = FF80 0448



LCOL **Late Collision Window**

The LCOL field defines the number of bytes that are within the nominal slot time or collision window for properly configured networks. This value defines the number of bytes after the SFD that will allow a normal collision to occur in the event of a collision. Any collisions after the time defined by LCOL will result in a LATE

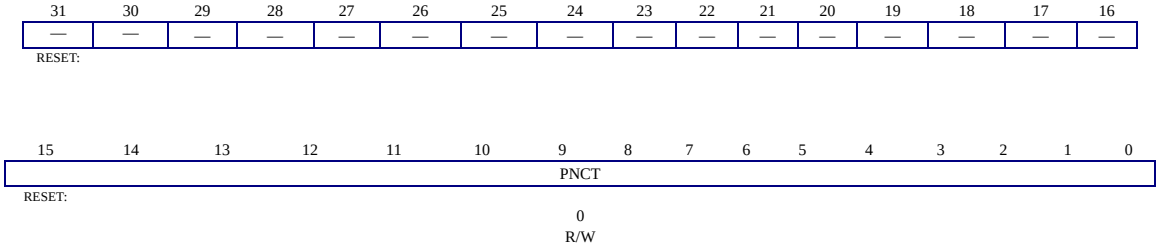
collision event. The recommended value for LCOL is 0x37 (55d), which results in a bit-time window of 512 bits when considering the 8-byte preamble/SFD.

RETRY **Maximum Number of Retries after a Collision**

The RETRY field specifies the number of retransmission attempts following a collision before aborting the packet due to excessive collisions. The 802.3u Standard specifies the attemptLimit to be 0xF (15d); as such, 0xF is the recommended value for this field.

5.3.12.4 Transmit Packet Nibble Counter

Address = FF80 0460

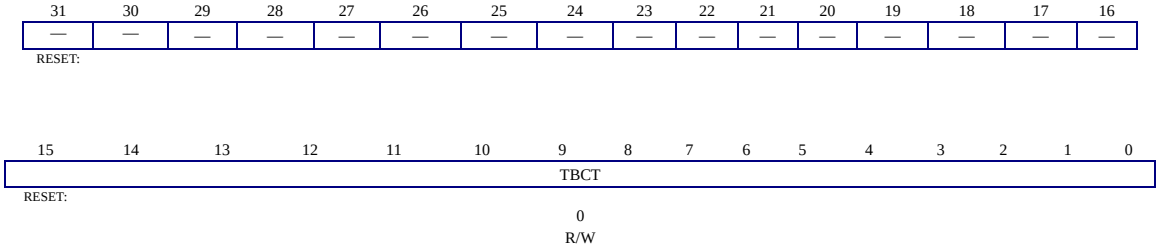


PNCT **Transmit Packet Nibble Counter**

The PNCT field is used for simulation of the Transmitter module. This field should be untouched for normal functional operation of the MAC Transmitter Module.

5.3.12.5 Transmit Byte Counter Register

Address = FF80 0464

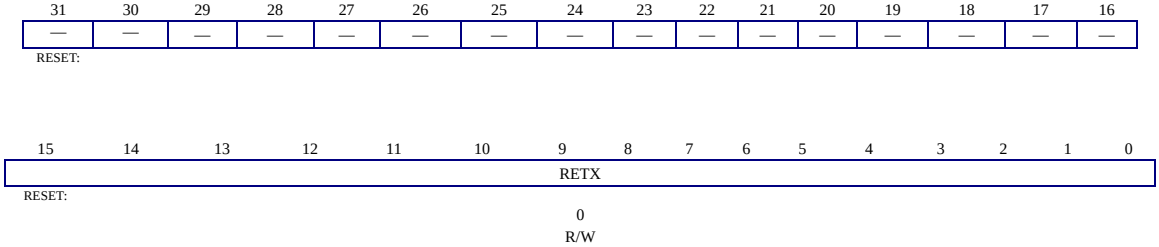


TBCT **Transmit Byte Counter Register**

The TBCT field is used for simulation of the Transmitter module. This field should be untouched for normal functional operation of the MAC Transmitter Module.

5.3.12.6 Retransmit Byte Counter Register

Address = FF80 0468

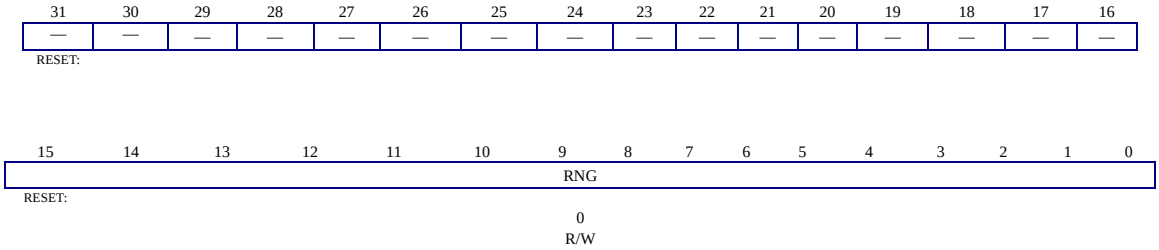


RETX **Retransmit Byte Counter Register**

The RETX field is used for simulation of the Transmitter module. This field should be untouched for normal functional operation of the MAC Transmitter Module.

5.3.12.7 Transmit Random Number Generator

Address = FF80 046C

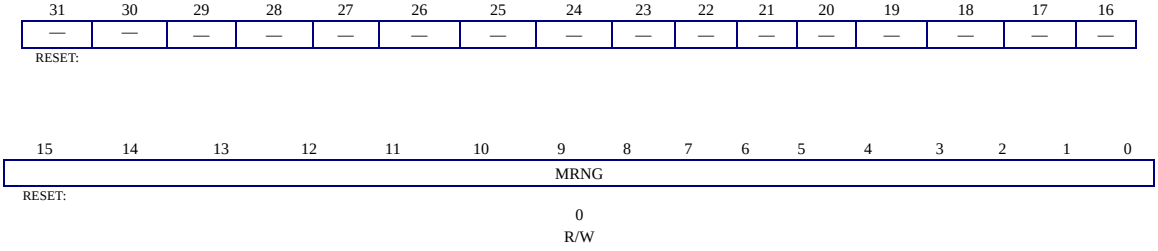


RNG **Transmit Random Number Generator**

The RNG field is used for simulation of the Transmitter module. This field should be untouched for normal functional operation of the MAC Transmitter Module.

5.3.12.8 Transmit Masked Random Number

Address = FF80 0470

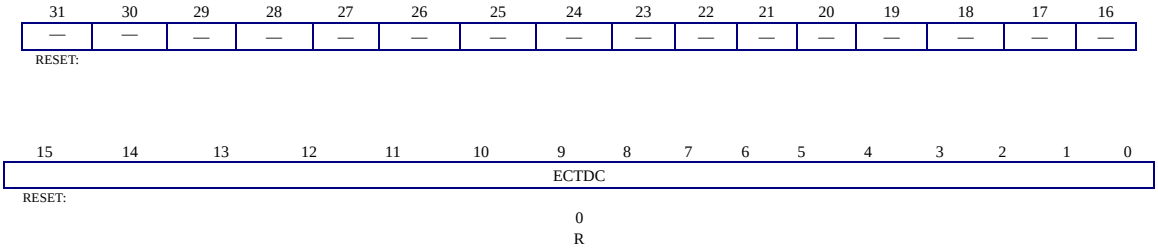


MRNG **Transmit Masked Random Number Generator**

The MRNG field is used for simulation of the Transmitter module. This field should be untouched for normal functional operation of the MAC Transmitter Module.

5.3.12.9 Transmit Counter Decodes

Address = FF80 0474

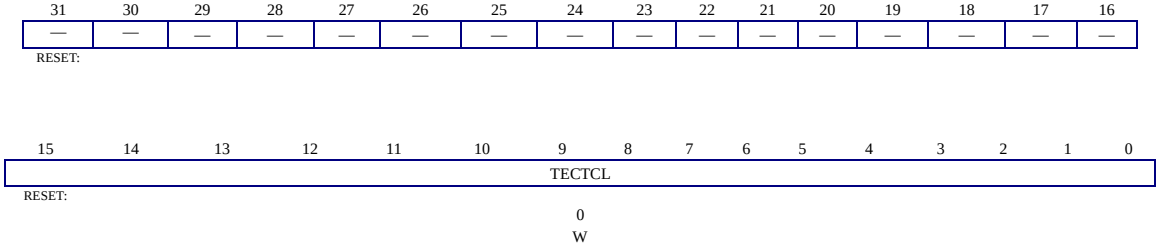


ECTDC **Transmit Counter Decodes**

The ECTDC field is used for simulation of the Transmitter module. This field should be untouched for normal functional operation of the MAC Transmitter Module.

5.3.12.10 Test Operate Transmit Counters

Address = FF80 0478



TECTCL **Test Operate Transmit Counters**

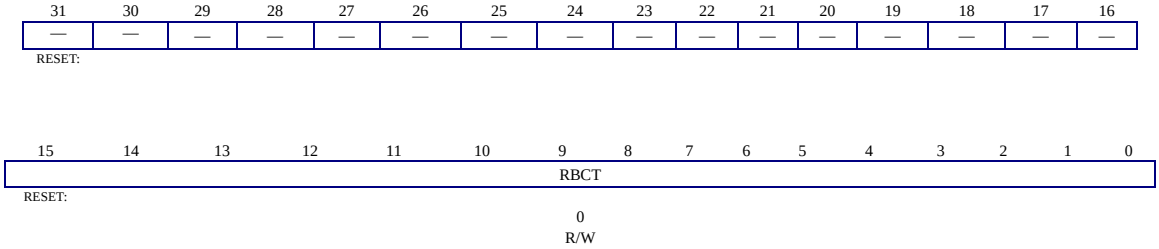
The TECTDL field is used for simulation of the Transmitter module. This field should be untouched for normal functional operation of the MAC Transmitter Module.

5.3.13 Receive Control Registers

There are 3 receive control registers. Each register is a 32-bit register. All bits are initialized to 0 on reset.

5.3.13.1 Receive Byte Counter

Address = FF80 04A0

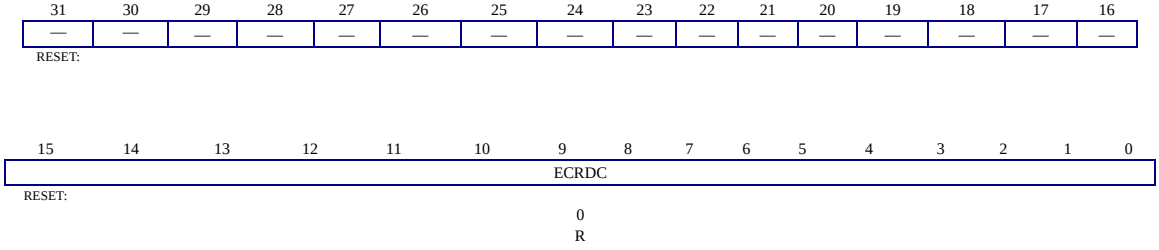


RBCT **Receive Byte Counter**

The RBCT field is used for simulation of the Receiver module. This field should be untouched for normal functional operation of the MAC Receiver Module.

5.3.13.2 Receive Counter Decodes

Address = FF80 0488

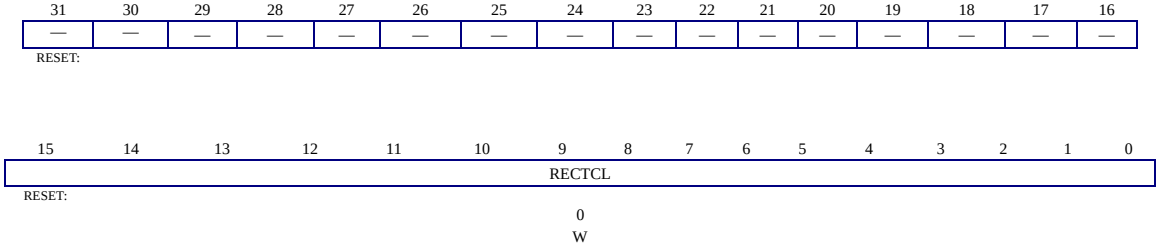


ECRDC **Receiver Counter Decodes**

The ECRDC field is used for simulation of the Receiver module. This field should be untouched for normal functional operation of the MAC Receiver Module.

5.3.13.3 Test Operate Receive Counters

Address = FF80 0484



RECTCL **Test Operate Receive Counters**

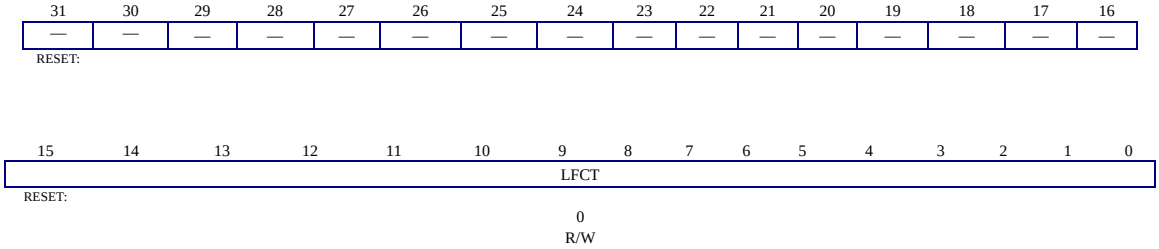
The RECTCL field is used for simulation of the Receiver module. This field should be untouched for normal functional operation of the MAC Receiver Module.

5.3.14 PCS Control Registers

There are 3 physical sublayer control registers. Each register is a 32-bit register. All bits are initialized to 0 on reset (unless otherwise noted).

5.3.14.1 Link Fail Counter

Address = FF80 04C0

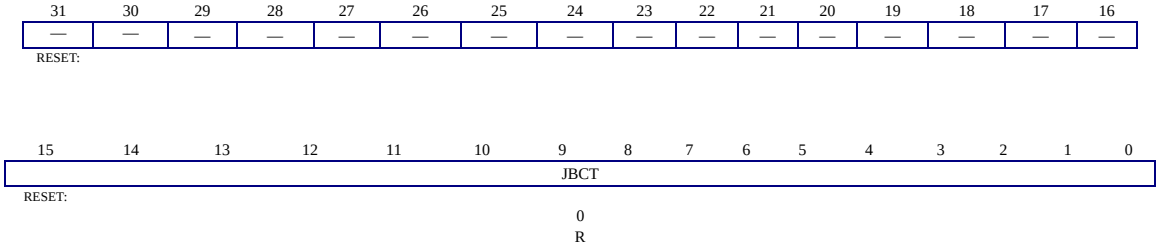


LFCT **Link Fail Counter**

The LFCT field is used for simulation of the PCS module. This field should be untouched for normal functional operation of the PCS Module.

5.3.14.2 10 MB Jabber Counter

Address = FF80 0500

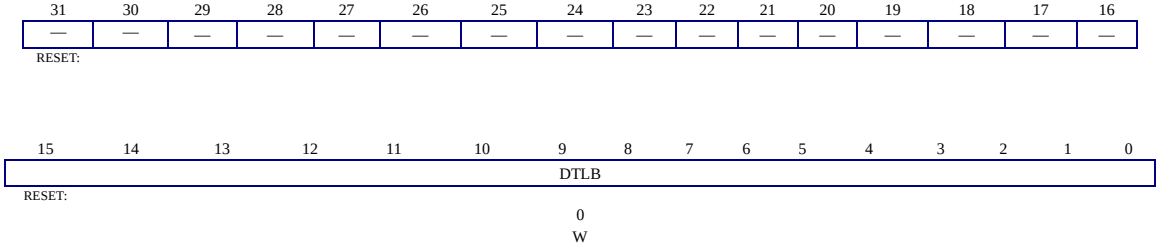


JBCT **10 MB Jabber Counter**

The JBCT field is used for simulation of the PCS module. This field should be untouched for normal functional operation of the PCS Module.

5.3.14.3 10 MB Loss of Carrier Counter

Address = FF80 0504



DTLB **10 MB Loss of Carrier Counter**

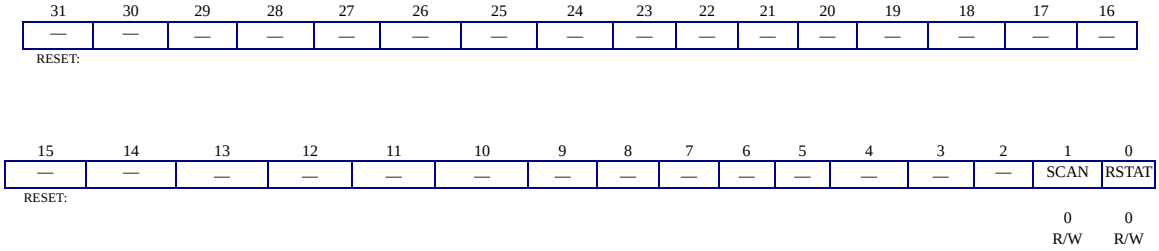
The DTLB field is used for simulation of the PCS module. This field should be untouched for normal functional operation of the PCS Module.

5.3.15 MII Control Registers

There are 5 media independent interface control registers. Each register is a 32-bit register. All bits are initialized to 0 on reset (unless otherwise noted).

5.3.15.1 MII Command Register

Address = FF80 0540



SCAN

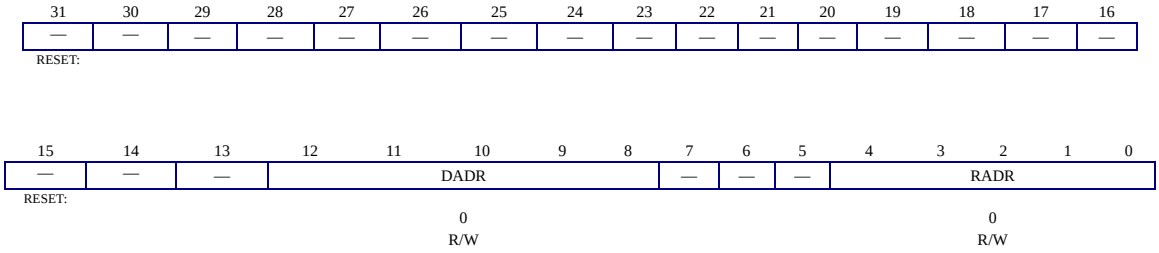
This bit is set to 1 to cause the MII Management module to perform Read cycles continuously. This is useful for monitoring Link Fail for example. The default value for this field is 0.

RSTAT

This bit is set to 1 to cause the MII Management module to perform a single Read cycle. The Read data is returned in the MII Read Data Register after the BUSY bit in the MII Indicators Register has returned to a value of 0. The default value for this field is 0.

5.3.15.2 MII Address Register

Address = FF80 044C



DADR

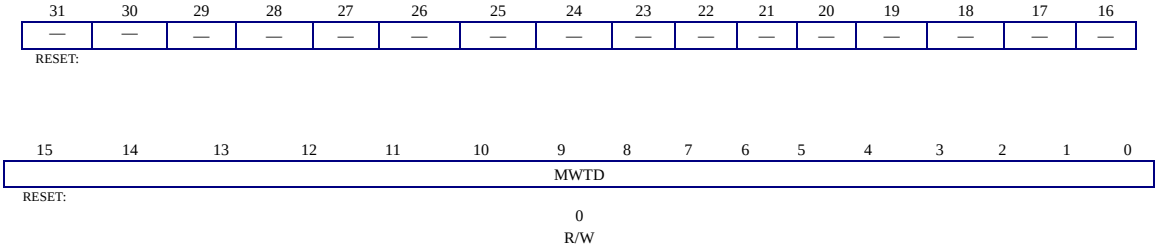
Device Address: This field represents the 5-bit PHY Device Address field for management cycles. Up to 31 different PHY devices can be addressed; address 0 is reserved. The default value for this field is 0.

RADR

Register Address: This field represents the 5-bit PHY Register Address field for management cycles. Up to 32 registers within a single PHY Device can be addressed. The default value for this field is 0.

5.3.15.3 MII Write Data Register

Address = FF80 044C

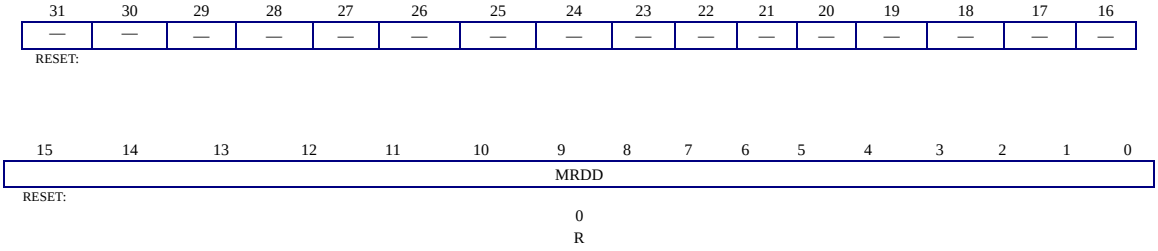


MWTD

Write Data: When this register is written an MII Management write cycle is performed using the 16-bit data the pre-configured PHY Device and PHY Register addresses defined in the PHY Address register. The write operation is completed when the BUSY bit in the MII Indicators Register returns to 0.

5.3.15.4 MII Read Data Register

Address = FF80 054C

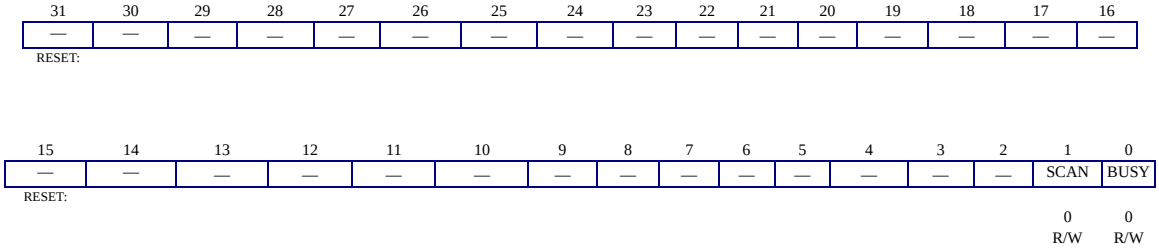


MRDD

Read Data: Following an MII Management read cycle, the read data is obtained by reading from this register. An MII Management read cycle is executed by loading the PHY Address Register then setting the READ bit in the MII Command Register to 1. Read data is available after the BUSY bit in the MII Indicators register returns to 0.

5.3.15.5 MII Indicators Register

Address = FF80 0550



SCAN

Scanning: A 1 in this bit position indicates continuous MII Management Scanning read operations are in progress.

BUSY

Busy: A 1 in this bit position indicates the MII Management module is currently performing an MII Management Read or Write Cycle. This bit will return to 0 when the operation is complete.

5.3.16 Statistics Monitoring

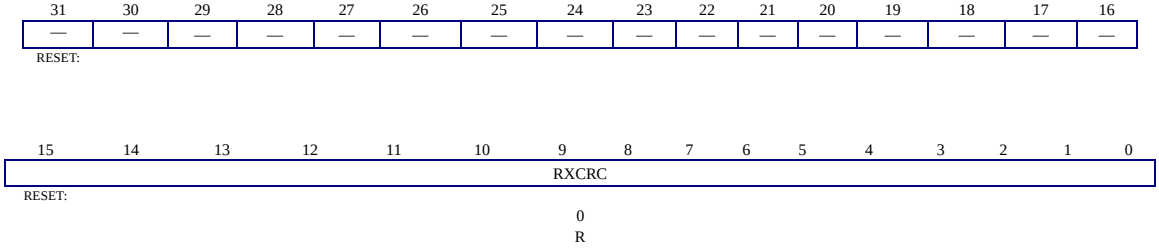
The statistics block (STAT) is responsible for monitoring the transmit and receive status vectors from the MAC. The status vectors identify transmit and receive packets that have encountered errors. The EFE module uses the STAT block to record those events. Firmware can read the statistics errors at any time. Furthermore, firmware can configure the STAT block to automatically clear the error counters when read via a bit in the STL configuration register.

The EFE module supports 8 different error statistics registers.

5.3.16.1 CRC Error Counter

The maximum collision counter is incremented each time a receive frame is discarded because it was received with a CRC error.

Address = FF80 0580



RXCRC

The RXCRC counter is incremented each time a packet is received with a CRC error. The packet is automatically discarded if the ERXBAD field in the General Control Register is set to 0. This counter will increment for each packet received with a bad CRC regardless of the value in the ERXBAD field. The counter is automatically cleared when read provided the AUTOZ bit is set in the STL Configuration Register.

5.3.16.2 Alignment Error Counter

The alignment error counter is incremented each time a receive frame is discarded because it was received with an alignment error.

Address = FF80 044C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RESET:															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXAEC															
RESET:															
0															
R															

RXAEC

The RXAEC counter is incremented each time a packet is received with an alignment (dribble bit) error. The packet is automatically discarded if the ERXBAD field in the General Control Register is set to 0. This counter will increment for each packet received with an alignment condition regardless of the value in the ERXBAD field. The counter is automatically cleared when read provided the AUTOZ bit is set in the STL Configuration Register.

5.3.16.3 Code Error Counter

The code error counter is incremented each time a receive frame is discarded because it was received with a coding error.

Address = FF80 0588

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RESET:															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXCEC															
RESET:															
0															
R															

RXCEC

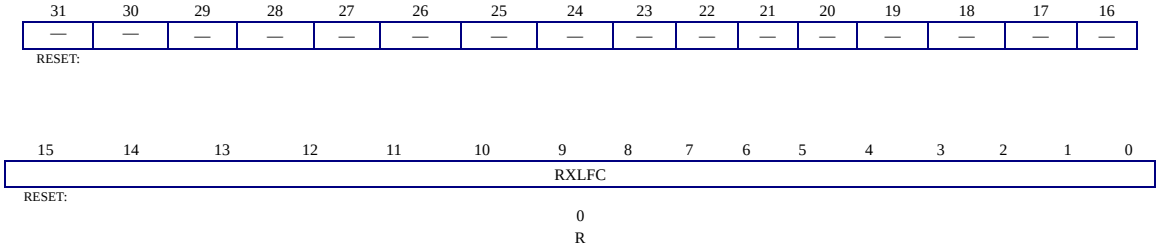
The RXCEC counter is incremented each time a packet is received with a coding error. A coding error occurs when the PHY asserts the RXER input to the NET+ARM

MAC. The packet is automatically discarded if the ERXBAD field in the General Control Register is set to 0. This counter will increment for each packet received with a coding error regardless of the value in the ERXBAD field. The counter is automatically cleared when read provided the AUTOZ bit is set in the STL Configuration Register.

5.3.16.4 Long Frame Counter

The long frame counter is incremented each time a receive frame is discarded because it was received with a length exceeding 1518 bytes.

Address = FF80 058C



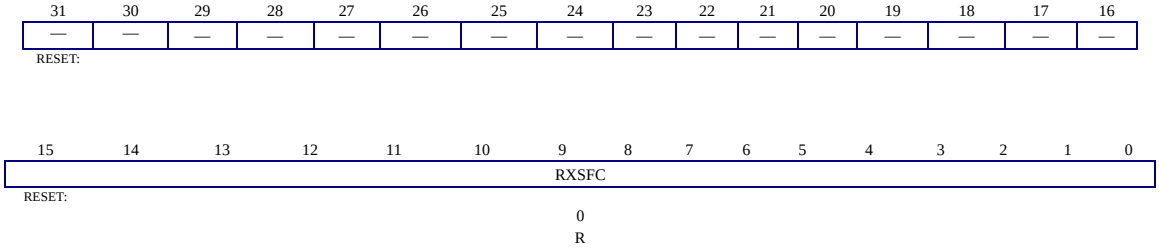
RXLFC

The RXLFC counter is incremented each time a packet is received that is larger than 1518 bytes. The packet is automatically discarded if the ERXLNG field in the General Control Register is set to 0. This counter will increment for each packet received that is larger than 1518 bytes regardless of the value in the ERXLNG field. The counter is automatically cleared when read, provided the AUTOZ bit is set in the STL Configuration Register.

5.3.16.5 Short Frame Counter

The short frame counter is incremented each time a receive frame is discarded because it was received with less than 64 bytes.

Address = FF80 0590



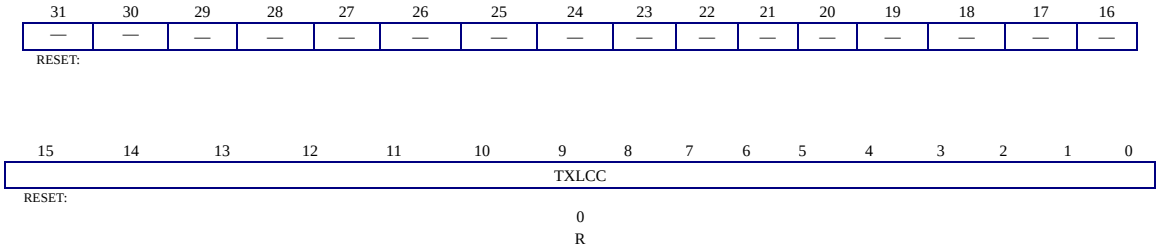
RXSFC

The RXSFC counter is incremented each time a packet is received that is shorter than 64 bytes. The packet is automatically discarded if the ERXSHT field in the General Control Register is set to 0. This counter will increment for each packet received that is shorter than 64 bytes regardless of the value in the ERXSHT field. The counter is automatically cleared when read provided the AUTOZ bit is set in the STL Configuration Register.

5.3.16.6 Late Collision Counter

The late collision counter is incremented each time a transmit frame is aborted because it receives a collision after transmitting at least 64 bytes of a packet.

Address = FF80 0594



TXLCC

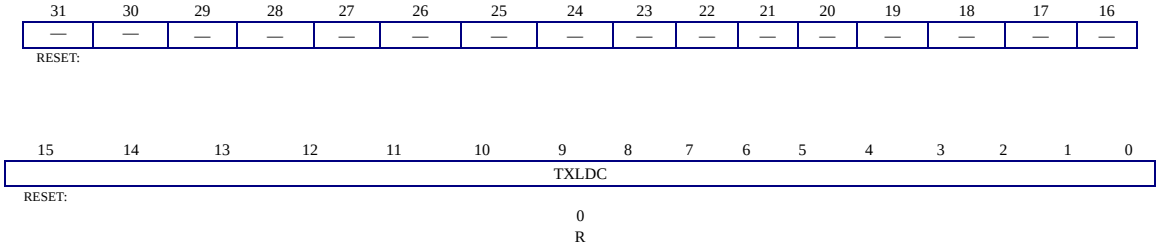
The TXLCC counter is incremented each time a packet transmission is aborted because it received a late collision. A late collision is defined as a collision that occurs

after the timeframe configured via the LCOL field in the Collision Window / Collision Retry Register. The counter is automatically cleared when read provided the AUTOZ bit is set in the STL Configuration Register.

5.3.16.7 Excessive Deferral Counter

The excessive deferral counter is incremented each time a transmit frame is aborted because it executed too many deferrals.

Address = FF80 0598



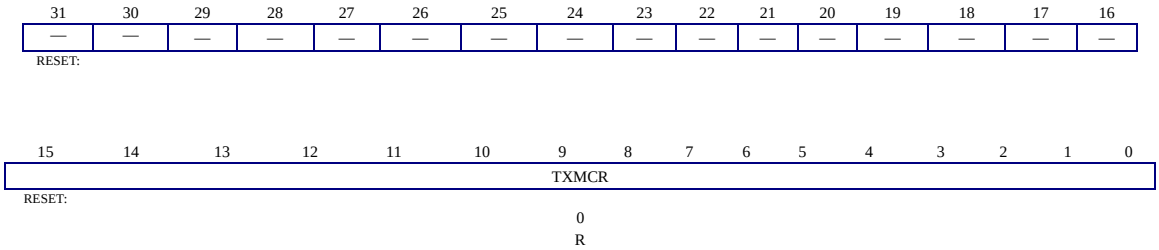
TXLDC

The TXLDC counter is incremented each time a packet transmission is aborted because it encountered excessive deferral. Excessive deferral occurs when carrier is detected on the Ethernet receive medium longer than expected. The counter is automatically cleared when read provided the AUTOZ bit is set in the STL Configuration Register.

5.3.16.8 Maximum Collision Counter

The maximum collision counter is incremented each time a transmit frame is aborted because it encountered too many collisions in its attempt to transmit.

Address = FF80 059C



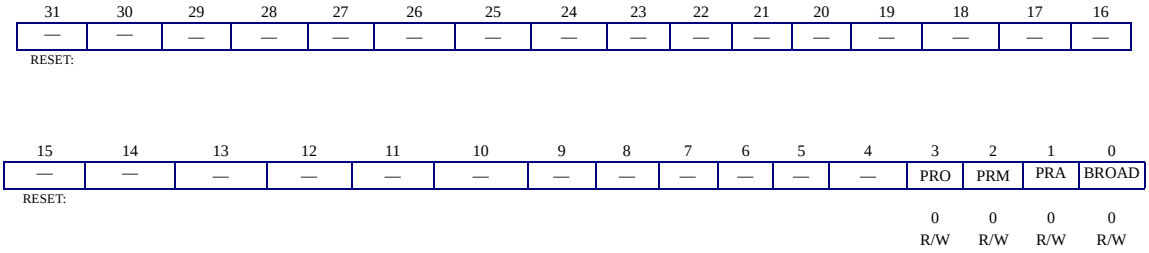
TXMCR

The TXMCR counter is incremented each time a packet transmission is aborted because it received too many collisions. The maximum number of allowable collisions is defined by the RETRY field in the Collision Window / Collision Retry Register. The counter is automatically cleared when read provided the AUTOZ bit is set in the STL Configuration Register.

5.3.17 Station Address Filter Register

The station address filter register is a 32-bit register containing various filter controls. The bits in this register initialize to 0 during reset.

Address = FF80 05C0



PRO

Setting the PRO bit to 1 causes the MAC to enter promiscuous mode. When operating in promiscuous mode, all packets are received without any filtering.

PRM

Setting the PRM bit to 1 causes the MAC to receive ALL Multicast Ethernet packets without any filtering.

PRA

Setting the PRA bit to 1 causes the MAC to receive only those Multicast Ethernet packets that are selected by the Multicast Hash Table. Please refer to Section *5.3.19 Multicast Hash Table* for more details.

BROAD

Setting the BROAD bit to 1 causes the MAC to receive ALL Broadcast Ethernet packets without any filtering.

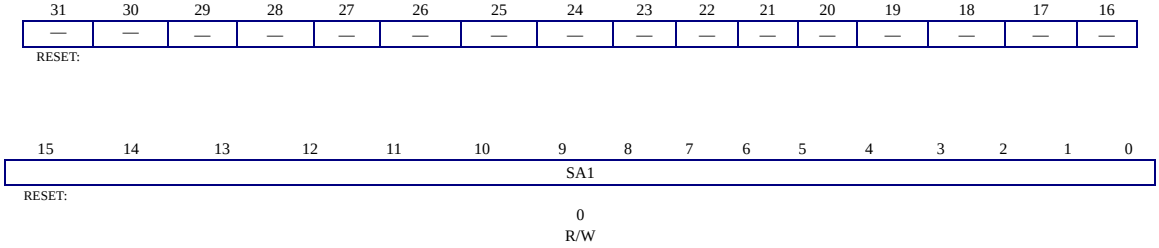
5.3.18 Station Address Register

The NET+ARM chip supports a single station address. The station address is used by the Station Address Logic (SAL) for address comparison of inbound receive packets.

The station address requires three 32-bit registers to hold the 48-bit value. Each register contains 16 bits of the 48-bit total. The first register (low address) stores bits 47:32, the second register stores bits 31:16, and the third register stores bits 15:00.

The station address bytes are loaded into the station address Register(s) in little endian format.

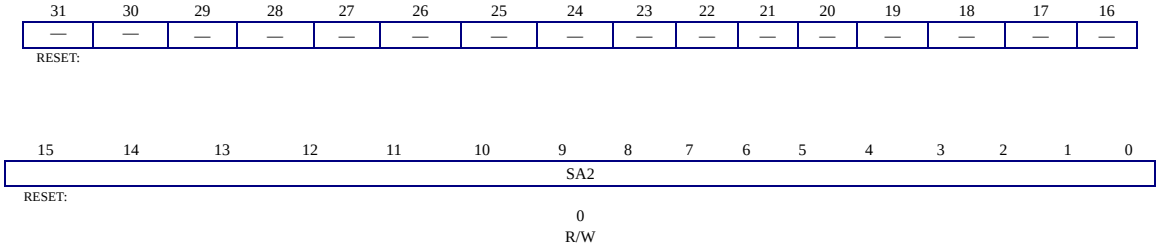
Address = FF80 05C4



SA1

The SA1 field provides the most significant (upper) 16 bits of the 48-bit Station Address value.

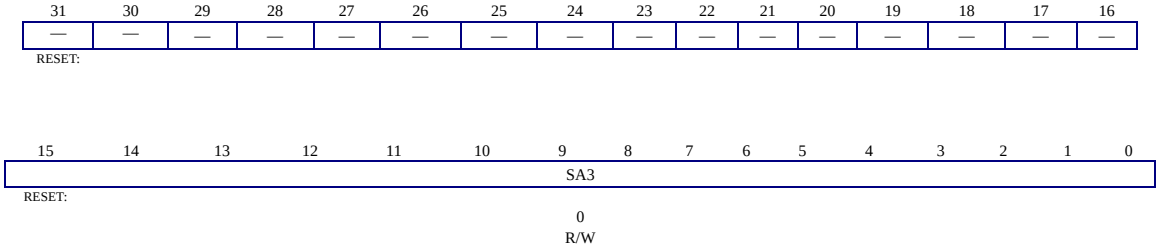
Address = FF80 05C8



SA2

The SA2 field provides the middle significant (middle) 16 bits of the 48-bit Station Address value.

Address = FF80 05CC



SA3

The SA3 field provides the least significant (lower) 16 bits of the 48-bit Station Address value.

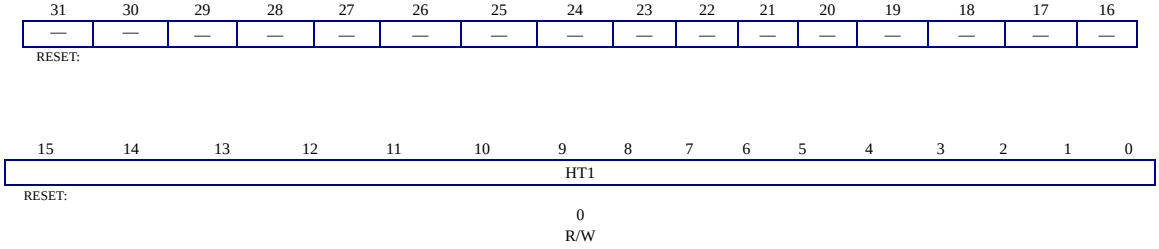
5.3.19 Multicast Hash Table

The MAC receiver provides the SAL logic with a 6-bit CRC value that addresses the 64-bit multicast hash table. If the current receive frame is a multicast frame and the 6-bit CRC addresses a bit in the hash table that is set to 1, then the receive packet is accepted, otherwise, it gets rejected. All register bits initialize to 0 on reset.

The 64-bit multicast hash table requires four 32-bit registers for storage. The first register (lower address) stores enables for the upper 32 CRC addresses. The other registers stores enables for the lower 32 CRC addresses. The CRC addresses are stored right to left in ascending order.

The multicast address bytes are loaded into the station address register(s) in little endian format.

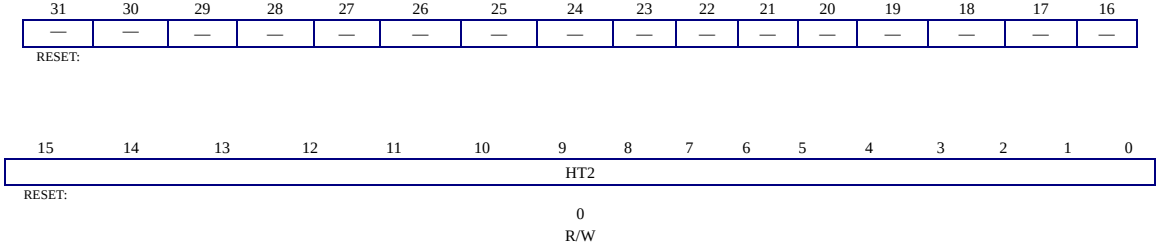
Address = FF80 05D0



HT1 – Hash Table Entries [15:0]

The HT1 field provides the least significant sixteen bits of the 64-bit hash table.

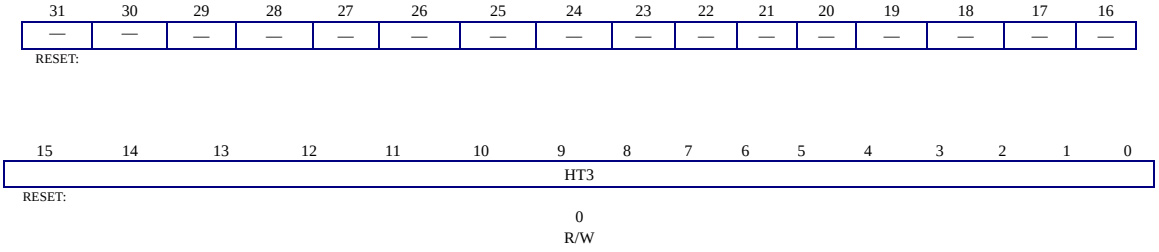
Address = FF80 05D4



HT2 – Hash Table Entries [31:16]

The HT2 field provides the lower middle significant sixteen bits of the 64-bit hash table.

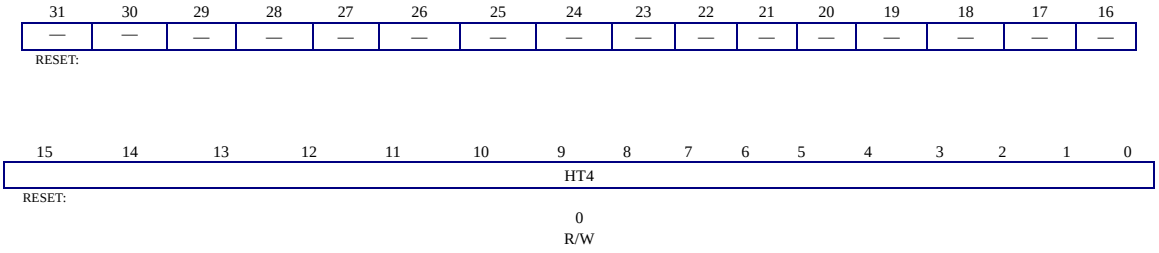
Address = FF80 05D8



HT3 – Hash Table Entries [47:32]

The HT3 field provides the upper middle significant sixteen bits of the 64-bit hash table.

Address = FF80 05DC



HT4 – Hash Table Entries [63:48]

The HT4 field provides the most significant sixteen bits of the 64-bit hash table.

5.3.19.1 Calculating Hash Table Entries

The following “C” code describes how to calculate the hash table entries based upon 6-byte Ethernet destination addresses.

```
static ETH_ADDRESS mca_address[MAX_MCA];        /*list of MCA addresses*/
static INT16 mca_count;                         /*# of MCA addresses*/

/*
 *
 *  Function: void eth_load_mca_table (void)
 *
```

```
 *  Description:
 *
 *  This routine loads the MCA table. It generates a hash table for
 *  the MCA addresses currently registered and then loads this table
 *  into the NET+ARM chip.
 *
 *  Parameters:
 *
 *      none
 *
 *  Return Values:
 *
 *      none
 *
 */

static void eth_load_mca_table (void)

{
   WORD16 hash_table[4];

   // create hash table for MAC address
   eth_make_hash_table (hash_table);

      (*NetARM_EFE).ht4.bits.data = SWAP16(hash_table[3]);
      (*NetARM_EFE).ht3.bits.data = SWAP16(hash_table[2]);
      (*NetARM_EFE).ht2.bits.data = SWAP16(hash_table[1]);
      (*NetARM_EFE).ht1.bits.data = SWAP16(hash_table[0]);
}


/*
 *
 *  Function: void eth_make_hash_table (WORD16 *hash_table)
 *
```

```
 *  Description:
 *
 *     This routine creates a hash table based on the CRC values of
 *     the MAC addresses setup by eth_add_mca(). The CRC value of
 *     each MAC address is calculated and the lower six bits are used
 *     to generate a value between 0 and 64. The corresponding bit in
 *     the 64-bit hash table then set.
 *
 *  Parameters:
 *
 *     hash_table          pointer to buffer to store hash table in.
 *
 *  Return Values:
 *
 *     none
 *
 */

static void eth_make_hash_table (WORD16 *hash_table)

{
   int index;

   memset (hash_table, 0, 8);                /* clear hash table*/

   for (index = 0; index < mca_count; index++) /* for each mca address*/
   {
      set_hash_bit ((BYTE *) hash_table, calculate_hash_bit (mca_address [index]));
   }
}



/*
 *
 *  Function: void set_hash_bit (BYTE *table, int bit)
```

```
 *
 *  Description:
 *
 *      This routine sets the appropriate bit in the hash table.
 *
 *  Parameters:
 *
 *      table       pointer to hash table
 *      bit         position of bit to set
 *
 *  Return Values:
 *
 *      none
 *
 */

static void set_hash_bit (BYTE *table, int bit)

{
   int byte_index, bit_index;

   byte_index = bit >> 3;
   bit_index = bit & 7;
   table [byte_index] |= (1 << bit_index);
}



/*
 *
 *  Function: int calculate_hash_bit (BYTE *mca)
 *
 *  Description:
 *
 *      This routine calculates which bit in the CRC hash table needs
```

```
 *     to be set for the NET+ARM chip to recognize incoming packets with the
 *     MCA passed to us.
 *
 *  Parameters:
 *
 *     mca     pointer to multi-cast address
 *
 *  Return Values:
 *
 *     bit position to set in hash table
 *
 */

#define POLYNOMIAL 0x4c11db6L

static int calculate_hash_bit (BYTE *mca)

{
   WORD32 crc;
   WORD16 *mcap, bp, bx;
   int result, index, mca_word, bit_index;
   BYTE lsb;
   WORD16 copy_mca[3];

   memcpy (copy_mca, mca, sizeof (copy_mca));
   for (index = 0; index < 3; index++)
   {
copy_mca [index] = SWAP16 (copy_mca [index]);
   }

   mcap = copy_mca;
   crc = 0xffffffffL;

   for (mca_word = 0; mca_word < 3; mca_word++)
   {
```

```
        bp = *mcap;
        mcap++;
        for (bit_index = 0; bit_index < 16; bit_index++)
        {
          bx = (WORD16) (crc >> 16);              /* get high word of crc*/
          bx = rotate (bx, LEFT, 1);              /* bit 31 to lsb*/
           bx ^= bp;                              /* combine with incoming*/
          crc <<= 1;                              /* shift crc left 1 bit*/
          bx &= 1;                                /* get control bit*/
          if (bx)                                 /* if bit set*/
          {
                  crc ^= POLYNOMIAL;               /* xero crc with polynomial*/
          }
          crc |= bx;                  /* or in control bit*/
          bp = rotate (bp, RIGHT, 1);
      }
    }

      // CRC calculation done. The 6-bit result resides in bit
      // locations 28:23

      result = (crc >> 23) & 0x3f;

      return result;

    }
```

5.4 External CAM Filtering

The NET+ARM chip provides support for external Ethernet CAM filtering. External Ethernet CAM filtering requires an external CAM controller to operate in concert with the MAC within the NET+ARM chip. To support this feature, the signals PORTB1 and PORTB5 must be configured for the special function signals RPSF* and REJECT* respectively. Refer to section *8.2.8 PORT B Register* for details on configuring special function mode for PORTB1 and PORTB5.

The RPSF* signal is driven active low by the NET+ARM chip to identify the beginning of each Ethernet packet being transferred from the Ethernet PHY to the NET+ARM chip internal MAC. The RPSF* signal is driven active low while the fifth nibble is being transferred. The external CAM hardware must monitor the MII receive interface between the PHY and MAC waiting for the RPSF* assertion. When RPSF* is asserted, the CAM hardware can then extract the destination address field from the MII receive bus.

After performing the necessary destination address lookup, the incoming packet can be rejected by the CAM filtering hardware by simply asserting the REJECT* input active low during any nibble-time between RPSF* assertion and nibble number 128.

Figure 5-2 shows the timing relationship between the RPSF*, REJECT*, and MII receive interface signals. In this example, the MII receive interface is transferring a packet whose first 6 nibbles have the values 1,2,3,4,5, and 6. Notice that the RPSF* signal is activated while the fifth nibble is being transferred from the MII (nibbles are transferred in Little Endian order within a BYTE).

The external CAM hardware uses the RPSF* signal to find the alignment for the destination address. After the lookup is performed, the CAM hardware can assert the REJECT* signal to discard the frame. The REJECT* signal can be asserted during any RXCLK cycle between when RPSF* is asserted and nibble number 128.

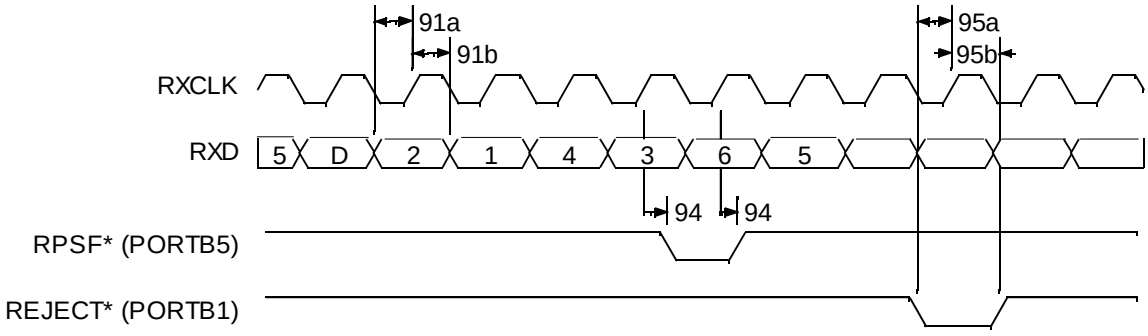


Figure 5-2: External Ethernet CAM Filtering

Num	Characteristic	Min	Max	Unit
90	TXCLK high to TXD, TXDV, TXER valid		20	ns
91a	RXD, RXER, RXDV valid to RXCLK high (setup)	10		ns
91b	RXCLK high to RXD, RXER, RXDV invalid (hold)	0		ns
92	MDC high to MDIO change		20	ns
93a	MDIO valid to MDC high (setup)	10		ns
93b	MDC high to MDIO invalid (hold)	0		ns
94	RXCLK high to RPSF* change		20	ns
95a	REJECT* valid to RXCLK high (setup)	10		ns
95b	REJECT* valid from RXCLK high (hold)	0		ns

Table 5-13: Ethernet Timing

Chapter 6
ENI Controller Module

The Embedded Network Interface (ENI) module provides the interface between the NET+ARM chip and some external device. The NET+ARM chip supports five styles of interfaces; only one mode can be operational at any given time.

6.1 ENI Controller Modes of Operation

1. IEEE 1284 Host Port—4 Ports
2. ENI Host Shared RAM-16 Interface—16-bit Shared Memory Only Interface
3. ENI Host Shared RAM-8 Interface—8-bit Shared Memory Only Interface
4. ENI Host FIFO-16 Interface—16-bit Register and Shared Memory Interface
5. ENI Host FIFO-8 Interface—8-bit Register and Shared Memory Interface

Figure 6-1 provides a high-level view of the ENI controller module. The three blocks in the middle of this diagram depict the various ENI interfaces. The remaining circuitry makes up the common support hardware required to complete the module.

The ENI controller module is allocated a total of 40 NET+ARM chip I/O pins. The usage of these 40 pins is automatically determined when the ENIMODE configuration bits are written in the general control register. The I/O pins for the ENI controller have two configurations; IEEE 1284 mode and ENI mode. Refer to section *1.4 NET+ARM Chip Pinout,* for details concerning how the pins are shared between modes.

The IEEE 1284 mode is used in commercial network print server applications. The commercial application provides a bridge between a LAN and up to 4 external devices using the IEEE 1284 Parallel Port interface. When the IEEE 1284 mode is used, the ENI shared RAM and FIFO mode interfaces are disabled.

The ENI host modes are used in the OEM network server application. The OEM application provides a bridge between a LAN and a shared memory or FIFO mode interface. The shared memory interface provides up to 64K of random access shared RAM between the NET+ARM chip and an external CPU bus. The FIFO mode interface provides a data-streaming FIFO interface between the NET+ARM chip and the external system. The FIFO mode interface supports two 32-byte FIFOs, one for each data direction. When any of the ENI modes are used, the IEEE 1284 interface is disabled.

The two ENI shared RAM modes provide support for shared RAM only. The two ENI FIFO modes support both the data-streaming FIFOs and shared memory at the same time. Note that when the data-streaming FIFOs are configured for DMA, the shared memory is limited to only 8K since PA13, PA14, PA15 are reassigned for DMA handshake signals.

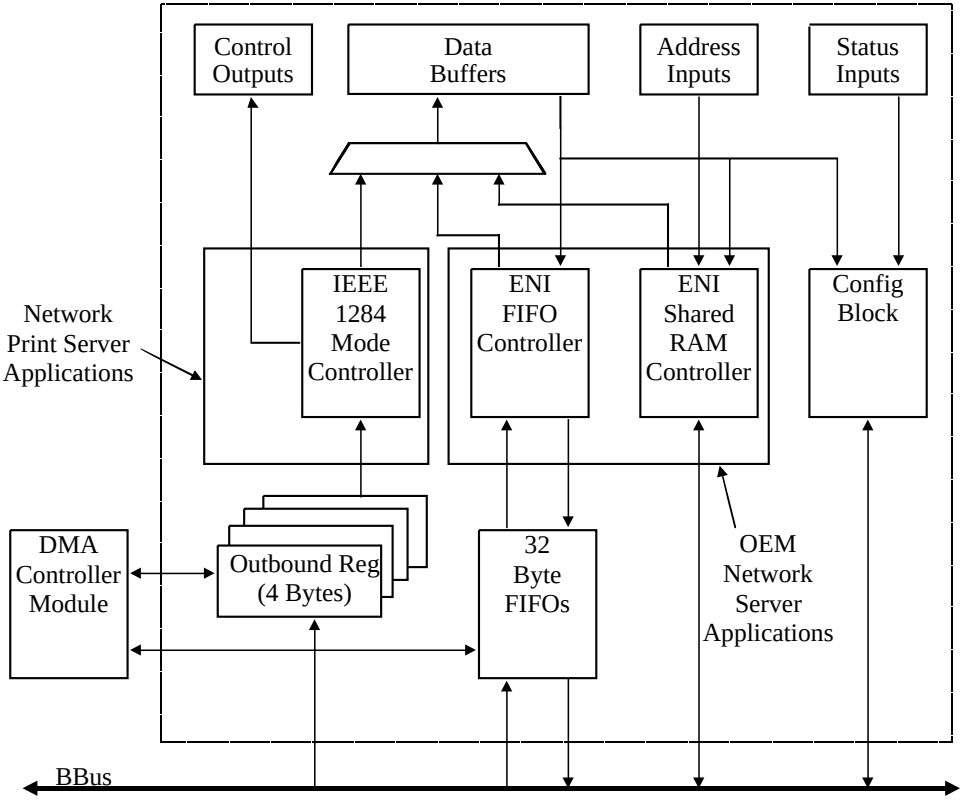


Figure 6-1: ENI Controller High-Level Block Diagram

6.2 IEEE 1284 Host Interface (4-Port) Module

When the ENI interface is configured in IEEE 1284 mode, the NET+ARM chip supports four (4) IEEE 1284 parallel port interfaces. The NET+ARM chip uses external hardware (latching transceivers) to minimize the pin count requirements. The IEEE 1284 interface uses 40 pins; all IEEE 1284 output signals are shared between ports using a simple multiplexing scheme. The NET+ARM chip provides unique CLK signals to load the shared output data into a specific 1284 port driving register.

The NET+ARM IEEE 1284 interface can only be used to provide the host side of the IEEE 1284 protocol. The NET+ARM chip cannot be used to provide the peripheral side of the IEEE 1284 protocol.

Each port requires two external 8-bit latches (a 16-bit FCT16646 device works well in this application). One latch stores the IEEE 1284 data signals while the other stores the IEEE 1284 control signals. The NET+ARM chip provides two CLK signals per port (one for data and one for control) and one output enable. The output enable allows input data flow (from multiple port transceivers) for bi-directional IEEE 1284 modes.

Figure 6-2 shows the external components required to implement this scheme.

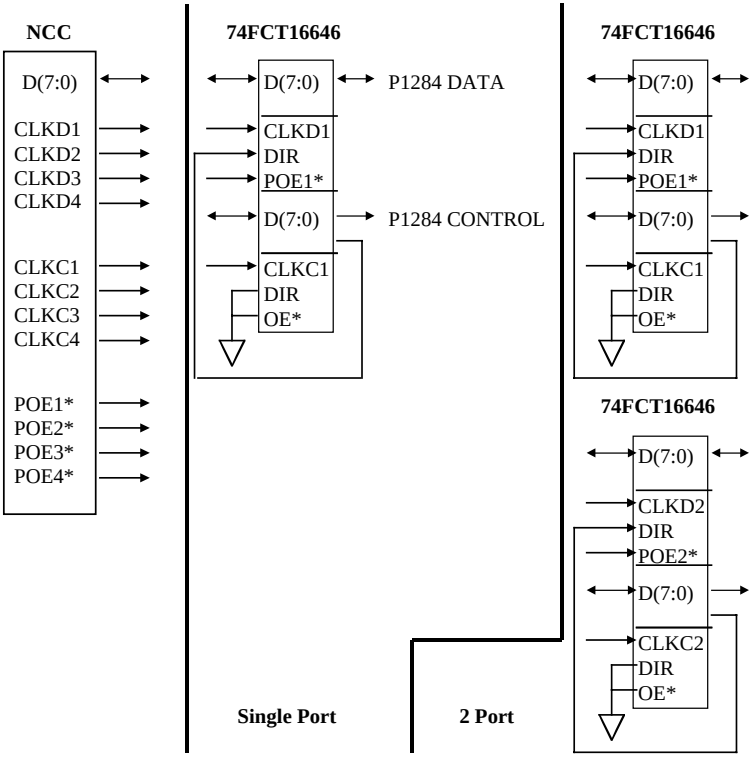


Figure 6-2: IEEE 1284 External Transceivers

The IEEE 1284 data and control signals are multiplexed using a single 8-bit data bus. The following table identifies the signal assignments.

Data Bit	Data Phase	Control Phase
D7	D7	STROBE*
D6	D6	AUTOFD*
D5	D5	INIT*
D4	D4	HSELECT*
D3	D3	PDIR
D2	D2	PIO
D1	D1	LOOPBACK
D0	D0	LOOP Strobe

Table 6-1: IEEE 1284 Data Bus Assignments

The STROBE*, AUTOFD*, INIT*, and HSELECT* signals are the four control outputs for the IEEE 1284 interface. These signals are driven through a parallel port cable.

The PDIR signal defines the direction of the external data transceiver. Its state is directly controlled by the BIDIR bit in the IEEE 1284 control register. When configured in the 0 state, data is driven from the external transceiver towards the cable. When configured in the 1 state, data is received from the cable (used for bi-directional IEEE 1284 modes).

The PIO signal is controlled by firmware. Its state is directly controlled by the PIO bit in the IEEE 1284 control register.

The LOOPBACK signal configures the port in external loopback mode. This signal can be used to control the Mux line in the external FCT646 devices. Its state is directly controlled by the LOOP bit in the IEEE 1284 control register. When set to 1, the FCT646 transceivers drive inbound data from the input latch and not the real-time (cable) interface. The LOOP strobe signal is responsible for writing outbound data into the inbound latch (completing the loopback path). The LOOP strobe signal is an inverted “copy” of the STROBE* signal. It works automatically in hardware.

Figure 6-3 provides a simple view of the IEEE 1284 controller hardware.

The IEEE 1284 control state machine works to keep the FIFOs empty by strobing data to the parallel port using the port configuration options defined by the 1284 port control register. The control state machine must manipulate the port control and data signals to adhere to the configured port mode. The control state machine services all four ports in a round-robin fashion. All four ports

share the same NET+ARM chip outputs. External latches are used to store the output signals.

The ENI 1284 controller interfaces with the BBus using four-byte outbound registers and four-byte FIFOs (one register and FIFO per port). DMA channels 3-6 work to keep the outbound registers full by causing a word write each time one empties.

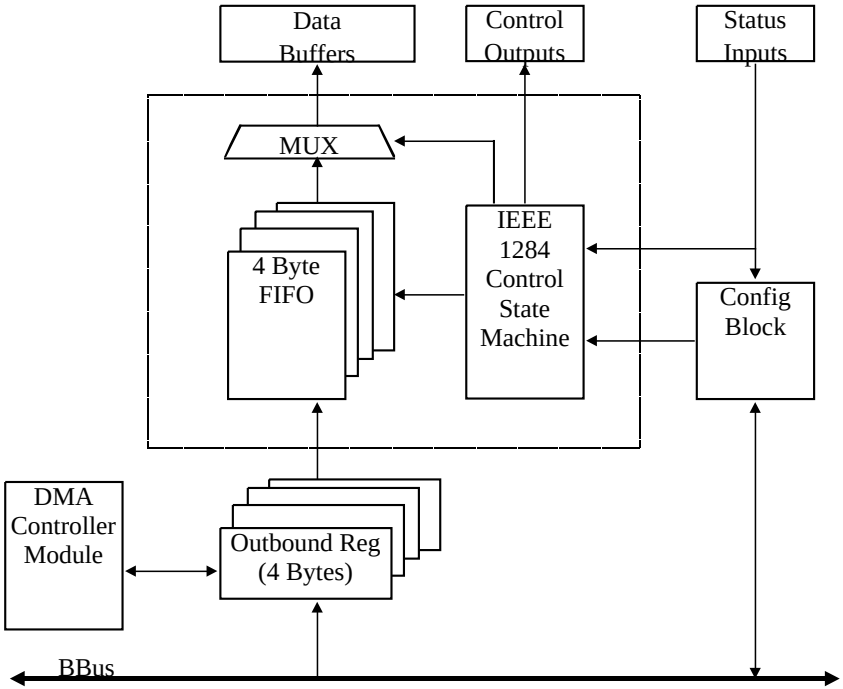


Figure 6-3: IEEE 1284 Host Controller Block Diagram

6.2.1 IEEE 1284 Signal Cross Reference

The following table is useful when cross-referencing the names between the IEEE 1284 specification and the bit names in the NET+ARM chip architecture. The input/output column defines the direction with respect to the NET+ARM chip 1284 host interface.

NET+ARM Chip Name	Host Input Output	Standard Parallel Port	Nibble Mode	Byte Mode	EPP Mode	ECP Mode
STROBE*	OUT	nSTROBE	nSTROBE	HostClk	nWRITE	HostClk
AUTOFD*	OUT	nAUTOFEED	HostBusy	HostBusy	nDATASTB	HostAck
HSELECT*	OUT	nSELECTIN	1284Active	1284Active	nADDRSTB	1284Active
INIT*	OUT	nINIT	NINIT	nINit	nRESET	nReverseRequest
ACK*	IN	nACK	PtrClk	PtrClk	nINTR	PeriphClk
BUSY	IN	BUSY	PtrBusy	PtrBusy	NWAIT	PeriphAck
PE	IN	PE	AckDataReq	AckDataReq	user defined	nAckReverse
PSELECT	IN	SELECT	Xflag	Xflag	user defined	Xflag
FAULT*	IN	nERROR	nDataAvail	nDataAvail	user defined	nPeriphRequest
DATA	OUT/IN	DATA[8:1]	Not Used	DATA[8:1]	AD[8:1]	DATA[8:1]

Table 6-2: 1284 Signal Cross Reference

6.2.2 IEEE 1284 Port Multiplexing

The NET+ARM chip multiplexes information to the external 1284 transceivers using a combination of the PDATA[7:0] data bus and the PCLKDx and PCLKCx clock signals. The PCLKDx signals are used to latch the PDATA bus into the port data transceiver. Each port is provided with a unique PCLKD signal. Similarly, the PCLKCx signals are used to latch the PDATA bus into the port control transceiver. Each port is also provided with a unique PCLKC signal. Each of the 8 PCLK signals are unique and two signals will never be activated at the same time.

Figure 6-4 and Table 6-3 provide the timing information showing the relationship between the PCLK and PDATA signals.

