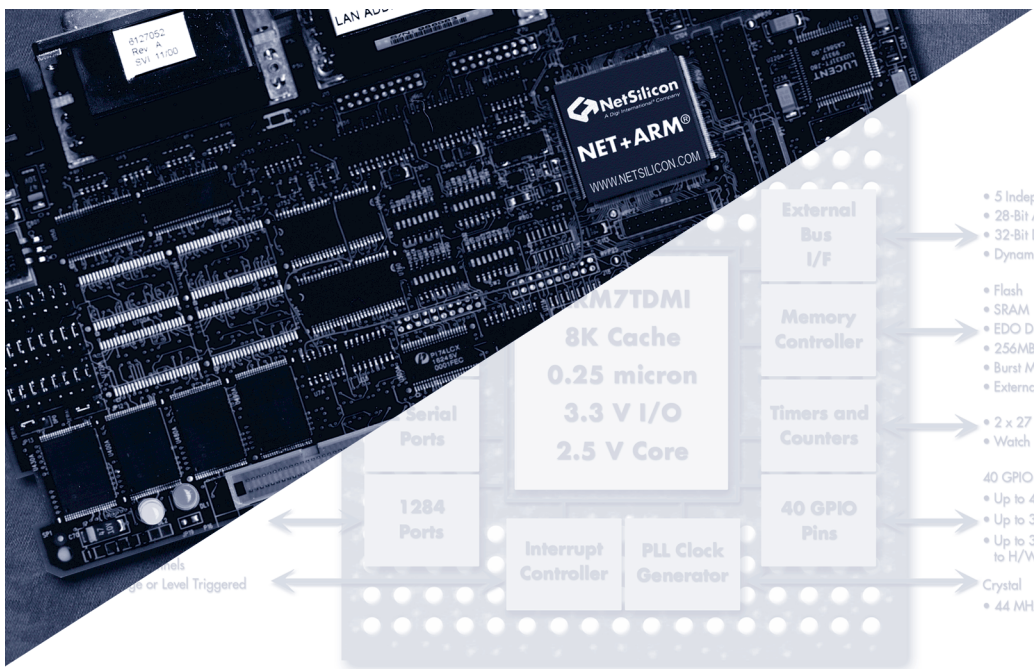
NetSilicon
A Digi International® Company

NET+50/20M Hardware Reference



8833450A

NET+50/20M Hardware Reference

Operating system/version: NET+OS 5.0
Part number/version: 8833450A
Release date: September 2002
www.netsilicon.com

Contents

Tables

Figures

Using This Guide

Review this section for basic information about the guide you are using, as well as general support and contact information.

About this guide

This guide provides information about the NetSilicon NET+50 and NET+20M 32-bit networked microprocessors. The NET+50 and NET+20M are part of the NetSilicon NET+ARM line of SoC (System-on-Chip) products, and support any type of high-bandwidth applications in Intelligent Networked Devices.

The NET+ARM chip is part of the NET+Works integrated product family, which includes the NET+OS network software suite.

Who should read this guide

This guide is for hardware developers, system software developers, and applications programmers who want to use the NET+50 or NET+20M for development.

To complete the tasks described in this guide, you must:

- Understand the basics of hardware and software design, operating systems, and microprocessor design.
- Understand the NET+50/20M architecture.

What's in this guide

This table shows where you can find specific information in this guide:

To read about	See
Similarities and differences between the NET+50 chip and the 20M chip	Chapter 1, "About the NET+50 chip and NET+20M chip"
NET+50 chip and NET+20M chip key features	Chapter 2, "NET+50/20M Chip Features"
NET+50 PQFP/BGA pin/ball grid array assignments and packaging	Chapter 3, "NET+50 Chip Package"
NET+50/20M CPU and the ARM *Thumb* concept	Chapter 4, "Working with the CPU"
BBus functionality	Chapter 5, "BBus Module"
General (GEN) module functionality	Chapter 6, "The GEN Module"
ARM7TDMI stand-alone core and instruction/data cache, write protection, and pre-fetch control **Note:** This information applies to the NET+50 chip only.	Chapter 7, "Cache"
How the NET+50/20M can be configured to interface with different types of memory devices	Chapter 8, "Memory Controller Module"
DMA controller, supported DMA channels, and internal and externalDMA transfers	Chapter 9, "DMA Controller Module"
Ethernet controller module	Chapter 10, "Ethernet Controller Module"
Serial channel A and serial channel B	Chapter 11, "Serial Controller Module"
Multi interface controller (MIC) **Note:** This information applies to the NET+50 chip only	Chapter 12, "MIC Controller Module"
NET+50/20M timing information and diagrams	Chapter 13, "Timing"
ARM exceptions	Appendix A, "ARM Exceptions"
NET+20M BGA (ball grid array) assignments and packaging	Appendix B, "NET+20M Chip Package"

Conventions used in this guide

This table describes the typographic conventions used in this guide:

This convention	Is used for
italic type	Emphasis, new terms, variables, and document titles.
`monospaced type`	Filenames, pathnames, and code examples.

Related documentation

- *NET+50/20M Jumpers and Components* provides a hardware description of the NET+Works Development Board, and includes information about jumpers, connectors, switches, and interface configuration as well as development board diagrams.
- Review the documentation CD-ROM that came with your development kit for information on third-party products and other components.
- Refer to the NET+OS software documentation for information appropriate to the chip you are using.

Customer support

To get help with a question or technical problem with this product, or to make comments and recommendations about our products or documentation, use the contact information listed in this table:

For	Contact information
Technical support	Telephone: 1 800 243-2333/ 1 781 647-1234 Fax: 1 781 893-1388 Email: tech_support@netsilicon.com
Documentation	techpubs@netsilicon.com

For	Contact information
NetSilicon home page	www.netsilicon.com
Online problem reporting	www.netsilicon.com/EmbWeb/Support/forms/bugreport.asp An engineer will analyze the information you provide and call you about the problem.

About the NET+50 chip and NET+20M chip

CHAPTER 1

NetSilicon offers two versions of its system-on-a-chip ASIC that is designed for use in intelligent network devices and Internet appliances:

- **NET+50 chip.** Consists of a 208 plastic-quad-flat-pack (PQFP) package and a 208 ball-grid-array (BGA) package. The NET+50 is a high-performance, highly-integrated 32-bit chip.
- **NET+20M chip.** Consists of a 208 BGA package only. The NET+20M is a cost-effective, highly-integrated 32-bit chip.

Introduction

The NET+50 and NET+20M chips support almost any networking scenario, with a 10/100 BaseT Ethernet MAC with MII interface and two independent serial ports, each of which can run in UART, HDLC, or SPI modes. The CPU is an ARM7TDMI 32-bit RISC processor core with a rich complement of support peripherals, including memory controllers for various types of memory (including flash, SDRAM, and EEPROM), programmable timers, and a 10-channel DMA controller.

The NET+50 and NET+20M provide all tools required for any embedded networking application.

ARM7TDMI

The heart of the NET+50/NET+20M hardware is provided by the ARM7TDMI. The ARM7TDMI is a member of the ARM Ltd. family of general purpose 32-bit microprocessors, which offer high-performance while maintaining very low power-consumption and size.

The ARM architecture is based on Reduced Instruction Set Computer (RISC) principles. Compared with Complex Instruction Set Computers (CISC) architecture, the RISC instruction set, and related decoding and execution mechanisms, is simpler and more streamlined. This simplicity results in a high instruction throughput and impressive real-time interrupt response, as well as a small, cost-effective circuit. The RISC architecture is conducive to pipelining, which allows the instruction fetch, decode, and execution units to operate simultaneously.

NET+50 chip

The NET+50 chip can be used in any embedded environment requiring networking services in an Ethernet LAN. The NET+50 contains an ARM RISC processor, 10/100 Ethernet MAC, serial ports, IEEE 1284 parallel ports, memory

controllers, and parallel I/O. The NET+50 chip can interface with another processor using a RAM or shared RAM interface.

Hardware design

The NET+50 chip can attach to another processor using the 1284, ENI shared RAM, or ENI FIFO interfaces. Application-specific hardware can be attached to the NET+50 system bus for custom applications that use the NET+50 internal RISC processor.

Device modules

Figure 1 is an overview of the modules that make up the NET+50 device.

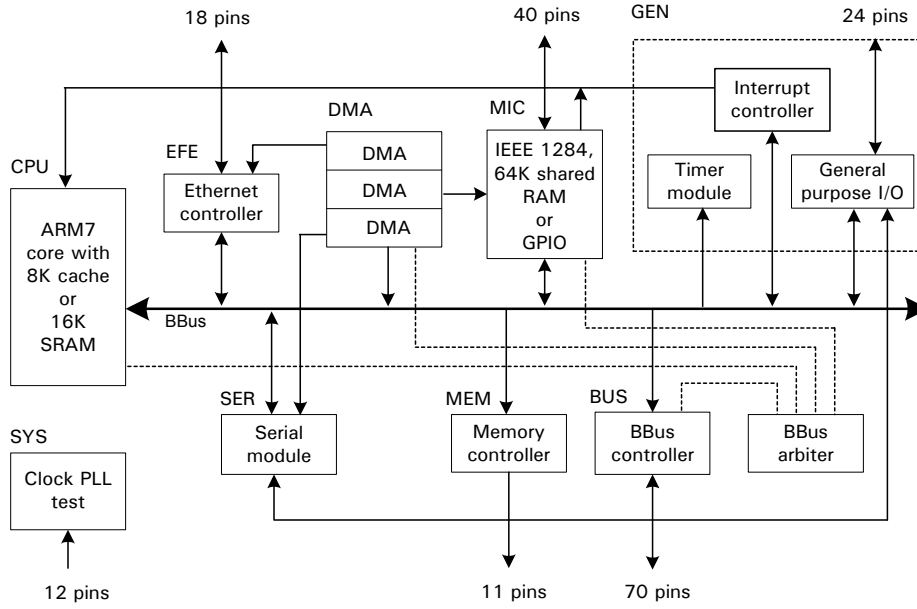


Figure 1: NET+50 device modules

NET+20M chip

The NET+20M chip can be used in any embedded environment requiring networking services in an Ethernet LAN. The NET+20M chip contains an integrated ARM RISC processor, 10/100 Ethernet MAC, serial ports, memory controllers, and parallel I/O. The NET+20M chip can interface with another processor using a register or shared RAM interface.

Hardware design

Application-specific hardware can be attached to the NET+50 system bus for custom applications that use the NET+50 internal RISC processor.

Device modules

Figure 2 provides an overview of the modules that make up the NET+20 device.

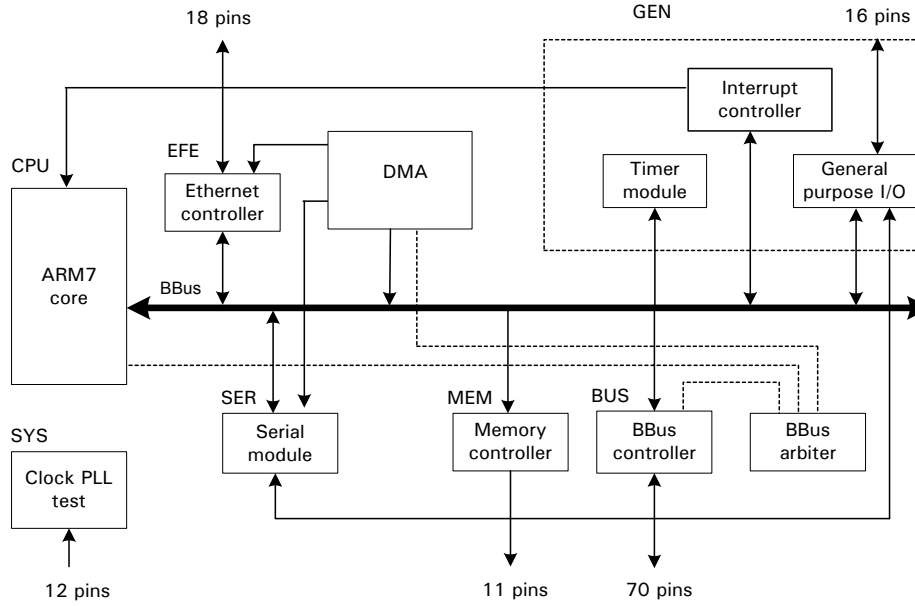


Figure 2: NET+20M device modules

Differences between the NET+50 and NET+20M chips

Although the NET+50 and NET+20M chips are similar in many aspects, there are several significant differences:

- Cache support
- Multi interface controller support
- GPIO PORTA/B/C pin availability and assignments
- Pin support
- Pinout and features

Cache support

The NET+20M does not support cache.

If you are using the NET+20M chip, you can ignore Chapter 7, "Cache."

Multi Interface Controller support

The NET+20 M does not support the multi interface controller (MIC) module. This module contains IEEE 1284 and ENI interface information.

If you are using the NET+20M chip, you can ignore Chapter 12, "MIC Controller Module."

GPIO PORTA/B/C pin availability and assignments

The GPIO PORTA/B/C registers contain eight pins each, each of which can be configured in general purpose input mode, general purpose output mode, special function input mode, and special function output mode. The NET+50 chip allows use of all 24 pins. The NET+20M chip, however, allows use of only 16 pins. This difference is noted in "General Purpose I/O (GPIO) Registers," beginning on page 73, in Chapter 6, "The GEN Module."

Pin support

The NET+50 chip supports up to 40 general-purpose I/O pins and 16 general-purpose input pins.

The NET+20M chip supports 16 general purpose input pins.

Pinout and features

Pinout and packaging

The pin/ball grid assignments and packaging for the NET+50 chip differs from the NET+20M chip. To make it easier to find chip-specific information for these items, separate chapters have been created for each chip:

- Chapter 3, "NET+50 Chip Package," beginning on page 15, contains pin/ball grid array assignments and package information for the NET+50 chip.
- Appendix B, "NET+20M Chip Package," beginning on page 481, contains ball grid array assignments and package information for the NET+20M chip.

Features

The chip features are listed in Chapter 2, "NET+50/20M Chip Features," beginning on page 7. To make it easier to review chip- specific information, the features have been presented in two lists:

- "NET+50 chip key features," beginning on page 8.
- "NET+20M chip key features," beginning on page 11.

NET+50/20M Chip Features

CHAPTER 2

The NET+50 and NET+20M chips are based on the standard architecture in the NET+ARM family of devices. NET+ARM is the hardware foundation for the NET+Works family of integrated hardware and software solutions for device networking.

This chapter lists the features supported and provided by the NET+50 and NET+20M chips.

NET+50 chip key features

The NET+50 chip key features include these components:

CPU core

- Full 32-bit ARM7TDMI RISC processor
- 32-bit internal bus
- 16-bit Thumb mode
- 8 Kbyte cache, configurable as 16 Kbyte RAM
- 15 general-purpose 32-bit registers
- 32-bit program counter and status register
- 5 supervisor modes, 1 user mode
- 2 programmable timers
- 2 async serial ports

Bus interface

- Five independent programmable chip selects with 256 Mbyte addressing per chip select
- 0–15 wait states per chip select
- 8-bit, 16-bit, and 32-bit peripheral support
- Normal and burst cycle support
- Glueless interface with all chip selects to SRAM, FP/EDO DRAM, SDRAM, and devices such as flash and EEPROM with SRAM interfaces
- External address decoding and cycle termination support
- Dynamic bus sizing support
- ASYNC and SYNC peripheral timing support
- Internal DRAM address multiplexing
- Internal refresh controller (CAS before RAS)
- Bootstrap support
- Internal bus arbiter support
- Configurable Endian support

Integrated 10/100 Ethernet MAC

- 10/100 MII-based interface to PHY
- 10 Mbit ENDEC interface
- TP-PMD and fiber-PMD device support
- Full-duplex and half-duplex modes
- Optional 4B/5B coding
- Full statistics gathering (SNMP and RMON)
- Station, broadcast, and multicast address detection and filtering
- 128 byte transmit FIFO, 2 Kbyte receive FIFO
- Intelligent receive-side buffer selection

P1284/MIC interface

- IEEE 1284 host interface with four parallel ports
- DMA support
- GPIO mode interface
- 64K shared RAM ENI interface (8- or 16-bit)
- Full-duplex FIFO mode interface (8- or 16-bit), including 32 byte transmit/receive mode FIFOs

10-channel DMA controller

- Two channels dedicated to Ethernet transmit and receive
- Four channels dedicated to serial transmit and receive
- Four channels dedicated to P1284/MIC interface
- Flexible buffer management
- Fly-by and memory-to-memory support

Serial ports

- Two fully-independent serial ports (UART, HDLC, SPI)
- Digital phase lock loop (DPLL) for receive clock extractions
- 32 byte transmit/receive FIFOs
- Internal programmable bit-rate generators
- Bit rates from 75 to 230400 in 16X mode

Bit rates from 1200 to 4 Mbps in 1X mode

(**Note:** Higher rates may be possible, depending on your design; consult factory for more information.)

- Odd, even, or no parity
- 5, 6, 7, or 8 bits; 1 or 2 stop bits
- Internal and external clock support
- Receive side character and buffer gap timers
- Four receive side data match detectors

Programmable Timers

- Two independent programmable timers (2µs to 20.7 Hours)
- Watch-dog timer (interrupt or reset on expiration)
- Bus timer

General-purpose I/O

- 40 programmable I/O interface pins and 16 input-only interface pins
- 36 pins with programmable interrupt

Clock generator

- Only a simple external crystal required
- Programmable phase lock loop (PLL), which generates a 44MHz operating frequency from an 18.432 MHz crystal. Allows a range of frequencies from 25–44 MHz, *only* by using external crystals.
- Direct external clock input support

Power and operating voltage

- 552 mW maximum (typically 368 mW), outputs switching, V_{DD} max and V_{CC} max
- 3.3 volts – I/O
- 2.5 volts – Core

NET+20M chip key features

The NET+20M chip key features include the following:

CPU core

- Full 32-bit ARM7TDMI RISC processor
- 32-bit internal bus
- 16-bit Thumb mode
- 15 general-purpose 32-bit registers
- 32-bit program counter and status register
- 5 supervisor modes and 1 user mode

Bus interface

- 5 independent programmable chip selects with 256 Mbyte addressing per chip
- 0-15 wait states per chip select
- 8-bit, 16-bit, and 32-bit peripherals
- Normal and burst cycle support
- Glueless interface with all chip selects to SRAM, FP/EDO DRAM, SDRAM, and devices such as flash and EEPROM with SRAM interfaces
- External address decoding and cycle termination support
- Dynamic bus sizing support
- ASYNC and SYNC peripheral timing support
- Internal DRAM address multiplexing
- Internal refresh controller (CAS before RAS)
- Bootstrap support
- Internal bus arbiter support
- Configurable Endian support

Integrated 10/100 Ethernet MAC

- 10/100 MII-based interface to PHY
- 10Mbit ENDEC interface

- TP-PMD and fiber-PMD device support
- Full-duplex and half-duplex modes
- Optional 4B/5B coding
- Full statistics gathering (SNMP and RMON)
- Station, broadcast, and multicast address detection and filtering
- 128-byte transmit FIFO, 2 Kbytes receive FIFO
- Intelligent receive-side buffer selection

10-channel DMA controller

- Two channels dedicated to Ethernet transmit and receive
- Four channels dedicated to serial transmit and receive
- Two channels at a time configurable for external peripherals
- Two channels reserved
- Flexible buffer management

Serial ports

- Two fully-independent serial ports (UART, HDLC, SPI)
- Digital phase locked loop (DPLL) for receive clock extractions
- 32-byte transmit/receive FIFOs
- Internal programmable bit-rate generators
- Bit rates from 75-230400 in 16X mode,

  Bit rates from 1200 bps-4 Mbps in 1X mode

  (**Note:** Higher rates may be possible, depending on your design; consult factory for more information.)
- Odd, even, or no parity
- 5, 6, 7, or 8 bits; 1 or 2 stop bits
- Internal and external clock support
- Receive-side character and buffer gap timers
- Four receive-side data match detectors

Programmable timers

- Two independent, programmable timers (2µs to 20.7 Hours)

- Watch-dog timer (interrupt or reset on expiration)
- Bus timer

Clock generator

- Only a simple external crystal required
- Programmable phase lock loop (PLL), which generates a 44MHz operating frequency from an 18.432 MHz crystal. Allows a range of frequencies from 25–44 MHz, *only* by using external crystals.
- Direct external clock input support

Power and operating voltage

- 446mW maximum (typically 297mW), outputs switching, V_{DD} max and V_{CC} max
- 3.3 volts – I/O
- 2.5 volts – core

NET+50 Chip Package

CHAPTER 3

The NET+50 chip can be used in any embedded environment requiring networking services in an Ethernet LAN. The NET+50 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+50 chip can interface with another processor using a register or shared RAM interface. The NET+50 chip provides all the tools required for any embedded networking application.

NET+50 chip pinouts

There are two variations of the NET+50 chip:

- **PQFP:** Plastic Quad Flat Pack
- **BGA:** Ball Grid Array

Figure 3 shows the NET+50 PQFP pinout. Figure 4 shows the BGA pinout. Table 1 through Table 8 provide the details for both pinouts.

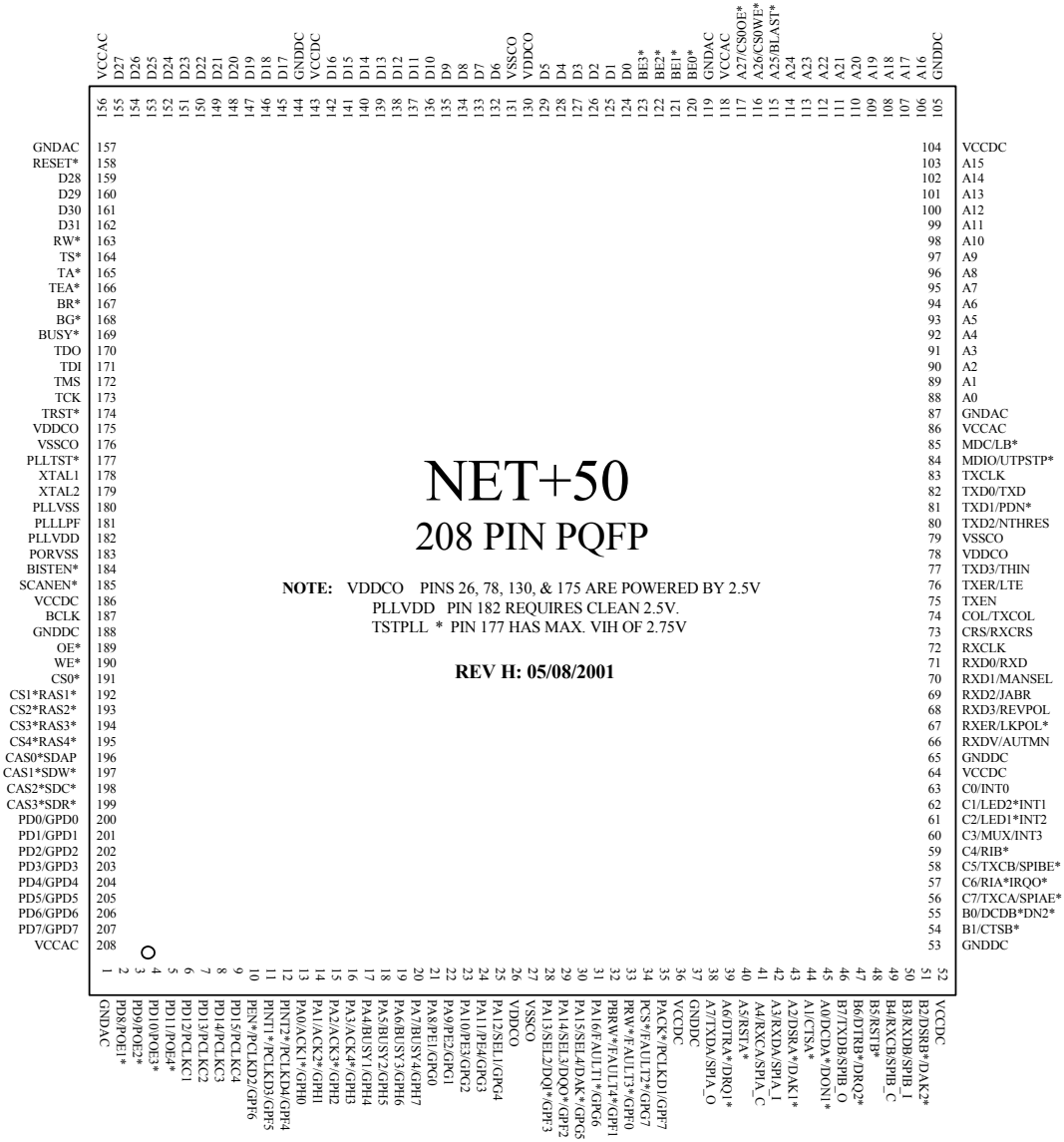


Figure 3: NET+50 PQFP pinout

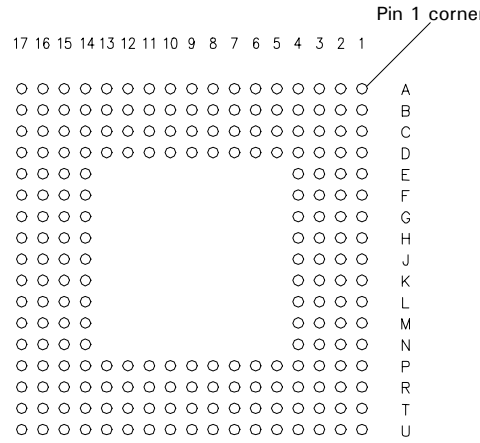


Figure 4: NET+50 BGA pinout

Table Information and Tables

Table 1 through Table 8 identify and describe the pin number assignments and ball number assignments contained in the NET+50 PQFP and BGA chips. Each table pertains to an interface, and contains the following information:

- **Signal column.** Identifies the pin name for each I/O signal. Some signals have multiple modes and are identified accordingly. You configure the mode through firmware using a configuration register. Some modes may require hardware configuration during a RESET condition.
- **208 QFP (PQFP) column.** Identifies the pin number assignment for a specific I/O signal. A dagger (†) next to the pin number indicates that the pin is an input current source.
- **BGA column.** Identifies the ball number assignment for a specific I/O signal. A dagger (†) next to the pin number indicates that the pin is an input current source.
- **I/O column.** Indicates whether the signal is input (I), output (O), or both (I/O).

- **OD (Drive) column.** Indicates the drive strength of an output buffer. The NET+50 chip uses one of three driver types:
  - 2mA
  - 4mA
  - 8mA

System bus interface

Signal		PQFP	BGA	I/O	OD	Description
BCLK		187	T10	O	8	Synchronous bus clock
ADDR27	CS0OE*	117†	F4†	I/O	4	Address bus
ADDR26	CS0WE*	116†	F3†	I/O	4	
ADDR25	BLAST*	115†	E2†	I/0	4	
ADDR24		114†	D2†	I/O	4	
ADDR23		113†	E3†	I/O	4	
ADDR22		112†	E4†	I/O	4	
ADDR21		111†	D1†	I/O	4	
ADDR20		110†	C2†	I/O	4	
ADDR19		109†	D3†	I/O	4	
ADDR18		108†	C1†	I/O	4	
ADDR17		107†	B1†	I/O	4	
ADDR16		106†	B2†	I/O	4	
ADDR15		103†	B3†	I/O	4	
ADDR14		102†	A3†	I/O	4	
ADDR13		101†	A4†	I/O	4	
ADDR12		100†	B4†	I/O	4	
ADDR11		99†	C3†	I/O	4	
ADDR10		98†	A5†	I/O	4	

Table 1: NET+50 PQFP/BGA Chip Pinout - System bus interface

Signal	PQFP	BGA	I/O	OD	Description
ADDR9	97†	D4†	I/O	4	
ADDR8	96†	C4†	I/O	4	
ADDR7	95†	B5†	I/O	4	
ADDR6	94†	A6†	I/O	4	
ADDR5	93†	D5†	I/O	4	
ADDR4	92†	C5†	I/O	4	
ADDR3	91†	B6†	I/O	4	
ADDR2	90†	A7†	I/O	4	
ADDR1	89†	D6†	I/O	4	
ADDR0	88†	C6†	I/O	4	
DATA31	162	T3	I/O	4	Data bus
DATA30	161	R4	I/O	4	
DATA29	160	U3	I/O	4	
DATA28	159	U2	I/O	4	
DATA27	155	R2	I/O	4	
DATA26	154	R1	I/O	4	
DATA25	153	P1	I/O	4	
DATA24	152	P2	I/O	4	
DATA23	151	R3	I/O	4	
DATA22	150	N1	I/O	4	
DATA21	149	P4	I/O	4	
DATA20	148	P3	I/O	4	
DATA19	147	N2	I/O	4	
DATA18	146	M1	I/O	4	
DATA17	145	N4	I/O	4	
DATA16	142	L1	I/O	4	
DATA15	141	M4	I/O	4	

Table 1: NET+50 PQFP/BGA Chip Pinout - System bus interface

Signal	PQFP	BGA	I/O	OD	Description
DATA14	140	M3	I/O	4	
DATA13	139	L2	I/O	4	
DATA12	138	K1	I/O	4	
DATA11	137	L4	I/O	4	
DATA10	136	L3	I/O	4	
DATA9	135	K2	I/O	4	
DATA8	134	J1	I/O	4	
DATA7	133	K4	I/O	4	
DATA6	132	K3	I/O	4	
DATA5	129	J4	I/O	4	
DATA4	128	J3	I/O	4	
DATA3	127	H2	I/O	4	
DATA2	126	G1	I/O	4	
DATA1	125	H4	I/O	4	
DATA0	124	H3	I/O	4	
TS*	NO CONNECT				
BE3*	123	G2	I/O	2	Byte enable D31:D24
BE2*	122	F1	I/O	2	Byte enable D23:D16
BE1*	121	G4	I/O	2	Byte enable D15:D08
BE0*	120	G3	I/O	2	Byte enable D07:D00
RW*	163	U4	I/O	2	Transfer direction
TA*	165†	R5†	I/O	8	Data transfer acknowledge
TEA*	166†	T4†	I/O	8	Transfer error/Last acknowledge
BR*	NO CONNECT				
BG*	NO CONNECT				
BUSY*	NO CONNECT				

Table 1: NET+50 PQFP/BGA Chip Pinout - System bus interface

Chip select controller

Signal	PQFP	BGA	I/O	OD	Description
CS0*	191	T11	O	4	Chip select (Boot select)
CS1*/RAS1*	192	R12	O	4	Chip select/DRAM RAS*
CS2*/RAS2*	193	P12	O	4	Chip select/DRAM RAS*
CS3*/RAS3*	194	U11	O	4	Chip select/DRAM RAS*
CS4*/RAS4*	195	T12	O	4	Chip select/DRAM RAS*
CAS3*/SDRAS*	199	T13	O	4	DRAM column strobe D31:24/ SDRAM RAS
CAS2*/SDCAS*	198	U12	O	4	DRAM column strobe D23:16/ SDRAM RAS
CAS1*/SDWE*	197	P13	O	8	DRAM column strobe D15:08/ SDRAM RAS
CAS0*/SD(AP)	196	R13	O	4	DRAM column strobe D07:00/ SDRAM RAS
WE*	190	R11	O	4	Write enable
OE*	189	P11	O	4	Output enable

Table 2: NET+50 PQFP/BGA Chip Pinout - Chip select controller

Ethernet interface

Signal		PQFP	BGA	I/O	OD	Description	
MII	**10BaseT**					**MII**	**10BaseT**
MDC	LB*	85	D7	O	2	MII clock	Loopback enable
MDIO	UTPSTP*	84†	C7†	I/O	2	MII data	Cable type
TXCLK	TXCLK	83	B8	I		TX clock	
TXD0	TXD	82	A9	O	2	TX data 0	TX data
TXD1	PDN* (OD)	81	D8	O	2	TX data 1	Power down

Table 3: NET+50 PQFP/BGA Chip Pinout - Ethernet interface

Signal		PQFP	BGA	I/O	OD	Description	
TXD2	NTHRES	80	C8	O	2	TX data 2	Normal threshold
TXD3	THIN	77	D9	O	2	TX data 3	Enable Thinnet
TXER	LTE	76	C9	O	2	TX code error	Link test enable
TXEN	TXEN	75	B10	O	2	TX enable	
COL	TXCOL	74	A11	I		Collision	
CRS	RXCRS	73	D10	I		Carrier sense	
RXCLK	RXCLK	72	C10	I		RX clock	
RXD0	RXD	71	B11	I		RX data 0	RX data
RXD1	MANSENSE	70	A12	I		RX data 1	Sense jumper
RXD2	JABBER	69	D11	I		RX data 2	Jabber
RXD3	REVPOL	68	C11	I		RX data 3	Reverse polarity
RXER	LINKPUL*	67	B12	I		RX error	Link pulse detection
RXDV	AUTOMAN	66	A13	I		RX data valid	10B2 selected

Table 3: NET+50 PQFP/BGA Chip Pinout - Ethernet interface

MIC interface

Signal			PQFP	BGA	I/O	OD	Description
IEEE 1284	**MIC**	**GPIO**					
PDATA0	PDATA0	GPIOD0	200†	R14†	I/O	2	Parallel 1284/ENI/GPIO
PDATA1	PDATA1	GPIOD1	201†	P14†	I/O	2	
PDATA2	PDATA2	GPIOD2	202†	U13†	I/O	2	
PDATA3	PDATA3	GPIOD3	203†	R15†	I/O	2	
PDATA4	PDATA4	GPIOD4	204†	T14†	I/O	2	
PDATA5	PDATA5	GPIOD5	205†	U14†	I/O	2	

Table 4: NET+50 PQFP/BGA Chip Pinout - MIC interface

Signal			PQFP	BGA	I/O	OD	Description
PDATA6	PDATA6	GPIOD6	206†	U15†	I/O	2	
PDATA7	PDATA7	GPIOD7	207†	T15†	I/O	2	
POE1*	PDATA8		2†	T16†	I/O	2	
POE2*	PDATA9		3†	T17†	I/O	2	
POE3*	PDATA10		4†	R17†	I/O	2	
POE4*	PDATA11		5†	P15†	I/O	2	
PCLKC1	PDATA12		6†	R16†	I/O	2	
PCLKC2	PDATA13		7†	P17†	I/O	2	
PCLKC3	PDATA14		8†	N14†	I/O	2	
PCLKC4	PDATA15		9†	N15†	I/O	2	
PCLKD1	PACK*	GPIOF7	35	G16	I/O	8	
PCLKD2	PEN*	GPIOF6	10†	P16†	I/O	2	
PCLKD3	PINT1*	GPIOF5	11	N16	I/O	2	
PCLKD4	PINT2*	GPIOF4	12	M15	I/O	2	Or ENI DMA output PDRQIO
ACK1*	PA0	GPIOH0	13	M14	I		
ACK2*	PA1	GPIOH1	14	N17	I		
ACK3*	PA2	GPIOH2	15	M16	I		
ACK4*	PA3	GPIOH3	16	L15	I		
BUSY1	PA4	GPIOH4	17	L14	I		
BUSY2	PA5	GPIOH5	18	M17	I		
BUSY3	PA6	GPIOH6	19	L16	I		
BUSY4	PA7	GPIOH7	20	K15	I		
PE1	PA8	GPIOG0	21	K14	I		
PE2	PA9	GPIOG1	22	L17	I		
PE3	PA10	GPIOG2	23	K16	I		
PE4	PA11	GPIOG3	24	J15	I		
PSELECT1	PA12	GPIOG4	25	J14	I		

Table 4: NET+50 PQFP/BGA Chip Pinout - MIC interface

Signal			PQFP	BGA	I/O	OD	Description
PSELECT2	PA13	GPIOF3	28	H15	I/O	2	Or ENI DMA output PDRQI*
PSELECT3	PA14	GPIOF2	29	H14	I/O	2	Or ENI DMA output PDRQO*
PSELECT4	PA15	GPIOG5	30	J17	I		Or ENI DMA input PDACK*
FAULT1*	PA16	GPIOG6	31	H16	I		
FAULT2*	PCS*	GPIOG7	34	H17	I		
FAULT3*	PRW*	GPIOF0	33	G14	I/O	2	
FAULT4*	PBRW*	GPIOF1	32	G15	I/O	2	

Table 4: NET+50 PQFP/BGA Chip Pinout - MIC interface

UARTS-SPI-GPIO

Signal		PQFP	BGA	I/O	OD	Description	
PORTA7	TXDA	38†	G17†	I/O	2	SPI-S-TXD-O-A	SPI-M-TXD-O-A
PORTA6	DTRA*/DRQ1*	39†	F16†	I/O	2		
PORTA5	RTSA*	40†	E15†	I/O	2		
PORTA4	RXCA	41†	E14†	I/O	2	SPI-S-CLK-I-A	SPI-M-CLK-O-A*
PORTA3	RXDA	42†	F17†	I/O	2	SPI-S-RXD-I-A	SPI-M-RXD-I-A
PORTA2	DSRA*/DAK1*	43†	E16†	I/O	2		
PORTA1	CTSA*	44†	D15†	I/O	2		
PORTA0	DCDA*/DON1*	45†	D14†	I/O	2		
PORTB7	TXDB	46†	E17†	I/O	2	SPI-S-TXD-O-B	SPI-M-TXD-O-B
PORTB6	DTRB*/DRQ2*	47†	C15†	I/O	2		
PORTB5	RTSB*	48†	D16†	I/O	2	Reject* Ethernet packet	
PORTB4	RXCB	49†	D17†	I/O	2	SPI-S-CLK-I-B	SPI-M-CLK-O-B*
PORTB3	RXDB	50†	C17†	I/O	2	SPI-S-RXD-I-B	SPI-M-RXD-O-B
PORTB2	DSRB*/DAK2*	51†	C16†	I/O	2		

Table 5: NET+50 PQFP/BGA Chip Pinout - UARTS-SPI-GPIO

Signal		PQFP	BGA	I/O	OD	Description	
PORTB1	CTSB*	54†	B16†	I/O	2	RPSF* Ethernet frame boundary	
PORTB0	DCDB*/DON2*	55†	A16†	I/O	2		
PORTC7	TXCA	56†	A15†	I/O	4	SPI-S-EN-I-A	SPI-M-EN-O-A
PORTC6	RIA*/IRQO*	57†	C14†	I/O	4		
PORTC5	TXCB	58†	B15†	I/O	4	SPI-S-EN-I-B	SPI-M-EN-O-B
PORTC4	RIB*	59†	A14†	I/O	4		
PORTC3	AMUX	60†	D13†	I/O	8	Interrupt 3	
PORTC2		61†	C13†	I/O	8	Interrupt 2	
PORTC1		62†	B14†	I/O	8	Interrupt 1	
PORTC0		63†	B13†	I/O	8	Interrupt 0	

Table 5: NET+50 PQFP/BGA Chip Pinout - UARTS-SPI-GPIO

Clock generation/system reset

Signal	PQFP	BGA	I/O	OD	Description
XTAL1	178	U7	I		Crystal oscillator circuit
XTAL2	179	T8	O		
PLLVDD (2.5V)	182	U8			2.5 V PLL clean power
PLLLPF	181	P9			PLL loop filter capacitor
PLLVSS	180	R9			PLL clean ground
PLLTST* (2.5V)	177†	P8†	I		PLL test mode
BISTEN*	184†	R10†	I		Enable internal BIST operation
SCANEN*	185†	P10†	I		Enable internal SCAN testing
System reset					
RESET*	158†	T2†	I		System reset

Table 6: NET+50 PQFP/BGA Chip Pinout - Clock generation/system reset

JTAG port for ARM core

Signal	PQFP	BGA	I/O	OD	Description
TDI	171†	T6†	1		Test data in
TDO	170	U5	O	2	Test data out
TMS	172†	R7†	I		Test mode select
TRST*	174	R8	I		Test mode reset. This is a current input sink.
TCK	173	P7	I		Test mode clock

Table 7: NET+50 PQFP/BGA Chip Pinout - JTAG port for ARM core

Power supply

Signal	Mode	Pin/Ball Number	Description
V_{CC}DC 3.3V DC	BGA	F15, B17, C12, A2, M2, U9	I/O steady state (6 pairs)
	PQFP	36, 52, 64, 104, 143, 186	
V_{SS}DC GND Returns	BGA	F14, A17, D12, A1, N3, U10	
	PQFP	37, 53, 65, 105, 144, 188	
V_{CC}AC 3.3V	BGA	A8, E1, T1, U16	I/O switching (4 pairs[a])
	PQFP	86, 118, 156, 208	
V_{SS}AC GND Returns	BGA	B7, F2, U1, U17	
	PQFP	87, 119, 157, 1	
V_{DD}CO 2.5V	BGA	K17, A10, H1, T7	Core power (4 pairs)
	PQFP	26, 78, 130, 175	
V_{SS}CO GND Returns	BGA	J16, B9, J2, U6	
	PQFP	27, 79, 131, 176	

Table 8: NET+50 PQFP/BGA Chip Pinout - Power supply

Signal	Mode	Pin/Ball Number	Description
$PLLV_{DD}$ 2.5V	BGA	U8	PLL bead filtered clean power
	PQFP	182	
$PLLV_{SS}$ GND Return	BGA	R9	
	PQFP	180	
$PORV_{SS}$ GND	BGA	T9	Powerup reset GND reference
	PQFP	183	

Table 8: NET+50 PQFP/BGA Chip Pinout - Power supply

a. NetSilicon recommends that you use separate power pairs for AC and DC power, to prevent the noise in the AC power buses from reaching the DC power buses. NetSilicon recommends and uses a ferrite bead to filter the AC power pins.

Signal description summary

The signal description summary defines and briefly describes the signals included in the interfaces in the NET+50 chip. The description tables in the section are presented in the same order as the NET+50 chip pinout tables. For details about any of the signals/pinouts, see the appropriate chapter:

Interface	Chapter
System bus	Chapter 8, "Memory Controller Module"
Chip select controller	Chapter 8, "Memory Controller Module"
Ethernet MII	Chapter 10, "Ethernet Controller Module"
MIC	Chapter 12, "MIC Controller Module"
UARTS-SPI-GPIO; GPIO ports A, B, C	Chapter 6, "The GEN Module"
Clock generation	Chapter 6, "The GEN Module"

The JTAG Port for ARM Core (ARM Debugger) and Power sections are addressed fully in their respective discussions.

System bus interface

The NET+50 chip uses the system bus interface to interface with memory-mapped peripheral devices such as flash, SRAM, and DRAM.

Code	Definition	Description
BCLK	Bus clock	Provides the bus clock. All system bus interface signals are referenced to the BCLK signal.
ADDR	Address bus	Identifies the address of the peripheral being addressed by the current bus master. The address bus is bi-directional.
DATA	Data bus	Provides the data transfer path between the NET+50 chip and external peripheral devices.The data bus is bi-directional.
TS*	Transfer start	NO CONNECT
BE*	Byte enable	Identifies which 8-bit bytes of the 32-bit data bus are active during any given system bus memory cycle. The BE* signals are active low and bi-directional.
RW*	Read/write indicator	Indicates the direction of the system bus memory cycle. RW* high identifies a read operation; RW* low identifies a write operation. The RW* signal is bi-directional.
TA*	Transfer acknowledge	Indicates the end of the current system bus memory cycle. This signal is bi-directional.
TEA*	Transfer error acknowledge	Indicates an error termination or burst cycle termination. TEA* is bi-directional. The NET+50 chip or the external peripheral can drive the TEA* signal.
BR*	Bus request	NO CONNECT
BG*	Bus grant	NO CONNECT
BUSY*	Bus busy	NO CONNECT

Table 9: System bus interface signal description

Chip select controller

The NET+50 chip supports five unique chip select configurations.

Code	Definition	Description
CS0* CS1* CS2* CS3* CS4*	Chip select 0 Chip select 1 Chip select 2 Chip select 3 Chip select 4	Unique chip select outputs supported by the NET + 50 chip. Each chip select can be configured to decode a portion of the available address space and can address a maximum of 256 Mbytes of address space. The chip selects are configured using registers in the memory module.
CAS0* CAS1* CAS2* CAS3*	Column address strobe signals	Activated when an address is decoded by a chip select module 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.
WE*	Write enable	Active low signal indicating that a memory write cycle is in progress. This signal is activated only during write cycles to peripherals controlled by one of the chip selects in the memory module.
OE*	Output enable	Active low signal indicating that a memory read cycle is in progress. This signal is activated only during read cycles from peripherals controlled by one of the chip selects in the memory module.

Table 10: Chip select controller signal description

Ethernet MII

The Ethernet MII (Media Independent Interface) provides the connection between the Ethernet PHY and the MAC (Media Access Controller).

Code	Definition	Description
MDC	Management data clock	Provides the clock for the MDIO serial data channel. The MDC signal is a NET + 50 chip output. The maximum frequency is 2.5 MHz.
MDIO	Management data IO	A bi-directional signal that provides a serial data channel between the NET + 50 chip and the external Ethernet PHY module.
TXCLK	Transmit clock	An input to the NET + 50 chip from the external PHY module. TXCLK provides the synchronous data clock for transmit data.

Table 11: Ethernet MII signal description

Code	Definition	Description
TXD3 TXD2 TXD1 TXD0	Transmit data signals	Nibble bus used by the NET+50 chip to drive data to the external Ethernet PHY. All transmit data signals are synchronized to TXCLK. In ENDEC mode, only TDX0 is used for transmit data.
TXER	Transmit coding error	Output asserted by the NET+50 when an error has occurred in the transmit data stream.
TXEN	Transmit enable	Asserted when the chip drives valid data on the TXD outputs. This signal is synchronized to TXCLK.
COL	Transmit collision	Input signal asserted by the external Ethernet PHY when a collision is detected.
CRS	Receive carrier sense	Asserted by the external Ethernet PHY whenever the receive medium is non-idle.
RXCLK	Receive clock	An input to the NET+50 chip from the external PHY module. The receive clock provides the synchronous data clock for receive data.
RXD3 RXD2 RXD1 RXD0	Receive data signals	Nibble bus used by the NET+50 chip to input receive data from the external Ethernet PHY. All receive data signals are synchronized to RXCLK. In ENDEC mode, only RXD0 is used for receive data.
RXER	Receive error	Input asserted by the external Ethernet PHY when the Ethernet PHY encounters invalid symbols from the network.
RXDV	Receive data valid	Input asserted by the external Ethernet PHY when the Ethernet PHY drives valid data on the RXD inputs.

Table 11: Ethernet MII signal description

MIC interface

The NET+50 chip uses the MIC interface to interface with other embedded computer systems. The MIC interface provides various modes: GPIO, IEEE 1284, shared RAM, and FIFOs.

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

If this port is not used, it must be defaulted during bootstrap to a known value, such as 100 (ENI Shared-16 RAM Mode), by providing input current sink resistors

on NET+50 pins A22, A21, and A20. Unused input pins must be terminated according to the NET+50 minimum configuration.

When the MIC interface is set to GPIO or 1284 Mode, unused inputs in each of these modes must be terminated.

This section provides details for each MIC interface mode.

IEEE 1284 mode

Code	Definition	Description
PDATA	1284 data bus	A bi-directional data bus that interfaces the NET + 50 chip with the external 1284 port data and control transceivers. The NET + 50 chip interfaces only with a single 1284 port at a time.
	Output enables:	Channel output enable signals that control the output enable for each specific port data register.
POE1*	1284 Channel 1	
POE2*	1284 Channel 2	
POE3*	1284 Channel 3	
POE4*	1284 Channel 4	
	Control clocks:	Channel control clock signals that are driven by the NET + 50 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.
PCLKC1	1284 Channel 1	
PCLKC2	1284 Channel 2	
PCLKC3	1284 Channel 3	
PCLKC4	1284 Channel 4	
	Data clocks:	Channel data clock signals that are driven by the NET + 50 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 data clocks are never active at the same time as any channel control clocks.
PCLKD1	1284 Channel 1	
PCLKD2	1284 Channel 2	
PCLKD3	1284 Channel 3	
PCLKD4	1284 Channel 4	
	Acknowledge:	Channel ACK inputs to the NET + 50 chip from the external IEEE 1284 port. The ACK input meaning changes as the IEEE 1284 operational mode changes. **Note:** For additional information about IEEE 1284 operational modes, see your IEEE documentation.
ACK1*	1284 Channel 1	
ACK2*	1284 Channel 2	
ACK3*	1284 Channel 3	
ACK4*	1284 Channel 4	

Table 12: MIC IEEE 1284 mode signal description

Code	Definition	Description
	BUSY:	Channel BUSY inputs to the NET+50 chip from the external IEEE 1284 port. The BUSY input meaning changes as the IEEE 1284 operational mode changes. **Note:** For additional information about IEEE 1284 operational modes, see your IEEE documentation.
BUSY1*	1284 Channel 1	
BUSY2*	1284 Channel 2	
BUSY3*	1284 Channel 3	
BUSY4*	1284 Channel 4	
	ERROR:	Channel ERROR inputs to the NET+50 chip from the external IEEE 1284 port. The ERROR input meaning changes as the IEEE 1284 operational mode changes. **Note:** For additional information about IEEE 1284 operational modes, see your IEEE documentation.
PE1*	1284 Channel 1	
PE2*	1284 Channel 2	
PE3*	1284 Channel 3	
PE4*	1284 Channel 4	
	PSELECT:	Channel PSELECT (peripheral select) inputs to the NET+50 chip from the external IEEE 1284 port. The PSELECT input meaning changes as the IEEE 1284 operational mode changes. **Note:** For additional information about IEEE 1284 operational modes, see your IEEE documentation.
PSELECT1*	1284 Channel 1	
PSELECT2*	1284 Channel 2	
PSELECT3*	1284 Channel 3	
PSELECT4*	1284 Channel 4	
	FAULT:	Channel FAULT inputs to the NET+50 chip from the external IEEE 1284 port. The FAULT input meaning changes as the IEEE 1284 operational mode changes. **Note:** For additional information about IEEE 1284 operational modes, see your IEEE documentation.
FAULT1*	1284 Channel 1	
FAULT2*	1284 Channel 2	
FAULT3*	1284 Channel 3	
FAULT4*	1284 Channel 4	

Table 12: MIC IEEE 1284 mode signal description

ENI host mode

Code	Definition	Description
PCS*	ENI chip select	Provides the chip enable for the ENI interface from the external processor system.
PRW*	ENI read/write	Driven by the external processor system during an ENI access cycle to indicate the data transfer
PA[16:0]	ENI address bus	17 required address signals that identify the resource being accessed. The PA bus must be valid while PCS* is low. During DMA cycles, the PA address bus is ignored and only the FIFO data register is accessible.

Table 13: ENI host mode signal description

Code	Definition	Description
PDACK*/ PA15	ENI DMA acknowledge	PA15 input used for the DMA acknowledge input PDACK*, when DMA FIFO operations are enabled (when the DMAE* bit in the ENI control register is set to 0).
PDRQO*/ PDRQO/PA14	ENI outbound FIFO DMA request	PA14 output used for the active low outbound FIFO DMA ready output PDRQO*, when DMA FIFO operations are enabled.The PDRQO* signal is routed through PA14 only when both DMAE* and DMAE2 are set to 0 in the ENI Control register.
PDRQI*/ PDRQI/PA13	ENI outbound FIFO DMA request	PA13 output used for the active low inbound FIFO DMA ready output PDRQI*, when DMA FIFO operations are enabled. THE PDRQI* signal is routed through PA13 only when both DMAE* and DMAE2 are set to 0 in the ENI Control register.
PDATA[15:0]	ENI data bus	16-bit data bus supported by ENI.
PACK*	ENI acknowledge	Indicates the requested ENI cycle is complete. PACK* is driven by the ENI interface in response to an active low PCS* signal.
PINT1*	ENI interrupt 1	Indicates an interrupt condition to the external ENI processor.
PINT2*	ENI interrupt 2	Indicates an interrupt condition to the external ENI processor.
PBRW*	ENI data buffer read/write	An output that controls the data direction pin for an external data bus transceiver.
PEN*	ENI data buffer enable	An output that controls the output enable pin for an external data bus transceiver.

Table 13: ENI host mode signal description

MIC interface configured for GPIO mode

Full configuration information for Ports D-H appears in the GPIO Mode section in Chapter 12, "MIC Controller Module."

Code	Definition	Description
GPIOD [7:0]	General purpose I/O	Each pin can be configured for input, output, or level-sensitive interrupt.
GPIOF [7:0]	General purpose I/O	Each pin can be configured for input, output, or level-sensitive interrupt.

Table 14: MIC GPIO mode signal description

Code	Definition	Description
GPIOG [7:0]/ GPIO [7:0]	General purpose I/O	Each pin can be configured input **low** level-sensitive interrupt.
GPIOH [7:0]/ GPIO [7:0]	General Purpose I/O	Each pin can be configured for input **low** level-sensitive interrupt.

Table 14: MIC GPIO mode signal description

GPIO (PORTA/B/C)

See Chapter 6, "The GEN Module," for full configuration information for PORTA/B/C. All ports can be can be configured for general purpose I/O. The descriptions below highlight some of each port's special functions.

Code	Definition	Description
PORTA7	Input/Output/TxDA/SPI-A-M + S-TXD-O	Can be configured as the special function TxDA output.
PORTA6	Input/Output/DREQ1*/DTRA*	Can be configured as the special function DREQ1* input or the special function DTRA.
PORTA5	Input/Output/RTSA*	Can be configured as the special function RTSA output.
PORTA4	Input/Output/RxCA/SPI-A-S-CLK-I/ SPI-A-M-CLK-O	Can be configured for special function RxCA input or the special function OUT1A output.
PORTA3	Input/Output/RxDA/SPI-A-S + M-RXD-I	Can be configured for special function RxDA input.
PORTA2	Input/Output/DSRA*/DACK1*	Can be configured for special function DSRA input or for special function DACK1* output.
PORTA1	Input/Output/CTSA*	Can be configured for special function CTSA input.
PORTA0	Input/Output/DCDA*/DONE1*	Can be configured for special function DCDA input or DONE1* I/O.
PORTB7	Input/Output/TxDB/SPI-B-M and S-TXD-O	Can be configured as the special function TxDB output.
PORTB6	Input/Output/DREQ2*/DTRB*	Can be configured as the special function DREQ2* input or the special function DTRB.

Table 15: GPIO PORTA/B/C signal description

Code	Definition	Description
PORTB5	Input/Output/REJECT*/RTSB*	Can be configured as the special function REJECT input or for special function RTSB output.
PORTB4	Input/Output/RxCB/SPI-B-M-CLK-O/ SPI-B-S-CLK-I	Can be configured for special function RxCB input or special function OUT1B.
PORTB3	Input/Output/RxDB/SPI-B-S + M-RXD-I	Can be configured for special function RxDB input.
PORTB2	Input/Output/DSRB*/DACK2*	Can be configured for special function DSRB input or special function DACK2* output.
PORTB1	Input/Output/CTSB*/RPSF*	Can be configured for special function RPSF* output or special function CTSB input.
PORTB0	Input/Output/DCDB*/DONE2*	Can be configured for special function DCDB input or special function DONE2* I/O.
PORTC7	Input/Output/TxCA/SPI-A-M-ENB-O/ SPI-A-S-ENB-I	Can be configured for special function TxCA input or special function OUT2A output.
PORTC6	Input/Output/RIA*/IRQ*	Can be configured as the special function RIA* input or for special function IRQ* output.
PORTC5	Input/Output/TxCB/SPI-B-M-ENB-O/ SPI-B-S-ENB-I	Can be configured for special function TxCB input or special function OUT2B output.
PORTC4	Input/Output/RIB*	Can be configured for special function RIB input.
PORTC3	Input/Output/CI3/AMUX	Can be configured for special function C3 interrupt input or to provide the DRAM address multiplexer function.
PORTC2	Input/Output/CI2	Can be configured for special function C2 interrupt input.
PORTC1	Input/Output/CI1	Can be configured for special function C1interrupt input.
PORTC0	Input/Output/CI0	Can be configured for special function C0 interrupt input.

Table 15: GPIO PORTA/B/C signal description

Clock generation and reset

The NET+50 has three main clock domains:

- System clock (SYSCLK)
- Bit rate generation and programmable timer reference clock (XTAL)

- System bus clock (BCLK)

The SYS module provides the NET+50 chip with these clocks, as well as system reset and boot-up resources.

Code	Definition	Description
XTAL1	Crystal oscillator input	A standard parallel-resonant crystal can be attached to these pins to provide the main input clock to the NET+50 chip.
XTAL2	Crystal oscillator output	The oscillator frequency equals SYSCLK. XTAL1 is in the 2.5V ring and has a maximum VIH of 2.75V. A typical 3.3V oscillator requires a 220 ohm series resistor with its output.
$PLLV_{DD}$	Clean PLL power	Powers the internal 2.5V PLL circuit.
$PLLV_{SS}$	Clean PLL ground	Grounds the internal PLL circuit.
PLLPF	PLL loop filter	Provides the PLL loop filter circuit.
RESET*	System reset	Resets the NET+50 chip hardware.

Table 16: Clock generation and reset signal description

Test support

The PLLTST*, BISTEN*, and SCANEN* primary input pins put the NET+50 into various test modes, for both functional and manufacturing test operations.

Code	Definition	Description
PLLTST*	PLL test and disable	Disables the internal PLL when driven active low. Enables the crystal oscillator and internal PLL when left unconnected or driven high.
BISTEN*	BIST enable	Used for testing purposes only; the pin should always be left unconnected or driven high.
SCANEN*	SCAN enable	Used for testing purposes only; the pin should always be left unconnected or driven high.

Table 17: Test support signal description

For more information about PLLTST*, see Chapter 6, "The GEN Module."

ARM debugger

Five pins provide a dedicated connection to the internal ARM processor core. These five signals connect to the ARM processor only, 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

The following figure illustrates the pin connection:

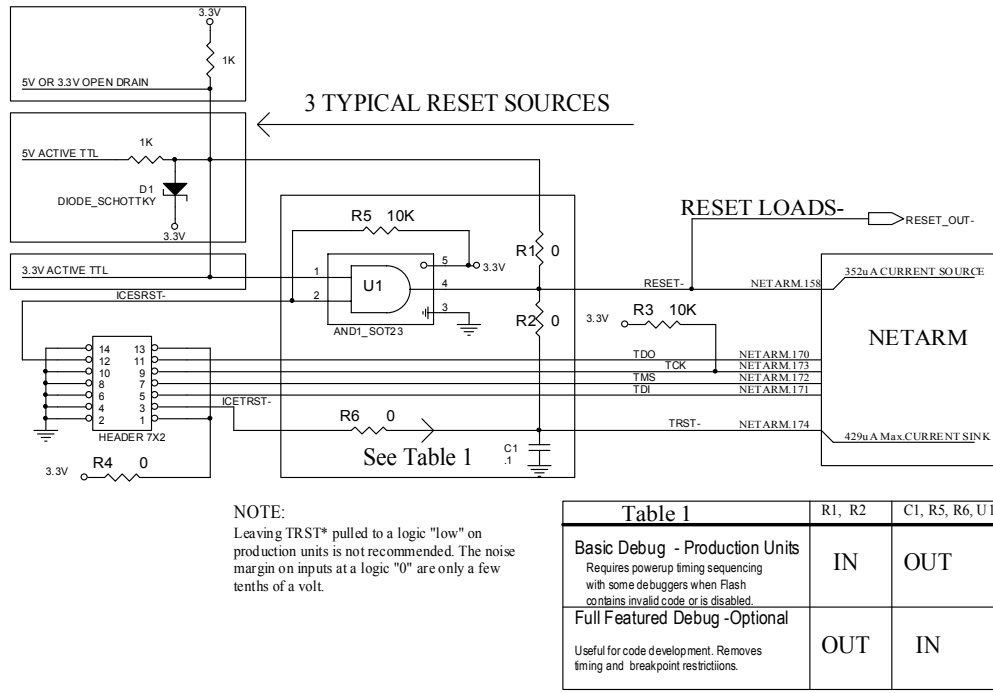


Figure 5: NET+50/20M reset and ARM debugger circuit

Power

This section describes the pins that provide power for various components of the I/O output drivers and the internal NET+50 chip core.

Code	Definition	Description
$V_{CC}AC$	Power A/C switching	The $V_{CC}AC$ 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. Each $V_{CC}AC$ pin has a GNDAC ground return.
$V_{CC}DC$	Power DC drive	The $V_{CC}DC$ 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. Each $V_{CC}DC$ pin has a GNDDC ground return.
$V_{DD}CO$	Power core	The $V_{DD}CO$ pins provide the 2.5V power for the internal NET+50 chip core. Each of these pins must have a bypass capacitor placed as close to the pin as physically possible. Each $V_{DD}CO$ pin has a $V_{SS}CO$ ground return.
$PORV_{SS}$	Power on reset	Grounds the power-on-reset circuitry.
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+50 chip core. These pins should not require any filtering other than standard decoupling practices.

Table 18: Power signal description

Packaging

NET + 50 PQFP package

Figure 6 shows the package for the NET+50 PQFP. Table 19 provides the NET+50 PQFP dimensions:

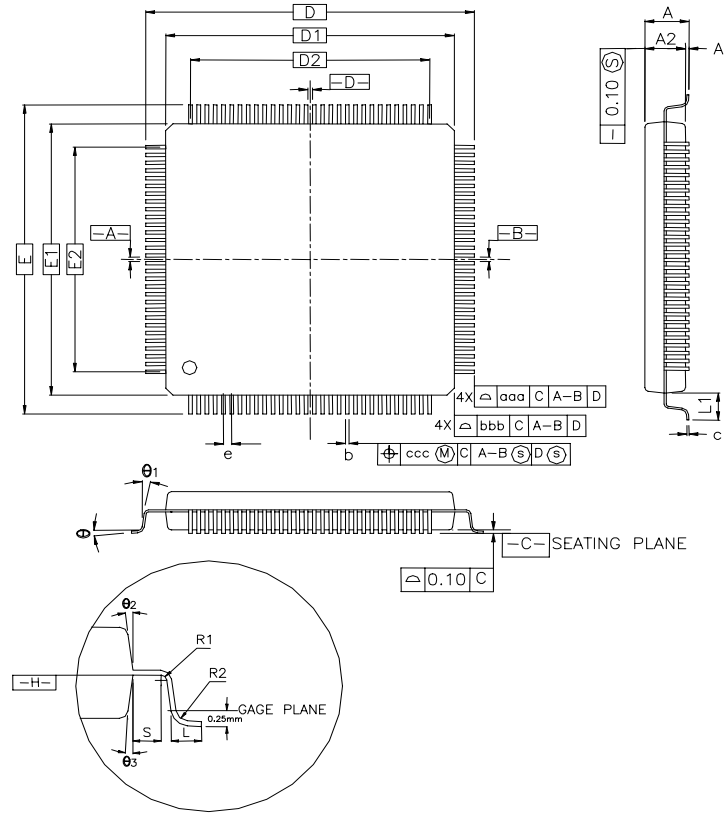


Figure 6: Short plastic quad flat pack (PQFP)

	Millimeter			Inches		
Symbol	**Min**	**Nom**	**Max**	**Min**	**Nom**	**Max**
A	----------	----------	4.10	----------	----------	0.161
A_1	0.25	----------	----------	0.010	----------	----------
A_2	3.20	3.32	3.60	0.126	0.131	0.142
D		31.20 BASIC			1.228 BASIC	
D_1		28.00 BASIC			1.102 BASIC	
E		31.20 BASIC			1.228 BASIC	
E_1		28.00 BASIC			1.102 BASIC	
R_2	0.13	----------	0.30	0.005	----------	0.012
R_1	0.13	----------	----------	0.005	----------	----------
θ_1	0°	----------	7°	0°	----------	7°
θ_{12}	0°	----------	----------	0°	----------	----------
θ_{23}		8° REF			8° REF	
θ_3		8° REF			8° REF	
c	0.11	0.15	0.23	0.004	0.006	0.009
L	0.73	0.88	1.03	0.029	0.035	0.041
L_1		1.60 REF			0.063 REF	
S	0.20	----------	----------	0.008	----------	----------
b	0.17	0.20	0.27	0.007	0.008	0.011
e		0.50 BSC.			0.020 BSC.	
D2		25.50			1.004	
E2		25.50			1.004	
TOLERANCES OF FORM AND POSITION						
aaa		0.25			0.010	
bbb		0.20			0.008	
ccc		0.08			0.003	

Table 19: PQFP dimensions

BGA package

Figure 7 shows the package for the NET+50 BGA. Table 20 provides the NET+50 BGA dimensions:

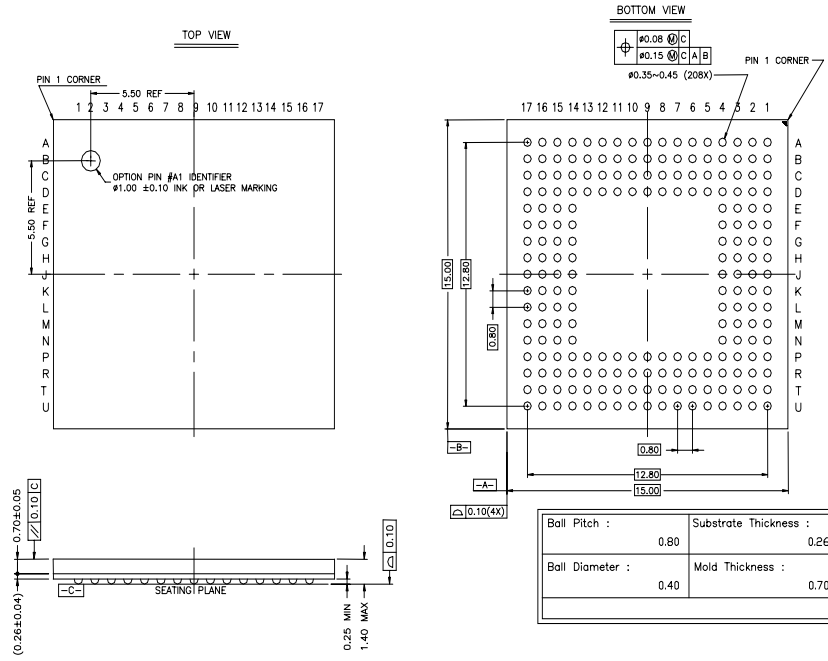


Figure 7: BGA package

Dimension	Minimum	Nominal	Maximum
A	----------	----------	1.40
A1	0.22	0.26	0.30
A2	0.65	0.70	0.75
A3	0.25	----------	----------
B	----------	5.50 REF	----------
D	----------	15.00	----------
D1	----------	12.80	----------

Table 20: BGA dimensions

Dimension	Minimum	Nominal	Maximum
E	----------	15.00	----------
E1	----------	12.80	----------
P	----------	0.80	----------

Table 20: BGA dimensions

Working with the CPU

CHAPTER 4

The CPU module uses an ARM7TDMI core processor, which provides high performance while maintaining low power consumption and size. This chapter describes the ARM Thumb concept and provides an overview of ARM exceptions and hardware interrupts.

Note: The information in this chapter applies to both the NET+50 and NET+20M chips, unless otherwise noted.

Thumb concept

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

The primary attribute of the Thumb is a super-reduced instruction set. The ARM7TDMI processor has essentially 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. Thumb code operates on the same 32-bit register set as the ARM code, but uses only about 65% of the same code size compiled in ARM mode.

Thumb instructions operate with the standard ARM register configuration. 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; the Thumb instruction set, however, can fit twice as many instructions in the same code space. The net result is a 35% decrease in overall code density.

Working with ARM exceptions

Exceptions occur whenever the normal flow of a program is halted temporarily; for example, to service an interrupt from a peripheral. Each ARM exception causes the ARM processor to save some state information and then jump to a location in low memory. Before an exception can be handled, the current processor state must be preserved so the original program can resume when the handler routine has finished.

Exceptions

The ARM processor can be interrupted by seven basic exceptions:

- **Reset exception.** After a reset condition, the ARM7TDMI saves the current values of PC and CPSR, then changes the value of CPSR such that ARM mode is re-established. The system then fetches the next instruction from 0x00.
- **Undefined exception.** The ARM7TDMI takes the undefined instruction trap when it encounters an instruction it cannot handle. This exception can extend either the Thumb or ARM instruction set by software emulation.
- **SWI exception.** The software interrupt instruction (SWI) is used to enter supervisor mode, usually to request a particular supervisor instruction.
- **Abort exception.** An abort indicates that the current memory access cannot be completed. There are two types of abort exception:
  - **Prefetch.** Occurs during an instruction prefetch.
  - **Data.** Occurs during a data operand access.
- **IRQ.** The interrupt request (IRQ) exception is a normal interrupt sourced by the NET+50 interrupt controller.
- **FIRQ.** The fast interrupt request (FIRQ) exception supports a data transfer or channel process. Only the GEN module timers and GEN module watch-dog timer can generate a FIRQ interrupt.

See Appendix A, "ARM Exceptions" for more information about these exceptions.

Exception priorities

Several exceptions can arise at the same time. If this happens, 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 the same time.

- Undefined instructions and SWIs 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 the FIRQ is enabled (that is, the CPSR F flag is clear), the data abort takes priority. ARM7TDMI enters the data abort handler and then immediately proceeds 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.

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 exists and 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 + 50 interrupt controller
0x1C	FIRQ	Fast interrupt from NET + 50 interrupt controller

Table 21: Exception vector table

All internal NET+50 peripheral interrupts are presented to the CPU using the IRQ or FIRQ interrupt inputs. The ARM can mask various NET+50 peripheral interrupts at the global level, using the NET+50 interrupt controller. The ARM also can 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+50 interrupt controller can assert the IRQ input to the ARM processor.

When entering an interrupt service routine (ISR), the bit is automatically set by the ARM processor. This disables recursive interrupts. The ISR's first task is to read the interrupt status register, which identifies all active sources for the IRQ interrupt. The firmware, at bootup, sets the priorities for servicing interrupts, using the bits defined in the interrupt status register.

Entering and exiting an exception (software action)

The ARM7TDMI processor performs specific steps when handling exceptions.

Entering an exception

When an ARM exception occurs, the processor:

1. Preserves the address of the next instruction in the appropriate link register, irrespective of the state from the exception is entered.
   - If the exception has been entered from ARM state, the address of the next instruction is copied into the link register. The address is calculated as:

     `current PC + 4` or `current PC + 8` depending on the exception — see Table 22, "Exception entry/exit," on page 51).
   - If the exception has been entered from Thumb state, the value written into the link register is the current PC (program counter), offset by a value that lets the program resume from the correct place on return from the exception.
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. Sets the I (for IRQ interrupts) or F (for FIRQ interrupts) bits to disable interrupts to prevent unmanageable nesting of exceptions.

Note: If the processor is in Thumb state when an exception occurs, it automatically switches into ARM state when the PC is loaded with the exception vector address.

Exiting an Exception

On completion, the ARM processor:

1. Moves the link register, minus the offset where appropriate, to the PC. The offset value varies 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.

An explicit switch back to Thumb state is never needed. Restoring the CPSR from the SPSR automatically sets the T bit to the value it held immediately before the exception.

Exception entry / exit summary

Table 22 summarizes the PC values preserved in the relevant link registers on exception entry and provides the recommended instruction for exiting the exception handler.

Return exception	Return instruction	Previous state ARM R14[a]_x[b]	Previous state Thumb R14[c]_x[d]	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_abt, #4	PC + 4	PC + 4	2
FIRQ	SUBS PC, R14_abt, #4	PC + 4	PC + 4	2

Table 22: Exception entry/exit

a.R14 = Link Register
b._x = The previous state of the processor
c.R14 = Link Register
d._x = The previous state of the processor

Notes:

1 Where PC is the address of the BL/SWI/undefined instruction fetch that had the prefetch abort. BL is a "branch with link" instruction.

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.

Hardware interrupts

There are two wires that go into the ARM7CPU core that can interrupt the processor:

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

Although the interrupts are basically the same, FIRQ can interrupt IRQ.

The FIRQ and IRQ Lines

The FIRQ line adds a simple, two-tier priority scheme to the interrupt system. Most sources of interrupts on the NET+50 come from the IRQ line. The only potential sources for FIRQ interrupts on the NET+50 are the two built-in timers and the watch-dog timer; there is no way to generate a FIRQ signal externally. These timers are controlled by registers in the GEN module (see "Timing Registers," beginning on page 69):

- The built-in timers are controlled using the timer control registers (`0xFFB0 0010/18`). The corresponding bit in the Interrupt Enable register must be set for either IRQ or FIRQ to function.
- The watch-dog timer is controlled using the System Control register (`0xFFB0 0000`).

Interrupts can come from many different sources on the NET+50. The GEN module interrupt controller manages the interrupts (see "Interrupt generation and control," beginning on page 93, for information). Interrupts are enabled/disabled on a per-source basis using the Interrupt Enable Register (`0xFFB0 0030`). This register serves as a mask for the interrupt sources and ultimately controls whether an interrupt from a NET+50 module can reach the IRQ line.

Read-only registers

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

- **The Interrupt Status Register Raw (`0xFFB0 0038`).** Indicates the source of a NET+50 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.
- **The Interrupt Status Register Enabled (`0xFFB0 0034`).** Identifies the current state of all interrupt sources that are enabled. The register is defined by performing a logical AND of the Interrupt Status Register Raw and the Interrupt Enable Register. All the bits in the Interrupt Status Register Enabled are ORed together and the output is fed directly to the IRQ line, which then interrupts the ARM.

See "Interrupt generation and control," beginning on page 93, for more information about these registers.

Interrupt sources

Interrupt sources include the following:

- **DMA interrupts.** DMA channels 1-10, including the four sub-channels of the Ethernet receiver. (See Chapter 9, "DMA Controller Module.")
- **MIC interrupts.** The MIC can be operated in one of two modes: IEEE 1284 or ENI host mode. Interrupts are handled differently for each mode. (See Chapter 12, "MIC Controller Module.")
- **Ethernet receive and transmit.** These interrupts are used only when the Ethernet receiver/transmitter are in interrupt mode instead of DMA mode. Ethernet interrupts are part of the Ethernet General Status register (`0xFF80 0004`). (See Chapter 10, "Ethernet Controller Module.")
- **Serial interrupts.** There are many sources for serial interrupts. (See Chapter 11, "Serial Controller Module.")
- **Watch-dog timer interrupts.** When the watch-dog timer expires, the system can generate either an IRQ interrupt, a FIRQ interrupt, or a system reset. (See the discussion of the Timer registers in Chapter 6, "The GEN Module.")

- **Timer 1 and Timer 2 interrupts.** Two types of interrupts can be generated by Timers 1 and 2. The type of interrupt is configured in the timer control register (`0xFFB0 0010/18`) while the interrupt itself is contained within the Timer Status register (`0xFFB0 0014/1C`). See the discussion of the Timer Control and Timer Status registers in Chapter 6, "The GEN Module.")
- **PORTC interrupts.** The lower four pins of PORTC on the NET+50 can be used as interrupt sources (C3, C2, C1, C0). (See the discussion of the PORTC register in Chapter 6, "The GEN Module.")

Each of these sources is enabled/disabled within their respective module (and sub-modules) within the NET+50 ASIC; the interrupt controller in the GEN module, however, does not latch any of the interrupt signals.

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

BBus Module

CHAPTER 5

This chapter describes the BBus module, which provides the data path between the NET+50 internal modules.

Note: The information in this chapter applies to both the NET+50 and NET+20M chips, unless otherwise noted.

BBus functionality

BBus functionality includes:

- Address and multiplexing logic that supports the data flow through the NET+50 chip.
- Central arbiter for all NET+50 bus masters. The CPU, DMA, MIC, and BUS modules are all potential BBus masters. The BBus arbiter prevents one bus master from obtaining consecutive back-to-back cycles while another bus master is waiting. See "Cycles and BBus arbitration" on page 56 for more information.
- Internal register decoding for all addressable modules. Each module is given a small portion of the system address map for configuration and status. See "Address decoding" on page 57.

Cycles and BBus arbitration

BBus masters are given ownership in a round-robin manner. As a rule, if a potential bus does not require the BBus resources when its turn comes around, that bus is skipped until the next round-robin slot.

During a normal (normal operand) cycle, each bus master cycle is allowed only one read/write cycle if another bus master is waiting. There are two exceptions to this rule: burst transactions and read-write-modify transactions.

Order of arbitration

The order of arbitration between the bus masters is CPU → DMA → ENI → BUS. The BBus arbiter maintains the top-level round-robin arbitration loop; the DMA controller maintains its own round-robin arbitration loop for the different DMA channels.

You should expect an access pattern similar to this:

```
[CPU]  [DMA]  [MIC]  [CPU]  [DMA]  [MIC]  ...
```

This pattern expands as shown:

```
[CPU]  [ERXDMA]  [MIC]  [CPU]  [ETXDMA]  [MIC]  ...
```

Note that the expanded pattern is the ideal representation. The DMA channel is governed by two mechanisms: the internal arbitration mechanism and the context-switching mechanism. The context-switching mechanism incurs a penalty of 13 clock cycles to switch from one channel to another, which can cause delays. The access pattern is more likely to appear as:

```
[CPU]  [ERXDMA]  [MIC]  [CPU]  [MIC]  [CPU]  [ETXDMA] ...
```

To properly budget the bandwidth between various modules, you must recognize the limitations placed on the bus by arbitration algorithms and context-switch delays. The sizes of the peripheral FIFOs have an impact on budgeting also, as does control of burst access modes, CPU cache, and DMA duplex settings.

Address decoding

The CPU address map is divided to allow access to the various internal modules and external resources routed through the internal peripherals, as shown in Table 23.

Address range	Module
0x0000 0000 - 0xFF7F FFFF	BUS module
0xFF80 0000 - 0xFF8F FFFF	EFE module
0xFF90 0000 - 0xFF9F FFFF	DMA module
0xFFA0 0000 - 0xFFAF FFFF	MIC module
0xFFB0 0000 - 0xFFBF FFFF	GEN module
0xFFC0 0000 - 0xFFCF FFFF	MEM module
0xFFD0 0000 - 0xFFDF FFFF	SER module
0xFFF0 0000 - 0xFFFF FFFF	Cache RAM

Table 23: BBus address decoding

All resources defined in this manner are addressed using the upper memory addresses. Each internal module is given 1 Mbyte of address space for its own internal decoding. Each module defines its own specific register map.

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

The GEN Module

CHAPTER 6

The GEN (General) module provides the NET+50 with its main system control functions and an interrupt priority controller, as well as miscellaneous functions:

- Two programmable timers with interrupt support
- One programmable bus-error timer
- One programmable watch-dog timer
- Three 8-bit programmable parallel I/O ports with interrupt support

Note: The information in this chapter applies to both the NET+50 and NET+20M chips, unless otherwise noted.

Module configuration

The GEN module is configured as follows:

Address	Register
0xFFB0 0000	System Control register
0xFFB0 0004	System Status register
0xFFB0 0008	PLL Control register
0xFFB0 000C	Software Service register
0xFFB0 0010	Timer 1 Control register
0xFFB0 0014	Timer 1 Status register
0xFFB0 0018	Timer 2 Control register
0xFFB0 001C	Timer 2 Status register
0xFFB0 0020	PORTA register
0xFFB0 0024	PORTB register
0xFFB0 0028	PORTC register
0xFFB0 0030	Interrupt Enable register
0xFFB0 0034	Interrupt Enable register – SET
0xFFB0 0038	Interrupt Enable register – CLEAR
0xFFB0 0034	Interrupt Status register – Enabled
0xFFB0 0038	Interrupt Status register – Raw

Table 24: GEN module address map

A note about interrupts

The type of interrupt produced and the length of the watch-dog timer is configured using the System Control register. The watch-dog is strobed using the Software Service register. The corresponding bit in the Interrupt Enable register must be set for either IRQ or FIRQ to function. See the appropriate discussions for more information.

GEN module hardware initialization

Many internal NET+50 configuration features are application-specific and need to be configured at powerup before the CPU begins executing boot-up code. System bus address bits are used to do this during a powerup reset. The NET+50 provides internal pull-up resistors on all address lines. Weak external pull-down resistors can be used to configure internal register bits to a zero state.

Table 25 specifies which system bus address bits control which GEN module functions.

Address bit	Function	Setting
ADDR27	GEN_LENDIAN	0 - Little Endian configuration 1 - Big Endian configuration **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.
ADDR26	GEN_BUSER	Must be set to 1 — ARM CPU enabled; GEN_BUSER set to 0.
ADDR25	GEN_IARB	Must be set to 1 — internal system bus arbiter.
ADDR19:09	GEN_ID	Product identification code

Table 25: ADDR bits/function control

NET + 50 chip bootstrap initialization

Many internal NET+50 chip configuration features are application-specific and need to be configured at powerup before the CPU boots. The system bus address bits are used for this purpose during a powerup reset. The NET+50 provides internal pull-up resistors on all address signals. Weak external pull-down resistors can be used to configure different internal register bits to a zero state.

Table 26 specifies which ADDR bits control which functions during bootstrap initialization:

Address bit	Function
ADDR[27]	Endian configuration ■ 0 - Little Endian configuration ■ 1 - Big Endian configuration The inverted ENDIAN bit (ADDR7) is loaded into the LENDIAN bit within the system control register, where: ■ LENDIAN = 1 (for Little Endian mode) ■ LENDIAN = 0 (for Big Endian mode)
ADDR[26]	CPU bootstrap. Must be set to 1.
ADDR[25]	GEN_IARB setting. Must be set to 1.
ADDR[24:23]	CS0 bootstrap setting ■ 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]	MICMODE[2:0] configuration bits
ADDR[19:09]	GEN_ID setting
ADDR[8]	Reserved
ADDR[07]	MIC control PSIO* ■ 0 = PSIO ■ 1 = Normal The inverted PSIO address bit (ADDR07) is loaded into the PSIO configuration bit within the ENI Control register, where: ■ 0 = Normal ■ 1 = PSIO)
ADDR0[6]	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*

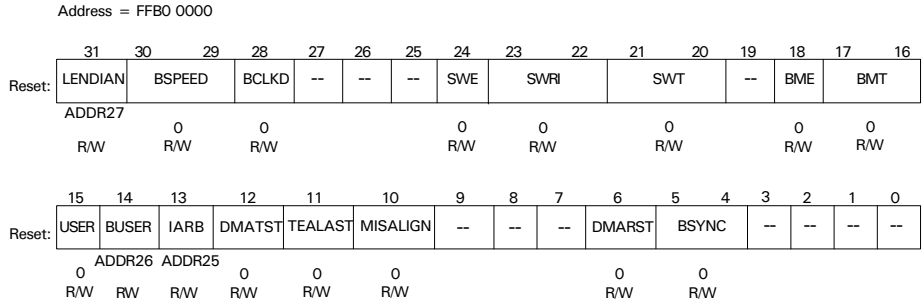
Table 26: GEN module ADDR bit function control

System registers

The GEN module uses two system registers:

- System Control register. 32-bit read/write register. All control bits are active high unless an asterisk (*) appears in the signal name; an asterisk indicates active low. All control bits are set to their respective inactive state on reset.
- System Status register. 32-bit register. All bits in this register, except LOCK, are loaded only during a hardware reset. The GEN_ID value is not affected when an external jumper is changed — unless a hardware reset is executed first. The LOCK bit is dynamic and changes.

System Control register



Code (Bit #)	Definition of Code	Description
LENDIAN (D31)	Configure chip to run in Little Endian mode	0 - Configures the NET+50 to run in Big Endian mode 1 - Configures the NET+50 to operate in Little Endian mode Controls the Endian configuration for the NET+50 chip. During reset, the LENDIAN bit defaults to the value defined by the inverted ADDR27 bit (see "GEN module hardware initialization" on page 61).

Table 27: System Control register bit description

Code (Bit #)	Definition of Code	Description
BSPEED (D30:29)	BUS speed configuration	00 - 1/4 speed 01 - 1/2 speed 10 - Full speed 11 - Reserved Controls the speed of the system bus clock (BCLK) relative to the speed of the internal system clock (SYSCLK). BCLK can be configured to operate at 1/4 speed, 1/2 speed, or full speed.
BCLKD (D28)	BCLK output disable	0 - BCLK output enabled 1 - BCLK output forced to LOW state Shuts down the operation of the BCLK signal. Turning off the BCLK signal minimizes electro-magnetic interference (EMI) when the BCLK signal is not required for the application.
SWE (D24)	Software watch-dog enable	Set to 1 to enable the watch-dog timer circuit. The watch-dog timer can be configured, using SWRI, to generate an interrupt or reset condition if and when the watch-dog timer expires. Once the SWE bit is set to 1, only a hardware reset sets the bit back to 0.
SWRI (D23:22)	Software watch-dog reset/interrupt select	00 - Software watch-dog causes normal (IRQ) interrupt 01 - Software watch-dog causes fast (FIRQ) interrupt 10 - Software watch-dog causes reset 11 - Reserved 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 (D21:20)	Software watch-dog timeout (in seconds)	00 - $2^{20}/F_{XTAL}$ 01 - $2^{22}/F_{XTAL}$ 10 - $2^{24}/F_{XTAL}$ 11 - $2^{25}/F_{XTAL}$ Controls the timeout period for the watch-dog timer. The timeout period is a function of the F_{XTAL} value. See "Internal XTAL clock reference" on page 99 for information about the F_{XTAL} value.

Table 27: System Control register bit description

Code (Bit #)	Definition of Code	Description
BME (D18)	Bus monitor enable	0 - Disable bus monitor operation 1 - Enable bus monitor operation Set to 1 to enable the bus monitor timer. Required to avoid a system lockup condition that can occur when a bus master attempts to address a memory space that is not decoded by any peripheral. The bus monitor timer detects when a bus master is accessing a peripheral and there is no transfer acknowledge (TA*) response. 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.
BMT (D17:16)	Bus monitor timer	00 - 128 BCLKs 01 - 64 BCLKs 10 - 32 BCLKs 11 - 16 BCLKs Controls the timeout period for the bus monitor timer. The BMT field generally is set to its maximum value, but it 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 (D15)	Enable access to internal chip registers in CPU user mode	Controls whether applications operating in ARM user mode (as opposed to supervisor mode) can access internal registers within the NET+50 chip. ■ If the USER bit is set to 0 and an application, operating in ARM user mode, tries to access (read or write) an internal NET+50 register, the application receives a data abort. ■ When the USER bit is set to 1, any application can access the NET+50 internal registers.
BUSER (D14)	Enable ARM CPU	Must be set to 0. During reset, BUSER defaults to the value defined by ADDR26 (see "GEN module hardware initialization" on page 61).
IARB (D13)	Define system bus to internal arbiter	Must be set to 1. During reset, IARB defaults to the value defined by ADDR25 (see "GEN module hardware initialization" on page 61).

