

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

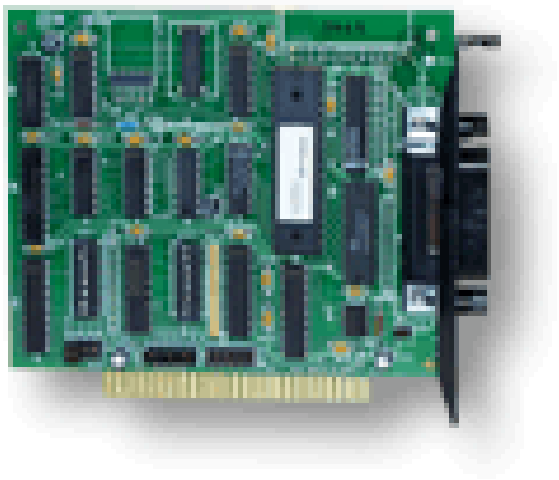
Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Visit our website - *Click HERE*

ZT 1444A

IEEE 488 Interface for Personal Computers



ÛZIATECH

DECLARATION OF CONFORMITY

We: **ZIATECH CORPORATION**

1050 SOUTHWOOD DRIVE

SAN LUIS OBISPO, CA 93401 USA

declare under our sole responsibility that the product

ZT 1444A

to which this declaration relates is in conformity with the following standard(s) or other normative document(s)

EN 55022 1994 EN 50082-1 1992

following the provisions of **89/336/EEC** directive.

San Luis Obispo CA USA **date: 2/4/96** **BY: Bert Forbes**

President

Signature: 

CONTENTS

WHAT'S IN THIS MANUAL? 6

1. INTRODUCTION 8
- PRODUCT DEFINITION 8
- FUNCTIONAL BLOCKS 9
- HARDWARE FEATURES OF THE ZT 1444A 9
- SOFTWARE FEATURES OF THE ZT 1444A 10

2. GETTING STARTED 11
- UNPACKING 11
- WHAT'S IN THE BOX? 11
- SYSTEM REQUIREMENTS 11
- SETTING UP YOUR WORKING DISKS 12
- SETTING UP YOUR INTERFACE BOARD 12
- INTERFACE AND SOFTWARE CAPABILITIES 13
- INSTALLING YOUR INTERFACE BOARD 14

3. THEORY OF OPERATION 15
- IEEE 488 ADAPTER AND TRANSCEIVERS 16
- DIP SWITCHES AND JUMPERS 16
- CARD SELECT LOGIC, I/O PORT DECODE LOGIC 16
- DMA CONTROL LOGIC 16
- DMA CHANNEL SELECT LOGIC 17
- INTERRUPT PRIORITY SELECT LOGIC 17
- SECURITY KEY OPTION 17
- SOFTWARE INTERFACING 17

4. HARDWARE 18
- SUMMARY OF DMA AND INTERRUPT LINE USAGE 18

5. INTERRUPTS AND DMA 19
- ZT 1444A INTERRUPTS AND DMA 19
  - ZT 1444A CONTROL REGISTER 19

6. THE IEEE 488 INTERFACE (NAT9914BPD) 22
- NAT9914BPD REGISTERS 22
  - ADDRESS REGISTER - TALKER/LISTENER 22
  - ADDRESS SWITCH REGISTER - GENERAL PURPOSE 26
  - ADDRESS STATUS REGISTER - TALKER/LISTENER 26
    - ULPA 27
    - TPAS/LPAS 28
    - TADS/LADS 28
    - ATN 28
    - LLO 29
    - REM 29
  - BUS STATUS REGISTER - DEBUGGING 29
  - COMMAND PASS-THROUGH REGISTER - TALKER/LISTENER 30
  - PARALLEL POLL REGISTER - TALKER/LISTENER 31
    - PARALLEL POLL SUBSET PP2 32
    - PARALLEL POLL SUBSET PP1 32
    - PARALLEL POLL VERSUS SERIAL POLL 34
    - PARALLEL POLL IEEE 488 DRIVERS 34
  - SERIAL POLL REGISTER - TALKER/LISTENER 34
  - DATA IN REGISTER 36
  - DATA OUT REGISTER 38

INTERRUPT MASK/STATUS REGISTERS 39
INTERRUPT MASK/STATUS REGISTER 0 40
INT0/INT1 41
BI 41
BO 41
END 41
SPAS 41
RLC 41
MAC 42
INTERRUPT MASK/STATUS REGISTER 1 42
GET 43
ERR 43
UNC 43
APT 43
DCAS 44
MA 44
IFC 44
AUXILIARY COMMAND REGISTER 45
AUXILIARY COMMANDS 45
SWRST (SOFTWARE RESET) 0/1XX00000 47
USING THE NAT9914BPD AS A CONTROLLER 54
USING THE NAT9914BPD AS A DEVICE 56

7. IEEE 488 TRANSCEIVERS (75160/75162) 57

8. OPTIONAL SECURITY KEY INTERFACE 58
PROGRAMMING SEQUENCE 58
PROGRAMMING SUMMARY 59
READING AND WRITING TO THE KEY 59
WRITE SEQUENCE 59
READ SEQUENCE 60
SECURITY METHODS 60
DEVICE CAPABILITIES 62

A. JUMPER CONFIGURATIONS 63
ZT 1444A JUMPERS 63
ZT 1444A VS. ZT 1444 63
ZT 1444A I/O PORT ADDRESS SWITCH CONFIGURATIONS 65
ZT 1444A JUMPER DESCRIPTIONS 65
CONFIGURING THE ZT 1444A 66

B. CUSTOMER SUPPORT 69
TECHNICAL/SALES ASSISTANCE 69
RELIABILITY 69
RETURNING FOR SERVICE 70
ZIATECH WARRANTY 70
FIVE-YEAR LIMITED WARRANTY 70
LIFE SUPPORT POLICY 71
TRADEMARKS 71

C. IEEE 488 OVERVIEW 72
WHAT IS THE IEEE 488 (GPIB)? 72
DESIGN OBJECTIVES 72
BUS CHARACTERISTICS 73
DATA RATE 73
MULTIPLE DEVICES 74
BUS LENGTH 74
BYTE-ORIENTED 74

BLOCK-MULTIPLEXED 74
INTERRUPT-DRIVEN 75
DIRECT MEMORY ACCESS (DMA) 75
ASYNCHRONOUS TRANSFERS 75
I/O-TO-I/O TRANSFERS 75
IEEE 488 SIGNAL LINES 75
DATA BUS 76
MANAGEMENT BUS 76
TRANSFER BUS 77
IEEE 488 INTERFACE FUNCTIONS 78
THE IEEE 488 CONNECTOR 79
IEEE 488 SIGNAL LEVELS 80

D. IEEE 488 REMOTE MESSAGE CODING 81
INTRODUCTION 81
MESSAGE CODING 82

E. IEEE 488 DATA RATES 84
INTRODUCTION 84
DATA RATES 84

WHAT'S IN THIS MANUAL?

Editor's Note: *This manual originally documented both the ZT 1444A and the ZT 1488A Interface boards. Please note that Ziatech has discontinued the ZT 1488A, and has therefore removed from this manual whole topics devoted exclusively to the ZT 1488A. Please ignore any incidental references to the ZT 1488A still contained in this manual.*

This manual describes Ziatech's ZT 1444A IEEE 488 Interface for Personal Computers and explains how to use it. Every effort is made to include all the information you will need to get quick, accurate results from your personal computer-based system.

The following summarizes the focus of each major section in this manual.

Chapter 1, "Introduction**,"** offers an overview of the ZT 1444A IEEE 488 interface. It includes a product definition, a list of hardware and software features, and a discussion of I/O expansion possibilities for the interface. If you are evaluating the ZT 1444A to determine if it fits your needs, this information will be especially useful to you.

Chapter 2, "Getting Started**,"** summarizes the information essential to getting your ZT 1444A up and running.

Chapter 3, "Theory Of Operation**,"** presents an operational overview of the ZT 1444A by subdividing the boards into blocks and describing the function of each block in detail.

Chapter 4, "Hardware**,"** provides a summary of current IBM PC/XT DMA and interrupt hardware utilization.

Chapter 5, "Interrupts and DMA**,"** presents a discussion of selectable interrupt lines and DMA requests generated by the ZT 1444A interface board.

Chapter 6, "The IEEE 488 Interface (NAT9914BPD)**,"** explains the use of the Texas Instruments NAT9914BPD IEEE 488 adapter to implement the IEEE 488 bus interface.

Chapter 7, "IEEE 488 Transceivers (75160/75162)**,"** discusses the 75160/75162 transceiver chips that ensure all relevant bus driver/receiver specifications are met.

Chapter 8, "Security Key Interface**,"** provides a description of the Dallas Semiconductor DS1204 electronic key used for securing software and machine operation.

Appendix A, "Jumper Configurations**,"** provides detailed descriptions of the ZT 1444A jumper selectable options which are summarized in "Getting Started."

Appendix B, "Customer Support**,"** offers a product revision history, technical assistance, and the necessary information should you need to return your ZT 1444A for repair.

Appendix C, "IEEE 488 Overview**,"** provides an introduction to the IEEE 488 GPIB (HP-IB, IEC) bus specification.

Appendix D, "IEEE 488 Remote Message Coding**,"** lists the encoding required for all messages capable of being sent or received by an interface function.

Appendix E, "IEEE 488 Data Rates**,"** illustrates theoretical data rates for sending and receiving data.

1. INTRODUCTION

The Ziatech ZT 1444A Interface for Personal Computers gives an IBM PC™, XT™, AT™, or equivalent the ability to control IEEE 488-compatible equipment in a variety of applications such as product testing and laboratory automation. Each controller occupies one PC I/O slot and accommodates up to 15 of the more than 4,000 instruments, peripherals, computers, and other devices that share this popular interface. The IEEE 488-1978 Digital Interface for Programmable Instrumentation, also known as the General Purpose Interface Bus (GPIB), conforms to a well-defined specification that you can obtain from the following address:

IEEE Service Center
P.O. Box 1331
Piscataway, New Jersey 08855-1331

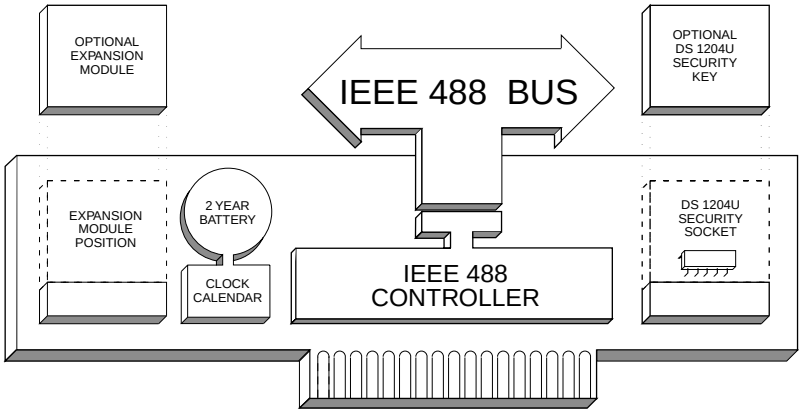
PRODUCT DEFINITION

The ZT 1444A and ZT 1488A differ in size and functionality. The ZT 1444A is 5.2" (13.2 cm) long. Because it supports extra features, the ZT 1488A is 10.5" (26.7 cm) in length. The ZT 1444A's capability includes IEEE 488 control and an optional security key. See Chapter 8, "Security Key Interface" for additional information. The ZT 1488A contains IEEE 488 control, a clock/calendar, and an expansion socket. The on-board clock/calendar reduces the need for entering the time and date upon power-up. It can be used as an interval timer or it can provide a stream of interrupts for exact measurement pacing.

Editor's Note: *This manual originally documented both the ZT 1444A and the ZT 1488A Interface boards. Please note that Ziatech has discontinued the ZT 1488A, and has therefore removed from this manual whole topics devoted exclusively to the ZT 1488A. Please ignore any incidental references to the ZT 1488A still contained in this manual.*

FUNCTIONAL BLOCKS

A functional block diagram of the ZT 1444 is Illustrated below.



Functional Block Diagram

HARDWARE FEATURES OF THE ZT 1444A

The ZT 1444A has the following features:

- Fully compatible with the IEEE 488 *Standard Interface for Programmable Instrumentation*
- Automatic direct memory access (DMA) sharing with other I/O devices using the PC's built-in DMA
- DMA channel user-selectable
- Eight I/O port addresses
- Interrupt enabling and disabling capability
- User-selectable interrupt line
- System controller enabling and disabling
- The ZT 1444A fits into the short format PC I/O slots
- Security key socket for Dallas Semiconductor DS 1204

SOFTWARE FEATURES OF THE ZT 1444A

The IEEE 488 interfaces are supported by comprehensive software that provides the PC/XT/AT (or compatible) user complete access to IEEE 488 devices as defined by the IEEE 488 specification. Optional software is available in efficient linkable format for controllers and talker/listeners, and in installable device driver format. These optional software packages support the following languages:

- Linkable Controller and Talker/Listener (C.488)
- Borland C++,
- Turbo C™
- Microsoft C™ and compatible
- Installable Device Driver (EZ.488)
- Not language-dependent

The standard software included with purchase is capable of the following:

- Initializing IEEE 488 devices
- Sending and receiving IEEE 488 device messages/data
- Polling IEEE 488 devices

Ziatech also supplies an interactive program with each software product. This program allows you to exercise IEEE 488 *send data* and *receive data* functions without writing programs. A menu-driven question and answer session is all you need to use EZTEST. This is handy for checking unknown devices as well as for verifying system operation.

The source diskettes also include example files to help you develop your application.

2. GETTING STARTED

This section summarizes the information essential to getting your ZT 1444A up and running. You should read this section before you attempt to use the board.

UNPACKING

Please check the shipping carton for damage. If the shipping carton and contents are damaged, notify the carrier and Ziatech for an insurance settlement. Retain the shipping carton and packing material for inspection by the carrier. ***Do not return any product to Ziatech without a Return Material Authorization (RMA) number.*** "Returning For Service" in Appendix B explains the procedure you should follow to obtain an RMA number from Ziatech.

WHAT'S IN THE BOX?

The items listed below are included with a ZT 1444A order.

- The ZT 1444A IEEE 488 interface board in anti-static bag
- Either the standard software disk (EZ.488) supporting all DOS languages, or optional software disk(s), if ordered

Save the anti-static bag for storing or returning the ZT 1444A.



Warning: Like all equipment utilizing MOS devices, the ZT 1444A must be protected from static discharge. Never remove any of the socketed parts except at a static-free workstation. Use the anti-static bag shipped with the ZT 1444A to handle the board.

SYSTEM REQUIREMENTS

The Ziatech ZT 1444A IEEE 488 Interface for Personal Computers must be installed in an IBM PC/XT/AT, TI, or equivalent personal computer, including the PS/2 Model 30. Support for Interpreted and Compiled BASIC comes with your board, unless you have ordered support for a separate language. Software is available in linkable format and as a DOS Installable Device Driver. Many third party software developers, such as Asyst Software Technologies, also support the ZT 1444A.

The ZT 1444A requires 0.70 A maximum. The ambient temperature must be maintained between 0° and +65° Celsius to guarantee operation of either board.

SETTING UP YOUR WORKING DISKS

Before using your software, copy the distribution diskette onto a working diskette. This prevents corruption of the original should you make mistakes while learning to use the IEEE 488 driver subroutines.

To make your backup, use the DISKCOPY utility provided with your MS-DOS™ or PC DOS™ system. See the DOS reference manual for further details.

Store the distribution diskette in a cool, dry, anti-static environment. Be sure your working copy is clearly marked with the Ziatech software version number.

SETTING UP YOUR INTERFACE BOARD

The IEEE 488 bus can function as three types of devices:

- Talker
- Listener
- Controller

A listener can be addressed by an interface message to receive messages or data. A talker can be addressed by an interface message to send data. A controller can address instruments (talkers and/or listeners) to send or receive data and can also send other interface messages. Most controllers have talker/listener capability as well.

The Ziatech IEEE 488 interface board can perform any of the three device functions. As a controller and talker/listener, the board can control up to 15 other IEEE 488-compatible devices or instruments.

No changes from the factory default jumper configuration are necessary when the board is used as a controller with Ziatech's standard software. It may be useful, however, to review the jumper descriptions; see the "ZT 1444A Jumper Descriptions" topic.

Some jumper changes are required when the board is used as a device. These changes are listed in the jumper description tables mentioned in the preceding paragraph. A complete description of jumper functions can be found in the "Jumper Configurations" topic.

Because devices do not drive the Interface Clear (IFC) and Remote Enable (REN) signals, the system controller option is not needed.

INTERFACE AND SOFTWARE CAPABILITIES

Ziatech's IEEE 488 interface boards use the NAT9914BPD chip, which supports all capabilities shown in the NAT9914BPD Capabilities table shown below. You can implement any of these functions by using Ziatech's software or by writing your own. Some infrequently used functions, such as the Pass Control capability, are supported by the NAT9914BPD but require additional software.

You can also obtain technical assistance from Ziatech; see "Technical/Sales Assistance" in Appendix B.

NAT9914BPD Capabilities.

SH1 (2.3):	Complete Source Handshake capability
AH1 (2.3):	Complete Acceptor Handshake capability
TE1 (2.5):	Extended Talker capability (secondary address allowable)
LE1 (2.6):	Extended Listener capability (secondary address allowable)
SR1 (2.7):	Service Request capability
RL1 (2.8):	Remote Local capability
PP1 (2.9):	Parallel Poll capability
DC1 (2.0):	Device Clear capability
DT1 (2.11):	Device Trigger capability
C1-4, C9:	Controller capability
	C1-4: System controller, IFC, REN, and SRQ capability
	C9: Messages, Receive/Pass Control, Parallel Poll, and Take Control Synchronously capability

INSTALLING YOUR INTERFACE BOARD

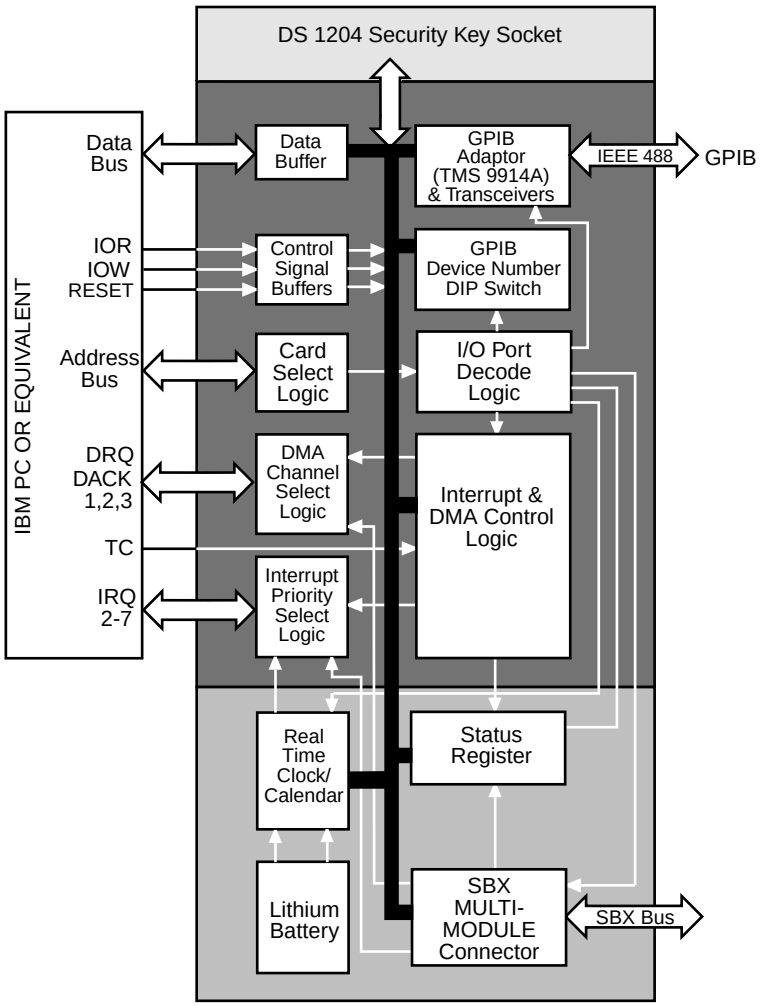
You can install the 488 interface board in an IBM PC, XT, AT, or equivalent computer. To install your interface board, follow the steps below.