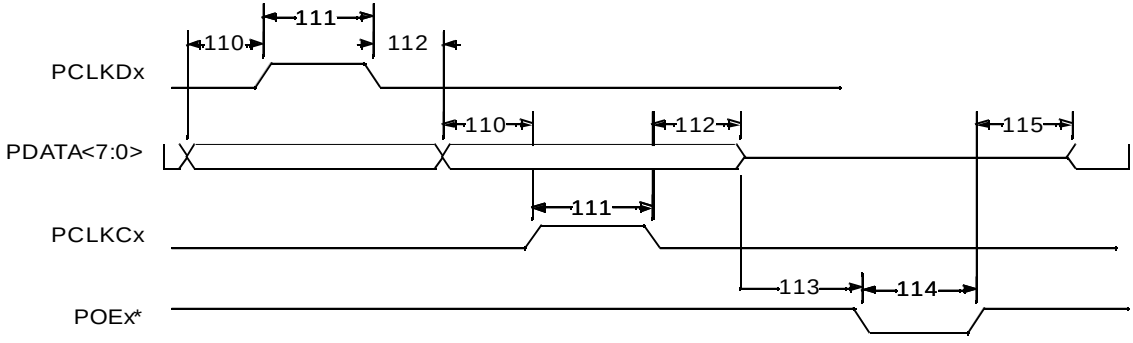


Figure 6-4: 1284 Multiplexing Timing

Num	Characteristic	Min	Max	Unit
110	PDATA valid to PCLKDx or PCLKCx high (setup)	T_{sys}		ns
111	PCLKDx and PCLKCx time high (width)	T_{sys}	T_{sys}	ns
112	PCLKDx and PCLKCx low to PDATA valid (hold)	T_{sys}		ns
113	PDATA high impedance to POEx* active low (read only)	T_{sys}		ns
114	POEx* minimum low time (read only)	T_{sys}		
115	POEX* high to PDATA driven (read only)	T_{sys}		ns

Table 6-3: 1284 Port Multiplexing Timing

6.2.3 IEEE 1284 Mode Configuration

Table 6-4 identifies the various modes for which each 1284 port can be configured. The configuration mode is provided using configuration bits in the 1284 port control registers. The proper operation mode is determined by the results of the IEEE 1284 negotiation process. The IEEE 1284 negotiation process must be done in firmware.

The NET+ARM chip provides hardware DMA support while operating in forward compatibility mode, forward ECP mode, and reverse ECP mode.

Mode	EPP	ECP	BIDIR	MAN
Manual Forward Compatibility	0	0	0	1
Automatic Forward Compatibility	0	0	0	0
Nibble Mode	0	0	0	0
Byte Mode	0	0	1	1
Forward ECP	0	1	0	0
Reverse ECP	0	1	1	0
EPP	1	0	1	0

Table 6-4: IEEE 1284 Mode Configuration

6.2.4 IEEE Negotiation

The NET+ARM chip does not offer any hardware support for IEEE negotiation. Negotiation is handled using the manual mode version of forward compatibility mode. Refer to Section *6.2.5 IEEE 1284 Forward Compatibility Mode*. Negotiation must be executed by manually manipulating the IEEE 1284 control signals according to the IEEE 1284 specification.

6.2.5 IEEE 1284 Forward Compatibility Mode

While configured to operate in forward compatibility mode, the 1284 interface signals can operate in either manual or automatic mode. Manual mode is established when the MAN bit is set to 1 in the port control register. Automatic mode is established when the MAN bit is set to 0. The manual forward compatibility mode configuration supports the IEEE 1284 negotiation function.

When configured to operate in automatic forward compatibility mode, each 1284 port can operate in one of 2 different timing modes. The timing modes are configured using the FAST bit in the IEEE 1284 port control registers. Each timing mode is slightly different. The timing modes are broken down into three major phases: 1) Data Setup Time, 2) Port Strobe Time, 3) Data Hold Time.

The AUTOFD*, HSELECT*, and INIT* outputs are statically driven based upon the associative bit settings in the port control registers. In manual forward compatibility mode, the STROBE* signal is manually controlled using the MSTB* control bit in the port control register. The DATA signals are driven from the last byte value written to the 1284 channel data register.

The ACK*, BUSY, PE, PSELECT, and FAULT* bits are manually read via the port control register. The ACK* input can be configured to generate an interrupt on the high-to-low transition on ACK*.

When the port is configured to operate in automatic mode (MAN bit set to 0), the hardware automatically manipulates the STROBE* and DATA signals based upon the current state of the BUSY input. Data bytes are strobed as they are written into the channel data registers. The channel data register can be manually filled by the processor or automatically filled using a DMA channel (when DMAE is set to 1). When manually loading the channel data register, the OBE status bit must be honored. When using DMA, the DMA channel automatically honors the OBE status bit.

Figure 6-5 provides the relative time between STROBE*, DATA, and BUSY when operating in automatic forward compatibility mode. The terminology STROBE refers to the 1284 strobe configuration found in the IEEE 1284 port configuration registers. The STROBE width can be configured between 0.5 and 10 us. The STROBE widths are not exact times, only minimum times.

There is some jitter introduced in the STROBE widths due to the port multiplexing.

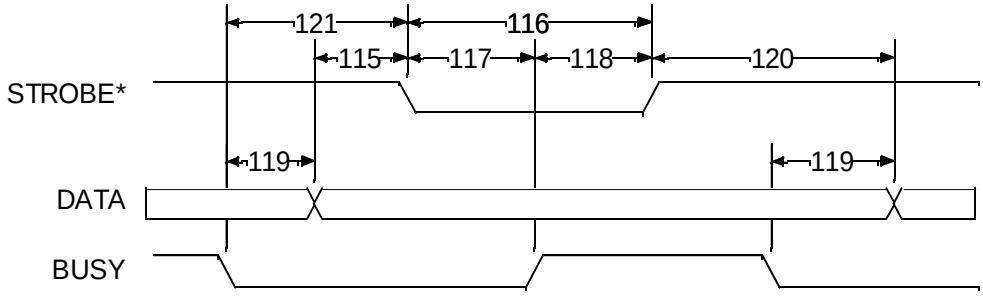


Figure 6-5: 1284 Compatibility Mode Timing

6.2.5.1 Compatibility SLOW Mode

SLOW mode is configured with FAST set to 0. This mode provides a full STROBE width of data setup time and a full STROBE width for STROBE* time. This mode provides a minimum of STROBE data hold time. This mode also maintains data hold as long as BUSY is active high.

Num	Characteristic	Min	Max	Unit
115	DATA valid to STROBE* low (setup)	STROBE		ns
116	STROBE* width low	STROBE		ns
117	STROBE* low to BUSY (input) high	0		ns
119	BUSY low to DATA change (hold)	3 * T_{sys}		ns
120	STROBE* high to DATA change (hold)	STROBE		ns

Table 6-5: 1284 Compatibility SLOW Mode Timing (FAST = 0)

6.2.5.2 Compatibility FAST Mode

FAST mode is configured with FAST set to 1. This mode provides a full STROBE width of data setup time and a full STROBE width for STROBE* time. This mode provides a minimum STROBE width of data hold time. Compatible FAST mode does not wait for BUSY low for data hold time. STROBE* does not go low until BUSY is low.

.

Num	Characteristic	Min	Max	Unit
115	DATA valid to STROBE* low (setup)	STROBE		ns
116	STROBE* width low	STROBE		ns
117	STROBE* low to BUSY (input) high	0		ns
119	BUSY low to DATA change (hold)	3 * T_{sys}		ns
120	STROBE* high to DATA change (hold)	STROBE		ns

Table 6-6: 1284 Compatibility FAST Mode Timing (FAST = 1)

6.2.6 IEEE 1284 Nibble Mode

Nibble mode is the most common way to obtain reverse channel data from a printer or peripheral device. Nibble mode data can be obtained while compatibility forward channel data is in process. The NET+ARM chip hardware provides no special support for nibble mode. Nibble mode must be handled by firmware alone.

Figure 6-6 describes the nibble mode cycle. The nibble input is a 4-bit value that can be read using the least significant nibble of the 1284 port control register. The nibble is comprised of the BUSY, PE, PSELECT, and FAULT* inputs.

1. Host signals ability to take data by asserting HostBusy low (AUTOFD*).
2. Peripheral responds by placing first nibble on status lines.
3. Peripheral signals valid nibble by asserting PtrClk low (ACK*).
4. Host sets HostBusy high to indicate that it has received the nibble and is not yet ready for another nibble.
5. Peripheral sets PtrClk high to acknowledge host.
6. States 1 through 5 repeat for the second nibble.

Figure 6-6: Nibble Mode Cycle

6.2.7 IEEE 1284 Byte Mode

A second method for obtaining reverse channel data for printers and peripherals is via the byte mode cycle. The byte mode cycle offers an enhancement over the nibble mode cycle since an entire byte can be transferred during a single cycle. The disadvantage to byte mode is that it cannot proceed while forward channel compatibility mode is in process.

Figure 6-7 describes the byte mode cycle. The AUTOFD* and STROBE* signals are manually manipulated.

1. The host signals ability to take data by asserting HostBusy low (AUTOFD*).
2. The peripheral responds by placing first byte on data lines.
3. The peripheral signals valid byte by asserting PtrClk low (ACK*).
4. The host sets HostBusy high to indicate that it has received the byte and is not ready for another byte yet.
5. The peripheral sets PtrClk high to acknowledge the host.
6. The host pulses HostClk (STROBE* via MSTB*) as an acknowledgment to the peripheral device.
7. States 1 through 5 repeat for additional bytes.

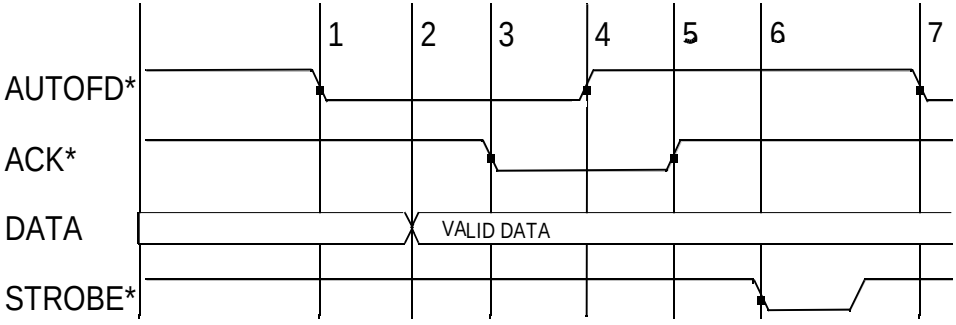


Figure 6-7: Byte Mode Cycle

The transition between forward compatibility mode and byte mode requires software intervention. When transitioning between forward compatibility and byte mode, the firmware must set the AUTOFD* (HostBusy) signal to 0 and the BIDIR signal to 1.

The transition between byte mode and forward compatibility mode requires software intervention. When transitioning between byte mode and forward compatibility mode, the firmware must wait for ACK* to be set high by the peripheral. After ACK* is set to 1, the firmware can set the BIDIR signal to 0. After the BIDIR signal is set to 0, the hardware immediately begins managing the forward channel data traffic.

6.2.8 IEEE 1284 Forward ECP Mode

When configured to operate in forward ECP mode, each 1284 port can operate in one of 2 different timing modes. The timing modes are configured using the FAST bit in the IEEE 1284 port control registers. Each timing mode is slightly different. The timing modes are broken down into three major phases: 1) Data Setup Time, 2) Port Strobe Time, 3) Data Hold Time.

The AUTOFD*, HSELECT*, and INIT* outputs are statically driven based upon the associative bit settings in the port control registers.

The ACK*, BUSY, PE, PSELECT, and FAULT* bits are manually read via the port control register. The ACK* input can be configured to generate an interrupt on the high-to-low transition on ACK*.

When the port is configured for forward ECP mode, the hardware automatically manipulates the STROBE* and DATA signals based upon the current state of the BUSY input. Data bytes are strobed as they are written into the channel data registers. The channel data register can be manually filled by the processor or automatically filled using a DMA channel (when DMAE is set to 1). When manually loading the channel data register, the OBE status bit must be honored. When using DMA, the DMA channel automatically honors the OBE status bit.

Figure 6-8 provides the relative time between STROBE*, DATA, and BUSY when operating in forward ECP mode. The terminology STROBE refers to the 1284 strobe configuration found in the IEEE 1284 port configuration registers. The STROBE width can be configured between 0.5 and 10 us. The STROBE widths are not exact times, only minimum times. There is some jitter introduced in the STROBE widths due to the port multiplexing.

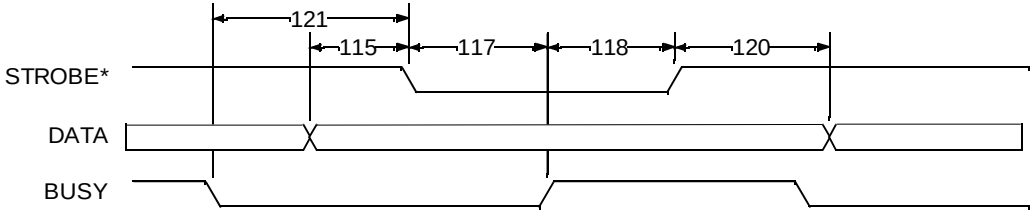


Figure 6-8: 1284 Forward ECP Mode Timing

The IEEE 1284 ECP definition provides a concept of command and data codes in the ECP mode of operation. The host command vs. data encoding is provided by the AUTOFD* (HostAck) control signal. When AUTOFD* is high, the byte transfer represents a data operand. When AUTOFD* is low, the byte transfer represents a command operand. Command and data operands cannot be intermixed within a single DMA buffer descriptor. The AUTOFD* signal must be manipulated by firmware. Careful coordination of setting AUTOFD* and writing to the channel data registers must be established to ensure 1284 protocol compliance.

6.2.8.1 ECP SLOW Mode

SLOW mode is configured with FAST set to 0. This mode provides a full STROBE width of data setup time. The STROBE* remains active until BUSY becomes active. This mode provides a minimum STROBE width of data hold time. STROBE* does not go low until BUSY is low.

Num	Characteristic	Min	Max	Unit
115	DATA valid to STROBE* low (setup)	STROBE		ns
117	STROBE* low to BUSY (input) high	STROBE		ns
118	BUSY high to STROBE* high (hold)	3 * T_{sys}		ns
120	STROBE* high to DATA change (hold)	STROBE		ns
121	BUSY low to STROBE* low (setup)	3 * T_{sys}		ns

Table 6-7: 1284 SLOW Forward ECP Mode Timing (FAST = 0)

6.2.8.2 ECP FAST Mode

FAST mode is configured with FAST set to 1. This mode drives STROBE* active immediately after data is valid. The STROBE* remains active until BUSY becomes active. Data can change immediately after STROBE* is removed. STROBE* does not go low until BUSY is low.

Num	Characteristic	Min	Max	Unit
115	DATA valid to STROBE* low (setup)	3 * T_{sys}		ns
117	STROBE* low to BUSY (input) high	0		ns
118	BUSY high to STROBE* high (hold)	3 * T_{sys}		ns
120	STROBE* high to DATA change (hold)	3 * T_{sys}		ns
121	BUSY low to STROBE* low (setup)	3 * T_{sys}		ns

Table 6-8: 1284 FAST Forward ECP Mode Timing (FAST = 1)

6.2.9 IEEE 1284 Reverse ECP Mode

The 1284 reverse ECP mode is a very fast and effective way of obtaining reverse channel data from a 1284 peripheral. The 1284 reverse ECP mode of operation provides hardware DMA support.

The transition between forward and reverse ECP mode requires software intervention. When transitioning between forward and reverse ECP mode, the firmware must set the INIT* (nReverseRequest) signal to 0 and the BIDIR signal to 1. After the BIDIR signal is set to 1, the hardware immediately begins managing the reverse channel data traffic.

The transition between reverse and forward ECP mode requires software intervention. When transitioning between reverse and forward ECP mode, the firmware must set the INIT* (nReverseRequest) signal to 1 and wait for the peripheral to respond with PE (nAckReverse) at 1. After PE is set to 1, the firmware can set the BIDIR signal to 0. After the BIDIR signal is set to 0, the hardware immediately begins managing the forward channel data traffic.

The HSELECT*, and INIT* outputs are statically driven based upon the associative bit settings in the port control registers. The STROBE* signal is always high when a channel is configured for reverse ECP mode.

The PE, PSELECT, and FAULT* bits are manually read via the port control register.

When the port is configured for reverse ECP mode, the hardware automatically manipulates the AUTOFD* signal based upon the current state of the ACK* input. Inbound DATA bytes are stored in the channel data register. When four bytes have been received, or the reverse ECP channel is “closed,” the inbound buffer ready (IBR) bit is set in the channel control register. While IBR is set, the RXFDB field identifies how many bytes are available in the channel data register.

The IBR bit can be configured to generate an interrupt. The IBR bit can also be used by a DMA channel (provided DMAE is set to 1) to automatically move the data to a receive data block.

The BUSY input provides the ECP reverse mode command/data indicator. The hardware automatically monitors the BUSY input. When BUSY is high, then an inbound reverse channel byte is considered DATA. The DATA byte is simply moved to the channel data register. When the BUSY input is sample low, then the inbound reverse channel byte is considered command information. The command byte is placed into the channel data register and the receive buffer is marked “closed.” The RBCC bit is set in the channel command register. The RBCC bit indicates the last byte in the channel data register is a command code. The RBCC bit is automatically cleared when the channel data register is read.

The reverse ECP mode provides a character timer that can be used to mark the receive buffer as “closed.” Each time a byte is moved to the channel data register, the character timer is reset. If the character timer reaches 10 times the STROBE configuration, the IBR and RBCT bits are automatically set in the port control register. The RBCT bit is automatically cleared when the channel data register is read. The purpose of the character timer is to guard against stale data

sitting in the port data register.

The RBCC and RBCT bits are also used to close a DMA buffer descriptor. When either bit is set, the DMA buffer descriptor is “closed.” The RBCC and RBCT status bits are written to the two most significant bits (D15:14) of the DMA buffer descriptor’s status field. The character timer begins operation when at least one data byte has been written to the receive data block. The character timer is cleared each time a byte is written to the channel data register. The purpose of the character timer is to guard against stale data sitting in the DMA receive data block.

Figure 6-9 describes the ECP reverse mode timing cycle. The INIT* signal is controlled manually. The PE input is polled. The AUTOFD* signal is automatically controlled by hardware. The ACK* and BUSY inputs are automatically monitored by hardware.

1. The host requests a reverse channel transfer by setting INIT* low.
2. The peripheral signals ready to continue by setting PE low.
3. The peripheral places data on the DATA bus and drives the command/data code on the BUSY signal.
4. The peripheral asserts ACK* low to indicate reverse channel data is ready.
5. The host acknowledges the transaction by driving AUTOFD* high.
6. The peripheral drives ACK* high again.
7. The host drives AUTOFD* low to indicate it’s ready for the next byte.
8. The peripheral can now change the DATA bus for the next byte.
9. Steps 4 through 7 are continuously repeated.

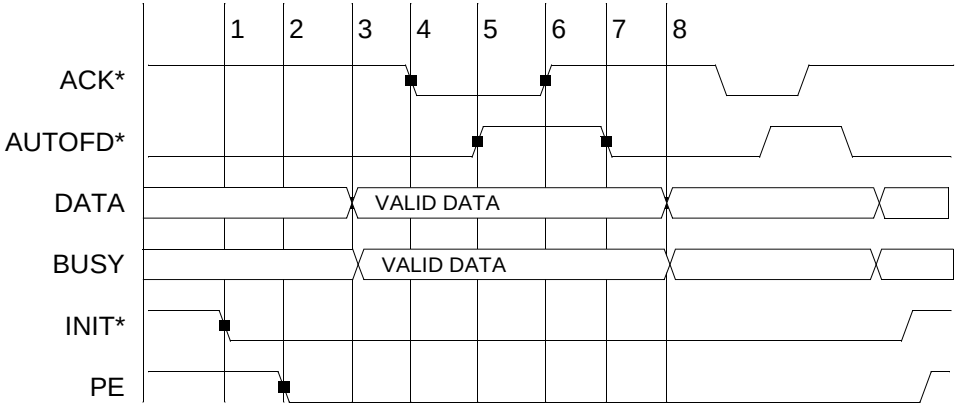


Figure 6-9: ECP Reverse Mode Timing

6.2.10 IEEE 1284 EPP Mode

The EPP protocol provides yet another mechanism for the host to access information in the peripheral. The EPP protocol allows the host to directly write/read information in the peripheral device. The host can access both address and data information in the peripheral.

When an EPP write or read cycle occurs, an interlocking handshake occurs between the host and peripheral. During the read or write cycle, wait states are inserted into the NET+ARM chip bus master memory cycle until the transaction is complete. The number of wait states is a function of the speed of the peripheral device.

To execute an address or data cycle, the NET+ARM chip bus master must access a specific byte within the port data register address. The byte address is a function of the configured Endian mode for the NET+ARM chip. Refer to section *6.5.6 IEEE 1284 Channel Data Registers* to identify which data byte to access. Accessing a half-word, full-word, or incorrect byte location within the port data register (while configured in EPP mode) results in an abort exception.

When an EPP write/read cycle is executed by the NET+ARM chip bus master, the hardware automatically manipulates the STROBE*, AUTOFD*, and HSELECT* signals. The hardware automatically monitors the BUSY input.

A DMA channel can be configured to perform EPP address and/or data write/read cycles. The DMA channel must be configured to perform memory-to-memory operations to the proper byte address within the port data register. The DMA channel must be configured to not increment the source/destination address (depending upon the direction).

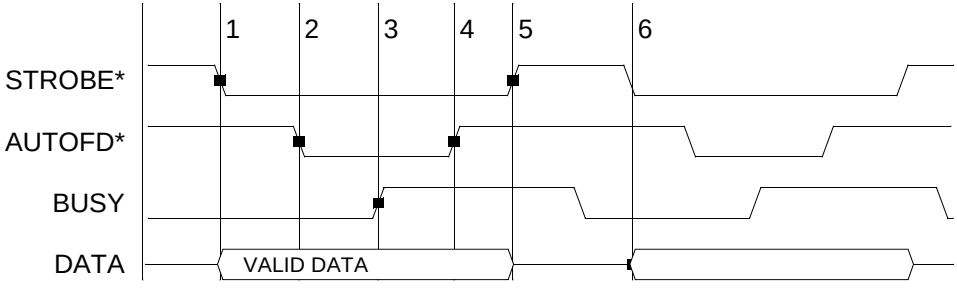


Figure 6-10: EPP Data Write Cycle

EPP Data Write Cycle:

1. NET+ARM chip bus master executes a write cycle to the EPP data byte address in the port data register. The host issues DATA and STROBE* low as a result. STROBE* low indicates a write cycle.
2. The AUTOFD* signal is asserted active since BUSY is inactive. The host does not assert AUTOFD* until BUSY is inactive.

3. The host waits for the acknowledgment from the peripheral via an active BUSY.
4. The host drives AUTOFD* inactive high to acknowledge the peripheral.
5. The host drives STROBE* inactive and tri-states the DATA bus to end the EPP write cycle.
6. States 1 through 5 are repeated as necessary.

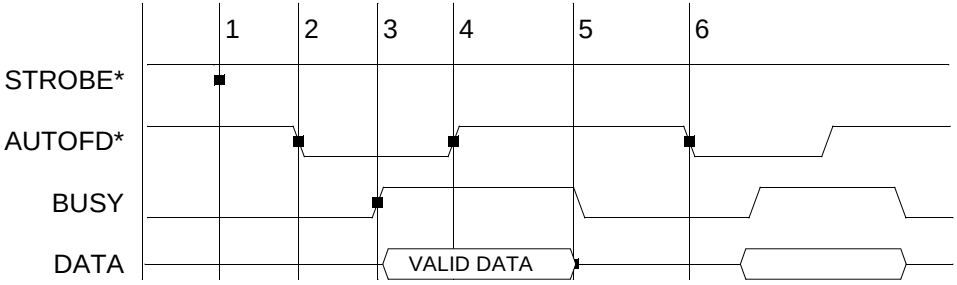


Figure 6-11: EPP Data Read Cycle

EPP Data Read Cycle:

1. NET+ARM chip bus master executes a read cycle from the EPP data byte address in the port data register. The Host asserts STROBE* high as a result. STROBE* high indicates a read cycle.
2. The AUTOFD* signal is asserted active since BUSY is inactive. The host does not assert AUTOFD* until BUSY is inactive.
3. The host waits for the acknowledgment from the peripheral via an active BUSY. The peripheral drives the data bus when BUSY is driven active.
4. The host drives AUTOFD* inactive high to acknowledge the peripheral. Read data is latched at this time.
5. The peripheral removes BUSY and tri-states the data bus.
6. States 1 through 5 are repeated as necessary.

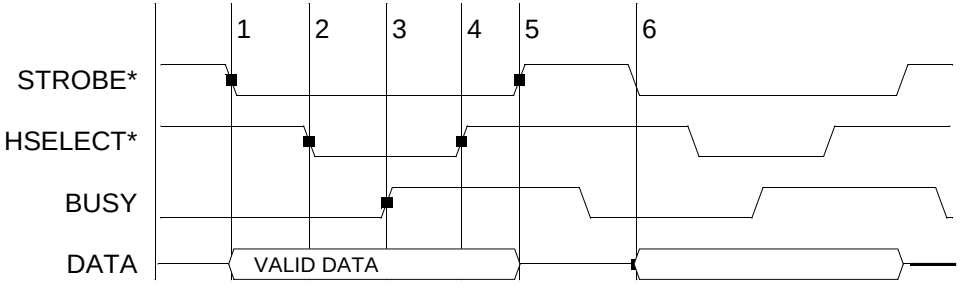


Figure 6-12: EPP Address Write Cycle

EPP Address Write Cycle:

1. NET+ARM chip bus master executes a write cycle to the EPP address byte address in the port data register. The host issues DATA and STROBE* low as a result. STROBE* low indicates a write cycle.
2. The HSELECT* signal is asserted active since BUSY is inactive. The host does not assert HSELECT* until BUSY is inactive.
3. The host waits for the acknowledgment from the peripheral via an active BUSY.
4. The host drives HSELECT* inactive high to acknowledge the peripheral.
5. The host drives STROBE* inactive and tri-states the DATA bus to end the EPP write cycle.
6. States 1 through 5 are repeated as necessary.

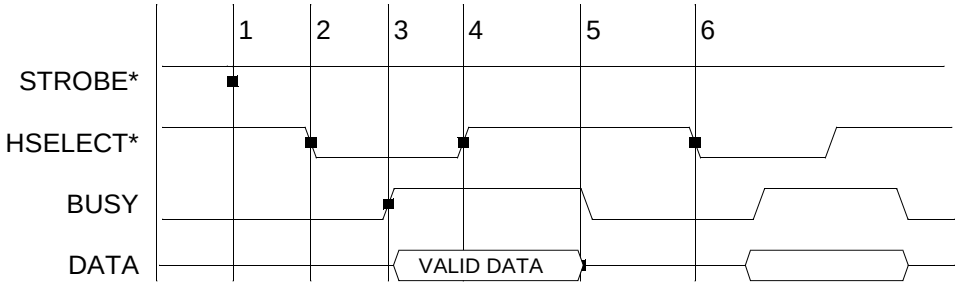


Figure 6-13: EPP Address Read Cycle

EPP Address Read Cycle:

1. NET+ARM chip bus master executes a read cycle from the EPP address byte address in the port data register. The host asserts STROBE* high as a result. STROBE* high indicates a read cycle.
2. The HSELECT* signal is asserted active since BUSY is inactive. The host does not assert HSELECT* until BUSY is inactive.
3. The host waits for the acknowledgment from the peripheral via an active BUSY. The peripheral drives the data bus when BUSY is driven active.
4. The host drives HSELECT* inactive high to acknowledge the peripheral. Read data is latched at this time.
5. The peripheral removes BUSY and tri-states the data bus.
6. States 1 through 5 are repeated as necessary.

6.3 ENI Shared RAM Module

When the ENI Interface is configured for ENI host shared RAM mode, the NET+ARM chip provides up to 64 Kbytes of shared RAM between the NET+ARM chip and external ENI device. Figure 6-14 shows the hardware required for the ENI shared RAM configuration. The ENI control state machine interprets the control signals for the configured mode to determine when the external CPU wants to access the shared RAM interface.

When the ENI shared RAM state machine determines that the external CPU wants to access shared RAM, the state machine requests ownership of the BBus. When the BBus arbiter grants the BBus to the ENI controller module, the state machine translates the address provided by the external ENI device to the physical address for the shared RAM (using values in the ENI shared RAM address register). The state machine uses the translated address to access the proper data in external RAM (either read or write). When the bus controller module completes the external RAM cycle, the control state machine signals the external CPU to indicate the cycle is complete (providing read data during read cycles).

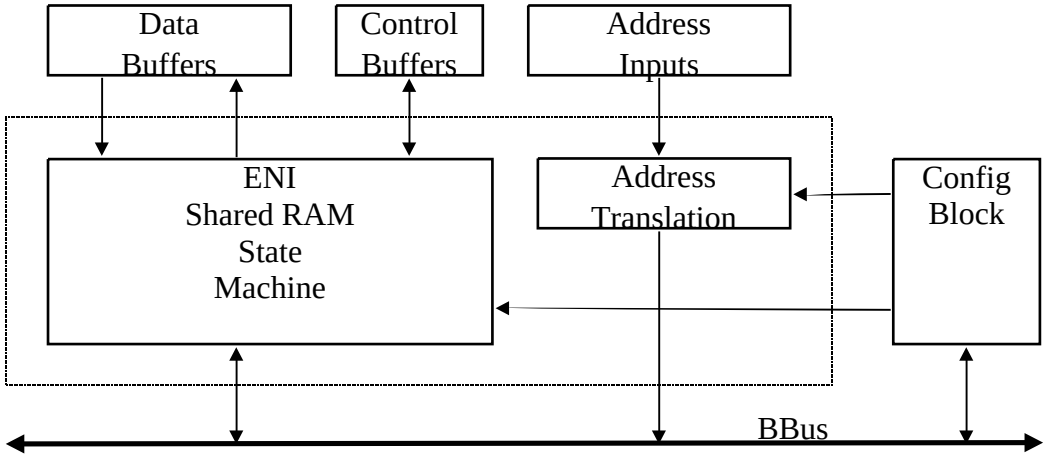


Figure 6-14: ENI Shared RAM Block Diagram

The ENI shared RAM occupies a physical block within external NET+ARM chip system RAM. The ENI controller accesses the shared RAM through the ENI interface. The NET+ARM chip firmware directly accesses the shared RAM within its system RAM. To aid in moving data between shared RAM and normal protocol stack memory, the DMA controller module provides a means of executing memory to memory DMA transfers.

6.4 ENI FIFO Mode Module

The ENI host FIFO mode interface provides a FIFO interface between the NET+ARM chip and some external CPU system. This register interface provides a streaming bi-directional data transfer. DMA channels 3 and 4 are used by the NET+ARM chip to move the data between the register interface and external RAM.

DMA Channel 4 moves data from external RAM to the register interface. DMA channel 3 moves data from the register interface to external RAM.

Figure 6-15 provides a view of the hardware required to support FIFO mode. The FIFO mode control state machine is responsible for running the register interface.

The FIFO mode interface has two 32-byte FIFOs. The DMA controllers work to keep the output FIFO full and the input FIFO empty.

The FIFO mode operates in a byte/word streaming fashion. Data from buffers is moved between external RAM and the FIFO mode peripheral interface automatically using DMA operations. In the outbound direction, the transmit buffers can be of any size (including an odd number of bytes).

In the inbound direction, the receive buffer size must be an integral of four bytes. This restriction is imposed to keep the receive FIFOs working efficiently. Without this restriction, residual bytes can

remain in the FIFO and stall the completion of the receive data buffer.

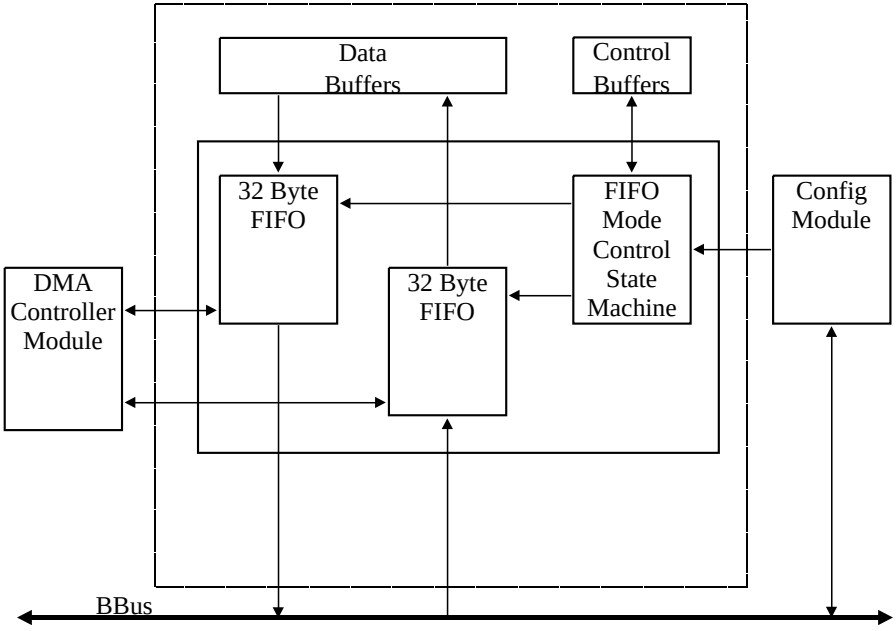


Figure 6-15: ENI FIFO Mode Block Diagram

6.5 ENI Controller Configuration

The ENI controller module has a block of configuration space that is mapped into the ENI module configuration space as defined in Table 2-1: BBus Address Decoding.

Address	Register
FFA0 0000	General Control Register
FFA0 0004	General Status Register
FFA0 0008	FIFO Mode FIFO Data Register
FFA0 0010	IEEE 1284 Port 1 Control Register
FFA0 0014	IEEE 1284 Port 2 Control Register
FFA0 0018	IEEE 1284 Port 3 Control Register
FFA0 001C	IEEE 1284 Port 4 Control Register
FFA0 0020	IEEE 1284 Channel 1 Data Register
FFA0 0024	IEEE 1284 Channel 2 Data Register
FFA0 0028	IEEE 1284 Channel 3 Data Register
FFA0 002C	IEEE 1284 Channel 4 Data Register
FFA0 0030	ENI Control Register
FFA0 0034	ENI Pulsed Interrupt Register
FFA0 0038	ENI Shared RAM Address Register
FFA0 003C	ENI Shared Register

Table 6-9: ENI Controller Configuration Registers

6.5.1 ENI Module Hardware Initialization

The ENIMODE bits (in the general control register) and the ENI control register bits are implementation-dependent. Since external hardware circuitry relies on this configuration during the time of RESET, the ENI module allows external jumpers to configure their initial values. The system bus address bits are used for this purpose during a powerup reset. The NET+ARM chip provides internal pullup resistors on all address signals. Weak external pull-down resistors can be employed to configure the ENIMODE and ENICTL register bits. The following identifies which ADDR bits control what functions.

ADDR[22:20]	:	ENIMODE[2:0]	Configuration Bits
ADDR[07]	:	ENI Control	PSIO* (0 = PSIO, 1 = Normal)
ADDR[06]	:	ENI Control	WR_OC
ADDR[05]	:	ENI Control	DINT2*
ADDR[04]	:	ENI Control	I_OC
ADDR[03]	:	ENI Control	DMAE*
ADDR[02]	:	Reserved	
ADDR[01]	:	ENI Control	EPACK*
ADDR[00]	:	ENI Control	PULINT*

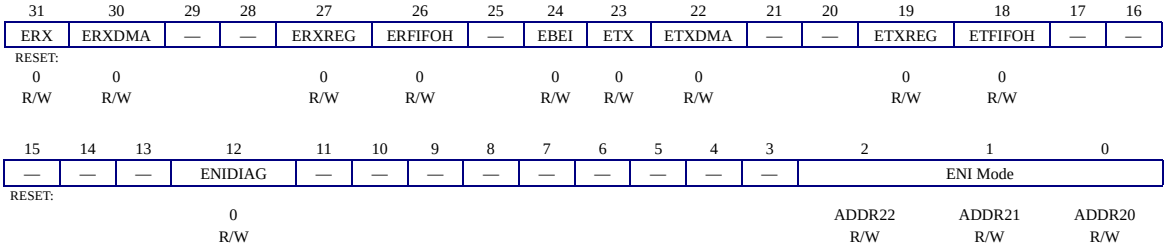
(Refer to Section *11.4 NET+ARM Chip Bootstrap Initialization*.)

Note: The inverted PSIO address bit (ADDR7) is loaded into the PSIO configuration bit within the ENI Control Register. In the ENI Control Register, 0=Normal, 1=PSIO).

6.5.2 General Control Register

The general control register is a 32-bit register that contains control bits which affect the entire ENI controller module.

Address = FFA0 0000



ERX **Enable Receive FIFO**

0 = Disables inbound data flow and resets the FIFO

1 = Enables inbound data flow

The ERX bit must be set to active high to allow data to be received through the FIFO interface. The ERX bit can be used to reset the receive side FIFO. In general, the ERX bit should be set once on device open.

ERXDMA **Enable Receive FIFO DMA**

0 = Disables inbound DMA data request (use to stall receiver)

1 = Enables inbound DMA data request

The ERXDMA bit must be set to active high to allow the receive FIFO to issue receive data move requests to the DMA controller. DMA Channel 3 is used for the ENI receive FIFO. This bit can be cleared to temporarily stall receive side FIFO DMA. This bit should not be set when operating the Ethernet receiver in interrupt service mode. In general, the ERXDMA bit should be set once on device open.

ERXREG **Enable Receive FIFO Data Ready Interrupt**

The ERXREG bit must be set to active high to generate an interrupt when data is available in the receive FIFO. The ERXREG bit should only be set when operating the receive FIFO in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable than interrupt service mode. The receive FIFO interrupt is routed to the GEN Module Interrupt Controller via the ENI Port 1 Interrupt (refer to Section *6.5.9 ENI Module Interrupts*).

ERFIFOH **Enable Receive FIFO Half Full Interrupt**

The ERFIFOH bit must be set to active high to generate an interrupt when the receive FIFO is at least half full (16 bytes). The ERFIFOH bit should only be set when operating the receive FIFO in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable than

interrupt service mode. The receive FIFO interrupt is routed to the GEN Module Interrupt Controller via the ENI Port 1 Interrupt (refer to Section *6.5.9 ENI Module Interrupts*).

EBEI **Enable ENI Bus Error Interrupt**

The EBEI bit can be set to 1 in order to cause an ARM interrupt to occur when the BEI bit is set to 1 in the ENI General Status Register. The BEI bit will be set to 1 when the ENI Shared RAM module encounteres a Bus Error (same as Data Abort) condition, while attempting to address Shared RAM. This typically indicates the Shared RAM pointer is invalid. The ENI Bus Error interrupt is routed to the GEN Module Interrupt Controller via the ENI Port 4 Interrupt (refer to Section *6.5.9 ENI Module Interrupts*).

ETX **Enable Transmit FIFO**

0 = Disables outbound data flow and resets the FIFO

1 = Enables outbound data flow

The ETX bit must be set to active high to allow data to be written to the transmit FIFO. The ETX bit can be used to reset the transmit FIFO by setting ETX to 0. In general, the ETX bit should be set once on device open.

ETXDMA **Enable Transmit FIFO DMA**

0 = Disables outbound DMA data request (use to stall transmitter)

1 = Enables outbound DMA data request

The ETXDMA bit must be set to active high to allow the transmit FIFO to issue transmit data move requests to the DMA controller. DMA Channel 4 is used for the ENI transmit FIFO. This bit can be cleared to temporarily stall receive side FIFO DMA. This bit should not be set when operating the receive FIFO in interrupt service mode. In general, the ETXDMA bit should be set once on device open.

ETXREG **Enable Transmit FIFO Data Register Ready Interrupt**

The ETXREG bit must be set to active high to generate an interrupt when the transmit FIFO is ready to accept more data. The ETXREG bit should only be set when operating the transmit FIFO in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable than interrupt mode. The transmit FIFO interrupt is routed to the GEN Module Interrupt Controller via the ENI Port 2 Interrupt (refer to Section *6.5.9 ENI Module Interrupts*).

ETFIFOH **Enable Transmit FIFO Half Empty Interrupt**

The ETFIFOH bit must be set to active high to generate an interrupt when the transmit FIFO is at least half empty (< 16 bytes). The ETFIFOH bit should only be set when operating the transmit FIFO in interrupt service mode instead of DMA mode. In general, DMA is more desirable than interrupt mode. The transmit FIFO interrupt is routed to the GEN Module Interrupt Controller via the ENI Port 2 Interrupt (refer to Section *6.5.9 ENI Module Interrupts*).

ENIDIAG **Enable ENI Diagnostic Mode**

0 = Disable firmware change of ENIMODE or ENICTL Register

1 = Enable firmware change

The ENIDIAG bit provides protection against accidentally modifying the ENIMODE or ENI Control Register Settings. Setting ENIDIAG to 1 allows the software to modify the ENIMODE field in the General Control Register or the least significant eight bits of the ENI Control Register. When ENIDIAG is set to 0, the ENIMODE and least significant eight bits of the ENI Control Register cannot be modified.

ENIMODE **ENI Mode of Operation**

000 = Reserved

001 = IEEE 1284 Host Mode

010 = Reserved

011 = Reserved

100 = ENI Shared-16 RAM Mode; 16-bit Shared RAM only

101 = ENI Shared-8 RAM Mode; 8-bit Shared RAM only

110 = ENI FIFO Mode; 16-bit with 8 Kbytes Shared RAM

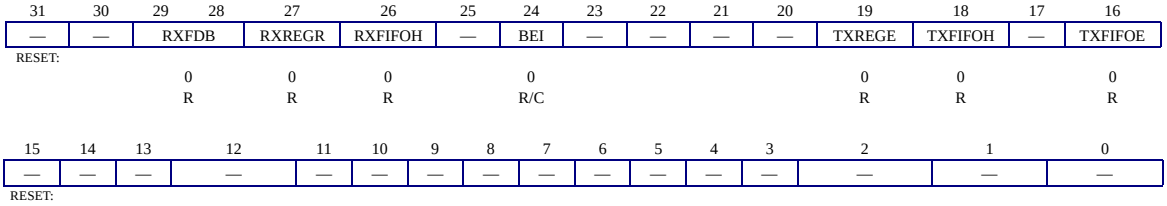
111 = ENI FIFO Mode; 8-bit with 8 Kbytes Shared RAM

The ENIMODE field configures the mode of operation for the ENI Module. The ENIMODE field will be automatically loaded on Power Up or Hardware Reset (RESET* pin) using the Bootstrap configuration bits defined on address pins A22, A21, and A20. For more detailed information on Bootstrap, please refer to Section 6.4.1 ENI Module Hardware Initialization.

6.5.3 General Status Register

The general status register is a 32-bit register that contains status bits which affect the entire ENI controller module.

Address = FFA0 0004



RXFDB **Receive FIFO Data Available**

00: Full-word

01: One-byte

10: Half-word

11: Three Bytes (LENDIAN determines which three)

This field must be used in conjunction with RXREGR. When RXREGR is set, this field identifies how many bytes are available in the receive data register. The RXFDB field is only used when operating the receive FIFO in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable than interrupt mode.

RXREGR **Receive FIFO Read Register Ready**

The RXREGR bit is set to 1 whenever data is available to be read from the receive FIFO data register. This bit, when active high, can cause an interrupt to occur when the ERXREGR bit is set in the General Control Register. The RXREGR bit is only used when operating the receive FIFO in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable than interrupt mode.

RXFIFOH **Receive FIFO Half Full**

The RXFIFOH bit is set to 1 whenever the receive FIFO is at least half full (> 16 bytes). This bit, when active high, can cause an interrupt to occur when the ERFIFOH bit is set. The RXFIFOH bit is only used when operating the receive FIFO in interrupt service mode. In general, DMA mode is more desirable than interrupt mode.

BEI **ENI Bus Error**

The BEI bit is set to 1 when the ENI Shared RAM interface receives a Bus Error (Data Abort) signal when attempting to address the Shared RAM in external memory. This typically means that the Shared RAM pointer is configured in the ENI Shared RAM Address Register. This bit, when

active high, can cause an interrupt to occur when the EBEI bit is set in the General Control Register.

TXREGE **Transmit FIFO Register Ready**

The TXREGE bit is set to 1 whenever the transmit FIFO is ready to accept data. This bit, when active high, can cause an interrupt to occur when the ETXREG bit is set in the General Control Register. The TXREGE bit is only used when operating the transmit FIFO in interrupt mode instead of DMA mode. In general, DMA mode is more desirable than interrupt mode.

TXFIFOH **Transmit FIFO Half Full**

The TXFIFOH bit is set to 1 whenever the transmit FIFO is at least half empty (< 16 bytes). This bit, when active high, can cause an interrupt to occur when the ETFIFOH bit is set in the General Control Register. The TXFIFOH bit is only used when operating the transmit FIFO in interrupt mode instead of DMA mode. In general, DMA mode is more desirable than interrupt mode.

TXFIFOE **Transmit FIFO Empty**

The TXFIFOE bit is set to 1 whenever the transmit FIFO is empty. The TXFIFOE bit cannot be used to generate an interrupt to the ARM processor.

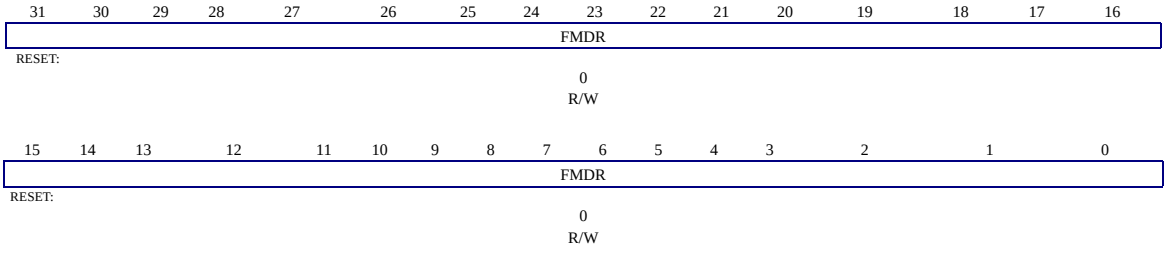
6.5.4 ENI Mode FIFO Data Register

The ENI mode FIFO data register is a 32-bit register used by the ARM processor to interface with the ENI mode FIFO. Writing to this register loads the transmit FIFO. This register can only be written when the transmit register empty (TXREGE) bit is set in the general status register. Reading from this register empties the receive FIFO. Read data is available when the RXREGR bit is set in the general status register. The RXFDB bit in the general status register indicates how many bytes are available.

When writing to the FIFO Data Register, the number of bytes written to the FIFO is controlled by the operand mode of the ARM processor. The ARM processor can write bytes, half-words, or full-words to the FIFO Data Register. These different writes can be accomplished in assembler using the following instructions respectively; STRB, STRH, STR. Similarly, these different writes can be accomplished in “C” using the following constructs respectively; *(* char)0xFFA00008, *(* short)0xFFA00008, and *(int *)0xFFA00008.

When reading from the FIFO Data Register, all the bytes available (as defined by the RXFDB field) must be read with a single instruction. If only a single byte is available, then a LDRB or *(char *) operation can be used. The LDR or *(int *) operations can also be used to read the one byte. If only a half-word is available, then a LDRH or *(short *) operation can be used. The LDR or *(int *) operations can also be used to read the half-word. When a full-word is available, the LDR or *(int *) operations must be performed.

Address = FFA0 0008



6.5.5 IEEE 1284 Port Control Registers

The NET+ARM chip supports (4) IEEE 1284 control registers; one register for each port. This is a 32-bit register containing both control and status bits. All control/status bits are active high unless an asterisk (*) appears in the signal name, in which case, it is active low. All control bits are set to their respective inactive state on reset.

Address = FFA0 0010 / 14 / 18 / 1C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PORTE	DMAE	INTEP	INTEA	ECP	LOOP	STROBE		MAN	FAST	BIDIR	PIO	MSTB*	AUTOFD*	INIT*	HSELECT*
RESET:															
0	0	0	0	0	0	0		0	0	0	0	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTEF	EPP	IBRIE	IBR	RXFDB		RBCC	RBCT	ACK*	FIFOE	OBE	ACKI	BUSY	PE	PSELECT	FAULT*
RESET:															
0	0	0	0	0		0	0	1	0	0	0	0	0	0	1
R/W	R/W	R/W	R	R		R	R	R	R	R	R/C	R	R	R	R

PORTE **1284 Port Enable**

The PORTE bit must be set to 1 to enable the 1284 port to operate. When the PORTE bit is set to 0, the 1284 port is held in reset.

DMAE **1284 Port DMA Request Enable**

The DMAE bit must be set to active high to allow the 1284 port to issue transmit data move requests to the DMA controller. This bit can be set to 0 to temporarily stall 1284 DMA operations. This bit should not be set when operating the 1284 port in interrupt service mode. In general, the DMAE bit should be set once on device open.

INTEP **1284 Port Outbound Interrupt Request Enable**

The INTEP bit must be set to active high to generate an interrupt when the 1284 Port is ready to accept more data. Setting the INTEP bit to 1 will cause an interrupt to be generated when the OBE status bit is 1. The INTEP bit should only be set when operating the 1284 Port in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable than interrupt mode.

INTEA **1284 Port Acknowledge Transition Interrupt Request Enable**

The INTEA bit must be set to active high to generate an interrupt when the 1284 Port has detected a high-to-low transition on the ACK* input. Setting the INTEA bit to 1 will cause an interrupt to be generated when the ACKI status bit is 1.

ECP **1284 ECP Mode of Operation**

The ECP bit must be set to 1 to allow the 1284 port to operate in the ECP mode of operation. Refer to Section *6.2.8 IEEE 1284 Forward ECP Mode* for more details on how to operate the 1284 port in ECP mode.

LOOP **1284 External Loopback**

When the LOOP bit is set to 1, the 1284 port is configured to operate in external loopback mode. Please refer to Section *6.5.8 IEEE 1284 External Loopback Mode* for additional information. When LOOP is set to 1, the LOOP Strobe signal is asserted instead of the STROBE* signal in the external control latch (refer to Table 6-1 *IEEE 1284 Data Bus Assignments*). The LOOP Strobe signal provides a low-to-high transition allowing the 1284 outbound data to be latched in the external latch.

The value written into the LOOP field will appear on the LOOPBACK signal in the external control latch (refer to Table 6-1 *IEEE 1284 Data Bus Assignments*).

STROBE **1284 Manual Strobe Width Configuration**

The STROBE field defines the width of the IEEE 1284 strobe timing when the 1284 port is configured to operate in Compatibility Mode. Refer to Section *6.5.7 IEEE 1284 Strobe Pulse Width* for more information.

MAN **1284 Manual Operation Mode**

The MAN bit must be set to 1 in order to manually manipulate the state of the IEEE 1284 STROBE* signal. When MAN is set to 1, the state of the STROBE* signal is defined by the MSTB* field. When MAN is set to 0, the state of the STROBE* signal is automatically controlled by hardware. The MAN setting of 1 is typically used to manipulate STROBE* for auto negotiation purposes. Refer to Section *6.2.4 IEEE Negotiation* for more details.

FAST **1284 Fast Configuration**

The FAST bit is used to control data setup and data hold times relative to the STROBE* signal when the 1284 port is configured to operate in Compatibility or ECP modes of operation. Refer to Sections *6.2.5 IEEE 1284 Forward Compatibility Mode* and *6.2.8 IEEE 1284 Forward ECP Mode* for more details concerning the configuration of the FAST field.

BIDIR **1284 Direction Configuration**

0 – Output Mode

1 – Input Mode

The BIDIR field controls the data direction for the 1284 port. When BIDIR is 0, the port is configured in output mode; data is sent from the Host to the Peripheral. When BIDIR is 1, the port is configured in input mode; data is sent from the Peripheral to the Host.

The value written into the BIDIR field will appear on the PDIR signal in the external control latch (refer to Table 6-1 IEEE 1284 Data Bus Assignments).

PIO **1284 Spare Bit**

The PIO field provides a spare control signal for application specific use.

The value written into the PIO field will appear on the PIO signal in the external control latch

(refer to Table 6-1 *IEEE 1284 Data Bus Assignments*).

MSTB* **1284 Manual Strobe Configuration**

The MSTB* bit controls the state of the STROBE* signal in the outside control latch when the MAN bit is set to 1. The MSTB* bit is used to manually manipulate the state of the STROBE* signal. This feature is primarily used for 1284 Negotiation. Refer to Section *6.2.4 IEEE Negotiation* for more details.

AUTOFD* **1284 AUTOFD* Setting**

The AUTOFD* bit controls the state of the AUTOFD* signal in the external control latch. The value written into the AUTOFD* field will appear on the AUTOFD* signal in the external control latch (refer to Table 6-1 *IEEE 1284 Data Bus Assignments*).

INIT* **1284 INIT* Setting**

The INIT* bit controls the state of the INIT* signal in the external control latch. The value written into the INIT* field will appear on the INIT* signal in the external control latch (refer to Table 6-1 *IEEE 1284 Data Bus Assignments*).

HSELECT* **1284 HSELECT* Setting**

The HSELECT* bit controls the state of the HSELECT* signal in the external control latch. The value written into the HSELECT* field will appear on the HSELECT* signal in the external control latch (refer to Table 6-1 *IEEE 1284 Data Bus Assignments*).

INTEF **1284 Port FIFO Empty Interrupt Request Enable**

The INTEF bit must be set active high to generate an interrupt when the 1284 data FIFO is empty. Setting the INTEF bit to 1 will cause an interrupt to be generated when the FIFOE status bit is 1. The INTEF bit should only be set when operating the 1284 Port in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable than interrupt mode.

EPP **1284 EPP Mode of Operation**

The EPP bit must be set to 1 to allow the 1284 port to operate in the EPP mode of operation. Refer to Section *6.2.10 IEEE 1284 EPP Mode* for more details on how to operate the 1284 port in EPP mode.

IBRIE **1284 Inbound Buffer Ready Interrupt Enable**

The IBRIE bit must be set to active high to generate an interrupt when the 1284 Port Inbound Buffer is available for reading. Setting the IBRIE bit to 1 will cause an interrupt to be generated when the IBR status bit is 1. The IBRIE bit should only be set when operating the 1284 Port in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable than interrupt mode.

IBR **1284 Inbound Buffer Ready**

The IBR bit will go active high when the inbound buffer is ready to be read by the processor. The

IBR status bit will only be active when the 1284 port is configured for operation in the ECP Reverse Channel Mode (ECP = 1; BIDIR = 1).

RXFDB **1284 Inbound Buffer FIFO Data Available**

00: Full-word

01: One-byte

10: Half-word

11: Three Bytes (LENDIAN determines which three)

This field must be used in conjunction with IBR. When IBR is set, this field identifies how many bytes are available in the inbound data register. The RXFDB field is only used when operating the receive 1284 port in interrupt service mode instead of DMA mode. In general, DMA mode is more desirable than interrupt mode.

RBCC **1284 Receive Buffer Close Command Byte**

The RBCC bit can be set at the same time IBR is set. The RBCC bit indicates that the last byte in the next read of the inbound data buffer is defined as the 1284 ECP Command Byte. The RBCC bit is automatically cleared when the inbound data buffer is read.

RBCT **1284 Receive Buffer Character Timeout**

The RBCT bit can be set at the same time IBR is set. The RBCT bit indicates that a timeout condition has caused the inbound buffer to become ready. This also implies that the RXFDB field will indicate that less than four bytes will be made available during the next read of the inbound data buffer. The timeout condition indicates there was too much time elapsed since the receipt of the last character. The maximum time between characters is defined by the following equation:

TIMEOUT = STROBE configuration * 16

Where:

TIMEOUT refers to the maximum time since the receipt of the last character.

STROBE refers to the time value defined by the STROBE field.

ACK* **1284 ACK* Input State**

The ACK* field provides the current state of the 1284 ACK* input signal. A high-to-low transition of the ACK* input will cause the ACKI status bit to be latched active high.

FIFOE **1284 FIFO Empty Status**

The FIFOE bit indicates that the 1284 outbound data register and FIFO are empty of characters. Any interrupt can be generated to the CPU when FIFOE goes active high provided that the INTEF control bit is set to 1.

OBE **1284 Outbound Buffer Empty**

The OBE bit is set to 1 when the 1284 outbound data buffer is ready to accept more data. An interrupt can be generated to the CPU when OBE goes to active high provided that the INTEP control bit is set to 1.

ACKI **1284 ACK* Transition Detected**

The ACKI bit is set to 1 whenever a high-to-low transition is detected on the 1284 ACK* signal input. The ACKI bit will remain 1 until acknowledged. The ACKI bit is acknowledged by writing a 1 to the same bit position in this register. An interrupt can be generated to the CPU when ACKI goes active high provided that the INTEA control bit is set to 1.

BUSY **1284 Busy Input State**

The BUSY field provides the current state of the 1284 BUSY input signal. The BUSY bit is one of the 4-bits in the 1284 reverse channel nibble.

PE **1284 PE Input State**

The BUSY field provides the current state of the 1284 PE input signal. The PE bit is one of the 4-bits in the 1284 reverse channel nibble.

PSELECT **1284 PSELECT Input State**

The PSELECT field provides the current state of the 1284 PSELECT input signal. The PSELECT bit is one of the 4-bits in the 1284 reverse channel nibble.

FAULT* **1284 FAULT* Input State**

The FAULT* field provides the current state of the 1284 FAULT* input signal. The FAULT* bit is one of the 4-bits in the 1284 reverse channel nibble.

6.5.6 IEEE 1284 Channel Data Registers

The IEEE 1284 channel data registers are 32-bit registers used to manually interface with the IEEE 1284 FIFOs in lieu of using DMA support. Each port has its own data register. These registers are anticipated to be used primarily as a diagnostic tool.

When writing to the Channel Data Register, the number of bytes written to the FIFO is controlled by the operand mode of the ARM processor. The ARM processor can write bytes, half-words, or full-words to the Channel Data Register. These different writes can be accomplished in assembler using the following instructions respectively: STRB, STRH, STR. Similarly, these different writes can be accomplished in "C" using the following constructs respectively: *(char *)0xFFA00020, *(short *)0xFFA00020, and *(int *)0xFFA00020.

When reading from the Channel Data Register, all the bytes available (as defined by the RXFDB field) must be read with a single instruction. If only a single byte is available, then a LDRB or *(char *) operation can be used. The LDR or *(int *) operations can also be used to read the one byte. If only a half-word is available, then a LDRH or *(short *) operation can be used. The LDR or *(int *) operations can also be used to read the half-word. When a full-word is available, the LDR or *(int *) operations must be performed.

Address=FFA0 0020 / 24 / 28 / 2C

Bits	R/W	Bit Name	Description
D31:00	W	DATA	Outbound Channel Data Register
D31:08	R		Reserved
D31:00	R	DATA	ECP Mode Reverse Mode Data
D15:08	R/W	DATA	EPP Mode Address Cycle
D31:00	W	DATA	Outbound Channel Data Register
D31:08	R	DATA	Reserved
D07:00	R/W		EPP Mode Data Cycle
D07:00	R	DATA	Byte Mode Reverse Channel Data

6.5.7 IEEE 1284 Strobe Pulse Width

The width of the IEEE 1284 strobe timing (used in compatibility mode) is a function of both the external crystal frequency and the value programmed in the STROBE field of the IEEE 1284 control register. Using an 18.432 MHz crystal as an example, the following STROBE times are possible:

STROBE	Equation	Example Value
00	$5 / F_{XTAL}$	0.542 us
01	$10 / F_{XTAL}$	1.08 us
10	$50 / F_{XTAL}$	5.42 us
11	$100 / F_{XTAL}$	10.85 us

Table 6-10: Example IEEE 1284 Strobe Widths

6.5.8 IEEE 1284 External Loopback Mode

The LOOP bit in the port control register puts the 1284 port in an external loopback mode. Each external “strobe” action forces the outbound data to be latched in the external inbound data path.

When configured in loopback mode, the inbound BUSY signal is connected to the ACKI status bit. Whenever a data byte is “strobed” in the external data latch, the outbound data transfer is stalled (by virtue of the internal BUSY connection) until the ACKI status bit is serviced. This feature allows the firmware to read each inbound data path one byte at a time. The outbound path can be either interrupt or DMA driven. The inbound data path can be either interrupt driven or polled.

6.5.9 ENI Module Interrupts

The ENI module provides 4 status indictors to the GEN module for generating an interrupt to the CPU module. The source for each of these interrupts is a function of the mode for which the ENI module is configured.

GEN Interrupt	IEEE 1284 Mode	ENI Host Mode
ENI Port 1	Port 1	FIFO Receiver
ENI Port 2	Port 2	FIFO Transmitter
ENI Port 3	Port 3	ENI Interrupt
ENI Port 4	Port 4	ENI Bus Error

Table 6-11: ENI Interrupt Sources

6.5.10 ENI Control Register

The ENI control register is a 32-bit read/write register which affects the operation of the ENI interface. Only the ARM processor can modify the ENI control register, it cannot be accessed through the ENI interface. The PSIO* and least significant 7 bits can be initialized during bootstrap by sampling the A7:A0 address inputs during reset.

Address = FFA0 0030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RESET:															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAIT			DMAE2	FAST	DPACK*	PSIO	KYOINT	INTP*	WR_OC	DINT2*	I_OC	DMAE*	IRQEN*	EPACK*	PULINT*
RESET:															
0			0	0	0	$\overline{\text{ADDR7}}$	0	1	ADDR6	ADDR5	ADDR4	ADDR3	1	ADDR1	ADDR0
R/W			R/W	R/W	R/W	R/W	R/W	R/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W

WAIT **ENI Added Wait States for Data Setup to PACK***

The WAIT field inserts additional wait states between when PDATA is made available and PACK* is asserted. This field can be used to help those environments requiring longer data settling time. This field extends the cycle time for both read and write cycles.

DMAE2 **Enable DMA Request through the PINT2 Pin**

0 = Disable

1 = Enable

The DMAE2 bit, when set, drives a single active high ENI FIFO mode DMA request signal out the PINT2 pin. The DMA request output is a logic-OR of the inbound and outbound DMA request signals.

This configuration mode should only be used when FIFO DMA is enabled in one direction or another, never both. The FIFO DMA enable configuration is controlled via the FIFO mode mask/ status register. When the DMAE2 configuration bit is set, the DINT2* configuration is ignored and the ENI interface cannot accept or issue an interrupt via the PINT2* pin. When the DMAE2 configuration bit is set, both PA14 and PA13 are returned to their address input function allowing for a maximum of 32K bytes of available shared RAM.

FAST **ENI FAST Timing Mode**

0 = Classic ENI Timing

1 = Fast ENI Timing

The ENI Fast mode bit can be set to improve performance in the ENI interface. The FAST mode configuration removes 2 SYSCLK clocks from the ENI interface access timing. Setting FAST to 1 requires the external ENI interface to drive PRW* valid before PCS* or PDACK* signals are

asserted. Setting FAST to 0 allows the external ENI interface to change PRW* slightly after PCS* or PDACK* are asserted.

DPACK* DMA PACK* Generation

0 = Drive PACK* active during DMA cycles

1 = Disable PACK* during DMA cycles

The DPACK* bit can be used to disable the assertion of PDACK* during DMA cycles. PDACK* should be disabled during DMA cycles when the external ENI interface is not designed to wait for PDACK* during DMA cycles.

PSIO PSIO Configuration for Shared Register

0 = Shared Register configured for NORMAL mode

1 = Shared Register configured for PSIO mode

The PSIO bit controls the bit formatting for the ENI shared register. Refer to Table 6-13. When the PSIO bit is set to 0, both the ARM processor and ENI interface use the same bit format for the shared register. When the PSIO bit is set to 1, the ENI Interface has an altered view of the ENI shared register as defined in Table 6-13. The default state for PSIO is established at bootstrap by sampling the A7 signal during reset. The inverted version of the A7 value is loaded into the PSIO configuration bit.

KYOINT Kyocera Interrupt Option

0 = Normal Interrupt Mode

1 = Kyocera Interrupt occurs when ENI interface writes to the most significant address of shared RAM.

The Kyocera interrupt option is an alternate mechanism for the ENI Interface to issue and interrupt to the ARM processor. When the KYOINT bit is set to 1, the ENI interface can issue an ARM interrupt by writing to the most significant word of shared RAM. The INTP* bit is set active low when the interrupt is issued. The KYOINT bit is a useful mechanism for generating an interrupt to the ARM processor when the ENI interface cannot address the shared register.

INTP* Interrupt Pending

The INTP* bit is set active low when the ENI Interface issues an interrupt to the ARM processor. The INTP* bit remains low until cleared by the ARM processor. The ARM processor clears the INTP* bit by writing a 1 to the same bit position in the ENI control register. An active low setting in INTP* causes an ARM processor interrupt when both IRQEN* (ENI control register) is set active low and ENI3 (GEN module interrupt enable register) is set active high.

This bit, when low, indicates that an interrupt from the ENI is pending. This bit is returned to the inactive 1 state by writing a 1 to this bit position.

WR_OC **Wait/Ready/Acknowledge Open Collector**

0 = Output Driver is “TTL,” used for LN14, MIO, Kyocera, others

1 = Output Driver is “OPEN COLLECTOR;” GCC, others

The WR_OC bit defines the driver personality for the PACK* signal. When set to 1, the PACK* signal is driven with an open-collector driver. When set to 0, PACK* is driven with a TTL driver. The default state for WR_OC is established at bootstrap by sampling the A6 signal during reset.

DINT2* **Disabled Interrupt 2 to ENI**

0 = INT2* pin can be used as a pulsed interrupt from the ENI, used for MIO, LN14, others

1 = ENI can select between PINT1* or PINT2* using the SINTP2 bit in the ENI shared register

The DINT2* bit controls the usage of the PINT2* signal. When DINT2* is set to 1, the PINT2* signal is an interrupt output. The ENI Interface selects either PINT1* or PINT2* using the SINTP2 bit in the ENI shared register.