Table 27: System Control register bit description

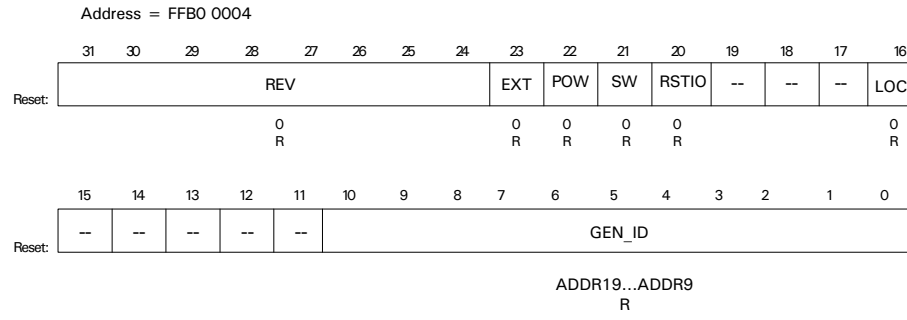
Code (Bit #)	Definition of Code	Description
DMATST (D12)	DMA module test mode	0 - Test mode disabled 1 - Allow unrestricted access to DMA context RAM Resets the DMA controller subsystem. Also used to allow the ARM processor direct access to the internal context RAM found in the DMA controller. ■ When the DMATST bit is set to 1, the DMA controller subsystem is held in reset and the ARM processor can access all the internal DMA context RAM. This is useful for diagnostic purposes. ■ When the DMATST bit is set to 0, the DMA controller subsystem operates normally. Only the bits in the DMA control register space can be accessed by the ARM processor.
TEALAST (D11)	Bus interface TEA/LAST configuration	0 - Use TEA_pin for error indication only 1 - Use TEA_ pin for LAST word of burst sequence indicator and error indication Determines how the NET+50 will use the TEA* signal. ■ When TEALAST is set to 0, the TEA* signal is used only to indicate a data abort. ■ When TEALAST is set to 1, the TEA* signal is used in conjunction with the TA* signal to indicate either a data abort or a burst completion.
MISALIGN (D10)	Bus error on misaligned cycles	0 - Disable misaligned data transfer bus abort generation 1 - Generate a bus abort during a misaligned transfer When set to 1, misaligned address transfers cause a data abort to be issued to the offending bus master. A misaligned 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 bit is useful during software debugging to detect misaligned cycles.

Table 27: System Control register bit description

Code (Bit #)	Definition of Code	Description
DMARST (D06)	DMA module reset	0 - DMA module operational 1 - DMA module reset Provides a way to issue a soft reset to the DMA module without affecting any other modules. ■ Setting the DMARST bit to 1 causes the DMA module to be held in reset. ■ Setting the DMARST bit to 0 allows the DMA module to operate.
BSYNC (D05:04)	TA* input synchronizer	00 - 1-stage synchronizer 01 - 1-stage synchronizer 10 - 2-stage synchronizer 11 - No synchronizer Defines the level of synchronization performed within the NET+50 chip for the TA* input. The NET+50 can process the TA* input signal using a 1-stage flip-flop synchronizer, a 2-stage synchronizer, or no synchronizer. The 1- or 2-stage synchronizer must be used if the TA* input is asynchronous to the BCLK signal. The 2-stage synchronizer is better, however, as it introduces one additional BCLK of latency in the access cycle. If no synchronizer is used, the TA* input must be synchronous to the BCLK signal and may not operate at or near 44 MHz. This level is not recommended.

Table 27: System Control register bit description

System Status register



Code (Bit #)	Definition of code	Description
REV (D31:24)	NET+ARM chip revision ID	Identifies the version of the NET+ARM chip and its revision. All NET+ARM chips have a REV identification (see Table 29 on page 68.)
EXT (D32)	Last reset caused by external reset	Indicates that the RESET* pin was used to cause the last hardware reset condition. This bit is set/reset during every hardware reset condition.
POW (D22)	Last reset caused by powerup reset	Indicates that a powerup reset caused the last hardware reset condition. The POW bit is set/reset during every hardware reset condition.
SW (D21)	Last reset caused by software watch-dog timer	Indicates that a software watch-dog timeout caused the last hardware reset condition. The SW bit is set/reset during every hardware reset condition.
RSTIO (D20)	Last reset caused by ENI RSTIO control	Indicates that 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 (D16)	System PLL in Lock state	Indicates that the internal phase lock loop (PLL) is currently in a Lock condition.
GEN_ID (D10:00)	Product ID defined by external resistor jumpers	During reset, defaults to the value defined by ADDR19:09 bits (see "GEN module hardware initialization" on page 61). The GEN_ID is loaded only once during a hardware reset condition.

Table 28: System Status register bit description

NET+50 chip device	Rev IDs
NET+5 & 10	0x00
NET+12-1	0x01
NET+15-0	0x04
NET+15-1	0x05

Table 29: NET+ARM chip devices and revision IDs

NET+50 chip device	Rev IDs
NET+15-2	0x06
NET+15-3	0x07
NET+15-4	0x07
NET+40-0	0x08
NET+40-1	0x09
NET+40-2	0x0A
NET+40-3	0x0B
NET+40-4	0x0B
NET+50-1	0x19
NET+50-3	0x1B

Table 29: NET+ARM chip devices and revision IDs

PLL

PLL comes up for hardware reset to allow the internal system clock (SYSCLK) to run at 44.3MHz. If you need to use a different frequency for SYSCLK, you must use a different external configuration; that is a crystal with a different output frequency and/or an external oscillator.

Timing Registers

The NET+50 uses four timing registers:

- Software Service register (SWSR). Acknowledges the system watch-dog timer. To do so, 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, but the two operations must occur in the proper sequence with the proper data values.

  The SWSR is also used to request a software reset of the NET+50 chip hardware. To request a soft reset, firmware must write $123 and $321 to

the SWSR using two separate write operations. Again, there is no restriction on the time between the two operations and the two operations must occur in the proper sequence with the proper data values.

- **Timer Control registers** (Timer 1 and Timer 2). 32-bit register(s) used to provide the CPU with programmable interval timers. The timers operate using the external crystal clock and an optional 9-bit prescaler. The timers provide a 27-bit programmable down-counter mechanism. An initial count register is loaded by the CPU to define the timeout 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.

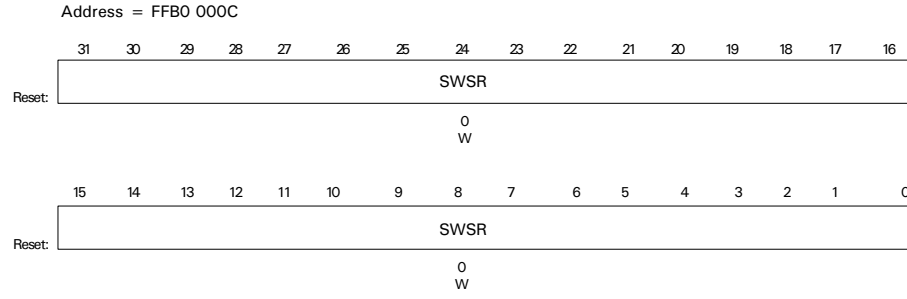
  The effective value of the timeout is determined by these equations:

  ```
  TIMEOUT = [8 * (TC + 1)] / FXTAL          TCLK = 0; TPRE = 0
  TIMEOUT = [4096 * (TC + 1)] / FXTAL       TCLK = 0; TPRE = 1
  TIMEOUT = [8 * (TC + 1)] / FSYSCLK        TCLK = 1; TPRE = 0
  TIMEOUT = [4096 * (TC + 1)] / FSYSCLK     TCLK = 1; TPRE = 1
  ```

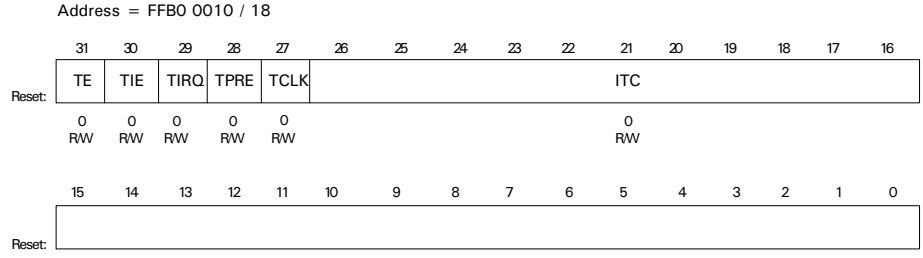
  All bits are set to 0 on reset.

- **Timer Status register.** 32-bit read-only register. The TIP (Timer Interrupt Pending) bit is set when the current timer count register decrements to zero. If the TIE (Timer Interrupt Enabled) bit is set in the Timer Control register, the CPU is interrupted. The TIP bit is cleared by writing a 1 to the respective bit position on this register.

Software Service register



Timer Control registers



Code (Bit #)	Definition of code	Description
TE (D31)	Timer enable	Must be set to 1 to allow the timer to operate. Setting the TE bit to 0 resets the timer and prevents it from operating. The other fields in this register should be configured before or during the same memory cycle in which TE is set to 1.
TIE (D30)	Timer interrupt enable	Can be set to 1 to allow the timer to interrupt the CPU. A timer interrupt is generated when the TIP bit is set in the timer status register.
TIRQ (D29)	Timer interrupt mode	0 – Normal interrupt 1 – Fast interrupt Controls the type of interrupt the timer will trigger. The Timer can generate either a normal interrupt (ARM IRQ pin) or a fast interrupt (ARM FIRQ pin).
TPRE (D28)	Timer prescaler	0 – Disable 9-bit prescaler 1 – Enable 9-bit prescaler Determines whether the 9-bit prescaler is used in calculating the TIMEOUT parameter. Using the prescaler allows for longer values of TIMEOUT.
TCLK (D27)	Timer clock source	0 – Use F_{XTAL} as timer clock source 1 – Use F_{SYSCLK} as timer clock source Selects the reference clock for the timer module. The reference clock can be based off the F_{XTAL} clock or the F_{SYSCLK} clock.

Table 30: Timer Control register bit definition

Code (Bit #)	Definition of code	Description

ITC (D26:00)	Initial timer count	Defines the TIMEOUT parameter for interrupt frequency. The TIMEOUT period is a function of the F_{XTAL} frequency. (See *"System clock generation" on page 97*.) The effective value of the timer TIMEOUT is determined by these equations:
		`TIMEOUT = [8 * (TC + 1)] / FXTAL  TCLK = 0; TPRE = 0`
		`TIMEOUT = [4096 * (TC + 1)] / FXTALTCLK = 0; TPRE = 1`
		`TIMEOUT = [8 * (TC + 1)] / FSYSCLKTCLK = 1; TPRE = 0`
		`TIMEOUT = [4096 * (TC + 1)] / FSYSCLKTCLK = 1; TPRE = 1`
		Using an 18.432 crystal as an example, Table 31 identifies the possible minimum and maximum values of the timers.

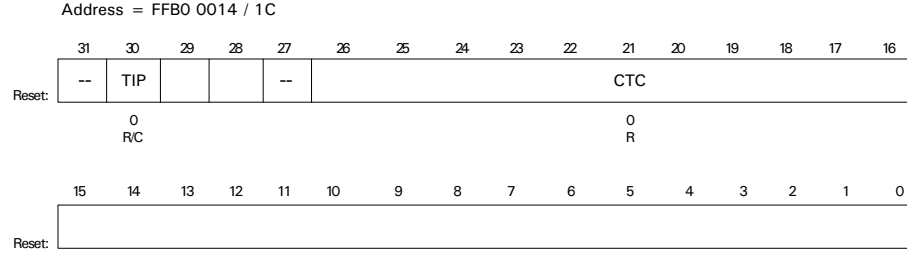
Table 30: Timer Control register bit definition

$F_{XTAL} = F_{CRYSTAL} / 5$
$F_{XTAL} = 18432000 / 5 = 3686400$ Hz

TPRE = 0		TPRE = 1	
ITC = 0	ITC = 511	ITC = 0	ITC = 511
2 μS	1.1 mS	1.1 mS	568 mS

Table 31: Minimum and maximum timer values

Timer Status register



Code (Bit #)	Definition of code	Description
TIP (D30)	Timer interrupt pending	Set to 1 when the timer is enabled and the CTC value reaches 0. The TIP can be used to generate an interrupt to the CPU as long as the TIE bit is set in the Timer Control register. Writing a 1 to the same bit position in this register clears the TIP bit. **Note:** The TIP bit is immediately set when the TE bit is changed from 0 to 1. This is because the CTC field initially starts out at zero, which means an interrupt occurs 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.
CTC (D26:00)	Current timer count	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.

Table 32: Timer Status register bit definition

General Purpose I/O (GPIO) Registers

The GEN module provides registers to configure the personality of each set of general purpose I/O pins:

- PORTA register. Configures each of the eight PORTA pins. All pins can be configured for general purpose I/O, as well as for special functions.
- PORTB register. Configures each of the eight PORTB pins. All pins can be configured for general purpose I/O as well as for special functions.
- PORTC register. Configures each of the eight PORTC pins. All pins can be configured for general purpose I/O, as well as for special functions.

Important: The NET+50 chip uses all 24 GPIO port pins. The NET+20M chip uses only 16 pins. You can use any combination of pins from the three ports, in both GPIO mode and special function mode, as long as you use only 16 pins.

PORTA Register

You can program each of the PORTA GPIO pins to be one of the following, as applicable:

- General purpose input
- General purpose output
- Special function input
- Special function output

Table 33: "PORTA configuration" describes the possible configurations for each of the PORTA signals. Each column in the table denotes one of the four possible configurations for each bit. If there is no entry in one of the columns (for example, PORTA7>AMODE=1>ADIR=0), the bit cannot be used in that configuration.

The MODE field in the PORTA register (see page 78) determines which PORTA bits are configured for GPIO and which are configured for special function. The DIR field in the PORTA register controls the direction of the configuration.

- When the MODE bit is set to 0, the bit is configured for GPIO.
- When the MODE bit is set to 1, the bit is configured for special function.
- When the DIR bit is set to 0, the bit is configured for input mode.
- When the DIR bit is set to 1, the bit is configured for 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-A* RXCA-IN	SPI-M-CLK-OUT-A RXCA-OUT OUT1A*
PORTA3	GPIO IN	GPIO OUT	RXDA	

Table 33: PORTA configuration

PORTA Bit	GPIO mode		Special function mode	
	AMODE = 0		AMODE = 1	
	ADIR = 0	ADIR = 1	ADIR = 0	ADIR = 1
PORTA2	GPIO IN	GPIO OUT	DSRA*	DACK1*
PORTA1	GPIO IN	GPIO OUT	CTSA*	
PORTA0	GPIO IN	GPIO OUT	DCDA* DONE-IN1*	DONE-OUT1*

Table 33: PORTA configuration

General purpose I/O

The bit is configured for GPIO when the MODE bit is set to 0. The DIR setting then determines whether the bit is input or output mode.

- **When both MODE and DIR are set to 0, the PORTA bit is configured in input mode.** The NET+50 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+50 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 returns the current setting of the DATA field.

Special function mode

Special function mode is used primarily for connecting the internal NET+50 serial ports to the external interface pins. This mode provides other features as well.

- When the MODE bit is set to 1 and the DIR bit is set to 0, the PORTA bit is configured to operate in special function *input* mode.
- When both the MODE bit and the DIR bit are set to 1, the PORTA bit is configured to operate in special function mode.

PORTA7

Provides the transmit data output function for serial port A. The TXDA signal is used for serial transmit data when configured to operate in UART, SPI, or HDLC modes.

PORTA6

- Special function *input*: Provides the active low DMA request input for DMA channel 3 or 5 when that DMA channel is configured for external channel sourcing. Used only when the REQ bit is set to 1 in the DMA Channel 3 or DMA Channel 5 Control register.
- Special function *output*: Provides the data terminal ready (DTR) control signal output for serial port A. Control Register A in the SER module configuration controls the state of DTR. The DTR signal configuration is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion before driving the PORTA6 I/O pad.

PORTA5

Provides the request to send (RTS) control signal for serial port A. Control Register A in the SER module configuration controls the state of RTS. The RTS signal configuration is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion before driving the PORTA5 I/O pad.

PORTA4

- Special function *input:* Provides one of two serial channel A features, depending on 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, the serial channel receiver clock (RXCA) will be accepted from the PORTA4 pin.
- Special function *output:* Provides one of three serial channel A features, depending on the configuration of the serial channel.
  - When the serial channel is configured for SPI master mode, PORTA4 special function output drives the SPI master clock signal out the PORTA4 pin.

- When the serial channel is not configured for SPI Master mode and the RXEXT bit in the Serial Channel Bit-Rate register is set to 1, 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 in the Serial Channel Bit-Rate Register is set to 0, the serial channel general purpose OUT1 signal will be driven out the PORTA4 pin. Control Register A in the SER module configuration controls the state of OUT1. The OUT1 signal configuration is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion before driving the PORTA4 I/O pad.

PORTA3

When configured for special function input, the PORTA3 signal provides the receive data (RXD) signal for serial port A. The RXD signal is used for serial receive data when configured to operate in UART, SPI, or HDLC modes.

PORTA2

- Special function *input*: Provides the data set ready (DSR) signal for serial port A. Status Register A in the SER module configuration returns the state of RTS. The RTS signal status is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion after receiving the PORTA2 I/O pad.
- Special function *output*: Provides the active low DMA acknowledge (DACK1*) signal for DMA channel 3 or 5. DACK1 is driven active low when DMA channel 3 or 5 is performing an external DMA cycle. DMA acknowledge is used only when the REQ bit in the DMA Channel 3 or DMA Channel 5 Control register is set.

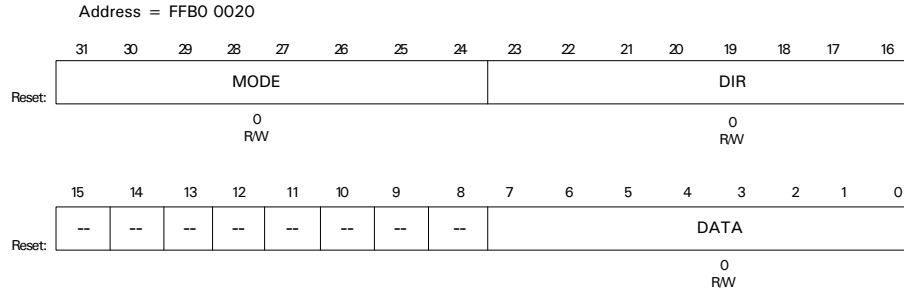
PORTA1

Provides the clear to send (CTS) signal for serial port A. Status Register A in the SER module configuration returns the state of CTS. The CTS signal status is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion after receiving the PORTA1 I/O pad.

PORTA0

- Special function *input*: Provides the data carrier detect (DCD) signal for serial port A, as well as the active low DMA DONE signal for DMA channel 3 or 5.
  - **DCDA*.** Status register A in the SER module configuration returns the state of DCD. The DCD signal status is active high inside the NET+50 and active low outside the NET+50. 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 in the SER module must be ignored.
  - **DONE1*.** The DMA DONE signal is driven active low by an external peripheral when DMA channel 3 or 5 is performing an external DMA cycle and this cycle represents the final transfer for a given block from the external peripheral. DMA DONE is used only when the REQ bit is set in the DMA Channel 3 or DMA Channel 5 Control register. The DMA DONE input causes the current DMA buffer descriptor to close and generates an interrupt to the software (under program control).

PORTA register and bit definitions



Code (Bit #)	Definition of code	Description
MODE (D31:24)	PORTA mode configuration	Used to individually configure each of the PORTA pins to operate in either GPIO mode or special function mode. 0 – Selects GPIO mode 1 – Selects special function mode Each bit in the MODE field corresponds to one of the PORTA bits. D31 controls PORTA7, D30 controls PORTA6, and so on.
DIR (D23:16)	PORTA data direction	Used to individually configure each of the PORTA pins to operate in either input mode or output mode. 0 – Selects input mode 1 – Selects output mode Each bit in the DIR field corresponds to one of the PORTA bits. D23 controls PORTA7, D22 controls PORTA6, and so on.
DATA (D07:00)	PORTA data register	Used when a PORTA bit is configured to operate in GPIO mode. ■ Reading the DATA field provides the current state of the GPIO signal, regardless of its configuration mode. ■ Writing the DATA field defines the current state of the 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 PORTA bits. D07 controls PORTA7, D06 controls PORTA6, and so on.

Table 34: PORTA register bit definition

PORTB Register

You can program each of the PORTB GPIO pins to be one of the following, as applicable:

- General purpose input
- General purpose output
- Special function input

- Special function output

Table 35: "PORTB configuration" describes the possible configurations for each of the PORTB signals. Each column in the table denotes one of the four possible configurations for each bit. If there is no entry in one of the columns (for example, PORTB7>BMODE=1>BDIR=0), the bit cannot be used in that configuration.

The MODE field in the PORTB register (see page 84) determines which PORTB bits are configured for GPIO and which are configured for special function. The DIR field in the PORTB register controls the direction of the configuration.

- When the MODE bit is set to 0, the bit is configured for GPIO.
- When the MODE bit is set to 1, the bit is configured for special function.
- When the DIR bit is set to 0, the bit is configured for input mode.
- When the DIR bit is set to 1, the bit is configured for 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	REJECT*	RTSB*
PORTB4	GPIO IN	GPIO OUT	SPI-S-CLK-IN-B* RXCB-IN	SPI-M-CLK-OUT-B* RXCB-OUT OUT1B*
PORTB3	GPIO IN	GPIO OUT	RXDB	
PORTB2	GPIO IN	GPIO OUT	DSRB*	DACK2*
PORTB1	GPIO IN	GPIO OUT	CTSB*	RPSF*
PORTB0	GPIO IN	GPIO OUT	DCDB* DONE-IN2*	DONE-OUT2*

Table 35: PORTB configuration

General purpose I/O

Remember: PORTB GPIO pins can be configured *only* in the NET+50 chip. If you are using the NET+20M chip, only the special function pins can be configured.

The bit is configured for GPIO when the MODE bit is set to 0. The DIR setting then determines whether the bit is input or output mode.

- **When both MODE and DIR are set to 0, the PORTB bit is configured in input mode.** The NET+50 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 PORTA bit is configured in output mode.** The NET+50 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 returns the current setting of the DATA field.

Special function mode

Special function mode is used primarily for connecting the internal serial ports to the external interface pins. This mode provides other features as well.

- When the MODE bit is set to 1 and the DIR bit is set to 0, the PORTB bit is configured to operate in special function *input* mode.
- When both the MODE bit and the DIR bit are set to 1, the PORTB bit is configured to operate in special function *output* mode.

PORTB7

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.

PORTB6

- Special function *input:* Provides the active low DMA request (DREQ*) input signal for DMA channel 4 or 6 when that DMA channel is configured for external channel sourcing. Used only when the REQ bit is set to 1 in the DMA Channel 4 or DMA Channel 6 Control Register.

- Special function *output:* Provides the data terminal ready (DTR) signal for serial port B. Control Register A in the SER module configuration controls the state of DTR. The DTR signal configuration is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion before driving the PORTB6 I/O pad.

PORTB5

- Special function *input:* Can be configured as the special function REJECT input used by an external Ethernet address filtering block to force the internal NET+50 chip Ethernet MAC to reject the current incoming data packet. See Chapter 10, "Ethernet Controller Module" for more information.
- Special function *output:* Provides the request to send (RTS) signal for serial port B. Control Register A in the SER module configuration controls the state of RTS. The RTS signal configuration is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion before driving the PORTB5 I/O pad.

PORTB4

- Special function *input:* Provides one of two serial channel B features, depending on 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, the serial channel receiver clock (RXCB) will be accepted from the PORTB4 pin.
- Special function *output:* Provides one of three serial channel B features, depending on the configuration of the serial channel.
  - When the serial channel is configured for SPI master mode, PORTB4 special function output drives the SPI master clock signal out the PORTB4 pin.
  - When the serial channel is not configured for SPI master mode and the RXEXT bit in the Serial Channel Bit-Rate Register is set to 1, 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 in the Serial Channel Bit-Rate Register is set to 0, the serial

channel general purpose OUT1 signal will be driven out the PORTB4 pin. Control Register A in the SER module configuration controls the state of OUT1. The OUT1 signal configuration is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion before driving the PORTB4 I/O pad.

PORTB3

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

- Special function *input:* Provides the data set ready (DSR) signal for serial port B. Status Register A in the SER module configuration returns the state of RTS. The RTS signal status is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion after receiving the PORTB2 I/O pad.
- Special function *output:* Provides the active low DMA acknowledge signal (DACK2*) for DMA channel 4 or 6. The DACK2 is driven active low when DMA channel 4 or 6 is performing an external DMA cycle. DMA acknowledge is used only when the REQ bit in the DMA Channel 4 or DMA Channel 6 Control register is set.

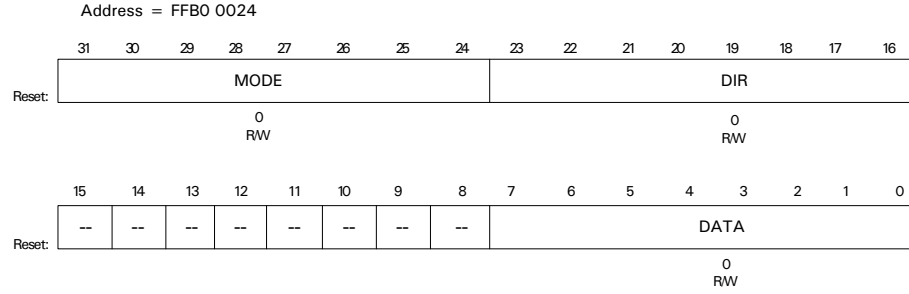
PORTB1

- Special function *input:* Provides the clear to send (CTS) signal for serial port B. Status Register A in the SER module configuration returns the state of CTS. The CTS signal status is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion after receiving the PORTB1 I/O pad.
- Special function *output:* Can be configured for special function RPSF* output used by external Ethernet address filtering logic to identify the 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+50's MII interface. See Chapter 10, "Ethernet Controller Module" for more information.

PORTB0

- Special function *input:* Provides the data carrier detect (DCD) signal for serial port B, as well as the active low DMA DONE input signal for DMA channel 4 or 6.
  - **DCDB*.** Status Register A in the SER module configuration returns the state of DCD. The DCD signal status is active high inside the NET+50 and active low outside the NET+50. 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 in the SER module must be ignored.
  - **DONE2*.** The DMA DONE signal is driven active low by an external peripheral when DMA channel 4 or 6 is performing an external DMA cycle and this cycle represents the final transfer for a given block from the external peripheral. DMA DONE is used only when the REQ bit is set in DMA Channel 4 or DMA Channel 6 Control register. The DMA DONE input causes the current DMA buffer descriptor to close and generates an interrupt to the software (under program control).

PORTB register and bit definitions



Code (Bit #)	Definition of code	Description
MODE (D31:24)	PORTB mode configuration	■ **NET+50.** Used to individually configure each of the PORTB pins to operate in either GPIO mode or special function mode. ■ **NET+20M.** Must be set to 1 for special function mode. Each bit in the MODE field corresponds to one of the PORTB bits. D31 controls PORTB7, D30 controls PORTB6, and so on.
DIR (D23:16)	PORTB data direction	Used to individually configure each of the PORTB pins to operate in either input mode or output mode. 0 – Selects input mode 1 – Selects output mode Each bit in the DIR field corresponds to one of the PORTB bits. D23 controls PORTB7, D22 controls PORTB6, and so on.
DATA (D07:00)	PORTB data register	Used when a PORTB bit is configured to operate in GPIO mode. ■ Reading the DATA field provides the current state of the GPIO signal, regardless of its configuration mode. ■ Writing the DATA field defines the current state of the 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 PORTB bits. D07 controls PORTB7, D06 controls PORTB6, and so on.

Table 36: PORTB register bit definition

PORTC register

You can program each of the PORTC GPIO pins to be one of the following, as applicable:

- General purpose input
- General purpose output
- Special function input
- Special function output

Table 37: "PORTC configuration" describes the possible configurations for each of the PORTC signals. Each column in the table denotes one of the four possible configurations for each bit. If there is no entry in one of the columns (for example, `PORTC4>CMODE=1>CDIR=1`), the bit cannot be used in that configuration.

The MODE field in the PORTC register (see page 92) determines which PORTC bits are configured for GPIO and which are configured for special function. The DIR field in the PORTC register controls the direction of the GPIO configuration.

- When the MODE bit is set to 0, the bit is configured for GPIO.
- When the MODE bit is set to 1, the bit is configured for special function.
- When the DIR bit is set to 0, the bit is configured for input mode.
- When the DIR bit is set to 1, the bit is configured for 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-ENABLE-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-ENABLE-B* TXCB-OUT OUT2B*

Table 37: PORTC configuration

PORTC BIT	GPIO Mode		Special Function Mode	
	CMODE = 0		CMODE = 1	
	CDIR = 0	CDIR = 1	CDIR = 0	CDIR = 1
PORTC4	GPIO IN	GPIO OUT	RIB*	
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 37: PORTC configuration

General purpose I/O

The bit is configured for GPIO when the MODE bit is set to 0. The DIR setting then determines whether the bit is input or output mode.

- **When both MODE and DIR are set to 0, the PORTC bit is configured in input mode.** The NET+50 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 1, the PORTC bit is configured in output mode.** The NET+50 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 returns the current setting of the DATA field.

Special function mode

Special function mode is used primarily for connecting the internal NET+50 serial ports to the external interface pins. This mode also provides other miscellaneous features.

- When the MODE bit is set to 1 and the DIR bit is set to 0, the PORTC bit is configured to operate in special function *input* mode.
- When both MODE and DIR are set to 1, the PORTC bit is configured to operate in Special Function Output Mode.

Interrupts

PORTC is the only register that enables, generates, and clears interrupts.

- To enable the interrupts, set the corresponding MODE bit to 1. The interrupts are *edge-sensitive*, and the transition can be selected using the DIR bit in the same register.
- To generate an interrupt on a high-to-low transition, set the DIR bit to 0. To generate a low-to-high transition, set the DIR bit to 1.

  For example, to enable C1 and C2 as low-to-high interrupts, write the following: `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. Be sure to use caution when using external interrupts and GPIO.

Note: Writing 0 to the least significant 4 bits in this register has no effect.

PORTC7

- Special function *input:* Provides one of 2 serial channel A features, depending on 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, the serial channel A transmit clock (TXCA) will be accepted from the PORTC7 pin.
- Special function *output:* Provides one of 3 serial channel A features, depending on the configuration of the serial channel.
  - When the serial channel is configured for SPI master mode, PORTC7 special function output drives the active low SPI master enable 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, the serial channel transmit clock (TXCA) 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, the serial channel general purpose OUT2 signal will be driven out the PORTC7 pin. Control Register A in the SER module configuration controls the

state of OUT2. The OUT2 signal configuration is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion before driving the PORTC7 I/O pad.

PORTC6

- Special function i*nput:* Provides the ring indicator (RI) signal for serial channel A. Status Register A in the SER module configuration returns the state of RI. The RI signal status is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion after receiving the PORTC6 I/O pad.
- Special function *output:* Provides the IRQ-OUT* signal. The IRQ-OUT* signal provides an active low interrupt request signal, which indicates that 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, and is typically used in environments where the NET+50 is used in the slave coprocessor configuration and the internal ARM processor is disabled during bootstrap.

PORTC5

- Special function *input:* Provides one of 2 serial channel B features, depending on 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, the serial channel B transmit clock (TXCB) will be accepted from the PORTC5 pin.
- Special function *output:* Provides one of 3 serial channel B features, depending on the configuration of the serial channel.
  - When the serial channel is configured for SPI master mode, PORTC5 special function output drives the active low SPI master enable 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, the serial channel transmit clock (TXCB) 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, the serial channel general purpose OUT2 signal will be driven out the PORTC5 pin. Control Register A in the SER module configuration controls the state of OUT2. The OUT2 signal configuration is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion before driving the PORTC5 I/O pad.

PORTC4

Provides the ring indicator (RI) signal for serial channel B. Status Register A in the SER module configuration returns the state of RI. The RI signal status is active high inside the NET+50 and active low outside the NET+50. The GEN module performs inversion after receiving the PORTC4 I/O pad.

PORTC3

When configured for special function mode, the PORTC3 signal becomes an edge-sensitive interrupt detector. When the corresponding DIR bit is set to 0, the PORTC3 pin detects high-to-low transitions on PORTC3 and asserts 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.

PORTC3 can also 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

When configured for special function mode, the PORTC2 signal becomes an edge-sensitive interrupt detector. When the corresponding DIR bit is set to 0, the PORTC2 pin detects high-to-low transitions on PORTC2 and asserts 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.

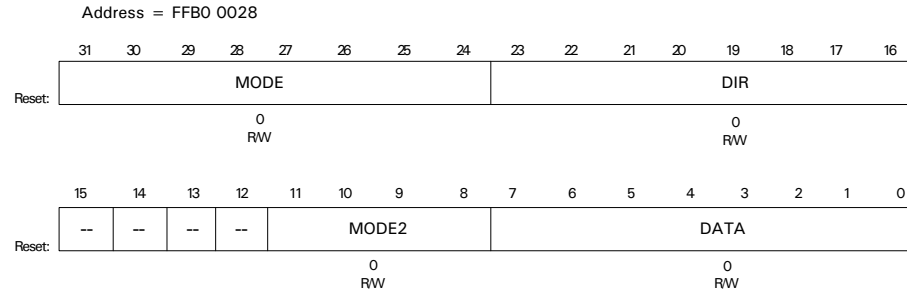
PORTC1

When configured for special function mode, the PORTC1 signal becomes an edge-sensitive interrupt detector. When the corresponding DIR bit is set to 0, the PORTC1 pin detects high-to-low transitions on PORTC1 and asserts 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

When configured for special function mode, the PORTC0 signal becomes an edge-sensitive interrupt detector. When the corresponding DIR bit is set to 0, the PORTC0 pin detects high-to-low transitions on PORTC0 and asserts 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.

PORTC register and bit definitions



Code (Bit #)	Definition of code	Description
MODE (D31:24)	PORTC mode configuration	Used to individually configure each of the PORTC pins to operate in either GPIO mode or special function mode. 0 – Selects GPIO mode 1 – Selects special function mode Each bit in the MODE field corresponds to one of the PORTC bits. D31 controls PORTC7, D30 controls PORTC6, and so on.
DIR (D23:16)	PORTC data direction	Used to individually configure each of the PORTC pins to operate in either input mode or output mode. 0 – Selects input mode 1 – Selects output mode. Each bit in the DIR field corresponds to one of the PORTC bits. D23 controls PORTC7, D22 controls PORTC6, and so on.
MODE2 (D11:08)	PORTC mode 2 configuration	Must be set to 0000 to ensure correct operation of the GPIO port. If set to any other value, the port can behave unpredictably.

Table 38: PORTC register bit definition

Code (Bit #)	Definition of code	Description
DATA (D07:00)	PORTC data register	Used when a PORTC bit is configured to operate in GPIO mode. ■ Reading the DATA field provides the current state of the signal, regardless of its configuration mode. ■ Writing the DATA field defines the current state of the 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 PORTC bits. D07 controls PORTC7, D06 controls PORTC6, and so on.

Table 38: PORTC register bit definition

Interrupt generation and control

The five Interrupt Control registers allow you to define and review the behavior of the NET+50's Interrupt Controller. The Interrupt Control registers are:

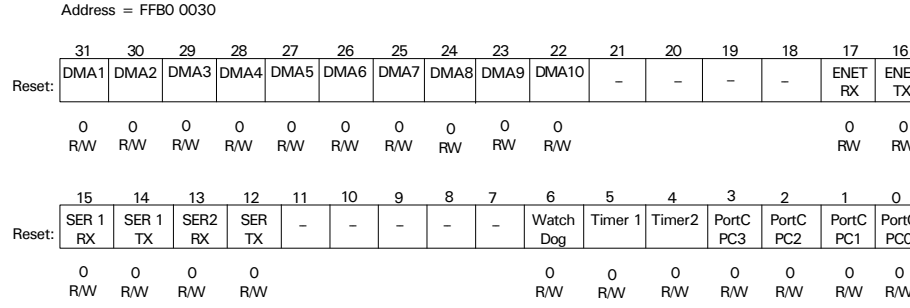
- **Interrupt Enable register** (FFB0 0030). Standard read/write register that can be updated in its entirety in a single write. When written to the fields in this register, 1s enable the corresponding interrupts and 0s disable them.
- **Interrupt Enable Register — Set** (FFB0 0034). 32-bit write-only register. This register sets specific bits without affecting the other bits. Writing a 1 to a bit position in this register sets the respective bit in the Interrupt Enable register. A zero in any bit position has no effect.

- **Interrupt Enable Register — Clear** (FFB0 0038). 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 effect.
- **Interrupt Status Register — Enabled** (FFB0 0034). The read-only personality of the Interrupt Enable Register — Set register. It identifies the current state of all the enabled interrupt sources. The sources that are not enabled are gated off by the Enable register, and therefore do not appear in this register.
- **Interrupt Status Register — Raw** (FFB0 0038). The read-only personality of the Interrupt Enable Register — Clear register. It identifies the current state of all interrupt sources regardless of whether the interrupts are enabled.

These registers, along with the NET+50's Interrupt Controller, control the generation and handling of interrupts. (See "Working with ARM exceptions" on page 46 for more information.)

Interrupt Control register and bit definition

All five Interrupt Control registers use the same 32-bit register layout, as shown.



Code (Bit #)	Description
DMA 1 (D31)	Corresponds to interrupts sourced by DMA channel 1. See Chapter 9, "DMA Controller Module."

Table 39: Interrupt Control register bit definition

Code (Bit #)	Description
DMA 2 (D30)	Corresponds to interrupts sourced by DMA channel 2. See Chapter 9, "DMA Controller Module."
DMA 3 (D29)	Corresponds to interrupts sourced by DMA channel 3. See Chapter 9, "DMA Controller Module."
DMA 4 (D28)	Corresponds to interrupts sourced by DMA channel 4. See Chapter 9, "DMA Controller Module."
DMA 5 (D27)	Corresponds to interrupts sourced by DMA channel 5. Chapter 9, "DMA Controller Module."
DMA 6 (D26)	Corresponds to interrupts sourced by DMA channel 6. Chapter 9, "DMA Controller Module."
DMA 7 (D25)	Corresponds to interrupts sourced by DMA channel 7. See Chapter 9, "DMA Controller Module."
DMA 8 (D24)	Corresponds to interrupts sourced by DMA channel 8. See Chapter 9, "DMA Controller Module."
DMA 9 (D23)	Corresponds to interrupts sourced by DMA channel 9. See Chapter 9, "DMA Controller Module."
DMA 10 (D22)	Corresponds to interrupts sourced by DMA channel 10. See Chapter 9, "DMA Controller Module."
ENET RX (D17)	Corresponds to an interrupt sourced by the Ethernet receiver. See Chapter 10, "Ethernet Controller Module."
ENET TX (D16)	Corresponds to an interrupt sourced by the Ethernet transmitter. See Chapter 10, "Ethernet Controller Module."
SER 1 RX (D15)	The SER 1 RX bit position corresponds to an interrupt sourced by the serial channel A receiver. See Chapter 11, "Serial Controller Module."
SER 1 TX (D14)	Corresponds to an interrupt sourced by the serial channel A transmitter. See Chapter 11, "Serial Controller Module."
SER 2 RX (D13)	Corresponds to an interrupt sourced by the serial channel B receiver. See Chapter 11, "Serial Controller Module."
SER 2 TX (D12)	Corresponds to an interrupt sourced by the serial channel B transmitter. See Chapter 11, "Serial Controller Module."

Table 39: Interrupt Control register bit definition

Code (Bit #)	Description
Watch-Dog (D6)	Corresponds to an interrupt condition sourced by the watch-dog timer. See "System Control register" on page 63 for details about the watch-dog timer.
TIMER 1 (D5)	Corresponds to an interrupt condition sourced by the TIMER 1 module. See "Timer Status register" on page 72 for details on timer interrupts.
TIMER 2 (D4)	Corresponds to an interrupt condition sourced by the TIMER 2 module. See "Timer Status register" on page 72 for details on timer interrupts.
PORTC PC3 (D3)	Corresponds to an interrupt condition caused by an edge transition detected on the PORTC3 pin. See "PORTC3" on page 90 for more information about the PORTC edge detection interrupts.
PORTC PC2 (D2)	Corresponds to an interrupt condition caused by an edge transition detected on the PORTC2 pin. See "PORTC2" on page 91 for more information about the PORTC edge detection.
PORTC PC1 (D1)	Corresponds to an interrupt condition caused by an edge transition detected on the PORTC1 pin. See "PORTC1" on page 91 for more information about the PORTC edge detection interrupts.
PORTC PC0 (D0)	Corresponds to an interrupt condition caused by an edge transition detected on the PORTC0 pin. See "PORTC0" on page 91 for more information about the PORTC edge detection interrupts.

Table 39: Interrupt Control register bit definition

NET+50 internal clock generation

The NET+50 chip has three main clock domains:

- SYSCLK — System clock. Drives the CPU and the NET+50 internal logic.
- XTAL — Bit-rate generation and programmable timer reference clock. Defines the serial modules.
- BCLK — System Bus Clock. Defines the speed of peripheral communication.

The SYS module generates these three clocks. Figure 8 shows the clock generation circuit.

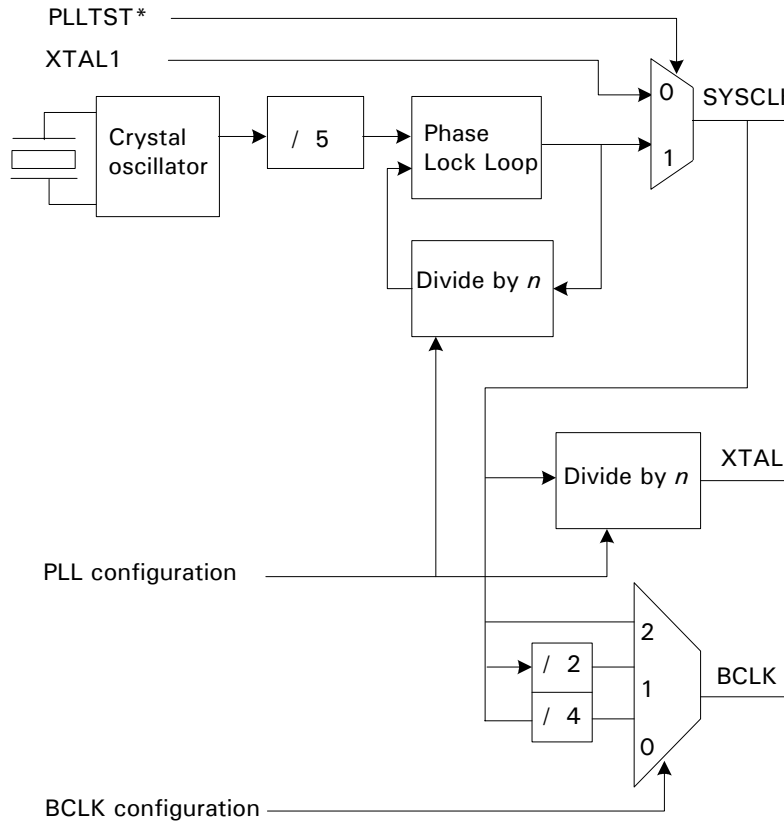


Figure 8: NET+50 chip clock generation

System clock generation

The system clock is provided by an external crystal, an internal crystal oscillator, and an internal phase locked loop (PLL). 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 PLL.

The PLL is configured to operate in multiplier mode. The PLL output is fed back through a divider, which allows the output to operate in a multiplier mode. The system clock frequency is **44.3** MHz.

Important: *PLL comes up for hardware reset to allow the internal system clock (SYSCLK) to run at 44.3MHz. If you need to use a different frequency for SYSCLK, you must use a different external configuration; that is a crystal with a different output frequency and/or an external oscillator.*

The PLL can be disabled and bypassed (or isolated) by asserting the PLLTST* pin active low. In this mode, the system clock is provided by the XTAL1 input. When the PLL is bypassed, a digital signal is required on the XTAL1 input, similar to the signal produced by an external CMOS clock source.

System clock frequency definitions

The system clock frequency is defined in one of two ways, depending on mode.

- **Inactive high.** When PLLTST* is inactive high (PLLTST*= 1), system clock frequency is defined as follows:

  $F_{SYSCLK} = (F_{CRYSTAL} / 5) * (PLLCNT + 3)$

  where:

  - F_{SYSCLK} identifies the frequency for ARM processor operation.
  - PLLCNT represents the value loaded into the PLLCNT field of the PLL Control register (PLLCNT value is **9** by default).
  - $F_{CRYSTAL}$ identifies the frequency of the external crystal component.

- **Active low.** When PLL is bypassed (PLLTST* = 0), system clock frequency is defined as follows:

  $F_{SYSCLK} = F_{CRYSTAL}$

  where:

  - F_{SYSCLK} identifies the frequency for ARM processor operation.
  - $F_{CRYSTAL}$ identifies the frequency of the CMOS clock input on the XTAL1 pin.

Internal XTAL clock reference

The internal XTAL clock signal is created by dividing the system clock by the same value as is loaded in the PLL multiplier. The internal XTAL clock signal is always 1/5th the frequency of the crystal input, irrespective of the PLL setting. The internal XTAL clock is used as a reference in both the GEN and SER (Serial) modules. With an 18.432 MHz crystal, for example, the internal XTAL timing reference is 3.6864 MHz (useful for serial bit-rate generation and timers).

Internal XTAL clock behavior can be emulated with an external oscillator; note the conditions in the following example:

- The PLL is bypassed.
- The external oscillator is running at 44.2368 MHz.
- The PLL Control register is set up to produce that same frequency by setting PLLCNT to 9.

With these conditions, the internal XTAL clock reference produces the 3.6864 MHz timing reference.

Internal XTAL clock frequency definitions

Internal XTAL clock frequency is defined in one of two ways, depending on mode:

- **Inactive high.** When PLLTST* is inactive high (PLLTST* = 1), the value of F_{XTAL} is 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.

- **Active low.** When PLLTST* is active low (PLLTST* = 0), the value of F_{XTAL} is defined as follows:

  ```
  IF (PLLCNT <= 3)
    FXTAL = FCRYSTAL / 6
  ELSE
    FXTAL = FCRYSTAL / (PLLCNT + 3)
  ```

  where:

- PLLCNT represents the value loaded into the PLL Control register (PLLCNT value is 9 by default).
- $F_{CRYSTAL}$ identifies the frequency of the external TTL clock on the XTAL1 pin.

Clock generation signals

Code (Bit #)	Description	Definition
XTAL1 XTAL2	Crystal oscillator input Crystal oscillator output	Provide the main input clock to the NET+50 chip, using a standard parallel-resonant crystal attached to the pins. See "Crystals and crystal oscillators," beginning on page 100, for more information.
PLLVDD	Clean PLL power	Powers the internal 2.5V PLL circuit. it is recommended that this pin be provided with clean power. The PLLVDD pin has an adjacent PLLVSS ground return.
PLLVSS	Clean PLL ground	Grounds the internal PLL circuit. It is recommended that this pin be provided with clean ground.
PLLLPF	PLL loop filter	Provides the PLL loop filter circuit. An external RC circuit (100 ohms in series with a .47μF with a .047 across both) must be attached between PLLPF and ground.
RESET*	System reset	Resets the NET+50 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+50 chip. **Note:** Both RESET* and TRST* must be pulsed active low after power up and should be active at the same time. These pins can be connected together.

Table 40: Clock generation signal description

Crystals and crystal oscillators

A standard parallel-resonant crystal can be attached to the XTAL1 and XTAL2 pins to provide the main input clock to the NET+50 chip.

Quartz crystals can be chosen from a wide range of manufacturers. The crystals' characteristics vary significantly, depending on their resonance frequency. Be sure the crystal you choose for the NET+50 chip is *parallel resonance with a load*

capacitance CL of 8pF. If the crystal-recommended load capacitance is greater, add capacitors external to XTAL1 and XTAL2.

The crystal oscillator is a low-power oscillator — do not use any external capacitors or resistors to reduce the current driven in the crystal. If the crystal being used has a drive level smaller than the drive level specified, however, add a series resistor between XTAL 2 and the crystal connection.

The NET+50 chip can be driven using an external TTL oscillator rather than the XTAL1/XTAL2 crystal oscillator circuit. 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.

Note: When this configuration is used, the internal PLL is bypassed and disabled.

The oscillator frequency equals *SYSCLK*. XTAL1 is in the 2.5V ring and has a maximum V_{IH} of 2.75V. A typical 3.3V oscillator requires a 220 ohm series resistor with its output.

Reset circuit

The NET+50 chip has four sources of hardware reset and one software reset:

- Powerup reset
- External reset
- Watch-dog reset
- ENI reset
- Software reset

Powerup reset

A powerup reset comes from a special ground PAD, which 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 powerup 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 powerup reset pulses), the counter triggers a hardware reset condition.

External reset

It is recommended that a system include an external powerup reset circuit that drives the RESET* pin active low during powerup. RESET* must be driven active low a minimum of 40ms after V_{DD} lines reach their minimal operating voltages. The RESET* pulse width must be a minimum of 1us at other times.

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, a hardware reset condition is triggered when the RESET* input is removed.

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.

Watch-Dog reset

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.

ENI reset

An 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. (See "ENI mode registers," beginning on page 400.)

Software reset

The software reset is issued by the CPU using the Software Service register in the GEN module. The software reset can be used to get all internal NET+50 peripherals into a known reset state.

Reset conditions

Table 41 identifies which internal NET+50 modules are reset under various reset conditions. The PORTC4 special function output works only with software and ENI resets.

NET+50 chip module	RESET condition				
	Power up	RESET*	Watchdog	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

[a.] All registers in the GEN Module are reset during powerup, RESET*, watch-dog, ENI, and software reset with a few exceptions. The following GEN Module fields are reset during the powerup, RESET*, and watch-dog conditions only (*not* ENI or software reset):

- BSPEED (GCR)
- BCLKD (GCR)
- PORTA
- PORTB
- PORTC

Table 41: Reset conditions

PLLTest Mode

When the PLLTST* signal is low, the PLL is isolated and the internal system clock is provided by the XTAL1 input. In addition, the external tester has access to all

PLL inputs. While PLLTST* and BISTEN* are both active low, the external tester has visibility into all PLL outputs.

When the PLLTST* input is high, the `indiv`, `icp`, `outdiv`, and `poltst` inputs are provided by a firmware programmable register; the `enb` input is driven low and the `test` input is driven low. The SYS_PLL_DIV signal is the output of the internal divider.

PLLTST	BISTEN*	PLL057 Signal	NETA50 Primary I/O
0	0	SYS_PLL_DIV	PORTA3 (output)
0	0	pdu	PORTA4 (output)
0	0	lock	PORTA5 (output)
0	0	pdd	PORTA6 (output)
0	0	ck	PORTA7 (output)
0	X	indiv[1]	PA0 (input)
0	X	indiv[0]	PA1 (input)
0	X	icp[3]	PA2 (input)
0	X	icp[2]	PA3 (input)
0	X	icp[1]	PA4 (input)
0	X	icp[0]	PA5 (input)
0	X	outdiv[1]	PA6 (input)
0	X	outdiv[0]	PA7 (input)
0	0	enb	PA8 (input)
0	1	enb	1 (disabled mode)
1	X	enb	0 (active mode)
0	X	poltst[0]	PA9 (input)
0	X	poltst[1]	PA10 (input)
0	X	poltst[2]	PA11 (input)
0	X	poltst[3]	PA12 (input)
0	X	ckr	TCK (input)

Table 42: PLL Test Mode Connections

PLLTST	BISTEN*	PLL057 Signal	NETA50 Primary I/O
0	X	ckdiv	TD1 (input)
0	X	test	PA13 (input)
0	X	dataload	XTAL1
X	X	lft	PLLLPF

Table 42: PLL Test Mode Connections

When PLLTST* is active low, the oscillator is disabled and provides a pass-through of the XTAL1 input as the main internal system clock.

PLLTST*	osc25fm16 signal	osc25fm16 value
0	onosc	0
1	onosc	1

Table 43: POR25 test mode connections

Cache

CHAPTER 7

The NET+50 CPU module uses the ARM7TDMI stand-alone core and adds an instruction/data cache, write protection, and pre-fetch control. The CPU module contains its own internal bus, referred to as the *CBUS*.

This chapter applies to the NET+50 chip only.

CBUS and BBus system buses

Figure 9 shows a block diagram of the CPU module structure within the NET+50 chip.

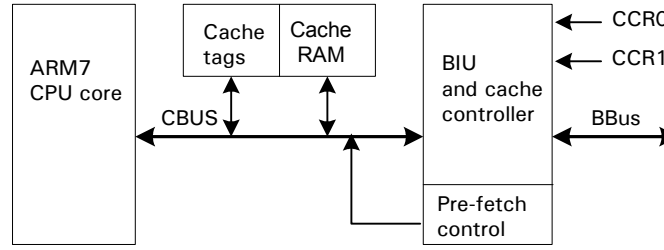


Figure 9: NET+50 CPU core block diagram

The CBUS and BBus system buses are separated using the BIU (bus interface unit). This structure allows the CPU to execute instructions from the cache while the BBus is being used for DMA data transfer operations.

All transactions that take place on the two buses are defined to operate in the logical memory space. Memory access requests mapped to external memory, through the MEM module, are considered to be in a physical addressing context. All internal buses, however, carry logical addresses.

BIU and cache controller

The BIU block provides the control functions necessary to separate the CBUS and BBus. 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 pre-fetch control when the ARM7TDMI is executing in Thumb mode and the external instruction storage peripheral is configured in 32-bit mode. Because Thumb instructions are 16 bits wide, you can use the full width of the data bus to fetch two instructions in one cycle.

Cache

The NET+50 chip contains an 8-KByte mixed instruction and data cache. The cache is arranged in a four-way set associative configuration.

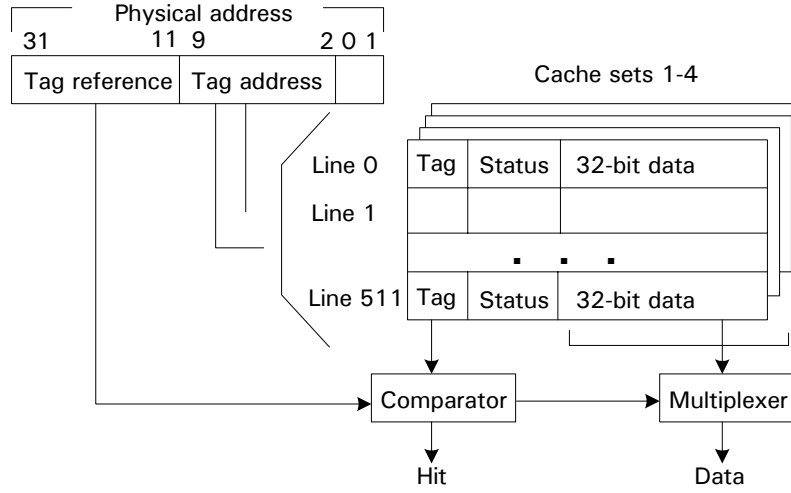


Figure 10: 4-way associated cache

When a cache-set is not enabled for caching, it can be used as a simple block of RAM. Each cache-set can be independently configured as a 2-KByte cache block or a 4-KByte RAM block.

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 unnecessary memory cycles when the program flow changes direction. Each line in the cache RAM contains a 21-bit tag, 2 control bits, an 8-bit status field, and a 32-bit data field.

See "Cache RAM" on page 114 and "Cache TAG format" on page 115 for more information.

Cache control registers

The NET+50 supports two Cache Control registers — CCR0 and CCR1. The Cache Control register identifies all the cache and buffer features for the identified memory block.

Each control register identifies a configurable block of logical address space. Typically, one Cache Control register identifies cacheable instructions and the other Cache Control register identifies cacheable data, which allows you to separate instruction and data fetch operations.

The Cache Control register selects cacheable address blocks using an address mask and compare operation. See "Cache Control registers" on page 112 for an illustration of the register and an explanation of each bit.

Cache operation

A read from a cacheable location results in data being taken from the cache if a cache hit occurs. 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 dismissed to make room for the new entry, a round-robin algorithm identifies which cache-set to dismiss.

Writes to areas defined as cacheable are written to the cache (when the data exists in the cache) and external memory at the same time.

Cache line fill

A cache line can be filled using 8-, 16-, or 32-bit operations, depending on the width of the bus transactions. The NET+50 chip cache controller does not force the entire line to be filled at one 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 corresponding address block. The cache controller inserts the new entry into any invalid set. If no empty sets are available, the cache controller discards one of the current lines to make room for the new data.

When the CPU 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 (see "Cache Control registers" on page 112). When

FRC32 is set, the cache controller forces a 32-bit operand fetch whenever the ARM fetches any operand on a zero byte boundary.

Cache line replacement

The appropriate, valid bits are cleared when a cache line is replaced, making the line empty and available. Any cache lines with the INVALID or LOCK bits set (see "Cache TAG format" on page 115) are not affected by a cache line replacement operation.

Cache line locking

You can lock instructions and data 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. These lines are not replaced by the round-robin algorithm.

32-bit pre-fetch

The 32-bit pre-fetch feature offers a performance improvement for those 16-bit Thumb applications using a 32-bit bus for instruction storage. Pre-fetch causes the memory interface to fetch a 32-bit word each time the ARM processor fetches any cacheable operand on a zero byte boundary. The FRC32 bit must be set to 1 in the Cache Control register to force fetching on a zero byte boundary. When the ARM processor fetches the next consecutive 16-bit instruction, the next instruction is most likely available in the cache.

Cache initialization

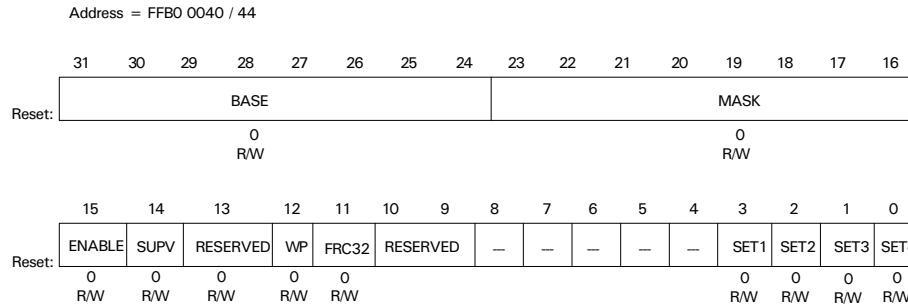
Before cache can be enabled for operation, the cache TAG area must be initialized to zero. A software loop can do this by writing zeros between byte addresses `0xFFF00000` and `0xFFF03FFC`.

While writing zeros to the cache RAM block, the CINIT bit in the GEN System Control register must also be set. The CINIT bit must be reset to zero after the cache RAM block has been initialized to zero.

Cache Control registers

The Cache Control registers, CCR0 and CCR1, provide control for the instruction and data cache. Each register selects a specific block of logical address space and determines how the cache is used.

Register and bit definitions



Code (Bit #)	Definition of code	Description
BASE (D31:D24)	Logical address base	Defines the base address of a logical block of memory that is to be cached using Cache Control register configuration parameters. The MASK field (D23:D16) defines the size of the logical block of memory. The BASE field is processed with a logical *AND* of the MASK field and then compared with the upper eight bits of the CPU bus (A31:A24) to determine which locations are to be cached with the CCR configuration. The upper eight CPU address bits are also processed with a logical *AND* of the MASK field before comparison with the BASE field.
MASK (D23:D16)	Logical address mask	Defines the size of the cacheable region. The MASK field identifies those bits in the BASE field definition that are used in the address comparison operation with the CPU bus address. Only the bit positions with 1 in the MASK field can be used in the address comparison function.

Table 44: CCR0/CCR1 register bit definition

Code (Bit #)	Definition of code	Description
ENABLE (D15)	Enable	0 — Disable CCR configuration 1 — Enable CCR configuration Must be set to 1 to allow the address defined by BASE and MASK to be cacheable according to the CCR parameters.
SUPV (D14)	Supervisor-only mode	0 — Allow both user and supervisor accesses to be cached 1 — Allow only supervisor accesses to be cached Set to 1 to cause this CCR configuration to cache only supervisor mode accesses. Useful for applications that want to cache operating system kernel execution only.
RESERVED (D13)	Reserved bit	Must always be set to 0. If set to 1, can result in unpredictable behavior of the cache controller.
WP (D12)	Write protect	0 — Allows writes to this block 1 — Does not allow writes to this block Provides the ability to write-protect a cacheable region of memory. Generally used to protect a cacheable region of memory used solely for instruction execution. When the bit is set 1, writing to a cacheable region of memory causes a data abort exception to the processor.
FRC32 (D11)	Force 32-bit prefetches	0 — Disable 1 — Force 32-bit fetches on 0 byte boundaries Forces 32-bit operands to always be fetched when address boundaries allow. ■ Should always be set to 1 when the CCR region is used with a peripheral that is 32-bits wide. ■ Must be set to 0 when the CCR region is used with any peripheral that is not 32-bit wide.
RESERVED (D10:D09)	Reserved bits	Must always be set to 0. If set to 1, can result in unpredictable behavior of the cache controller.

Table 44: CCR0/CCR1 register bit definition

Code (Bit #)	Definition of code	Description
SET1 (D03)	SET1 Enable	0 — Disable SET1; 1 — Enable SET1
SET2 (D02)	SET2 Enable	0 — Disable SET2; 1 — Enable SET2
SET3 (D01)	SET3 Enable	0 — Disable SET3; 1 — Enable SET3
SET4 (D00)	SET4 Enable	0 — Disable SET4; 1 — Enable SET4 Used typically to separate instruction and data regions, as well as apply different amounts of cache to each region Set to 1 attach the SET1 (and/or SET2, SET3, SET4) cache segment to the CCR region. Each cache set 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 allows the use of more cache sets for different memory regions.

Table 44: CCR0/CCR1 register bit definition

Cache RAM

The processor can fully address internal cache RAM memory. Data aborts occur when:

- The cache RAM memory block is accessed in non-supervisor mode when the USER bit in the GEN System Control register is set to 0 (see "System Control register" on page 63).
- The cache RAM memory block is accessed beyond its physical size.
- Attempts are made to access the cache RAM block when the CACHE bit is not enabled in the GEN System Control register (see "System Control register" on page 63).

Table 45 identifies how the cache RAM memory block is addressed.

Offset	Base Address			
	0xFFF00000 (set1)	0xFFF01000 (Set2)	0xFFF02000 (set3)	0xFFF03000 (set4)
0	Set1 Entry1 TAG	Set2 Entry1 TAG	Set3 Entry1 TAG	Set4 Entry1 TAG
4	Set1 Entry1 DATA	Set2 Entry1 DATA	Set3 Entry1 DATA	Set4 Entry1 DATA
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
0xFF8	Set1 Entry512 TAG	Set2 Entry512 TAG	Set3 Entry512 TAG	Set4 Entry512 TAG
0xFFC	Set1 Entry512 Data	Set2 Entry512 DATA	Set3 Entry512 DATA	Set4 Entry512 DATA

Table 45: Cache RAM memory addressing

The information contained in each set falls into contiguous memory locations:

- Each set, if not enabled in any CCR for cache, provides 4 KBytes of contiguous random access memory. This results in a total of 16 KBytes of fast, 0-wait state RAM.
- If 2 adjacent sets are not enabled as cache, then 8 KBytes of contiguous RAM 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; 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.

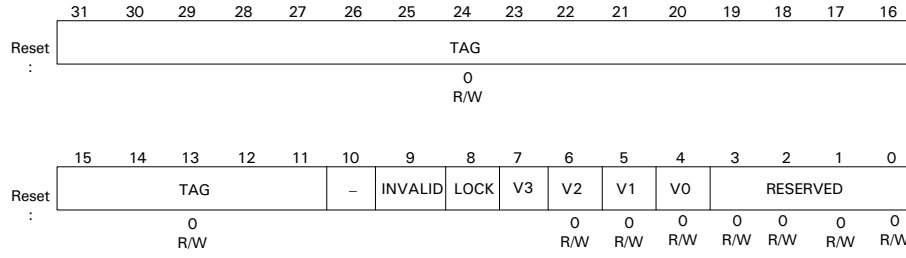
Cache TAG format

Each cache line stores 32 bits of data. A cache line is always in one of two states: valid or invalid. Each cache line can contain 0 to 4 bytes of valid 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.
- When a line is valid, it contains data consistent with external memory.

Figure 10, "4-way associated cache," on page 109 provides a simplified view of how the cache mechanism works. The caches always use logical addressing. To determine if the logical address is currently located in the cache, address bits A31 through A2 are used to reference the cache tag or to update the cache tag field. If a match occurs, the selected cache line is used and flagged as a hit.

Cache TAG format and definitions



Code (Bit #)	Definition of code	Description

TAG (D31:11)	Tag reference	Identifies which memory location is currently stored within this cache entry. This field is valid only when one of the four V (valid) bits is set to 1. Use the following formula to calculate the actual physical address of the information for this cache entry:
		`physical address =`
		`(TAG << 10 || ((cacheAddress & 0x000007F8) >> 1)`

D10		Not used
INVALID (D09)	Invalidate location	0 — Location is valid for cacheable entries 1 — Location is invalid for any cache entries Prevents specific cache lines within a set from being used for cache entries. Typically used to reserve cache entries for code that will be locked down in the cache at a later time.

Table 46: Cache tag bit definitions

Code (Bit #)	Definition of code	Description
LOCK (D08)	Locked state	0 — Entry not locked 1 — Entry locked in cache Causes an entry to remain static within the cache. Can be set to 1 to have the current entry remain in the cache set indefinitely. Typically used to preload software functions in the cache.
V3 (D07)	Byte 3 valid	0 — Byte 3 is invalid 1 — Byte 3 is valid 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, and to the most significant byte when operating in Little Endian mode.
V2 (D06)	Byte 2 valid	0 — Byte 2 is invalid 1 — Byte 2 is valid Set to 1 to indicate that byte 2 in this cache entry is valid.
V1 (D05)	Byte 1 valid	0 — Byte 1 is invalid 1 — Byte 1 is valid Set to 1 to indicate that byte 1 in this cache entry is valid.
V0 (D04)	Byte 0 valid	0 — Byte 0 is invalid 1 — Byte 0 is valid 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, and to the least significant byte when operating in Little Endian mode.
RESERVED (D03:D00)	Reserved bits	Should always be set to 0. If set to 1, can result in unpredictable behavior of the cache controller

Table 46: Cache tag bit definitions

Memory Controller Module

CHAPTER 8

This chapter explains how the NET+50 chip can be configured to interface with various types of memory devices.

Note: The information in this chapter applies to both the NET+50 and NET+20M chips, unless otherwise noted.

MEM module features

The MEM module provides these features:

- Five chip select configurations
- 28-bit address bus that allows access to 256 Mbytes per chip select
- 8-, 16-, or 32-bit data bus sizes
- Two timing variations for access to static RAM (SRAM)
- Fast Page (FP) DRAM timing and EDO DRAM timing

  Note: Both FP and EDO DRAM can use one of two internal RAS/CAS address multiplexers, which source 10-CAS or 8-CAS.
- Synchronous DRAM (SDRAM) timing

  Note: SDRAM can use the internal 8-CAS internal address multiplexer.
- Use of an external RAS/CAS multiplexer by all DRAM timing, allowing any RAS/CAS combination
- 0 to 15 wait states
- Externally-controlled transfers
- Write protect (WP)

Memory control signals

The NET+50 uses these signals to interface with memory. This section provides a brief description; see "Memory control signals" on page 134 for more information about each signal.

Signal	Definition
D[31:0]	32-bit data bus
A[27:0]	28-bit address bus
CS[4:0]	5 chip selects
RAS[4:0]	5 row address strobes for DRAM

Table 47: Memory control signal overview

Signal	Definition
CAS[3:0]	4 column address strobes for DRAM
R/W*	Read/Write signal
OE*	Output enable signal
WE*	Write enable signal
BE[3:0]	4 byte enable signals, one for each byte in the 32-bit data bus
TA*	Transfer acknowledge
TEA*	Transfer error acknowledge

Table 47: Memory control signal overview

Configuration registers

The Memory module has four configuration registers for each of the five chip selects. These registers adjust the behavior of the memory control signals to produce the optimum required for the particular memory device connected to the NET+50.

Registers

The NET+50 chip uses these four configuration registers:

- Memory Module Control register (MMCR)
- Chip Select Base Address register (CSBAR)
- Chip Select Option register (CSOR)
- Chip Select Option Register B (CSORB)

Table 48 lists the names and addresses of the configuration registers.

Address	Name	Description
FFC0 0000	MMCR	Memory Module Configuration register

Table 48: MEM module configuration registers

Address	Name	Description
FFC0 0010	BAR0	Chip Select 0 Base Address register
FFC0 0014	OR0	Chip Select 0 Option register
FFC0 0018	ORB0	Chip Select 0 Option Register B
FFC0 0020	BAR1	Chip Select 1 Base Address register
FFC0 0024	OR1	Chip Select 1 Option register
FFC0 0028	ORB1	Chip Select 1 Option Register B
FFC0 0030	BAR2	Chip Select 2 Base Address register
FFC0 0034	OR2	Chip Select 2 Option register
FFC0 0038	ORB2	Chip Select 2 Option Register B
FFC0 0040	BAR3	Chip Select 3 Base Address register
FFC0 0044	OR3	Chip Select 3 Option register
FFC0 0048	ORB3	Chip Select 3 Option Register B
FFC0 0050	BAR4	Chip Select 4 Base Address register
FFC0 0054	OR4	Chip Select 4 Option register
FFC0 0058	ORB4	Chip Select 4 Option Register B

Table 48: MEM module configuration registers

Memory Module Control register (MMCR)

The MMCR controls the functionality of address bits A27, A26, and A25. These address bits can function as other than address lines, and are defined within this register. The MMCR also controls how DRAM-configured chip selects are refreshed, and how the DRAM's RAS/CAS is multiplexed. All chip selects configured as DRAM must be set to the same type of DRAM. The DRAM configuration bits in this register are common to all DRAM chip selects.

The DRAM refresh controller always generates CAS before RAS refresh cycles. The RFCNT, REFEN, and RCYC bits in the MMCR work together to define DRAM refresh timing. The NET+50 will produce *background refresh* when accessing an SRAM device. The RAS and CAS lines are not needed for static RAM (SRAM), so a DRAM refresh can be accomplished when accessing SRAM.

Memory Module Control register and bit definitions

Address = FFC0 0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset:	RFCNT								REFEN	RCYC		AMUX	A27_F	A26_F	A25_F*	AMUX2
	0 R/W								0 R/W		0 R/W	0 R/W	0 R/W	0 R/W	I R/W	0 R/W

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset:	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

RFCNT (D31:24) **Refresh count value**

Defines the refresh period for the memory controller when Fast Page, EDO, or SDRAMs are being used. All DRAM devices require a periodic refresh cycle. The refresh cycle typically is 15 us. The NET+50 chip allows the refresh period to be programmable.

The refresh cycle is a function of the RFCNT field. The value for F_{XTAL} is defined by the following equation:

```
Refresh period = [(RFCNT + 1) * 4] / FXTAL
```

See "NET+50 internal clock generation" on page 96 for more information about the F_{XTAL} value.

REFEN (D23) **DRAM refresh enable**

0 – Disable DRAM refresh.

1 – Enable DRAM refresh.

RCYC (D22:21) **Refresh cycle count**

00 – Refresh cycle is 8 BCLK clocks long
(see Figure 74, "Fast Page and EDO DRAM Refresh (RCYC = 0)," on page 443)

01 – Refresh cycle is 6 BCLK clocks long
(see Figure 75, "Fast Page and EDO DRAM Refresh (RCYC = 1)," on page 443)

10 – Refresh cycle is 5 BCLK clocks long
(see Figure 76, "Fast Page and EDO DRAM Refresh (RCYC = 2)," on page 444)

11 – Refresh cycle is 4 BLCK clocks long
(see Figure 77, "Fast Page and EDO DRAM Refresh (RCYC = 3)," on page 444)

Controls the length of the refresh cycle. These settings apply only to FP/EDO DRAM. SDRAM is always refreshed as shown in Figure 84, "SDRAM Refresh," on page 452.

AMUX (D20) — **Enable external address multiplexing**

0 – Disable external address multiplexing on PORTC3 for all DRAM banks

1 – Enable external address multiplexing on PORTC3 for all DRAM banks

Controls whether the NET+50 chip uses its internal DRAM address multiplexer.

When AMUX is set to 1, the NET+50 chip uses an external DRAM address multiplexer. The RAS/CAS address select signal is routed out the PORTC3 signal. The external DRAM RAS/CAS address multiplexer function can use the PORTC3 signals to determine when to switch the address multiplexer.

A27_F (D19) — **A27 pin functionality**

0 – The CS0OE* signal is driven out the ADDR27 pin

1 – The ADDR27 pin is used for address bit 27

Determines the functionality of the A27 pin. A27F can be set to source the A27 address signal or the OR-gated CS0* and OE* signals.

- When A27_F is set to 0, the CS0OE* signal is driven out the A27 pin. The CS0OE* signal is generated by an internal OR-gating of CS0* and OE. The CS0OE* signal goes active low when both CS0* and OE* are low.
- When A27_F is set to 1, the A27 bit is sourced.

Figure 11 shows the functionality of the A27_F bit:

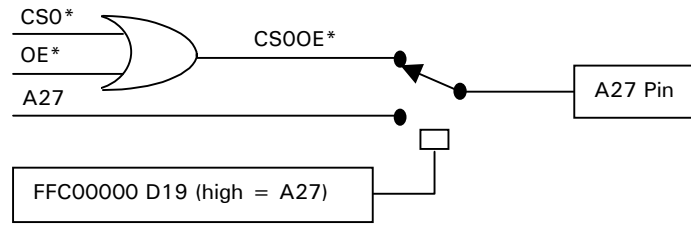


Figure 11: A27_F functionality

A26_F (D18) **A26 pin functionality**

0 – The CS0WE* signal is driven out the ADDR26 pin

1 – The ADDR26 signal is used for address bit 26

Determines the functionality of the A26 pin. A26_F can be set to source the A26 address signal or to the OR-gated CS0* and WE* signals.

- When A26_F is set to 0, the CS0WE* signal is driven out the A26 pin. The CS0WE* signal is generated by the internal OR-gating of the CS0* and WE* signals. The CS0WE* signal goes active low when both CS0* and WE* are low.
- When A26_F is set to 1, the A26 bit is sourced.

Figure 12 shows the functionality of the A26_F bit:

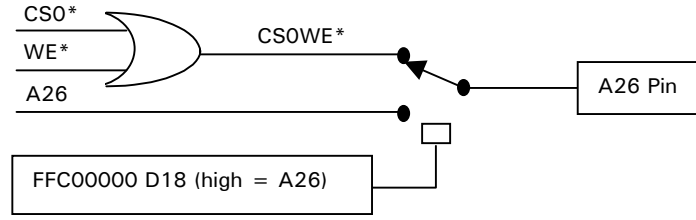


Figure 12: A26F functionality

A25_F (D17) — **Enable the A25 output**

0 – The ADDR25 pin is used as address bit 25

1 – Reserved. This bit is never set to 1.

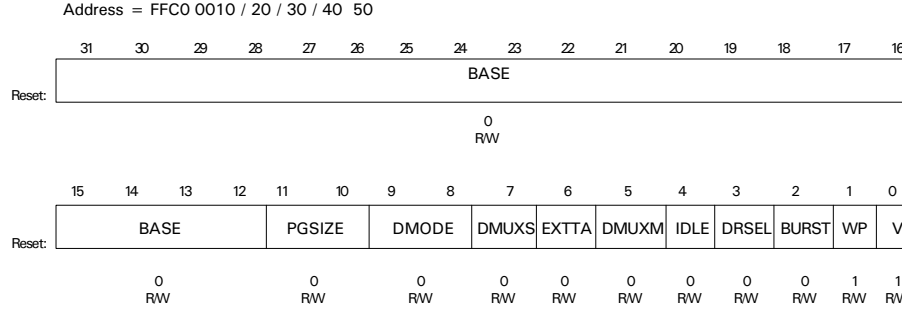
AMUX2 (D16) — **Internal/External RAS/CAS Mux**

0 – Normal operation

1 – Reserved. This bit is never set to 1.

Chip Select Base Address register and bit definitions

The CSBAR defines the base starting address for the chip select and configures other chip select options. Each chip select can be configured in size from 4 Kbytes to 256 Mbytes. All bits are reset to 0 on system reset. This register works in conjunction with the Chip Select Option register MASK field.



BASE (D31:12) — **Chip select base address**

Determines the starting physical address of the memory peripheral chip select. This field represents the 20 most significant bits of the physical address. To determine the base physical address of a memory peripheral decode space, add three zeros to the end of this field. For example:

```
BASE field value = 0x00200
plus three zeros = 0x00200000
0x00200000 = starting physical address of the peripheral chip select
```

The BASE field must be consistent with the size of the chip select as defined by the MASK field in the Chip Select Option register. The base address of a chip select must occur on a proper boundary in relation to the peripheral memory size. If the size of the memory device is 4 Mbytes, for example, the BASE field must be configured such that the base address is located on a 4 Mbyte address.

See "Setting the Chip Select address range" on page 136 for an explanation of how the BASE and MASK fields work together to define a chip select's address space.

PGSIZE (D11:10) — **Peripheral page size**

00 – 64 bytes

01 – 32 bytes

10 – 16 bytes

11 – 8 bytes

Defines the page size for the attached peripheral. This field controls the address burst boundary for the NET+50 memory controller. The NET+50 will not burst through an address boundary that is larger that defined by the PGSIZE field.

DMODE (D09:08) — **DRAM mode**

00 – Fast Page DRAM

01 – EDO DRAM

10 – SDRAM

11 – Reserved

Selects the type of DRAM. The DMODE field is valid only when the DRSEL bit is set to 1, indicating DRAM mode. All chip selects configured as DRAM must be set to the same type of DRAM.

DMUXS (D07) — **Internal/External RAS/CAS**

0 – Internal DRAM multiplexer

1 – External DRAM multiplexer

Controls whether the NET+ARM 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 is used to signal to the external address multiplexer when to switch the address bit. When the PORTC3 signal is high, the external DRAM RAS/CAS address multiplexer function drives the CAS address to the DRAM devices.

The AMUX or AMUX2 bit in the MMCR serves as a global control when set. When either of these bits is set to 1, all DRAM peripheral devices use the external address multiplexer and the DMUXS bit is ignored.

EXTTA (D06) — **Enable external transfers**

0 – Internal

1 – External

Determines how a memory cycle is terminated:

- When EXTTA is set to 0, the TA* signal is an output and signals the end of the transfer. The memory transfer time is consistent and controlled only by the NET+ARM's memory configuration.
- When EXTTA is set to 1, the TA* signal is an input and the transfer does not end until the external device drives the TA* signal low, to signal the end of the transfer. External transfers must be used when the device with which the NET+ARM is interfacing has an inconsistent cycle time.

Note: When the external transfer option is selected, the synchronizer bits in Chip Select Option Register B must be configured.

DMUXM (D05) — **Internal/External RAS/CAS**

0 – Mode-0 10 CAS
(see Figure 15, "DRAM Mode-0 addressing," on page 150 and Table 54: "DRAM Mode-0 addressing" on page 151)

1 – Mode-1 8 CAS
(see Figure 16, "DRAM Mode-1 addressing," on page 153 and Table 55: "DRAM Mode-1 addressing" on page 154)

When set to 0, the internal multiplexer option is selected. The DMUXM field offers two options: 10 CAS with Mode-0 and 8 CAS with Mode-1.

IDLE (D04) **Insert null clock cycle**

0 – Do not add null cycle after transfer

1 – Add one null cycle after every transfer for this chip select

Inserts an extra BCLK cycle at the end of the memory cycle for a chip select that has been configured as DRAM.

Note: *Do not set the idle bit in a non-DRAM chip select.*

DRSEL (D03) **DRAM select**

0 – Static RAM

1 – DRAM

- When DRSEL is set to 0, the chip select will be configured as static RAM and the timing is based on the WSYNC and RSYNC setting.
- When the DRSEL bit is set to 1, the chip select is configured as FP, EDO, or SDRAM based on the DMODE setting.

BURST (D02) **Enable burst transfers**

0 – Single cycle transfers only

1 – Burst cycle transfers are allowed

Controls whether the memory peripheral device supports bursting:

- When BURST is set to 1, burst cycles are allowed if the current bus master wants 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. The peripheral terminates the burst, however, when the maximum number of memory cycles defined by the BSIZE field is reached.
- When BURST is set to 0, burst cycles are not allowed.

WP (D01) **Write protect**

0 – Allow both read and write cycle transfers

1 – Do not allow memory write-cycles

WP can be used to prevent any bus master from writing to the memory device. When WP is set to 1, all memory write-cycles are terminated immediately with an data abort indicator. The WP bit can be used to protect non-volatile memory devices such as flash and EEPROM.

Note: The WP bit is set to 1 on hardware reset for CS0 only.

V (D00) **Valid**

0 – Disable the chip select

1 – Enable the chip select

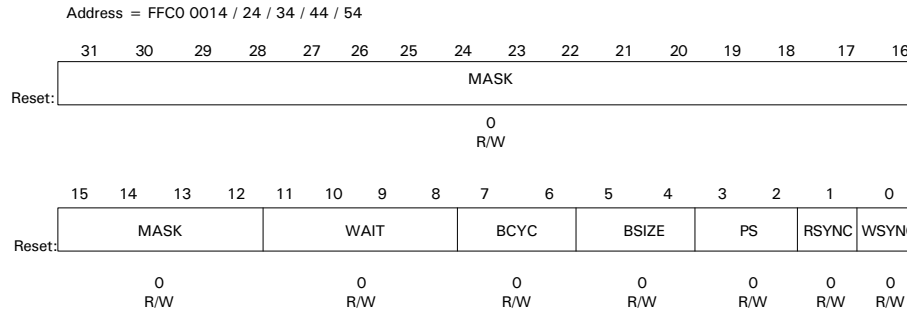
Enables the chip select. When the V bit is set to 1, the memory controller uses various fields in the Chip Select Base Address and Option registers to control the behavior of the peripheral memory cycles.

Note: For all configurations except SDRAM, it is recommended that you set the valid bit last, after all other bits in the MMCR, CSBAR, CSOR and CSORB have been configured.

When a chip select is configured as SDRAM, the V bit has an additional function—it produces the SDRAM load-mode command. (See "Load-Mode procedure: Setting up the Load-Mode command" on page 155 for more information.)

Chip Select Option register and bit definitions

The CSOR defines the physical address size of the chip select, and configures other chip select options. All bits in the register are reset to 0 on system reset.



MASK (D31:12) **Mask address**

Sets the size of the memory space that the chip select will respond to. The MASK field works in conjunction with the BASE field in the CSBAR to set the chip select's address space.

The memory controller performs a logical AND of the MASK and BASE fields to determine whether the address space is assigned to the chip select. The MASK field is used to identify those address bits from A31 through A12 that will be part of the address space.

See "Setting the Chip Select address range" on page 136 for an explanation of how the BASE and MASK fields work together to define a chip select's address space.

WAIT (D11:8) **Number of wait states**

Controls the number of wait states for all single cycle memory transfers and the first memory cycle of a burst transaction.

All NET+ARM 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.

Setting this 4-bit field will insert between 0 and 15 wait states in the memory transfer.

BCYC (D7:6) **Number of clocks for subsequent cycles**

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)

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.

Note: When the chip select is configured as SDRAM, the BCYC field has a different function when the V bit in the CSBAR is set. BCYC helps determine how the SDRAM load-mode command is configured (see "Setting the Chip Select address range" on page 136).

BSIZE (D5:4) — **Burst size**

00 – 2 system bus cycles in burst

01 – 4 system bus cycles in burst

10 – 8 system bus cycles in burst

11 – 16 system bus cycles in burst

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 not to 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.

Notes:

1. A potential limitation for this field is the BTE field in the DMA Control register (see page 187). If the data bus is 32 bits and the BTE field is set to a maximum of 16 bytes, the transfer will end after 4 cycles (16 bytes), even if the BSIZE field is set to 11 (16 system bus cycles in burst).
2. When the chip select is configured as SDRAM, the BSIZE field has a different function when the V bit in the CSBAR is set. BSIZE helps determine how the SDRAM load-mode command is configured (see "Setting the Chip Select address range" on page 136).

PS (D3:2) — **Port size**

00 – 32-bit Port Size

01 – 16-bit Port Size

10 – 8-bit Port Size

11 – 32-bit Port using external Data Acknowledge

Controls the size of the data bus associated with the chip select. Each chip select can be configured to operate as an 8-bit device, a 16-bit device, or a 32-bit device. The address bus adjusts accordingly.

RSYNC (D01) **Read synchronous mode**

0 – SRAM chip select read cycles will operate in asynchronous mode

1 – SRAM chip select read cycles will operate in synchronous mode

Controls the access timing of the OE* and CS* signals for SRAM timing. The RSYNC bit is used only when the DRSEL bit is set to 0.

- When RSYNC is set to 1, the memory peripheral operates in a mode in which the OE* signal is active low inside the low time of CS*.
- When RSYNC is set to 0, the memory peripheral operates in a mode in which the OE* signal is active low outside the low time of CS*.

See "SRAM Sync Read (WAIT = 2)" on page 427 and "SRAM Async Read (WAIT = 2)" on page 432.

WSYNC (D00) **Write synchronous mode**

0 – SRAM chip select write cycles will operate in asynchronous mode

1 – SRAM chip select write cycles will operate in synchronous mode

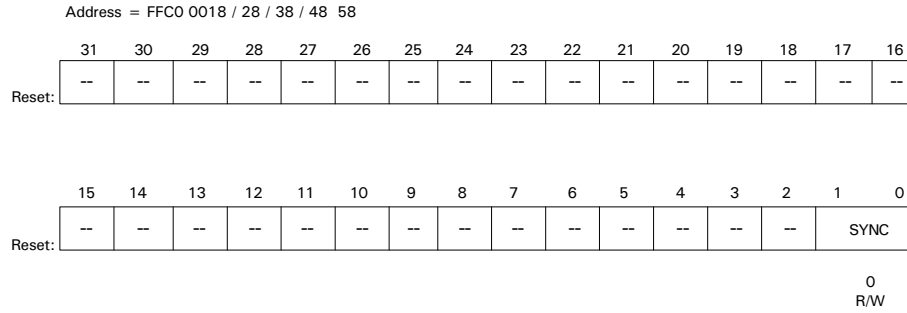
Controls the access timing of the WE* and CS* signals for non-DRAM memory peripherals. The WSYNC bit is used only when the DRSEL bit is set to 0.