1. Turn off the power to your system unit and disconnect the line cord.
2. Turn off the power to all externally attached devices (printer, display, etc.).
3. Remove the computer cover mounting screws and the computer cover.
4. Remove the slot cover screw and the existing slot cover.
5. Hold the interface board by the top. Firmly press it into the expansion slot while aligning the IEEE 488 connector with the expansion slot in the rear panel.
6. Re-install the slot cover screw.
7. Re-install the computer cover and connect the line cord.

You can test the installed interface by connecting it to an instrument and using the test routines supplied with the optional software.

3. THEORY OF OPERATION

This section presents an operational overview of the ZT 1444A hardware. The ZT 1444A Block Diagram shows the board's operations divided into sections. Refer to this diagram as you proceed through the chapter; the following topics discuss each major part of the diagram in order from top to bottom.



ZT 1444A Block Diagram

IEEE 488 ADAPTER AND TRANSCEIVERS

The heart of the ZT 1444A is the National Instruments IEEE 488 Adapter (NAT9914BPD).

The IEEE 488 adapter allows communication with up to 15 other IEEE 488-compatible devices. When used as a controller, it can control 15 other IEEE 488-compatible devices. When used as a device, it can be controlled by 14 other IEEE 488-compatible devices. Key features include pass control, parallel and serial poll, and secondary address capability. The NAT9914BPD interface adapter fully adheres to the IEEE 488 standard.

The IEEE 488 adapter chip (NAT9914BPD) interfaces with the IEEE 488 connector through a pair of on-board transceivers, 75160A/75162A. These transceivers convert the on-board TTL levels to IEEE 488 signal compatibility and prevent power-up or power-down glitches from affecting the IEEE 488 bus operation.

The IEEE 488 rear panel connector used on the ZT 1444A accepts standard stackable IEEE 488 connectors. Stainless steel, fiber-filled polycarbonate construction shields against EMI leakage.

DIP SWITCHES AND JUMPERS

On-board DIP switches and jumpers let you choose the IEEE 488 device address, card select address, DMA channel, and interrupt level. For the device address you can either read the DIP switch or arbitrarily set the address in software. Ziatech software uses this DIP switch for its device address. For card select addresses, you can individually set the base port address for the adapter, the clock, and the SBX expansion module (ZT 1488A only). You can choose the appropriate DMA channel and interrupt level with jumpers if you want to use these features.

CARD SELECT LOGIC, I/O PORT DECODE LOGIC

I/O port address decoding helps the computer determine which on-board function it is dealing with. The IEEE 488 adapter has eight addresses, the clock/calendar (ZT 1488A only) has 32 addresses, and the SBX bus (ZT 1488A only) has 16 unique I/O port addresses.

DMA CONTROL LOGIC

DMA and interrupt request buffers with logic for character counts, data locations, etc., allow the IEEE 488 interface to manage these functions after completion of the initial software-driven setup.

DMA CHANNEL SELECT LOGIC

The IEEE 488 data transfer rates (450 Kbytes) are very high when used with the PC's DMA facility. You may choose which DMA channel to use by means of jumpers (Ziatech controller software assumes Channel 1).

INTERRUPT PRIORITY SELECT LOGIC

You can generate interrupts in several ways. You must select the PC interrupt line you wish to use by means of jumpers. Additional ZT 1488A interrupt select jumpers accommodate other interrupt sources, including the expansion module and clock.

SECURITY KEY OPTION

The ZT 1444A provides a socket for an optional security key device. A software package can use this device interactively to prevent use of the software on more than one specific ZT 1444A. See Chapter 8, "Security Key Interface" for additional information.

SOFTWARE INTERFACING

Ziatech provides a variety of software packages designed to get your application up and running as easily as possible. Each ZT 1444A is supplied with a software package called EZ.488. This is a DOS installable device driver that interfaces through the file system and is not language dependent. Non-DOS-based software is optionally available.

4. HARDWARE

The hardware descriptions in this and the following sections apply to the ZT 1444A interface board. The interface generates interrupts and DMA requests to the Personal Computer via selectable interrupt lines. Subsequent sections discuss these interrupt lines. This section provides a summary of current IBM PC/XT DMA and interrupt hardware utilization.

SUMMARY OF DMA AND INTERRUPT LINE USAGE

IBM PC	
DRQ0/DACK0	RAM refresh
DRQ1/DACK1	Unused, 20-bit address
DRQ2/DACK2	Floppy Disk, 20-bit address
DRQ3/DACK3	Unused, 20-bit address
IRQ2	Unused
IRQ3	SDLC Communications option, otherwise available
IRQ4	SDLC Communications option, otherwise available
IRQ5	Unused
IRQ6	Floppy disk
IRQ7	Monochrome display
IBM XT/AT	
DRQ0/DACK0	RAM refresh
DRQ1/DACK1	SDLC Communications option, otherwise available, 20-bit address
DRQ2/DACK2	Floppy disk, 20-bit address
DRQ3/DACK3	Hard disk, 20-bit address
IRQ2	Unused
IRQ3	COM2 option
IRQ4	COM1 option
IRQ5	Hard disk (IBM XT)
IRQ6	Floppy disk
IRQ7	Monochrome display / LPT port
TI PC Interrupt Usage	
IR0, IR1, IR4	Unused
IR2	Timer
IR5	Parallel printer
IR6	Floppy disk

5. INTERRUPTS AND DMA

The ZT 1444A interface board generates interrupts and DMA requests through selectable interrupt lines. These interrupt lines are discussed in this section.

ZT 1444A INTERRUPTS AND DMA

The ZT 1444A can interrupt the PC on one or more of six interrupt lines. Use parallel jumpers W1-W6 to select any one of the IRQ2-IRQ7 lines. Refer to the figure ZT 1444A Interrupt Structure for an illustration of the ZT 1444A interrupt structure.

To enable the IEEE 488 interrupt, mask the NAT9914BPD INT0 and INT1 registers appropriately. "The IEEE 488 Interface (NAT9914BPD)" supplies additional information.

You can use the ZT 1444A Control register to disable interrupts without changing the NAT9914BPD Interrupt Mask registers.

ZT 1444A Control Register

(Base + 0002h, Write)

The Control register enables and disables various features. This is a 5-bit register with the following definitions:

D0: 1 - Reset DMA Terminal Count (TC) interrupt flip-flop
0 - Enable DMA TC interrupt flip-flop

D4: 1 - Disable DMA TC interrupt and DMA request
0 - Enable DMA TC interrupt and DMA request

D5: 1 - Disable interrupt output buffer
0 - Enable interrupt output buffer

D6: 1 - Enable open collector operation
0 - Enable three-state operation

D7: 1 - Disable system controller operation
0 - Enable system controller operation

After reset, the control register has the DMA TC interrupt flip-flop reset, DMA TC interrupt and DMA request enabled, interrupt output buffer enabled, three-state operation enabled, and system controller enabled.

When set, D5 enables the interrupt output buffer to generate an interrupt on IRQ6 when one jumper on W1-W6 is installed. The interrupt output has two sources: one is the output of the DMA TC flip-flop, and the other is from the NAT9914BPD. The NAT9914BPD interrupts are enabled via the interrupt mask registers. The TC flip-flop is enabled only when D4 is enabled.

Note: D4 also enables output of the DMA request line on the IBM backplane if W7, W9, or W11 are installed.

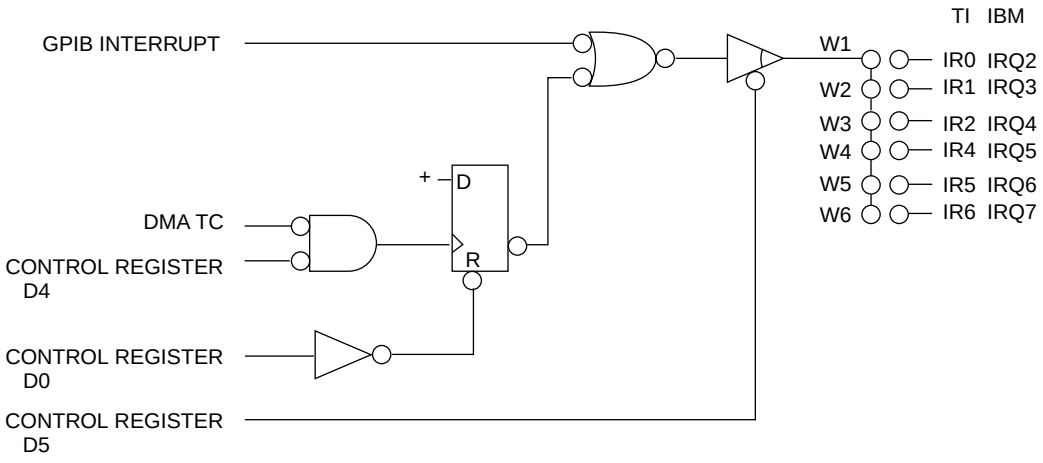
If you want to use the TC interrupt, enable the interrupt output buffer D5, the DMA TC interrupt, and the DMA request control D4. To clear the interrupt, write D0 with a 1.

D6 selects either a three-state or an open collector operation for the IEEE 488 bus drivers. For maximum performance, use three-state operation and use the Fast or Very Fast TI commands with the NAT9914BPD. ZT 1444A DMA Structure

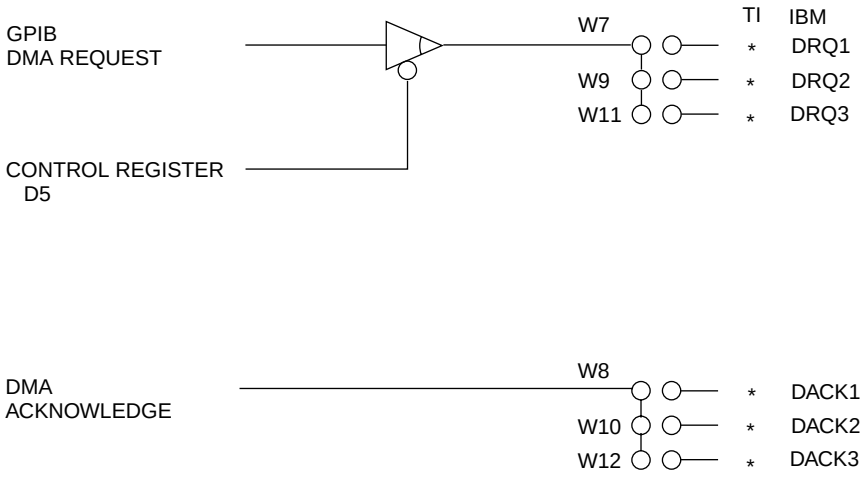
Use D7 to select system controller operation for the IEEE 488 bus. This control is typically used only in IEEE 488 systems that pass control.

The ZT 1444A can request DMA transfers for the IEEE 488 on one of the three DMA request/acknowledge lines (DRQ1-DRQ3, DACK1-DACK3. Refer to the ZT 1444A DMA Structure figure. If you use DMA with the IEEE 488, select one DMA request line, with its corresponding acknowledge line, via jumpers W7-W12. For DRQ1, insert W7 and its corresponding DACK1 W8 jumpers. For DRQ2, insert W9 and its corresponding DACK2 W10 jumpers. For DRQ3, insert W11 and its corresponding DACK3 W12 jumpers.

Note: Ziatech controller software that uses DMA utilizes DACK1 and DRQ1, corresponding to jumpers W8 and W7. Install these jumpers if you use DMA.



ZT 1444A Interrupt Structure



*TI PC does not support DMA.

ZT 1444A DMA Structure

6. THE IEEE 488 INTERFACE (NAT9914BPD)

The ZT 1444A uses the National Instruments NAT9914BPD IEEE 488 adapter to implement the IEEE 488 bus interface. Refer to the NAT9914BPD Block Diagram figure. The adapter interfaces to the IEEE 488 bus via IEEE 488 transceivers and is mapped into the CPU I/O system. It has 13 accessible registers, seven write and six read. All communication between the IEEE 488 and the PC's microprocessor is carried out using these registers. A summary of each register appears in the NAT9914BPD I/O Port Descriptions table shown below.

NAT9914BPD I/O Port Descriptions

I/O Port Address Base+	I/O Read Register	I/O Write Register
0000h	Interrupt Status 0	Interrupt Mask 0
0001h	Interrupt Status 1	Interrupt Mask 1
0002h	Address Status	Control Register/Interrupt Mask 2/ End of String/Accessory
0003h	Bus Status	Aux CMD
00004h	Address Switch/ Interrupt Status 2	Address Register
0005h	Serial Poll Status	Serial Poll
0006h	CMD Pass Thru	Parallel Poll
0007h	Data In	Data Out

NAT9914BPD REGISTERS

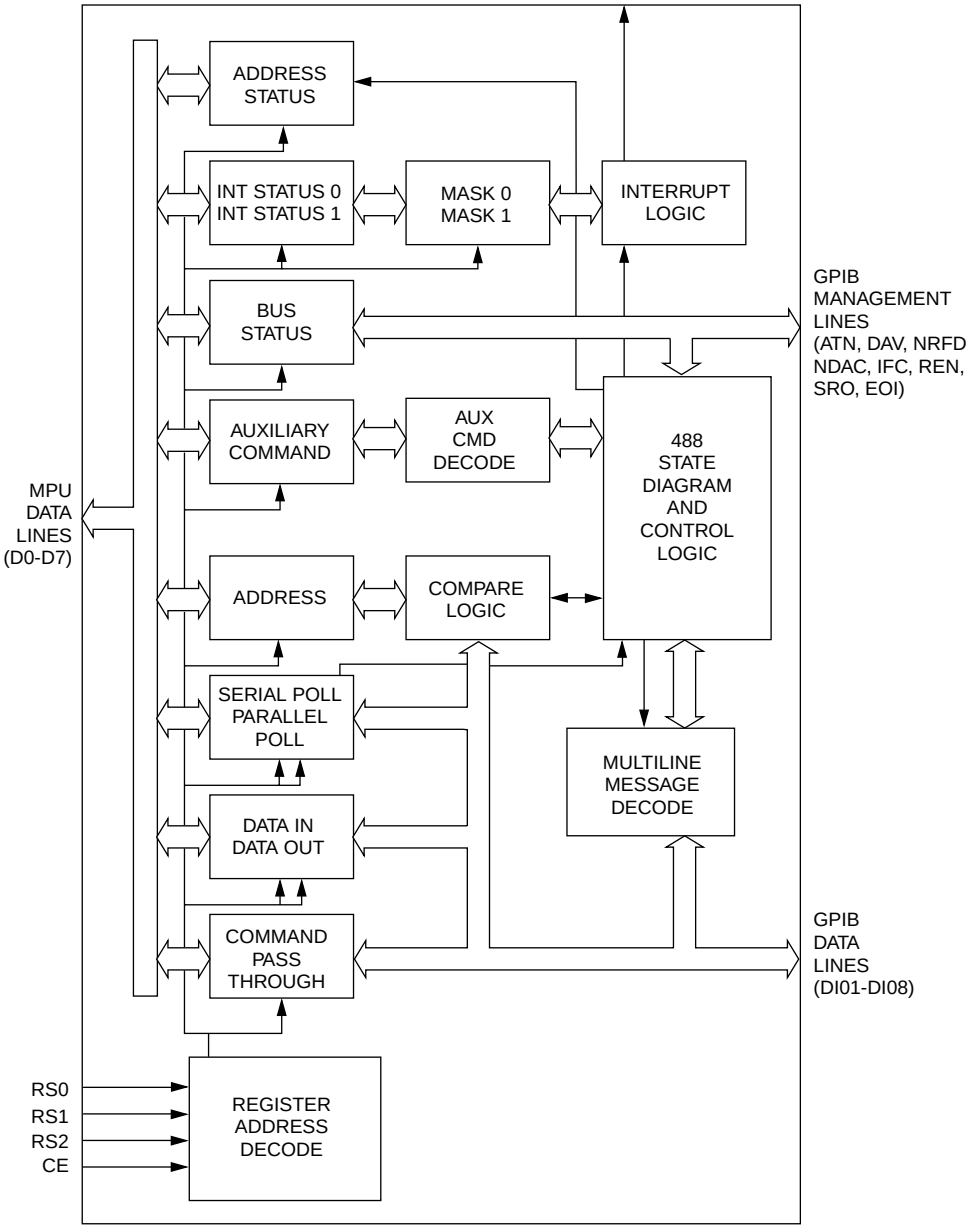
The following topics describe registers used for programming the NAT9914BPD IEEE 488 Interface.

Address Register - Talker/Listener

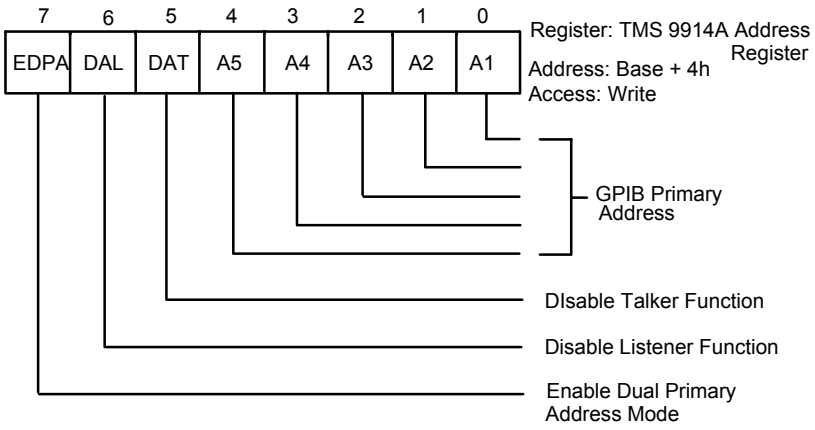
(Base + 4h, Write)

The Address register (ADDR) is a write-only register at Base + 4h that is written when the IEEE 488 interface is used as an IEEE 488 talker/listener but not as a controller. The Address register engages three major functions of the ZT 1444A as an IEEE 488 talker/listener:

- Establish the 5-bit IEEE 488 address
- Configure the IEEE 488 interface as either a talker or a listener, or both
- Enable dual IEEE 488 primary addressing



NAT9914BPD Block Diagram



NAT9914BPD Address Register

The following topics describe the NAT9914BPD Address register bits:

- A5-A1
- DAL/DAT (Disable Listener/Disable Talker)
- EDPA (Enable Dual-Primary Addressing)

A5-A1

Every IEEE 488 device requires a 5-bit address to distinguish it from other IEEE 488 devices. Address register bits A5-A1, when written to, establish the 5-bit IEEE 488 address of the IEEE 488 interface. You can obtain the address by reading the on-board DIP switch at the Address Switch register and also at port Base + 4h (see the "Address Switch Register - General Purpose" description). The address 11111B is not allowed by the IEEE 488 standard.

The System Reset signal generated by the CPU resets the Address register so that the IEEE 488 interface acts as a single IEEE 488 talker/listener with an address of 0. Your initialization, therefore, must enable the IEEE 488 talker/listener and simultaneously write the IEEE 488 address A5-A1. You can accomplish this in the following manner:

1. Select the IEEE 488 address using the five least significant bits of the on-board DIP switch SW4.

2. Read the DIP switch via the Address Switch register at port Base + 4h.

3. Mask out the remaining three bits by ANDing the DIP switch value with 1Fh (this bit pattern also enables talker/listener and single primary addressing).

4. Write the masked address to the Address register at port Base + 4h.

The IEEE 488 interface is then enabled for a single IEEE 488 primary talker/listener address of A5-A1.If you wish, you may skip steps 1-4 and write out the correct pattern to the Address register for your particular system.