When DINT2* is set to 0, the PINT2* signal is used as a pulse interrupt input from the ENI interface. An interrupt condition is latched on the low-to-high transition of the PINT2* input. The INTP* bit is set active low when the interrupt is issued. The pulsed interrupt input feature is a useful means for the ENI Interface to issue an interrupt to the ARM processor when the ENI Interface cannot address the ENI shared register. The default state for DINT2* is established at bootstrap by sampling the A5 signal during reset.

I_OC **Interrupt(s) Open Collector**

0 = Output Driver is “TTL,” used for LN14, MIO, Kyocera, others

1 = Output Driver is “OPEN COLLECTOR,” GCC, others

The I_OC bit defines the driver personality for the PINT1* and PINT2* output signals. When set to 1, the PINT1* and PINT2* output signals are driven with an open-collector driver. When set to 0, PINT1* and PINT2* output signals are driven with a TTL driver. The default state for I_OC is established at bootstrap by sampling the A4 signal during reset.

DMAE* **Enable ENI Interface FIFO Mode DMA Signals**

0 = Enable DACK*, DRQO*, and DRQI* for FIFO Mode

1 = Disable FIFO Mode DMA at the ENI interface

The DMAE* bit controls how the PA15, PA14, PA13, and PINT2* signals are used. The default state for DMAE* is established at bootstrap by sampling the A3 signal during reset.

When DMAE* is set to 0, the PA15, PA14, PA13, and PINT2* signals change personality to provide the DMA acknowledge and DMA request interface signals between the ENI FIFOs and the ENI Interface. The DMAE2 bit determines if the DMA Request signals are routed through PA14/PA13 or through PINT2*. When DMAE* and DMAE2 are both low, PA15, PA14, and PA13 are reallocated, causing the addressability of shared RAM to be limited to 8K bytes. When

DMAE* is low and DMAE2 is high, PA15 and PINT2* are reallocated limiting the addressability of shared RAM to 32Kbytes.

When DMAE* is set to 1, the ENI FIFO DMA interface signals are disabled; PA15, PA14, and PA13 are used for shared RAM addressing only; PINT2* is used for interrupts only.

IRQEN* **Enable Interrupt from ENI Interface**

0 = Interrupts from the ENI are enabled. An interrupt condition is setup when the ENI issues a Pulsed Interrupt via INT2* or the INTIO bit is set in the ENI shared register.

1 = Interrupts from the ENI are disabled.

The IRQEN* bit must be set active low to enable interrupts from the ENI Interface. An interrupt from the ENI interface can be set using the INTIOF bit in the ENI shared register, the Kyocera interrupt option, or the pulsed interrupt option on PINT2*. A pending interrupt is identified when the INTP* bit is active low in the ENI control register. The ENI3 bit (GEN module interrupt enable register) must also be set active high to generate an ARM processor interrupt.

EPACK* **Enable ENI ACK Pulse**

0 = The ACK pin is used as an active low data acknowledge signal; used for MIO, LN14, others

1 = The ACK pin is used as an active low WAIT signal; used for LN14, GCC, others

The EPACK* bit controls the personality of the PACK* output. The default state for EPACK* is established at bootstrap by sampling the A1 signal during reset.

When EPACK* is set to 0, the PACK* output is used as an active low data acknowledge signal. The PACK* signal is driven inactive high from the beginning of the ENI cycle until the end of the ENI cycle. At the end of the ENI cycle, the PACK* signal is driven active low.

When EPACK* is set to 1, the PACK* output is used as an active high data ready signal. The PACK* signal is immediately driven inactive low at the start of the ENI cycle and remains low until the end of the ENI cycle. At the end of the ENI cycle, the PACK* signal is driven active high to indicate “ready.”

PULINT* **Pulsed Interrupt Enable**

0 = Interrupts to the ENI are issued by writing to the pulsed interrupt register; used for MIO, LN14, others

1 = Interrupts are issued by setting the STSINT or VDAINT bits in the ENI shared register; used for GCC, others

The PULINT* bit controls how interrupts are issued from the ARM processor to the ENI interface. The PULINT* bit allows a pulsed interrupt signal to be driven out the PINT1* or PINT2* interrupt output signals. The default state for PULINT* is established at bootstrap by sampling the A0 signal during reset.

When PULINT* is set low, a write to the ENI pulsed interrupt register causes a 1us low-going pulse to be issued out the PINT1* pin. This interrupt mode is useful for those ENI interfaces that require an edge interrupt instead of a level interrupt.

When PULINT* is set high, interrupts to the ENI interface are solely controlled by the setting of the VDAINT and STSINT bits in the ENI shared register. When either of these bits are set, the PINT1* or PINT2* signal pins are driven active pins (controlled by the SINPT2 setting in the ENI shared register).

6.5.11 ENI Pulsed Interrupt Register

The ENI pulsed interrupt register contains no meaningful data. This address is simply used to issue an interrupt request to the ENI interface. An interrupt condition is setup for the ENI if this register is written to while the PULINT* bit in the ENI control register is set to its active low state.

When the NET+ARM chip is configured for MIO mode, reading this register provides the current status of the lower 8 bits of the PDATA pins. These pins can be used for various jumper settings. The 8 data bits are presented in the least significant 8 bits of this register.

6.5.12 ENI Shared RAM Address Register

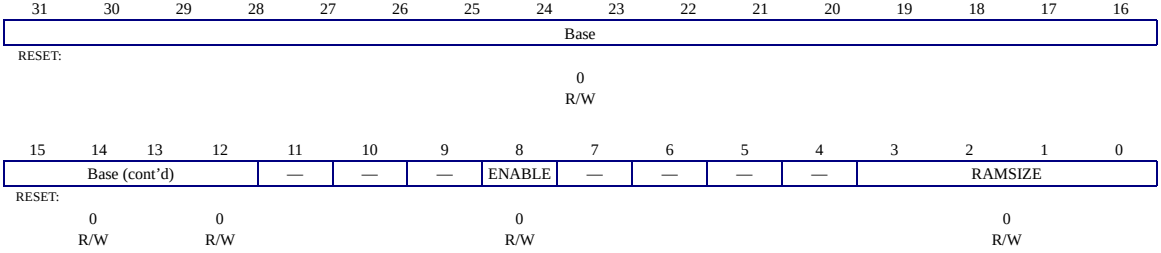
The ENI shared RAM address register is a 32-bit read/write register. The bits in this register are set to 0 upon reset. This register provides the base address and physical size of the shared memory window.

Shared memory is physically located somewhere in external RAM. The specified address offset must be on a multiple of the selected window size. The size of the shared RAM window can be configured from 4K to 64K bytes.

The base address value is used by the ENI controller module when addressing the shared memory block for the ENI interface. The effective address of the shared memory location is the sum of the base address and the index offset that is provided by the address inputs from the ENI interface. Any ENI address bits above what is specified to be the shared RAM physical size are ignored.

Shared RAM is only available to the ENI interface when the ENABLE bit is set to 1. The ENABLE bit must only be set after a valid BASE and RAMSIZE are configured. Before the ENABLE bit is set, any shared RAM access from the ENI interface is completed, however, write data is ignored and read data is returned with all zeros.

Address = FFA0 0038



BASE **Base Address of Shared RAM**

The BASE field defines the base location pointer for Shared RAM. Shared RAM resides in an external memory location whose base address is defined by this field. The BASE field provides the most significant 20-bits of the base address of Shared RAM. The actual value of the Shared RAM location can be defined as follow:

&(Shared RAM) = BASE << 12;

or

BASE = &(Shared RAM) >> 12;

Where:

& implies the “address of” operator.

>> implies the “shift right” operator.

<< implies the “shift left” operator.

ENABLE **Shared RAM Enable**

The ENABLE field must be set to 1 in order to allow the external ENI interface the ability to address Shared RAM. While the ENABLE bit is set to 0, the external ENI interface will be given a value of 0x0000 for all read operations to shared RAM. While the ENABLE bit is set to 0, writes to Shared RAM from the external ENI interface will be ignored.

RAMSIZE **Shared RAM Size**

0000: 64K

1000: 32K

1100: 16K

1110: 8K

1111: 4K

The amount of Shared RAM available to the external ENI interface is programmable between 4K and 64K bytes in size as defined above. All ENI address bits beyond the programmed size for Shared RAM will be ignored by the hardware. For example, address bit PA15 will be ignored when RAMSIZE is set to 32K (1110). Likewise, address bit PA15, PA14, PA13, and PA12 will be ignored when RAMSIZE is set to 4K (1111).

6.6 ENI Host Interface

The ENI host interface provides the interface between the NET+ARM chip and a separate processor subsystem. The ENI HOST interface acts as a slave memory device to the external processor subsystem. The ENI HOST provides both a shared RAM interface module and a byte-streaming FIFO module for inter-processor communications between the NET+ARM process and the external processor.

Figure 6-16 shows how to interface the NET+ARM ENI HOST interface to an external processor. The external processor is responsible for addressing the ENI HOST interface as a slave peripheral. The external processor activates the interface using the PCS* or PDACK* inputs. The NET+ARM

HOST interface services the request and responds with PACK* when the cycle is complete.

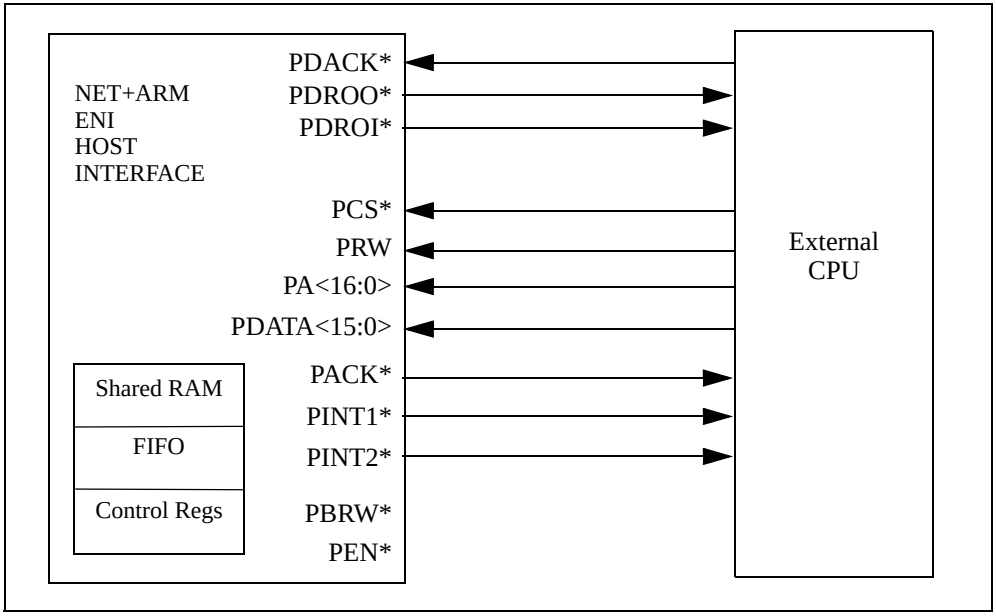


Figure 6-16: NET+ARM ENI HOST Conceptual View

6.6.1 Signal Description

PCS* **ENI Chip Select**

The PCS* signal provides the chip enable for the ENI interface from the external processor system. When PCS* is driven low, the ENI interface uses the PA address bus to determine which resource is being addressed. After the ENI interface cycle is complete, the ENI interface asserts the PACK* signal to complete the cycle. After PACK* is asserted active, the external processor must de-assert PCS* to complete the ENI host interface cycle.

PRW* **ENI Read/Write**

The PRW* is driven by the external processor system during an ENI host access cycle to indicate the data transfer direction. During normal PCS* cycles, PRW* high indicates a READ from the ENI host interface; PRW* low indicates a WRITE to the ENI host interface. During DMA cycles, PRW* high indicates a WRITE to the ENI host interface; PRW* low indicates a READ from the ENI host interface.

PA[16:0] **ENI Address Bus**

The ENI host interface requires 17 address signals to identify the resource being accessed. The PA bus must be valid while PCS* is low. During DMA cycles, the PA address bus is ignored. During DMA cycles, only the FIFO data register can be accessed.

PA16 low identifies an access to shared RAM. PA16 high identifies access to one of the ENI registers. Refer to Section *6.6 ENI Host Interface* for a detailed description of the registers available in the ENI host interface.

PDACK*/PA15 **ENI DMA Acknowledge**

When DMA FIFO operations are enabled, the PA15 input is used for the DMA acknowledge input PDACK*. DMA FIFO operations are enabled when the DMAE* bit in the ENI control register is set to 0.

The PDACK* input indicates a DMA cycle is in progress. Only the FIFO data register is accessed during a DMA cycle. During DMA cycles, the PCS* signal must not be asserted. During DMA cycles, the PA bus (other than PA15) is ignored. During DMA cycles, the PRW* defines the data transfer direction. PRW* high indicates a FIFO write operation while PRW* low indicates a FIFO read operation.

PDRQO*/PDRQO/PA14 **ENI Outbound FIFO DMA Request**

When DMA FIFO operations are enabled, the PA14 output is used for the active low outbound FIFO DMA ready output PDRQO*. The PDRQO* signal is only routed through PA14 when both DMAE* and DMAE2 in the ENI control register are set to 0. When DMAE* is set low and DMAE2 is set high, an active high version of PDRQO is routed out the active high PINT2 pin

instead. The PDRQO and PDRQI signals are internally “ORed” together before driving the PINT2 pin.

The PDRQO*/PDRQO signal is driven active to indicate the outbound FIFO has data available for reading. The PDRQO/PDRQI signal is only driven active when the outbound FIFO is not empty and the EDRDBUFRDY bit is set active high in the ENI FIFO mode mask/status register. The FIFO mode mask/status register is available in the ENI address space.

PDRQI*/PDRQI/PA13 ENI Inbound FIFO DMA Request

When DMA FIFO operations are enabled, the PA13 output is used for the active low inbound FIFO DMA ready output PDRQI*. The PDRQI* signal is only routed through PA13 when both DMAE* and DMAE2 in the ENI control register are both set to 0. When DMAE* is set low and DMAE2 is set high, an active high version of PDRQI is routed out the active high PINT2 pin instead. The PDRQO and PDRQI signals are internally “ORed” together before driving the PINT2 pin.

The PDRQI*/PDRQI signal is driven active to indicate the inbound FIFO has room available for writing. The PDRQI*/PDRQI signal is only driven active when the inbound FIFO is not full and the EDWRBUFEMP bit is set active high in the ENI FIFO mode mask/status register. The FIFO mode mask/status register is available in the ENI address space.

PDATA[15:0] ENI Data Bus

The ENI interface supports a 16-bit data bus and can be configured to operate in either 16-bit data mode or 8-bit data mode. When operating in 16-bit mode, all data bits are used, however, when operating in 8-bit mode only PDATA[15:8] are used.

PACK* ENI Acknowledge

The PACK* signal is driven by the ENI interface in response to an active low PCS* signal. The PACK* signal indicates the requested ENI bus cycle has been completed.

The PACK* signal can be configured to operate as either an active low acknowledge (ACK) indicator or an active high ready (RDY) indicator. This control is provided by the EPACK* bit in the ENI control register. The EPACK* bit can be automatically initialized to its proper state upon reset by including or not including a pull down resistor on the A1 signal (refer to section *6.5.1 ENI Module Hardware Initialization*). Figure 13-22, *ENI Shared RAM & Register Cycle Timing*, shows the difference between the ACK and RDY configuration for the PACK* signal.

The PACK* signal can be configured to operate as either a TTL or open-drain signal. The TTL configuration for the PACK* signal is provided by the WR_OC bit in the ENI control register. The WR_OC bit can be automatically initialized to its proper state upon reset by including or not including a pull down resistor on the A6 signal (refer to Section *6.5.1 ENI Module Hardware Initialization*).

PINT1* **ENI Interrupt 1**

The PINT1* signal is driven active low to indicate an interrupt condition to the external ENI processor. An interrupt condition is established when the ARM processor sets the STSINT or VDAINT bits in the ENI shared register. An interrupt condition is also established based upon the FIFO status and the FIFO interrupt enable bits in the FIFO mode mask/status register.

The PINT1* signal is further qualified by the EHWINT and SINTP2 bits in the ENI shared register. The ENI shared register is available in the ENI address space. The EHWINT provides global interrupt enable/disable control. The SINTP2 bit controls whether the interrupt is driven out the PINT1* pin or the PINT2* pin. The PINT1* is only driven active low when an interrupt condition is pending, the EHWINT bit is set to 1, and the SINTP2 bit is set to 0.

PINT2* **ENI Interrupt 2**

The PINT2* signal is driven active low to indicate an interrupt condition to the external ENI processor. An interrupt condition is established when the ARM processor sets the STSINT or VDAINT bits in the ENI shared register. An interrupt condition is also established based upon the FIFO status and the FIFO interrupt enable bits in the FIFO mode mask/status register.

The PINT2* signal is further qualified by the EHWINT and SINTP2 bits in the ENI shared register. The ENI shared register is available in the ENI address space. The EHWINT provides global interrupt enable/disable control. The SINTP2 bit controls whether the interrupt is driven out the PINT1* pin or the PINT2* pin. The PINT2* is only driven active low when an interrupt condition is pending, the EHWINT bit is set to 1, and the SINTP2 bit is set to 1.

The PINT2* signal can also be used to generate an interrupt from the external ENI Processor to the ARM processor. The PINT2* pin becomes an input when the DINT2* bit is set to 0 in the ENI control register. The DINT2* bit can be automatically initialized to its proper state upon reset by including or not including a pull down resistor on the A5 signal (refer to Section *6.5.1 ENI Module Hardware Initialization*). When DINT2* is configured as in the pulsed interrupt input, the low to high transition on the PINT2* input causes an interrupt condition to be established to the ARM processor. The low to high transition on the PINT2* input causes the INTIOF bit to be set in the shared register for the ARM processor.

The PINT2 signal can also be used to provide an active high DMA request when the ENI interface is configured to operate in FIFO mode and the DMAE2 bit is set in the ENI control register. In this condition, the inbound and outbound DMA ready signals are logically “ORed” together and routed through the PINT2 pin as an active high DMA request. When the DMAE2 bit is set, the PA14 and PA13 signals are no longer used as DMA request outputs and return to their shared RAM address function extending the amount of addressable shared RAM to 32K while operating in FIFO DMA mode.

PBRW* **ENI Data Buffer Read/Write**

The PBRW* signal is an output used to control the data direction pin for an external data bus transceiver. Some applications require a data bus buffer between the NET+ARM chip and the external ENI processor system. PBRW* high indicates a data transfer from the NET+ARM chip to the ENI processor; while PBRW* low indicates a data transfer from the ENI processor to the NET+ARM chip.

PEN* **ENI Data Buffer Enable**

The PEN* signal is an output used to control the output enable pin for an external data bus transceiver. Some applications require a data bus buffer between the NET+ARM chip and the external ENI processor system. PEN* low indicates a data transfer between the NET+ARM chip and the external ENI processor is in progress.

6.6.2 Memory Map

The ENI interface uses 17 address bits to access the shared RAM and various control registers. The address MAP from the external ENI perspective is defined as follows:

Address Range	A16	A2	A1	A0	Peripheral
00000 - 0FFFF	0	A2	A1	A0	Shared RAM
10000	1	0	0	0	Shared Register
10001	1	0	0	1	Clear Interrupts (Write Only)
10002	1	0	1	X	FIFO Mode Data Register
10004	1	1	0	0	FIFO Mode Mask/Status

Table 6-12: ENI Interface Address MAP

6.6.3 Shared RAM

The shared RAM is accessible from the ENI interface when the ENI controller is configured in ENI shared RAM or ENI FIFO mode. A total of 64Kbytes of shared RAM is nominally provided. When the interface is configured in FIFO mode and the ENI FIFO mode DMA interface is enabled in the ENI control register, a maximum of 8Kbytes of shared RAM is available (PA15, PA14, and PA13 are lost to the DMA control signals).

Each time the external ENI interface attempts to access Shared RAM, the PACK* signal will be delayed while the NET+ARM ENI shared RAM state machine arbitrates with the various internal NET+ARM bus masters (i.e., CPU and DMA) for access to external memory. Upon gaining access to the internal NET+ARM BBUS, the ENI shared RAM state machine will execute a memory read or memory write cycle, depending upon the state of the PRW* input, to a memory location defined by the BASE field located in the ENI Shared RAM Address Register (please refer to Section *6.5.12 ENI Shared RAM Address Register*).

After the NET+ARM memory subsystem completes the memory transfer for the ENI shared RAM state machine, the PACK* signal will be issued active to the external ENI interface along with read data during a read cycle. As such, the length of each and every Shared RAM access cycle from the external ENI interface can vary. The amount of time it takes to access Shared RAM from the external ENI interface is very dependent upon how the NET+ARM memory controller is configured, speed of various memory devices, DMA activity, bursting configurations, etc.

6.6.4 Shared Register

The NET+ARM chip supports an 8-bit shared register that allows the ENI interface some control over operation of the ENI interface and provides the NET+ARM chip with the means to interrupt the ENI. The NET+ARM chip accesses this register via the ENI controller configuration space. The ENI itself accesses this register through the ENI interface.

The shared register is accessible only when the ENI controller is configured to operate in ENI shared RAM or ENI FIFO mode.

The ENI Interface accesses the shared register by setting PA16 high. PA16 low accesses the shared RAM. When the ENI is configured to operate in 8-bit mode, the shared register is accessed using PDATA15 through PDATA8. When the ENI is configured to operate in 16-bit mode, the shared register is accessed using PDATA7 through PDATA0.

The VDAINT and STSINT bits in the shared register are set to 0 with a hardware or software reset from the local CPU. The other bits are set to 0 only by a hardware reset from the ENI interface reset (PRESET*).

The shared register can be configured to operate in either NORMAL mode or PSIO mode. The mode is configured using the PSIO bit in the ENI control register. Table 6-13 shows the two possible views for the shared register from the external ENI interface perspective. The register layout from the local CPU perspective always appears in NORMAL mode.

Unlike accessing Shared RAM, when accessing the Shared Register from the external ENI interface, the Shared Register access timing is always the same. The ENI hardware does not need to arbitrate with internal resources when accessing the Shared Register. As such, the Shared Register access timing is fast and consistent.

Data Bit	Normal Mode	PSIO Mode
D7	RSTIO	Reserved
D6	INTIOF	Reserved
D5	EMMINT	EMMINT
D4	EHWINT	RSTIO
D3	SINTP2	CLRINT
D2	VDAINT	EHWINT
D1	STSINT	STSINT
D0	CLRINT	INTIOF

Table 6-13: Shared Register Layout (ENI Interface Perspective Only)

CPU => FFA0 003C

ENI => 1 0000 ENI Shared Register (shown in NORMAL mode)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RESET:															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
—	—	—	—	—	—	—	—	RSTIO	INTIOF	EMMINT	EHWINT	SINTP2	VDAINT	STSINT	CLRINT
RESET:								0	0	0	0	0	0	0	0

RSTIO **ENI Reset NET+ARM Processor**

Only the external ENI interface can modify the RSTIO bit. The NET+ARM CPU cannot modify RSTIO. The external ENI interface can set the RSTIO bit to 1 in order to reset the NET+ARM processor and various peripherals. The NET+ARM processor and peripherals will remain in reset while RSTIO is set to 1. RSTIO must be set back to 0 to enable the NET+ARM CPU to reboot. The RSTIO bit is used by the external interface to reset the internal NET+ARM processor. The RSTIO bit does not reset either the ENI or MEM Modules within the NET+ARM architecture; the RSTIO bit resets all other internal NET+ARM modules.

The external ENI interface will read a 1 in the RSTIO bit position whenever the CLRINT bit in this register is also set.

INTIOF **ENI Interrupt NET+ARM Processor**

Only the external ENI interface can modify the INTIOF bit. The NET+ARM CPU cannot modify INTIOF directly. The external ENI interface can set the INTIOF bit to 1 in order to generate an interrupt to the NET+ARM CPU. When the INTIOF bit is set to 1, the INTP* bit in the ENI Control Register will be set to active low. The IRQEN* bit in the ENI Control Register can be used to allow the INTP* status condition to generate an interrupt to the NET+ARM CPU. Both the INTP* and INTIOF bits are returned to their inactive state when a 1 is written to the INTP* bit position in the ENI Control Register by the NET+ARM processor. The ENI INTIOF interrupt is routed to the GEN Module Interrupt Controller via the ENI Port 3 Interrupt (refer to Section *6.5.9 ENI Module Interrupts*).

EMMINT **ENI Spare Resource Bit**

Only the external ENI interface can modify the EMMINT bit. The NET+ARM CPU cannot modify EMMINT directly. The EMMINT bit controls no hardware within the NET+ARM. It can be used as a general-purpose control/status bit that can be controlled by the external ENI interface only.

EHWINT **ENI Enable Hardware Interrupt**

Only the external ENI interface can modify the EHWINT bit. The NET+ARM CPU cannot

modify EHWINT directly. The EHWINT bit must be set to 1 by the external ENI interface in order to allow either the PINT1* or PINT2* signals to generate an active low interrupt to the external ENI interface. Sources of interrupt for the external ENI interface include the STSINT and VDAINT bits in this register, the RDBUFRDY* and WRBUFEMP* bits in the FIFO Mask/Status Register, and a write by the NET+ARM CPU to the Pulsed Interrupt Register.

SINTP2 ENI PINT1*/PINT2* Interrupt Selection

0: PINT1*

1: PINT2*

Only the external ENI interface can modify the SINTP2 bit. The NET+ARM CPU cannot modify SINTP2 directly. The SINTP2 bit is controlled by the external ENI interface to determine which signal PINT1* or PINT2* will be used for interrupts to the external ENI interface.

VDAINT NET+ARM Interrupt to External ENI Interface

Only the internal ARM processor can modify the VDAINT bit. The external ENI interface cannot modify VDAINT directly. The VDAINT bit is set by the NET+ARM CPU to send an interrupt to the external ENI interface using either PINT1* or PINT2* depending upon the value defined in SINTP2. The EHWINT bit must also be set to active high for VDAINT to generate an interrupt to the external ENI interface. The VDAINT interrupt condition is cleared by the external ENI interface by setting the CLRINT bit to 1 then back to 0. The VDAINT can also be acknowledged by having the external ENI interface write to the Clear Interrupts memory mapped address location defined in Section *6.6.2 Memory Map*.

STSINT NET+ARM Interrupt to ENI

Only the internal ARM processor can modify the STSINT bit. The external ENI interface cannot modify STSINT directly. The STSINT bit is set by the NET+ARM CPU to send an interrupt to the external ENI interface using either PINT1* or PINT2* depending upon the value defined in SINTP2. The EHWINT bit must also be set to active high for STSINT to generate an interrupt to the external ENI interface. The STSINT interrupt condition is cleared by the external ENI interface by setting the CLRINT bit to 1 then back to 0. The STSINT can also be acknowledged by having the external ENI interface write to the Clear Interrupts memory mapped address location defined in Section *6.6.2 Memory Map*.

CLRINT ENI Clear Interrupt from NET+ARM

Only the external ENI interface can modify the CLRINT bit. The NET+ARM CPU cannot modify CLRINT directly. The CLRINT bit can be used by the external ENI interface to clear the VDAINT and STSINT interrupt conditions. The CLRINT bit must be set to 1 then back to 0 to clear these interrupt conditions.

6.6.5 Clear Interrupts

The clear interrupts command register is accessible from the ENI interface whenever the ENI controller is configured to operate in ENI shared RAM or ENI FIFO modes. The PA2 and PA1 address bits do not care when the ENI controller is configured in ENI shared RAM mode.

Unlike accessing Shared RAM, when writing to the Clear Interrupt location from the external ENI interface, the access timing is always the same. The ENI hardware does not need to arbitrate with internal resources when writing to the Clear Interrupt location. As such, the access timing is fast and consistent.

6.6.6 Kyocera Interrupts

When the KYOINT bit is set in the ENI control register, 2 additional command registers are available in the ENI interface address MAP. One address sets an interrupt condition from the ENI interface to the ARM processor, the other address clears an interrupt condition from the ARM processor to the ENI interface. These two address locations occupy the upper most two words of shared RAM.

The ENI interface uses 17 address bits to access the shared RAM and various control registers. The address MAP from the external ENI perspective is defined as follows:

Address Range	A16	A2	A1	A0	Peripheral
00000 - 0FFFB	0	X	X	X	Shared RAM
0FFFC - 0FFFD	0	1	0	X	Clear Interrupt to ENI
0FFFE - 0FFFF	0	1	1	X	Set Interrupt from ENI
10000	1	0	0	0	Shared Register
10001	1	0	0	1	Clear Interrupts (Write Only)
10002	1	0	1	X	FIFO Mode Data Register
10004	1	1	0	0	FIFO Mode Mask/Status

Table 6-14 - ENI Interface Address MAP

6.6.7 FIFO Data Register

The FIFO data register is only accessible from the ENI interface when the ENI controller is configured to operate in ENI FIFO mode.

The FIFO data register passes data between the NET+ARM chip and the ENI controller. The RDBUFRDY* and WRBUFEMP* flags in the mask/status register are used to identify the current state of the ENI FIFO mode data register. The FIFO Data Register must only be written when the WRBUFEMP* status bit in the FIFO Mask/Status Register is active low. The FIFO Data Register can only be read when the RDBUFRDY* status bit in the FIFO Mask/Status Register is active low.

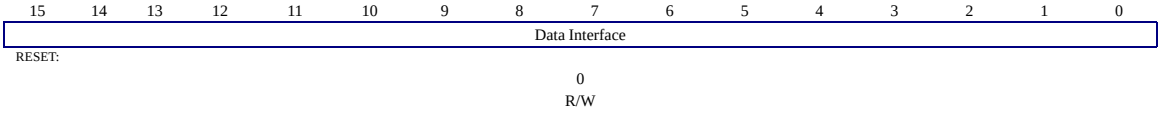
The data register can be memory mapped, using the PCS* select input, whose address is defined in Table 6-12. The memory mapped data register can be used in polled or interrupt-driven applications. The memory mapped data register can also be used in DMA applications that use the dual address, or memory-to-memory DMA operations. The dual address (or memory-to-memory) method requires two bus cycles per DMA operation; one bus cycle to access the data register and another bus cycle to access the RAM on the controller board.

The data register can also be DMA mapped using the PDACK* input. When using the PDACK* input, all ENI address inputs are ignored. Furthermore, the sense of PRW* is reversed. For DMA operations, PRW* high writes to the data register (read from controller memory), PRW* low reads from the data register (write to controller memory). The DMA mapped interface is designed for use in single address, or fly-by DMA operations. The single address (or fly-by) method requires a single bus cycle per DMA operation. The memory is simply accessed (read or write) on the controller. During DMA memory accesses, the PDACK* signal is asserted to the ENI interface. During DMA cycles, a PRW* high signal instructs the ENI interface to accept FIFO data from the controller, a PRW* low signal instructs the ENI interface to provide FIFO data to the controller.

The FIFO mode data register can be configured using an 8- or 16-bit interface. When configured for 8-bit operation, only the upper 8 data bits, PDATA(15:8), are used.

Unlike accessing Shared RAM, when accessing the FIFO Data Register from the external ENI interface, the access timing is always the same. The ENI hardware does not need to arbitrate with internal resources when accessing the FIFO Data Register. As such, the access timing for the FIFO Data Register is fast and consistent.

ENI => 1 0002



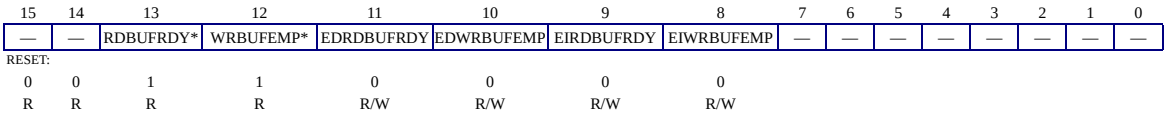
6.6.8 FIFO Mask/Status Register

The FIFO mode mask/status register is only accessible from the ENI interface when the ENI controller is configured to operate in ENI FIFO mode.

The FIFO mode mask/status register supports the data FIFO mode transfer mechanism between the NET+ARM chip and the ENI controller. The mask/status register can be written and read by the ENI controller. All bits are reset to their inactive state on power-up RESET.

Unlike accessing Shared RAM, when writing to the Clear Interrupt location from the external ENI interface, the access timing is always the same. The ENI hardware does not need to arbitrate with internal resources when writing to the Clear Interrupt location. As such, the access timing is fast and consistent.

ENI => 1 0004



RDBUFRDY* **FIFO Read Buffer Ready**

0 = Read Data Register is ready

1 = Read Data Register is empty

The RDBUFRDY* status bit will be active low when the FIFO had data available for the external ENI interface for reading. The external ENI interface must not read from the FIFO Data Register unless the RDBUFRDY* status bit is active low.

WRBUFEMP* **FIFO Write Buffer Empty**

0 = Write Data Register is empty

1 = Write Data Register is full

The WRBUFEMP* status bit will be active low when the FIFO has available room for the external ENI interface to write a new data word. The external ENI interface must not write to the FIFO Data Register unless the WRBUFEMP* status bit is active low.

EDRDBUFRDY **Enable DMA Read Buffer Ready**

0 = Mask Read Buffer Ready DMA Requests

1 = Enable Read Buffer Ready DMA

The EDRDBUFRDY bit can be set by the external ENI interface to enable an outbound FIFO DMA request to the external ENI interface. The outbound DMA request is active whenever the RDBUFRDY* status bit is active low. The outbound DMA request signal is routed out either PA14 or PINT2 depending upon the settings of DMAE* and DMAE2 fields in the ENI Control Register.

EDWRBUFEMP Enable DMA Write Buffer Empty

0 = Mask Write Buffer Empty DMA Requests

1 = Enable Write Buffer Empty DMA

The EDWRUFEMP bit can be set by the external ENI interface to enable an inbound FIFO DMA request to the external ENI interface. The inbound DMA request is active whenever the WRBUFEMP* status bit is active low. The inbound DMA request signal is routed out either PA13 or PINT2 depending upon the settings of DMAE* and DMAE2 fields in the ENI Control Register.

EIRDBUFRDY Enable Read Buffer Ready Interrupt

0 = Mask Read Buffer Ready Interrupt Requests

1 = Enable Read Buffer Ready Interrupts

The EIRDBUFRDY bit can be set by the external ENI interface to enable an interrupt request to the external ENI interface. The interrupt request is active whenever the RDBUFRDY* status bit is active low. The interrupt request signal is routed out either PINT1* or PINT2* depending upon the settings of SINTP2 fields in the Shared Register.

EIWRBUFEMP Enable Write Buffer Empty Interrupt

0 = Mask Write Buffer Empty Interrupt Requests

1 = Enable Write Buffer Empty Interrupts

The EDWRUFEMP bit can be set by the external ENI interface to enable an interrupt request to the external ENI interface. The interrupt request is active whenever the WRBUFEMP* status bit is active low. The interrupt request signal is routed out either PINT1* or PINT2* depending upon the settings of SINTP2 field in the Shared Register.

Chapter 7
Serial Controller Module

The NET+ARM chip supports two independent universal asynchronous/synchronous receiver/transmitter channels. Each channel supports the following features:

- Independent Programmable Bit-Rate Generator
- UART, HDLC, SPI Modes
- High Speed Data Transfer

  x1 Mode: 4Mbits/sec

  x16 Mode:230Kbits/sec
- 32-Byte TX FIFO
- 32-Byte RX FIFO
- Programmable Data Format

  5 to 8 Data Bits

  Odd, Even, or No Parity

  1, 2 Stop Bits
- Programmable Channel Modes

  Normal

  Local Loopback

  Remote Loopback
- Control Signal Support

  RTS, CTS, DTR, DSR, DCD, RI
- Maskable Interrupt Conditions

  Receive Break Detection

  Receive Framing Error

  Receive Parity Error

  Receive Overrun Error

  Receive FIFO Ready

  Receive FIFO Half-full

Transmit FIFO Ready

Transmit FIFO Half-empty

CTS, DSR, DCD, RI State Change Detection

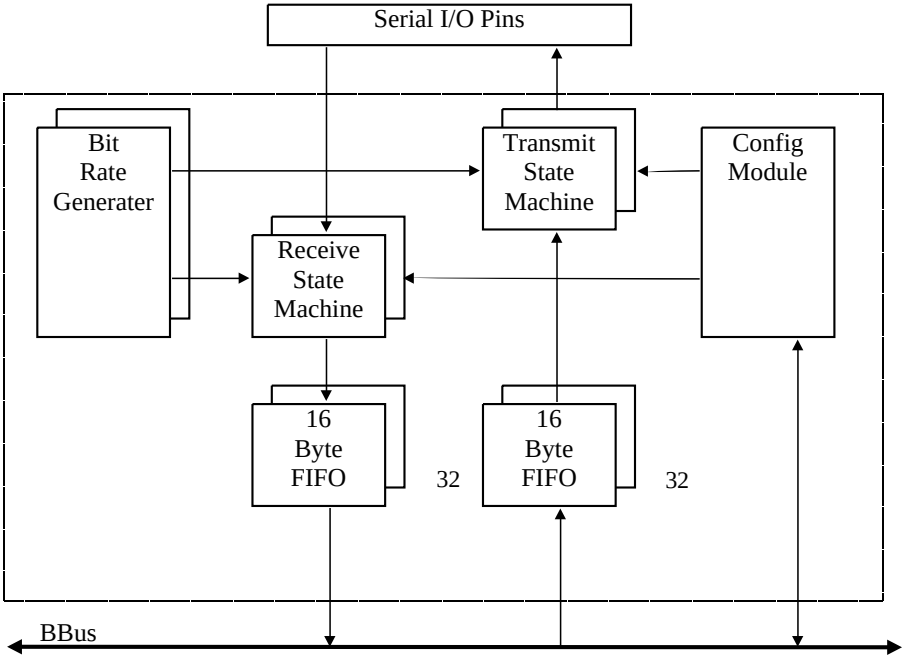


Figure 7-1: Serial Port Block Diagram

7.1 Bit-Rate Generator

Each serial channel supports an independent programmable bit-rate generator. The bit-rate generator runs both the transmitter and receiver of a given channel (no split speed support).

You can configure the bit-rate generator to use the external crystal, the external clock input or internal system timing as its timing reference. This allows for a wider range of possible bit-rates. For an example of the bit-rate configuration, refer to section *7.5.4 Serial Channel Bit-Rate Registers*.

7.2 Serial Protocols

7.2.1 UART Mode

Many applications require a simple mechanism for sending low-speed information between two pieces of equipment. The universal asynchronous receiver transmitter (UART) protocol is the de facto standard for simple serial communications. The term *asynchronous* is used since the protocol does not require the sending of clocking information between the two parties. Instead, the UART receiver operates using an over sampling technique to find the bit-level framing of the UART protocol.

Figure 7-2 shows the format for the UART framing protocol.

Start Bit	Data	Parity (Optional)	Stop Bit
0	5, 6, 7 or 8 bits	Odd or Even	1 or more

Figure 7-2: UART Framing Structure

Since the transmitter and receiver operate asynchronously, there is no need to connect transmit and receive clocks between them. Instead, the receiver over samples the receive data stream by a factor of 16. During synchronization, the receiver waits for the start bit. When it finds the high-to-low transition, the receiver counts 8 sample times and uses this point as the bit-center for all remaining bits in the UART frame. Each bit-time is 16 clock ticks apart.

When the UART is not transmitting data, it transmits a continuous stream of ones. This is referred to as the IDLE condition. When data transmission commences, the transmitter sends the start bit causing the receiver to go into action.

You can configure the UART to enable the transmitter using the CTS handshaking signal. In this mode, the transmitter cannot start a new UART data frame unless CTS is active. If CTS is dropped anywhere in the middle of a UART data frame, the current character is completed, however, the next character is *stalled*.

You can configure the UART to signal its receiver FIFO status using the RTS handshaking signal. When the receive FIFO has only four characters of available space, the RTS signal is dropped. The RTS and CTS pairs can be used for hardware flow control.

You can configure the UART to operate in synchronous timing mode. Using synchronous timing mode, the transmitter and receiver operate using a synchronous clock. Transmit and receive clocks must be connected between two end points. In synchronous mode, the receiver does not over sample the receive input data.

7.2.2 HDLC Mode

The HDLC protocol provides the data link layer for many WAN networking models such as Frame Relay, ISDN, and SDLC. The HDLC data frame is surrounded by a pair of flags (01111110) and a 16- or 32-bit CRC-CCITT checksum for error detection. The HDLC framing structure is defined as follows:

FLAG	Address	Control	Information	CRC	FLAG
01111110	8 or 16 bits	8 or 16 bits	Nx8 bits	16 or 32 bits	011111110

Figure 7-3: HDLC Framing Structure

The HDLC controller uses a zero insertion/deletion scheme in the data between flags. This process is more commonly referred to as bit-stuffing. When the transmitter encounters 5 consecutive ones in the data stream, it automatically inserts a zero. When the receiver encounters 5 consecutive ones followed by a zero, the zero is automatically removed. The bit-stuffing mechanism ensures that a 7E value in the data stream is not mistaken for a starting or ending flag.

Layer 2 frames can be transmitted over point-to-point links, broadcast networks, packet networks, or circuit switch networks. In many cases, the address field provides the frame's destination point. The address field is typically 8 or 16 bits. The SDLC and LAPB protocols use an 8-bit address while the LAPD protocol uses a 16-bit address field. The NET+ARM chip HDLC controller can detect four different 8-bit addresses and two different 16-bit addresses. Individual bits can be masked in the comparison function.

The 8-bit or 16-bit control field is typically used to define the frame type and flow control mechanism. The use of this field is protocol dependent. The NET+ARM chip HDLC controller offers no hardware support for handling of this field.

The data field contains the layer 3 protocol information.

The 16-bit or 32-bit CRC field provides error detection. The CRC is calculated on the entire frame including the address and control fields. The MSB of the CRC is transmitted first. For all other octets (address, control, data), the LSB is transmitted first.

The NET+ARM chip HDLC controller provides the following key features:

- Flag/Abort/Idle Generation and Detection
- Programmable Flags between Frames (0-15)
- Automatic Zero Bit-stuffing and Deletion
- 16-bit or 32-bit CRC-CCITT Generation and Checking
- Four Address Comparison Registers (both 8 and 16 bit addressing modes)
- Discarded Frame Counters (non-matching address and bad CRC)
- Detection of Frames that are Too Long
- Detection of Non-octet Aligned Frames
- Overrun/Underrun Detection

7.2.3 SPI Mode

The SPI controller provides a full-duplex, synchronous, character-oriented data channel between master and slave devices using a four-wire interface (TXD, RXD, CLK, SEL*). The master interface operates in a broadcast mode. The slave interface is activated using the SEL* signal. You can configure the master interface to address various slave interfaces using parallel I/O bits.

The transmitter and receiver use the same clock. When configured in master mode, the channel's BRG provides the timing reference.

The NET+ARM chip SPI controller provides the following key features:

- Four-wire Interface TXD, RXD, CLK, SEL*
- Master or Slave Configuration
- Programmable MSB/LSB Formatting

The SPI mode of operation is useful for providing simple parallel/serial data conversion to stream serial data between memory and a peripheral. The SPI port has no protocol associated with it other than it transfers information in multiples of 8-bits.

The SPI port simultaneously transmits and receives the same number of bytes. The transfer of information is controlled by a single clock signal. For the SPI master interface, the clock signal is an output. For the SPI slave interface, the clock signal is an input. Transfer of information is also qualified with an enable signal. The SPI enable signal must be active low for data transfer to occur, regardless of the SPI clock signal. The SPI enable function allows for multiple slaves to be individually addressed in a multi-drop configuration.

7.2.3.1 Limitations

The NETA15/40 are the first chips in the NET+ARM family to provide the SPI mode of operation.

The NETA15/40 SPI implementation has a few limitations, which make it difficult to use in certain applications. These limitations are described as follows.

1. When the master SPI port is transferring data, the master SPI port drives the SPI enable signal active low while transmitting characters. In between successive characters, the master SPI port drives the SPI enable signal inactive high for a short period of time. Although the SPI clock does not transition at this time, the inactive SPI enable can cause some applications difficulty.

   If the inactive high state of the SPI enable causes an application trouble, the simplest solution is to not use the PORTA4/PORTB4 signals as special function outputs, but instead as simple GPIO outputs. As GPIO outputs, the GPIO output signal can be set active low (via firmware) before the message is sent and returned to inactive high (via firmware) after the message transmission is complete. This guarantees that the SPI enable signal remains active low throughout the entire data transmission phase.

   Future versions of the NET+ARM chip family will be enhanced to ensure that the SPI enable remains active low between successive characters in a master SPI port transmit frame.

2. When the master SPI port is transferring data, the master SPI port controls the SPI clock signal. The master SPI port changes its TX data on the high to low transition of the SPI clock. The slave SPI port samples RX data on the low to high transition of the SPI clock. The master SPI port drives the SPI clock signal in the high state during the idle points between transmit bytes and transmit packets. Some applications have difficulty with the SPI clock being in the high state during the idle times.

   Future versions of the NET+ARM chip will be enhanced to allow for programmability of the SPI clock state during the idle times.

3. When the SPI (master or slave) is receiving information, receive data is moved into the receive FIFO when 4-bytes have been accumulated. This can present a problem when the SPI transmitter is sending a message whose length is not an integral number of 4-bytes. The FIFO logic uses the buffer GAP and character GAP timers to handle this situation. When either of the GAP counters expire, the residual receive data bytes are written to the receive FIFO. When SPI transmit data blocks are not an even multiple of four bytes in length, then either the buffer GAP or character GAP timers must be configured for operation to allow the receiver to operate properly.

Future versions of the NET+ARM chip will be enhanced to automatically update the receive FIFO (regardless of the residual data bytes) when the SPI transmitter has completed sending all of its information.

7.2.3.2 FIFO Management

Data flow between the SPI master/slave interfaces and memory occurs through the FIFO blocks within the SER module. Each serial port provides both a 32-byte transmit FIFO and a 32-byte receive FIFO. Both the transmit and receive FIFOs are memory mapped to the processor address space.

Transmit FIFO Interface

The processor can write either 1, 2, or 4 bytes at a time to the transmit FIFO. The number of bytes written is controlled by the data size defined by the ARM processor.

Using “C” terminology, the processor can write a BYTE to the transmit FIFO by using a byte pointer, for example, (char *)0xffd00010 = (char)data. The processor can write two bytes to the transmit FIFO using a short pointer, for example, (short *)0xffd00010 = (short)data. Finally, the processor can write four bytes to the FIFO using a standard long word pointer, for example, (int *)0xffd00010 = (int)data. Using assembler terminology, the processor can use STRB, STRH, or STR instructions to write a byte, half word, or long word respectively.

When the system is configured to operate in Big Endian mode, the most significant bytes in the word written to the FIFO are transmitted first. For example, the long word 0x11223344 would result in the character 0x11 being transmitted first, and 0x44 being transmitted last.

When the system is configured to operate in Little Endian mode, the least significant bytes in the word written to the FIFO are transmitted first. For example, the long word 0x11223344 would result in the character 0x44 being transmitted first, and 0x11 being transmitted last.

The transmit FIFO can be filled using processor interrupts or DMA. When not using DMA, the processor can write one long word (4 bytes) of data to the transmit FIFO whenever the TRDY bit in serial channel status register A is active high. If the THALF bit in serial channel status register A is active high, the processor can write four long words (16 bytes) of data to the transmit FIFO. To facilitate an interrupt when either the TRDY or THALF status bits are active, the processor can set one or both of the corresponding interrupt enables found in control register A (D03 and D02). The appropriate interrupt enable (SER 1 TX or SER 2 TX) bit in the GEN module interrupt enable register must also be set.

The transmit FIFO can also be filled using the DMA controller (DMA Channels 8 and 10). When using DMA, the processor need not interface with any of the serial port registers for data flow, instead, the processor must interface with the DMA channel registers and the DMA buffer descriptor block attached to DMA channels 8 and 10. To facilitate the use of transmit DMA, the ETXDMA bit in serial channel control register A must be set active high. When ETXDMA is set active high, the serial transmitter interrupts should be disabled.

Receive FIFO Interface

The receive FIFO presents up to 4 bytes of data at a time to the processor interface. The number of valid bytes found in the next read of the FIFO is defined by the information found in the RXFDB field in status register A. The same endian rules described for the transmit FIFO also apply for the receive FIFO. Whenever reading from the receive FIFO, the processor must perform a long word read operation. Each time a read cycle to the receive FIFO is performed, the receive FIFO advances to the next long word entry. The processor cannot read individual bytes from the same FIFO long word entry.

The receive FIFO can be emptied using processor interrupts or DMA. When not using DMA, the processor can read one long word (4 bytes) of data to the receive FIFO whenever the RRDY bit in serial channel status register A is active high. The long word read may have 1, 2, 3, or 4 bytes of valid data within the word. The number of valid bytes is determined by the bit encoding found in the RXFDB field in status register A. The RXFDB field must be read before the FIFO data register is read.

The RBC bit found in the status register A indicates that a receive data buffer has been closed and receiver status can be read from serial status register A. Before additional data can be read from the FIFO, the RBC bit must be acknowledged by writing a 1 to the same bit position in serial status register A.

The recommended process flow for the serial port receiver interrupt service routine is as follows:

1. Read serial status register A.
2. If RBC is True, then:
   a. Record receiver buffer closed status (if desired).
   b. Write a 1 to the RBC bit position in serial status A.
   c. Read serial status register A again.

3. If RRDY is True, then:
   a. Read the Data FIFO.
   b. Use the RXFDB field to pick out valid bytes.

To facilitate an interrupt when either the RRDY or RBC status bits are active, the processor must set one or both of the corresponding interrupt enables found in control register A (D11 and D09). The appropriate interrupt enable (SER 1 RX or SER 2 RX) bit in the GEN module interrupt enable register must also be set.

The receive FIFO can also be emptied using the DMA controller (DMA channels 7 and 9). When using DMA, the processor need not interface with any of the serial port registers for data flow, instead, the processor must interface with the DMA channel registers and the DMA buffer descriptor block attached to DMA channels 7 and 9. To facilitate the use of transmit DMA, the ERXDMA bit in serial channel control register A must be set active high. When ERXDMA is set active high, the serial receiver interrupts should be disabled.

7.2.3.3 SPI Master Mode

SPI master mode controls the flow of data between memory in the master SPI interface and an external SPI slave peripheral. The SPI master determines the numbers of bytes for transfer. The SPI master port simultaneously transmits and receives the same number of bytes. A single clock signal controls the transfer of information. For the SPI master interface, the clock signal is an output. Transfer of information is also qualified with an enable signal. The SPI master controls the SPI enable signal. The SPI enable signal must be active low for data transfer to occur, regardless of the SPI clock signal. The SPI enable function allows for multiple slaves to be individually addressed in a multi-drop configuration.

Configuration

The GEN module must be properly configured to allow the SPI interface signals from the SER module to interface with the PORTA/C GEN module GPIO pins.

SPI Channel A
PORTA7 - Configure as Special Function Output (SPI TXD)
PORTA3 - Configure as Special Function Input (SPI RXD)
PORTA4 - Configure as Special Function Output (SPI Enable)
PORTC7 - Configure as Special Function Output (SPI Clock)

SPI Channel B
PORTB7 - Configure as Special Function Output (SPI TXD)
PORTB3 - Configure as Special Function Input (SPI RXD)
PORTB4 - Configure as Special Function Output (SPI Enable)
PORTC5 - Configure as Special Function Output (SPI Clock)

The SER module must be properly configured to operate in either master or slave mode. For master mode operation, the MODE field in control register B must be set to a value of “10” before the CE field in control register A is set to 1.

The following is the suggested ordering of configuration for SPI master mode of operation.

1. Reset the serial port by writing a 0 to control register A.
2. Configure the bit-rate register:

   EBIT- 1 for enable
   TMODE- 1 for 1X mode
   RXSRC- 0 for internal
   TXSRC- 0 for internal
   RXEXT- 0 for disable
   TXEXT- 1 for enable
   CLKMUX- user defined
   TXCINV- 0 for normal
   RXCINV- 0 for normal
   N - user defined
3. Configure the buffer GAP timer.

   The buffer GAP timer needs to be configured to allow any residual receive data bytes to be written into the receive FIFO. Either the buffer GAP timer or character GAP timer can be used. Using one of the GAP timers is only required when the size of the transmit data block is not a multiple of four bytes. If the size of the transmit data block is a multiple of 4, then the GAP timers are not required.

   TRUN- 1 for enable
   CT - user defined
4. Configure the character GAP timer.

   The character GAP timer needs to be configured to allow any residual receive data bytes to be written into the receive FIFO. Either the buffer GAP timer or

character GAP timer can be used. Using one of the GAP timers is only required when the size of the transmit data block is not a multiple of four bytes. If the size of the transmit data block is a multiple of 4, then the GAP timers are not required.

TRUN- 1 for enable
CT- user defined

5. Configure control register B:

   RBGT- 1 To enable the buffer GAP timer
   RCGT- 1 To enable the character GAP timer
   MODE - “10” for master mode
   BITORDR- user defined

6. Configure control register A:

   CE - 1 for enable
   WLS - “11” for 8-bit operation

Transfer Waveform

Figure 7-4 shows the waveforms for the SPI master mode of operation.

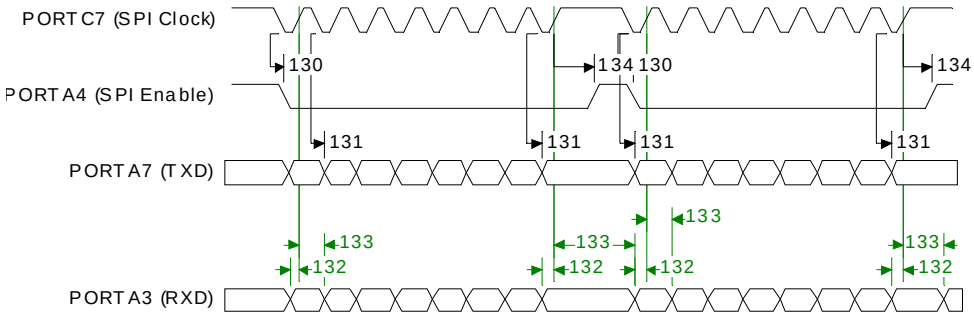


Figure 7-4: SPI Master Mode Waveform

Num	Characteristic	Min	Max	Unit
130	SPI Clock Low to SPI Enable Low	$-T_{SYS}$	T_{SYS}	ns
131	SPI Clock low to TXD valid	$-T_{SYS}$	T_{SYS}	ns
132	RXD Input Valid to SPI Clock High (setup)	20		ns
133	SPI Clock High to RXD Input Change (hold)	0		ns
134	SPI Clock High to SPI Enable High	½ Bit-Time		ns

Table 7-1: SPI Master Timing

T_{SYS} refers to the period of the NET+ARM chip system clock. If the NET+ARM chip is operating at 33Mhz, then T_{SYS} is equal to 30ns. ½ bit-time is a function of how fast the SPI port is configured to operate at. If the SPI port is configured to operate at its maximum speed of 4Mbps, then ½ bit-time equals 120ns.

SPI Master Transmitter

The SPI master transmitter changes its TXD output on the falling edge of the SPI clock signal while the SPI enable signal is driven active low. The SPI slave devices are expected to sample data in the rising edge of the SPI clock signal.

The SPI master transmitter drives the SPI enable signal active low from the falling edge of the SPI clock for the first bit of a byte being transmitted. The SPI master transmitter drives the SPI enable signal inactive high after the rising edge of the SPI clock signal during the eighth bit of the byte currently transmitted.

The SPI master transmitter drives the SPI enable signal active low to identify when data is being transmitted. With the NETA15/40 chips, the SPI enable pulses inactive high between successive transmit characters. (Future versions of the NET+ARM chip will remove the inactive pulses of SPI enable between successive transmit characters.) The SPI clock signal never transitions from low to high while the SPI enable signal is inactive high.

The SPI master transmitter transmits bytes whenever data is available in the TX FIFO. When the TX FIFO becomes empty, the SPI enable signal is driven inactive high until more data is available in the TX FIFO.

SPI Master Receiver

The SPI master receiver samples the RXD input on the rising edge of the SPI clock signal while the SPI enable signal is driven active low.

The SPI master receiver receives one byte of inbound data for each byte of transmit data sent. The SPI master receiver is therefore constrained by the amount of data being transmitted. The SPI master receiver cannot receive more data than what is transmitted.

Whenever the SPI master receiver collects four bytes, those four bytes are written to the RX FIFO. Receive data is read by the CPU or DMA controller from the other side of the FIFO. If the SPI master transmitter always sends data in multiples of four bytes, the SPI master receiver operates smoothly without any restrictions. However, when the master SPI transmitter sends an odd number of bytes, the SPI master receiver can be in a situation where there is an odd number of bytes (either 1, 2, or 3 bytes) waiting for the fourth byte before insertion into the FIFO. This can result in stale data sitting in the SPI master receiver. To commit these residual bytes to the RX FIFO, the buffer and/or character GAP timers must be used. When either timer expires, any residual RX data bytes are immediately written to the RX FIFO. Note that there will be some delay in the process of writing the final residual bytes. The amount of delay is determined by the configuration of the buffer and character GAP timers.

Future versions of the NET+ARM chip will remove the requirement of needing the GAP timers by automatically writing residual bytes when the SPI master transmitter is done transmitting.

7.2.3.4 SPI Slave Mode

SPI slave mode supports the peripheral side of a SPI interface. The SPI master controls the numbers of bytes for transfer. The SPI slave port simultaneously transmits and receives the same number of bytes. The transfer of information is controlled by a single clock signal. For the SPI slave interface, the clock signal is an input. Transfer of information is also qualified with an enable signal input. The SPI master controls the SPI enable signal. The SPI enable signal must be active low for data transfer to occur, regardless of the SPI clock signal. The SPI enable function allows for multiple slaves to be individually addressed in a multi-drop configuration

.

Configuration

The GEN module must be properly configured to allow the SPI interface signals from the SER module to interface with the PORTA/C GEN module GPIO pins.

SPI Channel A
PORTA7 - Configure as Special Function Output (SPI TXD)
PORTA3 - Configure as Special Function Input (SPI RXD)
PORTA4 - Configure as Special Function Input (SPI Clock)
PORTC7 - Configure as Special Function Input (SPI Enable)

SPI Channel B
PORTB7 - Configure as Special Function Output (SPI TXD)
PORTB3 - Configure as Special Function Input (SPI RXD)
PORTB4 - Configure as Special Function Input (SPI Clock)
PORTC5 - Configure as Special Function Input (SPI Enable)

The SER module must be properly configured to operate in either master or slave mode. For slave mode operation, the MODE field in control register B must be set to a value of “11” before the CE field in control register A is set to 1.

The following is the suggested ordering of configuration for SPI slave mode of operation.

1. Reset the serial port by writing a 0 to control register A.
2. Configure the bit-rate register:

   EBIT- 1 for enable
   TMODE- 1 for 1X mode
   RXSRC- 1 for external
   TXSRC- 1 for external
   RXEXT- 0 for disable
   TXEXT- 0 for disable
   CLKMUX- N/A (external timing source)
   TXCINV- 0 for normal
   RXCINV - 0 for normal
   N - N/A (external timing source)
3. Configure the buffer GAP timer:

The buffer GAP timer needs to be configured to allow any residual receive data bytes to be written into the receive FIFO. Either the buffer GAP timer or character GAP timer can be used. Using one of the GAP timers is only required when the size of the transmit data block is not a multiple of four bytes. If the size of the transmit data block is a multiple of 4, then the GAP timers are not required.

TRUN - 1 for enable
CT - user defined

4. Configure the Character GAP Timer:

   The character GAP timer needs to be configured to allow any residual receive data bytes to be written into the receive FIFO. Either the buffer GAP timer or character GAP timer can be used. Using one of the GAP timers is only required when the size of the transmit data block is not a multiple of four bytes. If the size of the transmit data block is a multiple of 4, then the GAP timers are not required.

   TRUN- 1 for enable
   CT - user defined

5. Configure control register B:

   RBGT- 1 To enable the buffer GAP timer
   RCGT- 1 To enable the character GAP timer
   MODE - “11” for slave mode
   BITORDR- user defined

6. Configure control register A:

   CE - 1 for enable
   WLS - “11” for 8-bit operation

Transfer Waveform

Figure 7-5 shows the waveforms for the SPI slave mode of operation.

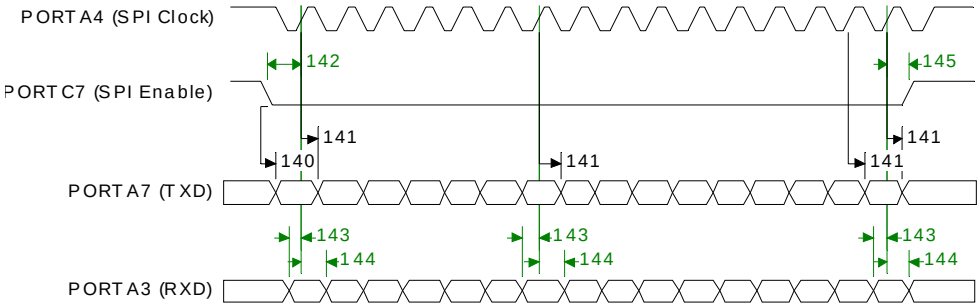


Figure 7-5: SPI Slave Mode Waveform

Num	Characteristic	Min	Max	Unit
140	SPI Enable Low to SPI Clock High	1/2 bit time		
141	SPI Clock High to TXD valid	3 * T_{SYS}	4 * T_{SYS}	ns
142	SPI Enable low to SPI Clock High (setup)	0		ns
143	RXD Input Valid to SPI Clock High (setup)	0		ns
144	SPI Clock High to RXD Input Change (hold)		4 * T_{SYS}	ns
145	SPI Clock High to SPI Enable High (hold)	4 * T_{SYS}		ns

Table 7-2: SPI Slave Timing

T_{SYS} refers to the period of the NET+ARM chip system clock. If the NET+ARM chip is operating at 33Mhz, then T_{SYS} is equal to 30ns. ½ bit-time is a function of how fast the SPI port is configured to operate at. If the SPI port is configured to operate at its maximum speed of 4Mbps, then ½ bit-time is equal to 120ns.

SPI Slave Transmitter

The SPI slave transmitter always has the first bit ready for transmission before the SPI enable signal is activated by the SPI master. On the rising edge of SPI clock while the SPI enable signal is active low, the SPI slave transmitter changes the TXD output to the next data bit for transmission. The SPI slave transmitter goes through a sampling of the SPI clock input, as such, the output changes 3 to 4 internal system clock ticks from the rising edge of the SPI clock input. The SPI master receiver does not sample the changing data until the next rising edge of SPI clock providing plenty of setup and hold time for the SPI master interface.

The SPI slave transmitter continues to change TXD data bits on the rising edge of SPI clock until the SPI enable signal is driven inactive high. While the SPI enable signal is inactive high, the TXD output remains constant.

SPI Slave Receiver

The SPI slave receiver samples the RXD input on the rising edge of the SPI clock signal while the SPI enable signal is driven active low. The SPI slave receiver actually goes through a sampling of the SPI clock input, as such, the input is actually sampled 3 to 4 internal system clock ticks from the rising edge of the SPI clock input.

Whenever the SPI slave receiver collects four bytes, those four bytes are written to the RX FIFO. Receive data is read by the CPU or DMA controller from the other side of the FIFO. If the SPI master transmitter always sends data in multiples of four bytes, the SPI slave receiver operates smoothly without any restrictions. However, when the master SPI transmitter sends an odd number of bytes, the SPI slave receiver can be in a situation where there is an odd number of bytes (either 1, 2, or 3 bytes) waiting for the fourth byte before insertion into the FIFO. This can result in stale data sitting in the SPI slave receiver. To commit these residual bytes to the RX FIFO, the buffer and/or character GAP timers must be used. When either timer expires, any residual RX data bytes are immediately written to the RX FIFO. Note there is some delay in the process of writing the final residual bytes. The amount of delay is determined by the configuration of the buffer and character GAP timers.

Future versions of the NET+ARM chip will remove the requirement of needing the GAP timers by automatically writing residual bytes when the SPI master transmitter is done transmitting.

7.3 General Purpose I/O Configurations

The GEN module provides the physical layer connections for NMSI (Non Multiplexed Serial Interface) and SPI interfaces. The NMSI interface is used for UART and HDLC protocols. PORTA provides the NMSI and SPI interface for serial channel A. PORTB provides the NMSI and SPI interface for serial channel B. Four PORTC signals are also required to support the NMSI and SPI interfaces. Each GEN Module interface signal can be enabled/disabled under firmware control.

The NMSI interface requires a minimum of two signals (TXD and RXD), however, each NMSI interface can be configured to use up to 10 signals. When serial channels A or B are configured to operate in NMSI mode, the GEN module interface signals are allocated as shown in Table 7-3.