- When WSYNC is set to 1, the memory peripheral operates in a mode in which the WE* signal is active low inside the low time of CS*.
- When WSYNC is set to 0, the memory peripheral operates in a mode in which the WE* signal is active low outside the low time of CS*.

See "SRAM Sync Write (WAIT = 2)" on page 428 and "SRAM Async Write (WAIT = 2)" on page 433.

Chip Select Option Register B and bit definitions

Chip Select Option Register B defines the level of synchronization performed within the NET+50 chip for TA* input. The SYNC field applies only when the EXTTA bit in the CSBAR is set to 1, for external transfers. The TA* input must be synchronized with the Net+ARM's BCLK. During external transfers, the external device sources the TA* signal. If the external device does not use BCLK when generating TA*, either the 1- or 2-stage synchronizer must be used.



SYNC (D1:0) **TA* input synchronizer**

00 – No synchronizer

01 – 1-stage synchronizer

10 – 2-stage synchronizer

11 – Reserved

- You must use the 1- or 2-stage synchronizer setting when the TA* input is asynchronous to the BCLK signal. (The 2-stage synchronizer is better as it introduces one additional BCLK of latency in the access cycle.)
- Use the "no synchronizer" setting when the TA* input is synchronous to the BCLK signal.

Memory control signals

Several signals interface with memory (see page 120). Their functionality and timing depend on the register settings.

Data bus (D31:0)

The data bus provides the data transfer path between the NET+50 chip and external devices. The data bus can be an 8-bit, 16-bit, or 32-bit size. The PS field in the Chip Select Option register determines the data bus size.

The address bus acts differently for each setting:

- An 8-bit data bus sources all addresses, and the NET+ARM's A0 pin attaches to the memory's A0 pin.
- A 16-bit data bus skips odd-numbered addresses, and the NET+ARM's A1 pin attaches to the memory's A0.
- A 32-bit data bus skips all addresses that are not divisible by four, and the NET+ARM's A2 pin attaches to the memory's A0.

Peripheral devices

- 8-bit peripheral devices must always attach to the NET+50 chip using D31:D24. The least significant data bit for 8-bit peripherals is D24. The least significant address bit for 8-bit peripherals is A0.
- 16-bit peripheral devices must always attach to the NET+50 chip using D31:D16. The least significant data bit for 16-bit peripherals is D16. The least significant address for 16-bit peripherals is A1.
- 32-bit peripheral devices must always attach to the NET+50 chip using D31:D00. The least significant data bit for 32-bit peripherals is D0. The least significant address bit for 32-bit peripherals is A2.

Address bus (A27:0)

The NET+ARM address bus is 28 bits long. Under most conditions, the NET+ARM can access 256 Mbytes of address space per chip select. To access all of the address space, the A27_F and A26_F bits in the MMCR must be set to 1; otherwise, those address bits are lost to another function and become unavailable as address signals.

The behavior of the address bus depends on the chip select configuration. The PS (data port size) field, DRSEL (static vs. DRAM) field and the type of RAS/CAS multiplexer all affect how the address reacts.

The address that eventually goes to a memory device originates in the ARM program counter or the DMA module.The address goes through the MEM module, where it is adapted to work with the type of memory for which the chip select is configured. Each of the five chip selects can be configured differently, to respond to different sections of the memory map and to interface with either DRAM or SRAM (flash and EEPROM use SRAM timing).

Three factors affect the order of the address bits that finally reach the memory device:

- Data bus size: 8-bit, 16-bit, 32-bit
- Memory type: SRAM, external MUX, DRAM, or SDRAM
- DRAM address mode

Chip selects/RAS (CSx*/RAS)

There are five chip select signals (CS[4:0]). Each chip select is enabled low when its address space is accessed. The chip select timing depends on the configuration. Each chip select has four configuration registers that determine the chip selects' properties. All five chip selects are interchangeable, and have identical functionality — with the exception of CS0*, which has an additional feature. CS0* can automatically be configured on powerup (see "CS0* boot configuration" on page 139).

When the chip select is configured as Fast Page or EDO DRAM, the chip select functions as the DRAM's RAS signal. DRAM does not use a chip select signal per se but, in many ways, the RAS (row address strobe) signal is similar. The RAS signal shares the same NET+ARM pins as the CS* signals, its timing is similar, and its address space is decoded with the same BASE and MASK values in the Chip Select Base Address and Chip Select Option registers. The RAS signal goes low when the row address is valid. DRAM uses this signal to internally latch the row address.

Setting the Chip Select address range

Each chip select should be configured to respond to a different portion of the memory map. Do this by setting the appropriate field in the Chip Select Base Address and Chip Select Option registers.

- The MASK field in the Chip Select Option register defines the chip select address space.
- The BASE field in the Chip Select Base Address register defines the starting address of the chip select address space.

The BASE and MASK fields are used in combination to decode the address space that will be assigned to a chip select. The memory controller performs a logical `AND` of the MASK and BASE fields to determine whether the address space is assigned to the peripheral device. The MASK field identifies those address bits, from A31 through A12, that are used in the address decoding function.

- All ones in the MASK field indicates that the process should use the associated address bit.
- All zeros in the MASK field indicates that the process should ignore the associated address bit.

Each chip select's address space is at least 4096 bytes wide. Bits [11:0] are always part of the address space no matter how bits [31:12] are configured. Any bit set to 0 in the MASK field is also part of the address space.

To determine if a logical address is associated with a chip select, apply the following boolean equation:

```
(({MASK,000} & 32-bit logical address)==(({BASE,000} & 32-bit logical
address
```

where:

- `{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
- `&` refers to the logical `AND` function
- == refers to the `equal to` operator

Example

This example illustrates setting the chip select address for 16 meg space.

```
Chip Select Option register = 0xFF000XXX (16 Meg space)
```

To determine the amount of address space, invert the value of the mask bits and change the 12 "don't care" bits to *1*:

```
0xFF000XXX => 0x00FFFFFF
```

If the base is set to 0, the memory space is from 0 to 0xFFFFFF. The allowable locations for the base are a multiple of 16; add *1* to 0xFFFFFF to get the correct address increment boundary:

0x00FFFFFF + 1 = 0x01000000

The correct values in the Chip Select Base Address register would be 0x00000XXX, 0x01000XXX, 0x02000XXX, 0x03000XXX, and so on, up to 0xFF000XXX.

Any other value is truncated to the allowable boundary. Although a truncated value can be used, the code can be hard to read. For example, the value 0x02034XXX truncates to begin at 0x02000000 rather than 0x2034000.

Memory space

The memory space associated with an individual chip select does not have to be continuous. Using the MASK field, you can set *gap*s in a single chip select range, essentially assigning the same chip select to different portions of the memory map.

Example

A 16 Mbyte device can be addresses in 4 different 64 Mbytes address locations by using a MASK field value of 0xF3000. The peripheral device is addressable at these address locations:

- BASE + 0x00000000
- BASE + 0x04000000
- BASE + 0x08000000
- BASE + 0x0C000000

Table 49 provides examples of mask settings and the related address space:

Mask	Size
0xFFFFF	4K
0xFFFFE	8K
0xFFFFC	16K
0xFFFF8	32K
0xFFFF0	64K
0xFFFE0	128K

Table 49: Mask settings and related address space

Mask	Size
0xFFFC0	256K
0xFFF80	512K
0xFFF00	1M
0xFFE00	2M
0xFFC00	4M
0xFF800	8M
0xFF000	16M
0xFE000	32M
0xFC000	64M
0xF8000	128M
0xF0000	256M

Table 49: Mask settings and related address space

CS0* boot configuration

Chip select 0 (CS0*) can be configured on powerup, which allows the ARM processor to boot from a flash or EEPROM device attached to CS0*. On powerup, the NET+ARM reads the address bus. If an address line is floating, the NET+ARM reads a `1`; if a 1K pull-down resistor is attached to the address line, the NET+ARM reads a `0`.

The binary values on A24 and A23 will configure CS0* on bootup. Table 50 shows the four possible bootup configurations for CS0*:

ADDR[24:23]	CS0 bootstrap setting	A27	A26
“00”	Disable	A27	A26
“01”	32-bit SRAM port; 15 wait-states	A27	A26
“10”	32-bit FP DRAM port; 15 wait-states	A27	A26
“11”	16-bit SRAM port; 15 wait-states	CSOOE	CSOWE

Table 50: CS0* bootstrap settings

These bootstrap settings set the values of the A27_F and A26_F bits in the Memory Module Configuration register, the V bit in the CS0* Chip Select Base Address register, and the WAIT and PS fields in the CS0* Chip Select Option register, as shown in the following table:

ADDR[24:23]	A27_F	A26_F	WAIT	PS	V
00	1	1	0000	00	0
01	1	1	1111	00	1
10	1	1	1111	00	1
11	0	0	1111	01	1

Column address strobes (CAS)

CAS* signals are activated when an address is decoded by a chip select module that is configured for DRAM mode. The signal goes low when the column address is valid. DRAM uses this signal to internally latch the column address. After receiving the row and column addresses, DRAM combines them internally so it can access the correct word. One CAS line is provided for each DRAM byte.

The CAS* signals also identify which bytes of the 32-bit data bus are active during a given memory cycle, as shown in Table 51. Note the similarity between this table and Table 52: "BE* lane configurations" on page 142.

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		
CAS* lane configuration for 32-bit peripherals					
CAS3*	D31:D24	1	0		

Table 51: CAS* lane configurations

CAS*	Active data bus lane	Little Endian byte address	Big Endian byte address	Little Endian word address	Big Endian word address
CAS2*	D23:D16	0	1		
CAS* lane configuration for 16-bit peripherals					
CAS3*	D31:D24				
CAS* lane configuration for 8-bit peripheral					

Table 51: CAS* lane configurations

Read/Write (R/W*)

The R/W* signal is high for a read cycle and low for a write cycle. This signal usually is connected to the direction pin on a data transceiver.

Output enable (OE*)

The OE* signal is an active low signal that indicates that a memory read cycle is in progress. The signal's timing is configuration-dependent.

Write enable (WE*)

The WE* signal is an active low signal that indicates that a memory write cycle is in progress. The signal's timing is configuration-dependent.

Byte enable (BEx*)

The BE* signals identify which 8 bytes of the 32-bit data bus are active during any given system bus memory cycle. The BE* signals are active low. The NET+ARM has a 32-bit data bus, which allows for four byte enable signals. The byte enable signals (BE3*, BE2*, BE1*, BE0*) control which portion of the data bus is active.

One BE* signal is active low for each valid data byte. This BE* signal is used when the microprocessor wants to transfer less than 32 bits in a memory cycle.

BE3* is active for 8-bit bytes. BE3* and BE2* are active for 16-byte transfers. All BE* signals are active for 32-bit transfers. The CAS* signals (see "Column address strobes (CAS)" on page 140) perform the same function when interfacing with FP or EDO DRAM. The BE* signals are used as DQM signals when interfacing to SDRAM.

Table 52 shows the relationship between the byte enable signals and the data bus lanes. Note the similarity between this table and Table 51: "CAS* lane configurations" on page 140.

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		
BE* lane configuration for 32-bit peripherals					
BE3*	D31:D24	1	0		
BE2*	D23:D16	0	1		
BE* lane configuration for 16-bit peripherals					
BE3*	D31:D24				
BE* lane configuration for 82-bit peripherals					

Table 52: BE* lane configurations

Transfer Acknowledge (TA*)

The TA* signal indicates the end of the current system bus memory cycle. TA* is sampled on the rising edge of BCLK.

The TA* signal is either a NET+ARM input or output.

- As an output, TA* signals the `T2` part of a transfer cycle, The NET+ARM reads the data bus on the rising edge of BCLK when TA* is low.

- As an input, TA* has the same function but is provided by the external device to tell the NET+ARM when to read the data bus and when to terminate the transfer.

Transfer Error Acknowledge (TEA*)

The TEA* signal is either a NET+ARM input or output, and has two functions.

- As an output, TEA* can be configured to signal a bus error or the last transfer in a burst cycle. The functionality depends on how the BURST bit (in the CSBAR register) is set.
- As an input, the TA* signal has the same functionality but the signal is provided by the external device.

Table 53 shows the TA* and TEA* signal configuration for terminating cycles.

Signal	TA*	TEA*
Burst cycle termination	0	0
Normal termination	0	1
Error termination	1	0
Waiting for completion	1	1

Table 53: Peripheral cycle termination

Basic configurations

Any of the five chip selects can be configured to any of four basic modes. The four basic modes are static RAM (SRAM), fast page (FP) DRAM, extended data output (EDO) DRAM, and synchronous DRAM (SDRAM). All chip selects configured as DRAM must be configured to the same mode.

Static RAM

Figure 13 shows the memory signals that are available when the chip select is configured as static RAM (SRAM). There are two timing variations — synchronous and asynchronous — for both the read and write cycles. See "SRAM timing," beginning on page 426, for synchronous and asynchronous read/write timing diagrams.

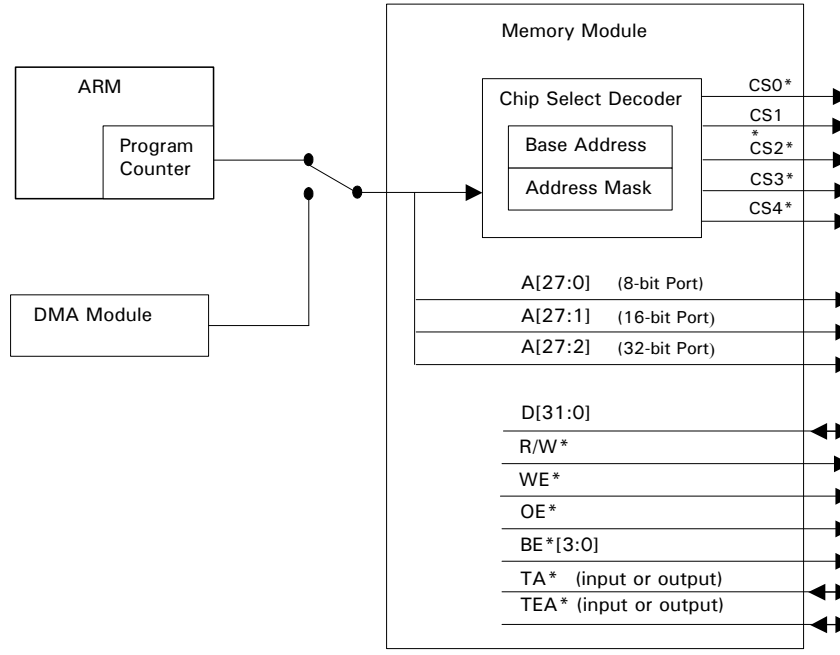


Figure 13: SRAM chip select configuration

Configuring a chip select for SRAM

Use these procedures to configure a chip select for SRAM.

Note: **SET** means you must use the value specified for the register field. For example, *set the V bit* means select 1 for the Vbit field.
CHOOSE means you can select a value for the field from two or more options. For example, *Choose either A27 or CS0OE** means set the bit (0 or 1) for the function you want to use.

Memory Module Control register
For CS0* only:

1. A27_F: Choose either A27 or CS0OE*.
2. A26_F: Choose either A26 or CS0WE*.

Chip Select Base Address register
The following fields are options for SRAM:

1. BASE: Works with the MASK field in the Chip Select Option register to configure address space.
2. PGSIZE: Choose the maximum page size for bursting if the BURST bit is set.
3. EXTTA: Choose to allow externally-controlled transfers.
4. BURST: Choose to allow bursting.
5. WP: Choose to disable the ability to write to a device.
6. V: Set to enable the chip select. Set this bit last.

Chip Select Option register
The following fields are options for SRAM:

1. MASK: Works with the BASE field in the Chip Select Base Address register to configure the address space.
2. WAIT: Choose to add wait states.
3. BCYC: Choose burst timing.
4. BSIZE: Choose the number of cycles in a burst.
5. PS: Choose the data bus size — 8-bit, 16-bit, or 32-bit.
6. RSYNC: Choose the read transfer timing.
7. WSYNC: Choose the write transfer timing.

The field in Chip Select Option Register B is used only when the EXTTA bit is set in the Chip Select Base Address register.

Fast page and EDO DRAM

The only difference between fast page (FP) and EDO DRAM is the RAS and CAS timing; see "Fast Page and EDO DRAM Timing," beginning on page 436. The two DRAM modes are otherwise the same, and the information in this section applies to both.

You have three options when producing the RAS/CAS addresses:

- External RAS/CAS multiplexer
- Mode-0 (10 CAS)
- Mode-1 (8 CAS)

Figure 14 shows the memory signals that are available when the chip select is configured for external DRAM multiplexing:

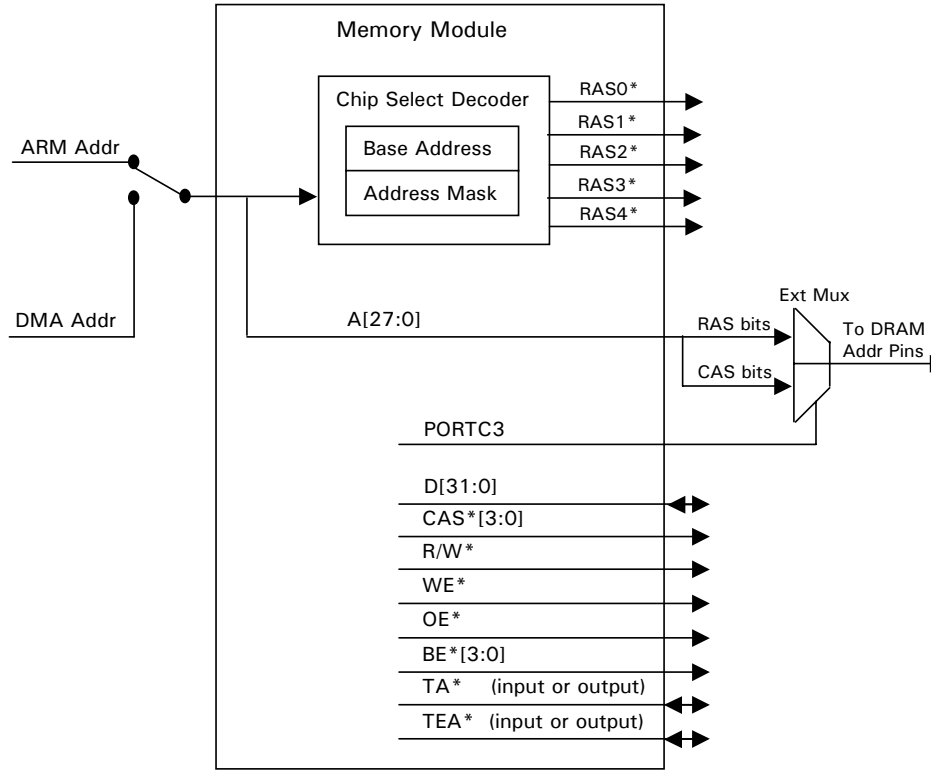


Figure 14: External DRAM multiplexer chip select configuration

Configuring a chip select for FP or EDO DRAM

Use these procedures to configure a chip select for SRAM.

Note: **SET** means you must use the value specified for the register field. For example, *set the V bit* means select 1 for the Vbit field.
CHOOSE means you can select a value for the field from two or more options. For example, *Choose either the FP or EDO timing mode* means set the bit (0 or 1) for the function you want to use.

Memory Module Control register
Set the refresh configuration:

1. RFCNT: Set the refresh count.
2. REFEN: Set the refresh enable.
3. RCYC: Set the refresh period.
4. AMUX: Choose an internal or external RAS/CAS mux for all DRAM chip selects.

Chip Select Base Address register
The following fields are options for FP or EDO DRAM:

1. BASE: Works with the MASK field in the Chip Select Option register to configure the address space.
2. PGSIZE: Choose the maximum page size for bursting if the BURST bit is set.
3. DMODE: Choose either the FP or EDO timing mode.
4. DMUXS: Choose either the internal or external RAS/CAS address multiplexer. Note that this bit is for an individual chip select and can be superseded by the AMUX setting in the Memory Module Control register.
5. EXTTA: Choose to allow externally-controlled transfers.
6. DMUXM: Choose the type of internal RAS/CAS multiplexing if the DMUXS bit is set to internal mux setting.
7. IDLE: Choose to set a null BCLK period after each DRAM memory transfer.
8. DRSEL: Set for DRAM timing.
9. BURST: Choose to allow bursting.
10. WP: Choose to disable the ability to write to a device.
11. V: Set to enable the chip select. Set this bit last.

Chip Select Option register
The following fields are options for FP or EDO RAM:

1. `MASK`: Works with the BASE field in the Chip Select Base Address register.
2. `WAIT`: Choose to add wait states.
3. `BCYC`: Choose the burst timing.
4. `BSIZE`: Choose the number of cycles in a burst.
5. `PS`: Choose the data bus size — 8-bit, 16-bit, or 32-bit.

The field in Chip Select Option Register B is used only when the EXTTA bit is set in the Chip Select Base Address register.

External multiplex RAS/CAS addressing

The NET+ARM can be configured to interface with an external address multiplexer. When in this mode, the entire address is provided as it would be when interfacing to SRAM. This mode provides enormous versatility in that virtually any RAS/CAS address combination can be wired to the multiplexer.

The MEM module sources the RAS*, CAS*, and R/W* as usual for DRAM. The module also provides a signal to control the external multiplexer. This signal appears on the NET+ARM's PORTC3 pin. The signal is low when the RAS* signal is valid and high for the CAS address. See Figure 14, "External DRAM multiplexer chip select configuration," on page 146.

DRAM Mode-0 addressing

The NET+ARM can provide an internal multiplexer to source the RAS/CAS address sequence. Many DRAMs use 10 CAS addresses during memory transfers. When the NET+ARM is configured in Mode-0 addressing, the MEM module divides the lowest 20 *used* address bits into 10 RAS and 10 CAS signals:

- A[19:0] for an 8-bit bus
- A[20:1] for a 16-bit bus
- A[21:2] for a 32-bit bus

Note, however, that the NET+ARM is not limited to interfacing to DRAMs with only 10 RAS lines. The higher order address bits that are not multiplexed can be connected to the DRAM's extra RAS pins.

Figure 15, "DRAM Mode-0 addressing," on page 150 shows how internal DRAM Mode-0 addressing interacts with the MEM module.

Table 54: "DRAM Mode-0 addressing" on page 151 describes how the NET+50 chip multiplexes the logical address signals through the physical address signals during the RAS and CAS timeframes.

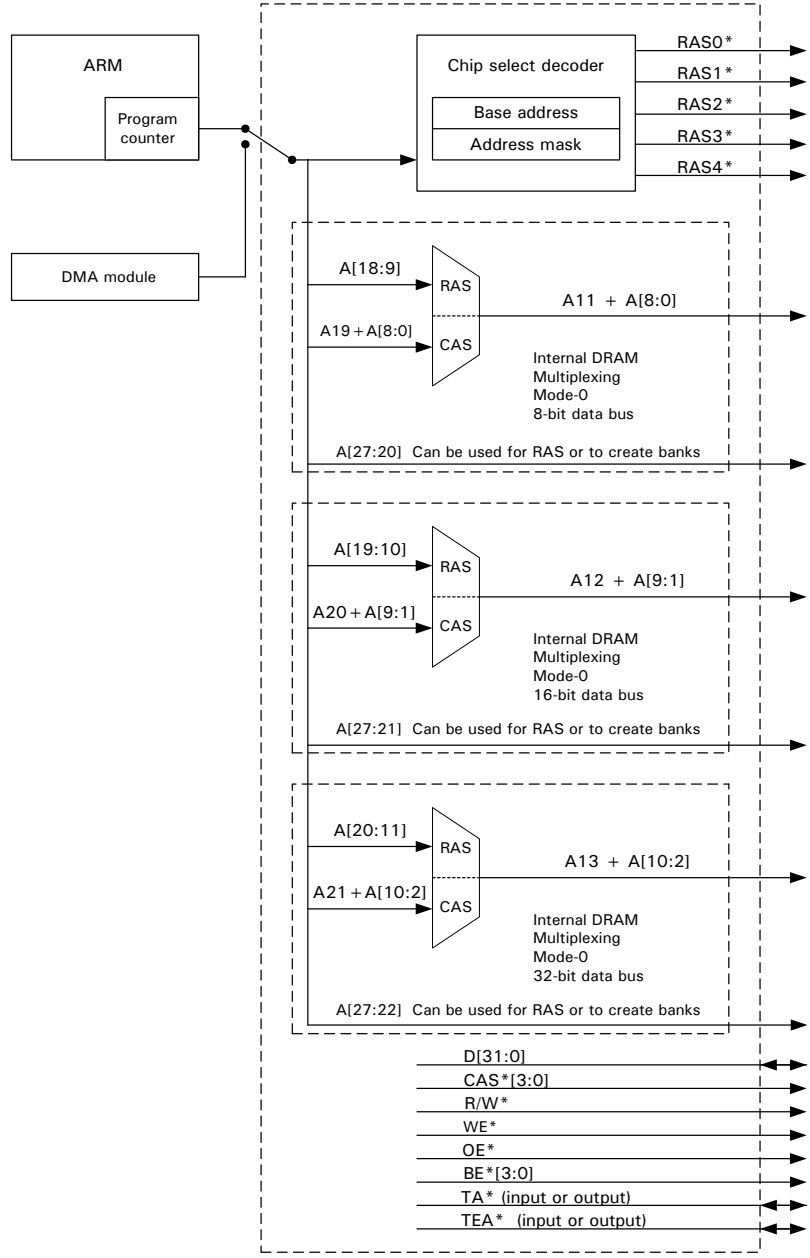


Figure 15: DRAM Mode-0 addressing

Mode-0 addressing is useful for DRAMs with 10 CAS address lines.

Multiplexed pins

NET + ARM pins		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
AMUX	FCTN														
0	RAS			18			17	16	15	14	13	12	11	10	9
1	CAS			19			8	7	6	5	4	3	2	1	0
		8- bit DRAM peripheral (20 Address Bits: 10 RAS and 10 CAS)													
DRAM pin			A9			A8	A7	A6	A5	A4	A3	A2	A1	A0	
AMUX	FCTN														
0	RAS		19			18	17	16	15	14	13	12	11	10	
1	CAS		20			9	8	7	6	5	4	3	2	1	
		16-bit DRAM peripheral (20 Address Bits: 10 RAS and 10 CAS)													
DRAM pin		A9			A8	A7	A6	A5	A4	A3	A2	A1	A0		
AMUX	FCTN														
0	RAS	20			19	18	17	16	15	14	13	12	11		
1	CAS	21			10	9	8	7	6	5	4	3	2		
		32-bit DRAM peripheral (20 Address Bits: 10 RAS and 10 CAS)													

Table 54: DRAM Mode-0 addressing

Notes:

1 Address lines A[27:20] can be used for extra RAS or to create banks with 8-bit DRAM peripherals.

2 Address lines A[27:21] can be used for extra RAS or to create banks with 16-bit DRAM peripherals.

3 Address lines A[27:22] can be used for extra RAS or to create banks with 32-bit DRAM peripherals.

DRAM Mode-1 multiplexing

Many DRAMs use 8 CAS addresses during memory transfers.

- When the NET+ARM is configured for Mode-1 addressing the MEM module divides A[21:0] into 14 RAS and 8 CAS signals when configured as an 8-bit bus.
- When configured as a 16-bus bit, A[21:1] is divided into 13 RAS and 8 CAS signals.
- When configured as a 32-bit bus, A[21:2] is divided into 12 RAS and 8 CAS signals.

In all cases, address bits A[27:22] can be used as extra RAS or bank select signals. EDO and FP DRAM can use either Mode-0 or Mode-1 addressing. SDRAM always uses Mode-1 addressing.

Figure 16, "DRAM Mode-1 addressing," on page 153 shows how internal DRAM Mode-1 addressing interacts with the MEM module.

Table 55: "DRAM Mode-1 addressing" on page 154 describes how the NET+50 chip multiplexes the logical address signals through the physical address signals during the RAS and CAS timeframes.

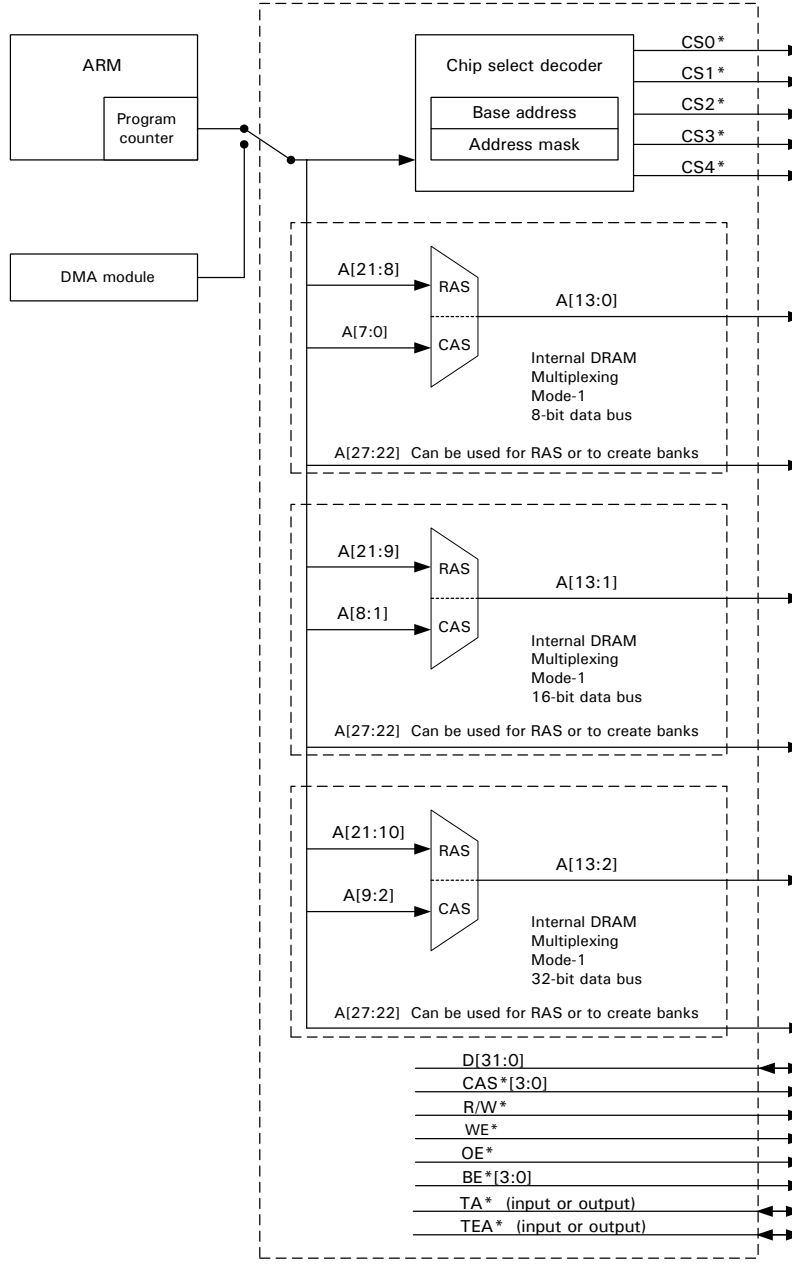


Figure 16: DRAM Mode-1 addressing

Mode-1 addressing is useful for DRAMs with 14RAS/8CAS address lines.

		Direct pins		Multiplexed pins												
NETARM pin		**A23**	**A22**	**A13**	**A12**	**A11**	**A10**	**A9**	**A8**	**A7**	**A6**	**A5**	**A4**	**A3**	**A2**	**A1**	**A0**
SDRAM pin				A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
AMUX	FCTN																
0	RAS			21	20	19	18	17	16	15	14	13	12	11	10	9	8
1	CAS			low	low	low	10	20	19	7	6	5	4	3	2	1	0
		8- bit DRAM peripheral (22 Addr Bits: 14 RAS and 8 CAS)															
SDRAM pin			A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
AMUX	FCTN																
0	RAS		22	21	20	19	18	17	16	15	14	13	12	11	10	9	
1	CAS		22	low	low	low	10	9	8	7	6	5	4	3	2	1	
		16-bit DRAM peripheral (22 Addr bits: 14 RAS and 8 CAS)															
SDRAM pin		A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0		
AMUX	FCTN																
0	RAS	23	22	21	20	19	18	17	16	15	14	13	12	11	10		
1	CAS	23	22	low	low	low	10	9	8	7	6	5	4	3	2		
		32-bit DRAM peripheral (22 Addr bits: 14 RAS and 8 CAS)															

Table 55: DRAM Mode-1 addressing

Notes:

1 Address lines A[27:22] can be used for extra RAS or to create banks with 8-bit DRAM peripherals.

2 Address lines A[27:23] can be used for extra RAS or to create banks with 16-bit DRAM peripherals.

3 Address lines A[27:24] can be used for extra RAS or to create banks with 32-bit DRAM peripherals.

SDRAM

When the Memory interface is set to SDRAM mode, the memory device must be initialized with the load-mode command before the registers are configured.

When the Load-Mode command is issued, the SDRAM looks at its address bus and configures itself based on the number it reads. The number that appears on the address bus depends on the BCYC and BSIZE field settings in the Chip Select Option register.

- BCYC can be set to a CAS latency of 1, 2, 3, or 4. This is the CAS latency you want the SDRAM to respond to.
- BSIZE must be set to 11, for full page. This is the only valid value at initialization.

The NET+ARM issues the SDRAM Load-Mode command whenever the V bit is set in the Chip Select Base Address register. You can change the BCYC and BSIZE fields after SDRAM is initialized, but the V bit should never be toggled. Otherwise, the SDRAM device can be inadvertently reconfigured with an erroneous setup.

Be sure the port size has been set to the proper width (PS field in the CHip Select Option register).

Load-Mode procedure: Setting up the Load-Mode command

Note: **SET** means you must use the value specified for the register field. For example, *set the refresh enable* means select 1 for the REFEN field. **CHOOSE** means you can select a value for the field from two or more options. For example, *Choose to allow bursting* means set the bit (0 or 1) for the function you want to use.

1 Set the BCYC field in the Chip Select Option register to the CAS latency you want to use:

BCYC	CAS latency
00	1
01	2
10	3
11	4

2 Set the BSIZE field in the Chip Select Option register to 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

3 Set the PS field in the Chip Select Option register to the proper bus size.

4 Set the DRSEL field in the Chip Select Base Address register for DRAM timing.

5 Set DMODE field in the Chip Select Base Address register to SDRAM timing mode.

6 Set the V bit in the Chip Select Base Address register to issue the Load-Mode command.

The MEM module loads the mode register with the values shown in this table:

16-bit port address	32-bit port address	Value and Function	SDRAM field
12:10	13:11	001 = Single access	Write burst mode
9:8	10:9	00 = Standard operation	Operation mode
7:5	8:6	BCYC + 1 = 001 to 100	CAS latency
4	5	0 = Sequential	Burst type
3:1	4:2	111 = Full page	Burst length

The next table shows the binary value that appears on the address during the Load-Mode command, for the four CAS latency options:

CAS latency setting	16-bit port address value	32-bit port address value
BCYC = 00, CAS lat = 1	xxx0 0100 0010 111x	xx00 1000 0101 11xx
BCYC = 01, CAS lat = 2	xxx0 0100 0100 111x	xx00 1000 1001 11xx

CAS latency setting	16-bit port address value	32-bit port address value
BCYC = 10, CAS lat = 3	xxx0 0100 0110 111x	xx00 1000 1101 11xx
BCYC = 11, CAS lat = 4	xxx0 0100 1000 111x	xx00 1001 0001 11xx

Only after the Load-Mode command has been configured can you configure the chip select. Use these procedures

Memory Module Control register
Set the refresh configuration:

1. `RFCNT`: Set the refresh count.
2. `REFEN`: Set the refresh enable.
3. `RCYC`: Set the refresh period.
4. `AMUX`: Choose an internal or external RAS/CAS mux for all DRAM chip selects.

Chip Select Base Address register
The following fields are options for SDRAM:

1. `BASE`: Works with the MASK field in the Chip Select Option register to configure the address space.
2. `PGSIZE`: Choose the maximum page size for bursting if the BURST bit is set.
3. `DMUXS`: Choose either the internal or external RAS/CAS address multiplexer. Note that this bit is for an individual chip select and can be superseded by the AMUX setting in the Memory Module Control register.
4. `EXTTA`: Choose to allow externally-controlled transfers.
5. `DMUXM`: Set the internal RAS/CAS multiplexing to Mode-1 if the DMUXS bit is set to internal mux setting.
6. `IDLE`: Choose to set a null BCLK period after each DRAM memory transfer.
7. `BURST`: Choose to allow bursting.
8. `WP`: Choose to disable the ability to write to a device.
9. `V`: This bit should have been set during the Load-Mode command. *Do not reset this bit.*

Chip Select Option register
The following fields are options for SDRAM:

1. MASK: Works with the BASE field in the Chip Select Base Address register to configure address space.
2. WAIT: Choose to add wait states after the Precharge and Active commands (see "SDRAM command descriptions" on page 161).
3. BCYC: Choose the burst timing. The value can be changed from what it was set to during the Load-Mode command.
4. BSIZE: Choose the number of cycles in a burst. The value can be can be changed from what it was set to during the Load-Mode command.
5. PS: This field should have been set before issuing the Load-Mode command. *Do not change the field.*

The field in Chip Select Option Register B is used only when the EXTTA bit is set in the Chip Select Base Address register.

Figure 17, "SDRAM (Mode-1) addressing," on page 159 shows how internal DRAM Mode-0 addressing interacts with the MEM module.

Table 56: "SDRAM (Mode-1) addressing" on page 160 describes how the NET+50 chip multiplexes the logical address signals through the physical address signals during the RAS and CAS timeframes.

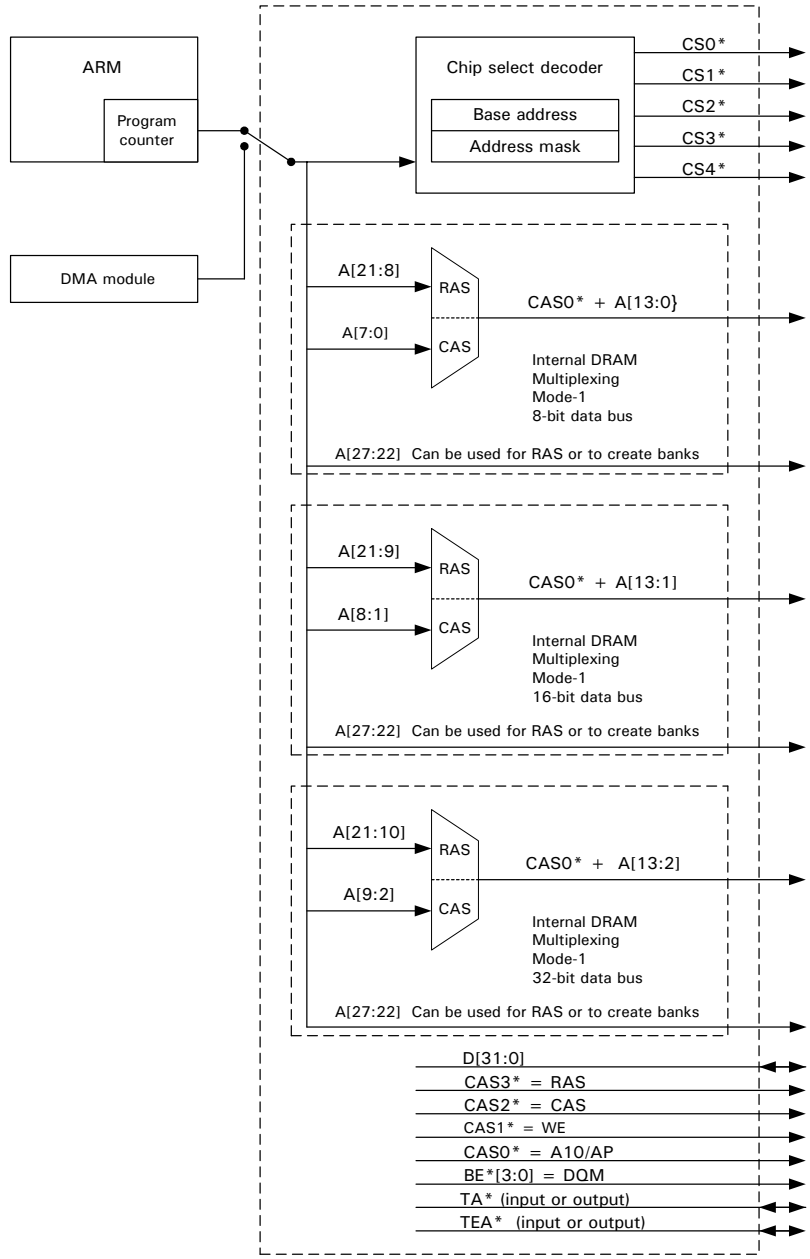


Figure 17: SDRAM (Mode-1) addressing

Mode-1 SDRAM multiplexing

SDRAM multiplexing is useful for SDRAMs with 14RAS/8CAS address lines.

		Direct pins		Multiplexed pins													
NETARM pin		**A23**	**A22**	**A13**	**A12**	**A11**	**A10**	**A9**	**A8**	**A7**	**A6**	**A5**	**A4**	**A3**	**A2**	**A1**	**A0**
SDRAM pin				A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
AMUX	FCTN																
0	RAS			21	20	19	CAS0*	17	16	15	14	13	12	11	10	9	8
1	CAS									7	6	5	4	3	2	1	0
1	L. Mode					11	CAS0*	9	8	7	6	5	4	3	2	1	0
		8- bit DRAM peripheral (22 Address Bits: 14 RAS and 8 CAS)															
SDRAM pin			A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
AMUX	FCTN																
0	RAS		22	21	20	CAS0*	18	17	16	15	14	13	12	11	10	9	
1	CAS								8	7	6	5	4	3	2	1	
1	L. Mode				12	CAS0*	10	9	8	7	6	5	4	3	2	1	
		16-bit DRAM peripheral (22 Address bits: 14 RAS and 8 CAS)															
SDRAM pin		A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0		
AMUX	FCTN																
0	RAS	23	22	21	CAS0*	19	18	17	16	15	14	13	12	11	10		
1	CAS							9	8	7	6	5	4	3	2		
1	L. Mode			13	CAS0*	11	10	9	8	7	6	5	4	3	2		
		32-bit DRAM peripheral (22 Address bits: 14 RAS and 8 CAS)															

Table 56: SDRAM (Mode-1) addressing

Important: Regarding NETARM address lines (A13), (A12), and (A11). During CAS cycles, A13:11 are equal to 0 and must not be connected to SDRAM bank select address lines. As an alternative, connect the RAS address to the SDRAM pin in the same column.

Notes:

1 Address lines A[27:22] can be used for extra RAS or to create banks with 8-bit DRAM peripherals.

2 Address lines A[27:23] can be used for extra RAS or to create banks with 16-bit DRAM peripherals.

3 Address lines A[27:24] can be used for extra RAS or to create banks with 32-bit DRAM peripherals.

SDRAM command codes

The SDRAM command codes are issued while the chip select input is active low, and are registered using the low to high transition of the synchronous clock (BCLK).

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

SDRAM command descriptions

Five signals control SDRAM functionality — chip select (CS), RAS, CAS, write enable (WE), and A10/AP. The signals are active low, and are sampled on the rising edge of the clock to perform the function indicated by the 5-bit code.

- Load-mode. Initializes SDRAM. The NET+ARM produces this command when the valid (V) bit is set in the Chip Select Base Address register.

When the V bit is set, the SDRAM loads the number on its address lines into its internal mode register. NET+ARM issues the load command and sets the address bus to produce the correct number. All SDRAM configuration options are defaulted by the NET+ARM except CAS latency. (See "Load-Mode procedure: Setting up the Load-Mode command," beginning on page 155, for more information.)

- **Precharge.** Alias *row deselect command*. You must deselect a row that has been previously accessed before you can select another row. Otherwise, two memory locations would be addressed at the same time.

  Precharge is not required for every situation; for example, it is not required when accessing a row in a different bank or accessing a different column in the same row. The NetARM keeps track of previous accesses and issues the precharge command only when needed.

- **Active.** Latches the row and bank address into the SDRAM. This command is similar to RAS in an EDO DRAM.

  Active is not required for every situation; for example, it is not required when accessing a different column in the same row as the previous transfer. The NetARM keeps track of previous accesses and issues the active command only when needed.

- **Write.** Writes the data on the data bus to a column location provided by the address bus. This command is similar to CAS in an EDO DRAM.

- **Read.** Latches the column address when issued; data will be available either 2 or 3 clocks later (depending on the CAS latency selected). This command is similar to CAS in an EDO DRAM.

- **No Operation (NOP).** Either "no operation" or continues the previous operation. If this command is issued after the read command, data continues to be sourced on each rising clock pulse (after the CAS latency).

- **Burst Terminate.** Must be issued at some point after a read command, to stop the SDRAM from sourcing data two or three cycles (depending on the CAS latency selected) after being issued. The NET+ARM reads data at this point, even though data will continue to burst for two cycles.

Burst terminate is always issued on the last data read even if there is only one data word that is read (that is, a single cycle rather than a burst cycle).

Note: If a hardware reset occurrs after a read command but before the burst terminate command, the SDRAM device continues to drive data onto the bus. This can cause a data clash with the boot code. To avoid this situation, you need to implement a *burst terminate solution*. See "Burst terminate solution," beginning on page 168, for complete information.

- **Inhibit.** Issued when the NET+ARM is accessing other devices, such as flash.

  When the CS* line is high, SDRAM is not enabled. At least two inhibit commands must be issued after the burst terminate command. Otherwise, SDRAM continues to source data for two cycles after the burst terminate command, preventing access to another memory device.

- **Refresh.** Performed when CS*, RAS*, and CAS* are low. The refresh operation is the same as a CAS-before-RAS refresh in an EDO or FP DRAM.

A10/AP Signal

The A10/AP (autoprecharge) signal has three functions, depending on which command is issued. The signal is connected to the SDRAM's address A10 input.

- During the **active** command, the signal acts as the A10 logical RAS address signal. The CAS0* pin is driven with the logical value of A20, A19, or A18, depending on the port size configuration defined in the Chip Select Option register.
  - A20 is driven on CAS0* for a 32-bit port size configuration.
  - A19 is driven on CAS0* for a 16-bit port size configuration.
  - A18 is driven on CAS0* for an 8-bit port size configuration.
- During the **precharge** command, the signal is high (1 on the CAS0* pin) to indicate that all banks should be precharged.
- During **read and write** commands, the signal is low to indicate that auto-precharge should not occur; the NET+50 chip always drives 0 during read

and write commands. When the MEM module is configured for SDRAM, the CAS0 pin performs the precharge command.

NET+50 chip SDRAM interconnect

This section describes how to interconnect the NET+50 parts to standard 16M and 64M SDRAM components.

SDRAM x16 bursting considerations

The NET+50 chip cannot perform 16-bit burst operations from an x32 SDRAM. Executing 16-bit Thumb code from a 32-bit SDRAM results in poor performance, as the NET+50 chip does not perform burst operations for the 16-bit thumb instruction fetches.

x32 SDRAM configuration

Table 57 identifies the interconnect between the NET+50 chip and SDRAM when SDRAM is used in an x32 configuration. An x32 SDRAM configuration typically is constructed using two x16 SDRAM components. In this table, the two components are called “Device 1” and “Device 2.”

NET+50 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

Table 57: X32 SDRAM configuration

NET+50 chip signal	16M SDRAM signal		64M SDRAM signal	
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	
A6	A4		A4	
A7	A5		A5	
A8	A6		A6	
A9	A7		A7	
A10	A8		A8	
A11	A9		A9	
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 57: X32 SDRAM configuration

x16 SDRAM configuration

Table 58 identifies the interconnect between the NET+50 chip and SDRAM when SDRAM is used in an x16 configuration. An x16 SDRAM configuration typically is constructed using one x16 SDRAM component.

NET+50 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
A12		A11

Table 58: X16 SDRAM configuration

NET + 50 chip signal	16M SDRAM signal	64M SDRAM signal
A20	BA	
A21		BA0
A22		BA1
BCLK	CLK	CLK
VCC	CKE	CKE
D31-D16	D15-D0	D15-D0

Table 58: X16 SDRAM configuration

WAIT configuration

The WAIT configuration provides the SDRAM T_{RCD} and T_{RP} parameters.

- When WAIT is configured with a value of 0, the active and precharge commands can be followed immediately by another command on the next active edge of BCLK.
- When WAIT is configured with a value larger than 0, wait states are inserted after the active and precharge commands before another command can be issued.

Burst terminate solution

If the NET+ARM experiences a reset after a read command and before a burst terminate command, SDRAM components continue to drive the data bus. This prevents the NET+ARM from being able to reboot from CS0*. Two options are available to avoid this situation.

- Option 1 is a three-gate solution used only when a hardware reset is the only possible way to reset the application.
- Option 2 requires more circuitry but can recover from hardware, watch-dog, or software reset conditions.

Option 1

Figure 18 presents both a timing diagram (top) and schematic (bottom) for the three-gate option.

The burst terminate command is asserted whenever the chip select (CS*) and the write enable (WE*) signals on the SDRAM are low. This condition occurs only during state 3 (B1) of the timing diagram. During normal operation (state 2), the BUR_TERM signal is high, which allows the CAS1*/WE* * and the CS* signal (see "SDRAM command codes" on page 161) to work in conjunction with the other SDRAM signals to produce other SDRAM commands (for example, precharge or active).

During reset, all GPIO lines are configured as inputs. The pullup resistor attached to the GPIO line assures that this signal will go high during reset. After reset, the firmware must set this GPIO signal low to produce the T3 state if a hardware reset occurs.

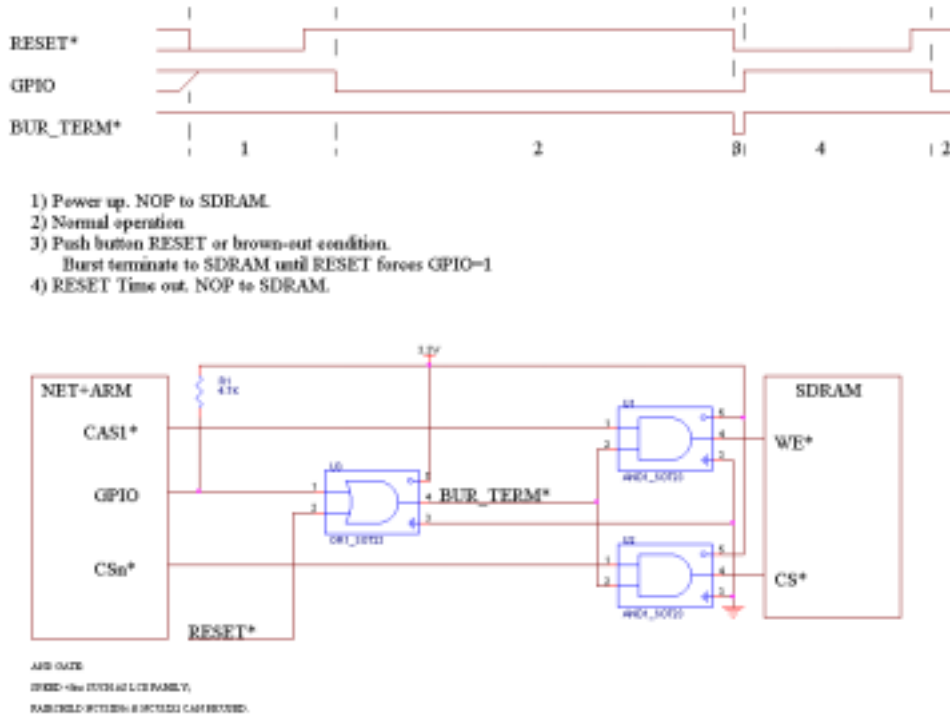


Figure 18: Option 1

Option 2

Option 2 has additional components that give the added capability of producing the burst terminate command when the watch-dog or software reset occurs.

Figure 19 presents both a timing diagram (top) and schematic (bottom) for option 2.

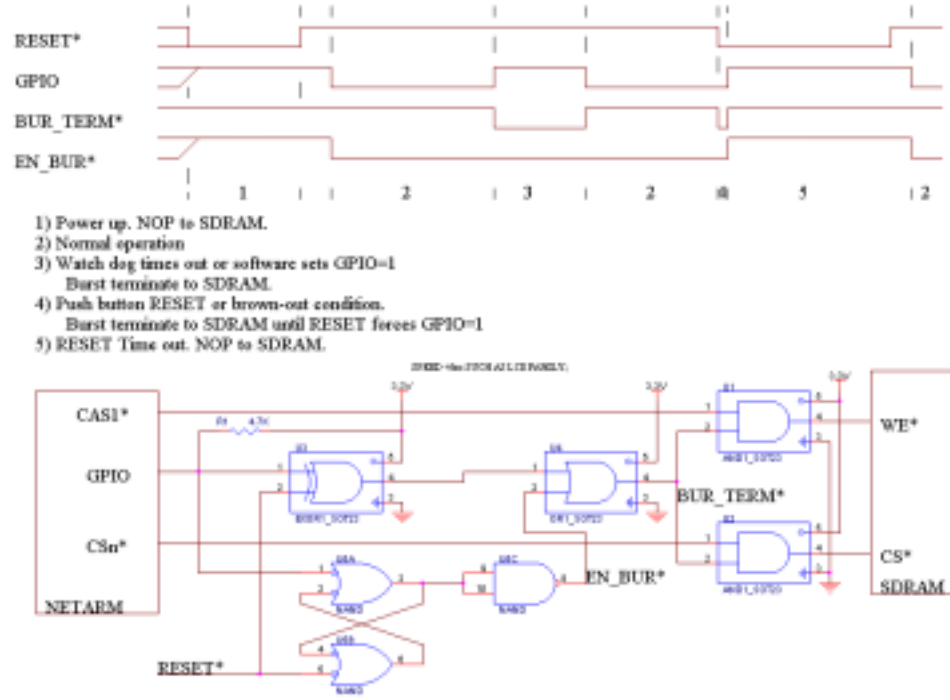


Figure 19: Option 2

If the application needs to use the internal watch-dog and software RESET condition, additional gates are required. The same three gates as used in Option 1 are used with the OR gate connected to a gate-constructed buffered flip-flop and to the output of an EOR gate. The flip-flop inhibits BUR_TERM during cold starts. RESET* low and GPIO high leaves this flip-flop in the inhibit state.

A powerup reset condition causes NOP commands to be issued to the SDRAM since RESET* is low and the GPIO is high (powerup default). After RESET* is driven high, system initialization firmware drives the GPIO signal low. An external RESET* input or a brownout condition causes the burst terminate command to be issued immediately, since both RESET* and GPIO are low. The EOR gate is forced low by its inputs being equal. The buffered flip-flop is forced to the enable mode by GPIO being low. The burst terminate command continues

until the GPIO is forced to a value of 1 as a result of the RESET after an internal sampling delay. NOP commands are sent to the SDRAM until the external RESET* signal returns to 1. The circuit becomes armed again when firmware drives the GPIO signal to 0.

A watchdog time-out or a software RESET causes an internal hardware RESET, forcing the GPIO to a value of 1. Since both RESET* and GPIO are equal, the EOR will output a low. The buffered flip-flop is still in the enable mode since RESET* has not been driven low. The SDRAM will be issued burst terminate commands until the GPIO bit is set to 0. Firmware must set the GPIO to a value of 0 to enable use of the SDRAM.

The following code example enables the GPIO fix:

```
// workaround for SDRAM burst terminate using PORTC6
// Set direction of PORTC6 to OUTPUT
(*NCC_GEN).portc.reg |= 0x00400000;
// Set C6 mode to GPIO, value to 0
(*NCC_GEN).portc.reg &= 0xbfffffbf;
```

This code must be placed soon after the detection and initialization of SDRAM. It is acceptable to enable the fix after the first read (which is likely to come during detection) from the SDRAM.

DMA Controller Module

CHAPTER 9

The NET+50 DMA controller supports ten DMA channels that facilitate the movement of data between external memory and internal peripherals, minimizing CPU intervention. All ten DMA channels support both peripheral-to-memory (fly-by) and memory-to-memory transfers. Four DMA channels support externally-controlled transfers in both fly-by and memory-to-memory modes.

Note: The information in this chapter applies to both the NET+50 and NET+20M chips, unless otherwise noted.

DMA transfers

Fly-by operation

There are two modes of fly-by operation:

- **Read mode.** Transfers data from memory to a peripheral device.
- **Write mode.** Transfers data from a peripheral device to memory.

When configured for fly-by operation, the DMA controller transfers data from one of the NET+50's internal peripherals (Ethernet, serial, 1284, and ENI) to a memory location. 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. Each peripheral is assigned a dedicated DMA channel, as shown in Table 59 on page 176.

Figure 20 is a simple representation of DMA fly-by mode:

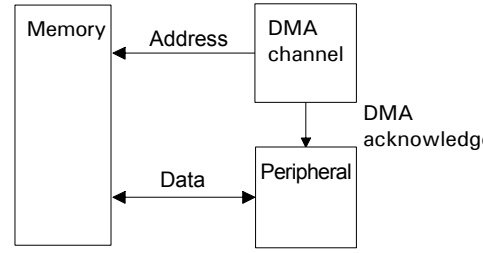


Figure 20: DMA fly-by transfers

Memory-to-memory operation

When configured for memory-to-memory (read-to-write) operations, the DMA controller transfers data between two memory locations, using a temporary holding register between read and write operations. Two memory cycles are executed; each read cycle is followed by a write cycle.

Figure 21 is a simple representation of DMA memory-to-memory mode:

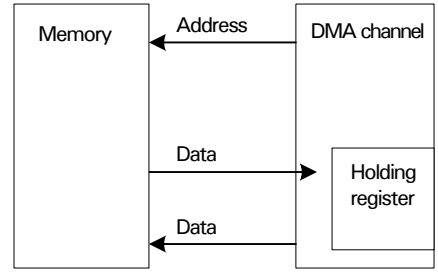


Figure 21: DMA memory-to-memory transfer

DMA module

Figure 22 shows a block diagram of the DMA module. The DMA module has a centralized single 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 the DMA channels in parallel by context-switching from channel to channel. The DMA channel arbiter determines the channel on which the machine is operating.

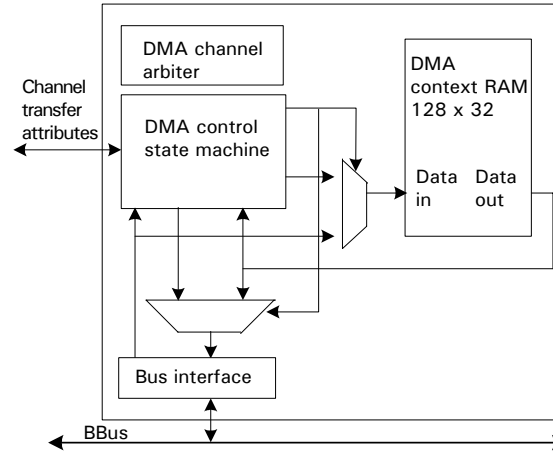


Figure 22: DMA channel block diagram

The arbiter requires six BCLK cycles to save the current context, one cycle to switch and six to load new context. The DMA controller, then, requires 13 clock cycles to transition from one channel to another before starting transfer activity. Consider this when evaluating bus bandwidth.

Note: The 13 DMA cycles are usually hidden by CPU instructions because closing and opening a DMA channel is handled independently by the DMA controller and occurs while the CPU is accessing the bus.

DMA controller assignments

Table 59 and Table 60 show how the DMA channels are used in NET+50 chip applications. There are two basic configurations:

- Configuration 1 is used for *IEEE 1284 applications,* which use one to four IEEE 1284 parallel ports and two serial ports.
- Configuration 2 is used in *MIC applications*, which provide shared RAM, FIFO mode, and two serial ports.

The D2:D0 bits in the MIC Control register determine which modes are set; see Table 106, "MIC configurations," on page 332 for information.

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

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

Table 59: DMA IEEE 1284 mode channel assignments

DMA channel	Configuration 1: IEEE 1284 mode	DMA direction
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

Table 59: DMA IEEE 1284 mode channel assignments

DMA channel	Configuration 2: MIC mode	DMA direction
1	Ethernet receiver	FB Write
2	Ethernet transmitter	FB Read
3	FIFO mode receiver/memory-to-memory	FB Write
4	FIFO mode transmitter/memory-to-memory	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 60: DMA MIC mode channel assignments

Important: The NET+20M chip uses a configuration similar to the MIC mode configuration, with the following changes:

DMA channel 3 is external DMA 1 instead of FIFO mode receiver/memory-to-memory.

DMA channel 4 is external DMA 2 instead of FIFO mode transmitter/memory-to-memory.

DMA Channel

The DMA channel is configured and controlled by a combination of configuration registers and a buffer descriptor. Figure 23 on page 179 shows all possible DMA configurations, combined into one diagram:

- Fly-by (peripheral-to-memory)
- Memory-to-memory
- External transfers

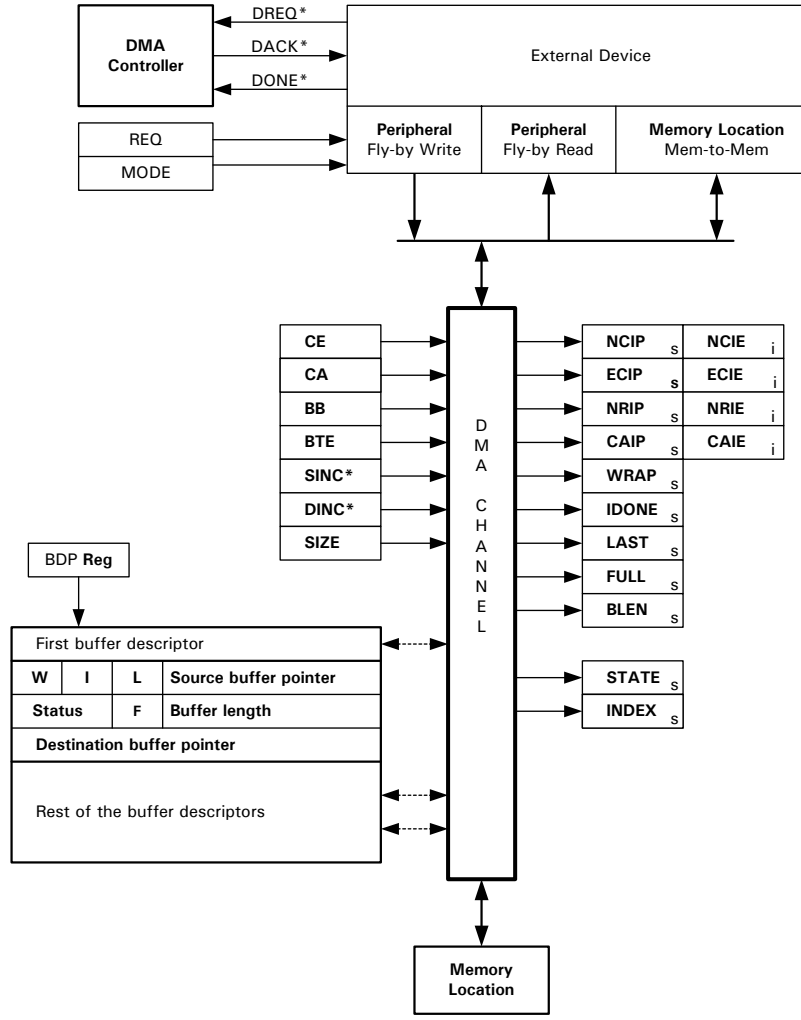


Figure 23: DMA channel structure

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 using the Enable bit in the Channel Control register. After 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 next to the first descriptor. The final buffer descriptor is defined with its W (wrap) bit set. When the DMA channel encounters the W bit, 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 unpredictably.

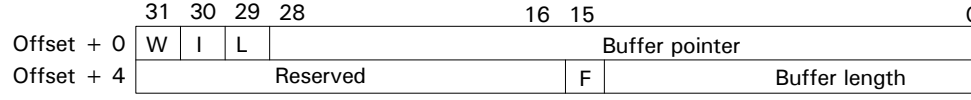


Figure 24: DMA buffer descriptor — Fly-by-mode

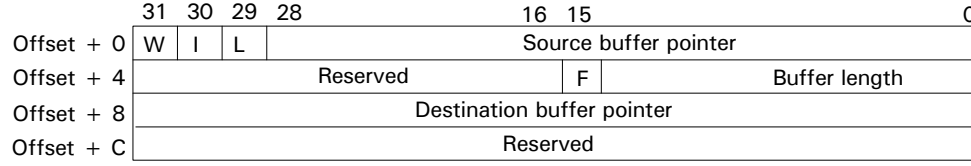


Figure 25: DMA buffer descriptor — Memory-to-memory mode

Bit	Description
W	When the W (wrap) bit is set, it informs the DMA controller that this is the last buffer descriptor within the continuous list of descriptors. The first buffer descriptor is found using the initial DMA channel buffer descriptor pointer. ■ When W is not set (0), the next buffer descriptor is found using an offset from the current buffer descriptor. An unset WRAP bit acts as a continuation indicator; that is, when W = 0, the DMA channel continues to address buffer descriptors until it encounters buffer descriptor with W set to 1. ■ When W is set (1), the bit wraps around to the first buffer descriptor. See Figure 26, "DMA Buffer descriptor structure," on page 183 for an illustration.
I	When the I bit is set, it tells 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	When the L bit is set, it informs the DMA controller that this buffer descriptor is the last descriptor and completes an entire message frame. The DMA controller uses this bit to signal the peripheral. Use this bit 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 and clears this bit after emptying a buffer. A DMA channel does not try to empty a buffer with the F bit clear. Similarly, a DMA channel does not try to fill a buffer with the F bit set. When 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 to begin DMA operations. Memory-to-memory transfers will not start unless the F bit is set to 1.

Table 61: DMA Buffer Bit Descriptions

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.

Buffer length

The buffer length field is used in peripheral-to-memory, memory-to-peripheral, and memory-to-memory operations. Buffer length can be up to 32,768 bytes.

Peripheral-to-memory

Uses this field 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 receive data byte count.

Memory-to-peripheral

Uses this field 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 data byte count (useful during error conditions).

Memory-to-memory

Uses this field in fly-by memory-to-memory 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 data byte count (useful during error conditions).

Destination address pointer

The destination address pointer is used only when the DMA channel is configured for memory-to-memory operations, and provides the "write" address portion of the transfer. The destination address pointer must start on the same byte boundary as the source address pointer.

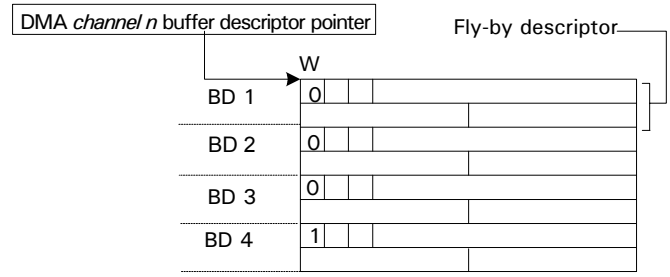


Figure 26: DMA Buffer descriptor structure

Multiple buffer descriptors are necessary when you need to provide for more bytes than the length of a single buffer descriptor. Maximum buffer length is 32,768 bytes. If the DMA channel is larger than 32,768 bytes, one or more buffer descriptors (as needed) must be appended to the first buffer descriptor. For example, for 40,000 bytes, two buffer descriptors are used.

DMA channel configuration

Each DMA channel has a block of configuration space that is mapped into the DMA module configuration space as shown:

Address	Register
FF90 0000	DMA 1 buffer descriptor pointer
FF90 0010	DMA 1 A control register

Table 62: DMA channel configuration

Address	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 1C 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
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

Table 62: DMA channel configuration

Address	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

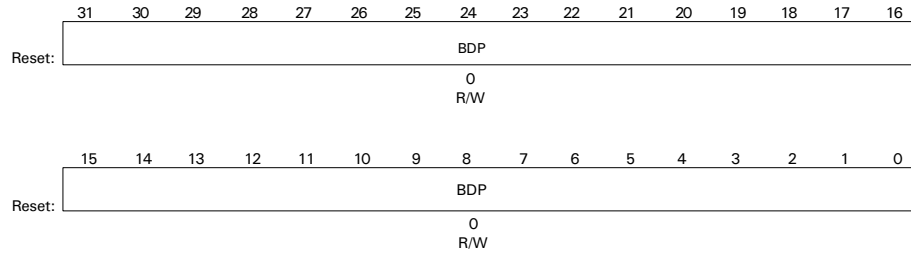
Table 62: DMA channel configuration

DMA registers

Each DMA channel has three registers: the buffer descriptor pointer, the DMA Control register, and the DMA Status/Interrupt Enable register. Each register contains the same information for each channel. The registers are 32-bits; all bits are set to 0 on reset for the buffer descriptor pointer and DMA Control register.

DMA buffer descriptor pointer

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

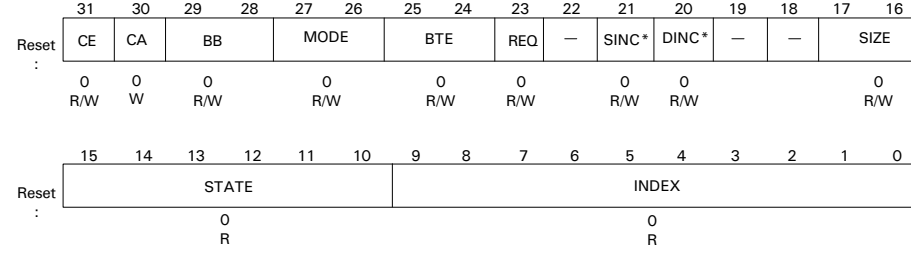


The DMA buffer descriptor pointer contains the address of the first buffer descriptor in a contiguous list of descriptors. The last descriptor in the list is identified with the W bit set in the buffer descriptor header.

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, and each buffer descriptor identifies a block of data buffers with a different size. This feature allows the Ethernet receiver to choose the optimum buffer size for the incoming packet.

DMA Control register and bit definition

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



Code (Bit #)	Definition of code	Description
CE (D31)	DMA channel enable	Must be set only after the other channel mode bits are set. The DMA channel begins reading the first buffer description when CE is set to 1.
CA (D30)	Channel abort request	When the channel abort (CA) bit is set, it 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 (D29:28)	Bus bandwidth field	00 – 100%; no limit 01 – 75% 10 – 50% 11 – 25% Determines how often the DMA channel can arbitrate for access to the bus. When set to 25%, the channel can only arbitrate one out of four times; 50% gets two out of four times; 75% gets three out of four times; 100% allows the DMA channel to arbitrate each time.
MODE (D27:26)	DMA operation 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 Identifies the data transfer mode. Each DMA channel can be configured to operate in either fly-by or memory-to-memory mode.
BTE (D25:24)	Burst transfer enable: fly-by mode and memory-to-memory mode	Determines whether the DMA channel can use burst transfers through the bus. 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). **Note:** The BURST bit in the Chip Select Base Address register for the chip select associated with the DMA memory location must also be set to allow bursting. See Chapter 8, "Memory Controller Module."

Table 63: DMA Control register bit definition

Code (Bit #)	Definition of code	Description
BTE *continued*	Burst transfer enable: fly-by mode and memory-to-memory mode	**Fly-by mode:** 00 – 1 Operand 01 – 2 Operands 10 – 4 Operands 11 – Reserved In fly-by mode, the BTE field controls the maximum number of operands the DMA controller moves each time it acquires control of the BBUS. When performing DMA to an internal peripheral, the operand size is always 32-bits. When performing DMA to an external peripheral, the SIZE field defines the operand size. The DMA controller can move information using burst cycles. If the attached memory peripheral device cannot support bursting or the peripheral terminates the burst, the maximum number of bytes defined by BTE is not achieved. **Memory-to-memory mode:** 00 – No burst 01 – 8 Byte burst 10 – 16 Byte burst 11 – Reserved When operating in memory-to-memory mode, the BTE field controls the maximum number of bytes the DMA controller moves each time it acquires control of the BBUS. The DMA controller moves information using burst cycles. If the attached source memory peripheral device cannot support bursting or the source peripheral terminates the burst, the maximum number of bytes defined by BTE is not achieved. The DMA channel always delivers to the destination peripheral the same number of bytes read from the source peripheral regardless of whether the destination peripheral can support bursting. If the destination peripheral cannot support bursting, the DMA controller issues multiple bus cycles to complete the data move. When operating in memory-to-memory mode, the SIZE field determines the size of each operand moved. In general, it is always more efficient to use a size of 32 bits; different values for SIZE are required, however, when the address alignment boundaries for the source and destination are mismatched.

Table 63: DMA Control register bit definition

Code (Bit #)	Definition of code	Description
REQ (D23)	Channel request source	0 – Internal Peripheral 1 – External Peripheral Allows channels 3, 4, 5, or 6 to be attached to either an internal peripheral or an external peripheral (THOUGHT IT WAS EXTERNAL ONLY). When an internal peripheral is selected, the DMA channel is attached to the peripheral defined in Table 59, "DMA IEEE 1284 mode channel assignments," on page 176. When an external peripheral is selected, channels 3 or 5 interface with an external peripheral using handshaking signals multiplexed through PORTA. Channels 4 or 6 interface with an external peripheral using handshaking signals multiplexed through PORTB. When the REQ bit is set to 1 in DMA channel 3, the REQ bit in DMA channel 5 is ignored and always assumed to be 0. Channel 5 can interface only with an external peripheral through PORTA with the REQ bit in DMA channel 3 set to 0. When the REQ bit is set to 1 in DMA channel 4, the REQ bit in channel 6 is ignored and always assumed to be 0. Channel 6 can interface only with an external peripheral through PORTB with the REQ bit in DMA channel 4 set to 0.
SINC* (D21)	Source address increment	SINC* 0 – Increment source address pointer 1 – Do not increment source address pointer Can be used to control whether the source address pointer is incremented after each DMA transfer. This bit is used by the DMA controller in all modes whenever referring to a memory address. This bit is not applicable when the source is an internal peripheral resource.
DINC* (D20)	Destination address increment	DINC* 0 – Increment destination address pointer 1 – Do not increment destination address pointer Can be used to control whether the destination address pointer is incremented after each DMA transfer. This bit is used by the DMA controller in all modes whenever referring to a memory address. This bit is are not applicable when the destination is an internal peripheral resource.

Table 63: DMA Control register bit definition

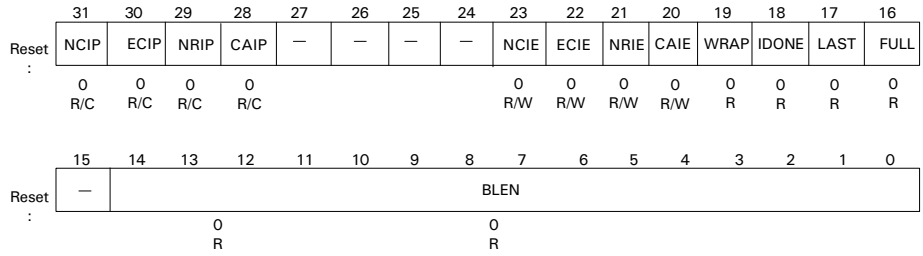
Code (Bit #)	Definition of code	Description
SIZE (D17:16)	Data operand size	00 – 32-bit 01 – 16-bit 10 – 8-bit 11 – Reserved Used by the DMA controller whenever configured for external request mode (REQ bit; Channels 3, 4, 5, and 6 only) or memory-to-memory DMA mode. These bits define the size of each DMA transaction.
STATE (D15:10)	Current DMA channel state	000000 – IDLE 000001 – Load source buffer address 000100 – Load destination buffer address 001000 – First operand 010000 – Memory-to-memory second operand 100000 – Update buffer description Describes the current state of the DMA controller state machine.
INDEX (D09:00)	Current DMA channel buffer descriptor index	Identifies the DMA controller's current byte offset pointer relative to the DMA buffer descriptor pointer.

Table 63: DMA Control register bit definition

DMA Status/Interrupt Enable register and bit definition

Under normal operations, only the NCIP and NRIP interrupts are used; the other interrupts indicate a failure. The interrupts are acknowledged by writing a 1 to the register location, which resets the interrupt condition.

Address = FF90 0014 /N34 / 54 / 74 / 94 / B4 / D4 / F4 / 114 / 134 / 154 / 194



The interrupt enable (IE) bits can be set to cause an interrupt to occur when the corresponding interrupt status bit is set in the DMA 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.

Code (Bit #)	Definition of code	Description
NCIP (D31)	Normal completion interrupt pending	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 (D30)	Error completion interrupt pending	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.

Table 64: DMA Status/Interrupt Enable register bit definition

Code (Bit #)	Definition of code	Description
NRIP (D29)	Buffer not ready interrupt pending	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 (D28)	Channel abort interrupt pending	Set when the DMA channel detects the CA bit set in the Control register. When CAIP is set, the DMA channel stops 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, using firmware, before the CAIP is cleared. Failure to reset the CA bit causes the subsequent buffer descriptor to also abort.
NCIE (D23)	Normal completion interrupt enable	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. **Note:** NCIE and the DMA buffer descriptor I bit should not be used at the same time.
ECIE D22)	Error completion interrupt enable	
NRIE (D21)	Buffer not ready interrupt enable	
CAIE (D20)	Channel abort interrupt enable	

Table 64: DMA Status/Interrupt Enable register bit definition

Code (Bit #)	Definition of code	Description
WRAP (D19)	Last descriptor in descriptor list	Provided for debugging purposes only. These bits serve no useful purpose to the firmware device driver.
IDONE (D18)	Interrupt on done	
LAST (D17)	Last buffer descriptor in current data frame	
FULL (D16)	Buffer full indicator	
BLEN (D14:00)	Remaining byte transfer count	

Table 64: DMA Status/Interrupt Enable register bit definition

Ethernet receiver considerations

When an Ethernet frame is received, DMA channel 1 searches the four buffer descriptors for the optimum buffer size. The search order is A, B, C, 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 DMA Control register whose channel enable (CE) bit is zero.

Because interrupts are set when the DMA channel encounters buffers that are not ready, 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.

An Ethernet receive FIFO overrun condition can occur (the FIFO becomes full while receiving an Ethernet packet) if insufficient buffers are allocated by the application. If this condition occurs (signaled by the NRIP bit in the DMA Channel 1 Status register; see "Ethernet Receive Status register and bit definitions,"

beginning on page 220) you must use the following procedure to guarantee successful operation:

1. Set the ERXDMA bit in the Ethernet General Control register to aero.
2. Set the ERX bit in the Ethernet General Control register to zero.
3. Set the channel enable bit in the DMA control register to zero.
4. Add new buffers for Ethernet DMA.
5. Restore the DMA Control register.
6. Restore the ERX bit.
7. Restore the ERXDMA bit.

External peripheral DMA support

NET+50 DMA channels 3, 4, 5, and 6 can be set up for external DMA transfers, and use three signals (DREQ*, DACK*, and DONE*) to facilitate communications between the NET+50 and an external device.

Note: You can use (turn on) only one DMA channel at a time.

It is up to the external device to source or react to these three signals. The NET+50 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.

Signal description

Figure 27 is a simple timing diagram of the DREQ*, DACK*, and DONE* signals. See "External DMA timing," beginning on page 454, for more detailed diagrams.

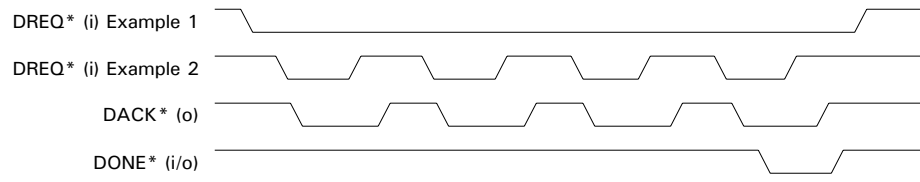


Figure 27: External DMA timing

Signal	Description
DREQ*	An input to the NET+50, 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 asserts the DREQ* signal. The DREQ* can either stay low until all data transfers are complete (see the first example of DREQ* in Figure 5) or be asserted once for each transfer (see the second example of DREQ* in Figure 5). Either method is treated the same by the NET+50.
DACK*	An input to the external device, sourced by the NET+50. After the NET+50 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*	Either an input or an output, depending on the configuration of the direction bit in the GPIO configuration register. **Input for fly-by write:** When the transfer is a fly-by write (external device to memory), the external device can send DONE* to the NET+50 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+50 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+50 receives the DONE* signal, it closes 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. **Output for fly-by read, fly-by write, and memory-to-memory:** When the transfer is a fly-by read (memory to external device), the NET+50 can send DONE* to the external device to indicate that the last DMA buffer has been emptied. The NET+50 asserts 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.

Table 65: External peripheral DMA support signal descriptions

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+50 pins that have the DREQ*, DACK*, and DONE* signals are multifunctional and are shared with GPIO ports A and B. PORTA has the DMA channel 3 and 5 signals and PORTB has the DMA channel 4 and 6 signals. Select the DMA 3 or 5 signals with the PORTA Register (0xFFB00020) and the DMA 4 or 6 signals with the PORTB 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.

	Pin number			
Channel signal	**PQFP**	**BGA**	**Signal name**	**Port A number**
DMA 3- or 5-channel signals	39	F16	DREQ1*	6
	43	E16	DACK1*	2
	45	D14	DONE1*	0
DMA 4- or-6-channel signals	47	C15	DREQ2*	6
	51	C16	DACK3*	2
	55	A16	DONE2*	0

Table 66: DMA channel signals

Enable the channel 3 or 5 signals with the PORTA register:

- `0xFFB00020 = 0x45040000` (enable DREQ1*, DACK1*, and DONE1* = input)
- `0xFFB00020 = 0x45050000` (enable DREQ1*, DACK1*, and DONE1* = output)

Enable the channel 4 or 6 signals with the PORTB register:

- `0xFFB00024 = 0x45040000` (enable DREQ2*, DACK2*, and DONE2* = input)
- `0xFFB00024 = 0x45050000` (enable DREQ2*, DACK2*, and DONE2* = output)

Fly-by mode

In fly-by mode, the NET+50 treats the external device as an I/O port that it can write to or read from. The NET+50 provides a 16- or 32-bit data bus, a 28-bit address bus, a read/write signal and the DMA transfer signals. Figure 28 shows the signals needed for external fly-by DMA transfers.

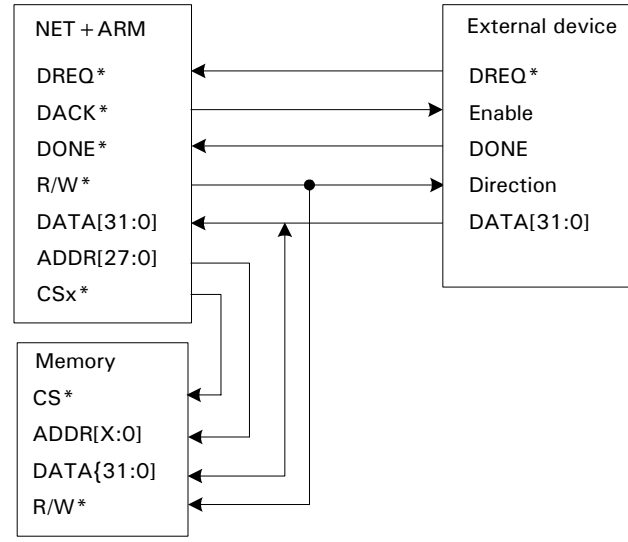


Figure 28: Hardware needed for external fly-by DMA transfers

Fly-by write

During a fly-by write cycle, the external device must provide data that will be written to memory. When the external device has data to write, it should assert DREQ* low. The NET+50 then asserts DACK* and R/W* low and provides the address of the memory location where the data will be written. The NET+50 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. When the external device has no more data to send, it should assert DONE* low.

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+50 then asserts DACK* low and provides the address of the memory location where the data will be read. Data is valid when DACK* goes low.

Memory-to-memory mode

In memory-to-memory mode, the NET+50 treats the external device as an addressable block of memory. The NET+50 provides the external device the same signals it would in fly-by mode, and it provides the address bus and a CSx* signal so it can access the external device's memory. Figure 29 shows the signals needed for external memory-to-memory DMA transfers.

The configuration of the buffer descriptor determines which side (NET+50 memory or external device memory) sources the data. If the source buffer pointer points to external device memory, the destination buffer pointer must point to NET+50 memory. The transfer is then similar to a fly-by write, and DREQ* and DACK* will act the same as in fly-by mode.

If the source buffer pointer points to NET+50 memory, the destination buffer pointer must point to external device memory. The transfer is then similar to a fly-by read, 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, and the NET+50 takes over and controls the memory cycles.

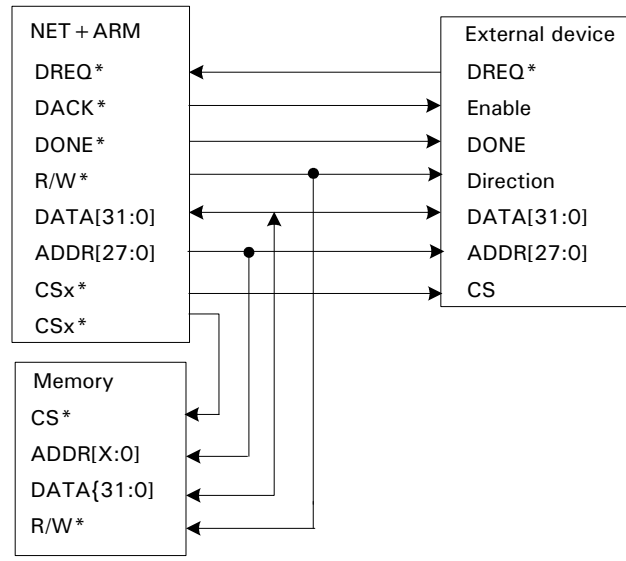


Figure 29: Hardware needed for external memory-to-memory DMA transfers

During memory-to-memory transfer, DACK* is asserted on both the read and write portions of the transfer. On the last access, the DONE* signal also asserts on both the read and write portions of the transfer.

DMA controller reset

You can set the entire DMA controller module without affecting any of the NET+50 modules by setting the DMA reset bit in the GEN System Control register to 1 and then back to 0 (see "DMARST" on page 67).

The DMA controller module is also reset by all forms of hardware and software resets.

Ethernet Controller Module

CHAPTER 10

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.