Note that the Address register is not cleared by a hardware or software reset. The NAT9914BPD as a controller talks and listens through use of TON and LON auxiliary commands and is not addressed as described above.

DAL/DAT

The disable-talker (DAT) and disable-listener (DAL) bits configure the IEEE 488 interface to be a talker, a listener, or both. The DAT and DAL bits enable the talker/listener functions in an inverse order; you need to think in reverse, therefore, to understand these functions. To enable the board as an IEEE 488 talker, disable the IEEE 488 listener function by writing a 1 to the DAL bit. To enable the board as a listener, disable the talker function by writing a 1 to the DAT bit. To enable the board as both an IEEE 488 talker and listener, don't disable either feature; rather, write zeros to bits DAT and DAL.

EDPA

Dual IEEE 488 addressing lets you partition the IEEE 488 interface into two separate devices. An example is one in which the IEEE 488 interface is programmed to measure temperature and pressure. One IEEE 488 address pertains to temperature measurement, the other to pressure measurement. Enabling the enable-dual-primary-addressing (EDPA) bit makes the IEEE 488 interface ignore IEEE 488 address bit A1, giving the board two consecutive IEEE 488 primary addresses. Be careful not to confuse this with the IEEE 488 primary-secondary addressing scheme that is described elsewhere (see the "TPAS/LPAS" topic). You can determine which of the dual primary addresses was sent (by the controller) by reading the Upper-Lower-Primary-Address bit (ULPA) in the Address Status register (see the "ULPA" topic). The ULPA bit is actually an image of the missing IEEE 488 address bit A1 that was ignored in the EDPA mode.

Note: You may want to build a system that uses a separate IEEE 488 talker and listener with the same IEEE 488 address. While the PC/IEEE 488 could be programmed to function in this manner, we recommend you avoid this mode of operation for two reasons: first, non-unique IEEE 488 addresses in the same system are very confusing; and second, the interrupt registers cannot differentiate between a talker or listener being addressed (see the discussion on NAT9914BPD interrupt registers in the "Interrupt Mask/Status Registers" topic).

Address Switch Register - General Purpose

(Base + 4h, Read)

The Address Switch register is actually a read-only port (Base + 4h) that reads the contents of the DIP switch at location 1D.

You would normally write the contents of the Address Switch register (that is, the DIP switch setting) to the Address register to enable the IEEE 488 talker/listener address. You may, however, use the DIP switch to input anything you find necessary in your system design. The "NAT9914BPD Address Switch Register" figure provides useful information on how to implement the DIP switch. This is how the IEEE 488 interface defines the DIP switch.

Switch positions SW5-SW1 represent the 5-bit IEEE 488 address. The switch position SW6 is user-defined for one of two possible operations. SW7, when in the *on* position, forces the IEEE 488 drivers into the open collector mode. Normally, for three-state operation, SW7 is *off* . The switch position SW8, when *off*, enables the interface as the system controller. When *off*, REN and IFC cannot be asserted. The DIP switch closures are inverted so that the *on* or *closed* position of a switch represents a binary 1.

The board is shipped from the factory with the DIP switch set for system controller operation as shown in the NAT9914BPD Address Switch Register figure below. The IEEE 488 address is 3; the user-defined DIP switch position is set for 0. The drivers are enabled for three-state operation. The Address Switch register thus reads C3h or 11000011B.

D7	D6	D5	D4	D3	D2	D1	D0
SW8	SW7	SW6	SW5	SW4	SW3	SW2	SW1
S.C.	O.C.	U.D.	GPIB Address				

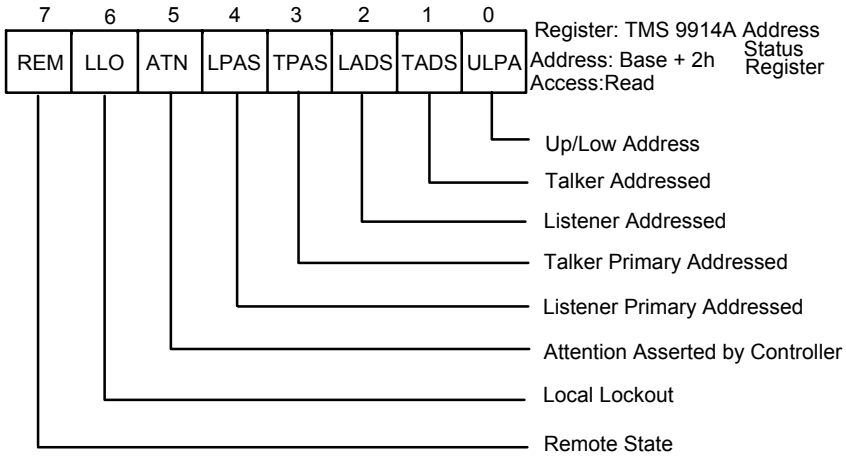
NAT9914BPD Address Switch Register

Address Status Register - Talker/Listener

(Base + 2h, Read)

The Address Status register (ADRST) is a read-only port at Base + 2h that is read only when the IEEE 488 interface is used as an IEEE 488 talker/listener. The Address Status register contains the IEEE 488 address status of the interface, which is determined by the current IEEE 488 controller-in-charge. The address status is not latched, which means it is valid only at the time of reading. This implies the IEEE 488 controller may change the address status at any time during or after reading; therefore, you should

carefully study the normal logical protocol of the IEEE 488 to anticipate any change in address status. Consult the NAT9914BPD Address Status Register figure below for details.



NAT9914BPD Address Status Register.

The following topics describe the NAT9914BPD Address register bits:

- ULPA (Upper-Lower-Primary Address)
- TPAS/LPAS (Talker or Listener Primary Addressed State)
- TADS/LADS (Talker or Listener Addressed State)
- ATN (Attention)
- LLO (Local Lockout)
- REM (Remote Enable)

ULPA

The ULPA bit detects upper or lower dual primary IEEE 488 addresses. Refer to the Address register "EDPA" bit description before you proceed. The only difference between dual primary addresses is the state of the least significant address bit A1 during addressing. If A1 is low, the lower address is the valid address and ULPA is cleared. If A1 is high, the higher address is valid and ULPA is set.

The ULPA bit feature is active regardless of whether or not you selected primary addressing. Once the ULPA bit is set, it can be cleared only by a valid address from the IEEE 488 interface with A1 equal to 0; by the Interface Clear (IFC) signal on the IEEE 488; or by removing power. Remember that the ULPA bit pertains only to the last valid interface address sent by the controller. Do not confuse dual primary addressing with secondary addressing, which is discussed in detail under "TPAS/LPAS".

TPAS/LPAS

The Talker or Listener Primary Addressed state (TPAS or LPAS) bits indicate the IEEE 488 primary talk or listen address of the IEEE 488 interface has been sent by the controller and that the IEEE 488 hardware has acknowledged that fact by settling into the talker or listener primary addressed state. Consult the IEEE 488 standard for the state diagrams.

The TPAS and LPAS bits are used when a secondary address is required to form a complete address. In normal primary addressing mode, a single 5-bit IEEE 488 address differentiates between 32 possible IEEE 488 addresses. Some systems require that you implement more than 32 device addresses; that is, a device within a device or a function within a device must be specified. This is typical of multiprocessor systems in which many subroutines must be specified via a second, or *secondary* address. The TPAS and LPAS indicate, therefore, that while a secondary address may be required, only the primary address has been received.

The IEEE 488 standard limits the total number of devices on the bus to 16 including the controller. This is a bus loading limitation and not a logical addressing restriction.

TADS/LADS

The Talker or Listener Addressed state (TADS or LADS) bits indicate the IEEE 488 interface has been fully addressed by the controller. If only primary addressing is used, TADS and TPAS or LADS and LPAS occur at the same time; that is, a single IEEE 488 primary address sent by the IEEE 488 controller completes the addressing state.

If secondary addressing is employed, TADS or LADS indicate both primary and secondary addresses have been received by the IEEE 488 interface. Secondary addressing is discussed in greater detail later in this chapter; see "APT".

The TADS or LADS bits do not necessarily mean the IEEE 488 interface is ready to talk or to listen. In order for the IEEE 488 interface to talk over the IEEE 488, you must closely monitor the Byte-In (BI) and Byte-Out (BO) bits in the Interrupt Status 0 register (INT0). See the INT0/INT1 discussion in "Interrupt Mask/Status Registers" for details.

ATN

The Attention (ATN) bit indicates the level of the IEEE 488 Attention line. Only the IEEE 488 controller currently in charge asserts ATN. When data is present on the IEEE 488 data bus and ATN is asserted, the data is actually an IEEE 488 bus message such as a talk or listen address. When data is present without ATN, then it is simply data. Advanced IEEE 488 system designers often need to know the level of the ATN line in order to determine the current IEEE 488 state of a device.

LLO

The Local Lockout (LLO) bit indicates the IEEE 488 interface has received the Local Lockout message. LLO is a message sent by the IEEE 488 controller to tell the talker/listeners to ignore their front panel controls, if any. This is useful in a system that needs to protect against an accidental switch closure at a control panel or against an inexperienced operator.

REM

The Remote Enable (REM) bit indicates the Remote Enable (REN) line on the IEEE 488 has been asserted by the controller and the IEEE 488 interface is in the Remote Enable state. The REN line lets a talker/listener (in this case the IEEE 488 interface) know it is enabled to be remotely programmed by the controller. Some devices ignore the REN line; that is, they accept control at any time from a controller.

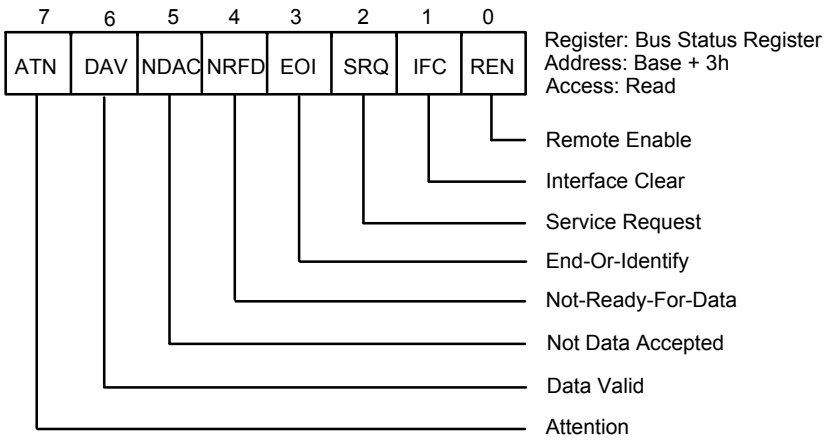
The REM bit has another subtle function. Power-up time for some systems presents many problems not incurred during normal operation. The system controller should power up, initialize, pulse Interface Clear (IFC), and assert REN. The IEEE 488 interface can then detect REN via the REM bit. REN tells the IEEE 488 interface the controller has powered up successfully and is ready to control the IEEE 488. The IEEE 488 interface talker/listener can then safely proceed.

Bus Status Register - Debugging

(Base + 3h, Read)

This read-only, non-latched register obtains the status of the IEEE 488 bus management lines. The Bus Status register (BUSTR) is not normally used in a system; its main purpose is to debug the IEEE 488 should a catastrophic failure occur. All eight IEEE 488 control lines can be monitored. The bits are positive true logic values of the IEEE 488 management lines. Note that this information is obtained from the internal logic of the NAT9914BPD and that no mechanism is provided to prevent status bits from changing during a read cycle.

If the IEEE 488 is configured as the system controller and is sending IFC, then the IFC bit in this register is not set. Refer to the figure below for bit assignments.



NAT9914BPD Bus Status Register.

Command Pass-Through Register - Talker/Listener

(Base + 6h, Read)

This read-only port is the Command Pass-Through register (CPTRG). It monitors the IEEE 488 data lines in a way similar to how the Bus Status register monitors the IEEE 488 control lines. It is a non-latched, unqualified image of the IEEE 488 data lines that may be read at any time. This register, normally used when the IEEE 488 interface is a talker/listener, reads secondary addresses, unrecognized commands, and secondary commands.

The register contents are not latched; therefore, you must suppress the handshake, thus forcing the data to remain stable long enough to read the address or command and then respond correctly. You must then complete the handshake to allow new data on the bus.

Handshake manipulation is controlled by the Auxiliary Command register; see "Auxiliary Command Register" for details. Handshake suppression is also affected by the Address Pass-Through bit in the Interrupt Mask 0 register; see the "Interrupt Mask/Status Registers" discussion.

Although the IEEE 488 standard does not permit you to define your own commands, provision for upgrades of the standard is made by the Command Pass-Through register. The number of possible available commands for future IEEE definition is thus increased. You can generate an interrupt to prompt the CPU to read the Command Pass-Through register.

When the IEEE 488 interface is the IEEE 488 controller, you can also use this register to obtain the parallel poll status bits when conducting a parallel poll. See "Parallel Poll Register - Talker/Listener" for details.

Parallel Poll Register - Talker/Listener

(Base + 6h, Write)

The Parallel Poll register (PRPR) is used only when the IEEE 488 interface is an IEEE 488 talker/listener. The parallel poll feature is used when the IEEE 488 controller needs to simultaneously check the request-for-service status of up to eight talker/listeners. Each of the eight devices has a dedicated IEEE 488 data line to drive when parallel-polled by the controller. When the IEEE 488 interface needs the attention of the IEEE 488 controller and the parallel poll feature is used, the IEEE 488 interface must save its own user-defined internal status indicating a request for service. When the controller routinely performs a parallel poll, the IEEE 488 must place a *yes* or *no* status bit on its own dedicated IEEE 488 data line. The mechanism for doing this is discussed below.

Note: Since most systems use Serial Poll rather than Parallel Poll because it is easier to implement, we recommend you use Serial Poll.

Whenever the Attention (ATN) and the End-Or-Identify (EOI) line on the IEEE 488 are asserted together by the controller, the contents of the IEEE 488 interface Parallel Poll register are asserted on the IEEE 488 data bus. A hardware reset clears the Parallel Poll register. You must execute a software reset (see the "Auxiliary Command Register" discussion) before writing to the Parallel Poll register.

You can write anything to the Parallel Poll register but to give each device a dedicated IEEE 488 data line from which to request service, only one bit of the parallel poll response byte may be active at any time. If the system uses a positive sense bit to indicate service requested, the byte you write to the Parallel Poll register must consist of one bit high with the remaining seven bits low. If negative sense is used, the complement byte must be written: one bit low and seven bits high. This is the normal PP mode because of the electrical nature of open collector drives with passive pull-ups.

If there are more than eight devices on the IEEE 488 and the system requires parallel polling from each device, devices may share one of the eight IEEE 488 data lines. You must then implement a way to determine which instrument(s) sharing a data line actually requested service. The controller can sequentially interrogate each device, or set up another parallel poll subsystem in which previously polled devices do not participate in the poll.

7	6	5	4	3	2	1	0
DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1
PP8	PP7	PP6	PP5	PP4	PP3	PP2	PP1

NAT9914BPD Parallel Poll Register.

The IEEE 488 standard calls out two subsets of parallel polling capability: an easy one and a not-so-easy one. These are Parallel Poll Two (PP2) and Parallel Poll One (PP1), respectively.

Parallel Poll Subset PP2

Protocol for PP2 can be simple. With PP2, the IEEE 488 controller can conduct a parallel poll by simply asserting EOI (End-Or-Identify - we are using the *Identify* portion now) while the ATN line is asserted; that is, while the controller is actively in charge. Each IEEE 488 device participating in the parallel poll must send its parallel poll response bit to the IEEE 488 data bus within 200 ms. The controller can then read all the response bits as one data byte and take appropriate action.

Polling frequency is determined solely by the IEEE 488 controller. The controller must poll often for busy systems because the talker/listeners have no direct means to attract attention or to interrupt the controller when using parallel poll. Polling frequency is a main consideration when you are deciding whether to use parallel or serial polling.

With serial poll designs, any device may interrupt the controller by asserting the Service Request (SRQ) line on the IEEE 488. The controller can then serially interrogate each device for a serial response byte that not only describes whether the device needs service, but also indicates the type of service required with the remaining seven bits. See the "Serial Poll Register - Talker/Listener" topic.

Obviously, configuring for a parallel poll requires many system considerations. Each device must know which IEEE 488 data line to drive during a parallel poll.

When using the PP2 subset, the IEEE 488 interface initialization routine must write the correct response byte to the Parallel Poll register, setting the assigned bit if positive sense is used or resetting the assigned bit if negative sense is used. All other bits must be complements of the assigned response bit.

Note: Most systems that use parallel poll use PP2.

Parallel Poll Subset PP1

You can build a system in which the IEEE 488 controller tells each device how to configure its response byte. The PP1 subset is defined for this purpose.

The four least significant bits of the Parallel Poll Enable (PPE) message are designated S, P1, P2, and P3. The Sense (S) bit, corresponding to the fourth IEEE 488 data line, tells the device which polarity the parallel poll response bit must be to be true; that is, an affirmative response. The binary-weighted bits P1, P2, and P3 tell the device which IEEE 488 data line to use for the response bit. The remaining four bits not shown, in conjunction with S, P1, P2, and P3, make up the PPE message.

When the IEEE 488 interface (as a talker/listener) receives the SPE message, the software must read the message via the Command Pass-Through register; interpret the S, P1, P2, and P3 information; and store the assigned parallel poll response byte somewhere in memory. Up to this point, zeros should have been written in the Parallel Poll register to avoid confusion.

The suggested protocol for implementing PP1 parallel polling in which the IEEE 488 interface is a talker/listener is as follows:

1. After power-up and software reset, write 00 to the Parallel Poll register.

2. The IEEE 488 controller addresses the IEEE 488 interface to listen.

3. The controller sends the Parallel Poll Configure (PPC) message. The IEEE 488 interface reads the command via the Command Pass-Through register and then gets ready for the PPE message.

4. The controller sends the customized PPE message for the IEEE 488 interface, which reads the PPE message via the Command Pass-Through register again; interprets the S, P1, P2, and P3 information; and stores the byte for further use.

5. The IEEE 488 interface is then set up for PP1 parallel polling. If the IEEE 488 interface needs service from the controller, an affirmative response byte is written to the Parallel Poll register; otherwise, the negative byte is written.

6. Whenever the controller requests a parallel poll response byte, the controller asserts the ATN and EOI lines. The *yes* or *no* response in the Parallel Poll register of the IEEE 488 interface is automatically placed on the IEEE 488 data bus.

7. If or when the controller re-addresses the IEEE 488 interface to listen and sends the Parallel Poll Disable (PPD) message, the IEEE 488 interface must not write an affirmative response byte into the Parallel Poll register until the IEEE 488 interface is re-enabled by the controller by repeating steps 3 and 4. This feature allows several devices to share a parallel poll response line by disabling the devices that are known to need no service and by enabling the devices in question.

8. If or when the controller sends the Parallel Poll Unconfigure (PPU) message, the IEEE 488 interface may interpret this message to imply no more parallel poll activity will take place until the controller again sends the re-configure (PPC) message.

Parallel Poll Versus Serial Poll

A Parallel Poll service request differs from the Serial Poll service request in the following ways:

1. A device using the parallel poll facility is assigned its own dedicated bus line to send its request, whereas devices using the serial poll facility (SRQ) are addressed individually to send an identifying service request byte. Parallel poll saves the talk addressing time and can identify up to eight devices at once.

2. Devices using the serial poll facility (SRQ) can request service from the controller any time a device requires service, whereas service requests sent via the parallel poll facility can be sent only when solicited by the current controller. Thus, if speed in servicing requests is of utmost importance and there is little IEEE 488 bus activity between requests (permitting frequent parallel polls by the controller), servicing requests should be done by the parallel poll method. However, the serial poll method is by far the easiest to use and is applicable for the majority of IEEE 488 systems.

3. The serial poll mechanism implicitly tells the device that the controller has seen its request and that it may stop asserting SRQ. Parallel Poll has no equivalent mechanism; the system software in both the device and the controller must explicitly set up some convention to inform the device that its parallel poll response has been recognized.