PORTA7	TXDA	PORTB7	TXDB
PORTA6	DTRA*	PORTB6	DTRB*
PORTA5	RTSA*	PORTB5	RTSB*
PORTA4	RXCA/OUT1A*	PORTB4	RXCB/OUT1B*
PORTA3	RXDA	PORTB3	RXDB
PORTA2	DSRA*	PORTB2	DSRB*
PORTA1	CTSA*	PORTB1	CTSB*
PORTA0	DCDA*	PORTB0	DCDB*
PORTC7	TXCA/OUT2A*	PORTC5	TXCB/OUT2B*
PORTC6	RIA*	PORTC4	RIB*

Table 7-3: PORTA/B/C Assignments for NMSI Interface

The SPI mode supports a four-wire interface. When serial channels A or B are configured to operate in SPI mode, the GEN module interface signals are configured as follows:

PORTA7	TXDA	PORTB7	TXDB
PORTA3	RXDA	PORTB3	RXDB
PORTC7	SPICLKA	PORTC5	SPICLKB
PORTA4	SPISELA*	PORTB4	SPISELB*

Table 7-4: PORTA/B/C Assignments for SPI Master Mode

PORTA7	TXDA	PORTB7	TXDB
PORTA3	RXDA	PORTB3	RXDB
PORTA4	SPICLKA	PORTB4	SPICLKB
PORTC7	SPISELA*	PORTC5	SPISELB*

Table 7-5: PORTA/B/C Assignments for SPI Slave Mode

7.4 Serial Port Performance

The serial ports have a finite performance limit on their ability to handle various serial protocols. The performance is limited by the speed of the SYSCLK operating the NET+ARM chip. The configured speed for the internal PLL defines the SYSCLK rate (refer to Section *8.2.3 PLL Control Register* for details).

Operating Mode	Serial Port Maximum Rate
UART (x16)	SYSCLK/64
UART (x1)	SYSCLK/4
SPI	SYSCLK/8
HDLC	SYSCLK/10

Table 7-6: Serial Port Performance

7.5 Configuration

The serial controller module has a block of configuration space that is mapped into the SER module configuration space as defined in Table 2-1, *BBus Address Decoding*.

Address	Register
FFD0 0000	Channel 1 Control Register A
FFD0 0004	Channel 1 Control Register B
FFD0 0008	Channel 1 Status Register A
FFD0 000C	Channel 1 Bit-Rate Register
FFD0 0010	Channel 1 FIFO Data Register
FFD0 0014	Channel 1 Receive Buffer Timer
FFD0 0018	Channel 1 Receive Character Timer
FFD0 001C	Channel 1 Receive Match Register
FFD0 0020	Channel 1 Receive Match Mask Register
FFD0 0024	Channel 1 Control Register C
FFD0 0028	Channel 1 Status Register B
FFD0 002C	Channel 1 Status Register C
FFD0 0030	Channel 1 FIFO Data Last Register
FFD0 0040	Channel 2 Control Register A
FFD0 0044	Channel 2 Control Register B
FFD0 0048	Channel 2 Status Register A
FFD0 004C	Channel 2 Bit-Rate Register
FFD0 0050	Channel 2 FIFO Data Register
FFD0 0054	Channel 2 Receive Buffer Timer

Table 7-7: Serial Channel Configuration Registers

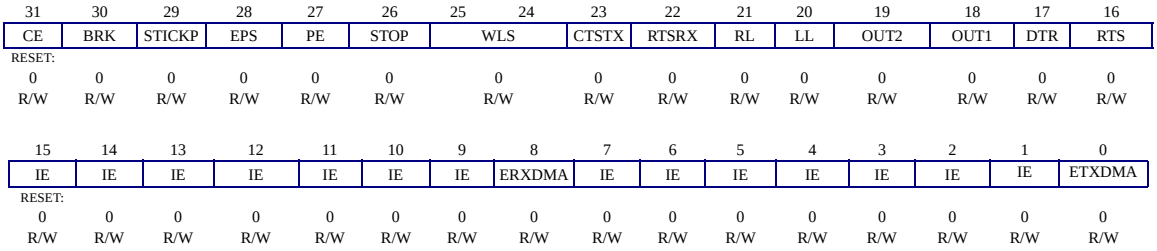
Address	Register
FFD0 0058	Channel 2 Receive Character Timer
FFD0 005C	Channel 2 Receive Match Register
FFD0 0060	Channel 2 Receive Match Mask Register
FFD0 0064	Channel 2 Control Register C
FFD0 0068	Channel 2 Status Register B
FFD0 006C	Channel 2 Status Register C
FFD0 0070	Channel 2 FIFO Data Last Register

Table 7-7: Serial Channel Configuration Registers (Continued)

7.5.1 Serial Channel Control Register A

All control bits are active high unless an asterisk (*) appears in the signal name, in which case, they are active low. All control bits are set to their respective inactive state on reset.

Address = FFD0 0000 / 40



CE

The CE field enables/disables the serial channel. Setting the CE field to 0 resets the port and the data FIFOs. Setting the CE to 1 enables serial channel operation. At a minimum, the CE field cannot be set until the MODE field is configured in control register B.

BRK

The BRK field forces a break condition in UART mode. While BRK is set to 1, the UART transmitter outputs a logic 0 or a space condition, on the TXD output signal.

STICKP

The STICKP field can be used to force the UART Parity field to a certain state, as defined by the EPS field, instead of a parity bit calculated against the data word. STICKP only applies when the PE field is also set to 1. Setting the STICKP field to 1 forces transmission of the static parity value.

EPS

The EPS field determines if the serial channel uses odd or even parity when calculating the parity bit in UART mode. When the STICKP field is set, the EPS field defines the static state for the parity bit.

0 - Odd Parity

1 - Even Parity

PE

When the PE field is set, parity is enabled for the UART transmitter and receiver. The transmitter generates proper parity. The receiver checks for proper parity. If the receiver encounters a bad parity bit, the RPE field is set in serial channel status register A.

STOP

The STOP field determines the number of stop bits in each UART transmitter.

0 - 1 stop bit

1 - 1.5 stop bits for 5-bit word, 2 stop bits for others

WLS

The WLS field determines the number of data bits in each UART data word.

“00” - 5 bits

“01” - 6 bits

“10” - 7 bits

“11” - 8 bits

CTSTX

The CTSTX field supports hardware handshaking. When CTSTX is set, the transmitter only operates when the external CTS signal is in the active state. An external device can therefore use CTS to temporarily stall data transmission.

RTSRX

The RTSRX field supports hardware handshaking. When RTSRX is set, the RTS output provides the receiver FIFO almost-full condition. When the receiver FIFO

backs up, the RTS signal is dropped. The RTS output stalls an external transmitter from delivering data.

RL

The RL field provides a remote loopback feature. When the RL field is set to 1, the TXD transmit output is connected to the RXD receive input. The RL field immediately echoes receive data back as transmit data. The RL field is primarily used as a test vehicle for external data equipment.

LL

The local loopback (LL) field provides an internal local loopback feature. When the LL field is set to 1, the internal receive data stream is connected to the TXD output signal. The LL field connects the serial channel receiver directly to the serial channel transmitter. The LL field is primarily used as a test vehicle for the serial channel driver firmware.

OUT1/OUT2

The OUT1 and OUT2 fields are used to provide extra miscellaneous serial channel control signal outputs. The OUT1 and OUT2 signals can be routed through PORTA/B/C pins using the special function mode of the PORTA/B/C ports.

DTR

The data terminal ready (DTR) field controls the state of the external data terminal ready signal. Setting DTR to 1 causes the DTR output to go active. Setting DTR to 0 causes the DTR output to go inactive.

RTS

The request to send (RTS) field controls the state of the external request to send signal. Setting RTS to 1 causes the RTS output to go active. Setting RTS to 0 causes the RTS output to go inactive.

IE

The various interrupt enable fields are used to enable an interrupt to occur when the respective status bit is set in serial status register A. Setting the IE field to a 1 enables the interrupt, a 0 disables it.

Bits D15:D04 constitute a Receiver Interrupt Condition

D15	R/W	IE	Receive Break Interrupt Enable
D14	R/W	IE	Receive Framing Error Interrupt Enable
D13	R/W	IE	Receive Parity Error Interrupt Enable
D12	R/W	IE	Receive Overrun Interrupt Enable

D11	R/W	IE	Receive Register Ready Interrupt Enable
D10	R/W	IE	Receive FIFO Half-Full Interrupt Enable
D09	R/W	IE	Receive Buffer Closed Interrupt Enable
D08	R/W	ERXDMA	Enable Receive DMA requests (use to pause receiver)
D07	R/W	IE	Change in DCD Interrupt Enable
D06	R/W	IE	Change in RI Interrupt Enable
D05	R/W	IE	Change in DSR Interrupt Enable
Bits D4:D0 constitute a Transmitter Interrupt Condition			
D04	R/W	IE	Change in CTS Interrupt Enable
D03	R/W	IE	Transmit Register Empty Interrupt Enable
D02	R/W	IE	Transmit FIFO Half-Empty Interrupt Enable
D01	R/W	IE	Transmit Buffer Closed Interrupt Enable
D00	R/W	ETXDMA	Enable Transmit DMA requests (use to pause transmitter)

ERXDMA

The ERXDMA field enables the receiver to interact with a DMA channel. When configured to operate in DMA mode, the DMA controller empties the receive data FIFO and delivers the data to memory. The receive status information from status registers B and C automatically moves to the receive DMA buffer descriptor.

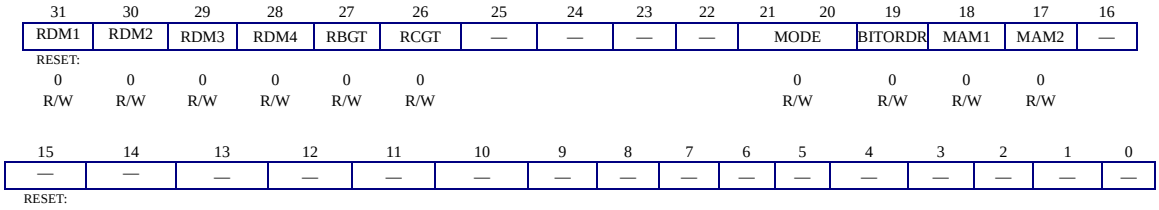
ETXDMA

The ETXDMA field enables the transmitter to interact with a DMA channel. When configured to operate in DMA mode, the DMA controller loads the transmit data FIFO from memory. The transmit status information from status register C moves automatically to the transmit DMA buffer descriptor.

7.5.2 Serial Channel Control Register B

All control bits are active high unless an asterisk (*) appears in the signal name, in which case, they are active low. All control bits are set to their respective inactive state on reset.

Address = FF80 0010



RDM1/2/3/4

When the serial channel is configured to operate in UART mode, the RDM bits are used to enable the receive data match comparators. A receive data match comparison detection can be used to close the current receive buffer descriptor. The last byte in the current receive data buffer contains the match character. Each of these bits enables the respective byte found in the receive match register.

When the serial channel is configured to operate in HDLC mode, the RDM bits are used to enable the address match comparators. The first character (or two characters) in the HDLC frame is compared to the byte values found in the receive match register. A match enables the HDLC receive packet to be received. By setting all RDM bits to the 0 state, the HDLC receiver effectively receives all HDLC frames regardless of an address match. The address match comparators can be configured as four 8-bit address values or as two 16-bit address values using the MAM1 and MAM2 field settings.When the MAM1 field is set to 1, the RDM1 field is used and the RDM2 field is ignored. When the MAM2 field is set to 1, the RDM3 field is used and the RDM4 field is ignored.

RBGT

The receive buffer gap timer (RBGT) field enables the receive buffer gap timer. When the RBGT field is set to 1, the BGAP field in serial status register A is set when the time out value defined in the RBGT register has expired. This RBGT field detects the maximum allowed time from when the first byte is placed into the receive data buffer and when the receive data buffer is closed.

RCGT

The receive character gap timer (RCGT) field enables the receive character gap timer. When the RCGT field is set to 1, the CGAP field in serial status register A is set when the time out value defined in the receive buffer character timer register has expired.

This RCGT field detects the maximum allowed time from when the last byte is placed into the receive data buffer and when the receive data buffer is closed.

MODE

The MODE field configures the serial channel to operate in UART, HDLC, or SPI modes. The MODE field must be established before the CE bit is set to 1 in serial channel control register A.

“00” - UART Mode

“01” - HDLC Mode

“10” - SPI Master Mode

“11” - SPI slave Mode

BITORDR

The BITORDR field controls the order in which bits are transmitted and received in the serial shift register. The BITORDR field applies to all modes, UART, HDLC, and SPI. When the BITORDR field is set to 0, the bits are processed LSB first, MSB last. When the BITORDR field is set to 1, the bits are processed MSB first, LSB last. In HDLC mode, the processing of the CRC bit field is an exception. The HDLC CRC bits are always processed MSB first, LSB last.

0 - Normal; Transmit/Receive LSB first

1 - Reverse; Transmit/Receive MSB first

MAM1/MAM2

The MAM1 and MAM2 fields are used to combine together two data bytes in the receive match register to form a single 16-bit HDLC address match value. Setting the MAM1 field to 1 causes RDMB1 and RDMB2 in the receive match register to be combined to form a single 16-bit address match value. Setting the MAM2 field to 1 causes RDMB3 and RDMB4 in the receive match register to be combined to form a single 16-bit address match value.

MAM1:

0 - Match1/Match2 each 8-bit address

1 - Match1/Match2 combined for 16-bit address

MAM2:

0 - Match3/Match4 each 8-bit address

1 - Match3/Match4 combined for 16-bit address

7.5.3 Serial Channel Status Register A

All status bits are active high unless an asterisk (*) appears in the signal name, in which case, they are active low.

The Receive Status bits (D31:16) are stuffed into the StatusOrIndex field in the DMA buffer descriptor when DMA operations are enabled and a buffer is closed. The upper 8 bits of this register (D31:24) can be cleared by writing a 1 to the respective bit position (to be used when interrupt driven). The upper 8 bits are automatically updated whenever a receive buffer is closed.

The Receive Interrupt Pending bits (D15:0) are cleared by writing a 1 to the respective bit position.

Address = FFD0 0008 / 48

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MATCH1	MATCH2	MATCH3	MATCH4	BGAP	CGAP	—	—	—	—	RXFDB		DCD	RI	DSR	CTS
RESET:															
0	0	0	0	0	0					0		0	0	0	0
R	R	R	R	R	R					R		R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBRK	RFE	RPE	ROVER	RRDY	RHALF	RBC	RFULL	DCDI	RII	DSRI	CTSI	TRDY	THALF	TBC	TEMPTY
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/C	R/C	R/C	R/C	R	R	R/C	R	R/C	R/C	R/C	R/C	R	R	R/C	R

MATCH1/2/3/4

The MATCH bit is set as a result of a MATCH character being configured in the receive match register in conjunction with the enable receive data match bit being set in control register B. The MATCH bit indicates a data match was found in the receive data stream and the current receive data buffer has been closed. The last character in the receive data buffer contains the actual MATCH character found. The MATCH bits are valid whenever the RBC bit in this register is set. When the receiver is configured for DMA operation, the MATCH status bits are automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the MATCH fields are only valid while the RBC field is set in status register A.

BGAP

The BGAP bit is set whenever the enable receive buffer gap timer is set in control register B and the time out value defined in the receive buffer gap timer register has expired. This bit indicates the maximum allowed time has occurred since the first byte was placed into the receive data buffer. The receive data buffer is closed under this condition. The BGAP bit is only valid when the RBC bit in this register is set. When the receiver is configured for DMA operation, the BGAP status bit is automatically

written to the DMA receive buffer descriptor's status field. When not using DMA, the BGAP field is only valid while the RBC field is set in status register A.

CGAP

The CGAP bit is set whenever the enable receive character gap timer is set in control register B and the time out value defined in the receive character gap timer register has expired. This bit indicates the maximum allowed time has occurred since the previous byte was placed into the receive data buffer. The receive data buffer is closed under this condition. The CGAP bit is only valid when the RBC bit in this register is set. When the receiver is configured for DMA operation, the CGAP status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the CGAP field is only valid while the RBC field is set in status register A.

RXFDB

The RXFDB field identifies the number of valid bytes contained in the next long word to be read from the channel FIFO data register. The next read of the FIFO can contain one, two, three, or four valid bytes of data. This field must be read before the FIFO is read to determine which bytes of the 4-byte long word contain valid data. Normal endian byte ordering rules apply to the channel FIFO data register.

“01” - one byte

“10” - half-word

“11” - three bytes (LENDIAN determines which three)

“00” - full-word

DCD

The data carrier detect (DCD) field identifies the current state of the EIA data carrier detect signal. A 1 indicates active, a 0 indicates inactive.

RI

The ring indicator (RI) field identifies the current state of the EIA ring indicator signal. A 1 indicates active, a 0 indicates inactive.

DSR

The data set ready (DSR) field identifies the current state of the EIA data set ready signal. A 1 indicates active, a 0 indicates inactive.

CTS

The clear to send (CTS) field identifies the current state of the EIA clear to send signal. A 1 indicates active, a 0 indicates inactive.

RBRK

The receive break condition (RBRK) field indicates a UART receive break condition has been detected. When set, the RBRK field remains set until acknowledged. The RBRK bit is acknowledged by writing a 1 to the same bit position in status register A. The RBRK status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

RFE

The receive framing error (RFE) field indicates a receive framing error condition has been detected. When set, the RFE field remains set until acknowledged. The RFE bit is acknowledged by writing a 1 to the same bit position in status register A. The RFE status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

RPE

The receive parity error (RPE) field indicates a receive parity error condition has been detected. When set, the RPE field remains set until acknowledged. The RPE bit is acknowledged by writing a 1 to the same bit position in status register A. The RPE status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

ROVER

The receive overrun (ROVER) field indicates a receive overrun error condition has been detected. An overrun condition indicates the FIFO was full while data needed to be written by the receiver. When set, the ROVER field remains set until acknowledged. The ROVER bit is acknowledged by writing a 1 to the same bit position in status register A. The ROVER status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

RRDY

The RRDY field indicates data is available to be read from the FIFO data register. Before reading the FIFO data register, the RXFDB field in status register A must be read to determine how many active bytes are available during the next read of the FIFO data register. The RRDY field is typically only used in interrupt driven applications, the RRDY field is not used for DMA operation. The RRDY status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

The RRDY bit is never active while the RBC bit is active. The RBC bit must be acknowledged to activate the RRDY bit. When the receiver is configured to operate in DMA mode, the interlock between RBC and RRDY is handled automatically in hardware.

RHALF

The RHALF field indicates the receive data FIFO contains at least 16 bytes. The RHALF field is typically only used in interrupt driven applications, the RHALF field is not used for DMA operation. The RHALF status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

RBC

The receive buffer closed (RBC) field indicates a receive buffer closed condition. When set, the RBC field remains set until acknowledged. The RBC bit is acknowledged by writing a 1 to the same bit position in status register A. The RBC bit is automatically acknowledged by hardware when the receiver is configured to operate in DMA mode. The RBC status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

For UART and SPI applications, the RBC field indicates that bits D31:26 in status register A are valid. While the RBC field is active, the RRDY bit is not active. To activate RRDY (to read the data FIFO), the RBC bit must be acknowledged. This interlock between RBC and RRDY is defined to allow the firmware driver the ability to read the status bits in status register A D31:26. When operating in DMA mode, the D31:26 status bits are automatically written to the receive DMA buffer descriptor and the interlock between RBC and RRDY is handled automatically in hardware.

For HDLC applications, the RBC field indicates that bits D25:16 in status register B are valid. While the RBC field is active, the RRDY bit is not active. To activate RRDY (to read the data FIFO), the RBC bit must be acknowledged. This interlock between RBC and RRDY is defined to allow the firmware driver the ability to read the status bits in status register B D25:16. When operating in DMA mode, the D25:16 status bits are automatically written to the receive DMA buffer descriptor and the interlock between RBC and RRDY is handled automatically in hardware.

RFULL

The RFULL field indicates the receive data FIFO is currently full.

DCDI

The DCDI field indicates a state change in the EIA data carrier detect signal. When set, the DCDI field remains set until acknowledged. The DCDI bit is acknowledged by writing a 1 to the same bit position in status register A. The DCDI status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

RII

The RII field indicates a state change in the EIA ring indicator signal. When set, the RII field remains set until acknowledged. The RII bit is acknowledged by writing a 1

to the same bit position in status register A. The RII status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

DSRI

The DSRI field indicates a state change in the EIA data set ready signal. When set, the DSRI field remains set until acknowledged. The DSRI bit is acknowledged by writing a 1 to the same bit position in status register A. The DSRI status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

CTSI

The CTSI field indicates a state change in the EIA clear to send signal. When set, the CTSI field remains set until acknowledged. The CTSI bit is acknowledged by writing a 1 to the same bit position in status register A. The CTSI status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

TRDY

The TRDY field indicates data can be written to the FIFO data register. The TRDY field is typically only used in interrupt driven applications, the TRDY field is not used for DMA operation. The TRDY status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

The TRDY bit is never active while the TBC-bit is active. The TBC bit must be acknowledged to activate the TRDY-bit. When the transmitter is configured to operate in DMA mode, the interlock between TBC and TRDY is handled automatically in hardware.

THALF

The THALF field indicates the transmit data FIFO contains room for at least 16 bytes. The THALF field is typically only used in interrupt driven applications, the THALF field is not used for DMA operation. The THALF status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

TBC

The transmit buffer closed (TBC) field indicates a transmit buffer closed condition. When set, the TBC field remains set until acknowledged. The TBC bit is acknowledged by writing a 1 to the same bit position in status register A. The TBC bit is automatically acknowledged by hardware when the transmitter is configured to operate in DMA mode. The TBC status condition can be programmed to generate an interrupt by setting the IE bit in control register A.

TBC is only used in HDLC applications. For HDLC applications, the TBC field indicates that bits D15:14 in status register B are valid. While the TBC field is active, the TRDY-bit is not active. To activate TRDY (to write to the data FIFO), the TBC bit

must be acknowledged. This interlock between TBC and TRDY allows the firmware driver to read the status bits in status register B D15:14. When operating in DMA mode, the D15:14 status bits are automatically written to the transmit DMA buffer descriptor and the interlock between TBC and TRDY is handled automatically in hardware.

TEMPTY

The TEMPTY field indicates the transmit data FIFO is currently empty. TEMPTY simply reports the status of the FIFO, it does not indicate the character currently in the transmit shift register has been transmitted. The TBC-bit must be used for the “all-sent” condition.

7.5.4 Serial Channel Bit-Rate Registers

The serial channel bit-rate registers are 32-bit registers used to configure the bit-rate for each serial channel.

The serial channel can be configured to operate using an internal or external timing reference. When configured for internal, the timing reference is provided by the bit-rate generator. When configured for external timing, the timing reference is provided by PORTC signals (OUT1, OUT2).

The serial channel can be configured to operate in either 1X mode (synchronous) or 16X mode (asynchronous). When using the internal bit-rate generator, the frequency must be properly configured to match the desired mode.

Address = FF80 0540

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EBIT	TMODE	RXSRC	TXSRC	RXEXT	TXEXT	CLKMUX		TXCINV	RXCINV	—	—	—	—	—	—
RESET:															
0	0	0	0	0	0	0	0	0	0						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
—	—	—	—	—	N Register										
RESET:															

EBIT

The EBIT field enables or disables the internal bit-rate generator. A 1 in the EBIT field allows the bit-rate generator to operate.

TMODE

The TMODE field configures the serial channel to operate in X1 mode (synchronous timing) or X16 mode (asynchronous UART timing).

When the transmitter or receiver are configured to use an external timing source (TXSRC or RXSRC set to 1), the serial channel is automatically configured to operate in 1X mode. In this situation, the serial channel clock is provided directly from the OUT1/OUT2 inputs via the PORTA/B/C interface. The TXSRC and RXSRC bits, when set, override any configuration setting for PORTA[4] / B[4] / C[7,5] in the GEN Module. The setting of TXSRC to 1 overrides any setting of the TXEXT bit.

Figure 7-6 shows the relationship between TXCLK, RXCLK, TXD, and RXD. TXD is driven active from the falling edge of the TXCLK signal. The RXD input is sampled using the rising edge of the RXCLK signal.

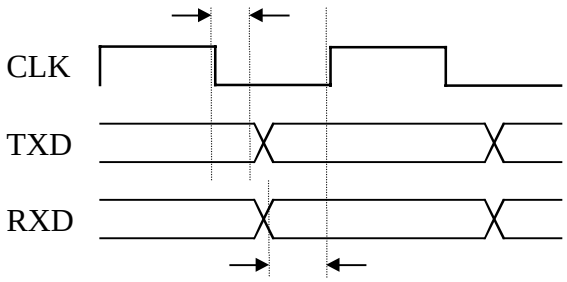


Figure 7-6: Serial Channel 1X Timing

RXSRC

The RXSRC field controls the source of the receiver clock. The receive clock can be provided by an internal source. Alternatively, the receiver clock can be provided by an input on the OUT1 signal attached to the PORTA/B ports. When using the OUT1 signal, the PORTA/B port pin must be configured as a special function input.

TXSRC

The TXSRC field controls the source of the transmitter clock. The transmitter clock can be provided by an internal source. Alternatively, the transmitter clock can be provided by an input on the OUT2 signal attached to the PORTC port pin. When using the OUT2 signal, the PORTC port pin must be configured as special function input.

RXEXT

The RXEXT field enables the receiver clock to be driven on the OUT1 signal attached to the PORTA/B ports. When using the OUT1 signal, the PORTA/B port pin must be configured as special function output.

TXEXT

The TXEXT field enables the transmitter clock to be driven on the OUT signal attached to the PORTC port. When using the OUT signal, the PORTC port pin must be configured as special function output.

CLKMUX

00 - Input Clock defined by F_{XTAL}

01 - Input Clock defined by F_{SYSCLK}

10 - Input Clock defined by Input on OUT1

11 - Input Clock defined by Input on OUT2

The CLKMUX field controls the bit-rate generator clock source. The bit-rate generator can be configured to use one of four clock sources; the external crystal oscillator, the internal PLL SYSCLK output, an input signal on the OUT1 signal on PORTA/B, or an input signal on the OUT2 signal attached to PORTC. When using either the OUT1 or OUT2 configurations, the PORTA/B/C port pin must be configured as special function input.

N Register

The effective frequency of the BRG is defined by the following equations:

$F_{BRG} = F_{XTAL} / [2 * (N + 1)]$ Using a CLKMUX setting of “00”

$F_{BRG} = F_{SYSCLK} / [2 * (N + 1)]$ Using a CLKMUX setting of “01”

$F_{BRG} = F_{OUT1} / [2 * (N + 1)]$ Using a CLKMUX setting of “10”

$F_{BRG} = F_{OUT2} / [2 * (N + 1)]$ Using a CLKMUX setting of “11”

Using an 18.432 MHz crystal, and a System PLL multiplier of 5 (SYS_CLK = 29.491 MHz) as an example, the following bit-rates can be generated using the bit-rate generator:

Bit-Rate	CLKMUX	N Register	
		X1 Mode	X16 Mode
75	00	n/a	1535
150	00	n/a	767
300	00	n/a	383
600	00	n/a	191
1200	00	1535	95
2400	00	767	47
4800	00	383	23
7200	00	255	15
9600	00	191	11
14400	00	127	7
19200	00	95	5
28800	00	63	3
38400	00	47	2
57600	00	31	1
115200	00	15	0
230400	01	63	3

Table 7-8: Bit-Rate Examples

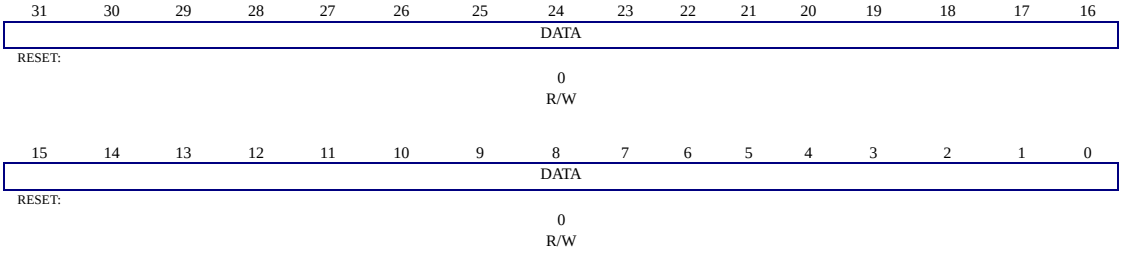
7.5.5 Serial Channel FIFO Registers

The serial channel FIFO data registers are 32-bit registers used to manually interface with the serial controller FIFOs in lieu of using DMA support.

Writing to this register loads the transmit FIFO. This register can only be written when the transmit data register ready bit (TRDY) is set in the serial control register. Writing to the serial channel FIFO register automatically clears the TRDY bit.

Reading from this register empties the receive FIFO. Data is available whenever the RRDY bit is set in the serial status register. The serial status register (RXFDB bits) identifies how many bytes are available to be read. Reading the FIFO register automatically clears the RRDY bit in the serial status register.

Address = FFD0 0010 / 50



7.5.6 Receive Buffer Gap Timer

The receive buffer gap timer closes out a receive serial data buffer. This timer is configured to provide an interval in the range of 100 us to 2.3 S. The timer is reset whenever the first character is received in a new buffer. New characters are received while the timer operates. When the timer reaches its programmed threshold, the receive data buffer is closed.

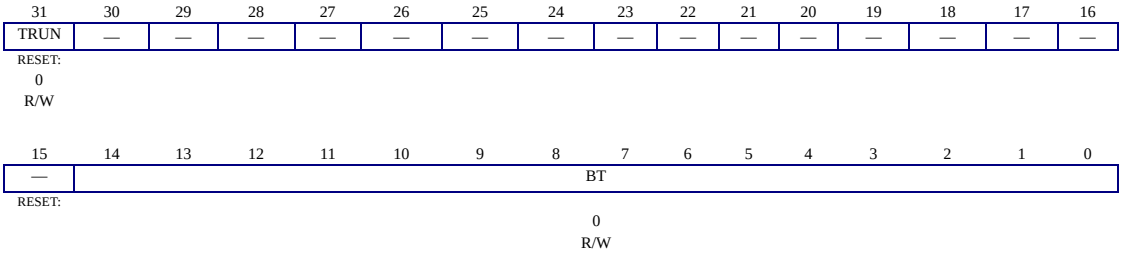
If the serial channel is configured to operate in DMA mode, the DMA channel is signaled to close the buffer and start a new buffer. If the serial channel is configured to operate in interrupt mode, then the expiration of the timer causes an interrupt to be generated.

The receive buffer timer operates using the external crystal clock (with divide by 5 prescaler within the SYS module) and a 9-bit prescaler within the SER module. The receive buffer timer is configured with a 15-bit programmable counter. The effective buffer timer value is defined by the following equation:

$$TIMEOUT = [512 * (BT + 1)] / F_{XTAL}$$

The receive buffer timer register is a 32-bit register. All bits are set to 0 on reset.

Address = FFD0 0014 / 54



TRUN

Set TRUN to 1 to allow the receive Buffer Gap Timer to operate.

BT

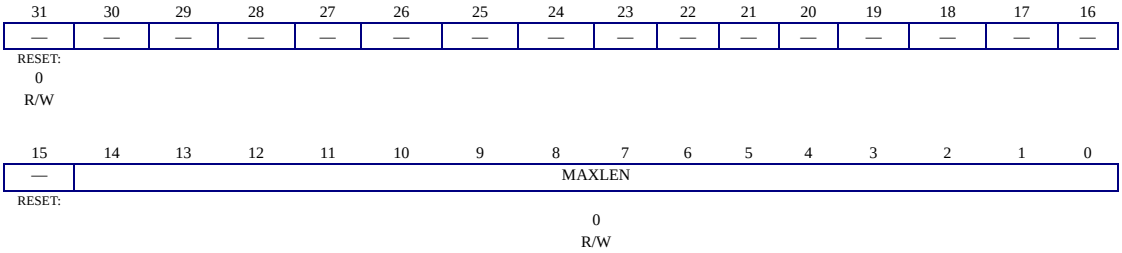
The required value for the receive buffer gap timer is a function of the channel bit-rate and the maximum receive buffer size. The recommended approach is to set the buffer gap timer to be a value that is slightly larger than the amount of time required to fill the maximum buffer size using the channel bit-rate. The following equation can be used to define the recommended buffer gap timer value. If the resulting value does not produce a value that fits within the range for BT, then you may want to reconsider the chosen size for MAX_RX_BUF_SIZE.

$$\text{BT RECOMMENDED} = [(\text{MAX\_RX\_BUF\_SIZE} * 1.10 * F_{XTAL}) / (\text{Bit-Rate} * 512)] - 1$$

7.5.7 HDLC Max Length Register

When a serial channel is configured to operate in HDLC mode, the receive buffer timer register provides the HDLC max frame length value

Address = FFD0 0014 / 54



MAXLEN

Maximum HDLC receive buffer frame size.

7.5.8 Receive Character Gap Timer

The receive character gap timer closes out a receive serial data buffer due to a gap between characters. This timer is configured to provide an interval in the range of 100us to 145ms. The timer is reset whenever a character is received. When the timer reaches its programmed threshold, the receive data buffer is closed.

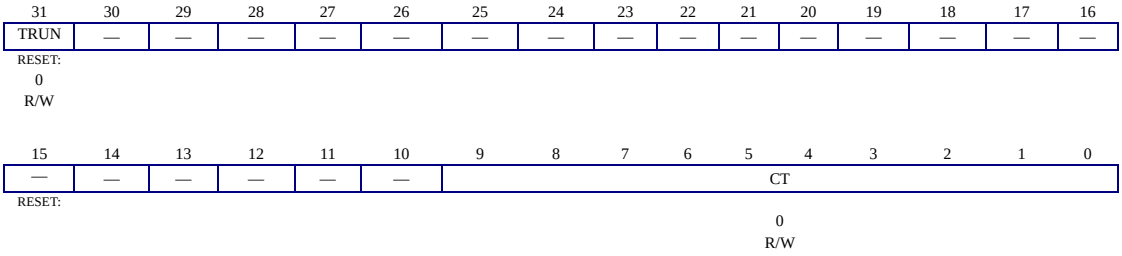
If the serial channel is configured to operate in DMA mode, the DMA channel is signaled to close the buffer and start a new buffer. If the serial channel is configured to operate in interrupt mode, then the expiration of the timer causes an interrupt to be generated.

The receive character timer operates using the external crystal clock (with divide by 5 pre-scaler) and a 9-bit prescaler within the SER module. The receive character timer is configured with a 10-bit programmable counter. The effective buffer timer value is defined by the following equation:

$$TIMEOUT = [512 * (CT + 1)] / F_{XTAL}$$

The receive character timer register is a 32-bit register. All bits are set to 0 on reset.

Address = FFD0 0018 / 58



TRUN

Set TRUN to 1 to allow the Receive Character Gap Timer to operate.

CT

The required value for the receive character timer is a function of the channel bit-rate. The recommended approach is to set the character timer to be a value that is 10 times the character period for the channel bit-rate. The following equation can be used to define the recommended character timer value.

CT RECOMMENDED = $[(10 * F_{XTAL}) / (\text{Bit-Rate} * 512)] - 1$

7.5.9 Receive Match Register

When the serial channel is configured for UART mode, the receive match register provides the data bytes, which the receiver uses to compare against the incoming receive data stream. If a match is found and the appropriate match enable bit is set in control register B, the current receive data buffer is closed by the serial channel and a new buffer is started.

When the serial channel is configured for HDLC mode, the receive match register provides the data bytes, which the receiver uses to compare against the incoming HDLC address (first and second bytes of the HDLC data frame). The address comparison function is enabled using the appropriate match enable bit settings in control register B. If the address comparison function is enabled and an address match occurs, the HDLC data packet is accepted. If the address comparison function is disabled, all HDLC packets are accepted.

In both UART and HDLC configurations, individual bits within the match register bytes can be masked using the receive match mask register.

Address = FFD0 001C / 5C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDMB1								RDMB2							
RESET:															
0								0							
R/W								R/W							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDMB3								RDMB4							
RESET:															
0								0							
R/W								R/W							

7.5.10 Receive Match MASK Register

The receive match MASK register masks those bits in the receive match DATA register that should NOT be included in the match comparison. To mask a bit in the match comparison function, place a 1 in the same bit position within this register.

Address = FFD0 0020 / 60

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMMB1								RMMB2							
RESET:															
0								0							
R/W								R/W							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMMB3								RMMB4							
RESET:															
0								0							
R/W								R/W							

7.5.11 Control Register C (HDLC)

Control register C provides the configuration bits required when the serial channel is configured to operate in HDLC mode.

Address = FFD0 0024 / 64

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFLAG				—	—	—	—	CRC		RXCRC	CHKCRC*	—	—	TXCRC*	FLAG*/IDL
RESET:															
0								0		0	0			0	0
R/W								R/W		R/W	R/W			R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RESET:															

NFLAG

The NFLAG field controls the minimum number of flags to be inserted between back-to-back HDLC frames. Setting this field to 0 allows a single shared flag between two successive HDLC frames. Additional flags are always inserted between HDLC frames when there is not data ready to send (conditioned by the setting of the FLAG*/IDL field).

CRC

The CRC field identifies the type of CRC encoding used in the HDLC frames. The settings include NONE, CRC-16, and CRC-32.

“00” - NO CRC

“01” - Reserved

“10” - 16-bit CCITT CRC-16

“11” - 32-bit CCITT CRC-32

RXCRC

The RXCRC field controls whether or not the HDLC CRC field is placed into the receive data FIFO. The HDLC CRC field is only required to be examined by firmware for either debugging purposes or possibly some encapsulation protocols. Setting the RXCRC field to 1 causes the HDLC CRC field to be inserted into the receive FIFO.

CHKCRC*

The CHKCRC* field controls whether or not the HDLC CRC field is checked by the receiver for errors. When CHKCRC* is set to 0, the HDLC receiver checks the HDLC CRC field and tags the HDLC data packet as good or bad. When CHKCRC* is set to 1, the HDLC receiver does not check the HDLC frame and marks all HDLC frames as good.

TXCRC*

The TXCRC* field controls whether or not the transmitter inserts the HDLC CRC field at the end of an HDLC frame. When TXCRC* is set to 0, the HDLC transmitter always inserts the HDLC CRC field at the end of the HDLC frame.

FLAG*/IDL

The FLAG*/IDL field controls how the HDLC transmitter operates while no data is available to transmit. When FLAG*/IDL is set to 0, the transmitter inserts flags while no data is available to transmit. When FLAG*/IDL is set to 1, the transmitter inserts marks while no data is available to transmit. The HDLC transmitter always sends, at a minimum, at least one flag immediately before the start of the HDLC frame and at least one flag immediately after the end of the HDLC frame.

7.5.12 Status Register B (HDLC)

Status register B provides status bits required when the serial channel is configured to operate in HDLC mode. The receive buffer closed status bits are only valid when the RXBC bit is set in status register A. The receive buffer closed status bits must be read before the RXBC bit is acknowledged. The RXBC bit must be acknowledged before subsequent receive frames can be read from the receive FIFO.

The transmit buffer closed (TXBC) status bits are only valid when the TXBC bit is set in status register A. The transmit buffer closed status bits must be read before the TXBC bit is acknowledged.

Address = FFD0 0028 / 68

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLAGS	IDLE	—	—	—	—	RXBCAM1	RXBCAM2	RXBCAM3	RXBCAM4	RXBCN	RXBCC	RXBCO	RXBCA	RXBCL	RXBCAB
RESET:															
0	0					0	0	0	0	0	0	0	0	0	0
R	R					R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXBCC	TXBCU	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RESET:															
0	0														
R	R														

FLAGS

The FLAGS field is set to 1 when the HDLC receiver has detected the remote transmitter is sending continuous flags.

0 - Receiver is active or not receiving flags

1 - Receiver is IDLE receiving flags

IDLE

The IDLE field is set to 1 when the receiver is inactive receiving all ones (1). The IDLE bit is set to 0 when the receiver is actively receiving a packet or not receiving the all ones (1) condition.

0 - Receiver is active or not receiving IDLE codes

1 - Receiver is IDLE receiving IDLE codes

RXBCAM1

The RXBCAM1 field indicates the HDLC receiver has found an 8-bit address comparison match between the first byte of the HDLC frame and the value found in the RDMB1 field. The RXBCAM1 field also indicates the HDLC receiver has found a 16-bit address comparison match between the first two bytes of the HDLC frame and the value found in the RDMB1/RDMB2 fields (provided the MAM1 field is set to 1).

When the receiver is configured for DMA operation, the RXBCAM1 status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the RXBCAM1 field is only valid while the RBC field is set in status register A.

RXBCAM2

The RXBCAM2 field indicates the HDLC receiver has found an 8-bit address comparison match between the first byte of the HDLC frame and the value found in the RDMB2 field. When the receiver is configured for DMA operation, the RXBCAM1 status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the RXBCAM2 field is only valid while the RBC field is set in status register A.

RXBCAM3

The RXBCAM3 field indicates the HDLC receiver has found an 8-bit address comparison match between the first byte of the HDLC frame and the value found in the RDMB3 field. The RXBCAM3 field also indicates the HDLC receiver has found a 16-bit address comparison match between the first two bytes of the HDLC frame and the value found in the RDMB3/RDMB4 fields (provided the MAM2 field is set to 1). When the receiver is configured for DMA operation, the RXBCAM3 status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the RXBCAM3 field is only valid while the RBC field is set in status register A.

RXBCAM4

The RXBCAM4 field indicates the HDLC receiver has found an 8-bit address comparison match between the first byte of the HDLC frame and the value found in the RDMB4 field. When the receiver is configured for DMA operation, the RXBCAM4 status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the RXBCAM4 field is only valid while the RBC field is set in status register A.

RXBCN

The RXBCN field is set to indicate the current HDLC frame was received without any CRC errors. When the receiver is configured for DMA operation, the RXBCN status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the RXBCN field is only valid while the RBC field is set in status register A.

RXBCC

The RXBCC field is set to indicate the current HDLC frame was received with CRC errors. The firmware driver can reject the packet, if so desired, using knowledge pro-

vided by RXBCC. When the receiver is configured for DMA operation, the RXBCC status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the RXBCC field is only valid while the RBC field is set in status register A.

RXBCO

The RXBCO field is set to indicate the current HDLC frame was received with a receive FIFO overrun condition. An overrun condition indicates the receive FIFO was not emptied at a fast enough rate. The firmware driver can reject the packet using knowledge provided by RXBCO. When the receiver is configured for DMA operation, the RXBCO status bit is automatically written to the DMA receive buffer descriptor’s status field. When not using DMA, the RXBCO field is only valid while the RBC field is set in status register A.

RXBCA

The RXBCA field is set to indicate the current HDLC frame was received with an alignment error condition. An alignment error condition occurs when an HDLC frame ends on a non-8-bit boundary. The firmware driver can reject the packet using knowledge provided by RXBCA. When the receiver is configured for DMA operation, the RXBCA status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the RXBCA field is only valid while the RBC field is set in status register A.

RXBCL

The RXBCL field is set to indicate the current HDLC frame was received that was too long. The HDLC frame length exceeded the value programmed in the HDLC max length register. The firmware driver can reject the packet using knowledge provided by RXBCL. When the receiver is configured for DMA operation, the RXBCL status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the RXBCL field is only valid while the RBC field is set in status register A.

RXBCAB

The RXBCAB field is set to indicate the current HDLC frame was received with an abort error condition. An abort error condition occurs when an HDLC receiver detects a continuous string of 8 consecutive ones (1). The firmware driver can reject the packet using knowledge provided by RXBCAB. When the receiver is configured for DMA operation, the RXBCAB status bit is automatically written to the DMA receive buffer descriptor's status field. When not using DMA, the RXBCAB field is only valid while the RBC field is set in status register A.

TXBCC

The TXBCC field is set to indicate the current HDLC transmit frame was aborted due to a loss of the CTS condition. The CTSTX field in control register A must be set for this to happen. The firmware driver can use the TXBCC to indicate the HDLC frame was not successfully transmitted. When the transmitter is configured for DMA operation, the TXBCC status bit is automatically written to the DMA transmit buffer descriptor's status field. When not using DMA, the TXBCC field is only valid while the TBC field is set in status register A.

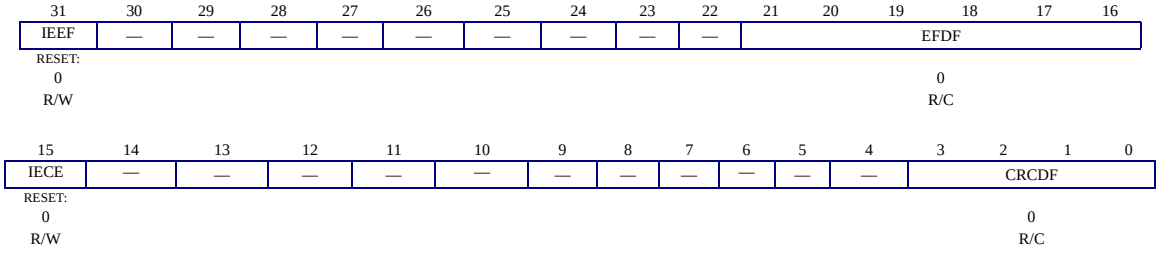
TXBCU

The TXBCU field is set to indicate the current HDLC transmit frame encountered an underrun condition. The firmware driver can use the TXBCC to indicate the HDLC frame was not successfully transmitted. When the transmitter is configured for DMA operation, the TXBCU status bit is automatically written to the DMA transmit buffer descriptor's status field. When not using DMA, the TXBCU field is only valid while the TBC field is set in status register A.

7.5.13 Status Register C (HDLC)

Status register C provides error counter registers required when the serial channel is configured to operate in HDLC mode.

Address = FFD0 002C / 6C



IEEF - Interrupt Enable

Setting this bit causes a receiver-interrupt to occur whenever the EFDF counter reaches a value of 24.

EFDF - Error Free Discarded Frame Counter

This 5-bit register identifies the number of frames discarded due to a failure to match the selected match address value. These frames did not result in a CRC error.

The counter stops counting when it reaches the maximum value of 31. An interrupt can be enabled when the counter reaches a value of 24. This register is automatically cleared each time this register is read.

IECE - Interrupt Enable

Setting this bit causes a receiver-interrupt to occur whenever the CRCDF counter reaches a value of 8.

CRCDF - Error Discarded Frame Counter

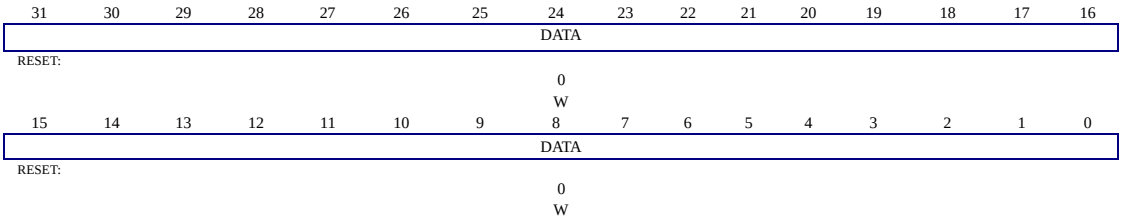
This 4-bit register identifies the number of frames discarded due to a failure to match the selected match address value. These frames also resulted in a CRC error. Note that this counter does not include those frames with a CRC error that had a valid matching address.

The counter stops counting when it reaches the maximum value of 15. An interrupt can be enabled when the counter reaches a value of 8. This register is automatically cleared each time this register is read.

7.5.14 FIFO Data Register LAST (HDLC)

This FIFO data register provides transmit data for the serial channel. This register identifies the last data value for the current transmit data frame. Writing to this register marks the end of the data frame and identifies when to send the closing flag in HDLC mode.

Address = FFD0 0030 / 70



Chapter 8
GEN Module

The GEN module provides the NET+ARM chip with some miscellaneous functions:

1. (2) Programmable Timers with Interrupt
2. (1) Programmable Bus-Error Timer
3. (1) Programmable Watch-Dog Timer
4. (3) 8-bit programmable Parallel I/O Ports with Interrupt
5. System Priority Interrupt Controller
6. Miscellaneous System Control Functions

8.1 Module Configuration

The General module has a block of configuration space is mapped into the GEN module configuration space as defined in Table 2-1, *BBus Address Decoding*.

Address	Register
FFB0 0000	System Control Register
FFB0 0004	System Status Register
FFB0 0008	PLL Control Register
FFB0 000C	Software Service Register
FFB0 0010	Timer 1 Control Register
FFB0 0014	Timer 1 Status Register
FFB0 0018	Timer 2 Control Register
FFB0 001C	Timer 2 Status Register
FFB0 0020	PORT A Register
FFB0 0024	PORT B Register
FFB0 0028	PORT C Register

Table 8-1: General Configuration

Address	Register
FFB0 0030	Interrupt Enable Register
FFB0 0034	Interrupt Enable Register - SET
FFB0 0038	Interrupt Enable Register - CLEAR
FFB0 0034	Interrupt Status Register - Enabled
FFB0 0038	Interrupt Status Register - Raw
FFB0 0040	Cache Control Register 1
FFB0 0044	Cache Control Register 2

Table 8-1: General Configuration (Continued)

8.2 GEN Module Hardware Initialization

Many internal NET+ARM chip configuration features are application specific and need to be configured at power-up before the CPU boots.

The system bus address bits are used for this purpose during a power-up reset. The NET+ARM chip provides internal pullup resistors on all Address signals. Weak external pulldown resistors can be employed to configure various internal register bits to a zero state. The following identifies which ADDR bits control what functions in the GEN Module.

ADDR27 : GEN_LENDIAN

0 - Little Endian Configuration

1 - Big Endian Configuration

ADDR26 : GEN_BUSER

0 - ARM CPU Disabled; GEN_BUSER set to 1.

1 - ARM CPU Enabled; GEN_BUSER set to 0.

ADDR25 : GEN_IARB

0 - External System Bus Arbiter

1 - Internal System Bus Arbiter

ADDR19:09 : GEN_ID

Product Identification Code

(Refer to Section *11.4 NET+ARM Chip Bootstrap Initialization*.)

Note: The inverted ENDIAN bit (ADDR27) is loaded into the LENDIAN bit within the System Control Register. In the System Control Register, LENDIAN=1 (for little endian mode) and LENDIAN=0 (for big endian mode).

8.2.1 System Control Register

The System Control register is a 32-bit read/write register. All control bits are active high unless an asterisk (*) appears in the signal name, in which case, it is active low. All control bits are set to their respective inactive state on reset.

Address = FFB0 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LENDIAN	BSPEED	BCKLD	EPS	—	—	—	SWE	SWRI		SWT		—	BME	BMT	
RESET: $\overline{\text{ADDR27}}$	0	0	0				0	0		0			0	0	
R/W	R/W	R/W	R/W				R/W	R/W		R/W			R/W	R/W	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USER	BUSER	IARB	DMATST	TEALAST	MISALIGN	CACHE	WB	CINIT	—	—	—	—	—	—	—
RESET: 0	ADDR26	ADDR25	0	0	0	0	0	0							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W							

LENDIAN

The LENDIAN bit controls the Endian configuration for the NET+ARM chip. A 1 in this bit position configures the NET+ARM chip to operate in Little Endian mode. A 0 configures the NET+ARM chip for Big Endian mode. During reset, the LENDIAN bit defaults to the value defined by the inverted sense ADDR27 (refer to Section *8.2 GEN Module Hardware Initialization*).

BSPEED

“00” - 1/4 Speed

“01” - 1/2 Speed

“10” - Full Speed

“11” - Reserved

The BSPEED field controls the relative speed between the internal system clock (SYSCLK) and the BCLK that operates the NET+ARM chip system bus protocol. The BCLK can be configured to operate at ¼ speed, ½ speed, or full speed.

BCLKD

0 - BCLK Output Enabled

1 - BCLK Output forced to LOW state

The BCLKD field can be used to shut-down the operation of the BCLK signal. Turning off the BCLK output can be used to minimize electro-magnetic interference (EMI) when the BCLK signal is not required for the application.

SWE

The SWE bit is set to 1 to enable the watch-dog timer circuit. The watch-dog timer can be configured, via SWRI, to generate an interrupt or a reset condition if and when the watch-dog timer expires. Once the SWE bit is set to 1, only a hardware reset resets the bit back to 0.

SWRI

“00” - Software watch Dog causes Normal IRQ Interrupt

“01” - Software watch Dog causes Fast FIRQ Interrupt

“10” - Software watch Dog causes Reset

“11” - Reserved

The SWRI field controls the action that occurs when the watch-dog timer expires. The watch-dog can be configured to generate a normal interrupt condition, a fast interrupt condition, or a reset condition.

SWT

“00” - $2^{20} / F_{XTAL}$

“01” - $2^{22} / F_{XTAL}$

“10” - $2^{24} / F_{XTAL}$

“11” - $2^{25} / F_{XTAL}$

The SWT field controls the time out period for the watch-dog timer. The time out period is a function of the F_{XTAL} value. Refer to section *11.3.2 XTAL Clock Generation* for information on the F_{XTAL} value.

BME

0 - Disable Bus Monitor operation

1 - Enable Bus Monitor operation

The BME bit is set to 1 to enable the bus-monitor timer. The bus-monitor timer detects when a bus master is accessing a peripheral and no peripheral responds with TA*. When the bus-monitor timer expires, the current Bus cycle is immediately terminated and the current system bus master is issued a data abort indicator. The bus-monitor timer is required to avoid a system lockup condition that can occur when a bus master attempts to address memory space that is not decoded by any peripheral.

BMT

“00” - 128 BCLKs (Bus Clocks)

“01” - 64 BCLKs

“10” - 32 BCLKs

“11” - 16 BCLKs

The bus-monitor timer (BMT) field controls the time out period for the bus-monitor timer. Typically, the BMT field is set to its maximum value, however, the BMT field can be set to a lower value to minimize the latency when issuing a data abort signal. The BMT field needs to be set to a value that is larger than the anticipated longest access time for all peripherals.

USER

The USER bit controls whether or not applications operating in ARM user-mode can access internal registers within the NET+ARM chip. If the USER bit is set to 0 and an application, operating in ARM user-mode, attempts to access (read or write) an internal NET+ARM chip register, the application is given a data abort causing a transfer in control to the data abort handler. When the USER bit is set to 1, any application can access the NET+ARM chip internal registers.

BUSER

The BUSER bit controls whether or not an external bus master can access internal registers within the NET+ARM chip. If the BUSER bit is set to 0 and an external bus master attempts to access (read or write) an internal NET+ARM chip register, the external bus master is given a data abort, via the TEA* signal. When the BUSER bit is set to 1, any external bus master can access the NET+ARM chip internal registers.

During reset, the BUSER bit defaults to the value defined by ADDR26 (refer to Section *8.2 GEN Module Hardware Initialization*).

IARB

0 - Use External System Bus Arbiter (BR* output, BG* input)
1 - Use Internal System Bus Arbiter (BR* input, BG* output)

The IARB bit determines if the NET+ARM chip operates using an internal bus arbiter or an external bus arbiter.

When IARB is low, the NET+ARM chip uses an external bus master. In this mode, the NET+ARM chip is not the default system bus owner. For any of the NET+ARM chip internal bus master (CPU, DMA, or ENI) to execute a memory-cycle, the NET+ARM chip asserts the BR* signal and waits for the external arbiter to grant the system bus to the NET+ARM chip by asserting the BG* input active low.

When IARB is high, the NET+ARM chip is the default system bus owner. If an external bus master desires to execute a system bus cycle, the external bus master must request the system bus by asserting the BR* input active low. The NET+ARM chip grants ownership to the external bus master by asserting the BG* output active low.

During reset, the IARB bit defaults to the value defined by ADDR25 (refer to section *8.2 GEN Module Hardware Initialization*).

DMATST

0 - Test Mode Disabled

1 - Allow unrestricted access to DMA Context RAM

The DMATST bit resets the DMA controller subsystem. It is also used to allow the ARM processor direct access to the internal context RAM found in the DMA controller.

When the DMATST bit is set to 1, the DMA controller subsystem is held in reset and the ARM processor can access all of the internal DMA context RAM. This is useful for diagnostics purposes. When the DMATST bit is set to 0, the DMA controller subsystem operates normally and only those bits in the DMA control register space are accessible by the ARM processor.

TEALAST

0 - Use TEA_ pin for Error Indication only

1 - Use TEA_ pin for LAST word of Burst Sequence Indicator and Error Indication

The TEALAST bit determines how the TEA* signal is to be used by the NET+ARM chip. When TEALAST is set to 0, the TEA* signal is only used to indicate a data abort condition. When TEALAST is set to 1, the TEA* signal is used in conjunction with the TA* signal to indicate either a data abort condition or a burst completion condition. Refer to Table 1-6: *Transfer Error Acknowledge* for additional information on the use of TA* and TEA*.

MISALIGN

When the MISALIGN bit is set to 1, mis-aligned address transfers cause a data abort to be issued to the offending bus master. A mis-aligned address transfer is defined as a half word access to an odd byte address boundary, or a full word access to either a half word or byte address boundary. This MISALIGN bit is useful during software debugging for detection of these cycles.

CACHE

0 - Disable Cache
1 - Enable Cache

The CACHE bit must be set to 1 to enable the internal cache memory found in the NET+40 device. If the internal cache memory is not being used, setting the CACHE bit to 0 lowers the power consumption of the NET+ARM chip.

WB

The WB is not used anywhere in the NET+ARM chip. This bit is reserved for future use and should be set to a value of 0 for future compatibility.

CINIT

0 - Normal Cache Operation

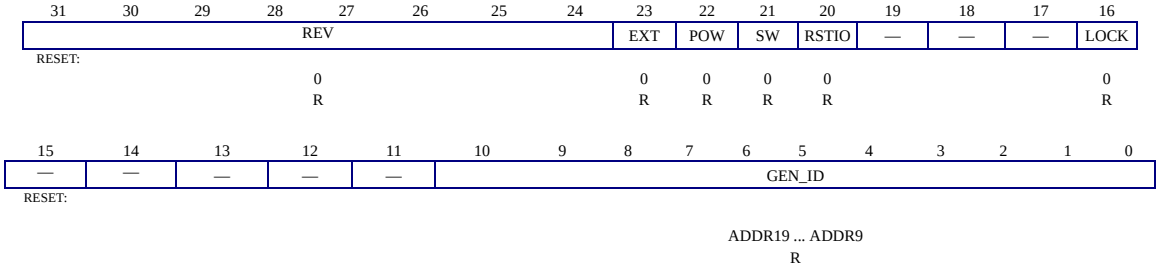
1 - Cache Initialization in progress

The CINIT bit is used during initialization of the cache subsystem. For more information concerning how to initialize the cache subsystem, refer to Section *3.4.12 Cache Initialization*.

8.2.2 System Status Register

The system status register is a 32-bit register. A bits in this register, except LOCK, are loaded during a hardware reset only. Changing an external jumper does not alter the GEN_ID value unless a hardware reset is first executed. The LOCK bit is dynamic and changes.

Address = FFB0 0004



REV

The REV field identifies the hardware identification of the NET+ARM chip and its revision. All NET+ARM chips have a unique REV identification.

NET+ARM Chip Device	REV ID
NET+5&10	0
NET+12-1	1
NET+15-0	4
NET+15-1	5
NET+15-2	6
NET+15-3	7
NET+15-4	7
NET+40-0	8
NET+40-1	9
NET+40-2	10
NET+40-3	11
NET+40-4	11

EXT

The EXT bit is set to indicate the RESET* pin was used to cause the last hardware reset condition. The EXT bit is set/reset during every hardware-reset condition.

POW

The power up reset (POW) bit is set to indicate a power up reset caused the last hardware reset condition. The POW bit is set/reset during every hardware-reset condition.

SW

The SW bit is set to indicate a software watch-dog time out caused the last hardware reset condition. The SW bit is set/reset during every hardware-reset condition.

RSTIO

The RSTIO bit is set to indicate the RSTIO bit in the ENI shared register caused the last hardware reset condition. The RSTIO bit is set/reset during every hardware-reset condition.

LOCK

The LOCK bit is set to 1 to indicate the internal phase lock loop (PLL) is currently in a Lock condition.

GEN_ID

During reset, the GEN_ID field defaults to the value defined by ADDR19:09 bits (refer to section *8.2 GEN Module Hardware Initialization*). The GEN_ID is only loaded once during a hardware reset condition.

8.2.3 PLL Control Register

The PLL control register defines the multiplication factor between the input crystal and the SYS_CLK operating frequency. The system frequency is controlled by the frequency control bit in the PLLCR as follows:

IF (PLLCNT <=3)

$F_{SYSCLK} = (F_{CRYSTAL} / 5) * 6$

ELSE

$F_{SYSCLK} = (F_{CRYSTAL} / 5) * (PLLCNT + 3)$

The PLLCR register is reset to 0 on a hardware reset only. It is not affected by a soft reset.

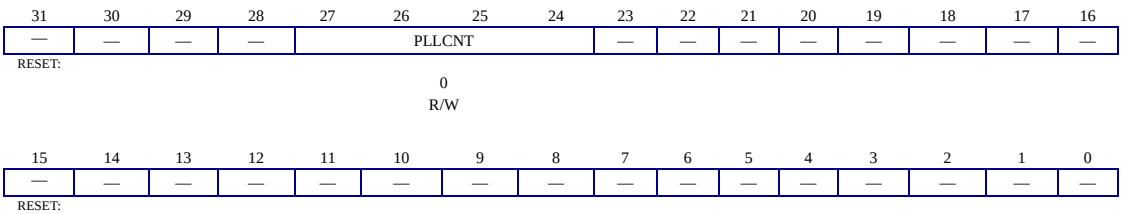
The PLLCR also defines the value F_{XTAL}. Refer to section *11.3.2 XTAL Clock Generation*.

Using an 18.432 MHz crystal as an example, the following table identifies the possible values of the SYS_CLK operating frequency:

PLLCNT	F_{SYSCLK} (MHz)
0	22.1184
1	22.1184
2	22.1184
3	22.1184
4	25.8048
5	29.4912
6	33.1776
7	36.8684
8	40.5500
9	44.2368
10	47.9232
11	51.6096
12	55.2960
13	58.9824
14	62.6688
15	66.3552

Table 8-2: PLL Configuration Examples

Address = FFB0 0008

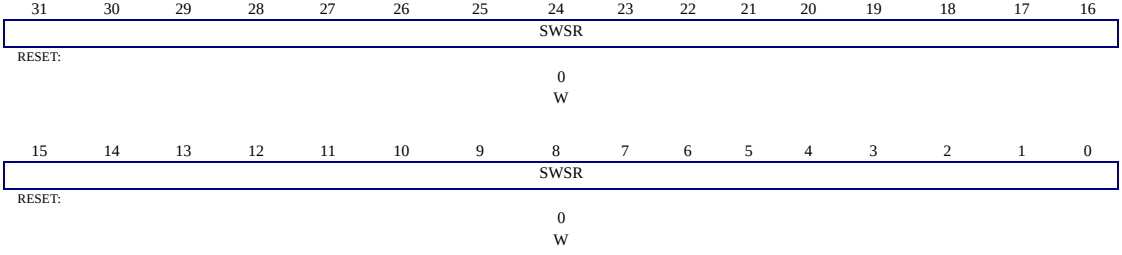


8.2.4 Software Service Register

The software service register (SWSR) acknowledges the system watch-dog timer. To acknowledge the watch-dog timer circuit, firmware must write $5A and $A5 to the SWSR using two separate write operations. There is no restriction on the time between the two operations, the two operations must simply occur in the proper sequence with the proper data values.

The SWSR is also used to request a software reset of the NET+ARM chip hardware. To request a soft reset, firmware must write $123 and $321 to the SWSR using two separate write operations. There is no restriction on the time between the two operations, the two operations must simply occur in the proper sequence with the proper data values. A software reset is as easy as 123.

Address = FFB0 000C



8.2.5 Timer Control Register

Timers 1 and 2 are used to provide the CPU with programmable interval timer(s). The timers operate using the external crystal clock and an optional 9-bit prescaler. The timers provide a 9-bit programmable down counter mechanism. An initial count register is loaded by the CPU to define the time-out period. When the current counter decrements to zero, an interrupt is provided to the CPU and the counter is reloaded. The current count value can be read by the CPU at any time.

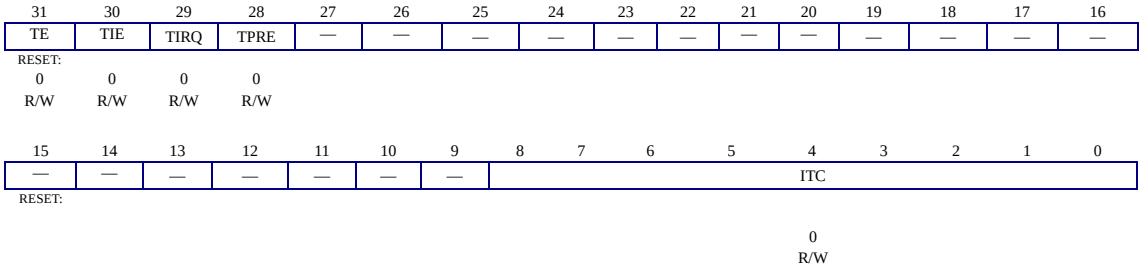
The effective value of the time-out is determined by the following equation:

TIMEOUT = [8 (ITC + 1)] / F_{XTAL} Without the 9-bit prescaler

TIMEOUT = [4096 * (ITC + 1)] / F_{XTAL} With the 9-bit prescaler

The Timer Control register is a 32-bit register. All bits are set to 0 upon reset.

Address = FFB0 0010 / 18



TE **Timer Enable**

The TE bit must be set to 1 in order to allow the Timer to operate. Setting the TE bit to 0 will reset and disable the timer from operating. The other fields within this register should be configured before TE is set to 1 or during the same memory cycle in which TE is set to 1.

TIE **Timer Interrupt Enable**

The TIE bit can be set to 1 in order to allow a Timer interrupt to be asserted to the CPU. A Timer interrupt is generated when the TIP bit is set in the Timer Status Register. The TIE bit can be used to either enable or disable the TIP Timer Interrupt.