Note: The information in this chapter applies to both the NET+50 and NET+20M chips, unless otherwise noted.

Ethernet (EFE) front-end module

Figure 30 shows the hardware blocks required in the EFE module:

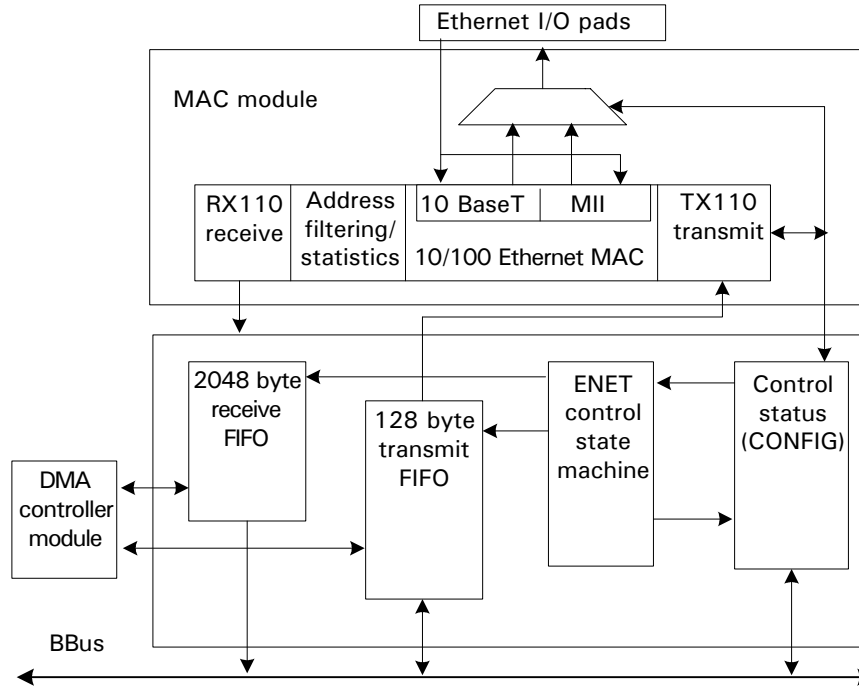


Figure 30: Ethernet controller module block diagram

The EFE provides the FIFOs and glue logic required to interface the MAC with the NET+50 chip architecture.

The basic features of the EFE include:

- Control/status registers for MAC, transmitter, and receiver.
- 128-byte transmit 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.

- 2048-byte receive FIFO.

  The large receive FIFO allows the entire Ethernet frame to be received and to 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 can use one of four differently-sized receive buffers. Only successfully received frames, with acceptable destination addresses, are committed to external system memory.

- 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.

Media access controller (MAC) module

The MAC module interfaces with the EFE module on one side and with 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:

- 100 Mbit Ethernet MAC
- MII interface
- Optional 100 Mbit physical coding sublayer
- 10 Mbit ENDEC interface
- MII management function
- Address filtering
- Statistics gathering

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

The MAC module provides both 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.

Note: 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.

Ethernet controller configuration

The Ethernet controller has a block of configuration space that is mapped into the DMA module configuration space as shown:

Address	Register
FF80 0000	Ethernet General Control register
FF80 0004	Ethernet General Status register
FF80 0008	Ethernet FIFO Data register (primary address)
FF80 000C	Ethernet FIFO Data register (secondary address)
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	Back-to-Back IPG Gap Timer register

Table 67: Ethernet controller configuration

Address	Register
FF80 0444	Non Back-to-Back IPG Gap Timer register
FF80 0448	Collision Window/Collision Retry register
FF80 0460	Transmit Packet Nibble Counter register
FF80 0464	Transmit Byte Counter register
FF80 0468	Retransmit Byte Counter register
FF80 046C	Transmit Random Number Generator
FF80 0470	Transmit Masked Random Number
FF80 0474	Transmit Counter Decodes
FF80 0478	Test Operate Transmit Counters
FF80 0480	Receive Byte Counter
FF80 0484	Receive Counter Decodes
FF80 0488	Test Operate Receive Counters
FF80 04C0	Link fail counter
FF80 0500	10 Mbit jabber counter
FF80 0504	10 Mbit 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
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

Table 67: Ethernet controller configuration

Address	Register
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 Multicast hash table
FF80 05D4	SAL Multicast hash table
FF80 05D8	SAL Multicast hash table
FF80 05DC	SAL Multicast hash table

Table 67: Ethernet controller configuration

Ethernet Registers

There are six standard Ethernet registers in the Ethernet controller:

- General Control
- General Status
- FIFO Data register (primary address)
- FIFO Data register (secondary address)
- Transmit Status
- Receive Status

Each register is a 32-bit register unless otherwise noted.

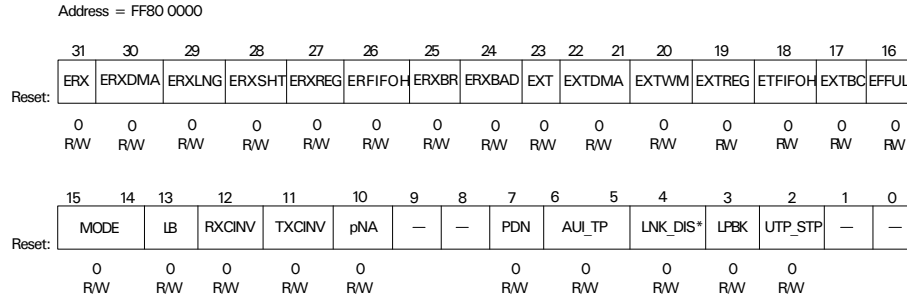
Ethernet General Control register and bit definitions

The following fields should be set only once, on device open:

- ERX
- ERXDMA
- ERXLNG
- ERXSHT
- ERXBAD
- ETX
- ETXDMA
- ETXWM
- EFULLD

The following fields can be ignored if using DMA mode (DMA interface logic) rather than interrupt service mode. DMA mode provides higher performance out of the Ethernet chip, and can be turned on and off.

- ERXREG
- ERFIFOH
- ERXBR
- ETXREG
- ETFIFOH
- ETXBC



Code (Bit #)	Definition of code	Description
ERX (D31)	Enable RX FIFO	0 – Disables inbound data flow and resets the FIFO 1 – Enables inbound data flow Must be set to 1 to allow data to be received from the MAC receiver. Can be used to reset the receive side FIFO.
ERXDMA (D30)	Enable receive DMA	0 – Disables inbound DMA data request (use to stall receiver) 1 – Enables inbound DMA data request Must be set to 1 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. Do not set this bit when operating the Ethernet receiver in interrupt service mode.
ERXLNG (D29)	Accept long (> 1518 bytes) receive packets	When set to 1, allows packets that are larger than 1518 bytes to be accepted by the MAC.
ERXSHT (D28)	Accept short (< 64 bytes) receive packets	When set to 1, allows packets that are smaller than 64 bytes to be accepted by the MAC. The ERXSHT bit is used primarily for debugging.
ERXREG (D27)	Enable Receive Data register ready interrupt	Must be set to 1 to generate an interrupt when data is available in the RX FIFO.
ERFIFOH (D26)	Enable receive data FIFO half full interrupt	Must be set to 1 to generate an interrupt when the RX FIFO is at least half full (1024 bytes).
ERXBR (D25)	Enable receive buffer ready interrupt	Must be set to 1 to generate an interrupt when a new data packet is available in the RX FIFO.
ERXBAD (D24)	Accept bad receive packets	When set to 1, 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. The ERXBAD bit is used primarily for debugging.

Table 68: Ethernet General Control register bit definition

Code (Bit #)	Definition of code	Description
ETX (D23)	Enable transmit FIFO	0 – Disables outbound data flow and resets the FIFO 1 – Enables outbound data flow Must be set to 1 to allow data to be written to the TX FIFO. Can be used to reset the transmit side FIFO.
ETXDMA (D22)	Enable transmit DMA	0 – Disables outbound DMA data request (use to stall transmitter) 1 – Enables outbound DMA data request Must be set to 1 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. Do *not* set this bit when operating the Ethernet receiver in interrupt service mode.
ETXWM (D21:20)	Transmit FIFO water mark before transmit start	00 – 25% FIFO full 01 – 50% FIFO full 10 – 75% FIFO full 11 – 100% FIFO full Identifies the minimum number of bytes required in the transmit FIFO before the Ethernet transmitter tries sending the packet. A larger watermark setting increases transmit packet latency, allowing for more slack in the memory system FIFO fill rate.
ETXREG (D19)	Enable Transmit Data Register ready interrupt	Must be set to 1 to generate an interrupt when the TX FIFO is ready to accept data.
ETFIFOH (D18)	Enable transmit data FIFO half empty interrupt	Must be set to 1 to generate an interrupt when the TX FIFO is at least half empty (< 64 bytes). Should be set only when operating the Ethernet transmitter in interrupt service mode instead of DMA mode. In general, DMA mode is preferred and ETFIFOH is not required to be set.
ETXBC (D17)	Enable transmit buffer complete interrupt	Must be set to 1 to generate an interrupt when the transmit packet transmission is complete.

Table 68: Ethernet General Control register bit definition

Code (Bit #)	Definition of code	Description
EFULLD (D16)	Enable full duplex operation	Must be set to 1 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.
MODE (D15:14)	Ethernet interface mode	00 – 10/100 Mbit MII mode 01 – Reserved 10 – 10 Mbit level 1 ENDEC mode 11 – 10 Mbit SEEQ ENDEC mode Identifies the type of Ethernet PHY attached to the NET+50 chip. The NET+50 chip supports two types of Ethernet PHY: the MII PHY and the ENDEC PHY. The NET+50 chip also supports two types of ENDEC interfaces: ■ 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 (D13)	MAC configured in internal loopback	Must be set to 1 when 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 (D12)	Invert the receive clock input	Should be set to 1 only when the external Ethernet PHY generates a clock that is inverted in phase relative to what the MAC is expecting. This bit is not required for most commercially available PHY devices.
TXCINV (D11)	Invert the transmit clock input	Should be set to 1 only when the external Ethernet PHY generates a clock that is inverted in phase relative to what the MAC is expecting. This bit is not required for most commercially available PHY devices.

Table 68: Ethernet General Control register bit definition

Code (Bit #)	Definition of code	Description
pNA (D10)	pSOS pNA buffer descriptors	0 – Standard receiver format. The data block immediately follows the 14-byte header block. 1 – pSOS pNA receiver format When set to 1, configures the EFE module to allow the receiver to provide receive data that 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 (D07:00)	ENDEC media control bits	The least significant 8 bits of the Ethernet General Control register, used only when the MODE field is configured for ENDEC mode. In this configuration, several NET+50 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. See "Media control (ENDEC mode only)" for more information.

Table 68: Ethernet General Control register bit definition

Media control (ENDEC mode only)

Table 69 shows the relationship between the lower bits in the Ethernet General Control register and the NET+50 chip pin. The NET+50 chip pins can be controlled by the ENDEC media control bits only when the MODE field is configured for ENDEC mode.

Data bit	Signal name	NET+50 chip pin	State inversion	Description
D7	PDN	TXD1	SEEQ: inverted TTL Level1: inverted open drain	Power down mode (D07): 0 - Normal 1 - External PHY in low power mode

Table 69: ENDEC mode media control bits

Data bit	Signal name	NET+50 chip pin	State inversion	Description
D6	AUI_TP[1]	TXD2	No inversion	AUI/TP selection (D06:D05): 00 - Auto select (with LNK_DIS*=1) 01 - AUI port 10 - TP port 11 - Reserved
D5	AUI_TP[0]	TXD3	No inversion	
D4	LNK_DIS*	TXER	No inversion	10BaseT link pulse disable (D04): 0 - Disable link pulse detection 1 - Enable link pulse detection
D3	LPBK	MDC	SEEQ: inverted Level1: inverted	Enable loopback mode (D03): 0 - Normal mode 1 - External loopback mode
D2	UTP_STP	MDIO	No inversion	Unshielded/shielded selection (D02): 0 - STP wiring 1 - UTP wiring

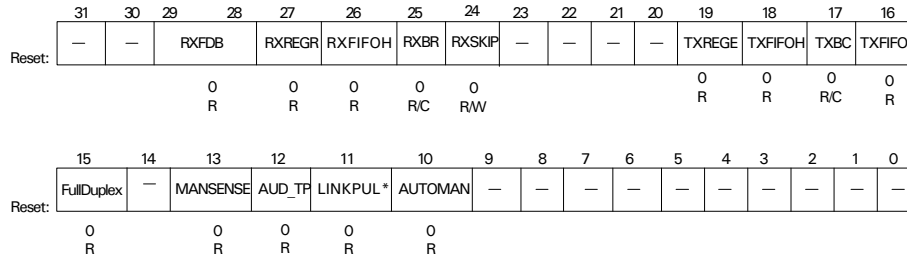
Table 69: ENDEC mode media control bits

Ethernet General Status register and bit definitions

The following fields can be ignored if using DMA mode (DMA interface logic) rather than interrupt service mode. DMA mode provides higher performance out of the Ethernet chip, and can be turned on and off.

- RXFDB
- RXREGR
- RXFIFOH
- RXBR
- RXSKIP
- TXREGE
- TXFIFOH
- TXBC

Address = FF80 0004



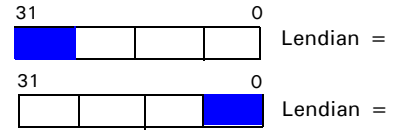
Code (Bit #)	Definition of code	Description
RXFDB (D29:28)	Receive FIFO data bytes available	00 – Full-word 01 – One byte 10 – Half-word 11 – Three bytes (LENDIAN determines which three)  Must be used in conjunction with RXREGR (D27). When RXREGR is set, RXFDB identifies how many bytes are available in the Receive Data register.
RXREGR (D27)	Receive Register ready	Is set to 1 whenever data is available to be read from the FIFO Data register. This bit, when set active high, can cause an interrupt to occur when the ERXREG bit is also set (in the Ethernet General Control register). The RXREGR bit is never active when RXBR (D25) is set. The RXBR bit must be cleared to activate RXREGR.
RXFIFOH (D26)	Receive FIFO half full	Is set to 1 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 also set (in the Ethernet General Control register).

Table 70: Ethernet General Status register bit definition

Code (Bit #)	Definition of code	Description
RXBR (D25)	Receive buffer ready	Is set to 1 whenever a new packet is available in the receive FIFO. This bit, when active high, can cause an interrupt to occur when the ERXBR bit is also set (in the Ethernet General COntrol register). The RXBR bit is set as an indicator to the interrupt service routine to read the Ethernet Receive Status register. After the Ethernet Receive Status register is read, the RXBR bit can be cleared by writing a 1 to the RXBR bit position in the Ethernet General Status register. After the RXBR bit is cleared, the RXREGR and RXFIFOH bits become active.
RXSKIP (D24)	Receive buffer skip	Can be set to 1 instead 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 bit provides a simple means of performing receive packet filtering in the interrupt service routine. When the Ethernet Receive Status register has been read and RXSKIP is set to 1, the interrupt service routine can flush the packet from the FIFO rather than reading it out of the FIFO.
TXREGE (D19)	Transmit Register empty	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 ETXREGE bit is also set (in the Ethernet General Control register). The TXREGE bit is never active while TXBC(D17) is set. The TXBC bit must be cleared to activate TXREGE.
TXFIFOH (D18)	Transmit FIFO half empty	Is set to 1 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 also set (in the Ethernet General Control register).
TXBC (D17)	Transmit buffer complete	Is set to 1 when packet transmission has completed. This bit, when active high, can cause an interrupt to occur when the ETXBC bit is also set (in the Ethernet General Control register). The TXBC bit is set as an indicator to the interrupt service routine to read the Ethernet Transmit Status register. After the Ethernet Transmit Status register is read, the TXBC bit can be cleared by writing a 1 to the TXBC bit position in the Ethernet General Status register. After the TXBC bit is cleared, the TXREGE and TXFIFOH bits become active.

Table 70: Ethernet General Status register bit definition

Code (Bit #)	Definition of code	Description
TXFIFOE (D16)	Transmit FIFO empty	Is set to 1 (active) when the transmit FIFO is empty.
D15:D0 (D15:00)	ENDEC media status bits	The least significant 16 bits of the Ethernet General Status register are used only when the MODE field is configured for ENDEC mode. In this configuration, several NET+50 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. See "Media status bits (ENDEC mode only)" for more information.

Table 70: Ethernet General Status register bit definition

Media status bits (ENDEC mode only)

Table 71 shows the relationship between the lower bits in the Ethernet General Status register and the NET+50 chip pin. The NET+50 chip pins can be monitored by the ENDEC media control bits only when the MODE field is configured for ENDEC mode.

Data bit	Signal name	NET+50 chip pin	Description
D15	FullDuplex	RXD2	Full duplex indicator (D15): 0 - Full duplex mode 1 - Half duplex mode
D13	Mansense	RXD1	Jumper sense (D13): 0 - Jumper in 1 - Jumper out
D12	AUI_TP	RXD3	10 Mbit reverse polarity indicator (D12): 0 - TP port 1 - AUI port
D11	LINKPUL*	RXER	10 Mbit link pulse indicator (D11): 0 - Disable link pulse detection 1 - Enable link pulse detection

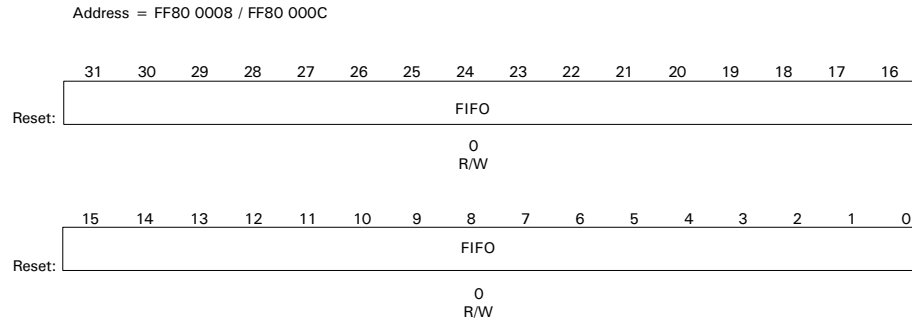
Table 71: ENDEC mode media status bits

Data bit	Signal name	NET+50 chip pin	Description
D10	Automan	RXDV	AUTOMAN jumper sense (D10): 0 - AUTOMAN jumper in 1 - AUTOMAN jumper out

Table 71: ENDEC mode media status bits

Ethernet FIFO Data register

The Ethernet FIFO Data register is used to interface manually with the Ethernet FIFO instead of using DMA support. This register can be used as a diagnostic tool.



Writing to the Ethernet FIFO Data register

Writing to the Ethernet FIFO Data register loads the transmit FIFO. This register can be written to only when the TXREGE bit (D19) is set in the Ethernet General Status register, indicating that 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; this is used for polled or interrupt-driven access to the Ethernet transmitter, and kicks off the transmission of a short packet onto the wire. The first and middle words of the frame must use the primary address (FF80 0008).

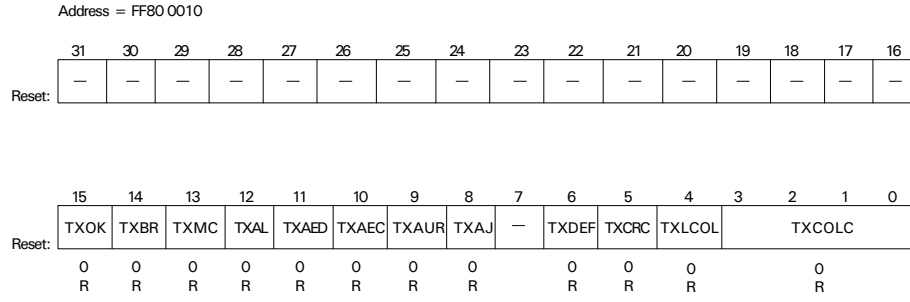
Reading from the Ethernet FIFO Data register

Reading from the Ethernet FIFO Data register empties the receive FIFO. The Ethernet General Status register identifies how many bytes are available to be read. The receive FIFO is available only after the RXBR (D25) bit has been cleared in the Ethernet General Status register. The Ethernet Receive Status register (see page 220) should be read before clearing the RXBR bit.

Note: When operating in DMA mode, the Receive Status register is read automatically and the RXBR bit is cleared automatically.

Ethernet Transmit Status register and bit definitions

The Ethernet Transmit Status register contains the status for the last completed transmit buffer. The transmit buffer complete bit (TXBC, D17) is set in the Ethernet General Status register when a transmit frame is completed and the Ethernet Transmit Status register is loaded. The lower 16 bits of the Ethernet Transmit Status register are also loaded into the StatusOrIndex field of the DMA buffer descriptor when using DMA mode.



Code (Bit #)	Definition of code	Description
TXOK (D15)	Transmit packet OK	■ Is set to 1 when the last Ethernet packet was transmitted without error. ■ Is set to 0 when the last Ethernet packet was not transmitted.

Table 72: Ethernet Transmit Status register bit definition

Code (Bit #)	Definition of code	Description
TXBR (D14)	Transmit broadcast packet	Is set to 1 to indicate that the last Ethernet packet transmitted was a broadcast packet.
TXMC (D13)	Transmit multicast packet	Is set to 1 to indicate that the last Ethernet packet transmitted was a multicast packet.
TXAL (D12)	Transmit abort late collision	Is set to 1 to indicate that the last Ethernet packet was not transmitted successfully — packet transmission was aborted due to a late collision problem.
TXAED (D11)	Transmit abort excessive deferral	Is set to 1 to indicate that the last Ethernet packet was not transmitted successfully — packet transmission was aborted due to *excessive deferral*. Excessive deferral means that there was a receive carrier on the Ethernet channel for a long period of time, making it impossible for the transmitter to try to transmit.
TXAEC (D10)	Transmit abort excessive collisions	Is set to 1 to indicate that the last Ethernet packet was not transmitted successfully — packet transmission was aborted due to an excessive number of collisions. The maximum number of collisions allowed is determined by the RETRY field in the Collision Window/Collision Retry register (see page 238).
TXAUR (D09)	Transmit abort underrun	Is set to 1 to indicate that the last Ethernet packet was not transmitted successfully — packet transmission was aborted due to a FIFO *underrun* condition. A FIFO underrun condition indicates that the DMA controller was unable to fill the FIFO at a fast enough rate compared to the rate of transmission on the Ethernet medium, for one of these reasons: ■ The DMA controller was not configured for bursting. ■ The memory peripheral device was not configured for bursting. ■ The memory peripheral device is too slow to support the Ethernet interface.

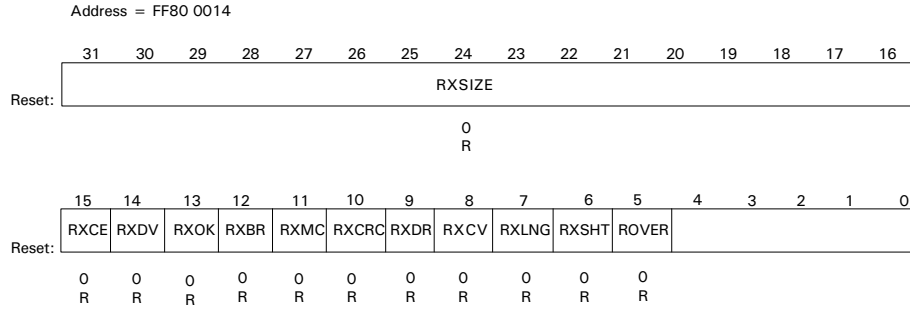
Table 72: Ethernet Transmit Status register bit definition

Code (Bit #)	Definition of code	Description
TXAJ (D08)	Transmit abort jumbo	Is set to 1 to indicate that the last Ethernet packet was not transmitted successfully — packet transmission was aborted due to a *jumbo* condition. The jumbo condition means the packet was too large; that is, greater than 1518 bytes. Packets larger than 1518 bytes are not transmitted successfully unless the HUGEN bit is set in the MAC Configuration register (see page 225)).
TXDEF (D06)	Transmit packet deferred	Is set to 1 to indicate that the last successfully-transmitted Ethernet packet encountered a deferral. Transmission was delayed because the Ethernet medium was busy at the time of first transmission attempt.
TXCRC (D05)	Transmit packet CRC error	Is set to 1 to indicate that the last successfully-transmitted Ethernet packet had an embedded CRC error. This condition occurs only when the CRCEN bit in the MAC Configuration register is set to 0 (see page 224). When CRCEN is set to 0, the MAC does not insert a CRC; the MAC expects a precompiled CRC to be contained in the last four bytes of the Ethernet packet. If the MAC finds that the precompiled CRC is incorrect, the MAC sets the TXCRC bit in the Ethernet Transmit Status register.
TXLCOL (D04)	Transmit packet late collision	Is set to 1 to indicate that the MAC encountered a late collision while trying to transmit the Ethernet packet. TXLCOL indicates only that a late collision event has occurred; if the packet transmission was aborted, the TXAL bit (D12) will be set. The MAC tries to retransmit the packet when the RTRYL bit in the MAC Configuration register is set to 1 (see page 224).
TXCOLC (D03:00)	Transmit packet collision count	Indicates how many collisions the MAC encountered while it was trying to transmit the packet. TXCOLC indicates only that collision events have occurred; if the packet transmission was aborted, the TXAEC (D10) bit will be set. The MAC tries to retransmit the packet up to the maximum number of collision retries defined by the RETRY field in the Collision Window/Collision Retry register (see page 238).

Table 72: Ethernet Transmit Status register bit definition

Ethernet Receive Status register and bit definitions

The Ethernet Receive Status register contains the status for the last completed receive buffer. The receive buffer complete bit (RXBR, D25) is set in the Ethernet General Status register when a receive frame is completed and the Ethernet Receive Status register is loaded. The lower 16 bits of the Ethernet Receive Status register are also loaded into the StatusOrIndex field of the DMA buffer descriptor (when using DMA mode).



Code (Bit #)	Definition of code	Description
RXSIZE (D31:16)	Receive packet size	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 (D15)	Receive carrier event	Is set to 1 to indicate that 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 particular packet. A carrier event is defined as any activity on the receive channel that does not result in a packet receive attempt being made.

Table 73: Ethernet Receive Status register bit definition

Code (Bit #)	Definition of code	Description
RXDV (D14)	Receive data event	Is set to 1 to indicate that, at some point since the last recorded receive packet event, a *receive data event* was detected, noted, and reported with this receive packet event. The receive data event is not associated with this particular packet. A receive data event is an event that occurs without a valid preamble and start frame delimiter (SFD) in the data stream.
RXOK (D13)	Receive packet OK	Is set to 1 to indicate that the next packet in the receive FIFO has been received without any errors.
RXBR (D12)	Receive broadcast packet	Is set to 1 to indicate that the next packet in the receive FIFO is a broadcast packet.
RXMC (D11)	Receive multicast packet	Is set to 1 to indicate that the next packet in the receive FIFO is a multicast packet.
RXCRC (D10)	Receive CRC error	Is set to 1 to indicate that the next packet in the receive FIFO was received with a CRC error.
RXDR (D09)	Receive dribble nibble	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 (dribble nibble) were received. This packet is still considered valid as long as the RXCRC bit (D10) is not set.
RXCV (D08)	Receive code violation	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 (D07)	Receive long packet	Is set to 1 to indicate that the next packet in the receive FIFO is longer than 1518 bits. A long packet is accepted only when the ERXLNG bit (D29) is set in the Ethernet General Control register.
RXSHT (D06)	Receive short packet	Is set to 1 to indicate that the next packet in the receive FIFO is smaller than 64 bytes. A short packet is accepted only when the ERXSHT bit (D28) is set in the Ethernet General Control Register.

Table 73: Ethernet Receive Status register bit definition

Code (Bit #)	Definition of code	Description
ROVER (D05)	Receive overrun	Is set to 1 to indicate that a receive FIFO *overrun* condition has occurred. An overrun condition occurs when the FIFO becomes full while receiving an Ethernet packet. The FIFO overrun condition indicates that the DMA controller was not able to empty the FIFO at a fast enough rate compared to the rate that information was received from the Ethernet medium for one of these reasons: ■ The DMA controller was not configured for bursting. ■ The memory peripheral device was not configured for bursting. ■ The memory peripheral device is too slow to support the Ethernet interface. ■ All of the buffers in the Ethernet receive buffer were exhausted. This can happen if the application software is unable to keep up with incoming Ethernet traffic. One solution is to add more buffers. See "Ethernet receiver considerations" on page 193.

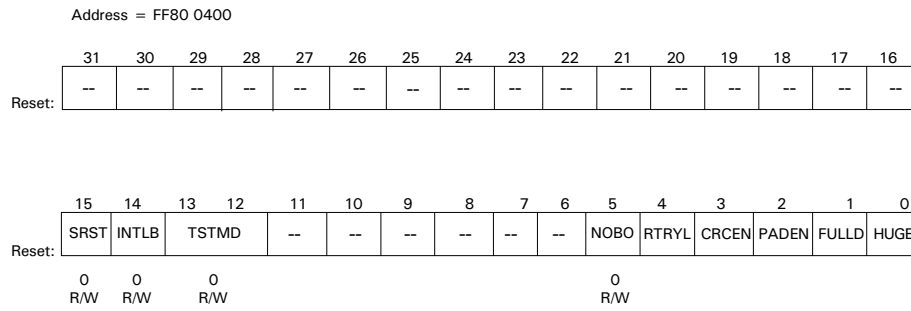
Table 73: Ethernet Receive Status register bit definition

MAC registers

The Ethernet has two MAC registers:

- **MAC Configuration register.** A 32-bit register that contains MAC configuration options. All bits initialize to 0 on reset.
- **MAC Test register.** A 32-bit register used only during hardware simulation — never under normal operation conditions. All bits initialize to 0 on reset.

MAC Configuration register and bit definitions



Code (Bit #)	Definition of code	Description
SRST (D15)	MAC soft reset	Is set to 1 to soft-reset the MAC module. **Note:** The SRST bit has been known to be unreliable. Because the standard NET+50 hardware and software resets all reset the MAC, a soft reset of the MAC is not required.
INTLB (D14)	Internal loopback	Places the MAC module in internal loopback mode. This configuration is useful for diagnostic testing of the MAC module. **Note:** When the INTLB is set to 1, the LB bit (D13) in the Ethernet General Control Register must also be set to 1.

Table 74: MAC Configuration register bit definition

Code (Bit #)	Definition of code	Description
TSTMD (D13:12)	Transmit test mode	00 – Normal operation 01 – Reserved 10 – Transmitter test mode 11 – Receive test mode Used for simulation of the MAC module. This field must be set to 00 for normal functional operation of the MAC module.
NOBO (D05)	No backoff	0 – Backoff enabled 1 – Backoff disabled Can be set to 1 to disable the *backoff* feature. The backoff feature is used when a collision is detected, and causes the MAC to wait a random amount of time before trying retransmission. When NOBO is set, the MAC immediately tries to retransmit after the jamming period is complete.
RTRYL (D04)	Retry late collision	Can be set to 1 to allow the MAC to try a retransmission when a late collision is detected. When RTRYL is set to 0 and a late collision is detected, the transmission is aborted through the TXAL bit setting in the Ethernet Transmit Status register (see page 218).
CRCEN (D03)	Append CRC	Can be set to 1 to allow 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 verifies that the precompiled 4-byte CRC value is correct. If the MAC finds an incorrect CRC value, the MAC reports the condition by setting the TXCRC bit in the Ethernet Transmit Status register (see page 219).
PADEN (D02)	Auto pad enable	Can be set to 1 to force the MAC to automatically pad fill all Ethernet packets that are smaller than 64 bytes. The MAC automatically inserts zero-byte padding at the end of the Ethernet data frame to make the Ethernet packet the minimum required 64 bytes in size.

Table 74: MAC Configuration register bit definition

Code (Bit #)	Definition of code	Description
FULLD (D01)	Full duplex enable	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 ignores any and all collision indications from the TXCOL input pin.
HUGEN (D00)	Enable huge frames	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 transmits Ethernet packets of any size. When HUGEN is set to 0, the MAC aborts transmission of all packets larger than 1518 bytes and sets the TXAJ bit in the Ethernet Transmit Status register (see page 219) to indicate this condition.

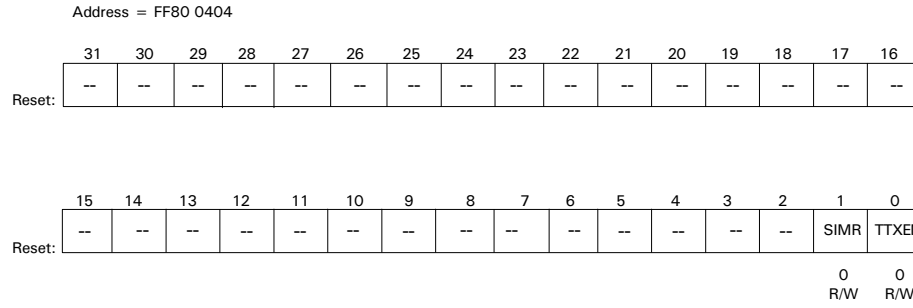
Table 74: MAC Configuration register bit definition

MAC Test register and bit definitions

There are two fields in the MAC Test register:

- **SIMR (**D01**).** Simulation reset.
- **TTXEN (**D00**).** Test transmit enable.

Both fields, when set to 1, are used only forsimulation of the MAC module. Both fields must be set to 0 for normal functional operation of the MAC module.

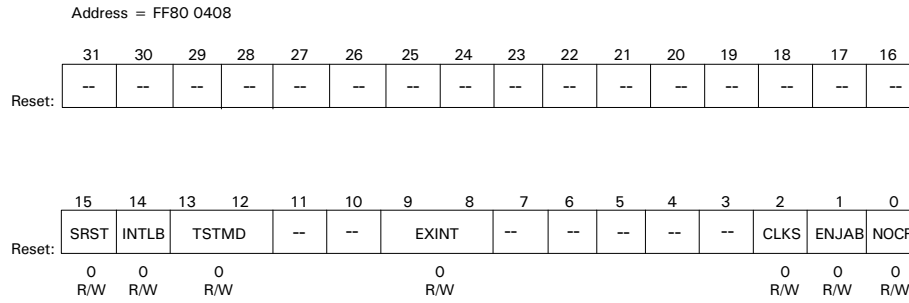


PCS registers

The Ethernet module uses two Physical Coding Sublayer (PCS) registers:

- **PCS Configuration register.** A 32-bit register that contains PCS configuration options. All bits initialize to 0 on reset.
- **PCS Test register.** A 32-bit register that is used only during hardware simulation — never under normal operating conditions. All bits initialize to 0 on reset.

PCS Configuration register and bit definitions



Code (Bit #)	Definition of code	Description
SRST (D15)	Soft reset	Is set to 1 to soft-reset the PCS module. The SRST bit has been known to be unreliable. The standard NET+50 hardware and software resets all reset the PCS; as such, a soft reset of the PCS is not required.
INTLB (D14)	Internal loopback	Is used to place the PCS module in internal loopback mode. This configuration is useful for diagnostic testing of the PCS module. When the INTLB is set to 1, the LB bit (D13) in the Ethernet General Control register must also be set to 1.

Table 75: PCS Configuration register bit definition

Code (Bit #)	Definition of code	Description
TSTMD (D13:12)	Test mode	00 – Normal operation 01 – Reserved 10 – PE100X test mode 11 – PE10T test mode Used for simulation of the PCS module. This field must be set to 00 for normal functional operation of the PCS module.
EXINT (D09:08)	External interface mode	00 – MII normal operation 01 – TP–PMD mode 10 – 10 Mbit ENDEC mode 11 – Reserved Used for all modern MII style 10/100 PHY devices. TP–PMD mode is used for older PHY that contain their own non-standard physical coding sublayer (PCS) circuitry. ENDEC mode configuration is used for the older 10 Mbit-only PHY devices that predate the MII interface standard.
CLKS (D02)	Clock select	0 – 25 MHz 1 – 44 MHz Used by the MAC to select the divide-down-ratio for the SYS_CLK system clock to produce the management data clock (MDC). Setting the bit divides SYS_CLK by 14; clearing the bit divides SYS_CLK by 10. MDC cannot exceed the specified frequency for the chosen PHY. The timing of the MDIO pin versus the clock at the resulting frequency should be verified. See "System clock generation" on page 97 for system clock frequency information.
ENJAB (D01)	Enable jabber protection	When set to 1, enables the *jabber* protection logic within the PCS when the PCS is operating in ENDEC mode (see "EXINT (D09:08)"). Jabber is the condition in which a transmitter is stuck *on* for longer than 50mS, preventing other stations from transmitting.
NOCFR (D00)	Disable ciphering	When set to 1, disables the ciphering login within the PCS when the PCS is operating in TP–PMD mode (see "EXINT (D09:08)"). The ciphering operation converts the 100Base-X service data unit (SDU) into the protocol data unit (PDU).

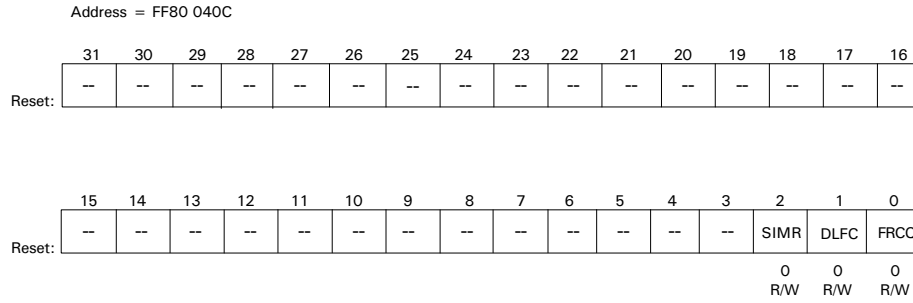
Table 75: PCS Configuration register bit definition

PCS Test register

There are three fields in the PCS Test register:

- **SIMR (D02).** Simulation reset.
- **DLFC (D01).** Disable link fail counter.
- **FRCQ (D00).** Force quiet.

All fields, when set to 1, are used only for simulation of the PCS module. All fields must be set to 0 for normal functional operation of the PCS module.

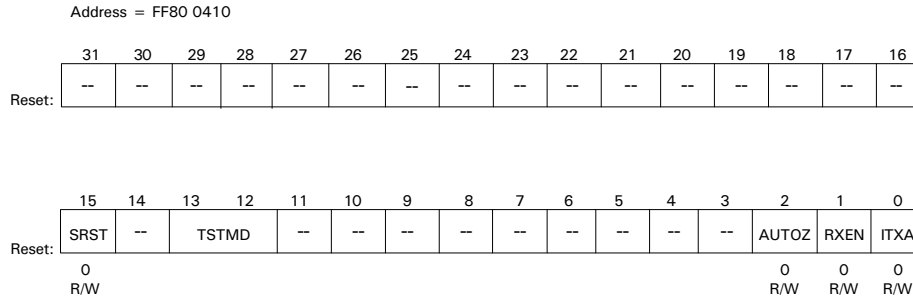


STL registers

The Ethernet module uses two Station Logic (STL) registers:

- **STL Configuration register.** 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.
- **STL Test register.** A 32-bit register that is used only during hardware simulation — never under normal operation conditions. All bits initialize to 0 on reset.

STL Configuration register and bit definitions



Code (Bit #)	Definition of code	Description
SRST (D15)	Soft reset	Is set to 1 to soft-reset the STL module. **Note:** The SRST bit has been known to be unreliable. Because the standard NET+50 hardware and software resets all reset the STL, a soft reset of the STL is not required.
TSTMD (D13:12	Test mode	00 – Normal operation 01 – Reserved 10 – Statistics module test mode 11 – Station address logic test mode Used for simulation of the STL module. This field must be set to 00 for normal functional operation of the STL module.
AUTOZ (D02)	Auto zero statistics	0 – Do not automatically reset Statistics registers to zero. 1 – Automatically reset Statistics registers to zero after reading. Can be set to 1 to cause the Statistics registers (see "Statistics monitoring" on page 251) to be automatically reset to 0 after reading. When set to 0, the Statistics registers must be manually reset to 0 by writing a 0 value to the register.

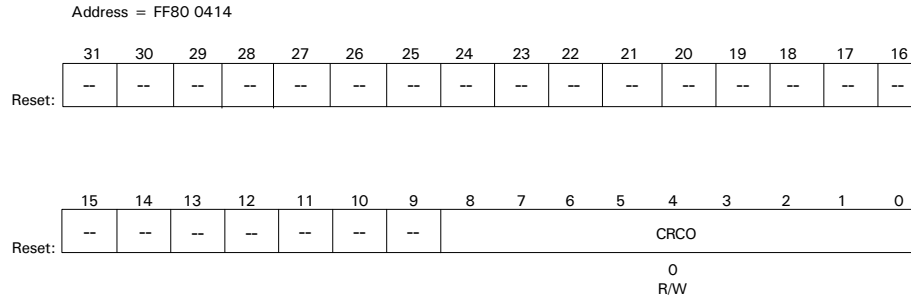
Table 76: STL Configuration register bit definition

Code (Bit #)	Definition of code	Description
RXEN (D01)	Receiver enable	Must be set to 1 to enable the Ethernet receiver logic. The RXEN bit should be set to 1 after the fields in the Station Address Filter register (see "Transmit Random Number Generator (RNG)" on page 240) have been configured. The Ethernet receiver can be disabled at any time by setting the RXEN bit to 0.
ITXA (D00)	Insert transmit source address	Can be set to 1 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 (see "Transmit Masked Random Number (MRNG)" on page 240). When the ITXA bit is set to 0, the 7^{th} through 12^{th} data bytes in the Ethernet packet are ignored and are replaced by the size bytes found in the Station Address register (see "Station Address register and bit definitions" on page 256).

Table 76: STL Configuration register bit definition

STL Test register and bit definitions

There is one field in the STL Test register — **CRCO:** CRC output (D08:00). When set to 1, this field is used only for simulation of the STL module. This field must be set to 0 for normal functional operation of the module.



Transmit Control registers

There are 10 Transmit Control registers. Each register is 32-bits, and all bits are initialized to 0 on reset, unless otherwise noted.

Inter-Packet Gap (IPG) registers

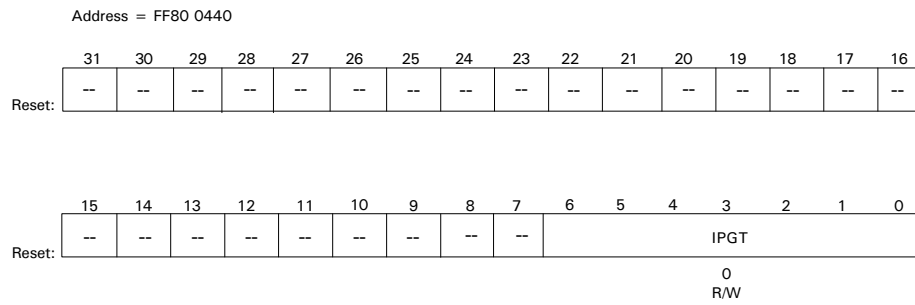
The inter-packet gap (IPG) is the measurement between the last nibble (or *bit* in ENDEC mode) of frame check sequence and the first nibble (or *bit* in ENDEC mode) of the preamble for the next packet.

- For 100 Mbyte Ethernet, the IEEE 802.3u standard requires IPG to be a minimum of 0.96 microseconds.
- For 10 Mbyte Ethernet, the IEEE 802.3u 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 finetune the IPG:

- **IPGT.** The IPGT field is in the Back-to-Back IPG Gap Timer register.
- **IPGR1** and **IPGR2**. These two fields are in the Non Back-to-Back Gap Timer register.

Back-to-Back IPG Gap Timer register



The *IPGT* (D06:00), or back-to-back transmit IPG, spaces back-to-back transmit packets. The Ethernet transmit state machine has an intrinsic delay of four

Ethernet clocks between packets. With the IPGT field set to 0, the IPG will be a minimum of four 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 in MII mode is 0x14.

Operating in MII mode

The information and examples below pertain to operating in MII mode.

MII mode transmit clock information

Clock frequency	Clock period	Maximum bit rate
25 MHz	40 nsec	100 Mbit
2.5 MHz	400 nsec	10 Mbit

Examples of IPGT configuration for 100BT operating in MII mode

0x14, or 0.96uS, is recommended.

100 Mbit 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

Examples of IPGT configuration for 10BT operating in MII mode

0x14, or 9.6uS is recommended.

10 Mbit IPGT	802.3 IPG
0x00	1.6 uS
0x03	2.8 uS
0x07	4.4 uS

10 Mbit IPGT	802.3 IPG
0x0B	6.0 uS
0x0F	7.6 uS
0x14	9.6 uS
0x1F	14.0 uS

Operating in ENDEC mode

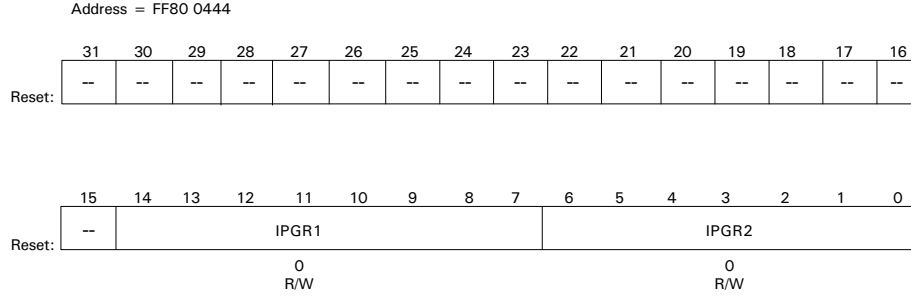
When operating in ENDEC mode, the IPGT values are different from MII mode. The ENDEC transmit clock is 10MHz rather than 25MHz, but there is still a 4-clock intrinsic delay in the transmitter state machine.

Examples of IPGT configuration for 10BT operating in ENDEC mode

`0x5C` or `9.6uS` is recommended.

10 Mbit IPGT	802.3 IPG
0x07	1.1 uS
0x0F	1.9 uS
0x3F	6.7 uS
0x5C	9.6 uS
0x5F	9.9 uS

Non Back-to-Back IPG Gap Timer register



The IPGR1 and IPGR2 timers begin their interval timing at the same time.

IPGR1 (D14:08), or non back-to-back IPG length part I, is the interval in which the transmitter defers if carrier sense goes high. The interval is usually set to 2/3 of the IPGR2 value. During the period of time after IPGR1 times out but before IPGR2 times out, the transmitter does not defer if it sees carrier sense. The transmitter will cause a collision if it has a packet to send, resulting in activation of the random back off fairness algorithm.

IPGR2 (D06:00), or non back-to-back IPG length part II, spaces 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 three 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 is a minimum of seven Ethernet clocks, or 0.28 uS, when operating at 100BT.

Operating in MII mode — IPGR1

The following examples pertain to operating in MII mode.

Examples of IPGR1 configuration for 100BT operating in MII mode

0x09, or 0.64uS, is recommended.

100Mbit 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

Examples of IPGR1 configuration for 10BT operating in MII mode

`0x09`, or `06.4uS`, is recommended.

10Mbit 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

Operating in ENDEC mode — IPGR1

When operating in ENDEC mode, the IPGR1 values are different from MII mode. The ENDEC transmit clock is 10MHz rather than 25MHz, but there is still a 7-clock intrinsic delay in the transmitter state machine.

Examples of IPGR1 configuration for 10BT operating in ENDEC mode

`0x39`, or `6.4uS`, is recommended.

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

Operating in MII mode — IPGR2

The IEEE 802.3u standard requires an IPG value of `9.6 uS`. The recommended value for IPGR2 when operating in MII mode is `0x11`.

Examples of IPGR2 configuration for 100BT operating in MII mode

`0x11`, or `0.96uS`, is recommended.

100Mbit 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

Examples of IPGR2 configuration for 10BT operating in MII Mode

`0x11`, or `9.6uS`, is recommended.

10 Mbit 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

Operating in ENDEC mode — IPGR2

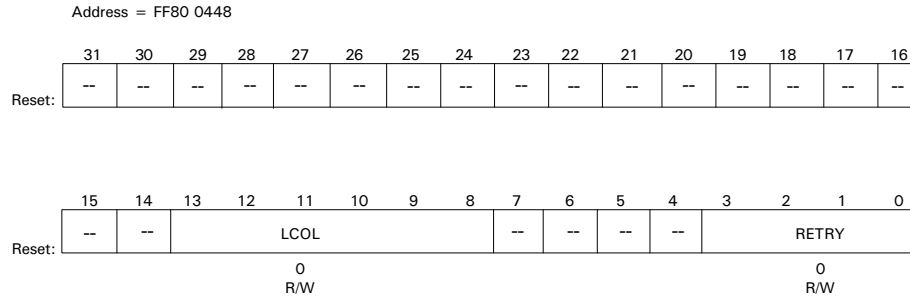
When operating in ENDEC mode, the IPGR2 values are different from MII mode. The ENDEC transmit clock is 10MHz rather than 25MHz, but there is still a 7-clock Ethernet clock intrinsic delay in the transmitter state machine.

Examples of IPGR2 configuration for 10BT operating in ENDEC mode

`0x59`, or `9.6uS`, is recommended.

10 Mbit IPGR2	802.3 IPG
0x07	1.4 uS
0x0F	2.2 uS
0x3F	6.9 uS
0x59	9.6 uS
0x5F	10.2 uS

Collision Window/Collision Retry register



Code (Bit #)	Definition of code	Description
LCOL (D13:08)	Late collision window	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 start frame delimiter (SFD) that allow a normal collision to occur. Any collisions after the time defined by LCOL 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 (D03:00)	Maximum number of retries after a collision	Specifies the number of retransmission attempts allowed following a collision, before aborting the packet due to excessive collisions. The 802.3u standard specifies the attemptLimit to be 0xF (15d) and is the recommended value for this field.

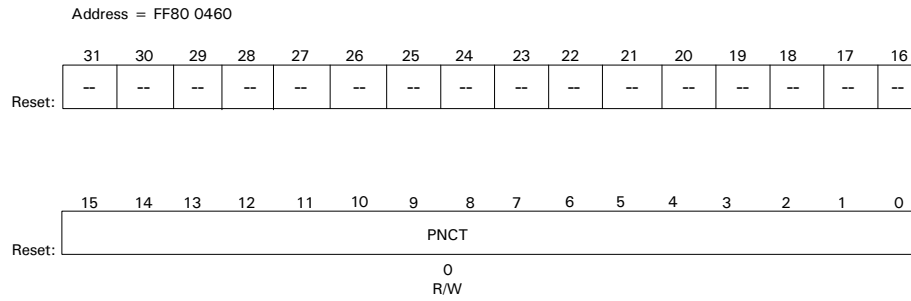
Table 77: Collision Window/Collision Retry register bit definition

"Simulation" registers

The remaining Transmit Control registers are used only for transmitter module simulation. For normal functional operation of the MAC transmitter, leave the bit settings within the registers as set.

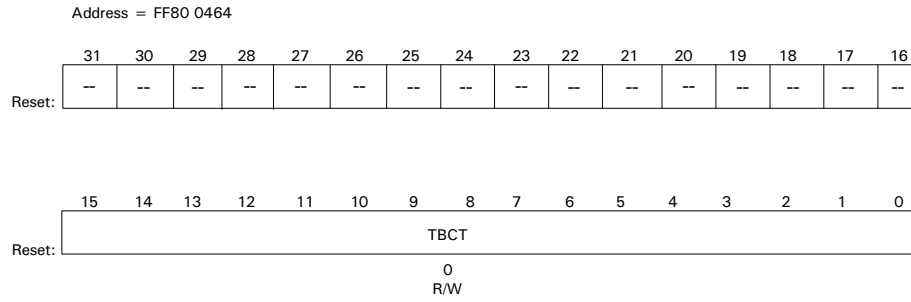
Transmit Packet Nibble Counter (PCNT)

D15:00



Transmit Byte Counter register (TBCT)

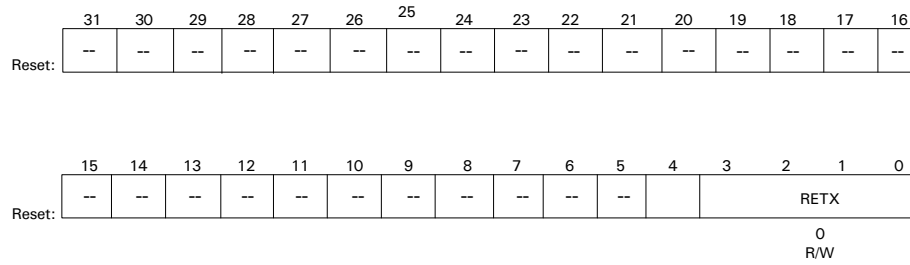
D15:00



Retransmit Byte Counter register (RETX)

(D03:00)

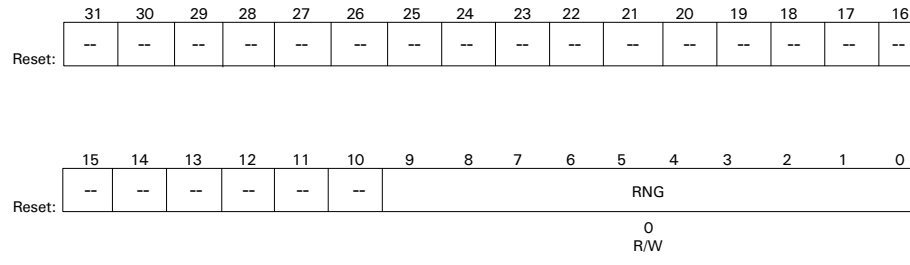
Address = FF80 0468



Transmit Random Number Generator (RNG)

(D09:00)

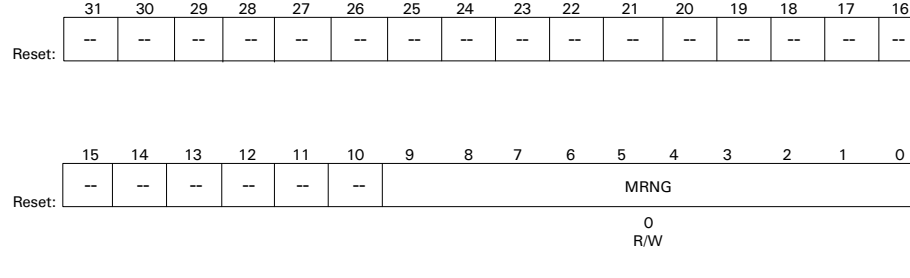
Address = FF80 046C



Transmit Masked Random Number (MRNG)

(D09:00)

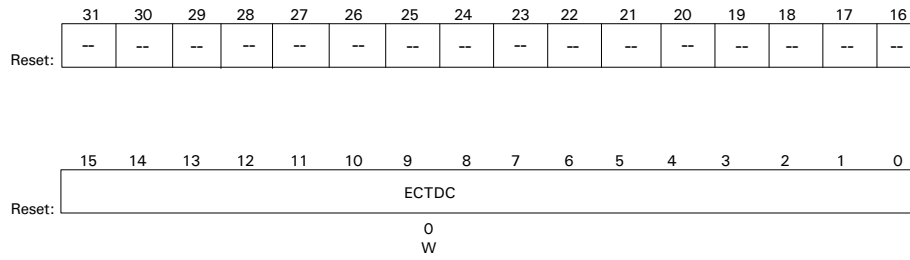
Address = FF80 0470



Transmit Counter Decodes (ECTDC)

(D15:00)

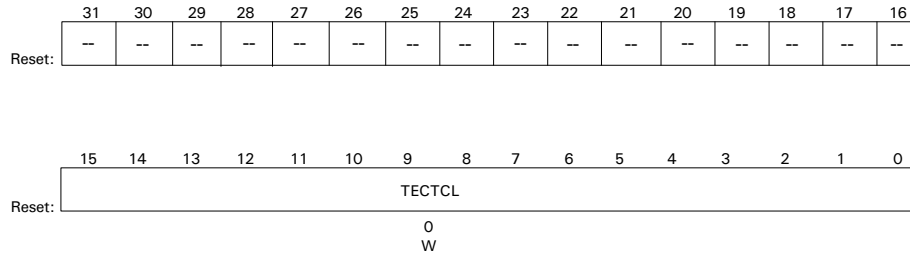
Address = FF80 0474



Test Operate Transmit Counters (TECTCL)

(D15:00)

Address = FF80 0478



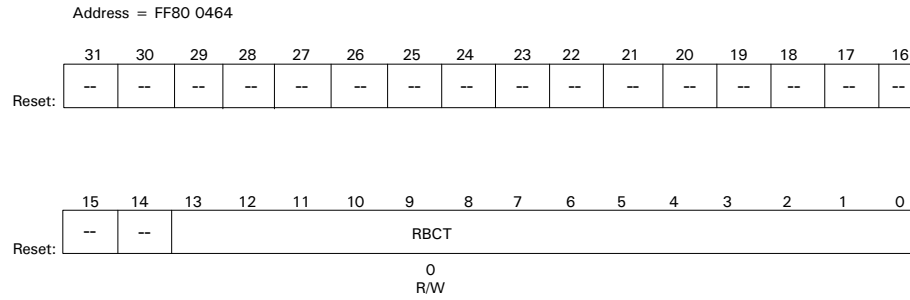
Receive Control registers

There are three Receive Control registers. Each register is a 32-bit register, and all bits are initialized to 0 on reset.

The Receive Control registers are used only for simulation of the MAC receiver module. For normal functional operation of the module, leave the bit settings within the registers as set.

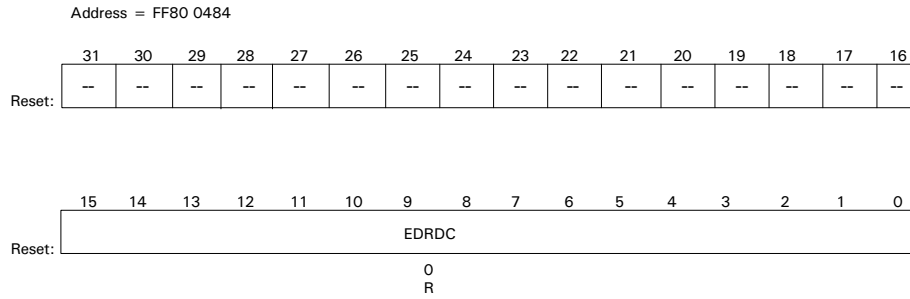
Receive Byte Counter (RBCT)

(D13:D00)



Receive Counter Decodes (ECRDC)

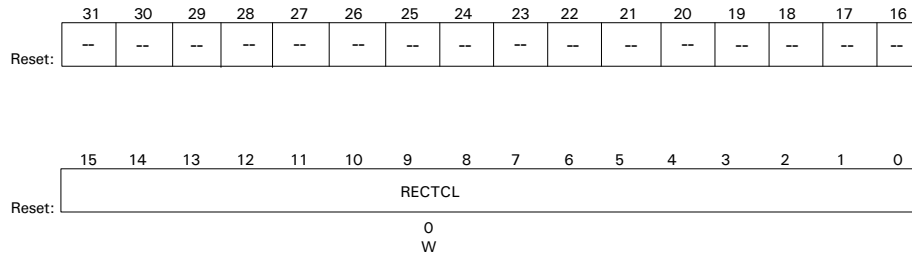
(D15:00)



Test Operate Receive Counters (RECTCL)

(D15:00)

Address = FF80 0488



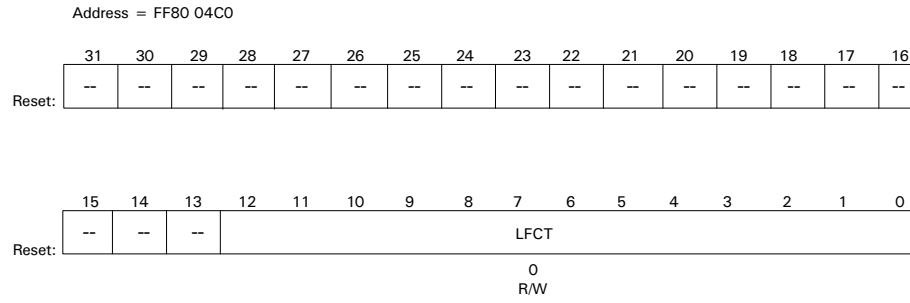
PCS Control registers

There are three Physical Sublayer Control (PCS) registers. Each register is a 32-bit register, and all bits are initialized to 0 on reset.

The PCS Control registers are used only for simulation of the PCS module. For normal functional operation of the PCS module, leave the bit settings within the registers as set.

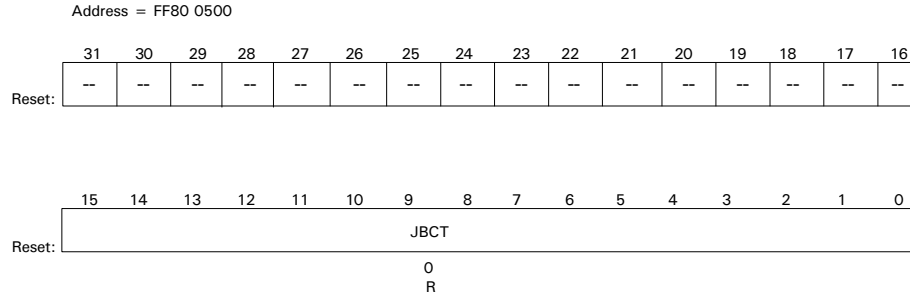
Link Fail Counter (LFCT)

(D12:00)



10 MB Jabber Counter (JBCT)

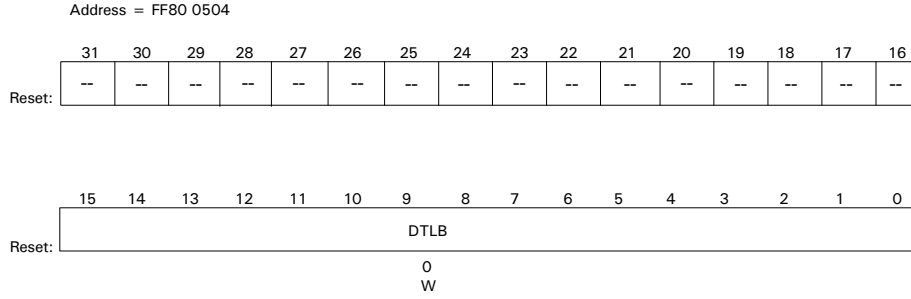
(D15:00)



10 MB Loss of Carrier Counter (DTLB)

(D15:00)

Address = FF80 0504



Ethernet MII interface signals

Table 78 defines and briefly describes the signals included in the MII interface. See Chapter 3, "NET+50 Chip Package," for more information

Code	Definition	Description
MDC	Management data clock	Provides the clock for the MDIO serial data channel. The MDC signal is a NET + 50 chip output. The maximum frequency is 2.5 MHz.
MDIO	Management data IO	A bi-directional signal that provides a serial data channel between the NET + 50 chip and the external Ethernet PHY module.
TXCLK	Transmit clock	An input to the NET + 50 chip from the external PHY module. TXCLK provides the synchronous data clock for transmit data. The transmit clock operates as follows: At 25 MHz for 100 Mbit operation At 2.5 MHz for 10 Mbit operation At 10 MHz in ENDEC mode
TXD3 TXD2 TXD1 TXD0	Transmit data signals	Nibble bus used by the NET + 50 chip to drive data to the external Ethernet PHY. All transmit data signals are synchronized to TXCLK. In ENDEC mode, only TXD0 is used for transmit data.
TXER	Transmit coding error	Output asserted by the NET + 50 when an error has occurred in the transmit data stream. When operating at 100 Mbps, the PHY responds to this signal by sending "`invalid code symbols`" on the line.
TXEN	Transmit enable	Asserted when the chip drives valid data on the TXD outputs. This signal is synchronized to TXCLK.
COL	Transmit collision	Input signal asserted by the external Ethernet PHY when a collision is detected. This input must remain high for the duration of the collision.
CRS	Receive carrier sense	Asserted by the external Ethernet PHY when the receive medium is non-idle. During half-duplex operation, this can also occur when the transmitter is active.

Table 78: MII interface signals

Code	Definition	Description
RXCLK	Receive clock	An input to the NET+50 chip from the external PHY module. The receive clock provides the synchronous data clock for receive data, and operates as follows: At 25 MHz for 100 Mbit operation At 2.5 MHz for 10 Mbit operation At 10 MHz in ENDEC mode The receive clock must remain active a minimum of 27 bit times after the end of each received Ethernet frame.
RXD3 RXD2 RXD1 RXD0	Receive data signals	Nibble bus used by the NET+50 chip to input receive data from the external Ethernet PHY. All receive data signals are synchronized to RXCLK. In ENDEC mode, only RXD0 is used for receive data.
RXER	Receive error	Input asserted by the external Ethernet PHY when the Ethernet PHY encounters invalid symbols from the network. This signal must be synchronous to RXCLK.
RXDV	Receive data valid	Input asserted by the external Ethernet PHY when the Ethernet PHY drives valid data on the RXD inputs. This signal must be synchronous to RXCLK.

Table 78: MII interface signals

MII Control registers

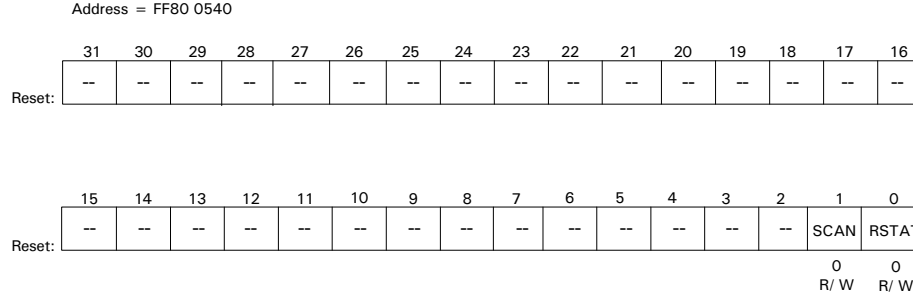
There are five MII Control registers:

- MII Command register
- MII Address register
- MII Write register
- MII Read register
- MII Indicators register

Each register is a 32-bit register. All bits are initialized to 0 on reset unless otherwise noted.

MII Command register and bit definitions

The default value for both fields in this register is 0.

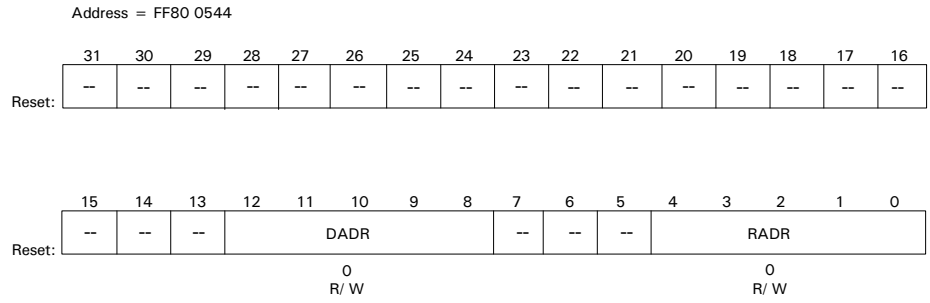


Code (Bit #)	Definition of code	Description
SCAN (D01)	Automatically scan for read data	When set to 1, the MII management module performs read cycles continuously. This is useful for monitoring link fail, for example.
RSTAT (D00)	Single scan for read data	When set to 1, the MII management module performs 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.

Table 79: MII Command register bit definition

MII Address register and bit definitions

The default value for both fields in this register is 0.

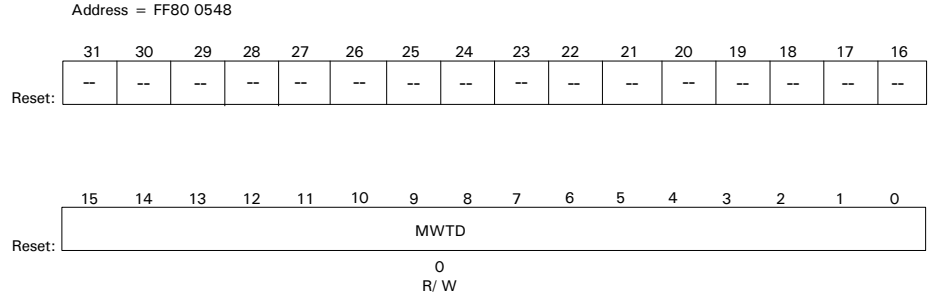


Code (Bit #)	Definition of code	Description
DADR (D12:08)	MII PHY device address	Represents the 5-bit PHY device address field for management cycles. Up to 31 different PHY devices can be addressed; address 0 is reserved.
RADR (D04:00)	MII PHY register address	This field represents the 5-bit PHY register address field for management cycles. Up to 32 registers within a single PHY device can be addressed.

Table 80: MII Address register bit definition

MII Write Data register and bit definition

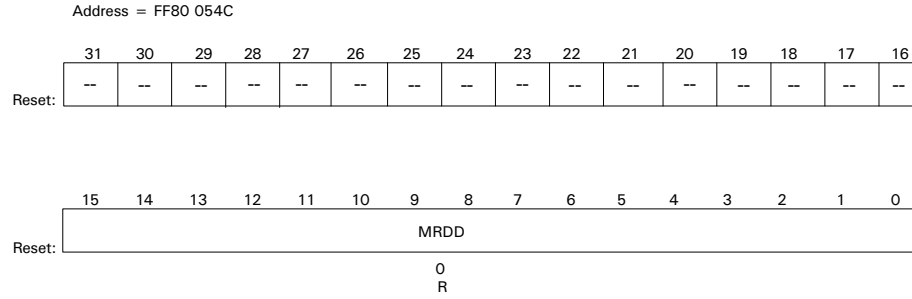
There is one field in the MII Write Data register — **MWTD:** MII write data (D15:00).



When this register is written, an MII management write cycle is performed using the 16-bit data defined in the PHY Address register by the pre-configured PHY device and PHY register addresses. The write operation is completed when the BUSY bit in the MII Indicators register returns to 0.

MII Read Data register and bit definition

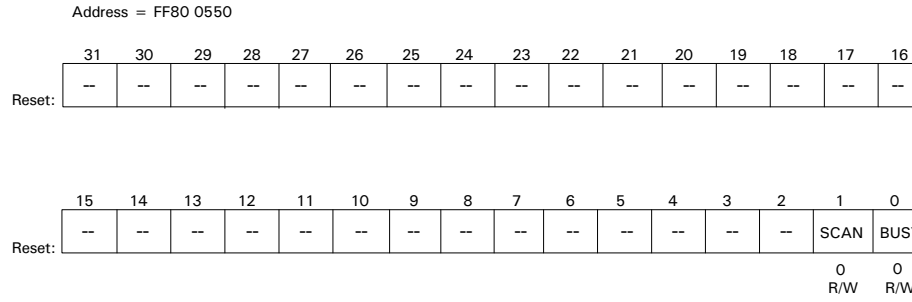
There is one field in the MII Read Data register — **MRDD:** MII read data (D15:00).



The MII Read Data register provides read data after an MII management read cycle. 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 ia available after the BUSY bit in the MII Indicators register returns to 0.

MII Indicators register and bit definitions



Code (Bit #)	Definition of code	Description
SCAN (D01)	Automatically scan for read data in progress	When set to 1, indicates that continuous MII management scanning read operations are in progress.
BUSY (D00)	MII interface BUSY with read/write operation	When set to 1, indicates that the MII management module is currently performing an MII management read or write cycle. This bit returns to 0 when the operation is complete.

Table 81: MII Indicators register bit definition

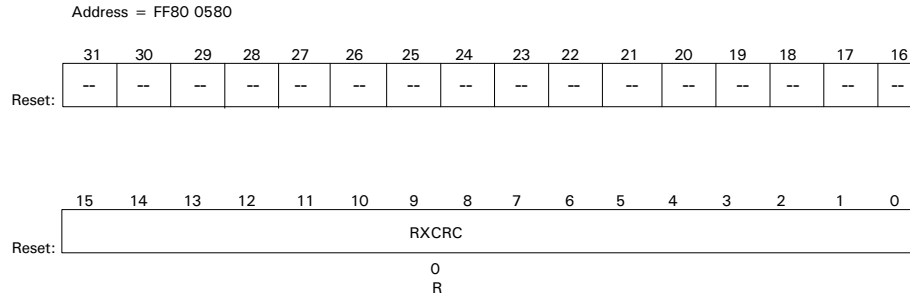
Statistics monitoring

The statistics block (STAT) serves two purposes:

- The STAT block monitors 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. The firmware can also configure the STAT block to automatically clear the error counters when they are read, using the AUTOZ bit in the STL Configuration register (see page 229).

Error Statistics registers

The EFE module supports eight Error Statistics registers. All registers use D15:D00 for the counter field, as shown in the CRC Error Counter register:



- **CRC Error Counter register** (FF80 0580) **– RXCRC.** Incremented each time a receive frame is discarded because it was received with a CRC error.

  The packet is discarded automatically if the ERXBAD field in the Ethernet General Control register is set to 0. RXCRC increments for each packet received with a bad CRC, however, no matter the value in ERXBAD.

  RXCRC is cleared automatically when read, if the AUTOZ bit in the STL Configuration register is set to 1.

- **Alignment Error Counter register** (FF80 0584) **– RXAEC.** Incremented each time a receive frame is discarded because it was received with an alignment (dribble bit) error.

  The packet is discarded automatically if the ERXBAD field in the Ethernet General Control register is set to 0. RXAEC increments for each packet received with an alignment condition, however, no matter the value in ERXBAD.

  RXAEC is cleared automatically when read, if the AUTOZ bit in the STL Configuration register is set to 1.

- **Code Error Counter register.** (FF80 0588) **– RXCEC.** Incremented each time a receive frame is discarded because it was received with a coding error. A coding error occurs when the PHY asserts the RXER input to the NET+50 MAC.

The packet is discarded automatically if the ERXBAD field in the Ethernet General Control register is set to 0. RXCEC increments for each packet received with a coding error, however, no matter the value in ERXBAD.

RXCEC is cleared automatically when read, if the AUTOZ bit in the STL Configuration register is set to 1.

- **Long Frame Counter** (FF80 058C) **– RXLFC.** Incremented each time a receive frame is discarded because it was received with a length exceeding 1518 bytes.

  The packet is discarded automatically if the ERXLNG field in the Ethernet General Control register is set to 0. RXLFC increments for each packet received with a length greater than 1518 bytes, however, no matter the value in ERXLNG.

  RXLFC is cleared automatically when read, if the AUTOZ bit in the STL Configuration register is set to 1.

- **Short Frame Counter** (FF80 0590) **– RXSFC.** Incremented each time a receive frame is discarded because it was received with less than 64 bytes.

  The packet is discarded automatically if the ERXSHT field in the Ethernet General Control register is set to 0. RXSFC increments for each packet received with a length less than 64 bytes, however, no matter the value in ERXSHT.

  RXSFC is cleared automatically when read, if the AUTOZ bit in the STL Configuration register is set to 1.

- **Late Collision Counter** (FF80 0594) **– TXLCC.** Incremented each time a transmit frame is aborted because it receives a *late collision*. A late collision is a collision that occurs after the timeframe defined in the LCOL field of the Collision Window/Collision Retry register.

  TXLCC is cleared automatically when read, if the AUTOZ bit in the STL Configuration register is set to 1.

- **Excessive (Late) Deferral Counter** (FF80 0598) **– TXLDC.** Incremented each time a transmit frame is aborted because of *excessive deferral*. Excessive deferral occurs when carrier is detected on the Ethernet receive medium for longer than expected.

TXLDC is cleared automatically when read, if the AUTOZ bit in the STL Configuration register is set to 1.

- **Maximum Collision Counter** (FF80 059C) **– TXMCR.** 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.

  TXMCR is cleared automatically when read, if the AUTOZ bit in the STL Configuration register is set to 1.

Station Address registers/multicast hash table

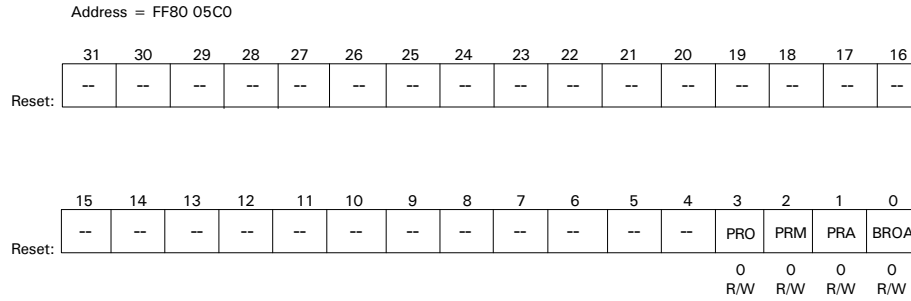
The Ethernet module uses three Station Address information registers:

- **Station Address Filter register.** A 32-bit register that contains filter controls. The bits in this register initialize to 0 during reset.
- **Station Address register.** The NET+50 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.
- **Multicast hash table.** The MAC receiver provides the SAL logic with a 6-bit CRC value that addresses the 64-bit multicast hash table. A packet is accepted 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. Otherwise, the packet is rejected.

  The 64-bit multicast hash table requires four 32-bit registers for storage. The first register (lower address) stores the enables for the lower 16 CRC addresses. The other registers store the enables for the upper 48 CRC addresses. The CRC addresses are stored right to left in ascending order.

Station Address Filter register and bit definitions



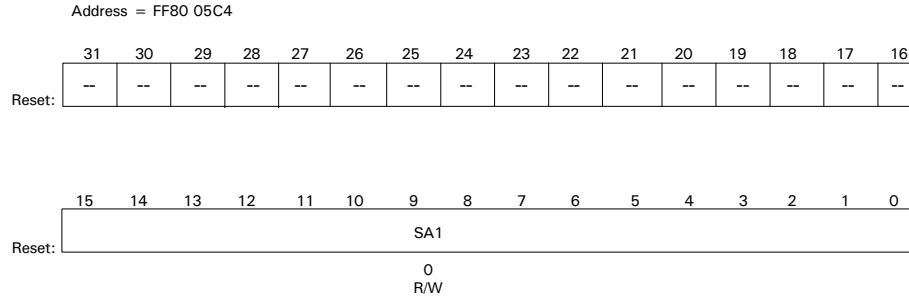
Code (Bit #)	Definition of code	Description
PRO (D03)	Enable promiscuous mode (receive *all* packets)	When PRO is set to 1, the MAC enters promiscuous mode. When operating in promiscuous mode, all packets are received without any filtering.
PRM (D02)	Accept *all* multicast packets	When PRM is set to 1, the MAC receives all multicast Ethernet packets without any filtering.
PRA (D01)	Accept multicast packets using hash table	When PRA is set to 1, the MAC receives only those multicast Ethernet packets that are selected by the multicast hash table. See "Multicast hash table entries and bit definitions" on page 257 for details.
BROAD (D00)	Accept *all* broadcast packets	When BROAD is set to 1, the MAC receives all broadcast Ethernet packets without any filtering.

Table 82: Station Address FIlter register bit definition

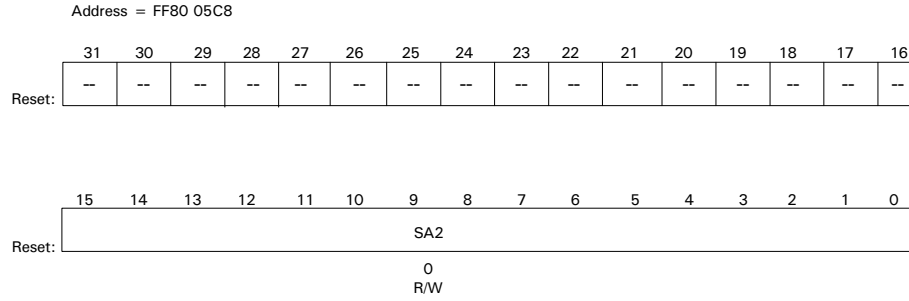
Station Address register and bit definitions

The station address bytes are loaded into the Station Address registers in Little Endian format.

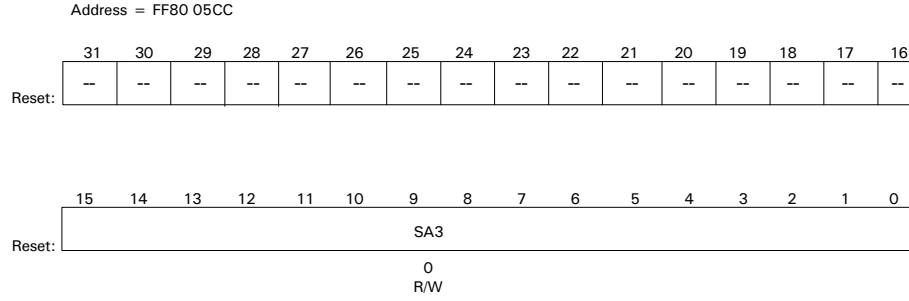
Station Register 1: Address bits 47:32



Station Register 2: Address bits 31:16



Station Address Register 3: Address bits 15:00



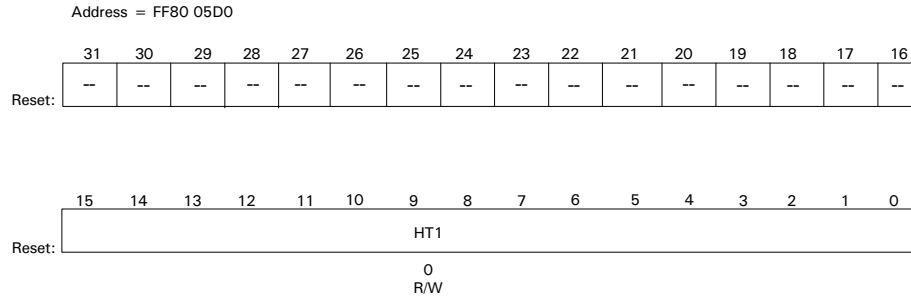
Code	Definition of Code	Description
SA1	Station Address 1	Provides the most significant (upper) 16 bits of the 48-bit station address value.
SA2	Station Address 2	Provides the middle significant (middle) 16 bits of the 48-bit station address value.
SA3	Station Address 3	Provides the least significant (lower) 16 bits if the 48-bit station address value.

Table 83: Station Address register bit definition

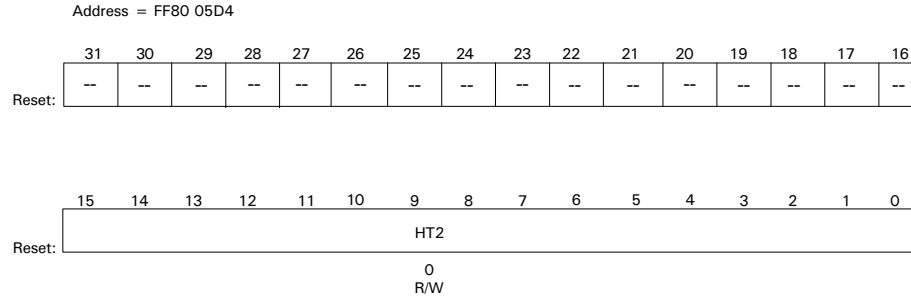
Multicast hash table entries and bit definitions

The multicast address bytes are loaded into the Station Address registers in Little Endian format.

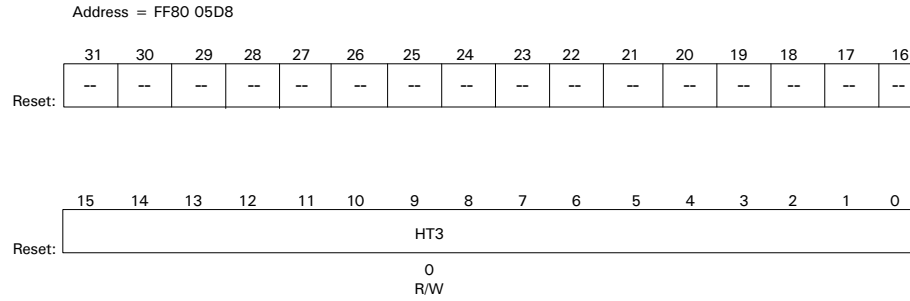
Hash table entries: Address bits 15:00



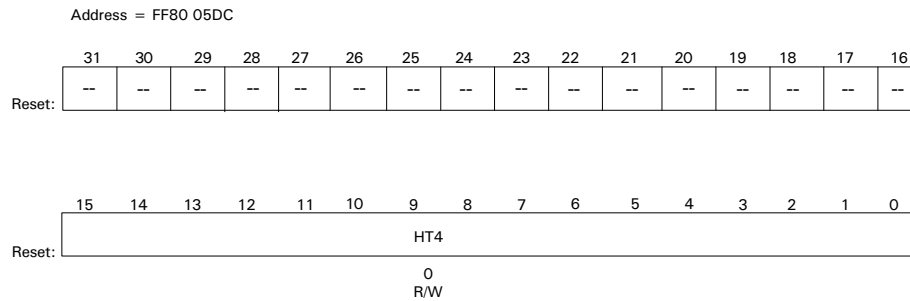
Hash table entries: Address bits 31:16



Hash table entries: Address bits 47:32



Hash table entries: Address bits: 63:48



Code	Definition of code	Description
HT1	Hash table entries [15:0]	Provides the least significant 16 bits of the 64-bit hash table.
HT2	Hash table entries [31:16]	Provides the lower middle significant 16 bits of the 64-bit hash table.
HT3	Hash table entries [47:32]	Provides the upper middle significant 16 bits of the 64-bit hash table.
HT4	Hash table entries [63:48]	Provides the most significant 16 bits of the 64-bit hash table.