Remember that protocol for the IEEE 488 bus has not been defined; it is left up to the designer. Bus messages and the effect thereof on IEEE 488 devices have been defined in such a manner that nearly all IEEE 488 devices are compatible when a reasonable systematic protocol is designed.

Parallel Poll IEEE 488 Drivers

Both parallel poll subsets require that open collector IEEE 488 transceivers be used to return the status byte when polled. The 75453 at pack location 12B automatically enables the IEEE 488 driver, pack 9A, for open collector operation during a parallel poll. During normal operation, the drivers operate in three-state mode for the fastest data transfers. See "IEEE 488 Transceivers (75160/75162)".

Serial Poll Register - Talker/Listener

(Base + 5h, Write)

The serial poll facility of the IEEE 488 is the easiest and most useful polling method used on the IEEE 488. The main distinction between serial polling and parallel polling is that in serial polling each talker/listener can interrupt the controller at any time via the Service Request (SRQ) line. When parallel polling has been implemented, the controller must periodically poll or interrogate the bus to check device status.

7	6	5	4	3	2	1	0
DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1
S8	rsv1	S6	S5	S4	S3	S2	S1

NAT9914BPD Serial Poll Register

When an SRQ is generated by a device using serial poll, the controller sends the Serial Poll Enable (SPE) message to the IEEE 488. Each device participating in the serial poll, regardless of whether or not it generated a service request, goes into the Serial Poll Mode state (SPMS) where each device must get ready to participate in the serial poll.

The controller then sequentially or serially runs down a device address list, addresses a device to talk, and then listens to or reads the device response called a *Serial Poll Response Byte*. If the polled device truly generated a service request (remember that more than one device could have requested service), the device must assert, as a minimum, bit 7 of its serial poll response byte. If bit 7 is not asserted, the controller knows the device did not request service. The controller keeps polling until all the devices in the device list have been polled.

The remaining seven bits of the serial poll response byte may contain user-defined information such as the type of service requested or some other machine status. This makes the serial poll mechanism the most popular of the IEEE 488 polling techniques.

The Serial Poll register (SPOLR) is used only when the IEEE 488 interface is a talker/listener. The interface must store its serial poll response byte in this register. When the IEEE 488 controller sends an SPE message to the bus followed by the board's talk address, the contents of the Serial Poll register are placed onto the IEEE 488 data bus. The IEEE 488 interface continues to assert the response byte until the controller re-addresses another device to talk or sends the Serial Poll Disable (SPD) message. The controller must read the serial poll response byte only once and then continue the serial poll.

Generating a Service Request

Method 1:

The easiest and most common method by which the IEEE 488 interface can generate a service request is with the auxiliary command called *Request Service Two (RSV2)* that is written to the Auxiliary Command register. Refer to the "Auxiliary Commands" topic.

This method should be used whenever possible. When the service request has been generated, the controller will eventually perform a serial poll. The suggested protocol follows:

1. The interface board requests service (asserts SRQ) via the RSV2 command.

2. The controller sends a Serial Poll Enable (SPE) message.

3. The controller addresses the interface to talk.

4. The controller de-asserts the ATN line, and the IEEE 488 interface serial poll response byte is automatically placed on the IEEE 488 data bus. The SRQ line is automatically cleared after being read.

5. The controller reads the response byte and the board generates a Serial Poll Active state (SPAS) interrupt, if enabled.

6. The controller reasserts ATN and again takes control of the IEEE 488. A second SPAS interrupt is generated, if enabled.

7. The controller continues polling the remaining devices on the IEEE 488. The serial poll terminates by way of the Serial Poll Disable (SPD) message sent by the controller after the controller polls the last device.

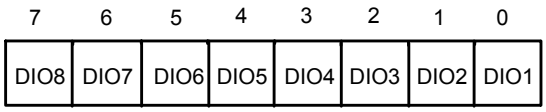
Method 2:

The second way to request service is to write a 1 to the RSV1 bit in the Serial Poll register. The same protocol is used as with RSV2 except that in order for the interface to generate another service request, you must first clear the RSV1 bit by writing a 0 to it. The RSV1 bit is then ready to be set again.

When the IEEE 488 interface has not requested service but is serial polled by the controller as a result of another device having requested service, the response byte is transferred to the IEEE 488 data bus as in the two cases above. The SPAS bit in the Interrupt Status register 0 (INT0) is never set and thus never generates an interrupt, if enabled. Also, bit 7 of the serial poll response byte is not asserted.

Data In Register

Controller, Talker/Listener (Base + 7h, Read)



NAT9914BPD Data In Register

The IEEE 488 interface reads all data from the IEEE 488 via the Data In register (DIN). The IEEE 488 hardware on the board is designed so that you do not lose data before the CPU has time to read the Data In register. IEEE 488 hardware suppresses the three-wire handshake either automatically or under software control, allowing an infinite length of time for the CPU to read the incoming data.

The Data In register accepts a byte of data from the IEEE 488 only if the previous data hold-offs (see "Auxiliary Commands") have been removed by the processor. Data can be read from the IEEE 488 only if the IEEE 488 interface has been addressed to listen, as when the board is a talker/listener, or if the board puts itself in a talk-only mode, as when it acts as the IEEE 488 controller.

The following suggested protocol can be used when the board is an IEEE 488 listener:

1. When the IEEE 488 controller addresses the IEEE 488 interface to listen, the My Address (MA) and My Address Change (MAC) interrupts occur, if enabled (see the "Interrupt Mask/Status Registers" discussion).The board is put in the Listener Primary Addressed state (LPAS) and Listener Addressed state (LADS).

2. The controller removes control by de-asserting ATN.

3. The active talker, which can be the controller or any other talking device, sends a valid data byte. The Byte In (BI) interrupt is generated, if enabled. The CPU must then read the byte from the Data In register.

4. Step 3 is repeated for each data byte sent by the active talker.

5. After the last data byte is sent by the talker and subsequently read from the Data In register, the controller "unaddresses" the board from listening by a Universal Unlisten (UNL) message. A MAC interrupt is generated, if enabled.

Making the IEEE 488 interface a listening controller is somewhat more difficult. The board must first be initialized as a controller. The following protocol is suggested:

1. Generate a chip reset and clear reset via the software reset (SWRST) auxiliary command.

2. Force the board to take control of the IEEE 488 and to send Interface Clear (IFC) to the IEEE 488 by issuing Send Interface Clear (SIC). The board then becomes the system controller.

3. Put the IEEE 488 interface into talk-only mode by issuing the talk only (TON) auxiliary command. This completes the board controller initialization. Talk-only mode can be considered the default controller mode.

4. Put the board into the listen-only mode by issuing the listen-only (LON) auxiliary command. An IEEE 488 controller should always default to the talk-only mode so that it can talk or send IEEE 488 messages.

5. Force the board to release control and de-assert ATN via the Go-To-Standby (GTS) auxiliary command.

6. The IEEE 488 interface then listens to the IEEE 488 data bus. The CPU reads the Data In register as in steps 3 and 4 above, with each byte preceded by a BI interrupt, if enabled.

7. After the last byte is read by the IEEE 488 interface, the board retakes control of the IEEE 488 by issuing the Take Control Synchronously (TCS) auxiliary command. The ATN line is reasserted.

8. Put the board back into the talk-only mode.

Note that the board has separate Data In and Data Out registers, which means that IEEE 488 data can be read and written without destroying the contents of the opposite register.

You can select several data hold-off modes via the auxiliary commands discussed in detail in the "Auxiliary Command Register" topic. The main function of data hold-off is to hold the handshake on the IEEE 488 long enough for the CPU to examine the data byte being listened to.

The data may be just data, but is more often an unrecognized command such as a Parallel Poll Enable or a secondary address. Messages or commands are different from data, for the controller is asserting ATN and the IEEE 488 hardware normally accepts the message without waiting for the CPU to read the Data In register. A held-off data byte or message is unheld or released by one of the release hold-off auxiliary commands discussed later in this chapter (see "DACR", "RHDF", "HDFA", "HDFE").

Data Out Register

Controller, Talker/Listener (Base + 7h, Write)

The IEEE 488 interface uses the Data Out register (DOUT) to send or output data to the IEEE 488 data bus. When ATN is asserted on the IEEE 488, the data becomes a command or message. Only the controller currently in charge of the bus can send commands. Every data output to the Data Out register initiates a handshake.

The Byte Out (BO) interrupt in the Interrupt Status register tells the CPU the previous byte sent by the board is accepted by all other devices on the IEEE 488; that is, the handshake is complete. When the current active controller first addresses the board to talk and ATN is not asserted, the BO bit goes high and a BO interrupt is generated, if enabled. This action tells the CPU that it is acceptable to write out to the Data Out register. The BO bit is not set again until the current byte is accepted by all IEEE 488 devices.

You must make provisions for terminating data transfers because the last byte written to the Data Out register that is accepted by the IEEE 488 sets the BO bit. To prevent the CPU from blindly trying to write another byte to the Data Out register, some convention must be invented to terminate data strings. The current talker must know either the

number of bytes it must send and/or the last character (end-of-string character) sent. The active listener(s) should also know this. The End-Or-Identify line on the IEEE 488 also serves the purpose of indicating the end of a data transfer. The talker should assert the EOI line with the last byte to tell all listeners to expect no more data. See the "Auxiliary Command Register" discussion for instructions on asserting EOI.

When you use the board as an IEEE 488 controller, you should observe the following protocol to initialize the Data Out register for sending data.

1. Execute a Software Reset and Clear Reset via the SWRST auxiliary command.

2. Assert Interface Clear (IFC) and take control (assert ATN) via the Send Interface Clear (SIC) auxiliary command. Do not forget to clear the command.

3. The Byte Out (BO) bit is then set, indicating a receptive IEEE 488 data bus. After reading the BO bit in the Interrupt Status register 0, it clears the BO bit, but the Data Out register is still ready.

4. Put the board in the talk-only mode via the Talk-Only (TON) auxiliary command.

5. A byte may be written to the Data Out register providing the BO bit was set as a result of the SIC auxiliary command and nothing else was written to the Data Out register prior to that point. You should implement a software polling loop to wait for BO to be set, keeping in mind that once the BO bit is read, it is cleared by the read operation.

6. When BO is set and a byte is written to the Data Out register, the byte is sent to the IEEE 488 as a command and not as data, because ATN was asserted by the SIC auxiliary command. Be sure this can be interpreted by the IEEE 488 devices.

7. To send data to a device on the IEEE 488, the IEEE 488 interface must first address the correct device(s) to listen to the data. After the devices are addressed to listen, the board must remove the ATN line by issuing the Go-To-Standby (GTS) auxiliary command.

8. When BO is set**,** a data byte may be written to the bus. All devices complete the handshake, causing BO to be set again, but only the active listeners actually read the data.

9. The controller takes control again by asserting ATN via the Take Control Asynchronously (TCA) auxiliary command.

Interrupt Mask/Status Registers

Controller, Talker/Listener

(Int Mask 0/Int Status 0: Base +0h, Read/Write)

(Int Mask 1/Ins Status 1: Base +1h, Read/Write)

The Interrupt Mask and Status registers are the registers most used when interfacing to the IEEE 488, whether or not interrupts are used. Study these Interrupt registers at length in order to understand the operation of the IEEE 488 interface. The Interrupt registers are usually the first and last registers read when using the IEEE 488 interface and usually point to the next operation, if any, to perform.

The Interrupt Status registers operate independently of the Mask register. No interrupt is generated if the corresponding mask bit is set to 0; that is, masked off. The Status registers are double buffered so that any event causing a Status register to change during a CPU read cycle is stored and sets the corresponding bit at the end of the read cycle. The previously set bits are cleared at the end of the read. The Interrupt Status registers are also cleared by either a hardware reset or a software reset (SWRST).

Except for INT0 and INT1, each bit is set when the corresponding event occurs. Once set, the corresponding register must first be read and then the interrupt condition be false and true again before that status bit is set again. However, INT0 and INT1 are set only when at least one event occurs in status register 0 or 1 and when the corresponding bit in the Interrupt Mask register is also set; that is, masked on so that interrupts are enabled.

Note that the INT0 and INT1 bits are cleared only when the Interrupt register causing the interrupt is read. Note also that an interrupt is enabled, that is, masked on when the mask bit is set to a 1. Both Mask registers are cleared by a hardware reset, but not by a software reset.

Interrupt Mask/Status Register 0

7	6	5	4	3	2	1	0	
X	X	BI	BO	END	SPAS	RLC	MAC	Mask
INT0	INT1	BI	BO	END	SPAS	RLC	MAC	Status

NAT9914BPD Interrupt 0 Register

The following topics discuss the NAT9914BPD INT0 register bits.

- INT0/INT1 (Interrupt 0 / Interrupt 1)
- BI (Byte In)
- BO (Byte Out)
- END (End)
- SPAS (Serial Poll Active State)
- RLC (Remote-To-Local Change)
- MAC (My Address Change)

INT0/INT1

The Interrupt 0 (INT0) and Interrupt 1 (INT1) bits indicate that a condition in the Interrupt Status register 0 or Interrupt Status register 1, respectively, caused an interrupt. Obviously, at least one of the conditions must have been enabled to generate an interrupt by having set a corresponding mask bit at an earlier time.

BI

The Byte In (BI) bit is set when a data byte or a command is received by the IEEE 488 Data In register. The primary function of the BI bit is to tell the CPU to promptly read the Data In register so that another byte may be input. The BI bit is reset when the CPU reads the INT0 register. The BI bit is not set when the board is in the shadow handshake mode. See "Auxiliary Commands".

BO

The Byte Out (BO) bit is set when the Data Out register is ready to be loaded with a data byte or IEEE 488 command. It basically tells the CPU that all the IEEE 488 devices have accepted the last byte and/or each device is ready for another byte or command. This bit is also reset by reading the INT0 register.

END

The End (END) bit indicates that the byte just received in the Data In register is the last byte, indicated by the End-Or-Identify (EOI) line on the IEEE 488 being asserted by the active talker. The talker could have been the controller.

SPAS

The Serial Poll Active state (SPAS) bit is set twice during serial polling. It is read only when the IEEE 488 interface is a talker/listener. If the corresponding mask bit is set, it also generates two interrupts with each set condition. The SPAS bit is first set when the controller reads the serial poll response byte from the IEEE 488 interface.

When the controller reasserts the ATN line after reading the board's response byte (usually to poll another device or to disable serial poll), the second setting of SPAS occurs. If the board did not request service but is serial polled as a result of another device having requested service, the SPAS bits are not set and no corresponding interrupt is generated. Remember that the Serial Poll register contents will still be read by the controller.

RLC

The Remote-To-Local Change (RLC) bit is set whenever the controller-in-charge sends a Remote or Local message (or REN) to the IEEE 488 interface. The RLC is used only

when the board is a talker/listener. The RLC message implies that an instrument may respond to its front panel controls if the front panel was previously disabled by the IEEE 488 controller. RLC does not mean anything inherently to the IEEE 488 interface because the interface is a microcomputer, which always has access to the IEEE 488 hardware so long as it is running. RLC is relevant to the IEEE 488 interface only if used to interface to a human interface, such as a keyboard or control panel. The IEEE 488 interface could then interpret the RLC message and subsequent RLC bit setting to "return-to-local" control of the operator by scanning and responding to the control panel again. The power-up configuration for an IEEE 488 instrument is normally a local control state that may be removed or superseded by the IEEE 488 controller.

MAC

The My Address Change (MAC) bit is read only when the IEEE 488 interface is a talker/listener. The MAC bit is set whenever the IEEE 488 interface address status has been changed by the IEEE 488 controller. The MAC should be the first bit examined when any change in talker/listener addressing is suspected by the board. As an example, when the controller addresses the IEEE 488 interface to listen, the MAC bit is set. The MAC bit is reset by reading the INT0 register. The board should then go into some listen routine designed by the user. When the controller addresses the board to listen (that is, the board is no longer an active listener), MAC is set again.

Interrupt Mask/Status Register 1

7	6	5	4	3	2	1	0	
GET	ERR	UNC	APT	DCAS	MA	SRQ	IFC	Mask
GET	ERR	UNC	APT	DCAS	MA	SRQ	IFC	Status

NAT9914BPD Interrupt 1 Register

The following topics discuss the NAT9914BPD INT1 register bits:

- GET (Group Execute Trigger)
- ERR (Error)
- UNC (Unrecognized Command Group)
- APT (Address Pass Through)
- DCAS (Device Clear Active State)
- MA (My Address)
- IFC (Interface Clear)

GET

The Group Execute Trigger (GET) bit, used only when the IEEE 488 interface is a talker/listener, indicates when the IEEE 488 controller sends the Group Execute Trigger message to the IEEE 488. The controller must have previously addressed the board to listen. The GET message can be used as an IEEE 488 system synchronization signal in which multiple listeners can respond to a command at the same instant. This is useful in a system in which the controller needs to start or stop a group of real-time clocks on the IEEE 488.

ERR

The Error (ERR) bit is used to detect errors in the handshake sequence. When the IEEE 488 interface is going to send a byte to the IEEE 488 and the Not Ready for Data (NRFD) and Not Data Accepted (NDAC) lines are both sense high, indicating an invalid source handshake, the ERR bit is set and the byte in the Data Out register is not sent. This is not a typical condition in IEEE 488 systems, and thus the ERR bit usually indicates that no devices in the system are addressed to listen.

UNC

The Unrecognized Command Group (UNC) bit tells the board that an IEEE 488 command sent by the controller is not known by the NAT9914BPD hardware. This means that software must handle the command's interpretation. The UNC bit is used only when the IEEE 488 interface is a talker/listener. The board could be an inactive controller currently acting as a talker/listener. The following three bus messages set the UNC bit:

1. Take Control (TCT) if the board is addressed to talk

2. My Secondary Address if the Pass Through Next Secondary auxiliary command was issued previously

3. Unrecognized Universal Command Groups (UUCG) or Unrecognized Addressed Command Group (UACG) (See the IEEE 488 standard, Section 2.13)

APT

The IEEE 488 interface uses the Address Pass Through (APT) bit only when it is a talker/listener. The APT bit tells the board that an extended or secondary address was sent by the IEEE 488 controller. To enable the board for secondary addressing, the APT bit in the Interrupt Mask register 1 must be set.

When the controller sends any secondary address, an APT interrupt is generated and an automatic Accepted Data state (ACDS) holdoff is initialized. No further IEEE 488 bus activity will take place until the CPU reads the secondary address from the Command Pass-Through register and issues one of two auxiliary commands.

If the CPU recognizes the secondary address as a valid secondary address, the data holdoff is released by sending the Data Accepted Release (DACR) auxiliary command with the most significant bit set high. This action completes the handshake, allowing IEEE 488 activity to continue and forcing the IEEE 488 interface to enter the completed address state.

If the CPU does not recognize the secondary address as being valid, a DACR auxiliary command is issued with the most significant bit set low. This forces the IEEE 488 interface to complete the handshake but not to enter the completed address state.

DCAS

The Device Clear Active state (DCAS) bit is used only when the IEEE 488 interface is a talker/listener. The DCAS tells the board that the IEEE 488 controller sent the Device Clear (DCL) message. DCL is sent by the controller to clear all or a subset of talker/listener on the bus individually selected by prior listen addressing.

The effect of DCL on a device is a function of system design. It is not meant to be a reset but could indirectly be used in that manner. You may implement the DCL message to force the listening device to enter the Power On (PON) state, thus forcing all states into an idle condition.

You can also define the DCL function to force listening devices into any "non-obtrusive" state. As an example, if the board is used as an IEEE 488 data logging system, DCL might be implemented to reset any internal software counters or timers, but not to clear data.

MA

The My Address (MA) bit is used only when the IEEE 488 interface is a talker/listener. The MA bit tells the board that it has been addressed by the controller to talk or listen. The MA bit is not set after the Serial Poll Enable (SPE) message has been sent by the controller; that is, during a serial poll sequence. The My Address Change (MAC) bit, however, is affected. See the "MAC" bit discussion.

The Service Request (SRQ) bit is used by the IEEE 488 interface when it is a controller only. The SRQ bit is set whenever a device on the IEEE 488 requests service from the controller by asserting the SRQ line.