TIRQ **Timer Interrupt Mode**

0: Normal Interrupt

1: Fast Interrupt

The TIRQ field controls the type of interrupt the Timer will assert to the CPU. The Timer can generate either a Normal Interrupt (ARM IRQ pin) or a Fast Interrupt (ARM FIRQ pin).

TPRE Timer Prescaler

0: Disable 9-bit prescaler

1: Enable 9-bit prescaler

The TPRE bit determines whether or not the 9-bit prescaler is to be used in calculating the TIMEOUT parameter; using the prescaler allows for longer values of TIMEOUT.

ITC Initial Timer Count

The ITC field defines the TIMEOUT parameter for interrupt frequency. The TIMEOUT period is a function of the F_{XTAL} frequency. (Refer to Section *11.3.1 SYSCLK Generation*.)

The effective value of the timer TIMEOUT is determined by the following equation:

TIMEOUT = [8 * (ITC + 1)] / F_{XTAL} Without the 9-bit prescaler

TIMEOUT = [4096 * (ITC + 1)] / F_{XTAL} With the 9-bit prescaler

Using an 18.432 MHz crystal as an example, the following table identifies the possible minimum and maximum value of the timers:

$F_{XTAL} = F_{CRYSTAL} / 5$

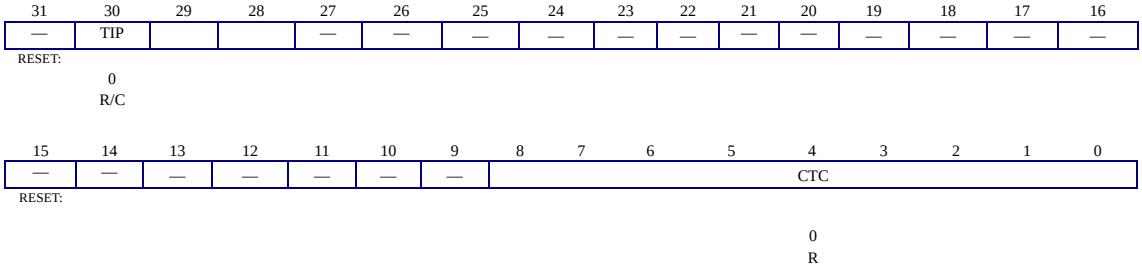
$F_{XTAL} = 18432000 / 5 = 3686400$ Hz

TPRE = 0		**TPRE = 1**	
ITC = 0	**ITC = 511**	**ITC = 0**	**ITC = 1**
2 uS	1.1 mS	1.1 mS	568 mS

8.2.6 Timer Status Register

The timer status register is a 32-bit read-only register. The TIP bit is set when the current timer count register decrements to zero. If the TIE bit is set in the control Register, an interrupt is provided to the CPU. The TIP bit is cleared by writing a 1 to the respective bit position in this register.

Address = FFB0 0014 / 1C



TIP **Timer Interrupt Pending**

The TIP bit will be set to 1 whenever the Timer is enabled and the CTC value counts down to 0. The TIP can be used to generate an interrupt to the CPU provided the TIE bit is set in the Timer Control Register. Writing a 1 to the same bit position in this register will clear the TIP bit.

Note: The TIP bit will immediately be set when the TE bit is changed from 0 to 1. This is true because the CTC field starts out initially at zero. This means an interrupt will occur immediately after TE transitions from 0 to 1. If this initial interrupt causes a problem in any specific application, the software must be designed to ignore the first interrupt after TE transitions from 0 to 1.

CTC **Current Timer Count**

The CTC field represents the Current Timer Count. Each time the CTC field reaches zero, the TIP bit is set and the CTC is reloaded with the value defined in the ITC field. The CTC continues to count back down to 0. The TIP bit can be used to generate an interrupt to the CPU.

8.2.7 PORT A Register

The PORT A Register is used to configure the personality of each PORT A General Purpose I/O pin. Each of the eight PORTA GPIO pins can be individually programmed to be one of the following:

- General Purpose Input
- General Purpose Output
- Special Function Input
- Special Function Output

Table 8.3 – PORT A Configuration describes the possible configurations for each of the PORTA signals. Many, but not all, of the PORTA signals can be configured for Special Function. Note that this table has four basic columns, each column denoting one of the four possible configurations for each PORTA bit.

The eight bits in the MODE field determine which of the PORTA bits are configured for GPIO or Special Function.

General Purpose I/O

When a MODE bit is set to 0, the respective PORTA bit is configured for General Purpose I/O (GPIO).

When a PORTA MODE bit is set to 0, the respective bit in the DIR field is used to control the direction of the GPIO configuration; a DIR bit of 0 configures Input Mode while a DIR setting of 1 configures Output Mode.

When both MODE and DIR are set to 0, the PORTA bit is configured in Input Mode. The NET+ARM processor can read the current logic value on the PORTA bit by reading the state of the respective bit in the DATA field.

When the MODE bit is set to 0 and the DIR bit is set to 1, the PORTA bit is configured in Output Mode. The NET+ARM processor can set the current logic value on the PORTA bit by setting the state of the respective bit in the DATA field. Reading the DATA field when configured in Output Mode simply returns the current setting of the DATA field.

Special Function Mode

Many of the PORT A signals can be configured for Special Function I/O. Special Function Mode is used primarily for connecting the internal NET+ARM Serial Ports to the external interface pins; however, Special Function Mode also provides other miscellaneous features.

When the PORTA MODE bit is set to 1 and the PORTA DIR bit is set to 0, the PORTA bit is configured to operate in Special Function Input Mode. When the PORTA MODE bit is set to 1 and the PORTA DIR bit is also set to 1, the PORTA bit is configured to operate in Special Function Output Mode.

PORTA BIT	GPIO Mode		Special Function Mode	
	AMODE=0		AMODE = 1	
	ADIR=0	ADIR=1	ADIR=0	ADIR=1
PORTA7	GPIO IN	GPIO OUT		TXDA
PORTA6	GPIO IN	GPIO OUT	DREQ1*	DTRA*
PORTA5	GPIO IN	GPIO OUT		RTSA*
PORTA4	GPIO IN	GPIO OUT	SPI-S-CLK-IN-B* RXCA-IN	SPI-M-ENABLE-A* RXCA-OUT OUT1A*
PORTA3	GPIO IN	GPIO OUT	RXDA	
PORTA2	GPIO IN	GPIO OUT	DSRA*	DACK1*
PORTA1	GPIO IN	GPIO OUT	CTSA*	
PORTA0	GPIO IN	GPIO OUT	DCDA* DONE-IN1*	DONE-OUT1*

Table 8-3: PORT A Configuration

PORTA7

The PORTA7 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTA7 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTA7 signal provides the Transmit Data (TXD) signal for Serial Port A. The TXDA signal is used for Serial Transmit Data when configured to operate in UART, SPI, or HDLC modes.

The PORTA7 bit has no useful function when configured to operate in Special Function Input mode.

PORTA6

The PORTA6 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTA6 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTA6 signal provides the Data Terminal Ready (DTR) signal for Serial Port A. Control Register A within the SER Module configuration controls the state of DTR. The DTR signal configuration is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion before driving the PORTA6 I/O pad.

The PORTA6 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTA6 signal provides the active low DMA Request (DREQ*) input signal for DMA Channel 3. DMA Channel 3 only uses the DMA Request Input on PORTA6 when the REQ bit is set to 1 in the DMA Channel 3 Control Register.

PORTA5

The PORTA5 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTA5 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTA5 signal provides the Request To Send (RTS) signal for Serial Port A. Control Register A within the SER Module configuration controls the state of RTS. The RTS signal configuration is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion before driving the PORTA5 I/O pad.

The PORTA5 bit has no useful function when configured to operate in Special Function Input mode.

PORTA4

The PORTA4 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTA4 bit can be configured for Special Function Output. When configured for Special Function Output, PORTA4 provides one of 3 Serial Channel A features, depending upon the configuration of the Serial Channel.

When the Serial Channel is configured for SPI Master mode, PORTA4 Special Function Output drives the active low SPI Master Enable signal out the PORTA4 pin.

When the Serial Channel is not configured for SPI Master mode and the RXEXT bit is set to 1 in the Serial Channel Bit-Rate Register, then the Serial Channel Receiver Clock will be driven out the PORTA4 pin.

When the Serial Channel is not configured for SPI Master mode and the RXEXT bit is set to 0 in the Serial Channel Bit-Rate Register, then the Serial Channel General Purpose OUT 1 signal will be driven out the PORTA4 pin. Control Register A within the SER Module configuration controls the state of OUT1. The OUT1 signal configuration is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion before driving the PORTA4 I/O pad.

The PORTA4 bit can be configured for Special Function Input. When configured for Special Function Input, PORTA4 provides one of 2 Serial Channel A features, depending upon the configuration of the Serial Channel.

When the Serial Channel is configured for SPI Slave mode, PORTA4 Special Function Input receives the SPI Slave clock signal from the PORTA4 pin.

When the RXSRC bit is set to 1 in the Serial Channel Bit-Rate Register, then the Serial Channel Receiver Clock will be accepted from the PORTA4 pin.

PORTA3

The PORTA3 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTA3 bit has no useful function when configured to operate in Special Function Output mode.

The PORTA3 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTA3 signal provides the Receive Data (RXD) signal for Serial Port A. The RXDA signal is used for Serial Receive Data when configured to operate in UART, SPI, or HDLC modes.

PORTA2

The PORTA2 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTA2 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTA5 signal provides the active low DMA Acknowledge signal for DMA Channel 3. The DMA Acknowledge signal is driven active low when DMA Channel 3 is performing an external DMA cycle. The DMA Acknowledge is only used when the REQ bit is set in DMA Channel 3 Control Register.

The PORTA2 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTA2 signal provides the Data Set Ready (DSR) signal for Serial Port A. Status Register A within the SER Module configuration returns the state of RTS. The RTS signal status is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion after receiving the PORTA2 I/O pad.

PORTA1

The PORTA1 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTA1 bit has no useful function when configured to operate in Special Function Output mode.

The PORTA1 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTA1 signal provides the Clear To Send (CTS) signal for Serial Port A. Status Register A within the SER Module configuration returns the state of CTS. The CTS signal status is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion after receiving the PORTA1 I/O pad.

PORTA0

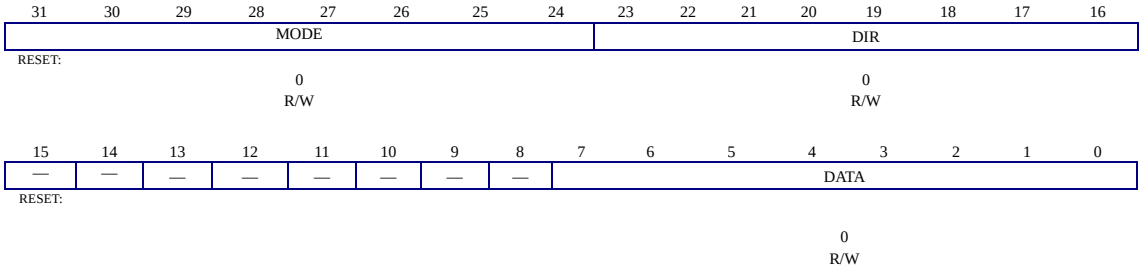
The PORTA0 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTA0 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTA0 signal provides the active low DMA DONE signal for DMA Channel 3. The DMA DONE signal is driven active low when DMA Channel 3 is performing an external DMA cycle and the current cycle represents the final transfer for the current DMA Buffer Descriptor. The DMA DONE is only used when the REQ bit is set in DMA Channel 3 Control Register.

The PORTA0 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTA0 signal provides the Data Carrier Detect (DCD) signal for Serial Port A. Status Register A within the SER Module configuration returns the state of DCD. The DCD signal status is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion after receiving the PORTA0 I/O pad. When the PORTB0 pin is being used for the DMA DONE input, the DCD information within the SER Module must be ignored.

The PORTA0 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTA0 signal provides the active low DMA DONE signal for DMA Channel 3. The DMA DONE signal is driven active low by an external peripheral when DMA Channel 3 is performing an external DMA cycle and this cycle represents the final transfer for a given block from the external peripheral. The DMA DONE is only used when the REQ bit is set in DMA Channel 3 Control Register. The DMA DONE input causes the current DMA Buffer Descriptor to be closed and an interrupt generated to the software (under program control).

Address = FFB0 0020



MODE

The 8-bit MODE field is used to individually configure each of the PORTA pins to operate in either GPIO mode or Special Function mode. Setting the MODE bit to 0 selects GPIO mode while a 1 selects Special Function Mode.

Each bit in the MODE field corresponds to one of the NET+ARM PORTA bits. D31 controls PORTA7, D30 controls PORTA6 … D24 controls PORTA0.

DIR

The 8-bit DIR field is used to individually configure each of the PORTA pins to operate in either Input Mode or Output Mode. Setting the DIR bit to 0 selects Input mode while a 1 selects Output Mode.

Each bit in the DIR field corresponds to one of the NET+ARM PORTA bits. D23 controls PORTA7, D22 controls PORTA6 … D16 controls PORTA0.

DATA

The 8-bit DATA field is used when a PORTA bit is configured to operate in GPIO mode. Reading the DATA field provides the current state of the NET+ARM GPIO signal (regardless of its configuration mode). Writing the DATA field defines the current state of the NET+ARM GPIO signal when the signal is defined to operate in GPIO output mode. Writing a DATA bit when configured in GPIO input mode or Special Function Mode has no effect.

Each bit in the DATA field corresponds to one of the NET+ARM PORTA bits. D07 controls PORTA7, D06 controls PORTA6 … D00 controls PORTA0.

8.2.8 PORT B Register

The PORTB Register is used to configure the personality of each PORTB General Purpose I/O pin. Each of the eight PORTB GPIO pins can be individually programmed to be one of the following:

- General Purpose Input
- General Purpose Output
- Special Function Input
- Special Function Output

Table 8.4 – PORT B Configuration describes the possible configurations for each of the PORTB signals. Many, but not all, of the PORTB signals can be configured for Special Function. Note that this table has four basic columns, each column denoting one of the four possible configurations for each PORTB bit.

The eight bits in the MODE field determine which of the PORTB bits are configured for GPIO or Special Function.

General Purpose I/O

When a MODE bit is set to 0, the respective PORTB bit is configured for General Purpose I/O (GPIO).

When a PORTB MODE bit is set to 0, the respective bit in the DIR field is used to control the direction of the GPIO configuration; a DIR bit of 0 configures Input Mode while a DIR setting of 1 configures Output Mode.

When both MODE and DIR are set to 0, the PORTB bit is configured in Input Mode. The NET+ARM processor can read the current logic value on the PORTB bit by reading the state of the respective bit in the DATA field.

When the MODE bit is set to 0 and the DIR bit is set to 1, the PORTB bit is configured in Output Mode. The NET+ARM processor can set the current logic value on the PORTB bit by setting the state of the respective bit in the DATA field. Reading the DATA field when configured in Output Mode simply returns the current setting of the DATA field.

Special Function Mode

Many of the PORTB signals can be configured for Special Function I/O. Special Function Mode is used primarily for connecting the internal NET+ARM Serial Ports to the external interface pins; however, Special Function Mode also provides other miscellaneous features.

When the PORTB MODE bit is set to 1 and the PORTB DIR bit is set to 0, the PORTB bit is configured to operate in Special Function Input Mode. When the PORTB MODE bit is set to 1 and the PORTB DIR bit is also set to 1, the PORTB bit is configured to operate in Special Function Output Mode.

PORTB BIT	GPIO Mode		Special Function Mode	
	BMODE=0		BMODE = 1	
	BDIR=0	BDIR=1	BDIR=0	BDIR=1
PORTB7	GPIO IN	GPIO OUT		TXDB
PORTB6	GPIO IN	GPIO OUT	DREQ2*	DTRB*
PORTB5	GPIO IN	GPIO OUT		RTSB*
PORTB4	GPIO IN	GPIO OUT	SPI-S-CLK-IN-B* RXCB-IN	SPI-M-ENABLE-B* RXCB-OUT OUT1B*
PORTB3	GPIO IN	GPIO OUT	RXDB	
PORTB2	GPIO IN	GPIO OUT	DSRB*	DACK2*
PORTB1	GPIO IN	GPIO OUT	CTSB*	
PORTB0	GPIO IN	GPIO OUT	DCDB* DONE-IN2*	DONE-OUT2*

Table 8-4: PORT B Configuration

PORTB7

The PORTB7 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTB7 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTB7 signal provides the Transmit Data (TXD) signal for Serial PORT B. The TXD signal is used for Serial Transmit Data when configured to operate in UART, SPI, or HDLC modes.

The PORTB7 bit has no useful function when configured to operate in Special Function Input mode.

PORTB6

The PORTB6 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTB6 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTB6 signal provides the Data Terminal Ready (DTR) signal for Serial PORT B. Control Register A within the SER Module configuration controls the state of DTR. The DTR signal configuration is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion before driving the PORTB6 I/O pad.

The PORTB6 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTB6 signal provides the active low DMA Request (DREQ*) input signal for DMA Channel 4. DMA Channel 4 only uses the DMA Request Input on PORTB6 when the REQ bit is set to 1 in the DMA Channel 4 Control Register.

PORTB5

The PORTB5 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTB5 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTB5 signal provides the Request To Send (RTS) signal for Serial PORT B. Control Register A within the SER Module configuration controls the state of RTS. The RTS signal configuration is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion before driving the PORTB5 I/O pad.

The PORTB5 bit has no useful function when configured to operate in Special Function Input mode.

PORTB4

The PORTB4 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTB4 bit can be configured for Special Function Output. When configured for Special Function Output, PORTB4 provides one of 3 Serial Channel B features, depending upon the configuration of the Serial Channel.

When the Serial Channel is configured for SPI Master mode, PORTB4 Special Function Output drives the active low SPI Master Enable signal out the PORTB4 pin.

When the Serial Channel is not configured for SPI Master mode and the RXEXT bit is set to 1 in the Serial Channel Bit-Rate Register, then the Serial Channel Receiver Clock will be driven out the PORTB4 pin.

When the Serial Channel is not configured for SPI Master mode and the RXEXT bit is set to 0 in the Serial Channel Bit-Rate Register, then the Serial Channel General Purpose OUT 1 signal will be driven out the PORTB4 pin. Control Register A within

the SER Module configuration controls the state of OUT1. The OUT1 signal configuration is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion before driving the PORTB4 I/O pad.

The PORTB4 bit can be configured for Special Function Input. When configured for Special Function Input, PORTB4 provides one of 2 Serial Channel B features, depending upon the configuration of the Serial Channel.

When the Serial Channel is configured for SPI Slave mode, PORTB4 Special Function Input receives the SPI Slave clock signal from the PORTB4 pin.

When the RXSRC bit is set to 1 in the Serial Channel Bit-Rate Register, then the Serial Channel Receiver Clock will be accepted from the PORTB4 pin.

PORTB3

The PORTB3 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTB3 bit has no useful function when configured to operate in Special Function Output mode.

The PORTB3 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTB3 signal provides the Receive Data (RXD) signal for Serial PORT B. The RXD signal is used for Serial Receive Data when configured to operate in UART, SPI, or HDLC modes.

PORTB2

The PORTB2 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTB2 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTB5 signal provides the active low DMA Acknowledge signal for DMA Channel 4. The DMA Acknowledge signal is driven active low when DMA Channel 4 is performing an external DMA cycle. The DMA Acknowledge is only used when the REQ bit is set in DMA Channel 4 Control Register.

The PORTB2 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTB2 signal provides the Data Set Ready (DSR) signal for Serial PORT B. Status Register A within the SER Module configuration returns the state of RTS. The RTS signal status is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion after receiving the PORTB2 I/O pad.

PORTB1

The PORTB1 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTB1 bit has no useful function when configured to operate in Special Function Output mode.

The PORTB1 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTB1 signal provides the Clear To Send (CTS) signal for Serial PORT B. Status Register A within the SER Module configuration returns the state of CTS. The CTS signal status is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion after receiving the PORTB1 I/O pad.

PORTB0

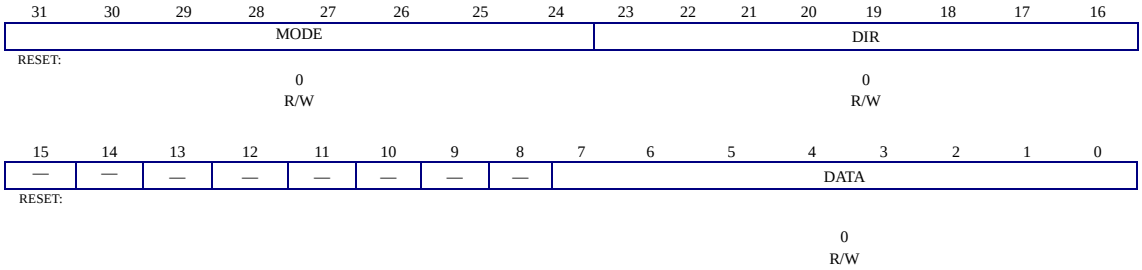
The PORTB0 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTB0 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTB0 signal provides the active low DMA DONE signal for DMA Channel 4. The DMA DONE signal is driven active low when DMA Channel 4 is performing an external DMA cycle and the current cycle represents the final transfer for the current DMA Buffer Descriptor. The DMA DONE is only used when the REQ bit is set in DMA Channel 4 Control Register.

The PORTB0 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTB0 signal provides the Data Carrier Detect (DCD) signal for Serial PORT B. Status Register A within the SER Module configuration returns the state of DCD. The DCD signal status is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion after receiving the PORTB0 I/O pad. When the PORTB0 pin is being used for the DMA DONE input, the DCD information within the SER Module must be ignored.

The PORTB0 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTB0 signal provides the active low DMA DONE input signal for DMA Channel 4. The DMA DONE signal is driven active low by an external peripheral when DMA Channel 4 is performing an external DMA cycle and this cycle represents the final transfer for a given block from the external peripheral. The DMA DONE is only used when the REQ bit is set in DMA Channel 4 Control Register. The DMA DONE input causes the current DMA Buffer Descriptor to be closed and an interrupt generated to the software (under program control).

Address = FFB0 0024



MODE

The 8-bit MODE field is used to individually configure each of the PORTB pins to operate in either GPIO mode or Special Function mode. Setting the MODE bit to 0 selects GPIO mode while a 1 selects Special Function Mode.

Each bit in the MODE field corresponds to one of the NET+ARM PORTB bits. D31 controls PORTB7, D30 controls PORTB6 … D24 controls PORTB0.

DIR

The 8-bit DIR field is used to individually configure each of the PORTB pins to operate in either Input Mode or Output Mode. Setting the DIR bit to 0 selects Input mode while a 1 selects Output Mode.

Each bit in the DIR field corresponds to one of the NET+ARM PORTB bits. D23 controls PORTB7, D22 controls PORTB6 … D16 controls PORTB0.

DATA

The 8-bit DATA field is used when a PORTB bit is configured to operate in GPIO mode. Reading the DATA field provides the current state of the NET+ARM GPIO signal (regardless of its configuration mode). Writing the DATA field defines the current state of the NET+ARM GPIO signal when the signal is defined to operate in GPIO output mode. Writing a DATA bit when configured in GPIO input mode or Special Function Mode has no effect.

Each bit in the DATA field corresponds to one of the NET+ARM PORTB bits. D07 controls PORTB7, D06 controls PORTB6 … D00 controls PORTB0.

8.2.9 PORT C Register

The PORTC Register is used to configure the personality of each PORTC General Purpose I/O pin. Each of the eight PORTC GPIO pins can be individually programmed to be one of the following:

- General Purpose Input
- General Purpose Output
- Special Function Input
- Special Function Output

Table 8.4 – PORT C Configuration describes the possible configurations for each of the PORTC signals. Many, but not all, of the PORTC signals can be configured for Special Function. Note that this table has four basic columns, each column denoting one of the four possible configurations for each PORTC bit.

The eight bits in the MODE field determine which of the PORTC bits are configured for GPIO or Special Function.

General Purpose I/O

When a MODE bit is set to 0, the respective PORTC bit is configured for General Purpose I/O (GPIO).

When a PORTC MODE bit is set to 0, the respective bit in the DIR field is used to control the direction of the GPIO configuration; a DIR bit of 0 configures Input Mode while a DIR setting of 1 configures Output Mode.

When both MODE and DIR are set to 0, the PORTC bit is configured in Input Mode. The NET+ARM processor can read the current logic value on the PORTC bit by reading the state of the respective bit in the DATA field.

When the MODE bit is set to 0 and the DIR bit is set to 1, the PORTC bit is configured in Output Mode. The NET+ARM processor can set the current logic value on the PORTC bit by setting the state of the respective bit in the DATA field. Reading the DATA field when configured in Output Mode simply returns the current setting of the DATA field.

Special Function Mode

Many of the PORTC signals can be configured for Special Function I/O. Special Function Mode is used primarily for connecting the internal NET+ARM Serial Ports to the external interface pins; however, Special Function Mode also provides other miscellaneous features.

When the PORTC MODE bit is set to 1 and the PORTC DIR bit is set to 0, the PORTC bit is configured to operate in Special Function Input Mode. When the

PORTC MODE bit is set to 1 and the PORTC DIR bit is also set to 1, the PORTC bit is configured to operate in Special Function Output Mode.

PORTC BIT	GPIO Mode		Special Function Mode	
	CMODE=0		CMODE = 1	
	CDIR=0	CDIR=1	CDIR=0	CDIR=1
PORTC7	GPIO IN	GPIO OUT	SPI-S-ENABLE-A* TXCA-IN	SPI-M-CLK-OUT-A TXCA-OUT OUT2A*
PORTC6	GPIO IN	GPIO OUT	RIA*	IRQ-OUT*
PORTC5	GPIO IN	GPIO OUT	SPI-S-ENABLE-B* TXCB-IN	SPI-M-CLK-OUT-B* TXCB-OUT OUT2B*
PORTC4	GPIO IN	GPIO OUT	RIB*	RESET-OUT*
PORTC3	GPIO IN	GPIO OUT	CI3-0	CI3-1
PORTC2	GPIO IN	GPIO OUT	CI2-0	CI2-1
PORTC1	GPIO IN	GPIO OUT	CI1-0	CI1-1
PORTC0	GPIO IN	GPIO OUT	CI0-0	CI0-1

Table 8-5: PORT C Configuration

PORTC7

The PORTC7 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTC7 bit can be configured for Special Function Output. When configured for Special Function Output, PORTC7 provides one of 3 Serial Channel A features, depending upon the configuration of the Serial Channel.

When the Serial Channel is configured for SPI Master mode, PORTC7 Special Function Output drives the SPI Master Clock signal out the PORTC7 pin.

When the Serial Channel is not configured for SPI Master mode and the TXEXT bit is set to 1 in the Serial Channel Bit-Rate Register, then the Serial Channel Transmit Clock will be driven out the PORTC7 pin.

When the Serial Channel is not configured for SPI Master mode and the TXEXT bit is set to 0 in the Serial Channel Bit-Rate Register, then the Serial Channel General Purpose OUT 2 signal will be driven out the PORTC7 pin. Control Register A within the SER Module configuration controls the state of OUT2. The OUT2 signal configuration is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion before driving the PORTC7 I/O pad.

The PORTC7 bit can be configured for Special Function Input. When configured for Special Function Input, PORTC7 provides one of 2 Serial Channel A features, depending upon the configuration of the Serial Channel.

When the Serial Channel is configured for SPI Slave mode, PORTC7 Special Function Input receives the active low SPI Slave Enable signal from the PORTC7 pin.

When the TXSRC bit is set to 1 in the Serial Channel Bit-Rate Register, then the Serial Channel A Receiver Clock will be accepted from the PORTC7 pin.

PORTC6

The PORTC6 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTC6 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTC6 signal provides the IRQ-OUT* signal. The IRQ-OUT* signal provides an active low interrupt request signal, which indicates an active enabled interrupt is pending at the output of the GEN Module Interrupt Controller. The IRQ-OUT* signal emulates the same signal given to the ARM processor. The IRQ-OUT* signal is typically used in environments where the NET+ARM is used in the slave coprocessor configuration where the internal ARM processor is disabled during bootstrap.

The PORTC6 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTC6 signal provides the Ring Indicator (RI) signal for Serial Channel A. Status Register A within the SER Module configuration returns the state of RI. The RI signal status is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion after receiving the PORTC6 I/O pad.

PORTC5

The PORTC5 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTC5 bit can be configured for Special Function Output. When configured for Special Function Output, PORTC5 provides one of 3 Serial Channel B features, depending upon the configuration of the Serial Channel.

When the Serial Channel is configured for SPI Master mode, PORTC5 Special Function Output drives the SPI Master Clock signal out the PORTC5 pin.

When the Serial Channel is not configured for SPI Master mode and the TXEXT bit is set to 1 in the Serial Channel Bit-Rate Register, then the Serial Channel Transmit Clock will be driven out the PORTC5 pin.

When the Serial Channel is not configured for SPI Master mode and the TXEXT bit is set to 0 in the Serial Channel Bit-Rate Register, then the Serial Channel General Purpose OUT 2 signal will be driven out the PORTC5 pin. Control Register A within the SER Module configuration controls the state of OUT2. The OUT2 signal configuration is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion before driving the PORTC5 I/O pad.

The PORTC5 bit can be configured for Special Function Input. When configured for Special Function Input, PORTC5 provides one of 2 Serial Channel B features, depending upon the configuration of the Serial Channel.

When the Serial Channel is configured for SPI Slave mode, PORTC5 Special Function Input receives the active low SPI Slave Enable signal from the PORTC5 pin.

When the TXSRC bit is set to 1 in the Serial Channel Bit-Rate Register, then the Serial Channel B Receiver Clock will be accepted from the PORTC5 pin.

PORTC4

The PORTC4 bit can be configured for GPIO Input mode or GPIO Output Mode.

The PORTC4 bit can be configured for Special Function Output. When configured for Special Function Output, the PORTC4 signal provides the RESET-OUT* signal. The RESET-OUT* signal provides an active low reset indicator that is activated during either a Software Reset or an ENI Host reset condition.

The PORTC4 bit can be configured for Special Function Input. When configured for Special Function Input, the PORTC4 signal provides the Ring Indicator (RI) signal for Serial Channel B. Status Register A within the SER Module configuration returns the state of RI. The RI signal status is active high inside the NET+ARM and active low outside the NET+ARM. The GEN Module performs inversion after receiving the PORTC4 I/O pad.

PORTC3

The PORTC3 bit can be configured for GPIO Input mode or GPIO Output Mode.

When PORTC3 is configured for Special Function mode, the PORTC3 signal becomes an edge sensitive interrupt detector. While the corresponding DIR bit is set to 0, the PORTC3 pin will detect high-to-low transitions on PORTC3 and assert an

interrupt using the PORTC PC3 bit in the GEN Module Interrupt Control Register. An active high interrupt status condition is reported in the corresponding PORTC D3 bit position. The interrupt condition is acknowledged by writing a 1 to the D3 position of the DATA field. A new interrupt will occur on the next high-to-low transition on PORTC3. Low-to-high transitions can also be detected by setting the corresponding DIR bit to 1.

PORTC2

The PORTC2 bit can be configured for GPIO Input mode or GPIO Output Mode.

When PORTC2 is configured for Special Function mode, the PORTC2 signal becomes an edge sensitive interrupt detector. While the corresponding DIR bit is set to 0, the PORTC2 pin will detect high-to-low transitions on PORTC2 and assert an interrupt using the PORTC PC2 bit in the GEN Module Interrupt Control Register. An active high interrupt status condition is reported in the corresponding PORTC D2 bit position. The interrupt condition is acknowledged by writing a 1 to the D2 position of the DATA field. A new interrupt will occur on the next high-to-low transition on PORTC2. Low-to-high transitions can also be detected by setting the corresponding DIR bit to 1.

PORTC1

The PORTC1 bit can be configured for GPIO Input mode or GPIO Output Mode.

When PORTC1 is configured for Special Function mode, the PORTC1 signal becomes an edge sensitive interrupt detector. While the corresponding DIR bit is set to 0, the PORTC1 pin will detect high-to-low transitions on PORTC1 and assert an interrupt using the PORTC PC1 bit in the GEN Module Interrupt Control Register. An active high interrupt status condition is reported in the corresponding PORTC D1 bit position. The interrupt condition is acknowledged by writing a 1 to the D1 position of the DATA field. A new interrupt will occur on the next high-to-low transition on PORTC1. Low-to-high transitions can also be detected by setting the corresponding DIR bit to 1.

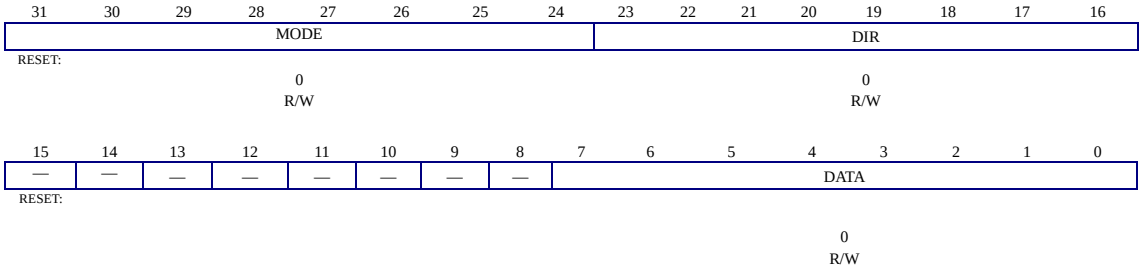
PORTC0

The PORTC0 bit can be configured for GPIO Input mode or GPIO Output Mode.

When PORTC0 is configured for Special Function mode, the PORTC0 signal becomes an edge sensitive interrupt detector. While the corresponding DIR bit is set to 0, the PORTC0 pin will detect high-to-low transitions on PORTC0 and assert an interrupt using the PORTC PC0 bit in the GEN Module Interrupt Control Register. An active high interrupt status condition is reported in the corresponding PORTC D0 bit position. The interrupt condition is acknowledged by writing a 1 to the D0 position of the DATA field. A new interrupt will occur on the next high-to-low transition on

PORTC0. Low-to-high transitions can also be detected by setting the corresponding DIR bit to 1.

Address = FFB0 0028



MODE

The 8-bit MODE field is used to individually configure each of the PORTC pins to operate in either GPIO mode or Special Function mode. Setting the MODE bit to 0 selects GPIO mode while a 1 selects Special Function Mode.

Each bit in the MODE field corresponds to one of the NET+ARM PORTC bits. D31 controls PORTC7, D30 controls PORTC6 … D24 controls PORTC0.

DIR

The 8-bit DIR field is used to individually configure each of the PORTC pins to operate in either Input Mode or Output Mode. Setting the DIR bit to 0 selects Input mode while a 1 selects Output Mode.

Each bit in the DIR field corresponds to one of the NET+ARM PORTC bits. D23 controls PORTC7, D22 controls PORTC6 … D16 controls PORTC0.

DATA

The 8-bit DATA field is used when a PORTC bit is configured to operate in GPIO mode. Reading the DATA field provides the current state of the NET+ARM GPIO signal (regardless of its configuration mode). Writing the DATA field defines the current state of the NET+ARM GPIO signal when the signal is defined to operate in GPIO output mode. Writing a DATA bit when configured in GPIO input mode or Special Function Mode has no effect.

When the lower four bits are used in Special Function mode, the corresponding DATA bits are used to indicate a pending interrupt condition. Pending interrupts are latched when an edge transition is detected. A pending interrupt is identified with a 1 in the DATA field. Writing a 1 to the same bit position within the DATA field clears the interrupt condition.

Each bit in the DATA field corresponds to one of the NET+ARM PORTC bits. D07 controls PORTC7, D06 controls PORTC6 … D00 controls PORTC0.

8.2.10 Interrupt Enable Register

The five Interrupt Control Registers make up the functionality for the NET+ARM Interrupt Controller. For more details concerning the NET+ARM Interrupt Controller, please refer to Section *3.3 ARM Exceptions*.

There are two wires that go into the ARM7 CPU core used for interrupting the processor. These lines are:

- IRQ (normal interrupt)
- FIRQ (fast interrupt)

They are basically the same except for the fact that FIRQ can interrupt IRQ. The purpose of the FIRQ line is to add a simple two-tier priority scheme to the interrupt system. Most all sources of interrupts on the NET+ARM come from the IRQ line. The only potential sources for FIRQ interrupts on the NET+ARM are the two built in timers and the watchdog timer. The timers are controlled using the Timer Control registers in the GEN module (0xFFB0 0010 / 18). The watchdog timer is controlled using the System Control Register in the GEN module (0xFFB0 0000).

Interrupts may come from many different sources on the NET+ARM and are managed by the interrupt controller within the GEN module (see Figure 3-1). Interrupts can be enabled/disabled on a per-source basis using the Interrupt Enable Register (0xFFB0 0030). This register serves as a mask for the various interrupt sources and ultimately controls whether or not an interrupt from a NET+ARM module can reach the IRQ line.

There are two read-only registers in the interrupt controller:

- The first is the Interrupt Status Register Raw that indicates the source of a NET+ARM interrupt regardless of the Interrupt Enable Register's state. All interrupts that are active in their respective module will be visible in the Interrupt Status Register Raw (0xFFB0 0038).
- The second read-only register is the Interrupt Status Register Enabled (0xFFB0 0034). This register identifies the current state of all interrupt sources that are enabled and is defined by performing a logical *AND* of the Interrupt Status Register Raw and the Interrupt Enable Register. All of the bits in the Interrupt Status Register Enabled are then *OR-ed* together; the output of which is fed directly to the IRQ line that then interrupts the ARM.

All of the five Interrupt Control Registers use the same 32-bit register layout definition as defined below.

Address = FFB0 0030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA1	DMA2	DMA3	DMA4	DMA5	DMA6	DMA7	DMA	DMA9	DMA10	ENI PORT 1	ENI PORT 2	ENI PORT 3	ENI PORT 4	ENET RX	ENET TX
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SER 1 RX	SER 1 TX	SER2 RX	SER 2 TX	—	—	—	—	—	Watch Dog	Timer 1	Timer 2	PortC PC3	PortC PC2	PortC PC1	PortC PC0
RESET:															
0	0	0	0						0	0	0	0	0	0	0
R/W	R/W	R/W	R/W						R/W	R/W	R/W	R/W	R/W	R/W	R/W

DMA 1

The DMA 1 bit position corresponds to interrupts source by DMA Channel 1. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

DMA 2

The DMA 2 bit position corresponds to interrupts source by DMA Channel 2. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

DMA 3

The DMA 3 bit position corresponds to interrupts source by DMA Channel 3. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

DMA 4

The DMA 4 bit position corresponds to interrupts source by DMA Channel 4. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

DMA 5

The DMA 5 bit position corresponds to interrupts source by DMA Channel 5. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

DMA 6

The DMA 6 bit position corresponds to interrupts source by DMA Channel 6. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

DMA 7

The DMA 7 bit position corresponds to interrupts source by DMA Channel 7. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

DMA 8

The DMA 8 bit position corresponds to interrupts source by DMA Channel 8. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

DMA 9

The DMA 9 bit position corresponds to interrupts source by DMA Channel 9. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

DMA 10

The DMA 10 bit position corresponds to interrupts source by DMA Channel 10. Please refer to Section *4.4.3 DMA Status/Interrupt Enable Register* for more details.

ENI Port 1

The ENI Port 1 bit position corresponds to interrupt sourced by the ENI controller. This bit position corresponds to either 1284 Parallel Port #1 or the ENI FIFO Receiver, depending upon configuration of the ENI Module. Please refer to Section *6.5.9 ENI Module Interrupts* for more details.

ENI Port 2

The ENI Port 2 bit position corresponds to interrupt sourced by the ENI controller. This bit position corresponds to either 1284 Parallel Port #2 or the ENI FIFO Transmitter, depending upon configuration of the ENI Module. Please refer to Section *6.5.9 ENI Module Interrupts* for more details.

ENI Port 3

The ENI Port 3 bit position corresponds to interrupt sourced by the ENI controller. This bit position corresponds to either 1284 Parallel Port #3 or the ENI Shared

Register INTIOF Interrupt Condition, depending upon configuration of the ENI Module. Please refer to Section *6.5.9 ENI Module Interrupts* for more details.

ENI Port 4

The ENI Port 4 bit position corresponds to interrupt sourced by the ENI controller. This bit position corresponds to either 1284 Parallel Port #4 or the ENI Bus Error (Data Abort) Interrupt Condition, depending upon configuration of the ENI Module. Please refer to Section *6.5.9 ENI Module Interrupts* for more details.

ENET RX

The ENET RX bit position corresponds to an interrupt sourced by the Ethernet Receiver. Please refer to Section *5.3.2 Ethernet General Status Register* for more details.

ENET TX

The ENET TX bit position corresponds to an interrupt sourced by the Ethernet Transmitter. Please refer to Section *5.3.2 Ethernet General Status Register* for more details.

SER 1 RX

The SER 1 RX bit position corresponds to an interrupt sourced by the Serial Channel A receiver. Please refer to Section *7.5.1 Serial Channel Control Register A* and *7.5.2 Serial Channel Control Register B* for more details.

SER 1 TX

The SER 1 TX bit position corresponds to an interrupt sourced by the Serial Channel A transmitter. Please refer to Section *7.5.1 Serial Channel Control Register A* and *7.5.2 Serial Channel Control Register B* for more details.

SER 2 RX

The SER 2 RX bit position corresponds to an interrupt sourced by the Serial Channel B receiver. Please refer to Section *7.5.1 Serial Channel Control Register A* and *7.5.2 Serial Channel Control Register B* for more details.

SER 2 TX

The SER 2 TX bit position corresponds to an interrupt sourced by the Serial Channel B transmitter. Please refer to Section *7.5.1 Serial Channel Control Register A* and *7.5.2 Serial Channel Control Register B* for more details.

WATCH DOG

The WATCH DOG bit position corresponds to an interrupt condition sourced by the watchdog timer. Please refer to Section *8.2.1 System Control Register* for more details regarding the watchdog timer.

TIMER 1

The TIMER 1 bit position corresponds to an interrupt condition sourced by the TIMER 1 module. Please refer to Section *8.2.6 Timer Status Register* for more details on timer interrupts.

TIMER 2

The TIMER 2 bit position corresponds to an interrupt condition sourced by the TIMER 2 module. Please refer to Section *8.2.6 Timer Status Register* for more details on timer interrupts.

PORTC PC3

The PORTC PC3 bit position corresponds to an interrupt condition caused by an edge transition detected on the PORTC3 pin. Please refer to Section *8.2.9 PORT C Register* for more information regarding the PORTC edge detection interrupts.

PORTC PC2

The PORTC PC2 bit position corresponds to an interrupt condition caused by an edge transition detected on the PORTC2 pin. Please refer to Section *8.2.9 PORT C Register* for more information regarding the PORTC edge detection interrupts.

PORTC PC1

The PORTC PC1 bit position corresponds to an interrupt condition caused by an edge transition detected on the PORTC1 pin. Please refer to *8.2.9 PORT C Register* for more information regarding the PORTC edge detection interrupts.

PORTC PC0

The PORTC PC0 bit position corresponds to an interrupt condition caused by an edge transition detected on the PORTC0 pin. Please refer to Section *8.2.9 PORT C Register* for more information regarding the PORTC edge detection interrupts.

8.2.11 Interrupt Enable Register - Set

The Interrupt Enable - Set register is a 32-bit write-only register. This register sets specific bits without affecting the other bits. Writing a one to a bit position in this register sets the respective bit in the Interrupt Enable Register. A zero in any bit position has no affect.

Address = FFB0 0034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA1	DMA2	DMA3	DMA4	DMA5	DMA6	DMA7	DMA	DMA9	DMA10	ENI PORT 1	ENI PORT 2	ENI PORT 3	ENI PORT 4	ENET RX	ENET TX
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SER 1 RX	SER 1 TX	SER2 RX	SER 2 TX	—	—	—	—	—	Watch Dog	Timer 1	Timer 2	PortC PC3	PortC PC2	PortC PC1	PortC PC0
RESET:															
0	0	0	0						0	0	0	0	0	0	0
R/W	R/W	R/W	R/W						R/W	R/W	R/W	R/W	R/W	R/W	R/W

8.2.12 Interrupt Enable Register - Clear

The Interrupt Enable - Clear register is a 32-bit write-only register. This register clears specific bits without affecting the other bits. Writing a 1 to a bit position in this register clears the respective bit in the Interrupt Enable Register. A zero in any bit position has no affect.

Address = FFB0 0038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA1	DMA2	DMA3	DMA4	DMA5	DMA6	DMA7	DMA	DMA9	DMA10	ENI PORT 1	ENI PORT 2	ENI PORT 3	ENI PORT 4	ENET RX	ENET TX
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SER 1 RX	SER 1 TX	SER2 RX	SER 2 TX	—	—	—	—	—	Watch Dog	Timer 1	Timer 2	PortC PC3	PortC PC2	PortC PC1	PortC PC0
RESET:															
0	0	0	0						0	0	0	0	0	0	0
R/W	R/W	R/W	R/W						R/W	R/W	R/W	R/W	R/W	R/W	R/W

8.2.13 Interrupt Status Register - Enabled

The Interrupt Status Register - Enabled register is read-only and identifies the current state of all interrupt sources that are currently enabled in the Interrupt Enable Register.

Address = FFB0 0034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA1	DMA2	DMA3	DMA4	DMA5	DMA6	DMA7	DMA	DMA9	DMA10	ENI PORT 1	ENI PORT 2	ENI PORT 3	ENI PORT 4	ENET RX	ENET TX
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SER 1 RX	SER 1 TX	SER2 RX	SER 2 TX	—	—	—	—	—	Watch Dog	Timer 1	Timer 2	PortC PC3	PortC PC2	PortC PC1	PortC PC0
RESET:															
0	0	0	0						0	0	0	0	0	0	0
R/W	R/W	R/W	R/W						R/W	R/W	R/W	R/W	R/W	R/W	R/W

8.2.14 Interrupt Status Register - Raw

The Interrupt Status Register - Raw register is read-only and identifies the current state of all interrupt sources regardless of the value in the Interrupt Enable Register.

Address = FFB0 0038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA1	DMA2	DMA3	DMA4	DMA5	DMA6	DMA7	DMA	DMA9	DMA10	ENI PORT 1	ENI PORT 2	ENI PORT 3	ENI PORT 4	ENET RX	ENET TX
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SER 1 RX	SER 1 TX	SER2 RX	SER 2 TX	—	—	—	—	—	Watch Dog	Timer 1	Timer 2	PortC PC3	PortC PC2	PortC PC1	PortC PC0
RESET:															
0	0	0	0						0	0	0	0	0	0	0
R/W	R/W	R/W	R/W						R/W	R/W	R/W	R/W	R/W	R/W	R/W

Chapter 9
BUS Controller Module

The bus controller is responsible for moving data between the BBus and the external system bus. The bus controller can support dynamic bus sizing for any logical addresses selected by the memory controller. The bus controller packs 8-bit bytes and 16-bit words into the proper location on the 32-bit BBus.

Figure 9-1 provides a simple block diagram of the bus controller module. The BUS module works in close concert with the MEM module to support the dynamic bus sizing feature. The BUS module is responsible for performing the byte/word data packing when accessing less than a 32-bit data operand.

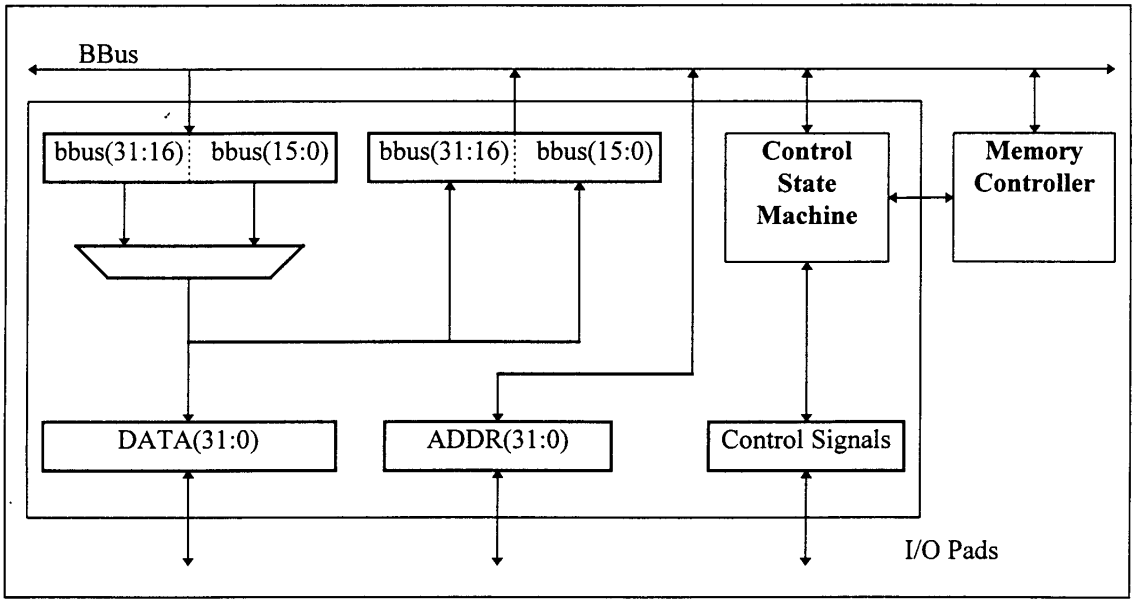


Figure 9-1: Bus Controller Module

When the NET+ARM chip interfaces with an external bus master or CPU, then the bus controller is responsible for providing arbitration of the system bus. The bus controller can be configured to allow the NET+ARM chip to be the default system master (other devices must request ownership from the NET+ARM chip). Alternately,

the bus controller can be configured to request ownership from an external default bus master (the NET+ARM chip must request ownership from another device).

An external bus master cannot access any NET+ARM chip internal registers unless the BUSER bit is set in the system control register.

The effective operation of the system bus interface (BCLK) must be configured proportional to the operation speed of the CPU Core. The system bus must be configurable to operate in full, half, and quarter speed modes.

The BUS module does not contain its own configuration block. All configurations for the BUS module are stored within the GEN module.

9.1 Data Transfer Modes

The bus controller works in concert with the memory controller to access external devices using the system bus. The bus controller can also interface with devices that are not controlled by the memory controller, however, the features are limited.

External system bus devices that use the memory controller for chip-select and acknowledge generation can support both dynamic bus sizing and burst line cycles.

Other devices on the system bus that do not use the memory controller for chip-select and acknowledge generation cannot use dynamic bus sizing. The bus controller assumes that these external devices are 32-bits in size. The bus controller can support byte operations to these devices provided the external device supports byte accesses.

9.2 Bus Operation

The bus controller provides a simple synchronous bus. All outputs change on the rising or falling edge of BCLK. All inputs are sampled on the rising edge of BCLK. All memory cycles start with the assertion of TS* and complete with the reception of an TA*. For memory cycles controlled by the memory controller, the TA* signal is generated within the memory controller.

9.3 Peripheral Cycle Termination

All peripherals terminate a memory cycle using a combination of TA* and TEA*. Peripherals include those controlled by the internal MEM module and those controlled using external hardware.

The TA* signal is always used to terminate a peripheral memory cycle. TEA*, in absence of TA*, signals a bus failure. TEA*, in conjunction with TA*, signals the end of a peripheral burst cycle.

Peripherals controlled by the MEM module can always support the peripheral burst cycle protocol. The MEM module always signals the end of a peripheral memory cycle using TA*. When the TEALAST bit is set in the system control register, the MEM module signals the end of a peripheral burst cycle by asserting TEA*. Setting TEALAST is not required for MEM module bursting. The TEALAST setting is required to allow signaling via TEA*.

Peripherals controlled using external hardware also signal peripheral cycle completion using TA*. For peripherals controlled using external hardware to take advantage of the burst protocol, the TEALAST bit must be set in the system control register. Internal bus masters do not burst to external peripherals controlled by external hardware unless the TEALAST bit is set. Internal bus masters use the TEA* signal to determine when a burst operation is complete.

Table 9-1 shows how the TA* and TEA* signals are used to signal peripheral cycle completion.

Signal	TA*	TEA*
Burst Cycle Termination	0	0
Normal Termination	0	1
Error Termination	1	0
Waiting for Completion	1	1

Table 9-1: Peripheral Cycle Termination

9.4 Dynamic Bus Sizing

An external device can take advantage of dynamic bus sizing provided the memory controller generates the chip-select and acknowledge for that device. The memory controller contains configuration identifying the data size of the external device. The external device can be configured for 8, 16, or 32-bit operands. If a word operand is requested from an external device configured as 8 bits in size, the bus controller automatically performs four external memory cycles.

9.5 Normal Operand Cycles

Normal operand cycles are used for single long, word, or byte access cycles. Figure 9-2 provides a diagram showing normal read and write cycles. The cycle begins with the assertion of TS* and ends with the reception of TA*. Additional wait-states can be inserted by delaying the assertion of TA*. The memory controller provides the ability to control the number of wait states.

The byte enable signals (BE3*, BE2*, BE1*, BE0*) control which portion of the data bus is active. The BE3* and BE2* signals are used to select a certain byte within a 16-bit device. For 8-bit devices, only BE3* is asserted; never BE2*.

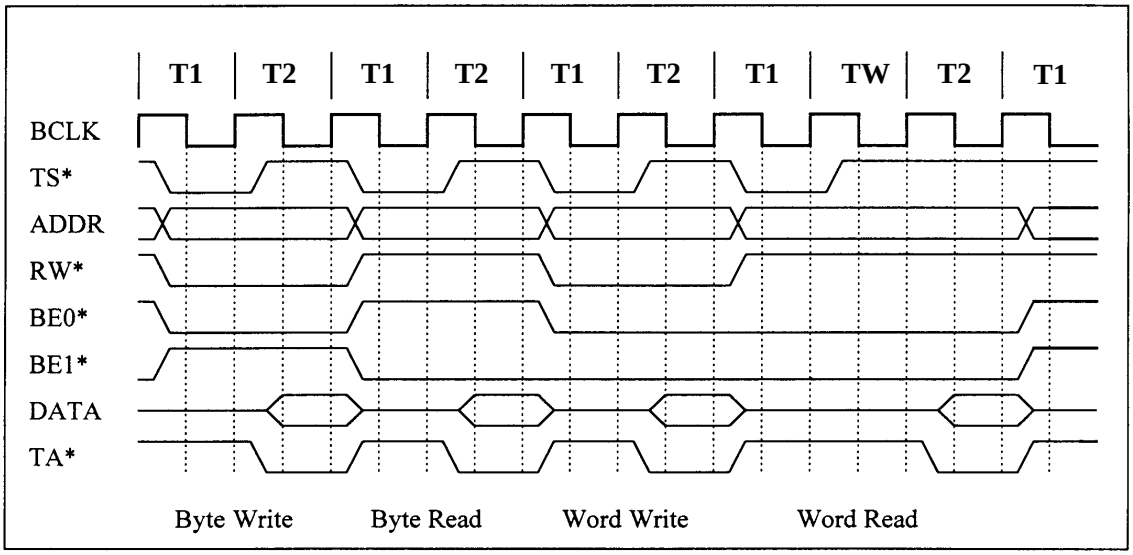


Figure 9-2: Normal System Bus Cycles

9.6 Burst Cycles

Figure 9-3 provides an example diagram showing burst read and write cycles. The burst cycle is controlled by configuration within the memory controller. The burst cycle starts with the assertion of TS* and continues until four long words are read. This requires 8 word reads to occur since the peripheral in this example is 16-bits wide. Each individual read cycle must be terminated with the assertion of TA*. The memory controller can be configured with separate wait states for the first cycle and all remaining cycles.

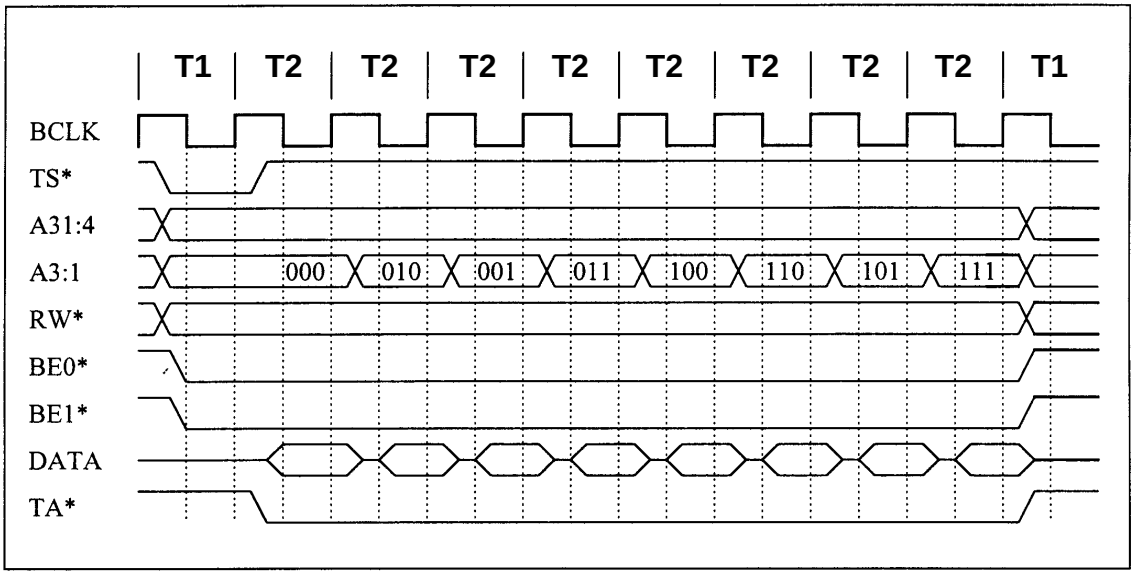


Figure 9-3: Burst Cycles

9.7 Read-Modify-Write Cycles

Read-Modify-Write, sometimes called LOCKED cycles, are required to support a hardware semaphore operation. An external bus master can accomplish this type of cycle by holding the BUSY* input active low between the two memory cycles required to be locked together.

9.8 System Bus Arbiter

The system bus arbiter can be configured to be internal or external. This configuration is provided by the IARB bit in the system control register. The IARB bit can be automatically set during system bootstrap (refer to section *11.4 NET+ARM Chip Bootstrap Initialization*).

The system bus arbiter uses a fully synchronous protocol using the BCLK signal at the system bus interface. When configured to use an internal arbiter, the BR* signal is an input and the BG* signal is an output. When configured to use an external arbiter, the BR* signal is an output while BG* is an input. In both cases, the BUSY* signal is bi-directional. As an input, BUSY* indicates an external master currently owns the bus. As an output, the NET+ARM chip drives BUSY* to indicate that the NET+ARM chip is the current system bus master. There is always one BCLK cycle where BUSY* must be high between active system bus masters.

A bus master drives its BR* signal active to request ownership of the bus. The arbiter provides the BG* signal to the device that is to be the next bus master. Upon receiving BG*, the master must hold BR* active low and wait for BUSY* to become inactive high. When BUSY* becomes inactive, the new master can assume ownership of the system bus, drive BUSY* low, and drive BR* to an inactive state. The new master continues to drive BUSY* active while it executes memory cycles.

The BBus and system bus are coupled such that the bus owner has control of both buses simultaneously. An internal master cannot own the BBus while an external master owns the system bus. This design approach removes the potential for a bus dead-lock condition.

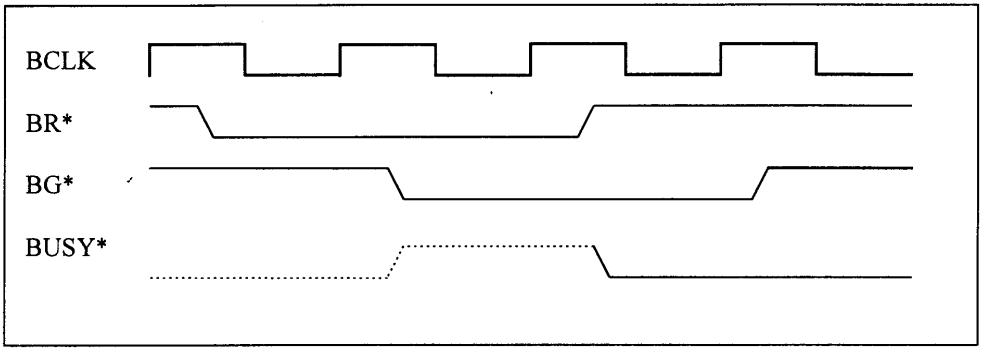


Figure 9-4: System Bus Arbitration Protocol

9.9 External Bus Master Support

The NET+ARM chip provides support for interfacing with an external bus master. The external bus master can request and acquire ownership of the system bus. Upon acquiring ownership, the external bus master can execute memory cycles using the standard NET+ARM chip bus protocol. The external bus master can execute memory cycles to peripherals that are controlled by the NET+ARM chip internal memory controller.

9.9.1 Signal Description

This section briefly describes the signals that an external bus master design must interface with.

BR*Bus Request

The external bus master asserts the BR* input active low to request ownership of the NET+ARM chip system memory bus. The BR* input must be synchronized to the BCLK output of the NET+ARM chip. The NET+ARM chip samples the BR* input using the rising edge of BCLK. The requesting device must drive the BR* signal active low until the requesting device drives the BUSY* signal low to assume ownership of the bus.

BG* Bus Grant

The NET+ARM chip asserts the BG* output active low to indicate to the External Bus Master that it can assume ownership of the NET+ARM chip system memory bus one BCLK cycle after the current bus master relinquishes the bus. The current bus master relinquishes the bus by asserting the BUSY* signal inactive high. The BG* output is synchronized to the BCLK signal and changes state on the rising edge of BCLK.

BUSY*Bus Busy

The BUSY* signal is driven active low by the current bus master to indicate to all other potential bus masters that the bus is owned. The NET+ARM chip system memory bus ownership changes on the rising edge of BCLK in which the BUSY* signal is inactive high. The external bus master knows that it is the next bus master when the BG* signal is asserted active low by the NET+ARM chip. The next system bus master assumes ownership of the bus by asserting BUSY* active low using a later rising edge of BCLK. The BUSY* output is synchronized to the BCLK signal and changes state on the rising edge of BCLK.

When the BUSY* signal is driven inactive high by the current bus master, the current bus master must also tri-state the following signal drivers: BUSY*, TS*, ADDR, RW*, and BE3:0. When driving the BUSY* signal inactive high, the current bus master must push the BUSY* signal to a valid 1 before tri-stating the BUSY* signal to ensure a reasonable rise-time occurs on the BUSY* signal.

TS*

The TS* signal must be driven active low by the external bus master to indicate the start of a memory signal. The TS* signal must be driven active low for one BCLK cycle only. The TS* signal can be driven active low during the same BCLK signal in which the external bus master drives BUSY* low to assume bus ownership.

A27-A0

The entire address bus must be driven active during the BCLK cycle while the TS* signal is active low. The address bus decodes the desired memory peripheral. For the NET+ARM chip memory controller to decode the proper address, the external bus master must provide the base address for the peripheral.

The NET+ARM chip memory controller requires 32-bits for address decode, however, the external bus master can only provide 28-bits of address. Internally, the NET+ARM chip interface hardware copies the values from A27:A24 to create the internal version of A31:A28 respectively. The memory controller can be configured to mask the A31:A28 bits when an external bus master desires to access one of the peripherals controlled by the memory controller.

The CS0OE* and CS0WE* feature can still be used with an external bus master, however, if these features are used, then address bits A27 and A26 must be masked in all the chip select option registers within the memory controller. When the A25 signal is configured (via the MMCR) to provide the BLAST* indicator, then A24 creates the internal version of A29 and A25 when an external bus master owns the bus.

When the external bus master is addressing a non-DRAM memory peripheral, the address bus must remain stable throughout the entire memory cycle. When the external bus master is addressing a DRAM memory peripheral, the upper address bits A27:A14 must remain stable, however, the lower address bits A13:A0 must change based upon the state of the PORTC3 signal. The PORTC3 signal is provided to the external bus master to indicate when the RAS/CAS multiplexed address bits must change.

A25

The A25 signal is used by the external bus master to indicate which memory cycle is the last memory cycle of a burst cycle. The external bus master identifies the last cycle of a burst by driving the A25 signal low during the last transfer of a burst sequence. The NET+ARM chip memory controller needs this indicator to know when the burst sequence should be terminated. To support bursting from the external bus master, the A25* configuration bit in the Memory Module Configuration Register (MMCR) must be set to a 1.

RW*

The RW* signal is driven by the external bus master to indicate a Read Cycle (RW* set to 1) or a Write Cycle (RW* set to 0).

TA*

The TA* signal is driven active low by the NET+ARM chip to indicate the end of the current memory cycle.

TEA*

The TEA* signal is driven active low by the NET+ARM chip to indicate the end of the current memory burst cycle. The TEA* signal is asserted in conjunction with the TA* signal. The TEA* signal is driven when either the external bus master indicates a burst completion (A25 signal driven low) or the memory controller peripheral indicates a burst completion.

BE3:0

The BE signals are driven active low by the external bus master to indicate which byte lanes are valid for any given memory cycle.

PORTC3

The external bus master can use the PORTC3 signal to determine when the A13:0 address signals must be changed for DRAM address multiplexing. While the PORTC3 signal is low, the external bus master must drive the RAS multiplexed address signals onto A13:A0. When the PORTC3 signal is driven high, the external bus master must drive the CAS multiplexed address signals onto A13:A0.

For the PORTC3 signal to be driven from the NET+ARM chip for DRAM address multiplexing, the AMUX2 bit in the Memory Module Configuration Register (MMCR) must be set to 1.