Table 84: Multicast hash table bit definition

Calculating hash table entries

The following C code describes how to calculate the hash table entries based on 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+50 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+50 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;
}
```

Serial Controller Module

CHAPTER 11

The NET+50 chip supports two independent universal asynchronous/synchronous receiver/transmitter channels: serial channel A and serial channel B. Each channel supports several modes, conditions, and formats.

Note: The information in this chapter applies to both the NET+50 and NET+20M chips, unless otherwise noted.

Features

Each serial channel supports these features:

- Independent programmable bit-rate generator
- UART, HDLC, SPI modes
- High speed data transfer
  - x1 Mode: 4Mbits/sec
  - x16 Mode: 230400 bits/sec
- Digital phase-locked loop (DPLL)
- 16-bit or 32-bit CRC-CCITT generation/checking
- 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
  - Bit order
  - Programmable flags (0-15)
- 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
  - Receive buffer close
  - Transmit FIFO empty
  - Transmit FIFO half-empty
  - Transmit buffer close
  - CTS, DSR, DCD, RI state change detection
- Clock/data encoding
  - NRZ
  - NRZB
  - NRZI-M
  - NRZI-S
  - FM0
  - FM1
  - Manchester
  - Differential Manchester
- Multi-drop capable

Figure 31 shows the modules and features that comprise the serial port.

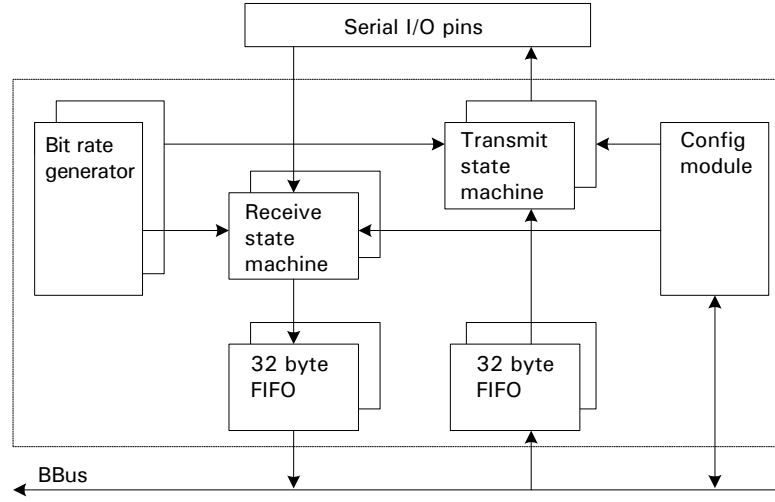


Figure 31: Serial port block diagram

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 bit-rates. See "Serial Channel Bit-Rate registers" on page 308 for more information.

Serial protocols

The serial port provides support for the following protocols:

- UART (Universal Asynchronous Receiver Transmitter)
- HDLC
- SPI

UART mode

Many applications require a simple mechanism for sending information between two pieces of equipment. The universal asynchronous receiver transmitter (UART) protocol is the de facto standard for simple serial communications.

The UART framing protocol format is:

Start bit:	0
Data:	5, 6, 7, or 8 bits
Parity (optional):	Odd or even
Stop bit:	1 or more

Because the transmitter and receiver operate asynchronously, the transmit and receive clocks between them do not need to be connected. Instead, the receiver over-samples the receive data stream by a factor of 16. Because the start bit is always a zero, the receiver can detect when real data is present on the line. (When

the UART is not transmitting data, it transmits a continuous stream of ones — considered the *idle* condition. When data transmission begins again, the transmitter sends the start bit and the receiver is enabled.)

During synchronization, the receiver waits for the start bit. When it finds the high-to-low transition, the receiver counts eight 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.

The UART can be configured to perform the following functions:

- **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 but the next character is stalled.
- **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 deasserted. The RTS and CTS pairs can be used for hardware flow control.
- **Operate in synchronous timing mode.** When using synchronous timing mode, a clock is provided with each bit; an over-sampling technique is not required. In this mode, transmit and receive clocks must be connected between two end points.

HDLC mode

The NET+50 chip HDLC controller provides these 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) with mask
- 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

Important: The NET+50 chip does not support HDLC DMA.

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:

FLAG: 01111110
Address: 01111110
Control: 01111110
Information:Nx8 bits
CRC: 16 or 32 bits
FLAG: 01111110

Bit-stuffing

The HDLC controller uses a zero insertion/deletion scheme (referred to as *bit-stuffing*) in the data between flags. When the transmitter encounters five consecutive ones (11111) in the data stream, it automatically inserts a zero. When the receiver encounters five 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 frame

Since layer 2 frames can be transmitted over point-to-point links, broadcast networks, packet networks, or circuit switch networks, an address field is needed to carry the frame's destination address. The address field is typically 8 or 16 bits, depending on the data link layer protocol. The SDLC and LAPB protocols use an 8-bit address, for example, and the LAPD protocol uses a 16-bit address field. The NET+50 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.

Note: The 8-bit or 16-bit control field is used to define the frame type and flow control mechanism. The use of this field is protocol-dependent.

The NET+50 chip HDLC controller offers no hardware support for handling this field.

Layer 3 frame

Data is transmitted in the data field. The data field contains the layer three protocol information.

Error detection

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 CRC most significant bit is transmitted first. For all other octets (address, control, data), the least significant bit is transmitted first.

SPI Mode

The SPI (serial peripheral interface) controller provides:

- **Full-duplex, synchronous interface.** The master interface operates in a broadcast mode, activating the slave interface with the select (SEL*) signal. The master interface can also be configured to address more than one slave interface by driving GPIO bits (in GPIO mode) using firmware rather than hardware.
- **Four-wire connection (TXD, RXD, CLK, SEL*).** The transmitter and receiver use the same clock, but different clock edges. The timing reference across a single SPI domain is provided by the master's bit rate generator.
- **Character-oriented data channel.** The protocol can provide simple parallel/serial data conversion to stream serial data between memory and a peripheral, as well as operating multi-drop serial connections.
- **Master and slave configurations.** The SPI port simultaneously transmits and receives the same number of bytes. The SPI can be configured as a master or a slave:
  - *SPI master* generates both the (SEL*) and the clock (CLK) signals.
  - *SPI slave* receives the SEL* and CLK signals as inputs.

  The transfer of information is controlled by a single clock signal and qualified with an enable signal. The slave transfers or accepts data only

when the SPI enable signal is active low in conjunction with the SPI clock signal. This allows for multiple slaves to be individually addressed in a multi-drop configuration.

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 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.

The NET+50 SPI FIFO interface is implemented as follows:

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 high during the idle points between transmit bytes and transmit packets.

When the SPI (master or slave) is receiving information:

- Receive data is moved into the receive FIFO when 4 bytes have been accumulated.
- The FIFO buffer is closed on the transition of the SEL* signal from low-to-high, when the SPI transmitter is sending a message whose length is not an integral number of 4 bytes.

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 as shown in these examples:

Terminology	What's being written	Value
C	Byte using a byte pointer	`(char*)0xffd00010=(char)data`
C	Two bytes using a short pointer	`(short*)0xffd00010=(short)data`
C	Four bytes using a standard long word pointer	`(int*)0xffd00010=(int)data`
Assembler	Byte	STRB instructions
Assembler	Half word	STRH instructions
Assembler	Long word	STR instructions

Operating in Endian modes

- Big Endian mode configuration. Transmits first the most significant bytes in the word written to the FIFO. For example, the long word `0x11223344` results in the character `0x11` being transmitted first and `0x44` being transmitted last.
- Little Endian mode configuration. Transmits first the least significant bytes in the word written to the FIFO. For example, the long word `0x11223344` results in the character `0x44` being transmitted first and `0x11` being transmitted last.

Processor interrupts vs. DMA

The transmit FIFO can be filled using processor interrupts or DMA.

- When using processor interrupts, the processor can write one long word (4 bytes) of data to the transmit FIFO when 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 — ETXRDY, ETXHALF — in Serial Channel Control Register A. The appropriate interrupt enable bit — SER 1 TX or SER 2 TX — in the GEN Interrupt Enable register must also be set.

Note: Software should check the TXEMPTY status bit rather than the TXHALF status bit when filling the FIFO using BYTE or HALF WORDS.

- When using the DMA controller, 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 and 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 Endian rules described for the transmit FIFO also apply for the receive FIFO (see "Operating in Endian modes" on page 271).

When reading from the receive FIFO, the processor must perform a long word read operation. Each time a read cycle 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.

Processor interrupts vs. DMA

The receive FIFO can be emptied using processor interrupts or DMA. The definition of a data buffer is different depending on whether you are in interrupt or DMA mode.

- In *interrupt mode*, the data buffer is one line of the FIFO; for example, 4 bytes (characters).
- In *DMA mode*, the data buffer is a complete buffer descriptor.

Using processor interrupts

When using processor interrupts, the processor can read one long word (4 bytes) of data to the receive FIFO when the RRDY bit in Serial Channel Status Register A is active high. This long word 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 (receive buffer closed) bit in Serial Channel 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.

This is the recommended process flow for the serial port receiver interrupt service routine:

1. Read Serial Channel Status Register A.
2. If RBC is true, then:
   a. Record the receiver buffer closed status (if desired).
   b. Write a 1 to the RBC bit position in serial status A.
   c. Read Serial Channel 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 — ERXDRDY, ERXBC — in Serial Channel Control Register A. The appropriate interrupt enable bit — SER 1 RX, SER 2 RX — in the GEN module Interrupt Enable register must also be set.

Using DMA

When using the DMA controller, 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 receive DMA, the ERXDMA bit in Serial Channel Control Register A must be set active high and the serial receiver interrupts should be disabled.

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.

Information transfer is controlled by a single SPI clock signal and qualified by an SPI enable signal. The clock signal is an output. The SPI enable signal must be active low for data transfer to occur, regardless of the SPI clock signal. SPI enable allows for multiple slaves to be individually addressed in a multi-drop configuration.

Configuration

The GEN module must be configured to allow the SPI interface signals from the SER module to propagate through the GEN GPIO ports (A, B and C) to the outside of the chip. The GEN module must be configured after the SPI interface has been configured; otherwise, there can be adverse effects on the devices connected to the SPI interface.

Table 85 shows how to configure the GEN module GPIO ports for the two SPI channels.

Channel	GPIO Port	Configuration
A	PORTA3	Special function input — SPI RXD
A	PORTA4	Special function output — SPI clock
A	PORTA7	Special function output — SPI TXD
A	PORTC7	Special function output — SPI enable. Use this if the hardware-generated SPI enable signal originating inside the NET+50 is needed outside the chip.
A	PORTC7 or any other available GPIO pins	GPIO output Use this if the slave SEL* signals (part of the four-wire SPI protocol) are going to be generated just by firmware.
B	PORTB3	Special function input — SPI RXD
B	PORTB4	Special function output — SPI clock

Table 85: SPI master mode channel A/B configuration

Channel	GPIO Port	Configuration
B	PORTB7	Special function output — SPI TXD
B	PORTC5	Special function output — SPI enable Use this if the hardware-generated SPI enable signal originating inside the NET+50 is needed outside the chip.
B	PORTC5 or any other available GPIO pins	GPIO output Use this if the slave SEL* signals (part of the four-wire SPI protocol) are going to be generated just by firmware. **Note:** The configuration of PORTC5 has no other effects on the operation of the SPI module.

Table 85: SPI master mode channel A/B configuration

SER module master mode configuration

Important: The MODE field in Serial Channel Control Register B must be set to '10' before the CE field in Serial Channel Control Register A is set to 1.

Follow these steps to configure SPI master mode for operation:

1 Reset the serial port by writing a 0 to Serial Channel Control Register A.

2 Configure the Serial Channel Bit-Rate register as shown:

Bit	Configuration
EBIT	1 for enable
TMODE	1 for x1 mode
RXSRC	0 for internal
TXSRC	0 for internal
CLKMUX	User-defined
TXCINV	0 for normal
RXCINV	0 for normal
CLKINV	According to the specification of the connected device
N	User-defined

If the size of the transmit data block is a multiple of 4, the FIFO FULL interrupt will be generated on the low-to-high transition of the receive

clock for the last bit; otherwise, interrupt will be generated on the low-to-high transition of the chip select signal.

3 Configure Serial Channel Control Register B as shown:

Bit	Configuration
MODE	10 for master mode
BITORDR	User-defined

4 Configure Serial Channel Control Register A as shown:

Bit	Configuration
CE	1 for enabled
WLS	11 for 8-bit operation

See "SPI timing," beginning on page 459, for SPI master mode timing information.

SPI master transmitter

The SPI master transmitter operates as follows:

- 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 should sample data on the rising edge of the SPI clock signal.
- Drives the SPI enable signal active low from the falling edge of the SPI clock for the first bit of a byte being transmitted, and inactive high after the rising edge of the SPI clock signal during the eighth bit of the byte currently transmitted.
- Drives the SPI enable signal active low to identify when data is being transmitted.The SPI clock signal never transitions from low to high while the internal SPI enable state is inactive high. A firmware-driven SEL* can still function.
- 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. This must be considered when choosing the interface's operating mode. For example, buffer underruns can occur while using a hardware-controlled SEL* in an interrupt-driven environment, particularly when operating at high bit rates.

When underruns occur, the SPI interface drives the SEL* line inactive high. In such a case, it is recommended that a firmware-driven enable is used to force the SEL* active low.

SPI master receiver

The SPI master receiver operates as follows:

- Samples the RXD input on the rising edge of the SPI clock signal while the SPI enable signal is driven active low.
- Receives one byte of inbound data for each byte of transmit data sent. The SPI master receiver cannot receive more data than is transmitted.

  When 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.

  When the SPI master transmitter sends an odd number of bytes, the SPI master receiver needs to wait 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. Some delay will occur in writing the final residual bytes; the delay is determined by the configuration of the buffer and character GAP timers.

SPI slave mode

SPI slave mode supports the peripheral side of an 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.

Information transfer is controlled by a single SPI clock signal and qualified by an SPI enable signal. The clock signal is an input. The SPI enable signal must be active low, in conjunction with the SPI clock signal, for data transfer to occur. The hardware SEL* control must be used; the slave should not use a firmware-controlled SEL* line.

Configuration

The GEN module must be configured to allow the SPI interface signals from the SER module to propagate through the GEN GPIO ports (A, B and C) to the outside of the chip. The GEN module must be configured after the SPI interface has been configured; otherwise, there can be adverse effects on the devices connected to the SPI interface.

Table 86 shows how to configure the GEN GPIO ports for the two SPI channels.

Channel	GPIO Port	Configuration
A	PORTA3	Special function input — SPI RXD
A	PORTA4	Special function input — SPI clock
A	PORTA7	Special function output — SPI TXD
A	PORTC7	Special function input — SPI enable
B	PORTB3	Special function input — SPI RXD
B	PORTB4	Special function input — SPI clock
B	PORTB7	Special function output — SPI TXD
B	PORTC5	Special function input — SPI enable

Table 86: SPI slave mode channel A/B configuration

SER module slave mode configuration

Important: The MODE field in Serial Channel Control Register B must be set to '11' before the CE field in Serial Channel Control Register A is set to 1.

Follow these steps to configure SPI slave mode for operation:

1. Reset the serial port by writing a 0 to Serial Channel Control Register A.
2. Configure the Serial Channel Bit-Rate register as shown:

Bit	Configuration
EBIT	1 for enable
TMODE	1 for x1 mode
RXSRC	1 for external
TXSRC	1 for external
CLKMUX	N/A (external timing source)

Bit	Configuration
TXCINV	0 for normal
RXCINV	0 for normal
CLKINV	According to the specification of the connected device
N	N/A (external timing source)

3 Configure Serial Channel Control Register B as follows:

Bit	Configuration
MODE	11 for slave mode
BITORDER	User-defined

4 Configure Serial Channel Control Register A as follows:

Bit	Configuration
CE	1 for enable
WLS	11 for 8-bit operation

See "SPI timing," beginning on page 459, for SPI slave mode timing information.

SPI Slave Transmitter

The SPI slave transmitter operates as follows:

- Has the first bit ready for transmission before the SPI enable signal is activated by the SPI master.
- Changes the TXD output to the next data bit for transmission on the rising edge of SPI clock, while the SPI enable signal is active low.
- Goes through a sampling of the SPI clock input. 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.
- 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 operates as follows:

- 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 goes through a sampling of the SPI clock input, which means the input is sampled three to four internal system clock ticks from the rising edge of the SPI clock input.
- When 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.

  When the master SPI transmitter sends an odd number of bytes, the SPI slave receiver waits 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 that there is some delay in the process of writing the final residual bytes. The delay is determined by the configuration of the buffer and character GAP timers.

General purpose I/O configurations

TheGEN module provides the physical layer connections for NMSI (Non Multiplexed Serial Interface) and SPI. NMSI is used for UART and HDLC protocols.

- PORTA provides NMSI and SPI for serial channel A.
- PORTB provides NMSI and SPI for serial channel B.
- Four PORTC signals are required to support NMSI and SPI.

Each GEN module interface signal can be enabled or disabled with firmware.

NMSI

NMSI requires a minimum of two signals (TXD and RXD) but each interface can be configured to use up to 10 signals. Table 87 shows how the GEN module

interface signals are allocated when serial channels A and B are configured to operate in NMSI mode.

GPIO port	Assignment for NMSI
PORTA7	TXDA
PORTA6	DTRA*
PORTA5	RTSA*
PORTA4	RXCA/OUT1A
PORTA3	RXDA
PORTA2	DSRA*
PORTA1	CTSA*
PORTA0	DCDA*
PORTC7	TXCA/OUT2A
PORTC6	RIA*
PORTB7	TXDB
PORTB6	DTRB*
PORTB5	RTSB*
PORTB4	RXCB/OUT1B
PORTB3	RXDB
PORTB2	DSRB*
PORTB1	CTSB*
PORTB0	DCDB*
PORTC5	TXCB/OUT2B
PORTC4	RIB*

Table 87: PORTA/B/C assignments for NMSI mode

SPI

The SPI mode supports a four-wire interface. Table 88 shows how the GEN module interface signals are configured when serial channels A and B are configured to operate in SPI slave mode:

GPIO port	Assignment for SPI slave
PORTA7	TXDA
PORTA3	RXDA
PORTA4	SPICLKA-IN
PORTC7	SPISELA-IN
PORTB7	TXDB
PORTB3	RXDB
PORTB4	SPICLKB-IN
PORTC5	SPISELB-IN

Table 88: PORTA/B/C assignments for SPI slave mode

Table 89 shows how the GEN module interface signals are configured when serial channels A and B are configured to operate in SPI master mode:

GPIO port	Assignment for SPI master
PORTA7	TXDA
PORTA3	RXDA
PORTA4	SPICLKA-OUT
PORTC7	SPISELA-OUT
PORTB7	TXDB
PORTB3	RXDB
PORTB4	SPICLKB-OUT
PORTC5	SPISELB-OUT

Table 89: PORTA/B/C assignments for SPI master mode

Figure 32 shows SPI master and slave/GPIO signal relationships.

Each entry reflects two GPIO pins and the related special function. For example, the entry *PORTA7/B7 Data Out* means *PORTA7* or *PORTB7*, depending on which serial channel you are using (Channel A or Channel B, respectively). *Data Out* is the transmit data output function.

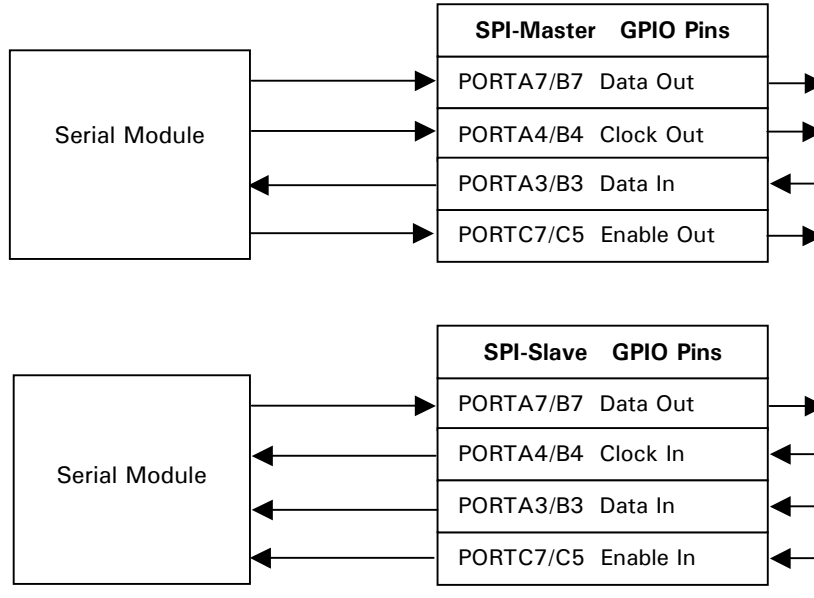


Figure 32: SPI mode/GPIO signal relationships

Serial port performance

The serial ports have a finite performance limit on their ability to handle serial protocols. The performance is limited by the speed of the SYSCLK operating the NET+50 chip. The configured speed for the internal PLL defines the SYSCLK rate:

Operating mode	Serial port maximum rate
UART (x16)	SYSCLK/64
UART (x1)	SYSCLK/4
SPI	SYSCLK/8
HDLC	SYSCLK/10

Configuration

The serial controller module has a block of configuration space that is mapped into the SER module configuration space as shown:

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
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

Table 90: Serial channel configuration registers

Address	Register
FFD0 0068	Channel 2 Status Register B
FFD0 006C	Channel 2 Status Register C
FFD0 0070	Channel 2 FIFO Data Last Register

Table 90: Serial channel configuration registers

Serial controller register diagrams

Figure 33 through Figure 36 present each serial controller register, with every bit contained in the register. Use these diagrams to gain an understanding of how the serial controller registers work within the serial controller module configuration.

Legend

Annotations in the lower right corner of bit-name boxes:

`s` = Status bit

`i` = Interrupt. This bit can cause an interrupt with the associated status bit.

`h` = HDLC only

Port/signal references:

The port references are written as `PortX-Xn`.

`X` = The port; for example, PortA.

`n` = The specific bit number; for example, 7.

The bit number applies to both ports in the entry; for example, `PortA-B7` means PortA7 and PortB7 bits.

The signal references shown with the port references are formatted similarly; for example, `TXDA-B` refers to `TXDA` and `TXDB`. Each signal "belongs" to its respective port, as shown:

Entry: `PortA-B7 TXDA-B`

Meaning: `PortA7 TXDA` and `PortB7 TXDB`

The registers are explained in the sections following these diagrams.

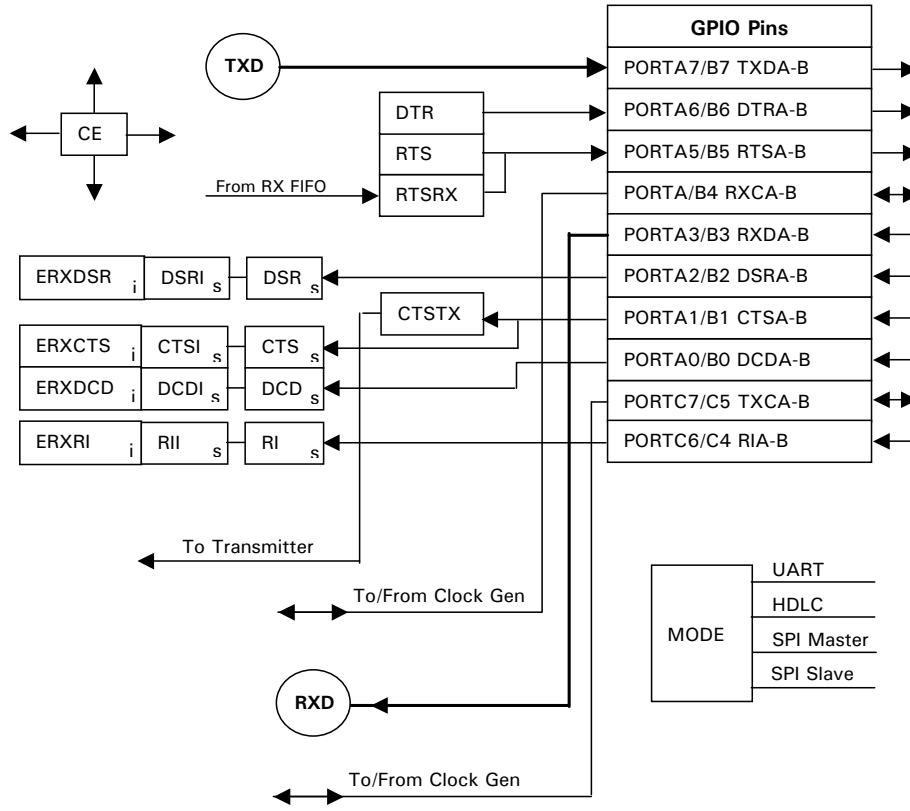


Figure 33: Serial Controller registers — 1

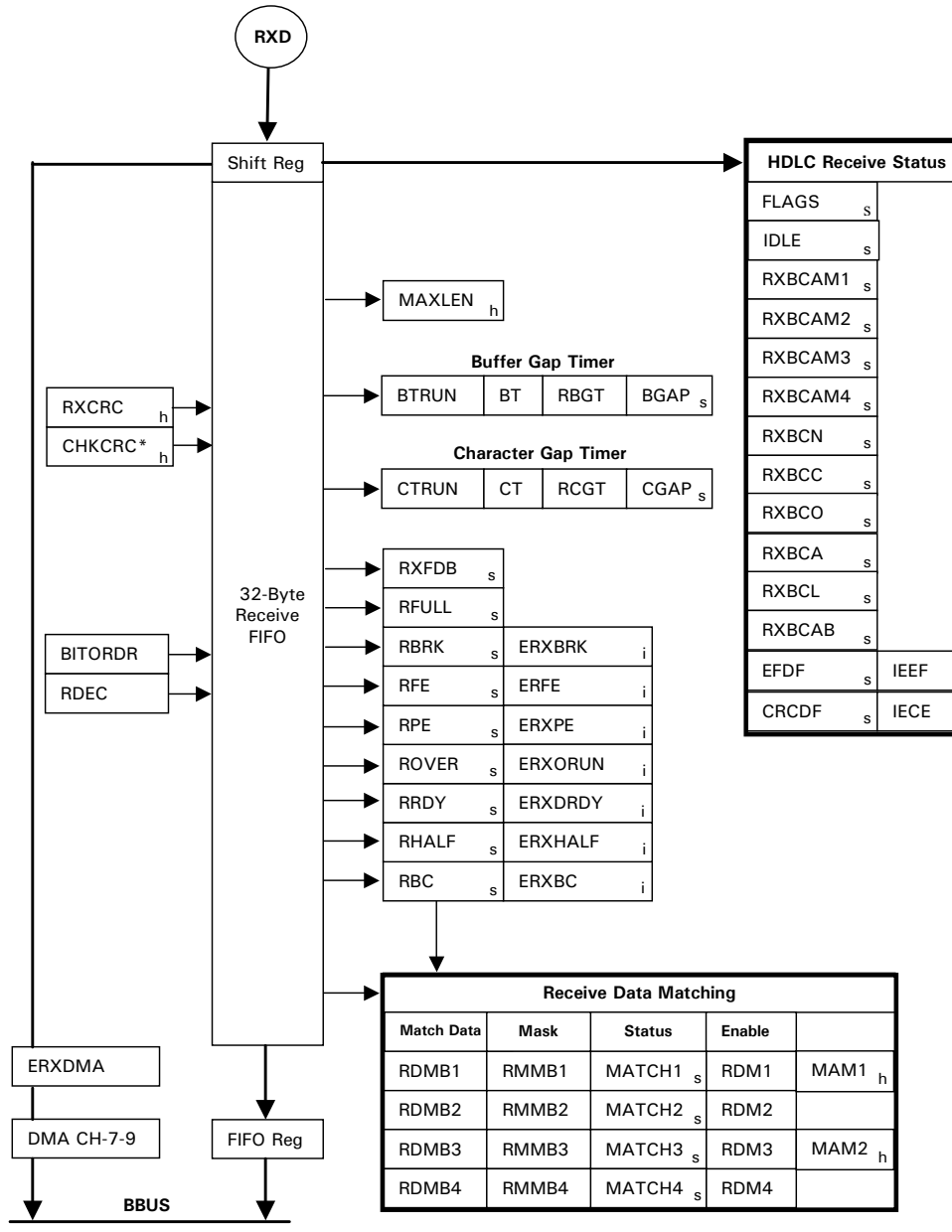


Figure 34: Serial controller registers — 2

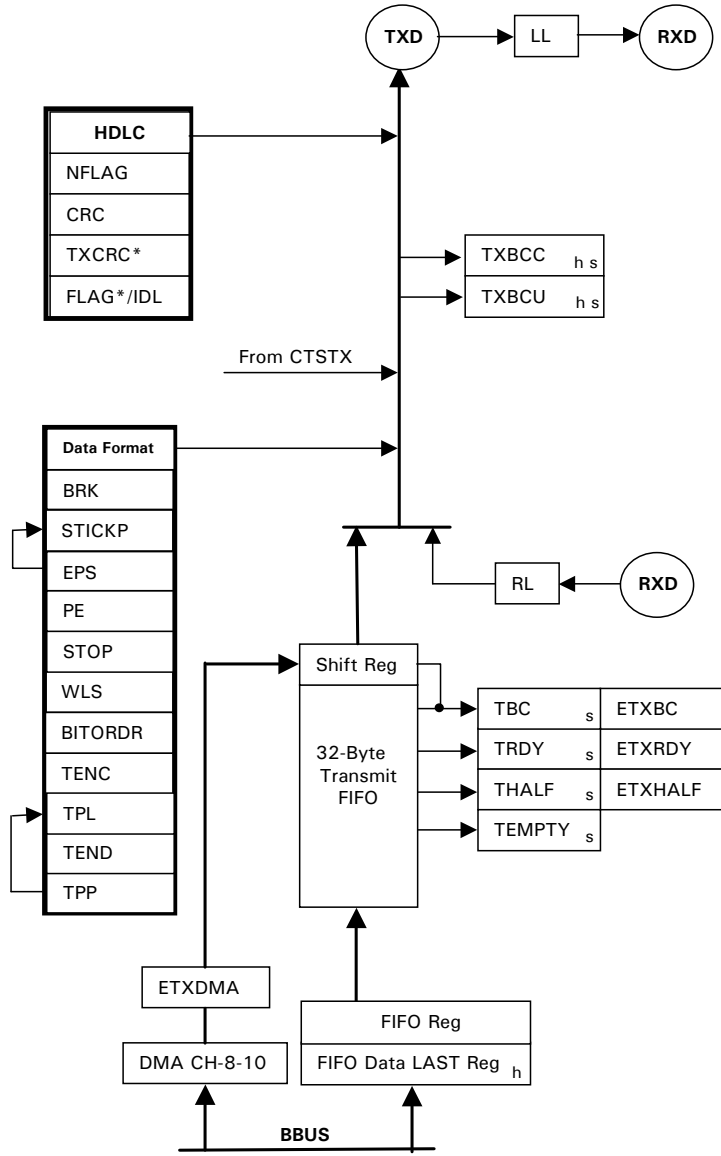


Figure 35: Serial controller registers — 3

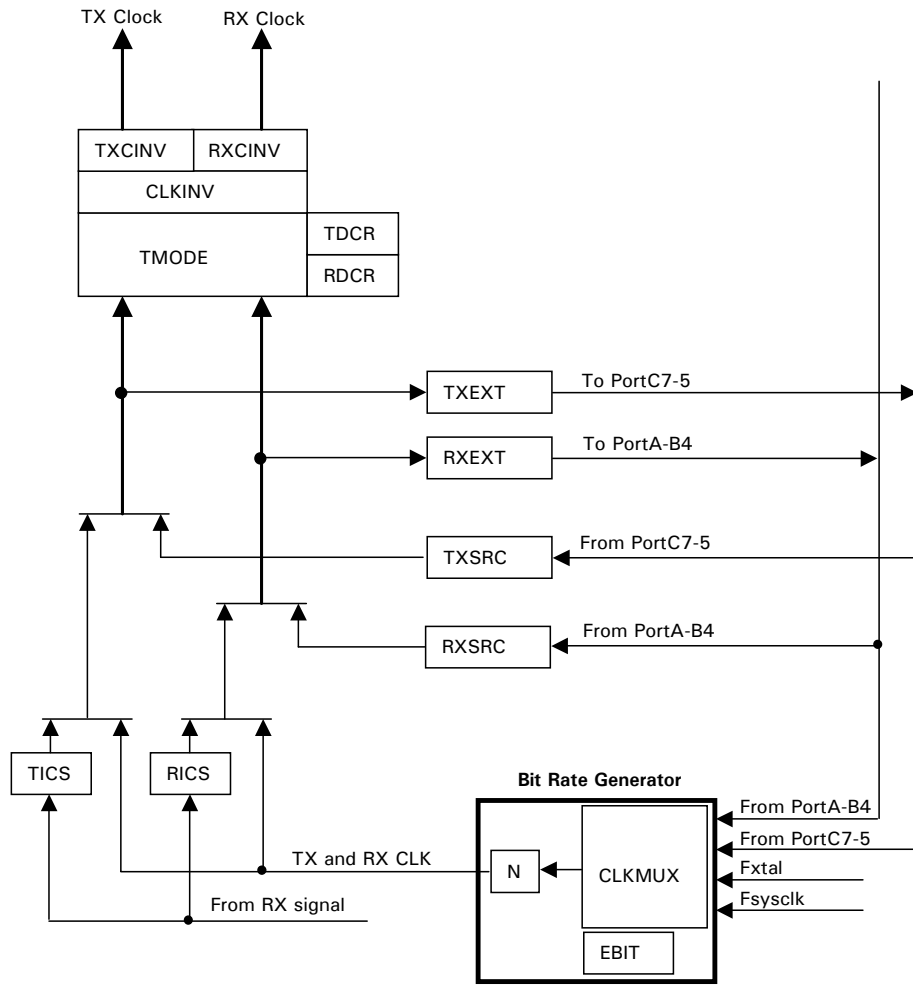


Figure 36: Serial controller registers — 4

Serial Channel registers

Serial channel registers provide the basic setup for each channel. There are five serial channel registers:

- **Serial Channel Control Register A.** All control bits are active high unless an asterisk (*) appears in the signal name; in this case, they are active low. All control bits are set to their respective inactive state on reset.
- **Serial Channel Control Register B.** All control bits are active high unless an asterisk (*) appears in the signal name; in this case, they are active low. All control bits are set to their respective inactive state on reset.
- **Serial Channel Status Register A.** All status bits are active high unless an asterisk (*) appears in the signal name; in this case, the status bits are active low.

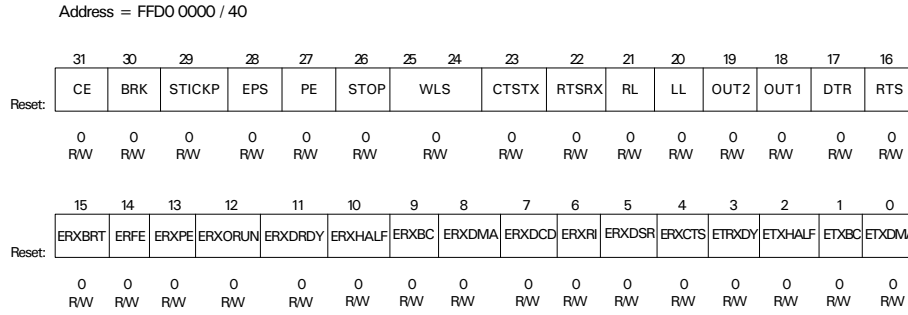
  The Receive Status bits (D31:16) are stuffed into the status field in the DMA buffer descriptor when DMA operations are enabled and a buffer is closed. The upper 8 bits of this register (D31:24) can be cleared by writing a 1 to the respective bit position (used when interrupt- driven). The upper 8 bits 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.
- **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 timing, the timing reference is provided by the bit-rate generator. When configured for external timing, the timing reference is provided by PORTA/B/C signals.
- **Serial Channel FIFO Registers.** The Serial Channel FIFO data registers are 32-bit registers used to manually interface with the serial controller FIFOs instead of using DMA support.

Serial Channel Control Register A

Important: The NET+50 chip does not support HDLC DMA.



Code (Bit #)	Definition of code	Description
CE (D31)	Channel enable	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. The CE field cannot be set until the MODE field is configured in Serial Channel Control Register B.
BRK (D30)	Send break	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 (D29)	Stick parity	Can be used to force the UART parity field to a certain state, as defined by the EPS field, instead of a parity bit being calculated against the data. STICKP applies only when the PE field is also set to 1. Setting the STICKP field to 1 forces transmission of the static parity value.
EPS (D28)	Even parity select	0 – Odd Parity 1 – Even Parity Determines whether 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.

Table 91: Serial Channel Control Register A bit definition

Code (Bit #)	Definition of code	Description
PE (D27)	Parity enable	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 (D26)	Number of stop bits	0 – 1 stop bit 1 – 2 stop bits Determines the number of stop bits in each UART transmitter.
WLS (D25:24)	Data word length select	00 – 5 bits 01 – 6 bits 10 – 7 bits 11 – 8 bits Determines the number of data bits in each UART data word.
CTSTX (D23)	Enable the transmitter with active CTS	Supports hardware handshaking. When CTSTX is set, the transmitter operates only when the external CTS signal is in the active state. An external device can use CTS to temporarily stall data transmission.
RTSRX (D22)	Enable active RTS (only when RX FIFO has space)	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 (D21)	Remote loopback	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. Primarily used as a test vehicle for external data equipment.
LL (D20)	Local loopback	Provides an internal local loopback feature. When the LL field is set to 1, it connects the serial channel receiver directly to the serial channel transmitter. Primarily used as a test vehicle for the serial channel driver firmware.

Table 91: Serial Channel Control Register A bit definition

Code (Bit #)	Definition of code	Description
OUT1/ OUT2 (D19:18)	General purpose output 1/General purpose output 2	The OUT1 and OUT2 fields provide extra miscellaneous serial channel control signal outputs. The OUT1 and OUT2 signals can be routed through PORTA/B/C pins using PORTA/B/C special function modes.
DTR (D17)	Data terminal ready active	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 (D16)	Request-to-send active	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 (D15:00)	Interrupt enable	The interrupt enable fields are used to enable an interrupt when the respective status bit is set in Serial Channel Status Register A. ■ Setting the IE field to 1 enables the interrupt. ■ Setting the IE field to 0 disables the interrupt. ■ Bits D15:D04 set up a receiver interrupt condition (see Table 92). ■ Bits D04:D00 set up a transmitter interrupt condition (see Table 93).

Table 91: Serial Channel Control Register A bit definition

This table describes each interrupt enable bit for a receiver interrupt condition. Each bit is read/write:

Bit #	Bit name	Interrupt enable description
D15	ERXBRT	Receive break
D14	ERFE	Receive framing error
D13	ERXPE	Receive parity error
D12	ERXORUN	Receive overrun
D11	ERXDRDY	Receive register ready
D10	ERXHALF	Receive FIFO half-full

Table 92: Interrupt enable bit definition - receiver

Bit #	Bit name	Interrupt enable description
D09	ERXBC	Receive buffer closed
D08	ERXDMA	UART and SPI modes only. Enables receive DMA requests (use to pause receiver). 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.
D07	ERXDCD	Change in DCD interrupt enable
D06	ERXRI	Change in RI interrupt enable
D05	ERXDSR	Change in DSR interrupt enable

Table 92: Interrupt enable bit definition - receiver

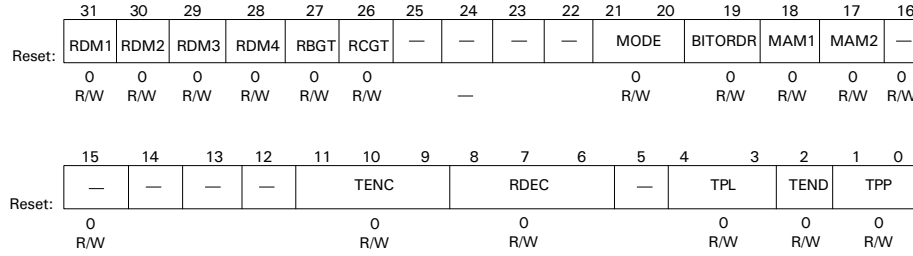
This table describes each interrupt enable bit for a transmitter interrupt condition. Each bit is read/write:

Bit #	Bit name	Description
D04	ERXCTS	Change in CTS interrupt enable
D03	ETXRDY	Transmit register empty
D02	ETXHALF	Transmit FIFO half-empty
D01	ERXBC	Transmit buffer closed
D00	ETXDMA	UART and SPI modes only. Enables transmit DMA requests (use to pause transmitter) 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.

Table 93: Interrupt enable bit definition - transmitter

Serial Channel Control Register B

Address = FFD0 0004 / 44



Code (Bit #)	Definition of code	Description
RDM1/2/3/4 (D31/D30/D29/D28)	Enable receive data match 1/2/3/4	■ When the serial channel is configured to operate in **UART** mode, the RDM bits 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 in the Receive Match register. ■ When the serial channel is configured to operate in **HDLC** mode, the RDM bits 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 two 16-bit address values, using the MAM1 and MAM2 field settings

Table 94: Serial Channel Control Register B bit definition

Code (Bit #)	Definition of code	Description
RBGT (D27)	Enable receive buffer gap timer	Enables the receive buffer gap timer. When the RBGT field is set to 1, the BGAP field in Serial Channel Status Register A is set when the timeout value defined in the Receive Buffer Gap Timer register has expired. The RBGT field determines 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 (D26)	Enable receive character gap timer	Enables the receive character gap timer. When the RCGT field is set to 1, the CGAP field in Serial Channel Status Register A is set when the timeout value defined in the Receive Character Gap Timer register has expired. The RCGT field determines the maximum allowed time from when the last byte is placed into the receiver data buffer and when the receive data buffer is closed.
MODE (D21:20)	SCC mode	00 – UART Mode 01 – HDLC Mode 10 – SPI Master Mode 11 – SPI Slave Mode 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.
BITORDR (D19)	Bit ordering	0 – Normal; Transmit/Receive LSB (least significant bits) first 1 – Reverse; Transmit/Receive MSB (most significant bits) first 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.

Table 94: Serial Channel Control Register B bit definition

Code (Bit #)	Definition of code	Description
MAM1/ MAM2 (D18/D17)	Match address mode 1/Match address mode 2	Used to combine 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 and 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 and 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

Table 94: Serial Channel Control Register B bit definition

Code (Bit #)	Definition of code	Description
TENC/ RDEC (D11:09/ D08:06)	Transmit/ receive data encoding	The NET+50 can be programmed to encode and decode the serial data stream using one of the methods listed next. The transmit and receive coding methods can be selected independently in the transmitter and receiver. ■ **NRZ (000) — Default.** A 1 is represented by a high level for the duration of the entire bit-time; a 0 is represented by a low level during the entire bit-time. ■ **NRZB (001)** — NRZB is NRZ inverted; that is, a 1 is represented by a low level during the entire bit-time and a 0 is represented by a high level during the bit-time. ■ **NRZI-Mark (010)** — A 1 is represented by a transition at the beginning of the bit-time; the level that appears during the previous cell is reversed. A 0 is represented by the absence of a transition at the beginning of the bit cell. ■ **NRZI-Space (011)** — A 0 is represented by a transition at the beginning of the bit-time; the level that appears during the previous cell is reversed. A 1 is represented by the absence of a transition at the beginning of the bit cell. ■ **FM0 (100)** — A 0 is represented by a transition at the beginning of the bit-time followed by another transition at the middle of the bit-time. A 1 is represented by a transition only at the beginning of the bit-time. ■ **FM1 (101)** — A 1 is represented by a transition at the beginning of the bit-time followed by another transition at the middle of the bit-time. A 0 is represented by a transition only at the beginning of the bit-time. ■ **Manchester (110)** — A 1 is represented by a high level during the first half of the bit-time and a low level during the second half of the bit-time. A 0 is represented by a low level during the first half of the bit cell and a high level during the second half of the bit cell. ■ **Differential Manchester (111)** — A 1 is represented by a transition at the center of the bit-time, with the opposite polarity from the transition of the bit-time at the center of the preceding bit-time. A 0 is represented by a transition at the bit-time with the same polarity as the transition at the center of the preceding bit-time. In both cases, transitions at the beginning of the bit-time set up the level required to make the correct center transition. See Figure 37, "Data coding example," on page 300 for an example of the transmit and receive coding methods.

Table 94: Serial Channel Control Register B bit definition

Code (Bit #)	Definition of code	Description
TPL (D04:03)	Transmit preamble length	00 – No preamble 01 – 8 bits 10 – 16 bits 11 – 32 bits Determines the length of the preamble pattern configured by the TPP field.
TEND (D02)	Transmitter frame ending	0 – TXD encoded only for valid data driven high otherwise 1 – TXD always encoded, even during IDLE bit times Controls whether the TXD line should idle in a high state or in an encoded *ones* state (which can be either high or low). ■ When TEND is 0, the TXD signal will be encoded during the preamble, flag, data, and ending flag conditions; the TXD pin idles in the high state when no data is available to transmit. ■ When TEND is 1, the TXD signal will always be encoded, even while the idle 1s are being transmitted.
TPP (D01:00)	Transmit preamble pattern	00 – All zeros 01 – Repeating 10s 10 – Repeating 01s 11 – All ones Determines the bit pattern that precedes the start of each transmit frame (provided the TPL field is non-zero). The preamble pattern is sent before the first flag character for the HDLC frame. The preamble is transmitted only if the serial port is configured for HDLC mode and to send 1s in the IDLE mode, and TPL is not set to 00. The preamble is not sent if the port is configured to Idle Flags. The preamble is typically transmitted to a receiving station that uses a DPLL for clock recovery. The receiving DPLL uses the regular pattern of the preamble to help it lock onto the received signal in a short, predictable time period.

Table 94: Serial Channel Control Register B bit definition

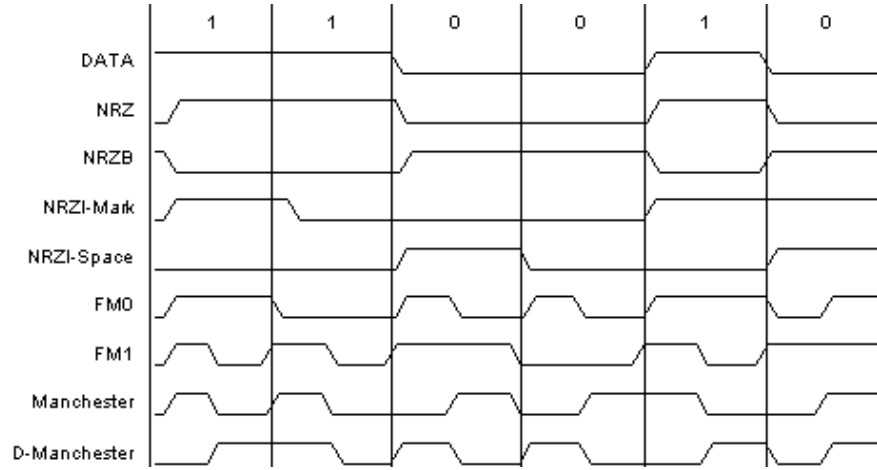


Figure 37: Data coding example

Serial Channel Status Register A

Address = FFD0 0008 / 48

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset:	MATCH1	MATCH2	MATCH3	MATCH4	BGAP	CGAP	—	—	—	—	RXFDB		DCD	RI	DSR	CTS
	0 R	0 R	0 R	0 R	0 R	0 R						0 R	0 R	0 R	0 R	0 R

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset:	RBRK	RFE	RPE	ROVER	RXRDY	RHALF	RBC	RFULL	RXDCD	RXRI	RXDSR	TXCTS	TRDY	THALF	TXBC	TXEMPTY
	0 R/C	0 R/C	0 R/C	0 R/C	0 R	0 R	0 R/C	0 R	0 R/C	0 R/C	0 R/C	0 R/C	0 R	0 R	0 R/C	0 R

Code (Bit #)	Definition of code	Description
MATCH1/ 2/3/4 (D31/D30/ D29/D28)	Character Match1/ Match2/ Match3/ Match4	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 Serial Channel Control Register B. The MATCH bit indicates that 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. When not using DMA, the MATCH bits are valid when the RBC bit in this register is set. **For UART and SPI modes only:** When the receiver is configured for DMA operation, the MATCH status bits are automatically written to the DMA receive buffer descriptor's status field.
BGAP (D27)	Buffer GAP timer	Set whenever the enable receive buffer gap timer is set in Serial Channel Control Register B and the timeout value defined in the Receive Buffer Gap Timer register has expired. This bit indicates that the maximum allowed time has occurred since the first byte was placed into the receive data buffer, closing that buffer. When not using DMA, the BGAP bit is valid only when the RBC bit in this register is set. **For UART and SPI mode only:** When the receiver is configured for DMA operation, the BGAP status bit is automatically written to the DMA receive buffer descriptor's status field.
CGAP (D26)	Character GAP timer	Set whenever the enable receive character gap timer is set in Serial Channel Control Register B and the timeout value defined in the Receive Character Gap Timer register has expired. This bit indicates that the maximum allowed time has occurred since the previous byte was placed into the receive data buffer, closing the receive data buffer. When not using DMA, the CGAP bit is valid only when the RBC bit in this register is set. **For UART and SPI mode only:** When the receiver is configured for DMA operation, the CGAP status bit is automatically written to the DMA receive buffer descriptor's status field.

Table 95: Serial Channel Status Register A bit definition

Code (Bit #)	Definition of code	Description
RXFDB (D21:20)	Receive FIFO data available	00 – Full-word 01 – One byte 10 – Half-word 11 – Three bytes (LENDIAN determines which three) Identifies the number of valid bytes contained in the next long word to be read from the Serial 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 Serial Channel FIFO data register.
DCD (D19)	Current data carrier detect state	0 – Inactive 1 – Active Identifies the current state of the EIA data carrier detect signal.
RI (D18)	Current ring indicator state	0 – Inactive 1 – Active Identifies the current state of the EIA ring indicator signal.
DSR (D17)	Current data set ready state	0 – Inactive 1 – Active Identifies the current state of the EIA data set ready signal.
CTS (D16)	Current clear-to-send state	0 – Inactive 1 – Active Identifies the current state of the EIA clear-to-send signal.
RXBRK (D15)	Receive break interrupt pending	Indicates that a UART receive break condition has been detected. When set, the RXBRK field remains set until acknowledged. The field is acknowledged by writing a 1 to the same bit position in Serial Channel Status Register A. The RXBRK status condition can be programmed to generate an interrupt by setting ERXBRT (D15) in Serial Channel Control Register A.

Table 95: Serial Channel Status Register A bit definition

Code (Bit #)	Definition of code	Description
RXFE (D14)	Receive framing error interrupt pending	Indicates that a receive framing error condition has been detected. When set, the RXFE field remains set until acknowledged. The field is acknowledged by writing a 1 to the same bit position in Serial Channel Status Register A. The RXFE status condition can be programmed to generate an interrupt by setting ERFE (D14) in Serial Channel Control Register A.
RXPE (D13)	Receive parity error interrupt pending	Indicates that a receive parity error condition has been detected. When set, the RXPE field remains set until acknowledged. The field is acknowledged by writing a 1 to the same bit position in Serial Channel Status Register A. The RXPE status condition can be programmed to generate an interrupt by setting ERXPE (D13) in Serial Channel Control Register A.
RXORUN (D12)	Receive overrun interrupt pending	Indicates that a receive overrun error condition has been detected. An overrun condition indicates that the FIFO was full while data needed to be written by the receiver. When the FIFO is full, any new receive data will be discarded; the contents of the FIFO prior to the overrun condition will remain the same. When set, the RXORUN field remains set until acknowledged. The field is acknowledged by writing a 1 to the same bit position in Serial Channel Status Register A. The RXORUN status condition can be programmed to generate an interrupt by setting ERXORUN (D12) in Serial Channel Control Register A.

Table 95: Serial Channel Status Register A bit definition

Code (Bit #)	Definition of code	Description
RXRDY (D11)	Receive register ready interrupt pending	Indicates data is available to be read from the FIFO data register. Before reading the FIFO data register, the RXFDB field in Serial Channel Status Register A must be read to determine how many active bytes are available during the next read of the FIFO data register. The RXRDY field is typically used only in interrupt-driven applications; it is not used for DMA operation. The RRDY status condition can be programmed to generate an interrupt by setting ERXDRDY (D11) in Serial Channel Control Register A. The RXRDY bit is never active while the RXBC bit is active. The RXBC bit must be acknowledged to activate the RXRDY bit. **For UART and SPI mode only:** When the receiver is configured to operate in DMA mode, the interlock between RXBC and RXRDY is handled automatically in hardware.
RXHALF (D10)	Receive FIFO half-full interrupt pending	Indicates that the receive data FIFO contains at least 16 bytes. The RXHALF field is typically used only in interrupt-driven applications; it is not used for DMA operation. The RXHALF status condition can be programmed to generate an interrupt by setting ERXHALF (D10) in Serial Channel Control Register A.

Table 95: Serial Channel Status Register A bit definition

Code (Bit #)	Definition of code	Description
RXBC (D09)	Receive buffer closed interrupt pending	Indicates a receive buffer closed condition. When set, the RXBC field remains set until acknowledged. This field is acknowledged by writing a 1 to the same bit position in Serial Channel Status Register A. When configured to operate in DMA mode, RXBC is automatically acknowledged by hardware (UART and SPI modes only). The RXBC status condition can be programmed to generate an interrupt by setting the appropriate IE bit in Serial Channel Control Register A. **For UART and SPI applications** The RXBC field indicates that bits D31:26 in Serial Channel Status Register A are valid. While the RXBC field is active, the RRDY bit is not active. To activate RXRDY (to read the data FIFO), the RXBC bit must be acknowledged. This interlock between RXBC and RXRDY is defined to allow the firmware driver to read status bits D31:26 in Serial Channel Status Register A. When operating in DMA mode, the D31:26 status bits are automatically written to the receive DMA buffer descriptor and the interlock between RXBC and RXRDY is handled automatically in hardware. **For HDLC applications** The RXBC field indicates that bits D25:16 in (HDLC) Status Register B are valid. While the RXBC field is active, the RXRDY bit is not active. To activate RXRDY (to read the data FIFO), the RBC bit must be acknowledged. This interlock between RXBC and RXRDY is defined to allow the firmware driver to read the status bits D25:D16 in Status Register B.
RXFULL (D08)	Receive FIFO full	Indicates that the receive data FIFO is currently full.
RXDCD (D07)	Change in DCD interrupt pending	Indicates a state change in the EIA data carrier detect signal. When set, the RXDCD field remains set until acknowledged. The field is acknowledged by writing a 1 to the same bit position in Serial Channel Status Register A. The RXDCD status condition can be programmed to generate an interrupt by setting ERXDCD (D07) in Serial Channel Control Register A.

Table 95: Serial Channel Status Register A bit definition

Code (Bit #)	Definition of code	Description
RXDSR (D05)	Change in DSR interrupt pending	Indicates a state change in the EIA data set ready signal. When set, the DSRI field remains set until acknowledged. The field is acknowledged by writing a 1 to the same bit position in Serial Channel Status Register A. The RXDSR status condition can be programmed to generate an interrupt by setting ERXDSR (D05) in Serial Channel Control Register A.
TXCTS (D04)	Change in CTS interrupt pending	Indicates a state change in the EIA clear-to-send signal. When set, the TXCTS field remains set until acknowledged. The field is acknowledged by writing a 1 to the same bit position in Serial Channel Status Register A. The TXCTS status condition can be programmed to generate an interrupt by setting ERXCTS (D04) in Serial Channel Control Register A.
TXRDY (D03)	Transmit register empty interrupt pending	Indicates data can be written to the FIFO data register. The TXRDY field is typically used only in interrupt-driven applications; it is not used for DMA operation. The TXRDY status condition can be programmed to generate an interrupt by setting ETXRDY (D03) in Serial Channel Control Register A. The TXRDY bit is never active while the TXBC bit is active. The TXBC bit must be acknowledged to activate the TXRDY bit. **For UART and SPI mode only:** When the transmitter is configured to operate in DMA mode, the interlock between TXBC and TXRDY is handled automatically in hardware.
TXHALF (D02)	Transmit FIFO half-empty interrupt pending	Indicates that the transmit data FIFO contains room for at least 16 bytes. The TXHALF field is typically used only in interrupt-driven applications; it is not used for DMA operation. The TXHALF status condition can be programmed to generate an interrupt by setting ETXHALF (D02) in Serial Channel Control Register A. **Note:** Software should check the TXEMPTY status bit rather than the TXHALF status bit when filling the FIFO using BYTE or HALF WORDS.

Table 95: Serial Channel Status Register A bit definition

Code (Bit #)	Definition of code	Description
TXBC (D01)	Transmit buffer closed interrupt pending	Indicates a transmit buffer closed condition. When set, the TXBC field remains set until acknowledged. The field is acknowledged by writing a 1 to the same bit position in Serial Channel Status Register A. When the transmitter is configured to run in DMA mode, the TXBC bit is automatically acknowledged by hardware (for UART and SPI modes only). The TXBC status condition can be programmed to generate an interrupt by setting ERXBC (D01) in Serial Channel Control Register A. **For UART and SPI applications** The TXBC field is set when the last character in the transmit FIFO has been shifted out of the shift register and the FIFO is currently empty. While the TBC field is active, the TXRDY bit is not active. To activate TXRDY (to write to the data FIFO), the TXBC bit must be acknowledged. When operating in DMA mode, the interlock between TXBC and TXRDY is handled automatically in hardware. **For HDLC applications** The TXBC field indicates that bits D15:14 in (HDLC) Status Register B are valid. While the TXBC field is active, the TXRDY bit is not active. To activate TXRDY (to write to the data FIFO), the TXBC must be acknowledged. This interlock between TXBC and TXRDY is defined to allow the firmware driver to read the status bits D15:14 in Status Register B.
TXEMPTY (D00)	Transmit FIFO empty	Indicates that the transmit data FIFO is currently empty. TXEMPTY simply reports the status of the FIFO; it does not indicate that the character currently in the Transmit Shift register has been transmitted. The TXBC bit must be used for the “all-sent” condition. **Note:** Software should check the TXEMPTY status bit rather than the TXHALF status bit when filling the FIFO using BYTE or HALF WORDS.

Table 95: Serial Channel Status Register A bit definition

Serial Channel Bit-Rate registers

The serial channel can be configured to operate in either X1 mode (synchronous) or X16 mode (asynchronous). When using the internal bit-rate generator, the frequency must be properly configured to match the mode being used.

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 X1 mode. The serial channel clock is provided directly from the PORTA/B/C interface's OUT1/OUT2 inputs. The TXSRC and RXSRC bits, when set, override any configuration setting for the PORTA4, PORTB4, and PORTC7 and PORTC5 pins in the GEN module. Setting TXSRC to 1 overrides any setting of the TXEXT bit (D26) in the Serial Channel Bit-Rate registers.

Figure 38 shows the relationship between CLK, TXD, and RXD. TXD is driven active from the falling edge of the CLK signal. The RXD input is sampled using the rising edge of the CLK signal.

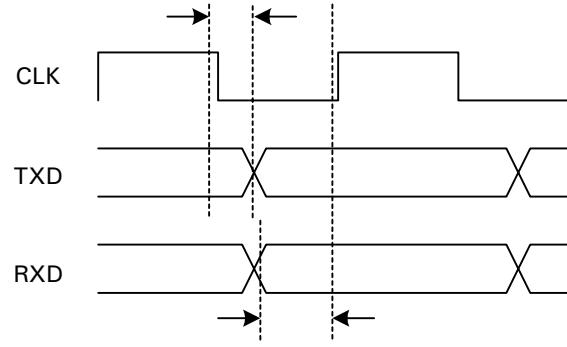
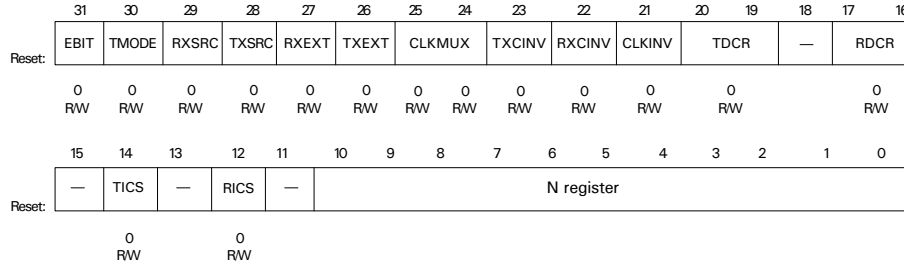


Figure 38: Serial channel X1 timing

Serial Channel Bit-Rate register

Address = FFD0 000C / FFD0 004C



Code (Bit #)	Definition of code	Description
EBIT (D31)	Bit-rate generator enable	Enables or disables the internal bit-rate generator. A 1 in this field allows the bit-rate generator to operate.
TMODE (D30)	Timing mode	0 – X16 Mode (asynchronous UART timing) 1 – X1 Mode (synchronous timing) Configures the serial channel to operate in X1 mode or X16 mode. When the TMODE field is set to 1, additional timing configuration is provided by the TDCR (D20:19) and RDCR (17:16) fields.
RXSRC (D29)	Receive clock source	0 – Internal 1 – External (input via PORTA4/B4) Controls the source of the receiver clock. The receiver clock can be provided by an internal source, as determined by the value in the RICS (D12) field, or by an input on PORTA4/B4.
TXSRC (D28)	Transmit clock source	0 – Internal 1 – External (input via PORTC5/C7) Controls the source of the transmitter clock. The transmitter clock can be provided by an internal source, as determined by the value in the TICS (D14) field, or by an input on PORTC5/C7.

Table 96: Serial Channel Bit-Rate register bit definition

Code (Bit #)	Definition of code	Description
RXEXT (D27)	Drive receive clock external	0 – Disable 1 – Enable; drive RXCLK out of PORTA4/B4 Enables the receiver clock to be driven on the PORTA4/B4 pin. The PORTA4/B4 pin must be configured as special function output.
TXEXT (D26)	Drive transmit clock external	0 – Disable 1 – Enable; drive TXCLK out of PORTC5/C7 Enables the transmitter clock to be driven on the PORTC5/C7 pin. The PORTC5/C7 pin must be configured as special function output.
CLKMUX (D25:24)	BRG input clock	00 – Input clock defined by F_{XTAL} 01 – Input clock defined by F_{SYSCLK} 10 – Input clock defined by input on PORTA4/B4 11 – Input clock defined by input on PORTC5/C7 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 SYSCLK output, an input signal on the PORTA4/B4 pin, or an input signal on the PORTC5/C7 pin. The PORTA4/B4 or PORTC5/C7 pin must be configured as special function input.
TXCINV (D23)	Transmit clock invert	0 – Normal; TXD driven on falling edge of TX clock 1 – Inverted; TXD driven on rising edge of TX clock Controls the relationship between transmit clock and transmit data. When TXCINV is set to 0, transmit data changes relative to the high-to-low transition of the transmit clock. When TXCINV is set to 1, transmit data changes relative to the low-to-high transition of the transmit clock.
RXCINV (D22)	Receive clock invert	0 – Normal; RXD sampled on rising edge of RX clock 1 – Inverted; RXD sampled on falling edge of RX clock Controls the relationship between receive clock and receive data. ■ When RXCINV is set to 0, the receive data input is sampled at the low-to-high transition of the receive clock. ■ When RXCINV is set to 1, the receive data input is sampled at the high-to-low transition of the receive clock.

Table 96: Serial Channel Bit-Rate register bit definition

Code (Bit #)	Definition of code	Description
CLKINV (D21)	Clock invert	0 - Initial value of the clock signal is low 1 - Initial value of the clock signal is high
TDCR/ RDCR (D20:19/ D17:16)	Transmit/ receive divide clock rate	00 – 1x clock mode (only NRZ or NRZI allowed) 01 – 8x clock mode 10 – 16x clock mode 11 – 32x clock mode Determine the divide ratio for the transmitter and receiver clocks. ■ **If DPLL is not used and you are not using UART mode**, select a value of 1x. ■ **If DPLL is not used and you are using UART**, select 8x, 16x, or 32x (16x is recommended). In most applications, the TDCR and RDCR configurations should match. ■ **When the DPLL is used,** the selection of TDCR/RDCR is a function of the transmitter encoding. The NRZ and NRZI modes can use the 1x configuration; all other encoding must use the 8x, 16x, or 32x configuration mode. The 8x configuration provides the highest possible data rate; the 32x mode provides the highest possible resolution. The TMODE bit (D30) in the Bit-Rate register is maintained for NET+50 family backward compatibility. When setting the TDCR or RDCR registers to a non-zero value, the TMODE bit must be set to 1. When the TMODE, TCDR, and RDCR fields are all set to 0, the port defaults to 16x mode of operation.
TICS (D14)	Transmit internal clock source	0 – BRG 1 – DPLL When the TXSRC field is set to zero, the transmitter operates using an internal clock. There are two sources for internal clocks: the bit-rate generator (BRG) and the receiver digital phase lock loop (DPLL). The BRG uses a divider mechanism for clock generation. The DPLL is used to extract the clock from the incoming receive data stream. When TICS is set to 0, the transmitter uses the BRG output for this clock. When TICS is set to 1, the transmitter uses the extracted clock provided by the DPLL.

Table 96: Serial Channel Bit-Rate register bit definition

Code (Bit #)	Definition of code	Description
RICS (D12)	Receiver internal clock source	0 – BRG 1 – DPLL When the RXSRC field is set to zero, the receiver operates using an internal clock. There are two sources for internal clocks: the bit-rate generator (BRG) and the receiver digital phase lock loop (DPLL). The BRG uses a divider mechanism for clock generation. The DPLL is used to extract the clock from the incoming receive data stream. When RICS is set to 0, the receiver uses the BRG output for this clock. When RICS is set to 1, the receiver uses the extracted clock provided by the DPLL.
N Register (D10:00)		The effective frequency of the BRG is defined by these equations: $F_{BRG} = F_{XTAL} / [2 * 16 * (N + 1)]$ 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 The maximum value for F_{OUT1} and F_{OUT2} is $F_{SYSCLK}/ 4$. See Table 97 on page 313 for sample bit rates.

Table 96: Serial Channel Bit-Rate register bit definition

Table 97 shows bit rates generated by the bit-rate generator using an 18.432 MHz crystal and a system PLL multiplier of 9 (SYS_CLK = 44.2368 MHz). The following equations were used to calculate these rates:

- X1 mode: `FBRG = FXTAL/[2*(N+1)]`

  SPI master uses this mode.

- X16 mode: `FBRG = FXTAL/[2*16*(N+1)]`

  RS232 uses this mode.

		N register	
Bit-rate	**CLKMUX**	**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	95	5

Table 97: Bit-rate examples

The following shows how X1 and X16 mode values were calculated for a bit rate of 9600:

```
FXTAL = crystal/5 = 18.432MHz/5 = 3.6864MHz
```

For X1 mode:

```
9600 = 3686400/2*(N+1) or
2*(N+1) = 3686400/9600 or
2*(N+1) = 384 or
N = 384/2 -1 = 191
```

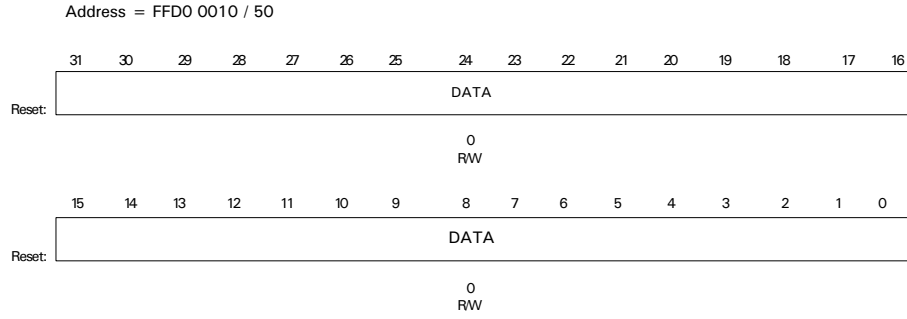
For X16 mode:

```
9600 = 3686400/2*16*(N+1) or
2*16*(N+1) = 3686400/9600 or
2*16*(N+1) = 384 or
N = 384/32-1 = 11
```

Serial Channel FIFO registers

Writing to the Serial Channel FIFO register loads the transmit FIFO. This register can be written only when the transmit data register ready bit (TXRDY) is set in Serial Channel Status Register A. Writing to the Serial Channel FIFO register automatically clears the TXRDY bit.

Reading from this register empties the receive FIFO. Data is available whenever the RXRDY bit is set in Serial Channel Status Register A. The Serial Channel Status register identifies how many bytes are available to be read (RXFDB field). Reading the FIFO register automatically clears the RXRDY bit in the Serial Channel Status register.



Receive Gap Timer registers

The SER module uses two "gap timer" registers to close out serial data buffers:

- **Receive Buffer Timer register.** A 32-bit register; all bits are set to 0 on reset.

  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 when 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 (UART and SPI modes only.)
– If the serial channel is configured to operate in interrupt mode, the expiration of the timer causes an interrupt to be generated.

The Receive Buffer Timer register operates using the external crystal clock (with a divide-by-5 prescaler) and a 9-bit prescaler within the SER module. The register is configured with a 15-bit programmable counter. The effective buffer timer value is defined by the following equation:

```
TIMEOUT =  [512 * (BT + 1)] / FXTAL
```

- **Receive Character Timer register.** A 32-bit register; all bits are set to 0 on reset.

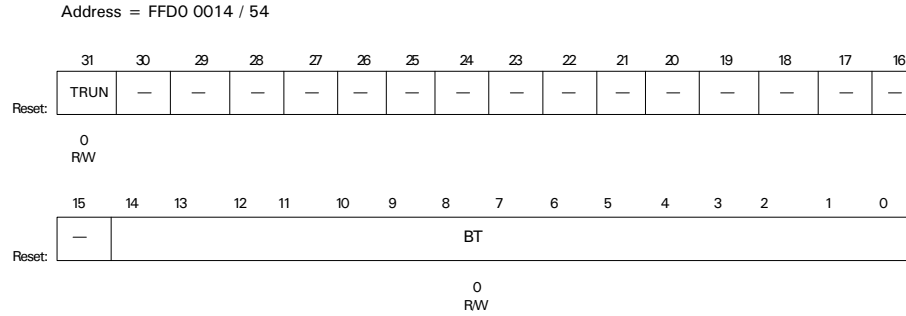
  The receive character gap timer closes out a receive serial data buffer due to a gap between characters. This timer is configured to provide an interval in the range of 100us to 145ms. The timer is reset whenever a character is received. When the timer reaches its programmed threshold, the receive data 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 (UART and SPI modes only).
  – If the serial channel is configured to operate in interrupt mode, the expiration of the timer causes an interrupt to be generated.

  The Receive Character Timer register operates using the external crystal clock (with a divide-by-5 prescaler) and a 9-bit prescaler within the SER module. The register is configured with a 10-bit programmable counter. The effective buffer timer value is defined by the following equation:

```
TIMEOUT =  [512 * (CT + 1)] / FXTAL
```

Receive Buffer Timer register and bit definitions

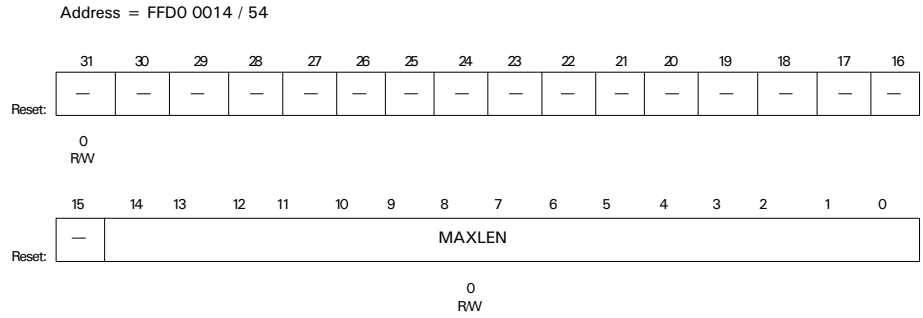


Code (Bit #)	Definition of code	Description
TRUN (D31)	Enable timer to run	Set TRUN to 1 to allow the receive buffer gap timer to operate.
BT (D14:00)	BT timer	The required value for the receive buffer gap timer is a function of the channel bit-rate and the maximum receive buffer size. The recommended approach is to set the buffer gap timer to 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: `BT RECOMMENDED = [(MAX_RX_BUF_SIZE * 1.10 * FXTAL) / (Bit-Rate * 512)] - 1` If the resulting value does not produce a value that fits within the range for BT, reconsider the size selected for MAX_RX_BUF_SIZE.

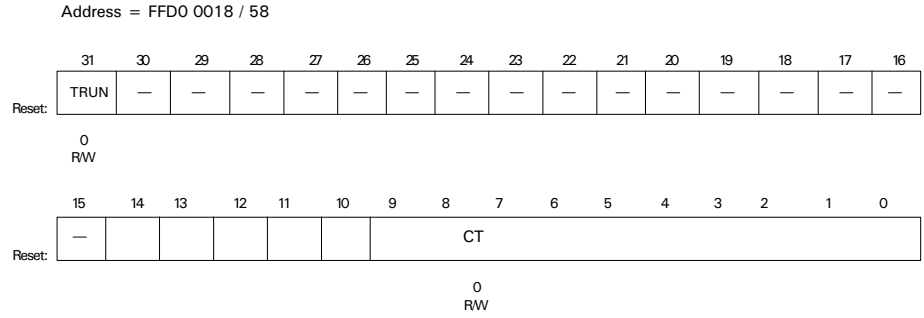
Table 98: Receive Buffer Timer register bit definition

HDLC mode

When a serial channel is configured to operate in HDLC mode, the Receive Buffer Timer register provides the HDLC maximum frame length value — `MAXLEN`.



Receive Character Timer register and bit definitions



Code (Bit #)	Definition of code	Description
TRUN (D31)	Enable timer to run	Set TRUN to 1 to allow the receive character gap timer to operate.
CT (D09:00)	CT Value	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 * FXTAL) / (Bit-Rate * 512)] - 1`

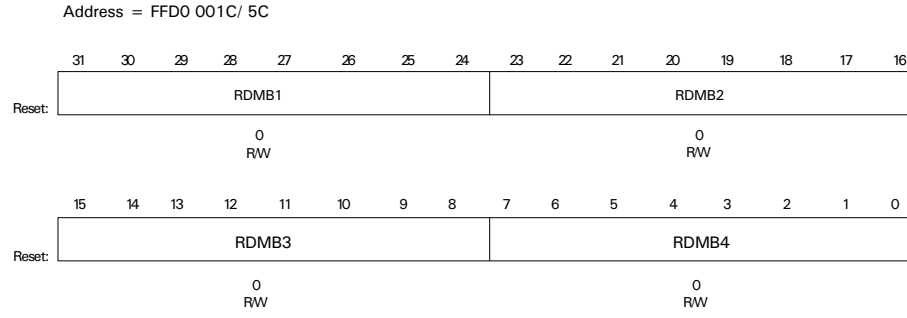
Table 99: Receive Character Timer register bit definition

Receive Match registers

The SER module uses two Receive Match registers:

- **Receive Match register.** Can be configured for either UART or HDLC:
  - **When configured for UART.** Provides the data bytes that 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 Serial Channel Control Register B, the current receive data buffer is closed by the serial channel.
  - **When configured for HDLC.** Provides the data bytes that 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 settings in Serial Channel Control Register B. If the address comparison function is enabled and a match occurs, the HDLC data packet is accepted. If the address comparison function is disabled, all HDLC packets are accepted. The address match comparators can be configured as four 8-bit address values or two 16-bit address values using the MAM1 and MAM2 bits in Serial Channel Control Register B.
- **Receive Match MASK register.** Can mask individual bits within the Receive Match register, for both UART and HDLC configurations. This register masks those bits in the Receive Match data register that should *not* be included in the match comparison.

Receive Match register and bit definitions

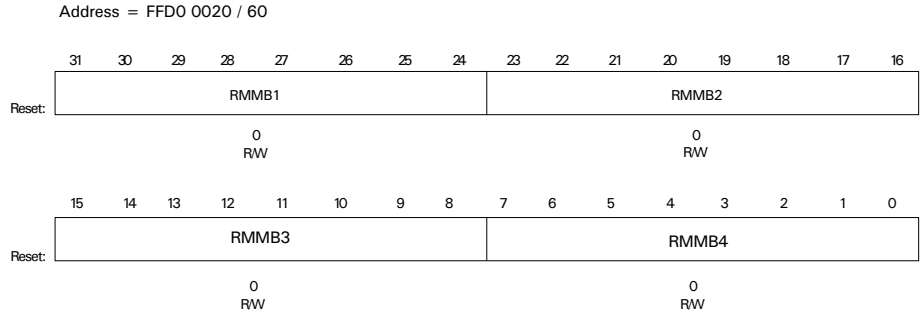


Code (Bit #)	Definition of code
RDMB1 (D31:24)	Receive data match byte 1
RDMB2 (D23:16)	Receive data match byte 2
RDMB3 (D15:08)	Receive data match byte 3
RDMB4 (D07:00)	Receive data match byte 4

Table 100: Receive Match register bit definition

Receive Match MASK register and bit definitions

To mask a bit in the match comparison function, place a 1 in the same bit position within this register.



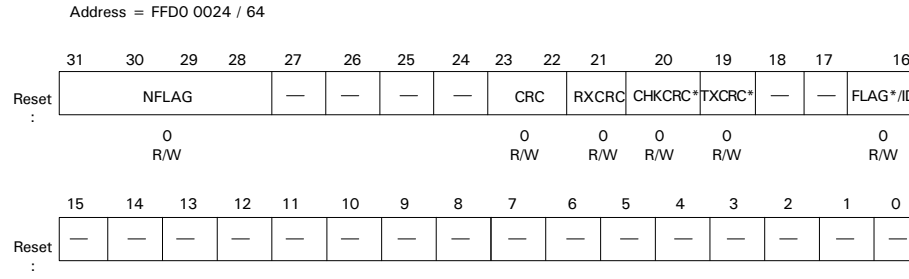
Code (Bit #)	Definition of code
RMMB1 (D31:24)	Receive mask match byte 1
RMMB2 (D23:16)	Receive mask match byte 2
RMMB3 (D15:08)	Receive mask match byte 3
RMMB4 (D07:00)	Receive mask match byte 4

Table 101: Receive Match MASK register bit definition

Control Register C (HDLC)

Control Register C provides the configuration bits required when the serial channel is configured to operate in HDLC mode.

Reminder: DMA is not supported in HDLC mode.



Code (Bit #)	Definition of code	Description
NFLAG (D31:28)	Number of flags between frames (0-15)	0 – Shared flag between frames 1-15– Additional flags 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 no data ready to send (conditioned by the setting of the FLAG*/IDL field).
CRC (D23:22)	CRC Mode	00 – NO CRC 01 – Reserved 10 – 16-bit CCITT CRC-16 11 – 32-bit CCITT CRC-32 Identifies the type of CRC encoding used in the HDLC frames.
RXCRC (D21)	Receive CRC	0 – Do not include CRC in receive data buffer 1 – Place receive CRC in receive data buffer Controls whether the HDLC CRC field is placed into the receive data FIFO. The HDLC CRC field is required to be examined by firmware only, for either debugging purposes or encapsulation protocols. Setting the RXCRC field to 1 causes the HDLC CRC field to be inserted into the receive FIFO.

Table 102: Control Register C (HDLC) bit definition

Code (Bit #)	Definition of code	Description
CHKCRC* (D20)	Check the receive CRC	0 – Discard bad receive frames with incorrect CRC 1 – Allow bad receive frames to be accepted Controls whether 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* (D17)	Transmit CRC	0 – Send calculated CRC before closing flag 1 – Do not send calculated CRC before closing Controls whether 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 (D16)	Send flag or idle between frames	0 – Send flags (7E) between frames 1 – Send idle (FF) between frames 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 least one flag immediately before the start of the HDLC frame and at least one flag immediately after the end of the HDLC frame.

Table 102: Control Register C (HDLC) bit definition

Status Register B (HDLC)

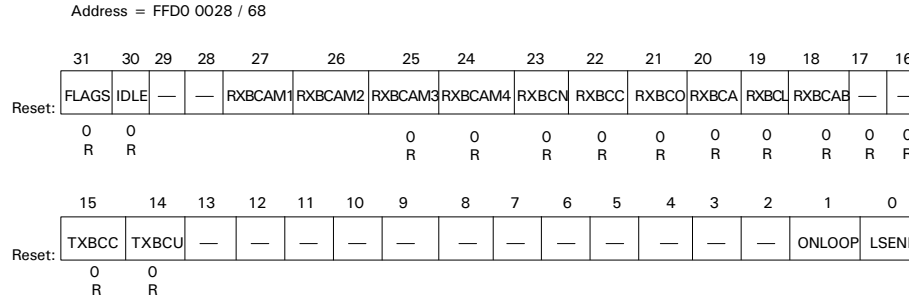
Status Register B provides the status bits required when the serial channel is configured to operate in HDLC mode.

The receive buffer closed (RXBC) status bits are valid only when the RXBC bit is set in Serial Channel 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 valid only when the TXBC bit is set in Serial Channel Status Register A. The transmit buffer closed status bits must be read before the TXBC bit is acknowledged.

Reminder: DMA is not supported in HDLC mode.

Status Register B and bit definitions



Code (Bit #)	Definition of code	Description
FLAGS (D31)	Static receiver flag status	0 – Receiver is active or not receiving flags 1 – Receiver is IDLE receiving flags Set to 1 when the HDLC receiver has detected that the remote transmitter is sending continuous flags.
IDLE (D30)	Static receiver IDLE status	0 – Receiver is active or not receiving IDLE codes 1 – Receiver is IDLE receiving IDLE codes ■ Set to 1 when the receiver is inactive and receiving all "1"s. ■ Set to 0 when the receiver is actively receiving a packet or not receiving the all "1s" condition.

Table 103: Status Register B (HDLC) bit definition

Code (Bit #)	Definition of code	Description
RXBCAM1 (D27)	Receive buffer closed due to address match 1	Indicates that 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 that 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). The RXBCAM1 field is valid only when the RBC field is set in Serial Channel Status Register A.
RXBCAM2 (D26)	Receive buffer closed due to address match 2	Indicates that 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. The RXBCAM2 field is valid only when the RBC field is set in Serial Channel Status Register A.
RXBCAM3 (D25)	Receive buffer closed due to address match 3	Indicates that 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 that 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). The RXBCAM3 field is valid only when the RBC field is set in Serial Channel Status Register A.
RXBCAM4 (D24)	Receive buffer closed due to address match 4	Indicates that 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. The RXBCAM4 field is valid only when the RBC field is set in Serial Channel Status Register A.
RXBCN (D23)	Receive buffer closed NORMAL	Indicates that the current HDLC frame was received without any errors. The RXBCN field is valid only when the RBC field is set in Serial Channel Status Register A.
RXBCC (D22)	Receive buffer closed due to CRC error	Indicates that the current HDLC frame was received with CRC errors. The firmware driver can reject the packet using knowledge provided by RXBCC. The RXBCC field is valid only when the RBC field is set in Serial Channel Status Register A.

Table 103: Status Register B (HDLC) bit definition

Code (Bit #)	Definition of code	Description
RXBCO (D21)	Receive buffer closed due to OVERRUN error	Indicates that the current HDLC frame was received with a receive FIFO overrun condition. An overrun condition indicates that the receive FIFO was not emptied at a fast enough rate. The firmware driver can reject the packet using knowledge provided by RXBCO. The RXBCO field is valid only when the RBC field is set in Serial Channel Status Register A.
RXBCA (D20)	Receive buffer closed due to ALIGNMENT error	Indicates that 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. The RXBCA field is valid only when the RBC field is set in Serial Channel Status Register A.
RXBCL (D19)	Receive buffer closed due to LARGE error	Indicates that the current HDLC frame that was received 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. The RXBCL field is valid only when the RBC field is set in Serial Channel Status Register A.
RXBCAB (D18)	Receive buffer closed due to ABORT	Indicates that 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 "1"s. The firmware driver can reject the packet using knowledge provided by RXBCAB. The RXBCAB field is valid only when the RBC field is set in Serial Channel Status Register A.

Table 103: Status Register B (HDLC) bit definition

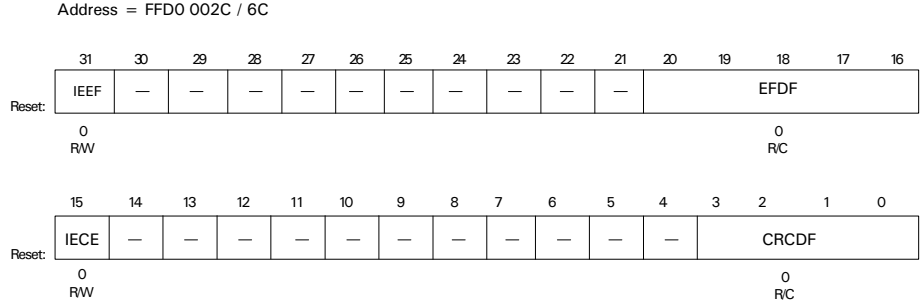
Code (Bit #)	Definition of code	Description
TXBCC (D15)	Transmit buffer closed due to loss of CTS	Indicates that the current HDLC transmit frame was aborted due to a loss of the CTS (clear-to-send) condition. The CTSTX field in Serial Channel Control Register A must be set for this to happen. The firmware driver can use TXBCC to indicate that the HDLC frame was not successfully transmitted. The TXBCC field is valid only when the TBC field is set in Serial Channel Status Register A.
TXBCU (D14)	Transmit buffer closed due to UNDERRUN error	Indicates that the current HDLC transmit frame encountered an underrun condition. The firmware driver can use TXBCU to indicate that the HDLC frame was not successfully transmitted. The TXBCU field is valid only when the TBC field is set in Serial Channel Status Register A.

Table 103: Status Register B (HDLC) bit definition

Status Register C (HDLC)

Status Register C provides error counter registers that are required when the serial channel is configured to operate in HDLC mode.

Status Register C and bit definitions



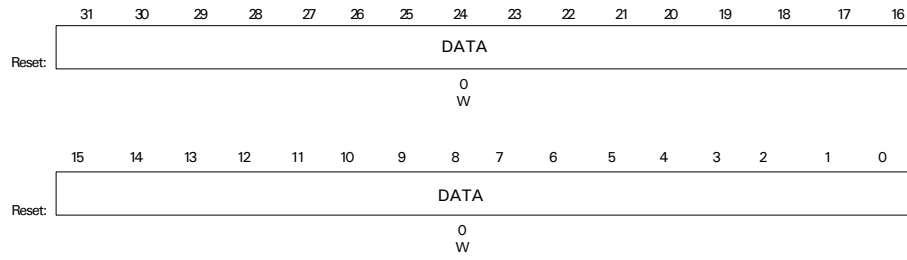
Code (Bit #)	Definition of code	Description
IEEF (D31)	Interrupt enable	Setting this bit causes a receiver-interrupt to occur when the EFDF counter reaches a value of 24.
EFDF (D20:16)	Error free discarded frame counter	Identifies the number of frames discarded due to a failure to match the selected 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 it is read.
IECE (D15)	Interrupt enable	Setting this bit causes a receiver-interrupt to occur when the CRCDF counter reaches a value of 8.
CRCDF (D03:00)	CRC error discarded frame counter	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 it is read.

Table 104: Status Register C (HDLC) bit definition

FIFO Data Register LAST (HDLC)

The FIFO Data Register LAST 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



MIC Controller Module

CHAPTER 12

This chapter applies to the NET+50 chip only.

Important: The previous NET_ARM hardware manuals used the term *ENI* to reference both the ENI module, which contains the IEEE 1284 ports and the GPIO ports, and the ENI interface, which is an address and data bus with its own protocol and timing.
The module that contains all three interfaces is now called the *Multi Interface Controller*, or *MIC.*

The Multi-Interface Controller (MIC) module provides three separate, mutually exclusive interfaces between the NET+50 chip and external devices:

- Four GPIO ports
- Four IEEE 1284 ports
- Four variations of the ENI Interface

Only one mode can be operational at any given time.

MIC controller modes of operation

The MIC controller module is allocated forty NET+50 chip I/O pins. The use of these 40 pins is determined automatically when the MICMODE configuration bits are written in the General Control register. The configuration bits are written either at boot-up with the information on the address lines or at any time with firmware. See Chapter 3, "NET+50 Chip Package" for more information.

Figure 39 provides a high-level overview of the MIC controller module.

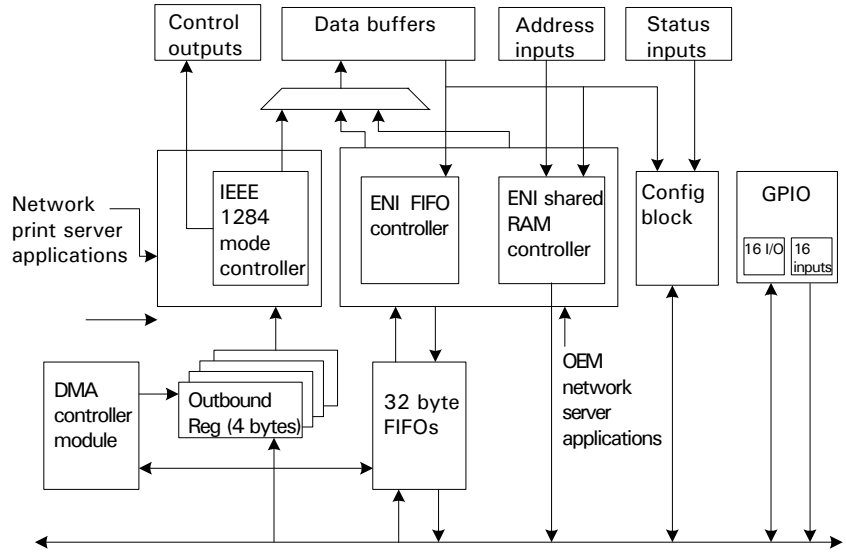


Figure 39: MIC controller high-level block diagram

- **GPIO mode.** Provides two 8-bit general purpose input/output ports and two 8-bit input ports. When the GPIO mode is used, the IEEE 1284 mode and the ENI mode interfaces are disabled.
- **IEEE 1284 mode.** Used in commercial network print server applications. The commercial application provides a bridge between a LAN and up to four external devices using the IEEE 1284 parallel port interface. When the IEEE 1284 mode is used, the GPIO mode and the ENI mode interfaces are disabled.
- **ENI modes.** 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+50 chip and the external system. The FIFO mode interface supports two 32-byte FIFOs, one for each direction streaming FIFO. When any of the ENI modes are used, the GPIO and the IEEE 1284 interface are 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 8K because PA13, PA14, and PA15 are reassigned for DMA handshake signals.

MIC modes of operation

The MIC module can be set to any of these six configurations:

Configuration	Description
GPIO interface	General purpose input/output
IEEE 1284 host port	Four ports
ENI host shared RAM-16 interface	16-bit shared memory only
ENI host shared RAM-8 interface	8-bit shared memory only
ENI host FIFO-16 interface	16-bit register and shared memory
ENI host FIFO-8 interface	8-bit register and shared memory

Table 105: MIC mode settings

The MICMODE field in the General Control register controls the MIC mode of operation. The MICMODE field can be loaded automatically during bootstrap, using or omitting 1K ohm pulldown resistors on address lines A22, A21, and A20. The state of these three address lines is read when power is applied to the system, and written to bits D2:D0 of the General Control register. The configuration can be changed any time after booting by rewriting to bits D2:D0.

Table 106 lists the address line characteristics and D2:D0 values that control each MIC configuration. See the discussions of each register for more information.

Configuration	A22	A21	A20	D2	D1	D0
GPIO	1K pulldown	1K pulldown	1K pulldown	0	0	0
IEEE 1284	1K pulldown	1K pulldown	Floating	0	0	1
Reserved	1K pulldown	Floating	1K pulldown	0	1	0
Reserved	1K pulldown	Floating	Floating	0	1	1
ENI 16-bit shared RAM only	Floating	1K pulldown	1K pulldown	1	0	0
ENI 8-bit shared RAM only	Floating	1K pulldown	Floating	1	0	1
ENI FIFO, 16-bit with shared RAM	Floating	Floating	1K pulldown	1	1	0
ENI FIFO, 8-bit with shared RAM	Floating	Floating	Floating	1	1	1

Table 106: MIC configurations

MIC controller configuration

The MIC controller module has a block of configuration registers, as shown:

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

Table 107: MIC controller configuration registers

Address	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
FFA0 0040	GPIO PORT D register
FFA0 0044	GPIO PORT D register
FFA0 0048	GPIO PORT F register
FFA0 004C	GPIO PORT G register
FFA0 0050	GPIO PORT H register

Table 107: MIC controller configuration registers

MIC module interrupts

The MIC module provides four status indicators 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 example, IEEE 1284) for which the MIC module is configured.

GEN interrupt	GPIO	IEEE 1284 mode	ENI host mode
MIC port 1	D	Port 1	FIFO receiver
MIC port 2	F	Port 2	FIFO transmitter
MIC port 3	G	Port 3	ENI interrupt
MIC port 4	H	Port 4	ENI bus error

Table 108: MIC interrupt sources

MIC module hardware initialization

The MICMODE bits (in the General Control register) and the MIC Control register bits are implementation-dependent. Since external hardware circuitry relies on

this configuration during reset, the MIC module allows external 1K ohm pulldown resistors on the address lines to configure their initial values. The system bus address bits are used to configure the values during a powerup reset.

The NET+50 chip provides internal pullup resistors on all address signals. 1K ohm external pulldown resistors can be used to configure the MICMODE (FFA0 0000) and MICCTL (FFA0 0030) register bits. This list identifies which ADDR bits control which functions:

Address bit	Function	Setting
ADDR[22:20]:	MICMODE[2:0]	Configuration bits
ADDR[07]:	ENI control D9	PSIO* (0 = PSIO, 1 = Normal)
ADDR[06]:	ENI control D6	WR_OC
ADDR[05]:	ENI control D5	DINT2*
ADDR[04]:	ENI control D4	I_OC
ADDR[03]:	ENI control D3	DMAE*
ADDR[02]:	Reserved	
ADDR[01]:	ENI control D1	EPACK*
ADDR[00]:	ENI control D0	PULINT*

Table 109: Address bit functionality

Note: The inverted PSIO address bit (ADDR7) is loaded into the PSIO configuration bit in the MIC Control register. In the MIC Control register, 0=Normal and 1=PSIO.

GPIO mode

The GPIO mode provides four 8-bit general purpose input/output ports: PORTD, PORTF, PORTG, and PORTH.

GPIO mode is configured when the MICMODE field in the MIC General Control register is set to 0. MICMODE can be defaulted to 0 during bootstrap by providing 1K ohm pulldown resistors on NET+50 pins A22, A21, and A20. (See "MIC module hardware initialization" on page 333 for more information.)

GPIO mode can be used instead of IEEE 1284 mode and ENI mode. GPIO mode is useful when parallel ports, shared RAM, or FIFO are not required. GPIO allows 32 of the 40 pins assigned to the MIC interface to be used for general purpose I/O.

Table 110 shows the PQFP and BGA pin assignments for PORTD, PORTF, PORTG, and PORTH.

	PORTD		PORTF		PORTG		PORTH	
Port	PQFP	BGA	PQFP	BGA	PQFP	BGA	PQFP	BGA
7	207	T15	35	G16	34	H17	20	K15
6	206	U15	10	P16	31	H16	19	L16
5	205	U14	11	N16	30	J17	18	M17
4	204	T14	12	M15	25	J14	17	L14
3	203	R15	28	H15	24	J15	16	L15
2	202	U13	29	H14	23	K16	15	M16
1	201	P14	32	G15	22	L17	14	N17
0	200	R14	33	G14	21	K14	13	M14

Table 110: PORTD/F/G/H PQFP/BGA pin assignments

These pins comprise four 8-bit GPIO ports:

- 16 GPIO pins (two 8-bit ports) can be configured for input, output, or level-sensitive interrupt.
- The other 16 GPIO pins (two 8-bit ports) can be configured as inputs or level-sensitive interrupts.

GPIO function configurations

The GPIO ports can be configured to provide any of four functions per pin: input, output, low-level interrupt, and high-level interrupt. The ports are individually configured and do not have to be set to the same function.

The `MODE` and `DIR` bits work together to set the GPIO port configuration. The GPIO port allows four configurations:

- Data in (input)

- Data out (output)
- Low level interrupt input (low interrupt)
- High level interrupt input (high interrupt)

Table 111 shows the functionality for PORTD. GPIO PORTD requires two registers to configure the functionality of each of the PORTD GPIO pins:

- Register `0xFFA00040` is used for input data and interrupts.
- Register `0xFFA00044` produces the output data.

FFA00040 MODE	FFA00040 DIR	FFA00044 MODE	FFA00044 DIR	PORTD function
0	0	0	0	Data input to register `FFA00040`
0	1	0	1	Output from register `FFA00044`
1	0	0	0	Low interrupt to register `FFA00040`
1	1	0	0	High interrupt to register `FFA00040`

Table 111: PORTD configuration

Table 112 shows the functionality for PORTF/G/H. Each port requires only one register to configure functionality.

Mode	Dir	PORTF FFA00048	PORTG FFA0004C	PORTH FFA00050
0	0	Input	Input	Input
0	1	Output		
1	0	Low interrupt	Low interrupt	Low interrupt
1	1	High interrupt		

Table 112: PORTF/G/H configuration

Functions

PORTF and PORTD allow all four functions. PORTG and PORTH are limited to data input and low-level interrupts.

- When the port is configured as an input, the system reads the data field at the address associated with the port to determine the logic level of the signal connected to the pin.
- When the port is configured as an output, the system writes to the data field at the address associated with the port to set the logic level of the pin's output.
- When the port is configured as a low-level or high-level interrupt, the system reads the data field at the address associated with the port to determine which of the eight pins caused the interrupt.

  The data bit is set to high to indicate a valid interrupt level on the pin.

Each port is assigned to a bit in the Interrupt Status register. Any of the eight pins in the port can set a bit in the Interrupt Status register. The following table shows how the bits are assigned:

GEN Interrupt Status Bit register	GPIO port
MIC port 1	D
MIC port 2	F
MIC port 3	G
MIC port 4	H

Examples using PORTF

Here is an example of an interrupt configuration and routine using PORTF. This example also shows how to configure the port for all four functions by setting two port pins to each function.

Example 1: Conditions

- PORTF7 and PORTF6 are inputs. A low signal is connected to PORTF7 and a high signal is connected to PORTF6.
- PORTF5 and PORTF4 are outputs. PORTF5 is set high and PORTF4 is set low.

- PORTF3 and PORTF2 are low-level interrupts. A high signal is connected to both pins, to start.
- PORTF1 and PORTF0 are high-level interrupts. A low signal is connected to both pins, to start.

Address	Action
0xFFA00000=0x000010000	Set the MIC mode to GPIO using the MIC mode General Control register. The MICDIAG bit (D12) must be set to 1 to change the MIC mode.
0xFFA00048=0x0F330020	Set PORTF as described in the "Example 1: Conditions" section. **Note:** If the port was read at this time, 0xFFA00048 would read 0x0F330060 because PORTF7 and PORTF6 are inputs and PORTF6 is high.
0xFFB00030=0x00200000	Set the Interrupt Enable register to allow an interrupt to be caused by PORTF. See "Interrupt generation and control" on page 93 for more information. At this point, the configuration is complete.

Example 2: Conditions

- The signal connected to PORTF2 goes low.
- The signal connected PORT0 goes high.
- Either event can cause an interrupt; this example shows both a low-level and high-level interrupt. This event sets the IRQ signal to the ARM. The ARM issues an interrupt exception vector to address 0x18. At address 0x18, there should be a jump instruction to the interrupt service routine.

Address	Action
0xFFB00038=0x00200000	The Interrupt Status register is read to determine that the interrupt is caused by PORTF. See "Interrupt generation and control" on page 93 Note that the interrupt is not latched. If the two signals that caused the interrupt revert to their original state very quickly, 0xFFB00038 reads 0.

Address	Action
0xFFA00048=0x0F330025	The PORTF GPIO register is read to determine which of the four signals configured as interrupts caused the interrupt. In this example, the interrupt is caused by either PORTF2 or PORTF0. Note that the interrupt is not latched. If the two signals that caused the interrupt revert to their original state very quickly, 0xFFA00048 reads 0x0F330020.

PORTD register

PORTD can be used for general purpose I/O or level-sensitive interrupt inputs. Each of the eight PORTD GPIO pins can be individually programmed to general purpose input, general purpose output, active low interrupt, or active high interrupt.

- See Table 110: "PORTD/F/G/H PQFP/BGA pin assignments" on page 335 for PORTD GPIO bit/pin assignments.
- See Table 111: "PORTD configuration" on page 336 for PORTD configuration functionality.

General purpose I/O

- When both MODE bits (0xFFA00040 and 0xFFA00044) are set to 0, the respective PORTD bit is configured for either input or output.

  The respective bits in the DIR field are used to control the direction of the GPIO data.

  – A DIR setting of 0 in both registers configures the input mode.
  – A DIR setting of 1 in both registers configures the output mode.
- When MODE and DIR are set to 0, the PORTD bit is configured in input mode. The NET+50 processor reads the current logic value on the PORTD pin by reading the state of the respective bit in the DATA field of register 0xFFA00040.
- When the MODE bit is set to 0 and the DIR bit is set to 1, the PORTD bit is configured in output mode. The NET+50 processor sets the logic value on the PORTD pin by setting the state of the respective bit in the

DATA field of register 0xFFA00044. Reading the DATA field when configured in output mode returns the current setting of the DATA field.

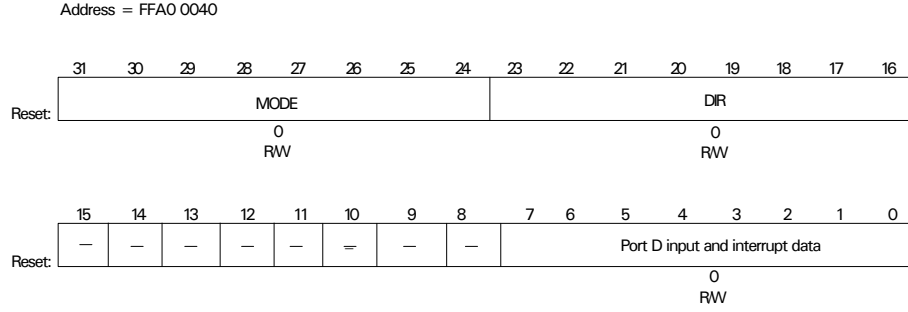
Interrupt Mode

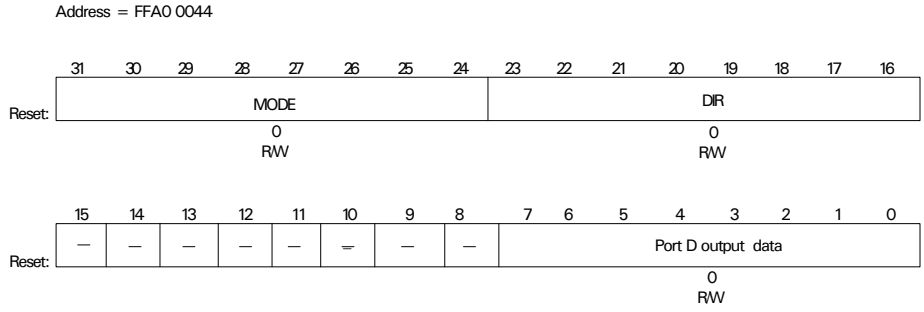
When both MODE bits are set to 1, the PORTD I/O pin is used for interrupt input. In this mode, the DIR register bits define the level sensitive state of the interrupt input (0 for active low level or 1 for active high level), and the PORTD data register provides the interrupt status. A 1 in this register indicates an active interrupt. To clear the interrupt condition, the external signal must change to the inactive state.

PORTD asserts an interrupt to the NET+50 interrupt controller using the MIC Port 1 connection. The MIC Port 1 interrupt is active when any one of the PORTD interrupts is active, and remains active until the external signal changes to the inactive state.

PORTD registers and bit definitions

- Each bit in the MODE fields corresponds to one of the NET+50 PORTD bits. D31 controls PORTD7, D30 controls PORTD6, and so on.
- Each bit in the DIR field corresponds to one of the NET+50 PORTD bits. D23 controls PORTD7, D22 controls PORTD6, and so on.
- Each bit in the DATA field corresponds to one of the NET+50 PORTD bits. D7 controls PORTFD, D6 controls PORTD6, and so on.





Code (Bit #)	Description
MODE	Used to configure each PORTD pin individually to operate in either GPIO mode or interrupt mode. Setting the MODE bit to 0 selects GPIO mode; setting the bit to 1 selects interrupt mode.
DIR	When the MODE bit in both registers is set to 0, the DIR bit is used to configure the PORTD pin to operate in either input mode or output mode. Setting the DIR bit to 0 selects input mode; setting the bit to 1 selects output mode. When the MODE bit in both registers is set to 1, the DIR bit is used to configure the PORTD pin to be either an active low interrupt input or active high interrupt. Setting the DIR bit to 0 selects active low interrupt mode; setting the bit to 1 selects active high interrupt mode.
DATA	When the MODE bit in both registers is set to 0 for GPIO mode, the DATA bit from register 0xFFA00040 provides the current state of the NET+50 GPIO signal (regardless of its configuration mode). Writing to the DATA field in register 0xFFA00044 defines the current state of the NET+50 GPIO signal when the signal is defined to operate in GPIO output mode. Writing a DATA bit when configured in GPIO input mode or interrupt mode has no effect. When the MODE bit is set to 1 for interrupt mode, the corresponding DATA bits are used to indicate a pending interrupt condition. A pending interrupt is identified with a 1 in the DATA field of register 0xFFA00040. The external signal input must be restored to its inactive state to remove the interrupt condition. Setting the MODE and DIR bits to 0 can also disable the interrupt.

Table 113: GPIO PORTD register bit definition

PORTF register

The PORTF I/O register can be used whenever the MIC module is configured to operate in GPIO mode. PORTF can be used for general purpose I/O or level-sensitive interrupt inputs.

Each of the eight PORTF GPIO pins can be individually programmed to general purpose input, general purpose output, active low interrupt, or active high interrupt.

- See Table 110: "PORTD/F/G/H PQFP/BGA pin assignments" on page 335 for PORTF GPIO bit/pin assignments.
- See Table 112: "PORTF/G/H configuration" on page 336 for PORTF configuration functionality.

General purpose I/O mode

- When a MODE bit is set to 0, the respective PORTF bit is configured for general purpose I/O (GPIO).
- When a PORTF MODE bit is set to 0, the respective bit in the DIR field controls the direction of the GPIO configuration:
  - A DIR setting of 0 configures input mode.
  - A DIR setting of 1 configures output mode.
- When both MODE and DIR are set to 0, the PORTF bit is configured in input mode. The NET+50 processor can read the current logic value on the PORTF pin 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 PORTF bit is configured in output mode. The NET+50 processor sets the current logic value on the PORTF pin by setting the state of the respective bit in the DATA field. Reading the DATA field when configured in output mode returns the current setting of the DATA field.

Interrupt mode

When the MODE bit position is configured as 1, the PORTF I/O pin can be used for interrupt input. In this mode, the DIR register bit defines the level-sensitive state of the interrupt input (0 for active low level or 1 for active high level), and the

PORTF data register provides the interrupt status. A 1 in this register indicates an active interrupt. To clear the interrupt condition, the external signal must change to the inactive state.

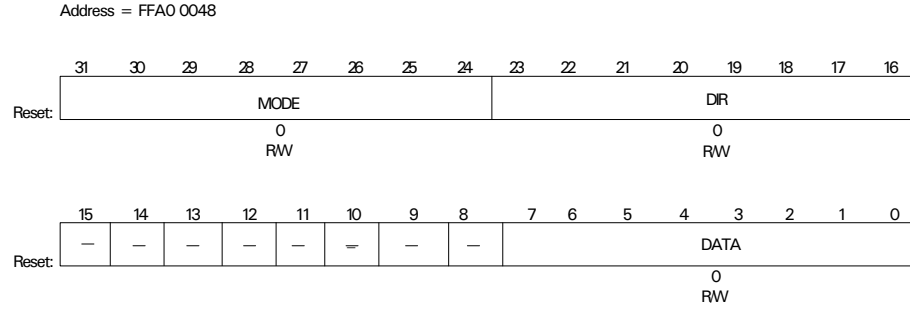
PORTF asserts an interrupt to the NET+50 interrupt controller using the MIC Port 2 connection. The MIC Port 2 interrupt is active when any one of the PORTF interrupts is active, and remains active until the external signal changes to the inactive state.

NET + 50 pin		GPIO mode, FMODE = 0		Interrupt mode, FMODE = 1	
PQFP	BGA	FDIR = 0	FDIR = 1	FDIR = 0	FDIR = 1
35	G16	D7 IN	D7 OUT	FI7-0	FI7-1
10	P16	D6 IN	D6 OUT	FI6-0	FI6-1
11	N16	D5 IN	D5 OUT	FI5-0	FI5-1
12	M15	D4 IN	D4 OUT	FI4-0	FI4-1
28	H15	D3 IN	D3 OUT	FI3-0	FI3-1
29	H14	D2 IN	D2 OUT	FI2-0	FI2-1
32	G15	D1 IN	D1 OUT	FI1-0	FI1-1
33	G14	D0 IN	D0 OUT	FI0-0	FI0-1

Table 114: PORTF pin/mode/dir configuration

PORTF register and bit definitions

- Each bit in the MODE field corresponds to one of the NET+50 PORTF bits. D31 controls PORTF7, D30 controls PORTF6, and so on.
- Each bit in the DIR field corresponds to one of the NET+50 PORTF bits. D23 controls PORTF7, D22 controls PORTF6, and so on.
- Each bit in the DATA field corresponds to one of the NET+50 PORTF bits. D07 controls PORTF7, D06 controls PORTF6, and so on.



Code (Bit #)	Description
MODE (D31:24)	Used to configure each of the PORTF pins to operate in either GPIO mode or interrupt mode. Setting the MODE bit to 0 selects GPIO mode; setting the bit to 1 selects interrupt mode.
DIR (D23:16)	When the MODE bit is set to 0, the DIR bit is used to configure the PORTF pin to operate in either input mode or output mode. Setting the DIR bit to 0 selects input mode; setting the bit to 1 selects output mode. When the MODE bit is set to 1, the DIR bit is used to configure the PORTF pin to be either active low interrupt input or active high interrupt input. Setting the DIR bit to 0 selects active low interrupt mode; setting the bit to 1 selects active high interrupt mode.
DATA (D07:00)	When the MODE bit is set to 0 for GPIO mode, the DATA bit provides the current state of the NET+50 GPIO signal (regardless of its configuration mode). Writing the DATA field defines the current state of the NET+50 GPIO signal when the signal is defined to operate in GPIO output mode. Writing a DATA bit when configured in GPIO input mode or interrupt mode has no effect. When the MODE bit is set to 1 for interrupt mode, the corresponding DATA bits are used to indicate a pending interrupt condition. A pending interrupt is identified with a 1 in the DATA field. The external signal input must be restored to its inactive state to remove the interrupt condition. Setting the MODE and DIR bits to 0 can also disable the interrupt.

Table 115: GPIO PORTF register bit definition

PORTG register

The PORTG I/O register can be used whenever the MIC module is configured to operate in GPIO mode. PORTG can be used for general purpose input or level-sensitive interrupt inputs.

Each of the eight PORTD GPIO pins can be individually programmed to general purpose input, active low interrupt, or active high interrupt.

- See Table 110: "PORTD/F/G/H PQFP/BGA pin assignments" on page 335 for PORTG GPIO bit/pin assignments.
- See Table 112: "PORTF/G/H configuration" on page 336 for PORTG configuration functionality.

General purpose I/O

- When a MODE bit is set to 0, the respective PORTG bit is configured for general purpose input.
- When a PORTG MODE bit is set to 0, a DIR bit of 0 configures input mode. PORTG does not support GPIO output mode.
- When both MODE and DIR are set to 0, the PORTG pin is configured in input mode. The NET+50 processor can read the current logic value on the PORTG pin by reading the state of the respective bit in the DATA field. PORTG does not support GPIO output mode.

Interrupt mode

When the MODE bit position is configured as 1, the PORTG I/O pin can be used for interrupt input. In this mode, the DIR register should be set to 0 for active low level, and the PORTG data register provides the interrupt status. A 1 in this register indicates an active interrupt. To clear the interrupt condition, the external signal must change to the inactive state.

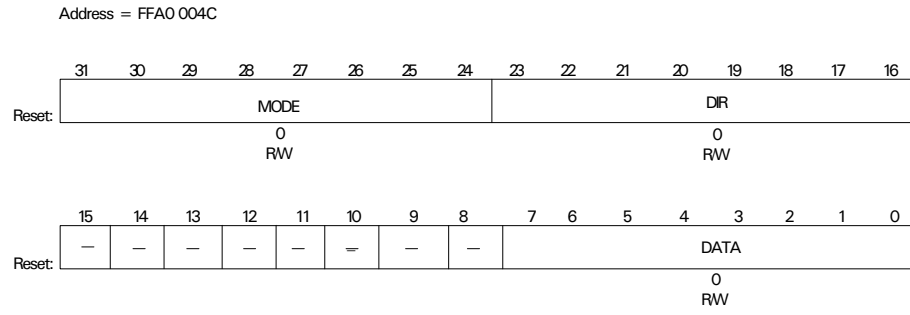
PORTG asserts an interrupt to the NET+50 interrupt controller using the MIC Port 2 connection. The MIC Port 2 interrupt is active when any one of the PORTG interrupts is active and remains active until the external signal changes to the high state.

NET + 50 pin		GPIO mode	Interrupt mode
PQFP	BGA	GMODE = 0 GDIR = 0	GMODE = 1 GDIR = 0
34	H17	D7 IN	GI7-0
31	H16	D6 IN	GI6-0
30	J17	D5 IN	GI5-0
25	J14	D4 IN	GI4-0
24	J15	D3 IN	GI3-0
23	K16	D2 IN	GI2-0
22	L17	D1 IN	GI1-0
21	K14	D0 IN	GI0-0

Table 116: PORTG pin/mode/dir configuration

PORTG register and bit definitions

- Each bit in the MODE field corresponds to one of the NET+50 PORTG bits. D31 controls PORTG7, D30 controls PORTG6, and so on.
- Each bit in the DATA field corresponds to one of the NET+50 PORTG bits. D07 controls PORTG7, D06 controls PORTG6, and so on.



Code (Bit #)	Description
MODE (D31:24)	Used to individually configure each of the PORTG pins to operate in either GPIO mode or interrupt mode. Setting the MODE bit to 0 selects GPIO mode; setting the bit to 1 selects interrupt mode.
DIR (D23:16)	Should always be set to 0.
DATA (D07:00)	When the MODE bit is set to 0 for GPIO mode, the DATA bit provides the current state of the NET+50 GPIO signal (regardless of its configuration mode). Writing a DATA bit when configured in GPIO input mode or interrupt mode has no effect. When the MODE bit is set to 1 for interrupt mode, the corresponding DATA bits are used to indicate a pending interrupt condition. A pending interrupt is identified with a 1 in the DATA field. The external signal input must be restored to its inactive state to remove the interrupt condition. Setting the MODE bit to 0 can also disable the interrupt.

Table 117: GPIO PORTG register bit definition

PORTH register

The PORTH I/O register can be used whenever the MIC module is configured to operate in GPIO mode. PORTH can be used for general purpose input or level-sensitive interrupt inputs.

Each of the eight PORTD GPIO pins can be individually programmed to general purpose input, active low interrupt, or active high interrupt.

- See Table 110: "PORTD/F/G/H PQFP/BGA pin assignments" on page 335 for PORTH GPIO bit/pin assignments.
- See Table 112: "PORTF/G/H configuration" on page 336 for PORTH configuration functionality.

General purpose I/O mode

- When a MODE bit is set to 0, the respective PORTH bit is configured for general purpose input.

- When a PORTH MODE bit is set to 0, a DIR bit of 0 configures input mode. PORTH does not support GPIO output mode.
- When both MODE and DIR are set to 0, the PORTH bit is configured in input mode. The NET+50 processor can read the current logic value on the PORTH bit by reading the state of the respective bit in the DATA field. PORTH does not support GPIO output mode.

Interrupt mode

When the MODE bit position is configured as 1, the PORTH I/O pin can be used for interrupt input. In this mode, the DIR register bit should be set to 0 for active low level. A MODE register bit value of 1 automatically enables the interrupt, and the PORTH data register provides the interrupt status. A 1 in this register indicates an active interrupt. To clear the interrupt condition, the external signal must change to the inactive state.

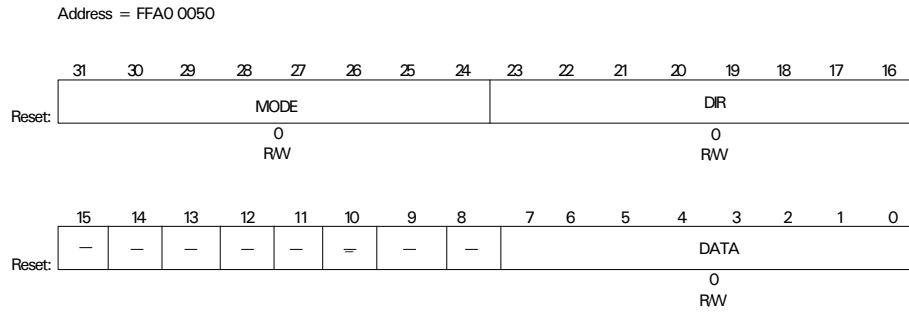
PORTH asserts an interrupt to the NET+50 interrupt controller using the MIC Port 2 connection. The MIC Port 2 interrupt is active when any one of the PORTH interrupts is active and remains active until the external signal changes to the high state.

NET + 50 pin		GPIO mode	Interrupt mode
PQFP	BGA	HMODE = 0 HDIR = 0	HMODE = 1 HDIR = 0
20	K15	D7 IN	GI7-0
19	L16	D6 IN	GI6-0
18	M17	D5 IN	GI5-0
17	L14	D4 IN	GI4-0
16	L15	D3 IN	GI3-0
15	M16	D2 IN	GI2-0
14	N17	D1 IN	GI1-0
13	M14	D0 IN	GI0-0

Table 118: PORTH pin/mode/dir configuration

PORTH register and bit definitions

- Each bit in the MODE field corresponds to one of the NET+50 PORTH bits. D31 controls PORTH7, D30 controls PORTH6, and so on.
- Each bit in the DIR field corresponds to one of the NET+50 PORTH bits. D23 controls PORTH7, D22 controls PORTH6, and so on.
- Each bit in the DATA field corresponds to one of the NET+50 PORTH bits. D07 controls PORTH7, D06 controls PORTH, and so on.



Code (Bit #)	Description
MODE (D31:24)	Used to individually configure each of the PORTH pins to operate in either GPIO mode or interrupt mode. Setting the MODE bit to 0 selects GPIO mode; setting the bit to 1 selects interrupt mode.
DIR (D23:16)	When the MODE bit is set to 0, the DIR bit is used to configure the PORTH pin. Setting the DIR bit to 0 selects input mode. When the MODE bit is set to 1, the DIR bit is used to configure the PORTH pin to be an active low interrupt input. Setting the DIR bit to 0 selects active low interrupt mode.

Table 119: GPIO PORTH register bit definition

Code (Bit #)	Description
DATA (D07:00)	When the MODE bit is set to 0 for GPIO mode, the DATA bit provides the current state of the NET+50 GPIO signal (regardless of its configuration mode). Note that writing a DATA bit when configured in GPIO input mode or interrupt mode has no effect. When the MODE bit is set to 1 for interrupt mode, the corresponding DATA bits are used to indicate a pending interrupt condition. A pending interrupt is identified with a 1 in the DATA field. The external signal input must be restored to its inactive state to remove the interrupt condition. Setting the MODE and DIR bits to 0 can also disable the interrupt.

Table 119: GPIO PORTH register bit definition

IEEE 1284 host interface (4-port) module

When the MIC controller is configured in IEEE 1284 mode, the NET+50 chip supports four IEEE 1284 parallel port interfaces. The NET+50 chip uses external hardware (latching transceivers) to minimize the pin count requirements. All IEEE 1284 output signals are shared between ports using a simple multiplexing scheme. The NET+50 chip provides unique CLK signals to load the shared output data into a specific 1284 port driving register.

The NET+50 IEEE 1284 interface can be used only to provide the host side of the IEEE 1284 protocol. The NET+50 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+50 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 40 shows the external components required to implement this scheme.

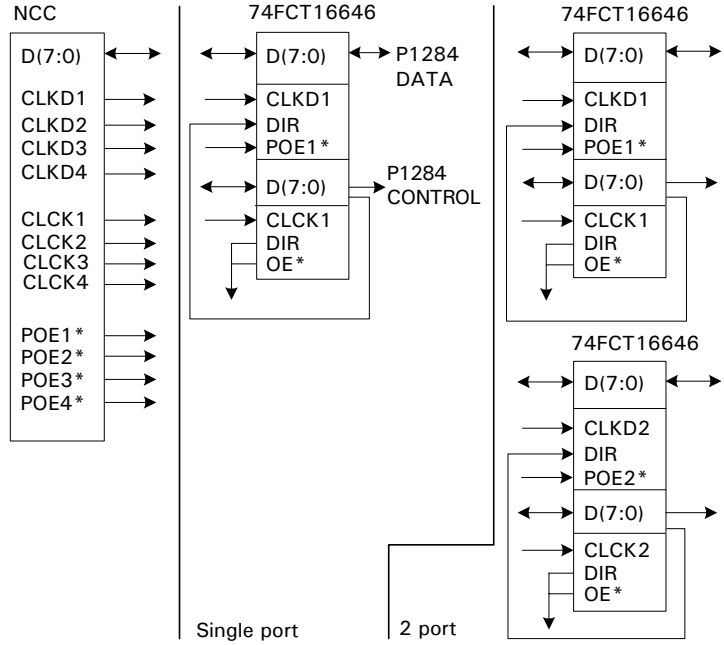


Figure 40: IEEE 1284 external transceivers

The IEEE 1284 data and control signals are multiplexed using a single 8-bit data bus. Table 120 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 120: IEEE 1284 data bus assignments

IEEE 1284 Signals

- **STROBE*, AUTOFD*, INIT*, and HSELECT*.** Four control outputs for the IEEE 1284 interface: These signals are driven through a parallel port cable.
- **PDIR.** 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).
- **PIO.** Controlled by firmware. Its state is directly controlled by the PIO bit in the IEEE 1284 Control register.
- **LOOPBACK.** 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 41 provides a simple view of the IEEE 1284 controller hardware.

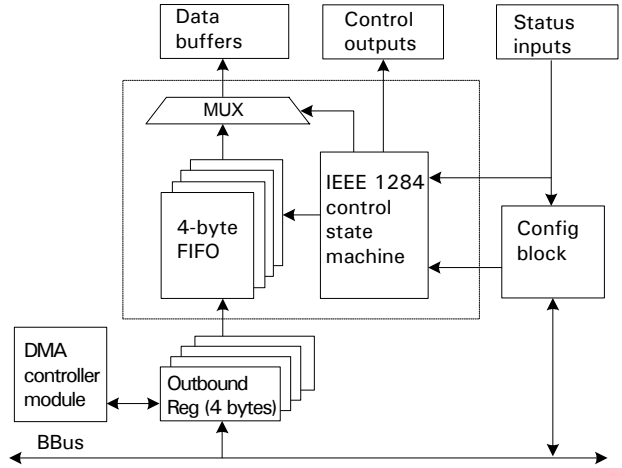


Figure 41: IEEE 1284 host controller block diagram

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+50 chip outputs. External latches are used to store the output signals.

The MIC 1284 controller interfaces with the BBus using 4-byte outbound registers and 4-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.

IEEE 1284 signal cross reference

Table 121 is useful when cross-referencing the names between the IEEE 1284 specification and the bit names in the NET+50 chip architecture. The input/output column defines the direction with respect to the NET+50 chip 1284 host interface.

NET+50 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 121: 1284 signal cross reference

IEEE 1284 port multiplexing

The NET+50 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 eight PCLK signals is unique, and two signals are never activated at the same time.

See "1284 Port Multiplexing timing," beginning on page 470, for timing information and an illustration of the relationship between the PCLK and PDATA signals.

IEEE 1284 mode configuration

Table 122 identifies the modes for which each 1284 port can be configured. The configuration mode is provided using configuration bits in the 1284 Port Control

registers. The operation mode is determined by the results of the IEEE 1284 negotiation process, which must be done in firmware.

The NET+50 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 122: IEEE 1284 mode configuration

IEEE negotiation

The NET+50 chip does not offer any hardware support for IEEE negotiation. Negotiation is handled using the manual mode version of forward compatibility mode (see "IEEE 1284 forward compatibility mode" on page 355). Negotiation must be executed by manually manipulating the IEEE 1284 control signals according to the IEEE 1284 specification.

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 two 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:

- Data setup time
- Port strobe time
- Data hold time

The AUTOFD*, HSELECT*, and INIT* outputs are statically-driven based on 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 on 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.

See "1284 Compatibility mode timing," beginning on page 471, for information about the relative time between STROBE*, DATA, and BUSY when operating in automatic forward compatibility mode. *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. Some jitter is introduced in the STROBE widths because of the port multiplexing.

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.

See Table 154: "1284 Compatibility SLOW mode timing (FAST = 0)" on page 471 for compatibility SLOW mode timing values.

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.

See Table 155: "1284 Compatibility FAST mode timing (FAST = 1)" on page 472 for compatibility FAST mode timing values.

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+50 chip hardware provides no special support for nibble mode; nibble mode must be handled by firmware alone.

Figure 42 shows 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 made up of the BUSY, PE, PSELECT, and FAULT* inputs. The steps involved are listed after the figure.

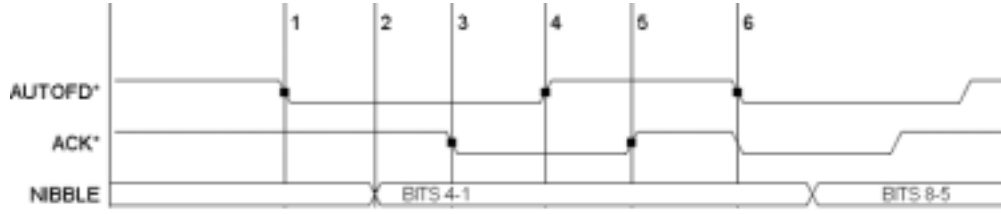


Figure 42: Nibble mode cycle

1. The host signals the ability to take data by asserting HostBusy low (AUTOFD*).
2. The peripheral responds by placing the first nibble on the status lines.
3. The peripheral signals a valid nibble by asserting PtrClk low (ACK*).

4 The host sets HostBusy high to indicate that it has received the nibble and is not yet ready for another nibble.

5 The peripheral sets PtrClk high to acknowledge host.

6 States 1 through 5 repeat for the second nibble.

IEEE 1284 byte mode

Use the byte mode cycle as a second method for obtaining reverse channel data for printers and peripherals. The byte mode cycle offers an advantage 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 43 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.

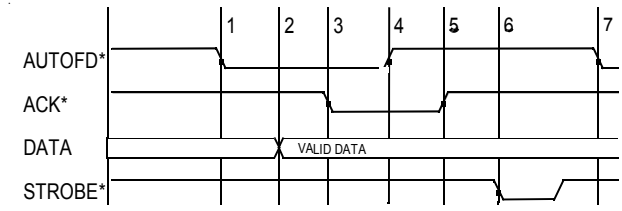


Figure 43: Byte mode cycle

- 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.

IEEE 1284 forward ECP mode

When configured to operate in forward ECP mode, each 1284 port can operate in one of two 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:

- Data Setup Time
- Port Strobe Time
- Data Hold Time.

The AUTOFD*, HSELECT*, and INIT* outputs are statically-driven based on the associative bit settings in the Port Control registers.

The ACK*, BUSY, PE, PSELECT, and FAULT* bits are manually read using 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 on 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.

See "Forward ECP mode timing," beginning on page 473, for information about the relative time between STROBE*, DATA, and BUSY when operating in forward ECP mode.*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. Some jitter is introduced in the STROBE widths because of the port multiplexing.

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.

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.

See Table 156: "1284 SLOW Forward ECP mode timing (FAST = 0)" on page 473 for timing values.

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.

See Table 157: "1284 FAST Forward ECP mode timing (FAST = 1)" on page 473 for timing values.

IEEE 1284 reverse ECP mode

The 1284 reverse ECP mode is a fast and effective way to obtain reverse channel data from a 1284 peripheral. The 1284 reverse ECP mode 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, 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 on 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 using the Port Control register.

When the port is configured for reverse ECP mode, the hardware automatically manipulates the AUTOFD* signal based on 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, an inbound reverse channel byte is considered DATA. The DATA byte is moved to the Channel Data register. When the BUSY input is sampled low, 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 that 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 44 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.

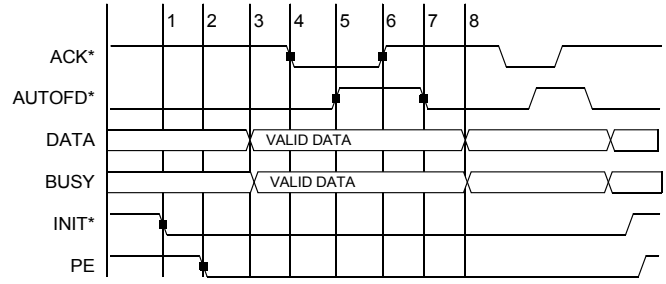


Figure 44: ECP reverse mode timing

1. The host requests a reverse channel transfer by setting INIT* low.
2. The peripheral signals that it is 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.

IEEE 1284 EPP mode

The EPP protocol provides 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+50 chip bus client 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+50 chip bus client 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+50 chip. See "IEEE 1284 Channel Data registers" on page 372 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+50 chip bus client, 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 or destination address (depending on the direction).

EPP data write cycle

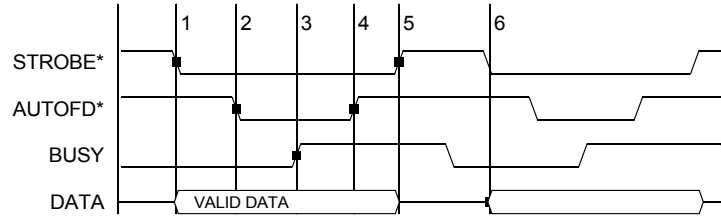


Figure 45: EPP data write cycle

1. The NET+50 chip bus client 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 through 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. Steps 1 through 5 are repeated as necessary.

EPP data read cycle

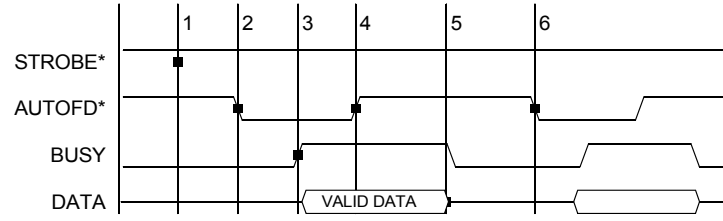


Figure 46: EPP data read cycle

1. The NET+50 chip bus client 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 through 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 Steps 1 through 5 are repeated as necessary.

EPP address write cycle

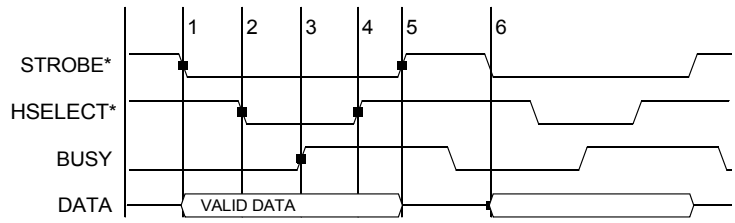


Figure 47: EPP address write cycle

1 The NET+50 chip bus client 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 through 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 Steps 1 through 5 are repeated as necessary.

EPP address read cycle

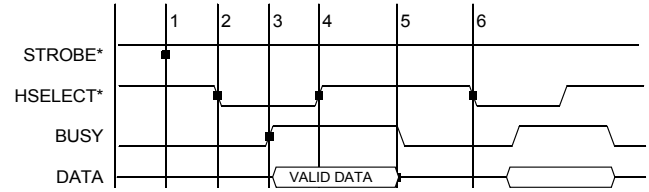


Figure 48: EPP address read cycle

1. The NET+50 chip bus client 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 because BUSY is inactive. The host does not assert HSELECT* until BUSY is inactive.
3. The host waits for the acknowledgment from the peripheral through 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.
5. The peripheral removes BUSY and tri-states the data bus.
6. Steps 1 through 5 are repeated as necessary.

IEEE 1284 Configuration registers

Table 123 shows the IEEE 1284 configuration registers.

Address	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

Table 123: IEEE 1284 registers

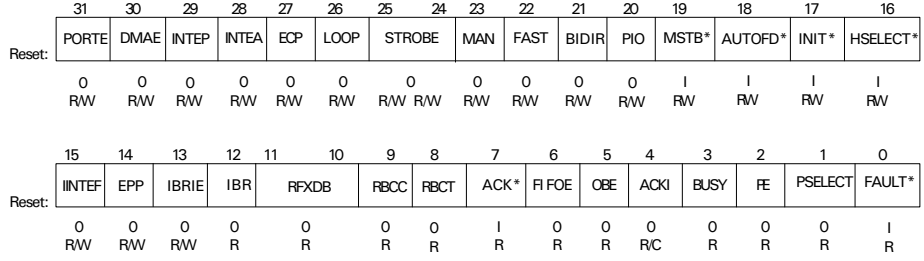
Address	Register
FFA0 0024	IEEE 1284 Channel 2 Data register
FFA0 0028	IEEE 1284 Channel 3 Data register
FFA0 002C	IEEE 1284 Channel 4 Data register

Table 123: IEEE 1284 registers

IEEE 1284 Port Control registers and bit definitions

The NET+50 chip supports four 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 this case, the bit is active low. All control bits are set to their respective inactive state on reset.

Note: The active edge of ACK Interrupt (ACKI) is active when the appropriate register bit is set low.



Code (Bit #)	Definition of code	Description
PORTE (D31)	1284 port enable	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 (D30)	1284 port DMA request enable	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.

Table 124: IEEE 1284 Port Control register bit definition

Code (Bit #)	Definition of code	Description
INTEP (D29)	1284 port outbound interrupt request enable	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 causes an interrupt to be generated when the OBE status bit is 1. The INTEP bit should be set only when operating the 1284 port in interrupt service mode instead of DMA mode.
INTEA (D28)	1284 port acknowledge transition interrupt request enable	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 causes an interrupt to be generated when the ACKI status bit is 1.
ECP (D27)	1284 ECP mode of operation	Must be set to 1 to allow the 1284 port to operate in the ECP mode of operation. See "IEEE 1284 forward ECP mode" on page 359 for details about operating the 1284 port in ECP mode
LOOP (D26)	1284 external loopback	When the LOOP bit is set to 1, the 1284 port is configured to operate in external loopback mode (see "IEEE 1284 external loopback mode" on page 374). When LOOP is set to 1, the LOOP strobe signal is asserted instead of the STROBE* signal in the external control latch (see Table 120: "IEEE 1284 data bus assignments" on page 351). 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 appears on the LOOPBACK signal in the external control latch.
STROBE (D25:24)	1284 manual strobe width configuration	■ 00 – 0.5 us ■ 01 – 1.0 us ■ 10 – 5.0 us ■ 11 – 10.0 us Defines the width of the IEEE 1284 strobe timing when the 1284 port is configured to operate in compatibility mode. See "IEEE 1284 strobe pulse width" on page 373 for more information.

Table 124: IEEE 1284 Port Control register bit definition

Code (Bit #)	Definition of code	Description
MAN (D23)	1284 manual operation mode	Must be set to 1 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. A setting of 1 is typically used to manipulate STROBE* for auto-negotiation purposes. See "IEEE negotiation" on page 355 for more information.
FAST (D22)	1284 fast configuration	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 mode of operation. See "IEEE 1284 forward compatibility mode" on page 355 and "IEEE 1284 forward ECP mode" on page 359 for information about configuring the FAST field.
BIDIR (D21)	1284 direction configuration	■ 0 – Output mode ■ 1 – Input mode 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 appears on the PDIR signal in the external control latch (see Table 120: "IEEE 1284 data bus assignments" on page 351).
PIO (D20)	1284 spare bit	Provides a spare control signal for application-specific use. The value written into the PIO field appears on the PIO signal in the external control latch (see Table 120: "IEEE 1284 data bus assignments" on page 351).
MSTB* (D19)	1284 manual strobe configuration	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. See "IEEE negotiation" on page 355 for more information.

Table 124: IEEE 1284 Port Control register bit definition

Code (Bit #)	Definition of code	Description
AUTOFD* (D18)	1284 AUTOFD* setting	Controls the state of the AUTOFD* signal in the external control latch. The value written into the AUTOFD* field appears on the AUTOFD* signal in the external control latch (see Table 120: "IEEE 1284 data bus assignments" on page 351).
INIT* (D17)	1284 INIT* setting	Controls the state of the INIT* signal in the external control latch. The value written into the INIT* field appears on the INIT* signal in the external control latch (see Table 120: "IEEE 1284 data bus assignments" on page 351).
HSELECT* (D16)	1284 HSELECT* setting	Controls the state of the HSELECT* signal in the external control latch. The value written into the HSELECT* field appears on the HSELECT* signal in the external control latch (seeTable 120: "IEEE 1284 data bus assignments" on page 351).
INTEF (D15)	1284 port FIFO empty interrupt request enable	Must be set active high to generate an interrupt when the 1284 data FIFO is empty. Setting the INTEF bit to 1 causes an interrupt to be generated when the FIFOE status bit is 1. The INTEF bit should be set only when operating the 1284 port in interrupt service mode instead of DMA mode.
EPP (D14)	1284 EPP mode of operation	Must be set to 1 to allow the 1284 port to operate in the EPP mode of operation. See "IEEE 1284 EPP mode" on page 363 for information about operating the 1284 port in EPP mode.
IBRIE (D13)	1284 inbound buffer ready interrupt enable	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 cause san interrupt to be generated when the IBR status bit is 1. The IBRIE bit should be set only when operating the 1284 port in interrupt service mode instead of DMA mode.
IBR (D12)	1284 inbound buffer ready	Goes active high when the inbound buffer is ready to be read by the processor. The IBR status bit is active only when the 1284 port is configured for operation in the ECP reverse channel mode (ECP = 1; BIDIR = 1).

Table 124: IEEE 1284 Port Control register bit definition

Code (Bit #)	Definition of code	Description
RXFDB (D11:10)	1284 inbound buffer FIFO data available	■ 00 – Full-word ■ 01 – One-byte ■ 10 – Half-word ■ 11 – Three bytes (LENDIAN determines which three) 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 used only when operating the receive 1284 port in interrupt service mode instead of DMA mode.
RBCC (D09)	1284 receive buffer close command byte	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 (D08)	1284 receive buffer character timeout	Can be set at the same time IBR is set. The RBCT bit indicates 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 this equation: `TIMEOUT = STROBE configuration * 16` where: ■ `TIMEOUT` is the maximum time since the receipt of the last character. ■ `STROBE` is the time value defined by the STROBE field.
ACK* (D07)	1284 ACK* input state	Provides the current state of the 1284 ACK* input signal. A high-to-low transition of the ACK* input causes the ACKI status bit to be latched active high.
FIFOE (D06)	1284 FIFO empty status	Indicates that the 1284 Outbound Data register and FIFO are empty of characters. An interrupt can be generated to the CPU when FIFOE goes active high, provided the INTEF control bit is set to 1.

Table 124: IEEE 1284 Port Control register bit definition

Code (Bit #)	Definition of code	Description
OBE (D05)	1284 outbound buffer empty	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 the INTEP control bit is set to 1.
ACKI (D04)	1284 ACK* transition detected	Set to 1 whenever a high-to-low transition is detected on the 1284 ACK* signal input. The ACKI bit remains 1 until acknowledged. To acknowledge the ACKI bit, write a 1 in the ACK1 bit position [D4] in the register. An interrupt can be generated to the CPU when ACKI goes active high, provided the INTEA control bit is set to 1.
BUSY (D03)	1284 busy input state	Provides the current state of the 1284 BUSY input signal. The BUSY bit is one of the four bits in the 1284 reverse channel nibble.
PE (D02)	1284 PE input state	Provides the current state of the 1284 PE input signal. The PE bit is one of the four bits in the 1284 reverse channel nibble.
PSELECT (D01)	1284 PSELECT input state	Provides the current state of the 1284 PSELECT input signal. The PSELECT bit is one of the four bits in the 1284 reverse channel nibble.
FAULT* (D00)	1284 FAULT* input state	Provides the current state of the 1284 FAULT* input signal. The FAULT* bit is one of the four bits in the 1284 reverse channel nibble.

Table 124: IEEE 1284 Port Control register bit definition

IEEE 1284 Channel Data registers

The IEEE 1284 Channel Data registers are 32-bit registers used to manually interface with the IEEE 1284 FIFOs instead of using DMA support. Each port has its own data register. These registers are 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 writes can be accomplished in assembler using these instructions respectively: STRB, STRH, and STR. Similarly, these writes can be accomplished in

C using these 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.

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

Table 125: IEEE 1284 Channel Data register bit definition

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.

Table 126 shows sample STROBE times using an 18.432 MHz crystal:

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 126: Sample IEEE 1284 strobe widths

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.

ENI mode overview

The ENI is a 16-bit port with its own data and address busses, protocol and timing. Its purpose is to allow an external device to access the NET+50's system RAM. Figure 49 shows an external processor talking to the NET+50's RAM through the ENI interface.

Note: The external processor is referred to as the *client* where the NET+50 containing the ENI interface port is referred to as the *host*. The external processor system (client) initiates all data transfers. The host responds to the client's commands and reads or writes the client's data into the NET+50 system RAM.

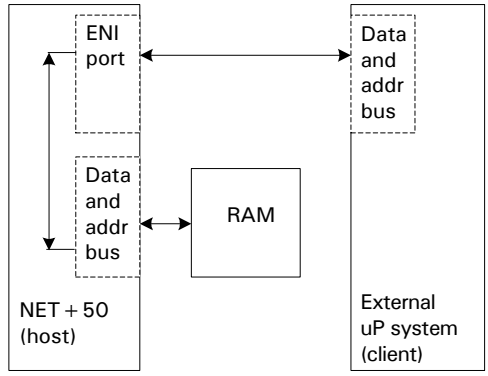


Figure 49: ENI basic functionality

The ENI's configuration can be adjusted in many ways for the client's convenience. There are four ENI modes:

- **8-bit shared RAM.** Allows the client to access 65536 bytes in the NET+50 system RAM.
- **16-bit shared RAM.** Allows the client to access 32768 word locations.
- **8-bit FIFO bus and 16-bit FIFO bus.** Allow shared RAM access with the memory space being either 8K, 16K, or 32K depending on the configuration.

Each mode has additional features that are configured with the seven registers that configure and control the ENI interface.

ENI host interface

The ENI host mode allows an external processor to access the NET+50 as a slave memory device. The NET+50 ENI is connected to the system bus of the external processor, which can generate four different interrupts to the NET+50. The ENI interrupts to the NET+50 share the same interrupt mask location bits as the IEEE 1284 port interrupts. Sharing these location bits is not a problem because ENI can be configured in only one mode at a time.

When the NET+50 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+50 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+50's system bus. The NET+50 also has the ability to interrupt the external processor using the PINT1 and PINT2 signals in the ENI. These signals are covered in greater detail in the ENI host port application note available from NetSilicon.

The ENI host interface provides the interface between the NET+50 chip and a separate processor subsystem, and acts as a host memory device to the external processor subsystem. The ENI host interface also provides both a shared RAM interface module and a byte-streaming FIFO module for interprocessor communications between the NET+50 process and the external processor.

Figure 50 shows the signals used when interfacing to an external processor. The ENI uses forty NET+50 pins. These are dual function pins that are shared with the IEEE 1284 ports.

Note: The P before all the ENI signal names stands for printer and is a legacy from when the ENI was considered strictly a printer interface.

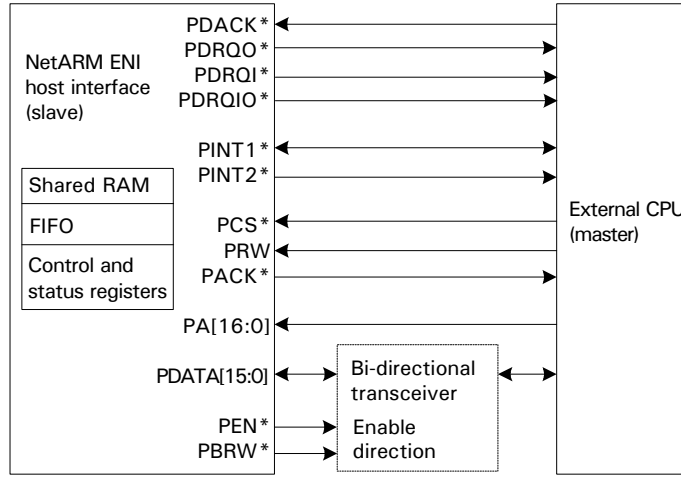


Figure 50: NET+50 ENI host signals

The external processor is responsible for addressing the ENI host interface as a host peripheral, and activates the interface using the PCS* or PDACK* inputs. The

NET+50 ENI host interface services the request and responds with PACK* when the cycle is complete.

In shared RAM mode, the ENI provides a 16-bit bi-directional data bus with 17 address bits. Under normal circumstances, seventeen address bits means that you can address 131072 locations. Bit 16 is a special case, however, and is used to access some of the ENI registers; it is not available for memory access. Therefore, a maximum of 65536 memory locations can be accessed.

The signals PCS* and PRW* are sourced by the client. They initiate the data transfer, and control the data's direction. The host acknowledges receiving these signals by issuing PACK. If the data bus needs buffering, the signals PEN* and PBRW control the buffer. PINT1* and PINT2* are interrupt outputs that allow the host to interrupt the client. PINT2* can also be configured as an interrupt input that allows the client to interrupt the host. DRQI*, DRQO*, DRQIO, and DACK are used when ENI is configured for the FIFO mode to facilitate DMA transfers.

Signal descriptions

PCS* **ENI chip select**

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**

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**

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 and 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. See "Memory map" on page 383 for a list 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 routed only 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.
- 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 routed only 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; 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 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 that 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 16-bit or 8-bit data mode. In 16-bit mode, all data bits are used; in 8-bit mode only PDATA[15:8] are used.

In 8-bit mode, PDATA[7:0] can be used as general-purpose inputs.

PACK* **ENI Acknowledge**

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 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 (see "MIC module hardware initialization" on page 333). Figure 90, "ENI Shared RAM and Register Cycle

timing," on page 464 shows the difference between the ACK and RDY configuration for the PACK* signal.

The PACK* signal can be configured to operate as 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 (see "MIC module hardware initialization" on page 333).

PINT1* **ENI Interrupt 1**

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 determined by 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 driven active low only when an interrupt condition is pending, the EHWINT bit is set to 1, and the SINTP2 bit is set to 0.

See Figure 53, "PINT1*/PINT2* interrupt pins," on page 390.

PINT2* **ENI Interrupt 2**

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 driven active low only 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 automatically be initialized to its proper state on reset by including or not including a pulldown resistor on the A5 signal (see "MIC module hardware initialization" on page 333). 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 32Kb in FIFO DMA mode.

See Figure 53, "PINT1*/PINT2* interrupt pins," on page 390.

PBRW* **ENI Data Buffer Read/Write**

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+50 chip and the external ENI processor system. PBRW* high indicates a data transfer from the NET+50 chip to the ENI processor; PBRW* low indicates a data transfer from the ENI processor to the NET+50 chip.

The ENI interface provides a clock synchronizer function for the PACK* output. The PBRW* pin can be reconfigured as a clock input for the PACK* synchronizer. See "ENI Control register and bit definitions" on page 408 for more information.

PEN* **ENI Data Buffer Enable**

An output that controls the output enable pin for an external data bus transceiver. Some applications require a data bus buffer between the NET+50 chip and the external ENI processor system. PEN* low indicates that a data transfer between the NET+50 chip and the external ENI processor is in progress.

ENI shared RAM mode

When the ENI interface is configured for ENI host shared RAM mode, the NET+50 chip provides up to 64 Kbytes of shared RAM between the NET+50 chip and an external device. Figure 51 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 that the cycle is complete (providing read data during read cycles).

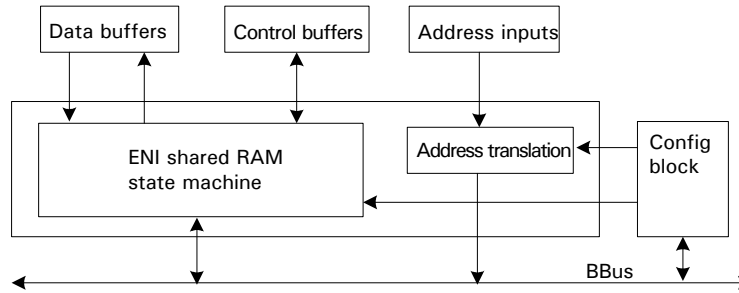


Figure 51: ENI shared RAM block diagram

The ENI shared RAM occupies a physical block within external NET+50 chip system RAM. The ENI controller accesses the shared RAM through the ENI interface. The NET+50 chip firmware directly accesses the shared RAM within its system RAM.

Memory map

The ENI interface uses 17 address bits to access the shared RAM and various control registers. Table 127 shows the address map from the perspective of the external ENI.

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 127: ENI interface address map

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 64 Kbytes 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 8 Kbytes 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 is delayed while the NET+50 ENI shared RAM state machine arbitrates with the internal NET+50 bus clients (CPU and DMA) for access to external memory. After gaining access to the internal NET+50 BBus, the ENI shared RAM state machine executes a memory read or memory write cycle (depending on the state of the PRW* input) to a memory location defined by the BASE field located in the ENI Shared RAM Address register (see "ENI Shared RAM Address register and bit definitions" on page 415).

After the NET+50 memory subsystem completes the memory transfer for the ENI shared RAM state machine, the PACK* signal is issued active to the external ENI interface along with read data during a read cycle. Therefore, the length of 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 depends on

how the NET+50 memory controller is configured, the speed of various memory devices, DMA activity, bursting configurations, and so on. See Figure 90, "ENI Shared RAM and Register Cycle timing," on page 464 for a detailed timing diagram of a shared RAM transfer.

This sequence of events makes up an ENI shared RAM transfer:

1. The printer sends a chip select (PCS*), an address (PA16:0), and a read/write (PRW-) signal to the NET+50.
2. The NET+50 drives PACK low, enables a bi-directional data transceiver by driving PEN low and determines the direction of that data with PBRW. (PBRW high is data from the NET+50 to the printer)
3. In the background (no P signals change while this is being done), the NET+50 shared RAM state machine takes the address provided by the printer and adds it to an offset (previously defined in a register) to access a word in the DRAM. When that word (copied to a shared RAM state machine register) is ready to be taken by the printer, the NET+50 drives the PACK- signal high.
4. When PACK- goes high, the printer reads the word and drives PCS high.
5. After PCS goes high, the NET+50 drives PEN high, ending the cycle.

Shared register

The NET+50 chip supports an 8-bit shared register that controls the personality of signals and provides the NET+50 chip with the means to interrupt the ENI. The NET+50 chip accesses this register through 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 128 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.

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 128: Shared register layout (ENI interface perspective only)

Figure 52 shows how the ENI shared register can be written. Both sides can read all eight bits.

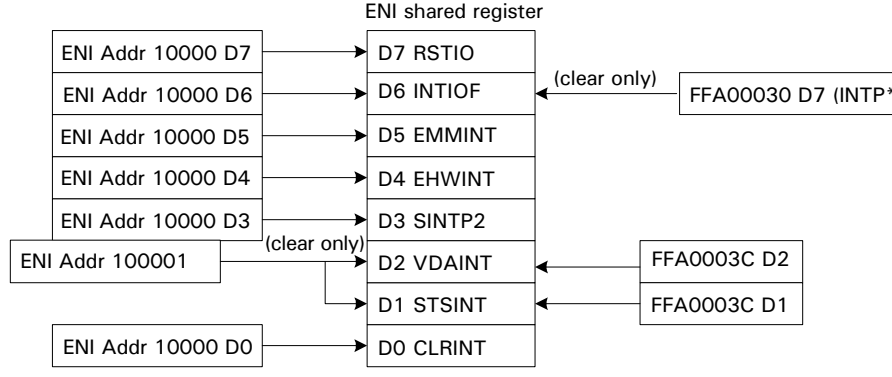
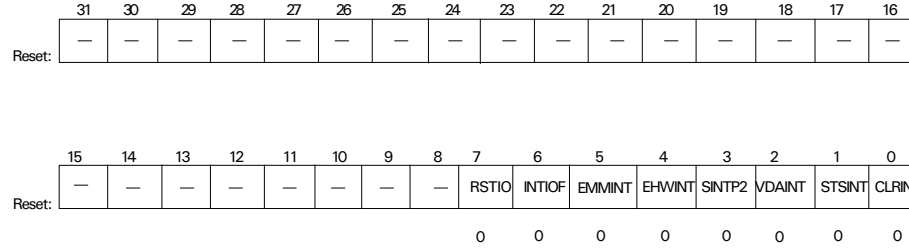


Figure 52: Writing to the ENI shared register

Unlike accessing shared RAM, the Shared register access timing is always the same when accessing the Shared register from the external ENI interface. Because the ENI hardware does not need to arbitrate with internal resources when accessing the shared register, the shared register access timing is fast and consistent.

ENI Shared register and bit definitions

```
CPU => FFA0 003C
ENI => 1 0000 ENI shared Register (shown in NORMAL mode)
```



Code (Bit #)	Definition of code	Description
RSTIO (D07)	ENI reset NET + 50 processor	Only the external ENI interface can modify the RSTIO bit; the NET + 50 CPU cannot modify this bit. The external ENI interface can set the RSTIO bit to 1 to reset the NET + 50 processor and peripherals. The processor and peripherals remain in reset while RSTIO is set to 1. RSTIO must be set back to 0 to enable the NET + 50 CPU to reboot. RSTIO is used by the external interface to reset the internal NET + 50 processor. The RSTIO bit does not reset either the ENI or MEM modules within the NET + 50 architecture but does reset all other internal NET + 50 modules. The external ENI interface reads a 1 in the RSTIO bit position whenever the CLRINT bit in this register is also set.

Table 129: ENI Shared register bit definition

Code (Bit #)	Definition of code	Description
INTIOF (D06)	ENI interrupt NET+50 processor	Only the external ENI interface can modify the INTIOF bit. The NET+50 CPU cannot modify INTIOF directly. The external ENI interface can set the INTIOF bit to 1 to generate an interrupt to the NET+50 CPU. When the INTIOF bit is set to 1, the INTP* bit in the ENI Control register is 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+50 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+50 processor. The ENI INTIOF interrupt is routed to the GEN module Interrupt Controller through the ENI Port 3 interrupt (see "MIC module interrupts" on page 333).
EMMINT (D05)	ENI spare resource bit	Only the external ENI interface can modify the EMMINT bit. The NET+50 CPU cannot modify EMMINT directly. The EMMINT bit controls no hardware within the NET+50. It can be used as a general-purpose control/status bit that can be controlled by the external ENI interface only.
EHWINT (D04)	ENI enable hardware interrupt	Only the external ENI interface can modify the EHWINT bit. The NET+50 CPU cannot modify EHWINT directly. The EHWINT bit must be set to 1 by the external ENI interface to allow either the PINT1* or PINT2* signals to generate an active low interrupt to that interface. Sources of interrupt for the external ENI interface include the STSINT and VDAINT bits in the ENI Shared register, the RDBUFRDY* and WRBUFEMP* bits in the FIFO mask/status register, and a write by the NET+50 CPU to the pulsed interrupt register.
SINTP2 (D03)	ENI PINT1*/ PINT2* interrupt selection	■ 0 – PINT1* ■ 1 – PINT2* Only the external ENI interface can modify the SINTP2 bit. The NET+50 CPU cannot modify SINTP2 directly. The SINTP2 bit is controlled by the external ENI interface to determine which signal — PINT1* or PINT2* — is used for interrupts to the external ENI interface. See Figure 53, "PINT1*/PINT2* interrupt pins," on page 390 for more information.

Table 129: ENI Shared register bit definition

Code (Bit #)	Definition of code	Description
VDAINT (D02)	NET + 50 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 + 50 CPU to send an interrupt to the external ENI interface using either PINT1* or PINT2*, depending on 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 "Memory map" on page 383.
STSINT (D01)	NET + 50 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 + 50 CPU to send an interrupt to the external ENI interface using either PINT1* or PINT2*, depending on 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 "Memory map" on page 383.
CLRINT (D00)	ENI clear interrupt from NET + 50	Only the external ENI interface can modify the CLRINT bit. The NET + 50 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.

Table 129: ENI Shared register bit definition

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.

When writing to the clear interrupt location from the external ENI interface, the access timing is always the same — unlike accessing shared RAM,. Because the ENI hardware does not need to arbitrate with internal resources when writing to the clear interrupt location, the access timing is fast and consistent.

Address 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 uppermost two words of shared RAM.

The ENI interface uses 17 address bits to access the shared RAM and various control registers. Table 130 shows the address map as defined from the perspective of the external ENI.

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 130: ENI interface address map

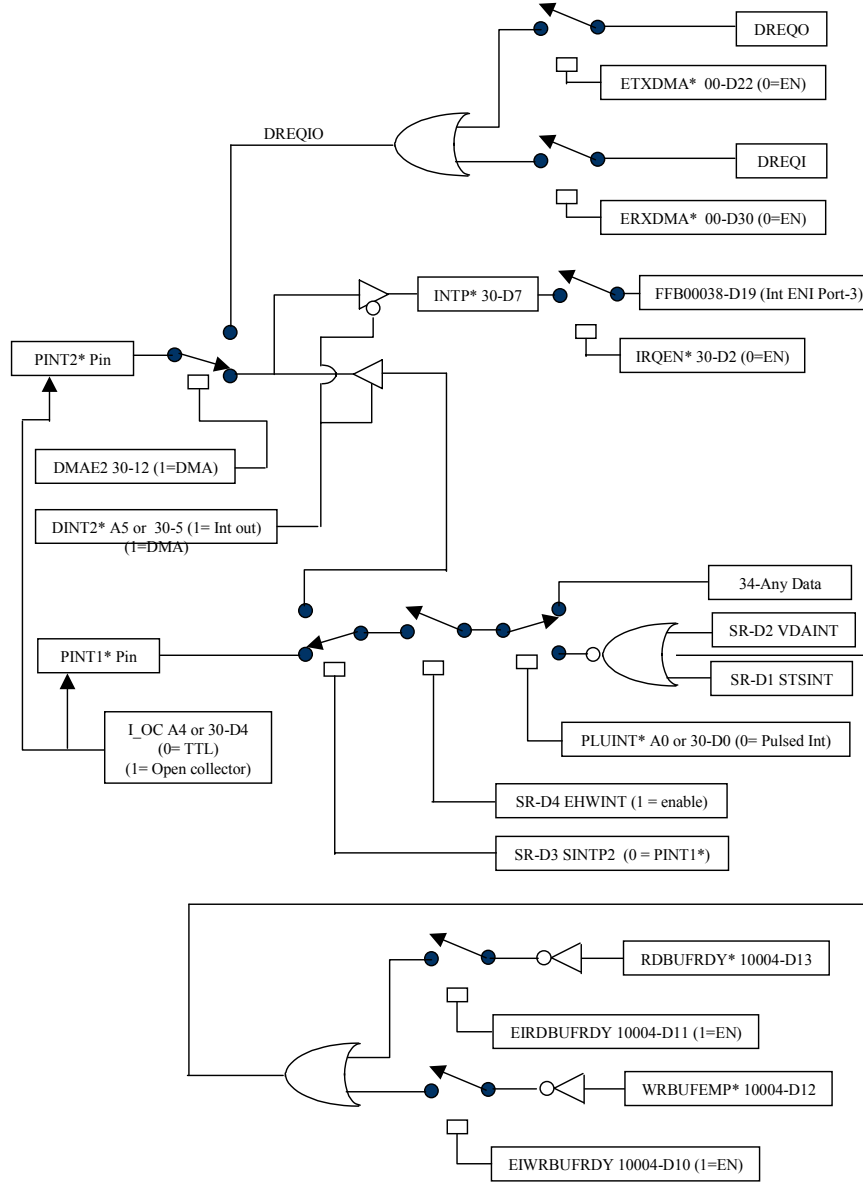


Figure 53: PINT1*/PINT2* interrupt pins

ENI bus error interrupts

Figure 54shows the ENI bus error (BEI) interrupt process.

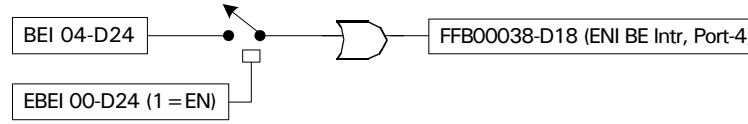


Figure 54: ENI bus error interrupt process

If the shared RAM address register (FFA00038) is improperly configured, the ENI device may try to access a non-existent memory location in the ENI's RAM. If this happens, the BEI bit will go high. The BEI bit must be cleared by writing a 1 to this bit location. If this bit is enabled by the EBEI bit in the General Control register, it can generate an interrupt. (See "General Control register and bit definitions" on page 401 and "General Status register and bit definitions" on page 405 for more information about these bits.)

ENI FIFO mode module

The ENI host FIFO mode interface provides a FIFO interface between the NET+50 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+50 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 55 provides a view of the hardware required to support FIFO mode. The FIFO mode control state machine is responsible for running the register interface.

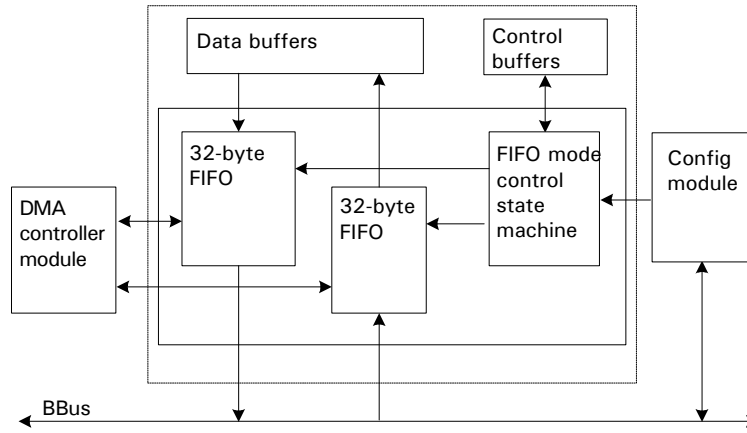


Figure 55: ENI FIFO mode block diagram

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 manner. 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 56 shows two NET+ARM systems; one is configured in ENI FIFO mode and is controlled by the other's NET+ARM memory bus signal.

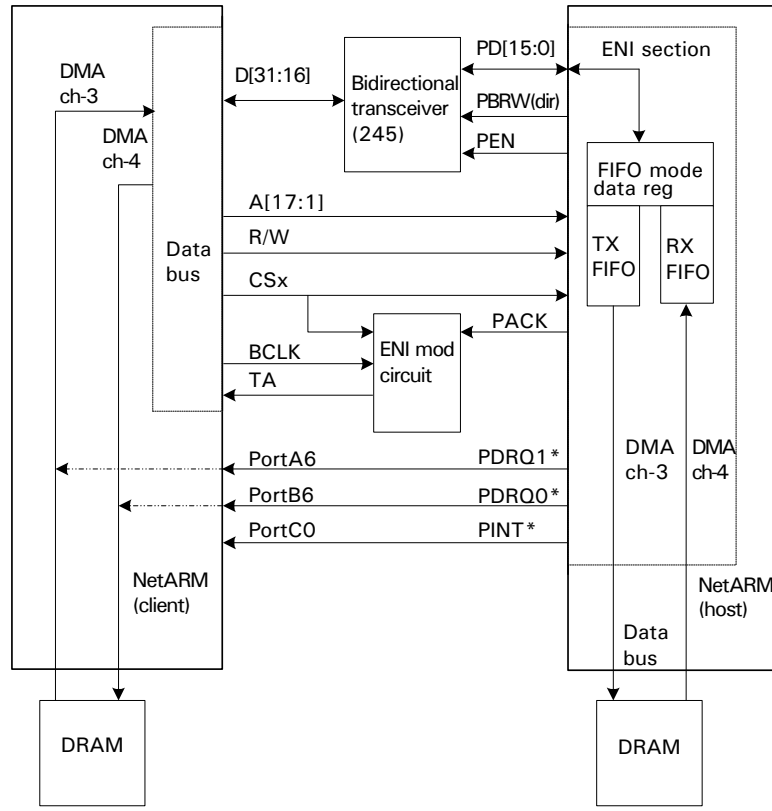


Figure 56: Sample application

FIFO Data register

The FIFO Data register is accessible only 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+50 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 be written only when the WRBUFEMP* status bit in the FIFO mask/status register is active low. The FIFO Data register can be read only 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. 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 FIFO can be configured to transport data through DMA channels, using three signals to communicate with an external device: DREQO, DREQI, and PDACK. These signals can be configured to act in conjunction with either 8K or 32K shared RAM. Figure 57 shows the configuration from inside NET+50.

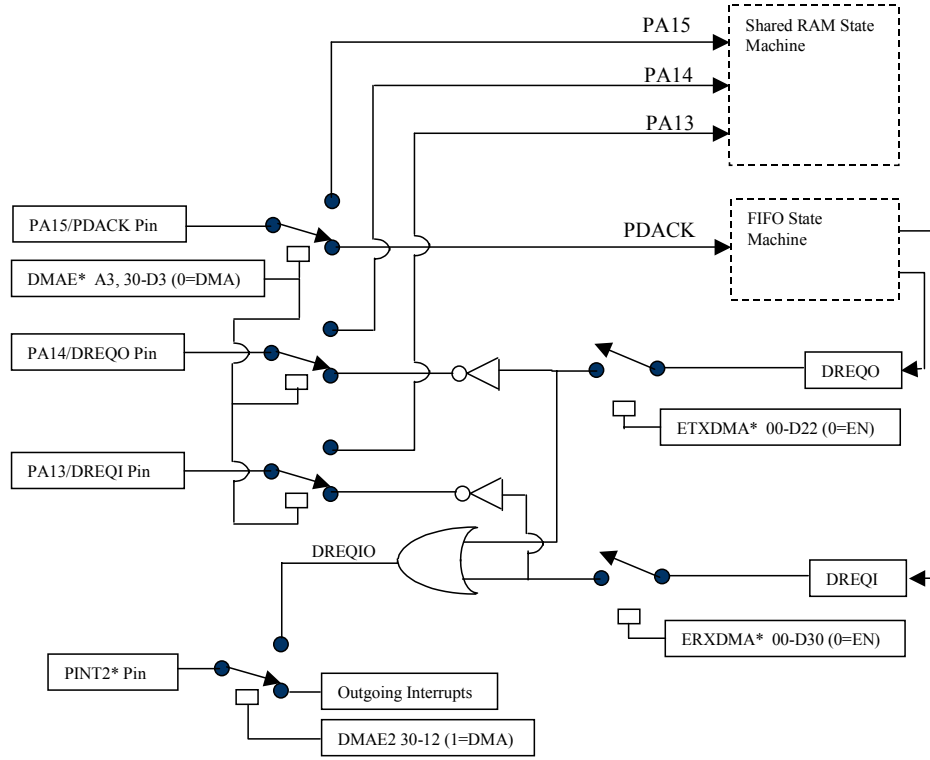
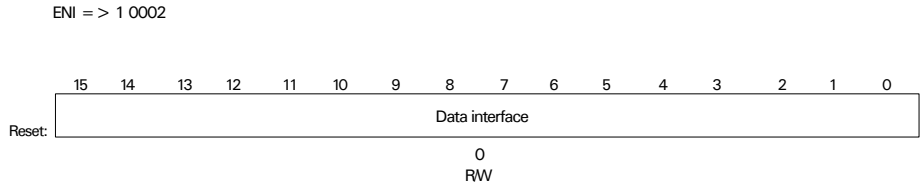


Figure 57: ENI FIFO — Transporting data through DMA channels

The data register can also be DMA-mapped using the PDACK* input. When using the PDACK* input, all ENI address inputs are ignored and the sense of PRW* is

reversed. For DMA operations, PRW* high writes to the Data register (read from controller memory) and PRW* low reads from the Data register (write to controller memory). The DMA-mapped interface is used in single address, or fly-by, DMA operations. The single address method requires a single bus cycle per DMA operation. The memory is 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 Data register can be configured using an 8- or 16-bit interface. When configured for 8-bit operation, only the upper eight data bits — PDATA(15:8) — are used.



When accessing the FIFO Data register from the external ENI interface, the access timing is always the same — unlike accessing shared RAM. Because the ENI hardware does not need to arbitrate with internal resources when accessing the FIFO Data register, the access timing for this data register is fast and consistent.

FIFO Mask/Status register and bit definition

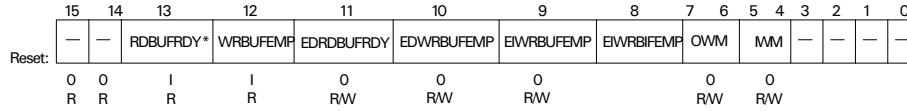
The FIFO Mask/Status register is accessible only from the ENI interface when the ENI controller is configured to operate in ENI FIFO mode.

The FIFO Mask/Status register supports the data FIFO mode transfer mechanism between the NET+50 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 powerup RESET.

When writing to the clear interrupt location from the external ENI interface, the access timing is always the same — unlike accessing shared RAM. Because the ENI

hardware does not need to arbitrate with internal resources when writing to the clear interrupt location, the access timing is fast and consistent.

ENI = > 1 004



Code (Bit #)	Definition of code	Description
RDBUFRDY* (D13)	FIFO read buffer ready	■ 0 – Read Data Register is ready ■ 1 – Read Data Register is empty The RDBUFRDY* status bit is active low when the FIFO has 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* (D12)	FIFO write buffer empty	■ 0 – Write Data Register is empty ■ 1 – Write Data Register is full The WRBUFEMP* status bit is 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 (D11)	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 when the RDBUFRDY* status bit is active low. The outbound DMA request signal is routed out either PA14 or PINT2, depending on the settings of the DMAE* and DMAE2 fields in the ENI Control register.

Table 131: FIFO Mask/Status register bit definition

Code (Bit #)	Definition of code	Description
EDWRBUFEMP (D10)	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 on the settings of the DMAE* and DMAE2 fields in the ENI Control register.
EIRDBUFRDY (D09)	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 when the RDBUFRDY* status bit is active low. The interrupt request signal is routed out either PINT1* or PINT2*, depending on the setting of the SINTP2 field in the Shared register.
EIWRBUFEMP (D08)	Enable write buffer empty interrupt	■ 0 – Mask write buffer empty interrupt requests ■ 1 – Enable write buffer empty interrupts Can be set by the external ENI interface to enable an interrupt request to that 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 on the setting of the SINTP2 field in the Shared register.

Table 131: FIFO Mask/Status register bit definition

Code (Bit #)	Definition of code	Description
OWM (D07:06)	Outbound watermark	Controls the watermark configured for the outbound FIFO determining when the RDBUFRDY* signal becomes active low. This results in either an interrupt or a DMA request. ■ 00 — One operand; the RDBUFRDY* signal will be active when at least one operand is available for reading from the outbound FIFO. ■ 01 — Half full; the RDBUFRDY* signal will be active when the outbound FIFO contains at least 16 bytes of read data. ■ 10 — Full; the RDBUFRDY* signal will be active only when the outbound FIFO contains 32 bytes of read data.
IWM (D05:04)	Inbound watermark	Controls the watermark configured for the inbound FIFO determining when the WRBUFEMP* signal becomes active low. This results in either an interrupt or a DMA request. ■ 00 — One operand; the WRBUFEMP* signal will be active when at least one operand can be written to the inbound FIFO. ■ 01 — Half empty; the WRBUFEMP* signal will be active when the inbound FIFO can accept at least 16 bytes of write data. ■ 10 — Empty; the WRBUFEMP* signal will be active only when the inbound FIFO can accept 32 bytes of write data.

Table 131: FIFO Mask/Status register bit definition

FIFO receiver interrupts

The FIFO receiver can generate interrupts if the receive buffer is either ready or half full. Figure 58 shows the receive FIFO interrupt process:

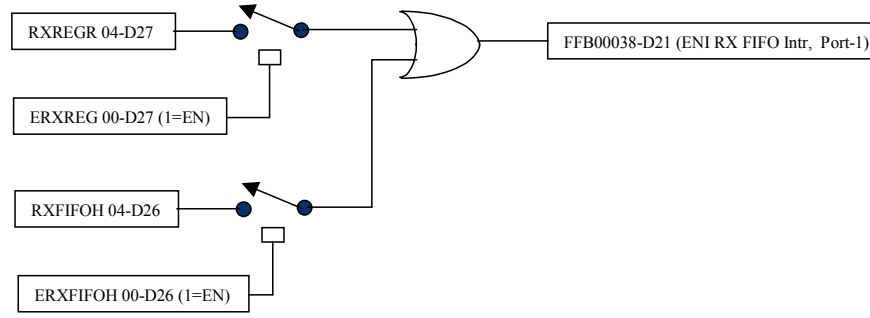


Figure 58: ENI receive FIFO interrupt process

The FIFO mode receive register ready (RXREGR) bit gets its information from the receive FIFO state machine, and is asserted whenever there is data that the NET+50 can read in the receive FIFO. This bit is cleared automatically when the FIFO is read. If this bit is enabled by the ERXREG bit in the General Control register, it can generate an interrupt. (See "General Control register and bit definitions" on page 401 and "General Status register and bit definitions" on page 405 for more information about these bits.)

The FIFO mode receive register half full (RXFIFOH) bit gets its information from the receive FIFO state machine, and is asserted when the receive FIFO is half full. This bit is cleared when enough data has been read from the FIFO so the FIFO is less than half full. If this bit is enabled by the ERFIFOH bit in the General Control register, it can generate an interrupt. (See "General Control register and bit definitions" on page 401 and "General Status register and bit definitions" on page 405 for more information about these bits.)

FIFO transmitter interrupts

The FIFO transmitter can generate interrupts if the transmit buffer is either ready or half full. Figure 59 shows the transmit FIFO interrupt process:

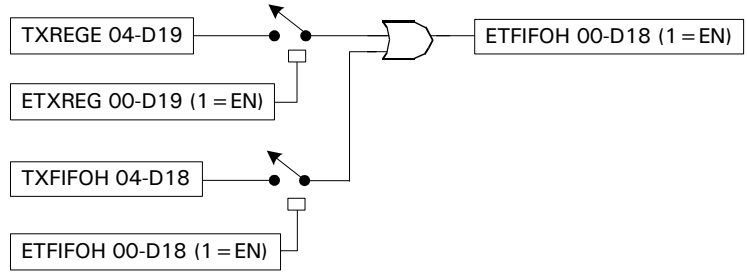


Figure 59: ENI transmit FIFO interrupt process

The FIFO mode transmit register ready (TXREGE) bit gets its information from the transmit FIFO state machine, and is asserted when the transmit FIFO bit is empty. This bit is automatically cleared when data is placed in the FIFO. If this bit is enabled by the ETXREG bit in the General Control register, it can generate an interrupt. (See "General Control register and bit definitions" on page 401 and "General Status register and bit definitions" on page 405 for more information about these bits.)

The FIFO mode transmit register half full (TXFIFOH) bit gets its information from the transmit FIFO state machine, and is asserted when the transmit FIFO is half empty. This bit is cleared when enough data has been placed in the FIFO so the FIFO is more than half full. If this bit is enabled by the ETFIFOH bit in the General Control register, it can generate an interrupt. (See "General Control register and bit definitions" on page 401 and "General Status register and bit definitions" on page 405 for more information about these bits.)

ENI mode registers

Table 132 shows the registers that control the ENI interface.

Address	Register
FFA0 0000	General Control register
FFA0 0004	General Status register

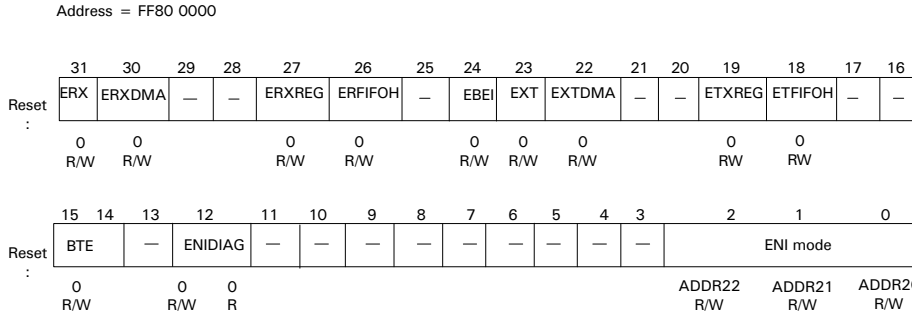
Table 132: ENI controller configuration registers

Address	Register
FFA0 0008	FIFO mode FIFO 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 132: ENI controller configuration registers

General Control register and bit definitions

The General Control register is a 32-bit register containing control bits that affect the entire ENI controller module.



Code (Bit #)	Definition of code	Description
ERX (D31)	Enable receive FIFO	0 – Disables inbound data flow and resets the FIFO 1 – Enables inbound data flow 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.

Table 133: General Control register bit definition

Code (Bit #)	Definition of code	Description
ERXDMA (D30)	Enable receive FIFO DMA	0 – Disables inbound DMA data request (use to stall receiver) 1 – Enables inbound DMA data request 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 receiver FIFO in interrupt service mode. In general, the ERXDMA bit should be set once on device open.
ERXREG (D27)	Enable receive FIFO data ready interrupt	Must be set to active high to generate an interrupt when data is available in the receive FIFO. The ERXREG bit should be set only when operating the receive FIFO in interrupt service mode instead of DMA mode. The receive FIFO interrupt is routed to the GEN module interrupt controller using the ENI Port 1 interrupt (see "MIC module interrupts" on page 333).
ERFIFOH (D26)	Enable receive FIFO half full interrupt	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 be set only when operating the receive FIFO in interrupt service mode instead of DMA mode. The receive FIFO interrupt is routed to the GEN module interrupt controller using the ENI Port 1 interrupt (see "MIC module interrupts" on page 333).
EBEI (D24)	Enable ENI bus error interrupt	Can be set to 1 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 encounters a bus error (data abort) condition while attempting to address shared RAM. This typically indicates that the shared RAM pointer is invalid. The ENI bus error interrupt is routed to the GEN module interrupt controller using the ENI Port 4 interrupt (see "MIC module interrupts" on page 333).

Table 133: General Control register bit definition

Code (Bit #)	Definition of code	Description
ETXDMA (D22)	Enable transmit FIFO DMA	0 – Disables outbound DMA data request (use to stall transmitter) 1 – Enables outbound DMA data request 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 (D19)	Enable transmit FIFO Data register ready interrupt	Must be set to active high to generate an interrupt when the transmit FIFO is ready to accept more data. The ETXREG bit should be set only when operating the transmit FIFO in interrupt service mode instead of DMA mode. The transmit FIFO interrupt is routed to the GEN module interrupt controller using the ENI Port 2 interrupt (see "MIC module interrupts" on page 333).
ETFIFOH (D18)	Enable transmit FIFO half empty interrupt	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 be set only when operating the transmit FIFO in interrupt service mode instead of DMA mode. The transmit FIFO interrupt is routed to the GEN module interrupt controller using the ENI Port 2 interrupt (see "MIC module interrupts" on page 333).

Table 133: General Control register bit definition

Code (Bit #)	Definition of code	Description
BTE (D15:14)	Burst transfer enable	■ 00 – No burst pre-fetching is performed ■ 01 – Burst pre-fetching of 8 bytes can be performed when reading from an 8-byte boundary ■ 10 – Burst pre-fetching of 16 bytes can be performed when reading from a 16-byte boundary ■ 11 – Reserved The shared RAM module supports a bursting pre-fetch option. When the external ENI client processor reads the shared RAM space and the address is on a 16- or 8-byte boundary, the NET+50 performs a burst read pre-fetch and stores the result in an internal storage buffer. If this feature is used, subsequent sequential accesses from the client interface result in increased performance because the anticipated read operations are pre-fetched and stored in a mini-cache block. The mini-cache is immediately flushed when accesses are no longer sequential. This feature cannot be used if shared RAM coherency is a concern.
MICDIAG (D12)	Enable MIC diagnostic mode	■ 0 – Disable firmware change of MICMODE or MICCTL register ■ 1 – Enable firmware change Provides protection against accidentally modifying the MICMODE or ENI Control register settings. Setting MICDIAG to 1 allows the software to modify either the MICMODE field in the General Control register or the least significant eight bits of the ENI Control register. When MICDIAG is set to 0, neither the MICMODE nor the least significant eight bits of the ENI Control register can be modified.

Table 133: General Control register bit definition

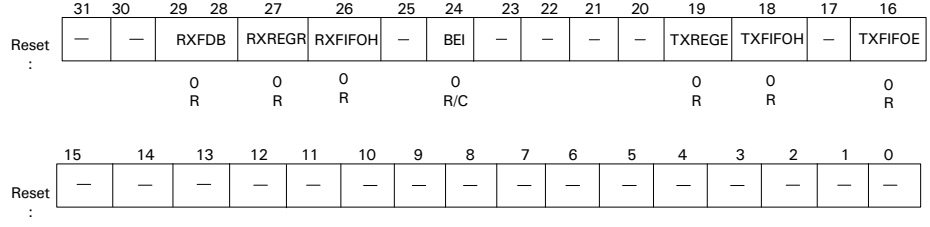
Code (Bit #)	Definition of code	Description
MICMODE (D02:00)	MIC mode of operation	■ 000 – GPIO ■ 001 – IEEE 1284 host mode ■ 010 – Reserved ■ 011 – Reserved ■ 100 – ENI shared-16 RAM mode; 16-bit shared RAM only ■ 101 – ENI shared-8 RAM mode; 8-bit shared RAM only ■ 110 – ENI FIFO mode; 16-bit with 8 Kbytes shared RAM ■ 111 – ENI FIFO mode; 8-bit with 8 Kbytes shared RAM Configures the mode of operation for the MIC module. The MICMODE field is automatically loaded on powerup or hardware reset (RESET* pin) using the bootstrap configuration bits defined on address pins A22, A21, and A20. See "MIC module hardware initialization" on page 333 for more information on bootstrap.

Table 133: General Control register bit definition

General Status register and bit definitions

The General Status register is a 32-bit register containing status bits that affect the ENI interface.

Address = FFA0 0004



Code (Bit #)	Definition of code	Description
RXFDB (D29:28)	Receive FIFO data available	■ 00 – Full-word ■ 01 – One-byte ■ 10 – Half-word ■ 11 – Three Bytes (LENDIAN determines which three) 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 used only when operating the receive FIFO in interrupt service mode instead of DMA mode.
RXREGR (D27)	Receive FIFO Read Register ready	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 if the ERXREGR bit is set in the General Control register. The RXREGR bit is used only when operating the receive FIFO in interrupt service mode instead of DMA mode.
RXFIFOH (D26)	Receive FIFO half full	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 used only when operating the receive FIFO in interrupt service mode.
BEI (D24)	ENI bus error	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 if the EBEI bit is set in the General Control register.
TXREGE (D19)	Transmit FIFO Register ready	Set to 1 whenever the transmit FIFO is ready to accept data. This bit, when active high, can cause an interrupt to occur if the ETXREG bit is set in the General Control register. The TXREGE bit is used only when operating the transmit FIFO in interrupt mode instead of DMA mode.

Table 134: General Status register bit definition

Code (Bit #)	Definition of code	Description
TXFIFOH (D18)	Transmit FIFO half full	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 if the ETFIFOH bit is set in the General Control register. The TXFIFOH bit is used only when operating the transmit FIFO in interrupt mode instead of DMA mode.
TXFIFOE (D16)	Transmit FIFO empty	Set to 1 whenever the transmit FIFO is empty. The TXFIFOE bit cannot be used to generate an interrupt to the ARM processor.

Table 134: General Status register bit definition

ENI mode FIFO Data register

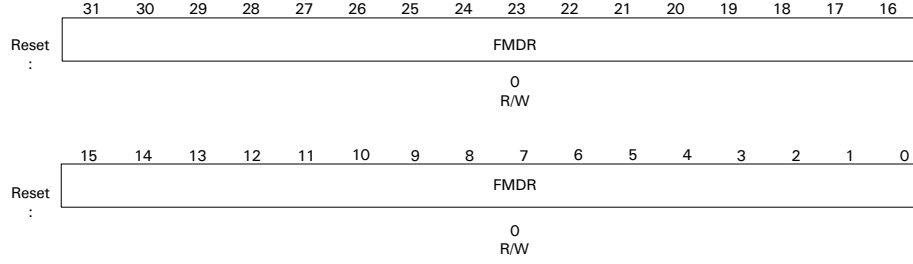
The ENI mode FIFO Data register is a 32-bit register used by the ARM processor to interface with the ENI mode FIFO. Writing to this register loads the transmit FIFO. This register can be written only when the transmit register empty (TXREGE) bit is set in the General Status register. Reading from this 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 in assembler or C:

ARM processor	Assembler instructions	C instructions
Bytes	STRB	`*(char*)0xFFA00008`
Half-word	STRH	`*(short*)0xFFA00008`
Full-word	STR	`*(int*)0xFFA00008`

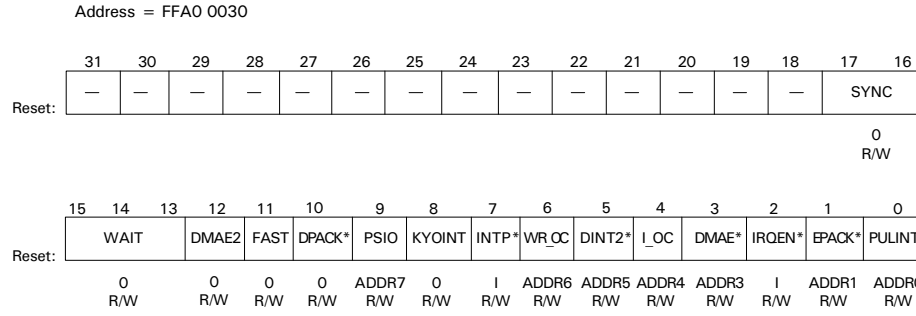
When reading from the ENI mode FIFO Data register, all bytes available (as defined by the RXFDB field) must be read with a single instruction:

- If only a single byte is available, an 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, an 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.



ENI Control register and bit definitions

The ENI Control register is a 32-bit read/write register that 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.



Code (Bit #)	Definition of code	Description
SYNC (D17:16)	PACK* output synchronization mode	Provides control over the PACK* output synchronization mode: ■ **00** – The PACK* output is driven synchronous to the internal SYSCLK timing domain. This configuration is directly compatible with older NET+50 devices. ■ **01** – The PACK* output is driven synchronous to an external clock, which is provided as an input on the PBRW* pin. The PACK* output is synchronized from the SYSCLK domain to the PBRW* input clock domain using a single-rank flip-flop structure. ■ **10** – The PACK* output is driven synchronous to an external clock, which is provided as an input on the PBRW* pin. The PACK* output is synchronized from the SYSCLK domain to the PBRW* input clock domain using a dual-rank flip-flop structure. ■ **11** – Reserved.
WAIT (D15:13)	ENI added wait states for data setup to PACK*	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.

Table 135: ENI Control register and bit definition

Code (Bit #)	Definition of code	Description
DMAE2 (D12)	Enable DMA request through the PINT2 pin	■ 0 – Disable ■ 1 – Enable 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 be used only when FIFO DMA is enabled in one direction or another — never both. The FIFO DMA enable configuration is controlled using 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 through the PINT2* pin. ■ Both PA14 and PA13 are returned to their address input function allowing for a maximum of 32K bytes of available shared RAM.
FAST (D11)	ENI FAST timing mode	■ 0 – Classic ENI timing ■ 1 – Fast ENI timing 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* (D10)	DMA PACK* generation	■ 0 – Drive PACK* active during DMA cycles ■ 1 – Disable PACK* during DMA cycles 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.

Table 135: ENI Control register and bit definition

Code (Bit #)	Definition of code	Description
PSIO (D09)	PSIO configuration for shared register	■ 0 – Shared register configured for NORMAL mode ■ 1 – Shared register configured for PSIO mode Controls the bit formatting for the ENI Shared register. ■ 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 (see Table 128: "Shared register layout (ENI interface perspective only)" on page 385.) 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 (D08)	Kyocera interrupt option	■ 0 – Normal interrupt mode ■ 1 – Kyocera interrupt. Occurs when the ENI interface writes to the most significant address of shared RAM. An alternate mechanism for the ENI interface to issue an 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 provides a useful means for generating an interrupt to the ARM processor when the ENI interface cannot address the Shared register.
INTP* (D07)	Interrupt pending	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 IRQEN* (ENI Control register) is set active low and ENI3 (GEN 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 state by writing a 1 in this same bit position [D7].

Table 135: ENI Control register and bit definition

Code (Bit #)	Definition of code	Description
WR_OC (D06)	Wait/ready/ acknowledge open collector	■ 0 – Output driver is TTL; used for LN14, MIO, Kyocera, others ■ 1 – Output driver is OPEN COLLECTOR, GCC, others 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* (D05)	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 Controls the use 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 provides 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 is established at bootstrap by sampling the A5 signal during reset.
I_OC (D04)	Interrupt(s) open collector	■ 0 – Output driver is TTL; used for LN14, MIO, Kyocera, others ■ 1 – Output driver is OPEN COLLECTOR; GCC, others 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 is established at bootstrap by sampling the A4 signal during reset.

Table 135: ENI Control register and bit definition

Code (Bit #)	Definition of code	Description
DMAE* (D03)	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 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 whether 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 addressing 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* (D02)	Enable interrupt from ENI interface	■ 0 – Interrupts from the ENI are enabled. An interrupt condition is set up when the ENI issues a pulsed interrupt using INT2* or the INTIO bit is set in the ENI Shared register. ■ 1 – Interrupts from the ENI are disabled. 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.

Table 135: ENI Control register and bit definition

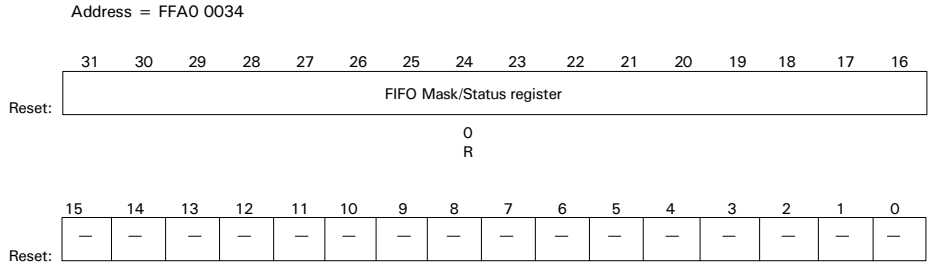
Code (Bit #)	Definition of code	Description
EPACK* (D01)	Enable ENI ACK pulse	■ 0 –The ACK pin is used as an active low data acknowledge signal; used for MIO, LN14, others ■ 1 –The ACK pin is used as an active low WAIT signal; used for LN14, GCC, others Controls the personality of the PACK* output. The default state 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* (D00)	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 Controls how interrupts are issued from the ARM processor to the ENI interface. This bit allows a pulsed interrupt signal to be drive 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 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 VDAINT and STSINT bit settings 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).

Table 135: ENI Control register and bit definition

ENI Pulsed Interrupt register and bit definition

The ENI Pulsed Interrupt register contains no meaningful data. This address is used to issue an interrupt request to the ENI interface. An interrupt condition is set up 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+50 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 jumper settings. The 8 data bits are presented in the least significant 8 bits of this register.



The external ENI client processor is responsible for setting the bits in the FIFO Mask/Status register (see "FIFO Mask/Status register and bit definition" on page 395). The ARM processor can read the values written by the ENI client processor by reading the ENI Pulsed Interrupt register. The lower sixteen bits in the FIFO Mask/Status register are mapped onto the upper sixteen bits of the ENI Pulsed Interrupt register.

The ARM processor cannot modify any values in either the FIFO Mask/Status register or the ENI Pulsed Interrupt register.

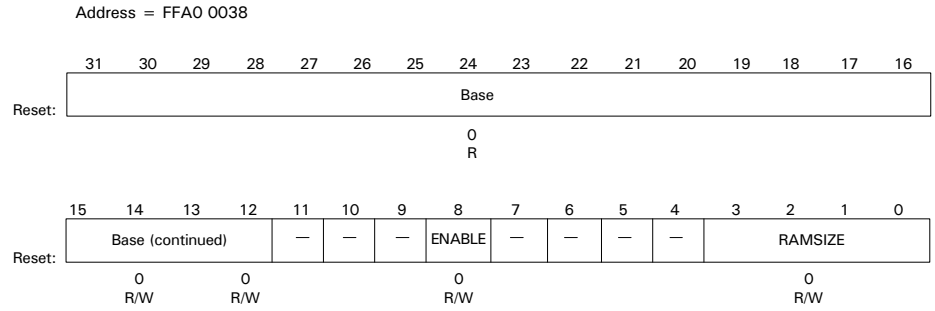
ENI Shared RAM Address register and bit definitions

The ENI Shared RAM Address register is a 32-bit read/write register. The bits in this register are set to 0 on 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 4 Kbytes to 64 Kbytes.

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 available to the ENI interface only when the ENABLE bit is set to 1. The ENABLE bit must be set only after a valid BASE and RAMSIZE are configured. Before the ENABLE bit is set, any shared RAM access from the ENI interface is completed; write data is ignored and read data is returned with all zeros, however.



Code (Bit #)	Definition of code	Description

BASE (D31:12)	Base address of shared RAM	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 value of the shared RAM location can be defined in either of these two ways:
		`& (Shared RAM) = BASE << 12;`
		`BASE = & (Shared RAM) >> 12;`
		where
		■ & implies the address of operator.
		■ >> implies the shift right operator.
		■ << implies the shift left operator.

Table 136: ENI Shared RAM Address register bit definition

Code (Bit #)	Definition of code	Description
ENABLE (D08)	Shared RAM enable	Must be set to 1 to allow the external ENI interface 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. ■ Writes to shared RAM from the external ENI interface are ignored.
RAMSIZE (D03:00)	Shared RAM physical size	0000 - 64K 1000 - 32K 1100 - 16K 1110 - 8K 1111 - 4K

Table 136: ENI Shared RAM Address register bit definition

Timing

CHAPTER 13

This chapter provides the electrical specifications, or timing, integral to the operation of the NET+50/20M chip. Timing includes information on DC and AC characteristics, output rise and fall timing, and crystal oscillator specifications.

Note: The information in this chapter applies to both the NET+50 and NET+20M chips, unless otherwise noted.

Thermal considerations

Table 137 describes the thermal requirements for the NET+50 chip:

Characteristic	Symbol	Value	Unit
Thermal resistance – junction to ambient	θ_{JA}	PQFP: 31 BGA: 37	ºC/W
Operating junction temperature	T_J	Min: -40 Max: +100	ºC
Operating ambient temperature	T_A	Min: -40 Max: +85	ºC
Storage temperature	T_{STG}	Min: -60 Max: +150	ºC
Core dynamic power @ V_{DD} (max)	P_{INT}	NET+50 chip max: 7.36 NET+20M chip max: 4.95	mW/MHz
I/O dynamic power @ V_{CC} (max)	$P_{I/O}$	NET+50 chip max: 5.18 NET+20M chip max: 4.95	mW/MHz

Table 137: Thermal considerations

The average chip junction temperature (T_J) in ºC is calculated as follows:

$T_J = T_A + (P_D * \theta_{JA})$

where:

T_A is the ambient temperature.

θ_{JA} is the package thermal resistance, junction-to-ambient, ºCW.

P_D is the calculated power consumption.

Power consumption is calculated by the following equation:

$P_D = (P_{INT} * F_P) + P_{I/O} * F_P * S_P$

where:

S_P is the estimated number of output pads switching simultaneously.

F_P is the operating frequency of outputs.

Absolute maximum ratings

Table 138 identifies the maximum values for the voltages that the NET+50 chip can withstand without being damaged.

Sym	Parameter	Conditions	Min	Max	Unit
V_{DD3}	DC supply voltage 2.5V core	Core	-0.3	3.15	V
V_{CC}	DC supply voltage 3.3V I/O	Standard I/Os	-0.3	4.00	
V_I	DC input voltage		-0.3	4.50	V
V_O	DC output voltage		-0.3	4.50	V
TEMP	Operating free air	Industrial	-40	+85	ºC
T_{STG}	Storage temperature		-40	+125	ºC

Table 138: Maximum voltage ratings

DC characteristics

Be sure the NET+50 chip operates within the voltages defined in the next two tables. Operating outside these voltages can result in unpredictable behavior.

Sym	Parameter	Conditions	Value	Unit
V_{DD}	DC supply voltage 2.5V core		Min: 2.25 Typ: 2.5 Max: 2.75	V
V_{CC}	DC supply voltage I/O pads		Min: 3.0 Typ: 3.3 Max: 3.6	V
V_{IH}	Input high voltage		Min: 2.0 Max: 3.6	V
V_{IL}	Input low voltage		Min: V_{SS}-0.3 Max: 0.8	V

Table 139: DC characteristics — Inputs

Sym	Parameter	Conditions	Value	Unit
I_{IL}	Input buffer	$V_{IN} = V_{CC}$	Min: -10 Max: 10	µA
	Input buffer with current input sink		Min: 99 Max: 429	
I_{IH}	Input buffer	$V_{IN} = V_{SS}$	Min: -10 Max: 10	µA
	Input buffer with current input source		Min: 130 Max: 352	
C_{IN}	Input capacitance	Any input	7	pF
V_T	Input switching threshold voltage	Any input	Min: 1.4 Typ: 2.0	V
P_D	V_{CC} power dissipation (@ 3.6V)	$F_{SYSCLK} = 44MHz$	Typ: 152 Max: 228	mW
P_D	V_{DD} power dissipation (@ 2.75V)	$F_{SYSCLK} = 44MHz$	**NET + 50 chip** Typ: 216 Max: 324 **NET + 20M** Typ:145 Max: 218	mW

Table 139: DC characteristics — Inputs

Sym	Parameter	Conditions	Min	Unit
V_{OL}	Output low voltage	Type: 1, $I_{OL} = 2$ mA;	Min: 0 Max: 0.4	V
		Type: 2, $I_{OL} = 4$mA;	Min: 0 Max: 0.4	V
		Type: 3, $I_{OL} = 8$mA;	Min: 0 Max: 0.4	V

Table 140: DC characteristics — Outputs

Sym	Parameter	Conditions	Min	Unit
V_{OH}	Output high voltage	Type: 1, I_{OH} = 2 mA;	Min: 2.4 Max: V_{CC}	V
		Type: 2, I_{OH} = 4mA;	Min: 2.4 Ma: V_{CC}	V
		Type: 3, I_{OH} = 8mA	Min: 2.4 Max: V_{CC}	V
I_{OZ}	High-Z leakage current	$V_O = V_{SS}$	Min: -10 Max: 10	uA
I_{OS}	Output short circuit current	V_{CC} = 3.6V, $V_O = V_{CC}$	55	uA
		V_{CC} = 3.6V, $V_O = V_{SS}$	-55	uA
C_{IN}	Input capacitance	Any input	7	pF
C_{IO}	Input/output capacitance	Any I/O	7	pF

Table 140: DC characteristics — Outputs

AC characteristics

Output pad timing

The total propagation delay for each of the output timing characteristics in the NET+50 chip is a function of the intrinsic delay of the path, the pad cell type, and the load capacitance.

The total propagation delay is the sum of the intrinsic delay time and the load-dependent delay time.

- Intrinsic delay is the delay of the path with no external load. The values shown in the timing diagram tables are intrinsic delay values.
- Load-dependent delay time is a function of the output capacitive load attached to the NET+50 chip output pin and the drive strength of the output buffer type used on that pin. The load-dependent factors for

propagation delay on the low-to-high transition of the output (T_{PLH}) may be different than the factors for the high-to-low transition (T_{PHL}).

Table 141 shows the factors for each pad type. The propagation delay times are calculated as follows:

```
TPLH or TPHL = Intrinsic time + (load-dependent delay factor * CL)
```

C_L is the output load capacitance attached to the NET+50 chip output pin.

Pad Cell	Load-dependent delay factor (ns/pF) for T_{PLH}	Load-dependent delay factor (ns/pF) for T_{PHL}
pc25d00u	0.178	0.123
pt33b01	0.168	0.160
pt33b01u	0.163	0.161
pt33b02	0.084	0.080
pt33b02u	0.083	0.080
pt33b03	0.042	0.040
pt33b03u	0.042	0.040
pt33d00	0.025	0.0144
pt33d00d	0.025	0.0144
pt33d00u	0.025	0.0144
pt33t01	0.168	0.160
pt33t02	0.084	0.080
pt33t03	0.042	0.084

Table 141: Output load-dependent delay factors

Clock relationships

Table 142 shows the relationship between the various clocks on the NET+50 chip.

Num	Characteristic	Symbol	Min	Max	Unit
1	On-chip VCO system frequency	F_{SYSCLK}	DC	44.3	MHz
2	VCO system cycle time	T_{SYS}	$1/F_{SYSCLK}$		ns
3	Crystal frequency	$F_{CRYSTAL}$	5	20	MHz
4	CPU core frequency	F_{CPU}		FSYSCLK	MHz
5	BCLK frequency	F_{BCLK}	$F_{SYSCLK}/4$	F_{SYSCLK}	MHz

Table 142: Operating frequencies

RESET* timing

Table 143 describes the values shown in Figure 60, "RESET* timing."

Num	Characteristic	Min	Max	Unit
7	VDD at 3.0V to RESET* high	40		ms
8	RESET* pulse width low	$10/F_{SYSCLK}$		µs
9	RESET* low to VDD below 3.0V	$8/F_{SYSCLK}$		µs

Table 143: RESET timing values

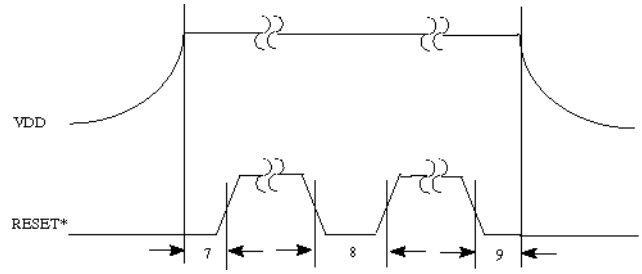


Figure 60: RESET* timing

SRAM timing

Table 144 describes the values shown in the SRAM timing diagrams (Figure 61 through Figure 69).

Num	Characteristic	Min	Max	Unit
31	BCLK high to RW* valid		19	ns
32	BCLK high to BE* valid		16	ns
33	BCLK high to Address valid	4	15	ns
34	BCLK high to Data Out high impedance		17	ns
35	BCLK high to Data Out valid		18	ns
36a	Data In valid to BCLK high (setup)	8		ns
36b	BCLK high to Data In invalid (hold)	0		ns
37a	TA* valid to BCLK high (setup)	8		ns
37b	BCLK high to TA* invalid (hold)	0		ns
38a	TEA* valid to BCLK high (setup)	8.5		ns
38b	BCLK high to TEA* invalid (hold)	0		ns
40	BCLK high to CS* valid		16	ns
41	BCLK low to OE* valid		14	ns
42	BCLK low to WE* valid		16	ns
43	BCLK high to TA* valid		11	ns
44	BCLK high to TEA* valid		14	ns

Table 144: SRAM timing characteristics

SRAM Sync Read (WAIT = 2)

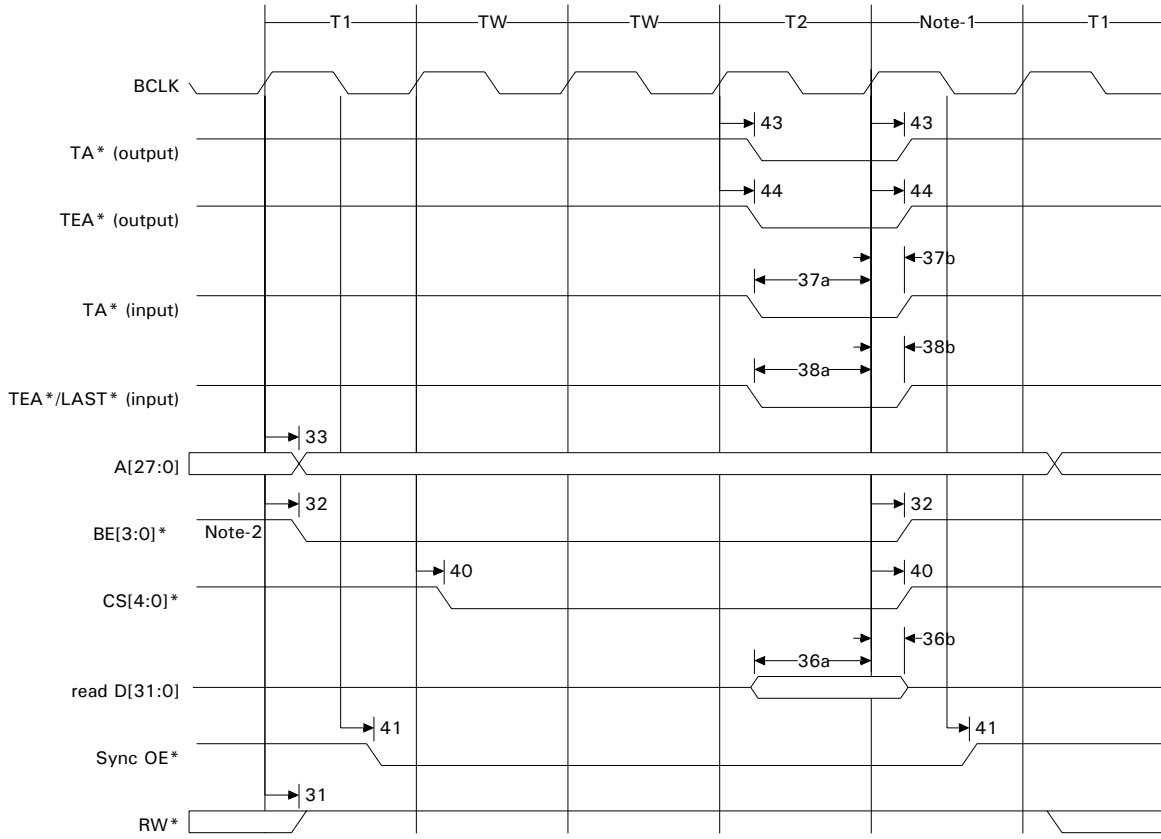


Figure 61: SRAM Synchronous Read (WAIT = 2)

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

SRAM Sync Write (WAIT = 2)

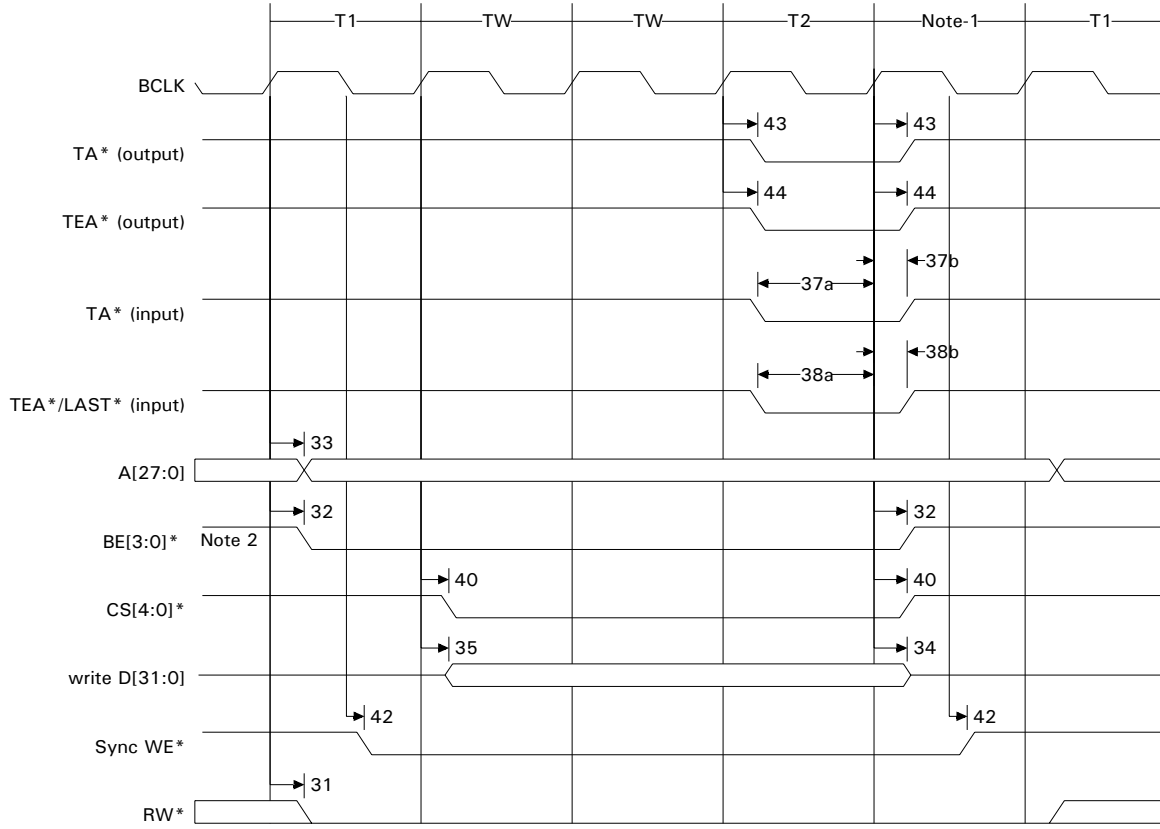


Figure 62: SRAM Synchronous Write (WAIT = 2)

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0[

SRAM Sync Burst Read (2-111) (WAIT = 0, BCYC = 00)

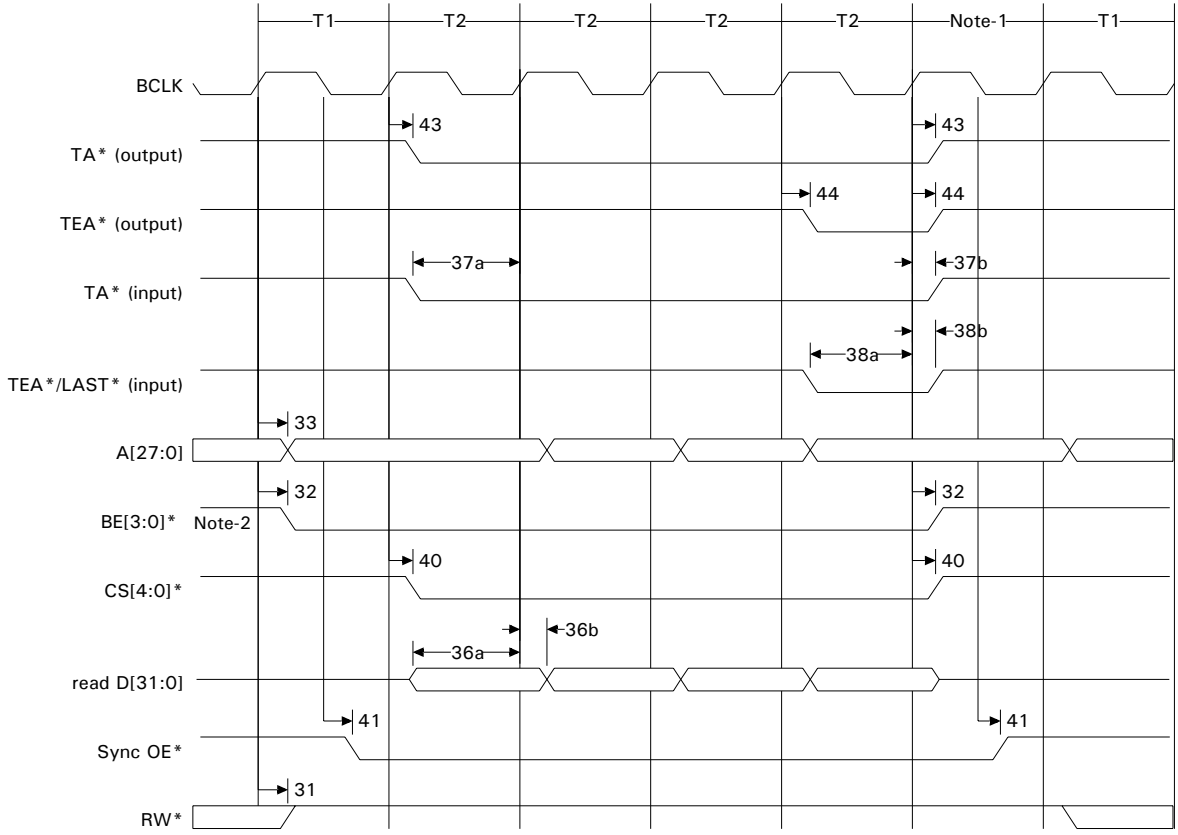


Figure 63: SRAM Synchronous Burst Read (2-111) (WAIT = 0, BCYC = 00)

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

SRAM Sync Burst Read (4-222) (WAIT = 2, BCYC = 01)

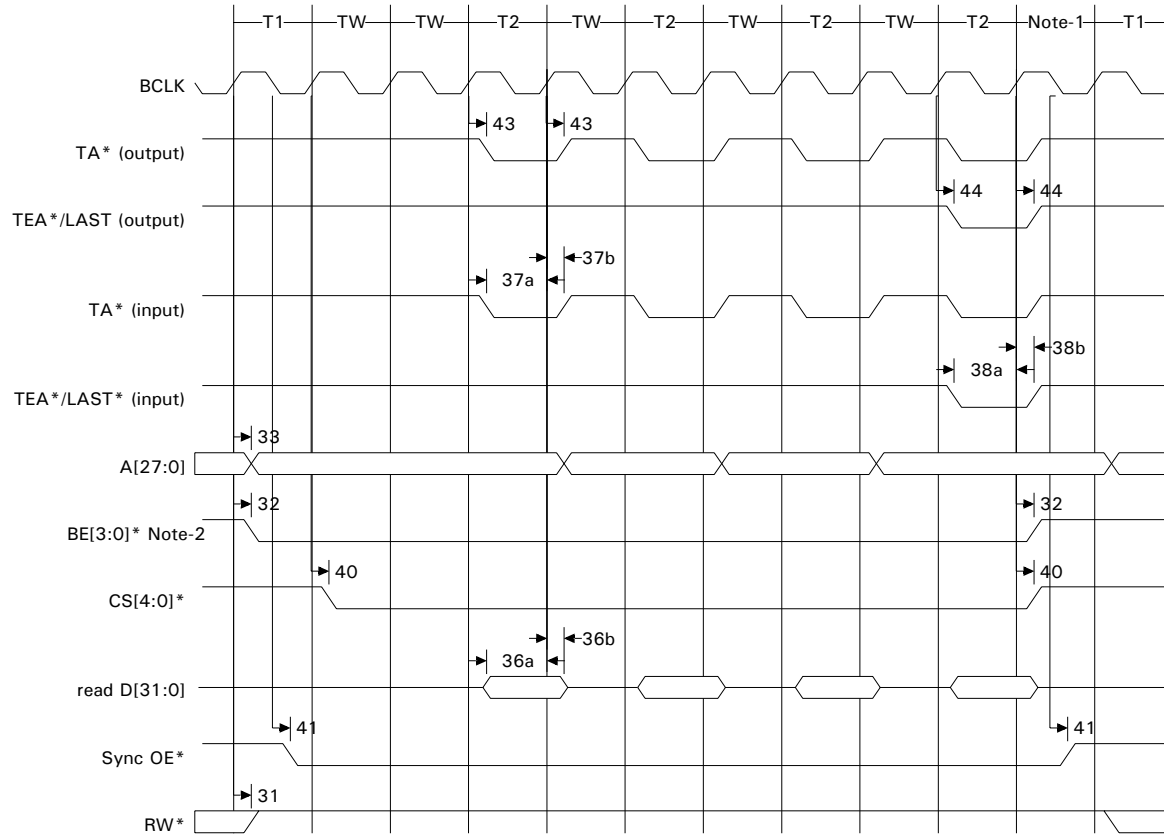


Figure 64: SRAM Synchronous Burst Read (4-222) (WAIT = 2, BCYC = 01)

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

SRAM Sync Burst Write (4-222) (WAIT = 2, BCYC = 01)

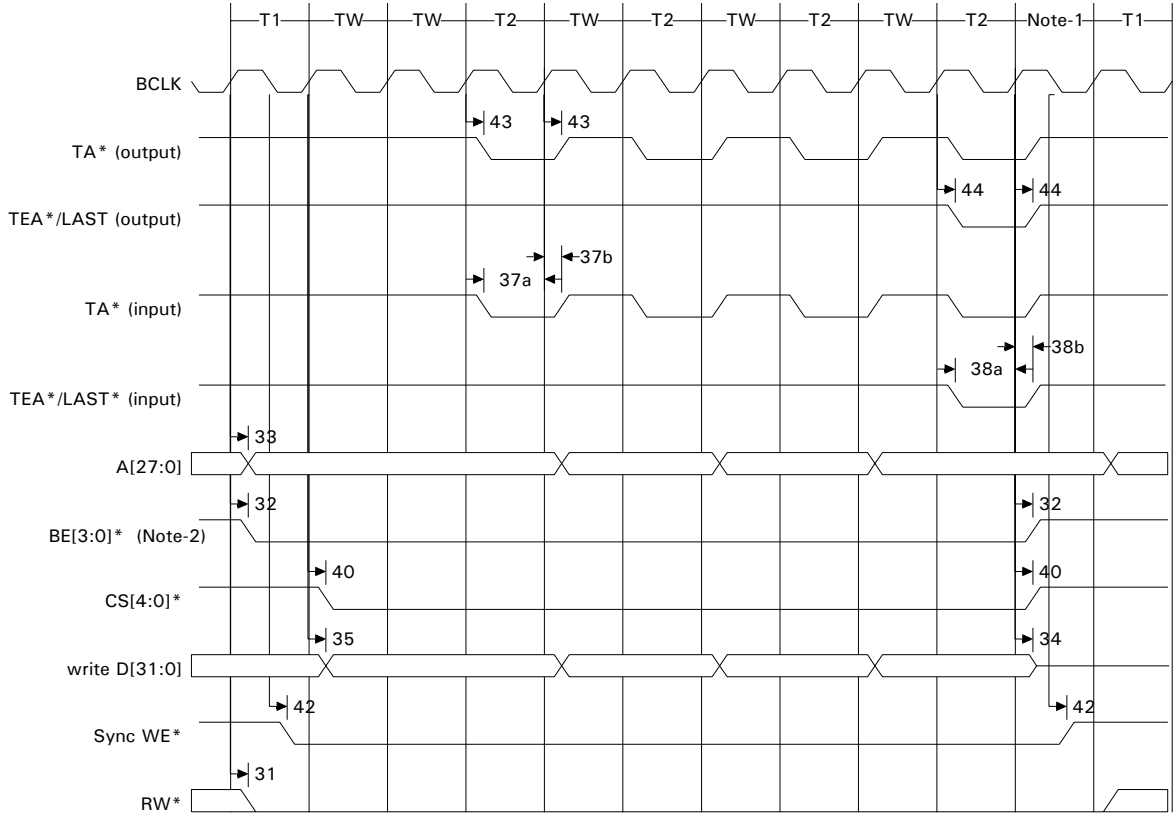


Figure 65: SRAM Synchronous Burst Write (4-222) (WAIT = 2, BCYC = 01)

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

SRAM Async Read (WAIT = 2)

The TW cycles are present when the WAIT field is set to 2 or more.

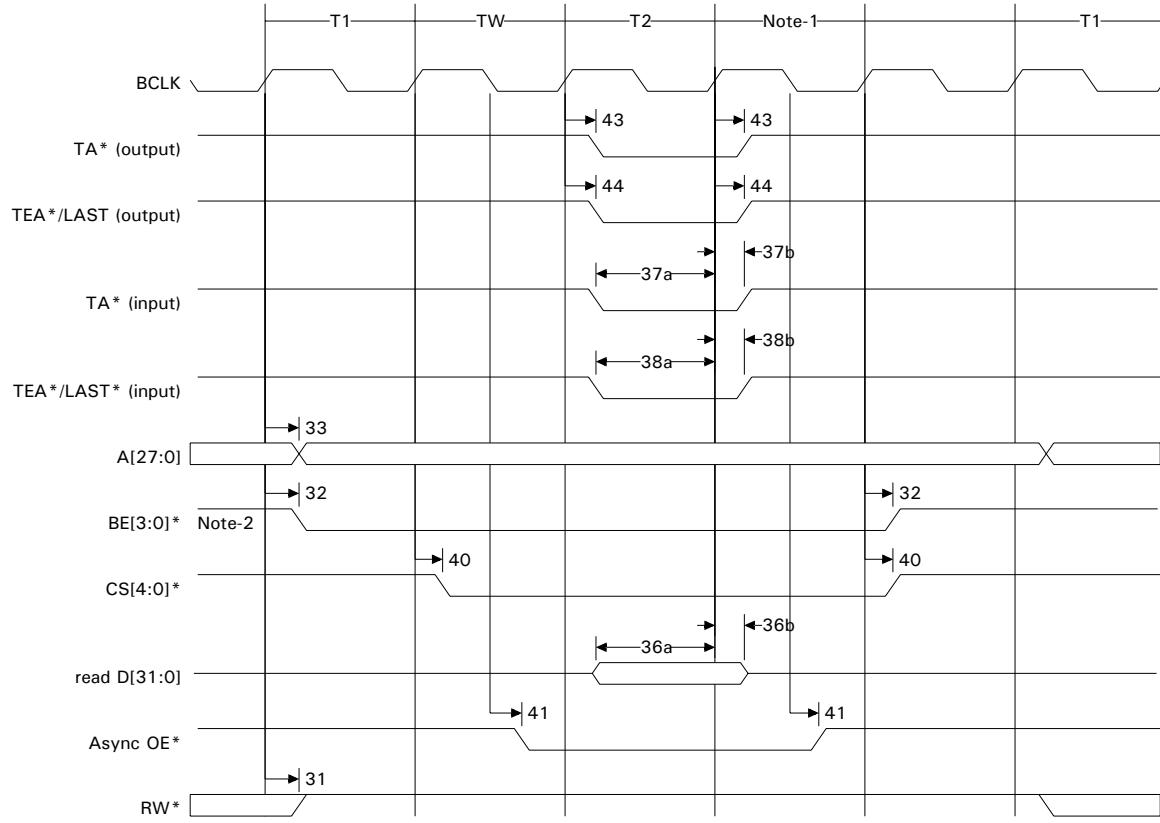


Figure 66: SRAM Asynchronous Read (WAIT = 2)

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

SRAM Async Write (WAIT = 2)

The TW cycles are present when the WAIT field is set to 2 or more.

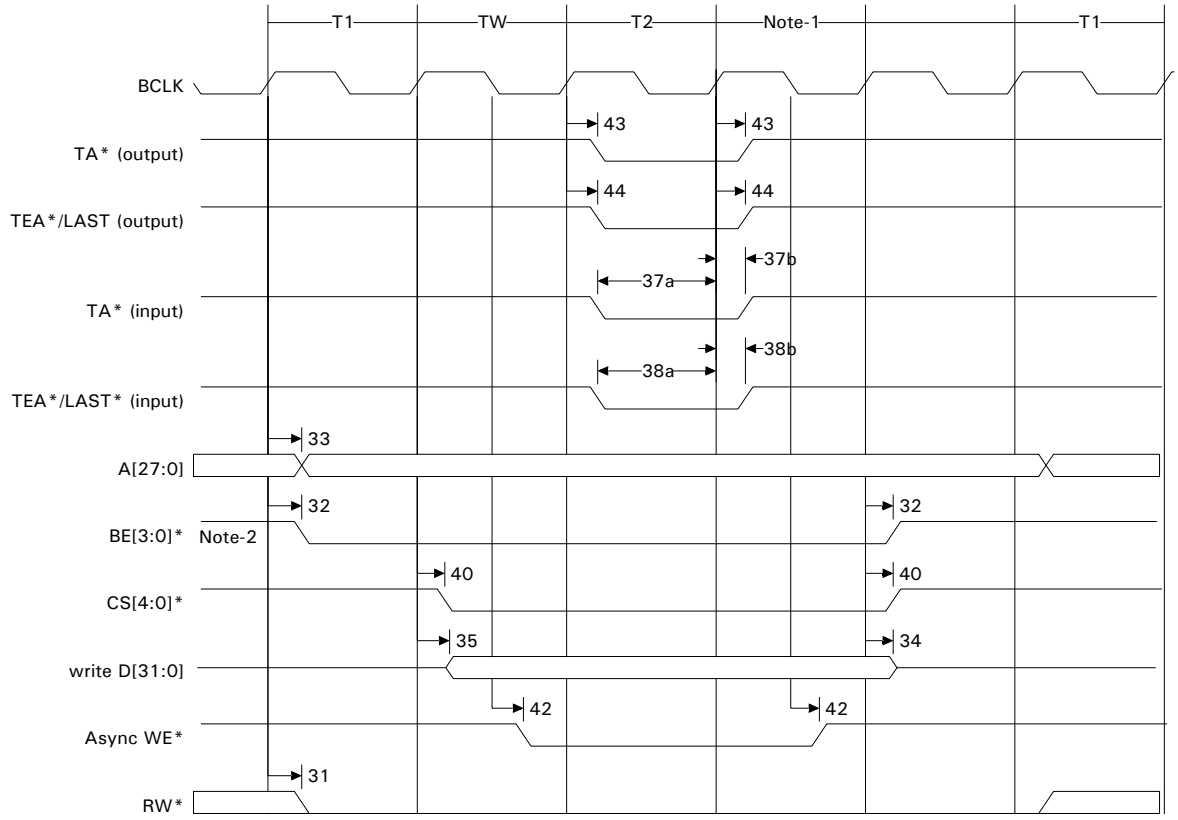


Figure 67: SRAM Asynchronous Write (WAIT = 2)

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

SRAM Async Burst Read (WAIT = 2, BCYC = 01)

The TW cycles are present when the WAIT field is set to 2 or more.

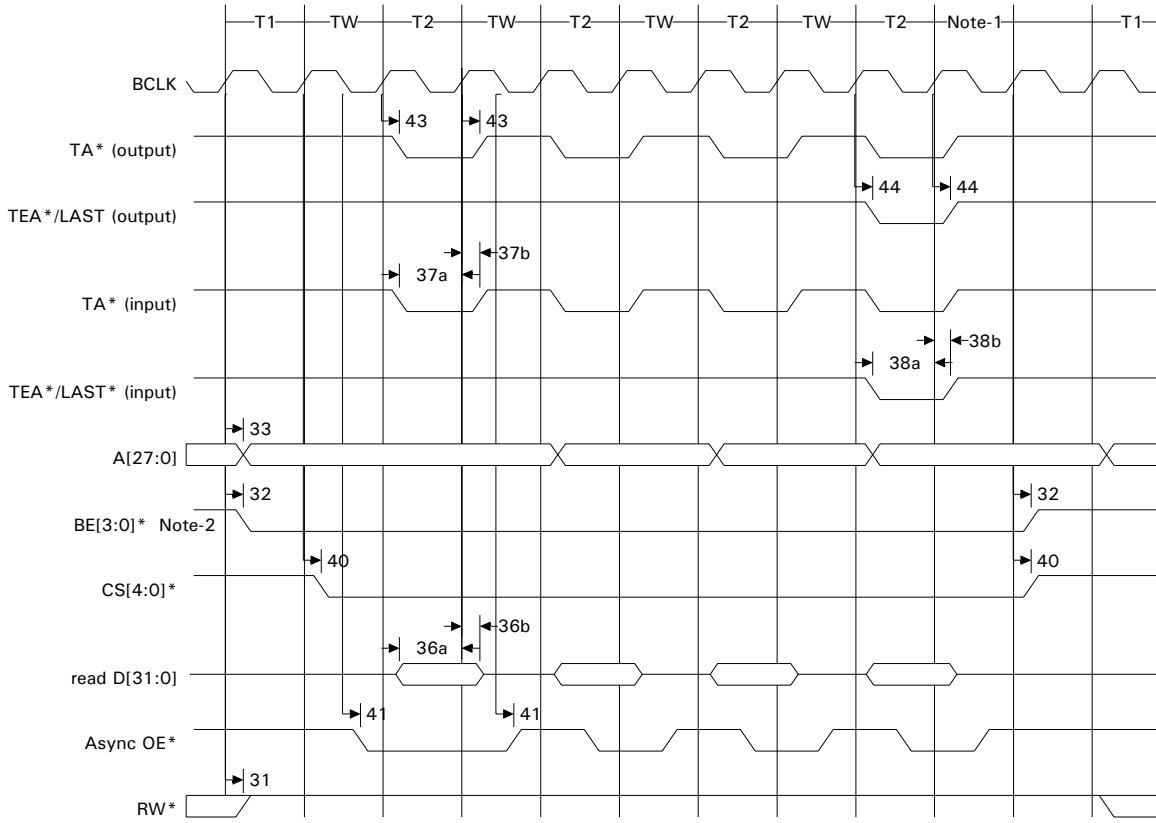


Figure 68: SRAM Asynchronous Burst Read (WAIT = 2, BCYC = 01)

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

SRAM Async Burst Write (WAIT = 2, BCYC = 01)

The TW cycles are present when the WAIT field is set to 2 or more.

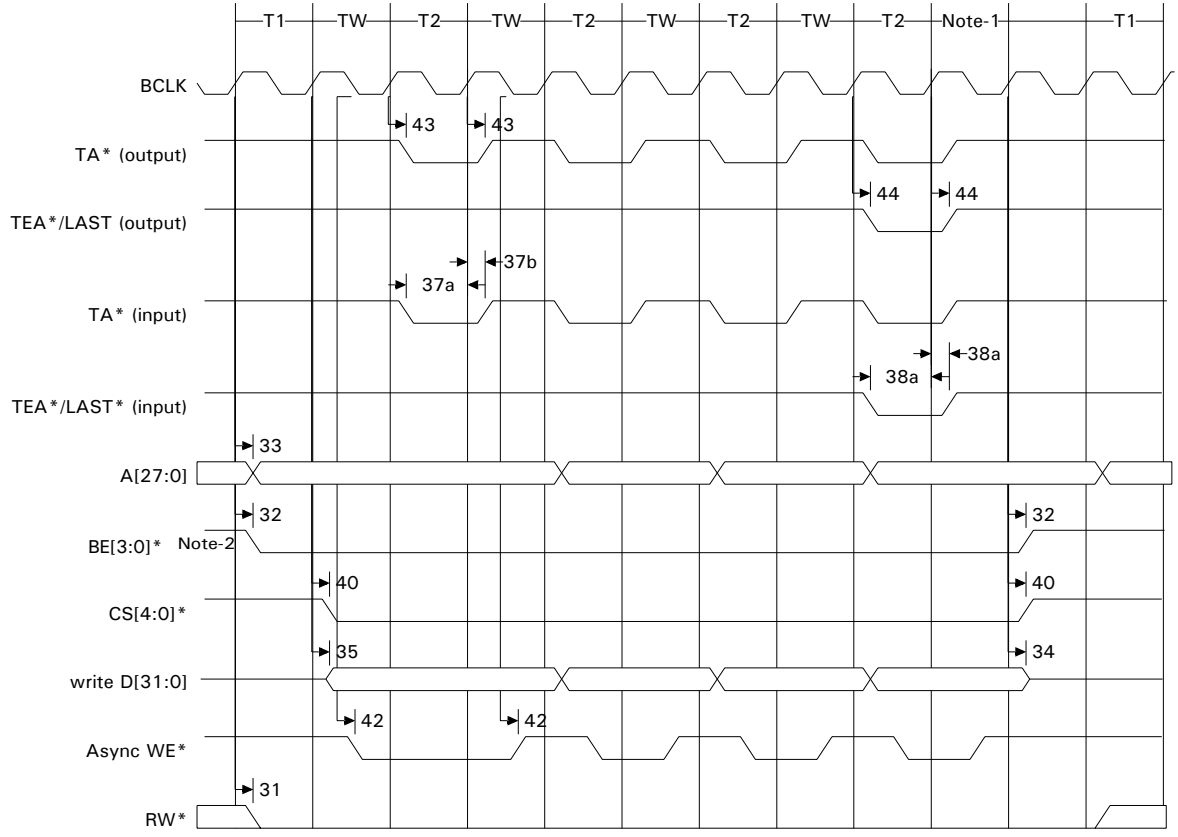


Figure 69: SRAM Asynchronous Burst Write (WAIT = 2, BCYC = 01)

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

Fast Page and EDO DRAM Timing

Table 145 describes the values shown in the fast page (FP) and EDO DRAM timing diagrams (Figure 70 through Figure 77).

Num	Characteristic	Min	Max	Unit
31	BCLK high to RW* valid		19	ns
32	BCLK high to BE* valid		16	ns‘
33	BCLK high to Address valid	4	15	ns
34	BCLK high to Data Out high impedance		17	ns
35	BCLK high to Data Out valid		18	ns
36a	Data In valid to BCLK high (setup)	8		ns
36b	BCLK high to Data In invalid (hold)	0		ns
37a	TA* valid to BCLK high (setup)	8		ns
37b	BCLK high to TA* invalid (hold)	0		ns
38a	TEA* valid to BCLK high (setup)	8.5		ns
38b	BCLK high to TEA* invalid (hold)	0		ns
41	BCLK low to OE* valid		14	ns
42	BCLK low to WE* valid		16	ns
43	BCLK high to TA* valid		11	ns
44	BCLK high to TEA* valid		14	ns
45	BCLK high to PORTC3/AMUX valid		8	ns
46a	BCLK low to CAS* valid		13	ns
46b	BCLK high to CAS* valid		13	ns
47	BCLK low to RAS* valid		16	ns

Table 145: Fast Page/EDO timing characteristics

Fast Page and EDO DRAM Read

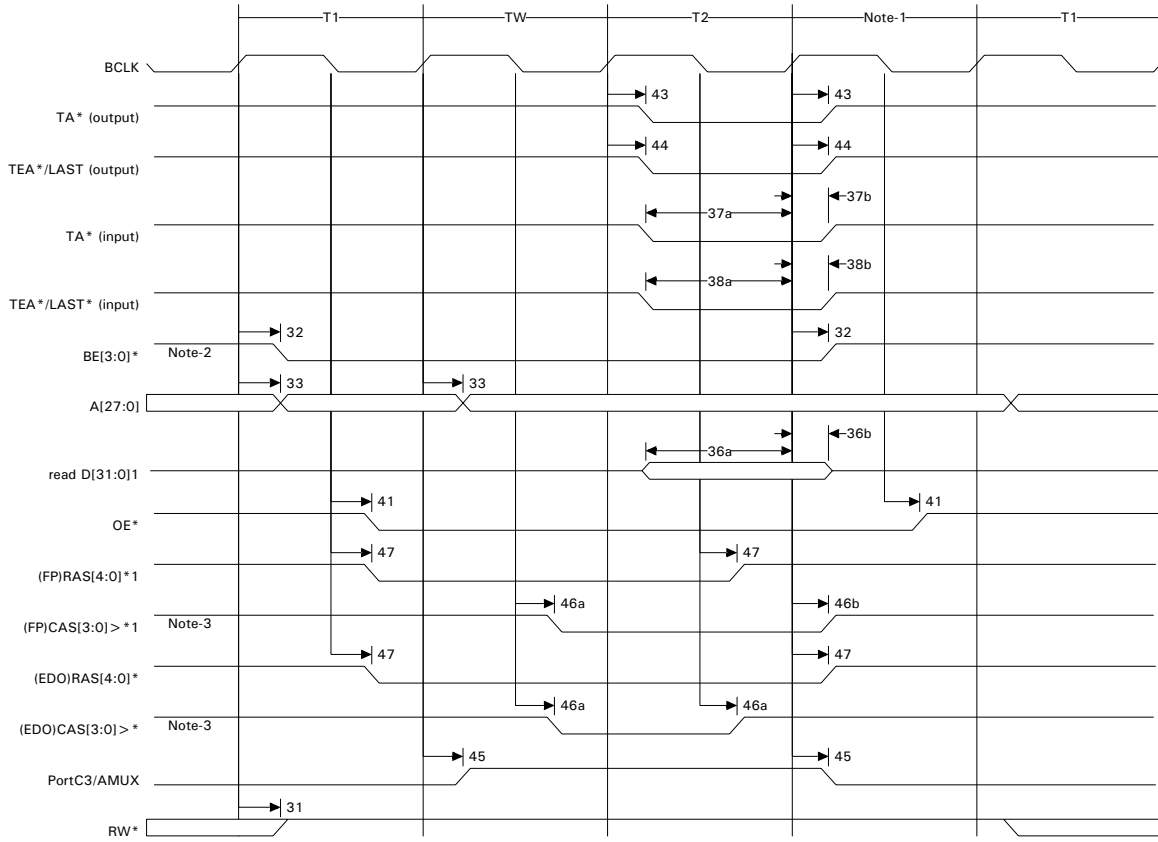


Figure 70: Fast Page and EDO DRAM Read

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

3 Port size determines which CAS* signals are active:
- 8-bit port = CAS3*
- 16-bit port = CAS[3:2]
- 32-bit port = CAS[3:0]

Fast Page and EDO DRAM Write

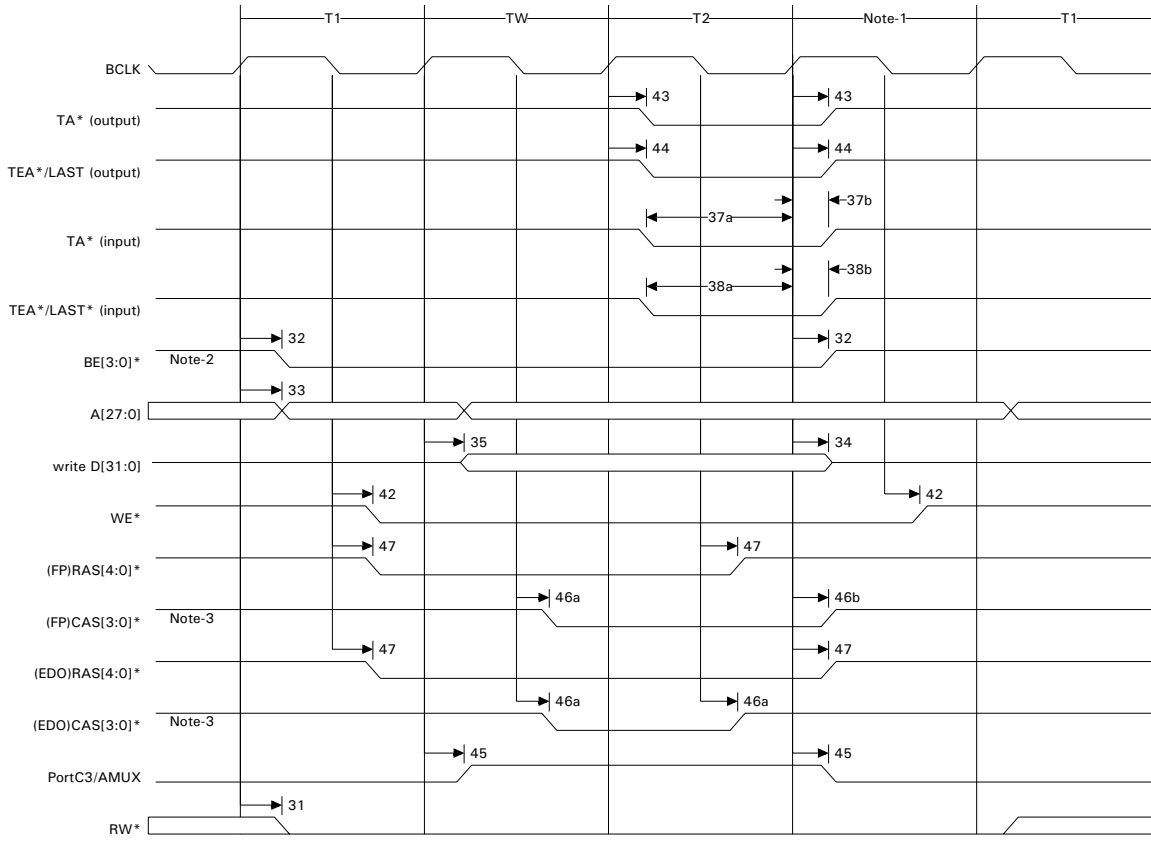


Figure 71: Fast Page and EDO DRAM Write

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

3 Port size determines which CAS* signals are active:
- 8-bit port = CAS3*
- 16-bit port = CAS[3:2]
- 32-bit port = CAS[3:0]

Fast Page and EDO DRAM Burst Read

Caution: The BCYC field in the Chip Select Option register should never be set to 00 for Fast Page/EDO DRAM.

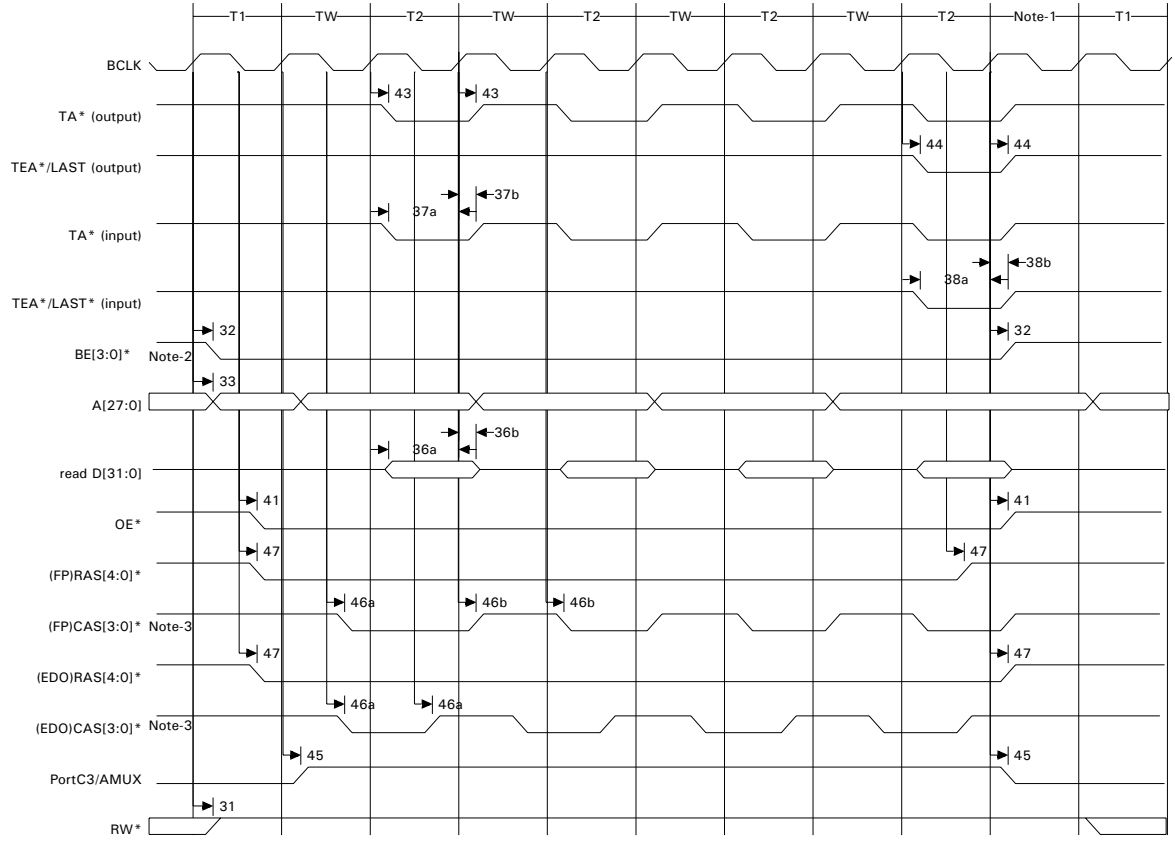


Figure 72: Fast Page and EDO DRAM Burst Read

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

3 Port size determines which CAS* signals are active:
   – 8-bit port = CAS3*
   – 16-bit port = CAS[3:2]
   – 32-bit port = CAS[3:0]

Fast Page and EDO DRAM Burst Write

Caution: The BCYC field in the Chip Select Option register should never be set to 00 for Fast Page/EDO DRAM.

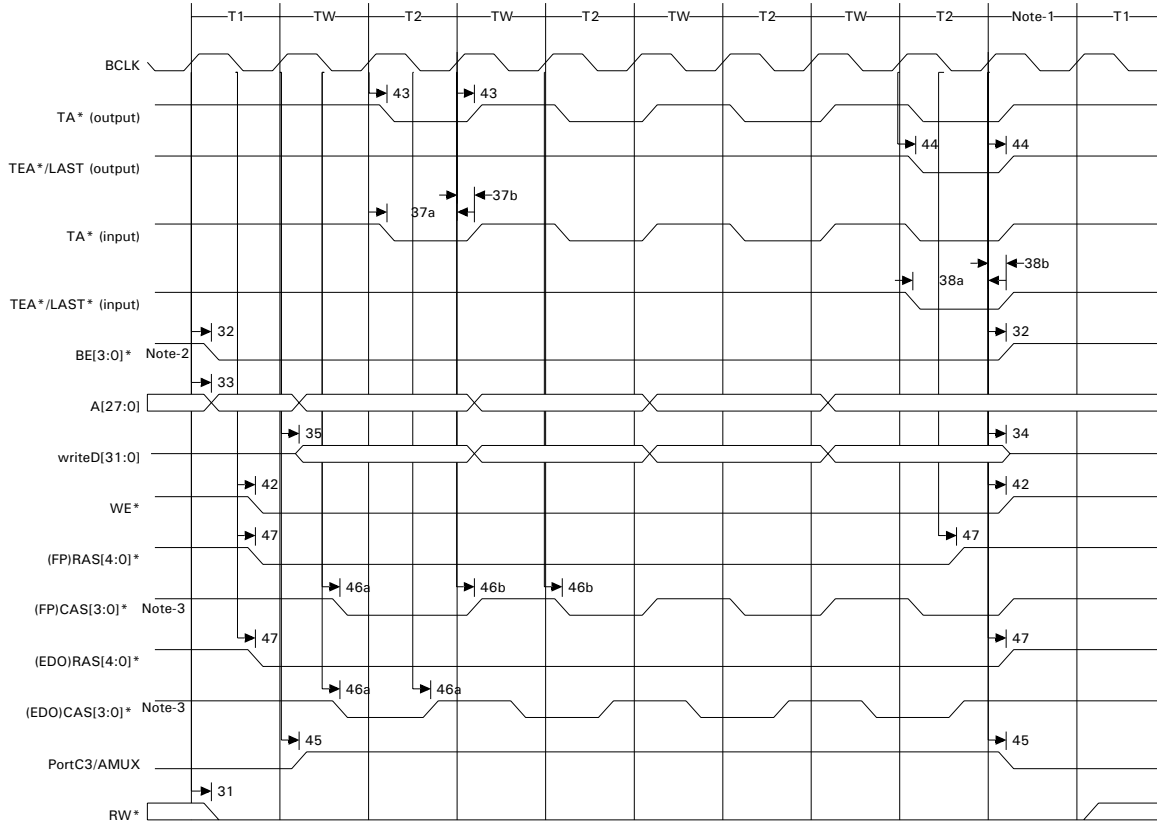


Figure 73: Fast Page and EDO DRAM Burst Write

Notes:

1 There can be 0, 1, or 2 null periods between memory transfers.

2 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

3 Port size determines which CAS* signals are active:
   – 8-bit port = CAS3*
   – 16-bit port = CAS[3:2]
   – 32-bit port = CAS[3:0]

Fast Page and EDO DRAM Refresh (RCYC = 0)

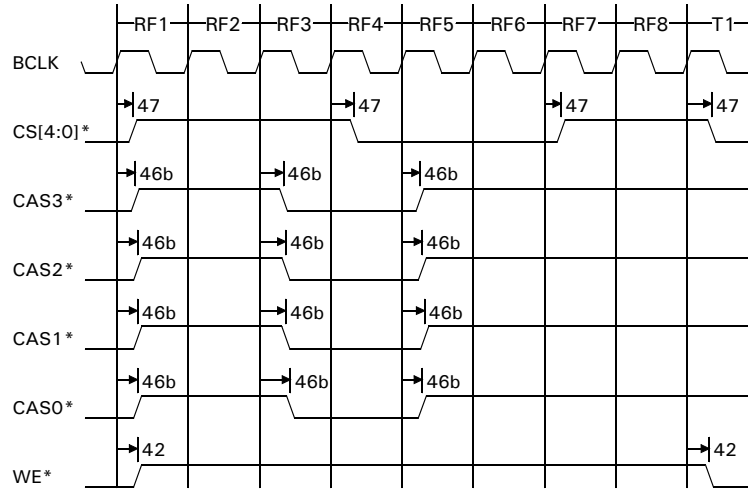


Figure 74: Fast Page and EDO DRAM Refresh (RCYC = 0)

Fast Page and EDO DRAM Refresh (RCYC = 1)

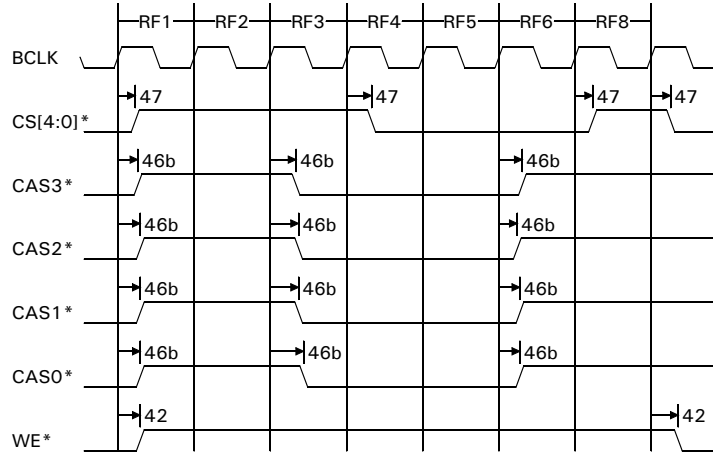


Figure 75: Fast Page and EDO DRAM Refresh (RCYC = 1)

Fast Page and EDO DRAM Refresh (RCYC = 2)

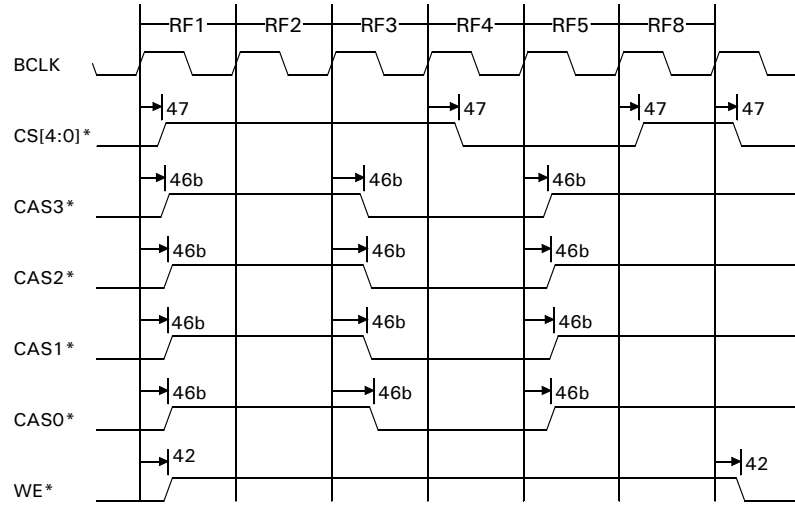


Figure 76: Fast Page and EDO DRAM Refresh (RCYC = 2)

Fast Page and EDO DRAM Refresh (RCYC = 3)

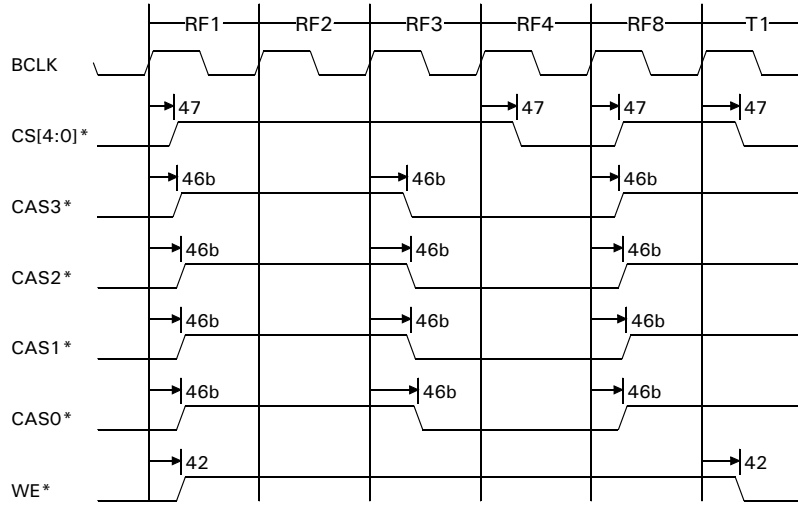


Figure 77: Fast Page and EDO DRAM Refresh (RCYC = 3)

SDRAM timing

Table 146 describes the values shown in the SDRAM timing diagrams (Figure 78 through Figure 85).

Num	Characteristic	Min	Max	Unit
31	BCLK high to RW* valid		19	ns
32	BCLK high to BE* valid		16	ns‘
33	BCLK high to Address valid	4	15	ns
34	BCLK high to Data Out high impedance		17	ns
35	BCLK high to Data Out valid		18	ns
36a	Data In valid to BCLK high (setup)	8		ns
36b	BCLK high to Data In invalid (hold)	0		ns
37a	TA* valid to BCLK high (setup)	8		ns
37b	BCLK high to TA* invalid (hold)	0		ns
38a	TEA* valid to BCLK high (setup)	8.5		ns
38b	BCLK high to TEA* invalid (hold)	0		ns
40	BCLK high to CS* valid		16	ns
41	BCLK low to OE* valid		14	ns
42	BCLK low to WE* valid		16	ns
43	BCLK high to TA* valid		11	ns
44	BCLK high to TEA* valid		14	ns

Table 146: SDRAM timing values

SDRAM Read (CAS Latency = 1)

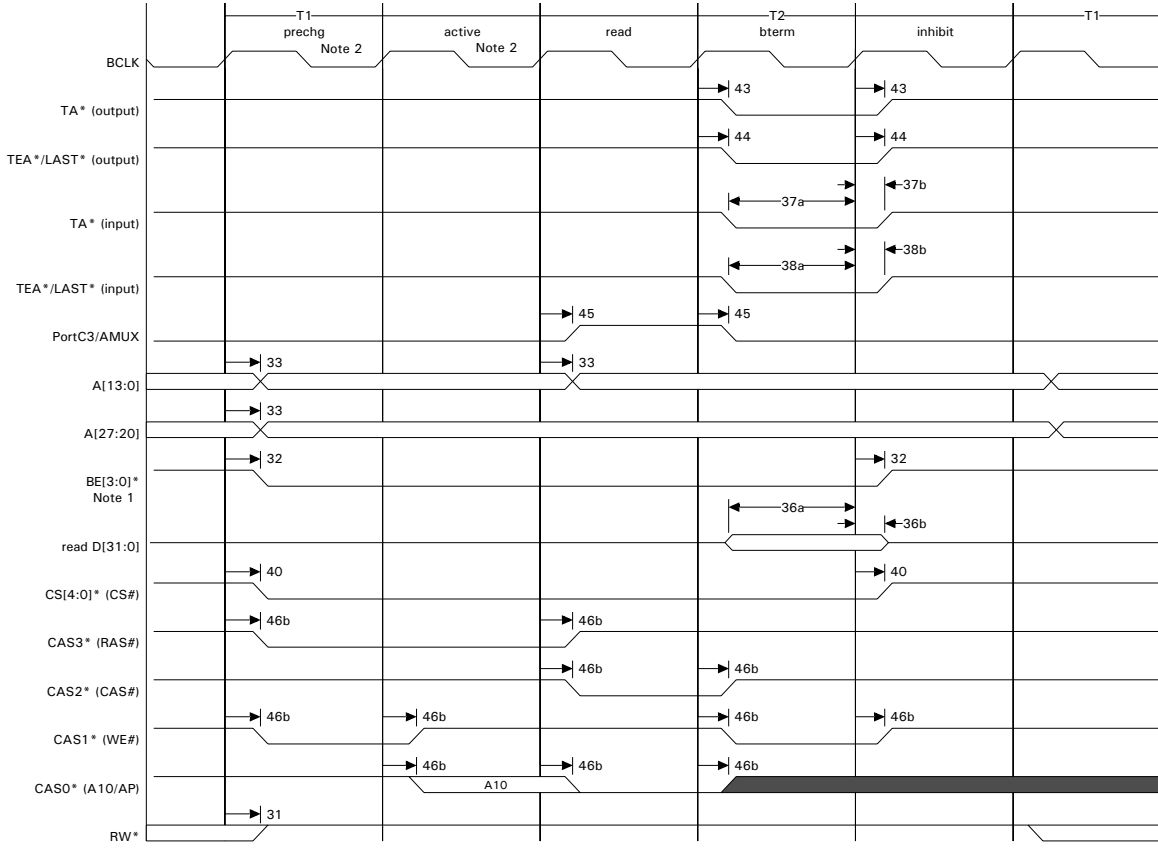


Figure 78: SDRAM Read (CAS Latency = 1)

Notes:

1 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

2 The precharge and active commands are not always present. Their presence depends on the address of the previous SDRAM access.

SDRAM Read (CAS Latency = 2)

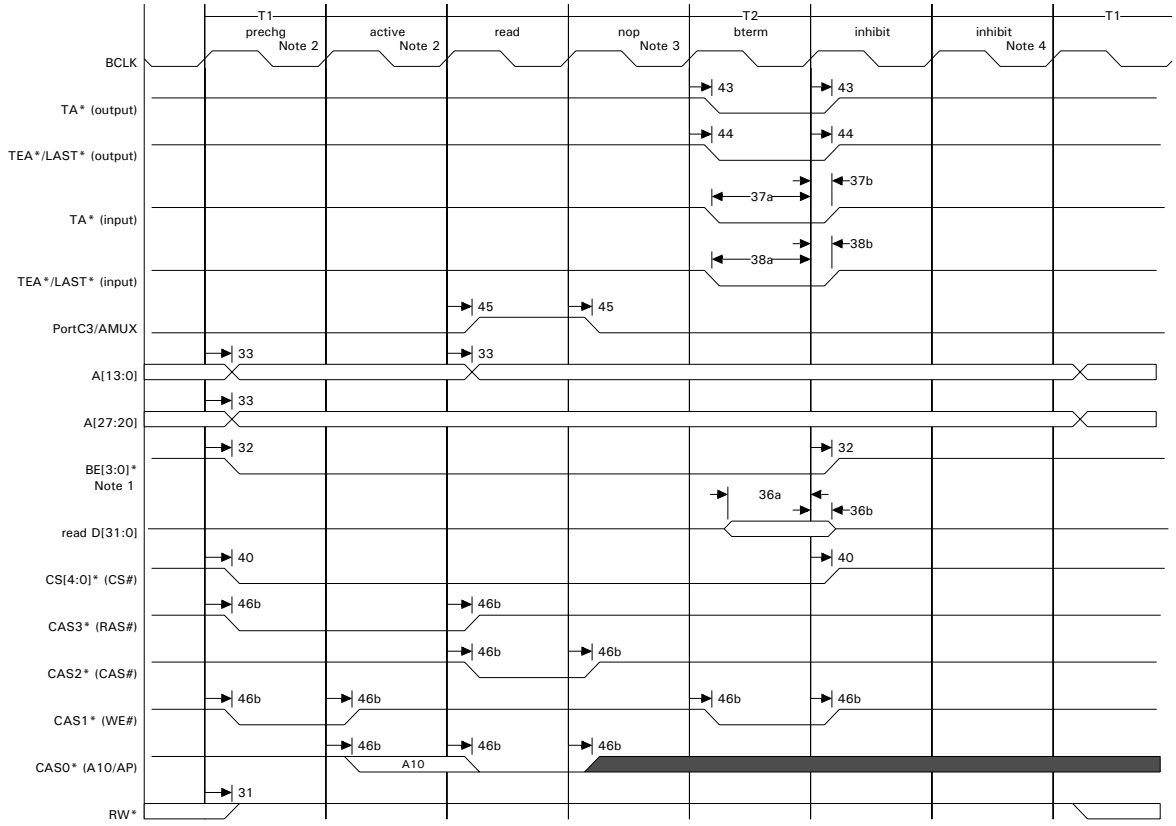


Figure 79: SDRAM Read (CAS Latency = 2)

Notes:

1 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

2 The precharge and active commands are not always present. Their presence depends on the address of the previous SDRAM access.

3 If CAS Latency = 3, there are 2 NOPs between the read and burst terminate commands.

4 If CAS Latency = 3, there are 3 inhibits after burst terminate.

SDRAM Write (CAS Latency = 2)

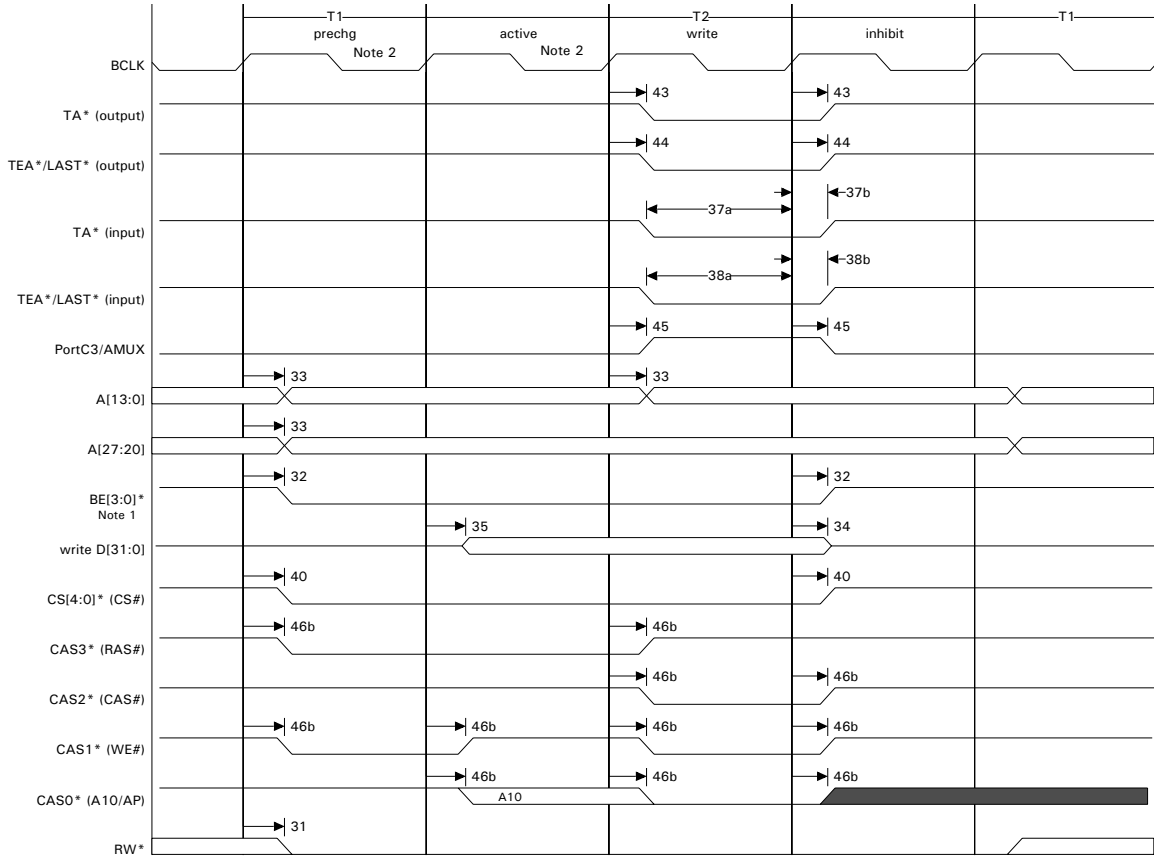


Figure 80: SDRAM Write (CAS Latency = 2)

Notes:

1 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = [3:0]

2 The precharge and active commands are not always present. Their presence depends on the address of the previous SDRAM access. When the active command is not present, parameter 35 (Write data[31:0]) is not valid until the write (T2) cycle.

SDRAM Burst Read (CAS Latency = 1)

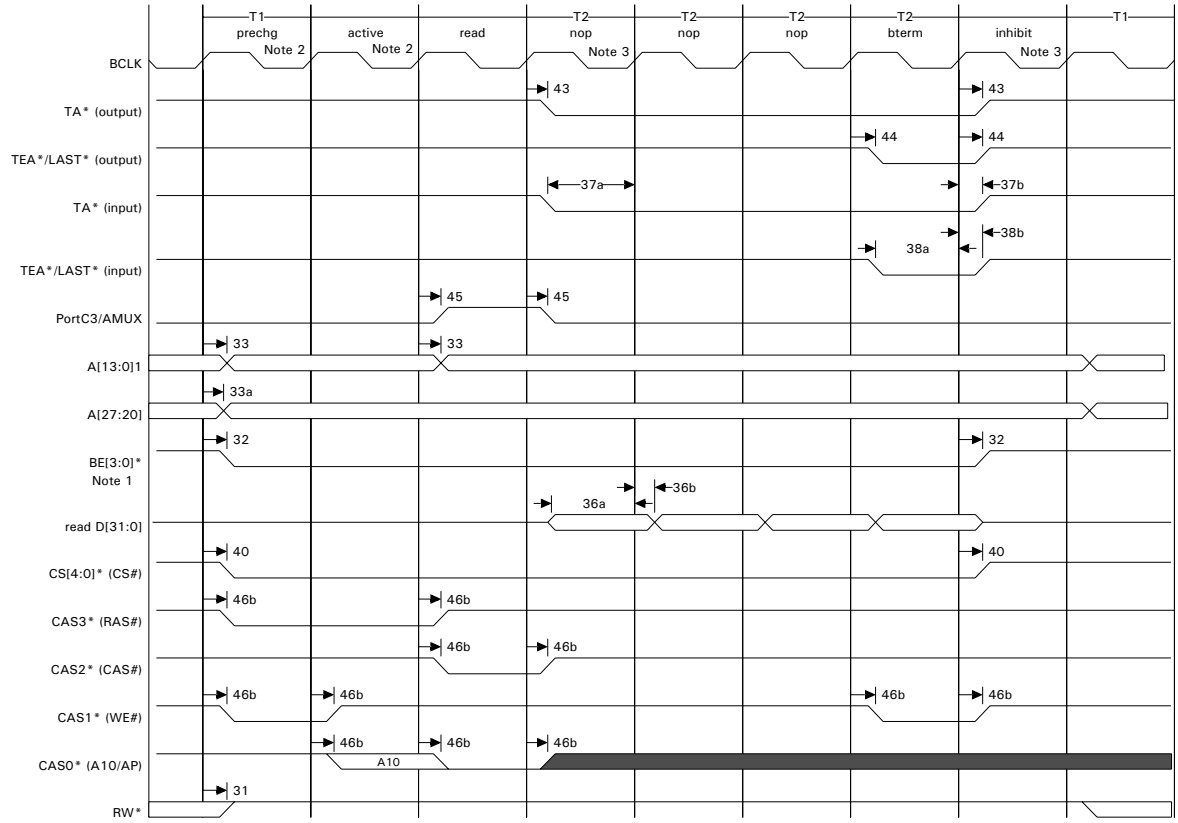


Figure 81: SDRAM Burst Read (CAS Latency = 1)

Notes:

1 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

2 The precharge and active commands are not always present. Their presence depends on the address of the previous SDRAM access.

3 If CAS Latency = 3, there are 5 NOPS between the read and burst terminate commands.

4 If CAS Latency = 3, there are 3 inhibits after burst terminate.

SDRAM Burst Read (CAS Latency = 2)

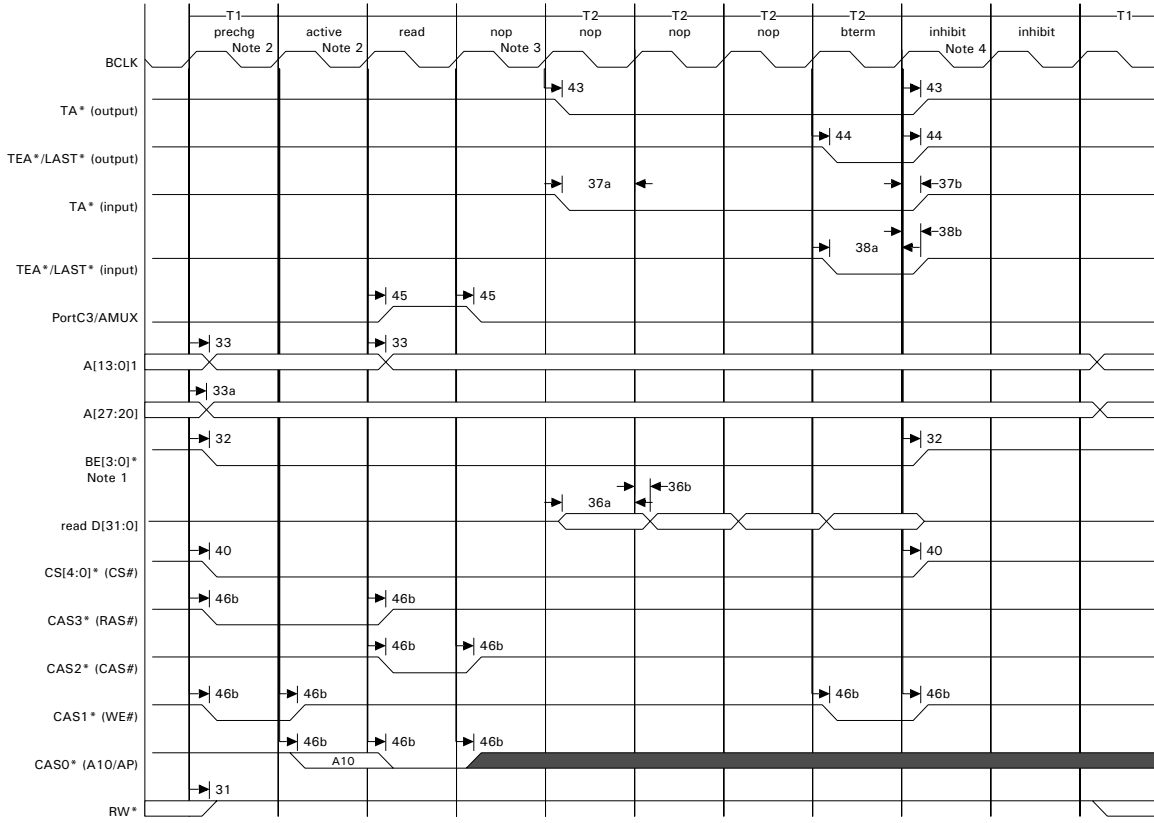


Figure 82: SDRAM Burst Read (CAS Latency = 2)

Notes:

1 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

2 The precharge and active commands are not always present. Their presence depends on the address of the previous SDRAM access.

3 If CAS Latency = 3, there are 5 NOPs between the read and burst terminate commands.

4 If CAS Latency = 3, there are 3 inhibits after burst terminate.

SDRAM Burst Write (CAS Latency = 2)

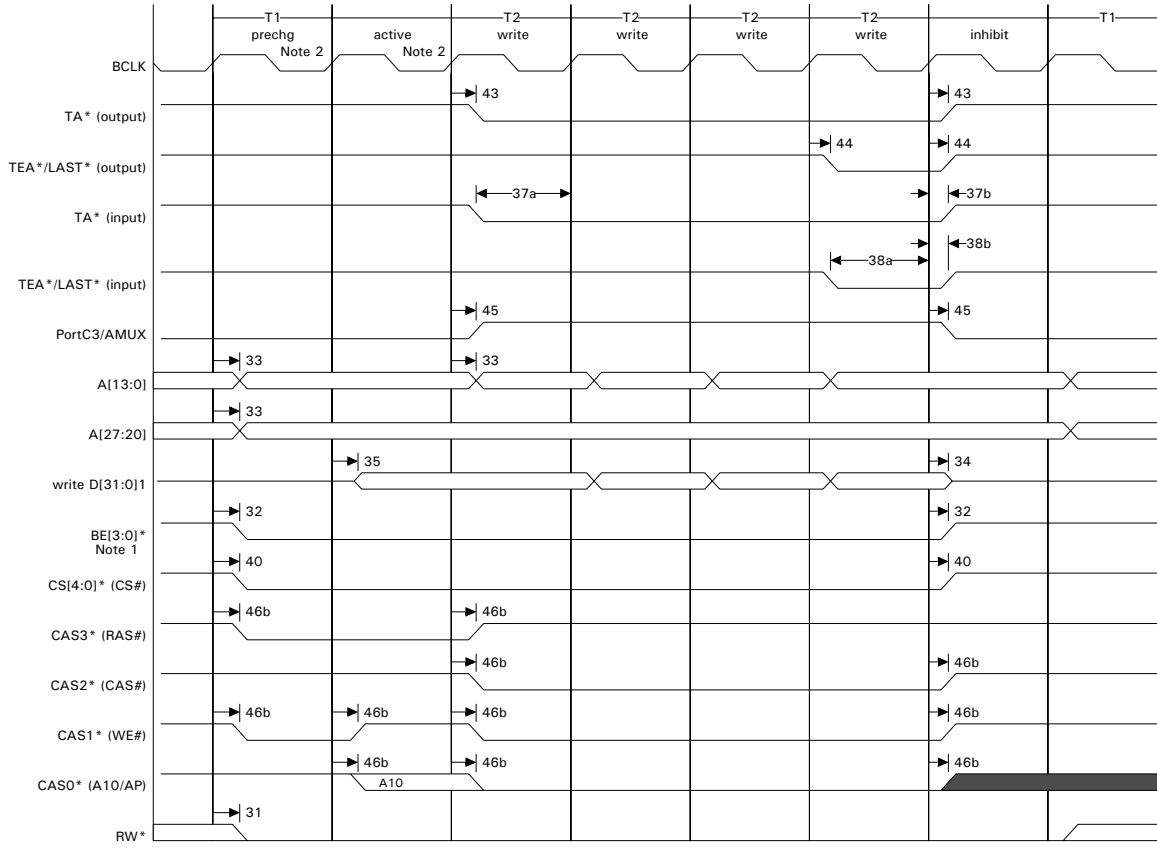


Figure 83: SDRAM Burst Write (CAS Latency = 2)

Notes:

1 Port size determines which byte enable signals are active:
- 8-bit port = BE3*
- 16-bit port = BE[3:2]
- 32-bit port = BE[3:0]

2 The precharge and active commands are not always present. Their presence depends on the address of the previous SDRAM access. When the active command is not present, parameter 35 (Write data[31:0]) is not valid until the write (T2) cycle.

SDRAM Refresh Command

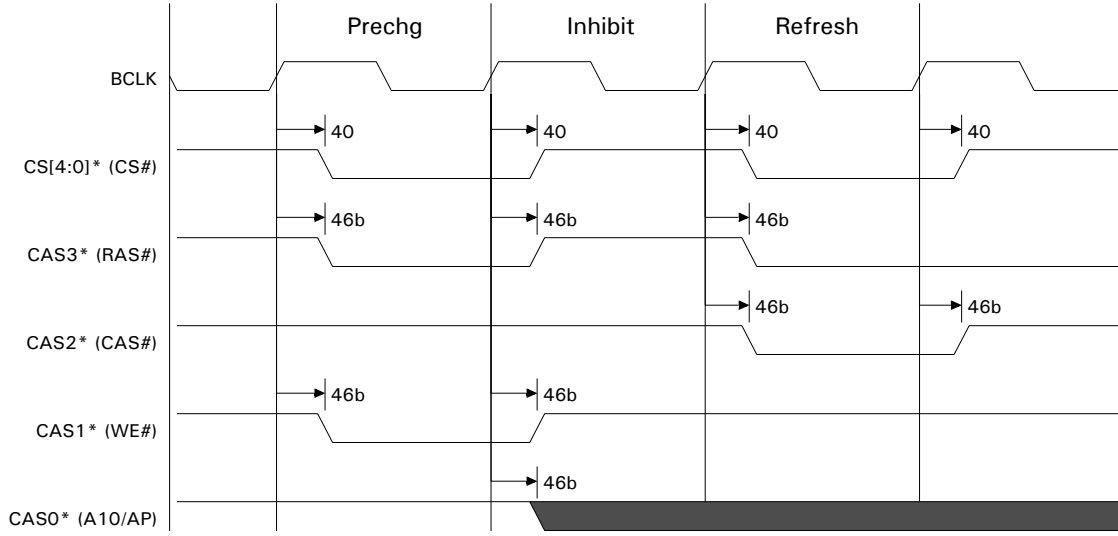


Figure 84: SDRAM Refresh

SDRAM Load-Mode Command

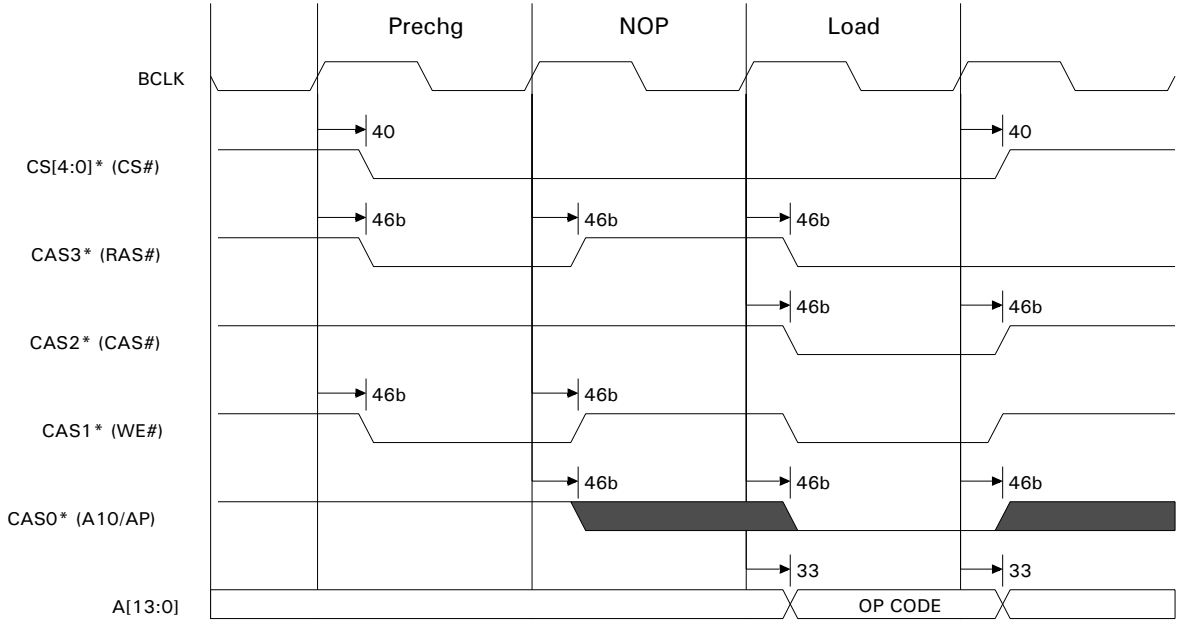


Figure 85: SDRAM Load Mode Command

External DMA timing

Table 147 describes the values shown in the external DMA timing diagrams (Figure 86 and Figure 87).

Num	Characteristic	Min	Max	Unit
50	BCLK high to DACK* valid		26	ns
51	BCLK high to DONE* (output) valid		27	ns
52	DACK* low to DREQ* high	0		ns
53a	DREQ* valid to BCLK high (setup)	11		ns
53b	BCLK high to DREQ* valid (hold)	0		ns
54a	DONE* (input) valid to BCLK high (setup)	14		ns
54b	BCLK high to DONE* (input) valid (hold)	0		ns

Table 147: External DMA timing values

External Fly-by DMA

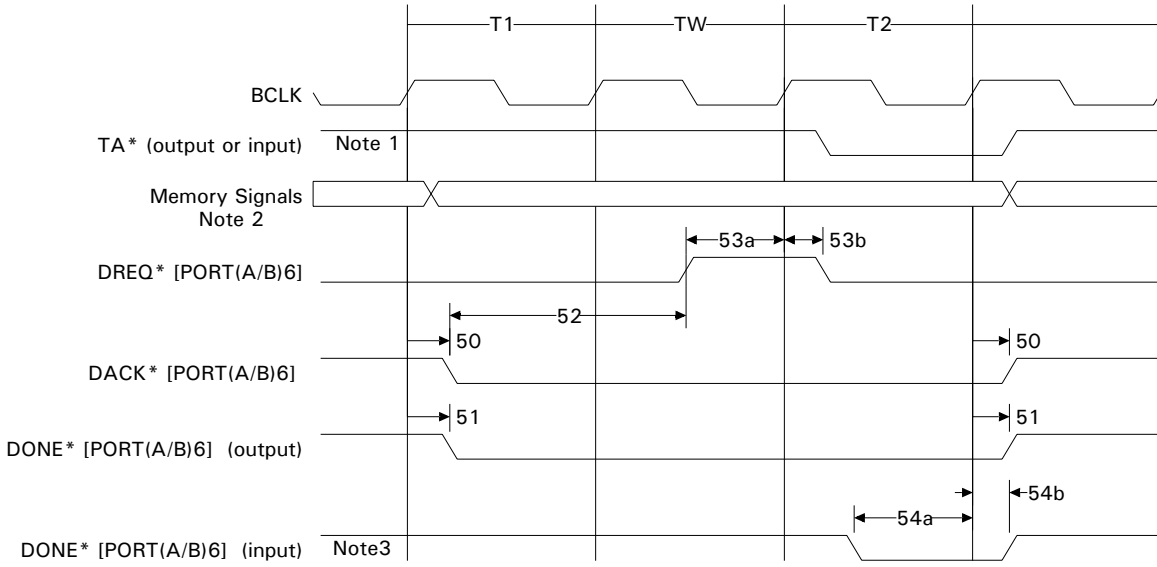


Figure 86: External Fly-by DMA

Notes:

1 TheTA* signal is shown here for reference only. See the following timing diagram sections for information about TA* timing:
- "SRAM timing," beginning on page 426.
- "Fast Page and EDO DRAM Timing," beginning on page 436.
- "SDRAM timing," beginning on page 445.

2 The memory signals consist of the following types:

DATA[31:0]	RW
ADDR[27:0]	OE*
BE[3:0]	WE*
CS/RAS[4:0]	PORTC3/AMUX
CAS[3:0]	

The timing of these signals depends on how the memory is configured. See the following sections for information:
- "SRAM timing," beginning on page 426.

- "Fast Page and EDO DRAM Timing," beginning on page 436.
- "SDRAM timing," beginning on page 445.

3 The DONE* signal works as an input only when the DMA channel is configured as fly-by write.

External Memory-to-Memory DMA

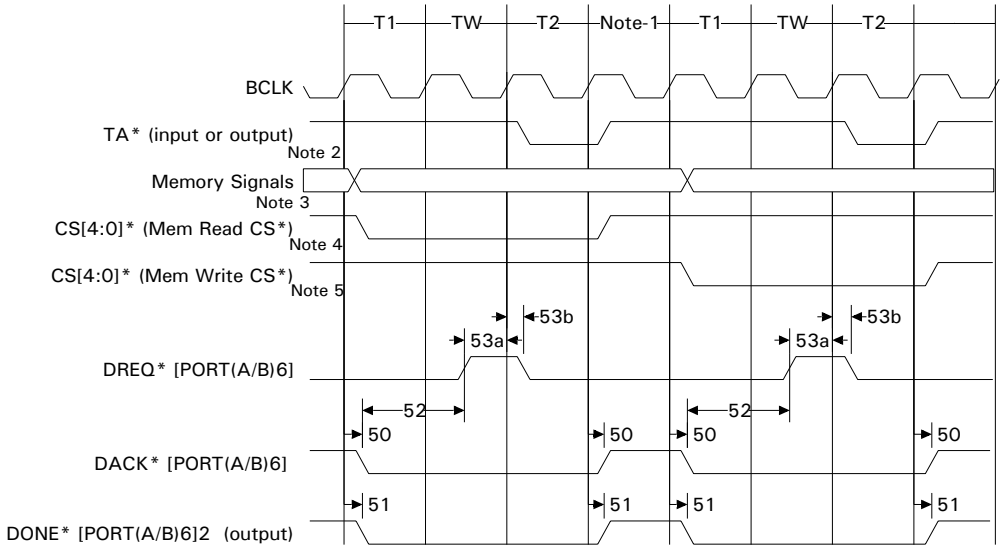


Figure 87: External Memory-to-Memory DMA

Notes:

1 There is sometimes a null period between memory cycles.

2 The TA* signal is shown here for reference only. See the following timing diagram sections for information about TA* timing:

- "SRAM timing," beginning on page 426.
- "Fast Page and EDO DRAM Timing," beginning on page 436.
- "SDRAM timing," beginning on page 445.

3 The memory signals consist of the following types:

DATA[31:0]	RW
ADDR[27:0]	OE*
BE[3:0]	WE*
CS/RAS[4:0]	PORTC3/AMUX
CAS[3:0]	

The timing of these signals depends on how the memory is configured. See the following sections for information:

- "SRAM timing," beginning on page 426.
- "Fast Page and EDO DRAM Timing," beginning on page 436.
- "SDRAM timing," beginning on page 445.

4 The timing of the chip select associated with the buffer descriptor's source address depends on how that chip select is configured.

5 The timing of the chip select associated with the buffer descriptor's destination address depends on how that chip select is configured.

SPI timing

SPI Master mode 0 and 1 (two-byte transfer)

Table 148 describes the values shown in Figure 88, "SPI Master mode 0 and 1 (two-byte transfer)."

Num	Characteristic	Min	Max	Unit
130	Data In valid to mode 0 rising clock (setup)	20		ns
131	Mode 0 rising clock to Data In valid	0		ns
132	Data In valid to mode 1 falling clock (setup)	20		ns
133	Mode 1 falling clock to Data In valid (hold)	0		ns
134	Enable low to mode 0 first rising clock		1.5	Bit-time
135	Enable low to mode 1 first falling clock		1.5	Bit-time
136	Mode 0 falling edge to Data Out valid	$-T_{SYS}$	T_{SYS}	ns
137	Mode 1 rising edge to Data Out valid	$-T_{SYS}$	T_{SYS}	ns
138	Mode 0 last falling clock to Enable high		1.5	Bit-time
139	Mode 1 last rising clock to Enable high		1.5	Bit-time
140	Enable low to Data Out valid	$-T_{SYS}$	T_{SYS}	ns

Table 148: SPI Master mode 0 and 1 timing values

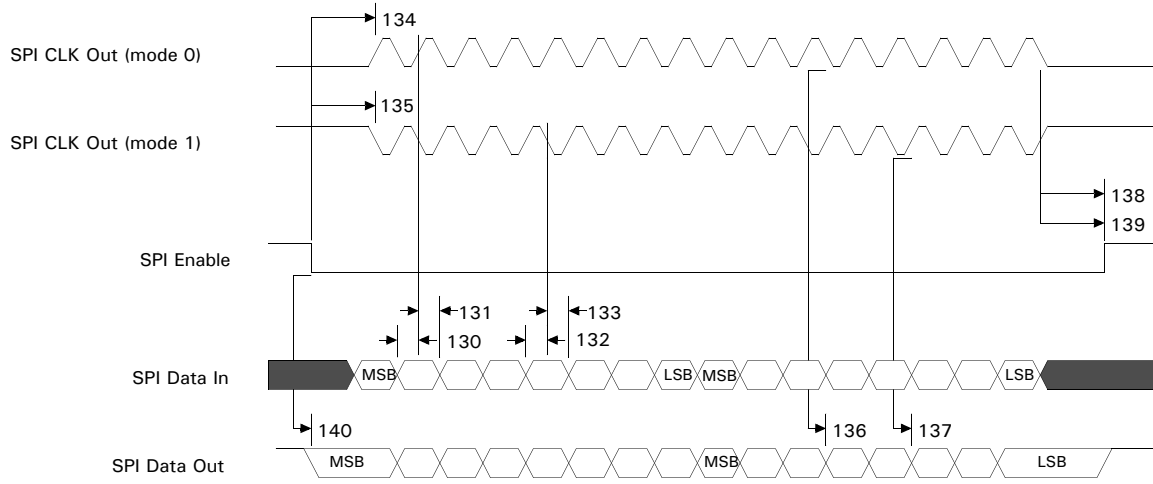


Figure 88: SPI Master mode 0 and 1 (two-byte transfer)

SPI Slave mode 0 and 1

Table 149 describes the values shown in Figure 89, "SPI Slave mode 0 and 1 (two-byte transfer)."

Num	Characteristic	Min	Max	Unit
Mode 0				
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)	3.4		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
Mode 1				
146	SPI clock low to TXD valid	$3*T_{SYS}$	$4*T_{SYS}$	ns
147	SPI Enable low to SPI clock low (setup)	0		ns
148	RXD Input valid to SPI clock low (setup)	3.4		ns

Table 149: SPI Slave mode timing values

Num	Characteristic	Min	Max	Unit
149	SPI clock low to RXD Input change (hold)	$4*T_{SYS}$		ns
150	SPI clock low to SPI Enable high (hold)	$4*T_{SYS}$		ns

Table 149: SPI Slave mode timing values

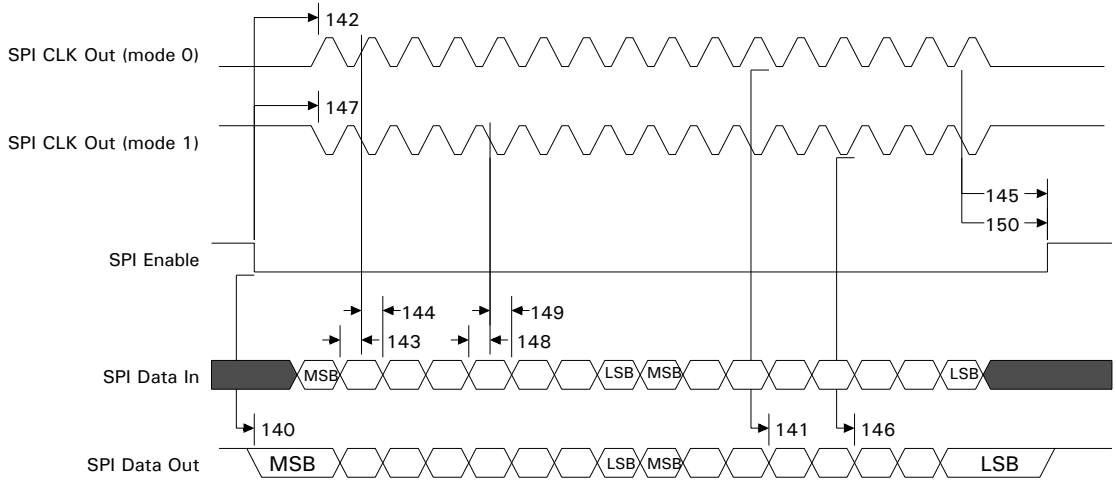


Figure 89: SPI Slave mode 0 and 1 (two-byte transfer)

MIC timing

Table 150 describes the values shown in the MIC timing diagrams (Figure 90 through Figure 92). Note that the data shown are independent of SYS_CLK and BCLK settings.

Num	Characteristic	Note	Min	Max	Unit
70a	PCS*/PDACK* low to PRW* sampled	1		T_{SYS} - 2.5	ns
70b	PCS*/PDACK* low to PRW* hold time	1	$4*T_{SYS}$		ns
71	Read Data Valid to PACK* valid		T_{SYS}		ns

Table 150: MIC timing values

Num	Characteristic	Note	Min	Max	Unit
72	PCS*/PDACK* low to PACK* low	3	T_{SYS}	$6*T_{SYS}$	ns
72a	PCS* low to PACK* low	4	$7*T_{SYS}$	Determined by shared RAM access time	ns
72b	PCS*/PACK* low to PACK* low	5	$2*T_{SYS}$	$4*T_{SYS}$	ns
72c	PCS* low to PACK* valid (shared RAM only)	6	$7*T_{SYS}$	Determined by shared RAM access time	ns
73	PCS*/PDACK* high to PDATA high impedance		0	7.84	ns
74	PCS*/PDACK* high to PACK* high		0	13	ns
75	PCS*/PDACK* low to PACK* (wait-) low		0	14	ns
76a	PCS*/PDACK* low to Write Data valid			$2*T_{SYS}$	ns
76b	PCS*/PDACK* high to Write Data hold time		0		ns
77a	PCS*/PDACK* width high (recovery)		16		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 hold time		0		ns
79	PCS* low to PINT1/2 change (write)		$3*T_{SYS}$	$5*T_{SYS}$	ns
80a	PDACK* low to PDRQI*, PDRQO* high			$5*T_{SYS}$	NS
80b	PDRQO* high width		$5*T_{SYS}$		ns
80c	PDRQI* high width		$4*T_{SYS}$		ns
80d	PDACK* low to PINT2 low			14	ns
80e	PDACK* high to PINT2 high		15		ns
81	PCS*/PDACK* low to PEN* low	2	$1*T_{SYS}$	$3*T_{SYS}$	ns
82	PCS*/PDACK* high to PEN* high	2	0	13	ns
83	PCS*/PDACK* low to PBRW* low	2	$1*T_{SYS}$	$2*T_{SYS}$	ns
84a	PRW* valid to PDACK* low (setup)	1	0		ns
84b	PDACK* high to PRW* hold time	1	0		ns

Table 150: MIC timing values

Num	Characteristic	Note	Min	Max	Unit
85	PDACK* low to PDATA valid	7	$3*T_{SYS}$	$5*T_{SYS}$	ns
85a	PDACK* low to PDATA valid	7	$1*T_{SYS}$	$2*T_{SYS}$	ns

Table 150: MIC timing values

Notes:

1 Parameters 70a and 70b apply only when the ENI FAST bit is set to 0. When ENI FAST is set to 1, parameters 84a and 84b apply.

2 The PEN* and PBRW* signals control an external bi-directional data bus transceiver for the PDATA bus that can drive only 2mA.

3 Parameter 72 applies only to ENI registers while FAST is set to 0. This specification does *not* apply to shared RAM access.

4 Parameter 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 depends on the speed of the external RAM assigned to provide the physical shared RAM. Consequently, the max specification for shared RAM accesses is system-dependent.

5 Parameter 72b applies to ENI register accesses when FAST is set to 1.

6 Parameter 72c applies to ENI shared RAM accesses when FAST is set to 1.

7 Parameter 85 applies when FAST is set to 0. Parameter 85a applies when FAST is set to 1.

8 Parameter 77c can be reduced to ($3*T_{SYS}$) when FAST is set to 1.

ENI Shared RAM and Register Cycle timing

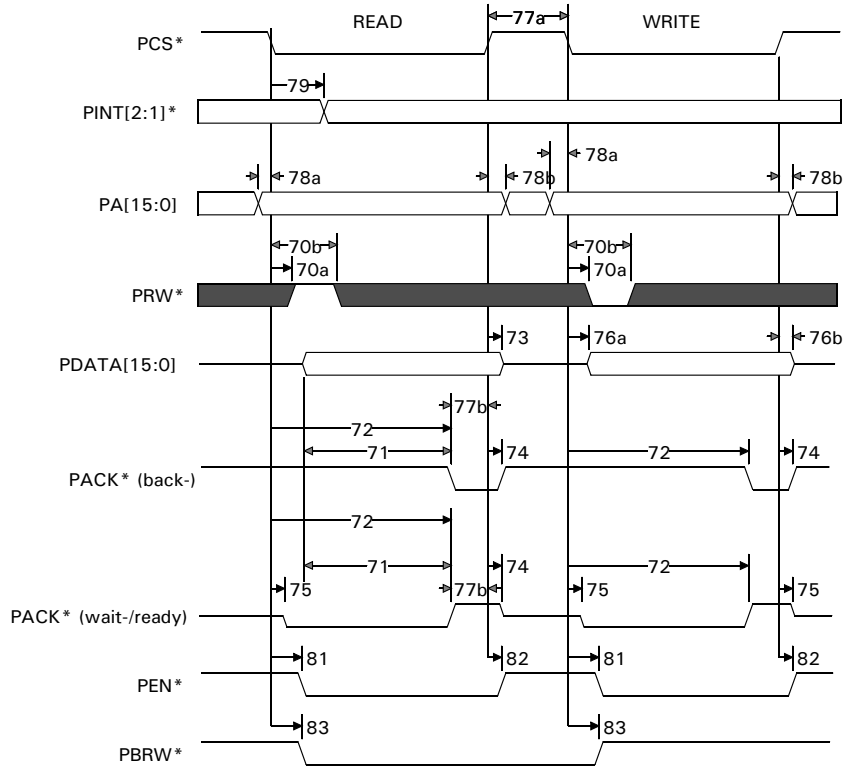


Figure 90: ENI Shared RAM and Register Cycle timing

ENI Single Direction DMA timing

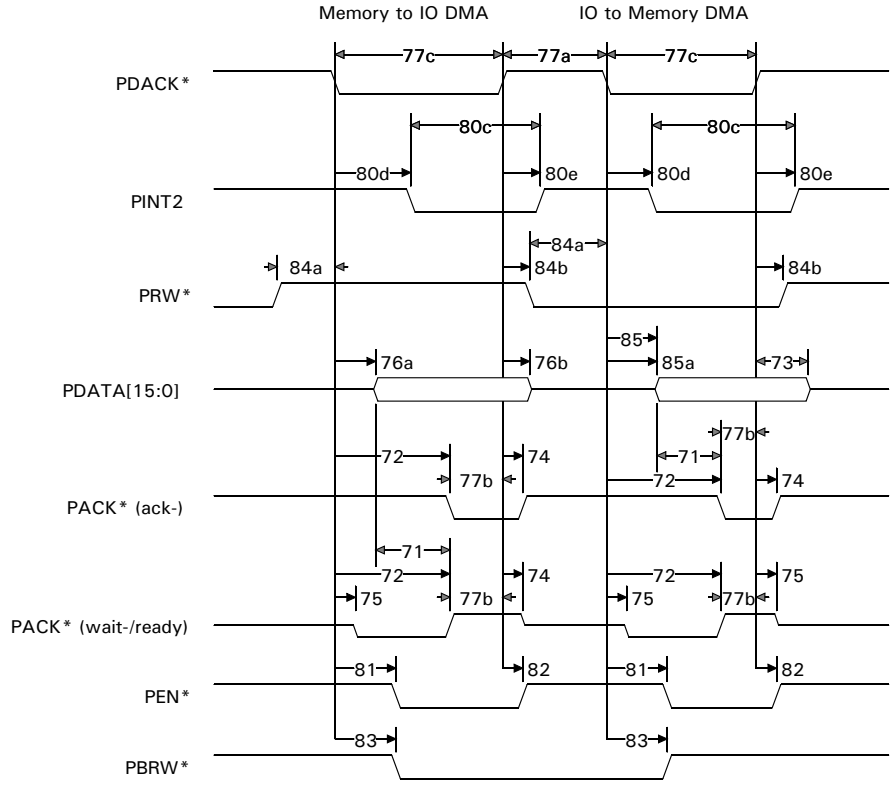


Figure 91: ENI Single Direction DMA timing

ENI Dual Direction DMA timing

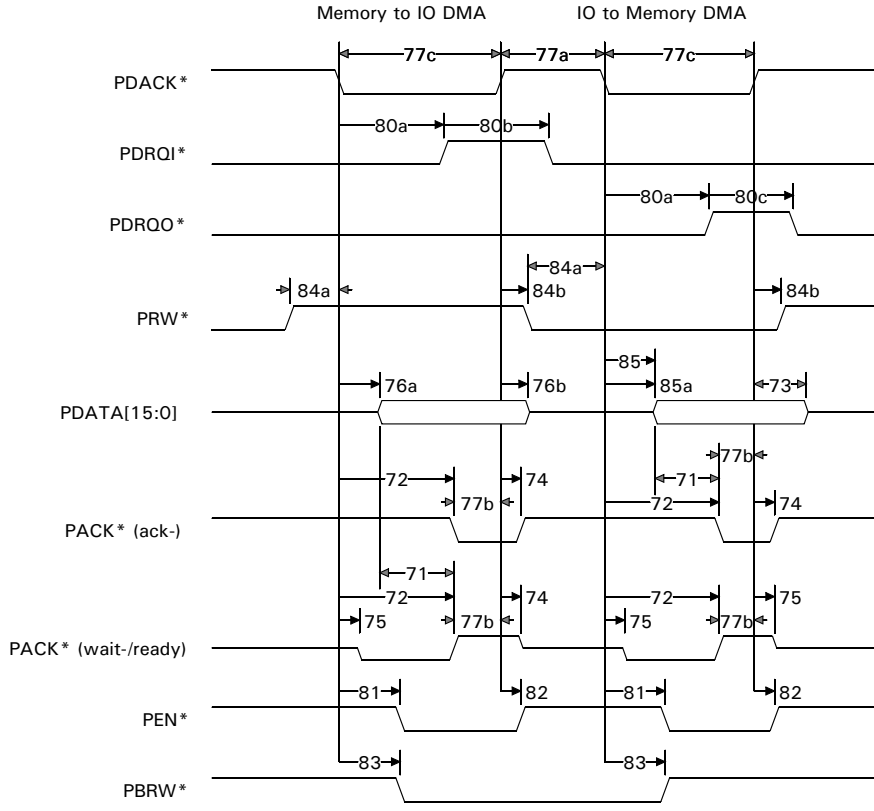


Figure 92: ENI Dual Direction DMA timing

Ethernet timing

Table 151 describes the values shown in the Ethernet timing diagrams (Figure 93 through Figure 95).

Num	Characteristic	Min	Max	Unit
90	TXCLK high to TXD, TXDV, TXER valid		17	ns
91a	RXD, RXER, RXDV valid to RXCLK high (setup)	8		ns
91b	RXCLK high to RXD, RXER, RXDV hold time	0		ns
91c	RXCLK high to RXD hold time	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 hold time	0		ns
94	RXCLK high to RPSF* change		6.5	ns
95a	REJECT* valid to RXCLK high (setup)	2.6		ns
95b	REJECT* valid from RXCLK high (hold)	0		ns
96	CRS low to RXCLK idle (bit-times)	27		Bit-times

Table 151: Ethernet timing values

Ethernet timing diagram

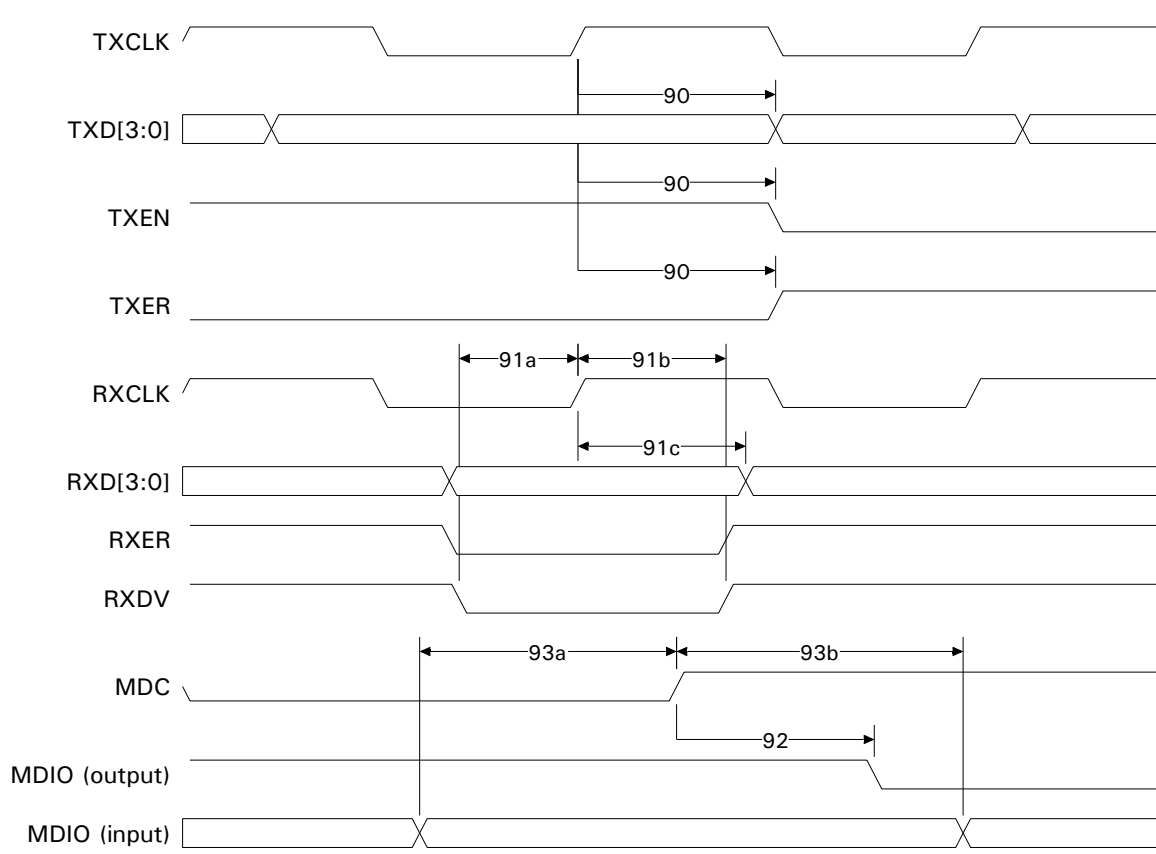


Figure 93: Ethernet timing

Ethernet Receive Clock Idle

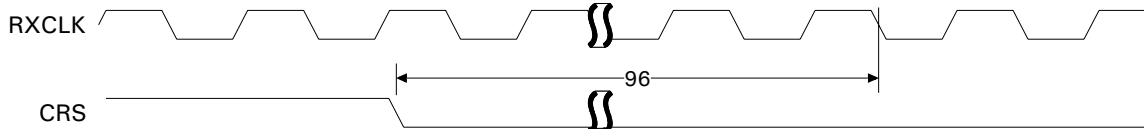


Figure 94: Ethernet Receive Clock Idle

External Ethernet CAM Filtering

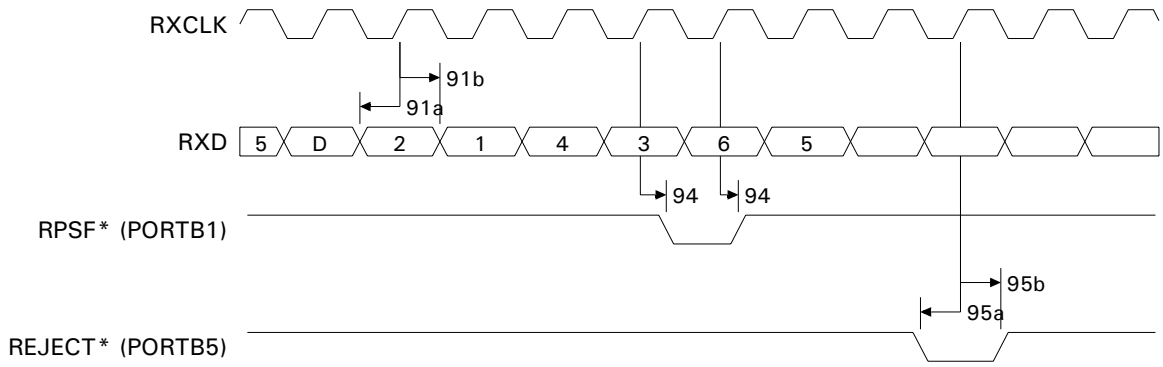


Figure 95: External Ethernet CAM Filtering

JTAG timing

Table 152 describes the values shown in Figure 96, "JTAG timing."

Num	Characteristic	Min	Max	Unit
100	TCK low to TDO valid	0	33.5	ns
101	TCLK low to TDO high impedance	0	32.3	ns
102a	TDI, TMS valid to TCK high (setup)	0.72		ns
102b	TCK high to TDI, TMS hold time	1.3		ns
103	TCK cycle time	31.2		ns
104	TCK pulse width	15.6		ns
105	TRST low time	27		ns

Table 152: JTAG timing values

Note: TRST* has an internal current sink. On production units, do not leave TRST* pulled low. The noise margin on inputs at a logic 0 is only a few tenths of a volt. NetSilicon recommends tying TRST* to RESET* on production units.

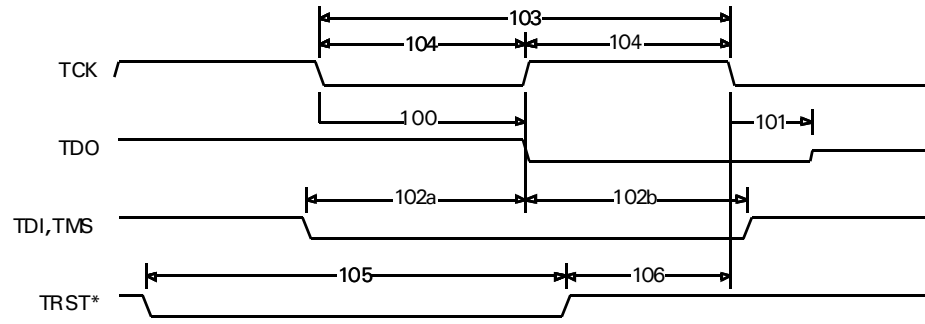


Figure 96: JTAG timing

1284 Port Multiplexing timing

Table 153 describes the values shown in Figure 97, "1284 Port 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}		ns
112	PCLKSx 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}		ns
114a	POEx* high to PDATA driven (read only)	T_{SYS}		ns

Table 153: 1284 Port Multiplexing timing values

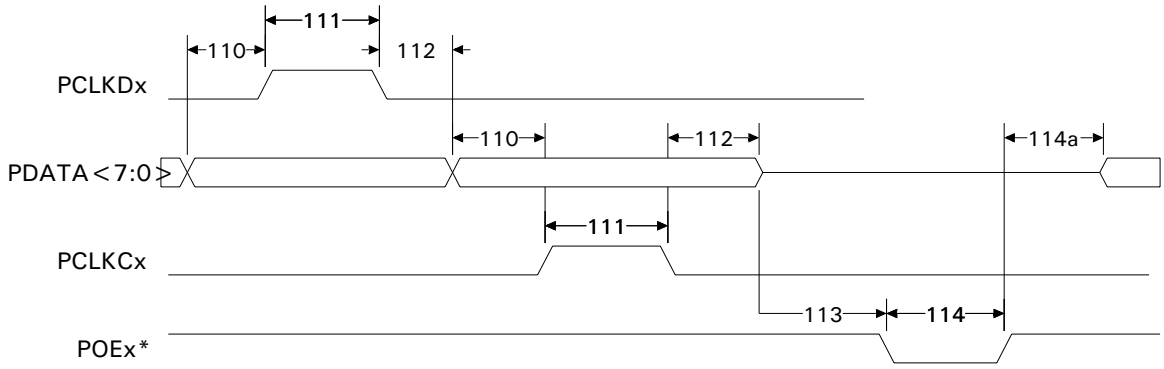


Figure 97: 1284 Port Multiplexing timing

1284 Compatibility mode timing

Table 154 and table 155 describe the timing values shown in Figure 98, "1284 Compatibility mode timing."

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
121	BUSY low to STROBE* (low)	$3*T_{SYS}$		ns

Table 154: 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
121	BUSY low to STROBE* (low)	$3*T_{SYS}$		ns

Table 155: 1284 Compatibility FAST mode timing (FAST = 1)

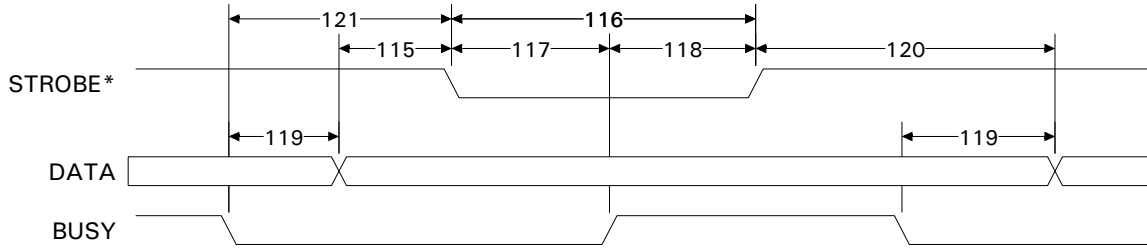


Figure 98: 1284 Compatibility mode timing

Forward ECP mode timing

Table 156 and Table 157 describe the timing values shown in Figure 99, "1284 Forward ECP mode timing."

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	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 156: 1284 SLOW Forward ECP mode timing (FAST = 0)

Num	Characteristic	Min	Max	Unit
115	DATA valid to STROBE* low (setup)	$3*T_{SY}$		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 157: 1284 FAST Forward ECP mode timing (FAST = 1)

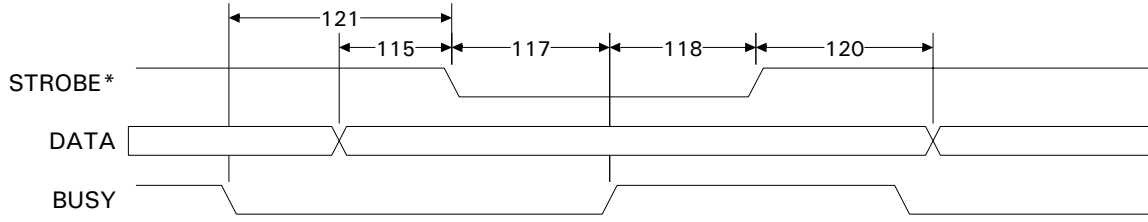


Figure 99: 1284 Forward ECP mode timing

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 vary significantly, depending on their resonant frequency. The choice must always be parallel resonance for the NET+50 chip with a load capacitance CL of 8pF. If the crystal recommended load capacitance is greater, additional capacitors must be added external to XTAL1 and XTAL2.

If the crystal used has a drive level smaller than the drive level specified, a series resistor can be added between XTAL2 and the crystal connection.

Instead of using the XTAL1/XTAL2 crystal oscillator circuit, the NET+50 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. Note that when this configuration is used, the internal PLL is bypassed and disabled.

Table 158 details the parameters for the external crystal:

Sym	Parameter	Conditions	Min	Typ	Max	Unit
f0	Operating frequency	16 Mhz		16	20	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 158: Crystal specification

ARM Exceptions

APPENDIX A

This appendix provides descriptions of the ARM exceptions outlined in Chapter 4, "Working with the CPU."

Note: The information in this appendix applies to both the NET+50 and NET+20M chips, unless otherwise noted.

About ARM exceptions

An exception occurs when the normal flow of a program is stopped temporarily. There are seven exceptions that can interrupt the ARM processor:

- Reset
- Undefined
- SWI
- Prefetch abort
- Data abort
- IRQ
- FIRQ

Reset exception

A reset exception is the highest priority exception. When the ARM is removed from reset, the ARM7TDMI:

1. Overwrites `R14_svc` and `SPSR_svc` by copying the current values of the PC (program counter) 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 system to fetch the next instruction from address 0x00.
4. Resumes execution in ARM state.

Undefined exception

When the ARM7TDMI comes across an instruction it cannot handle, it takes the undefined instruction trap. This exception can 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 instruction restores the PC and CPSR, and returns to the instruction following the undefined instruction.

SWI exception

The software interrupt instruction (SWI) is used to enter supervisor mode, usually to request a particular supervisor function. An 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.

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. A prefetch abort takes place during an instruction prefetch. If a prefetch abort occurs, the prefetch instruction is marked as invalid but the exception is not taken until the instruction reaches the head of the pipeline. The abort does not take place, however, if the instruction is not executed; for example, if a branch occurs while the instruction is in the pipeline.
- Data Abort. A data abort takes place during a data operand access. If a data abort occurs, the action taken depends on the instruction type:
  - Single data transfer instructions (LDR, STR) write back modified base registers; the abort handler must be aware of this.
  - The swap instruction (SWP) is aborted as though it had not been executed.
  - Block data transfer instructions (LDM, STM) complete. If write-back is set, the base is updated. If the instruction would have overwritten the base with data (that is, the data has the base in the transfer list), overwriting is prevented. All register overwriting is prevented after an abort is indicated, which means 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 this type of 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, and its state is not affected by the abort.

The handler executes one of the following instructions, irrespective of the state (ARM or THUMB), after fixing the cause of the abort:

- For a prefetch abort: `SUBS PC, R14_abt, #4`
- For a data abort: `SUBS PC, R14_abt, #8`

IRQ exception

The IRQ (interrupt request) exception is a normal interrupt sourced by the NET+50 interrupt controller. IRQ has a lower priority than FIRQ and is masked when a FIRQ sequence is entered. You can disable the IRQ interrupt at any time by setting the I bit in the CPSR to 1. You must be in privileged (non-user) mode to do this, however.

The IRQ handler should leave the interrupt by executing the following instruction, irrespective of the state (ARM or THUMB):

```
SUBS PC, R14_irq, #4
```

FIRQ exception

The FIRQ (fast interrupt request) exception supports a data transfer or channel process. In ARM state, FIRQ has enough registers to remove the need for register saving, which minimizes context switching overhead.

The only two internal peripherals that can generate a FIRQ interrupt are the GEN module built-in timers and the GEN module watch-dog timer.

The FIRQ handler should leave the interrupt by executing the following execution, irrespective of the state (ARM or THUMB):

```
SUBS PC, R14_firq, #4
```

You can disable the FIRQ interrupt by setting the CPSR F flag to 1; you must be in non-user mode, however. 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.

NET+20M Chip Package

APPENDIX B

The NET+20M chip can be used in any embedded environment requiring networking services in an Ethernet LAN. Application-specific hardware can be attached to the NET+20M system bus for custom applications that use the NET+20M internal RISC processor.

NET+20M chip pinout

The NET+20M is a 208 ball grid array (BGA) chip. Figure 100 shows the NET+20M BGA pinout.

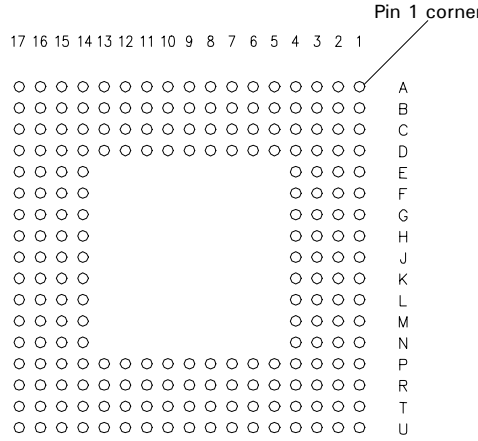


Figure 100: NET+20M BGA pinout (bottom view)

Table Information and Tables

Table 159 through Table 165 identify and describe the ball number assignments contained in the NET+2M chip. Each table pertains to an interface, and contains the following information:

- **Signal column.** Identifies the pin name for each I/O signal. Some signals have multiple modes and are identified accordingly. You configure the mode through firmware using a configuration register. Some modes may require hardware configuration during a RESET condition.
- **BGA column.** Identifies the ball number assignment for a specific I/O signal. A dagger (†) next to the pin number indicates that the pin is an input current source.
- **I/O column.** Indicates whether the signal is input (I), output (O), or both (I/O).

- **OD (Drive) column.** Indicates the drive strength of an output buffer. The NET+50 chip uses one of three driver types:
  - 2mA
  - 4mA
  - 8mA

System bus interface

Signal		BGA	I/O	OD	Description
BCLK		T10	O	8	Synchronous bus clock
ADDR27	CS0OE*	F4†	I/O	4	Address bus
ADDR26	CS0WE*	F3†	I/O	4	
ADDR25	BLAST*	E2†	I/0	4	
ADDR24		D2†	I/O	4	
ADDR23		E3†	I/O	4	
ADDR22		E4†	I/O	4	
ADDR21		D1†	I/O	4	
ADDR20		C2†	I/O	4	
ADDR19		D3†	I/O	4	
ADDR18		C1†	I/O	4	
ADDR17		B1†	I/O	4	
ADDR16		B2†	I/O	4	
ADDR15		B3†	I/O	4	
ADDR14		A3†	I/O	4	
ADDR13		A4†	I/O	4	
ADDR12		B4†	I/O	4	
ADDR11		C3†	I/O	4	
ADDR10		A5†	I/O	4	

Table 159: NET+20M BGA Chip Pinout - System bus interface

Signal	BGA	I/O	OD	Description
ADDR9	D4†	I/O	4	
ADDR8	C4†	I/O	4	
ADDR7	B5†	I/O	4	
ADDR6	A6†	I/O	4	
ADDR5	D5†	I/O	4	
ADDR4	C5†	I/O	4	
ADDR3	B6†	I/O	4	
ADDR2	A7†	I/O	4	
ADDR1	D6†	I/O	4	
ADDR0	C6†	I/O	4	
DATA31	T3	I/O	4	Data bus
DATA30	R4	I/O	4	
DATA29	U3	I/O	4	
DATA28	U2	I/O	4	
DATA27	R2	I/O	4	
DATA26	R1	I/O	4	
DATA25	P1	I/O	4	
DATA24	P2	I/O	4	
DATA23	R3	I/O	4	
DATA22	N1	I/O	4	
DATA21	P4	I/O	4	
DATA20	P3	I/O	4	
DATA19	N2	I/O	4	
DATA18	M1	I/O	4	
DATA17	N4	I/O	4	
DATA16	L1	I/O	4	
DATA15	M4	I/O	4	

Table 159: NET+20M BGA Chip Pinout - System bus interface

Signal	BGA	I/O	OD	Description
DATA14	M3	I/O	4	
DATA13	L2	I/O	4	
DATA12	K1	I/O	4	
DATA11	L4	I/O	4	
DATA10	L3	I/O	4	
DATA9	K2	I/O	4	
DATA8	J1	I/O	4	
DATA7	K4	I/O	4	
DATA6	K3	I/O	4	
DATA5	J4	I/O	4	
DATA4	J3	I/O	4	
DATA3	H2	I/O	4	
DATA2	G1	I/O	4	
DATA1	H4	I/O	4	
DATA0	H3	I/O	4	
TS*	NO CONNECT			
BE3*	G2	I/O	2	Byte enable D31:D24
BE2*	F1	I/O	2	Byte enable D23:D16
BE1*	G4	I/O	2	Byte enable D15:D08
BE0*	G3	I/O	2	Byte enable D07:D00
RW*	U4	I/O	2	Transfer direction
TA*	R5†	I/O	8	Data transfer acknowledge
TEA*	T4†	I/O	8	Transfer error/Last acknowledge
BR*	NO CONNECT			
BG*	NO CONNECT			
BUSY*	NO CONNECT			

Table 159: NET+20M BGA Chip Pinout - System bus interface

Chip select controller

Signal	BGA	I/O	OD	Description
CS0*	T11	O	4	Chip select (Boot select)
CS1*/RAS1*	R12	O	4	Chip select/DRAM RAS*
CS2*/RAS2*	P12	O	4	Chip select/DRAM RAS*
CS3*/RAS3*	U11	O	4	Chip select/DRAM RAS*
CS4*/RAS4*	T12	O	4	Chip select/DRAM RAS*
CAS3*/SDRAS*	T13	O	4	DRAM column strobe D31:24/SDRAM RAS
CAS2*/SDCAS*	U12	O	4	DRAM column strobe D23:16/SDRAM RAS
CAS1*/SDWE*	P13	O	8	DRAM column strobe D15:08/SDRAM RAS
CAS0*/SD(AP)	R13	O	4	DRAM column strobe D07:00/SDRAM RAS
WE*	R11	O	4	Write enable
OE*	P11	O	4	Output enable

Table 160: NET+20M BGA Chip Pinout - Chip select controller

Ethernet interface

Signal		BGA	I/O	OD	Description	
MII	**10BaseT**				**MII**	**10BaseT**
MDC	LB*	D7	O	2	MII clock	Loopback enable
MDIO	UTPSTP*	C7†	I/O	2	MII data	Cable type
TXCLK	TXCLK	B8	I		TX clock	
TXD0	TXD	A9	O	2	TX data 0	TX data
TXD1	PDN* (OD)	D8	O	2	TX data 1	Power down
TXD2	NTHRES	C8	O	2	TX data 2	Normal threshold
TXD3	THIN	D9	O	2	TX data 3	Enable Thinnet

Table 161: NET+20M BGA Chip Pinout - Ethernet interface

Signal		BGA	I/O	OD	Description	
TXER	LTE	C9	O	2	TX code error	Link test enable
TXEN	TXEN	B10	O	2	TX enable	
COL	TXCOL	A11	I		Collision	
CRS	RXCRS	D10	I		Carrier sense	
RXCLK	RXCLK	C10	I		RX clock	
RXD0	RXD	B11	I		RX data 0	RX data
RXD1	MANSENSE	A12	I		RX data 1	Sense jumper
RXD2	JABBER	D11	I		RX data 2	Jabber
RXD3	REVPOL	C11	I		RX data 3	Reverse polarity
RXER	LINKPUL*	B12	I		RX error	Link pulse detection
RXDV	AUTOMAN	A13	I		RX data valid	10B2 selected

Table 161: NET+20M BGA Chip Pinout - Ethernet interface

UARTS-SPI-GPIO

Important: The NET+20M supports only 16 GPIO/special function pins. Any combination of pins is acceptable.

Signal		BGA	I/O	OD	SPI Slave Mode	SPI Master Mode
PORTA7	TXDA	G17†	I/O	2	SPI-S-TXD-O-A	SPI-M-TXD-O-A
PORTA6	DTRA*/DRQ1*	F16†	I/O	2		
PORTA5	RTSA*	E15†	I/O	2		
PORTA4	RXCA	E14†	I/O	2	SPI-S-CLK-I-A	SPI-M-CLK-O-A*
PORTA3	RXDA	F17†	I/O	2	SPI-S-RXD-I-A	SPI-M-RXD-I-A
PORTA2	DSRA*/DAK1*	E16†	I/O	2		
PORTA1	CTSA*	D15†	I/O	2		
PORTA0	DCDA* DON1*	D14†	I/O	2		

Table 162: NET+20M BGA Pinout - UARTS-SPI-GPIO

Signal		BGA	I/O	OD	SPI Slave Mode	SPI Master Mode
PORTB7	TXDB	E17†	I/O	2	SPI-S-TXD-O-B	SPI-M-TXD-O-B
PORTB6	DTRB*DRQ2*	C15†	I/O	2		
PORTB5	RTSB*	D16†	I/O	2	Reject* Ethernet package	
PORTB4	RXCB	D17†	I/O	2	SPI-S-CLK-I-B	SPI-M-CLK-O-B*
PORTB3	RXDB	C17†	I/O	2	SPI-S-RXD-I-B	SPI-M-RXD-I-B
PORTB2	DSRB*/DAK2*	C16†	I/O	2		
PORTB1	CTSB*	B16†	I/O	2	RPSF* Ethernet frame boundary	
PORTB0	DCDB*/DON2*	A16†	I/O	2		
PORTC7	TXCA	A15†	I/O	2	SPI-S-EN-I-A	SPI-M-EN-O-A*
PORTC6	RIA*/IQRO*	C14†	I/O	2		
PORTC5	TXCB	B15†	I/O	2	SPI-S-EN-I-B	SPI-M-EN-O-B
PORTC4	RIB*	A14†	I/O	2		
PORTC3	AMUX	D13†	I/O	8	Interrupt 3	
PORTC2		C13†	I/O	8	Interrupt 2	
PORTC1		B14†	I/O	8	Interrupt 1	
PORTC0		B13†	I/O	8	Interrupt 0	

Table 162: NET+20M BGA Pinout - UARTS-SPI-GPIO

Clock generation/system reset

Signal	BGA	I/O	OD	Description
XTAL1	U7	I		Crystal oscillator circuit
XTAL2	T8	O		
PLLVDD (2.5V)	U8			2.5 V PLL clean power
PLLLPF	P9			PLL loop filter capacitor
PLLVSS	R9			PLL clean ground

Table 163: NET+20M BGA Chip Pinout - Clock generation/system reset

Signal	BGA	I/O	OD	Description
PLLTST* (2.5V)	P8†	I		PLL test mode
BISTEN*	R10†	I		Enable internal BIST operation
SCANEN*	P10†	I		Enable internal SCAN testing
System reset				
RESET*	T2†	I		System reset

Table 163: NET+20M BGA Chip Pinout - Clock generation/system reset

JTAG port for ARM core

Signal	BGA	I/O	OD	Description
TDI	T6†	1		Test data in
TDO	U5	O	2	Test data out
TMS	R7†	I		Test mode select
TRST*	R8	I		Test mode reset. This is a current input sink.
TCK	P7	I		Test mode clock

Table 164: NET+20M BGA Chip Pinout - JTAG port for ARM core

Power supply

Signal	Mode	Ball Number	Description
V_{CC}DC 3.3V DC	BGA	F15, B17, C12, A2, M2, U9	I/O steady state (6 pairs)
V_{SS}DC GND Returns	BGA	F14, A17, D12, A1, N3, U10	
V_{CC}AC 3.3V	BGA	A8, E1, T1, U16	I/O switching (4 pairs[a])
V_{SS}AC GND Returns	BGA	B7, F2, U1, U17	

Table 165: NET+20M BGA Chip Pinout - Power supply

Signal	Mode	Ball Number	Description
V_{DD}CO 2.5V	BGA	K17, A10, H1, T7	Core power (4 pairs)
V_{SS}CO GND Returns	BGA	J16, B9, J2, U6	
$PLLV_{DD}$ 2.5V	BGA	U8	PLL bead filtered clean power
$PLLV_{SS}$ GND Return	BGA	R9	
$PORV_{SS}$ GND	BGA	T9	Powerup reset GND reference

Table 165: NET+20M BGA Chip Pinout - Power supply

a. NetSilicon recommends that you use separate power pairs for AC and DC power, to prevent the noise in the AC power buses from reaching the DC power buses. NetSilicon recommends and uses a ferrite bead to filter the AC power pins.

Additional information about NET+20M pins

- The following pins are reserved:

Must be terminated*		Must be left unconnected	
G14	L14 - 17	G15 and 16	P14 - 17
H16 and 17	M14, 16, and 17	H14 and 15	R14 - 17
J14, 15, and 17	N17	M15	T14 - 17
K14 - 16		N14 - 16	U13 - 15

* These can be tied to a single input current source resistor.

- All outputs drive TTL levels. Outputs drive to 0.4 V maximum on low and 2.4 V maximum on high.
- The following pins require a 2.5 V input level:
  - XTAL1 (U7)
  - PLLTST (P8)
  - PLLVDD (U8)
- TRST* (R8) is the only pin that is an input current sink.

- All inputs are TTL levels and are 3.3 V-tolerant, allowing integration with 3.3 V devices.

Signal description summary

The signal description summary defines and briefly describes the signals included in the interfaces in the NET+20M chip. The description tables in the section are presented in the same order as the NET+20M chip pinout tables. For details about any of the signals/pinouts, see the appropriate chapter:

Interface	Chapter
System bus	Chapter 8, "Memory Controller Module"
Chip select controller	Chapter 8, "Memory Controller Module"
Ethernet MII	Chapter 10, "Ethernet Controller Module"
UARTS-SPI-GPIO; GPIO ports A, B, C	Chapter 6, "The GEN Module"
Clock generation	Chapter 6, "The GEN Module"

The JTAG port for ARM core (ARM debugger) and Power sections are addressed fully in their respective discussions.

System Bus Interface

The NET+20M chip uses the system bus interface to interface with memory-mapped peripheral devices such as flash, SRAM, and DRAM.

Code	Definition	Description
BCLK	Bus clock	Provides the bus clock. All system bus interface signals are referenced to the BCLK signal.
ADDR	Address bus	Identifies the address of the peripheral being addressed by the current bus master. The address bus is bi-directional.

Table 166: System bus interface signal description

Code	Definition	Description
DATA	Data bus	Provides the data transfer path between the NET+20M chip and external peripheral devices.The data bus is bi-directional.
TS*	Transfer start	NO CONNECT
BE*	Byte enable	Identifies which 8-bit bytes of the 32-bit data bus are active during any given system bus memory cycle. The BE* signals are active low and bi-directional.
RW*	Read/write indicator	Indicates the direction of the system bus memory cycle. RW* high identifies a read operation; RW* low identifies a write operation. The RW* signal is bi-directional.
TA*	Transfer acknowledge	Indicates the end of the current system bus memory cycle. This signal is bi-directional.
TEA*	Transfer error acknowledge	Indicates an error termination or burst cycle termination. TEA* is bi-directional. The NET+20M chip or the external peripheral can drive the TEA* signal.
BR*	Bus request	NO CONNECT
BG*	Bus grant	NO CONNECT
BUSY*	Bus busy	NO CONNECT

Table 166: System bus interface signal description

Chip select controller

The NET+20M chip supports five unique chip select configurations.

Code	Definition	Description
CS0* CS1* CS2* CS3* CS4*	Chip select 0 Chip select 1 Chip select 2 Chip select 3 Chip select 4	Unique chip select outputs supported by the NET+20M chip. Each chip select can be configured to decode a portion of the available address space and can address a maximum of 256 Mbytes of address space. The chip selects are configured using registers in the memory module.
CAS0* CAS1* CAS2* CAS3*	Column address strobe signals	Activated when an address is decoded by a chip select module 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.

Table 167: Chip select controller signal description

Code	Definition	Description
WE*	Write enable	Active low signal indicating that a memory write cycle is in progress. This signal is activated only during write cycles to peripherals controlled by one of the chip selects in the memory module.
OE*	Output enable	Active low signal indicating that a memory read cycle is in progress. This signal is activated only during read cycles from peripherals controlled by one of the chip selects in the memory module.

Table 167: Chip select controller signal description

Ethernet interface

The Ethernet MII (Media Independent Interface) provides the connection between the Ethernet PHY and the MAC (Media Access Controller).

Code	Definition	Description
MDC	Management data clock	Provides the clock for the MDIO serial data channel. The MDC signal is a NET+20M chip output. The maximum frequency is 2.5 MHz.
MDIO	Management data IO	A bi-directional signal that provides a serial data channel between the NET+20M chip and the external Ethernet PHY module.
TXCLK	Transmit clock	An input to the NET+20M chip from the external PHY module. TXCLK provides the synchronous data clock for transmit data.
TXD3 TXD2 TXD1 TXD0	Transmit data signals	Nibble bus used by the NET+20M chip to drive data to the external Ethernet PHY. All transmit data signals are synchronized to TXCLK. In ENDEC mode, only TDX0 is used for transmit data.
TXER	Transmit coding error	Output asserted by the NET+20M when an error has occurred in the transmit data stream.
TXEN	Transmit enable	Asserted when the chip drives valid data on the TXD outputs. This signal is synchronized to TXCLK.
COL	Transmit collision	Input signal asserted by the external Ethernet PHY when a collision is detected.
CRS	Receive carrier sense	Asserted by the external Ethernet PHY whenever the receive medium is non-idle.

Table 168: Ethernet MII signal description

Code	Definition	Description
RXCLK	Receive clock	An input to the NET+20M chip from the external PHY module. The receive clock provides the synchronous data clock for receive data.
RXD3 RXD2 RXD1 RXD0	Receive data signals	Nibble bus used by the NET+20M chip to input receive data from the external Ethernet PHY. All receive data signals are synchronized to RXCLK. In ENDEC mode, only RXD0 is used for receive data.
RXER	Receive error	Input asserted by the external Ethernet PHY when the Ethernet PHY encounters invalid symbols from the network.
RXDV	Receive data valid	Input asserted by the external Ethernet PHY when the Ethernet PHY drives valid data on the RXD inputs.

Table 168: Ethernet MII signal description

GPIO (PORTA/B/C)

See Chapter 6, "The GEN Module," for full configuration information for PORTA/B/C. All ports can be configured for general purpose I/O pins. Only special function codes are listed.

Code	Definition	Description
PORTA7	Input/Output/TxDA/SPI-A-M + S-TXD-O	Can be configured as the special function TxDA output.
PORTA6	Input/Output/DREQ1*/DTRA*	Can be configured as the special function DREQ1* input or the special function DTRA.
PORTA5	Input/Output/RTSA*	Can be configured as the special function RTSA output.
PORTA4	Input/Output/RxCA/SPI-A-S-CLK-I/ SPI-A-M-CLK-O	Can be configured for special function RxCA input or the special function OUT1A output.
PORTA3	Input/Output/RxDA/SPI-A-S + M-RXD-I	Can be configured for special function RxDA input.
PORTA2	Input/Output/DSRA*/DACK1*	Can be configured for special function DSRA input or for special function DACK1* output.
PORTA1	Input/Output/CTSA*	Can be configured for special function CTSA input.
PORTA0	Input/Output/DCDA*/DONE1*	Can be configured for special function DCDA input or DONE1* I/O.

Code	Definition	Description
PORTB7	Input/Output/TxDB/SPI-B-M and S-TXD-O	Can be configured as the special function TxDB output.
PORTB6	Input/Output/DREQ2*/DTRB*	Can be configured as the special function DREQ2* input or the special function DTRB.
PORTB5	Input/Output/REJECT*/RTSB*	Can be configured as the special function REJECT input or for special function RTSB output.
PORTB4	Input/Output/RxCB/SPI-B-M-CLK-O/ SPI-B-S-CLK-I	Can be configured for special function RxCB input or special function OUT1B.
PORTB3	Input/Output/RxDB/SPI-B-S + M-RXD-I	Can be configured for special function RxDB input.
PORTB2	Input/Output/DSRB*/DACK2*	Can be configured for special function DSRB input or special function DACK2* output.
PORTB1	Input/Output/CTSB*/RPSF*	Can be configured for special function RPSF* output or special function CTSB input.
PORTB0	Input/Output/DCDB*/DONE2*	Can be configured for special function DCDB input or special function DONE2* I/O.
PORTC7	Input/Output/TxCA/SPI-A-M-ENB-O/ SPI-A-S-ENB-I	Can be configured for special function TxCA input or special function OUT2A output.
PORTC6	Input/Output/RIA*/IRQ*	Can be configured as the special function RIA* input or for special function IRQ* output.
PORTC5	Input/Output/TxCB/SPI-B-M-ENB-O/ SPI-B-S-ENB-I	Can be configured for special function TxCB input or special function OUT2B output.
PORTC4	Input/Output/RIB*	Can be configured for special function RIB input.
PORTC3	Input/Output/CI3/AMUX	Can be configured for special function C3 interrupt input or to provide the DRAM address multiplexer function.
PORTC2	Input/Output/CI2	Can be configured for special function C2 interrupt input.
PORTC1	Input/Output/CI1	Can be configured for special function C1interrupt input.
PORTC0	Input/Output/CI0	Can be configured for special function C0 interrupt input.

Clock generation and reset

The NET+20M has three main clock domains:

- System clock (SYSCLK), which drives the CPU and internal logic.
- Bit rate generation and programmable timer reference clock (XTAL). Also produces SYSCLK through the PLL when PLLTST* = 1.
- System bus clock (BCLK), which drives devices external to the NET+20M.

The SYS module provides the NET+20M chip with these clocks, as well as system reset and boot-up resources.

Code	Definition	Description
XTAL1	Crystal oscillator input	A standard (18.432 MHz) parallel-resonant crystal can be attached to these pins to provide the main input clock to the NET + 20M chip.
XTAL2	Crystal oscillator output	The oscillator frequency equals SYSCLK. XTAL1 is in the 2.5V ring and has a maximum VIH of 2.75V. A typical 3.3V oscillator requires a 220 ohm series resistor with its output.
PLLVDD	Clean PLL power	Powers the internal 2.5V PLL circuit.
PLLVSS	Clean PLL ground	Grounds the internal PLL circuit.
PLLLPF	PLL loop filter	Provides the PLL loop filter circuit.
RESET*	System reset	Resets the NET + 20M chip hardware.

Table 169: Clock generation and reset signal description

Test support

The PLLTST*, BISTEN*, and SCANEN* primary input pins put the NET+20M into various test modes, for both functional and manufacturing test operations.

Code	Definition	Description
PLLTST*	PLL test and disable	Disables the internal PLL when driven active low. Enables the crystal oscillator and internal PLL when left unconnected or driven high.
BISTEN*	BIST enable	Used for testing purposes only; the pin should always be left unconnected or driven high.

Table 170: Test support signal description

Code	Definition	Description
SCANEN*	SCAN enable	Used for testing purposes only; the pin should always be left unconnected or driven high.

Table 170: Test support signal description

For more information about PLLTST*, see Chapter 6, "The GEN Module."

ARM debugger

Five pins provide a dedicated connection to the internal ARM processor core. These five signals connect to the ARM processor only, and are used for software development using the ARM Embedded ICE module (or similar device). These five signals cannot be used for 1149.1 JTAG testing of the NET+20M chip.

- TDI
- TDO
- TMS
- TRST*
- TCK

The following figure illustrates the pin connection:

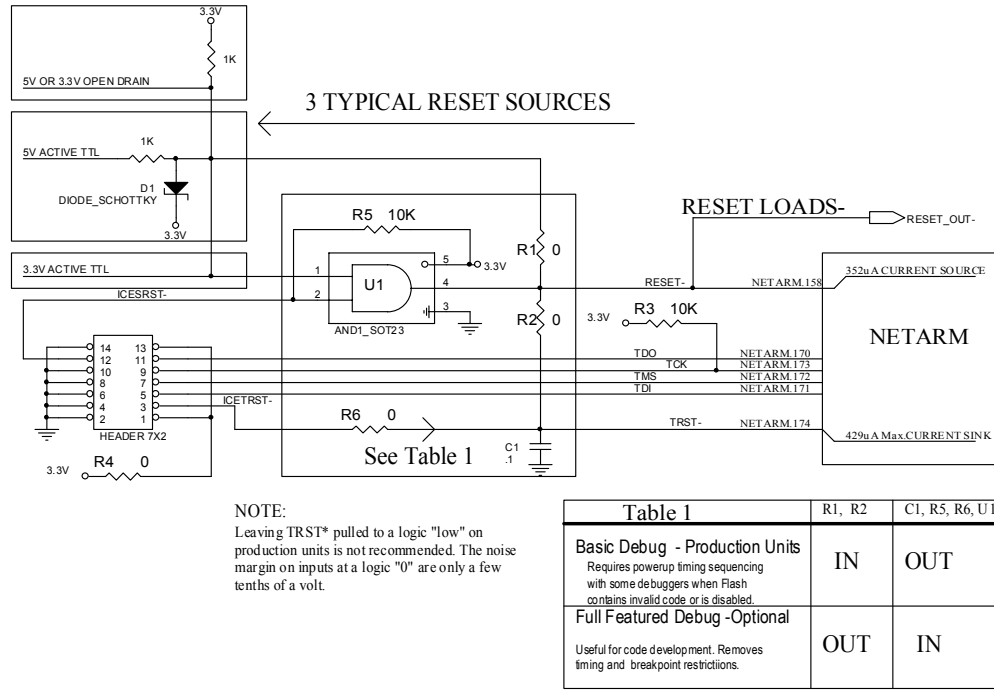


Figure 101: NET+50/20M reset and ARM debugger circuit

Power

This section describes the pins that provide power for various components of the I/O output drivers and the internal NET+20M chip core.

Code	Definition	Description
V_{CC}AC	Power A/C switching	The V_{CC}AC 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. Each V_{CC}AC pin has a GNDAC ground return.
V_{CC}DC	Power DC drive	The V_{CC}DC 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. Each V_{C}CDC pin has a GNDDC ground return.

Table 171: Power signal description

Code	Definition	Description
$V_{DD}CO$	Power core	The $V_{DD}CO$ pins provide the 2.5V power for the internal NET+20M chip core. Each of these pins must have a bypass capacitor placed as close to the pin as physically possible. Each $V_{DD}CO$ pin has a $V_{SS}CO$ ground return.
$PORV_{SS}$	Power on reset	Grounds the power-on-reset circuitry.
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+20M chip core. These pins should not require any filtering other than standard decoupling practices.

Table 171: Power signal description

Packaging

Figure 102 shows the package for the NET+20M. Table 172 provides the NET+20M dimensions:

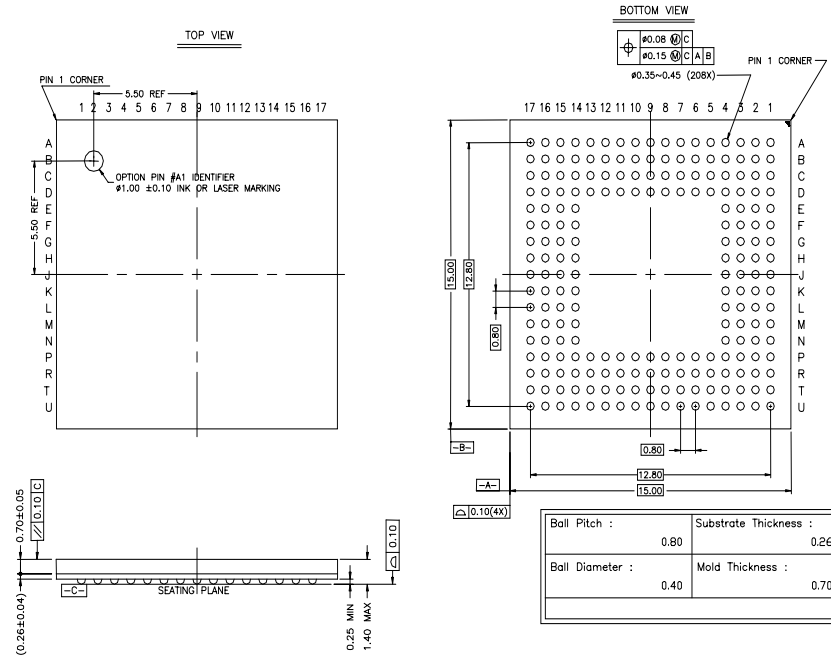


Figure 102: BGA package

Dimension	Minimum	Nominal	Maximum
A	----------	----------	1.40
A1	0.22	0.26	0.30
A2	0.65	0.70	0.75
A3	0.25	----------	----------
B	----------	5.50 REF	----------
D	----------	15.00	----------

Table 172: BGA dimensions

Dimension	Minimum	Nominal	Maximum
D1	----------	12.80	----------
E	----------	15.00	----------
E1	----------	12.80	----------
P	----------	0.80	----------

Table 172: BGA dimensions

Index

Numerics

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X


NetSilicon
A Digi International® Company