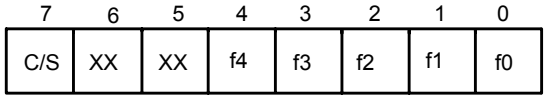
IFC

The Interface Clear (IFC) bit is used only when the board is a talker/listener. The IFC bit tells the board when the system controller asserts the IFC line on the IEEE 488. IFC is normally pulsed only during power-up and/or reset. When the board detects an IFC pulse, the software should completely reinitialize the system. All IEEE 488 functions should be in idle state.

Note: When GET, UNC, APT, DCAS, and MA bits have been enabled to generate an interrupt to the CPU and one of these states occurs, thus generating an interrupt, an Accept Data state (ACDS) holdoff is automatically effected. All IEEE 488 activity is then temporarily suspended and the handshake is suppressed. The on-board CPU must interpret the cause of the interrupt, take appropriate action depending on the system, then complete the handshake by issuing the Release Data Holdoff (DACR) auxiliary command. This necessary feature gives the on-board processor time to respond to interrupts without losing IEEE 488 information.

Auxiliary Command Register

Controller, Talker/Listener (Base + 3h, Write)



NAT9914BPD Auxiliary Command Register

The Auxiliary Command register (AUXCD) provides many of the special features of the IEEE 488 interface. An auxiliary command is issued by writing the command byte to the Auxiliary Command register. Refer to the "Auxiliary Commands" topic.

Auxiliary Commands

A number of the auxiliary commands are of the Clear/Set (C/S) type. If a command is loaded with the C/S bit set to 1, the function is selected and remains selected until the code is loaded with the C/S bit set to 0. The Talk Only (TON) and Listen Only (LON) commands operate in this manner.

Other commands, such as the Force EOI (FEOI) and Release RFD Holdoff (RHDF) commands, have a pulsed mode of operation in which the C/S bit is not applicable (NA), as shown in the "NAT9914BPD Auxiliary Commands" table.

The Force Group Execute Trigger (FGET) and Return To Local (RTL) commands can operate in either CLEAR/SET or pulsed modes. If the FGET command is loaded with the C/S bit set to 0, a pulse appears at the trigger output of the NAT9914BPD. If the command is loaded with the C/S bit set to 1, the trigger output goes high until the command is issued again with the C/S bit set to 0.

If the Return To Local (RTL) command is issued with the C/S bit set to 0, the REM status bit in the Address Status register is reset. REM can be set again at any time by a REN command from the IEEE 488 controller-in-charge.

If the RTL command is issued with the C/S bit set to 1, the REM bit is cleared and cannot be set until the RTL command is issued again with the C/S bit set to 0. The RTL

command has no effect if the Local Lockout (LLO) mode has been selected by the IEEE 488 controller.

NAT9914BPD Auxiliary Commands

C/S	F4	F3	F2	F1	F0	Mnemonic	Function
0/1	0	0	0	0	0	SWRST	Software Reset
0/1	0	0	0	0	1	DACR	Release ACDS Holdoff
NA	0	0	0	1	0	RHDF	Release RFD Holdoff
0/1	0	0	0	1	1	HDFA	Holdoff On All Data
0/1	0	0	1	0	0	HDFE	Holdoff On EOI Only
NA	0	0	1	0	1	NBAF	New Byte Available False
0/1	0	0	1	1	0	FGET	Force Group Execute Trigger
0/1	0	0	1	1	1	RTL	Return To Local
NA	0	1	0	0	0	FEOI	Send EOI with Next Bite
0/1	0	1	0	0	1	LON	Listen Only
0/1	0	1	0	1	0	TON	Talk Only
NA	0	1	0	1	1	GTS	Go To Standby
NA	0	1	1	0	0	TCA	Take Control Asynchronously
NA	0	1	1	0	1	TCS	Take Control Synchronously
0/1	0	1	1	1	0	RPP	Request Parallel Poll
0/1	0	1	1	1	1	SIC	Send Interface Clear
0/1	1	0	0	0	0	SRE	Send Remote Enable
NA	1	0	0	0	1	RQC	Request Control
NA	1	0	0	1	0	RLC	Release Control
0/1	1	0	0	1	1	DAI	Disable All Interrupts
NA	1	0	1	0	0	PTS	Pass Thru Next Secondary
0/1	1	0	1	0	1	STDL	Set T1 Delay
0/1	1	0	1	1	0	SHDW	Shadow Handshake
0/1	1	0	1	1	1	VSTDL	Set Very Fast T1 Delay
0/1	1	1	0	0	0	RSV2	Second Service Request

SWRST (Software Reset) 0/1XX00000

The Software Reset command is issued when the IEEE 488 interface is a talker/listener or a controller. In both cases, the SWRST must be issued at power-up and/or system reset to begin initialization of the IEEE 488 hardware. When the board is a talker/listener, the SWRST should be executed when the IEEE 488 controller asserts the Interface Clear (IFC) line.

Issuing the SWRST command with the C/S bit set to 1 causes all input to the IEEE 488 hardware to be ignored. The Serial Poll register and Parallel Poll registers 0 and 1 are cleared. Also, when the SWRST is set, the IEEE 488 hardware is forced into the following states:

SIDS - Source Idle state

CIDS - Controller Idle state

AIDS - Acceptor Idle state

LOCS - Local state

TIDS - Talker Idle state

NPRS - Negative Poll Response state

TPIS - Talker Primary Idle state

PPIS - Parallel Poll Idle state

LIDS - Listener Idle state

SPIS - Serial Poll Idle state

LPIS - Listener Primary Idle state

When a power-on or push-button reset is generated on the IEEE 488 interface, the SWRST is automatically generated internal to the IEEE 488 hardware, forcing the SWRST command into its set condition. Whenever the SWRST command is set, either by software or automatically, it must be cleared by reissuing the SWRST command with the C/S bit set to 0.

DACR (Release ACDS Holdoff) 0/1XX00001

The Release Accepted Data state Holdoff command (DACR) is used when the IEEE 488 interface is a talker/listener. The DACR command is issued to complete a handshake that was put into Data Accepted Holdoff as the result of receiving an unrecognized command, secondary address, device trigger, or device clear message. When the board is receiving data from the IEEE 488 and an ACDS holdoff occurs, the

talking device holds the data in its valid state to give the on-board CPU unlimited time to read and process the valid data. When the CPU has decided what to do with the data, it completes the handshake via the DACR command.

The DACR command is used in two modes: secondary addressing and primary addressing only. When an Address Pass Through (APT) interrupt is enabled, an ACDS holdoff will occur whenever a secondary address is received. If the CPU recognizes the secondary address as valid, the CPU must issue the DACR command with the C/S bit set to 1. If the secondary address is invalid, the DACR must be issued with the C/S bit set to 0. In any case, the handshake is complete; however, the board remains unaddressed for invalid secondary addresses.

When secondary addressing is not being used, ACDS holdoffs due to unrecognized commands are released by issuing the DACR command with the C/S bit set to 0.

RHDF (Release RFD Holdoff) naXX00010

The Release Ready For Data Holdoff command is used by the board when it is both a talker/listener and a controller. The RHDF command is issued to release any data holdoff caused by the auxiliary command HDFA or HDFE. The C/S bit is not applicable.

There is an important distinction between RFD and ACDS holdoff. The ACDS holdoff is used to give the on-board CPU time to read an IEEE 488 command. The byte remains valid so long as the CPU needs to process the data and issue the DACR command completing the handshake. Bus activity is terminated.

HDFA (Holdoff On All Data) 0/1XX00011

The HDFA command is used by the IEEE 488 interface when it is both a talker/listener and a controller. When the HDFA command is issued with the C/S bit set to 1, a Ready For Data Holdoff (RFD) is generated with every IEEE 488 data byte. The handshake must be completed by issuing the RHDF command. IEEE 488 commands, when ATN is asserted, are not affected by this command. The HDFA command is unasserted by issuing the command with the C/S bit set to 0.

HDFE (Holdoff On EOI Only) 0/1XX00100

The HDFE command is used by the board when it is a talker/listener or a controller. When the HDFE command is issued with the C/S bit set to 1, a Ready For Data Holdoff (RFD) is generated for the last IEEE 488 data byte indicated by the End-Or-Identify (EOI) signal. This holdoff gives the CPU time to respond to a string of data terminated by EOI before allowing another string to be received. The holdoff must be released by issuing the RHDF command, thus completing the handshake. The HDFE mode is de-asserted by issuing the HDFE command with the C/S bit set to 0.

NBAF (Set New Byte Available False) naXX00101

The NBAF command is used by the board only in the talker modes. The C/S bit is not applicable. The main function of NBAF is to back out of an IEEE 488 data send sequence by canceling the data previously written to the Data Out register. The protocol is as follows:

1. Assume the IEEE 488 interface has been an active talker, sending a string of data bytes to the addressed listeners.

2. Assume that in midstream of this string of data a particular byte sent out by the board is accepted by all devices; that is, the handshake is completed. If for some system-dependent reason (such as an error condition) the last byte sent demands the immediate attention of the IEEE 488 controller, the controller takes immediate control of the IEEE 488 (asserts ATN).

3. The IEEE 488 interface would still see the Byte Out (BO) bit set in the Interrupt Status register 0, indicating the need to write out the next byte in the data string to the Data Out register. If this happens, the Data Out register cannot be written again until the last byte written (that is, the byte following the error) is accepted by the IEEE 488 listeners.

    a. Assuming that happened, when the controller releases ATN again, the last byte written to the Data Out register is automatically sent to the IEEE 488, requiring all listeners to accept the byte. If this is not desirable; that is, if as a result of the previous byte sent a different byte from the one currently in the Data Out register is needed, the board may change its mind by issuing an NBAF command.

    b. When the NBAF command is issued, the current byte is not removed or reset, but the validity of the byte is canceled; when ATN is released, Data Valid (DAV) will not be asserted until a new byte is written to the Data Out register. After NBAF is issued, the Byte Out is also set again, indicating the Data Out register is free to be written again.

This command should be used only when you find yourself in a deadlock situation. An IEEE 488 controller should not take control in the middle of a block transfer. Provision is normally made to terminate a block transfer such as an End-Of-String (EOS) character, byte counter, or End-Or-Identify (EOI).

FGET (Force Group Execute Trigger) 0/1XX00110

This is a general purpose command. The state of the trigger output from the NAT9914BPD is affected when this command is issued. If the C/S bit is set to 0, the line is pulsed high for approximately five clock cycles (1.0 µsecond at 4.77 MHz). If the C/S bit is set to 1, the trigger line goes high until FGET is sent with the C/S bit set to 0. No

interrupts or handshakes are initiated and only the NAT9914BPD is affected. The trigger output pin is not used on the board.

RTL (Return To Local) 0/1XX00111

This is also a general purpose command. When the RTL command is issued, provided the local lockout (LLO) has not been previously enabled, the remote/local status bit is reset and an interrupt is generated (if enabled) to inform the on-board CPU that it should respond to front panel controls if applicable. If the C/S bit is set to 1, the RTL command must be cleared by issuing the RTL command with the C/S bit set to 0 before the device is able to return to remote control. If the C/S bit is set to 0, the device may return to remote without first clearing RTL.

FEOI (Force End-Or-Identify) naXX01000

This command is used by the board when it is a talker or talking controller. The command causes the End-Or-Identify message to be sent with the next data byte. The EOI line is then reset.

LON (Listen Only) 0/1XX01001

The Listen Only command is used by the IEEE 488 interface to set itself up as a listener. However, this command should be used only if the board is placed in a system in which there is no controller or in which the board is the controller.

After the LON command is issued with the C/S bit set to 1, the board becomes a self-addressed listener, indicated by the Listen Addressed state (LADS) bit being set in the Address Status register. The listener feature must not be disabled; that is, the Disable Listener (DAL) bit in the Address register must not be set.

The on-board CPU may read data from the IEEE 488 via the Data In register (listen activity) whenever the Byte In (BI) is set in the Interrupt Status register 0. The LON command is reversed by issuing the command with the C/S bit set to 0 or by issuing the TON auxiliary command.

TON (Talk Only) 0/1XX01001

The Talk Only (TON) command is analogous to the LON command. It is used by the IEEE 488 interface to address itself to talk although no real addressing takes place. Use the TON command only when the board is the controller or if there is no controller in the system. An example of a no-controller system would be a reporting (talking) voltmeter and a printer set up as a listener.

To enable the talk only mode, issue the TON command with the C/S bit set to 1. The TON should be removed by issuing the command with the C/S bit set to 0. After the TON mode has been enabled, the Byte Out (BO) bit in the Interrupt Status register 0 is

asserted to let the on-board CPU know that it is okay to write to the Data Out register. When all the listening devices have completed the handshake, the BO bit is set again. The Disable Talker (DAT) bit in the Address register must not be set; that is, do not disable the talk ability of the IEEE 488 hardware.

Note: The TON and LON commands are designed to be used with systems without a controller. However, when the board is the IEEE 488 controller, the TON and LON functions are used to set the board up to talk and listen, respectively. You should be aware that if the board as a controller is sending IEEE 488 messages such as Untalk (UNT), Unlisten (UNL), or Other Talk Address (OTA), the talk or listen status is subject to those messages. For example, UNT resets the TON feature, taking the board out of talk activity. Note also that the TON and LON auxiliary commands are mutually exclusive to the IEEE 488 hardware; in other words, the most recently issued command will be the one in effect.

GTS (Go To Standby) naXX01011

This command instructs the IEEE 488 interface to de-assert the ATN line, thus going to standby. This is an IEEE 488 controller function only.

TCA (Take Control Asynchronously) naXX01100

This command instructs the board to reassert ATN as controller-in-charge. The command is executed immediately; data corruption or loss may occur if a talker/listener is in the process of transferring a data byte. If a controller has been talking, use TCA after the last BO interrupt to reassert the ATN line without corrupting data. A BO interrupt is generated when the board has entered the controller active state.

TCS (Take Control Synchronously) naXX01101

This command is used by the controller-in-charge to set the ATN line true and to gain control of the IEEE 488. If the controller is not a true listener, the shadow handshake command must be used to monitor the handshake lines. The IEEE 488 interface is forced to synchronize with the talker/listeners, sending ATN true only at the end of a byte transfer so that no data will be lost or corrupted. A BO interrupt is generated when the NAT9914BPD has entered the controller active state.

RPP (Request Parallel Poll) 0/1XX01110

This command is used by the controller-in-charge to send the parallel poll command over the IEEE 488 (the board must be in the Controller Active state so that the ATN line is asserted). The poll is completed by reading the Command Pass-Through register to obtain the parallel poll response bits, then sending RPP with the C/S bit set to 0. Note that the IEEE 488 standard requires a minimum of 2 µseconds before the parallel response is output to the bus.

SIC (Send Interface Clear) 0/1XX01222

This command is used when the IEEE 488 interface is a system controller only. The Interface Clear (IFC) line is set true when this command is sent with the C/S bit set to 1. This must be sent by the system controller and reset to C/S equal to 0 only after the IEEE 488 standard minimum time for IFC (100 µseconds) has elapsed. A longer time of about 5 milliseconds is suggested. The system controller is put into the controller active state and a BO interrupt is generated, if enabled.

SRE (Send Remote Enable) 0/1XX10000

This command instructs the IEEE 488 interface to set the REN line true, thus sending the Remote Enable message over the IEEE 488. REN is set false by sending SRE with the C/S bit set to 0, causing the IEEE 488 devices to return to local mode. This command is used only when the board is the system controller.

RQC (Request Control) naXX10001

Multiple IEEE 488 controllers are allowed on the bus, although only one controller can be actively in control at one time. Also, only one can be the ultimate system controller. The function of the system controller is to take control at power-up time during system conflicts. Only the system controller is allowed to assert Interface Clear (IFC) and Remote Enable (REN).

A controller may pass control to another controller via the IEEE 488 message Take Control (TCT). The current controller-in-charge passes control to the board by sending the board talk address followed by the TCT message. The board recognizes the TCT by receiving an Unrecognized Command Group (UNC) interrupt, if enabled, and reading the TCT message from the Command Pass-Through register. The board responds to TCT by issuing the RQC command. The IEEE 488 hardware waits for the current controller-in-charge to release ATN and then asserts ATN itself, going into the Controller Active state. A BO interrupt is generated, if enabled.

RLC (Release Control) naXX10010

The IEEE 488 interface may pass control (or return control) to another controller by the same protocol as that used for RQC (see "RQC"). In this case, TCT is sent by the board following the new controller talk address. After the handshake is completed, the board issues the RLC auxiliary command which releases the ATN line, thus relinquishing control. The new controller must then take (retake) control.

Note: No standard protocol is available that enables a controller to regain control once it has passed control to another device. You must alert potential and current controllers that a transfer of control needs to take place. A serial poll protocol is a good way to do this. An inactive controller requests service of a current active controller via the SRQ line. When the inactive controller sends its serial poll response byte during the serial

poll, the response byte must contain a request for control. If your system contains more than two controllers, you must assign a priority scheme in the event that multiple controllers request control simultaneously.

DAI (Disable All Interrupts) 0/1XX10011

This command disables the IEEE 488 interrupt line. The interrupt registers and any selected holdoffs are not affected. This feature is useful in systems designed for polling operation as opposed to interrupt operation.

PTS (Pass Through Next Secondary) naXX10100

This command may be used to carry out a remote configuration of a parallel poll. The Parallel Poll Configure command (PPC) is passed through the board as an unrecognized command and must be identified by the CPU. When the PTS command is issued, the next byte received by the board is passed through via the Command Pass-Through register. This should be the Parallel Poll Enable (PPE) message which is read by the microprocessor.

STDL (Set T1 Delay) 0/1XX10101

The IEEE 488 interface uses this command to set the data to Data Valid delay time T1. The T1 delay time is set to six clock cycles if this command is sent with the C/S bit set to 1. The T1 delay time is 10 clock cycles following a power-on RESET or if a STDL is sent with the C/S bit set to 0.

Three-state driver mode is required when using the short T1 time to reduce the settling time of data on the DIO lines. See "IEEE 488 Transceivers (75160/75162)".

SHDW (Shadow Handshake) 0/1XX1011000

This auxiliary command enables the controller to carry out the listener handshake without participating in a data transfer. The Data Accepted line (DAC) is pulled true a maximum of three clock cycles after Data Valid (DAV) is received. Not Ready For Data (NRFD) is allowed to go false as soon as DAV is removed. It must be used in conjunction with the LON mode. The END interrupt can also indicate when to generate an ACDS holdoff. This permits the controller to sense the end-of-string transfer across the IEEE 488.

The shadow handshake function allows the TCS command to be synchronized with the Acceptor Not Ready state (ANRS) so that ATN can be reasserted without causing the loss or corruption of a data byte. The END interrupt can also be received to cause an RFD holdoff to be generated.

VSTDL (Set Very Fast T1 Delay) 0/1XX10111

The IEEE 488 specification allows the bus settling time T1 to be reduced to 400 ns on all bytes except the first byte after ATN is de-asserted. It should then be greater than 1100 ns. The IEEE 488 interface has a feature that reduces T1 to three clock cycles on all bytes but the first when ATN is de-asserted. When ATN is de-asserted or on the first byte, T1 is 2.1 µseconds with STDL not set or 1.26 µseconds with STDL set. This feature is programmable. The VSTDL is a Clear/Set (C/S) type command. If VSTDL is set, three-state drivers are required for shorter settling time for data.

RSV2 (Second Service Request) 0/1XX11000

This auxiliary command should be used by a device to request service from a IEEE 488 controller. Once set true and when a SPAS interrupt occurs (indicating that the serial poll response byte has been read), this bit is automatically reset by the NAT9914BPD logic. Most systems should request service with this command as opposed to RSV1 in the Serial Poll register. Refer to "Serial Poll Register - Talker/Listener" for more information.

USING THE NAT9914BPD AS A CONTROLLER

The following brief controller commands outline is based on previous discussions of auxiliary commands (AC), interrupt status, and data register. The outline does not represent a complete set of IEEE 488 functions, nor is it complete in every detail. However, the information should be useful as you develop your own code if you are not using Ziatech's optional software drivers.