9.9.2 Bus Arbitration

An external bus master requests ownership of the NET+ARM chip system memory bus by asserting the BR* signal active low. The BR* signal must be synchronized to BCLK. The NET+ARM chip samples the BR* input using the rising edge of BCLK.

After asserting BR* active low, the external bus master must wait for the rising edge of BCLK in which the BG* signal is active low and the BUSY* signal is inactive high. After this condition is found, the external bus master can drive BUSY* low to assume ownership of the NET+ARM chip system bus. The new external bus master must provide a minimum latency of 1 BCLK cycle between when BUSY* is driven high (by the previous bus master) and driven low (by the new bus master).

The minimum one BCLK cycle where BUSY* is inactive high is referred to as the turn-around cycle. During this time, the previous bus master must tri-state the following signals: BUSY*, TS*, ADDR, DATA, RW*, and BE3:0.

The new bus master can delay driving BUSY* low for as many BCLK clocks as desired after the previous master drives BUSY* high. While waiting to drive BUSY* active low, the new bus master must continue to drive BR* low to ensure the NET+ARM chip does not reassume ownership of the system bus.

The external bus master can drive the BR* input inactive high at the same time in which the BUSY* signal is driven active low. The external bus master can start driving the TS*, ADDR, DATA, RW*, and BE3:0 signals at the same time in which BUSY* is driven active low. The new bus master is not required to begin a bus cycle at the same time in which BUSY* is driven active low. The bus master can insert additional BCLK cycles between driving BUSY* active low and starting a memory cycle by driving TS* low.

The external bus master owns the NET+ARM chip system bus until it relinquishes the bus by driving the BUSY* signal inactive high. The external bus master is responsible for maintaining fairness. When the external bus master does relinquish the bus, it must

tri-state the following signals: BUSY*, TS*, ADDR, DATA, RW*, and BE3:0. Before tri-stating the BUSY* signal, the external bus master must drive BUSY* to a solid high first. This is required to ensure a good rise-time on the BUSY* signal.

9.9.3 Memory Cycle Start

After driving the BUSY* signal active low to assume ownership of the NET+ARM chip system memory bus, the external bus master must drive the TS* signal active low for one BCLK cycle to initiate a memory cycle. The external bus master may begin bus cycles any number of BCLK cycles after BUSY* is asserted low.

While the TS* signal is active low, the address, RW*, and BE signals must also be valid. All address bits must be driven to allow the internal MEM module peripheral decoders to decode the proper memory address. Only those address bits, which are masked for decoding (using the chip select option registers), do not need to be driven. The NET+ARM chip samples the address bus (for decoding purposes) on the rising edge of BCLK while TS* is active low.

9.9.4 DRAM Address Multiplexing

The address bus must remain stable throughout the entire memory cycle for all non-DRAM peripheral devices. For DRAM peripheral devices, the A13-A0 address signals can change for DRAM address multiplexing.

When interfacing with DRAM, the DRAM devices require multiplexing on the lower address signals, A13-A0. Normally, the NET+ARM chip can perform this function using its internal address multiplexer, however, when interfacing with an external bus master, the internal NET+ARM chip address multiplexer cannot be used when an external bus master owns the bus. Instead, DRAM address multiplexing must be performed using external multiplexing chips or by the external bus master itself (when the external bus master owns the bus).

To use external address multiplexing chips for both the NET+ARM chip bus masters and the external bus masters, then set the AMUX bit in the MMCR. When the AMUX bit is set, the PORTC3 output controls the select input for the external address multiplexer chips. While PORTC3 is low, the RAS address must be driven. While PORTC3 is high, the CAS address must be driven.

An alternative is to use the internal DRAM address multiplexer when the NET+ARM chip owns the system memory bus and have the external bus master perform its own multiplexing when it owns the system bus. With this scheme, the AMUX2 bit in the MMCR must be set. When the AMUX bit is set, the PORTC3 output controls the select input for the external address multiplexer chips. While PORTC3 is low, the

RAS address must be driven. While PORTC3 is high, the CAS address must be driven.

Additional multiplexing considerations are required to support synchronous DRAM. The NET+ARM chip requires the following when using SDRAM with an external bus master.

1. The external bus master must drive the CAS address values on A13:A0 while the PORTC3 signal is active high.
2. The external bus master must drive the RAS address values on A13:A0 while the PORTC3 signal is active low and the SDRAM controller is issuing the ACTIVE command. The ACTIVE command is encoded using the CAS3:1 signals. The CAS3:1 signals are in the “011” binary state during the ACTIVE command.
3. At all others times when PORTC3 is inactive and the SDRAM controller is not issuing the ACTIVE command, the external bus master must drive the non-multiplexed address signals on A13:A8. The NET+ARM chip samples the A23:A8 address bus 1 BCLK cycle before the PORTC3 signal is activated to sample the current SDRAM page address.

9.9.5 Burst Operation

An external bus master can perform burst operations to peripherals controlled by the NET+ARM chip memory controller. A burst cycle requires a single TS* to start the burst operation. Individual cycles within the burst cycle complete when the TA* signal is driven active low by the NET+ARM chip memory controller. The NET+ARM chip memory controller signals the end of the burst cycle by driving the TEA* signal active low while TA* is low during the last operation of the burst cycle.

The external bus master must also signal the boundaries of the burst. The external bus master might terminate the burst before the memory controller does. The external bus master uses the A25 signal to indicate the boundaries of the burst operation. The A25 signal must be driven high to indicate a burst continuing condition and driven low to indicate a burst completion.

The external bus master must advance the lower address bits when transitioning between individual cycles within the burst cycle. The external bus master must provide the necessary address timing required by the peripheral it is interfacing with. Note that when interfacing with SDRAM, the external bus master need not advance the lower address bits since the SDRAM uses an internal index pointer.

9.9.6 Completion

The external bus master owns the NET+ARM chip system bus until it relinquishes the bus by driving the BUSY* signal inactive high. The external bus master is responsible for maintaining fairness. When the external bus master does relinquish the bus, it must tri-state the following signals: BUSY*, TS*, ADDR, RW*, and BE3:0. Before tri-stating the BUSY* signal, the external bus master must drive BUSY* to a solid high first. This is required to ensure a good rise-time on the BUSY* signal.

9.9.7 Throttling

The external bus master needs to be concerned about bus utilization and fairness. The external bus master should not maintain ownership of the NET+ARM chip system memory bus for long extended periods of time. Doing so can cause bandwidth problems for the internal DMA controller. In general, an external bus master should maintain ownership of the system bus for something on the order of 4-8 memory cycles only.

Chapter 10
Memory Controller Module

The memory module provides a glueless interface to external memory devices such as Flash, DRAM, EEPROM, and so on. The memory controller contains an integrated DRAM controller. The memory controller supports 5 unique chip select configurations. Each chip select can be configured to interface with an asynchronous device (such as static RAM or flash) or a DRAM device.

The MEM module monitors the BBus interface for access to the BUS module. Those accesses are destined for external resources. If the desired address corresponds to an address base register within the MEM module, then the MEM module provides the memory access signals and responds to the BBus with the necessary completion signal.

The MEM module can be configured to interface with FD, EDO, or synchronous DRAM, however, the NET+ARM chip cannot interface with a mixture of synchronous DRAM and FP or EDO DRAM. All chip selects configured for DRAM must be configured with the same style of DRAM.

10.1 Module Configuration

The memory module has a block of configuration space that is mapped into the MEM module configuration space as defined in Table 2-1, BBus Address Decoding.

Address		Register
FFC0 0000	MMCR	Memory Module Configuration Register
FFC0 0010	BAR0	Chip Select 0 Base Address Register
FFC0 0014	OR0	Chip Select 0 Option Register
FFC0 0020	BAR1	Chip Select 1 Base Address Register
FFC0 0024	OR1	Chip Select 1 Option Register

Table 10-1: Memory Controller Configuration

Address		Register
FFC0 0030	BAR2	Chip Select 2 Base Address Register
FFC0 0034	OR2	Chip Select 2 Option Register
FFC0 0040	BAR3	Chip Select 3 Base Address Register
FFC0 0044	OR3	Chip Select 3 Option Register
FFC0 0050	BAR4	Chip Select 4 Base Address Register
FFC0 0054	OR4	Chip Select 4 Option Register

Table 10-1: Memory Controller Configuration (Continued)

10.1.1 MEM Module Hardware Initialization

Many internal NET+ARM chip configuration features are application specific and need to be configured at power-up before the CPU boots.

The System Bus Address bits are used for this purpose during a powerup reset. The NET+ARM chip provides internal pullup resistors on all Address signals. External pulldown resistors can be employed to configure various internal register bits to a zero state.

The following identifies which ADDR bits control what functions in the MEM Module.

ADDR[24:23] : CS0 Bootstrap Setting

“00” - Disable

“01” - 32-bit SRAM port; 15 wait-states

“10” - 32-bit FP DRAM port; 15 wait-states

“11” - 16-bit SRAM port; 15 wait-states

Refer to section *11.4 NET+ARM Chip Bootstrap Initialization*.

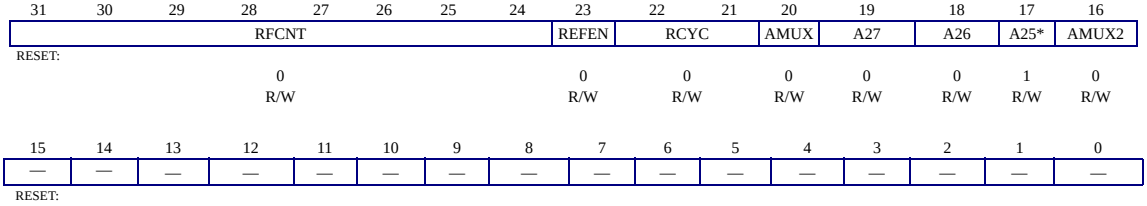
10.1.2 Memory Module Configuration Register

The MMCR is a 32-bit register that defines basic configurations that apply to allow chip selects.

The A27 and A26 bits in the MMCR are automatically set to 0 on bootstrap when the CS0 bootstrap configuration is defined for 16-bit SRAM port. The A27 and A26 bits are automatically set to 1 for all other CS0 bootstrap configurations.

The software reset command issued via the GEN module software service register has no effect on any MEM module configuration registers.

Address = FFC0 0000



The CS0WE* and CS0OE* signals are internally generated by gating CS0* with WE* and OE* respectively. CS0WE* is driven active low when both CS0* and WE* are active low. Likewise, CS0OE* is driven active low when both CS0* and OE* are active low. The A27 and A26 settings in the MMCR are used to multiplex the two highest order address bits with the CS0WE* and CS0OE* signals.

RFCNT

Refresh period = [(RFCNT + 1) * 4] / F_{XTAL}

The REFCNT field defines the refresh period for the memory controller when Fast-Page, EDO, or Synchronous DRAMs are being used. All DRAM devices require a periodic refresh cycle. Typically, the refresh cycle is 15 us. The NET+ARM chip allows the refresh period to be programmable. The refresh period is a function of the RFCNT field and the value for F_{XTAL}. Refer to section *11.3.2 XTAL Clock Generation* for more information on the F_{XTAL} value.

The DRAM refresh controllers always generates CAS before RAS refresh cycles.

REFEN

The REFEN bit must be set to enable DRAM refresh. DRAM refresh must be enabled whenever DRAMs are being used.

RCYC

“00” - Refresh cycle is 8 BCLK clocks long

“01” - Refresh cycle is 6 BCLK clocks long

“10” - Refresh cycle is 5 BCLK clocks long

“11” - Refresh cycle is 4 BLCK clocks long

The RCYC field controls the length of the refresh cycle. The NET+ARM chip provides for flexibility in the DRAM refresh cycle to cover a wide range of industry available DRAM components. Refer to section *13.3.2 SDRAM Cycles* concerning more information regarding the refresh waveforms for the various setting of RCYC. When using Synchronous DRAM (SDRAM) devices, an RCYC setting of “00” is recommended.

AMUX

The AMUX bit controls whether or not the NET+ARM chip uses it’s internal DRAM address multiplexer.

When AMUX is set to 0, the NET+ARM chip uses the internal DRAM address multiplexer for all DRAM access cycles. The DRAM RAS/CAS multiplexed address is routed through the A13:A0 pins as described in section *10.4 NET+ARM Internal DRAM Address Multiplexing*.

When AMUX is set to 1, the NET+ARM chip does not use the internal DRAM address multiplexer for DRAM cycles. Instead an external DRAM address multiplexer is employed. The RAS/CAS address select signal is routed out the PORTC3 signal when the AMUX bit is set to 1. The external DRAM RAS/CAS address multiplexer function can use the PORTC3 signals to determine when to switch the address multiplexer. When the PORTC3 signal is active high, the external DRAM RAS/CAS address multiplexer function must drive the CAS address to the DRAM devices. An external DRAM address multiplexer is required to be used when an external bus master must access the DRAM devices using the NET+ARM chip DRAM controller and that external bus master cannot perform DRAM address multiplexing itself.

A27

The A27 bit determines how the A27 bit is used by the NET+ARM chip.

When A27 is set to 0, the CS0OE* signal is driven out the A27 pin. The CS0OE* signal is generated by internally performing a logical AND of the CS0* and OE* signals. The CS0OE* signal goes active low when both CS0* and OE* are active low. The CS0OE* function is useful for maximizing the read access timing for external memory peripherals attached to CS0. When using CS0OE*, the CS0 peripheral’s chip-select input can be attached to GND causing the read-access time for that peripheral to be referenced from Address instead of Chip-Select. The NET+ARM chip provides the address signals during the earliest part of each memory cycle. The CS0OE* signal is then connected to the OE* input for the CS0 peripheral enabling the data bus during read cycles.

During reset, the A27 bit defaults to the value defined by A24 and A23 (refer to section *10.1.1 MEM Module Hardware Initialization*). Only the “11” setting on A24:23 defaults the A27 output for the CS0OE* configuration.

A24:A23	A27 Default
00	1
01	1
10	1
11	0

A26

The A26 bit determines how the A26 bit is used by the NET+ARM chip.

When A26 is set to 0, the CS0WE* signal is driven out the A26 pin. The CS0WE* signal is generated by internally performing a logical AND of the CS0* and WE* signals. The CS0WE* signal goes active low when both CS0* and WE* are active low.

During reset, the A26 bit defaults to the value defined by A24 and A23 (refer to section *10.1.1 MEM Module Hardware Initialization*). Only the “11” setting on A24:23 defaults the A26 output for the CS0WE* configuration.

A24:A23	A26 Default
00	1
01	1
10	1
11	0

A25*

The A25 bit determines how the A25 bit is used by the NET+ARM chip.

When the A25 bit is low, the A25 pin is used by the NET+ARM chip for address bit 25. The A25 control bit always defaults to 1 on reset.

When the A25 bit is set high, the current system bus master uses the A25 pin to signal the end of a burst cycle. The system bus master drives the A25 signal high to indicate it wishes to continue a burst cycle. The system bus master drives the A25 signal low to indicate it wishes to end the current burst cycle.

Refer to section *9.9 External Bus Master Support* for a detailed description on the operation of the A25/BLAST* signal.

AMUX2

The AMUX2 bit can be used to drive the DRAM RAS/CAS address multiplexing control signal out the PORTC3 pin regardless of the setting of the AMUX bit.

When AMUX2 is set to 1, the DRAM RAS/CAS address multiplexing control signal is driven out the PORTC3 pin. This is useful in applications using an external bus master when both the NET+ARM chip and external bus master can both use an internal DRAM address multiplexer. When the NET+ARM chip is the current bus master, the NET+ARM chip uses its internal DRAM address multiplexer for all DRAM cycles. When the NET+ARM chip is not the current bus master, the NET+ARM chip tri-states its address bus. The memory controller drives the DRAM RAS/CAS address multiplexing control signal through the PORTC3 pin for the external bus master to use for its own DRAM RAS/CAS address multiplexing control.

When the external bus master is capable of performing DRAM RAS/CAS address multiplexing on its own, the AMUX2 configuration can save the cost of external RAS/CAS address multiplexers.

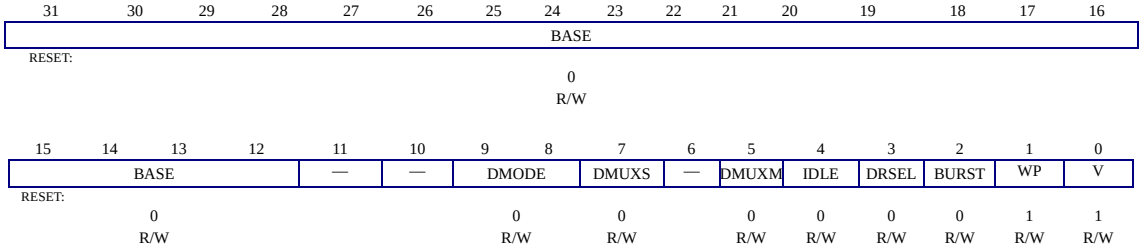
10.1.3 Chip Select Base Address Register

The BAR is a 32-bit read/write register that defines the base starting address for the chip select. Each chip select can be configured in size from 4K to 4G bytes. The base address must be positioned on an address boundary consistent with its programmed size. All bits are reset to 0 on system reset.

Note: The “WP” and “V” bits are set to 1 on hardware reset for Chip Select 0 only.

Note: All DRAM banks must be configured with the same DMODE setting.

Address = FFC0 0010 / 20 / 30 / 40 / 50



It is very important that the “V” bit in the chip select base address register not be set to 1 until the fields in the chip select option register have been configured. Failure to configure the values in the chip select option register before setting the “V” bit results in unpredictable behavior for the memory controller.

BASE

The BASE field determines the physical base address of the memory peripheral chip select. This 20-bit field represents the 20 most significant bits of the physical address. To determine the base physical address of a memory peripheral decode space, simply add three zeros to the end of this field. For example, if the BASE field is set to a value of 0x00200, then the base physical address of the peripheral chip select would be 0x**00200**000.

The BASE field must be consistent with the size of the chip select as defined by the MASK field. The base address of a chip select must occur on a proper boundary in relation to the peripheral memory size. For example, if the size of the memory device is 4M-bytes, then the BASE field must be configured such that the base address is located on a 4M-byte address boundary.

DMODE

“00” - Fast Page DRAM

“01” - EDO DRAM

“10” - Synchronous DRAM

“11” - Reserved

The DMODE field controls the DRAM type when the memory device is configured to operate in dynamic RAM mode. The DMODE field is only used when the DRSEL bit is set to 1 indicating DRAM mode. All DRAM memory peripherals must be configured to be the same type of DRAM.

DMUXS

0 – Internal DRAM Multiplexer

1 – External DRAM Multiplexer

The DMUXS field controls whether or not the NET+ARM chip internal address multiplexer is used for this DRAM memory peripheral. A setting of 1 indicates that an external DRAM multiplexer is to be used for this device. When set to 1, the PORTC3 pin signals to the external address multiplexer when to switch the address bits. When the PORTC3 signal is active high, the external DRAM RAS/CAS address multiplexer function must drive the CAS address to the DRAM devices. An external DRAM address multiplexer is required when an external bus master must access the DRAM

devices using the NET+ARM chip DRAM controller and that external bus master cannot perform DRAM address multiplexing itself.

Note that when either the AMUX or AMUX2 bits in the memory module configuration register are set, the AMUX or AMUX2 bits serve as a global control and all DRAM peripheral devices use the external address multiplexer. If either AMUX or AMUX2 are set, the DMUXS bit is ignored.

DMUXM

0 – 10 CAS

1 – 8 CAS

The DMUXM field controls which DRAM address-multiplexing style should be used for this DRAM memory peripheral. Refer to section *10.4 NET+ARM Internal DRAM Address Multiplexing* for information on the two different address-multiplexing modes.

IDLE

The IDLE bit can be used to insert an extra BCLK cycle at the end of the memory cycle for this peripheral. The IDLE bit can be used to insert one additional clock cycle for recovery purposes.

DRSEL

The DRSEL bit configures the memory peripheral to operate in DRAM mode. The DMODE, DMUXS, and DMUXM bits are only used while the DRSEL bit is set to 1. The WSYNC and RSYNC bits are ignored in DRAM mode. WSYNC and RSYNC are only used when DRSEL is configured as 0.

BURST

The BURST bit controls whether or not the memory peripheral device supports bursting. When BURST is set to 0, burst cycles are not allowed. All memory cycles are single cycles and the WAIT field controls the number of wait-states.

When the BURST bit is set to 1, burst cycles are allowed provided the current bus master desires to execute a burst cycle. During a burst cycle, the WAIT field controls the number of wait-states for the first memory access of the burst and the BCYC field controls the number of wait-states for all subsequent memory access for the burst cycle. The BSIZE field controls the maximum number of memory cycles that the peripheral can support. The current Bus Master can choose to burst a smaller number of memory cycles, however, the peripheral terminates the burst when the maximum number of memory cycles defined by the BSIZE field is reached.

WP

The WP bit can be used to prevent any bus master from writing to the memory device. When the WP bit is set to 1, all memory write-cycles are terminated immediately with a data abort indicator. The WP bit can be used to protect non-volatile memory devices such as Flash and EEPROM.

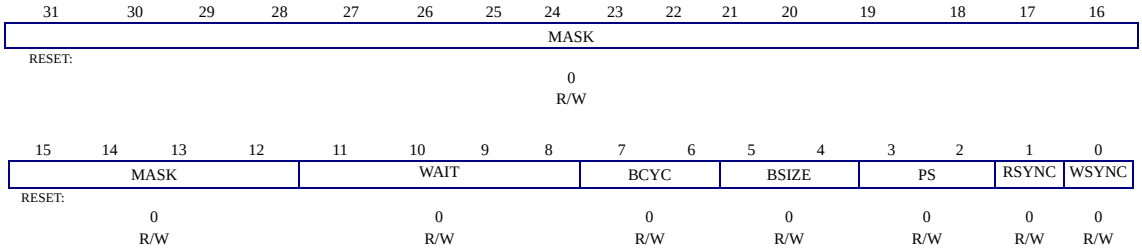
V

The V bit enables the chip select. When the V bit is set to 1, the memory controller uses the various fields in the chip select base and option registers to control the behavior of the peripheral memory cycles. It is very important that the “V” bit not be set to 1 until all other fields in both the base and option registers are configured. Failure to do so results in unpredictable behavior of the memory controller.

10.1.4 Chip Select Option Register

The OR is a 32-bit read/write register that defines the physical size of the chip select and other miscellaneous features. Each chip select can be configured in size from 4K to 4G bytes. All bits are reset to 0 on system reset

Address = FFC0 0010 / 20 / 30 / 40 / 50



It is very important that the “V” bit in the chip select base address register not be set to 1 until the fields in the chip select option register have been configured. Failure to configure the values in the chip select option register before setting the “V” bit results in unpredictable behavior for the memory controller.

MASK

The MASK field controls the size of the memory peripheral decode space. The MASK field can also be used to “alias” the peripheral device in different areas of the memory map.

The memory controller performs a logical and of the MASK and BASE fields to determine if the address space is assigned to the peripheral device. The MASK field identifies those address bits from A31 through A12, which are used in the address

decoding function. All ones (1) in the MASK field indicates the associated address bit is to be used in the decoding process. All zeros (0) in the MASK field indicates the associated address bit is to be ignored in the address decoding process.

To determine if a logical address is associated with a peripheral chip select, the following boolean equations can be applied.

Chip Select Decode is TRUE when:

(({MASK,000} & 32-bit logical address) = = ({BASE,000} & {MASK,000}))

{MASK,000} refers to the MASK field concatenated with three nibbles of 0.

{BASE,000} refers to the BASE field concatenated with three nibbles of 0.

The **&** symbol refers to the logical AND function. The = = symbol refers to the "equal to" operator.

The following table can help you to determine the physical size of the peripheral.

MASK	SIZE
0xFFFFF	4K
0xFFFFE	8K
0xFFFFC	16K
0xFFFF8	32K
0xFFFF0	64K
0xFFFE0	128K
0xFFFC0	256K
0xFFF80	512K
0xFFF00	1M
0xFFE00	2M
0xFFC00	4M
0xFF800	6M
0xFF000	16M
0xFE000	32M
0xFC000	64M
0xF8000	128M
0xF0000	256M

Since the NET+ARM chip only supports 28 address bits, addressing above 256M bytes of information per chip select is impossible.

The MASK field can also be used to alias a memory location in different areas. For example, a 16M-byte device can be addressed in 4 different 64M bytes address locations by using a MASK value of 0xF3000. In this example, the peripheral would be

addressable at address location BASE + 0x00000000, BASE + 0x04000000, BASE + 0x08000000, and BASE + 0x0C000000.

The BASE field must be consistent with the size of the chip select as defined by the MASK field. The base address of a chip select must occur on a proper boundary in relation to the peripheral memory size. For example, if the size of the memory device is 4M-bytes, then the BASE field must be configured such that the base address is located on a 4M-byte address boundary.

WAIT

The WAIT field controls the number of wait-states for all single memory cycle transactions and the first memory cycle of a burst transaction. All NET+ARM chip memory cycles are a minimum of 2 BCLK cycles. This field identifies the additional number of BCLK cycles for all single memory cycle transactions and the first memory cycle of a burst transaction.

BCYC

"00" - Burst cycles are 1 BCLK clock in length (x,1,1,1)

"01" - Burst cycles are 2 BCLK clocks in length (x,2,2,2)

"10" - Burst cycles are 3 BCLK clocks in length (x,3,3,3)

"11" - Burst cycles are 4 BCLK clocks in length (x,4,4,4)

The BCYC field controls the number of clock cycles for the secondary portion of a burst cycle. The WAIT field controls the first memory cycle of a burst cycle, the BCYC field controls the subsequent cycles.

BSIZE

"00" - 2 System Bus cycles in burst access

"01" - 4 System Bus cycles in burst access

"10" - 8 System Bus cycles in burst access

"11" - 16 System Bus cycles in burst access

The BSIZE field controls the maximum number of memory cycles that can occur in a burst cycle. This field determines only the maximum number of allowable cycles, the current bus master can choose to not burst the entire amount. If the current bus master continues to burst, the peripheral terminates the burst when the number of memory cycles reaches the maximum allowed by the BSIZE field.

PS

"00" - 32-bit Port Size

"01" - 16-bit Port Size

"10" - 8-bit Port Size

"11" - 32-bit Port using external Data Acknowledge

The PS field controls the size of the memory peripheral device. Each device can be configured to operate as either an 8-bit device, 16-bit device, or 32-bit device.

Note that for all configurations except 1, the NET+ARM chip operates the device using a fixed number of wait-states and drives the TA* signal active low to indicate the end of the memory cycle.

Only the "11" setting allows the NET+ARM chip to wait for an external peripheral to complete its cycle by signaling the end of the memory cycle by driving the TA* input active low. As such, the NET+ARM chip can only support 32-bit peripherals can terminate their own memory cycles using the TA* input.

RSYNC

0 - SRAM chip-select to operate in Asynchronous mode

1 - SRAM chip-select to operate in Synchronous mode

The RSYNC bit controls the access timing of the OE* and CS* signals for non-DRAM memory peripherals. The RSYNC bit is only used when the DRSEL bit is set to 0.

When RSYNC is set to 1, the memory peripheral operates in a mode where the OE* signal is active low inside the low time of CS*. This is typically used to interface with peripheral devices that use "Intel" timing.

When RSYNC is set to 0, the memory peripheral operates in a mode where the OE* signal is active low outside the low time of CS*. This is typically used to interface with peripheral devices that use "Motorola" timing.

WSYNC

0 - SRAM chip-select to operate in Asynchronous mode

1 - SRAM chip-select to operate in Synchronous mode

The WSYNC bit controls the access timing of the WE* and CS* signals for non-DRAM memory peripherals. The WSYNC bit is only used when the DRSEL bit is set to 0.

When WSYNC is set to 1, the memory peripheral operates in a mode where the WE* signal is active low inside the low time of CS*. This is typically used to interface with peripheral devices that use "Intel" timing.

When WSYNC is set to 0, the memory peripheral operates in a mode where the WE* signal is active low outside the low time of CS*. This is typically used to interface with peripheral devices that use "Motorola" timing.

10.2 Pin Configuration

The NET+ARM chip uses various pins in support of SRAM and the various DRAM devices. The MEM module can control the following signals: ADDR[13:0], CSx*, CASx*, WE*, and OE*.

Mode	A13:0	CSx*	CAS3*	CAS2*	CAS1*	CAS0*	OE*	WE*
SRAM	-	CS*	-	-	-	-	OE*	WE*
DRAM-FP	Address	RAS*	CAS3*	CAS2*	CAS1*	CAS0*	OE*	WE*
DRAM-EDO	Address	RAS*	CAS3*	CAS2*	CAS1*	CAS0*	OE*	WE*
DRAM-SYNC	Address	CS*	RAS*	CAS*	WE*	A10/ap	-	-

Table 10-2: MEM Module Pin Configuration

When a chip select is configured for SRAM, the MEM module controls the CS*, OE*, and WE* signals. The current bus master provides the entire address.

When a chip select is configured for FP or EDO DRAM, the MEM module controls the address bus when the internal DRAM multiplexer is selected. The CS* signals provide the RAS* function, the CAS* signals provide the CAS* function, and the OE* and WE* signals provide the output and write enables respectively.

When a chip select is configured for synchronous DRAM, the MEM module controls the address bus when the internal DRAM multiplexer is selected. The CS* signals provide the CS* function. The CAS3* signal provides the RAS* function. The CAS2* signals provide the CAS* function. The CAS1* signal provides the WE* function. The CAS0* signals provide the A10/ap multiplexed signal. The A10/ap multiplexes between the A10 pin for the DRAM and the Auto Pre-charge indicator. The CAS0* signal must always be connected to the Synchronous DRAM A10 pin.

Synchronous DRAMs also require a function referred to as DQM. The DQM function is provided by the NET+ARM chip BE*[3:0] signals.

10.3 Static Memory Controller

Each chip select can be configured to operate using a static memory interface. The memory controller must support the following features:

- Synchronous Mode: Transactions use rising edge of BCLK
- Asynchronous Mode: Force OE* and WE* pulses to be inside active low portion of CS*
- Burst Cycle Support
- Programmable Wait States
- Programmable Base Address and Chip Select Size

10.3.1 Single Cycle Read/Write

Figure 10-1 provides a diagram showing synchronous SRAM cycles. These cycles are designed primarily for peripherals that can use BCLK or CS* as the data transfer signal. All outputs change state relative to the rising edge of BCLK except for OE* and WE* that transition on the falling edge of BCLK. The OE* and WE* signals change state on the falling edge before CS* is asserted. OE* and WE* remain active until the falling edge after CS* is deasserted.

During synchronous read and write cycles, the rising edge of BCLK where TA* is low defines the end of the memory cycle, also referred to as the T2 state. During synchronous read cycles, read data is sampled on the rising edge of BCLK where TA* is low.

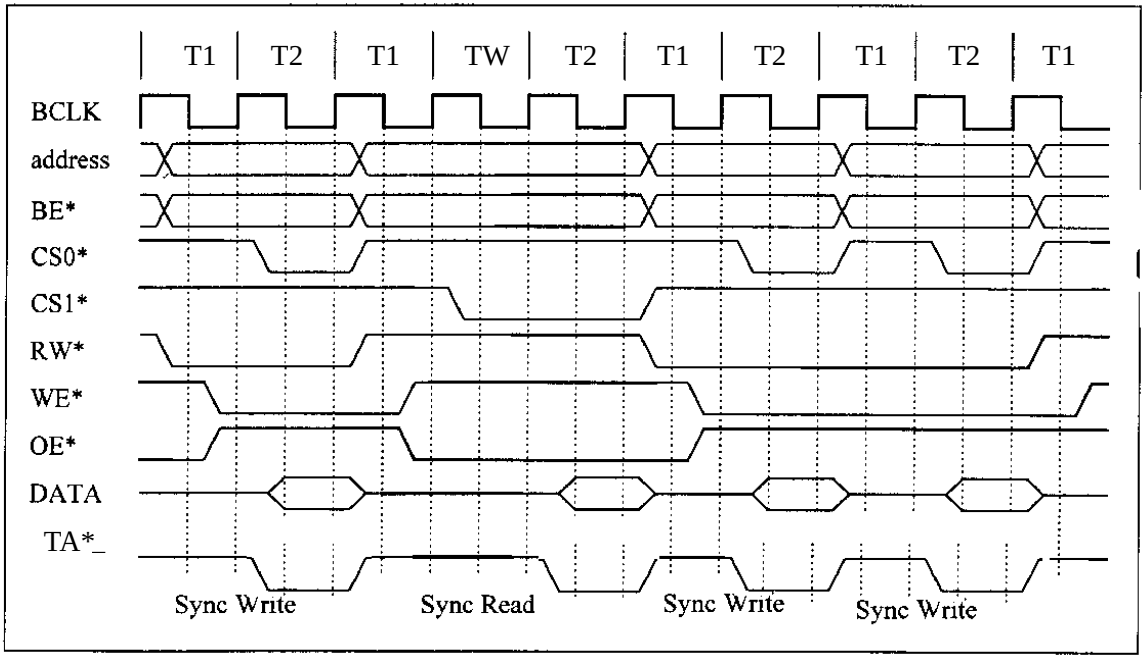


Figure 10-1: SRAM Synchronous Bus Cycles

Figure 10-2 provides a diagram showing asynchronous SRAM cycles. These cycles guarantee that the WE* and OE* pulses are inside the active low pulse of CS*. Asynchronous SRAM cycles operate a minimum of 1 wait-state at all times.

During an asynchronous read/write cycle, the BE*, OE* and WE* signals transition based upon the falling edge of BCLK. The BE*, OE* or WE* signal transitions low on the first falling edge after CS* is asserted. The BE*, OE* or WE* signal transitions high on the first falling edge after TA* is recognized (TA* is still sampled using the rising edge of BCLK.

During asynchronous read and write cycles the rising edge of BCLK where TA* is low defines the last TW cycle that is one full BCLK before T2. During asynchronous read cycles, read data is sampled on the rising edge of BCLK where TA* is low, which is also one full BCLK before T2.

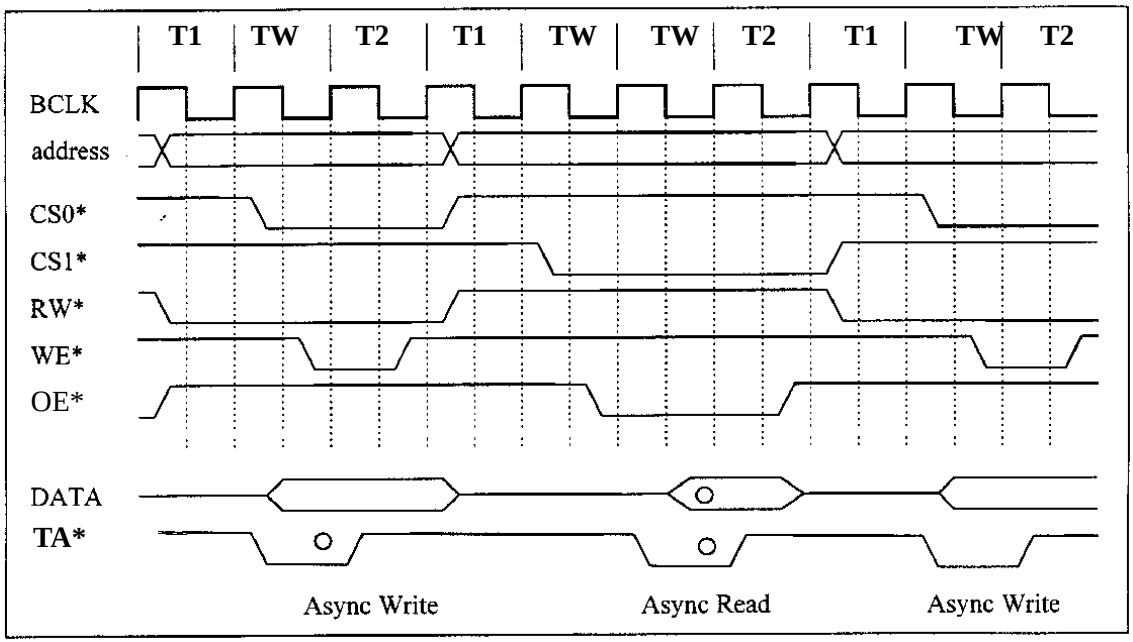


Figure 10-2: SRAM Asynchronous Bus Cycles

10.3.2 Burst Cycles

The SRAM controller supports both read and write burst cycles.

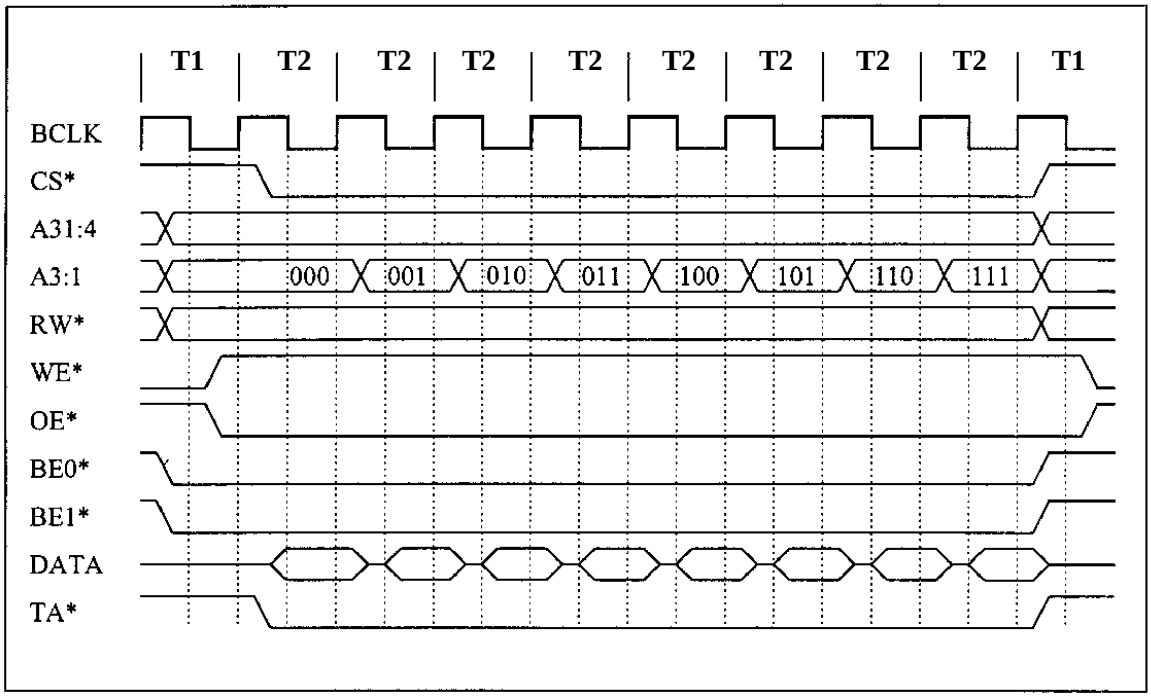


Figure 10-3: SRAM Synchronous Burst Read Cycle

10.4 NET+ARM Internal DRAM Address Multiplexing

The NET+ARM chip can be configured to use an internal DRAM address multiplexer or an external address multiplexer. The internal/external multiplexer selection is provided by a combination of the AMUX and AMUX2 bits in the MMCR and the DMUXS bit in the chip select base address register.

When configured to use the internal address multiplexer, the DRAM address signals are provided on system bus address pins A13 through A0. A 32-bit DRAM peripheral connects to A13 through A2; a 16-bit DRAM peripheral connects to A13 through A1, while an 8-bit DRAM peripheral connects to A13 through A0. When a particular DRAM has less than 14 address bits, the lower order NET+ARM chip address bits are to be used and the upper NET+ARM chip address bits are to be left disconnected from the DRAM. The SDRAM bank select pins must always be connected to the upper

order NET+ARM chip address pins. Never connect a bank select pin to one of the multiplexed address pins (A13:A0).

The NET+ARM chip supports two modes of internal address multiplexing; MODE 0 and MODE 1. The internal address-multiplexing mode is configured via the DMUXM bit in the chip select base address register. Each chip select can be configured for a different address-multiplexing mode. SDRAM requires the use of MODE 1 multiplexing.

Figure 10-5 describes how the NET+ARM chip multiplexes the logical address signals through the physical address signals during the RAS and CAS timeframes.

The top row identifies the physical address connection on the NET+ARM chip devices. The “DRAM” row identifies the physical address connection used on the DRAM device. The “RAS” row identifies the “logical” address driven by the NET+ARM chip during the RAS portion of the RAS/CAS addressing multiplexing sequence. The “CAS” row identifies the “logical” address driven by the NET+ARM chip during the CAS portion of the RAS/CAS address multiplexing sequence.

Carefully note that during the CAS portion of mux mode 1, the A13, A12, and A11 signals are always driven to a 0. The A13, A12, A11 signals must never be connected to the SDRAM bank select pins. The SDRAM bank select pins must remain stable throughout the entire memory cycle. If the bank select pins are attached to any of the multiplexed address pins, then the SDRAM part fails because the bank select pins are changing state in the middle of the memory cycle.

Note: When using synchronous DRAM parts, A10 of the SDRAM device must always connect to CAS0* of the NET+ARM chip. Refer to section *10.7 Synchronous DRAM*.

MODE 0 Multiplexing – Useful for DRAMs with Balanced RAS/CAS address lines

NET+ARM Chip Multiplexed Address Outputs														
NET+ARM Chip PIN	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
DRAM			A9			A8	A7	A6	A5	A4	A3	A2	A1	A0
RAS			18			17	16	15	14	13	12	11	10	9
CAS			19			8	7	6	5	4	3	2	1	0
			8-BIT			8- BIT DRAM PERIPHERAL (20 Addr Bits; 10 RAS and 10CAS)								
NET+ARM Chip PIN	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
DRAM PIN		A9			A8	A7	A6	A5	A4	A3	A2	A1	A0	
RAS		19			18	17	16	15	14	13	12	11	10	
CAS		20			9	8	7	6	5	4	3	2	1	
		16-BT			16-BIT DRAM PERIPHERAL (20 Addr Bits; 10 RAS and 10 CAS)									
NET+ARM Chip PIN	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
DRAM PIN	A9			A8	A7	A6	A5	A4	A3	A2	A1	A0		
RAS	20			19	18	17	16	15	14	13	12	11		
CAS	21			10	9	8	7	6	5	4	3	2		
	32-BT			32-BIT DRAM PERIPHERAL (20 Addr Bits; 10 RAS and 10 CAS)										

Figure 10-4: Internal DRAM Multiplexing - Mode 0

MODE 1 Multiplexing – Useful for DRAMs with Unbalanced RAS/CAS address lines

	NET+ARM Chip Direct		NET+ARM Chip Multiplexed Address Outputs													
NET+ARM Chip PIN	A23	A22	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
DRAM			A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
RAS			21	20	19	18	17	16	15	14	13	12	11	10	9	8
CAS			0	0	0	10	20	19	7	6	5	4	3	2	1	0
			8- BIT DRAM PERIPHERAL (22 Addr Bits; 14 RAS and 8CAS)													
NET+ARM Chip PIN	A23	A22	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
DRAM PIN		A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
RAS		22	21	20	19	18	17	16	15	14	13	12	11	10	9	
CAS		22	0	0	0	10	9	8	7	6	5	4	3	2	1	
		16-BIT DRAM PERIPHERAL (21 Addr Bits; 13 RAS and 8 CAS)														
NET+ARM Chip PIN	A23	A22	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
DRAM PIN	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0		
RAS	23	22	21	20	19	18	17	16	15	14	13	12	11	10		
CAS	23	22	0	0	0	10	9	8	7	6	5	4	3	2		
	32-BIT DRAM PERIPHERAL (20 Addr Bits; 12 RAS and 8 CAS)															

Figure 10-5: Internal DRAM Multiplexing - Mode 1

10.5 NET+ARM External DRAM Address Multiplexing

The NET+ARM chip can be configured to use DRAM components using an external DRAM address multiplexer.

An external address multiplexer is required when designing a system with an external bus master. When interfacing with an external bus master, the NET+ARM chip memory controller cannot multiplex the address signals out the A13:A0 address pins since the NET+ARM chip would require those pins to be both an output (for the multiplexed signal) and an input (address inputs from the external bus master) at the same time, which is impossible.

An external address multiplexer may also be required when the selected SDRAM component cannot interface with the NET+ARM chip internal multiplexer. This happens as DRAM densities grow and the DRAM requirements for RAS/CAS addressing change.

Even though an external address multiplexer is used, the NET+ARM chip memory controller can be used to control the basic DRAM signal protocol. The NET+ARM chip can be configured to output the DRAM address multiplexer select signal out the PORTC3 pin. This is accomplished by setting the AMUX or the AMUX2 bit in the memory module configuration register (MMCR) or the DMUXS bit in the chip select base address register.

The AMUX bit is set to indicate that the internal address multiplexer must be disabled. When AMUX is set, the NET+ARM chip drives the address bus using standard addressing without any multiplexing. When AMUX is set, the internal address multiplexer is disabled and the multiplexer indicator is driven out the PORTC3 pin.

The AMUX2 control bit allows the internal multiplexer to operate normally and forces the PORTC3 signal to be driven. Using the AMUX2 setting allows the internal bus masters to use the internal address multiplexer and the external bus masters to perform their own address multiplexing by watching the state of the PORTC3 pin. The AMUX2 feature removes the need for an external physical address multiplexer (for example, F157 chips) provided the external bus master can perform the address multiplexing itself.

The DMUXS bit is set to indicate that the internal address multiplexer must be disabled when the specific chip select is activated. When DMUXS is set, the NET+ARM chip drives the address bus using standard addressing without any multiplexing, but only for the specific chip select. When DMUXS is set, the internal address multiplexer is disabled and the multiplexer indicator is driven out the PORTC3 pin.

The PORTC3 signal is driven active high during the CAS addressing portion for FP and EDO DRAMS. The PORTC3 signal is also driven active high during the SDRAM WRITE and READ commands. The NET+ARM chip also drives PORTC3 active high during the SDRAM LOAD MODE command. The NET+ARM chip drives the SDRAM LOAD MODE configuration on its lower address pins. At all other times, the PORTC3 signal is driven inactive low.

The SDRAM A10/AP pin must still be connected to the NET+ARM chip CAS0* pin even when using an external bus master. The NET+ARM chip pin samples the A20, A19, A18 signals and drives the proper value on the CAS0* output pin for use as the SDRAM A10/AP signal.

Additional multiplexing considerations are required to support Synchronous DRAM when using the AMUX2 and PORTC3 option. The NET+ARM chip requires the following when using SDRAM with an external bus master and the external bus master is providing address multiplexing on its own.

1. The external bus master must drive the CAS address values on A13:A0 while the PORTC3 signal is active high.
2. The external bus master must drive the RAS address values on A13:A0 while the PORTC3 signal is active low and the SDRAM controller is issuing the ACTIVE command. The ACTIVE command is encoded using the CAS[3:1] signals. The CAS[3:1] signals are in the “011” binary state during the ACTIVE command.
3. At all others times when PORTC3 is inactive and the SDRAM controller is not issuing the ACTIVE command, the external bus master must drive the non-multiplexed address signals on A13:A8. The NET+ARM chip samples the A23:A8 address bus 1 BCLK cycle before the PORTC3 signal is activated to sample the current SDRAM page address.

10.6 FP/EDO DRAM Controller

The memory controller module contains an integrated FP/EDO DRAM controller. Each chip-select can be configured to operate using DRAM. The DRAM controller provides the following features:

- Support both Page Mode, EDO DRAMs, and Synchronous DRAM (refer to section *10.7 Synchronous DRAM*)
- CAS before RAS Refresh Operation
- Programmable Refresh Timer
- Background Refresh Cycles (refresh while another chip-select access in progress)

- Allow External Bus Master Access to DRAM bank
- Support Normal and Burst (page/EDO) cycles
- Programmable Wait States for Normal (also first cycle in burst access), and Burst Cycles
- Programmable Base Address and Chip Select Size

10.6.1 Single Cycle Read/Write

Figure 10-6 shows FP DRAM normal read and write cycles. All DRAM cycles must operate a minimum of 1 wait state. This figure shows DRAM cycles with 2 wait states. A single wait state DRAM cycle requires the DRAM devices to tolerate a single BCLK cycle for RAS pre-charge and CAS access timing. The CAS* signal is deasserted on the rising edge in which TA* is recognized. The RAS* signal is deasserted on the falling edge of BCLK after CAS* is asserted.

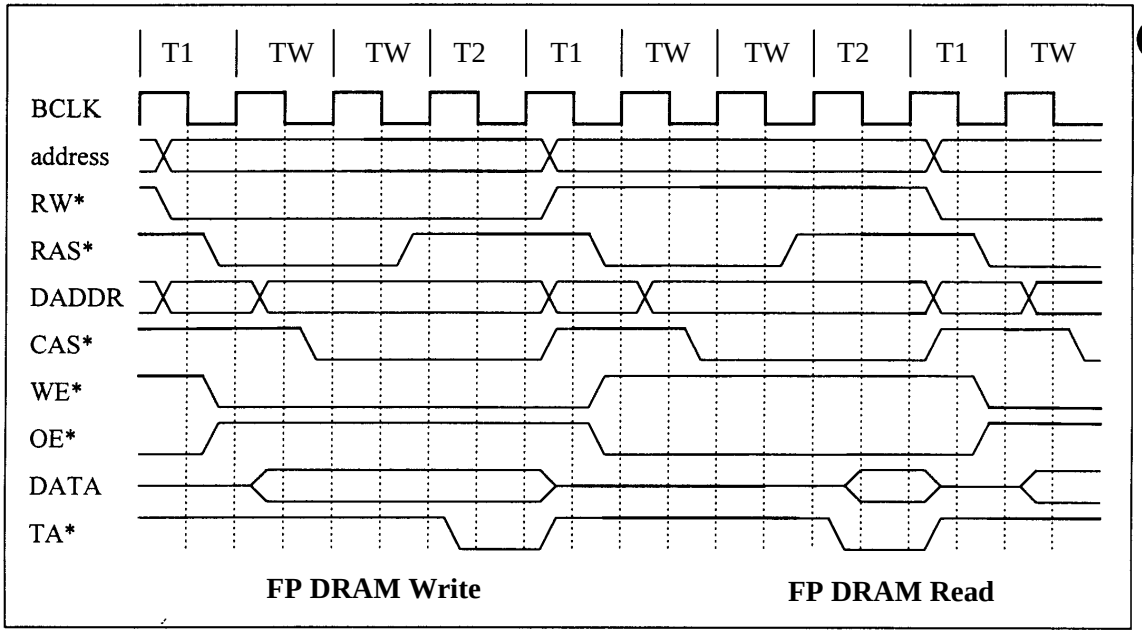


Figure 10-6: Normal FP DRAM Bus Cycles

10.6.2 Burst Cycles

The DRAM Controller can support both read and write burst cycles. A DRAM burst cycle must operate with a minimum of one wait state for the first cycle. Figure 10-7 shows a diagram of FP DRAM burst cycles

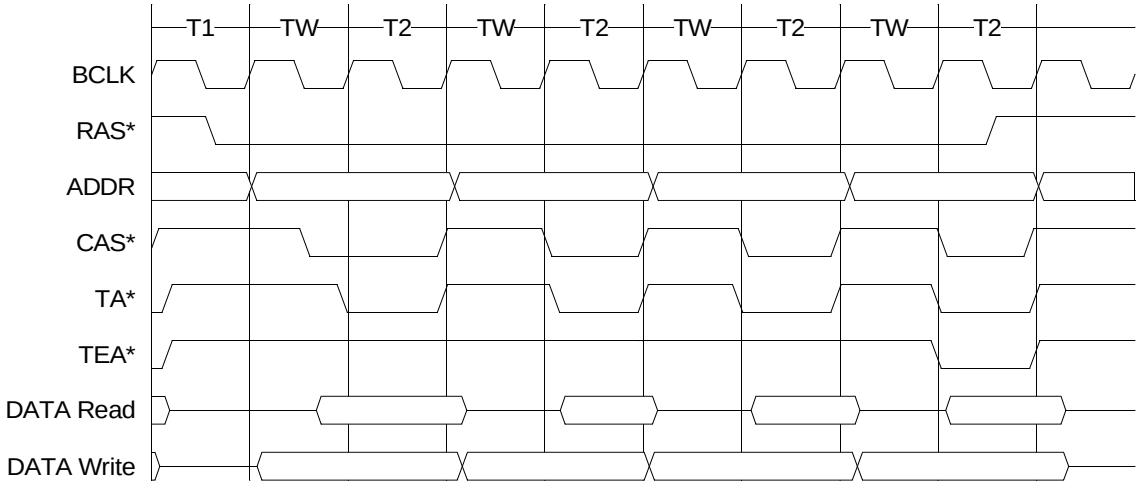


Figure 10-7: FP DRAM Burst Cycles

10.7 Synchronous DRAM

Starting with NETA15, the NET+ARM chip MEM module provides support for synchronous DRAMs. The DMODE field in the chip select base address register configures the chip select for synchronous DRAM. The NET+ARM chip does not provide the ability to bootstrap to synchronous DRAM. The NET+ARM chip architecture can support chips with either 1K or 4K refresh using the address multiplexing mode selections in the chip select base register.

10.7.1 SDRAM x16 Bursting Considerations

The NET+ARM chip cannot perform 16-bit burst operations from a x32 SDRAM. This is caused by the fact that once the SDRAM begins bursting, the SDRAM cannot go back in the page address. Bursting 16-bit values from a 32-bit SDRAM would require the SDRAM to stall its operation between 2 successive clock signals. The

SDRAM cannot operate in that fashion. As such, be advised that executing 16-bit Thumb code from a 32-bit SDRAM results in poor performance since the NET+ARM chip does not perform burst operations for the 16-bit thumb instruction fetches.

10.7.2 NET+ARM Chip SDRAM Interconnect

This section describes how to interconnect the NETA15 and NETA40 parts of the NET+ARM chip family to standard 16M and 64M SDRAM components.

X32 SDRAM Configuration

The following chart identifies the interconnect between the NET+ARM chip and SDRAM when the SDRAM is used in a x32 configuration. Typically, a x32 SDRAM configuration is constructed using (2) x16 SDRAM components. In the following chart, the two components are identified as “Device 1” and “Device 2.”

NET+ARM Chip Signal	16M SDRAM Signal		64M SDRAM Signal	
CS/RAS*	CS*		CS*	
CAS3*	RAS*		RAS*	
CAS2*	CAS*		CAS*	
CAS1*	WE*		WE*	
CAS0*	A10/AP		A10/AP	
BE3*	UDQM*	Device 1	UDQM*	Device 1
BE2*	LDQM*	Device 1	LDQM*	Device 1
BE1*	UDQM*	Device 2	UDQM*	Device 2
BE0*	LDQM*	Device 2	LDQM*	Device 2
A2	A0		A0	
A3	A1		A1	
A4	A2		A2	
A5	A3		A3	

Table 10-3: x32 SDRAM Interconnect

NET+ARM Chip Signal	16M SDRAM Signal		64M SDRAM Signal	
A6	A4		A4	
A7	A5		A5	
A8	A6		A6	
A9	A7		A7	
A10	A8		A8	
A11	A9		A9	
A12				
A13			A11	
A21	BA			
A22			BA0	
A23			BA1	
BCLK	CLK		CLK	
VCC	CKE		CKE	
D31-D16	D15-D0	Device 1	D15-D0	Device 1
D15-D00	D15-D0	Device 2	D15-D0	Device 2

Table 10-3: x32 SDRAM Interconnect (Continued)

X16 SDRAM Configuration

The following chart identifies the interconnect between the NET+ARM chip and SDRAM when the SDRAM is used in a x16 configuration. Typically, a x16 SDRAM configuration is constructed using (1) x16 SDRAM component.

NET+ARM Chip Signal	16M SDRAM Signal	64M SDRAM Signal
CS/RAS*	CS*	CS*
CAS3*	RAS*	RAS*
CAS2*	CAS*	CAS*
CAS1*	WE*	WE*
CAS0*	A10/AP	A10/AP
BE3*	UDQM*	UDQM*
BE2*	LDQM*	LDQM*
BE1*	-	-
BE0*	-	-
A1	A0	A0
A2	A1	A1
A3	A2	A2
A4	A3	A3
A5	A4	A4
A6	A5	A5
A7	A6	A6
A8	A7	A7
A9	A8	A8
A10	A9	A9
A11		
A12		A11
A13		
A20	BA	
A21		BA0
A22		BA1

Table 10-4: 16 SDRAM Interconnect

NET+ARM Chip Signal	16M SDRAM Signal	64M SDRAM Signal
A23		
BCLK	CLK	CLK
VCC	CKE	CKE
D31-D16	D15-D0	D15-D0

Table 10-4: 16 SDRAM Interconnect (Continued)

10.7.3 SDRAM A10/AP Support

SDRAM components have a peculiar signal referred to as A10/AP. This signal multiplexes one of the RAS signals with an Auto Pre-charge command indicator. During the ACTIVE command, the A10/AP signal must provide the DRAM A10 logical RAS address value. During the READ and WRITE commands, the AP signal provides the auto pre-charge command indicator (the NET+ARM chip does not use automatic pre-charge, the NET+ARM chip always drives 0 during READ and WRITE commands). During the PRECHARGE command, the A10/AP signal indicates if all banks should be pre-charged. The NET+ARM chip always causes all banks to be pre-charged.

The NET+ARM chip provides the A10/AP multiplexing function using the CAS0* pin. During the ACTIVE command, the CAS0* pin is driven with either the logical value of A20, A19, or A18 as a function of the port size configuration defined in the chip select option register. A port size configuration of 32-bit causes A20 to be driven on CAS0*, a 16-bit port size uses A19, and finally an 8-bit port size uses A18. During the READ or WRITE commands, the NET+ARM chip always drives a 0 on the CAS0* pin (indicating not to perform automatic pre-charge). During the PRECHARGE command, the NET+ARM chip always drives a 1 on the CAS0* pin (indicating to pre-charge all banks).

10.7.4 Command Definitions

Synchronous DRAMs operate according to a series of “command codes” that are issued while the chip select input is active low. The command codes are registered using the low to high transition of the synchronous clock. The NET+ARM chip implementation requires that all synchronous DRAMs are synchronized to the NET+ARM chip BCLK signal. Table 10-5 identifies the encoding of the commands supported by the NET+ARM chip MEM module.

Command	CSx*	A13:0	CAS3* RAS#	CAS2* CAS#	CAS1* WE#	CAS0* A10/ap
Inhibit	1	X	X	X	X	X
NOP	0	X	1	1	1	X
ACTIVE	0	Bank/Row	0	1	1	A10
READ	0	Column	1	0	1	0
WRITE	0	Column	1	0	0	0
Burst Term	0	X	1	1	0	X
Precharge	0	X	0	1	0	1
Refresh	0	X	0	0	1	X
Load Mode	0	Op-Code	0	0	0	0

Table 10-5: SDRAM Command Definitions

10.7.5 WAIT Configuration

The WAIT configuration in the chip select option register is responsible for providing the SDRAM T_{RCD} and T_{RP} parameters. When WAIT is configured with a value of 0, then ACTIVE and PRECHARGE commands can immediately be followed by another command on the next active edge of BCLK. When WAIT is configured with a value larger than 0, then wait states are inserted after the ACTIVE and PRECHARGE commands before another command can be issued.

10.7.6 BCYC Configuration

The BCYC configuration in the chip select option register is responsible for providing the SDRAM CAS Latency parameter. The BCYC field must be set to a value of CAS Latency – 1. The NET+ARM chip can support SDRAMs that have a CAS latency specification between 1 and 4 BCLK clocks.

CAS Latency	BCYC Configuration
1	00
2	01
3	10
4	11

10.7.7 BSIZE Configuration

The BSIZE configuration in the Chip Select Option Register is responsible for providing the SDRAM Burst Length parameter. The BSIZE field must be set to a value of "11" for Full Page.

BSIZE	Burst Length
00	2 Words (not supported)
01	4 Words (not supported)
10	8 Words (not supported)
11	Full Page

Table 10-6: SDRAM BSIZE Configuration

After the SDRAM Load Mode command has been issued to the SDRAM devices, the BSIZE field can be reconfigured for use as the memory peripheral's burst depth. It is safe to assume that the LOAD Mode command has been issued after at least one memory read cycle has been executed to the SDRAM device.

10.7.8 SDRAM Mode Register

The SDRAM Mode Register is automatically loaded by the MEM module during the first memory cycle to the SDRAM device after the "V" bit in the chip select base

address register is set to 1. The MEM module loads the mode register with the values identified in the following table.

Address	Field	Value
11:9	Write Burst Mode	001 - Single Location Access
8:7	Operating Mode	00 - Standard Operation
6:4	CAS Latency	BCYC + 1
3	Burst Type	0 - Sequential
2:0	Burst Length	7 - Full Page

Table 10-7: SDRAM Mode Register

The burst length value is a function of the BSIZE configuration. The NET+ARM requires that the burst length be configured to full page. Therefore, the BSIZE field in the chip select option register must be set to a value of “11” before the “V” bit is set in the chip select base address register. After the SDRAM Load Mode command has been issued to the SDRAM devices, the BSIZE field can be reconfigured for use as the memory peripheral's burst depth. It is safe to assume that the LOAD mode command has been issued after at least one memory read cycle has been executed to the SDRAM device.

10.7.9 SDRAM Read Cycles

Figure 13-10, *SDRAM Normal Read*, and Figure 13-11, *SDRAM Burst Read*, provide timing diagrams with both WAIT and BCYC configured with a value of 0.

The PRECHARGE command is issued, when necessary, during the T1 phase of a normal or burst read cycle. The PRECHARGE command is only issued when the ROW selection for the chip select has changed since the last access. Each chip select maintains a 14-bit register identifying the last row accessed. Additional clock cycles are inserted between the PRECHARGE command and the ACTIVE command depending upon the WAIT configuration.

The ACTIVE command is always issued after the PRECHARGE command. The ACTIVE command selects a newly activated row address. Additional clock cycles are inserted between the ACTIVE command and the READ command depending upon the WAIT configuration.

The READ command is issued in either the T1 or TW states dependent upon whether or not a PRECHARGE command was required. The READ command is issued to

select the starting column address for the current burst read operation. Additional wait states are inserted after the READ command depending upon the value of the BCYC configuration. The BCYC configuration identifies the CAS latency specification for the SDRAM.

The BURST STOP command is issued at the end of the current burst read operation. The SDRAM continues to burst read data for an additional x BCLK cycles after the BURST STOP command is issued. The value for x is calculated as CAS Latency minus 1. When the CAS latency value is greater than 1, additional wait states are inserted between T2 and the next system bus cycle to account for the delay. These additional bus cycles are identified as “TX” states.

10.7.10 SDRAM Write Cycles

Figure 13-12 and Figure 13-13 provide timing diagrams for SDRAM normal and burst writes respectively. These timing diagrams have both WAIT and BCYC configured with a value of 0.

The PRECHARGE command is issued, when necessary, during the T1 phase of a normal or burst read cycle. The PRECHARGE command is only issued when the ROW selection for the chip select has changed since the last access. Each chip select maintains a 14-bit register identifying the last row accessed. Additional clock cycles are inserted between the PRECHARGE command and the ACTIVE command depending upon the WAIT configuration.

The ACTIVE command is always issued after the PRECHARGE command. The ACTIVE command selects a newly activated row address. Additional clock cycles are inserted between the ACTIVE command and the READ command depending upon the WAIT configuration.

The WRITE command is always issued during the T2 state since data is only available at that time. If the PRECHARGE and ACTIVE commands are not required, then a NOP is inserted in the T1 state of the write cycle.

10.8 DRAM Refresh

The NET+ARM chip MEM module executes a refresh cycle that supports Fast Page, EDO, and SDRAM devices. The Fast Page and EDO devices are refreshed using the CAS-before-RAS technique, while the SDRAM devices are refreshed using the REFRESH command.

The figures in section *13.3.2 SDRAM Cycles* provide a timing diagram of the DRAM refresh cycle. This diagram clearly illustrates the CAS-before-RAS refresh cycle for

Fast Page and EDO DRAM. There is one BCLK transition that executes the SDRAM refresh cycle. This transition occurs on the rising edge of BCLK at the end of the RF4 state.

The RCYC field in the MMCR register controls how long the refresh cycle occurs. This setting provides the DRAM TRAS parameter. Table 10-8 identifies which of the states shown in the figures in Section *13.3.2 SDRAM Cycles* are executed for each of the RCYC configuration states. The RCYC field simply controls the time for which RAS* is asserted active low.

RCYC	Refresh States Executed
00	RF1 - RF8
01	RF1, RF2, RF3, RF4, RF5, RF6, RF8
10	RF1, RF2, RF3, Rf4, RF5, RF8
11	RF1, RF2, RF3, RF4, RF8

Table 10-8: Refresh Cycle Configuration

10.9 Peripheral Page Burst Size

For the NET+ARM chip to support Bursting to or from a memory peripheral device, the peripheral device must provide a minimum burst page size of 64 bytes.

The NET+ARM chip can begin a burst memory access on any address boundary. The NET+ARM chip continues the burst until any one of three conditions are met:

1. The current bus master terminates the burst.
2. The memory controller terminates the burst because the number of bus cycles for the burst cycle reached the maximum as defined in the BSIZE field of the chip select option register.
3. The current memory address has reached a 64-byte page boundary.