Note: When outputting to the Data Out (DO) register, you must first wait for the Byte Out bit. When inputting from the Data In (DI) register, first wait for the Byte In (BI) bit.

NAT9914BPD Controller Commands

INITIALIZATION

```
Software Reset      ; Auxiliary Command (AC)
Assert IFC          ; Assert for 5 ms (AC)
TON                 ; Set talk only (AC)
Return
```

SEND DATA

MTA	; My talk address (DO)
UNL	; Universal unlisten (DO)
Listen Address	; Listener address (DO)
GTS	; Go to standby (AC)
Output Data	; (DO)
Repeat until done	
TCA	
Return	

RECEIVE DATA

Talk Address	; Talk address (DO)
UNL	; Universal unlisten (DO)
MLA	; My listen address (DO)
LON	; Listen only (AC)
GTS	; Go to standby (AC)
Input Data	; (DI)
Repeat until done	
TCS	; Take control (AC)
TON	; Talk only (AC)
Return	

SRQ STATUS

Input SRQ status

Return

SERIAL POLL

UNL	; Universal unlisten (DO)
MLA	; My listen address (DO)
SPE	; Serial poll enable (DO)
Talk address	; Talk address (DO)
LON	; Listen only (AC)
GTS	; Go to standby (AC)
Input Data	; (DI)
TON	; Talk only (AC)
TCS	; Take control (AC)
SPD	; Serial poll disable (DO)
Return	

USING THE NAT9914BPD AS A DEVICE

Use the registers listed below when programming the NAT9914BPD interface as a device (talker/listener). Contact Ziatech for more information about software to implement the ZT 1444A as a device.

Register Name:

Address Switch register

Address register

Data In register

Data Out register

Interrupt Mask/Status registers

Auxiliary Command register

7. IEEE 488 TRANSCEIVERS (75160/75162)

The 75160/75162 IEEE 488 transceiver chips ensure that all relevant bus driver/receiver specifications are met. These transceivers feature:

- 500 mV receiver hysteresis
- Bus-terminating resistors
- No loading with no power
- Meets IEEE 488-1978 standard

Two devices implement the 16 signal lines required by the interface system. The 75160A handles the 8-bit data bus and the 75162A handles the handshake lines and bus management signals.

The 75160A has a Pull-up Enable (PE) pin that controls output characteristics. When PE is high, three-state characteristics are exhibited. The state of this line is normally three-state except during parallel polls.

The 75162A octal bus transceiver determines the direction of REN and IFC via the system controller (SC) input. This input is connected to a DIP switch at location 1D, position 8, and is shipped from the factory enabled. In the enabled mode, the board acts as a system controller by sending Remote/Local and Interface Clear messages.

Note: Older 75160/75162 chips glitch the IEEE 488 lines when powered on or off.

8. OPTIONAL SECURITY KEY INTERFACE

This is an optional feature available on the ZT 1444A. Ziatech utilizes a Dallas Semiconductor DS1204 electronic key for securing software and machine operation. The electronic key stores 64 bits of user-definable identification code and a 64-bit security match code which protects 128 bits of read/write non-volatile memory.

The 64-bit identification code and the security match code are programmed via a unique program mode operation (see the "Programming Sequence" topic). Once programmed, the key is accessed with a special sequence (see "Reading and Writing to the Key" for details. Re-programming the chip clears all data in the non-volatile memory.

Reading and writing data/commands to the key is done in a serial format to a port location (Base + 0Ch) using D0, the least significant bit. See the DS 1204 I/O Port Descriptions table. Programming and read/write capability is controlled by writing to a different port (Base + 0Dh). Outputting a **1** to this port disables access to the port. Writing a **0** to this port enables programming or read/write.

DS 1204 I/O Port Descriptions

I/O Port Address Base+	I/O Read Register	I/O Write Register
000Ch	DS 1204 Data (D0)	DS 1204 Data (D0)
000Dh	-----	DS 1204 RESET (D0)

PROGRAMMING SEQUENCE

(Example: base address = 0210h)

1. Output D0 = 0 to port 21Dh (Base + 0Dh).
2. Output eight bits serially to define whether access is read or write. Programming requires writing to the port. Output to port 21Ch, using D0 to define bit.

   Programming and Write sequence: 1, 0, 1, 1, 1, 0, 0, 1

   Read sequence: 0, 1, 0, 0, 0, 1, 1, 0
3. Output eight bits serially to define a program cycle versus a normal read/write cycle.

   Program: 0, 1, 0, 0, 0, 0, 0, 0

   Normal Read/Write: 1, 0, 0, 0, 0, 0, 0, 0

4. Output eight bits of factory defined information in serial fashion.

   Program and Read/Write: 0, 0, 0, 0, 0, 1, 0, 1

5. Output 64 bits of identification code in serial fashion.

6. Output 64 bits of security match information in serial fashion.

7. Output D0 = 1 to port 021Dh to conclude programming.

Programming Summary

1. Output D0 = 0 to port 021Dh to enable the port.

2. Output D0 = X to port 021Ch serially with the following sequence for X:

   1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1

3. Output D0 = X to port 021Ch serially with 64 bits of user-defined identification code.

4. Output D0 = X to port 021Ch serially with 64 bits of security match code information.

5. Output D0 = 1 to port 021Dh to conclude programming.

READING AND WRITING TO THE KEY

After the port has been programmed, the 128 bits of non-volatile memory may be read or written. The sequence used here is similar to that used for programming. See "Write Sequence" and "Read Sequence" for details.

Write Sequence

(Example: base address = 0210h)

1. Output D0 = 0 to port 21Dh to enable the port.

2. Output eight bits serially to port 21Ch to define the write sequence. D0 is used to define the bit.

   Write sequence: 1, 0, 1, 1, 1, 0, 0, 1

3. Output eight bits serially to port 21Ch to define normal sequence (versus program).

   Write sequence: 1, 0, 0, 0, 0, 0, 0, 0

4. Output eight bits of factory defined information to port 21Ch.

   Write sequence: 0, 0, 0, 0, 0, 1, 0, 1

5. Read back 64 bits of banner identification code (programmed during program cycle) by inputting port 21Ch. D0 contains the information.

6. Output 64 bits of security match information to be used by the key for comparison. Output the correct 64 bits (as programmed during the last program cycle) to port 21Ch. If incorrect, access is denied.

7. Output 128 bits of data to be retained by non-volatile memory to port 21Ch.

8. Output D0 = 1 to port 21Dh to conclude the sequence.

Read Sequence

The read sequence is the same as the write sequence except for steps 2 and 7.

1. Output D0 = 0 to port 21Dh to enable the port.

2. Output eight bits serially to port 21Ch to define the read cycle.

   Read sequence: 0, 1, 0, 0, 0, 1, 1, 0

3. Output eight bits serially to port 21Ch to define normal sequence (versus program).

   Write sequence: 1, 0, 0, 0, 0, 0, 0, 0

4. Output 8 bits of factory defined information to port 21Ch.

   Write sequence: 0, 0, 0, 0, 0, 1, 0, 1

5. Read back 64 bits of banner identification code (programmed during program cycle) by inputting port 21Ch. D0 contains the information.

6. Output 64 bits of security match information to be used by the key for comparison. Output the correct 64 bits (as programmed during the last program cycle) to port 21Ch. If incorrect, access is denied.

7. Input the 128 bits of non volatile-memory from port 21Ch.

8. Output D0 = 1 to port 21Dh to conclude the sequence.

SECURITY METHODS

The general method used to ensure that the application software is not used in an unauthorized manner is to match each distribution media with a unique DS 1204 electronic key. This matched pair can be provided by uniquely programming the DS 1204 prior to shipping the media/key combination.

Several techniques can be used to ensure that the software protection mechanisms are not easily broken. Under the most simplistic case, the application software can perform the following steps to ensure that the program is authorized to function:

1. Upon program initialization, reset the keyring and read the 64-bit identification sequence from the electronic key.

2. Compare this identification sequence with the sequence value contained in the application program (unique per software application package) and abort if the values do not match.

3. If the identification sequences match, then transmit the 64-bit unlocking code to the electronic key. This unlocking code can be unique for each distribution media produced from the software distribution factory.

4. If the unlocking code does not match the value contained in the electronic key, the electronic key essentially becomes passive and refuses further access. It is important to note that the security match field within the key can be written, but not read. This provides a secure entry mechanism to the lowest level of the electronic key non-volatile memory.

5. If the unlocking code matches the value contained in the electronic key, access will be granted to read or write 128 bits of non-volatile memory in the electronic key. At this point, information unique to this particular invocation of the software package should be inserted/removed from the electronic key. This acts as a tertiary check for valid access to the application software package.

6. If all the above checks pass, the application program may continue to execute. Otherwise, it should inform you of unauthorized access and abort processing.

The above approach has several flaws:

- By placing the initialization sequences at the beginning of application program execution, it becomes possible to disassemble the application program binary to determine when the security check is performed. Once this is known, the security check can be disabled by patching the application program binary. The relative difficulty of this increases linearly with the number of security checks included in your program. Therefore, it would be wise to include several dozen security checks throughout the user application.

- If the security check routines that access the electronic key are called as subroutines, it could be possible to bypass these calls and thus render the security check ineffective. This approach can be circumvented by conducting an interactive dialogue with the electronic key 128-bit non-volatile memory, making the contents of the electronic key vital to the correct operation of the program. In addition, the read from or write to the key is dynamically changed by the program during execution.

This is more sophisticated and requires a greater integration of the electronic key into the application software, but the results are a security system which is intertwined into the application software and extremely difficult to break.

The method of using the key within application software is completely arbitrary. More sophisticated suggestions can be obtained from Ziatech if the above case proves too simplistic.

A purely random approach at breaking the security match code of 64 bits, using a 32 MHz processor and a 50% hit ratio, would take approximately 13,000 years to break, assuming a random distribution of security codes assigned. This rough estimate does not take into account practical limitations in hardware and software which would increase the time taken to crack the code

Device Capabilities

The following is a brief summary of the capabilities of the security device:

1. *Media copy resistance*. It is possible for the end user to make millions of copies of the original distribution media without being able to use more than one remastered distribution kit at a given time.

2. *No interference with normal backup procedures*. End users may safely back up their distribution media as many times as necessary to ensure the integrity of their data and application software.

3. *Media hardware independence*. The application software is no longer dependent on distribution media hardware dependencies for copy protection. This functionality is now present in utility routines which communicate with the electronic key.

4. *Software licensing mechanisms*. The application software may now be licensed for a set period of time rather than sold with a perpetual right-to-use license. This differentiation allows software to be used or evaluated for a set period of time while at the same time preventing widespread distribution of the application software.

Using the security device should provide a secure environment in which developers of expensive software packages can prevent unauthorized usage.

A. JUMPER CONFIGURATIONS

The ZT 1444A has several options that you can select by changing jumper configurations. These options are discussed in this section.

ZT 1444A JUMPERS

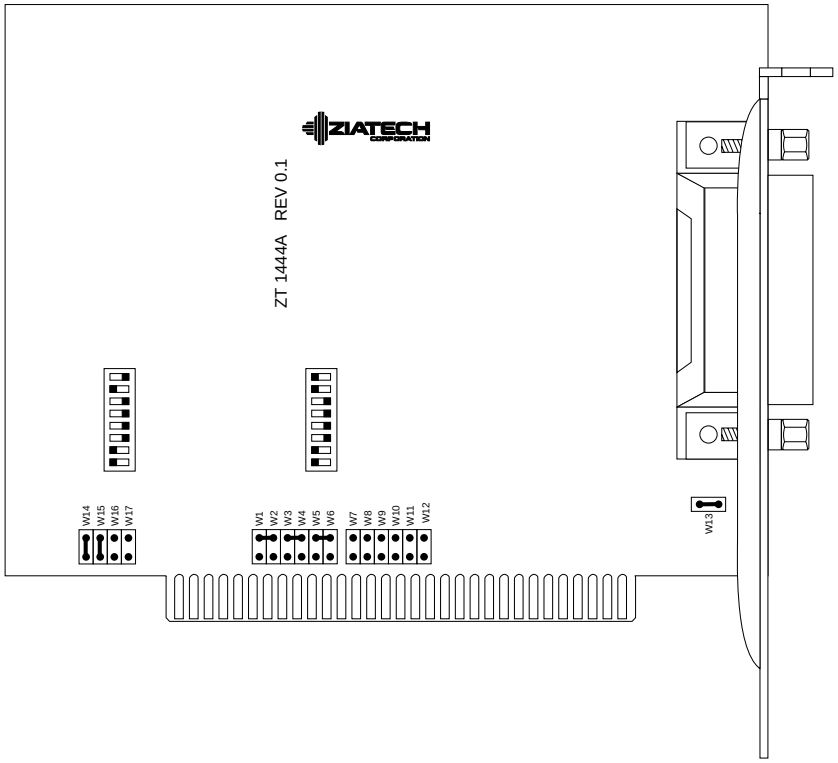
ZT 1444A jumper selections are summarized below. A detailed description of each jumper's function can be found in "ZT 1444A Jumper Descriptions".

Default jumper configurations for the ZT 1444A are illustrated in the ZT 1444A Default Jumpers (Controller) figure. You can use the ZT 1444A Customer Jumper Configuration figure to document your own configuration.

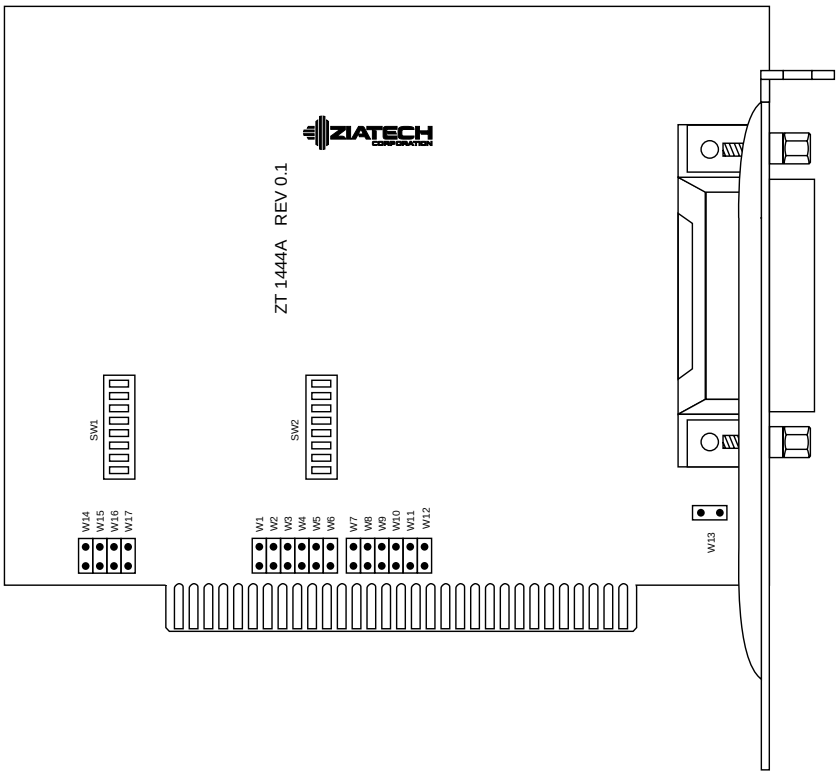
Note: DIP switch settings can occasionally become altered during shipping. If you encounter any addressing problems, reseat the DIP switches and check for the desired setting as described in the "ZT 1444A I/O Port Address Switch Configurations" topic.

ZT 1444A vs. ZT 1444

The ZT 1444A is a functional superset of the ZT 1444. A socket location for the Dallas Semiconductor DS 1204 security key has been added, which requires the board to decode a larger address space (from 8 ports to 16 ports). Jumpers W14-W17 allow this expansion. If you want your ZT 1444A to remain completely compatible with the ZT 1444, leave W14 and W15 installed (factory default). Note that W14 and W15 must not be installed at the same time as W16 and W17. Software written for the ZT 1444 will run in either mode, but some users may have another device mapped at the Base + 8 through Base + 15 location.



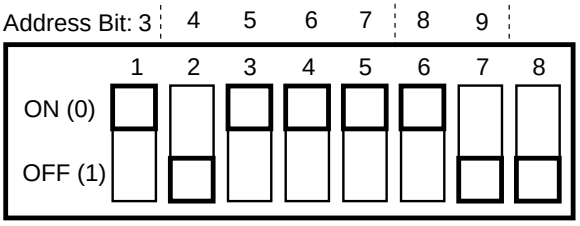
ZT 1444A Default Jumpers (Controller)



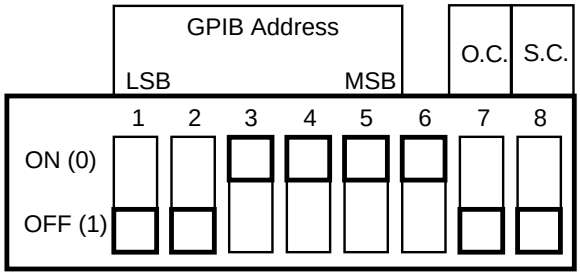
ZT 1444A Customer Jumper Configuration

ZT 1444A I/O Port Address Switch Configurations

The ZT 1444A has several options that you can set with two switches, SW1 and SW2.



ZT 1444A I/O Address DIP Switch (SW1)



ZT 1444A I/O Address DIP Switch (SW2)

SW1, Segments 1-7

Select I/O port address. SW1, segments 1-7, correspond to address lines A3-9, respectively. Factory default is 210h (segments 2 and 7 *off*). To change, open the SW1 switch segment corresponding to the new I/O address selected.

SW1, Segment 8

Switch position 8 is not used and is a "don't care' (it can be in either position).

SW2, Segments 1-5

Select IEEE 488 address (via switch segments 1-5) and system controller.

SW2, Segment 6

User-definable switch.

SW2, Segment 7

Selects IEEE 488 three-state when *off*, open collector when *on*.

SW2, Segment 8

Selects system controller when *off*, non system controller (including device) operation when *on*.

ZT 1444A Jumper Descriptions

W1-6

Select IRQ2-7, respectively. These jumpers have been factory installed in storage configuration perpendicular to jumper post locations.

W7, 9, 11

Select DRQ1-3, respectively.

W8,10,12

Select DACK1-3, respectively.

W13

Ground cable shield. Not installed for device operation.

W-14-15

Security key disable select. Installed for ZT 1444 compatibility. W14 and W15 must not be installed together.

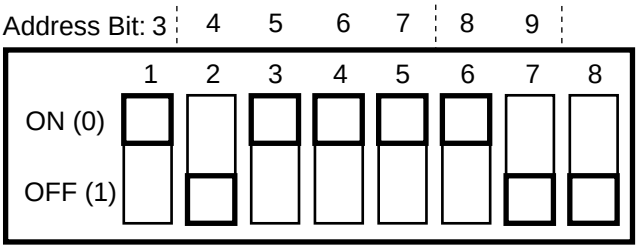
W16-17

Security key enable. Not installed. W16 and W17 must not be installed together.

Configuring the ZT 1444A

- **Is the IBM or TI I/O port address for the IEEE 488 to be other than 0210h? If yes, set address in DIP switch SW1.**

  The starting address of the eight IEEE 488 I/O ports must be on an 8-port boundary in the IBM I/O address space. Only 9 of 16 possible I/O address lines are decoded according to protocol. Illustrated below is the factory default setting of 0210h in switch positions 1-7 of SW1. Switch position 8 of SW1 is not used and is a *don't care*.



ZT 1444A I/O Address DIP Switch (SW1)

- **Are IEEE 488 or DMA Terminal Count (TC) interrupts to be used? If yes, select the desired IRQ line with jumpers W1-6.**

  Two possible interrupts occurring on the ZT 1444A are ORed together (see "ZT 1444A Interrupts and DMA" in Chapter 5 for details). The output of the ORed interrupt status must be enabled to generate an interrupt to the IBM PC. To select which interrupt line is used, select one of the interrupt request lines IRQ2-IRQ6 via W1-6.