The master can choose to terminate the burst at any time. The memory controller limits the maximum number of bus cycles that can occur via the BSIZE field. The memory controller also terminates the burst when a 64-byte page boundary is encountered.

Consider the following example:

1. The memory peripheral is configured as a 32-bit peripheral
2. The memory peripheral has a 16-byte page size
3. The BSIZE field is set to the maximum value of “11” that implies a maximum of 16 bus cycles for each burst
4. The bus master begins a 16-byte burst cycle starting with address offset of 0xC

The bus master begins a 16-byte burst cycle starting at address 0xC. It expects to receive four long words of data from address offsets 0xC, 0x10, 0x14, and 0x18. The memory controller will allow the full burst since the BSIZE allows a total of 16 long words to be accessed.

The memory controller does a normal access cycle to address 0xC using the WAIT field to determine the access timing. The memory controller then follows with 3 burst beats to address offsets 0x10, 0x14, and 0x18 using the BCYC field to determine the access timing.

The memory peripheral properly delivers the data at address offset 0xC, however, has trouble providing the value at 0x10 since the NET+ARM chip is operating with the BCYC burst timing and the memory peripheral needed an access cycle using the WAIT timing since the access address has crossed the memory peripheral’s page boundary.

Therefore, if a memory peripheral has a page size that is less than 64 bytes, then the memory peripheral can only interface with the NET+ARM chip, using burst cycles, when both the WAIT and BCYC fields result in the same number of clock cycles for normal and burst cycles. The following combinations are allowed.

WAIT	BCYC
0	1
1	2
2	3

Chapter 11
SYS Module

The SYS module provides the NET+ARM chip with its system clock and system reset resources.

11.1 System Clock Generation

The system clock is provided by an external crystal, an internal crystal oscillator, and an internal phase locked loop. The crystal oscillator circuit generates an internal CMOS transition clock signal. The clock signal from the crystal oscillator is initially divided by 5 and provided as an input to the phase locked loop (PLL).

The PLL is configured to operate in a multiplier mode. The output of the PLL is fed back through a four-bit programmable divider. This allows the PLL output to operate anywhere from 1 to 16 times the output of the prescaler, or 1/5th to 3X that of the input crystal frequency. The configuration for the PLL multiplication factor is provided by registers in the GEN module.

After multiplication, the system clock is divided down by the same value loaded in the PLL divider. This produces a clock signal that is always 1/5th the frequency of the crystal input regardless of what rate the PLL is programmed for. This clock is used as a reference timing pulse in both the GEN, and SER modules. Using a 18.432 MHz crystal as an example, the reference pulse generates a timing reference of 3.6864 MHz (quite useful for serial bit-rate generation and timers). The 1/5th reference timing pulse remains constant no matter what rate the PLL is programmed for system clock generation.

The PLL can be disabled and bypassed by asserting the PLLTST* input active low. In this mode, the system clock is provided by the XTAL1 input. Refer to Section *12.2 PLL* for more details on PLL testing.

11.2 Reset Circuit

The NET+ARM chip has four sources of hardware reset.

1. Power-up Reset
2. External Reset
3. Watch-Dog Reset
4. ENI Reset

The power-up reset comes from a special Ground PAD. This Ground PAD produces an active high reset pulse when the power voltage is outside specifications. The SYS module synchronizes this signal using a 5-bit counter with asynchronous clear. When the power up reset is active, the counter is cleared. When the reset is removed, the counter is allowed to count. When the counter reaches 31 (with no additional power up reset pulses), the counter triggers a hardware reset condition.

The External Reset comes from the RESET* input pin. The SYS module uses a six flip-flop stage to de-glitch the RESET* input. If the signal is low for this minimum pulse width, then a hardware reset condition is triggered when the RESET_ input is removed.

Both the Watch-Dog and ENI Reset are synchronous inputs. Whenever either of these inputs are active high, a hardware reset condition is asserted.

When a PowerUp or External reset condition is triggered, the SYS module drives the internal hardware reset signal active for a total of 512 system clock periods.

Table 11-1 identifies which internal NET+ARM chip modules are reset under various reset conditions. The Power UP, RESET*, Watch Dog, and ENI conditions are all hardware reset conditions. The Software Reset condition is issued by the CPU using the software service register within the GEN module.

The PORTC4 special function output only works with software and ENI resets.

NET+ARM Chip Module	RESET Condition				
	Power Up	RESET*	Watch Dog	ENI	Software
CPU	Yes	Yes	Yes	Yes	No
EFE	Yes	Yes	Yes	Yes	Yes
DMA	Yes	Yes	Yes	Yes	Yes
ENI	Yes	Yes	Yes	No	No
GEN[a]	Yes	Yes	Yes	Yes	Yes
MEM	Yes	Yes	Yes	No	No
SER	Yes	Yes	Yes	Yes	Yes

Table 11-1: RESET Operation

a. All registers in the GEN Module are reset during Power UP, RESET*, Watch Dog, ENI, and Software Reset with a few exceptions. The following GEN Module fields are reset during the POWER UP and RESET* condition only (NOT Watch Dog, ENI, Software Reset):
   BSPEED (GCR),
   BCLKD (GCR),
   PORTA,
   PORTB,
   PORTC
   The reason the above fields are not affected by Watch Dog, ENI, and Software Resets is to prevent the PORTA/B/C and CLOCK outputs from glitching when a reset occurs. This might cause a “valve” to accidentally switch.

The NET+ARM chip contains an integral power up reset circuit that generates the Power UP Reset condition. It is not recommended designs rely solely on the internal power up reset circuit. It is recommended that a system include an external power up reset circuit that drives the RESET* pin active low during power up.

The RESET* pin can be driven active low to reset the NET+ARM chip. RESET* must be driven active low a minimum of 40ms after Vdd reaches the minimum

operating value of 3V. The RESET* pulse width must be a minimum of 1us at other times.

The Watch Dog reset is provided by the Watch Dog timer in the GEN Module. Once the Watch Dog Timer is enabled, the software service register must be accessed to reset the Watch Dog Timer before it expires. When the Watch Dog Timer expires, the Watch Dog Reset is issued.

The ENI Reset is issued when the RSTIO bit is set in the ENI shared register. The ENI Reset condition remains active until the RSTIO bit is cleared in the ENI shared register.

The Software Reset is issued using the software service register in the GEN Module. The Software Reset can be used to get all internal NET+ARM peripherals into a known reset state.

11.3 Clock Generation Circuit

The clock generation circuit takes an external clock input (using an external CMOS clock or the internal crystal oscillator circuit) and generates the internal system clock, the XTAL timing reference (GEN and SER timer circuits), and the system bus clock (BCLK).

Figure 11-1 shows the NET+ARM chip clock generation circuit.

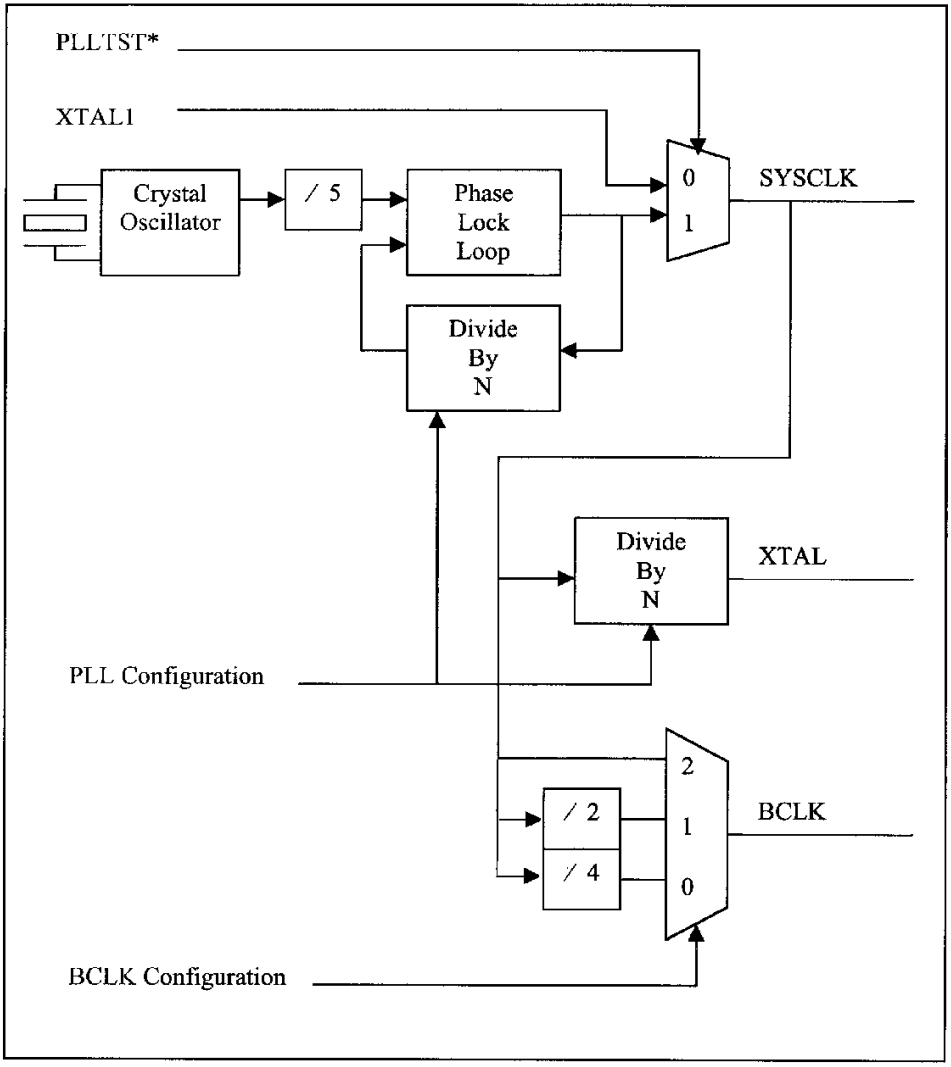


Figure 11-1: NET+ARM Chip Clock Generation

11.3.1 SYSCLK Generation

The SYSCLK output provides the central clock, which drives the ARM processor and internal logic clock for the NET+ARM chip. The frequency for SYSCLK is a function of either the PLL or an external TTL oscillator input.

When PLLTST* is inactive high, SYSCLK is a function of the Crystal Oscillator and Phase Lock Loop circuits. The PLL control register in the GEN module defines the multiplication factor between the input crystal oscillator and the SYSCLK operating frequency. The system frequency is controlled by the following equation:

IF (PLLCNT <= 3)

$$F_{SYSCLK} = (F_{CRYSTAL} / 5) * 6$$

ELSE

$$F_{SYSCLK} = (F_{CRYSTAL} / 5) * (PLLCNT + 3)$$

Where:

- PLLCNT represents the value loaded into the PLL control register
- F_{SYSCLK} identifies the frequency for ARM processor operation
- $F_{CRYSTAL}$ identifies the frequency of the External Crystal Component

Equation 1 - Fsysclk Equation when PLLTST = 1*

When the PLLTST* input is active low, the frequency of SYSCLK is a function of the TTL clock input applied to the XTAL1 pin.

$$F_{SYSCLK} = F_{XTAL1}$$

Where:

- F_{SYSCLK} identifies the frequency for ARM Processor Operation
- F_{XTAL1} identifies the frequency of the CMOS clock input on the XTAL1 pin.

Equation 2 - Fsysclk Equation when PLLTST = 0*

11.3.2 XTAL Clock Generation

The XTAL output provides the timing reference clock for the programmable timers in the GEN module and the Bit-Rate Generators in the SER module. The GEN and SER

modules use equations, which are a function of F_{XTAL}. The F_{XTAL} timing reference is always a constant regardless of the operating frequency of the Phase Lock Loop.

When PLLTST* is inactive high, then the value of F_{XTAL} is simply defined as follows:

$$F_{XTAL} = F_{CRYSTAL}/5$$

Where:

- F_{XTAL} identifies the timing reference used by the GEN and SER Modules.
- $F_{CRYSTAL}$ identifies the frequency of the External Crystal Component

Equation 3 - Fxtal Equation when PLLTST = 1*

When PLLTST* is active low, the value of FXTAL is defined as follows:

IF (PLLCNT <= 3)

$$F_{XTAL} = F_{XTAL1} / 6$$

ELSE

$$F_{XTAL} = F_{XTAL1} / (PLLCNT + 3)$$

Where:

- PLLCNT represents the value loaded into the PLL Control Register
- F_{XTAL1} identifies the frequency of the external TTL clock on the XTAL1 pin.

Equation 4 - Fxtal Equation when PLLTST = 0*

11.3.3 BCLK Generation

The BCLK output identifies the speed of the system bus. The system bus BCLK frequency is either 1/4, 1/2, or 1 times the value of SYSCLK. The frequency for BCLK is a function of the BCLK configuration value in the System Control Register found in the GEN module.

11.4 NET+ARM Chip Bootstrap Initialization

Many internal NET+ARM chip configuration features are application specific and need to be configured at power-up before the CPU boots. The system bus address bits are used for this purpose during a powerup reset. The NET+ARM chip provides internal pullup resistors on all address signals. Weak external pulldown resistors can be employed to configure various internal register bits to a zero state.

The following identifies which ADDR bits control what functions.

ADDR[27]	:	Endian Configuration (see Section *8.2.1 System Control Register*)
		0 - Little Endian Configuration
		1 - Big Endian Configuration
ADDR[26]	:	CPU Bootstrap
		0 - ARM CPU Disabled; GEN_BUSER set to 1.
		1 - ARM CPU Enabled; GEN_BUSER set to 0.
ADDR[25]	:	GEN_IARB Setting (see Section *8.2.1 System Control Register*)
		0 - External System Bus Arbiter
		1 - Internal System Bus Arbiter
ADDR[24:23]	:	CS0 Bootstrap Setting (see Section *10.1.1 MEM Module Hardware Initialization)*
		“00” - Bootstrap Disabled
		“01” - 32-bit SRAM port; 15 wait-states
		“10” - 32-bit DRAM port; 15 wait-states
		“11” - 16-bit SRAM port; 15 wait-states
ADDR[22:20]	:	ENIMODE[2:0] Configuration Bits (see Section *6.5.2 General Control Register*)
ADDR[19:09]	:	GEN_ID Setting (see Section *8.2.2 System Status Register*)
ADDR[8]	:	Reserved
ADDR[07]	:	ENI Control PSIO 0 = PSIO; 1 = Normal (see Section *6.5.10 ENI Control Register*)

ADDR[06]	:	ENI ControlWR_OC (see Section *6.5.10 ENI Control Register*)
ADDR[05]	:	ENI ControlDINT2* (see Section *6.5.10 ENI Control Register*)
ADDR[04]	:	ENI Control I_OC (see Section *6.5.10 ENI Control Register*)
ADDR[03]	:	ENI Control DMAE* (see Section *6.5.10 ENI Control Register*)
ADDR[02]	:	Reserved (see Section *6.5.10 ENI Control Register*)
ADDR[01]	:	ENI ControlEPACK* (see Section *6.5.10 ENI Control Register*)
ADDR[00]	:	ENI ControlPULINT* (see Section *6.5.2 General Control Register*)

Note: The inverted PSIO address bit (ADDR7) is loaded into the PSIO configuration bit within the ENI Control Register. In the ENI Control Register, 0=Normal, 1=PSIO).

Note: The inverted ENDIAN bit (ADDR27) is loaded into the LENDIAN bit within the System Control Register. In the System Control Register, LENDIAN=1 (for little endian mode) and LENDIAN=0 (for big endian mode).

Chapter 12
Test Support

The PLLTST*, BISTEN* and SCANEN* primary inputs are used to control various test modes for both functional and manufacturing test operations.

PLLTST*	BISTEN*	SCANEN*	Mode
0	0	0	ATPG Scan
0	0	1	BIST / ATPG Parallel
0	1	1	Normal w/ PLL Bypass
1	0	1	HIZ
1	1	1	Normal w/ PLL Operational

Table 12-1: NET+ARM Chip Test Modes

12.1 ATPG

The NET+ARM chip supports ATPG testing using the full-scan methodology. The NET+ARM chip uses multiple scan chains. The scan chains propagate from primary inputs to primary outputs. Scan testing is enabled when SCANEN* is defined in its active low state.

During scan testing on a tester, the PLL must be disabled such that the tester has direct control of the system clock. The PLLTST* signal, when pulled active low, allows the PLL to be bypassed and all system clocks to be provided by the XTAL1 primary input.

Scan Chain	PLLTST*	BISTEN*	SCANEN*	Input	Output
Chain 1	0	0	0	PA0	PORTA0
Chain 2	0	0	0	PA1	PORTA1
Chain 3	0	0	0	PA2	PORTA2
Chain 4	0	0	0	PA3	PORTA3
Chain 5	0	0	0	PA4	PORTA4
Chain 6	0	0	0	PA5	PORTA5
Chain 7	0	0	0	PA6	PORTA6
Chain 8	0	0	0	PA7	PORTA7
Chain 9	0	0	0	PA8	PORTB0
Chain 10	0	0	0	PA9	PORTB1
Chain 11	0	0	0	PA10	PORTB2
Chain 12	0	0	0	PA11	PORTB3
Chain 13	0	0	0	PA12	PORTB4
Chain 14	0	0	0	PA13	PORTB5
Chain 15	0	0	0	PA14	PORTB6
Chain 16	0	0	0	PA15	PORTB7

Table 12-2: ATPG Test Mode Connections

12.2 PLL

When the PLLTST* signal is active low, the PLL is isolated and the internal system clock is provided by the XTAL1 input. In this mode, the PLL “TEST” input is driven active high. The external tester has access to the PLL “DIV” input via the TDI primary input and the PLL “CKR” input via the TCK primary input. When BISTEN* is also active low, the PLL “CK,” “CKN,” and “LOCK” outputs are visible via primary output pins.

PLLTST*	BISTEN*	SCANEN*	Pll120m1 Signal	NET+ARM Chip Primary IO
0	0	1	DIV	TDI (input)
0	0	1	CKR	TCK (input)
0	0	1	CK	PORTA7 (output)
0	0	1	CK_	PORTA6 (output)
0	0	1	LOCK	PORTA5 (output)

Table 12-3: PLL Test Mode Connections

12.3 BIST

The NET+ARM chip supports Built In Self Test (BIST) for the various compiled RAM modules. The BISTEN* input, when pulled low, enables the BIST testing features. In this mode, various outputs are reconfigured to provide the BIST results from various internal modules. The BIST clock is provided by the PA16 primary input.

PLLTST*	BISTEN*	SCANEN*	MegaCell	Output
0	0	1	aram64	PORTC6
0	0	1	dpr32x33	PORTC5
0	0	1	dpr512kx32	PORTC4
0	0	1	dpr32x40	PORTC3
0	0	1	dpr8x36	PORTC1
0	0	1	dpr8x36	PORTC0
0	0	1	dpr8x36	PORTA2
0	0	1	dpr8x36	PORTA1
0	0	1	dpr8x40	PORTA0
0	0	1	dpr25664	PORTB7
0	0	1	dpr25664	PORTB6
0	0	1	dpr25664	PORTB5
0	0	1	dpr25664	PORTB4
0	0	1	dpr256x2	PORTB3
0	0	1	dpr8x10	PORTB2
0	0	1	dpr8x40	PORTB1
0	0	1	dpr8x10	PORTB0

Table 12-4: BIST Test Mode Connections

12.4 ATE Testing

The NET+ARM chip aids ATE testing by supporting an easy means to tri-state all outputs. When PLLTST* is inactive and BISTEN* is active, then all outputs are placed in a low current tri-state mode.

12.5 ARM DEBUG

The ARM7TDMI core contains hardware extensions for advanced debugging features. These are intended to ease the testing/development of application software, operating systems, and the hardware itself. The debug extensions allow the core to be stopped either on a given instruction fetch (breakpoint) or data access (watchpoint), or asynchronously by a debug-request. When this happens, the ARM7TDMI is said to be in debug state. At this point, the core's internal state and the system's external state may be examined. Once examination is complete, the core and system state may be restored and program execution resumed.

The ARM7TDMI is forced into debug state by an internal functional unit known as ICEBreaker. Once in debug state, the core isolates itself from the memory system. The core can then be examined while all other system activities continue as normal (for example, DMA operations).

ARM7TDMI's internal state is examined via the 5-pin JTAG interface. This interface allows instructions to be serially inserted into the core's pipeline without using the external data bus. Thus, when in debug state, a store-multiple (STM) could be inserted into the instruction pipeline and this would dump the contents of the ARM7TDMI's registers. This data can be serially shifted out without affecting the rest of the system.

Chapter 13
Electrical Specifications

13.1 Thermal Considerations

Characteristic	Symbol	Min	Max	Unit
Thermal Resistance – Junction to Ambient	θ_{JA}		31	$^oC/W$
Operating Junction Temperature	T_J	-40	100	^oC
Operating Ambient Temperature	T_A	-40	85	^oC
Storage Temperature	T_{STG}	-60	150	^oC
Internal Core Power @ 3.3V – Cache Enabled	P_{INT}		15	mW / MHz
Internal Core Power @ 3.3V – Cache Disabled	P_{INT}		9	mW / MHz

Table 13-1: Thermal Considerations

The average chip-junction temperature, T_J, in ^oC can be obtained from:

$$T_J = T_A + (P_D * \theta_{JA}) \quad (1)$$

Where:

T_A : Ambient Temperature

θ_{JA}: Package Thermal Resistance, Junction-to-Ambient, $^oC/W$

P_D: $(P_{INT} * F_P) + P_{I/O}$

P_{INT}: Internal Core Power (mW / MHz)

F_P: Frequency of Operation

$P_{I/O}$: Power Dissipation on Input and Output Pins – User Determined

The calculations for $P_{I/O}$ are very application specific.

$$P_{I/O} = V_{DD} * V_{DD} * C_L * S_P * F_P / 2000 \qquad (2)$$

Where:

V_{DD} : Output Driver Voltage

C_L : Average output capacitive load

S_P : Estimated number of output pads switching simultaneously

F_P : Operating Frequency of Outputs

For most applications, $P_{I/O} < 0.3 * P_{INT}$ and can be neglected.

An approximate relationship between P_D and T_J (assuming $P_{I/O}$ is neglected) is:

$$P_D = K / (T_J + 273^oC) \qquad (3)$$

Solving Equations (1) and (2) for K gives:

$$K = P_D * (TA + 273^oC) + \theta_{JA} * P_D^2 \qquad (4)$$

Where K is a constant pertaining to the particular part. K can be determined from equation (4) by measuring P_D (at thermal equilibrium) for a known T_A. Using this value of K, the value of P_D and T_J can be obtained by solving Equations (1) and (3) iteratively for any value of T_A.

13.2 DC Characteristics

Sym	Parameter	Conditions	Min	Max	Unit
V_{DD3}	DC supply voltage	Core and standard I/Os	-0.3	4.6	V
V_I	DC input voltage, 3.3 V I/Os		-0.3	V_{DD3}+0.3, 4.6 max	V
V_O	DC output voltage, 3.3 V I/Os		-0.3	V_{DD3}+0.3, 4.6 max	V
TEMP	Operating free air temperature range	Industrial	-40	+85	°C
T_{STG}	Storage Temperature		-60	+150	°C

Table 13-2: Absolute Maximum Ratings

Note: Table 13-2 identifies the maximum values for voltages that the NET+ARM chip can withstand without being damaged. Operating the NET+ARM chip outside the voltages defined in Table 13-3 will result in unpredictable behavior.

Sym	Parameter	Conditions	Min	Typ	Max	Unit
V_{DD}	DC supply voltage		3.0	3.3	3.6	V
V_{IH}	Input High Voltage		2.0		V_{DD}+0.3	V
V_{IL}	Input Low Voltage		V_{SS} - 0.3		0.8	V
I_{IL}	Input Leakage Current	V_{IH} = 3.6V V_{IN} = 0 to V_{DD}	 -10		+5 +5	uA uA
I_{PULLUP}	Internal Pullup Current		178		352	uA
$I_{PULLDOWN}$	Internal Pulldown Current		99		429	uA
V_T	Input Switching Threshold Voltage		1.4	2.0		V
C_{IN}	Input Capacitance	Any Input		5		pF
I_{CC}	V_{DD} Active Current	f_{SYS} = 33 Mhz		100	150	mA
V_{DD}	DC supply voltage		3.0	3.3	3.6	V

Table 13-3: DC Characteristics - Inputs

Sym	Parameter	Conditions	Min	Typ	Max	Unit
V_{OL}	Output Low Voltage	Type: S2, I_{OL} = 2 mA;	0		0.4	V
		Type: S4, I_{OL} = 4 mA;	0		0.4	V
		Type: S8, I_{OL} = 8 mA;	0		0.4	V
V_{OH}	Output High Voltage	Type: S2, I_{OH} = 2 mA;	2.4		V_{DD}	V
		Type: S4, I_{OH} = 4 mA;	2.4		V_{DD}	V
		Type: S8, I_{OH} = 8 mA;	2.4		V_{DD}	V
P_D	Power Dissipation	NET+15			500	mW
P_D	Power Dissipation	NET+40			750	mW
I_{OZ}	High-Z Leakage Current	V_{PAD}=3.6V, V_{PAD}=0V	-5 -5		+5	uA uA
C_{OUT}	Output Capacitance				10	pF
C_{IO}	Input/Output Capacitance	Any I/O			15	pF
C_L	Output Load Capacitance	BCLK, TA*, TEA*, BR*, BG*			20	pF
		BUSY*, CSO-4*, MDC, MDIO			20	pF
		TXD3-0, TXER, TXEN, PEN*			20	pF
		PBRW*, TDO			20	pF
		BE3-0*, RW*, CAS3-0*, WE*			30	pF
		OE*			30	pF
		ADDR, DATA, TS*, PDATA			50	pF
		PACK*, PINT1*, PINT2*			50	pF
		PA13, PA14, PORTA, PORTB			50	pF
		PORTC			50	pF

Table 13-4: DC Characteristics - Outputs

13.3 AC Characteristics

Num	Characteristic	Symbol	Min	Max	Unit
1	On-Chip VCO System Frequency	F_{SYS}	DC	33.3	MHz
2	VCO System Cycle Time	T_{sys}	$1/F_{SYS}$		ns
3	Crystal Frequency	$F_{CRYSTAL}$	5	20	MHz
4	CPU Core Frequency	F_{CPU}		F_{SYS}	MHz
5	BCLK Frequency	F_{BCLK}	F_{SYS} /4	F_{SYS}	MHz

Table 13-5: Operating Frequencies

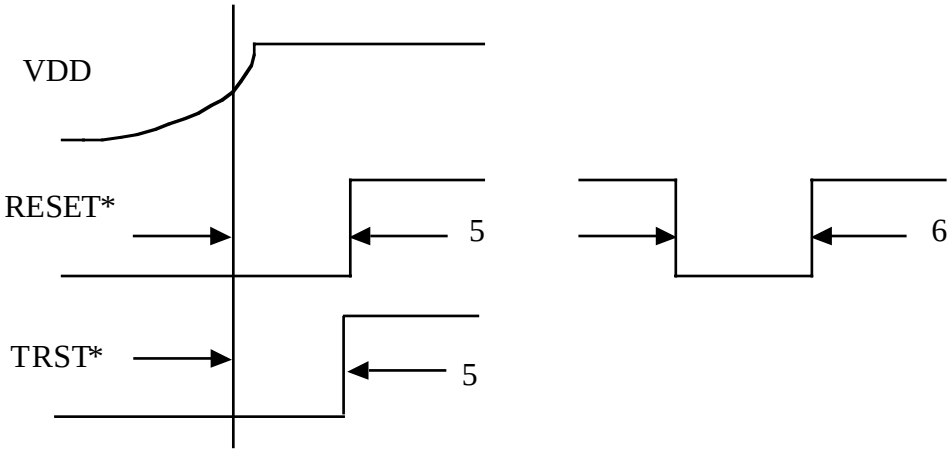


Figure 13-1: RESET* Timing

Num	Characteristic	Min	Max	Unit
5	VDD at 3.0 V to RESET* High	40		ms
6	RESET* Pulse Width Low	1		us

Table 13-6: Reset Timing

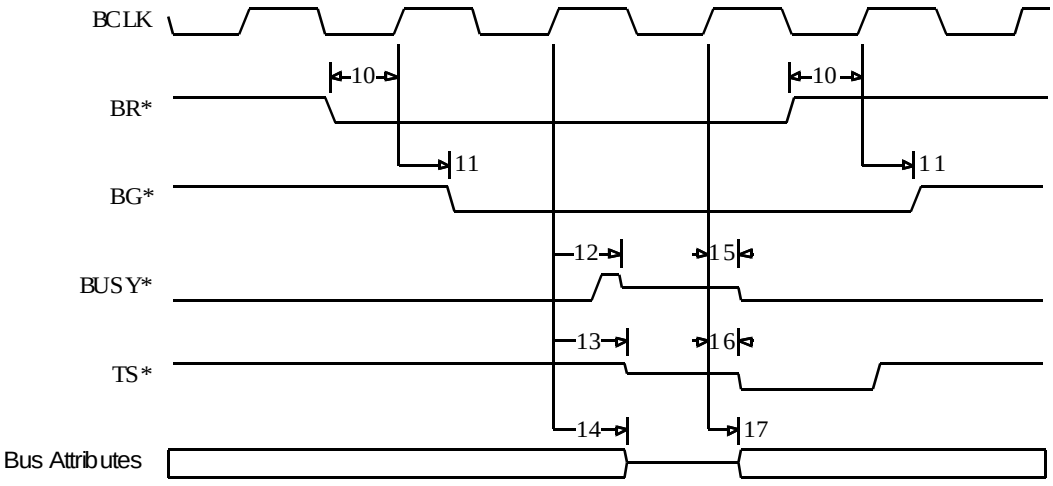


Figure 13-2: Internal Bus Arbitration Timing

Num	Characteristic	Min	Max	Unit
10	BR* valid to BCLK high (setup)	6		ns
11	BCLK high to BG* valid	2	4	ns
12	BCLK high to BUSY* high impedance	1	4	ns
13	BCLK high to TS* high impedance	1	4	ns
14	BCLK to Attributes high impedance	1	4	ns
15	BCLK to BUSY* valid	1	8	ns
16	BCLK high to TS* valid	1	4	ns
17	BCLK high to Attributes driven	3	15	ns

Table 13-7: Internal Bus Arbitration Timing

Num	Characteristic	Min	Max	Unit
20	BCLK high to BR* valid	1	4	ns
21	BCLK high to BG* valid (setup)	6		ns
22	BUSY* valid to BCLK high (setup)	6		ns
22B	BCLK high to BUSY* (hold)	0		ns
23	TS* valid to BCLK high (setup)	7		ns
23B	BCLK high to TS* (hold)	0		ns
24	Attributes high impedance to BCLK high (setup)	10		ns
25	BCLK high to BUSY* valid	1	8	ns
26	BCLK high to TS* valid	1	4	ns
27	BCLK high to Attributes valid	3	14	ns
28	BCLK high to BG* invalid (hold)	2		ns

Table 13-8: External Bus Arbitration Timing

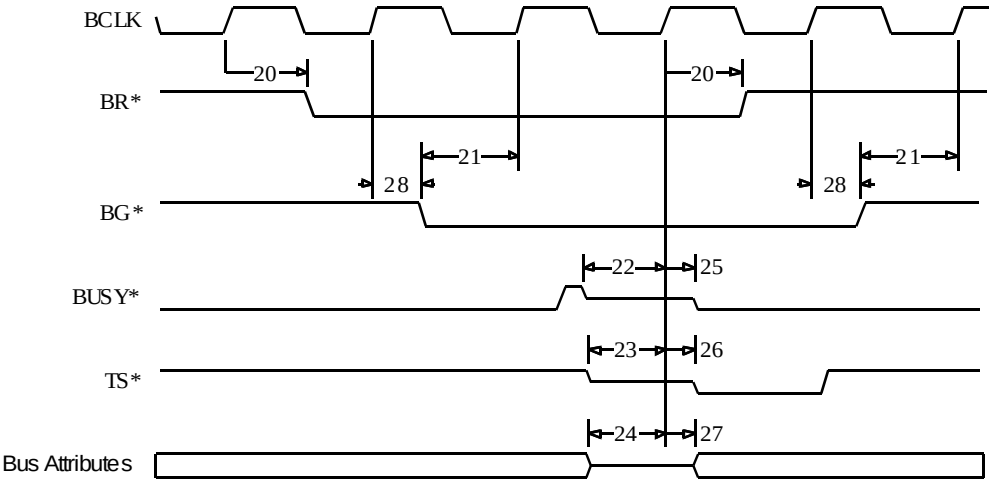


Figure 13-3: External Bus Arbitration Timing

Refer to Table 13-8.

Num	Characteristic	Min	Max	Unit
30	BCLK high to TS* valid	1	6	ns
31	BCLK high to RW* valid	4	8	ns
32	BCLK high to BE* valid	6	10	ns
33	BCLK high to Address valid	3	10	ns
34	BCLK high to Data High Impedance	1	8	ns
35	BCLK high to Data Valid	1	15	ns
36a	Data In valid to BCLK high (setup)	14		ns
36b	BCLK high to Data In invalid (hold)	0		ns
37a	TA* valid to BCLK high (setup)	7		ns
37b	BCLK high to TA* invalid (hold)	0		ns
38a	TEA* valid to BCLK high (setup)	6		ns
38b	BCLK high to TEA* invalid (hold)	0		ns
39	BCLK high to A25/BLAST* valid	3	10	ns

Table 13-9: External Bus Cycle Timing

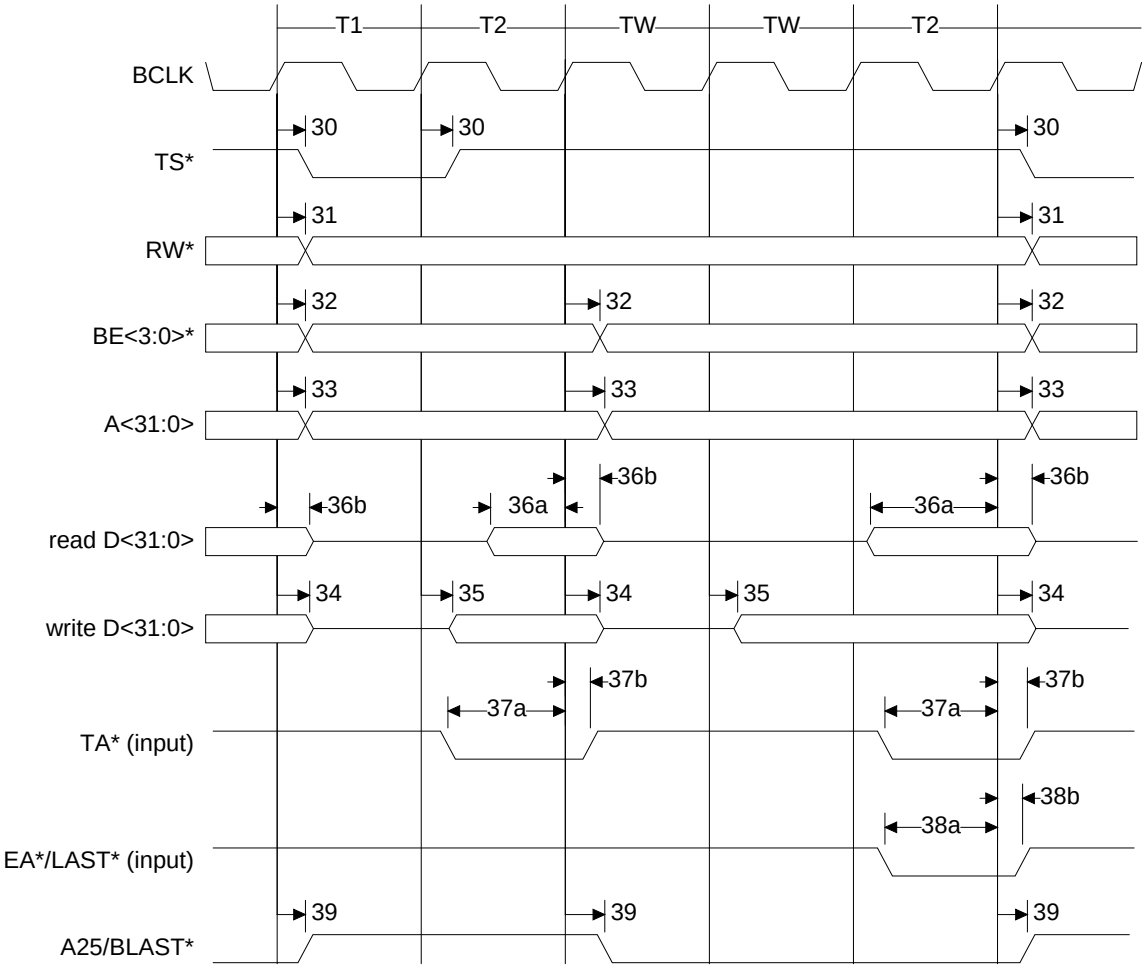


Figure 13-4: External Bus Cycle Timing

Refer to Table 13-9 for values.

Num	Characteristic	Min	Max	Unit
40	BCLK high to CS* valid	1	4	ns
41	BCLK low to OE* valid	2	8	ns
42	BCLK low to WE* valid	2	8	ns
43	BCLK high to TA* valid	2	10	ns
44	BCLK high to TEA* valid	1	12	ns
45	BCLK high to PORTC3 valid (external address multiplexing)	1	3	ns
46a	BCLK low to CAS* valid	1	5	ns
46b	BCLK high to CAS* valid	2	5	ns
46c	Address valid to CAS* low	0		ns
47	BCLK low to RAS* valid	1	5	ns

Table 13-10: Memory Controller Bus Timing

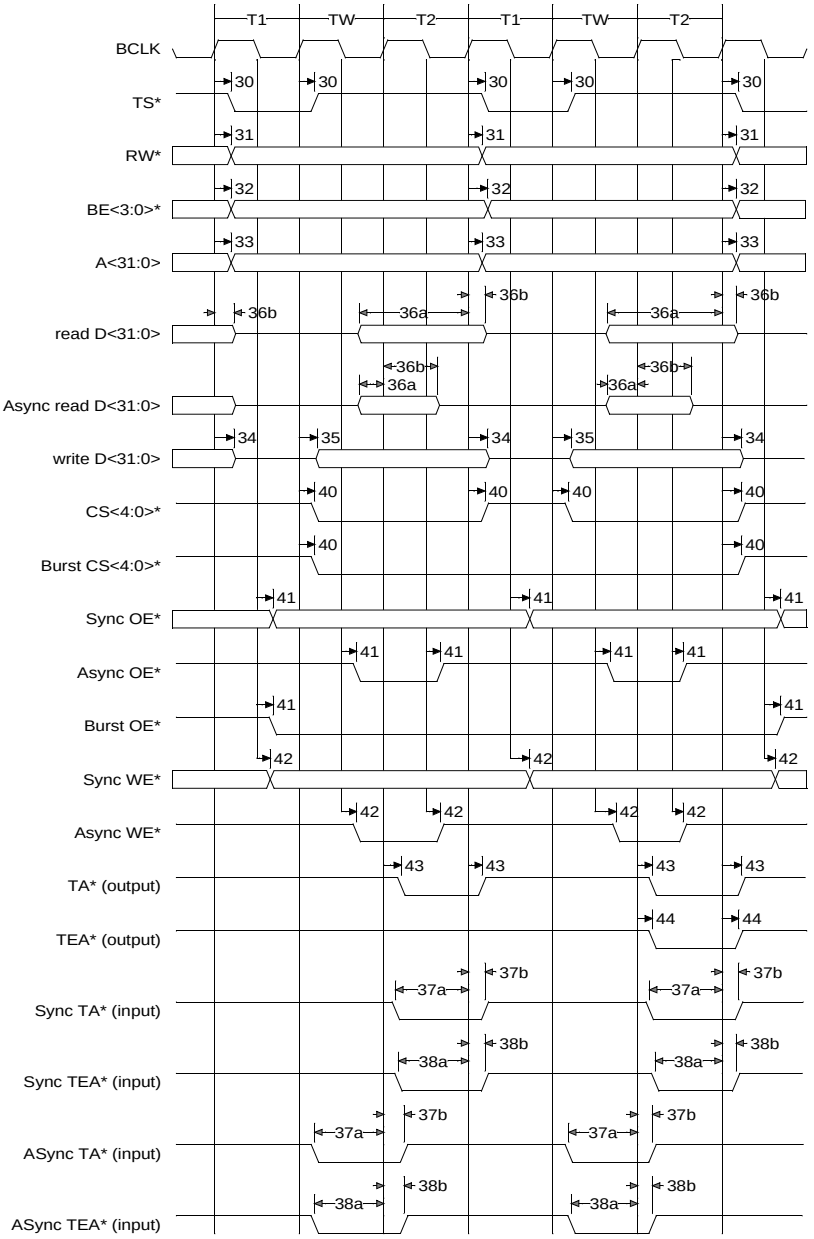


Figure 13-5: SRAM Bus Cycle Timing

Refer to Tables 13-9 and 13-10 for values.

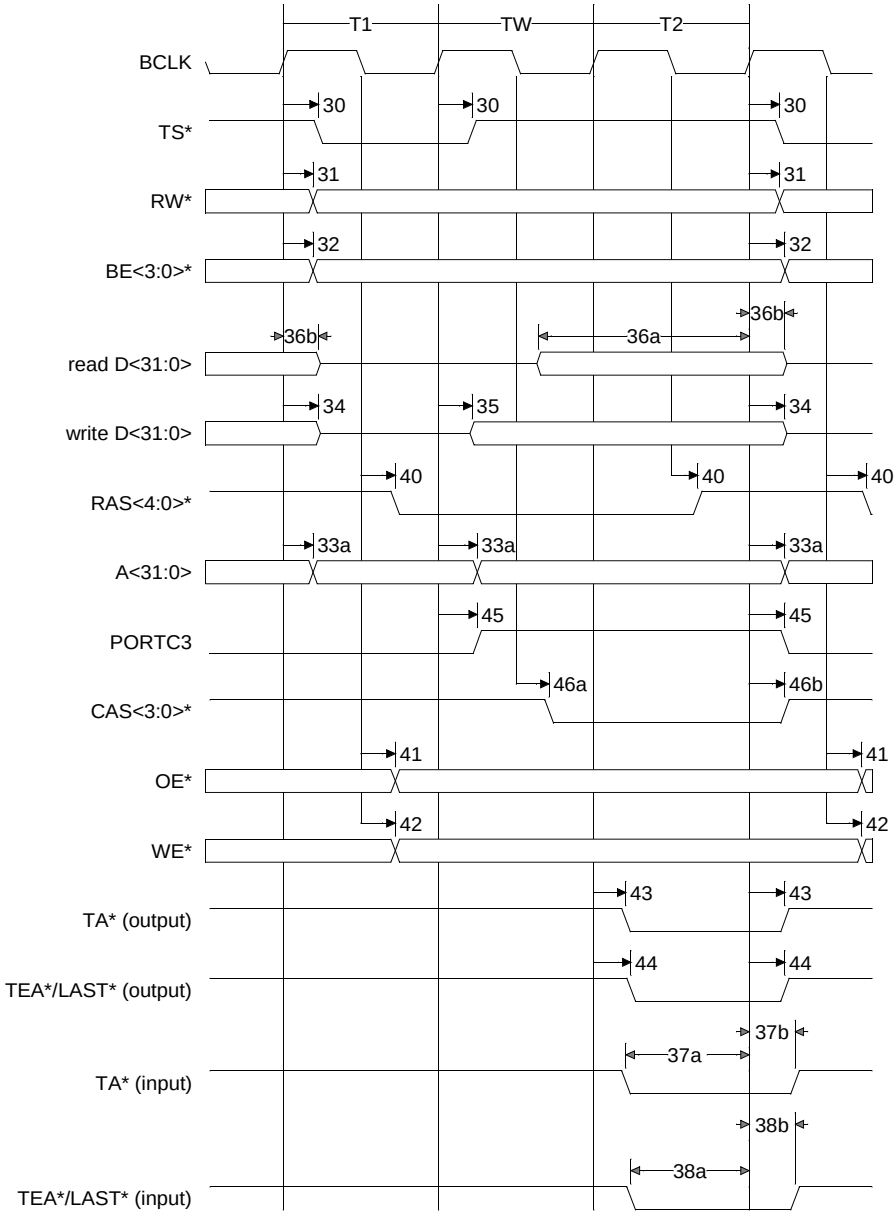


Figure 13-6: Fast Page DRAM Cycle - Normal

Refer to Tables 13-9 and 13-10 for values.

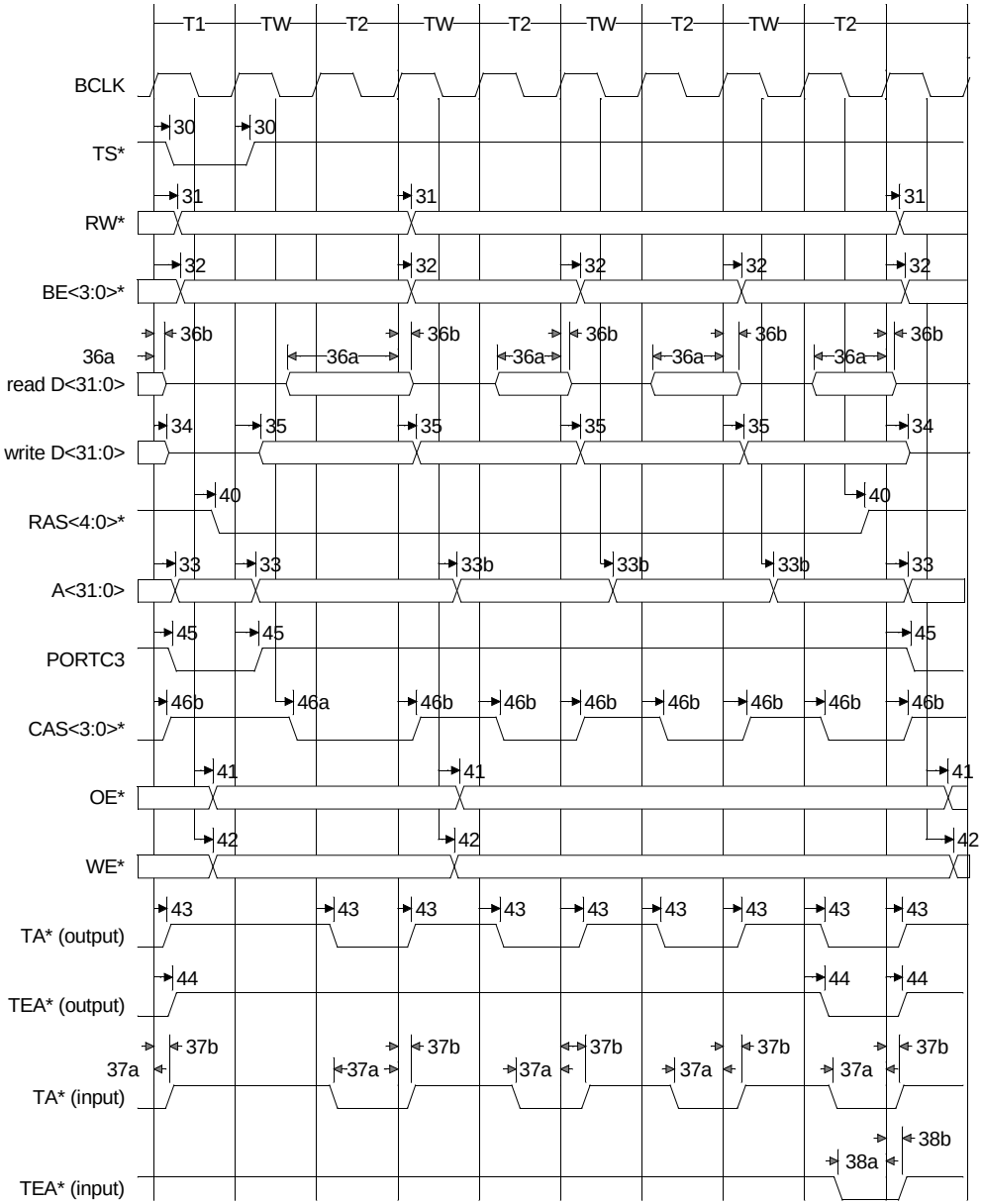


Figure 13-7: Fast Page DRAM Cycle - Burst

Refer to Tables 13-9 and 13-10 for values.

13.3.1 EDO DRAM Cycles

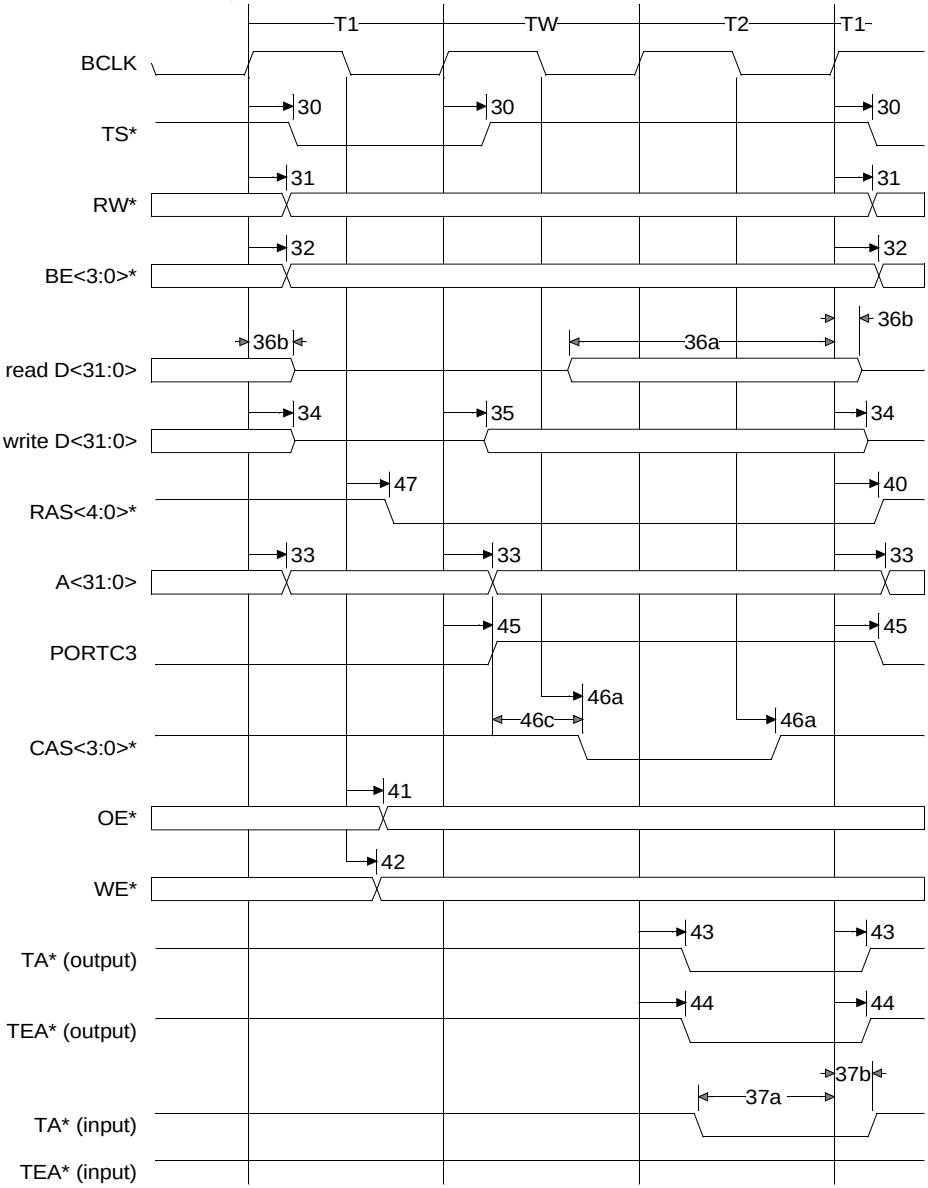


Figure 13-8: EDO DRAM Cycle - Normal

Refer to Tables 13-9 and 13-10 for values.

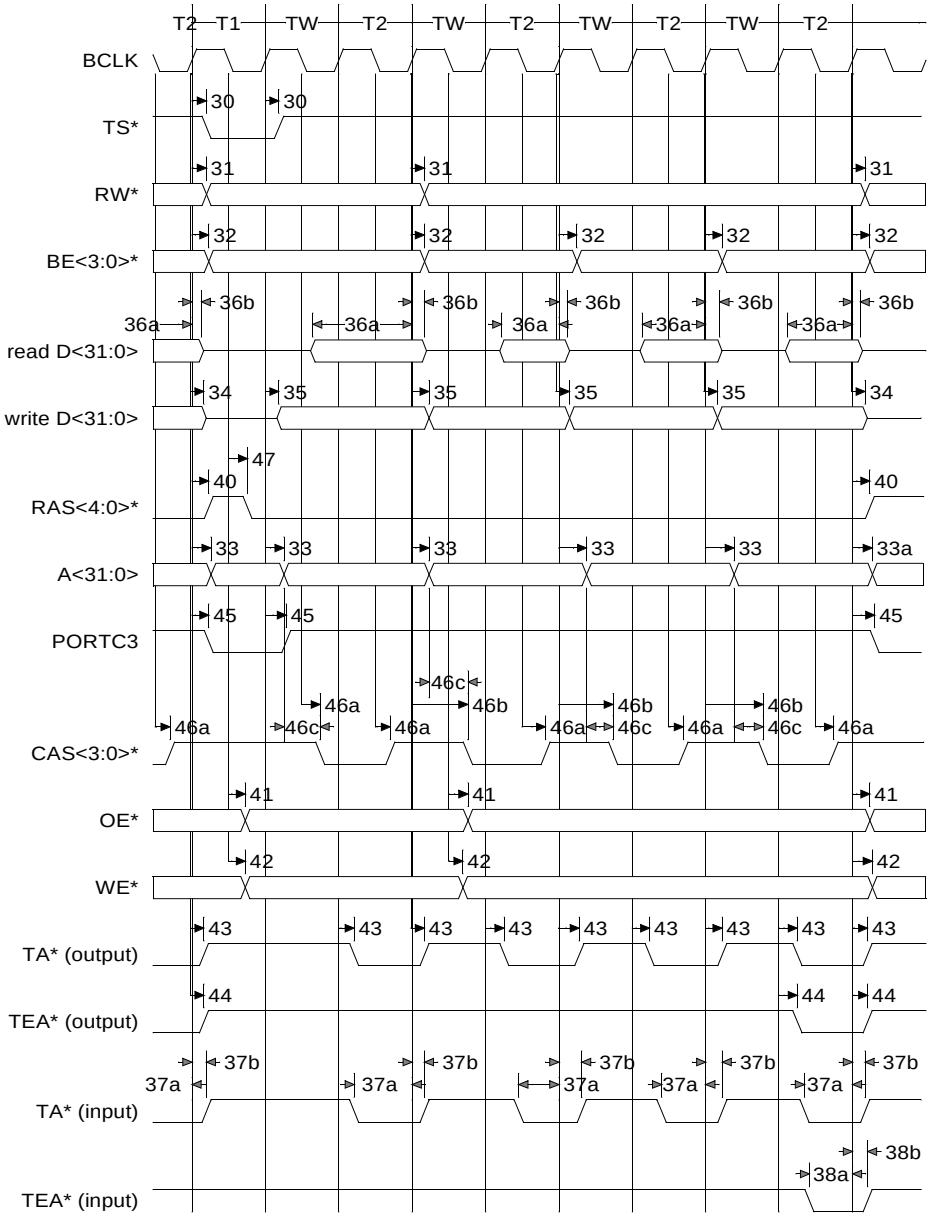


Figure 13-9: EDO DRAM Cycle - Burst

Refer to Tables 13-9 and 13-10 for values.

13.3.2 SDRAM Cycles

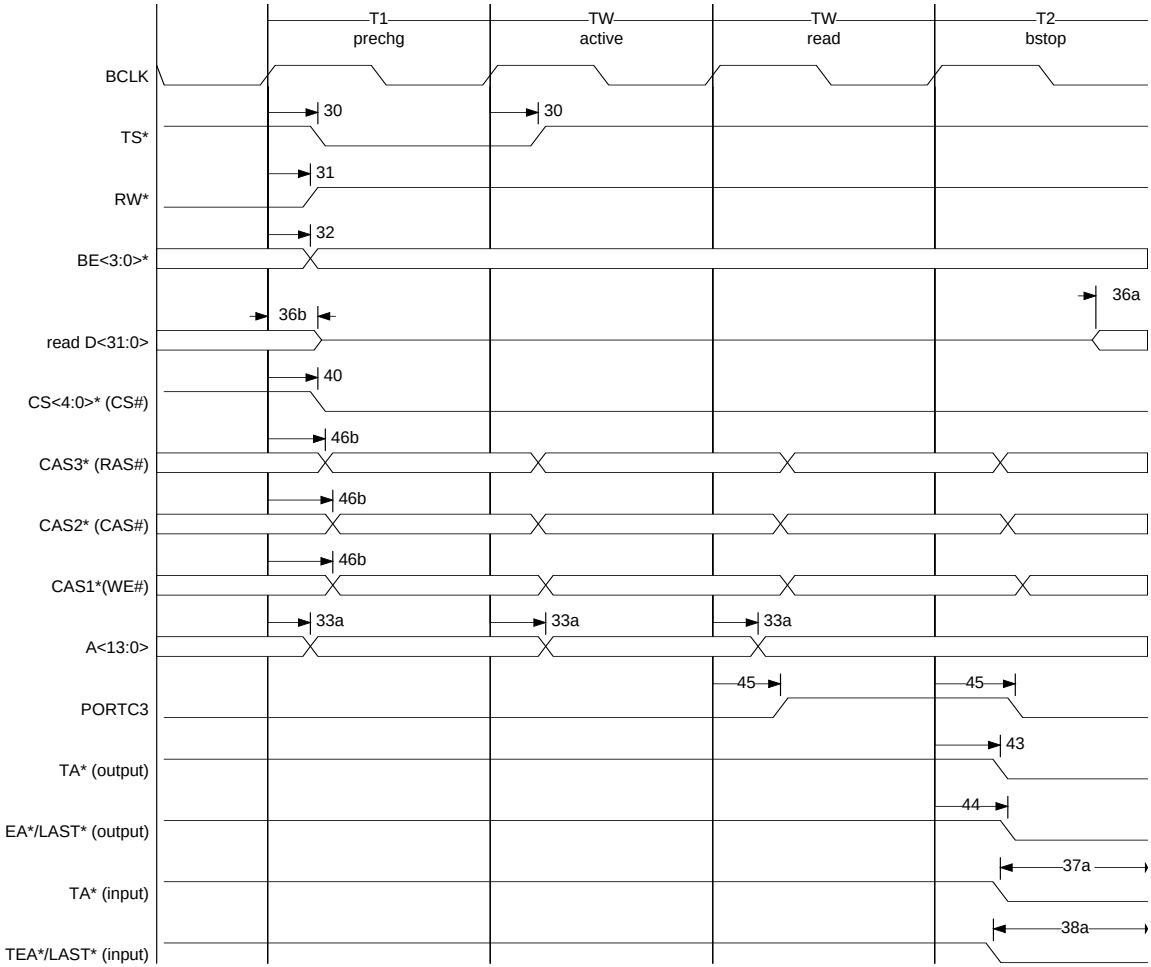


Figure 13-10: SDRAM Normal Read

Refer to Tables 13-9 and 13-10 for values.

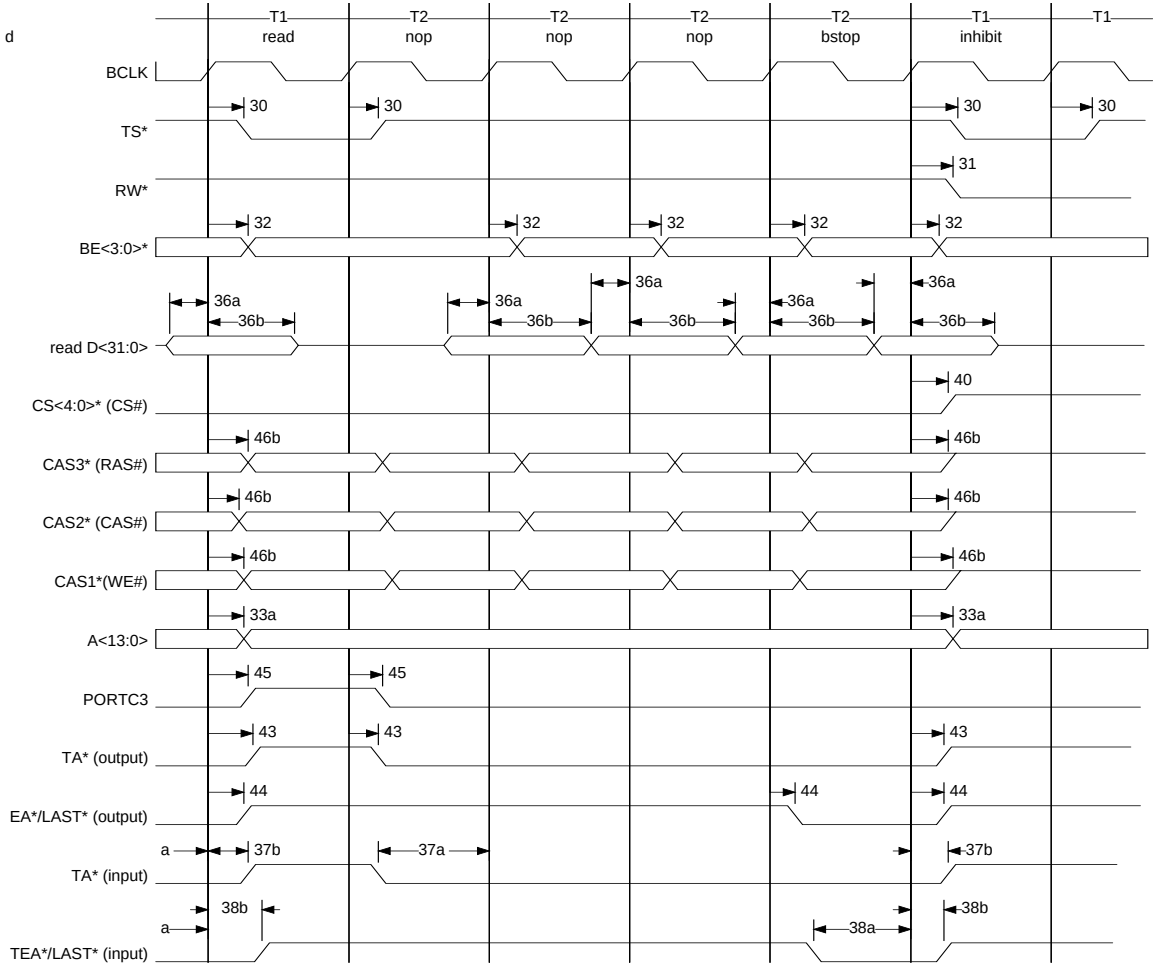


Figure 13-11: SDRAM Burst Read

Refer to Tables 13-9 and 13-10 for values.

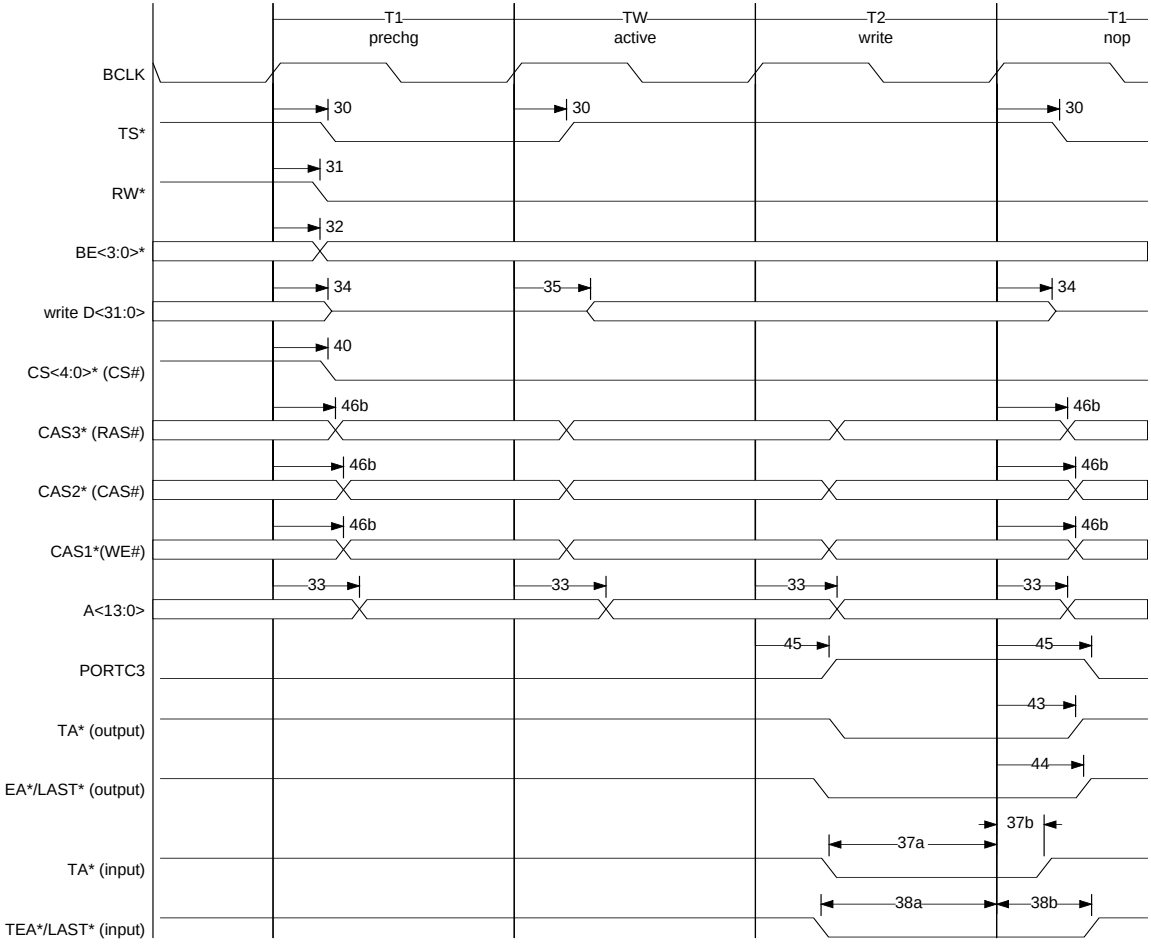


Figure 13-12: SDRAM Normal Write

Refer to Tables 13-9 and 13-10 for values.

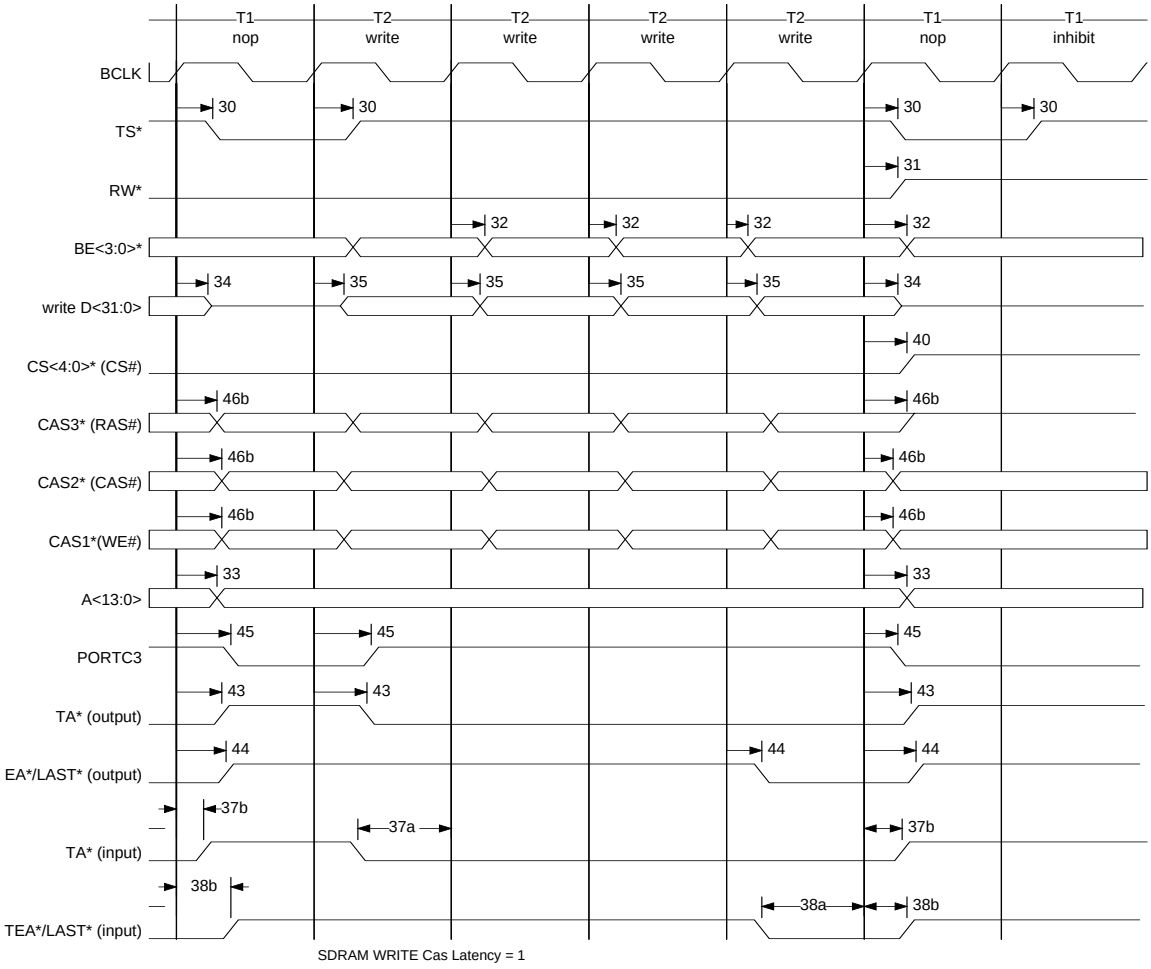


Figure 13-13: SDRAM Burst Write

Refer to Tables 13-9 and 13-10 for values.

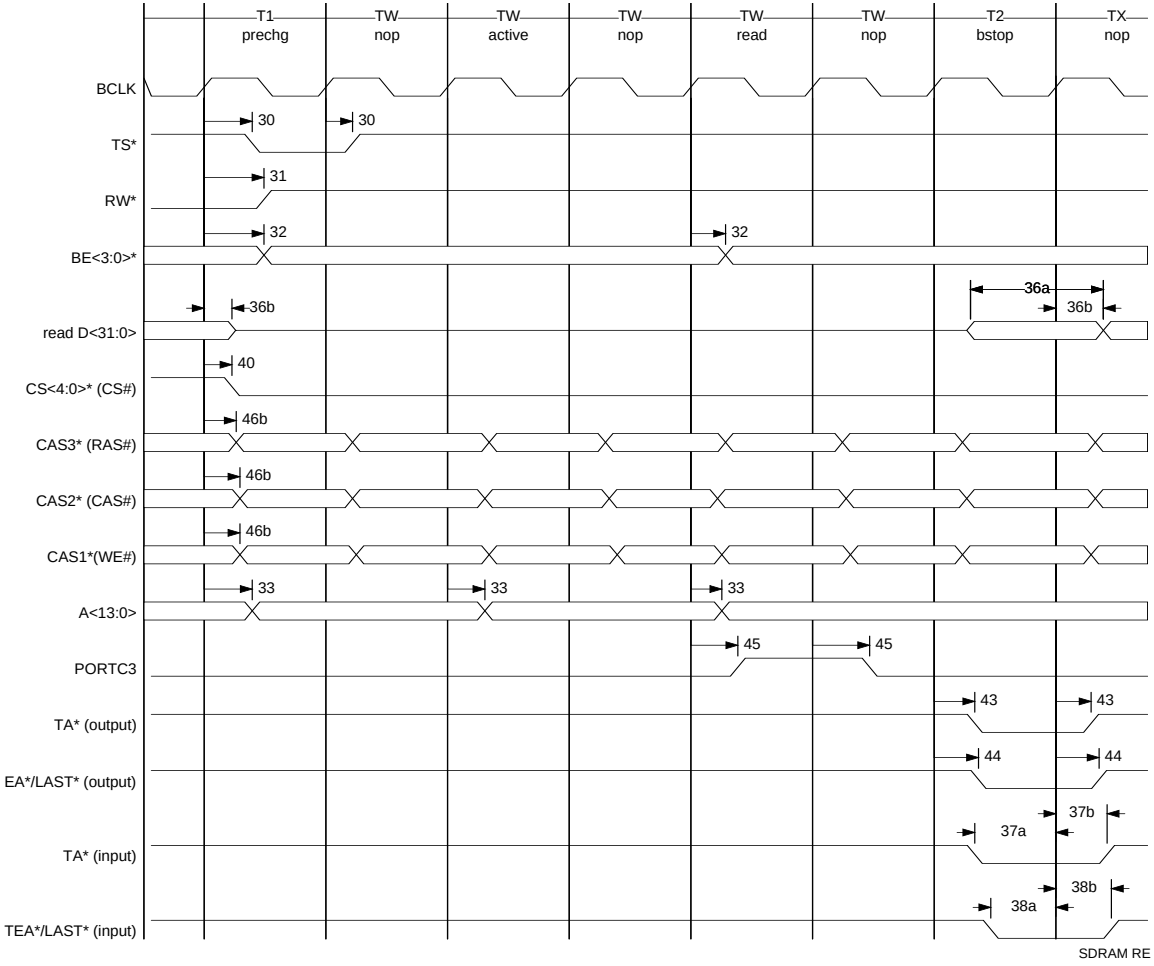


Figure 13-14: SDRAM Read WAIT = 1; BCYC = 1

Refer to Tables 13-9 and 13-10 for values.

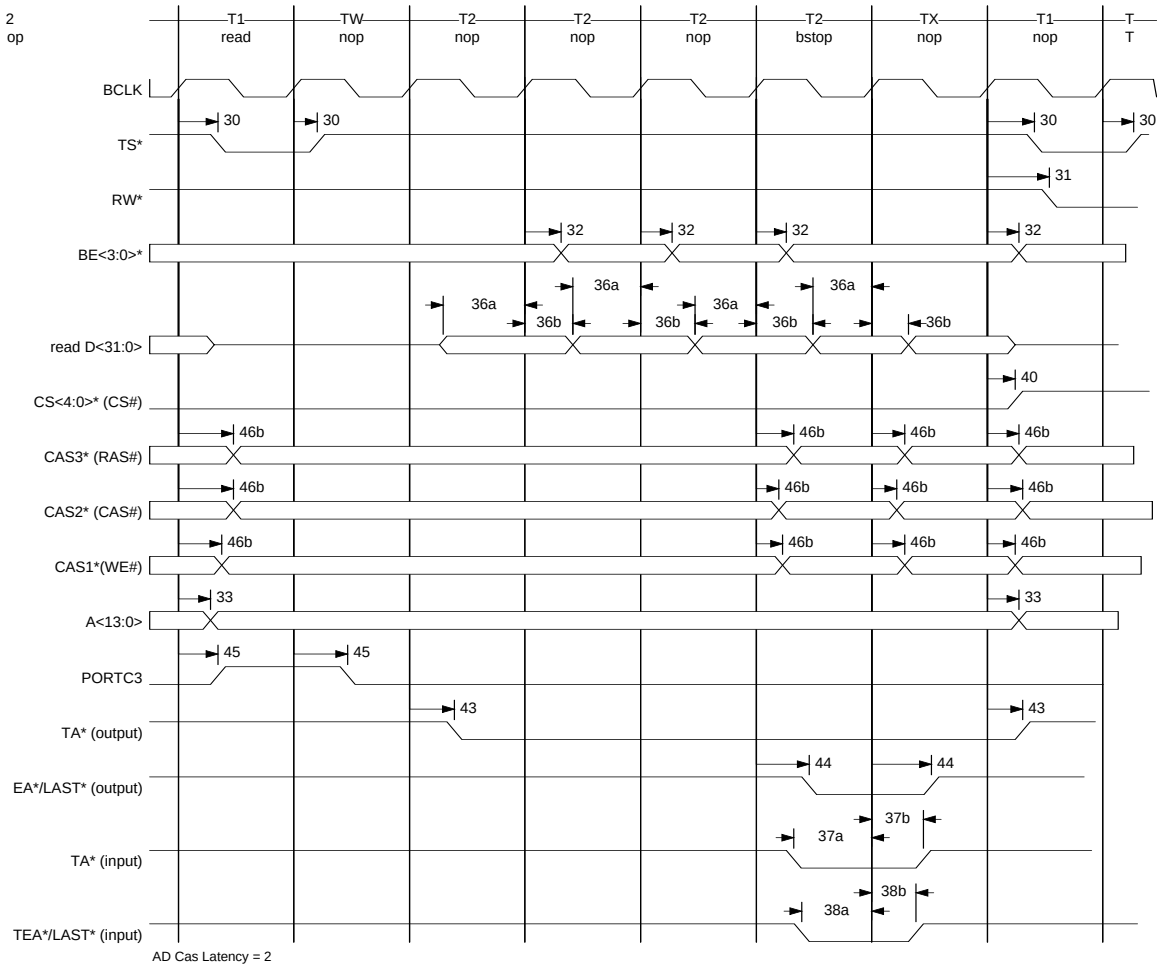


Figure 13-15: SDRAM Read Burst; WAIT = 1; BCYC = 1

Refer to Tables 13-9 and 13-10 for values.

13.3.3 Refresh

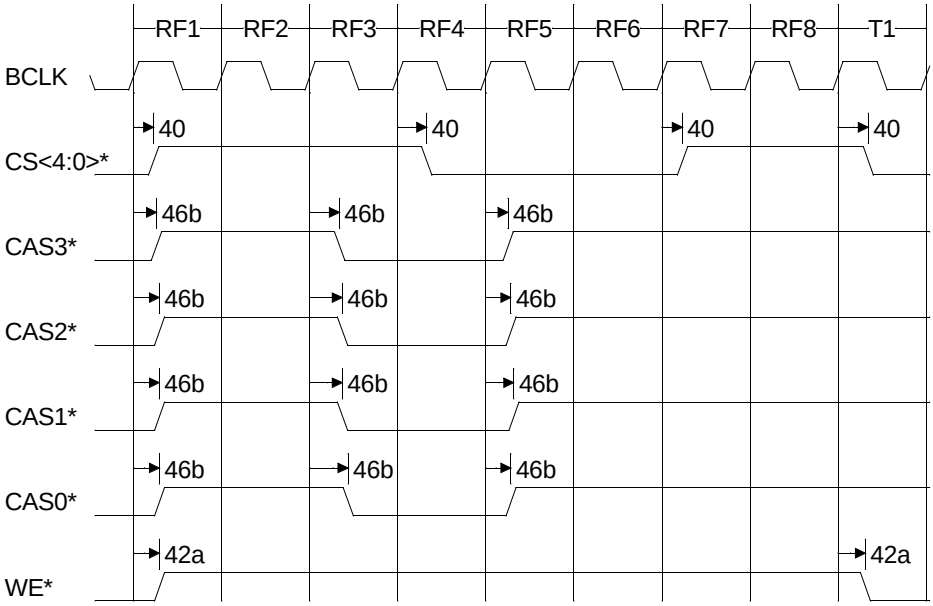


Figure 13-16: DRAM Refresh (RCYC = 0)

Refer to Table 13-10 for values.

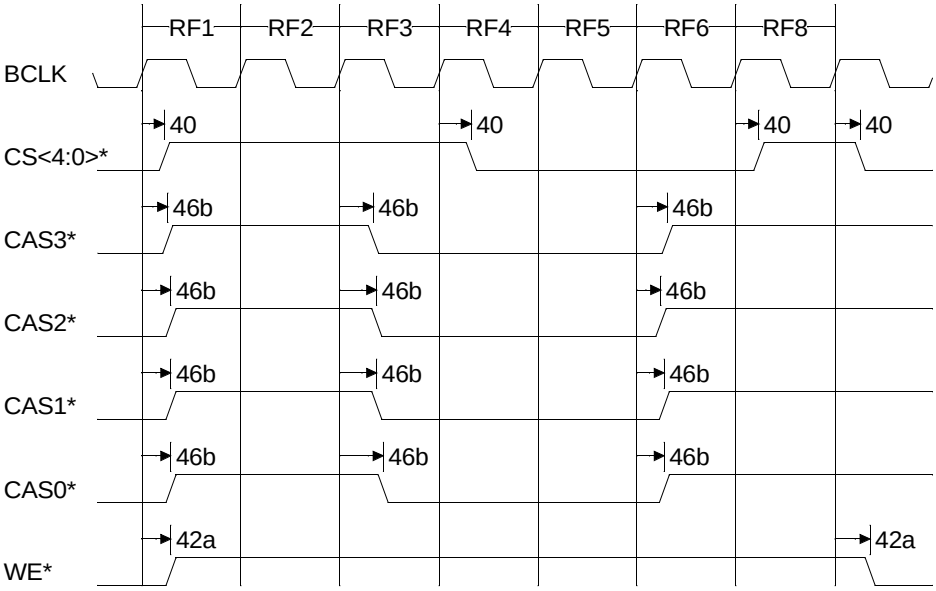


Figure 13-17: DRAM Refresh (RCYC = 1)

Refer to Table 13-10 for values.

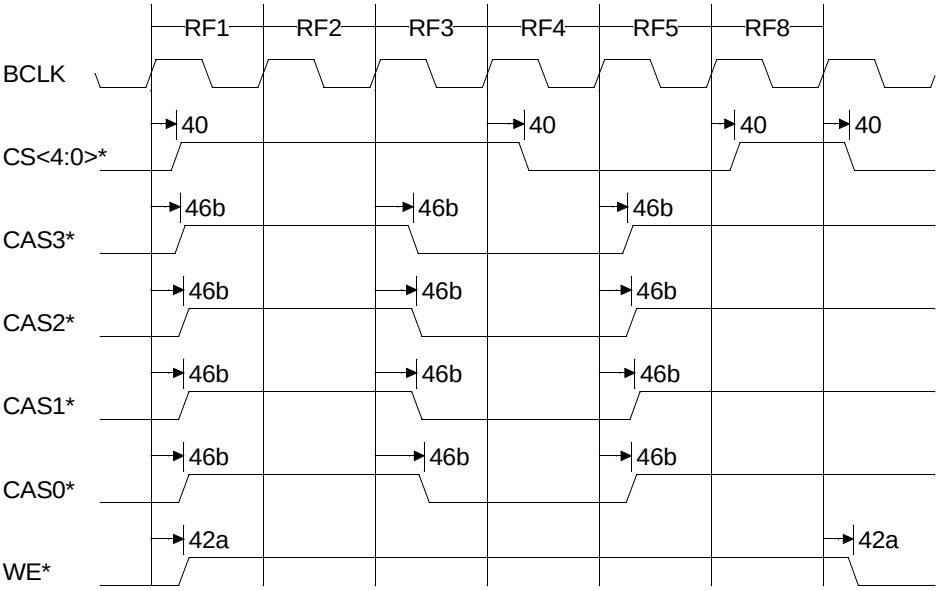


Figure 13-18: DRAM Refresh (RCYC = 2)

Refer to Table 13-10 for values.

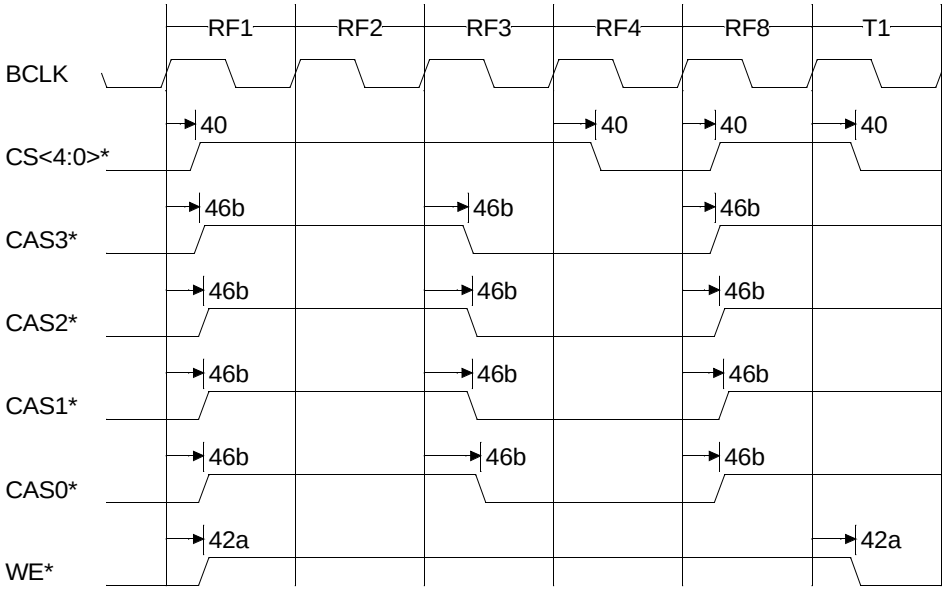


Figure 13-19: DRAM Refresh (RCYC = 3)

Refer to Table 13-10 for values.

Num	Characteristic	Min	Max	Unit
50	BCLK High to DACK* valid		12	ns
51	BCLK High to DONE* (output) valid	0	12	ns
52	DACK* low to DREQ* high (hold)	0		ns
53a	DREQ* valid to BCLK high (setup)	11		ns
53b	BCLK high to DREQ* valid (hold)	0		
54a	DONE* (input) valid to BCLK high (setup)	9		ns
54b	BCLK high to DONE* (input) valid (hold)	0		ns

Table 13-11: External DMA Timing

Num	Characteristic	Min	Max	Unit
60a	RW* valid to BCLK high (setup)	10		ns
60b	BCLK high to RW* invalid (hold)	0		ns
61a	BE* valid to BCLK high (setup)	8		ns
61b	BCLK high to BE* invalid (hold)	0		ns
62a	Address valid to BCLK high (setup)	8		ns
62b	BCLK high to Address invalid (hold)	0		ns
63a	BCLK high to Read Data valid	6	12	ns
63b	BCLK high to Data High Impedance	14	26	ns
64a	Data In valid to BCLK high (setup)	14		ns
64b	BCLK high to Data In invalid (hold)	0		ns
65a	A25/BLAST* valid to BCLK (T2) high (setup)	10		ns
65b	BCLK (T2) high to A25/BLAST* invalid (hold)	0		ns

Table 13-12: External Bus Master Timing

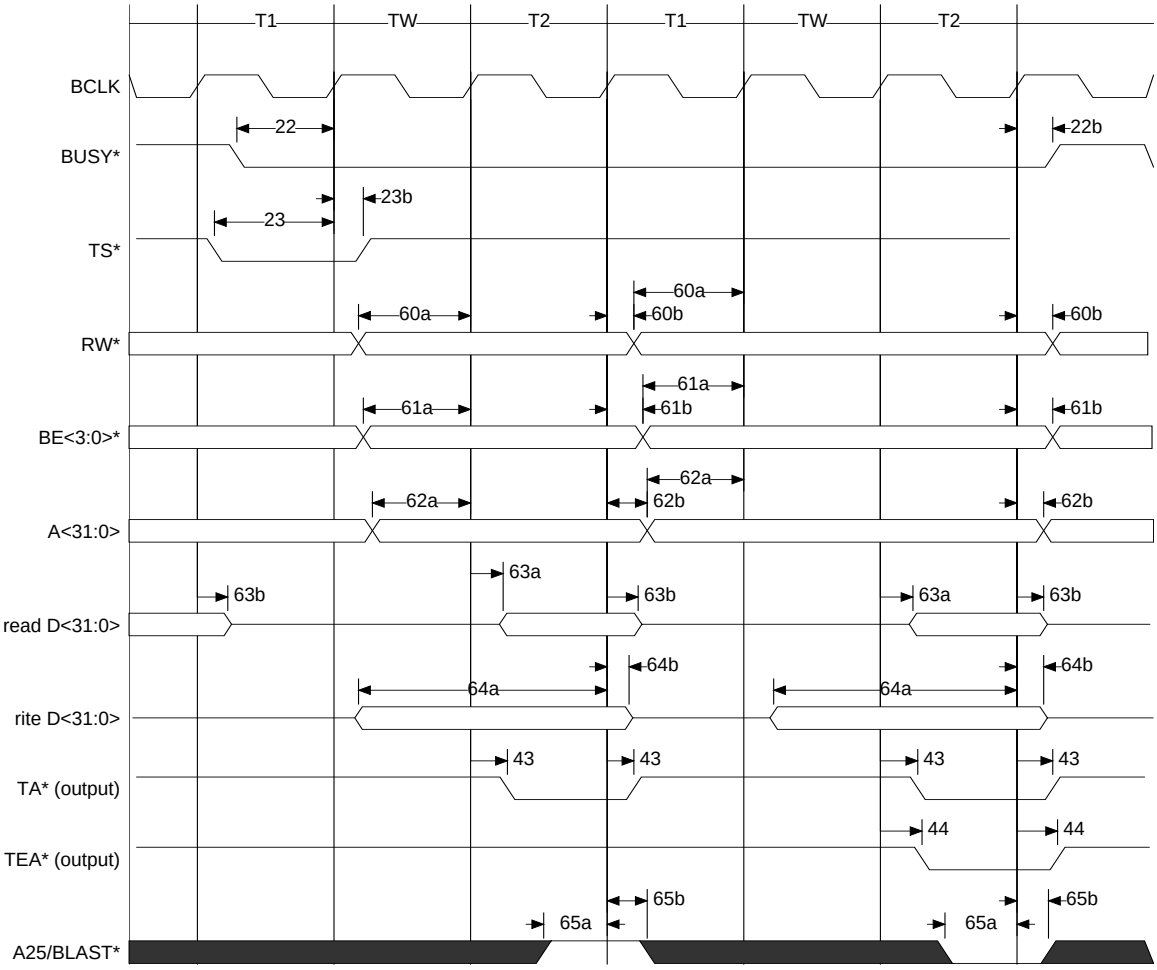


Figure 13-20: External Bus Master Timing

Refer to Tables 13-8 and 13-12 for values.

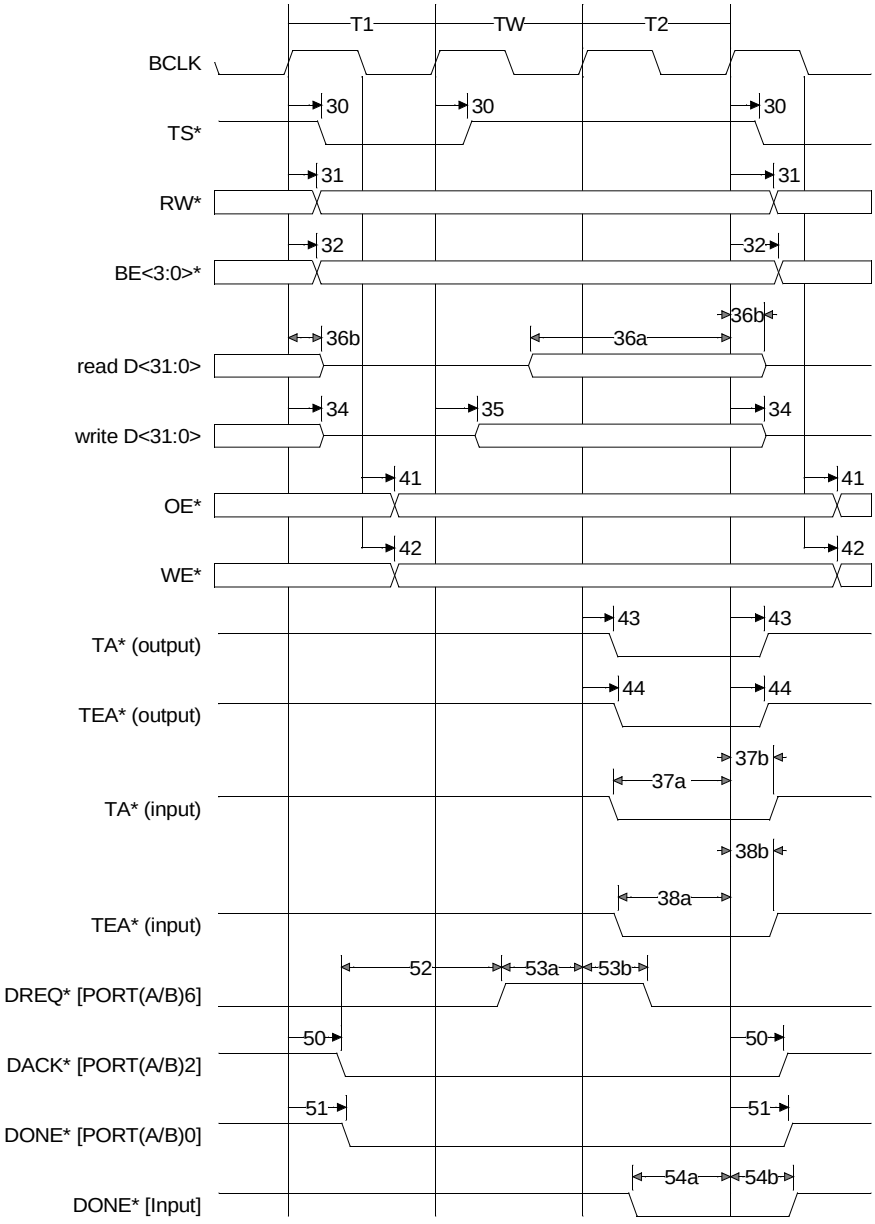


Figure 13-21: External DMA Timing

Refer to Table 13-9 and 13-11 for values.

Num	Characteristic	Note	Min	Max	Unit
70a	PCS*/PDACK* low to PRW* valid	1		2 * Tsys	ns
70b	PCS*/PDACK* low to PRW* invalid	1	4 * Tsys		ns
71	Read Data Valid to PACK* valid			Tsys	ns
72	PCS*/PDACK* low to PACK* valid	3	5 * Tsys	6 * Tsys	ns
72a	PCS* low to PACK* valid	4	8 * Tsys	System Dependent	ns
72b	PCS*/PACK* low to PACK* valid	5	3 * Tsys	4 * Tsys	ns
72c	PCS* low to PACK* valid (Shared RAM only)	6	8 * Tsys	System Dependent	ns
73	PCS*/PDACK* high to PDATA high impedance		0	15	ns
74	PCS*/PDACK* high to PACK* inactive		0	10	ns
75	PCS*/PDACK* low to PACK* (wait-) low		0	10	ns
76a	PCS*/PDACK* low to Write Data valid			2 * Tsys	ns
76b	PCS*/PDACK* high to Write Data invalid (hold)		0		ns
77a	PCS*/PDACK* width high (recovery)		10		ns
77b	PACK* low to PCS*/PDACK* high (hold)	8	0		ns
77c	PDACK* minimum low		120		ns
78a	Address valid to PCS* low		0		ns
78b	PCS* high to Address invalid		0		ns
79	PCS* low to PINT1/2 change (write)		4 * Tsys	5 * Tsys	ns
80a	PDACK* low to PDRQI*, PDRQO* high			5 * Tsys	ns
80b	PDRQO* high width		5 * Tsys		ns
80c	PDRQI* high width		4 * Tsys		ns
80d	PDACK* low to PINT2 low			30	ns
80e	PDACK* high to PINT2 high		30		ns
81	PCS*/PDACK* low to PEN* low	2	2 * Tsys	3 * Tsys	ns
82	PCS*/PDACK* low to PEN* high	2	0	10	ns
83	PCS*/PDACK* low to PBRW* valid	2	2 * Tsys	2 * Tsys	ns
84a	PRW* valid to PDACK* low (setup)	1	0		ns
84b	PDACK* high to PRW* invalid (hold)	1	0		ns
85	PDACK* low to PDATA valid	7	4 * Tsys	5 * Tsys	ns
85a	PDACK* low to PDATA valid	7	1 * Tsys	2 * Tsys	ns

Table 13-13: ENI Shared RAM & Register Timing

See notes on following page.

Note	Description
1	Parameters 70a/70b only apply when the ENI FAST bit is set to 0. When ENI FAST is set to 1, then parameters 84a/84b apply.
2	The PEN* and PBRW* signals are used to control an external bi-directional data bus transceiver for the PDATA bus that can only drive 2mA.
3	Specification 72 only applies to ENI Registers while FAST is set to 0. This specification does NOT apply to Shared RAM access.
4	Specification 72a applies to all Shared RAM accesses when FAST is set to 0. The Max specification for PCS* to PACK* valid is larger for shared RAM accesses. The additional delay is dependent upon the speed of the external RAM assigned to provide the physical shared RAM. As such, the Max specification for shared RAM accesses is system dependent.
5	Specification 72b applies to ENI register accesses when FAST is set to 1.
6	Specification 72c applies to ENI shared RAM accesses when FAST is set to 1.
7	Specification 85 applies when FAST is set to 0. Specification 85a applies when FAST is set to 1.
8	Specification 77c can be reduced to (3 *Tsys) when FAST is set to 1.

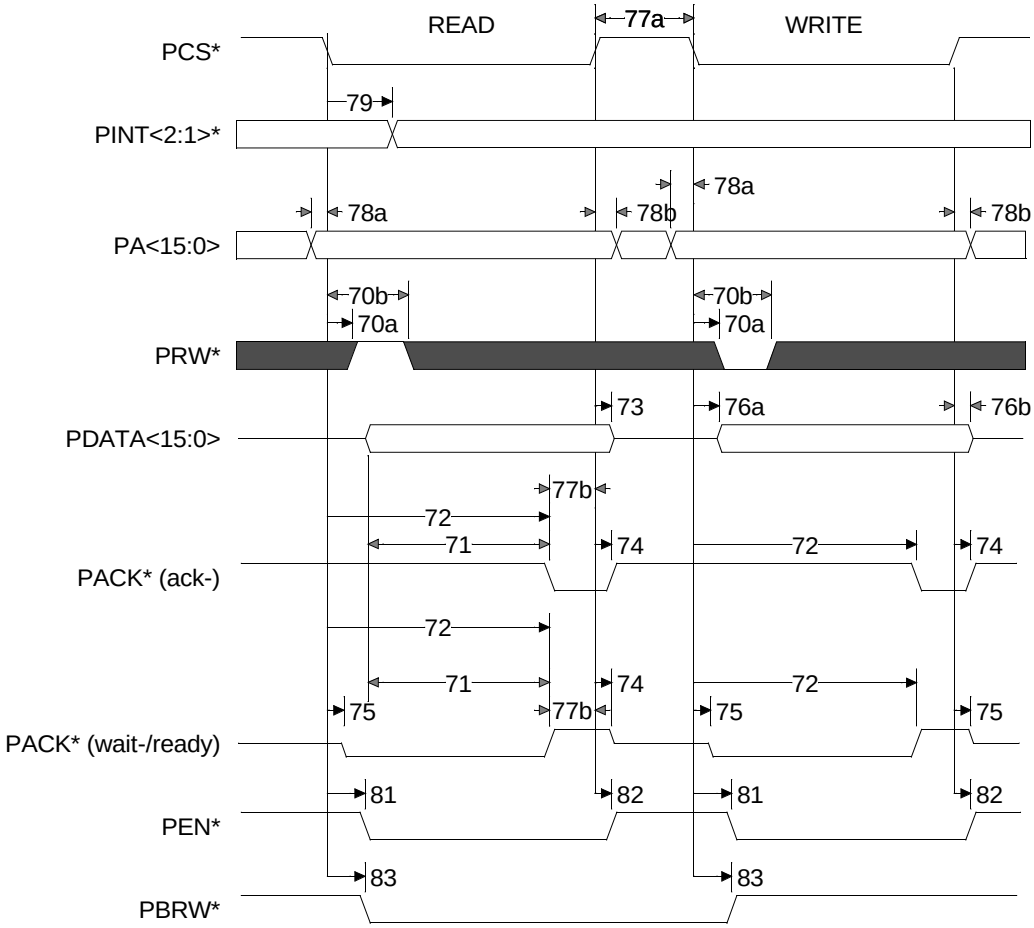


Figure 13-22: ENI Shared RAM & Register Cycle Timing
Refer to Table 13-13 for values.

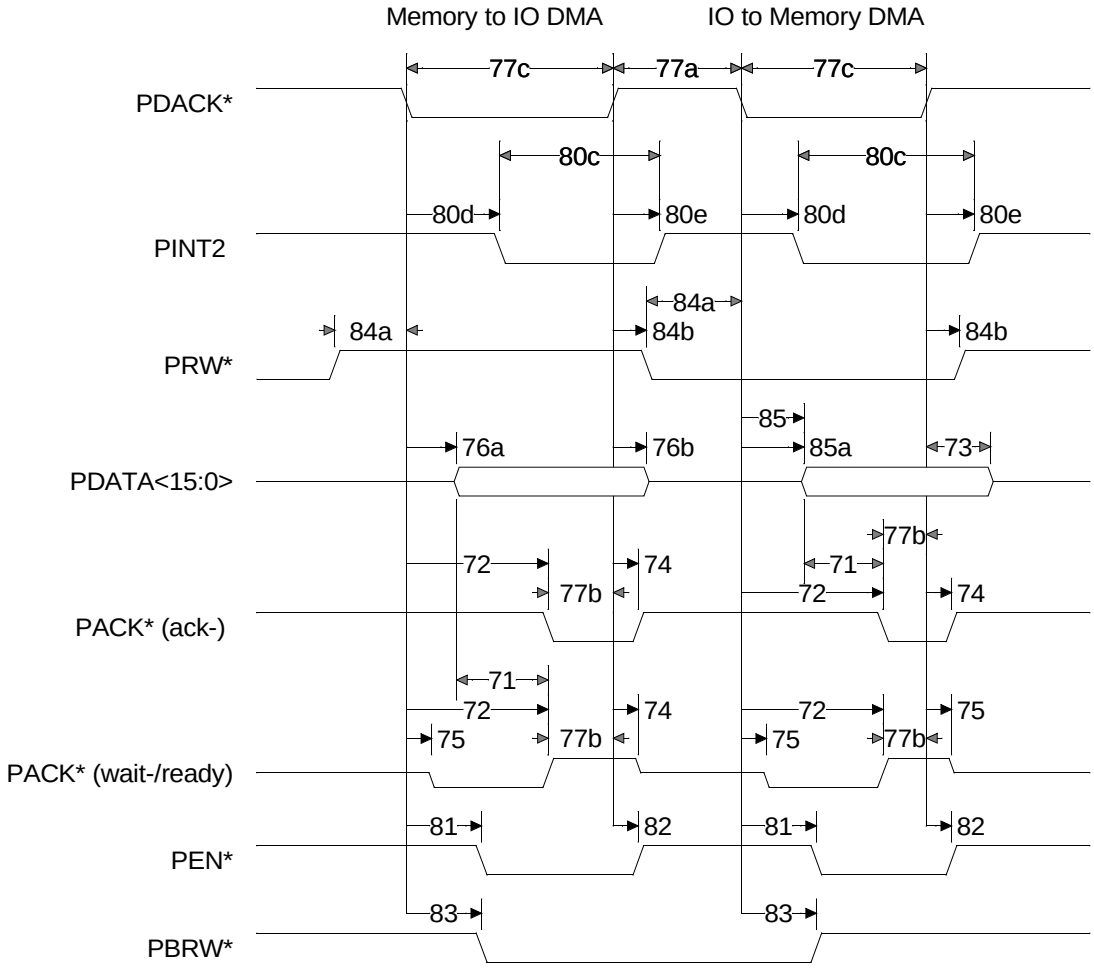


Figure 13-23: ENI Single Direction DMA Timing
Refer to Table 13-13 for values.

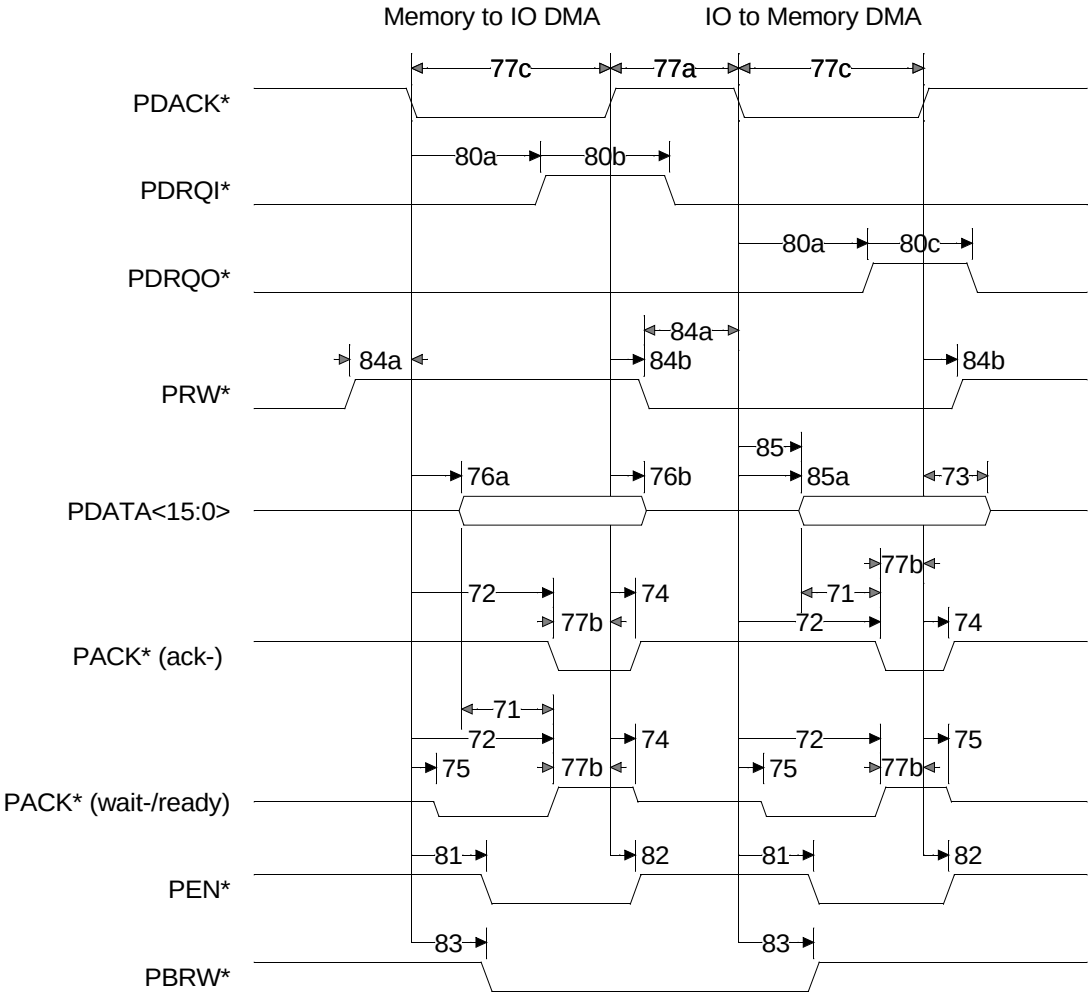


Figure 13-24: ENI Dual Direction DMA Timing
Refer to Table 13-13 for values.

Num	Characteristic	Min	Max	Unit
90	TXCLK high to TXD, TXDV, TXER valid		20	ns
91a	RXD, RXER, RXDV valid to RXCLK high (setup)	10		ns
91b	RXCLK high to RXD, RXER, RXDV invalid (hold)	0		ns
92	MDC high to MDIO change	40	50	ns
93a	MDIO valid to MDC high (setup)	10		ns
93b	MDC high to MDIO invalid (hold)	0		ns
94	RXCLK high to RPSF* change		20	ns
95a	REJECT* valid to RXCLK high (setup)	10		ns
95b	REJECT* valid from RXCLK high (hold)	0		ns
96	CRS low to RXCLK idle (bit-times)	27		Bit Times

Table 13-14: Ethernet Timing

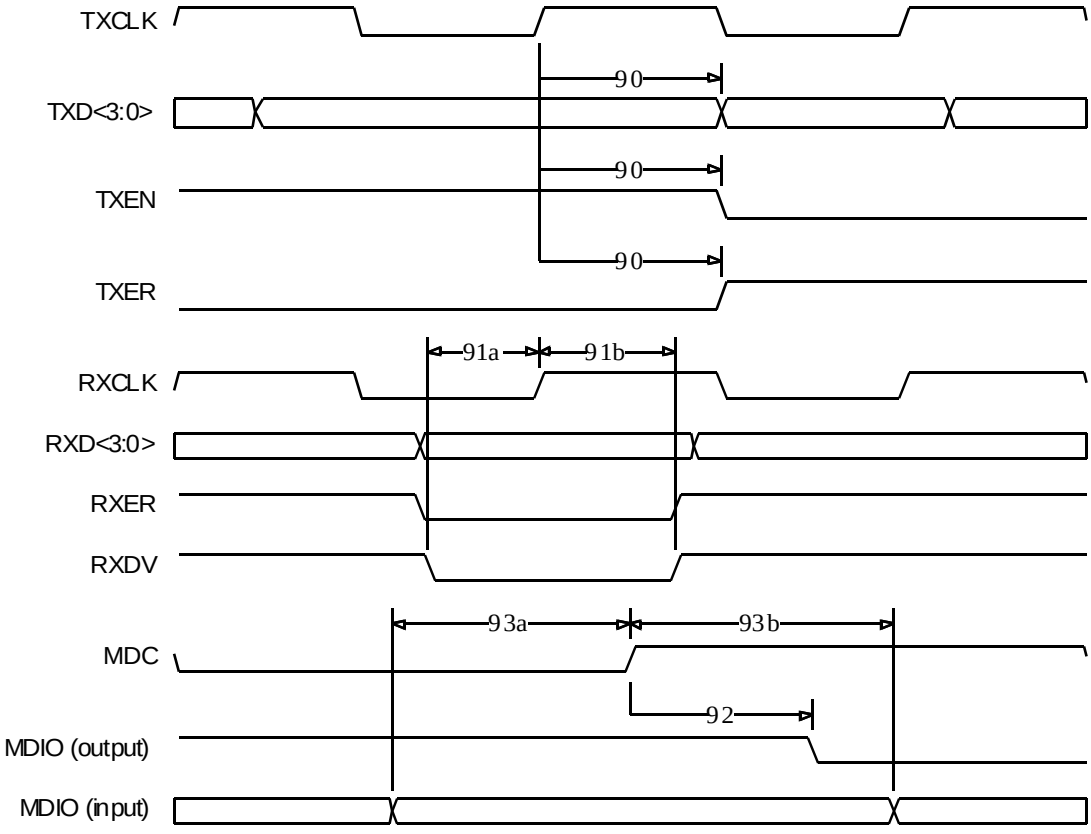


Figure 13-25: Ethernet Timing
Refer to Table 13-14 for values.

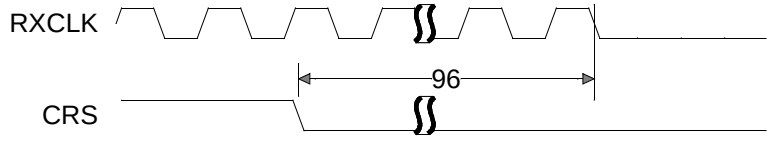


Figure 13-26: SDRAM
Refer to Table 13-14 for values.

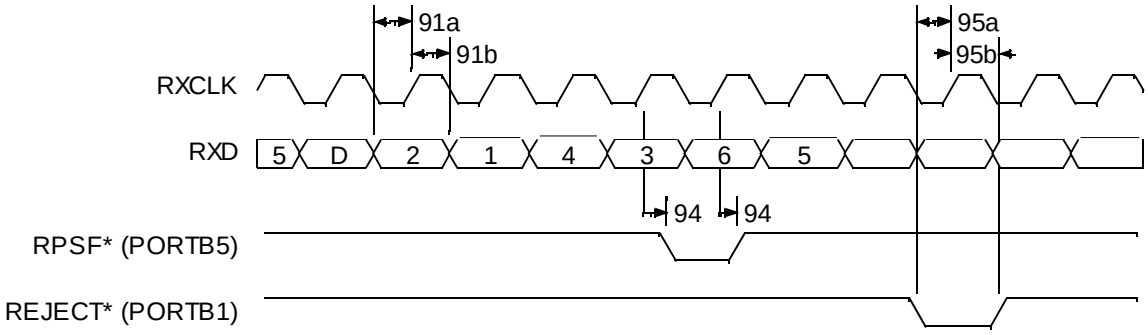


Figure 13-27: External Ethernet CAM Filtering

Refer to Table 13-14 for values.

Num	Characteristic	Min	Max	Unit
100	TCK low to TDO valid	0	50	ns
101	TCLK low to TDO high impedance	0	20	ns
102a	TDI, TMS valid to TCK high (setup)	50		ns
102b	TCK high to TDI, TMS invalid (hold)	50		ns
103	TCK Cycle Time	100		ns
104	TCK pulse width	40		ns
105	TRST* low time	100		ns
106	TRST* valid to TCK low (setup)	40		ns

Table 13-15: JTAG Timing

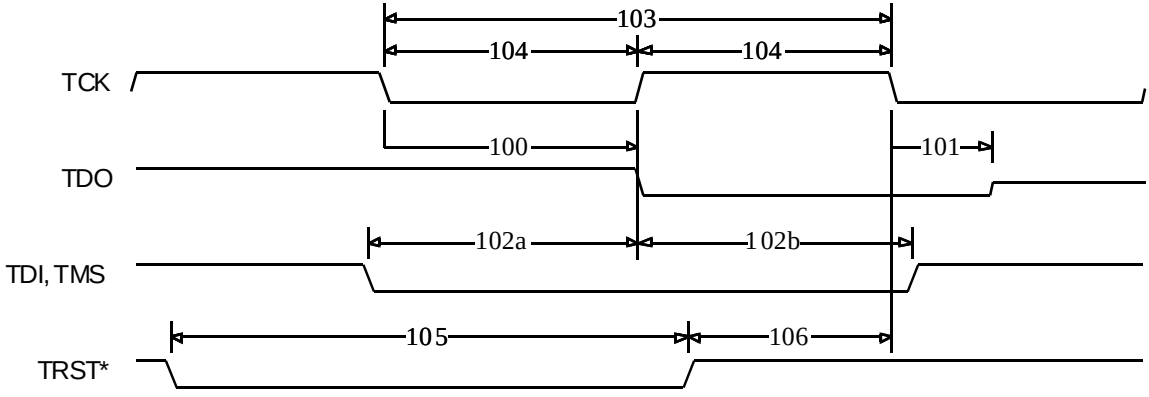


Figure 13-28: JTAG Timing
Refer to Table 13-15 for values.

Num	Characteristic	Min	Max	Unit
110	PDATA valid to PCLKDx or PCLKCx high (setup)	T_{sys}		ns
111	PCLKDx and PCLKCx time high (width)	T_{sys}		ns
112	PCLKDx and PCLKCx low to PDATA valid (hold)	T_{sys}		ns
113	PDATA high impedance to POEx* active low (read only)	T_{sys}		ns
114	POEx* minimum low time (read only)	T_{sys}		
115	POEX* high to PDATA driven (read only)	T_{sys}		ns

Table 13-16: 1284 Port Multiplexing Timing

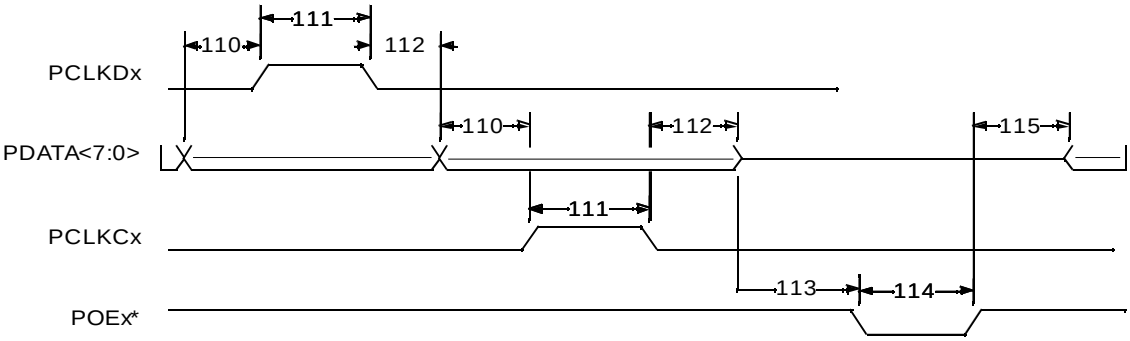


Figure 13-29: 1284 Multiplexing Timing
Refer to Table 13-16 for values.

Num	Characteristic	Min	Max	Unit
115	DATA valid to STROBE* low (setup)	STROBE		ns
116	STROBE* width low	STROBE		ns
117	STROBE* low to BUSY (input) high	0		ns
119	BUSY low to DATA change (hold)	3 * T_{sys}		ns
120	STROBE* high to DATA change (hold)	STROBE		ns

Table 13-17: 1284 Compatibility SLOW Mode Timing (FAST = 0)

Num	Characteristic	Min	Max	Unit
115	DATA valid to STROBE* low (setup)	STROBE		ns
116	STROBE* width low	STROBE		ns
117	STROBE* low to BUSY (input) high	0		ns
119	BUSY low to DATA change (hold)	3 * T_{sys}		ns
120	STROBE* high to DATA change (hold)	3 * T_{sys}		ns

Table 13-18: 1284 Compatibility FAST Mode Timing (FAST = 1)

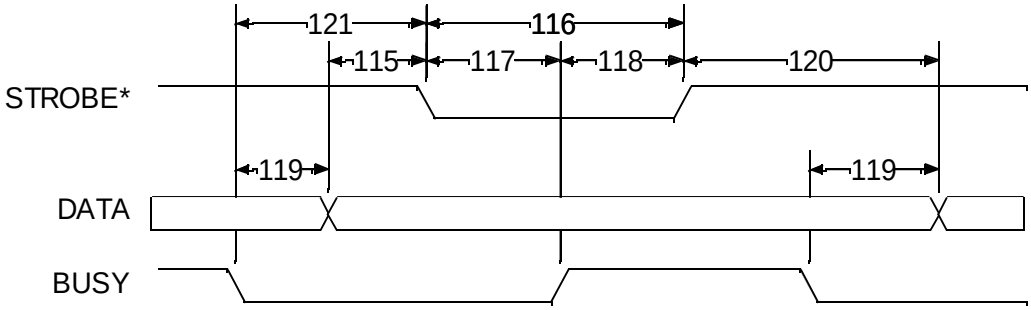


Figure 13-30: 1284 Compatibility Mode Timing

Refer to Tables 13-17 and 13-18 for values.

Num	Characteristic	Min	Max	Unit
115	DATA valid to STROBE* low (setup)	STROBE		ns
117	STROBE* low to BUSY (input) high	STROBE		ns
118	BUSY high to STROBE* high (hold)	3 * T_{sys}		ns
120	STROBE* high to DATA change (hold)	STROBE		ns
121	BUSY low to STROBE* (low)	3 * T_{sys}		ns

Table 13-19: 1284 SLOW Forward ECP Mode Timing (FAST = 0)

Num	Characteristic	Min	Max	Unit
115	DATA valid to STROBE* low (setup)	3 * T_{sys}		ns
117	STROBE* low to BUSY (input) high	0		ns
118	BUSY high to STROBE* high (hold)	3 * T_{sys}		ns
120	STROBE* high to DATA change (hold)	3 * T_{sys}		ns
121	BUSY low to STROBE* (low)	3 * T_{sys}		ns

Table 13-20: 1284 FAST Forward ECP Mode Timing (FAST = 1)

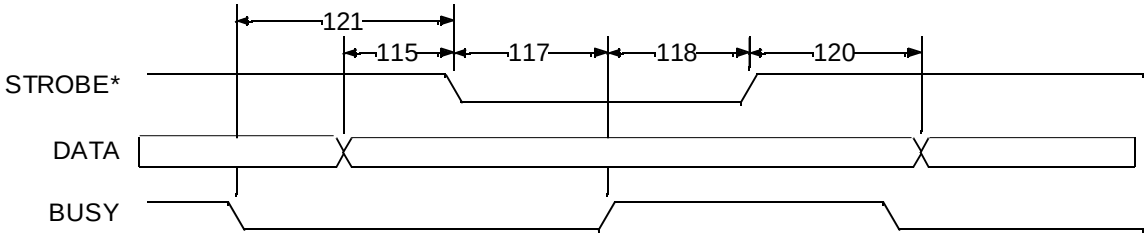


Figure 13-31: 1284 Forward ECP Mode Timing

Num	Characteristic	Min	Max	Unit
130	SPI Clock Low to SPI Enable Low	$-T_{SYS}$	T_{SYS}	ns
131	SPI Clock low to TXD valid	$-T_{SYS}$	T_{SYS}	ns
132	RXD Input Valid to SPI Clock High (setup)	20		ns
133	SPI Clock High to RXD Input Change (hold)	0		ns
134	SPI Clock High to SPI Enable High	½ Bit-Time		ns

Table 13-21: SPI Master Timing

T_{SYS} refers to the period of the NET+ARM chip system clock. If the NET+ARM chip is operating at 33Mhz, then T_{SYS} is equal to 30ns. ½ bit-time is a function of how fast the SPI port is configured to operate at. If the SPI port is configured to operate at its maximum speed of 4Mbps, then ½ bit-time would be equal to 120ns.

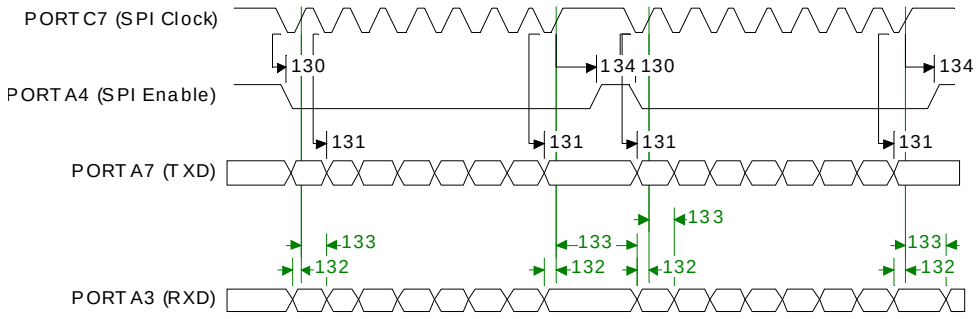


Figure 13-32: SPI Master Mode Waveform

Refer to Table 13-21 for values.

Num	Characteristic	Min	Max	Unit
141	SPI Clock High to TXD valid	3 * T_{SYS}	4 * T_{SYS}	ns
142	SPI Enable low to SPI Clock High (setup)	0		ns
143	RXD Input Valid to SPI Clock High (setup)	0		ns
144	SPI Clock High to RXD Input Change (hold)		4 * T_{SYS}	ns
145	SPI Clock High to SPI Enable High (hold)	4 * T_{SYS}		ns

Table 13-22: SPI Slave Timing

T_{SYS} refers to the period of the NET+ARM chip system clock. If the NET+ARM chip is operating at 33Mhz, then T_{SYS} is equal to 30ns. ½ bit-time is a function of how fast the SPI port is configured to operate at. If the SPI port is configured to operate at it's maximum speed of 4Mbps, then ½ bit-time is equal to 120ns

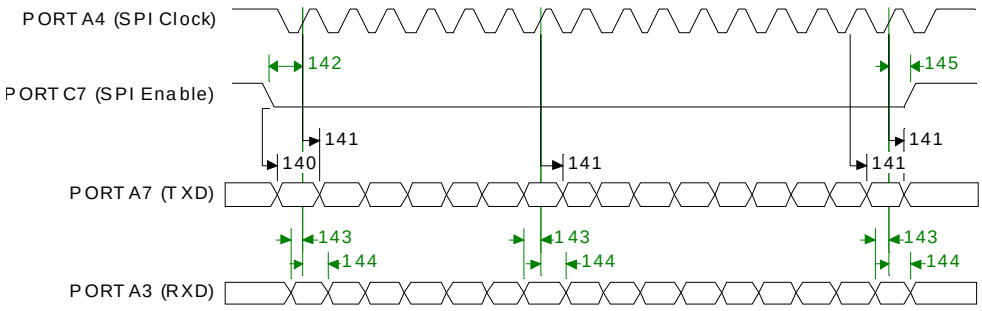


Figure 13-33: SPI Slave Mode Waveform

Refer to Table 13-22 for values.

13.4 Output Rise and Fall Timing

The rise and fall times for each of the NET+ARM chip output pins is a function of the pad cell type and the load capacitance. The pad cell type for the various NET+ARM chip outputs can be found in Table 1-1: *NET+ARM Chip Pinout*.

The rise and fall times equations are a function of the intrinsic delay time and the load delay time. The load delay time is a function of the output capacitance load attached to the NET+ARM chip output pin. The rise and fall times are calculated using the following equation:

T_{RISE} / T_{FALL} = Intrinsic Time + (Load Delay * C_L)

C_L is defined as the output load capacitance attached to the NET+ARM chip output pin.

Pad Cell	Rise Time		Fall Time	
	Intrinsic Delay (ns)	Load Delay (ns / pF)	Intrinsic Delay (ns)	Load Delay (ns / pF)
pt3b01	1.864	0.097	1.389	0.106
pt3b01u	1.840	0.095	1.397	0.107
pt3b02	1.785	0.049	1.259	0.053
pt3b02u	1.774	0.048	1.262	0.053
pt3b03u	2.103	0.026	1.450	0.027
pt3t01	1.861	0.097	1.391	0.106
pt3t02	1.784	0.049	1.259	0.053
pt3t03	2.111	0.026	1.454	0.027

Table 13-23: Output Rise/Fall Times

13.5 Crystal Oscillator Specifications

Quartz crystals can be chosen from a wide range of manufacturers and it is difficult to specify only one rule. The crystals' characteristics also vary significantly depending on their resonant frequency. The choice must always be parallel resonance for the NET+ARM chip with a load capacitance CL of 8pF. If the crystal recommended load capacitance is greater, then additional capacitors must be added external to XTAL1 and XTAL2.

The crystal oscillator is a low power oscillator so no external capacitors or resistors should be used to reduce the current driven in the crystal. If however, the crystal used has a drive level smaller than the drive level specified then a series resistor can be added between XTAL2 and the crystal connection.

Instead of using the XTAL1 / XTAL2 crystal oscillator circuit, the NET+ARM chip can be driven using an external TTL oscillator. The TTL oscillator provides a single TTL clock input on the XTAL1 pin. The XTAL2 pin is left unconnected in this configuration. To use this configuration, the PLLTST* pin must be driven low. Also note that when this configuration is used, the internal PLL is bypassed and disabled.

Sym	Parameter	Conditions	Min	Typ	Max	Unit
f0	Operating Frequency	16 Mhz		16		Mhz
Tsu	Startup Time				2	ms
C1	Internal Capacitance	Xtal1 / Gnd	14.8	16	17.1	pF
C2	Internal Capacitance	Xtal2 / Gnd	14.8	16	17.1	pF
CL	Local Capacitance	Xtal1 / Xtal2	7.4	8	8.6	pF
DL	Drive Level		50	100	150	uW
Rs	Series Resistance	Crystal			100	Ohm
Cs	Shunt Capacitance	Crystal				pF

Table 13-24: Crystal Specification

Chapter 14
NET+ARM Chip Package Guide

Figure 14-1 shows the chip package for the NET+ARM chip.

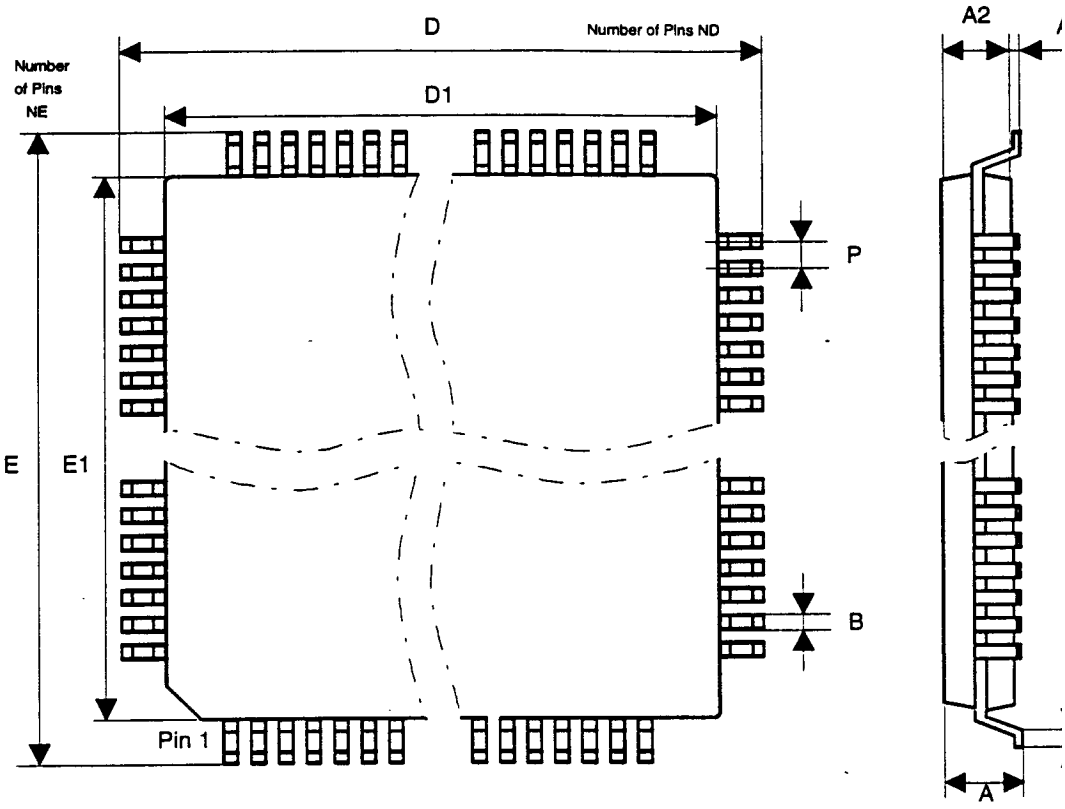


Figure 14-1: Short Plastic Quad Flat Pack

The table below specifies the dimensions for the NET+ARM chip.

208 Lead - All Dimensions are in millimeters			
	Minimum	**Nominal**	**Maximum**
A	3.22	3.57	3.97
A1	0.05	0.25	0.50
A2	3.17	3.32	3.47
D	30.95	31.20	31.45
D1	27.90	28.00	28.10
E	30.95	31.20	31.45
E1	27.90	28.00	28.10
L	0.35	0.50	0.65
P		0.50	
B	0.10	0.20	0.30

Table 14-1: NET+ARM Chip Dimensions

Index

Numerics

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T