As shipped default from Ziatech, no interrupt lines are enabled and the jumper is stored at right angles to W1-6 positions.

- **Is DMA to be used with the IEEE 488 and IBM PC? If yes, select desired DRQ and DACK lines via jumpers W7-12.**

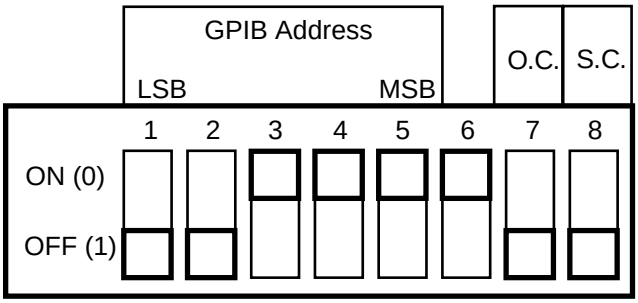
  DMA may be optionally used for the transfer of IEEE 488 data (refer to "ZT 1444A Interrupts and DMA" for details). IEEE 488 device requests are made to the IBM DMA controller via DRQ1-DRQ3. IEEE 488 device acknowledges are returned from the DMA controller via device acknowledge lines DACK1-DACK3. In a typical DMA system, DRQ1 corresponds to DACK1, DRQ2 to DACK2, DRQ3 to DACK3.

  Note that for IBM PC or XT operation, Ziatech software used DRQ1 and DACK1. For TI PC operation, DMA is not possible. See "ZT 1444A Interrupts and DMA" for jumper selection for DRQ1 and DACK1.

  The factory default has no DRQ or DACK lines enabled. Jumpers for these lines are stored at right angles to these jumper positions.

- **Is the IEEE 488 address to be different from 3? If yes, set address switch SW2.**

  This DIP switch (SW2) can be changed at any time to contain the IEEE 488 device address. Refer to the "Address Switch Register - General Purpose" topic in Chapter 6 for more information. The IEEE 488 address is set in switch numbers 1-5 requiring an AND with 1F hex to get the proper IEEE 488 address. See the ZT 1444A I/O Address DIP Switch (SW2) illustration below.



ZT 1444A I/O Address DIP Switch (SW2)

- **Is the ZT 1444A not the IEEE 488 System Controller? If yes, set system controller switch to "on."**

  This switch (SW2) located at pack position 4C #8, when *on*, disables IFC and REN lines from being asserted by the ZT 1444A. The system controller must now manage these lines. Refer to the ZT 1444A I/O Address DIP Switch (SW2) figure shown above.

  If yes, remove the shield ground, W13.

Any bus system that connects many pieces of equipment is a potential source of ground loops and spurious noise problems.

To help avoid these problems, only one IEEE 488 device should connect the shield in the IEEE 488 cable to earth ground. The ground is normally connected to ground by the system controller. The shield connection may be removed by cutting jumper W1 located next to the IEEE 488 header.

Switch position 7 on SW2 when in the *off* position enables three-state operation of the IEEE 488 drivers. This mode should be used when operating at high speeds with DMA. The *on* position enables open collector operation of the IEEE 488 drivers.

Switch position 6 on SW2 is a user-defined, software-readable switch. In the *on* position, a logical low is read.

- **Is a chassis ground via the mounting bracket not required? If yes, remove jumper W2.**

  Most systems do not have strict EMI emission requirements. Therefore W2 does not necessarily need to be installed even though the standard factory default loading shorts the metal mounting bracket to ground.

- **Is the ZT 1444A to remain completely compatible with the ZT 1444? If yes, install jumpers W14 and W15 and remove W16 and W17.**

- **Is the on-board security key option to be used? If yes, install jumpers W16 and W17 and remove W14 and W15.**

  These jumpers select whether or not the ZT 1444A will decode the I/O addresses where the security key is installed. This requires the mapping of eight additional I/O locations. W16 and W17 map the board for that configuration. DO NOT install both sets of jumpers W14-15 and W16-17.

B. CUSTOMER SUPPORT

This appendix offers technical assistance information for this product, and also the necessary information should you need to return a Ziatech product.

TECHNICAL/SALES ASSISTANCE

If you have a technical question, please call Ziatech's Customer Support Service at the number below, or e-mail our technical support team at *tech_support@ziatech.com*. Ziatech also maintains an FTP site located at *ftp.ziatech.com*.

If you have a sales question, please contact your local Ziatech Sales Representative or the Regional Sales Office for your area. Address, telephone and FAX numbers, and additional information is available at Ziatech's website, located at http://www.ziatech.com.

Corporate Headquarters

1050 Southwood Drive

San Luis Obispo, CA 93401 USA

Tel (805) 541-0488

FAX (805) 541-5088

RELIABILITY

Ziatech has taken extra care in the design of the ZT 1444A in order to ensure reliability. The product was designed in top-down fashion, using the latest in hardware and software design techniques, so that unwanted side effects and unclean interactions between parts of the system are eliminated. Each ZT 1444A has an identification number. Ziatech maintains a lifetime data base on each board and the components used. Any negative trends in reliability are spotted and Ziatech's suppliers are informed and/or changed.

Editor's Note: *This manual originally documented both the ZT 1444A and the ZT 1488A Interface boards. Please note that Ziatech has discontinued the ZT 1488A, and has therefore removed from this manual whole topics devoted exclusively to the ZT 1488A. Please ignore any incidental references to the ZT 1488A still contained in this manual.*

RETURNING FOR SERVICE

Before returning any of Ziatech's products, you must phone Ziatech at (805) 541-0488 and obtain a Returned Material Authorization (RMA) number. The following information is needed to expedite the shipment of a replacement to you:

1. Your company name and address for invoice
2. Shipping address and phone number
3. Product I.D. number
4. If possible, the name of a technically qualified individual at your company familiar with the mode of failure on the board

If the unit is out of warranty, service is available at a predesignated service charge. Contact Ziatech for pricing and please supply a purchase order number for invoicing the repair.

Pack the board in **anti-static** material and ship in a sturdy cardboard box with enough packing material to adequately cushion it. ***Any product returned to Ziatech improperly packed will immediately void the warranty for that particular product!*** Mark the RMA number clearly on the outside of the box before returning.

ZIATECH WARRANTY

Ziatech provides a five-year limited warranty to its customers. Ziatech also has an explicit policy regarding the use of Ziatech products in life support systems. These topics are covered in the following sections.

Five-Year Limited Warranty

Products manufactured by Ziatech Corporation are covered from the date of purchase by a five-year warranty against defects in materials, workmanship, and published specifications applicable to the date of manufacture. During the warranty period, Ziatech will repair or replace, solely at its option, defective units provided they are returned at customer expense to an authorized Ziatech repair facility. Products which have been subjected to misuse, abuse, neglect, alteration, or unauthorized repair, determined at the sole discretion of Ziatech, whether by accident or otherwise, are excluded from warranty. The warranty on fans and disk drives is limited to two years and the warranty on flat panel displays is limited to nine months from date of purchase. Other products and accessories not manufactured by Ziatech are limited to the warranty provided by the original manufacturer. Consumable items (fuses, batteries, etc.) and software are not covered by this warranty.

Ziatech Corporation warrants that for a period of ninety (90) days from the date of purchase; the media on which software is furnished will be free of defects in materials and workmanship under normal use; and the software contains the features described in the Ziatech price list. Otherwise, the software is provided "AS IS". This limited

warranty extends only to Customer as the original licensee. Customer's exclusive remedy and Ziatech's entire liability under this limited warranty will be, at Ziatech's option, to repair or replace the software, or refund the license fee paid therefore.

Ziatech may offer, where applicable and available, replacement products; otherwise, repairs requiring components, assemblies, and other purchased materials may be limited by market availability.

Ziatech assumes no liability resulting from changes to government regulations affecting use of materials, equipment, safety, and methods of repair. Ziatech may, at its discretion, offer replacement products.

THE ABOVE WARRANTY IS IN LIEU OF ANY OTHER WARRANTY, WHETHER EXPRESSED, IMPLIED, OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTY FOR FITNESS OF PURPOSE, MERCHANTABILITY, OR FREEDOM FROM INFRINGEMENT OR THE LIKE, AND ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATIONS, OR SAMPLE.

Ziatech neither assumes nor authorizes any person to assume for it any other liability. The liability of Ziatech under this warranty agreement is limited to a refund of the purchase price. In no event shall Ziatech be liable for loss of profits, use, incidental, consequential, or other damage, under this agreement.

Life Support Policy

Ziatech products are not authorized for use as critical components in life support devices or systems without the express written approval of the president of Ziatech Corporation. As used herein:

1. Life support devices or systems are devices or systems which support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.

2. A critical component is any component of a life support device or system whose failure to perform can be expected to cause the failure of the life support device or system, affect its safety, or limit its effectiveness.

TRADEMARKS

MULTIMODULE and iSBX are registered trademarks of Intel Corporation.
IBM PC/XT/AT, PS/2, and PC DOS are registered trademarks of International Business Machines, Inc.
MS-DOS and Microsoft C are registered trademarks of Microsoft Corporation.
Turbo C is a trademark of Borland International, Inc.

All other brand and product names may be trademarks or registered trademarks of their respective holders.

©Copyright 2000 Ziatech Corporation

C. IEEE 488 OVERVIEW

This section provides an introduction to the IEEE 488 GPIB (HP-IB, IEC) bus specification.

WHAT IS THE IEEE 488 (GPIB)?

In the early 1970s, Hewlett-Packard defined a standard mechanism to facilitate assembly of instrumentation systems of varying degrees of complexity. Prior to this, each interface was designed from scratch and inconsistent in electrical levels, connector types, and pin-outs. With every system built, new cables and documentation were invented to specify cabling and interconnection procedures.

When Hewlett-Packard defined the new standard interconnection scheme, they also specified typical communication protocol. Their first 1972 version of the bus has since been modified to the present IEEE 488 Interface bus (also known as the HP-IB, the GPIB, and the IEC bus). This specification is an amalgam of the goals of various instrumentation and computer peripheral manufacturers to produce a common interconnection mechanism. Design objectives for the GPIB are outlined below.

Design Objectives

1. Specify an easy-to-use system with all terminology and definitions related to that system precisely spelled out.

2. Define mechanical, electrical, and functional interface requirements of a system, yet leave device aspects to the designer).

3. Permit a wide capability range for instruments and computer peripherals which, when used simultaneously, do not degrade the performance of any other.

4. Allow different manufacturers' equipment to be interconnected and work together.

5. Define a system effective for limited distance interconnections.

6. Define a system with a minimum of performance restrictions.

7. Define a bus allowing asynchronous communications with a wide range of data rates.

8. Define a system not requiring extensive and elaborate interface logic for low-cost instruments, yet providing higher capability for higher cost instruments if desired.

9. Provide for systems not requiring a central controller; that is, communication directly from one instrument to another.

Bus Characteristics

While the IEEE 488 was originally designed for instrumentation systems, most of these systems are controlled by a computer. With this in mind, several modifications were made to the original proposal before its final adoption as an international standard. The following list highlights the salient characteristics of the IEEE 488 both as an instrumentation bus and as a computer I/O bus.

Data Rate	1 Mbyte/second, maximum 300-450 Kbytes/second, typical
Multiple Devices	15 devices, maximum (electrical limit) 8 devices, typical (interrupt flexibility)
Bus Length	20 meters, maximum 2 meters per device, typical
Byte-Oriented	8-bit commands 8-bit data
Block-Multiplexed	Optimum strategy on GPIB due to setup overhead for commands
Interrupt-Driven	Serial Poll (slower devices) Parallel Poll (faster devices)
Direct Memory Access	One DMA facility (controller) serves all devices on bus
Asynchronous	One talker Multiple listeners
I/O -to-I/O Transfers	Talker and listeners need not include microcomputer/controller

The above characteristics can best be understood by examining the IEEE 488 bus as though it were a general microcomputer I/O bus.

Data Rate

Most microcomputer systems utilize peripherals of differing operational rates, such as floppy discs at 31/62 Kbytes/second (single or double density), tape cassettes at 5 Kbytes to 10 Kbytes/second, and cartridge tapes at 40 Kbytes to 80 Kbytes/second. In general, the only devices that need high speed I/O are high speed magnetic tapes and hard discs that operate at speeds of 30 Kbytes to 781 Kbytes/second, respectively. Certainly the 300 Kbytes/second data rate that can be easily achieved by the IEEE 488 bus is sufficient for microcomputers and their peripherals and is more than enough for

typical analog instruments that take only a few readings per second. The 1 Mbyte maximum data rate is not easily achieved on the GPIB and requires a more complex and expensive implementation than is necessary for most instrument systems. Although not required, data buffering in each device improves the overall bus performance and allows better utilization of the system bandwidth.

Multiple Devices

The average instrumentation computer must handle from three to seven instruments/peripherals. With the IEEE 488, up to eight devices can be handled easily by one controller; and with some slowdown in interrupt handling, up to fifteen devices can be accommodated. The limit of eight is imposed by the number of unique parallel poll responses available; the limit of fifteen total devices is set by the electrical drive characteristics of the bus. Logically, the IEEE 488 standard is capable of accommodating more device addresses (31 primary, each potentially with 31 secondaries).

Bus Length

Physically, the majority of microcomputer systems fit easily on a desk top or in a standard 19" (48 cm) rack, eliminating the need for long cables. The IEEE 488 is designed to accommodate 2 meters of cable per device. A line printer, for example, might require greater cable lengths, but this can be handled by using dummy terminations. Overall cable length should be kept to a minimum (maximum of 20 meters) to ensure data integrity.

Byte-Oriented

The 8-bit byte is almost universal in I/O applications; even 16-bit and 32-bit computers use byte transfers for most peripherals. The 8-bit byte matches the ASCII code for characters and is an integral submultiple of most computer word sizes. The IEEE 488 has an 8-bit wide data path that may be used to transfer ASCII or binary data, as well as status and control bytes.

Block-Multiplexed

Many peripherals are block-oriented or are used in a block mode. Bytes are transferred in a group of fixed or variable length. There is then a wait before another group is sent to that device, for example one sector of a floppy disc or one line on a printer or tape punch. The IEEE 488 is, by nature, a block-multiplexed bus due to the overhead involved in addressing various devices to talk and listen. This overhead is less bothersome if it occurs only once for a large number of data bytes (once per block). This mode of operation matches the needs of microcomputers and most of their peripherals. Because of block multiplexing, the bus works best with buffered memory devices and/or devices that can operate with Direct Memory Access (DMA).

Interrupt-Driven

Many types of interrupt systems exist, ranging from complex, fast, vectored or priority networks to simple polling schemes. The main trade-off is usually cost versus speed of response. The IEEE 488 has two interrupt protocols. The first is the single service request (SRQ) line that may be asserted by any interrupting device to get the attention of the controller, which then polls to find the origin of the interrupt. The second is common polling of devices to determine which needs service. For higher performance, parallel polling allows up to eight devices to be polled at once; each device is assigned to one bit of the data bus. This mechanism provides fast recognition of an interrupting device.

Direct Memory Access (DMA)

In many applications, immediate processing of I/O data on a byte-by-byte basis is not required. Programmed transfers slow down the data transfer rate unnecessarily when higher speeds can be obtained using DMA. With the IEEE 488, one DMA facility at the controller serves all devices. There is no need to incorporate complex logic in each device.

Asynchronous Transfers

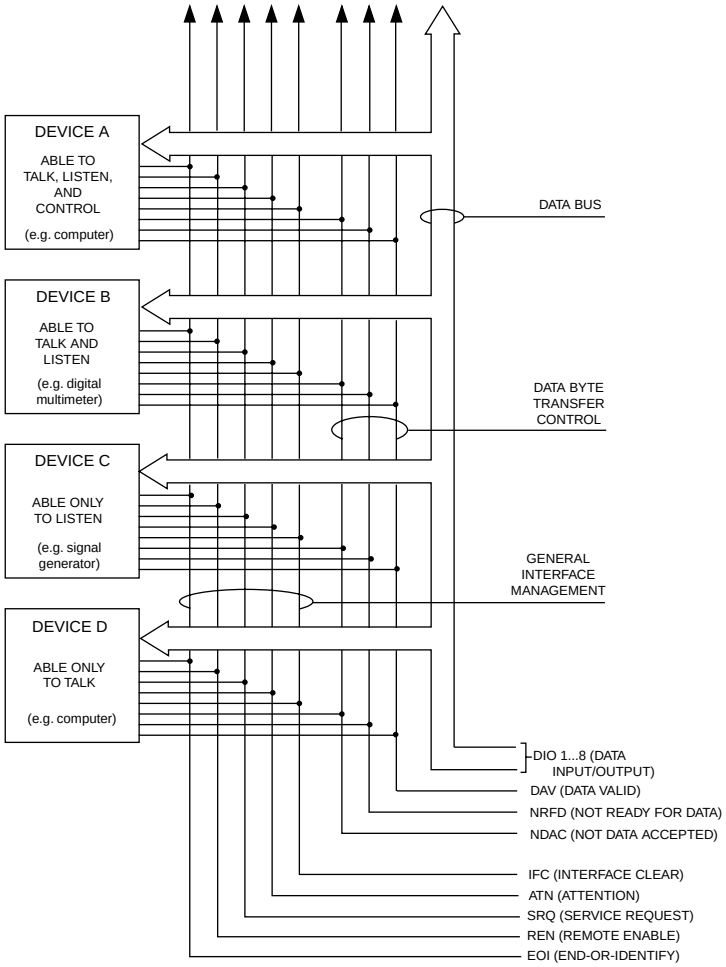
An asynchronous bus is desirable because it allows each device to transfer data at its own rate. However, there is still a good reason to buffer the data at each device when used in large systems: to speed up the aggregate data rate on the bus by allowing each device to transfer at its own top speed. The IEEE 488 is asynchronous and uses a special 3-wire handshake that allows data transfers from one talker to many listeners.

I/O-To-I/O Transfers

In practice, I/O-to-I/O transfers are seldom performed due to the need for processing data or changing formats, or because of mismatched data rates. However, the IEEE 488 can support this mode of operation in which the microcomputer is neither the talker nor one of the listeners. In this mode of operation, the operational speed of the devices determines the transfer rate.

IEEE 488 SIGNAL LINES

The IEEE 488 Interface figure illustrates the IEEE 488 signal lines, which include the Data Bus, Management Bus, and Transfer Bus. Discussion of the individual buses and signal lines follow.



IEEE 488 Interface

Data Bus

The lines DI01 through DI08 transfer addresses and control information and data. The IEEE 488 standard defines formats for addresses and control bytes. Data formats are undefined and may be ASCII (with or without parity) or binary. DI01 is the Least Significant Bit (note that this corresponds to bit 0 on most computers).

Management Bus

ATN: Attention

This signal is asserted by the controller to indicate that it is placing an address or control byte on the data bus. ATN is de-asserted to allow the assigned talker to place status or data on the data bus. The controller regains control by reasserting ATN; this is normally done synchronously with the handshake to avoid confusion between control and data bytes.

EOI: End Or Identify

This signal has two uses, as its name implies. A talker may assert EOI simultaneously with the last byte of data to indicate end-of-data. The controller may assert EOI along with ATN to initiate a Parallel Poll. Although many devices do not use Parallel Poll, all devices should use EOI to end transfers (many devices currently available do not).

SRQ: Service Request

This line, similar to an interrupt line, is asserted by any device to request the controller to take some action. The controller must determine which device is asserting SRQ by conducting a Serial Poll. The requesting device de-asserts SRQ when polled.

IFC: Interface Clear

This signal is asserted only by the system controller in order to initialize all device interfaces to a known state. After de-asserting IFC, the system controller is the active controller of the system.

REN: Remote Enable

This signal is asserted only by the system controller. Its assertion does not place devices into Remote Control Mode; REN enables a device to go remote only when addressed to listen. When in Remote, a device should ignore its front panel controls.

Transfer Bus

NRFD: Not Ready For Data

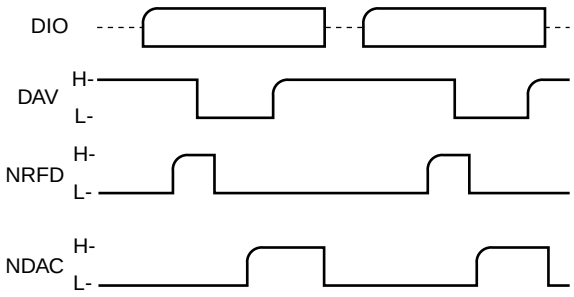
This handshake line is asserted by a listener to indicate it is not yet ready for the next data or control byte. Note that the controller will not see NRFD de-asserted (that is, ready for data) until all devices have de-asserted NRFD. See the IEEE 488 Handshake Sequence figure.

NDAC: Not Data Accepted

This handshake line is asserted by a listener to indicate it has not yet accepted the data or control byte on the DIO lines. Note that the controller will not see NDAC de-asserted (that is, data accepted) until all devices have de-asserted NDAC.

DAV: Data Valid

This handshake line is asserted by the talker to indicate that a data or control byte has been placed on the DIO lines and has had the minimum specified settling time.



IEEE 488 Handshake Sequence

IEEE 488 INTERFACE FUNCTIONS

The IEEE 488 standard specifies 10 interface functions. Not all devices have all functions and some may have only partial subsets. The 10 functions are summarized below with the relevant section number from the IEEE document shown in parentheses.

SH: Source Handshake (Section 2.3)

This function lets a device properly transfer data from a talker to one or more listeners using the three handshake lines.

AH: Acceptor Handshake (Section 2.4)

This function lets a device properly receive data from the talker using the three handshake lines. The AH function may also delay the beginning (NRFD) or end (NDAC) of any transfer.

T: Talker (Section 2.5)

This function lets a device send status and data bytes when addressed to talk. An address consists of one (primary) or two (primary and secondary) bytes. The latter is called an Extended Talker.

L: Listener (Section 2.6)

This function lets a device receive data when addressed to listen. There can be Extended Listeners (analogous to Extended Talkers above).

SR: Service Request (Section 2.7)

This function lets a device request service; that is, interrupt the controller. The SRQ line may be asserted asynchronously.

RL: Remote Local (Section 2.8)

This function lets a device be operated in two modes: Remote via the IEEE 488 or Local via the manual front panel controls.

PP: Parallel Poll (Section 2.9)

This function lets a device present one bit of status to the controller-in-charge. The device need not be addressed to talk, and no handshake is required.

DC: Device Clear (Section 2.10)

This function lets a device be cleared (initialized) by the controller. Note there is a difference between DC (device clear) and the IFC line (interface clear).

DT: Device Trigger (Section 2.11)

This function lets a device have its basic operation started either individually or as part of a group. This capability is often used to synchronize several instruments.

C: Controller (Section 2.12)

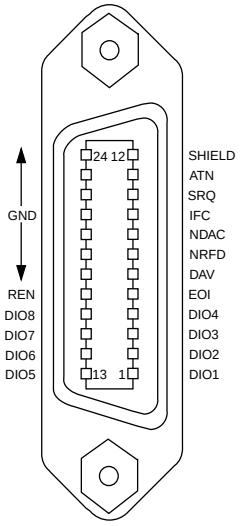
This function lets a device send addresses, as well as universal and addressed commands, to other devices. There may be more than one controller on a system, but only one may be the controller-in-charge at any one time.

At power-on time, the controller programmed to be the system controller becomes the active controller-in-charge. The system controller has several unique capabilities, including the ability to send Interface Clear (IFC clears all device interfaces and returns control to the system controller) and to send Remote Enable (REN allows devices to respond to bus data once they are addressed to listen). The system controller may optionally pass control to another controller if the system software has that capability.

THE IEEE 488 CONNECTOR

The IEEE 488 connector (see the figure below) is a standard 24-pin industrial connector such as Cinch or Amphenol series 57 micro-Ribbon. The IEEE standard specifies this connector, the signal connections, and the mounting hardware.

The cable has 16 signal lines and eight ground lines. The maximum length is 20 meters with no more than two meters average per device.



IEEE 488 Connector

IEEE 488 SIGNAL LEVELS

The IEEE 488 signals are all TTL-compatible, low true signals. A signal is asserted (true) when its electrical voltage is less than 0.5 V and is de-asserted (false) when it is greater than 2.4 V. Be careful not to confuse the two handshake signals, NRFD and NDAC, which are also low true. 0.5 V implies the device is Not Ready For Data.

D. IEEE 488 REMOTE MESSAGE CODING

IEEE Standard 488-1978 lists all messages capable of being sent (talk) or received (listen) by an interface function. Coding for these messages is provided in this section.

INTRODUCTION

The remote message coding shown below includes both the encoding required to send the message and the decoding required to receive it. The logical state of each bus line signal is specified in the following message coding as 0, 1, Y, or X as follows:

0 = logical zero

1 = logical one

X = don't care (for received message)

Y = don't care (for send message)

Other symbols used in the remote message coding are:

U = Uniline message

M = Multiline message

AC = Addressed Command

AD = Address (talk or listen)

DD = Device Dependent

HS = Handshake

UC = Universal Command

SE = Secondary

ST = Status

MESSAGE CODING

Remote message coding is presented in table form below.

Remote Message Coding

Mnemonic / Message Name		Type/ Class	Bus Signal Line(s) and Coding That Asserts the True Value of the Message															
			DIO8	7	6	5	4	3	2	DIO1	DAV	NRFD	NDAC	ATN	EOI	SRQ	IFC	REN
ACG	Addressed Command Grp.	M AC	Y	0	0	0	X	X	X	X	X	X	X	1	X	X	X	X
ATN	Attention	U UC	X	X	X	X	X	X	X	X	X	X	X	1	X	X	X	X
DAB	Data Byte[1,9]	M DD	D8	D7	D6	D5	D4	D3	D2	D1	X	X	X	0	X	X	X	X
DAC	Data Accepted	U HS	X	X	X	X	X	X	X	X	X	X	0	X	X	X	X	X
DAV	Data Valid	U HS	X	X	X	X	X	X	X	X	1	X	X	X	X	X	X	X
DCL	Device Clear	M UC	Y	0	0	1	0	1	0	0	X	X	X	1	X	X	X	X
END	End	U ST	X	X	X	X	X	X	X	X	X	X	X	0	1	X	X	X
EOS	End Of String[2,9]	M DD	E8	E7	E6	E5	E4	E3	E2	E1	X	X	X	0	X	X	X	X
GET	Group Execute Trigger	M AC	Y	0	0	0	1	0	0	0	X	X	X	1	X	X	X	X
GTL	Go To Local	M AC	Y	0	0	0	0	0	0	1	X	X	X	1	X	X	X	X
IDY	Identify	U UC	X	X	X	X	X	X	X	X	X	X	X	X	1	X	X	X
IFC	Interface Clear	U UC	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	X
LAG	Listen Address Group	M AD	Y	0	1	X	X	X	X	X	X	X	X	1	X	X	X	X
LLO	Local Lock Out	M UC	Y	0	0	1	0	0	0	1	X	X	X	1	X	X	X	X
MLA	My Listen Address[3]	M AD	Y	0	1	L5	L4	L3	L2	L1	X	X	X	1	X	X	X	X
MTA	My Talk Address[4]	M AD	Y	1	0	T5	T4	T3	T2	T1	X	X	X	1	X	X	X	X
MSA	My Secondary Address[5]	M SE	Y	1	1	S5	S4	S3	S2	S1	X	X	X	1	X	X	X	X
NUL	Null Byte	M DD	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X
OSA	Other Secondary Addr.	M SE	(OSA = SCG & !MSA)															
OTA	Other Talk Address	M AD	(OTA = TAG & !MTA)															
PCG	Primary Command Group	M --	(PCG = ACG + UCG + LAG + TAG)															
PPC	Par. Poll Configure	M AC	Y	0	0	0	0	1	0	1	X	X	X	1	X	X	X	X
PPE	Par. Poll Enable[6]	M SE	Y	1	1	0	S	P3	P2	P1	X	X	X	1	X	X	X	X
PPD	Par. Poll Disable[7]	M SE	Y	1	1	1	D4	D3	D2	D1	X	X	X	1	X	X	X	X
PPR1	Par. Poll Response 1[10]	U ST	X	X	X	X	X	X	X	1	X	X	X	1	1	X	X	X
PPR2	Par. Poll Response 2[10]	U ST	X	X	X	X	X	X	1	X	X	X	X	1	1	X	X	X
PPR3	Par. Poll Response 3[10]	U ST	X	X	X	X	X	1	X	X	X	X	X	1	1	X	X	X

Mnemonic	Message Name	Type	Class	Bus Signal Line(s) and Coding That Asserts the True Value of the Message: DIO8	7	6	5	4	3	2	DIO1	DAV	NRFD	NDAC	ATN	EOI	SRQ	IFC	REN
PPR4	Par. Poll Response 4[10]	U	ST	X	X	X	X	1	X	X	X	X	X	X	1	1	X	X	X
PPR5	Par. Poll Response 5[10]	U	ST	X	X	X	1	X	X	X	X	X	X	X	1	1	X	X	X
PPR6	Par. Poll Response 6[10]	U	ST	X	X	1	X	X	X	X	X	X	X	X	1	1	X	X	X
PPR7	Par. Poll Response 7[10]	U	ST	X	1	X	X	X	X	X	X	X	X	X	1	1	X	X	X
PPR8	Par. Poll Response 8[10]	U	ST	1	X	X	X	X	X	X	X	X	X	X	1	1	X	X	X
PPU	Par. Poll Unconfigure	M	UC	Y	0	0	1	0	1	0	1	X	X	X	1	X	X	X	X
REN	Remote Enable	U	UC	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1
RFD	Ready For Data	U	HS	X	X	X	X	X	X	X	X	X	0	X	X	X	X	X	X
RQS	Request Service[9]	U	ST	X	1	X	X	X	X	X	X	X	X	X	0	X	X	X	X
SCG	Secondary Command Grp.	M	SE	Y	1	1	X	X	X	X	X	X	X	X	1	X	X	X	X
SDC	Selected Device Clear	M	AC	Y	0	0	0	0	1	0	0	X	X	X	1	X	X	X	X
SPD	Serial Poll Disable	M	UC	Y	0	0	1	1	0	0	1	X	X	X	1	X	X	X	X
SPE	Serial Poll Enable	M	UC	Y	0	0	1	1	0	0	0	X	X	X	1	X	X	X	X
SRQ	Service Request	U	ST	X	X	X	X	X	X	X	X	X	X	X	1	X	X	X	X
STB	Status Byte[8,9]	M	ST	S8	X	S6	S5	S4	S3	S2	S1	X	X	X	0	X	X	X	X
TCT	Take Control	M	AC	Y	0	0	0	1	0	0	1	X	X	X	1	X	X	X	X
TAG	Talk Address Group	M	AD	Y	1	0	X	X	X	X	X	X	X	X	1	X	X	X	X
UCG	Universal Command Grp.	M	UC	Y	0	0	1	X	X	X	X	X	X	X	1	X	X	X	X
UNL	Unlisten	M	AD	Y	0	1	1	1	1	1	1	X	X	X	1	X	X	X	X
UNT	Untalk[11]	M	AD	Y	1	0	1	1	1	1	1	X	X	X	X	X	1	X	X

The I/O coding on ATN when sent concurrent with multiline messages has been added to this revision for interpretive convenience.

NOTES:
1. D1-D8 specify the device-dependent data bits.
2. E1-E8 specify the device-dependent code used to indicate the EOS message.
3. L1-L5 specify the device-dependent bits of the device's listen address.
4. T1-T5 specify the device-dependent bits of the device's talk address.
5. S1-S5 specify the device-dependent bits of the device's secondary address.
6. S specifies the sense of the PPR. P1-P3 specify the PPR message to be sent when a parallel poll is executed.

S	Response	P3	P2	P1	PPR Message
0	0	0	0	0	PPR1
1	1	.	.	.	.
		.	.	.	.
		.	.	.	.
		1	1	1	PPR8

7. D1-D4 specify don't-care bits that shall not be decoded by the receiving device. It is recommended that all zeroes be sent.
8. S1-S6, S8 specify the device-dependent status. (DIO7 is used for the RQS message.)
9. The source of the message on the ATN line is always the C function, whereas the messages on the DIO and EOI lines are enabled by the T function.
10. The source of the messages on the ATN and EOI lines is always the C function, whereas the source of the messages on the DIO lines is always the PP function.
11. This code is provided for system use; see 6.3.

E. IEEE 488 DATA RATES

The theoretical data rates illustrated in this section assume Ziatech's software is used to send and receive data. Data rates have been verified with a 1 Mbyte talker and listener.

INTRODUCTION

Theoretical data rates for sending and receiving are illustrated in the figures IEEE 488 Send Data Rate and IEEE 488 Receive Data Rate. These data rates are for sending and receiving data only; no IEEE 488 command time is included. To include command time and data transfer time, equations are given below. These equations are of the y=mx+b type where *x* is the number of bytes transferred, *m* is the transfer time per byte, and *b* is the command time. All times are in µseconds assuming a 4.77 MHz 8088 is used in the IBM PC. If a different speed 8088 is used, for example, 5.0 for the TI, multiply the transfer time (t) by the new clock speed divided by 4.77 MHz.

DATA RATES

ZT\ 1444A/1488A talking:

a) 21.2 Kbytes/second using Ziatech's *SENDST* statement with no timeout in effect.

Note: This is a theoretical data transmission rate of an infinite data string after initialization, assuming data accepted is true within 20.1 µseconds after the leading edge of data valid. If the device response is not 20.1 µseconds, refer to the IEEE 488 Send Data Rate figure. This figure illustrates the expected data rates for various response times for the IBM and TI (TI rates in parentheses).

To calculate transfer time for a given length of data (x) + IEEE 488 command time, use the following equation:

$$t(\mu seconds) = 47x + 300$$

b) 450 Kbytes/second using Ziatech's *SENDDM* DMA statement.

Note: This is a theoretical data transmission rate of an infinite data string after initialization using DMA. Typical DMA rates could be 170-200 Kbytes/second.

To calculate transfer time for a given length of data (x) + IEEE 488 command time, use the following equation:

$$t(\mu seconds) = 5x + 325$$

ZT\ 1444A/1488A listening:

a) 15.6 Kbytes/second using Ziatech's *RECVST* statement with no timeout or terminator in effect.

Note: This is a theoretical data transmission rate of an infinite data string after initialization, assuming data valid is true within 23.3 µseconds of ready-for-data. If the device response is not 23.3 µseconds, refer to the IEEE 488 Receive Data Rate figure. This figure illustrates data rates for various response times.

To calculate transfer time for a given length of data (x) + IEEE 488 command time, use the following equation:

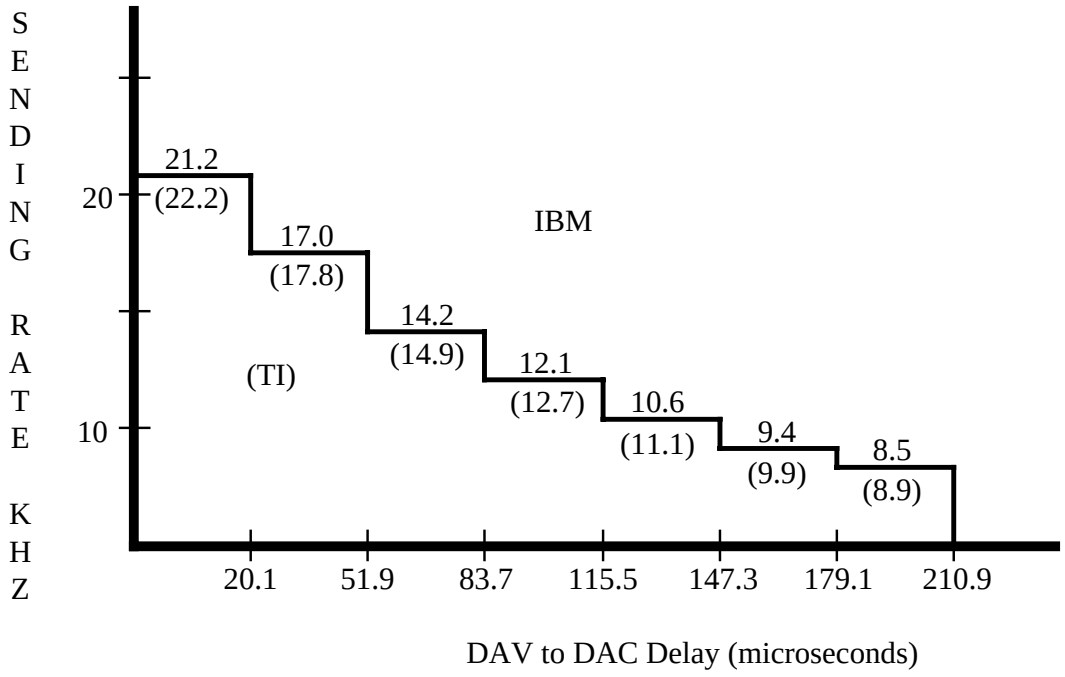
$$t(\mu seconds) = 64x + 370$$

b) 170 Kbytes/second using Ziatech's *RECVDM* DMA statement.

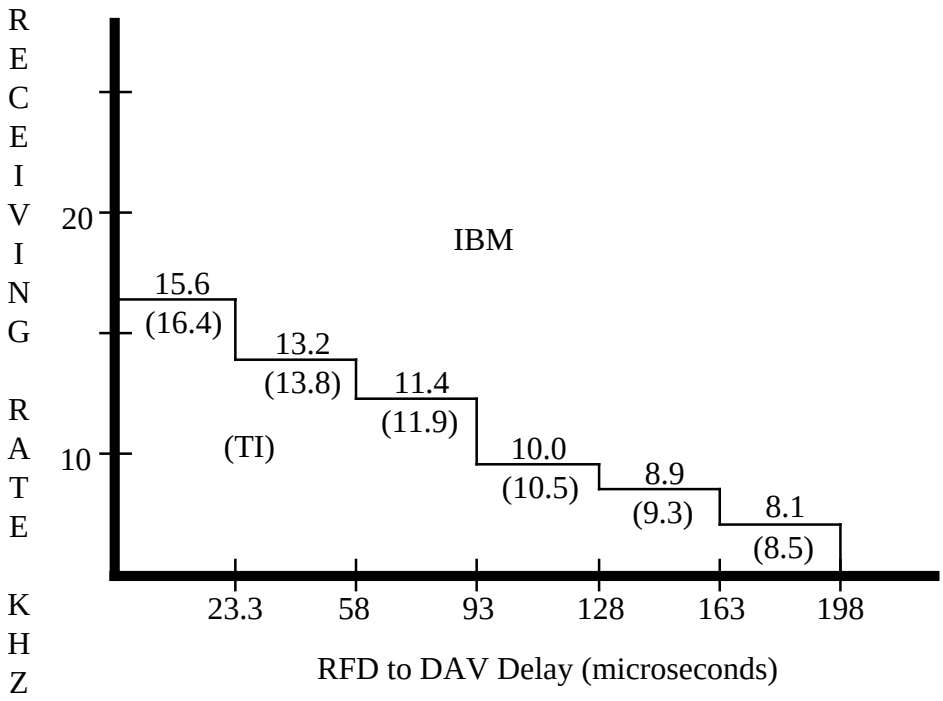
Note: This is a theoretical data transmission rate of an infinite data string after initialization using DMA. Typical DMA rates could be 170-200 Kbytes/second.

To calculate transfer time for a given length of data (x) + IEEE 488 command time, use the following equation:

$$t(\mu seconds) = 5x + 375$$



IEEE 488 Send Data Rate



IEEE 488 Receive Data Rate



1050 Southwood Drive
San Luis Obispo, CA 93401 USA
Tel: (805) 541-0488
FAX: (805) 541-5088
E-Mail: tech_support@ziatech.com
Internet: http://www.ziatech.com

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP