
COMPREHENSIVE SERVICES
We offer competitive repair and calibration services, as well as easily accessible documentation and free downloadable resources.
SELL YOUR SURPLUS
We buy new, used, decommissioned, and surplus parts from every NI series.
We work out the best solution to suit your individual needs.
Sell For Cash
Get Credit
Receive a Trade-In Deal
OBSOLETE NI HARDWARE IN STOCK & READY TO SHIP
We stock New, New Surplus, Refurbished, and Reconditioned NI Hardware.
APEX WAVES
Bridging the gap between the manufacturer and your legacy test system.
1-800-915-6216
www.apexwaves.com
sales@apexwaves.com
All trademarks, brands, and brand names are the property of their respective owners.
Request a Quote
CLICK HERE
GPIB-1014DP

GPIB-1014D
User Manual

March 1997 Edition

Part Number 320140-01

National Instruments Corporate Headquarters
6504 Bridge Point Parkway
Austin, TX 78730-5039
(512) 794-0100
Technical support phone: (512) 795-8248
Technical support fax: (512) 794-5678

Branch Offices:
Australia 03 9879 5166, Austria 0662 45 79 90 0, Belgium 02 757 00 20, Canada (Ontario) 905 785 0085,
Canada (Québec) 514 694 8521, Denmark 45 76 26 00, Finland 90 527 2321, France 1 48 14 24 24,
Germany 089 741 31 30, Hong Kong 2645 3186, Israel 03 5734815, Italy 02 413091, Japan 03 5472 2970,
Korea 02 596 7456, Mexico5 520 2635, Netherlands 0348 433466, Norway 32 84 84 00,
Singapore 2265886, Spain 91 640 0085, Sweden 08 730 49 70, Switzerland 056 20 51 51,
Taiwan 02 377 1200, U.K. 01635 523545

Limited Warranty

The GPIB-1014D is warranted against defects in materials and workmanship for a period of two years from the date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace equipment that proves to be defective during the warranty period. This warranty includes parts and labor.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. National Instruments will pay the shipping costs of returning to the owner parts which are covered by warranty.

National Instruments believes that the information in this manual is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, National Instruments reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult National Instruments if errors are suspected. In no event shall National Instruments be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, NATIONAL INSTRUMENTS MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF NATIONAL INSTRUMENTS SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. NATIONAL INSTRUMENTS WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF. This limitation of the liability of National Instruments will apply regardless of the form of action, whether in contract or tort, including negligence. Any action against National Instruments must be brought within one year after the cause of action accrues. National Instruments shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failures caused by owner's failure to follow the National Instruments installation, operation, or maintenance instructions; owner's modification of the product; owner's abuse, misuse, or negligent acts; and power failure or surges, fire, flood, accident, actions of third parties, or other events outside reasonable control.

Copyright

Trademarks

Warning Regarding Medical and Clinical Use of National Instruments Products

National Instruments products are not designed with components and testing intended to ensure a level of reliability suitable for use in treatment and diagnosis of humans. Applications of National Instruments products involving medical or clinical treatment can create a potential for accidental injury caused by product failure, or by errors on the part of the user or application designer. Any use or application of National Instruments products for or involving medical or clinical treatment must be performed by properly trained and qualified medical personnel, and all traditional medical safeguards, equipment, and procedures that are appropriate in the particular situation to prevent serious injury or death should always continue to be used when National Instruments products are being used. National Instruments products are NOT intended to be a substitute for any form of established process, procedure, or equipment used to monitor or safeguard human health and safety in medical or clinical treatment.

Contents

Figures

Tables

Preface

The GPIB-1014D is a double-height circuit board that interfaces the VMEbus to the IEEE-488 General Purpose Interface Bus (GPIB). The GPIB-1014D is used to implement VMEbus test and measurement systems using standard interconnecting cables.

Organization of This Manual

This manual describes the mechanical and electrical aspects of the GPIB-1014D and contains information concerning its operation and programming. The manual is divided into the following chapters and appendixes:

- Chapter 1, *Introduction*, describes the GPIB-1014D features, lists the contents of your GPIB-1014D kit, and explains how to unpack the GPIB-1014D kit.
- Chapter 2, *General Description*, contains the physical and electrical specifications for the GPIB-1014D and describes the characteristics of key interface board components.
- Chapter 3, *Configuration and Installation*, describes the steps needed to configure the GPIB-1014D hardware and to verify that it is functioning properly.
- Chapter 4, *Register Bit Descriptions*, contains detailed GPIB Interface register descriptions of the NEC µPD7210 GPIB Talker/Listener/Controller, the internal DMA registers of the 68450 DMA Controller, and Configuration Registers of the GPIB-1014D as well as summary tables for easy reference.
- Chapter 5, *Programming Considerations*, explains important considerations for programming the GPIB-1014D.
- Chapter 6, *Theory of Operation*, contains a functional overview of the GPIB-1014D board and explains the operation of each functional block making up the GPIB-1014D.
- Chapter 7, *Diagnostic and Troubleshooting Test Procedures*, contains test procedures for determining if the GPIB-1014D is installed and operating correctly.
- Appendix A, *Specifications*, lists the specifications of the GPIB-1014D.
- Appendix B, *PAL Drawings, Parts List, and Schematic Diagrams*, contains the PAL drawings, a parts list, and detailed schematic diagrams of the GPIB-1014D.
- Appendix C, *Sample Programs*, contains sample programs in 68000 Assembly Language code for implementing the most commonly used GPIB functions. Line-by-line comments provide an explanation of each function.
- Appendix D, *Multiline Interface Messages*, contains a listing of the multiline GPIB interface messages.
- Appendix E, *Operation of the GPIB*, describes the operation of the GPIB.

- Appendix F, *Mnemonics Key*, contains an alphabetical listing of all mnemonics used in this manual and indicates whether the mnemonic represents a bit, register, function, remote message, local message or a state.
- Appendix G, *Customer Communication*, contains forms you can use to request help from National Instruments or to comment on our products and manuals.
- The *Index* contains an alphabetical list of key terms and topics in this manual, including the page where you can find each one.

Conventions Used in This Manual

italic Italic text denotes emphasis, a cross reference, or an introduction to a key concept.

Abbreviations

The following are the metric system prefixes for use with abbreviations for units of measure:

Prefix	Meaning	Value
n	*nano*	10^{-9}
μ	*micro*	10^{-6}
m	*milli*	10^{-3}
k	*kilo*	10^{3}
M	*mega*	10^{6}

The following are the accepted abbreviations for units of measure used in the text of this manual.

A	ampere
C	Celcius
°	degree
hex	hexadecimal
Hz	hertz
in.	inch
K	kilobyte (1,024 bytes)
kbytes	1,000 bytes
m	meter
%	percent
sec	second
V	volt

Acronyms

VDC	volts direct current

Related Documents

The following manuals contain information that you may find helpful as you read this manual:

- ANSI/IEEE Std. 488-1978, *IEEE Standard Digital Interface for Programmable Instrumentation*.
- ANSI/IEEE Std. 1014-1987, *IEEE Standard for a Versatile Backplane Bus: VMEbus*.
- *µPD7210 GPIB-IFC User Manual*, NEC Electronics U.S.A., Inc..
- *µPD7210 Intelligent GPIB Interface Controller* Engineering Data Sheet, NEC Electronics U.S.A., Inc., Microcomputer Division.
- *How to Interface a Microcomputer System to a GPIB*, (& The NEC µPD7210 TLC), NEC Electronics U.S.A., Inc.
- *Motorola Semiconductor Technical Data MC68450 Advance Information Direct Memory Access Controller (DMAC)*, Motorola, Inc.
- *Hitachi Microcomputer System HD68450 DMAC (Direct Memory Access Controller).*

Customer Communication

National Instruments wants to receive your comments on our products and manuals. We are interested in the applications you develop with our products, and we want to help if you have problems with them. To make it easy for you to contact us, this manual contains comment and configuration forms for you to complete. These forms are in Appendix G, *Customer Communication*, at the end of this manual.

Chapter 1
Introduction

The GPIB-1014D is a high-performance IEEE-488 interface for the VMEbus. By using the GPIB-1014D, a VMEbus-based computer is able to control IEEE-488 compatible engineering, scientific, or medical instruments. The GPIB-1014D has the following features:

- Two independent IEEE-488 interfaces
- Complete IEEE-488 Talker/Listener/Controller (TLC) capability for each IEEE-488 interface using two NEC µPD7210 GPIB TLC chips
- Ability to monitor the status of all GPIB control lines for each interface (necessary for IEEE-488.2 compliance)
- DMA transfers
  - Data rates up to 500 kbytes/sec
  - Unlimited data block lengths
  - Full 32-bit addressing
  - Automatic carry cycle operation
  - GPIB synchronization detection
  - Release On Request capability
  - General purpose DMA capability
- Complete software control through programmable configuration parameters
  - One out of four Bus Request/Grant lines
  - One out of seven Interrupt Request lines
  - Supervisor or User access
  - Red/Green SYSFAIL LED indicator
  - Local Master Reset
- IEEE-1014 (VMEbus) standard compliance
- Complete software compatibility with the GPIB-1014

The GPIB-1014D conforms to all requirements and conventions specified in the ANSI/IEEE Std. 1014-1987. Hereafter, the General Purpose Interface Bus is referred to as the GPIB, the GPIB standard is referred to as the IEEE-488 standard, and the ANSI/IEEE Std. 1014-1987 is referred to as the IEEE-1014 standard.

Figure 1-1 shows the GPIB-1014D interface board.

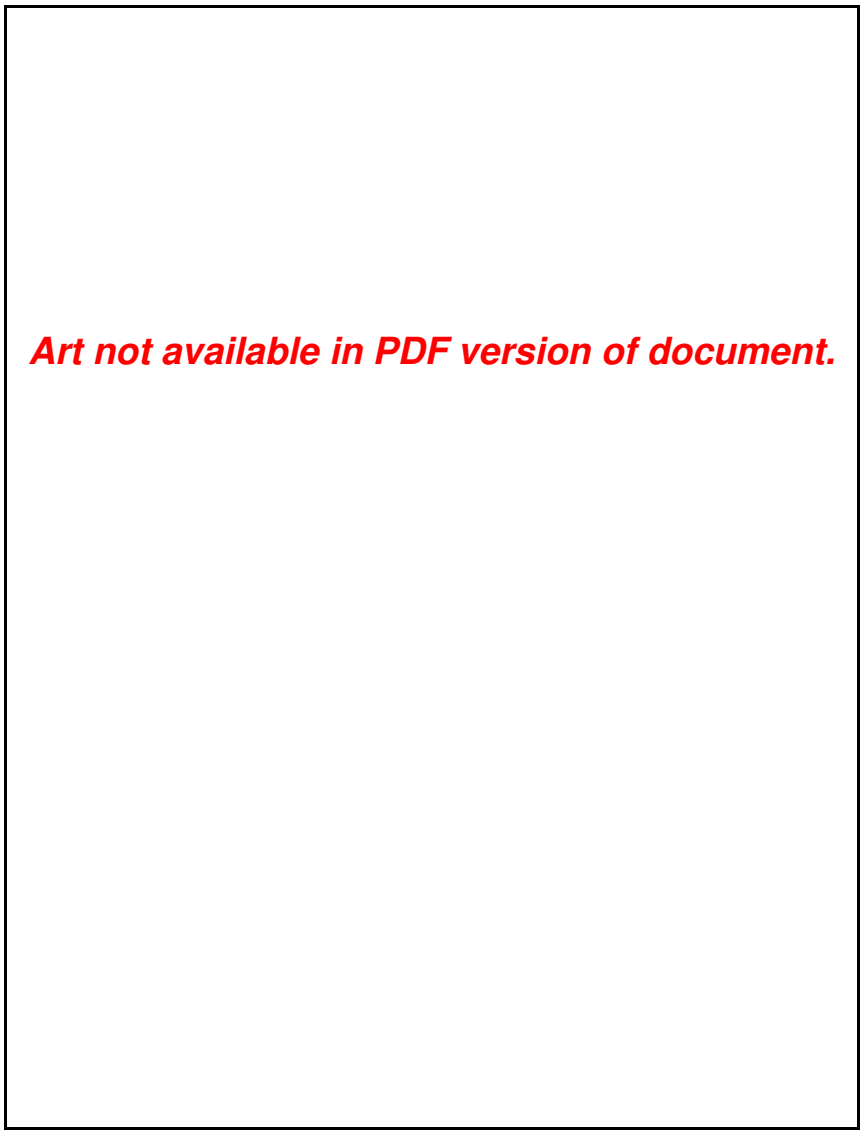


Figure 1-1. GPIB-1014D Interface Board

The GPIB-1014D interface kit includes hardware and programming examples to implement the GPIB functions. Optional cables are supplied for interconnection with other devices on the GPIB.

What Your Kit Should Contain

Your GPIB-1014D kit contains one of the following boards:

- GPIB-1014D-1
- GPIB-1014D-1S

The GPIB-1014D-1S board contains no front panel.

Optional Equipment

You can contact National Instruments to order the following optional equipment:

- Single-shielded Type X1 GPIB cables (1 m, 2 m, or 4 m)
- Double-shielded Type X2 GPIB cables (1 m, 2 m, or 4 m)
- GPIB Monitor/Analyzer
  - GPIB-400
  - GPIB-410

Unpacking

Follow these steps when unpacking your GPIB-1014D.

1. Your GPIB-1014D board is shipped packaged in an antistatic plastic bag to prevent electrostatic damage to the board. Several components on the board can be damaged by electrostatic discharge. To avoid such damage in handling the board, touch the plastic bag to a metal part of your VMEbus computer chassis before removing the board from the bag.

2. Remove the board from the bag and inspect the board for loose components or any other sign of damage. Notify National Instruments if the board appears damaged in any way. *Do not* install a damaged board into your computer.

Chapter 2
General Description

This chapter contains the physical and electrical specifications for the GPIB-1014D and describes the characteristics of key interface board components.

Electrical Characteristics

All integrated circuit drivers and receivers used on the GPIB-1014D meet the requirements of the VMEbus Specification and the IEEE-1014 standard. Table 2-1 contains a list of the VMEbus signals used by the GPIB-1014D and the electrical loading presented by the circuitry on the interface board (in terms of device types and their part numbers).

Note: The asterisk (*) after the bus signal indicates that the signal is active low.

Table 2-1. GPIB-1014D Signals

Bus Signals	Driver Device Part Number	Receiver Device Part Number
D00-D15	F245	F245
A15-A10	AS573	–
A9	AS573	F125
A8	AS573	AS373
A07-A01	F245	F245
AM5-AM3, AM0	F244	LS2521
AM2	F244	F32
DS0*, DS1*, WRITE*	F1243	F1243
AS*	68172 (BUSCON)	68172 (BUSCON)
LWORD*	F244	LS2521
IACK*	F244	LS240
SYSCLK	–	LS241
BG0IN*-BG3IN*	–	LS241
BG0OUT*-BG3OUT*	F04	–

(continues)

Table 2-1. GPIB-1014D Signals (continued)

Bus Signals	Driver Device Part Number	Receiver Device Part Number
DTACK*	F38	F125
BR0*-BR3*	F38	PAL22PIO
BBSY*	68172 (BUSCON)	68172 (BUSCON)
IACKIN*	–	LS241
IACKOUT*	F32	–
IRQ1*-IRQ7*	74145	–
BERR*	F38	F125
SYSFAIL*	F38	–
SYSRESET*	–	LS241

All GPIB transceivers meet the requirements of the IEEE-488 standard. The components used are as follows:

Transceivers	Component Designation
Data Transceivers	DS75160AN
Control Transceivers	DS75162AN

Note: The GPIB-1014D requires regulated +5 VDC power from the VMEbus. Current load is typically 2.5 A (2.9 A maximum).

VMEbus Characteristics

The following paragraphs describe each of the VMEbus modules on the GPIB-1014D: slave, master, interrupter, and requester. Table 2-5 at the end of this chapter summarizes the capabilities of these modules.

VMEbus Slave-Addressing

The GPIB-1014D occupies 1,024 bytes (512 words) in the A16 (short) I/O space. As a VMEbus slave, it only responds when the address modifier (AM) lines specify a short supervisory access (AM code = 2D) or short nonpriviledged access (AM code = 29). The board responds to short 16-bit addresses.

The GPIB-1014D is divided into two ports: Port A and Port B. Each port has an identical set of registers. The GPIB-1014D first compares address lines A15 through A10 with its base address to generate its board select signal. It then decodes address line A9 in determining to which port it will talk and decodes the lowest eight address lines A8 through A1 to address the following components and registers.

- The 68450 DMA Controller (DMAC)
- The µPD7210 GPIB Talker/Listener/Controllers (TLC) (one for each port)
- Two 8-bit, write-only Configuration Registers (two for each port)
- One GPIB Status Register (one for each port)
- One Page Register

The base address of the board is configured through the switch set U82 on the interface board. See Chapter 3, *Configuration and Installation*, for instructions on how to set the base address of the board.

VMEbus Slave-Data

The GPIB-1014D can function as a VMEbus slave, decoding short I/O addresses and commands from a VMEbus master. The µPD7210 TLCs, the Configuration Registers, the GPIB Status Registers, and the Page Register function as 8-bit slaves, allowing data to be transferred to and from the VMEbus master on data lines D07 through D00. The 68450 can function as an 8- or 16-bit slave, allowing 8- or 16-bit transfers on data lines D15 through D00. The board is designed to accommodate Address Only (ADO) cycles. In VMEbus terminology, the slave module of the board is designated as SADO16 / D16 and D08(EO).

Ports A and B of the GPIB-1014D have an identical set of registers that consist of the GPIB Interface Registers, two Configuration Registers, and one GPIB Status Register. The Page Register, along with some bits of the Configuration Registers, are shared by both ports, but can be written in the address space of either port. The ports were designed so that the base address of Port A is at the base address of the board, and the base address of Port B is at the base address of the board + 200 (hex). Thus, if the Data In Register (DIR) of Port A is located at base address + 111 (hex), then the DIR of Port B is located at base address + 311 (hex).

The GPIB Interface Registers associated with the µPD7210s are addressed relative to the base address of the GPIB-1014D as shown in Table 2-2. The DMA registers internal to the 68450 are shown in Table 2-3. The Configuration Registers, the GPIB Status Registers, and the Page Register of the GPIB-1014D are shown in Table 2-4.

Table 2-2. µPD7210 Internal GPIB Interface Registers (Port A and Port B)

Address (Base+ hex offset) Port A	Port B	Mode	Register		Size
111	311	R	Data In Register	(DIR)	8 bits
111	311	W	Control/Data Out Register	(CDOR)	8 bits
113	313	R	Interrupt Status Register 1	(ISR1)	8 bits
113	313	W	Interrupt Mask Register 1	(IMR1)	8 bits
115	315	R	Interrupt Status Register 2	(ISR2)	8 bits
115	315	W	Interrupt Mask Register 2	(IMR2)	8 bits
117	317	R	Serial Poll Status Register	(SPSR)	8 bits
117	317	W	Serial Poll Mode Register	(SPMR)	8 bits
119	319	R	Address Status Register	(ADSR)	8 bits
119	319	W	Address Mode Register	(ADMR)	8 bits
11B	31B	R	Command Pass Through Register	(CPTR)	8 bits
11B	31B	W	Auxiliary Mode Register	(AUXMR)	8 bits
11D	31D	R	Address Register 0	(ADR0)	8 bits
11D	31D	W	Address Register	(ADR)	8 bits
11F	31F	R	Address Register 1	(ADR1)	8 bits
11F	31F	W	End Of String Register	(EOSR)	8 bits

Table 2-3. 68450 Internal DMA Registers

Address (Base + hex offset)	Mode	Register		Channel	Size
0A	R/W	Memory Transfer Counter Register	(MTCR0)	0	16 bits
0C	R/W	Memory Address Register	(MAR0)	0	32 bits
29	R/W	Memory Function Code Register	(MFCR0)	0	8 bits
14	R/W	Device Address Register	(DAR0)	0	32 bits
31	R/W	Device Function Code Register	(DFCR0)	0	8 bits
1A	R/W	Base Transfer Counter Register	(BTCR0)	0	16 bits
1C	R/W	Base Address Register	(BAR0)	0	32 bits
39	R/W	Base Function Code Register	(BFCR0)	0	8 bits
00	R/W	Channel Status Register	(CSR0)	0	8 bits
01	R	Channel Error Register	(CER0)	0	8 bits
04	R/W	Device Control Register	(DCR0)	0	8 bits
05	R/W	Operation Control Register	(OCR0)	0	8 bits
06	R/W	Sequence Control Register	(SCR0)	0	8 bits
07	R/W	Channel Control Register	(CCR0)	0	8 bits
2D	R/W	Channel Priority Register	(CPR0)	0	8 bits
25	R/W	Normal Interrupt Vector Register	(NIVR0)	0	8 bits
27	R/W	Error Interrupt Vector Register	(EIVR0)	0	8 bits

(continues)

Table 2-3. 68450 Internal DMA Registers (continues)

Address (Base + hex offset)	Mode	Register	Channel	Size
4A	R/W	Memory Transfer Counter Register (MTCR1)	1	16 bits
4C	R/W	Memory Address Register (MAR1)	1	32 bits
69	R/W	Memory Function Code Register (MFCR1)	1	8 bits
54	R/W	Device Address Register (DAR1)	1	32 bits
71	R/W	Device Function Code Register (DFCR1)	1	8 bits
5A	R/W	Base Transfer Counter Register (BTCR1)	1	16 bit
5C	R/W	Base Address Register (BAR1)	1	32 bits
79	R/W	Base Function Code Register (BFCR1)	1	8 bits
40	R/W	Channel Status Register (CSR1)	1	8 bits
41	R	Channel Error Register (CER1)	1	8 bits
44	R/W	Device Control Register (DCR1)	1	8 bits
45	R/W	Operation Control Register (OCR1)	1	8 bits
46	R/W	Sequence Control Register (SCR1)	1	8 bits
47	R/W	Channel Control Register (CCR1)	1	8 bits
6D	R/W	Channel Priority Register (CPR1)	1	8 bits
65	R/W	Normal Interrupt Vector Register (NIVR1)	1	8 bits
67	R/W	Error Interrupt Vector Register (EIVR1)	1	8 bits
8A	R/W	Memory Transfer Counter Register (MTCR2)	2	16 bits
8C	R/W	Memory Address Register (MAR2)	2	32 bits
A9	R/W	Memory Function Code Register (MFCR2)	2	8 bits
94	R/W	Device Address Register (DAR2)	2	32 bits
B1	R/W	Device Function Code Register (DFCR2)	2	8 bits
9A	R/W	Base Transfer Counter Register (BTCR2)	2	16 bits
9C	R/W	Base Address Register (BAR2)	2	32 bits
B9	R/W	Base Function Code Register (BFCR2)	2	8 bits
80	R/W	Channel Status Register (CSR2)	2	8 bits
81	R	Channel Error Register (CER2)	2	8 bits
84	R/W	Device Control Register (DCR2)	2	8 bits
85	R/W	Operation Control Register (OCR2)	2	8 bits
86	R/W	Sequence Control Register (SCR2)	2	8 bits
87	R/W	Channel Control Register (CCR2)	2	8 bits
AD	R/W	Channel Priority Register (CPR2)	2	8 bits
A5	R/W	Normal Interrupt Vector Register (NIVR2)	2	8 bits
A7	R/W	Error Interrupt Vector Register (EIVR2)	2	8 bits

(continues)

Table 2-3. 68450 Internal DMA Registers (continued)

Address (Base + hex offset)	Mode	Register		Channel	Size
CA	R/W	Memory Transfer Counter Register	(MTCR3)	3	16 bits
CC	R/W	Memory Address Register	(MAR3)	3	32 bits
E9	R/W	Memory Function Code Register	(MFCR3)	3	8 bits
D4	R/W	Device Address Register	(DAR3)	3	32 bits
F1	R/W	Device Function Code Register	(DFCR3)	3	8 bits
DA	R/W	Base Transfer Counter Register	(BTCR3)	3	16 bits
DC	R/W	Base Address Register	(BAR3)	3	32 bits
F9	R/W	Base Function Code Register	(BFCR3)	3	8 bits
C0	R/W	Channel Status Register	(CSR3)	3	8 bits
C1	R	Channel Error Register	(CER3)	3	8 bits
C4	R/W	Device Control Register	(DCR3)	3	8 bits
C5	R/W	Operation Control Register	(OCR3)	3	8 bits
C6	R/W	Sequence Control Register	(SCR3)	3	8 bits
C7	R/W	Channel Control Register	(CCR3)	3	8 bits
ED	R/W	Channel Priority Register	(CPR3)	3	8 bits
E5	R/W	Normal Interrupt Vector Register	(NIVR3)	3	8 bits
E7	R/W	Error Interrupt Vector Register	(EIVR3)	3	8 bits
FF	R/W	General Control Register	(GCR)	all	8 bits

Table 2-4. Configuration, Page, and GPIB Status Registers of Port A and Port B

Address (Base + hex offset) Port A	Port B	Mode	Register		Size
101	301	W	Configuration Register 1	(CFG1)	8 bits
105	305	W	Configuration Register 2	(CFG2)	8 bits
105	305	R	GPIB Status Register	(GSR)	8 bits
109	309	W	Page Register	(PREG)	8 bits

VMEbus Master-Direct Memory Access

The GPIB-1014D can function as a VMEbus master, performing data transfers to and from VMEbus memory. In most applications, the 68450 controls the data transfer to and from the GPIB during DMA, and can transfer the 8-bit data on data lines D07 through D00 or D15 through D08, allowing packing of data in VMEbus memory. In addition to GPIB-to-VMEbus memory DMA transfers, the GPIB-1014D can also perform 8- or 16-bit memory-to-memory DMA transfers, if the DMA channels are not being used for GPIB transfers.

Memory addresses generated by the GPIB-1014D are 32-bits wide (A31 through A01, DS0* and DS1*) and the VMEbus Address Modifier Lines (AM5 through AM0) are fully programmable

using function code registers located in the 68450 and three hardware jumpers (W5, W6, and W8). (See Chapter 3 for instructions on how to set the hardware jumpers. See Chapter 4 for a description of the DMAC Function Code Registers.) The 32-bit addresses, along with selectable Address Modifier codes, eliminate artificial memory boundaries and allow data transfers between the GPIB and data area, program area, or even devices located in the short I/O area. In VMEbus terminology, the GPIB-1014D has an A32 / D08(EO) and D16 master capability. The board does not implement Unaligned Transfer (UAT), Block Transfer (BLT), and Read Modify Write (RMW) cycles. The chaining feature of the 68450 allows data blocks of unlimited size to be transferred.

Interrupter

Interrupt events that can drive a hardware-programmed VMEbus interrupt request line are as follows:

- GPIB Data In (DI)
- GPIB Data Out (DO)
- END message received (END RX)
- GPIB Command Out (CO)
- Remote Mode Change (REMC)
- GPIB Handshake Error (ERR)
- Lockout Change (LOKC)
- GPIB DMA Transfer Finished and GPIB Synchronized (FIN)
- Address Status Change (ADSC)
- Secondary Address Pass Through (APT)
- Service Request Input (SRQI)
- Device Execute Trigger (DET)
- Device Clear Received (DEC RX)
- Command Pass Through (CPT)
- Bus Error (BERR)

You can select one of seven VMEbus interrupt request lines (IRQ1* through IRQ7*) through software using three bits located in Configuration Register 1 Port A or Configuration Register 1 Port B (CFG1A or CFG1B). Because Port A and Port B share configuration bits, both ports use the same interrupt request line.

The onboard hardware implements the VMEbus interrupt acknowledge protocol. Interrupt Vector Registers located in the 68450 let you select, through software, the 8-bit Interrupt Status/ID byte supplied by the GPIB-1014D during an interrupt acknowledge cycle of the correct priority. The GPIB-1014D is a D08(O) interrupter because it responds to an interrupt acknowledge cycle by providing an 8-bit status/ID byte on data lines D00 through D07. In addition, the board is a Release On Register Access (RORA) Interrupter since it releases its interrupt line when the MASTER accesses its an onboard register. In VMEbus terminology, the GPIB-1014D has a D08(O) / RORA Interrupter.

Data Transfer Bus (DTB) Requester

The GPIB-1014D arbitrates for the DTB before each DMA transfer. The board is designed for you to select, through software, one of four VMEbus request lines (BR0* through BR3*) using two bits in Configuration Register 1 (CFG1A or CFG1B). The onboard 68172 Bus Controller

(BUSCON) performs the necessary VMEbus protocol to request, obtain, and release control of the VMEbus. Circuitry on the board passes unused BGxIN* signals down the BGxOUT* lines. To maximize the use of the DTB, the board can be programmed to become a Release On Request (ROR) DTB Requester. Unless programmed, the GPIB-1014D is a Release When Done (RWD) Requester.

VMEbus Modules Not Provided

Because the GPIB-1014D is not designed to be VMEbus System Controller, it does not have the following modules:

- Bus Timer
- Arbiter
- Interrupt Handler
- IACK Daisy-Chain Driver
- System Clock Driver
- Serial Clock Driver
- Power Monitor

Diagnostic Aids

The GPIB-1014D is designed to permit stand-alone verification of I/O and DMA functions. See Chapter 7, *Diagnostic and Troubleshooting Test Procedures*, for details.

Data Transfer Features

The GPIB-1014D can be used to transfer data to and from the GPIB using Direct Memory Access (DMA) and programmed I/O. The rates given in the following paragraphs are block transfer rates and do not necessarily indicate the overall throughput rate for the particular GPIB system. The overall throughput rate is dependent on the factors in the following list as well as the time limits given in the paragraph after this list.

- The number of GPIB commands sent
- The amount of time spent setting up the GPIB-1014D DMA transfers
- The size of the DMA data buffers (number of bytes transferred in one DMA operation)
- Operating system overhead
- Interrupt service time

Transfer rates of 250 kbytes/sec to 350 kbytes/sec can be expected in typical systems. Rates up to 500 kbytes/sec can be achieved under optimum conditions listed in the following paragraphs.

DMA Transfers from VMEbus Memory to the GPIB

In most systems, the response time of the Listeners on the GPIB is the limiting factor for transfer speed but, under the following set of conditions, the GPIB-1014D can run at speeds of 500 kbytes/sec:

- VMEbus memory read response time (DS* low to DTACK* low): 150 nsec or less
- GPIB Listener response time (DAV* low to NDAC* high): 50 nsec or less
- GPIB-1014D transfer mode: cycle steal with hold, programmable timeout
- T1 timing: high speed, ICR = 5

DMA Transfers from the GPIB to VMEbus Memory

Under the following set of conditions, the GPIB-1014D can read data from a GPIB Talker and store it in VMEbus memory at speeds of 500 kbytes/sec:

- VMEbus memory write response time (DS* low to DTACK* low): 150 nsec or less
- GPIB Talker response time (NRFD* high to DAV* low): 350 nsec or less
- GPIB-1014D transfer mode: cycle steal with hold, programmable timeout

Programmed I/O Transfers

The GPIB-1014D is able to transfer data to and from the GPIB using programmed I/O. Transfer rates under programmed I/O depend on many factors, including how fast the program code executes, how fast the microprocessor services interrupts, and operating system overhead. Typically, the GPIB-1014D transfers data at rates ranging from 10 kbytes/sec to 80 kbytes/sec using programmed I/O.

GPIB-1014D Functional Description

In the simplest terms, the GPIB-1014D can be thought of as a bus translator, converting messages and signals present on the VMEbus into appropriate GPIB messages and signals. Expressed in GPIB terms, the GPIB-1014D implements GPIB interface functions for communicating with other GPIB devices and device functions for communicating with the central processor and memory. Expressed in VMEbus terms, the GPIB-1014D is an interface to the outside world.

Figures 2-1 and 2-2 show typical applications for the GPIB-1014D. In Figure 2-1, the GPIB-1014D is used to interface an assortment of test instruments to a VMEbus computer system, which then functions as an intelligent System Controller. This is the traditional role of the GPIB.

In Figure 2-2, the GPIB-1014D is used along with other National Instruments interface boards to connect a VMEbus computer to other processors to transfer files electrically rather than manually (via a removable storage medium) or to perform other interprocessor communication functions.

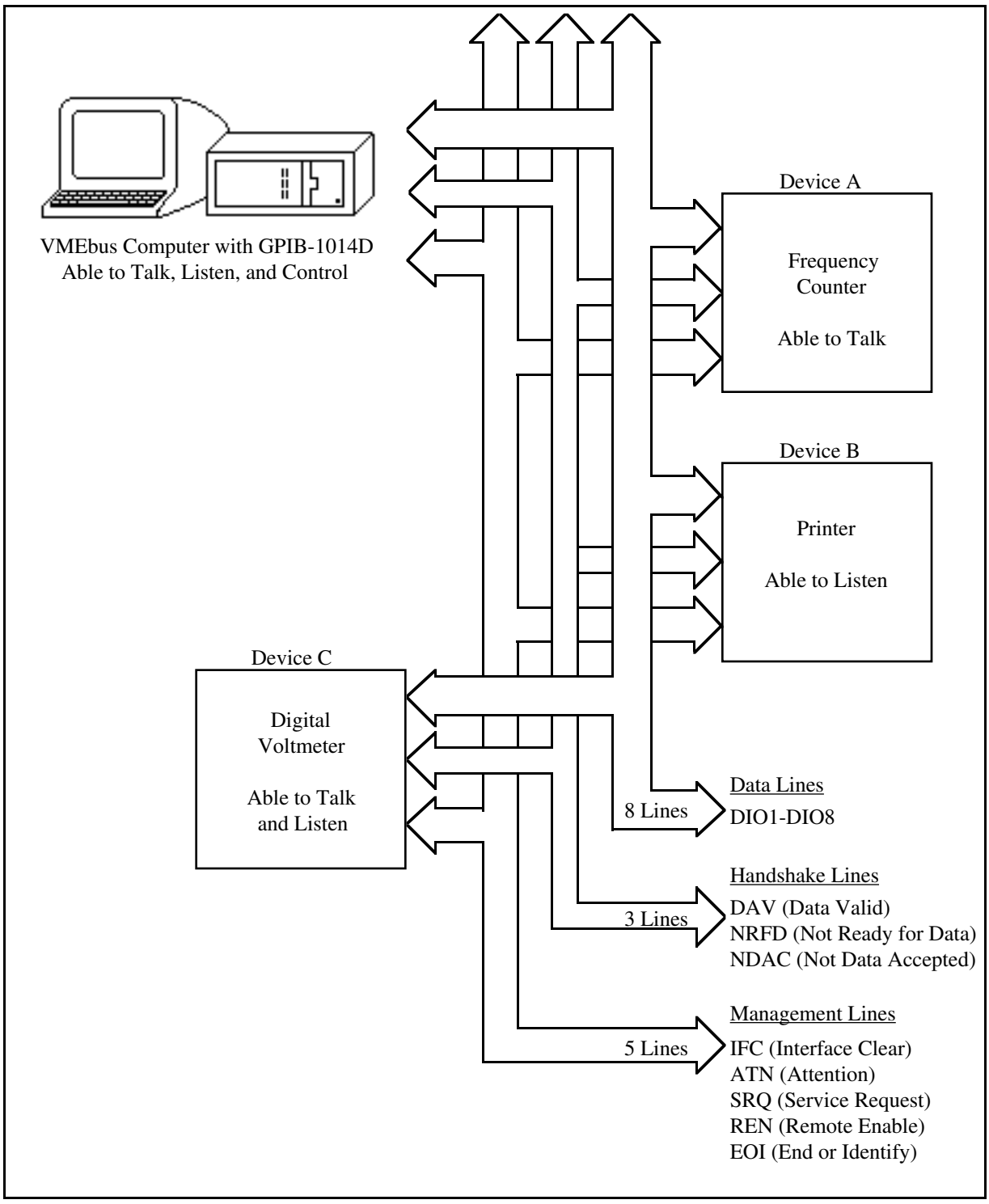


Figure 2-1. GPIB-1014D with a VMEbus Computer

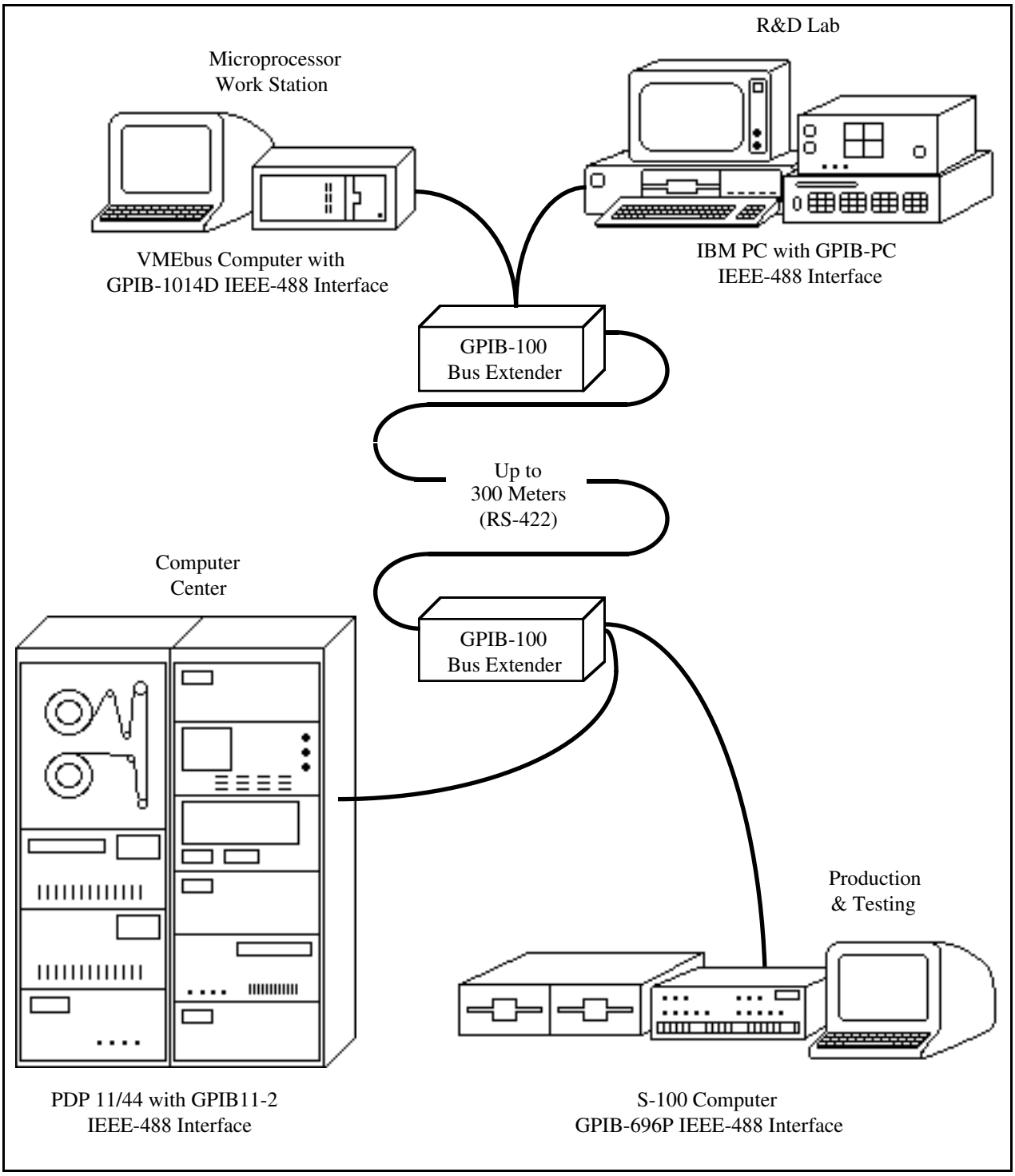


Figure 2-2. GPIB-1014D in a Multiprocessor Application

Figure 2-3 is a block diagram of the GPIB-1014D.

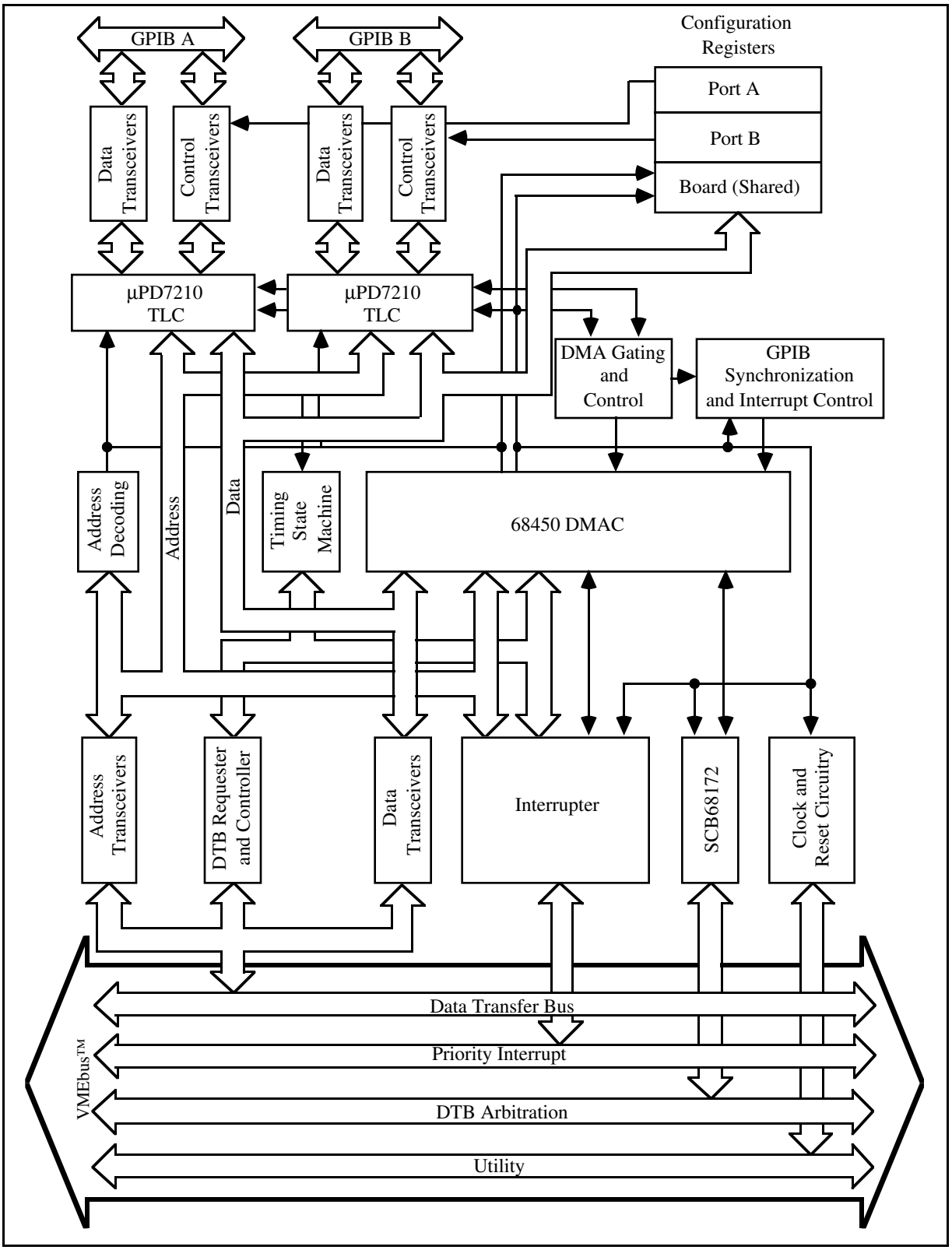


Figure 2-3. GPIB-1014D Block Diagram

The interface consists of these major components which are discussed in greater detail in Chapter 6.

- VMEbus Interface — Consists of the buffers, drivers, and transceivers for the address, data, status, and control lines used on the VMEbus. Also consists of logic circuitry that converts internal signals to bus-compatible signals.
- Address Decoder — Recognizes when the VMEbus master addresses one of the GPIB-1014D registers and generates the appropriate strobe to cause the data transfer.
- Clock and Reset Circuitry — Monitors the VMEbus utility signals to generate the 8 MHz clock used by the TLC and to detect System Reset and Bus Error conditions.
- Configuration Registers — Programmably configures some of the operating parameters of the GPIB-1014D.
- Timing State Machine — Controls the timing of DMA transfers and accesses to the GPIB-1014D from the VMEbus.
- DMA Gating and Control — Controls the DMA request/acknowledge interface between the DMAC and the two TLCs.
- Interrupter — Implements the VMEbus priority interrupt protocol, allowing the GPIB-1014D to request and respond to an interrupt acknowledge cycle. All interrupt conditions are also detectable by polling.
- DTB Requester and Controller — The circuitry consists of a 68172 BUSCON that performs the necessary VMEbus protocol to request, obtain, and release control of the VME system bus. Once configured for a DMA transfer, the GPIB-1014D automatically performs data transfers between the GPIB and VMEbus memory.
- GPIB Synchronization and Interrupt Control — Detects the synchronization of the GPIB after the last byte in a DMA transfer (all devices on the GPIB have accepted the last byte) and detects interrupting conditions from the TLC. TLC interrupt requests are routed through the DMAC, which notifies that the VMEbus system interrupt handler when either a TLC interrupt or one of its own internal interrupt conditions is detected.

- DMAC (68450) — Controls DMA transfers between the GPIB and the VMEbus. The DMA Gating and Control circuitry controls the DMA request/acknowledge interface between the TLC and the DMAC.
- GPIB TLC (NEC µPD7210) — Implements many of the GPIB interface functions, either independently or with assistance of or interpretation by the controlling program. Together with special transceivers, the TLC forms the GPIB interface side of the GPIB-1014D.

Table 2-5 lists the capabilities of the GPIB-1014D in terms of the IEEE-488 standard codes

Table 2-5. GPIB-1014D IEEE-488 Interface Capabilities

Capability Code	Description
SH1	Complete Source Handshake capability
AH1	Complete Acceptor Handshake capability DAC and RFD Holdoff on certain events
T5	Complete Talker capability Basic Talker Serial Poll Talk Only mode Unaddressed on MLA Send END or EOS Dual primary addressing
TE5	Complete Extended Talker capability Basic Extended Talker Serial Poll Talk Only mode Unaddressed on MSA*LPAS Send END or EOS Dual primary addressing
L3	Complete Listener capability Basic Listener Listen Only mode Unaddressed on MTA Detect END or EOS Dual extended addressing with software assist
LE3 (continues)	Complete Extended Listener capability Basic Listener Listen Only mode Unaddressed on MSA*TPAS

(continues)

Table 2-5. GPIB-1014D IEEE-488 Interface Capabilities (continued)

Capability Code	Description
LE3 (continued)	Complete Extended Listener capability (continued) Detect END or EOS Dual extended addressing with software assist
SR1	Complete Service Request capability
RL1	Complete Remote/Local capability with software interpretation
PP1	Remote Parallel Poll configuration
PP2	Local Parallel Poll configuration with software assist
DC1	Complete Device Clear capability with software interpretation
DT1	Complete Device Trigger capability with software interpretation
C1 through C5	Complete Controller capability System Controller Send IFC and take charge Send REN Respond to SRQ Send interface messages Receive control Pass control Parallel Poll Take control synchronously or asynchronously
E1, E2	Tri-state bus drivers with automatic switch to open Collector drivers during Parallel Poll

The GPIB-1014D has complete Source and Acceptor Handshake capability.

- The GPIB-1014D can operate as a basic Talker or Extended Talker and can respond to a Serial Poll. It can be placed in a Talk Only mode, and is unaddressed to talk when it receives its listen address.

- The interface can operate as a basic Listener or Extended Listener. It can be placed in a Listen Only mode, and is unaddressed to listen when it receives its talk address.

The GPIB-1014D has full capabilities for requesting service from another Controller. The ability to place the GPIB-1014D in local mode is included, but the interpretation of remote versus local mode is software-dependent. Full Parallel Poll capability is included in the interface although local configuration requires software assistance. Device Clear and Trigger capability is included in the interface, but the interpretation is software-dependent. All Controller functions as

specified by the IEEE-488 standard are included in the GPIB-1014D. These include the ability to do the following tasks:

- Be System Controller
- Initialize the interface
- Send Remote Enable
- Respond to Service Request
- Send multiline command messages
- Receive control
- Pass control
- Conduct a Parallel Poll
- Take control synchronously or asynchronously

Table 2-6 contains the GPIB-1014D IEEE-1014 compliance levels.

Table 2-6. GPIB-1014D IEEE-1014 Compliance Levels

Compliance Notation	Description
Bus Slave Compliance Levels	
D8(O)	8-bit data path to the TLCs, Configuration Registers, Page Registers, and GPIB Status Registers.
D16 and D8(EO)	8- or 16-bit data path to DMAC registers.
A16	Responds to 16-bit short I/O addresses when specified on the address modifier lines.
ADO	Accommodate Address Only cycles.
Interrupter Compliance Levels	
D8(O)	Provides an 8-bit status/ID byte on D00 through D07.
RORA	Releases its interrupt request line when its onboard register is accessed by a master.
Bus Master Compliance Levels	
D8(EO)	8-bit data path for GPIB DMA transfers.
D16 and D8(EO)	8- or 16-bit memory-to-memory DMA transfers.
A32	32-bit memory address path.
DTB Requester Compliance Level	
ROR	Programmable Release on Request Feature.

Chapter 3
Configuration and Installation

This chapter describes the configuration and installation of the GPIB-1014D.

Configuration

Before installing the GPIB-1014D in the VMEbus backplane, the following options must be configured with hardware switches that are located on the GPIB-1014D interface board:

- Access mode (switch W4)
- Base address (switch block U82)
- DMA Address Modifier (AM) code output (switches W5, W6, and W8)
- Extended (32-bit) Addressing Enable (switch W9)
- DMAC Channel 2 Interrupt Source (switch W3)
- VMEbus SYSFAIL* Driver Enable (switch W2)
- Board Reset Source (switch W7)

Figure 3-1 shows the locations of the GPIB-1014D configuration switches.

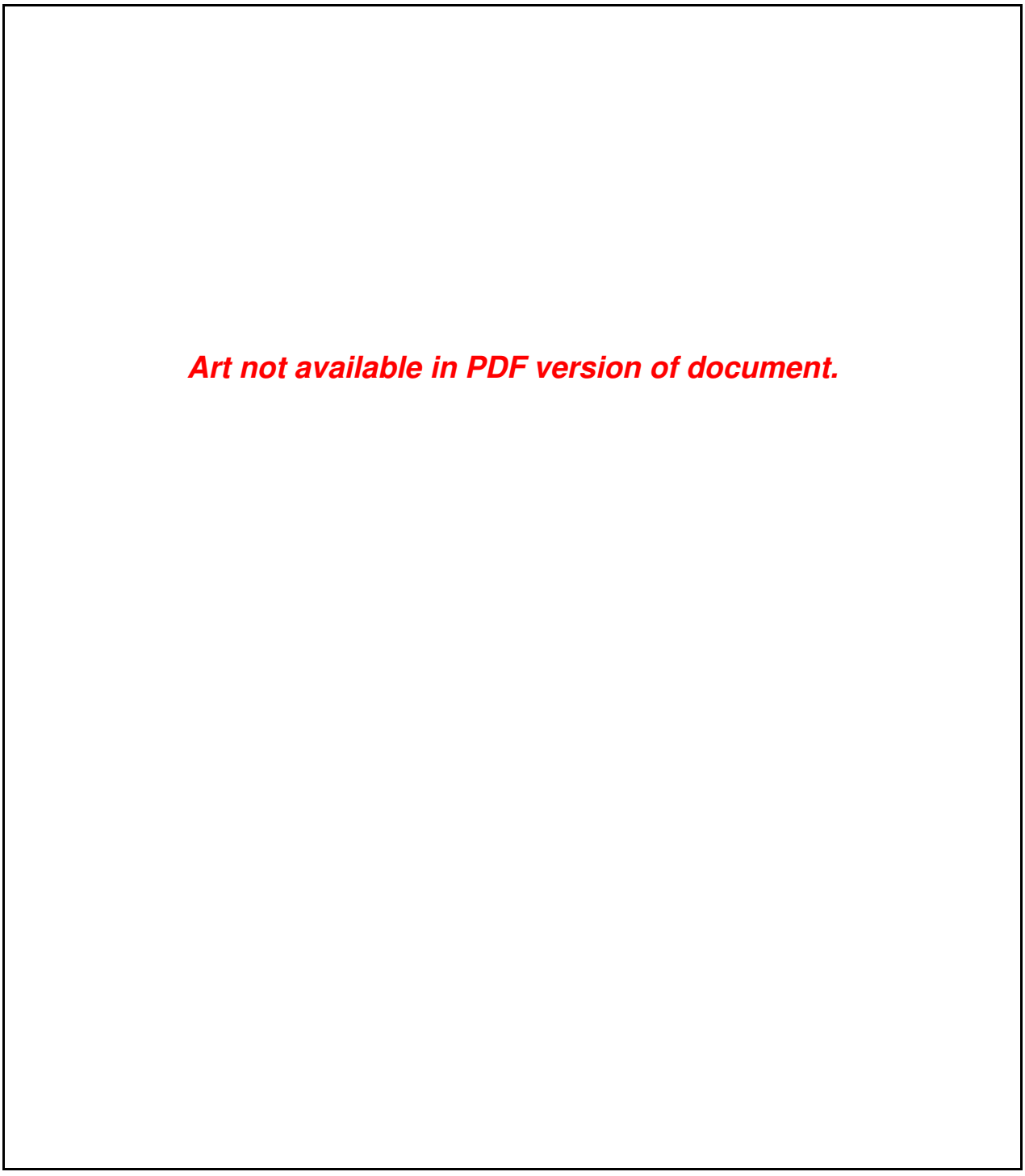


Figure 3-1. GPIB-1014D Parts Locator Diagram

Access Mode

The GPIB-1014D can be configured to respond to Supervisor (privileged) or User (non-privileged) access. Hardware switch W4 is used to select the access mode that is automatically in effect upon a power-up or a system reset. The access mode can then be changed by software via a bit in Configuration Register 2. Figure 3-2 shows the placement of the jumper for the desired mode after a power-up or a system reset. Move the switch to the side labeled *SUP* if you require Supervisor mode. Move the switch to the side labeled *USR* if you need User mode. The GPIB-1014D is configured at the factory for Supervisor access; that is, Supervisor mode is the default setting. If the board is configured for Supervisor mode, it initially responds to a 16-bit address and an AM code of 2D. If the board is configured for User mode, the board will initially respond to a 16-bit address and an AM code of 29 or 2D. (Refer to the ANSI/IEEE Std. 1014-1987, *IEEE Standard for a Versatile Backplane Bus: VMEbus* for more information on Supervisor and User modes.)

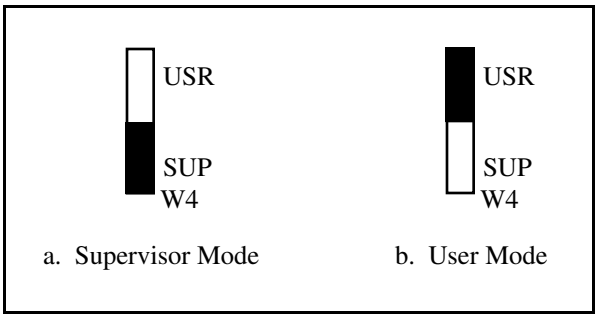


Figure 3-2. Access Mode After RESET

Base Address

The GPIB-1014D occupies a total of 1,014 bytes of 16-bit I/O space. The base address is selected with DIP switch U82 on the interface board. Press the side labeled *1* to select a logical one for the corresponding address bit. Press the side labeled *0* to select a logical zero for the corresponding address bit. Figure 3-3 shows the configuration for a base address 2000 (hex), which is the default address configured at the factory.

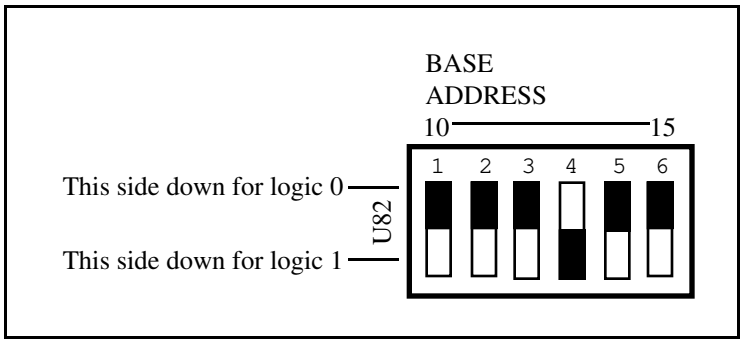


Figure 3-3. Configuration for GPIB-1014D Base Address 2000 (hex) (Default Setting)

DMA Address Modifier Code Output

During a DMA cycle, the GPIB-1014D sends out a 6-bit Address Modifier (AM) code to the VMEbus lines AM5 through AM0. The correct code is obtained by programming the DMAC and setting switches W5, W6, and W8. Figure 3-4 shows the default settings of W5, W6, and W8.

Note: Because switch W8 is not located near switches W5 and W6, Figure 3-4 outlines and labels the switches on the GPIB-1014D interface board that are near switch W8 to help you locate it.

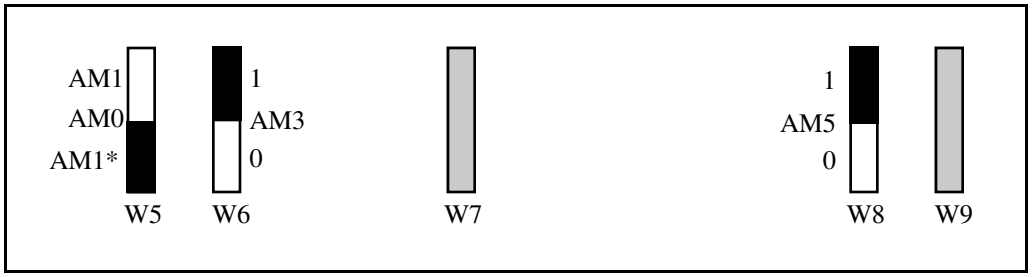


Figure 3-4. Default Settings of AM Code Switches W5, W6, and W8

The GPIB-1014D is compatible with software written for the GPIB-1014 and earlier versions of the board.

The GPIB-1014D can produce 12 AM codes from the default settings of W5, W6, and W8. These codes are most commonly used. Table 3-1 lists these codes next to the corresponding DMAC Function Code Register (FCR) values needed to produce the codes. (See Chapter 4 for a description of the DMAC FCR.)

Table 3-1. Programming Values for Default Settings of Jumpers W5, W6, and W8

AM Codes	FCR Bits M2 through M0
29	000
2A	001
2D	100
2E	101
39	010
3A	011

(continues)

Table 3-1. Programming Values for Default Settings of Jumpers W5, W6, and W8 (continued)

AM Codes	FCR Bits M2 through M0
3D	110
3E	111
09*	000
0A*	001
0D*	100
OE*	101

* If this code is used for extended (32-bit) addressing, the 32-bit addressing option must be enabled for the board. See *Select Extended (32-Bit) Addressing* later in this chapter.

If you do not want to use any of the default settings listed in Table 3-1 then you can produce any arbitrary AM code by first setting switch W9 to 24-bit addressing then changing the switches W5, W6, and W8 along with programming the DMAC.

Table 3-2 shows how each of the six AM code bits is effected by the switches and the values of the FCR bits of the DMAC.

Table 3-2. Setting the Address Modifier Code bits (AM5-AM0)

AM Bits	Controlled By	Value	Default
AM(5)	Switch W8	0 or 1	1
AM(4)	Bit M1 in FCR	0 or 1	None
AM(3)	Switch W6	0 or 1	1
AM(2)	Bit M2 in FCR	0 or 1	None
AM(1)	Bit M0 in FCR	0 or 1	None
AM(0)	Switch W5	AM(1) or AM(1)'	AM(1)'

For example, to produce an AM code of 17 hex (a binary value of 010111), you need to complete the following steps:

1. Set switch W8 to 0.
2. Set switch W6 to 0.
3. Set switch W5 to AM(1).
4. Program the FCR with M2=1, M1=1, and M0=1.

Note: When changing the default settings of W5, W6, and W8 to produce an arbitrary AM code, you must also set switch W9 to 24-bit addressing for Table 3-2 to be applicable.

Select Extended (32-Bit) Addressing

The GPIB-1014D is capable of providing 32-bit addressing to the VMEbus on DMA transfers. The 32-bit addressing feature is enabled by setting switch W9 (as shown in Figure 3-5) and setting the fourth bit (32BIT) in CFG2A or CFG2B.

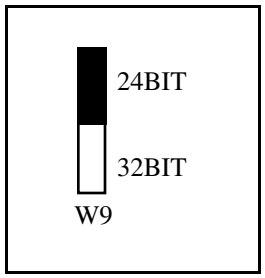


Figure 3-5. 32-Bit Addressing Switch

By placing the GPIB-1014D in 32-bit mode, the address modifier codes generated are changed to those defined by the IEEE-1014 specification for 32-bit addressing. For compatibility with GPIB-1014 software, this can be completely disabled by selecting the 24-bit position; AM codes will be generated as described previously.

Select Interrupt Source for DMAC Channel 2

The GPIB-1014D monitors either signal REN* of Port A's GPIB or signal SRQ* of Port B's GPIB using pin PCL2* of the DMAC. By setting switch W3, you can select the line to be monitored.

Note: Switch W3 is used to determine compatibility with the software written for the GPIB-1014 series. By using the GPIB-1014 series, you can monitor signal SRQ* on PCL0* and signal REN* on PCL2*.

Figure 3-6 shows switch W3.

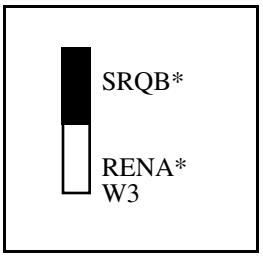


Figure 3-6. DMAC PCL2* Signal Selection

If you select RENA*, Port A of the GPIB-1014D will have complete software compatibility with the GPIB-1014 (which has only one Port). If SRQB* is selected, Port A and B of the GPIB-1014D will be functionally equivalent. For more information on how to port GPIB-1014 software to the GPIB-1014D, see *Porting GPIB-1014 Software to the GPIB-1014D* in Chapter 5.

VMEbus SYSFAIL* Driver Enable

Switch W2 determines whether the GPIB-1014D will drive its local SYSFAIL* message onto the VMEbus. Driving the local SYSFAIL* message onto the VMEbus is the proper response as defined by the specification, but is incompatible with some systems. Moving the switch to the side labeled 1 enables the SYSFAIL* driver (as shown in Figure 3-7), while switch position 0 disables the feature.

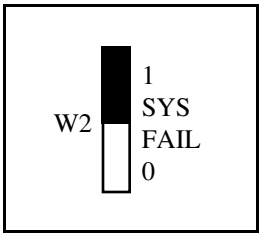


Figure 3-7. VMEbus SYSFAIL* Driver Enabled

Board Reset Source

Switch W7 selects the reset signal source for circuitry shared by both ports of the GPIB-1014D, such as the DMAC. Selecting switch position SYS chooses only the VMEbus SYSRESET* (as shown in Figure 3-8). Switch position LMR permits both LMRA and LMRB of Configuration Register 2 to reset the shared circuitry.

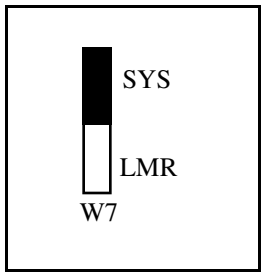


Figure 3-8. Board Reset Source

Other Configuration Parameters

All other configuration parameters for the GPIB-1014D are software selectable. Some parameters are *board-specific* for both Port A and Port B (or the GPIB-1014D as a whole) and some are *port-specific* for an individual port. Some of the board-specific parameters include the following:

- Selecting one of seven interrupt request lines (IRQ1* through IRQ7*)
- Selecting one of four Bus Request/Grant lines (BR0* through BR3*, BG0IN* through BG3IN*, BG0OUT* through BG3OUT*)
- Enabling the Release On Request (ROR) feature.

The ROR feature minimizes bus arbitration overhead by holding the system bus between transfers unless another request is pending.

Some of the port-specific parameters include the following:

- Selecting the Automatic Carry Cycle feature
- Selecting the GPIB System Controller
- Selecting the DMA transfer direction

Software control of these and other parameters, both board- and port-specific, give you the flexibility to tailor the operation and performance of each port to meet the exact requirements of separate tasks within your system.

Installation

The GPIB-1014D is a dual-height card that interfaces to the VMEbus P1 and P2 connectors. The following paragraphs describe the GPIB-1014D interface to the VMEbus backplane and to the IEEE-488 bus.

Verification of System Compatibility

The GPIB-1014D does not receive or drive all signals on the P1 connector that are included in the VMEbus specification. Compare signals listed in Tables 3-3 and 3-4 to those signals used by the VMEbus system in which the GPIB-1014D will be installed. (VMEbus system signal information is provided with your system documentation.) This is to ensure that the two are compatible (that is, that the GPIB-1014D has all the necessary signals needed by the system and vice versa).

Table 3-3. GPIB-1014D Pin Assignment on VMEbus Connector P1

Pin No.	Signal Used	Signal Not Used	Pin No.	Signal Used	Signal Not Used
A1	D00		A17	GND	
A2	D01		A18	AS*	
A3	D02		A19	GND	
A4	D03		A20	IACK*	
A5	D04		A21	IACKIN*	
A6	D05		A22	IACKOUT*	
A7	D06		A23	AM4	
A8	D07		A24	A07	
A9	GND		A25	A06	
A10	SYSCLK		A26	A05	
A11	GND		A27	A04	
A12	DS1*		A28	A03	
A13	DS0*		A29	A02	
A14	WRITE*		A30	A01	
A15	GND		A31		-12V
A16	DTACK		A32	+5V	
B1	BBSY*		B17	AM1	
B2		BCLR*	B18	AM2	
B3		ACFAIL*	B19	AM3	
B4	BG0IN*		B20	GND	
B5	BG0OUT*		B21		SERCLK
B6	BG1IN*		B22		SERDAT
B7	BG1OUT*		B23	GND	
B8	BG2IN*		B24	IRQ7*	
B9	BG2OUT*		B25	IRQ6*	
B10	BG3IN*		B26	IRQ5*	
B11	BG3OUT*		B27	IRQ4*	
B12	BR0*		B28	IRQ3*	
B13	BR1*		B29	IRQ2*	
B14	BR2*		B30	IRQ1*	
B15	BR3*		B31		+5V STDBY
B16	AM0		B32	+5V	
C1	D08		C17	A21	
C2	D09		C18	A20	
C3	D010		C19	A19	
C4	D011		C20	A18	
C5	D012		C21	A17	
C6	D013		C22	A16	
C7	D014		C23	A15	
C8	D015		C24	A14	
C9	GND		C25	A13	
C10	SYSFAIL*		C26	A12	
C11	BERR*		C27	A11	
C12	SYSRESET*	C28	C28		
C13	LWORD*		C29	A09	
C14	AM5		C30	A08	
C15	A23		C31		+12V
C16	A22		C32	+5V	

Table 3-4. GPIB-1014D Pin Assignment on VMEbus Connector P2

Pin No.	Signal Used	Signal Not Used	Pin No.	Signal Used	Signal Not Used
A1		User I/O	A17		User I/O
A2		User I/O	A18		User I/O
A3		User I/O	A19		User I/O
A4		User I/O	A20		User I/O
A5		User I/O	A21		User I/O
A6		User I/O	A22		User I/O
A7		User I/O	A23		User I/O
A8		User I/O	A24		User I/O
A9		User I/O	A25		User I/O
A10		User I/O	A26		User I/O
A11		User I/O	A27		User I/O
A12		User I/O	A28		User I/O
A13		User I/O	A29		User I/O
A14		User I/O	A30		User I/O
A15		User I/O	A31		User I/O
A16		User I/O	A32		User I/O
B1	+5V		B17		D19
B2	GND		B18		D20
B3 †			B19		D21
B4	A24		B20		D22
B5	A25		B21		D23
B6	A26		B22	GND	
B7	A27		B23		D24
B8	A28		B24		D25
B9	A29		B25		D26
B10	A30		B26		D27
B11	A31		B27		D28
B12	GND		B28		D29
B13	+5V		B29		D30
B14	D16		B30		D31
B15	D17		B31	GND	
B16	D18		B32	+5V	
C1		User I/O	C17		User I/O
C2		User I/O	C18		User I/O
C3		User I/O	C19		User I/O
C4		User I/O	C20		User I/O
C5		User I/O	C21		User I/O
C6		User I/O	C22		User I/O
C7		User I/O	C23		User I/O
C8		User I/O	C24		User I/O
C9		User I/O	C25		User I/O
C10		User I/O	C26		User I/O
C11		User I/O	C27		User I/O
C12		User I/O	C28		User I/O
C13		User I/O	C29		User I/O
C14		User I/O	C30		User I/O
C15		User I/O	C31		User I/O
C16		User I/O	C32		User I/O

† Reserved pin. Defined by the VMEbus specification.

Cabling

The GPIB-1014D interface board has two standard 24-pin IEEE-488 connectors on the front panel of the board. A standard GPIB cable can plug directly to any of the two connectors. Two GPIB cables cannot be connected side by side in this configuration; that is, if more than one GPIB-1014D is installed in the system, they cannot be placed side by side due to the width of the GPIB cable connector housing (another board must be placed between any two GPIB-1014D interface boards).

Verification Testing

A performance verification test can be run to ensure that the board has not been damaged during shipment and also to ensure that the board has been configured correctly. To do this you need an interactive control program or an equivalent mechanism, such as front panel control switches or front panel emulator, that provides a way to load and read memory and I/O addresses.

The tests presented in Chapter 7, *GPIB-1014D Diagnostic and Troubleshooting Test Procedures*, consist of a series of steps written in a pseudo (processor-independent) language with instructions. The steps generally involve writing data to specific GPIB-1014D device registers followed by reading other GPIB-1014D registers to verify that the programming is correct. These tests exercise virtually all of the major functions of the GPIB-1014D, including VMEbus I/O communications, DMA operation, and GPIB communications. All functions except GPIB communications can be performed as stand-alone operations (that is, without another GPIB device). To completely check the GPIB functions, you must use a bus tester or analyzer (such as National Instruments GPIB-400) that can monitor and control GPIB signal lines; emulate GPIB Talker, Listener, and Controller devices; and single-step through the Source and Acceptor Handshakes.

Chapter 4
Register Bit Descriptions

This chapter presents detailed information on the use of the GPIB-1014D registers. The GPIB-1014D has two ports, Port A and Port B. Each port has an identical set of TLC and configuration registers. In addition, Port A uses two DMAC channels, 0 and 1, and Port B uses the other two DMAC channels, 2 and 3. Figure 4-1 shows a breakdown of the port address space for the GPIB-1014D registers.

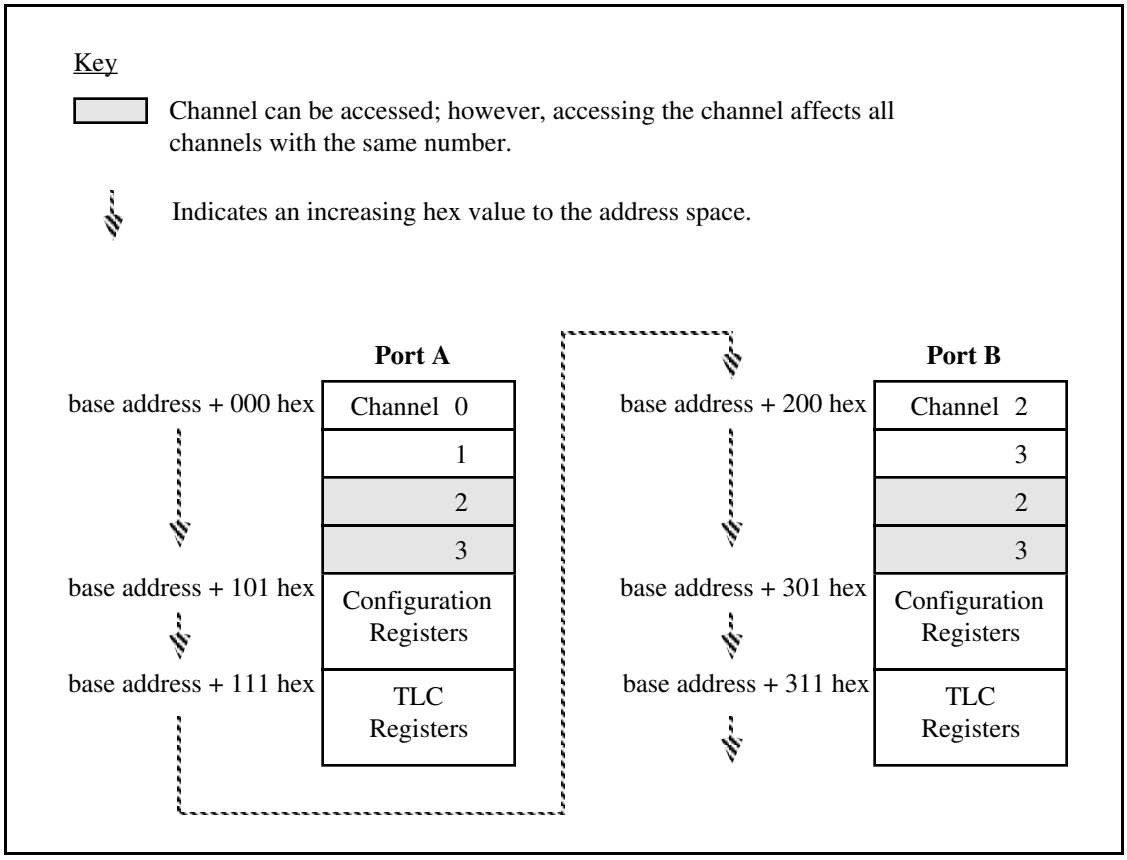


Figure 4-1. GPIB-1014D Registers Address Space

Register Maps

The register map for Port A registers is shown in Table 4-1. The register map for Port B registers is shown in Table 4-2. These tables give the register name, the register address, the size of the register in bits, and the type of the register (read only, write only, or read and write).

Table 4-1. GPIB-1014D Port A Register Map

Register Name	Address (Hex)	Type	Size
TLC Register Group:			
Data In Register	Base address + 111	Read only	8-bit
Command/Data Out Register	Base address + 111	Write only	8-bit
Interrupt Status Register 1	Base address + 113	Read only	8-bit
Interrupt Mask Register 1	Base address + 113	Write only	8-bit
Interrupt Status Register 2	Base address + 115	Read only	8-bit
Interrupt Mask Register 2	Base address + 115	Write only	8-bit
Serial Poll Status Register	Base address + 117	Read only	8-bit
Serial Poll Mode Register	Base address + 117	Write only	8-bit
Address Status Register	Base address + 119	Read only	8-bit
Address Mode Register	Base address + 119	Write only	8-bit
Command Pass Through Register	Base address + 11B	Read only	8-bit
Auxiliary Mode Register	Base address + 11B	Write only	8-bit
Hidden Registers			
Internal Counter Register	Base address + 11B	Write only	8-bit
Parallel Poll Register	Base address + 11B	Write only	8-bit
Auxiliary Register A	Base address + 11B	Write only	8-bit
Auxiliary Register B	Base address + 11B	Write only	8-bit
Auxiliary Register E	Base address + 11B	Write only	8-bit
Address Register 0	Base address + 11D	Read only	8-bit
Address Register	Base address + 11D	Write only	8-bit
Address Register 1	Base address + 11F	Read only	8-bit
End Of String Register	Base address + 11F	Write only	8-bit
DMA Register Group (Channel 0):			
Address Registers			
Memory Address Register	Base address + 00C	Read/Write	32-bit
Base Address Register	Base address + 01C	Read/Write	32-bit
Device Address Register	Base address + 014	Read/Write	32-bit
Transfer Count Registers			
Memory Transfer Counter Register	Base address + 00A	Read/Write	16-bit
Base Transfer Counter Register	Base address + 01A	Read/Write	16-bit
Function Code Registers			
Memory Function Code Register	Base address + 029	Read/Write	8-bit
Base Function Code Register	Base address + 039	Read/Write	8-bit
Device Function Code Register	Base address + 031	Read/Write	8-bit
Device Control Register	Base address + 004	Read/Write	8-bit
Operation Control Register	Base address + 005	Read/Write	8-bit
Sequence Control Register	Base address + 006	Read/Write	8-bit
Channel Control Register	Base address + 007	Read/Write	8-bit
Channel Status Register	Base address + 000	Read/Write	8-bit
Channel Error Register	Base address + 001	Read Only	8-bit
Channel Priority Register	Base address + 02D	Read/Write	8-bit
Interrupt Vector Registers			
Normal Interrupt Vector Register	Base address + 025	Read/Write	8-bit
Error Interrupt Vector Register	Base address + 027	Read/Write	8-bit
General Control Register *	Base address + 0FF	Read/Write	8-bit

(continues)

Table 4-1. GPIB-1014D Port A Register Map (continued)

Register Name	Address (Hex)	Type	Size
DMA Register Group (Channel 1):			
Address Registers			
Memory Address Register	Base address + 04C	Read/Write	32-bit
Base Address Register	Base address + 05C	Read/Write	32-bit
Device Address Register	Base address + 05A	Read/Write	32-bit
Transfer Count Registers			
Memory Transfer Counter Register	Base address + 04A	Read/Write	16-bit
Base Transfer Counter Register	Base address + 05A	Read/Write	16-bit
Function Code Registers			
Memory Function Code Register	Base address + 069	Read/Write	8-bit
Base Function Code Register	Base address + 079	Read/Write	8-bit
Device Function Code Register	Base address + 071	Read/Write	8-bit
Device Control Register	Base address + 044	Read/Write	8-bit
Operation Control Register	Base address + 045	Read/Write	8-bit
Sequence Control Register	Base address + 046	Read/Write	8-bit
Channel Control Register	Base address + 047	Read/Write	8-bit
Channel Status Register	Base address + 040	Read/Write	8-bit
Channel Error Register	Base address + 041	Read Only	8-bit
Channel Priority Register	Base address + 06D	Read/Write	8-bit
Interrupt Vector Registers			
Normal Interrupt Vector Register	Base address + 065	Read/Write	8-bit
Error Interrupt Vector Register	Base address + 067	Read/Write	8-bit
General Control Register *	Base address + 0FF	Read/Write	8-bit
Configuration Register Group:			
Configuration Register 1	Base address + 101	Write only	8-bit
Configuration Register 2	Base address + 105	Write only	8-bit
GPIB Status Register	Base address + 101	Read only	8-bit
Page Register	Base address + 109	Write only	8-bit

* This register is shared by all four DMAC channels.

Table 4-2. GPIB-1014D Port B Register Map

Register Name	Address (Hex)	Type	Size
TLC Register Group:			
Data In Register	Base address + 311	Read only	8-bit
Command/Data Out Register	Base address + 311	Write only	8-bit
Interrupt Status Register 1	Base address + 313	Read only	8-bit
Interrupt Mask Register 1	Base address + 313	Write only	8-bit
Interrupt Status Register 2	Base address + 315	Read only	8-bit
Interrupt Mask Register 2	Base address + 315	Write only	8-bit
Serial Poll Status Register	Base address + 317	Read only	8-bit
Serial Poll Mode Register	Base address + 317	Write only	8-bit
Address Status Register	Base address + 319	Read only	8-bit
Address Mode Register	Base address + 319	Write only	8-bit
Command Pass Through Register	Base address + 31B	Read only	8-bit
Auxiliary Mode Register	Base address + 31B	Write only	8-bit
Hidden Registers			
Internal Counter Register	Base address + 31B	Write only	8-bit
Parallel Poll Register	Base address + 31B	Write only	8-bit
Auxiliary Register A	Base address + 31B	Write only	8-bit
Auxiliary Register B	Base address + 31B	Write only	8-bit
Auxiliary Register E	Base address + 31B	Write only	8-bit
Address Register 0	Base address + 31D	Read only	8-bit
Address Register	Base address + 31D	Write only	8-bit
Address Register 1	Base address + 31F	Read only	8-bit
End Of String Register	Base address + 31F	Write only	8-bit
DMA Register Group (Channel 2):			
Address Registers			
Memory Address Register	Base address + 20C	Read/Write	32-bit
Base Address Register	Base address + 21C	Read/Write	32-bit
Device Address Register	Base address + 214	Read/Write	32-bit
Transfer Count Registers			
Memory Transfer Counter Register	Base address + 20A	Read/Write	16-bit
Base Transfer Counter Register	Base address + 21A	Read/Write	16-bit
Function Code Registers			
Memory Function Code Register	Base address + 229	Read/Write	8-bit
Base Function Code Register	Base address + 239	Read/Write	8-bit
Device Function Code Register	Base address + 231	Read/Write	8-bit
Device Control Register	Base address + 204	Read/Write	8-bit
Operation Control Register	Base address + 205	Read/Write	8-bit
Sequence Control Register	Base address + 206	Read/Write	8-bit
Channel Control Register	Base address + 207	Read/Write	8-bit
Channel Status Register	Base address + 200	Read/Write	8-bit
Channel Error Register	Base address + 201	Read Only	8-bit
Channel Priority Register	Base address + 22D	Read/Write	8-bit
Interrupt Vector Registers			
Normal Interrupt Vector Register	Base address + 225	Read/Write	8-bit
Error Interrupt Vector Register	Base address + 227	Read/Write	8-bit
General Control Register *	Base address + 2FF	Read/Write	8-bit

(continues)

Table 4-2. GPIB-1014D Port B Register Map (continued)

Register Name	Address (Hex)	Type	Size
DMA Register Group (Channel 3):			
Address Registers			
Memory Address Register	Base address + 24C	Read/Write	32-bit
Base Address Register	Base address + 25C	Read/Write	32-bit
Device Address Register	Base address + 254	Read/Write	32-bit
Transfer Count Registers			
Memory Transfer Counter Register	Base address + 24A	Read/Write	16-bit
Base Transfer Counter Register	Base address + 25A	Read/Write	16-bit
Function Code Registers			
Memory Function Code Register	Base address + 269	Read/Write	8-bit
Base Function Code Register	Base address + 279	Read/Write	8-bit
Device Function Code Register	Base address + 271	Read/Write	8-bit
Device Control Register	Base address + 244	Read/Write	8-bit
Operation Control Register	Base address + 245	Read/Write	8-bit
Sequence Control Register	Base address + 246	Read/Write	8-bit
Channel Control Register	Base address + 247	Read/Write	8-bit
Channel Status Register	Base address + 240	Read/Write	8-bit
Channel Error Register	Base address + 241	Read Only	8-bit
Channel Priority Register	Base address + 26D	Read/Write	8-bit
Interrupt Vector Registers			
Normal Interrupt Vector Register	Base address + 265	Read/Write	8-bit
Error Interrupt Vector Register	Base address + 267	Read/Write	8-bit
General Control Register *	Base address + 2FF	Read/Write	8-bit
Configuration Register Group:			
Configuration Register 1	Base address + 301	Write only	8-bit
Configuration Register 2	Base address + 305	Write only	8-bit
GPIB Status Register	Base address + 301	Read only	8-bit
Page Register	Base address + 309	Write only	8-bit

* This register is shared by all four DMAC channels.

Register Sizes

The VMEbus computers support three different transfer sizes for read and write operations: 8-, 16-, or 32-bit. Table 4-1 shows the size of the registers in Port A and Table 4-2 shows the size of the registers in Port B. The 8-bit registers in the TLC and Configuration Register Group are accessible using 8- or 16-bit read/write operations. All 8-, 16-, or 32-bit DMAC Channel Registers are accessed using 8- or 16-bit read/write operations.

Register Description

Table 4-1 divides Port A registers into three different register groups and Table 4-2 divides Port B registers into three different register groups. A bit description of each of the registers making up these groups is included later in this chapter.

For each port, the GPIB Interface Register Groups consists of 16 registers located in the NEC µPD7210 TLC chip. (There are two TLC chips on the GPIB-1014 interface board.) Each DMA Channel Register Group consists of 18 registers on the 68450 DMAC chip. These DMA registers support unchained, continue, array-chained, or link-chained DMA operations between memory and memory or between memory and device (GPIB). For each port, the Configuration Register Groups contain three registers used to configure some of the board operating parameters and one register to monitor the status of the port's GPIB.

Register Description Format

The remainder of this chapter discusses each register in the order shown in Tables 4-1 and 4-2. Each register group is introduced, followed by a detailed bit description of each register. The individual register description gives the register offset, type, size, and bit map of the register, followed by a description of each bit.

The register bit map shows a diagram of the register with the most significant bit (bit 15 for a 16-bit register, bit 7 for an 8-bit register) shown on the left, and the least significant bit (bit 0) shown on the right. A square is used to represent each bit. Each bit is labeled with a name inside its square. An asterisk (*) after the bit name indicates that the signal is active low. An asterisk is equivalent to an overbar.

In many of the registers, several bits are labeled with an X, indicating "don't care bits." When a register is read, these bits may appear set or cleared but should be ignored because they have no significance. If a register is written to, these bits should be written as zeros. Many of the bits in the Configuration Register Group are shaded, indicating "board-specific bits." These bits configure board-specific parameters that are shared by both ports such as the VMEbus interrupt level or the bus grant/request level. Each shared bit is implemented with one memory element; therefore, it cannot hold a different value for each port. The configuration register bits that are not shaded are port-specific such as the direction of data transfer and the carry cycle feature selection. These bits are implemented using separate storage elements.

Mnemonics are assigned to messages, states, registers and bits. Most mnemonics contain a clue to their meaning. Table 4-3 contains a list of clues to look for.

Table 4-3. Clues to Understanding Mnemonics

Clue	Probable Mnemonic Meaning
Ends in IE	Interrupt Enable Bit
Ends in EN	Enable Bit
Ends in A or B	Register in Port A or Port B
4 letters, ends in S	Interface function as defined in the IEEE-488 standard
Ends in R, R0, R1, R2	GPIB Program Register

(continues)

Table 4-3. Clues to Understanding Mnemonics (continued)

Clue	Probable Mnemonic Meaning
3 letters, uppercase	Remote GPIB Message
3 letters, lowercase	Local GPIB Message

Appendix F contains an alphabetical list of all mnemonics within this manual.

Interface Registers

Port A and Port B each have sixteen GPIB Interface registers, eight of which are readable and eight of which are writable. The registers are located within the NEC µPD7210 Talker/Listener/Controller (TLC) integrated circuit. Each of the 16 interface registers is addressed relative to the GPIB-1014D VMEbus base address which is set with hardware switches (refer to *Base Address* in Chapter 3).

Figures 4-2 and 4-3 show the register and bit mnemonics for each TLC internal register, the read/write accessibility of the register, and the address relative to the base address of the port. A detailed function description of all 16 interface registers is provided in the paragraphs following the figures.

Figure 4-2 shows the GPIB Interface registers of Port A and Port B. Figure 4-2 shows the hidden GPIB interface registers of Port A and Port B and illustrates the method of writing to those registers via the Auxiliary Mode Register.

Note: The address offsets in Figure 4-2 are for Port A. To figure out the address offsets for Port B you must add 200 to the offset shown. For example, the address offset for Port B ISR2 is 315 (115 + 300).

Legend

(Contents of Read Register)							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
(Contents of Write Register)							

	Address Offset (hex)									Read/ Write
DIR	+111	DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0	R
CDOR		CDO7	CDO6	CDO5	CDO4	CDO3	CDO2	CDO1	CDO0	W
ISR1	+113	CPT	APT	DET	END RX	DEC	ERR	DO	DI	R
IMR1		CPT IE	APT IE	DET IE	END IE	DEC IE	ERR IE	DO IE	DI IE	W
ISR2	+115	INT	SRQI	LOK	REM	CO	LOKC	REMC	ADSC	R
IMR2		0	SRQI IE	DMAO	DMAI	CO IE	LOKC IE	REMC IE	ADSC IE	W
SPSR	+117	S8	PEND	S6	S5	S4	S3	S2	S1	R
SPMR		S8	rsv	S6	S5	S4	S3	S2	S1	W
ADSR	+119	CIC	ATN*	SPMS	LPAS	TPAS	LA	TA	MJMN	R
ADMR		ton	lon	TRM1	TRM0	0	0	ADM1	ADM0	W
CPTR	+11B	CPT7	CPT6	CPT5	CPT4	CPT3	CPT2	CPT1	CPT0	R
AUXMR		CNT2	CNT1	CNT0	COM4	COM3	COM2	COM1	COM0	W
ADR0	+11D	X	DT0	DL0	AD5-0	AD4-0	AD3-0	AD2-0	AD1-0	R
ADR		ARS	DT	DL	AD5	AD4	AD3	AD2	AD1	W
ADR1	+11F	EOI	DT1	DL1	AD5-1	AD4-1	AD3-1	AD2-1	AD1-1	R
EOSR		EOS7	EOS6	EOS5	EOS4	EOS3	EOS2	EOS1	EOS0	W

Note: X indicates a don't care bit.

Figure 4-2. Interface Registers

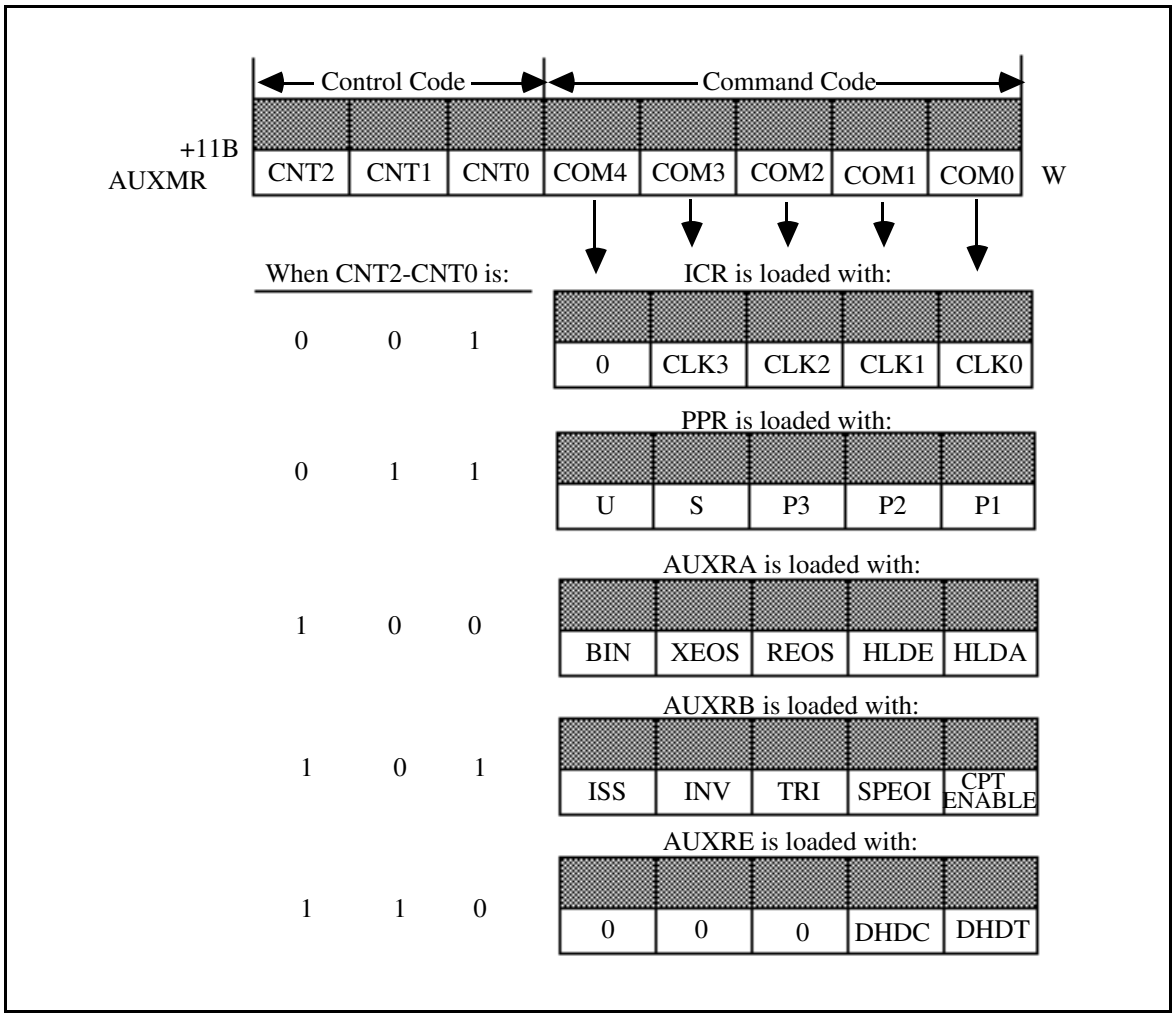


Figure 4-3. Writing to the Hidden Registers

Data In Register (DIR)

VMEbus Address: Port Address + 111 (hex)

Attributes: Read Only, Internal to TLC

7	6	5	4	3	2	1	0	R
DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0	

The Data In Register (DIR) is used to move data from the GPIB to the computer when the interface is a Listener. The GPIB Ready For Data (RFD) message is held false until the byte is removed from the DIR, either by a DMA transfer from the DIR to the VMEbus memory or by an I/O read from a VMEbus master. In Ready For Data (RFD) Holdoff mode (refer to *Auxiliary Register A* later in this chapter), the GPIB Handshake is not finished until the Finish Handshake (FH) auxiliary command is issued telling the TLC to release the Holdoff. By using the RFD Holdoff mode, the same byte can be read several times or a GPIB Talker that is ready to provide more data can be held off until the program is ready to proceed.

DI0 is the least significant bit of the data byte and corresponds to GPIB DI01. DI7 is the most significant bit of the data byte and corresponds to GPIB DI08.

Bit	Mnemonic	Description
7-0r	DI[7-0]	Data In Bits 7 through 0

Command/Data Out Register (CDOR)

VMEbus Address: Port Address + 111 (hex)

Attributes: Write Only, Internal to TLC

7	6	5	4	3	2	1	0
CDO7	CDO6	CDO5	CDO4	CDO3	CDO2	CDO1	CDO0

The Command/Data Out Register (CDOR) is used to move data from the VMEbus to the GPIB when the interface (TLC) is the GPIB Talker or the Active Controller. Outgoing data is separately latched by this register and is not destroyed by a read from the DIR. When a byte is written to the CDOR, the TLC GPIB Source Handshake (SH) function is initiated and the byte is transferred to the GPIB.

Bit	Mnemonic	Description
7-0w	CDO[7-0]	Command/Data Out Bits 7 through 0

Interrupt Status Register 1 (ISR1)

VMEbus Address: Port Address + 113 (hex)

Attributes: Read Only, Internal to TLC
Bits are cleared when read

Interrupt Mask Register 1 (IMR1)

VMEbus Address: Port Address + 113 (hex)

Attributes: Write Only, Internal to TLC

7	6	5	4	3	2	1	0	
CPT	APT	DET	END RX	DEC	ERR	DO	DI	R
CPT IE	APT IE	DET IE	END IE	DEC IE	ERR IE	DO IE	DI IE	W

The Interrupt Status Register 1 (ISR1) is composed of eight Interrupt Status bits. The Interrupt Mask Register 1 (IMR1) is composed of eight Interrupt Enable bits that directly correspond to the Interrupt Status bits in ISR1. As a result, ISR1 and IMR1 service eight possible interrupt conditions, where each condition has an Interrupt Status bit and an Interrupt Enable bit associated with it. If the Interrupt Enable bit is true when the corresponding status condition or event occurs, a hardware interrupt request is generated. Bits in ISR1 are set and cleared by the TLC regardless of the status of the Interrupt bits in IMR1. If an interrupt condition occurs at the same time ISR1 is being read, the TLC holds off setting the corresponding Status bit until the read has finished.

Bit **Mnemonic** **Description**

7r CPT Command Pass Through Bit
7w CPT IE Command Pass Through Interrupt Enable Bit

CPT is set on:

[UCG + ACG and (TADS + LADS)] and undefined and ACDS and (CPT ENABLE) + UDPCF and SCG and ACDS and CPT ENABLE

CPT is cleared by:

pon + (read ISR1)

Notes

UCG: GPIB Universal Command Group message
ACG: GPIB Addressed Command Group message
TADS: GPIB Talker Addressed State
LADS: GPIB Listener Addressed State
defined: GPIB command automatically recognized and executed by TLC

Bit	Mnemonic	Description

undefined: GPIB command not automatically recognized and executed by TLC
ACDS: GPIB Accept Data State
CPT ENABLE: AUXRB[0]w
UDPCF: Undefined Primary Command Function
SCG: GPIB Secondary Command Group message
pon: Power On Reset
TAG: GPIB Talk Address Group message
LAG: GPIB Listen Address Group message
read ISR1: Bit is cleared immediately after it is read

UDPCF is set on:

[UCG + ACG and (TADS + LADS)] and undefined and ACDS and CPT ENABLE

UDPCF is cleared on:

[(UCG + ACG) and defined + TAG + LAG] and ACDS + CPT ENABLE* + pon

The CPT bit flags the occurrence of a GPIB command not recognized by the TLC when the Command Pass Through feature is enabled by the CPT ENABLE bit, AUXRB[0]w. Any GPIB command message not decoded by the TLC is treated as an undefined command, the Go To Local [GTL] command); however, any addressed command is automatically ignored when the TLC is not addressed.

Undefined commands are read using the Command Pass Through Register (CPTR). The TLC holds off the GPIB Acceptor Handshake in the Accept Data State (ACDS) until the Valid auxiliary command function code, 0F hex, is written to the AUXMR. If the CPT feature is not enabled, undefined commands are simply ignored.

6r	APT	Address Pass Through
6w	APT IE	Address Pass Through Interrupt Enable

APT is set by:

ADM1 and ADM0 and (TPAS + LPAS) and SCG and ACDS

APT is cleared by:

pon + (read ISR1)

Notes

ADM1: Address Mode Register Bit 1, ADMR[1]w
ADM0: Address Mode Register Bit 0, ADMR[0]w
TPAS: GPIB Talker Primary Addressed State
LPAS: GPIB Listener Primary Addressed State
SCG: GPIB Secondary Command Group

Bit	Mnemonic	Description

ACDS: GPIB Accept Data State
pon: Power On Reset
Read ISR1: Bit is cleared immediately after it is read

The APT bit indicates that a secondary GPIB address has been received and is available in the CPTR for inspection

Note: The application program must check this bit when using TLC address mode 3.

When APT is set, the Data Accepted (DAC) message is held and the GPIB Handshake stops until either the Valid or Non-Valid auxiliary command is issued. The secondary address can be read from the CPTR.

5r DET
5w DET IE

Device Execute Trigger Bit
Device Execute Trigger Interrupt Enable Bit

DET is set by:

DTAS

DET is cleared by:

pon + (read ISR1)

Notes

DTAS: GPIB Device Trigger Active State
pon: Power On Reset
read ISR1: Bit is cleared immediately after it is read

The DET bit indicates that the GPIB Device Execute Trigger (DET) command has been received while the TLC was a GPIB Listener (the TLC has been in DTAS).

4r END RX
4w END IE

End Received Bit
End Received Interrupt Enable Bit

END RX is set by:

LACS and (EOI + EOS and REOS) and ACDS

END RX is cleared by:

pon + (read ISR1)

Notes

LACS: GPIB Listener Active State
EOI: GPIB End Of Identify Signal
EOS: GPIB END Of String message

Bit	Mnemonic	Description

REOS: Reception of GPIB EOS allowed, AUXRA[2]w
ACDS: GPIB Accept Data State
pon: Power On Reset
read ISR1: Bit is cleared immediately after it is read

The END RX bit is set when the TLC is a Listener and the GPIB uniline message, END, is received with a data byte from the GPIB Talker, or the data byte in the DIR matches the contents of the End Of String Register (EOSR).

3r DEC
3w DEC IE

Device Clear Bit
Device Clear Interrupt Enable Bit

DEC is set by:

DCAS

DEC is cleared by:

pon + (read ISR1)

Notes

DCAS: GPIB Device Clear Active State
pon: Power On Reset
read ISR1: Bit is cleared immediately after it is read

The DEC bit indicates that the GPIB Device Clear (DCL) command has been received or that the GPIB Selected Device Clear (SDC) command has been received while the TLC was a GPIB Listener (the TLC is in DCAS).

2r ERR
2w ERR IE

Error Bit
Error Interrupt Enable Bit

ERR is set by:

TACS and SDYS and DAC and RFD + SIDS and (write CDOR) + (SDYS to SIDS)

ERR is cleared by:

pon + (read ISR1)

Notes

TACS: GPIB Talker Active State
SDYS: GPIB Source Delay State
DAC: GPIB Data Accepted message
RFD: GPIB Ready For Data message
SIDS: GPIB Source Idle State

Bit	Mnemonic	Description

write CDOR: Bit is set immediately after writing to the Command/Data Out Register
SDYS to SIDS: Transition from GPIB Source Delay State to Source Idle State
pon: Power On Reset
read ISR1; Bit is cleared immediately after it is read

The ERR bit indicates that the contents of the CDOR have been lost. ERR is set when data is sent to the GPIB without a specified Listener or when a byte is written to the CDOR during SIDS or during the SDYS to SIDS transition.

1r DO Data Out Bit
1w DO IE Data Out Interrupt Enable Bit

DO is set as:

(TACS and SGNS) becomes true

DO is cleared by:

(read ISR1) + TACS* + SGNS*

Notes

TACS: GPIB Talker Active State
SGNS: GPIB Source Generate State
read ISR1: Bit is cleared immediately after it is read

The DO bit indicates that the TLC is ready to accept another data byte from the VMEbus for transmission on to the GPIB when the TLC is the GPIB Talker. The DO bit is cleared when a byte is written to the CDOR and also when the TLC ceases to be the Active Talker. When performing a DMA operation, DO IE must be clear so that an interrupt request does not occur. Instead, the DMA0 bit in the Interrupt Mask Register 2 (IMR2[5]w) must be set to enable a DMA cycle request when DO is asserted.

0r DI Data In Bit
0w DI IE Data In Interrupt Enable Bit Bit

DI is set by:

LACS and ACDS and continuous mode

DI is cleared by:

pon + (read ISR1) + (Finish Handshake) and (Holdoff mode) + (read DIR)

Bit **Mnemonic** **Description**

Notes

LACS:	GPIB Listener Active State
ACDS:	GPIB Accept Data State
continuous mode:	Listen in continuous mode auxiliary command in effect
pon:	Power On Reset
read ISR1:	Bit is cleared immediately after it is read
finish Handshake:	Finish Handshake auxiliary command issued
Holdoff mode:	RFD Holdoff state
read DIR:	Read Data In Register

The DI bit indicates that the TLC, as a GPIB Listener, has accepted a data byte from the GPIB Talker. When performing a DMA operation, DI IE must be clear so that an interrupt request does not occur. Instead, the DMAI bit in the Interrupt Mask Register 2 (IMR1[4]]w) must be set to enable a DMA cycle request when DI is asserted.

Interrupt Status Register 2 (ISR2)

VMEbus Address: Port Address + 115 (hex)

Attributes: Read Only, Internal to TLC
Bits are cleared when read

Interrupt Mask Register 2 (IMR2)

VMEbus Address: Port Address + 115 (hex)

Attributes: Write Only, Internal to TLC

7	6	5	4	3	2	1	0	
INT	SRQI	LOK	REM	CO	LOKC	REMC	ADSC	R
0	SRQI IE	DMAO	DMAI	CO IE	LOKC IE	REMC IE	ADSC IE	W

The Interrupt Status Register 2 (ISR2) consists of six Interrupt Status bits and two TLC Internal Status bits. The Interrupt Mask Register 2 (IMR2) consists of five Interrupt Enable bits and two TLC Internal Control bits. If the Interrupt Enable bit is true when the corresponding status condition or event occurs, an interrupt request is generated. Bits in ISR2 are set and cleared regardless of the status of the bits in IMR2. If a condition occurs which requires the TLC to set or clear a bit or bits in ISR2 at the same time ISR2 is being read, the TLC holds off setting or clearing the bit or bits until the read is finished.

Bit **Mnemonic** **Description**

7r INT Interrupt Bit

This bit is the logical OR of all the Enabled Interrupt Status bits in both ISR1 and ISR2, each one ANDed with its Interrupt Enable bit. There is no corresponding Mask bit for INT.

INT is set by:

(CPT and CPT IE) + (APT and APT IE) + (DET and DET IE) + ERR and ERR IE) + (END RX and END IE) + (DEC and DEC IE) + (DO and DO IE) + (DI and DI IE) + (SRQI and SRQI IE) + (REMC and REMC IE) + (CO and CO IE) + (LOKC and LOKC IE) + (ADSC and ADSC IE)

Notes

CPT: Command Pass Through Bit
CPT IE: Enable Interrupt on Command Pass Through Bit
APT: Address Pass Through Bit
APT IE: Enable Interrupt on Address Pass Through Bit
DET: Device Execute Trigger Bit
DET IE: Enable Interrupt on Device Execute Trigger Bit
ERR: Error Bit

Bit	Mnemonic	Description
		ERR IE: Enable Interrupt on Error Bit END RX: End Received Bit END IE: Enable Interrupt on End Received Bit DEC: Device Clear Bit DEC IE: Enable Interrupt on Device Clear Bit DO: Data Out Bit DO IE: Enable Interrupt on Data Out Bit DI: Data In Bit DI IE: Enable Interrupt on Data In Bit SRQI: Service Request Input Bit SRQI IE: Enable Interrupt on Service Request Input Bit REMC: Remote Change Bit REMC IE: Enable Interrupt on Remote Change Bit CO: Command Output Bit CO IE: Enable Interrupt on Command Output Bit LOKC: Lockout Change Bit LOKC IE: Enable Interrupt on Lockout Change Bit ADSC: Address Status Change Bit ADSC IE: Enable Interrupt on Address Status Change Bit
7w	0	Reserved Bit Write zero to this bit.
6r 6w	SRQI SRQI IE	Service Request Input Bit Service Request Input Interrupt Enable Bit SRQI is set when: (CIC and SRQ and -(RQS and DAV)) becomes true or (CIC and SRQ and RQS and DAV) becomes true. SRQI is cleared by: pon + (read ISR2) **Notes** CIC: GPIB Controller-In-Charge SRQ: GPIB Service Request message RQS: GPIB Request Service message DAV: GPIB Data Valid message pon: Power On Reset read ISR2: Bit is cleared immediately after it is read The SRQI bit indicates that a GPIB Service Request (SRQ) message has been received while the TLC function is active (CIC=1). **Note:** The set SRQI equation only applies to situations in which two or more devices are issuing the SRQ message.

Bit	Mnemonic	Description
5r	LOK	Lockout Bit LOK is used, along with the REM bit, to indicate the status of the TLC GPIB Remote/Local (RL) function. If set, the LOK bit indicates that the TLC is in Local With Lockout State (LWLS) or Remote With Lockout State (RWLS). LOK is a Non-Interrupt bit.
5w	DMAO	DMA Out Enable Bit The DMAO bit must be set to allow data transfers from VMEbus memory to the CDOR. DO IE must be clear, DMAO must be set, and the TLC must be the active GPIB Talker when a DMAO bit is set, the DO condition causes a data transfer request rather than an interrupt request.
4r	REM	Remote Bit This bit is true when the TLC GPIB RL function is in one of two states: Remote State (REMS) or Remote With Lockout State (RWLS). The TLC RL function transfers to one of these states when the System Controller has asserted the Remote Enable line (REN), and the CIC addresses the TLC as a Listener.
4w	DMAI	DMA Input Enable Bit The DMAO bit must be set to allow data transfers from the DIR to VMEbus memory. DI IE must be clear, DMAI must be set, and the TLC must be an active GPIB Listener when a *DMA in* operation is initiated. If DMAI is set, the DI condition causes a data transfer request rather than an interrupt request.
3r 3w	CO CO IE	Command Out Bit Command Out Interrupt Enable Bit CO is set when: (CACS and SGNS) becomes true CO is cleared by: (read ISR2) + CACS* + SGNS* **Notes** CACS: GPIB Controller Active State SGNS: GPIB Source Generate State read ISR2: Bit is cleared immediately after it is read CO = 1 indicates that the CDOR is empty and that another command can be written to it for transmission over the GPIB without overwriting a previous command.

Bit	Mnemonic	Description
2r 2w	LOKC LOKC IE	Lockout Change Bit Lockout Change Interrupt Enable Bit

LOKC is set by:

any change in LOK

LOKC is cleared by:

pon + (read ISR2)

Notes

LOK: ISR2[5]r
pon: Power On Reset
read ISR2: Bit is cleared immediately after it is read

LOKC is set when there is a change in the LOK bit, ISR2[5]r, (REMS +RELS).

1r 1w	REMC REMC IE	Remote Change Bit Remote Change Interrupt Enable Bit

REMC is set by:

any change in REM

REMC is cleared by:

pon + (read ISR2)

Notes

REM: ISR2[4]r
pon: Power On Reset
read ISR2: Bit is cleared immediately after it is read

REMC is set when there is a change in the REM bit, ISR2[4]r, (REMS + RELS).

0r 0w	ADSC ADSC IE	Addressed Status Change Bit Addressed Status Change Interrupt Enable Bit

ADSC is set by:
[(any change in TA) + (any change in LA) + (any change in CIC) + (any change in MJMN)] and -(lon + ton)

ADSC is cleared by:
pon + (read ISR2)

Bit **Mnemonic** **Description**

Notes

TA:	Talker Active bit, ADSR[1]r
LA:	Listener Active bit, ADSR[2]r
CIC:	Controller-In-Charge bit, ADSR[7]r
MJMN:	Major/Minor bit, ADSR[0]r
lon:	Listen Only bit, ADMR[6]w
ton:	Talk Only bit, ADMR[7]w
pon:	Power On Reset
read ISR2:	Bit is cleared immediately after it is read
ADSR:	Address Status Register
ADMR:	Address Mode Register

ADSC is set when there is a change in one of the four bits–TA, LA, CIC, or MJMN–of the Address Status Register (ADSR).

Serial Poll Status Register (SPSR)

VMEbus Address: Port Address + 117 (hex)

Attributes: Read Only, Internal to TLC

Serial Poll Mode Register (SPMR)

VMEbus Address: Port Address + 117 (hex)

Attributes: Write Only, Internal to TLC

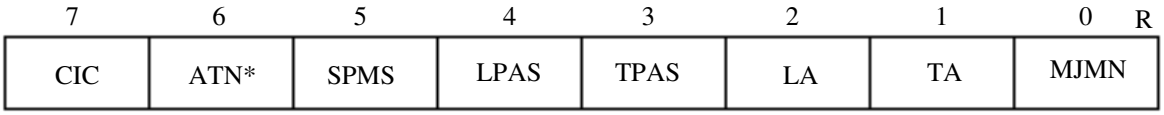
7	6	5	4	3	2	1	0	R
S8	PEND	S6	S5	S4	S3	S2	S1	
S8	rsv	S6	S5	S4	S3	S2	S1	
								W

Bit	Mnemonic	Description
7r	S8	Serial Poll Status Bit 8
7w, 5-0r, 5-0w	S[6-1]	Serial Poll Status Bits 6 through 1 Cleared by Power On Reset (pon), and by issuing the Chip Reset auxiliary command. These bits are used for sending device- or system-dependent status information to the GPIB when the TLC is serially polled. When the TLC is addressed as the GPIB Talker and receives the GPIB multiline Serial Poll Enable (SPE) command message, the TLC transmits a byte of status information, SPMR[7- 0], to the Controller-In-Charge after the Controller Goes to Standby and becomes an Active Listener.
6r	PEND	Pending Bit PEND is set when rsv=1 and cleared when Negative Poll Response State (NPRS) and Request Service (rsv) = 1. Reading the PEND status bit can confirm that a request was accepted and that the Status Byte (STB) was transmitted (PEND=0).
6w	rsv	Request Service Bit The rsv bit is used for generating the GPIB local rsv message. When rsv is set and the GPIB Active Controller is not serially polling the TLC, the TLC enters the Service Request State (SRQS) and asserts the GPIB SRQ signal. When the Active Controller reads the STB during the poll, the TLC clears rsv at the Affirmative Poll Response State (APRS). The rsv bit is also cleared by pon, and by issuing the Chip Reset auxiliary command.

Address Status Register (ADSR)

VMEbus Address: Port Address + 119 (hex)

Attributes: Read Only, Internal to TLC



The Address Status Register (ADSR) contains information that can be used to monitor the TLC GPIB address status.

Bit	Mnemonic	Description
7r	CIC	Controller-In-Charge Bit CIC = -(CIDS + CADS) CIC indicates that the TLC GPIB Controller function is in an active or standby state, with ATN* on or off, respectively. If CIC=0, the Controller function is in an idle state, with ATN* off.
6r	ATN*	Attention* Bit ATN* is a Status bit which indicates the current level of the GPIB ATN* signal. If ATN* = 0, the GPIB ATN* signal is asserted.
5r	SPMS	Serial Poll Mode State Bit If SPMS=1, the TLC GPIB Talker (T) or Talker Extended (TE) function is enabled to participate in a Serial Poll. SPMS is set when the TLC has been addressed as a GPIB Talker and the GPIB Active Controller has issued the GPIB Serial Poll Enable (SPE) command message. SPMS is cleared when the GPIB Serial Poll Disable (SPD) command is received by pon, by LMR (CR0[2]w), or by issuing the Chip Reset auxiliary command.
4r	LPAS	Listener Primary Addressed State Bit The LPAS bit is used when the TLC is configured for extended GPIB addressing and, when set, indicates that the TLC has received its primary listen address. In mode 3 addressing (see *Address Mode Register* in this chapter), LPAS=1 indicates that the secondary address being received on the next GPIB command may represent the TLC extended (secondary) GPIB listen address. LPAS is cleared by pon, by LMR (CR0[2]w), or by issuing the Chip Reset auxiliary command.

Bit	Mnemonic	Description
3r	TPAS	Talker Primary Addressed State Bit TPAS is used when the TLC is configured for extended GPIB addressing, and, when set, indicates that the TLC has received its primary GPIB talk address. In extended mode addressing (mode 3 addressing), TPAS=1 indicates that the secondary address being received as the next GPIB command message can represent the TLC extended (secondary) GPIB talk address.
2r	LA	Listener Active Bit LA is set when the TLC has been addressed or programmed as a GPIB Listener; that is, the TLC is in the Listener Active State, LACS, or the Listener Addressed State (LADS). The TLC can be addressed to listen either by sending its own listen or extended listen address while it is Controller-In-Charge or by receiving its listen address from another Controller-In-Charge. It can also be programmed to listen using the Listen Only (lon) bit in the Address Mode Register (ADMR). If the TLC is addressed to listen, it is automatically unaddressed to talk. LA is cleared by pon, or by issuing the Chip Reset auxiliary command.
1r	TA	Talker Active Bit TA is set when the TLC has been addressed or programmed as the GPIB Talker; that is, the TLC is in the Talker Active State (TACS), the Talker Addressed State (TADS), or the Serial Poll Active State (SPAS). The TLC can be addressed to talk either by sending its own talk or extended talk address while it is CIC or by receiving its talk address from another CIC. It can also be programmed to talk using the Talk Only (ton) bit in the ADMR. TA is cleared by pon, or by issuing the Chip Reset auxiliary command.
0r	MJMN	Major-Minor Bit The MJMN bit is used to determine whether the information in the other ADSR bits applies to the TLC major or minor Talker and Listener functions. MJMN is set to one when the TLC GPIB minor talk address or minor listen address is received. MJMN is cleared on receipt of the TLC major talk or major listen address. **Note:** Only one Talker and Listener may be active at any one time; thus, the MJMN bit indicates which, if either, of the TLC Talker and Listener functions is addressed or active. MJMN is always zero unless a dual primary addressing mode (mode 1 or mode 3) is enabled (see *Address Mode Register* later in this chapter).

Address Mode Register (ADMR)

VMEbus Address: Port Address + 119 (hex)

Attributes: Write Only, Internal to TLC

7	6	5	4	3	2	1	0
ton	1on	TRM1	TRM0	0	0	ADM1	ADM0
							W

Bit	Mnemonic	Description
7w	ton	Talk Only Bit By setting ton programs, the TLC becomes a GPIB Talker. If ton is set, the lon, ADM1, and ADM0 bits must be cleared. This method must be used in place of the addressing method when the TLC will be only a Talker. **Note:** Clearing ton does not by itself take the TLC out of GPIB Talker Active State (TACS). It is also necessary to execute the Chip Reset or Immediate Execute pon auxiliary command.
6w	lon	Listen Only Bit By setting lon programs, the TLC becomes a GPIB Listener. If lon is set, ton, ADM1, and ADM0 must be cleared. **Note:** Clearing lon does not, by itself, take the TLC out of GPIB Listener Active State (LACS). It is also necessary to execute the Chip Reset or Immediate Execute pon auxiliary command.
5-4w	TRM[1-0]	Transmit/Receive Mode Bit TRM1 and TRM0 control the function of the TLC T/R2 and T/R3 output pins. For proper operation, set both TRM1 and TRM0 to 1.
3-2w	0	Reserved Bits Write zeros to these bits.
1-0w	ADM[1-0]	Address Mode Bits 1 through 0 These bits state the addressing mode currently in effect–that is, the manner in which the information in ADR0 and ADR1 is interpreted (see *Address Register 0* and *Address Register 1* later in this chapter). If both bits are zero, then the TLC does not respond to GPIB address

Bit **Mnemonic** **Description**

commands. Instead, the ton and lon bits are used to program the Talker and Listener functions, respectively. The ton and lon bits must be cleared if mode 1, 2, or 3 addressing is selected, and the ADM1 through 0 bits must be cleared if either of the bits ton or lon are set.

Mode	ADM1	ADM0	Title
0	0	0	ton/lon
1	0	1	Normal dual addressing
2	1	0	Extended single addressing
3	1	1	Extended dual addressing

In mode 1 ADR0 and ADR1 contain the major and minor addresses, respectively, for dual primary GPIB address applications; that is, the TLC responds to two GPIB addresses: a major address and a minor address. The MJMN bit in the ADSR indicates which address was received. In applications where the TLC needs to respond to only one address, the major Talker and Listener function is used and the minor Talker and Listener function should be disabled. The minor Talker and Listener function can be disabled by setting the Disable Talker (DT) and Disable Listener (DL) bits in ADR1 (set ADR and ADR1).

In mode 2 (ADM1=1, ADM0=0), the TLC recognizes two sequential GPIB address bytes, a primary followed by a secondary. Both GPIB address bytes must be received in order to enable the TLC to talk or listen. In this manner, mode 2 addressing uses the Extended Talker and Extended Listener functions as defined in IEEE-488, without requiring computer program intervention. In mode 2, ADR0 and ADR1 contain the TLC primary and secondary GPIB addresses, respectively.

In mode 3 (ADM1=1, ADM0=1), the TLC handles addressing just as it does in mode 1, except that each major or minor GPIB primary address must be followed by a secondary address. All secondary GPIB addresses must be verified by computer program when mode 3 is used. When the TLC is in Talker Primary Addressed State (TPAS) or Listener Primary Addressed State (LPAS) and a secondary address byte is on the GPIB DIO lines, the APT bit of ISR2 is set and the secondary GPIB address may be inspected in the CPTR. The TLC Acceptor Handshake is held up in the Accept Data State (ACDS) until the Valid or Non-Valid auxiliary command is written to the AUXMR, signaling a valid or invalid secondary address, respectively, to the TLC.

ADM0 and ADM1 must be cleared when either of the two programmable bits ton or lon is set.

Command Pass Through Register (CPTR)

VMEbus Address: Port Address + 11B (hex)

Attributes: Read Only, Internal to TLC

7	6	5	4	3	2	1	0	R
CPT7	CPT6	CPT5	CPT4	CPT3	CPT2	CPT1	CPT0	

Bit **Mnemonic** **Description**

7-0r CPT[7-0] Command Pass Through Bits 7 through 0

These bits are used to transfer undefined multiline GPIB command messages from the GPIB DIO lines to the computer. When the CPT feature is enabled (CPT ENABLE=1, AUXRB[0]w), any GPIB Primary Command Group (PCG) message not decoded by the TLC is treated as an undefined command. The multiline GPIB commands recognized by the µPD7210 are listed in Table 4-4. All GPIB Secondary Command Group (SCG) messages following an undefined GPIB PCG message are also treated as undefined. When an undefined GPIB message is encountered, it is held in the CPTR and the TLC Acceptor Handshake function is held off (in ACDS) until the Valid auxiliary command is written to the AUXMR. The CPTR is also used to inspect secondary addresses when mode 3 addressing is used. The TLC Acceptor Handshake function is held off (in ACDS) until the Valid or Non-Valid auxiliary command is written to the AUXMR.

Table 4-4. Multiline GPIB Commands Recognized by the µPD7210

Hex Number	Message	Description
01	GTL	Go To Local
04	SDC	Selected Device Clear
05	PPC	Parallel Poll Configure
08	GET	Group Execute Trigger
09	TCT	Take Control
11	LLO	Local Lockout
14	DCL	Device Clear

(continues)

Bit Mnemonic Description

Table 4-4. Multiline GPIB Commands Recognized by the µPD7210 (continued)

Hex Number	Message	Description
15	PPU	Parallel Poll Unconfigure
18	SPE	Serial Poll Enable
19	SPD	Serial Poll Disable
20-3E	MLA	My Listen Address
3F	UNL	Unlisten
40-5E	MTA	My Talk Address
5F	UNT	Untalk
60-6F	MSA,PPE	My Secondary Address or Parallel Poll Enable
70-7E	MSA,PPD	My Secondary Address or Parallel Poll Disable

The CPTR is read during a TLC-initiated Parallel Poll operation to fetch the Parallel Poll response. The PPR message is latched into the CPTR when CPPS is set, until CIDS is set, or until a command byte is sent over the GPIB.

Auxiliary Mode Register (AUXMR)

VMEbus Address: Port Address + 11B (hex)

Attributes: Write Only, Internal to TLC
Permits Access to Hidden Registers

7	6	5	4	3	2	1	0
CNT2	CNT1	CNT0	COM4	COM3	COM2	COM1	COM0
							W

The Auxiliary Mode Register (AUXMR) is used to issue auxiliary commands. It is also used to program the five hidden registers:

- Auxiliary Register A (AUXRA)
- Auxiliary Register B (AUXRB)
- Parallel Poll Register (PPR)
- Auxiliary Register E (AUXRE)
- Internal Counter Register (ICR)

Table 4-2 shows the control and command codes used.

Bit	Mnemonic	Description
7-5w	CNT[2-0]	Control Code Bits 2 through 0 These bits specify the control code (that is, the manner in which the information in bits COM[4-0] is to be used). If CNT[2-0] are all zero, then the special command selected by COM[4-0] is executed. Otherwise the hidden register selected by CNT[2-0] is loaded with the data from COM[4-0].
4-0w	COM[4-0]	Command Code Bits 4 through 0 These bits specify the command code of the special function if the control code is 000. Table 4-5 is a summary of the implemented special functions. Table 4-6 explains the details of each special function. If the control code is not 000, then these bits are written to one of the hidden registers (indicated by the control code in CNT[2-0]).

Table 4-5. Auxiliary Command Summary

Function Code* (COM4-COM0) 4 3 2 1 0	**Hex Code****	**Auxiliary Command**
0 0 0 0 0	00	Immediate Execute pon
0 0 0 1 0	02	Chip Reset
0 0 0 1 1	03	Finish Handshake
0 0 1 0 0	04	Trigger
0 0 1 0 1	05	Return to Local
0 0 1 1 0	06	Send EOI
0 0 1 1 1	07	Non-Valid Secondary Command or Address
0 1 1 1 1	0F	Valid Secondary Command or Address
0 0 0 0 1 0 1 0 0 1	01 09	Clear Parallel Poll Flag Set Parallel Poll Flag
1 0 0 0 1 1 0 0 1 0 1 1 0 1 0	11 12 1A	Take Control Asynchronously (Pulsed) Take Control Synchronously Take Control Synchronously on End
1 0 0 0 0	10	Go To Standby
1 0 0 1 1 1 1 0 1 1 1 1 1 0 0	13 1B 1C	Listen Listen in Continuous Mode Local Unlisten
1 1 1 0 1	1D	Execute Parallel Poll
1 1 1 1 0 1 0 1 1 0	1E 16	Set IFC Clear IFC
1 1 1 1 1 1 0 1 1 1	1F 17	Set REN Clear REN
1 0 1 0 0	14	Disable System Control

* CNT[2-0] set to 000 binary
** Represents all eight bits of the Auxiliary Mode Register

Table 4-6 shows the functions that are executed when the AUXMR Control Code (CNT2-CNT0) is loaded with 000 (binary) and the Command Code (COM4-COM0) is loaded.

Table 4-6. Auxiliary Commands: Detail Description

Command Code (COM4-COM0) 4 3 2 1 0	**Description**
0 0 0 0 0	Immediate Execute Pon This command generates a local pon message that places the following GPIB interface functions into these idle states: AIDS Acceptor Idle State CIDS Controller Idle State LIDS Listener Idle State LOCS Local State LPIS Listener Primary Idle State NPRS Negative Poll Response State PPIS Parallel Poll Idle State PUCS Parallel Poll to Unaddressed to Configure State SIDS Source Idle State SIIS System Control Interface Clear Idle State SPIS Serial Poll Idle State SRIS System Control Remote Enable Idle State TIDS Talker Idle State TPIS Talker Primary Idle State If the command is sent while a pon message is already active (by either an external reset pulse or the Chip Reset auxiliary command) the local pon message becomes false.
0 0 0 1 0	Chip Reset The Chip Reset command resets the TLC in the same way as an external reset pulse. The System Controller bit is also cleared. The TLC is reset to the following conditions: • The local pon message is set and the interface functions are placed in their idle states. • All bits of the SPMR are cleared. • The EOI bit is cleared. • All bits of the AUXRA, AUXRB, and AUXRE are cleared. • The Parallel Poll Flag and RSC local message are cleared. • The contents of the ICR is set to eight (F3 set to 1; F2, F1, and F0 set to 0). • The TRM0 bit and the TRM1 bit are cleared. The interface functions are held in their idle states until released by an Immediate Execute pon command. Between these commands, the TLC unitable bits may be programmed to their desired states.

(continues)

Table 4-6. Auxiliary Commands: Detail Description (continues)

Command Code (COM4-COM0) 4 3 2 1 0	Description
0 0 0 1 1	Finish Handshake (FH) The Finish Handshake command finishes a GPIB Handshake that was stopped because of a Holdoff on RFD or DAC.
0 0 1 0 0	Trigger **Note:** Trigger cannot be used with the GPIB-1014D. The Trigger command generates a high pulse on the TRIG pin (T/R3 pin when TRM1=0) of the TLC. The Trigger command performs the same function as if the DET (Device Trigger) bit (ISR1[5]r) were set. (The DET bit is not set by issuing the Trigger command.)
0 0 1 0 1 0 1 1 0 1	Return to Local (rtl) Return to Local (rtl) The two Return to Local commands implement the rtl message as defined by IEEE-488. When COM3 is zero, the message is generated in the form of a pulse. When COM3 is one, the rtl command is set in the standard manner.
0 0 1 1 0	Send EOI (SEOI) The Send EOI command causes the GPIB End Or Identify (EOI) line to go true with the next byte transmitted. The EOI line is then cleared upon completion of the Handshake for that byte. The TLC recognizes the Send EOI command only if TA=1 (that is, the TLC is addressed as the GPIB Talker).
0 0 1 1 1	Non-Valid Secondary Command or Address The Non-Valid command releases the GPIB DAC message held off by the Address Pass Through (APT). The TLC is permitted to operate as if an Other Secondary Address (OSA) message has been received.
0 1 1 1 1	Valid Secondary Command or Address The Valid command releases the GPIB DAC message held off by APT and allows the TLC to function as if a My Secondary Address (MSA) message had been received. The DAC message is released at the time of Command Pass Through (CPT). DAC is also released if DCAS or DTAS is in Holdoff state.

(continues)

Table 4-6. Auxiliary Commands: Detail Description (continues)

Command Code (COM4-COM0) 4 3 2 1 0	**Description**
0 0 0 0 1 0 1 0 0 1	Clear Parallel Poll Flag Set Parallel Poll Flag These commands set the Parallel Poll Flag to the value of COM3. The value of the Parallel Poll Flag is used as the local message ist when bit four of Auxiliary Register B is zero. The value of SRQS is used as the ist when ISS=1.
1 0 0 0 0	Go To Standby The Go To Standby command sets the local message gts if the TLC is in Controller Active State (CACS) or when it enters CACS. When the TLC leaves CACS, gts is cleared.
1 0 0 0 1	Take Control Asynchronously The Take Control Asynchronously command pulses the local message tca.
1 0 0 1 0	Take Control Synchronously The Take Control Synchronously command sets the local message tcs. The local message tcs is effective only when the TLC is in Controller Standby State (CSBS) or Controller Synchronous Wait State (CSWS). The local message tcs is cleared when the TLC enters Controller Active State (CACS).
1 1 0 1 0	Take Control Synchronously on END The Take Control Synchronously on END command sets the local message tcs when the data block transfer End message (END bit equal to one) is generated at CSBS. The tcs message is cleared when the TLC enters CACS.
1 0 0 1 1	Listen The listen command generates the local message ltn in the form of a pulse.

(continues)

Table 4-6. Auxiliary Commands: Detail Description (continues)

Command Code (COM4-COM0) 4 3 2 1 0	**Description**
1 1 0 1 1	Listen in Continuous Mode The Listen in Continuous Mode command generates the local message ltn in the form of a pulse and places the TLC in continuous mode. In continuous mode, the local message rdy is issued when the Acceptor Not Ready State (ANRS) is initiated unless data block transfer end is detected (END RX bit equals one). When END is detected, the TLC is placed in the RFD Holdoff state, preventing generation of the rdy message. In continuous mode, the DI bit is not set when a data byte is received. The continuous mode caused by the Listen in Continuous Mode command is released when the Listen auxiliary command is issued or the TLC enters the Listener Idle State (LIDS).
1 1 1 0 0	Local Unlisten The Local Unlisten command generates the local message lun in the form of a pulse.
1 1 1 0 1	Execute Parallel Poll The Execute Parallel Poll command sets the local message Request Parallel Poll (rpp). The rpp message is cleared when the TLC enters either Controller Parallel Poll State (CPPS) or Controller Idle State (CIDS). The transition of the TLC interface function is not guaranteed if the local messages rpp and Go To Standby (gts) are issued simultaneously when the TLC is in Controller Active State (CACS) and Source Transfer State (STRS) or Source Delay State (SDYS).
1 1 1 1 0 1 0 1 1 0	Set IFC Clear IFC These commands generate the local message request system control (rsc) and set Interface Clear (IFC) to the value of COM3. These commands should only be issued if the System Controller (SC) bit in CFG2 is set; that is, the GPIB-1014D is SC. To meet the IEEE-488 requirements, you must not issue the Clear IFC command until IFC has been held true for at least 100 µsec.

(continues)

Table 4-6. Auxiliary Commands: Detail Description (continued)

Command Code (COM4-COM0) 4 3 2 1 0	**Description**
1 1 1 1 1 1 0 1 1 1	Set REN Clear REN These commands generate the local message rsc and set REN to the value in COM3. These commands should only be issued if the SC bit in CFG2 is set; that is, the GPIB-1014D is SC. To meet IEEE-488 requirements, you must not issue the Set REN command until REN has been held false for at least 100 µsec.
1 0 1 0 0	Disable System Control The Disable System Control command clears the local message rsc and clears the SC bit.

Hidden Registers

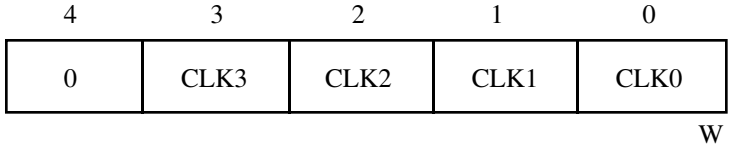
The hidden registers are loaded through the Auxiliary Mode Register (AUXMR). AUXMR[7-5] is loaded with the hidden register number, and AUXMR[4-0] is loaded with the data to be transferred to the hidden register. The hidden registers cannot be read, and in some cases the contents can only be set; that is, they can be cleared or reset to initialized conditions only by issuing the Chip Reset auxiliary command, or by a pon. Figure 4-3 shows the five hidden registers and illustrates how they are loaded with data from the AUXMR.

Internal Counter Register (ICR)

VMEbus Address: Port Address + 11B (hex)

AUXMR Control Code: 001 (Binary, Bits 7 - 5)

Attributes: Write Only, Internal to TLC
Accessed through AUXMR



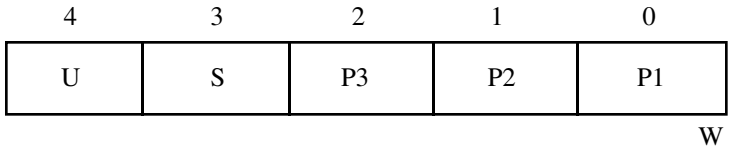
Bit	Mnemonic	Description
4w	0	Reserved Bit Write zero to this bit.
3-0w	CLK[3-0]	Clock Bits 3 through 0 The contents of the ICR are used to divide internal counters that generate TLC state change delay times used by the IEEE-488 specification. The most familiar of these delay times, T1, is the minimum delay between placing the data or command bytes on the GPIB DIO lines and asserting DAV. These delay times vary depending on the type of transfer in progress and the value of the AUXRB bit TRI. To mee all GPIB timing specifications, ICR should be set to eight because the TLC is clocked at 8 MHz. Because the T1 delay time has a direct impact on performance, you may want to reduce the value of the ICR to reduce the T1 delay. At an ICR value equal to eight, the T1 delay is greater than 700 nsec by measurement. By lowering the value of ICR to five during a GPIB write operation, the T1 delay is reduced to 500 nsec, which is the GPIB specification requirement, and performance is maximized. You should return the value of the ICR to eight after the GPIB write operation completes.

Parallel Poll Register (PPR)

VMEbus Address: Port Address + 11B (hex)

AUXMR Control Code: 011 (Binary, Bits 7 - 5)

Attributes: Write Only, Internal to TLC
Accessed through AUXMR



Writing to the Parallel Poll Register (PPR) is done via the AUXMR. Writing the binary value 011 into the Control Code (CNT[2-0]) and writing a bit pattern into the command code portion (COM[4-0]) of the AUXMR causes the command code to be written to the PPR. When COM[4-0] is written to the PPR, the bits are named as shown in the PPR. This 5-bit command code determines the manner in which the TLC responds to a Parallel Poll.

When using the remote Parallel Poll Configure (IEEE-488 capability code PP1), do not write to the PPR. The TLC implements remote configuration fully and automatically without software assistance. The hardware recognizes, interprets, and responds to Parallel Poll Configure (PPC), Parallel Poll Enable (PPE), Parallel Poll Disable (PPD), and Identify (IDY) messages. The user need only set or clear the individual status (ist) message (using Set/Clear Parallel Poll Flag auxiliary commands) according to pre-established system protocol convention. Writing to the PPR after it is remotely configured will corrupt the configuration.

When using the local PPC (capability code PP2), a valid PPE or PPD message must be written to the PPR prior to the poll.

Bit	Mnemonic	Description
4w	U	Unconfigure Bit The U bit determines whether or not the TLC participates in a Parallel Poll. If U=0, the TLC participates in Parallel Polls and responds in the manner defined by PPR[3] through PPR[0] and by ist. If U=1, the TLC does not participate in a Parallel Poll. The U bit is equivalent to the Local Poll Enable, active low (lpe*) message. When U=0, S and P[3-1] mean the same as the bit of the same name in the PPE message, and the I/O write operation (to the PPR) is the same as the receipt of the PPE message from the GPIB Controller. When U=1, S and P[3-1] do not carry any meaning, but they must be cleared.

Bit	Mnemonic	Description
3w	S	Status Bit Polarity (Sense) Bit The S bit is used to indicate the polarity (or sense) of the TLC local ist message. If S=1, the status is *in phase*, meaning that if, during a Parallel Poll response, S=ist=1, and U=0, the TLC responds to the Parallel Poll by driving one of the eight GPIB DIO lines low, thus asserting it to a logic one. If S=1 and ist=0, the TLC does not drive the DIO line. If S=0, the status is *in reverse phase*, meaning that if, during a Parallel Poll, ist=0, and U=0, the TLC responds to the Parallel Poll by driving one of the eight GPIB DIO lines low. If S=0 and ist=1, the TLC does not drive the DIO line. For more information, refer to *Auxiliary Register B* and the *Clear Parallel Poll Flags/Set Parallel Poll Flags* later in this section.
2-0w	P[3-1]	Parallel Poll Response Bits 3 through 1 PPR bits 3 through 1, designated P[3-1], contain an encoded version of the Parallel Poll response. P[3-1] indicate which of the eight DIO lines is asserted during a Parallel Poll (equal to N-1). The GPIB-PC normally drives the GPIB DIO lines using three-state drivers. During Parallel Poll responses, however, the drivers automatically convert to Open Collector mode, as required by IEEE-488. For example, if P[3-1]=010 (binary), GPIB DIO line DIO3* is driven low (asserted) if the GPIB-PC is Parallel Polled (and S=ist).

Some examples of configuring the Parallel Poll Register are listed in Table 4-7:

Table 4-7. Examples of Configuring the Parallel Poll Register

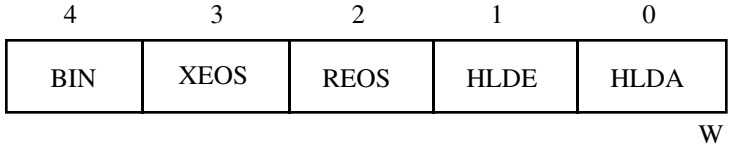
Written to the AUXMR								
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**Result**
0	1	1	1	0	0	0	0	Unconfigures PPR
0	1	1	0	0	0	0	0	0 0 0 0 0 is written to the PPR. The GPIB-1014D participates in a Parallel Poll, asserting the DIO1 line if ist=0. Otherwise, the GPIB-1014D does not participate.
0	1	1	0	1	0	0	1	0 1 0 0 1 is written to the PPR. The GPIB-1014D participates in a Parallel Poll, asserting the DIO2 line if ist=1. Otherwise, the GPIB-1014D does not participate.

Auxiliary Register A (AUXRA)

VMEbus Address: Port Address + 11B (hex)

AUXMR Control Code: 100 (Binary, Bits 7 - 5)

Attributes: Write Only, Internal to TLC
Accessed through AUXMR



Writing to Auxiliary Register A (AUXRA) is done via the AUXMR. Writing the binary value 100 into the Control Code (CNT[2-0]) and a bit pattern into the Command Code (COM[4-0]) portion of the AUXMR causes the Command Code to be written to AUXRA. When the data is written to AUXRA, the bits are denoted by the mnemonics shown in the register bit map above. This 5-bit code controls the data transfer messages Holdoff and EOS/END.

Bit	Mnemonic	Description
4w	BIN	Binary Bit The BIN bit selects the length of the EOS message. Setting BIN causes the End Of String Register (EOSR) to be treated as a full 8-bit byte. When BIN=0, the EOSR is treated as a 7-bit register (for ASCII characters) and only a 7-bit comparison is done with the data on the GPIB.
3w	XEOS	Transmit END with EOS Bit The XEOS bit permits or prohibits automatic transmission of the GPIB END message at the same time as the EOS message when the TLC is in Talker Active State (TACS). If XEOS is set and the byte in the CDOR matches the contents of the EOSR, the EOI line is sent true along with the data.
2w	REOS	END on EOS Received Bit The REOS bit permits or prohibits setting the END bit (ISR1[4]r) at reception of the EOS message when the TLC is in Listener Active State (LACS). If REOS is set and the byte in the DIR matches the byte in the EOSR, the END bit (ISR1[4]r) is set.

Bit	Mnemonic	Description
1-0w	HLDE HLDA	Holdoff on End Bit Holdoff on All Bit

HLDE and HLDA together determine the GPIB data receiving mode. The four possible modes are as follows:

HLDE	HLDA	Data Receiving Mode
0	0	Normal Handshake
0	1	RFD Holdoff on All Data
1	0	RFD Holdoff on END
1	1	Continuous

In Normal Handshake mode, the local message rdy is generated when data is received from the GPIB. When the received data is read from the DIR, rdy is generated in Acceptor Not Ready State (ANRS), the RFD message is transmitted, and the GPIB Handshake continues.

In RFD Holdoff on All Data (HLDA) mode, RFD is not sent true after data is received until the Finish Handshake (FH) auxiliary command is issued. Unlike Normal Handshake mode, the RFD HLDA mode does not generate the rdy message even if the received data is read through the DIR; that is, the GPIB RFD message is not generated.

In RFD Holdoff on End mode, operation is the same as the RFD HLDA mode, but only when the end of the data block (EOS or END message) is detected; that is, the END message is received or, if REOS is set, the EOS character is received. Handshake Holdoff is released by the FH auxiliary command.

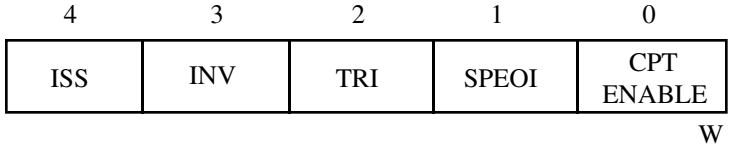
In continuous mode, the rdy message is generated when in ANRS until the end of the data block is detected. A Holdoff is generated at the end of a data block. The FH auxiliary command must be issued to release the Holdoff. The continuous mode is useful for monitoring the data block transfer without actually participating in the transfer (no data reception). In continuous mode, the DI bit (ISR1[0]r) is not set by the reception of a data byte.

Auxiliary Register B (AUXRB)

VMEbus Address: Port Address + 11B (hex)

AUXMR Control Code: 101 (Binary, Bits 7 - 5)

Attributes: Write Only, Internal to TLC
Accessed through AUXMR



Writing to Auxiliary Register B (AUXRB) is done via the AUXMR. Writing the value 101 into the Control Code (CNT[2-0]) and a bit pattern into the Command Code portion (COM[4-0]) of the AUXMR causes the Command Code to be written to AUXRB. When the data is written to AUXRB, the bits are denoted as shown in the figure above. This 5-bit code affects several interface functions, as described in the following paragraphs.

Bit	Mnemonic	Description
4w	ISS	Individual Status Select Bit The ISS bit determines the value of the TLC ist. When ISS=1, ist becomes the same value as the TLC Service Request State (SRQS). (The TLC is asserting the GPIB SRQ message when it is in SRQS.) When ISS=0, ist takes on the value of the TLC Parallel Poll Flag. The Parallel Poll Flag is set and cleared using the Set Parallel Poll Flag and Clear Parallel Poll Flag auxiliary commands.
3w	INV	Invert Bit The INV bit affects the polarity of the TLC INT pin. Setting INV causes the polarity of the Interrupt (INT) pin on the TLC to be active low. As implemented on the GPIB-1014D, configuring the INT pin to active low results in interrupt request errors. Consequently, INV must always be cleared and must never be set except for diagnostic purposes. INV = 0 : INT pin is active high INV = 1 : INT pin is active low

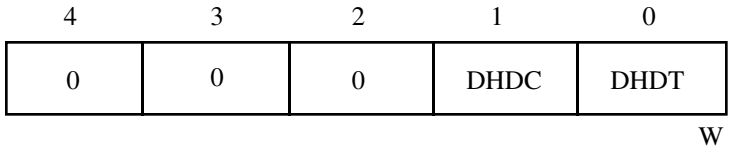
Bit	Mnemonic	Description
2w	TRI	Three-State Timing Bit The TRI bit determines the TLC GPIB Source Handshake Timing, T1. TRI can be set to enable high speed data transfers (T1 ≥ 500 nsec) when tri-state GPIB drivers are used. (The GPIB-1014D uses tri-state GPIB drivers except during Parallel Poll responses, in which case the GPIB drivers automatically switch to Open Collector.) Setting TRI enables high-speed timing as T1 of the GPIB Source Handshake after transmission of the first byte. Clearing TRI sets the low-speed timing (T1 ≥ 2 μsec).
1w	SPEOI	Send Serial Poll EOI Bit The SPEOI bit permits or prohibits the transmission of the END message in Serial Poll Active State (SPAS). If SPEOI is set, EOI is sent true when the TLC is in SPAS; otherwise, EOI is sent false in SPAS.
0w	CPT ENABLE	Command Pass Through Enable Bit The CPT ENABLE bit permits or prohibits the detection of undefined GPIB commands and permits or prohibits the setting of the CPT bit (ISR1[7]r) on receipt of an undefined command. When CPT ENABLE is set and an undefined command has been received, the DAC message is held and the Handshake stops until the Valid auxiliary command is issued. The undefined command can be read from the CPTR and processed by the software.

Auxiliary Register E (AUXRE)

VMEbus Address: Port Address + 11B (hex)

AUXMR Control Code: 110 (Binary, Bits 7 - 5)

Attributes: Write Only, Internal to TLC
Accessed through AUXMR



Writing to Auxiliary Register E (AUXRE) is done via the AUXMR. Writing the binary value 110 into the Control Code (CNT[2-0]) and a bit pattern into the the lower five bits of the AUXMR (COM[4-0]) causes the two lowest order bits to be written to AUXRE. The 2-bit code, DHDC and DHDT, determines how the TLC may be placed into DAC Holdoff.

Bit	Mnemonic	Description
4-2w	0	Reserved Bits Write zeros to these bits.
1w	DHDC	DAC Holdoff on DCAS Bit Setting DHDC enables DAC Holdoff when the TLC enters Device Clear Active State (DCAS). Clearing DHDC disables DAC Holdoff on DCAS. Issuing the Finish Handshake auxiliary command releases the Holdoff.
0w	DHDT	DAC Holdoff on DTAS Bit Setting DHDT enables DAC Holdoff when the TLC enters Device Trigger Active State (DTAS). Clearing DHDT disables DAC Holdoff on DTAS. Issuing the Finish Handshake auxiliary command releases the Holdoff.

Address Register 0 (ADR0)

VMEbus Address: Port Address + 11D (hex)

Attributes: Read Only, Internal to TLC

7	6	5	4	3	2	1	0 R
X	DT0	DL0	AD5-0	AD4-0	AD3-0	AD2-0	AD1-0

Address Register 0 (ADR0) reflects the internal GPIB address status of the TLC as configured using the ADMR. In addressing mode 2, ADR0 indicates the address and enable bits for the primary GPIB address of the TLC. In dual primary addressing (Modes 1 and 3) ADR0 indicates the TLC major primary GPIB address. Refer to the description of the *Address Mode Register* in this section for information on addressing modes.

Bit	Mnemonic	Description
7r	X	Don't Care Bit Reads as a zero or one.
6r	DT0	Disable Talker 0 Bit If DT0 is set, it indicates that the mode 2 primary (or mode 1 and 3 major) Talker is not enabled; that is, the TLC does not respond to a GPIB talk address matching AD[5-0 – 1-0]. If DT0=0, the TLC responds to a GPIB talk address matching bits AD[5-0 – 1-0].
5r	DL0	Disable Listener 0 Bit If DL0 is set, it indicates that the mode 2 primary (or mode 1 and 3 major) Listener is not enabled; that is, the TLC does not respond to a GPIB listen address matching bits AD[5-0 – 1-0]. If DL0=0, the TLC responds to a GPIB listen address matching bits AD[5-0 – 1-0].
4-0r	AD[5-0 – 1-0]	Mode 2 Primary TLC GPIB Address Bits 5-0 through 1-0 These are the lower 5 bits of the TLC GPIB primary (or major) address. The primary talk address is formed by adding hex 40 to AD[5-0 – 1-0], while the listen address is formed by adding hex 20.

Address Register (ADR)

VMEbus Address: Port Address + 11D (hex)

Attributes: Write Only, Internal to TLC

7	6	5	4	3	2	1	0
ARS	DT	DL	AD5	AD4	AD3	AD2	AD1
							W

The Address Register (ADR) is used to load the internal registers ADR0 and ADR1. Both ADR0 and ADR1 must be loaded for all addressing modes.

Bit	Mnemonic	Description
7w	ARS	Address Register Select Bit ARS is zero or one to select whether the seven low-order bits of ADR must be loaded into internal registers ADR0 or ADR1, respectively.
6w	DT	Disable Talker Bit DT must be set if recognition of the GPIB talk address formed from AD5 through AD1 (ADR[4-0]w) is not to enable.
5w	DL	Disable Listener Bit DL should be set if recognition of the GPIB listen address formed from AD5 through AD1 (ADR[4-0]w) is not to enable.
4-0w	AD[5-1]	TLC GPIB Address Bits 5 through 1 These bits specify the five low-order bits of the GPIB address that is to be recognized by the TLC. The corresponding GPIB talk address is formed by adding hex 40 to AD[5-1], while the corresponding GPIB listen address is formed by adding hex 20. The value written to AD[5-1] must not be all ones; otherwise, the corresponding talk and listen addresses would conflict with the GPIB Untalk (UNT) and GPIB Unlisten (UNL) commands.

Address Register 1 (ADR1)

VMEbus Address: Port Address + 11F (hex)

Attributes: Read Only, Internal to TLC

7	6	5	4	3	2	1	0 R
EOI	DT1	DL1	AD5-1	AD4-1	AD3-1	AD2-1	AD1-1

Address Register 1 (ADR1) indicates the status of the GPIB address and enable bits for the secondary address of the TLC if mode 2 addressing is used, or the minor primary address of the TLC if dual primary addressing is used (modes 1 and 3). If mode 1 addressing is used and only a single primary address is needed, then both the talk and listen addresses must disable in this register. If mode 2 addressing is used, then the talk and listen disable bits in this register must match those in ADR0.

Bit	Mnemonic	Description
7r	EOI	End Or Identify Bit EOI indicates the value of the GPIB EOI line latched when a data byte is received by the TLC GPIB Acceptor Handshake (AH) function. If EOI=1, the EOI line was asserted with the received byte. EOI is cleared by pon, or by using the Chip Reset auxiliary command. EOI is updated after each byte is received.
6r	DT1	Disable Talker 1 Bit If DT1 is set, the mode 2 secondary (or mode 1 and 3 minor) Talker function is not enabled; that is, the TLC does not respond to a secondary address (or minor primary talk address) formed from bits AD[5-1 – 1-1]. If DT1 is cleared, the secondary address is checked only if the TLC received its primary talk address (that is, is in TPAS).
5r	DL1	Disable Listener 1 Bit If DL1=1, the mode 2 secondary (or mode 1 and 3 minor) listen function is not enabled; that is, the TLC does not respond to a secondary address (or minor primary listen address formed from bits AD5-1 through AD1-1). If DL1 is cleared and the TLC received its primary listen address (that is, is in LPAS), the secondary address is checked.

Bit	Mnemonic	Description
4-0r	AD[5-1 – 1-1]	Mode 2 Secondary TLC GPIB Address Bits 5-1 through 1-1 These are the lower five bits of the TLC secondary or minor address. The secondary address is formed by adding hex A0 to bits AD[5-1 – 1-1]. The minor talk address is formed by adding hex 40 to AD[5-1 – 1-1], while the listen address is formed by adding a hex 20.

End Of String Register (EOSR)

VMEbus Address: Port Address + 11F hex

Attributes: Write Only, Internal to TLC

7	6	5	4	3	2	1	0
EOS7	EOS6	EOS5	EOS4	EOS3	EOS2	EOS1	EOS0
							W

The End Of String Register (EOSR) holds the byte used by the TLC to detect the end of a GPIB data block transfer. A 7- or 8-bit byte (ASCII or binary) may be placed in the EOSR to be used in detecting the end of a block of data. The length of the EOS byte to be used in the comparison is selected by the BIN bit in AUXRA (AUXRA[4w]).

If the TLC is a Listener and bit REOS of AUXRA is set, then the END bit is set in ISR1 whenever the byte in the DIR matches the EOSR. If the TLC is a Talker and the data is being transmitted, and bit XEOS of AUXRA is set, the END message (GPIB EOI* line asserted low) is sent along with the data byte whenever the contents of the CDOR match the EOSR.

Bit	Mnemonic	Description
7-0w	EOS[7-0]	End of String Bits 7 through 0

DMA Registers

The onboard DMA Controller is a 68450 DMAC. This chip is extremely flexible and contains four independent DMA channels. The DMAC can support single address (flyby) transfers or dual address (flowthrough) transfers. The GPIB-1014D uses all four channels (0 through 3) for 8-bit flyby DMA transfers between VMEbus memory and the two GPIB ports (A and B). All four channels are available for 8- or 16-bit flowthrough memory-to-memory DMA transfers when not being used for GPIB transfers. The DMAC supports unchained, continue, array chained, or linked chained operations between memory and memory or between memory and device (GPIB). The TLC-to-DMAC interface includes lines for requesting, acknowledging, and providing incidental control of the two onboard TLCs.

The DMAC contains a large number of internal configuration and status registers. These registers define and control the activity of the DMAC in processing a channel operation. The registers are addressed relative to the base address of the board. Locations not used in the board address space are reserved.

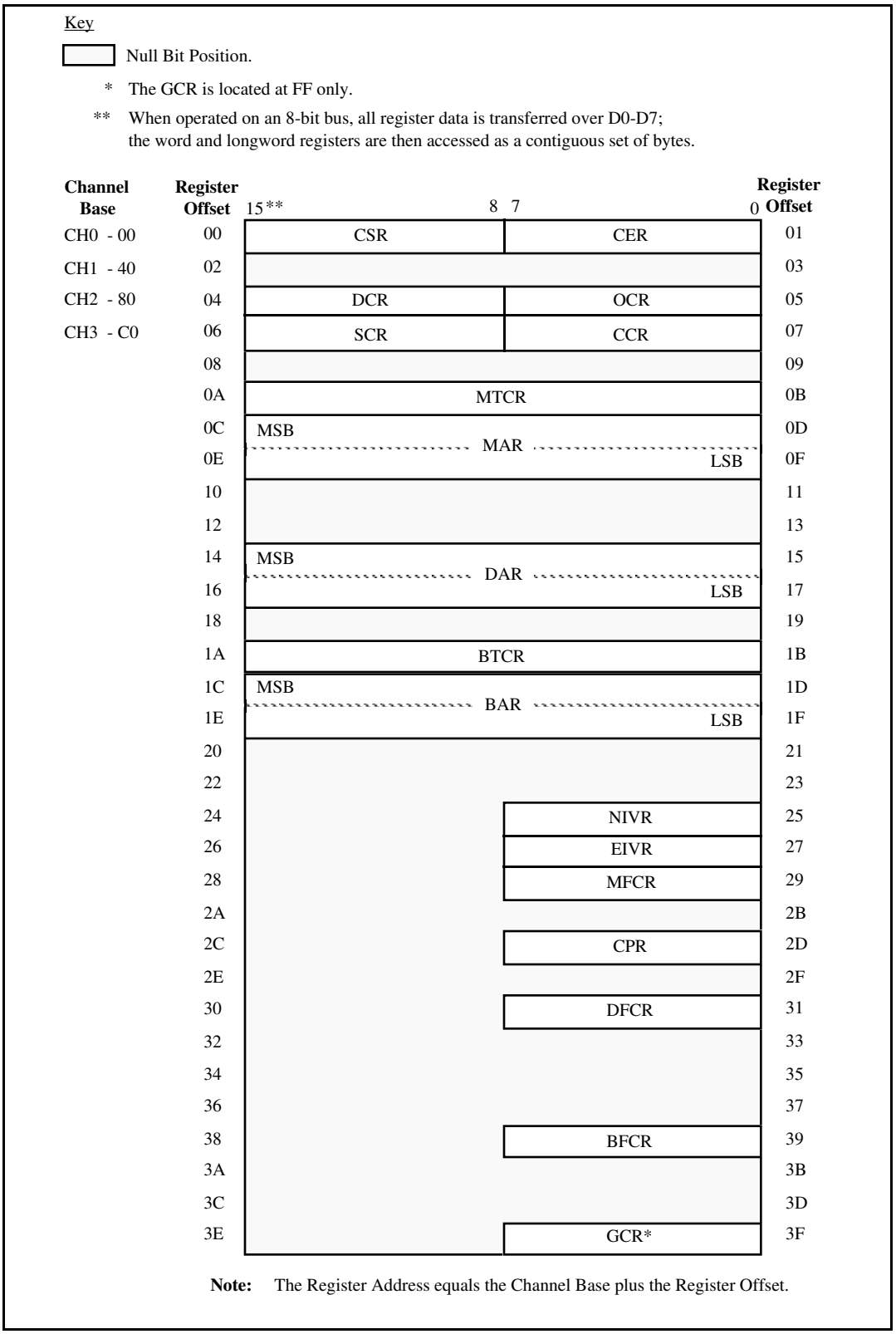
The register set associated with *each* DMA channel is shown in Table 4-8.

Table 4-8. DMAC DMA Channel Register Set

Register Name	Mnemonic	Size
Memory Address Register	MAR	32 bits
Memory Transfer Counter Register	MTCR	6 bits
Memory Function Code Register	MFCR	8 bits
Device Address Register	DAR	32 bits
Device Function Code Register	DFCR	8 bits
Base Address Register	BAR	32 bits
Base Transfer Counter Register	BTCR	16 bits
Base Function Code Register	BFCR	8 bits
Channel Status Register	CSR	8 bits
Channel Error Register	CER	8 bits
Device Control Register	DCR	8 bits
Operation Control Register	OCR	8 bits
Sequence Control Register	SCR	8 bits
Channel Control Register	CCR	8 bits
Channel Priority Register	CPR	8 bits
Normal Interrupt Vector Register	NIVR	8 bits
Error Interrupt Vector Register	EIVR	8 bits

The register set for each channel is addressed relative to the base address of the GPIB-1014D as outlined in Table 2-2. Figure 4-4 shows the DMA registers in order of their register offset.

Figure 4-4 is reprinted with permission of the copyright owner from the Motorola *MC68440 Dual-Channel Direct Memory Access Controller*, *Advance Information*, February 1984 Edition, p. 3-20. © Copyright 1984 Motorola, Inc.



Reprinted from the Motorola *MC68440 Advance Information* manual.

Figure 4-4. DMA Register Memory Map

The following paragraphs describe the channel configuration and status registers. More information on the MC68450 can be found in the *Motorola MC68450 DMAC Advance Information* or the *Hitachi Microsystem HD68450 DMAC (Direct Memory Access Controller)*. Each channel contains the same status and configuration registers.

Address Registers

The Memory Address Register (MAR), Base Address Register (BAR), and Device Address Register (DAR) are 32-bit registers. Although they are 32-bit registers, only the least significant 24 bits are provided to the VMEbus by the DMA Controller. The GPIB-1014D adds an additional 8-bit Page Register which contains information on the top eight address lines (A32 through A24).

The Memory Address Register (MAR) is used to hold the address of the VMEbus memory location, which is where the data is transferred. This register is used in all DMA operations, memory-to-memory or memory-to-device.

The Base Address Register (BAR) is used in continue, array-chained, and link-chained operations. In the continue mode of operation, the BAR holds the address of the next block of data to be transferred. At the end of the last block transfer, the content of BAR is automatically transferred to MAR and a next block transfer is started. In array-chained and link-chained modes of operation, the BAR is used as a pointer to a table in memory that holds the address(es) of the data block(s) to be transferred. Using the BAR in the chained modes of operation, the DMAC first reads (using DMA) the address of the next block to be transferred into the MAR, then starts the actual data transfer. In most GPIB-to-memory DMA transfers, the array chained mode of operation is used. For more information on this mode, see Chapter 5 under *Sending/Receiving Messages* and Chapter 6 under *Block Termination.*

The Device Address Register (DAR) is used to hold the device address when the DMA transfer is in dual address (flowthrough) mode. This mode is used for devices that cannot be implicitly addressed with an acknowledge (ACK) signal, but must be addressed via the address bus. Flowthrough transfers address both the memory and the device by executing two bus cycles and storing the data temporarily in an internal register. In single address (flyby) transfers, the device is implicitly addressed with an acknowledge signal, so the data transfer takes one cycle and the data is transferred directly between the device and memory. DMA transfers between the GPIB and VMEbus memory use only flyby mode, so the DAR is not used. General purpose memory-to-memory (Flowthrough) transfers use both the MAR and DAR to hold the source and destination VMEbus memory addresses.

While data transfers between the GPIB and VMEbus must use flyby mode, flowthrough memory-to-memory DMA transfers can be accomplished using any of the four available full-function DMA channels (channels 0 through 3). This not only makes the GPIB-1014D available as a general purpose VMEbus DMA Controller, but also enables comprehensive stand-alone diagnostics to be performed on the DMA circuitry without using the GPIB.

Transfer Count Registers

The Memory Transfer Counter Register (MTCR) and the Base Transfer Counter Register (BTCR) are 16-bit registers. The MTCR is used to specify how many operands will be transferred. (An operand can be either a byte (8 bits) or a word (16 bits).) This register is loaded prior to starting the channel and will be decremented with each operand transfer. When the contents of this register are zero and the operation is unchained (or the chain is exhausted), the channel has reached terminal count and the COC bit of the CSR is set. In continue mode of operation, the BTC holds the size of the next block to be transferred which is then transferred into the MTR when the last block is finished. In array chaining mode of operation, the BTC holds the number of memory blocks to be transferred. Linked chaining mode of operation does not use the BTC.

Function Code Registers

VMEbus Address: Port Address + 29 (hex) for Memory Function Code Register (MAR)
Port Address + 31 (hex) for Device Function Code Register (DAR)
Port Address + 39 (hex) for Base Function Code Register (BAR)

Attributes: Read/Write, Internal to DMAC

7	6	5	4	3	2	1	0	
X	X	X	X	X	M2	M1	M0	R/W

On each of the four DMAC channels, there are three Function Code Registers (FCRs) associated with the three address registers (MAR, DAR, and BAR). The three FCRs are MFCR, DFCR, and BFCR. These registers are three bits wide and are loaded via the lowest three bits of the data bus. During a DMA cycle, when the DMAC outputs the contents of one of the three address registers, the DMAC also outputs the associated FCR. The 3-bit value of an FCR along with switches W5, W6, and W8 determine the 6-bit Address Modifier (AM) codes on the VMEbus. The AM codes are used to identify what type of cycle (long, short, supervisor, user) is specified by the DMAC when the GPIB-1014D is the bus master.

Table 3-1 shows how to program bits M2 through M0 to produce AM codes supported by the default settings of W5, W6, and W8. Tables 3-2 shows how to program bits M2 through M0 to produce any arbitrary AM code.

Bit	Mnemonic	Description
7-3r/w	X	Don't Care Bits Read zeros or ones to these bits.
2r/w	M2	Supervisor/User Access Bit If this bit is a one, the GPIB-1014D indicates Supervisor access. If it is a zero, the GPIB-1014D indicates User access.
1r/w	M1	Standard/Short Addressing Bit If this bit is a one, the GPIB-1014D indicates a standard 24-bit cycle. If it is a zero, the GPIB-1014D indicates a 16-bit short I/O cycle.
0r/w	M0	Program/Data Access Bit If this bit is a one, the GPIB-1014D accesses the program area. If it is a zero, the GPIB-1014D accesses the data area.

Device Control Register (DCR)

VMEbus Address: Port Address + 004 (hex)

Attributes: Read/Write, Internal to DMAC

7 6	5 4	3	2	1 0	
XRM	DTYP	DPS	0	PCL	R/W

The Device Control Register (DCR) is a device-oriented control register.

Bit	Mnemonic	Description
7-6r/w	XRM	External Request Mode Bits 7 through 6

The External Request Mode bits specify whether the channel is in cycle steal or cycle steal with hold transfer mode. These two modes are used in all GPIB applications. Burst mode is not used in GPIB-1014D GPIB data transfers, but may be used in memory-to-memory transfers.

00 = Burst Transfer Mode
10 = Cycle Steal Mode
01 = Undefined, Reserved
11 = Cycle Steal with Hold Mode

5-4r/w DTYP — Device Type Bits 5 through 4

The Device Type bits specify what type of device is on the channel. For the GPIB-1014D GPIB application, set the device type to 10 (device with ACK). For memory-to-memory transfers, set the device type to 00 (68000 compatible).

00 = 68000 compatible, explicitly addressed
01 = 6800 compatible, explicitly addressed
10 = Device with ACK, implicitly addressed
11 = Device with ACK and READY.

3r/w DPS — Device Port Size Bit

The Device Port Size bit indicates the size of the device port. For GPIB-1014D GPIB transfers, the device port size is 8 bits. For memory-to-memory transfers, the device port size can be 8 or 16 bits.

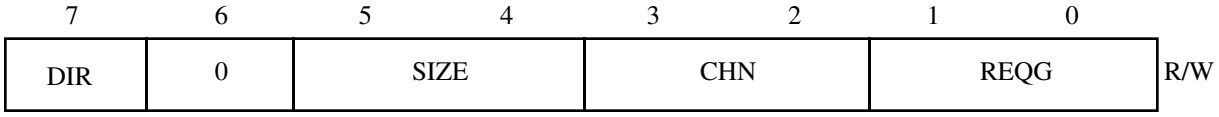
0 = 8-bit port
1 = 16-bit port

Bit	Mnemonic	Description
2r/w	0	Reserved Bit Write a zero to this bit.
1-0r/w	PCL	Peripheral Control Line Bits 1 through 0 Each of the four DMAC channels has a Peripheral Control Line (called PCL0* through PCL3*). The two PCL bits define the function of each line. The GPIB-1014D uses the four lines as status input. On PCL0*, GPIB signal SRQ* of Port A is connected. On PCL2*, signal REN* of Port A or signal SRQ* of Port B is connected (switch selectable). If programmed as status input, you can determine the state (high or low) of these two GPIB signals by reading the PCS bit in the appropriate CSR. For Port A, PCL1* is designed to detect an interrupt from the TLC, synchronization of the GPIB, and a bus error during a DMA transfer. For Port B, PCL3* is used in the same manner. PCL1* and PCL3* must be set to 01 (status input with interrupt) if interrupts are used, or 00 (status input) if polling is used. This is described in more detail in Chapter 5 under *Interrupts*. 00 = Status Input (can be read by reading CSR) 01 = Status Input with Interrupt 10 = Start Output Pulse, Negative 1/8 CLK 11 = Abort Input

Operation Control Register (OCR)

VMEbus Address: Port Address + 005 (hex)

Attributes: Read/Write, Internal to DMAC



The Operation Control Register (OCR) is an operation oriented register.

Bit	Mnemonic	Description
7r/w	DIR	Direction Bit The Direction bit specifies the direction of the transfer, to or from VMEbus memory : 0 = Transfer from memory to device 1 = Transfer from device to memory In GPIB applications, 0 indicates transfers from memory-to-GPIB and 1 indicates transfers from GPIB-to-memory.
6r/w	0	Reserved Bit Write zero to this bit.
5-4r/w	SIZE	Size Bits 5 through 4 The Size bits indicate the size of the data transfer. For the GPIB-1014D GPIB transfers, the size is always byte 00. For memory-to-memory transfers, the size can be byte, word or longword. 00 = Byte (8 bits) 01 = Word (16 bits) 10 = Long-word (32 bits) 11 = (undefined, reserved)
3-2r/w	CHN	Chaining Mode Bits 3 through 2 The Chain bits are used to indicate what type of chaining, if any, is used: 00 = Chain operation is disabled 01 = (undefined, reserved) 10 = Array Chaining 11 = Linked Chaining

Bit	Mnemonic	Description
		In most GPIB applications, either no chaining or array chaining is used. See Chapter 5 for details.
1-0r/w	REQG	DMA Request Generation Bits 1 through 0

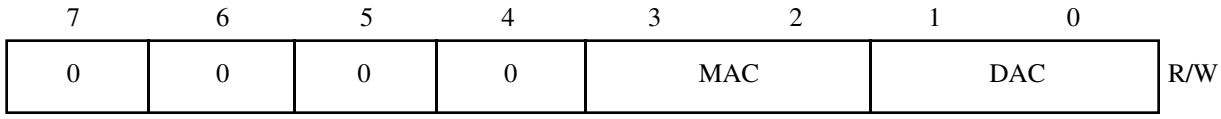
The DMA Request Generation method bits define how requests for transfers are generated. For the GPIB-to-memory DMA transfers, the request mode is always 10 (the REQ line initiates an operand transfer). For memory-to-memory transfers, automatic request mode must be used.

00 = Automatic request at a rate limited by the General Control Register (GCR).
01 = Automatic request at maximum rate.
10 = REQ line initiates an operand transfer.
11 = Automatic request the first operand, external request the remaining operands.

Sequence Control Register (SCR)

VMEbus Address: Port Address + 006 (hex)

Attributes: Read/Write, Internal to DMAC



The Sequence Control Register (SCR) is used to define the sequencing of memory and device addresses.

Bit	Mnemonic	Description
7-4r/w	0	Reserved Bits Write zeros to these bits.
3-2r/w	MAC	Memory Address Count Bits 3 through 2 The Memory Address Count bits indicate the count sequence of the Memory Address Register: 00 = Memory address does not count 01 = Memory address register counts up 10 = Memory address register counts down 11 = (undefined, reserved)
1-0r/w	DAC	Device Address Count Bits 1 through 0 The Device Address Count bits indicate the address sequence of the Device Address Register. This is only used in flowthrough memory-to-memory DMA transfers. 00 = Device address does not count 01 = Device address register counts up 10 = Device address register counts down 11 = (undefined, reserved)

Channel Control Register (CCR)

VMEbus Address: Port Address + 007 (hex)

Attributes: Read/Write, Internal to DMAC

7	6	5	4	3	2	1	0	
STR	CNT	HLT	SAB	EINT	0	0	0	R/W

This register is used to control the operation of the channel. By writing to this register, a channel operation can be started and set to be continued, halted, and aborted. Also, the channel can be enabled to issue interrupts when an operation is terminated (normal or error termination). Setting the STR bit causes immediate activation of the channel; the channel will be ready to accept requests immediately. The STR and CNT bits of the register can not be reset by a write to the register. The software abort bit (SAB) can be used to terminate the operation. Setting the SAB resets the STR and CNT. Setting the HLT bit halts the channel and resetting the HLT bit resumes the operation.

Bit	Mnemonic	Description
7r/w	STR	Start Bit The Start bit is used to start the operation of the channel. 0 = No operation is pending 1 = Start operation
6r/w	CNT	Continue Bit The Continue bit is used to select the continue option. This bit must be set when the channel is active or at the same time you set the STR bit. This is not generally used for GPIB-1014D GPIB transfers. 0 = No continuation is pending 1 = Continue operation
5r/w	HLT	Halt Bit The Halt bit is used to temporarily halt channel operation. 0 = Operation not halted 1 = Operation halted
4r/w	SAB	Software Abort Bit The Software Abort bit is used to abort channel operation. 0 = Channel operation not aborted 1 = Abort channel operation

Bit	Mnemonic	Description
3r/w	EINT	Interrupt Enable Bit The Interrupt Enable bit in used to enable or disable interrupts from the channel. GPIB-1014D interrupts are discussed in more detail in Chapter 5. 0 = No interrupts enabled 1 = Interrupts enabled
2-0r/w	0	Reserved Bits Write zeros to these bits.

Channel Status Register (CSR)

VMEbus Address: Port Address + 000 (hex)

Attributes: Read/Write, Internal to DMAC

7	6	5	4	3	2	1	0	
COC	BTC	NDT	ERR	ACT	0	PCT	PCS	R/W

The Channel Status Register (CSR) contains the status of the channel. Bits are set automatically by DMAC. Bits are cleared by writing a one (1) to each register bit or by resetting the DMAC.

Bit	Mnemonic	Description
7r/w	COC	Channel Operation Complete bit The Channel Operation Complete bit is set if the DMA transfer has completed. This bit is set following the termination, whether successful or not, of any DMA operation. This bit must be cleared in order to start another channel operation. 0 = Channel operation incomplete 1 = Channel operation complete
6r/w	BTC	Block Termination Complete Bit The Block Termination Complete bit is set when the memory transfer count is exhausted, the operation is unchained, and the continue bit is set. This bit must be cleared before another continuation is attempted; otherwise, an operation timing error is signaled. See the *Continue Mode of Operation* section in Chapter 6 under *Operation Continuation* for more information. 0 = Block transfer incomplete 1 = Block transfer complete
5r/w	NDT	Normal Device Termination Bit The Normal Device Termination bit is set when the transfer operation is terminated by the device. This is not used in the GPIB-1014D application. 0 = No device termination 1 = Device terminated operation normally

Bit	Mnemonic	Description
4r/w	ERR	Error Bit The Error bit is used to report the occurrence of error conditions. It is set if any errors have been signaled. If bit ERR is set, the CER logs the exact cause of the error. If this bit is cleared, the CER is also cleared. 0 = No errors 1 = Error as coded in CER
3r/w	ACT	Channel Active Bit The Channel Active bit is asserted after the channel has been started. The bit remains set until the channel operation terminates. This bit is unaffected by write operations. 0 = Channel not active 1 = Channel active
2r/w	0	Reserved Bit Write zero to this bit.
1r/w	PCT	Peripheral Control Transition Bit The Peripheral Control Transition bit is set if a falling edge transition has occurred on the Peripheral Control Line (PCL*) of the channel. The PCL* is active low. 0 = No PCL transition occurred 1 = High-to-Low PCL transition occurred
0r/w	PCS	Peripheral Control Status Bit The Peripheral Control Status reflects the state of the channel's PCL. This bit is unaffected by write operations. 0 = PCL low 1 = PCL high

Channel Error Register (CER)

VMEbus Address: Port Address + 001 (hex)

Attributes: Read Only, Internal to DMAC

7	6	5	4	3	2	1	0	R
0	0	0	ERROR CODE					

The Channel Error Register (CER) is an error condition status register. The ERR bit of the CSR indicates if there is an error. Bits 0 through 4 of the CER indicate what type of error occurred.

Bit **Mnemonic** **Description**

7-5r/w 0 Reserved Bits

Write zeros to these bits.

4-0r/w ERROR CODE

Error Code:

00000 = No error
00001 = Configuration Error
00010 = Operation Timing Error
00011 = (undefined, reserved)
001rr = Address error
011rr = Count error
010rr = Bus Error
10000 = External Abort
10001 = Software Abort

where rr = register or counter code:

01 = Memory address or memory counter
10 = Device address
11 = Base address of base counter

Channel Priority Register (CPR)

VMEbus Address: Port Address + 02D (hex)

Attributes: Read/Write, Internal to DMAC

7	6	5	4	3	2	0	
0	0	0	0	0	0	CP	R/W

The Channel Priority Register (CPR) is used to define the priority level for each channel. The priority of a channel is a number from 0 through 3, with 3 being the highest priority level. When multiple requests for DMA service are pending at the DMAC, the channel with the highest priority receives first service. Channel priority is also used to determine which channel is serviced first when multiple channels have interrupts pending. If there are several requesting channels at the highest priority level, a round-robin resolution is used. For the GPIB-1014D application, channel 0 and channel 1 priority should be the same.

Bit	Mnemonic	Description
7-2r/w	0	Reserved Bits Write zeros to these bits.
1-0r/w	CP	Channel Priority Bits 1 through 0 The Channel Priority bits specify the channel priority. 00 = Priority level 0 01 = Priority level 1 10 = Priority level 2 11 = Priority level 3

Interrupt Vector Registers

Each channel has an Normal Interrupt Vector Register (NIVR) and an Error Interrupt Vector Register (EIVR), each consisting of eight bits. The CPU responds to an interrupt request from the DMAC by executing an Interrupt Acknowledge Cycle. The GPIB-1014D hardware detects this cycle and checks to see if the indicated priority matches its own programmable priority level (for the VMEbus this can be level 1 through 7). If it does, the GPIB-1014D hardware acknowledges the interrupt service to the DMAC and the DMAC completes the cycle by placing the appropriate programmable interrupt vector on the lower eight bits of the data bus for the channel requesting an interrupt.

The EINT bit of the CCR determines if an interrupt can be generated by the channel. The interrupt request is generated if EINT is set and any of the three following conditions is met:

- The COC bit is set in the CSR.
- The BTC bit is set in the CSR.
- The PCT bit is set and the PCL line is programmed to be an interrupt input.

For GPIB DMA transfers, Port A uses channel 1 for status input with interrupt while Port B uses channel 3. This is described in more detail in Chapter 5.

The interrupt vector returned to the CPU comes from either the NIVR or the EIVR for the channel. The NIVR is used unless the ERR bit of the CSR is set, in which case the error interrupt vector is used. All interrupt vector registers in the DMAC are initialized to 0x0F on power-up or reset.

General Control Register (GCR)

VMEbus Address: Port Address + 0FF (hex)

Attributes: Read/Write, Internal to DMAC

7	6	5	4	3	2	1	0	
0	0	0	0	BT		BR		R/W

When the transfer mode is cycle steal with hold (like in most GPIB applications), the General Control Register (GCR) is used to define how long the DMAC, after transferring the last byte, will wait for another DMA request before relinquishing the bus. The DMAC will retain control of the bus unless the device (TLC) pauses. The TLC is determined to have paused if it does not make any requests during a full sample interval after the previous operand was transferred. The sample interval is programmed via the GCR and is expressed in clock cycles. The DMAC clock is 8 MHz. If any of the four DMAC channels is programmed to operate in cycle steal with hold mode, the same sample interval is used. The GCR is shared by all four channels in the DMAC.

Sample Interval = $2^{BT+BR+5}$ clock cycles

Bit	Mnemonic	Description
7-4r/w	0	Reserved Bits 7 through 4 Read/write zeros from/to these bits.
3-2r/w	BT	Burst Transfer Time Bits 3 through 2 00 = 16 clocks 01 = 32 clocks 10 = 64 clocks 11 = 128 clocks
1-0r/w	BR	Bandwidth Available to DMAC Bits 1 through 0 00 = 50.00% 01 = 25.00% 10 = 12.50% 11 = 6.25%

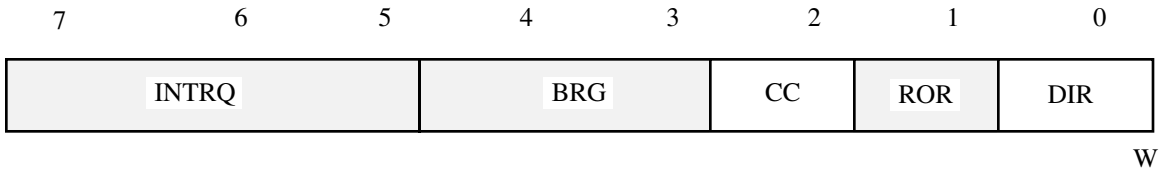
Configuration and Status Registers

Port A and Port B each contains two 8-bit write-only registers that are used to configure some of the board operating parameters. Some parameters are *board-specific* and some are *port-specific*. Board-specific parameters are shared by both ports, such as the VMEbus interrupt level or bus grant/request level. The bits that determine board-specific parameters in the configuration registers are shaded within the register bit map. Each shared bit is implemented with one flip-flop; therefore, it cannot hold a different value for each port. The configuration register bits that are not shaded are port-specific and are implemented using separate storage elements. Port-specific parameters provide some flexibility for each port. The GPIB status register in each port is used to monitor the state of the GPIB control lines. In each port, the GPIB status register is located at the same address as Configuration Register 1. A single Page Register is provided to set the VMEbus address lines A31 through A24 during Extended Addressing mode. The same Page Register is accessible from Port A and B.

Configuration Register 1 (CFG1A and CFG1B)

VMEbus Address: Port Address + 101 (hex) for CFG1A
Port Address + 301 (hex) for CFG1B

Attributes: Write Only



Configuration Register 1 (CFG1A and CFG1B) are 8-bit write-only registers used to configure some board-specific and port-specific parameters. The register bits define the following operating parameters.

Bit	Mnemonic	Description
7-5r/w	INTRQ	Interrupt Request Bits 7 through 5

The interrupt request bits are used to select the seven VMEbus interrupt request lines used by the board to request service from the interrupt handler of the CPU.

Note: Port A and Port B use the same VMEbus interrupt request line.

000 = No interrupt request line selected
001 = Interrupt request line IRQ1 selected
010 = Interrupt request line IRQ2 selected
011 = Interrupt request line IRQ3 selected
100 = Interrupt request line IRQ4 selected
101 = Interrupt request line IRQ5 selected

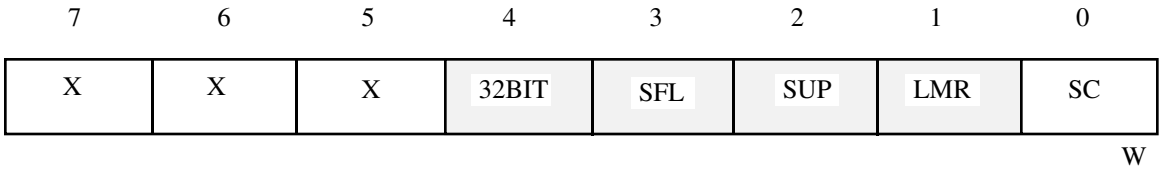
Bit	Mnemonic	Description
		110 = Interrupt request line IRQ6 selected 111 = Interrupt request line IRQ7 selected No interrupt request line is selected after power-up or after the board is reset.
4-3r/w	BRG	Bus Request/Grant Bits 4 through 3 The Bus Request/Grant bits are used to select which pair of the VMEbus request/grant lines are used by the GPIB-1014D to request and obtain control of the system bus. Again, the two Ports used the same pair of the VMEbus request/grant lines. 00 = BR0*/BG0IN*-BG0OUT* selected 01 = BR1*/BG1IN*-BG1OUT* selected 10 = BR2*/BG2IN*-BG2OUT* selected 11 = BR3*/BG3IN*-BG3OUT* selected BR0*/BG0IN*-BG0OUT* is selected automatically after power-up or after the board is reset.
2r/w	CC	Carry Cycle Bit The Carry Cycle bit is used to enable or disable the automatic carry cycle feature for Port A and B. Either one or both Ports can use the carry cycle feature. 1 = Carry Cycle feature enabled 0 = Carry Cycle feature disabled The feature is not enabled after power-up or after the board is reset.
1r/w	ROR	Release On Request Bit The Release On Request bit is used to indicate the status of the Release on Request feature for the GPIB-1014D. This is a board-specific parameter. 1 = Release On Request feature disabled 0 = Release On Request feature enabled The feature is automatically disabled (ROR=1) after power-up or after the board is reset.
0r/w	DIR	Direction Bit This bit is used to indicate the direction of the GPIB DMA data transfer on each port. The two ports can have different transfer direction. 0 = Transfer from system memory to GPIB 1 = Transfer from GPIB to system memory

Bit	Mnemonic	Description
		This bit is cleared to 0 (transfer from system memory to GPIB) after power-up or reset.

Configuration Register 2 (CFG2A and CFG2B)

VMEbus Address: Port Address + 105 (hex) for CFG2A
Port Address + 305 (hex) for CFG2B

Attributes: Write Only



Configuration Register 2 (CFG2A and CFG2B) are 8-bit write-only registers that are used to set the board access mode, determine if a port is a System Controller, and drive the VMEbus SYSFAIL* line. They also contains a Local Master Reset bit that can be used to reset the GPIB-1014D to a known state.

Bit	Mnemonic	Description
7-5r/w	0	Reserved Bits Write zeros to these bits.
4r/w	32BIT	Extended Addressing (32-bit) Bit This board-specific bit *and* the setting of switch W9 are used to place the GPIB-1014D in an Extended Addressing (32-bit) mode. The various VMEbus addressing modes (extended, standard, or short) are differentiated by a 6-bit address modifier (AM) code. If the 32-bit mode is disabled, the AM code is determined by the function code registers of the DMAC and the setting of switches W5, W6, and W8. If the 32-bit mode is enabled, the top two bits of the 6-bit AM code are automatically cleared to zero to reflect an extended addressing modifier code. The other four bits are determined by the function code registers of the DMAC and the setting of switches W5, W6, and W8.
3r/w	SFL	System Fail Bit This board-specific bit is used to drive the VMEbus SYSFAIL* line (if switch W2 is set to position 1) and to control the color of the LED on the board. On power-up, this bit is cleared, the LED is RED, and the board may drive the SYSFAIL* line active. After you have completed the diagnostics, setting this bit to a 1 will turn the LED green and release the SYSFAIL* line. This bit is *not* cleared by a Local Master Reset. 0 = LED red, SYSFAIL* is driven active. 1 = LED green, SYSFAIL* is not driven active.

Bit	Mnemonic	Description
2r/w	SUP	Supervisor Bit

The Supervisor bit is used to select Supervisor-Only or Supervisor-and-User access to the board. A switch is provided on the board to automatically select the value of this bit after power-up (or a system reset). If switch W7 is in the LMR position, all common circuitry will be reset by the LMR of either port. All port-specific circuitry is always reset by LMR.

0 = Supervisor-and-User access to the board
1 = Supervisor-Only access to the board

The factory default setting of switch W4 sets this bit to 1 after power-up or reset.

1r/w	LMR	Local Master Reset Bit

The Local Master Reset bit is used to reset the GPIB-1014D to a known state. Setting this bit to a 1 drives the local reset line active while clearing this bit releases the local reset line. The local reset line must be left in the active state for at least 10 msec to ensure that the onboard circuitry is reset properly. If switch W7 is in the LMR position, all common circuitry will be reset by the LMR of either port. All port-specific circuitry is always reset by LMR.

0 = Local reset line inactive
1 = Local reset line active

VMEbus signal SYSRESET*, if asserted, will automatically force a Local Master Reset.

0r/w	SC	System Controller Bit

The System Controller bit is used to control whether a port is to be the GPIB System Controller.

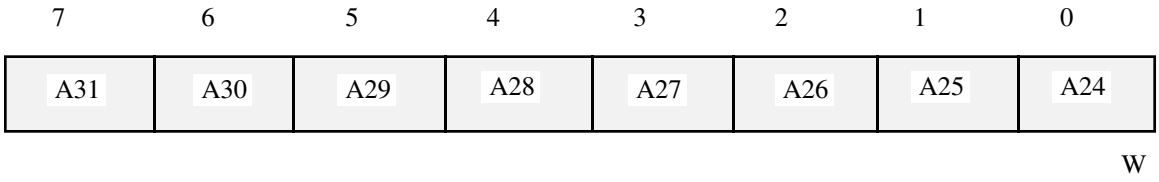
0 = GPIB-1014D port is not System Controller
1 = GPIB-1014D port is System Controller

This bit is cleared (not System Controller) upon reset or power-up.

Page Register (PGREG)

VMEbus Address: Port Address + 109 (hex) for PGREGA
Port Address + 309 (hex) for PGREGB

Attributes: Write Only



Page Register (PGREG) is an 8-bit write-only register used to set the upper eight address lines (A31-A24) of the VMEbus when using the Extended Addressing mode. The content of the register is driven onto the VMEbus when the DMAC drives its address lines. You must set this register manually to access the appropriate page of the VMEbus 32-bit address space. Thus, if a DMA operation crosses the page boundary, the you must reload the Page Register with the new page number. The same Page Register is accessible from both Port A and Port B.

GPIB Status Register (GSRA and GSRB)

VMEbus Address: Port Address + 101 (hex) for GSRA
Port Address + 301 (hex) for GSRB

Attributes: Read Only, Internal to DMAC

7	6	5	4	3	2	1	0 R
EOI	ATN	SRQ	REN	IFC	NRFD	NDAC	DAV

The GPIB Status Register (GSRA and GSRB) is an 8-bit read-only register used to monitor the status of the GPIB control lines on each port. Because the lines are monitored from TLC control line pins, the validity of each bit is determined by the direction of the GPIB transceiver. The direction of the transceiver is determined by the addressing mode of the TLC and the value of the SC bit of CFG2. Refer to the TLC register descriptions for more information. VMEbus data lines D7-D0 are driven with the content of the register during the read cycle. The response values are active high. If a bit is set, the corresponding GPIB line is asserted.

Chapter 5
Programming Considerations

This chapter explains important considerations for programming the GPIB-1014D.

Note: Detailed descriptions of all register bits can be found in Chapter 4, *Register Bit Descriptions*.

Initialization

On power on (pon), the VMEbus system typically issues a system reset (SYSRESET*) that drives the GPIB-1014D RESET* signal active. This action clears the Configuration Registers of both ports (Port A and Port B) and initializes the following circuitry:

- Timing State Machine
- DMA Gating and Control
- GPIB Synchronization and Interrupt Control
- Interrupter
- DTB Requester
- µPD7210 TLC
- 68450 DMAC

The GPIB-1014D also has another method for initializing the circuitry on the interface board. If the Local Master Reset (LMR) bit in Configuration Register 2 of Port A or Port B (CFG2A or CFG2B) is set, the RESET signal is driven and the GPIB-1014D is initialized in the same manner.

The NEC µPD7210 Talker/Listener/Controller (TLC) integrated circuit for each of the two ports is initialized as follows:

1. The local message pon is set and the interface functions are placed in their idle states (SIDS, AIDS, TIDS, SPIS, TPIS, LIDS, LPIS, NPRS, LOCS, PPIS, PUCS, CIDS, SRIS, SIIS).
2. All bits of the Serial Poll Mode Register (SPMR) are cleared.
3. The End Or Identify (EOI) bit is cleared.
4. All bits of the Auxiliary Registers A, B, and E (AUXRA, AUXRB, and AUXRE) are cleared.
5. The Parallel Poll Flag and request system control (rsc) local message are cleared.
6. The Internal Clock Register (ICR) is set to a count of eight.

7. The Transit Receive Mode 0 (TRM0) and Transit Receive Mode 1 (TRM1) bits in the Address Mode Register (ADMR) are cleared.

All other TLC register contents should be considered as undefined while the LMR is asserted and after LMR has been cleared. All Auxiliary Mode Register (AUXMR) commands are cleared and cannot be executed. All other TLC registers can be programmed while the TLC internal signal pon is set. When pon is released or cleared (by issuing an Immediate Execute pon auxiliary command to the TLC), the interface functions are released from the pon state and the auxiliary commands can be executed.

The 68450 DMAC is initialized as follows:

1. All bits of the GCR, DCR, OCR, SCR, CCR, CSR, CPR, and CER are cleared for all channels. This resets the STR, CNT, ACT and the interrupt generation bits and clears the status and error bits.

2. The Interrupt Vector Registers (Normal Interrupt Vector Register (NIVR) and Error Interrupt Vector Register (EIVR)) for all channels are set to 0x0F (hex).

Many bits of Configuration Registers 1 and 2 for Ports A and B overlap; thus, you can write to either port and accomplish the same task. See Chapter 4, *Register Bit Description*s, for information on the overlapping bits. A typical programmed initialization sequence for the GPIB-1014D includes the following steps:

1. Set and then clear the Local Master Reset (LMR) bit (bit 1 in either CFG2A or CFG2B) to place the GPIB-1014D in a known, quiescent state.

2. Load the onboard Configuration Registers (CFG1A, CFG1B, CFG2A, and CFG2B) with board-specific and port-specific parameters. Board-specific parameters include a Bus Request/Grant level, an Interrupt level, a Release On Request Enable, and 32-bit addressing whereas port-specific parameters include a transfer direction, a carry cycle feature enabled, and a GPIB System Controller. (See the CFG1 and CFG2 (A and B) description in Chapter 4 for the bits used to configure the GPIB-1014D operation.) Remember to set the System Controller (SC) bit in Configuration Register 1 (CFG1A or CFG1B) if the GPIB-1014D is to be the GPIB System Controller.

3. For each port, set or clear the desired Interrupt Enable bits in Interrupt Mask Register 1 (IMR1) and Interrupt Mask Register 2 (IMR2) of the TLC.

4. If Port A uses a VMEbus interrupt then the PCL bits in Channel 1 DCR (DCR1) must be set to 01 for status input with interrupt. In addition, bit EINT in Channel 1 CCR (CCR1) must be set for Channel 1 to generate an interrupt. Similarly, if Port B uses a VMEbus interrupt, then the PCL bits in DCR3 must be set to 01 and bit EINT in CCR3 must be set. If polling (no interrupt) is used, then set PCL bits in DCR1 or DCR3 to 00 for status input without interrupt. Bit EINT in CCR1 or CCR3 should be cleared.

5. If the VMEbus interrupt is used, load the DMAC Channel 1 (NIVR1) and (EIVR1) registers with the desired GPIB-1014D Status/ID byte. Similarly, for Port B, NIVR3 and EIVR3 should be loaded. As described in *Interrupt* later in this chapter, it is possible to use the same Status/ID byte for both NIVs and EIVs.

6. For each port, load the TLC primary GPIB address in Address Register 0 (ADR0) and Address Register 1 (ADR1).

7. For each port, enable or disable the GPIB Talker and Listener functions and the addressing mode using the ADMR.

8. For each port, load the Serial Poll response in the SPMR.

9. For each port, load the Parallel Poll response in the Parallel Poll Register (PPR) if local configuration is used. If using remote configuration, clear the PPR.

10. For each port, clear power on (pon) by issuing the Immediate Execute pon auxiliary command to the TLC to bring TLC online.

11. For each port, execute the desired TLC auxiliary commands.

The Page Register and the DMAC internal registers do not need to be configured further until just prior to a DMA operation.

The GPIB-1014D as GPIB Controller

The GPIB-1014D Controller function is generally in one of two modes: idle or in charge. When in charge, the Controller function is either active (asserting ATN) or standby (not asserting ATN). Because only one GPIB device can be in charge at a time, the following discussion, *Becoming Controller-In-Charge and Active Controller*, concerns only one of the GPIB-1014D ports which has programmed to be in charge. The following paragraphs discuss the various transitions between the idle and in charge modes.

Becoming Controller-In-Charge and Active Controller

The TLC can become Controller-In-Charge (CIC) either by being the System Controller and taking control (by issuing the Set IFC auxiliary command) or by being passed control of the GPIB from the current Active Controller.

The GPIB-1014D is only capable of driving the GPIB IFC and REN lines (which allows the TLC to function as GPIB System Controller) when the SC bit in CFG1 is set. To take control:

1. Issue the Set IFC auxiliary command.

2. Wait for a minimum of 100 µsec.

3. Issue the Clear IFC auxiliary command.

The GPIB IFC message initializes the GPIB interface functions of all devices on the bus. As soon as any existing CIC goes to idle (dropping ATN if it was active), the TLC becomes CIC and Active Controller and asserts the GPIB ATN line.

Another Active Controller passes control to the GPIB-1014D by sending the TLC GPIB Talk Address (MTA) followed by the GPIB Take Control (TCT) message. The TLC, upon receiving these two messages (MTA and TCT), automatically becomes CIC when ATN is dropped. The exact sequence of events is as follows:

1. The TLC receives the MTA. The TLC then enters into Talker Addressed State (TADS). This operation can be transparent to a program. The Talker Active (TA) bit in the Address Status Register (ADSR) is set when the TLC receives its MTA.

2. The TLC receives the GPIB TCT message.

3. The current Active Controller sees the completed handshake, goes to idle mode, and unasserts ATN.

4. As soon as the ATN line on the GPIB is unasserted, the TLC automatically becomes CIC and asserts ATN.

As soon as the TLC becomes CIC, the CIC bit in the ADSR, and the Command Output (CO) bit in Interrupt Status Register 2 (ISR2) are set. Using these two bits, the program can unambiguously determine that the TLC is the GPIB Active Controller and can send remote messages.

Sending Remote Multiline Messages (Commands)

The GPIB-1014D sends commands as the Active Controller by writing to the Command/Data Out Register (CDOR) in response to the CO status bit in ISR2. DMA transfers are not supported when the TLC is GPIB Active Controller, and must not be attempted.

To make the TLC a Listener, you must write the TLC listen address to the CDOR. The TLC recognizes any command that applies to its own talk or listen address on the GPIB.

Going from Active to Standby Controller

If the TLC is GPIB Active Controller, the Controller Standby State (CSBS) is entered upon receipt of the Go To Standby auxiliary command. The ATN line is unasserted as soon as the TLC enters CSBS. Do not issue the Go To Standby auxiliary command unless the CO bit in ISR2 is set.

There are three cases to consider when going from Active Controller to Standby Controller:

Case 1: The TLC becomes the GPIB Talker when ATN is unasserted. To do this, wait for CO to be set, send the TLC MTA, wait for CO to be set again, and then issue the Go To Standby auxiliary command.

Case 2: The TLC becomes a GPIB Listener when ATN is unasserted. To do this, wait for CO to be set, issue the TLC GPIB Listen Address (MLA), wait for CO to be set again, and then issue the Go To Standby auxiliary command.

Case 3: The TLC is neither GPIB Talker nor Listener. In this case, issue the Listen In Continuous Mode auxiliary command before going to standby. Once this mode is enabled, the TLC participates in the GPIB handshake without setting the DI (Data In)

bit (or holding off the handshake at DAC until the DIR is read. Then it can take control synchronously when necessary.

Going from Standby to Active Controller

The manner in which the TLC resumes GPIB Active Control depends on how it went to standby. Consider the following three cases when going from Standby Controller to Active Controller:

Case 1: The TLC, as a Talker, takes control upon receipt of the Take Control Asynchronously auxiliary command. Do not issue the Take Control Asynchronously auxiliary command until there are no more bytes to send and the DO bit is set.

Case 2: The TLC, as a Listener, takes control upon receipt of the Take Control Synchronously auxiliary command. If a Programmed I/O is used, the Take Control Synchronously auxiliary command must be issued between seeing a DI status bit and reading the last byte from the DIR.

If DMA is used, a handshake holdoff must be in effect after the last data byte is read in order for the Take Control Synchronously auxiliary command to work properly. This is accomplished by:

1. Writing 100XX110 (binary) to the AUXMR. Writing this byte to the AUXMR issues the RFD Holdoff On End mode auxiliary command to the TLC, and also causes the END bit in ISR1 to be set upon receipt of the EOS message.

   Note: If the carry cycle feature is used, the RFD holdoff on all Data Mode auxiliary commands must be written to AUXRA before the DMAC reads the last data byte from the TLC.

2. Remember to set the CC bit in CFG1 before starting the DMA transfer.

3. When the DMA transfer has finished, either from an END interrupt or a FIN interrupt which occurs when the GPIB is synchronized, issue the Take Control Synchronously auxiliary command followed by the Finish Handshake auxiliary command.

Case 3: The TLC, as neither Talker nor Listener, takes control synchronously with the Take Control Synchronously auxiliary command. Since the Listen in Continuous Mode auxiliary function is active, the Take Control Synchronously auxiliary command can be sent at any time.

When the Take Control Synchronously auxiliary command is used, the TLC takes control of the GPIB only at the end of a data transfer. This implies that one transfer must follow or be in progress when the Take Control Synchronously auxiliary command is issued. If this is not the case, the Take Control Asynchronously auxiliary command must be used. Of course, the Take Control Asynchronously auxiliary command can be used in place of the Take Control Synchronously auxiliary command when the possibility of disrupting an in-progress GPIB handshake (before all GPIB Listeners have accepted the data byte) is acceptable.

In all cases, a CO status indicates that the GPIB-1014D is now Active Controller.

Going from Active to Idle Controller

Going from Active to Idle GPIB Controller, also known as passing control, requires that the TLC be the Active Controller initially (to send the necessary GPIB command messages). After the TLC has become the GPIB Active Controller, it must complete the following procedures to pass control:

1. Write the MTA of the device being passed control to the CDOR.

2. In response to the next CO status, write the GPIB TCT message to the CDOR.

3. As soon as the TCT command message is accepted by all devices on the GPIB, the TLC automatically unasserts ATN and the new Controller asserts ATN.

The GPIB-1014D as GPIB Talker and Listener

Each port can be either GPIB Talker or Listener. For each port, the Talker or Listener function is deactivated automatically if the other is activated. The following paragraphs describe the GPIB Talker and Listener in general, and do not refer to any specific port. The TA, LA, and ATN* bits in the ADSR together indicate the specific state of the TLC as follows:

ATN*	TA	LA	
0	1	0	Addressed Talker–cannot send data
1	1	0	Active Talker–can send data
0	0	1	Addressed Listener–cannot receive data
1	0	1	Active Listener–can receive data

The status bits Address Status Change (ADSC), Command Output (CO), Address Pass Through (APT), Data Out (DO), and Data In (DI) are used to prompt the program (possibly with an interrupt request) when a change of state occurs.

The following paragraphs discuss several aspects of data transfers.

Programmed Implementation of Talker and Listener

When there is no Controller in the GPIB system, the ton and lon address modes (refer to the description of the ADMR in Chapter 4) are used to activate the TLC GPIB Talker and Listener functions. If used, ton or lon must be set during TLC initialization.

When the TLC is GPIB Active Controller, the Listen and Local Unlisten programmed auxiliary commands are used to activate and deactivate the TLC GPIB Listener function.

Addressed Implementation of the Talker and Listener

The TLC, when GPIB Active Controller, can address itself by sending its own GPIB Talk or Listen address across the GPIB bus. When another device on the GPIB is acting as Controller, the TLC is addressed with GPIB command messages to become a Talker or Listener.

Address Mode 1

If the TLC ADMR has been configured for Address Mode 1, the TLC responds to the reception of two primary GPIB addresses: major and minor. On receipt of the TLC major or minor talk address or its major or minor listen address from the GPIB Active Controller, the TLC is addressed as Talker or Listener, respectively. If the TLC has received its talk address, the TA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DO bit in ISR1 sets when ATN* is unasserted. If the TLC has received its listen address, the LA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DI bit in ISR1 sets when a GPIB data byte is received. The MJMN bit in the ADSR indicates whether the address status refers to the major or minor address.

Address Mode 2

Address Mode 2 implements Talker Extended (TE) or Listener Extended (LE) functions. TE and LE functions require the receipt of two addresses (primary and secondary) before setting TA or LA. The TLC GPIB primary address is specified by the byte written to ADR0. The secondary address is specified by the byte written to ADR1. Upon receipt of both the primary and secondary GPIB addresses the TLC becomes an addressed Talker or Listener. If the TLC has received its primary GPIB talk address, the Talker Primary Addressed State (TPAS) bit in the ADSR is set. If the TLC has received its primary talk address, followed by its secondary address, the TA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DO bit in ISR1 sets when ATN* is unasserted. If the TLC has received its primary listen address, the LPAS bit in the ADSR is set. If the TLC has received its primary listen address, immediately followed by its secondary MLA, the LA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DI bit in ISR1 sets when the first GPIB data byte is received.

Address Mode 3

Address Mode 3, like Address Mode 2, is used to implement Extended GPIB talk and listen address functions. However, unlike Address Mode 2, Address Mode 3 uses both major and minor primary addresses. Your program must identify the secondary address by reading the CPTR. Proper operation using Address Mode 3 is listed as follows:

1. During initialization of the TLC, enable Address Mode 3 and optionally set the APT IE bit in IMR1 to enable an interrupt request on receipt of a secondary GPIB address.

2. Write the TLC major GPIB primary address to ADR0 and the TLC minor GPIB primary address to ADR1.

   Note: Receipt of the TLC major or minor primary address sets TPAS or LPAS accordingly, indicating that a primary address has been received.

3. If the next GPIB command following the primary address is a secondary address,

   a. The APT bit is set and a DAC handshake holdoff is activated (the GPIB DAC message is held false)

   b. In response to the APT, the program must complete the following tasks:

      - Determine whether the command previously received was a listen, talk, major, or minor address by reading the LPAS, TPAS, and MJMN bits of the ADSR; and

- Read the secondary address in the CPTR and determine whether or not it is the address of the TLC.

4. If it *is not* the TLC address, issue the Non-Valid auxiliary command. If it *is* the TLC address, issue the Valid auxiliary command.

5. If the Valid auxiliary command is issued, the TLC assumes that the My Secondary Address (MSA) message has been received, which causes the following events to take place:

   a. The LA bit is set and the TA bit is cleared (LADS=TIDS=1) if LPAS was set, or the TA bit is set and the LA bit is cleared (TADS=LIDS=1) if TPAS was set

   b. The GPIB DAC message is sent true, and the GPIB handshake is finished

   If the Non-Valid auxiliary command is issued, the TLC assumes that the Other Secondary Address (OSA) message has been received, which causes the following events to take place:

   a. The TLC Talker or Listener function goes to its idle state (TIDS=1 or LIDS=1) if the either the TPAS or LPAS bit was set

   b. The GPIB DAC message is sent true, and the handshake is finished

Until a GPIB Primary Command Group (PCG) message is received (that is, as long as the subsequent messages are secondary addresses), the APT bit is set and a DAC holdoff is in effect each time a GPIB secondary address is received. In this way, the GPIB CIC can address several devices having the same primary address without repeating the primary address each time. If a PCG message is received before a secondary address is received, the TPAS and LPAS bits are cleared.

Sending/Receiving Messages

Since each port can be a GPIB Talker or Listener, the following paragraph includes either port and can be interpreted accordingly.

When the TLC is a GPIB Talker or Listener, data (device-dependent messages) can be sent or received using either DMA or programmed I/O. When the TLC is Active Controller, commands (remote multiline messages) can be sent using programmed I/O only. The default configuration is programmed I/O (DMA mode is activated only by issuing a specific sequence of commands to the GPIB-1014D). In either programmed I/O or DMA modes, the GPIB interface functions are programmed or addressed in the same way, with minor exceptions described in the following discussion, *Using Direct Memory Access*.

Using Direct Memory Access

The onboard DMA Controller is the 68450 (DMAC). This chip provides four independent DMA channels, all of which are used by the GPIB-1014D to transfer data between the VMEbus memory and the GPIB. Port A uses Channel 0 and/or 1 and Port B uses Channel 2 and/or 3. The GPIB-1014D only supports single address (fly by mode) DMA transfers to the TLCs. During single address DMA cycles, data bytes transfer directly between VMEbus memory and the TLCs without being retained in a temporary register inside the DMA Controller. Although the 68450 supports several different DMA Request Generation modes, it must be programmed for external

and Cycle Steal DMA request, since the TLCs assert the DMA request line before each transfer. The DMAC can be programmed via the REQG bits in the OCR to perform Cycle Steal with Hold or Cycle Steal Without Hold mode transfers. In Cycle Steal Without Hold mode, the DMAC will request use of the VMEbus upon receiving a DMA request from the TLCs. Once the VMEbus is granted to the GPIB-1014D, the DMAC performs the DMA transfer. After the DMAC finishes the transfer, it immediately releases the bus. In Cycle Steal With Hold mode, after performing a DMA transfer, the DMAC will hold the bus for a programmable time period waiting for another DMA request. If no request is received in the time period allotted, the DMAC will relinquish the VMEbus. If a request is received in the time period allotted, the DMAC must request the use of the VMEbus in response to the next DMA request. To reduce the base arbitration time, you must select the hold option. In addition, either or both ports can include Cycle Steal With Hold mode.

For each port, the onboard TLC acts as a buffer between the VMEbus memory and the GPIB. When the TLC is a GPIB Talker and its CDOR is empty, the TLC asserts the DMA request line. The DMAC, in response to the DMA request, transfers data from VMEbus memory to the CDOR. The TLC then performs the necessary handshake sequence to transfer the byte to all GPIB listeners. Similarly, when the TLC is listener and a byte of received data is in the DIR, the TLC asserts the DMA request line. The DMAC then transfers the data from the DIR to VMEbus memory.

The GPIB-1014D also has the Release On Request (ROR) feature. This board-specific feature, which is enabled in CFG1A or CFG1B, causes the board to hold onto the VMEbus if no other device on the VMEbus requests the bus. Even after the DMAC has relinquished the bus, the board still asserts VMEbus signal BBSY* to hold onto the bus. If there is a DMAREQ (from any port) while the board is holding the bus, the DMAC is regranted the bus immediately to start the next DMA transfer. If the board is holding the bus and some other device on the VMEbus requests the bus, the GPIB-1014D immediately releases the bus.

The two Ports can be configured to transfer data between the GPIB and the VMEbus system memory with or without the carry cycle feature. This port-specific feature is enabled by setting bit 2 in CFG1A or CFG1B. In general, during a DMA transfer from the VMEbus memory to the GPIB, the carry cycle feature is used to make the TLC (as a GPIB talker) set the EOI bit active during the transfer of the last data byte. Sending EOI along with the last data byte is useful to inform all GPIB listeners that the byte is the last one in the transfer. From the port's point of view (as a GPIB Talker), the last byte is defined as the terminal count in the MTCR of the active channel regardless if carry cycle feature is used or not. During a DMA transfer from the GPIB to the VMEbus memory (the port is a GPIB listener), the carry cycle feature is used to make the TLC hold off the handshake sequence after the last byte has been accepted by all GPIB Listeners. The hold off of the handshake sequence by a GPIB listener is used to prevent the GPIB talker from sending more bytes. From the port's point of view (as a GPIB listener), the last byte is defined as the terminal count in the MTCR of the active channel regardless if carry cycle feature is used or not. In general, the carry cycle feature facilitates the termination of a GPIB transfer.

Without using the carry cycle feature, each port uses only one DMAC channel to transfer its data. Using the carry cycle feature, each port uses two DMAC channels to transfer data. Particularly, Channel 0 is used for Port A to control the transfer of the entire block (n bytes) of data when the carry cycle feature is not selected. Channel 2 is used for Port B to control the transfer of the entire block (n bytes) of data when the carry cycle feature is not selected. If Port A uses the carry cycle feature, Channel 0 is used to control the transfer of the first (n-1) bytes of data and Channel 1 is used to transfer one byte of the GPIB command (the carry cycle byte) from the VME system memory to the TLC Auxiliary Register, and to transfer the last (nth) byte of data. The carry cycle byte must be written to the TLC just prior to the last data byte in order to instruct the TLC to handle the last byte as needed (to send EOI or the holdoff the handshake). The carry

cycle byte is a local GPIB command that is inserted at the end of an n-byte transfer. There is a slight preparation process you must do if you use the carry cycle (See *DMA Transfers with a Carry Cycle* later in this chapter). Similarly, if Port B uses the carry cycle feature, Channel 2 is first used to transfer n-1 bytes of data and Channel 3 is used to transfer the carry cycle byte and the last data byte.

Regardless if the carry cycle feature is used or not, after each Port has transferred n bytes of data, it then automatically detects for synchronization on its GPIB, that is, if all Listeners have accepted the data. For each port, this synchronization signal along with interrupt signals from the TLC and VMEbus signal BERR* are first ORed together and the result is routed to a pin on the DMAC. Particularly, Port A sends its interrupt to PCL1* pin while Port B sends its interrupt request to PCL3*. If the ports are enabled for interrupts, the assertion of PCL1* or PCL3* will cause the DMAC to send an interrupt to the VMEbus. Thus, for each port, you can set up and start the DMA
transfer and wait for the GPIB synchronization interrupt of the port. This interrupt indicates, for the port, n bytes have been transferred and the GPIB is synchronized.

Note: If the interrupt is not enabled, polling can be used to detect GPIB synchronization.

DMA Transfers without Carry Cycle

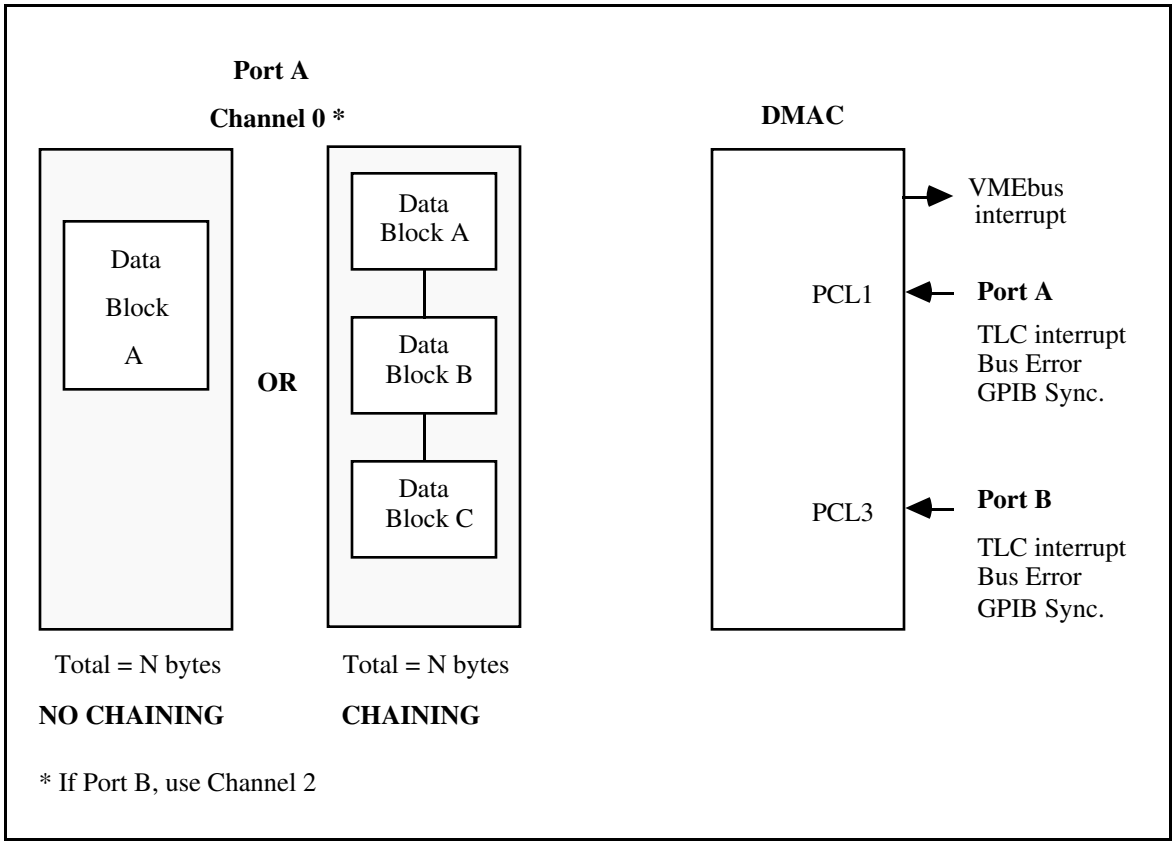


Figure 5-1. Channel Preparation without Carry Cycle

If the carry cycle feature is not needed in a transfer sequence, there is little programming needed. For Port A, Channel 0 is used to transfer the entire block of data, since the carry cycle byte does not need to be inserted in the data. Channel 1 DMA channel is not used and must not be started. For Port B, Channel 2 is used but Channel 3 is inactive. As indicated in Figure 5-1, you can use no-chaining or chaining mode, depending on the number of data blocks to be transferred for each port.

Detailed instructions for a GPIB DMA transfer without the carry cycle feature using Port A is described in the following programming sequence.

Note: Port B can be programmed by writing to the equivalent configuration registers and TLC within Port B. In addition, when programming Port B, all references to Channel 0 can be changed to Channel 2 and all references to Channel 1 can be changed to Channel 3.

1. In CFG1A, set the CC bit to 0. Set the DIR bit to reflect the direction of the DMA transfer for the port (1=GPIB-to-Memory, 0=Memory-to-GPIB). Set the ROR* bit if the ROR feature is enabled. If interrupts are used, set the INTRQ bits to select the interrupt level. Set the BRG bits to choose one of four VMEbus request lines.

2. In CFG2A, set the 32-bit bit in CFG1B to select 32-bit addressing. If 32-bit addressing is used, load the Page Register with the value of address lines A32 through A24.

3. Configure Channel 0 to use a flyby transfer for the *n* data bytes between the GPIB and the VME system memory. The sequence is as follows:

   a. Write Channel 0 CCR with the SAB bit set to abort the channel operation in case it is still active.

   b. Write a 0xFF (hex) to Channel 0 CSR to clear any leftover error or status bits.

   c. Load the XRM bits of Channel 0 DCR with the proper value to select the DMA transfer mode (Cycle Steal Without Hold or Cycle Steal With Hold). Set the DTYP bits to 10 (device with ACK*, implicitly addressed), set the DPS bit to 0 (8-bit port size), and set the PCL bits to 00 (status input). If the Cycle Steal With Hold transfer mode is selected, write to the GCR to select the required timeout.

   d. Write to Channel 0 OCR. Set the DIR bit to reflect the port's direction of transfer (0=Memory-to-GPIB, 1=GPIB-to-Memory). Set the SIZE bits to 00 (byte); set the CHAIN bits to the desired value (00 if no chaining is used, 10 if array chaining is used, 11 if linked chaining is used), and set the REQG bits to 10 (REQ* line initiates transfers). Use Chaining to transfer a block of data greater than 64K or to transfer multiple blocks of data.

   e. Write to Channel 0 SCR. Set the MAC bits to determine if the MAC bits count up or down (01=Up, 10=Down). The DAC bits are not used.

   f. If no chaining is required:

      - Load Channel 0 MFC with the proper data to generate the required Address Modifier Code to access the data buffer. (See Table 3-1 and 3-2 for recommended values.)

      - Load Channel 0 MAR with the starting physical address of the buffer of data that is to be transferred.

- Load Channel 0 MTCR with the number of bytes in the buffer to be transferred (must be less than or equal to 65535 bytes).

If chaining is required:

- Chaining modes (array or linked) use an address and transfer count array which is an array of pointers that point to the data blocks to be transferred.
- For array or linked chaining, load Channel 0 BFC with the proper data to generate the required Address Modifier Code to access the address and transfer count array. (See Tables 3-1 and 3-2 for recommended values.)
- For array or linked chaining, load Channel 0 BAR with the starting physical address of the address and transfer count array.
- For array chaining, load Channel 0 BTCR with the number of entries in the address and transfer count array. Linked chaining does not use BTCR.
- For array or linked chaining, load Channel 0 MFC with the proper data to generate the required Address Modifier Code to access the data blocks. (See Tables 3-1 and 3-2 for recommended values.)
- Set up the data blocks and the address and transfer count array in VMEbus memory. (Figures 6-1 and 6-2 show how to set up the array for both chaining modes.)

4. Channel 1 must be programmed before starting the transfer in the following manner:
   - If the VMEbus interrupt is used:
     a. Set the PCL bits in Channel 1 DCR to 01 for status input with interrupts.
     b. Load Channel 1 NIV and EIV with the proper status/ID byte to return to the VMEbus interrupt handler.
     c. Set the EINT bit in Channel 1 CCR to enable interrupts.
   - If the VMEbus interrupt is not used, set the PCL bits in Channel 1 DCR to 00 for status input. During the DMA transfer, you can check the PCT and PCS bits in Channel 1 CSR to see if TLC has interrupted, BERR* has occurred, or if GPIB synchronization has occurred.
5. Once Channels 0 and 1 have been configured, you must start the DMAC by setting the STR bit in CCR of Channel 0.
6. Finally, configure the TLC for a DMA operation by completing the following steps:
   a. Set END IE in IMR1 if the TLC is a GPIB Listener. Set ERR IE in IMR1 if the TLC is a GPIB Talker. In all cases, clear all other IE bits.
   b. Set the DMAO bit in IMR2 if the TLC is a GPIB Talker. Otherwise, clear DMAO.
   c. Set the DMAI bit in IMR2 if the TLC is a GPIB Listener. Otherwise, clear DMAI.
   d. The TLC will request for DMA operations as soon as DMAO or DMAI is set.

DMA Transfers with Carry Cycle

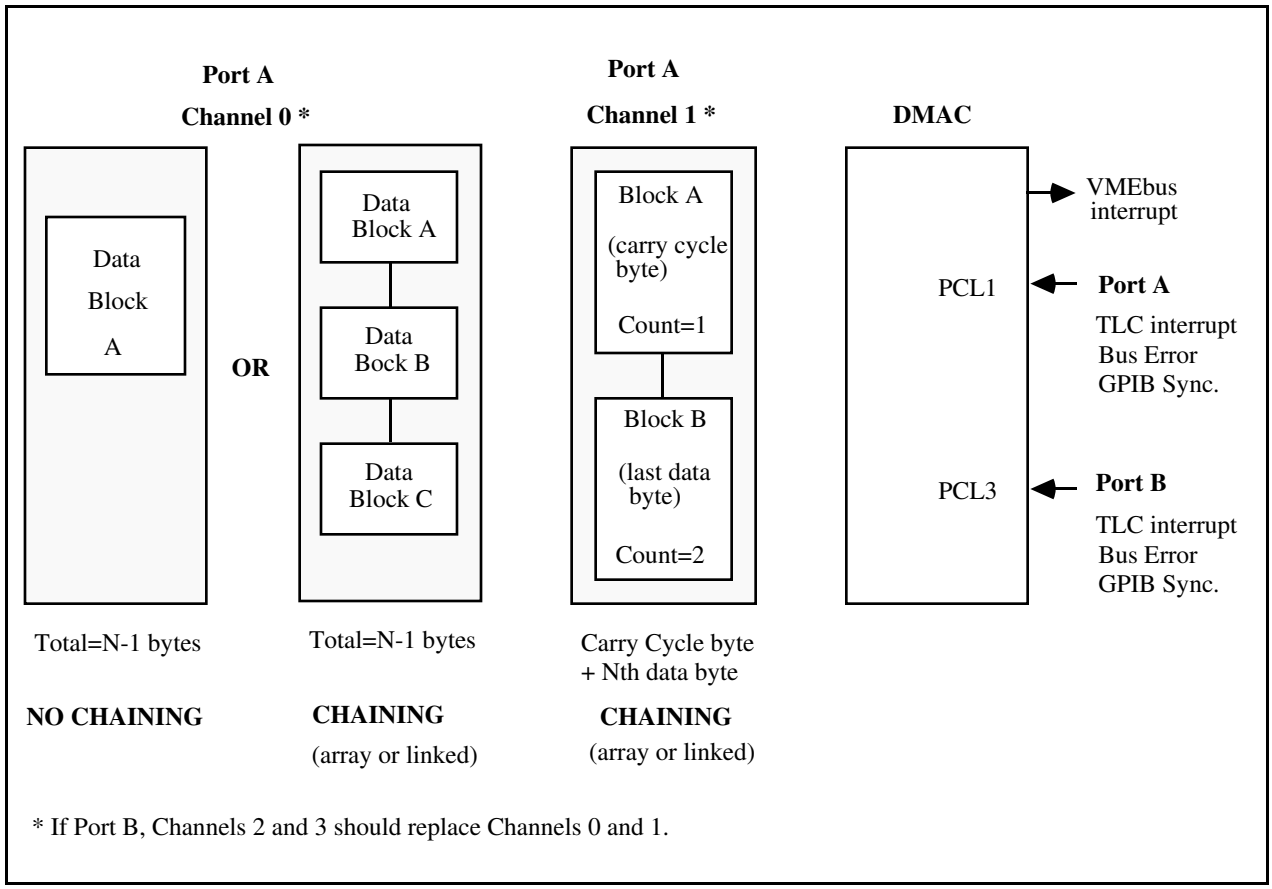


Figure 5-2. DMA Transfer with Carry Cycle

When the carry cycle feature is needed in a transfer, it is transparent to the system CPU and is automatically handled by the GPIB-1014D once the DMAC channels have been properly configured. As indicated earlier in this chapter, Port A and B use both of their assigned channels for DMA operations. For each port, the second of its two assigned channels (Channel 1 for Port A or Channel 3 for Port B) is used to transfer the carry cycle byte and the last data byte. Setting up the first channel (Channel 0 for Port A or Channel 2 for Port B) to transfer n-1 bytes is similar to the steps used to program the DMAC explained earlier in this chapter. The second channel is set up to transfer two 1-byte data blocks with the first block containing the carry cycle byte and the second block containing the last data byte. It is easiest to use the array chaining mode on the second channel; however, linked chaining is possible. The PCL of the second channel can also be configured to accept interrupts of the port. When the GPIB is finally synchronized, you can check the COC and ERR bit in the CSR of the first channel (CSR0 for Port A or CSR2 for Port B) to determine the status of the DMA transfer of the first n-1 bytes. The success of the second channel is determined by examining its MTCR (MTCR1 for Port A and MTCR3 for Port B). This is described in more detail under *Terminating the Transfer and Checking the Result* later in this chapter.

Detailed instructions for a GPIB DMA transfer with the carry cycle feature using Port A is described in the following programming sequence.

Note: Port B can be programmed by writing to the equivalent configuration registers and TLC within Port B. In addition, when programming Port B, all references to Channel 0 can be changed to Channel 2 and all references to Channel 1 can be changed to Channel 3.

1. In CFG1A, set the CC bit to a 1 to enable the carry cycle feature. Set the DIR bit to reflect the DMA transfer direction (1=GPIB-to-Memory, 0=Memory-to-GPIB) of the port. Also set the ROR* bit if the ROR feature is to be enabled. If the interrupt is used, set the INTRQ bits to select the interrupt level. Set the BRG bits to choose one of four VMEbus request lines.

2. In CFG2A, set the 32-bit bit in CFG1B to select 32-bit addressing. If 32-bit addressing is used, load the Page Register with the value of address lines A32-A24.

3. Configure Channel 0 to use a flyby transfer for the n-1 data bytes between the GPIB and the VME system memory. The sequence is as follows:

   a. Write to Channel 0 CCR with the SAB bit set to abort the channel operation in case it is still active.

   b. Write a 0xFF (hex) to Channel 0 CSR to clear any remaining error or status bits.

   c. Load Channel 0 DCR with the proper value to select the DMA transfer mode, Cycle Steal Without Hold, or Cycle Steal With Hold. Set the DTYP bits to 10 (device with ACK*, implicitly addressed); set the DPS bit to 0 (8-bit port size); and set the PCL bits to 00 (status input). If the Cycle Steal With Hold transfer mode is selected, write to the GCR to select the required timeout.

   d. Load Channel 0 SCR to define the sequencing of memory and device address registers. Set the MAC bits to determine if the MAC counts up or down (01=Up, 10=Down). The DAC bits are not used.

   e. Load Channel 0 OCR to select the direction of transfers, the size of the data transfer, which chaining method is used, and which DMA Request generation method is used. Set the DIR bit to reflect the port's direction of transfer (0=Memory-to-GPIB, 1=GPIB-to-Memory). Set the SIZE bits to 00 (byte); set the CHAIN bits to the desired value (00 if no chaining is used, 10 if array chaining is used, 11 if linked chaining is used), and set the REQG bits set to 10 (REQ* line initiates transfer). Chaining is used to transfer a block of data greater than 64K or to transfer multiple blocks of data.

   f. If no chaining is required:

      - Load Channel 0 MFC with the proper data to generate the required Address Modifier Code to access the data buffer. See Table 3-1 and 3-2 for recommended values.

      - Load Channel 0 MAR with the starting physical address of the buffer of data that is to be transferred.

      - Load Channel 0 MTCR with the number of bytes (n-1 total) in the buffer to be transferred (must be less than or equal to 65,535 bytes).

      If chaining is required:

      - Chaining modes (array or linked) all use an address and transfer count array which is an array of pointers that point to the data blocks to be transferred.

- For array or linked chaining, load Channel 0 BFC with the proper data to generate the required Address Modifier Code to access the address and transfer count array. (See Table 3-1 and 3-2 for recommended values.)
- For array or linked chaining, load Channel 0 BAR with the starting physical memory address of the address and transfer count array.
- For array chaining, load Channel 0 BTCR with the number of entries in the address and transfer count array. Linked chaining does not use BTCR.
- For array or linked chaining, load Channel 0 MFC with the proper data to generate the required Address Modifier Code to access the data blocks. (See Table 3-1 and 3-2 for recommended values.)
- Set up the data blocks and the address and transfer count array in VMEbus memory. The total number of bytes in the data block should be n-1 bytes. Figures 6-1 and 6-2 describe how to set up the array for both chaining modes.

4. Use Channel 1 to control the interrupts from the board and to implement the carry cycle feature. Construct an address and transfer count array in memory and use chaining to transfer the data blocks defined by the array. The address and transfer count array is an array of pointers that point to the data blocks to be transferred. The procedure for configuring Channel 1 is as follows:

   a. Write to Channel 1 CCR with the SAB bit set to abort the channel operation in case it is still active.

   b. Write an 0xFF (hex) to Channel 1 CSR to clear any leftover error or status bits.

   c. Load Channel 1 DCR with the proper value to select the DMA transfer mode, Cycle Steal Without Hold or Cycle Steal With Hold. Set the DTYP bits to 10 (device with ACK*, implicitly addressed); set the DPS bit to 0 (8 bit port size); and set the PCL bits to 00 (status input) or 01 (status input with interrupt). If the Cycle Steal With Hold transfer mode is selected, write to the GCR to select the required timeout. (See the GCR description in Chapter 4 for more details.)

   d. Write to Channel 1 OCR. Set the DIR bit to reflect the port's direction of transfer (0=Memory-to-GPIB, 1=GPIB-to-Memory), set the SIZE bits to 00 (byte), set the CHAIN bits to 10 or 11 (array chaining or linked chaining), and set the REQG bits to 10 (the REQ* line initiates the transfer). The array chaining feature is usually used to implement the carry cycle because of its simplicity.

   e. Set Channel 1 SCR to 00 (addresses do not count).

   f. For array or linked chaining, load Channel 1 BFC with the proper value to generate the required address modifier code to access the address and transfer count array. (See Table 3-1 and 3-2 for recommended values.)

   g. For array or linked chaining, load Channel 1 BAR with the beginning address of the address and transfer count array.

   h. For array chaining, load Channel 1 BTC with 2 (two entries in the carry cycle array). Linked chaining does not use BTC.

i. For array or linked chaining, load Channel 1 MFC with the proper value to generate the desired address modifier code to access the data blocks. (See Table 3-1 and 3-2 for recommended values.)

j. Construct an address and transfer count array in memory. The array must begin at an even address. The contents and setup of the array are as follows :

First 4 bytes = physical address of carry cycle byte

Next 2 bytes = 0001 (hex)

Next 4 bytes = physical address of last data byte in the data buffer to be transferred (the address can be even or odd)

Last 2 bytes = 0002 (hex) (see *Checking Transfer Results and Handling Interrupts* or Chapter 6 on why two bytes are required)

For linked chaining mode, the address and transfer count array is similar to the one above except the link address to the next array entry is needed. (Figures 6-1 and 6-2 show how to set up the array for both chaining modes.)

Write the carry cycle byte to the memory allocated for its storage. This may be a command such as send EOI or RFD Holdoff on ALL. This byte must be located somewhere in memory where it can be accessed by the DMAC. The address of this byte is the first element in the address and transfer count array (first four bytes). The DMAC uses this array and the chaining mode with Channel 1 to insert the command in the data string. The MAR and MTCR are initialized and the carry cycle byte is transferred. The MAR and MTCR are then reloaded (the DMAC transfers the bytes in from the array using DMA) and the last byte of the data buffer is transferred.

k. Program Channel 1 to detect a TLC interrupt, bus error, and GPIB synchronization in the following manner:

- If the VMEbus interrupt is used:

  1. Set the PCL bits in Channel 1 DCR to 01 for status input with interrupts.

  2. Set the EINT bit in Channel 1 CCR to enable interrupts.

  3. Load Channel 1 NIV and EIV with the proper status/ID byte to return to the VMEbus interrupt handler.

- If the VMEbus interrupt is not used, set the PCL bits in Channel 1 DCR to 00 for status input. You can check the PCT and PCS bits in Channel 1 CSR to see if TLC has interrupted, if BERR* has occurred, or if GPIB synchronization has occurred.

5. Once Channels 0 and 1 have been configured properly, start the DMA channels. Channel 1 must be started before Channel 0. Start the channels by writing to the CCRs with the STR bits set. (Channel 1 should also have the EINT bit set if you are using interrupts.)

6. Finally, configure the TLC for a DMA operation by completing the following steps:

   a. Set the END IE bit in IMR1 if the TLC is a GPIB Listener. Set the ERR IE bit in IMR1 if the TLC is a GPIB Talker. In all cases, clear all other IE bits.

b. Set the DMAO bit in IMR2 if the TLC is a GPIB Talker. Otherwise, clear DMAO.

c. Set the DMAI bit in IMR2 if the TLC is a GPIB Listener. Otherwise, clear DMAI.

The TLC will request for DMA operations as soon as DMA0 or DMAI is set.

Polling During DMAs

All the GPIB-1014D registers are accessible during DMA operations while the CPU has control of the bus. If the interrupt is not enabled, Channel 1 CSR can be read to check if the PCT bit is set.

Sending END or EOS

To send the GPIB END message with the last data byte, use the carry cycle feature as described in the discussion *DMA Transfers With A Carry Cycle* earlier in this chapter.

Checking Transfer Results and Handling Interrupts

When any DMA channel is improperly programmed, the ERR bit in the CSR of the active channel will be set by the DMAC and the CER of the channels will indicate a configuration error. Examples of configuration errors include locating the address and transfer count array at an odd address or using a transfer count of zero. If a configuration error occurs, the active channel is terminated automatically and the GPIB transfer can never finish. This will probably cause a time-out error in your program. The following paragraphs assume the channels are properly programmed to allow normal termination in a GPIB DMA transfer. They describe how to check the channel's CSR to determine the success of the GPIB DMA transfers.

The termination of a GPIB DMA transfer will cause an interrupt if interrupts have been enabled in Channel 1 and/or 3. If interrupts are not used, this condition can be detected by polling. Regardless of whether or not the carry cycle feature is used, when a DMA transfer is complete (that is, all data blocks have been transferred) and the GPIB is synchronized (that is, all listeners have accepted the last byte), a high to low transition on PCL1* or PCL3* occurs (depending on which port is active). The PCT bit in CSR1 or CSR3 will be set by this transition. This will generate an interrupt if Channel 1 or 3 is configured to interrupt on a PCL transition. This transition can also be detected by polling CSR1 or CSR3 and waiting for the PCT bit to be set. Once detected, the software must service the interrupt condition.

For simplicity, the following paragraphs describe how to check the transfer result, assuming only Port A is active. It also assumes that a single interrupt handler is used for both Port A and B. If Port B is also activated, its transfer result can be checked by examining the equivalent registers in the TLC of Port B. In addition, while examining DMAC registers for Port B, all references to Channel 0 can be changed to Channel 2 and all references to Channel 1 can be changed to Channel 3.

If Port A and B share the same interrupt handler, then the first step in the interrupt handler is to check which port is interrupting. The PCT bit in CSR1 is set if Port A is interrupting, likewise, the PCT bit in CSR3 is set if Port B is interrupting. If the PCT bit is set in any port, then the interrupt handler must determine the cause of the interrupt on that port.

For each port, an interrupt can be caused by the following:

- An interrupt from the TLC
- A bus error that occurred during a DMA transfer
- The GPIB synchronization event

For example, to determine which above condition caused an interrupt in Port A, you must first examine Channel 1 CSR. If Channel 1 ERR bit is set, then a bus error has occurred while Channel 1 was transferring data. Channel 1 CER must also record a bus error. If Channel 1 is improperly programmed, its operation terminates automatically and the ERR bit is set. This does not cause an interrupt but rather a timeout error in your program. The cause of the error can be found by closely examining Channel 1 CER. (See the CER register description in Chapter 4 for more information.) If Channel 1 ERR bit is cleared while its PCT bit is set (which is the usual case), one of the following conditions may occur:

- A bus error occurred while Channel 0 was transferring data
- The TLC in Port A is interrupting
- Synchronization of Port A's GPIB is detected

If a Channel 0 bus error has occurred, then the ERR bit is set in Channel 0 CSR and Channel 0 CER must indicate a bus error.

Note: A bus error on Channel 0 will only set the PCT bit in Channel 1 CSR but not the ERR bit. Instead, it sets the ERR bit in Channel 0 CSR. A timeout error (no interrupt) occurs if Channel 0 is improperly programmed.

If the PCT bit in Channel 1 is set (Port A is interrupting) but both ERR bits in CSR0 and CSR1 are clear, then either:

- The TLC of Port A is interrupting
- The DMA operation is complete and GPIB synchronization is detected for Port A.

The TLC of Port A must be examined for interrupt conditions (if TLC interrupts are enabled). This can be done by examining the INT bit in ISR2. Since reading ISR1 can clear the INT bit in ISR2, ISR2 should always be read before ISR1. If the TLC did not request an interrupt, then the PCL transition was caused by the GPIB becoming synchronized. To ensure that the DMA transfer was completed, Channel 0 CSR should always be examined. The COC bit must be set and the ERR bit is cleared. This check is sufficient if carry cycle feature was not used.

If the carry cycle feature was used, the CSR bit in Channel 1 must also be examined. The Channel Operation Complete (COC) bit in Channel 1 must *not* be set, and the MTCR must contain 1. This is because the second entry of the address and transfer count array requests a 2-byte DMA transfer but only one byte (the nth data byte) is allowed to transfer. A 2-byte transfer is requested for the following reason: If Channel 1 was allowed to reach terminal count (by requesting a 1-byte transfer), the COC bit would set and an interrupt would be generated (the DMAC will interrupt when either the COC bit or the PCT bit is set). It is convenient to detect if the nth byte has been transferred *and* accepted by all Listeners on the GPIB (indicating a GPIB synchronization). For this reason, Channel 1 is programmed to transfer two bytes to avoid a premature COC interrupt. After the last data byte (the nth byte) is transferred, Channel 1 MTCR

contains 1 and the next DMA request from the TLC is not allowed to reach the DMAC by onboard hardware. Channel 1 is still active, waiting for another request, but never detects one. The PCL transition is generated when the GPIB becomes synchronized.

Note: After a DMA transfer in which the carry cycle was used, Channel 1 must be stopped by issuing a software abort command before another DMA transfer takes place. This is done by writing to Channel 1 CCR with the SAB bit set.

In general for Port A, after a DMA transfer has completed with or without an error, the following actions should be taken.

1. Read ISR2 to clear any interrupt bit set.
2. Read ISR1.
3. Clear IMR1.
4. Clear IMR2.
5. Write a value to CFG1A to unassert Channel 1 PCL line.
6. Write a software abort to Channel 0.
7. Clear the Channel 0 status register.
8. Write a software abort to Channel 1 if a carry cycle is used.
9. Clear Channel 1 status register to clear any interrupt remaining.

If Port B was activated and interrupting, the above process must be used to check the interrupt source from Port B. Instead of checking the TLC and Channels 0 and 1 of Port A, the TLC and DMAC Channels 2 and 3 of Port B must be checked. Finally, remember that the DMAC has separate interrupt vector registers for each DMAC channel and it will send out the vector of the channel that has the interrupt condition pending.

Terminating on END or EOS

The END RX bit in ISR1 is set when a GPIB END message is received. The END RX bit is also set when an EOS message is received and the REOS bit of the AUXR has been previously set. (Receipt of the EOS message is determined by the contents of the DIR, the EOSR, and the value of the BIN bit in the AUXRA.) The END RX bit in ISR1 may be polled during DMA transfers, or if END IE bit in IMR1 and the EINT bit in CCR (Channel 1) are both set, a VMEbus interrupt request (GPIB IR) occurs when END RX is set.

Whether terminating on the END message or the EOS message (or whenever the DMA transfer does not complete properly) the DMAC must be stopped by issuing a software abort to channels 0 and 1 by writing to the CCR with the SAB bit set. You then must write to the CSR in order to clear the ERR flag set by the software abort.

Using Programmed I/O

Programming considerations for using programmed I/O data transfers are explained in the following paragraphs.

Sending and Receiving Data

When using a programmed I/O to send or receive a GPIB data byte, the DMAO and DMAI bits in IMR2 must be cleared. The contents of other GPIB-1014D DMA registers are irrelevant.

For each port, to send data, wait until the TLC has been programmed or addressed to talk and the CDOR is empty. When this occurs, the DO bit in the ISR1 is set, indicating that it is safe to write a byte to the CDOR. The DO bit is set again once that byte has been received by all Listeners.

Similarly, to receive data, wait until the TLC has been programmed or addressed to listen and the DIR is full. When this occurs, the DI bit in ISR1 is set, indicating that the GPIB Talker has written a byte to the DIR. Once that byte has been read, the DI bit is set again when a new byte is received from the Talker.

Determining that the CDOR is empty or the DIR is full can be done by polling ISR1 until the DO or DI status first appears, or by allowing a program interrupt to occur on the respective event.

Sending END or EOS

The GPIB END message is sent by issuing the Send EOI auxiliary command just before writing the last data byte to the CDOR. The GPIB EOS message is sent simply by making the last byte the EOS code.

Terminating on END or EOS

The END status bit or interrupt is used to inform the program of the occurrence of an END message or an EOS message.

Interrupts

The GPIB-1014D has two ports, each of which can generate interrupts on the VMEbus. There are three events that can cause an interrupt on the port (and the VMEbus):

1. An interrupt from the TLC of the port
2. GPIB synchronization that occurs when a DMA transfer is finished and the GPIB is synchronized
3. A bus error which occurs during a DMA transfer

Interrupts must be enabled in software to be recognized.

The Interrupter circuitry of the GPIB-1014D allows the board to interrupt the CPU to request service. The hardware provides programmable selection of the interrupt priority level (level 1 to level 7) via CFG1A or CFG1B.

Note: Ports A and B share a single VMEbus interrupt line.

Interrupts in each Port can be generated on numerous conditions. These conditions can also be detected by polling. Each port, if programmed to interrupt, can generate an interrupt on any of the following conditions:

1. An interrupt request from the Port's TLC.

2. A bus error which occurs during a DMA transfer.

3. A DMA transfer from memory to the GPIB is complete and the GPIB is synchronized (that is, all devices have accepted the last byte).

   or

   A DMA transfer from the GPIB to memory is complete and the GPIB is synchronized.

For each port, the three interrupt events above are ORed together and routed to a pin on the DMAC which is programmed to be a status input. For Port A, pin PCL1* is used and for Port B pin PCL3* is used. The assertion of either of these pins will generate an interrupt request on the VMEbus. When programmed as a status input, the status level of the PCL can be determined by reading the PCS bit in the CSR. If a negative transition occurs on this input, the PCT bit of the CSR is automatically set. Any of the four DMAC channels may be programmed to generate an interrupt on a negative transition (high to low) on its PCL line. This enables an interrupt which is requested when the PCT bit of the CSR is set, indicating that a transition has occurred. If an interrupt operation is used, the DMAC Channel 1 for Port A or Channel 3 for Port B must be configured to interrupt on a high to low transition of the PCL1* or PCL3*. Channel 0 of Port A and Channel 2 of Port B are not enabled for interrupt in most GPIB applications.

The TLC contains its own internal registers that are used to control and enable interrupts. The interrupt output from the TLC, however, is routed to the PCL of the DMAC and the DMAC will then issue an interrupt to the VMEbus.

If the DMAC encounters a bus error during operation, a negative transition is caused on PCL1* or PCL3* depending on which port is active when the bus error occurs.

The GPIB-1014D hardware provides automatic GPIB synchronization after the data transfer is complete. This allows an interrupt (or a set status bit for polling) when the last data byte has been transferred and the GPIB is synchronized (that is, all devices on the GPIB have accepted the last byte). When the DMA transfer is complete and the GPIB is synchronized, a negative transition is also generated on the PCL1* or PCL3*.

Since the PCL detects a high to low transition on its input, the input, once pulled low to generate an interrupt, must be pulled back to high by your program for the PCL to work properly on the next transition. For example, if an interrupt from the TLC of Port A pulls PCL1 low, your program must first clear the PCT bit in CSR1 to release the VMEbus IRQx* line then read ISR1 or ISR2 of the TLC to clear the TLC interrupt of the TLC and pull PCL1 high. If GPIB synchronization occurs, you must write any value to the CFG1 of the port to pull PCL high. If a VMEbus error occurs during a DMA transfer, the PLC1 is pulled high automatically when the VMEbus memory releases signal BERR* high.

After an interrupt is generated, the operating system will ask the interrupting source (the DMAC) for a status/ID byte so that it can branch to the appropriate interrupt handler. Depending on the value of the ERR bit, Channel 1 for Port A and Channel 3 for Port B will send out the Normal Interrupt Vector (NIV) or Error Interrupt Vector (EIV). Since all errors are logged, it is possible and convenient to use the same vector for NIV and EIV and the same interrupt handler for both ports. The port and the source of the interrupt from each port must be determined. You must clear the PCT bit in CSR1 for the DMAC to release the VME bus interrupt line. The status of the TLC interrupt can be found by reading the appropriate TLC status registers.

The status bits in ISR1 or ISR2 are all automatically cleared when the register is read, even if the conditions are still true. If two conditions are true at the same time (that is, more than one bit in ISR1 or ISR2 is set), a software copy of the register must be maintained if the program is going to analyze the conditions one at a time. Since reading ISR1 can clear the INT bit in ISR2, ISR2 must always be read *before* ISR1.

The PCL lines of Channels 0 and 2 are also connected to two GPIB signals for you to directly detect them. The PCL of Channel 0 is connected to the GPIB Service Request signal (active low) of Port A. This can be used to read the status of the SRQ* line at any time or to generate an interrupt when the SRQ* line is asserted and the TLC is Controller-in-Charge. The PCL of Channel 2 is connected to either the GPIB Remote Enable signal (active low) of Port A or the SRQ signal (active low) of Port B. An onboard switch is used to select which of these two signals to monitor. PCL2* can be used to read the status of the GPIB line at any time or to generate an interrupt when the GPIB line is asserted.

Serial Polls

Conducting Serial Polls

Any of the two onboard TLCs, as CIC, serially polls other devices as described in the IEEE-488 specification. From the programming point of view, the TLC must first become Active Controller to send the addressing and enabling commands to the device being polled, make itself a GPIB Listener by issuing the Listen auxiliary command, and then go to standby with the Go To Standby auxiliary command in order to read the status byte.

Responding to a Serial Poll

The CIC can conduct Serial Polls to determine which device is asserting the GPIB SRQ signal to request service. Since the board has two ports, the following procedure must be used by each port.

Before requesting the service, the recommended practice is to wait until the PEND bit in the SPSR is zero, indicating that the TLC is not presently in the middle of a Serial Poll (SPAS=0). If PEND=0, write the desired Status Byte (STB) into the SPMR with the rsv bit set. At that time, PEND sets and remains set until the Serial Poll completes.

Once rsv is set, the TLC waits until any current Serial Poll is complete and then asserts the GPIB SRQ signal. In response to that signal, the CIC starts the poll, addressing the TLC to talk. When the CIC unasserts ATN, the TLC unasserts SRQ and transfers the STB message onto the GPIB data bus with DIO7 (the RQS signal) asserted.

RQS, rsv, and PEND are cleared when the CIC asserts ATN to terminate the poll.

The GPIB EOI line is asserted along with the status byte (that is, the END message is sent) during the serial poll if bit B1 of the AUXRB is set.

Parallel Polls

Parallel Polls are used by the GPIB Active Controller to check the status of several devices simultaneously. The meaning of the status returned by the devices being polled is device-dependent. There are two general ways in which Parallel Polls are useful:

- When the GPIB Controller sees SRQ asserted in a system with several devices, it can quickly determine which one needs to be serially polled usually using only one Parallel Poll.
- In systems in which the Controller response time requirement to service a device is low and the number of devices is small, Parallel Polls can replace Serial Polls entirely, provided that the Controller polls frequently.

Although the Controller can obtain a Parallel Poll response quickly and at any time, there can be considerable front-end overhead during initialization to configure the devices to respond appropriately. Whereas the overhead, (in the form of addressing and enabling command messages in Serial Polls) occurs with each poll.

Conducting a Parallel Poll

Any of the two onboard TLCs as Active Controller has the capability to conduct a Parallel Poll. When the Execute Parallel Poll auxiliary command is issued, the Parallel Poll is executed (that is, the GPIB message IDY is sent true) as soon as the TLC Controller interface function is placed in the proper state (CAWS or CACS). The Parallel Poll Response (PPR) is automatically read from the GPIB DIO lines into the CPTR and the rpp local message is cleared. A program can determine that the Parallel Poll operation is complete based on the condition of CO (CO=1 when the poll is complete). The response can be obtained by reading the contents of the CPTR. The response is held in the CPTR until a GPIB command is transmitted or the TLC Controller function becomes inactive.

In response to IDY, each device participating in the Parallel Poll drives one and only one GPIB DIO line (its Parallel Poll response or PPRn) active true or passive false, while it drives the other lines passive false.

Since there are eight data lines, and for each line there can be only one response (true or false) for each device (two lines/device), there are 16 possible responses. The line that a device uses and how that device drives the line depends on how it was configured and whether its local individual status (ist) message is one or zero. Thus, each device on the GPIB can be configured to drive its assigned DIO line true if ist=1 and to drive the DIO line false if ist=0; or it can be configured to do exactly the opposite; that is, to drive the DIO line true if ist=0 and false if ist=1. (The meaning of the value of ist, whether one or zero, is system- or device-dependent.)

Because the data lines are driven Open Collector during Parallel Polls, more than one device can respond on each line. The device or devices asserting the line true overrides any device asserting the line false. The Controller must know in advance whether a true response means the

local ist message of the device is one or zero. To do this, the device must be configured to respond in the desired way. Two methods can be used to accomplish this:

- Local configuration (Parallel Poll function subset PP2) involves assigning a response line and sense from the device side in a manner similar to assigning the device GPIB address. Thus, one device might be assigned to respond with remote message PPR1 (driving DIO1), while a second device might be assigned to respond with the remote message PPR3 (driving DIO3), both positive (that is, true response if ist=1). Local configuration is static in that it does not change after the system is integrated (that is, hardware configured and installed).

- Remote configuration (Parallel Poll function subset PP1) involves the dynamic assigning of the response line and sense to devices on the GPIB. This is accomplished using Parallel Poll Enable (PPE) and Parallel Poll Disable (PPD) commands, which are issued by the Active Controller. The sequence for remotely configuring devices on the GPIB is as follows:

  1. Become Active Controller.

  2. Send the GPIB UNL message to unaddress all GPIB Listeners.

  3. Send the listen address of the first device to be configured.

  4. Send the GPIB PPC message to all devices followed by the PPE message for that device.

  5. Repeat from the second step (UNL) for each additional device.

The same procedure should be followed to disable polling with PPD (for example, when changing responses during reconfiguration).

Responding to a Parallel Poll

Before the GPIB-1014D can be polled by the CIC, both TLCs must be configured either locally by your program at initialization time or remotely by the CIC. Configuration involves the following:

- Enabling the TLC(s) to participate in polls

- Selecting the sense or polarity of the response

- Selecting the GPIB data line on which the response will be asserted when the CIC issues the IDY message

With remote configuration (PP1), the TLC(s) interprets the configuration commands received from the CIC, without any software assistance or interpretation from your program. With local configuration (PP2), the three actions listed must be explicitly handled in the software by writing the appropriate values to the U, S, and P3 through P1 bits of the PPR. Refer to the PPR description in Chapter 4 for more information.

Once the PPR is configured, all that remains for your program is determining the source and value of the local individual status (ist) message. If the ISS bit in the AUXRB is zero, ist is set and cleared via the Set and Clear Parallel Poll auxiliary commands. If ISS is one, ist is set if the Service Request function of the TLC is in the Service Request State (SRQS) and the TLC is

asserting the GPIB SRQ signal line and cleared otherwise. Consequently, setting ISS ties the Parallel Poll function to the Service Request function and also to the Serial Poll process.

As explained under *Conducting a Serial Poll* earlier in this chapter, one of the 16 possible responses that is transmitted by the TLC during a Parallel Poll is a function of the value of ist and the configuration of the TLC. What ist indicates and what configuration is used must be decided by the GPIB system integrator. The response can be changed dynamically during program execution by changing the value of ist and, when remote configuration is used, by reconfiguration.

Software Porting

Porting GPIB-1014 Software to GPIB-1014D

As described in Chapter 3 under *Select Interrupt Source for DMAC Channel 2*, if switch W3 is set to RENA*, Port A of the GPIB-1014D is completely software compatible with the GPIB-1014. This assumes that the GPIB-1014 software uses both PCL0* and PCL2* to monitor SRQ* and REN*, respectively. If the GPIB-1014 software does *not* use either PCL0* and PCL2* to monitor SRQ* and REN*, but rather uses only DMAC Channel 0 and 1 to transfer data and PCL1* to monitor onboard interrupt, it is easy to convert that software to work on both ports of the GPIB-1014D. The following steps list the sequence in converting the GPIB-1014 software:

1. Notice that Port A's registers start at the base address and Port B's registers start at base address + 200 (hex); that is, Port B appears as another GPIB-1014 board located at base address + 200 (hex). The old GPIB-1014 software works unmodified with Port A. By changing the base address in the software to base + 200(hex), the new software works on Port B as well. In addition, all read/write to Channel 0 and 1 of the DMAC in the old software will be *automatically* mapped to Channel 2 and 3 of the DMAC. You do *not* need to explicitly change all references to Channel 0 and 1 DMAC registers to Channel 2 and 3. As an example, if Channel 0's MTCR locates at base address + 0A (hex), then the MTCR of Channel 2 is accessible when you specify either Port B's base address + 0A (hex) or the base address + 8A (hex) of the board. Similarly, with the new base address, all read/write to the TLC and Configuration Registers will be directed to Port B's TLC and Configuration Registers.

2. Since Port A and B share a single VMEbus interrupt line, the interrupt handler in the old software must be changed to examine Port B as well as Port A. If a VMEbus system allows Port A and B to send out two different status/ID bytes during the interrupt acknowledge cycle, you simply need to add another interrupt handler–one to examine Port A's registers and another to examine Port B's. If only one interrupt handler is allowed for the GPIB-1014D, then at the beginning of the handler, you must determine the interrupt port. Once the interrupt port is determined, the old handler can then be used, unchanged, to determine the exact source of interrupt in the interrupting port.

3. If the 32-bit addressing feature is used, then the old software must ensure that the 32-bit bit in CFG2 is set (switch W9 must also be set to 32BIT). In addition, you must load the Page Register accordingly to access the proper page in 32-bit space.

Chapter 6
Theory of Operation

This chapter explains the operation of each functional unit making up the GPIB-1014D. A brief description of the GPIB-1014D interface board along with a functional block diagram can be found in Chapter 2.

The following are the major components making up the GPIB-1014D board:

- VMEbus Interface
- Address Decoder
- Clock and Reset Circuitry
- Configuration Registers
- Timing State Machine Circuitry
- DMA Gating and Control Circuitry
- Interrupter Circuitry
- DTB Requester and Controller (68172 BUSCON) Circuitry
- GPIB Synchronization and Interrupt Control Circuitry
- DMA Controller (MC68450L8)
- GPIB Talker/Listener/Controller (NEC μPD7210)

The internal data and control buses interconnect the components. The theory of operation of each of these components is explained in the remainder of this chapter.

Note: Signal names in the following discussion are referenced in terms of logic value (true or false, and asserted or not asserted), and also in terms of logic level (TTL high or low). Both positive and negative logic symbols are used in the schematic diagram. The terms *clear*, *negate*, *unassert*, *reset*, and *set false* are synonymous as are *set*, *assert*, and *set true*.

VMEbus Interface

Address, data, control, and status signals to or from the VMEbus are buffered with LSTTL, ASTTL, or FTTL logic devices. All drivers drive the proper amount of current as required by the VMEbus specification and all receivers present the required bus load as called out by the VMEbus specification.

Data Lines

Two F245 octal bus transceivers connect the VMEbus data lines (D15 through D00) to the circuitry on the GPIB-1014D. All 16 of these data lines are routed directly to the DMAC, while only the lower eight data lines (D07 through D00) are connected to the 8-bit data bus of the two TLCs.

Slave Read and Write Transfers

During slave read/write to the onboard registers, both transceivers are enabled but either one or two are driven with correct data depending on the size of the transfer (byte or word). For example, during a read to the TLCs, only the lower eight data lines are driven with correct data. The GPIB-1014D drives the VMEbus data lines during onboard register reads and receives (is driven by) the data on the VMEbus during onboard register writes.

DMA Transfers

During DMA transfers, the 8-bit data bus of the TLC can be directed to either the lower eight bits (D7 through D0) or the upper eight bits (D15 through D08) of the VMEbus data bus. This is accomplished using an F245 8-bit data transceiver that gates the upper data byte to the TLCs. This data transceiver is automatically controlled by the DMAC signal HIBYTE*. When the data transfer is on VMEbus data lines D07-D00, the HIBYTE* is not active and the data bus of the TLC is connected to the lower eight bits of the VMEbus data bus. When the data transfer is on VMEbus data lines D15 through D09, HIBYTE* is active to connect the data bus of the TLC to the upper eight bits of the VMEbus data bus. This feature allows the 8-bit GPIB data to be packed in the 16-bit VMEbus memory. When the GPIB-1014D is acting as a VMEbus master, it drives the VMEbus data lines on VMEbus memory writes and receives from (is driven by) the VMEbus data lines VMEbus memory reads.

Control Signals

An F1243 transceiver is used to transceive the data bus control signals WRITE*, DS0*, and DS1*. The BUSCON transceives signal AS*. The OWN* signal of the DMAC is used to control the direction of the F1243 and BUSCON. When the GPIB-1014D is a slave (that is, it does not have control of the bus), control signals are directly routed onboard. In contrast, during DMA cycles, control signals from the DMAC are somewhat altered before passing out to the VMEbus. For example, signal WRT* of the DMAC is ORed with the onboard signal CCBYTE to implement the carry byte cycle (see *DMA Gating and Control* later in this chapter). Signals UDS* and LDS* of the DMAC are delayed by the timing state machine output before being sent to the VMEbus (see *Timing State Machine* later in this chapter). When signal MAS* of the DMAC is asserted, BUSCON will assert signal AS* on the VMEbus.

Address

When the board is a VMEbus slave, address information A15-A10, AM5-AM0, IACK*, and LWORD* are received and decoded by two LS2521 8-bit comparators. (For more information, see Chapter 5, *Address Decoding*.)

When the board is a VMEbus master, it drives the address lines (A32-A1), the address modifier lines (AM5-AM1), IACK*, and LWORD*. The address modifier lines (AM5-AM1), IACK*, and LWORD* are driven by an F244. Address lines A32-A24 are driven by an AS374 driver. The upper 16 bits of the address are driven by two transparent D-type AS573 flip-flops while the lower eight bits are driven by an F245. The UAS* signal of the DMAC is used to clock the AS573s. The signal OWN* of the DMAC enables all address transceivers and drivers.

Address Decoder

During non-DMA operations, the GPIB-1014D acts as a VMEbus slave and monitors the lower 16 lines of the VMEbus address bus (A15 through A01), the Address Modifier Lines (AM5 through AM0), and the VMEbus signals LWORD* and IACK*. Receivers and comparators are used to recognize the GPIB-1014D base address during short I/O transfers.

Two 25LS2521 comparators recognize the short I/O base address of the GPIB-1014D. The base address is selectable with switches located on the board. The base address of the GPIB-1014D is recognized whenever the VMEbus signal AS* is active, IACK* is inactive, the Address Modifier lines indicate a short I/O operation, and VMEbus address lines A15-A10 match the selected base address. BUSCON will assert local signal SLVSEL* when the above condition is true. The board will not start the slave read/write cycle until DS* is asserted, at which point local signal BRD_SLV_CYCLE* becomes active. Further decoding is necessary to determine which register is being addressed.

Address line A9 is used to determine whether Port A or B registers are accessed. If A9 is clear then the configuration registers of Port A and Channels 0 and 1 of the DMAC are accessed; otherwise, the configuration registers of Port B and Channels 2 and 3 of the DMAC are accessed.

When BRD_SLV_CYCLE* is active and VMEbus address line A8 is a logic zero, the DMAC select signal DMACS* is driven active, indicating that the DMAC has been addressed. Address lines A7 through A1, DS0* and DS1*, are then used to select one of the internal registers of the DMAC. The DMAC can function as an 8- or 16-bit slave. All registers within the chip can be accessed with 8- or 16-bit transfers. A 16-bit access to an 8-bit register will return arbitrary data in the unused bits. Reading from unused locations within the address range occupied by the DMAC results in a normal bus cycle but returns all ones. DMAC will assert the DTACK* to prevent a bus error. A write to an unused location results in a normal bus cycle but no write occurs. Like above, DMAC will assert DTACK* to prevent a bus error. Port A uses DMAC's Channels 0 and 1 registers while Port B uses Channel 2 and 3 registers. When address line A9 is high, it forces address line A7 to be high allowing accesses to Channel 2 and 3 of the DMAC. Channel 2 and 3 registers can also be accessed if user explicitly sets A7 high. Address line A9 thus provides a convenient way of mapping read/write accesses to Port A's DMAC registers (Channel 0 and 1 registers) to Port B's DMAC registers (Channel 2 and 3 registers).

When BRD_SLV_CYCLE* is active and VMEbus address line A8 is a logic 1, A4 is a logic 1, the lower data strobe DS0* is active, TLC select signals TLCCSA* or TLCCSB* is driven active, indicating that one of the two TLCs has been addressed. Address line A9 determines which Port's TLC is selected. Like above, when A9 is set, Port B's TLC is selected, otherwise Port A's TLC is selected. Address lines A3-A1 are used to select the internal registers of the TLCs. These lines are gated with the onboard signal CCBYTE and are connected to the register select lines RS2-RS0 of the TLC. The TLCs function as 8-bit slaves. They transfer data on the lower eight bits of the VMEbus data bus. The TLC registers are located at consecutive odd addresses and will respond only when DS0* is asserted. The TLC will respond to 16-bit transfers, but the upper data byte is not used during an I/O write and will return all ones during

an I/O read. As described in chapter *DMA Gating and Control*, signals CCBYTEA or CCBYTEB will force a write cycle to the Auxiliary Mode Register inside the TLCA or TLCB when it is asserted.

When BRD_SLV_CYCLE* is active and VMEbus address line A8 is a logic 1, A4 is a logic 0, A2 is a logic 0, either Port A or B's Configuration Register 1 is addressed (CFG1A or CFG1B). When A8 is a logic 1, A4 is a logic 0, A2 is a logic 1, either Port A or B's Configuration Register 2 is addressed (CFG2B or CFG2B). Like above, address line A9 will select Port A or B. When A8 is a logic 1, A4 is a logic 0, A3 is a logic 1, the Page Register is accessed (Port A and B share a single Page Register). CFG1A/B and CFG2A/B respond only to I/O write operations (write-only) and are loaded via the lower eight data lines. They can be accessed with an 8- or 16-bit I/O write operation. The GPIB status registers for Port A or B are located at the same address as their port's CFG1A or CFG1B. A read from either of two locations will give the status of the GPIB lines from each Port on VMEbus data line D07-D00. A read from CFG2A, CFG2B, and Page Register will result in a normal bus cycle but will return all ones.

Accesses to locations within the 1024 byte block that do not specifically address either the DMAC, the two TLCs, the Page Register or one of the Configuration Registers will result in a bus error because the GPIB-1014D will not respond.

Clock and Reset Circuitry

The clock and reset circuitry on the GPIB-1014D is used to generate the onboard clock and reset signals. The VMEbus signal SYSCLK (16 MHz) is received with a 74LS241 receiver and is divided by a 74LS74A flip-flop to generate the onboard 8 MHz clock signal (CLK). This clock is used by the TLC. The 16 MHz clock is used by the Timing State Machine to control the timing of local signal DTACK* when the board is a slave and to control RD* and WR* signals to the TLC (see *Timing State Machine*).

The VMEbus signals SYSRESET* is monitored by the GPIB-1014D. It is received with a 74LS241 receiver and is ORed with the LMR bit in CFG2 to generate the onboard RESET signal. The RESET signal is used to initialize all circuitry on the GPIB-1014D except the two bits in CFG2 (LMR and SYSFAIL*), the Address Decoding circuitry, the BUSCON, and the Timing State Machine. The two bits in CFG2 are reset only by the SYSRESET* signal or by a write to CFG2. The Address Decoding circuitry, the BUSCON, and the Timing State Machine can only be reset by SYSRESET*.

The VMEbus signal BERR* is monitored by the GPIB-1014D while it is bus master. If asserted, BUSCON will assert local signal LBERR*. LBERR* is ANDed with the onboard signal OWN*, which is active while the GPIB-1014D is bus master, to drive the DMAC input BEC1, indicating to the DMAC that the cycle cannot be completed. The DMAC responds to this condition by terminating the cycle and indicating a bus error condition in the channel CER.

Configuration Registers

The GPIB-1014D is divided into two ports, Port A and Port B, each of which contains an identical set of configuration registers.

For each of the two Ports A and B, two registers (CFG1 and CFG2) are 8-bit write-only registers used by the controlling software program (application program and/or interface handler) to configure some of the operating characteristics of the GPIB-1014D. The GPIB status register for

each port, located at the same location as CFG1, provides the state of the GPIB lines for each port. The Page Register, a write-only register that is accessible from either Port A or Port B, provides VMEbus address lines A32-A24 with correct information during a DMA operation.

Configuration Register 1 (CFG1A or CFG1B)

There are six bits in CFG1A and CFG1B that specify board-specific configurations such as interrupt level, bus request/grant level, and so on. These six bits are implemented by the same hardware. There are two bits that are port-specific and these bits are implemented by separate hardware. Data is written into these bits on the rising edge of the WCFG1A* or WCFG1B* signals generated by the Timing State Machine circuitry. Except for the ROR* bit, all other bits in CFG1 are cleared by the onboard RESET* signal which is generated by the Clock and Reset circuitry.

The overlapping bits are used for a variety of board-specific configuration functions, including:

- Selecting the VMEbus Interrupt Request Priority for the GPIB-1014D. The three most significant bits are used to encode the desired interrupt priority. This encoded value is used by the Interrupter circuitry to drive the appropriate VMEbus Interrupt Request line and to identify an interrupt acknowledge cycle of the proper priority.

- Selecting the VMEbus Bus Request/Grant line. Two bits are used to encode the desired VMEbus Bus Request/Grant line. This encoded value is used by the DTB Requester and Controller circuitry to drive the appropriate Bus Request line and to monitor the appropriate Bus Grant In line.

- Enabling the Release On Request feature. Writing a zero to this bit (ROR*) enables the Release On Request feature while writing a one disables the Release On Request feature. This bit is set to one during reset or power up. This bit is used by the DTB Requester circuitry.

The two port-specific bits are used to configure port-specific parameters, including:

- Enabling the automatic carry cycle feature. Writing a one to this bit (CC) enables the carry cycle feature while writing a zero disables the carry cycle feature. This bit is used by the DMA Gating and Control circuitry. Either one or both Ports can use the carry cycle feature.

- Selecting the direction of the DMA transfer. Writing a one to this bit (DIR) the transfer direction is from GPIB to VMEbus memory, while writing a zero indicates the direction is from VMEbus memory to the GPIB. This bit is used by the GPIB Synchronization and Interrupt Control circuitry. The Ports can have different transfer directions.

Lastly, writing any value to CFG1A or CFG1B resets the separate circuitry which detects GPIB synchronization for each Port after a DMA transfer. (For more information, see *GPIB Synchronization and Interrupt Control* later in this chapter).

Configuration Register 2 (CFG2A or CFG2B)

There are four bits in CFG2A and CFG2B that specify board-specific configurations and are implemented by the same hardware. There is one bit that is port-specific and is implemented by separate hardware. Data is written into each bit of this register on the rising edge of the

WCFG2A* or WCFG2B* signals generated by the Timing State Machine circuitry. The SC bit is cleared by the onboard RESET* signal which is generated by the Clock and Reset circuitry, while the LMR and SFL bits are cleared only by the VMEbus signal SYSRESET* or by writing to the register.

The overlapping bits are used for a variety of board-specific configuration functions, including:

- Issuing a Local Master Reset (LMR) to the GPIB-1014D. The LMR bit is cleared to a zero on SYSRESET*. Writing to CFG2 with this bit set to one will drive the GPIB-1014D local signal RESETA* or RESETB*. Writing a zero to this bit will release the Reset signal.
- Selecting Supervisor-only or User access to the GPIB-1014D. A switch on the GPIB-1014D is used to automatically determine the state of this bit (SUP) after a system or local reset. After reset, you can write a one to this bit to select Supervisor-only access or a zero to select User access.
- Controlling the GPIB-1014D LED SYSFAIL* indicator. If this bit (SFL) is a zero, the VMEbus SYSFAIL* line is driven low with an F38 open-collector driver and the Bi-Color Light Emitting Diode (LED) on the GPIB-1014D turns red. Writing a one to this bit releases the VMEbus SYSFAIL* line and turns the LED green. Driving the VMEbus SYSFAIL* can be disabled by setting switch W2 to position 0. This bit is set to zero on SYSRESET* and is unaffected by a local master reset (LMR).
- Enabling 32-bit Extended Addressing. If switch W9 is set for 32-bit and this bit is active, the GPIB-1014D will set the VMEbus Address Modifier Lines (AM5-AM0) to indicate Extended Addressing.

The port-specific bit is used to configure port-specific parameters, including configuring Port A or B to be System Controller. Writing a one to the SC bit will configure the port as System Controller. This bit is used by the 75162 GPIB control transceivers to determine whether the port drives or receives the GPIB signal IFC. If this bit is clear in both ports, the GPIB-1014D will receive signal IFC.

GPIB Status Register (GSRA or GSRB)

For each port, the status register is located at the same address as CFG1. The inputs of these buffers are connected directly to the GPIB lines of each port. One of the two buffers will drive VMEbus data line D00-D07 when local signal RCFG1A* or RCFG1B* is low.

Page Register (PGREG)

A single AS374 D-type transparent flip-flop is used to implement the Page Register for both Ports. During DMA cycles, the content of the Page Register is driven into VMEbus address line A32-A24.

Timing State Machine Circuitry

The Timing State Machine (TSM) circuitry is designed to control the timing of local signal LDTACK* in slave cycles and the RD* and WR* signals that control the TLC in both the slave

and DMA cycles. During a DMA cycle that transfers data from GPIB to memory, the TSM helps to delay the assertion of VMEbus signals DS0* and DS1* until the TLC has supplied valid data.

Slave Cycles

When the GPIB-1014D is functioning as a VMEbus slave, and the DMAC has been addressed, local signal LDTACK* (and thus VMEbus signal DTACK*) is driven by the signal DDTACK* of the DMAC. The timing is controlled entirely by the DMAC. When the TLCs or one of the read/write Configuration Registers has been addressed, however, the timing is controlled by the TSM. The following paragraphs describe the operation of the TSM doing slave read/write cycles.

The TSM starts (signal TSM_START/STOP* is high) when one of the TLC select or register select signals is asserted, indicating that one of the TLCs or registers has been addressed. During a read cycle to either one of the TLCs, the RD* signal of the TLC is immediately driven low. The circuitry then counts a minimum delay of 250 nsec, which corresponds to the read access time of the TLCs. Output signal PDTACK of the TSM is asserted to drive local signal LDTACK* (and thus VMEbus signal DTACK*) active to indicate that the data is valid on the VMEbus data lines D07 through D00. The data remains valid until DS0* is released by the bus master. Local signals PDTACK and LDTACK* are kept asserted by a flip-flop until both DS0* and DS1* are released by the bus master. When DS0* is released, signal TSM_START/STOP* is low to start the recovery cycle and release the RD* signal. The TSM counter will count for a recovery time of 250 nsec. Signal END_RECOVERY is asserted at the end of the 250 nsec period to reset the TSM. At this point, the TSM will allow another read cycle issued to the TLCs or registers.

During a write cycle to the TLCs or one of the onboard registers, the timing is similar. Once DS0* is asserted, the TLCs' WR* line is driven low immediately. The WR* signal is ANDed with each register select signal to drive the clock inputs of those registers. Once WR* is driven low, the circuitry counts a data setup time of 250 nsec. TSM output signal PDTACK is then asserted to drive LDTACK* (and subsequently VMEbus signal DTACK*) low. At the same time, the WR* signal is immediately driven high to latch the data into the TLCs or the onboard registers. When DS0* is released, signal TSM_START/STOP* is low to start the recovery cycle. The TSM counter will count for a recovery time of 250 nsec. Signal END_RECOVERY is asserted at the end of the 250 nsec period to reset the TSM. At this point, the TSM will allow another write cycle issued to the TLCs or registers. Notice that recovery cycle is necessary to prevent another read or write cycle to the TLCs.

DMA Cycles

The TSM also controls the timing of the RD* and WR* signals to the TLCs during a DMA operation. The sequence is similar to a normal TLC access, except the 74S139 decoder is used to drive the TLC RD* and WR* signals as required to effect the DMA transfer. The following paragraphs describe the operation of the TSM doing a DMA read/write cycle. The DMAC begins a DMA transfer for the TLC by driving one of its DMA acknowledge lines (ACK3* - ACK0*) low. The TMS also starts (signal TSM_START/STOP* is high) when this is detected. When the transfer is from the TLC to the VMEbus memory, the DMAC drives the VMEbus WRITE* line low, while the timing circuitry drives the TLC RD* line low. The timing circuitry counts 250 nsec data access time and the TSM then asserts PDTACK to gate the DMAC data strobe signals UDS* and LDS* onto the VMEbus to drive the DS1* and DS0* lines. The DMAC continues to drive the data strobes and the TLC continues to drive the data lines (D15-

D08 or D07-D00) with correct data until the memory drives VMEbus DTACK*. When DMAC has received DTACK* low, it first releases its data strobes (UDS* and LDS*), then the DMA acknowledge lines (ACK3* - ACK0*). When ACK3* - ACK0* are released, signal TSM_START/STOP* is low to start the recovery cycle and release the RD* signal. The TSM counter will count for a recovery time of 250 nsec. Signal END_RECOVERY is asserted at the end of the 250 nsec period to reset the TSM. At this point, the TSM will allow another read cycle to be issued to the TLCs or registers.

When the DMA transfer is from VMEbus memory to the TLC, the sequence is similar. The timing circuitry starts when one of ACK3* - ACK0* is detected low. The DMAC drives the VMEbus data strobes as required and drives the VMEbus WRITE* line high, while the timing circuitry drives the TLC WR* line low. Unlike the case when the board reads the TLC and writes to the memory, data strobes from DMAC (UDS* and LDS*) are not held off by TSM signal PDTACK. When the memory responds with DTACK*, indicating that the data is valid on the bus, the DMAC waits a minimum of 190 nsec then releases the data strobes. When the timing circuitry sees the data strobes released, it immediately drives the TLC WR* line high, latching the data into the TLC. When the DMAC has released ACK3* - ACK0*, the timing circuitry then counts a recovery time of 250 nsec while the DMAC and memory finish the cycle.

DMA Gating and Control Circuitry

The DMA Gating and Control circuitry is designed to control the DMA request/acknowledge interface between the DMAC and the two TLCs. There is separate circuitry for each Port A and each Port B. Port A uses Channel 0 and/or Channel 1 of the DMAC for its DMA operation whereas Port B uses Channel 2 and/or 3. Each DMA Gating and Control circuitry consists of a flip-flop and miscellaneous logic gates to generate the DMA request signals (DREQ0* and/or DREQ1* for Port A and DREQ2* and/or DREQ3* for Port B), carry cycle byte (CCBYTEA or CCBYTEB) strobe. Depending on the value of CCBYTEA or CCBYTEB, a flip-flop generates a DMA Acknowledge Enable (DACKEN*) signal for both ports.

The main function of the DMA Gating and Control circuitry is to direct the DMA request signal from the TLCs (DMAREQs) to the proper DMA channel of the DMAC in order to perform the carry cycle function; that is, the carry cycle feature uses one DMAC channel to transfer the first N-1 data byte and another channel to transfer the GPIB command byte and the last (nth) byte. The GPIB command byte is also called the carry cycle byte. If programmed to use the carry cycle feature, Port A uses Channel 0 and 1, while Port B uses Channel 2 and 3.

Upon RESET, the DMA requests from the TLCs are directed to Channel 0 for Port A and to Channel 2 for Port B. This configuration remains unchanged unless Port A and/or B have selected to use the carry cycle feature. Depending on the port, the DMA requests are gated to Channel 0 or 2 until the DMAC DONE* signal is active during a DMA transfer, indicating that Channel 0 or 2 has completely finished (that is, all the blocks in the chain have been transferred). Depending on which channel (0 or 2) is active when DONE* is asserted, the circuitry automatically gates the DONE* pulse from Channel 0 to the DMA request pin (REQ1*) of Channel 1, requesting a transfer (DREQ1* is asserted). Similarly, if Channel 2 is active when DONE* is asserted, the DREQ3* is automatically asserted. There is only one DONE* signal for all four DMA channels.

When the DMAC answers the request on Channel 1 with ACK1* asserted, CCBYTEA is asserted to indicate that the carry cycle byte transfer is in progress for Port A. Similarly, when the DMAC answers the request on Channel 3 with ACK3* asserted, CCBYTEB is asserted to indicate that the carry cycle byte transfer is in progress for Port B. During either carry cycle byte

transfer, signal DACKEN* is kept high to prevent the circuitry from asserting the DMAACK* signal of DMAC. In every DMA transfer (except a carry cycle byte transfer), DACKEN* is low, causing the DMAACK* signal of either TLCs to go low, to acknowledge a DMA request from the TLCs. The TLCs can keep their DMA request pin asserted during the carry cycle transfer but will not be acknowledged until the carry cycle has completed. In addition, since Channel 0 or 2 has stopped after DONE* was asserted, DREQ0* or DREQ2* can be asserted by the circuitry but has no effect.

The first byte to be transferred from Channel 1 for Port A or Channel 3 for Port B is the carry cycle byte. This byte, a GPIB command, must be transferred by the DMAC to the auxiliary registers of the TLC. Since the TLCs do not allow DMA transfers to the auxiliary registers, discrete circuitry must make this cycle appear as a normal write cycle to the TLCs. Furthermore, the circuitry must ensure that this byte is always transferred from memory to the TLCs even though the DMAC may be configured to transfer the data from the GPIB ports to the VMEbus memory. The circuitry knows that the first byte to be transferred from Channel 1 or 3 is always the carry cycle byte, so when ACK1* or ACK3* is driven low at the start of DMA cycle, the onboard signals CCBYTEA or CCBYTEB becomes active. These signals are ORed with the DMAC WR* signal to ensure that the onboard circuitry performs a read from the VMEbus memory and a write to one of the TLCs. Signal CCBYTE (which is the OR result of CCBYTEA and CCBYTEB) is also used to gate the register select lines RS2-RS0 of the TLC to drive them as needed to address the auxiliary register. Signals CCBYTEA and CCBYTEB are used to drive the TLCCSA* or TLCCSB* line instead of the DMAACKA* or DMAACKB* line. As mentioned above, DACKEN* is kept high during the carry byte cycle to prevent the DMAACKA* or DMAACKB* line from asserting. The TSM circuitry and the TLCs interpret the cycle as a normal write (since the CS* and WR* of the TLCs are asserted), while the VMEbus memory interprets the cycle as a DMA read from memory.

The DMAC signal ACK1* or ACK3* goes high at the end of the transfer of the carry cycle byte, and this low to high transition is used to set a flip-flop in each circuitry of the port that gates further DMA requests from the TLCs to DMAC Channel 1 or 3 and disables generation of the CCBYTEA or CCBYTEB signals. When the TLCs issue another DMA request, Channel 1 or 3 respond by transferring the last byte of data with a normal DMA transfer cycle. The low to high transition of ACK1* or ACK3* at the end of this cycle clears the flip-flop in each circuitry of the port, which masks any further DMA requests from the TLCs. Again, the flip-flop state determines if a carry byte cycle has been completed. It is always cleared (Q=0) before and during the carry byte cycle and set (Q=1) at the end of the carry byte cycle. The output of the flip-flop is also used to generate signals ALLDONEA or ALLDONEB which is used in the GPIB synchronization circuitry (discussed later in this chapter).

If the carry cycle feature is not used in a DMA transfer, the CC bit in CFG1A or CFG1B is zero, and DMA Gating and Control circuitry directs all DMA requests from the TLCs to DMAC Channel 0 or 2. DMAC Channel 1 or 3 is not used and must not be started. Either one or both Ports can use the carry cycle feature.

Interrupter Circuitry

The Interrupter circuitry provides the ability for the GPIB-1014D to request service. The two ports share the same circuitry.

The interrupt priority level of the GPIB-1014D is software selectable via three overlapping bits in CFG1A or CFG1B. When the DMAC drives its IREQ* pin low, the VMEbus interrupt request line corresponding to the interrupt priority code in CFG1A or CFG1B is driven low.

When the GPIB-1014D detects an interrupt acknowledge cycle (AS*, LDS*, and IACKIN* all low), the 74F85 compares VMEbus address lines A3-A1 against the 3-bit interrupt priority code in CFG1A and CFG1B. If IACKIN* is asserted and the indicated priority does not match the GPIB-1014D priority, the daisy chain signal IACKOUT* is asserted. This signal remains asserted until AS* from the interrupt handler is released. However, if the priority indicated matches the GPIB-1014D priority, the IACK* pin of the DMAC is driven low (local signal DIACK* is active). The DMAC then finishes the acknowledge cycle by placing the contents of Channel 1 or 3 NIVR (EIVR if a DMA error was encountered) on VMEbus data lines D07-D00 and driving the DDTACK* line. A flip-flop keeps the IACK* line of the DMAC continuously asserted until the interrupt handler releases both data strobes. As mentioned in Chapter 5, *Programming Considerations*, the DMAC will continuously assert the VMEbus interrupt line until the PCT bit in Channel 1 or 3 CSR is cleared.

DTB Requester and Controller Circuitry

The DTB Requester and Controller circuitry is designed to translate the bus request (BR*) from the DMAC into the appropriate VMEbus arbitration protocol. This circuitry is responsible for informing the DMAC that it has been granted the bus and for implementing the programmable Release On Request feature. This circuitry consists of the 68172 BUSCON and various gates to drive and receive VMEbus signals BG3IN*-BG0IN*, BG3OUT*-BG0OUT*, and BR3*-BR0*. The Signetics SCB68172 VME Bus Controller handles the bus arbitration process for the GPIB-1014D.

The GPIB-1014D bus arbitration process begins when the TLCs request DMA service by asserting one of DREQ3*-DREQ0* lines of the DMAC. The DMAC asserts local signal LBR*. The BUSCON accepts bus requests (LBR*) from the DMAC and asserts BR* if it does not have control of the VMEbus. The BR* of the BUSCON is transferred onto VMEbus request lines (BR0*-BR3*) by decoding the two overlapping bits in CFG1A and CFG1B. When the BUSCON receives BGIN* low while holding BR* low, it drives VMEbus signal BBSY* low and thereafter releases BR*. The BUSCON then grants the bus to the DMAC by driving its output LBG* low. In addition, the BUSCON drives the state of the VMEbus signal AS* to the onboard signal MAS*. Even with its BG* input low, the DMAC will wait until MAS* is released before asserting OWN* to begin its DMA cycle. At this point, the BUSCON reverses its AS* transceiver, waits for its input MAS* to be asserted by the DMAC, then drives the VMEbus signal AS* active. If the BUSCON receives BGIN* low when it does not drive BR* low, it will drive its output BGOUT* low and continue to do so until its input BGIN* is high (to pass the bus grant down the daisy-chain).

The BUSCON is used to implement both the Release When Done (RWD) and the Release On Request (ROR) features. The ROR* bit in CFG1A and CFG1B is used to enable the ROR feature on the GPIB-1014D. At the end of every DMA cycle, the BUSCON waits for its RELSE input to be asserted before releasing the bus. If the RWD feature is enabled (ROR*=1), RELSE is asserted to release the VMEbus when the DMAC releases its BGACK* output at the end of every DMA transfer. If the ROR feature is enabled (ROR*=0), RELSE is kept unasserted and GPIB-1014D remains the bus master until there is a request on one of the four VMEbus request lines BR3* - BR0* (local signal ALLBR becomes active). During the period in which the GPIB-1014D is holding the bus, if LBR* is asserted (due to DMA requests from the TLCs), the BUSCON will assert LBG* to immediately grant the bus to the DMAC but will not assert BR* (as described in the previous paragraph).

Note: The ROR feature can be used along with the cycle steal with hold feature to maximize the use of the VMEbus before relinquishing the bus to other VMEbus masters. The ROR and cycle steal with hold features are available to avoid bus arbitration time.

GPIB Synchronization and Interrupt Control Circuitry

The GPIB circuitry is designed to allow the two ports of the GPIB-1014D to detect GPIB synchronization. This synchronization event can cause an interrupt to the DMAC. There are two identical GPIB Synchronization Detector and Interrupt Control circuitries, one for each port. The GPIB synchronization circuitry allows each port of the GPIB-1014D to detect when the last byte of data in a data transfer has been accepted by all devices on the GPIB. In this manner, the host CPU can be notified (by an interrupt) that the DMA transfer is complete and the GPIB is synchronized.

TLC Interrupt

For each Port, the INT pin of the TLCs is connected to one of three inputs of a NOR gate. Each of the TLC interrupt request events or conditions are controlled by mask bits located within the TLCs. Each of the 13 TLC events is individually enabled by bits in the mask registers IMR1 and IMR2. When a TLC detects an interrupt condition, it drives its INT pin high. This action causes a negative transition on the NOR gate output and also on PCL1* or PCL3* (depending on which port is active). The high-to-low transition also sets the PCT status bit and generates an interrupt if interrupts are enabled in Channel 1 or 3. The exact TLC interrupt condition can be detected by examining the interrupt mask in IMR1 and IMR2 of the TLC. Reading from the TLC status register will release the TLC INT line and enable further interrupts. Notice that the TLC must be properly configured in order to present an active high interrupt request output. This is accomplished by clearing the INV bit of AUXRB.

GPIB Synchronization Interrupt

This circuitry detects GPIB synchronization after the last byte in a DMA transfer has been transferred to or from the TLCs. This is accomplished by monitoring the DAV* line on the GPIB. The circuitry begins to monitor synchronization when either signal ALLDONEA or ALLDONEB becomes high, which occurs when Port A or Port B is transferring the last data byte in its DMA transfer sequence.

If Port A does not use the carry cycle feature (as indicated by the CC bit in CFG1A), the transfer of the last byte is detected by DONE* and ACK0*, both active at the same time. Similarly, if Port B does not use the carry cycle feature, the transfer of the last byte is detected by DONE* and ACK2*. DMAC asserts DONE* when it is transferring the last data byte in the last data block in any data chain. Depending on whether channel 0 or 2 is active, ALLDONEA or ALLDONEB is driven high during this time. If a carry cycle feature was enabled and the carry cycle byte was sent (as indicated by the flip-flop in the DMA Gating circuitry being set), the next DMA cycle to transfer the last data byte (indicated by an active ACK1* for Port A or ACK3* for Port B) will make signal ALLDONEA or ALLDONEB high. After this last data transfer, the DMAREQ from the TLC of the active port is masked (see *DMA Gating and Control* earlier in this chapter) and Channel 1 or 3 is still active (see Chapter 5, *Programming Considerations*); thus, Channel 1 or 3 cannot reach its terminal count, which is normal. In both cases, the positive

transition (0 to 1) in ALLDONEA or ALLDONEB causes the GPIB synchronization circuitry in each Port to begin monitoring its GPIB DAV* line.

The GPIB synchronization is detected differently depending on the direction of the DMA transfer. The direction of the transfer in each Port is specified in the DIR bit in CFG1A or CFG1B. If the transfer is from the VMEbus memory to the GPIB (DIR = 0), the ALLDONE (A or B) signal indicates that the last byte has just been transferred to the TLC (A or B). The DAV* line at this time is high. The TLC (as a GPIB Talker) will drive DAV* low as it places the data on the GPIB. As soon as all listeners have accepted the byte, the TLC will release DAV* to high again. It is this positive transition of the DAV* line that indicates synchronization. This generates a negative transition on PCL1* for Port A or PCL3* for Port B.

If the direction of the DMA transfer is from the GPIB to the VMEbus system memory (DIR =1), the GPIB synchronization is detected by watching the level of the DAV* line rather than looking for a transition. The ALLDONE (A or B) signal indicates that the last byte in the DMA transfer has just been transferred from the TLC (A or B). This means that the TLC has already accepted the byte from the GPIB. By the time the DMA transfer is complete, all devices on the GPIB may have already accepted the byte and the Talker may have already released DAV*. For this reason, the synchronization circuitry looks at the level of the DAV* line rather than for a transition. When the DAV* line is detected high, all devices have accepted the byte and a negative transition is generated on PCL1* or PCL3*, requesting an interrupt.

Bus Error Interrupt

The third event which can cause a transition on the PCL1* or PCL3* can occur when the VMEbus BERR* line is driven low while the DMAC is bus master, to indicate that the cycle cannot be completed. This condition will also cause the DMAC BEC1* line to go low, which will cause the DMAC to terminate the cycle, and set the COC and ERR bits in the active channel's CSR (can be any of the four DMAC channels). Whether or not a carry cycle was used and depending on which port is active, a bus error will also cause a transition on the PCL1* for Port A or PCL3* for Port B, requesting an interrupt on Channel 1 or 3. The host CPU can detect the error condition by examining the CSR of all four DMAC channels.

DMA Controller

The DMAC is a flexible high-performance device. It provides four independent DMA channels (0-3), two of which (channel 0 and 1) are used by Port A to provide GPIB to memory DMA flyby transfers and two of which (channel 2 and 3) are used by Port B. All four channels are available for general use, such as memory to memory DMA transfers. The DMAC provides 24-bit addressing, a 16-bit data path, and programmable selection of the Address Modifier (AM) lines. 32-bit addressing is available from the onboard Page Register. During DMA operations, the GPIB-1014D functions as a full VMEbus master. The DMAC and BUSCON control the direction of the GPIB-1014D data and address bus transceivers as needed to carry out the transfer. Details of the DMAC operation, with emphasis on GPIB-1014D applications, are given in the following paragraphs.

DMAC Channel Operation

A DMAC channel operation proceeds in three principal phases. During the initialization phase, the CPU configures the channel control registers, sets initial addresses, and starts the channel.

During the transfer phase, the DMAC accepts requests for data operand transfers, and provides addressing and bus controls for the transfers. The termination phase occurs after the operation is complete, when the DMAC reports the status of the operation. The following paragraphs discuss these three phases of the channel operation.

Initialization and Transfer Phases

The following paragraphs describe the communication between the TLC (the device) and the DMAC as well as how the DMAC controls the data transfer during DMA operations.

Device (TLC)/DMAC Communication. Communication between the Devices (TLCs) and the DMAC is accomplished mainly by two signals (are for each port). Each of the four DMA channels has a DMA request input (REQ0*-REQ*3) and a DMA acknowledge output (ACK0*-ACK3*). The TLC requests service by:

1. Asserting its DMAREQ line (this is called external request)

2. Waiting for its DMAACK* pin to assert

3. Waiting for its RD* or WR* pins to assert before sending or receiving data

The DMA Gating and Control circuitry for Port A will gate the DMAREQ of the TLCA to REQ0* or REQ1*, depending on whether or not the carry cycle function is used. The DMA Gating and Control circuitry for Port B will gate the DMAREQ of the TLCB to REQ2* or REQ3*, depending on whether or not the carry cycle function is used. Depending on which port and channel is being serviced, any one ACK* line is activated during DMA transfers to or from the TLCs. The ACK* lines are used to tell the TLCs if their current DMA request is being serviced (the DMAACKA* of TLCA or the DMAACKB* of TLCB is asserted). Flowthrough memory-to-memory transfers must use the automatic request mode.

Each channel also has a peripheral control line (PCL0*-PCL3*). The function of this line is programmable via the Device Control Register (DCR). When programmed as a Status input with interrupt, in GPIB-1014D applications, the status level of the PCL can be determined by reading the PCS bit in the CSR (TTL high or low). If a negative transition occurs and remains stable for two DMAC clock cycles, the PCT bit of the CSR is set. The setting of the PCT bit will cause an interrupt, if an interrupt was previously enabled, by setting the EINT bit in CCR.

DMA Requests. Internal or external requests activate the DMAC to transfer an operand. The REQG bits of the OCR determine the manner in which requests are generated. Requests may be externally generated by the device or internally generated by the DMAC using its internal automatic request mechanism. Internal automatic requests can be generated at a maximum rate, so that the channel always has a request pending, or at a limited rate, monitoring the bus bandwidth use. Any of the two modes are used when performing memory-to-memory DMA transfers. In GPIB-1014D GPIB applications, the REQG bit must be set to 10 to indicate that the external REQ line will initiate a transfer. After selecting the External Request Generation mode, you must also set the XRM bit in DCR to specify whether a channel must operate in cycle steal or cycle steal with hold mode. You must not set XRM to 00 to select the burst transfer mode in the GPIB DMA application. If internal request mode was chosen, the XRM bit can be ignored.

For GPIB DMA data transfers, the GPIB-1014D uses cycle steal mode (with or without hold) . In the cycle steal mode, the device (TLC) requests an operand transfer by generating a falling

edge on the REQ* line (REQ0* or REQ1*). The DMAC services the request by arbitrating for the system bus and then driving one of the ACK* lines active to transfer the operand. If the XRM bits specify cycle steal without hold, the DMAC will relinquish the bus after each operand transfer. If the XRM bits specify cycle steal with hold, the DMAC will retain bus ownership for a short time after an operand transfer in anticipation of a new request from the TLC within that time period. If a new request is not present during the hold period, the DMAC relinquishes the bus by unasserting its signal OWN*. The sample period is defined by the values programmed into the GCR.

Data Transfers. All DMAC transfers are assumed to be between a 16-bit 68000 device (VMEbus memory) and another device. By programming the DCR, the characteristics of the device may be assigned. Each channel can communicate using the following protocols. For GPIB-1014D GPIB DMA data transfers, the device type must be set to 10 for *Device with ACK - single addressing* for all four channels.

Single Addressing Mode is used for implicitly addressed devices that do not require addressing of the data register before the data can be transferred. Such peripherals use the acknowledge (ACK) line to access the data register and require only one bus cycle to transfer the data between themselves and memory. In case of a DMA write to a device, data is transferred directly from the VMEbus memory into the device. Similarly, during a DMA read of a device, data from a device is transferred directly into the VMEbus memory.

Operands and Addressing. Three factors affect how the actual data is handled: device (destination) port size, operand size (from source), and address sequencing.

- Device Port Size — The DCR is also used to program the device port size to be 8 or 16 bits. The port size is the number of bits of data which the device can handle (transmit or receive) in a single bus cycle. For GPIB DMA data transfers, the device (or TLC) port size is eight bits. For memory-to-memory transfers, the port size can be 8 or 16 bits.

- Operand Size — The OCR is used to program the operand size to be either byte, word, or longword. The operand size is the number of bits of data to be transferred to honor a single DMA request. Multiple bus cycles may be required to transfer the entire operand through the device port if the operand is larger than the device port size. For single addressing operations, like GPIB DMA operations, the device port size and the operand size must be equal. The transfer counter counts the number of operands transferred. For GPIB DMA data transfers, the operand size is eight bits and the transfer counter indicates the number of bytes to be transferred. For memory-to-memory transfers, the operand size can be byte, word or longword.

- Address Sequencing — The sequence of addresses generated to address the memory and device depends on the port size, operand size, whether the addresses are to count up, down, or not change, and whether the transfer is explicitly or implicitly addressed. The Sequence Control Register (SCR) is used to program the memory address count method and the device address count method.

For single-address transfers, such as GPIB DMA transfers, the device port size and operand size must be equal. If the operand size is a byte, the address increment/decrement is one (1). If the operand size is a word, the address increment is two (2). The Memory Address Register and the Transfer Counter are updated after the operand is transferred.

For dual-address transfers (memory-to-memory), the device port size does not need not to equal the operand size. If the operand size is larger than the memory width or the device port size, the DMAC will run multiple bus cycles to transfer operand parts until the entire operand has been transferred. Regardless of the way the address register is counting (up or down), the DMAC will access the parts of the operand in a linear sequence. The step between the addresses of the parts depends on the memory width or the device port size. The size of the parts is the size of the device port or memory width. The number of parts is the operand size divided by the port size or memory width. The memory address register (MAR) increment is added or subtracted after the operand is transferred.

Address Register Operation. The DMAC has three 32-bit address registers per channel: the Memory Address Register (MAR), the Device Address Register (DAR), and the Base Address Register (BAR). The MAR is used in all DMA operations. This register is either initialized before the channel operation is started or is loaded during chaining or continue operations, which are discussed later in this chapter. The DAR is used to address the device (VMEbus memory) in dual-address transfers. It is initiated before starting the channel operation. The BAR register is used only in chaining or continue operations.

Transfer Count Register Operation. The DMAC has two 16-bit transfer counter registers per channel: the Memory Transfer Counter (MTC) and the Base Transfer Counter (BTC). The MTC is used in all operations to count the number of operands transferred in a block. It is decrement by one after each operand is transferred. This register is either initialized before the channel operation is started or is loaded during chaining or continue operations. The BTC is used for chaining and continue operations. Both the MTC and the BTC have a terminal count of zero. If either register is initialized or loaded with a terminal count when the channel is configured to use that same register, a count error is signaled.

Initiation and Control of Channel Operation

The Channel Control Register (CCR) provides mechanisms for starting, continuing, halting, or aborting an operation. It also controls the enabling of interrupts from a channel.

Initiating the Operation. To initiate the operation of a channel, the STR bit of the CCR is set to start the operation. Setting the STR bit causes the immediate activation of the channel. The channel will be ready to accept requests immediately. In the GPIB applications, the DMAC is ready to accept DMA requests from the TLC. The channel initiates the operation by first clearing the STR bit and then setting the channel active (ACT) bit in the CSR (internal to the DMAC). Any pending requests are cleared and the channel is then ready to receive requests for the new operation. If the channel is configured for an illegal operation, the configuration error is signaled and no channel operation is run. The illegal operations include the selection of any of the operations marked *undefined, reserved*. If the DMA operation is dual-address (for memory-to-memory DMA), the device address register must have been previously initialized. The channel cannot be started if any of the ACT, COC, BTC, NDT, or ERR bits is set in the CSR. In this case, the channel signals an operation timing error.

If the operation is unchained, as to transfer a single block of data, the MAR and the MTC should have been previously initialized. If the operation is chained, as to transfer multiple blocks of data, the BAR and/or the BTC must have been previously initialized. For array chained operation, both the BAR and the BTC must have been previously initialized. For a link chained operation, only the BAR must be initialized.

The Continue Mode of Operation. The continue bit (CNT) allows multiple blocks to be transferred in unchained operations. The CNT bit is set in order to continue the current channel operation. If an attempt is made to continue a chained operation, a configuration error is signaled. The BAR and BTC must have been previously initialized. The continue bit can be set at the same time as the STR bit (to start a channel) or it can be set while the channel is still active. If the CNT bit is set at any other time, an operation timing error will occur. GPIB-1014D applications generally do not use the continue mode of operation.

Halt. The CCR has a halt bit which allows suspension of the operation of the channel. If this bit is set, a request may still be generated and recognized, but the DMAC does not attempt to acquire the bus to service the request for the halted channel. When this bit is reset, the channel resumes operation and services any request that may have been received while the channel was halted. The HLT bit must be cleared to zero when writing to the STR bit to avoid immediate halt of the channel.

Software Abort. The CCR has an SAB which allows the current operation of the channel to be aborted. Writing a 1 into the SAB causes a channel abort error to be signaled. The active channel is then terminated immediately, the ACT bit is cleared, and both COC and ERR bits are set. The abort status can be read in the Channel Error Register (CER). When the CCR is read, the SAB always reads as a 0.

Interrupt Enable. The CCR has an Interrupt Enable (EINT) bit that allows the channel to request interrupts on the completion of block transfers (bit BTC is set), on termination of channel operations (bit COC is set), or on a negative transition on the PCL (bit PCT is set). While Port A of the GPIB-1014D uses Channel 0 and 1, the interrupt is usually enabled in Channel 1 only, similarly, Port B has interrupt enabled in Channel 3 (see Chapter 5, *Programming Considerations*).

Block Termination

At the end of the transfer of an operand, the DMAC decrements the MTC. If this counter is decreased to the terminal count, the transfer counter is exhausted and the operand is the last operand of the block. The channel operation is complete if the operation is unchained and there is no continuation or if the operation is chained and the chain is exhausted (last block in the chain has been transferred). When the last transfer of the last data block is complete, the ACT bit of the CSR is cleared and the COC bit is set, indicating the channel operation is complete.

A bus exception during a bus cycle being run for a channel or an error in the channel terminates the block transfer and the channel operation. The bit of the CER corresponding to the error is set. The ACT of the CSR is cleared and the COC and ERR bits are set.

Continued Operations. When the memory transfer counter is exhausted and the continue bit of the CCR is set, the DMAC performs a continuation of the channel operation.

1. The Base Address Register, the Base Function Code Register, and the Base Transfer Count Registers are automatically copied into the Memory Address Registers, the Function Code Registers, and Memory Transfer Count Registers.

2. The Block Transfer Complete (BTC) bit of the CSR is set.

3. The Continue (CNT) bit is cleared.

4. The channel begins a new block transfer.

If the Interrupt bit in the CCR is set when the BTC bit is set, an interrupt will be generated. The interrupt handler can reload the BFCR, BAR, and BTCR with information describing the next data block if necessary, clear the BTC bit, and set the CNT to repeat the operation. In all cases, if the MTC is loaded with a terminal count, the count error is signaled. The GPIB-1014D usually does not use continue operations for its GPIB DMA transfers.

Multiple Block Operations. When the memory transfer counter is exhausted, there are additional blocks to be transferred if the channel is chained and the chain is not exhausted. The DMAC provides the re-initialization of the MAR and the MTC as described in the following paragraphs.

Array Chaining Operations. Chaining is not necessary when transferring a single small block of data. When data to be transferred is fragmented or is larger than 64K, array chaining is used to transfer these blocks of data. This type of chaining uses an array in memory which holds the addresses and transfer counts of the data blocks. The address and transfer count array must occupy continuous memory locations. Each entry in the array is six bytes long – four bytes to hold the starting address of a data block and two bytes to hold the length of the data blocks. The beginning address of this array is in the BAR, and the number of entries in the array is in the BTC; that is, the BAR points to the address and transfer count array. Before starting any block transfers, the DMAC fetches (using DMA) the entry (a total of six bytes in three DMA cycles) currently pointed to by the BAR. The address information is placed in the MAR and the count information is placed in the MTC. After each chaining entry is fetched, the BTC is decrement by one and the BAR is incremented by six to point to the next entry of the address and transfer count array. When the base transfer counter reaches a terminal count, the chain is exhausted, and the entry just fetched determines the last block of the channel operation. After this data block is transferred, the channel operation is complete and the COC bit is set.

As described in Chapter 5, *Programming Considerations*, the array chaining mode is used in the GPIB-1014D applications to implement the carry cycle feature. It is also used to transfer multiple blocks of data on Channel 0 for Port A and Channel 3 for Port B.

The address and transfer count array must start at an even address or the entry fetch results in an address error. The transfer count (the last two bytes of each array entry) must be non-zero. The starting address of the data blocks (the first four bytes of each array entry) can be even or odd. Since the base registers can be read by the CPU, appropriate error recovery information is available should the DMAC encounter an error anywhere in the chain.

An example of the array format for array chaining is shown in Figure 6-1.

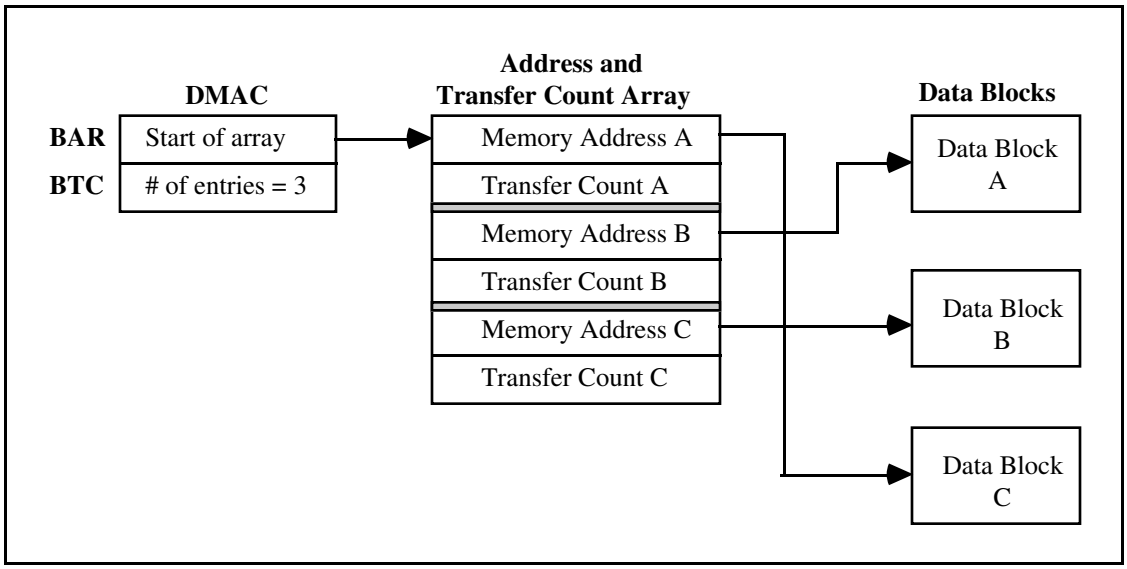


Figure 6-1. Array Format for Array Chaining Modes

Linked Chaining Operations. This type of chaining is very similar to array chaining. The difference is that it uses a list in memory consisting of memory addresses, transfer counts, and linked addresses. Whereas array chaining requires a continuous address and transfer count array, the linked address in each entry allows a non-continuous address and transfer count array (by a mean of linking). Each entry in the chain list is ten bytes long - four bytes to hold the address of the data block, two bytes to hold the transfer count, and the last four bytes to hold the linked address of the next array entry. The linked address of the last array entry must be set to zero. Similar to array chaining, the BAR points to the first array entry. However, the BTC is unused. Before starting any block transfers, the DMAC fetches (using DMA) the entry currently pointed to by the BAR. The address information is placed in the MAR, the count information is placed in the MTC, and the link address replaces the current contents of the BAR. The channel then begins a new block transfer. As each chaining entry is fetched, the update BAR is examined for the terminal link, which has all 32 bits equal to zero. When the new base address is the terminal address, the chain is exhausted and the entry just fetched determines the last block of the channel operation.

Linked chaining allows entries to be easily removed or inserted without having to reorganize data within the chain. Since the end of the chain is indicated by a terminal link, the number of entries in the chain need not be specified. The last four bytes in each entry (the linked address) must be even or the entry fetch results in an address error. The middle two bytes (the transfer count of the data block) must be non-zero. The first four bytes in each entry (the starting address of the data block) can be even or odd. Altogether, the address of the address and transfer count array must be even. If a terminal count is loaded into the MTC, the count error is signaled.

The linked chaining method can be used in GPIB-1014D applications to transfer large data blocks arbitrarily much like array chaining. Similarly, you can use link chaining to implement the carry cycle feature, although array chaining is simpler.

An example of the array format for link chaining is shown in Figure 6-2. Notice that the data blocks do not have to be in the same order as they were in the memory address.

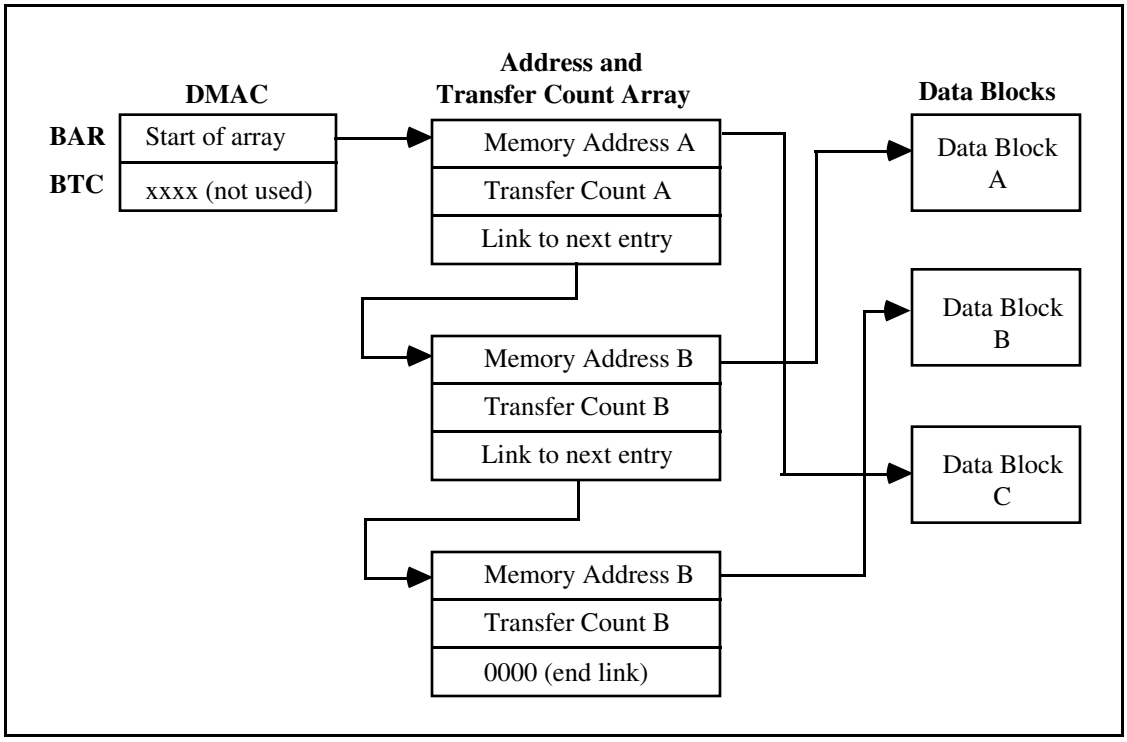


Figure 6-2. Array Format for Linked Chaining Modes

Error Conditions

When an error is signaled on a channel, all activity on that channel is stopped:

- The ACT bit of the CSR is cleared
- The COC bit is set
- The ERR bit of the CSR is set
- The error code is recorded in the CER

All pending operations are cleared, so that both the STR and CNT bits of the CCR are cleared. The following paragraphs explain the various error conditions.

Configuration Error. The Configuration Error occurs when any undefined or reserved bit pattern or illegal device/operand size combination is programmed into a channel and an attempt is made to set the STR bit. This error will occur, for example, when chaining is programmed and the continue bit is also set; if the DTYP bit specifies a single-address transfer and the device port size is not the same as the operand size, or if DTYP is 68000 or 6800 (DPS is 16 bits), SIZE is eight bits, and REQG is 10 bits or 11 bits (request pin). Setting an undefined configuration will

signal a configuration error. The undefined configurations are: XPM=01, MAC=11, DAC=11, CHN=01, SIZE=11. In any case, the channel will not be started.

Operation Timing Error. If a write access is attempted to certain registers or bits within registers while a channel is active or if certain status bits are set, an operation timing error will occur and the channel operation will abort if the channel is active. This error will occur, for example, if an attempt is made to continue an operation without STR being simultaneously set, if the channel is not active, if an attempt to set STR is made with ACT, COC, BTC, NPT, or ERR asserted, if an attempt is made to write to the DCR, OCR, SCR, GCR, MAR, DAR or MTC with STR or ACT asserted, if an attempt to assert CNT is made when CHN is 10 or 11, or if an attempt to assert CNT is made when BTC and ACT are asserted.

Address Error. This is signaled if a word or longword operation is attempted to an odd address.

Bus Error. This indicates a bus error occurred during the last bus cycle generated by any of the active channel.

Count Error. This is signaled if the memory or Base Transfer Count registers are initialized with terminal count. A count error is signaled if a terminal count is encountered during continue or chain processing.

Abort. This is signaled if the PCL was configured as an abort input and made an active transition, or if the channel operation was aborted by the SAB bit of the OCR. If the PCL is used as an abort input, the PCT bit must be cleared prior to starting the channel.

Error Recovery

The DMAC allows access to internal registers to implement error recovery procedures. If an error occurs during a DMA transfer, appropriate information is available to the operating system to allow a soft-failure operation. The operating system must be able to determine how much data was transferred, where the data was transferred to, and what type of error occurred.

The information available to the operating system consists of the present values of the MAR, DAR, BAR, MTC, BTC, Control Registers, Status Registers, and Error Registers. After the successful completion of any transfer, the MAR and DAR point to the location of the next operand and the MTC contains the number of operands to be transferred. If an error occurs during a transfer, that transfer will not complete and the registers will contain the same values they had before the transfer was attempted.

The DMAC will log the first error encountered in the channel error register. If an error is pending in the error register and another error is encountered, the second error will not be logged.

GPIB Interfaces (NEC μPD7210s)

The GPIB-1014D is interfaced to two GPIB busses, each using NEC μPD7210 Talker/Listener/ Controller (TLC) large scale integrated circuits. Access to each TLC is through a block of 16 VMEbus addresses.

When the GPIB-1014D is operating as a VMEbus slave, one of the two onboard TLCs is enabled when the base address of the GPIB-1014D has been decoded, VMEbus address bit A8 is a logic 1, and bit A4 is a logic 1. Address bit A9 is used to select which TLC is accessed. The TLC register select signals (RS2 through RS0) are derived from VMEbus address lines A3 through A1. Data is strobed into the TLC registers using WR*. Data is read from the TLC using RD*. Both RD* and WR* are generated by the Timing State Machine and are derived from the VMEbus WRITE* line.

During DMA transfers, the GPIB-1014D is acting as the VMEbus master, and the TLCCS* (A or B) and RS2 through RS0 signals are ignored by the TLC (except when the carry cycle byte is written to the Auxiliary Register by the DMAC). Instead, the chip is enabled and selected by the DMAACK* (DMA Acknowledge A or B) signal and the internal registers of the TLCs, CDOR and DIR, are automatically accessed. Data is strobed into the CDOR at the rising edge of the TLC WR* signal. If RD* is asserted, data from the DIR is placed on the internal three-state data bus a minimum access time after DMAACK* is asserted low.

Most of the TLC GPIB interface functions can be implemented or activated from either side; that is, the TLC may be programmed to do these functions by the VMEbus master or it may be addressed to do them by the GPIB Controller. In terms of the IEEE-488 standard, the distinction between these two modes of operation is generally the same as that between local and remote interface messages, respectively, as defined in the IEEE-488 standard.

The ADSR is the primary register for monitoring the current status of the TLC, that is, in determining if it is a GPIB Talker, GPIB Listener, GPIB Active Controller, or in GPIB remote or local mode. The CPTR provides a means to read the GPIB data bus directly and is used to recognize interface messages that are not automatically decoded and implemented by the TLC.

The ADR is used to program the primary and secondary GPIB addresses of the TLC and is also used to disable talking or listening and to enable dual primary addressing. The SPMR is used to program the serial poll status byte.

The IMR1 and IMR2 are interrupt mask registers for enabling and disabling the interrupt from the TLC on the occurrence of 13 GPIB conditions or events. The status of these conditions can be read from the ISR1 and ISR2 registers. The status bits in these registers function independently of the corresponding mask bits; that is, they are set and cleared regardless of whether an interrupt request is enabled for the condition. An important fact to remember is that ISR1 and ISR2 are always cleared when read, even if the condition which caused the bit to be initially set remains true.

Data to and from the GPIB is pipelined through the CDOR and DIR respectively. An 8 MHz clock is used as the CLOCK input to the TLC. For proper GPIB timing, the internal counter must be programmed to eight. The TLC RESET pin is driven by the GPIB-1014D RESET signal.

Connecting the TLC to the GPIB itself are two multi-function transceivers, one handling the data lines (a 75160A) and the other handling the handshake and management lines (a 75162A). The 75160A and 75162A transceivers are specifically designed to meet the IEEE-488 driver and

receiver specifications. In particular, they do not affect bus (GPIB) operations when the GPIB-1014D power is removed or during power up or down transitions.

The TLC controls the direction of the 16 GPIB signals, depending on whether the chip is functioning as a GPIB Talker, Listener, or Controller. The SC bit of CFG2A or CFG2B, when set to a logic one, programs one of two onboard 75162 transceivers to drive or receive the GPIB IFC* and REN* control lines. The data lines are always driven in the three-state mode unless the TLC receives a parallel poll message (ATN and EOI both true), at which time the drivers are switched to open collector mode.

Test and Troubleshooting

The GPIB-1014D is designed to aid acceptance testing and troubleshooting of either hardware failures or software bugs. The hardware provides several features that enable stand-alone testing.

DMA Stand Alone Testing

Most of the GPIB-1014D ICs and interconnecting circuitry are associated with implementing the DMA function. From the system's point of view, the best acceptance test for the interface is one in which DMA data is transferred between the VMEbus memory and a GPIB device. Should there be problems associated with implementing this type of test, a good troubleshooting tactic is to start removing variables in the system, starting with the DMAC itself. The GPIB-1014D is designed to perform DMA transfers without needing an external GPIB device. This can be implemented using the flowthrough mode of the DMAC to perform memory-to-memory DMA transfers. This feature allows test and verification of the DMAC, the DTB Requester, the Interrupter, and the VMEbus interface. Any of the DMAC channels can be configured to transfer a buffer of data from one location in memory to another, and the new buffer may be compared against expected results. The DMAC must be configured to provide flowthrough transfers using the automatic request mode.

GPIB Interface Testing

The NDAC* and DIO1* bits can be used to determine if the output signals of the TLCs, the 75160A, and the 75162A are functioning properly. Since most failures (including problems with shorts or open on the PWB) prevent the TLCs from working at all, this test gives limited assurance that the TLCs and their associated circuitry are working and that the output signals can be manipulated properly.

NDAC* is the GPIB Not Data Accepted signal. By programming the TLCs to Listen or not to Listen via the AUXMR, NDAC* can be asserted or cannot be asserted, respectively.

DIO1* is the GPIB Data Input/Output bit 1 (LSB). By programming the TLCs as active GPIB Controller and sending command bytes using the CO bit, the CDOR, DIO1* can be asserted and unasserted for testing.

Chapter 7
Diagnostic and Troubleshooting Test Procedures

This chapter contains test procedures for determining if the GPIB-1014D is installed and operating correctly. The tests are similar to those used by National Instruments to verify correct hardware functioning. The procedure used for testing is as follows:

1. Program specific internal functions by writing to one or more registers.

2. Read other registers to confirm that the functions were implemented. You must have an appropriate mechanism available for writing to and reading from I/O and memory locations. A program such as an interactive control program, console emulator, monitor, or program debugger is ideal for this purpose.

Interpreting Test Procedures

The following test procedures are written in the form of simple equations. The left side of the equation contains the hexadecimal address offset (from the GPIB-1014D base address) and mnemonic for the register. The right side of the equation contains a hexadecimal value. Converting the hex value to binary results in a representation of the bit pattern in the register. For example, a hex value of FF corresponds to a bit pattern of 11111111 and 40 hex corresponds to a bit pattern of 01000000.

Equations not followed by a question mark are instructions that you must follow to load the value shown into the designated register. Equations followed by a question mark are instructions that you must follow to read the register and verify that the value in the register is the one indicated.

The column to the left of each test step contains the relative register address. Comments written to the right of each test step briefly describe the action taken, and sometimes suggest the purpose.

The test procedures are designed to check the most elementary levels of functioning first, and then progress to tests of higher complexity. The onboard circuitry is reset at the beginning of every test. For this reason, perform the tests in the order given.

Since there are two identical ports on board, each test should be repeated twice to exercise the circuitry on each port. Note that equivalent registers in Port B are easily accessed by adding 200 hex to the register addresses of Port A. As described under *Porting GPIB-1014 Software to GPIB-1014D* in Chapter 5, when 200 hex is added to DMAC Channel 0 and 1 register addresses, the equivalent DMAC registers in Channel 2 and 3 are automatically accessed. In addition, a Local Master Reset is performed before the repeat of each test for Port B.

If the GPIB-1014D does not perform as described in the test procedures, carefully perform the following steps:

1. Verify that the test instructions have been followed correctly.

2. Examine any read and write routines being used in connection with the checkout procedure for errors.

3. Recheck the switch settings described in Chapter 3.

After these items have been carefully checked, if the interface is still not functioning properly, gather together the information concerning what the GPIB-1014D is and is not doing with regard to the expected results and contact National Instruments.

GPIB-1014D Hardware Installation Tests

Complete the following steps to test the installation of your GPIB-1014D interface board.

1. Initialize the TLCs (Port A).

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
11B	AUXMR = 2	TLC Reset
11B	AUXMR = 0	Immediate execute pon

2. Send LMR. Then read registers and compare to reset values.

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
111	DIR = 0?	
113	ISR1 = 0?	
115	ISR2 = 0?	
117	SPSR = 0?	
119	ADSR = 40?	
11B	CPTR = 0?	
0FF	GCR = 0?	
007	CCR0 = 0?	
047	CCR1 = 0?	
000	CSR0 = 01?	
040	CSR1 = 01?	
065	NIV1 = 0F?	
067	EIV1 = 0F?	

3. Test the DMAC Channel Address Registers.

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
00C	MAR0 = 5533	
00C	MAR0 = 5533?	
04C	MAR1 = 5533	
04C	MAR1 = 5533?	
01C	BAR1 = 5533	
01C	BAR1 = 5533?	

4. Test the Local Master Reset.

105	CFG2A = 08	Clear LMR
065	NIV1 = 55	
065	NIV1 = 55?	
067	EIV1 = 55	
067	EIV1 = 55?	
105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
065	NIV1 = 0F?	
067	EIV1 = 0F?	

5. Test ton, DO, ERR, CPTR, and TA.

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
11B	AUXMR = 2	TLC Reset
119	ADMR = 80	ton
11B	AUXMR = 0	Immediate execute pon
119	ADSR = 42?	TA
113	ISR1 = 2?	DO
111	CDOR = 51	write data byte
11B	CPTR = 51?	verify
113	ISR1 = 6?	DO + ERR
113	ISR1 = 0?	bits cleared when read
11B	AUXMR = 2	TLC Reset
119	ADMR = 0	disable ton
11B	AUXMR = 0	Immediate execute pon
119	ADSR = 40?	not TA

6. Check lon and LA.

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
11B	AUXMR = 2	TLC Reset
113	IMR1 = 0	no interrupts
115	IMR2 = 0	no interrupts
119	ADMR = 40	lon
11B	AUXMR = 0	Immediate execute pon
119	ADSR = 44?	LA
11B	AUXMR = 2	TLC Reset
119	ADSR = 40?	not LA

7. Test ATN, CIC, and CO.

105	CFG2A = 0B	Set LMR and turn LED green
105	CFG2A = 09	Clear LMR and become System Controller
11B	AUXMR = 2	TLC Reset
119	ADMR = 31	Address Mode 1
11B	AUXMR = 0	Immediate execute pon
11B	AUXMR = 1E	set IFC

11B	AUXMR = 16	clear IFC
119	ADSR = 80?	CIC
115	ISR2 = 9?	CO + ADSC
11B	AUXMR = 10	go to standby
119	ADSR = C0?	CIC + ATN*

8. Test the DMA Error.

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
007	CCR0 = 80	start channel 0
007	CCR0 = 10	abort channel 0
000	CSR0 = 91?	COC and Error
000	CSR0 = 90	clear bits
000	CSR0 = 01?	bits cleared

9. Test the DMAC Interrupt Detection.

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
044	DCR1 = 00	PCL Status Input, channel 1
040	CSR1 = FF	Clear bits, channel 1
11B	AUXMR = 2	TLC Reset
11B	AUXMR = A0	Clear INV
119	ADMR = 80	ton
113	IMR1 = 2	DO IE
11B	AUXMR = 0	Immediate execute pon
040	CSR1 = 2?	PCL transition occurred
113	ISR1 = 2?	DO
040	CSR1 = 3?	TLC interrupt cleared
040	CSR1 = FF	
040	CSR1 = 01?	bit reset

10. Test the memory-to-memory (flowthrough) DMA transfer.

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
000	CSR0 = FF	Reset Channel 0 status register
004	DCR0 = 08	Burst mode, 16-bit transfer. w/ 68000 dev
005	OCR0=11	Mem to dev, word xtfr, no chain, internal req
006	SCR0=05	MAC and DAC both count up
00A	MTC0=0005	5 words (10 bytes) transfer
029	MFC0=06	
031	DFC0=06	
00C	MAR0=daddr	8-byte data address (Ex: 00200000)
014	DAR0=daddr+0A	8-byte data address (Ex: 0020000A)
101	CFG1A=00	no interrupt, BR0*&BG0*, no carry cycle, enable ROR feature
	daddr=FF	Set data values at source locations
	daddr+1=FE	
	.....	

	daddr+9=F6	
007	CCR0=80	Start DMA on Channel 0
000	CSR0=81?	DMA completed w/o error? Bit 4 = 1 if error
	daddr+0A=FF?	Verify data values at destination locations
	daddr+0B=FE?	
	.....	
	daddr+13(hex)=F6?	

11. Test the DMA transfer (flyby) from memory to GPIB,and one-byte transfer.

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
00A	MTC0 = 0001	one byte
004	DCR0 = A0	
005	OCR0 = 02	
006	SCR0 = 0	
000	CSR0 = FF	
040	CSR1 = FF	
045	OCR1 = 0	
029	MFC0 = 06	
00C	MAR0 = daddr	4 byte data address (any free data area)
	daddr= data	put data byte in memory
101	CFG1A = 18	BRG3*, OUT, enable ROR feature
11B	AUXMR = 2	TLC Reset
119	ADMR = C0	ton,lon
007	CCR0 = 80	Start channel 0
115	IMR2 = 20	DMA out enable
11B	AUXMR = 0	Immediate execute pon, TLC immediately sets DO to request for a byte to be transferred using DMA to its internal register.
000	CSR0 = 81?	DMA channel finished (COC)
040	CSR1 = 02?	GPIB synchronized (PCL1* is pulled low, PCT bit in CSR1 is set)
113	ISR1 = 03?	DO and DI are both set (DO is currently set to request another byte to be transferred)
111	CDOR = data?	verify data that was transferred to the TLC
101	CFG1A = 18	clear GPIB synchronization detecting circuitry, also to pull PCL1* high
040	CSR1 = 02	clear PCT bit in CSR1
040	CSR1 = 01?	PCT bit cleared, PCL1* high

12. Test the DMA transfer (flyby) from GPIB to memory, one-byte transfer, and memory write.

105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
00A	MTC0 = 0001	one byte
004	DCR0 = A0	
005	OCR0 = 82	
006	SCR0 = 0	
000	CSR0 = FF	
040	CSR1 = FF	
045	OCR1 = 0	

029	MFC0 = 06	
00C	MAR0 = daddr	4-byte data address
	daddr= 0	clear data location
101	CFG1A = 19	BRG3*, IN, enable ROR feature
11B	AUXMR = 2	TLC Reset
119	ADMR = C0	ton,lon
007	CCR0 = 80	Start channel 0
115	IMR2 = 10	DMA in enable
11B	AUXMR = 0	Immediate execute pon
113	ISR1 = 02?	wait for DI cleared before writing a byte to TLC Data In Register
111	DIR = 55	send data to TLC, DIR is full, TLC request for a DMA transfer to put the byte in DIR to memory
000	CSR0 = 81?	DMA channel finished (COC)
040	CSR1 = 02?	GPIB synchronized (PCL)
113	ISR1 = 02?	DIR is empty, DI is cleared
	daddr= 55?	verify data has been transferred from TLC to memory
101	CFG1A = 18	clear GPIB synchronization detecting circuitry, also to pull PCL1 high
040	CSR1 = 02	clear PCT bit in CSR1
040	CSR1 = 01?	PCT bit cleared, PCL1 high

13. Test the DMA transfer (flyby) from GPIB to memory, one-byte transfer. This test uses the carry cycle feature.

    Note: Addresses 3000 to 300E hex are used for this test. Other locations can be used if required.

3000	data = 0102	two data bytes (01 and 02) to be transferred on Channel 0
3002	data = 0306	data byte (03) and CC byte (06=SEOI) to be transferred on Channel 1
3004	data = 0000	define carry cycle array's first entry
3006	data = 3003	4-byte address of carry cycle byte (00003003)
3008	data = 0001	first entry's transfer count (0001)
300A	data = 0000	define carry cycle array's second entry
300C	data = 3002	4-byte address of last data byte (00003002)
300E	data = 0002	second entry's transfer count (0002)
105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR
101	CFG1A = 1C	BRG3*, OUT, CC, enable ROR feature
004	DCR0 = A0	
044	DCR1 = A0	
005	OCR0 = 02	
045	OCR1 = 0A	
006	SCR0 = 04	
046	SCR1 = 04	
029	MFC0 = 06	
00C	MAR0 = 00003000	4-byte address (00003000) of the first two data bytes
00A	MTC0 = 0002	2-byte transfer count (0002)
069	MFC1 = 06	

079	BFC1 = 06	
05C	BAR1 = 00003004	4-byte address (00003004) of the carry cycle array
05A	BTC1 = 0002	carry cycle array has two entries - two small memory blocks to be transferred
000	CSR0 = FF	
040	CSR1 = FF	
11B	AUXMR = 2	TLC Reset
119	ADMR = C0	ton,lon
115	IMR2 = 20	DMA out enable
047	CCR1 = 80	start channel 1
007	CCR0 = 80	start channel 0
11B	AUXMR = 0	Immediate execute pon, TLC immediately sets DO in ISR1, requests a DMA transfer for a byte from memory
113	ISR1 = 1?	DO is cleared here because a byte has been transferred from memory to TLC's CDOR. TLC does not currently request for DMA transfer
111	CDOR = 1?	check the first data byte that was transferred from memory to TLC, after this read the TLC will request for another DMA transfer
113	ISR1 = 1?	DO is cleared here because a byte has been transferred from memory to TLC's CDOR
111	DIR = 2?	second data byte that was transferred from memory to TLC, after this read the TLC will request for another DMA transfer
000	CSR0 = 81?	channel 0 finished (COC)
04A	MTC1 = 0001?	last data byte transferred will make count=1
113	ISR1 = 13?	END, DO, and DI (carry cycle byte will set END bit)
111	DIR = 3?	last data byte
040	CSR1 = 0A?	GPIB synchronized
047	CCR1 = 10	software abort
040	CSR1 = FF	clear status bits

14. Test the DMA transfer (flyby) from GPIB to memory, one-byte, and memory write. This text uses the carry cycle feature.

    Note: Addresses 3000 to 300E hex are used for this test. Other locations can be used if required.

3000	data = 0000	clear two memory locations, these will be written over by data from GPIB
3002	data = 0081	clear memory location 3002 as it will be written over and put carry cycle byte (81 = HLDA) in location 3003
3004	data = 0000	define carry cycle array's first entry
3006	data = 3003	4-byte address of carry cycle byte (00003003)
3008	data = 0001	first entry's transfer count (0001)
300A	data = 0000	define carry cycle array's second entry
300C	data = 3002	4-byte address of last data byte (00003002)
300E	data = 0002	second entry's transfer count (0002)
105	CFG2A = 0A	Set LMR and turn LED green
105	CFG2A = 08	Clear LMR

101	CFG1A = 1D	BRG3*, IN, CC, enable ROR feature
004	DCR0 = A0	Configure DMAC
044	DCR1 = A0	
005	OCR0 = 82	
045	OCR1 = 8A	
006	SCR0 = 04	
046	SCR1 = 04	
029	MFC0 = 06	
00C	MAR0 = 00003000	4-byte address of the first two data bytes
00A	MTC0 = 0002	two data bytes
069	MFC1 = 06	
079	BFC1 = 06	
05C	BAR1 = 00003004	4-byte address of the carry cycle array
05A	BTC1 = 0002	carry cycle array has two entries - two small memory blocks to be transferred
000	CSR0 = FF	
040	CSR1 = FF	
11B	AUXMR = 2	TLC Reset
119	ADMR = C0	ton,lon
115	IMR2 = 10	DMA in enable
047	CCR1 = 80	start channel 1
007	CCR0 = 80	start channel 0
11B	AUXMR = 0	Immediate execute pon
113	ISR1 = 2?	check if DI is cleared before write data byte to DIR
111	DIR = 1	write first data byte to TLC, since DIR is full, TLC will request for a DMA transfer to put the byte in DIR to memory
113	ISR1 = 2?	after transferred the first data byte, check if DI is cleared before write second byte to DIR
111	DIR = 2	write second data byte to TLC, since DIR is full, TLC will request for a DMA transfer to put the byte in DIR to memory
000	CSRO = 81?	channel 0 finished (COC)
113	ISR1 = 2?	after transferred the first two bytes on Channel 0 and the carry cycle byte on Channel 1, check if DI is cleared before write the last data byte to DIR
111	DIR = 3	write the last data byte to TLC, since DIR is full, TLC will request for a DMA transfer to put the byte in DIR to memory
04A	MTC1 = 0001?	count of 1 indicates last data byte has been transferred
040	CSR1 = 0A?	GPIB synchronized
047	CCR1 = 10	software abort
040	CSR1 = FF	
3000	data = 01?	check the first data byte that was transferred from TLC to memory
3001	data = 02?	check second data byte
3002	data = 03?	check last data byte

Appendix A
Specifications

IEEE-488 Bus Transfer Rate

DMA	Up to 500 kbytes/sec
Programmed I/O	Up to 80 kbytes/sec

Power Requirement

+5 VDC	2.5 A typical 2.9 A maximum

Physical

Board dimensions	6.3 x 9.2 in.
Input/output connector	IEEE-488 standard 24-pin

Operating Environment

Component temperature	0° to 70° C
Relative humidity	10% to 90% noncondensing

Storage Environment

Temperature	-55° to 71° C
Relative humidity	0% to 100% noncondensing

Appendix B
PAL Drawings, Parts List, and Schematic Diagrams

This appendix contains the PAL drawings, parts list, and schematic diagrams for the GPIB-1014D.

Art not available in PDF version of document.

Appendix C
Sample Programs

This appendix contains listings of routines in 68000 assembly language code that implement the essential elements of these major utility functions:

- Initialize the GPIB-1014 interface (INIT).
- Initialize the interface functions of the GPIB devices (IFC).
- Set or clear the GPIB REN line (REN).
- Accept data bytes from a Talker (RCV).
- Address Talker and read device-dependent messages (READ).
- Send data bytes to Listeners (DSEND).
- Address Listener and write device-dependent messages (WRITE).
- Send command bytes to Listeners (CSEND).
- Write interface messages (CMD).
- Pass GPIB control to another device (PASSC).

Assumptions regarding the state of the GPIB-1014 appear at the beginning of each routine and must be adhered to for proper, error-free operation.

The following characteristics of the code must be considered:

  - Standard 24-bit DMA memory addressing is used.
  - The GPIB-1014 base address is FF2000 hex.
  - Normal (non-extended) GPIB addressing is used.

- Time-out on subroutine calls is not implemented.
- Register values are not saved on subroutine calls.
- Program interrupt is not used; status checking is by register polling.
- Constants and variables listed in the User-Specified Constants section of the listings must be initialized to correct values.
- In operands containing expressions, + is used in place of logical OR for convenience. An arithmetic addition yields the same result in the instances used here

```
                                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                                |                                                       ;
                                                |   GPIB-1014 Sample Functions for Driver:              ;
                                                |                                                       ;
                                                |           INIT     (Initialize the GPIB-1014)         ;
                                                |           IFC      (Send Interface Clear)             ;
                                                |           REN      (Set/Clear Remote Enable)          ;
                                                |           RCV      (Receive)                          ;
                                                |           READ     (Read Data)                        ;
                                                |           DSEND    (Data Send)                        ;
                                                |           WRITE    (Write Data)                       ;
                                                |           CSEND    (Command Send)                     ;
                                                |           CMD      (Write Commands)                   ;
                                                |           PASSC    (Pass Control)                     ;
                                                |                                                       ;
                                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                                |
                                                |
BASE    =   0xFF2000                            | Base address of GPIB-1014 interface
DIR     =   BASE + 0x111                        | Data In Register (read)
CDOR    =   BASE + 0x111                        | Control/Data Out Register (write)
ISR1    =   BASE + 0x113                        | Interrupt Status Register 1 (read)
IMR1    =   BASE + 0x113                        | Interrupt Mask Register 1 (write)
ISR2    =   BASE + 0x115                        | Interrupt Status Register 2 (read)
IMR2    =   BASE + 0x115                        | Interrupt Mask Register 2 (write)
SPSR    =   BASE + 0x117                        | Serial Poll Status Register (read)
SPMR    =   BASE + 0x117                        | Serial Poll Mask Register (write)
ADSR    =   BASE + 0x119                        | Address Status Register (read)
ADMR    =   BASE + 0x119                        | Address Mode Register (write)
CPTR    =   BASE + 0x11B                        | Command Pass Thru Register (read)
AUXMR   =   BASE + 0x11B                        | Auxiliary Mode Register (write)
ADR0    =   BASE + 0x11D                        | Address Register 0 (read)
ADR     =   BASE + 0x11D                        | Address Register (write)
ADR1    =   BASE + 0x11F                        | Address Register 1 (read)
EOSR    =   BASE + 0x11F                        | End Of String Register (write)
MTC0    =   BASE + 0x00A                        | Channel 0 Memory Transfer Count
MAR0    =   BASE + 0x00C                        | Channel 0 Memory Address Register
MFC0    =   BASE + 0x029                        | Channel 0 Memory Function Code
CSR0    =   BASE + 0x000                        | Channel 0 Status Register
DCR0    =   BASE + 0x004                        | Channel 0 Device Control Register
OCR0    =   BASE + 0x005                        | Channel 0 Operation Control Register
SCR0    =   BASE + 0x006                        | Channel 0 Sequence Control Register
CCR0    =   BASE + 0x007                        | Channel 0 Channel Control Register
MTC1    =   BASE + 0x04A                        | Channel 1 Memory Transfer Count
MAR1    =   BASE + 0x04C                        | Channel 1 Memory Address Register
MFC1    =   BASE + 0x069                        | Channel 1 Memory Function Code
BTC1    =   BASE + 0x05A                        | Channel 1 Base Transfer Count
BAR1    =   BASE + 0x05C                        | Channel 1 Base Address Register
BFC1    =   BASE + 0x079                        | Channel 1 Base Function Code
CSR1    =   BASE + 0x040                        | Channel 1 Status Register
DCR1    =   BASE + 0x044                        | Channel 1 Device Control Register
OCR1    =   BASE + 0x045                        | Channel 1 Operation Control Register
SCR1    =   BASE + 0x046                        | Channel 1 Sequence Control Register
CCR1    =   BASE + 0x047                        | Channel 1 Channel Control Register
CFG1    =   BASE + 0x101                        | Configuration Register 1
CFG2    =   BASE + 0x105                        | Configuration Register 2
```

```
                                | ISR1 Bits
DI      =   001 (octal)             | Data in
DO      =   002                     | Data out
ERR     =   004                     | Error
ENDRX   =   020                     | END received

                                | ISR2 Bits
CO      =   010                     | Command out

                                | ADSR Bits
NATN    =   0100                    | Not ATN

                                | ADMR Bits
MODE1   =   001                     | Address Mode 1
TRM     =   060                     | GPIB-1014 functions for T/R2 and T/R3

                                | ADR Bits
DT1     =   0100                    | Disable Talker
DL1     =   0-40                    | Disable Listener

                                | AUXMR Hidden Registers
ICR     =   040                     | Internal Counter Register
PPR     =   0140                    | Parallel Poll Register
AUXRA   =   0200                    | Auxiliary Register A
AUXRB   =   0240                    | Auxiliary Register B
AUXRE   =   0300                    | Auxiliary Register E

                                | AUXMR Commands
IEPON   =   000                     | Immediate execute power on
FH      =   003                     | Finish (release) handshake
GTS     =   020                     | Go to standby
TCA     =   021                     | Take control asynchronously
TCS     =   022                     | Take control synchronously
TCSE    =   032                     | Take control synchronously on END
LTN     =   023                     | Listen
LTNC    =   033                     | Listen continuously
LUN     =   034                     | Unlisten
SIFC    =   036                     | Set IFC
CIFC    =   026                     | Clear IFC
SREN    =   037                     | Set REN
CREN    =   027                     | Clear REN
CRST    =   002                     | Chip Reset

                                | OCR Bits
GTM     =   202                     | Transfer from GPIB to memory
MTG     =   002                     | Transfer from memory to GPIB
ACHN    =   010                     | Array Chaining

                                | SCR Bits
MCU     =   004                     | Memory address counts up

                                | CCR Bits
GO      =   0200                    | Start DMA channel
STOP    =   020                     |Stop DMA channel
```

```
                                | CSR Bits
CLRS    =       0377            |       Clear all status bits
COC     =       0200            |       Channel operation complete
CERR    =       020             |       Error in channel operation
ACT     =       010             |       Channel active
PCT     =       002             |       PCL transition occurred

                                | CFG1 Bits
ECC     =       004             |       Enable Carry Cycle feature
IN      =       001             |       Accepting data from GPIB
OUT     =       000             |       Sending data to GPIB

                                | CFG2 Bits
SLMR    =       012             |       Set Local Master Reset bit
CLMR    =       010             |       Clear Local Master Reset bit

                                GPIB Commands
TCT     =       011             |       Take control
UNL     =       077             |       Universal unlisten
UNT     =       0137            |       Universal untalk

                                | User Specified Constants
SEL0    =       0               |       Select ADR0
SEL1    =       0200            |       Select ADR1
MA      =       0               |       GPIB address of GPIB-1014
SC      =       011             |       System Controller (set to 010 if not Sys. Con.)
ROR     =       002             |       Release on Request feature (set to 0 if not used)
TMODE   =       0240            |       Cycle Steal DMA transfer mode (set to 340 if Cycle Steal
                                |       with Hold is desired)
BRG     =       030             |       Bus Request BR3* selected, other options are:
                                |       (BR2*=020,BR1*=010,BR0*=000)
ADMC    =       06              |       Address Modifiers set to 24 bit, supervisor access of
                                |       data area

                                | Program Variables/Buffers
        .even
cmdbuf: .=      .+100           |       Command buffer for interface messages
cmdct:  .word       0           |       Number of commands to be sent
datbuf: . =     .+100           |       Data buffer for device-dependent messages
count:  .word       0           |       Current number of commands transferred
datct:  .word       0           |       Number of data bytes to be sent
ccary:  .word       0,0,1,0,0,2 |       Carry Cycle Array
ola:    .byte       0           |       Listen address passed to WRITE
sre:    .byte       0           |       REN flag (zero to not set REN, non-zero to set REN)
tctadr: .byte       0           |       TCT address of new Active Controller
vecc:   .byte       0           |       ECC flag (set appropriately by RCV)
vseoi:  .byte       0           |       SEOI flag (zero to not send, non-zero to send
                                |       END message with last DSEND byte)
cic:    .byte       0           |       Controller-In-Charge flag (non-zero if CIC)
HLDA:   .byte   0x81            |       Holdoff on all command AUXMR
SEOI:   .byte   0x06            |       Send END command to AUXMR
```

```
|
|  * * * * * * * * * * * * * * *       *
|  *   INITIALIZE - INIT *
|  * * * * * * * * * * * * * * *       *
|
| Summary:
|   - Initialize the GPIB-1014 hardware
|
| Assumptions on entry:
|   - User specified constants MA, ADMC, and SC have
|     been initialized
|   - Mode 1 primary addressing is used
|   - Low speed timing is used
|   - Interrupts are not used
|   - Status byte will be set elsewhere
|   - Remote Parallel Poll configuration will be used
|
| Actions:
|   - Pulse LMR to put hardware in known reset state
|   - Disable interrupts and clear status
|   - Set hardware registers to desired values
|
| Status on return:
|   - The following registers are cleared:  SR1/2, IMR1/2,
|     SPMR, SPSR, BCR, ACR, PPR, AUXRA, AUXRB,
|     AUXRE
|   - Other registers are configured as described
|   - The GPIB-1014 interface functions are reset to idle
|     and are enabled
```

```
                    68000     Code          |          Comments
                                            |
--------------------------------------------------------------------------------------------------------------------------
INIT:  movb  #SLMR,CFG2                     | Pulse Local Master Reset
       movb  #CLMR,CFG2                     |
                                            |
       movb  #CRST,AUXMR                    | Chip Reset
                                            |
       movb  #0,IMR1                        | Disable TLC interrupts
       movb  #0,IMR2                        |
                                            |
       tstb  ISR1                           | Clear status bits by reading registers
       tstb  ISR2                           |
                                            |
       movb  #MODE1+TRM,ADMR                | Set address mode, Talker/Listener inactive, and proper
                                            | T/R signal mode
                                            |
       movb  #MA+SEL0,ADR                   | Set GPIB address (mode 1 primary only), with
                                            | Talker/Listener enabled
                                            |
       movb  #DT1+DL1+SEL1,ADR              | Disable secondary address recognition
                                            |
       movb  #ICR+8,AUXMR                   | Set clock divider for 8MHz, low speed
                                            |
       movb  #ADMC,MFC0                     | Set DMA address modifier codes
       movb  #ADMC,MFC1                     |
       movb  #ADMC,BFC1                     |
                                            |
       movb  #SC,CFG2                       | By default be system controller
                                            |
       movb  #IEPON,AUXMR                   | Execute pon to bring TLC online
                                            |
       rts                                  |
```

```
                                        |
                                        | ************************      *
                                        | *  INTERFACE CLEAR - IFC*
                                        | ************************      *
                                        |
                                        | Summary:
                                        |   - Initialize the interface function of other GPIB
                                        |     devices
                                        |
                                        | Assumptions on entry:
                                        |   - GPIB-1014 has been initialized
                                        |   - GPIB-1014 is System Controller (SC is true)
                                        |
                                        | Actions:
                                        |   - Assert GPIB IFC
                                        |   - Wait at least 100 microseconds
                                        |   - Unassert IFC
                                        |
                                        | Status on return:
                                        |   - GPIB-1014 is Active Controller
                                        |   - Interface functions of other GPIB devices are reset
                                        |     to their idle states
                                        |
                  68000    Code         |          Comments
--------------------------------------------------------------------------------------------------------------------------------
                                        |
                                        |
IFC:    movb    #SIFC,AUXMR             | Set the IFC signal
                                        |
        movb    #50,d1                  | Wait at least 100 microseconds
IFC1:   subb    #1,d1                   |
        bne     IFC1                    |
                                        |
        movb    #CIFC,AUXMR             | Clear IFC
                                        |
        rts                             |
```

```
                                        |
                                        | * * * * * * * * * * * * * * * * * * *         *
                                        | * REMOTE ENABLE - REN    *
                                        | * * * * * * * * * * * * * * * * * * *         *
                                        |
                                        | Summary:
                                        |   - Set or clear GPIB Remote Enable signal
                                        |
                                        | Assumptions on entry:
                                        |   - User specified sre is non-zero if REN is to be
                                        |     asserted and is zero if REN is to be unasserted
                                        |   - GPIB-1014 is System Controller and Active
                                        |     Controller
                                        |
                                        | Actions:
                                        |   - Check sre flag.
                                        |     if non-zero (true) send REN else send clear REN
                                        |
                                        | Status on return:
                                        |   - REN is asserted or unasserted
                                        |
          68000    Code                 |          Comments
-------------------------------------------------------------------------------------------------------------------------------
                                        |
REN:    tstb    sre                     | Turn on the REN signal if sre is non-zero
        beq     REN1                    |
                                        |
        movb    #SREN,AUXMR             |
        bra     REN2                    |
                                        |
REN1:   movb    #CREN,AUXMR             | Else, turn off REN if sre is zero
                                        |
REN2:   rts                             |
                                        |
```

```
|
| * * * * * * * * * * * * * *  *
|   *  RECEIVE - RCV   *
| * * * * * * * * * * * * * *  *
|
| Summary:
|   - Called by READ to receive data if GPIB-1014 is
|     Controller-In-Charge
|   - Called directly from main program to receive data if
|     GPIB-1014 is Idle Controller
|
| Assumptions on entry:
|   - GPIB-1014 is Standby or Idle Controller
|   - GPIB-1014 is or will be addressed to listen
|   - The GPIB Talker has been or will be addressed
|   - The Talker will send END with last byte if the number
|     of bytes sent is less than the byte count
|   - The d0 register contains the byte count
|   - The a0 register contains the address of the data
|     buffer
|   - The user-specified variable cic is set properly
|
| Actions:
|   - Clear DMAC channel 0 and 1 status registers
|   - Configure channel 0 to transfer all but the last byte
|   - Configure channel 1 for carry cycle to implement
|     handshake holdoff after last byte
|   - Start the DMA channels
|   - Release any holdoff in progress
|   - Set IMR2 to enable DMAs
|   - Wait for DMA done or GPIB END message
|   - Disable DMA
|   - Set d0 register to number of bytes received
|
| Status on return:
|   - a NRFD handshake holdoff is in effect
|   - The number of bytes transferred is in d0
```

```
          68000    Code                  |          Comments
-------------------------------------------------------------------------------------------------------------------------
                                         |
RCV:      movb     #TMODE,DCR0           | Set DMA transfer mode
          movb     #TMODE,DCR1           |
                                         |
          movb     #GTM,OCR0             | Set control registers
          movb     #MCU,SCR0             |
                                         |
          movb     #FF,CSR0              | Clear status registers
          movb     #FF,CSR1              |
                                         |
          movl     a0,MAR0               | Point channel 0 to buffer
                                         |
          cmpb     #0,cic                | Is GPIB-1014 Controller-In-Charge
          beq      RCV1                  |
                                         |
                                         | Yes, set up carry cycle feature
                                         |
          movb     #GTM+ACHN,OCR1        |  - Enable chaining on channel 1
                                         |
          movl     #ccary,BAR1           |  - Point channel 1 to ccary
                                         |
          movw     #2,BTC1               |  - 2 elements in ccary
                                         |
          movl     #HLDA,ccary           |  - First ccary address points to HLDA
                                         |
          movw     d0,d1                 |  - Set channel 0 transfer count
          subw     #1,d1                 |    to transfer all but the last
          movw     d1,MTC0               |    byte
                                         |
          movl     a0,d2                 |  - Second ccary address points to
          addl     d2,d1                 |    last data byte
          movl     d1,ccary+6            |
                                         |
                                         |
          movb     #BRG+ECC+IN,CFG2      |  - configure CFG2 for carry cycle
                                         |
          movb     #AUXRA+022,AUXMR      |  - Set HLDE and BIN in AUXRA
                                         |
          movb     #GO,CCR1              |  - Start channel 1
          bra      RCV2                  |
                                         |
RCV1:     movb     #BRG+IN,CFG2          | No - no carry cycle
          movb     #AUXRA,AUXMR          | Clear any HLDE or HLDA in effect
          movw     d0,MTC0               | Channel 0 transfers all bytes
                                         |
RCV2:     movb     #GO,CCR0              | Start channel 0
          movb     #DMAI,IMR2            | Enable DMAs from the DIR
          movb     #FH,AUXMR             | Release any handshake holdoff in progress
                                         |
RCV3:     btst     #ENDRX,ISR1           | Loop waiting for ENDRX or PCT (DMA done)
          bne      RCV4                  |
          btst     #PCT,CSR1             |
          beq      RCV3                  |
```

```
                                    |
RCV4:  btst    #COC,CSR0            | Calculate number of bytes transferred
       bne     RCV5                 |
       movw    MTC0,d1              |
       subw    d1,d0                |
       btst    #ECC,CFG1            | If carry cycle, MTC0 was initialized to (d0)-1
       beq     RCV6                 |
       subw    #1,d0                |
       bra     RCV6                 |
                                    |
RCV5:  btst    #ECC, CFG1           | If no carry cyle, leave d0 as is
       bne     RCV6                 |
       movw    MTC1,d1              |
       subw    d1,d0                |
       addw    #1,d0                |
                                    |
RCV6:  movb    #0,IMR2              | Disable DMAs and stop DMA channels
       movb    #STOP,CCR1           |
       movb    #STOP,CCR0           |
                                    |
       rts                          |
                                    |
```

```
|
| * * * * * * *    *
| *  READ *
| * * * * * * *    *
| Summary:
|   - Called to read device-dependent (data) messages
|     when the GPIB-1014 is Controller-In-Charge (RCV
|     is called when the GPIB-1014 is Idle Controller)
|
| Assumptions on entry:
|   - GPIB-1014 is Controller-In-Charge
|   - The Talker address is placed in first location of
|     cmdbuf
|   - The variable cmdct is set to 1
|   - The buffer datbuf is free to place incoming data
|   - The number of bytes to read is placed in datct
|
| Actions:
|   - Set up cmdbuf and cmdct and call CMD to address
|     the Talker and unaddress all other devices
|   - Program the GPIB-1014 to listen
|   - Go to standby and unassert ATN
|   - Transfer the contents of datct to the d0 register
|   - Load the a0 register with the address of datbuf
|   - Call RCV to receive the data
|   - Call CMD to unaddress all devices
|   - Program the GPIB-1014 to unlisten
|
| Status on return:
|   - GPIB-1014 is Active Controller
|   - Acceptor handshake is held off at NRFD
|   - All GPIB devices are unaddressed
|
```

```
          68000    Code                |          Comments
--------------------------------------------------------------------------------------------------------------------------
                                       |
READ:     movb     cmdbuf,cmdbuf+2     | Put Untalk and Unlisten commands before
          movb     #UNT,cmdbuf         | Talker address in the buffer
          movb     #UNL,cmdbuf+1       |
          addw     #2,cmdct            |
                                       |
          bsr      CMD                 | Command routine will address the Talker
                                       |
          movb     #LTN,AUXMR          | Program GPIB-1014 to be a Listener
          movb     #GTS,AUXMR          | so it can take control synchronously
                                       | later; then go to standby and drop ATN
                                       |
          movw     #datct,d0           | Preset d0 register with byte count
          movl     #datbuf,a0          | Preset a0 register with buffer address
          bsr      RCV                 | Receive routine will read data
                                       |
          movb     #TCS,AUXMR          | Take control
                                       |
READ1:    btst     #NATN,ADSR          | Wait for ATN, indefinitely
          bne      READ1               |
                                       |
          subw     #1,cmdct            | Unaddress all Talkers and Listeners
          bsr      CMD                 | using CMD
                                       |
          movb     #LUN,AUXMR          | Send Local Unlisten command
                                       |
          rts                          |
```

```
|
| * * * * * * * * * * * * * * * * * * *
| * DATA SEND - DSEND    *
| * * * * * * * * * * * * * * * * * * *
|
| Summary:
|   - Called directly from the main program if the
|     GPIB-1014 is not CIC
|
| Assumptions on entry:
|   - The GPIB-1014 is Standby or Idle Controller
|   - GPIB-1014 is or will be addressed to talk
|   - If the GPIB-1014 is Idle Controller, the current CIC
|     will go to standby
|   - The d0 register contains the byte count
|   - The a0 register contains the address of the data
|     buffer
|   - The user specified variable veoi has been set
|     properly
|
| Actions:
|   - Configure DMA channel 0
|   - Clear DMAC status registers
|   - Configure channel 1 for carry cycle if END is to be
|     sent with last byte
|   - Start DMA channels
|   - Set IMR2 to enable DMAs
|   - Wait for DMA done or a GPIB error
|   - Place in the d0 register the number of bytes
|     transferred or -1 on an error
|   - Disable further DMAs
|
| Status on return:
|   - The d0 register contains the number of bytes
|     transferred or a -1 to indicate an error
```

```
          68000    Code                    |        Comments
-----------------------------------------------------------------------------------------------------------------------
                                           |
DSEND:    movb     #TMODE,DCR0             | Set DMA transfer mode
          movb     #TMODE,DCR1             |
                                           |
          movb     #MTG,OCR0               | Set control registers
          movb     #MCU,SCR0               |
                                           |
          movb     #FF,CSR0                | Clear status registers
          movb     #FF,CSR1                |
                                           |
          movl     a0,MAR0                 | Point channel 0 to buffer
          tstb     vseoi                   | Send END with last byte?
          beq      DSEND1                  |
                                           |
                                           | Yes, enable carry cycle feature
                                           |
          movb     #MTG+ACHN,OCR1          |  - Enable chaining on channel 1
                                           |
          movw     d0,d1                   |  - Channel 0 transfer all but the
          subw     #1,d1                   |    last data byte
          movw     d1,MTC0                 |
                                           |
          movl     #ccary,BAR1             |  - Point channel 1 to ccary
                                           |
          movw     #2,BTC1                 |  - 2 elements in ccary
                                           |
          movl     #SEOI,ccary             |  - First ccary address points to SEOI
                                           |
          movl     a0,d2                   |  - Second ccary address points to last
          addl     d2,d1                   |    data byte
          movl     d1,ccary+6              |
                                           |
          movb     #BRG+ECC+OUT,CFG1       |   - Set CFG2 for carry cycle
          movb     #G0,CCR1                |   - Start channel 1
          bra      DSEND2                  |
                                           |
                                           |
DSEND1:   movb     #BRG+OUT,CFG2           | No--Configure CFG2 without ECC
                                           |
          movw     #d0,MTC0                | Channel 0 transfers all bytes
                                           |
DSEND2:   movb     #GO,CCR0                | Start channel 0
                                           |
          movb     #DMAO,IMR2              | Enable DMA to the CDOR
                                           |
DSEND3:   btst     #PCT,CSR1               | Loop waiting for GPIB error or DMA done
          bne      DSEND4                  |
          btst     #ERR,ISR1               |
          beq      DSEND3                  |
          movw     #-1,d0                  | set count to -1 if error occurred
          bra      DSEND6                  |
                                           |
```

```
DSEND4:  btst    #COC,CSR0        | Calculate number of bytes transferred
         bne     DSEND5           |
         movw    MTC0,d1          |
         subw    d1,d0            |
         btst    #ECC,CFG1        |
         beq     DSEND6           |
         subw    #1,d0            |
         bra     DSEND6           |
                                  |
DSEND5:  btst    #ECC,CFG1        |
         bne     DSEND6           |
         movw    MTC1,d1          |
         subw    d1,d0            |
         addw    #1,d0            |
                                  |
DSEND6:  movb    #0,IMR2          | Disable DMAs
                                  |
         movb    #STOP,CCR1       | Stop DMA channels
         movb    #STOP,CCR0       |
                                  |
         rts                      |
                                  |
```

```
|
| * * * * * * * * *
| *  WRITE    *
| * * * * * * * * *
|
| Summary:
|   - Called to send device-dependent (data) messages
|     when the GPIB-1014 is Controller-In-Charge
|      (DSEND is called when the interface is Idle
|     Controller)
|
| Assumptions on entry:
|   - GPIB-1014 is CIC
|   - One Listener is addressed and its address is placed
|     in the variable ola
|   - The data to be sent is placed in datbuf
|   - The variable datct contains the number of bytes to
|     send
|
| Actions:
|   - Set up cmdbuf and cmdct and call CMD to address
|     the GPIB-1014 as Talker, to address the Listener,
|     and to unaddress all other devices
|   - Go to standby and unassert ATN
|   - Transfer the contents of datct to the d0 register
|   - Load a0 register with the address of datbuf
|   - Call DSEND to write the data
|   - When the last byte has been sent, take control
|   - Call CMD to unaddress all devices
|
| Status on return:
|   - The GPIB-1014 is Active Controller
|   - All GPIB devices are unaddressed
|
```

```
                 |
  68000   Code   |          Comments
-------------------------------------------------------------------------------
                               |
WRITE:  movw  #4,cmdct         | Put Untalk, Unlisten, MTA, and OLA
        movb  #UNT,cmdbuf      | commands in the buffer
        movb  #UNL,cmdbuf+1    |
        movb  #MA+100,cmdbuf+2 |
        movb  #ola,cmdbuf+3    |
                               |
        bsr   CMD              | Call CMD to address GPIB devices
                               |
        movb  #GTS,AUXMR       | Go to standby and drop ATN
                               |
        movw  #datct,d0        | Preset d0 register with byte count
        movl  #datbuf,a0       | Preset a0 register with address of buffer
                               |
        bsr   DSEND            | Call DSEND to send data
                               |
WRITE1:btst   #DO,ISR1         | Wait until last byte has been sent
        beq   WRITE1           |
                               |
        movb  #TCA,AUXMR       | Then take control
                               |
        subw  #2,cmdct         | Use CMD to unaddress GPIB devices
        bsr   CMD              |
                               |
        rts                    |
                               |
```



```
                                        |
                                        | ***************************
                                        | *  COMMAND SEND - CSEND      *
                                        | ***************************
                                        |
                                        | Summary:
                                        |     - Called by CMD to send interface  messages
                                        |
                                        | Assumptions on entry:
                                        |   -  The GPIB-1014 is Active Controller
                                        |   -  The d0 register contains the number of bytes to
                                        |      send
                                        |   -  The a0 register contains the address oc cmdbuf
                                        |
                                        | Actions:
                                        |   -  Initialize a count variable
                                        |   -  Wait until the CDOR is empty
                                        |   -  Write a byte and increment the counter
                                        |   -  Check for a GPIB error
                                        |   -  Loop until all bytes are transferred
                                        |   -  On an error, set d0 to -1
                                        |
                                        | Status on return:
                                        |   -  d0 register contains number of bytes sent or -1 if an
                                        |      error occurred
                                        |
                                        |
          68000    Code                 |          Comments
-------------------------------------------------------------------------------------------------------------------------------
                                        |
CSEND:    clrw     count                | Initialize count variable
                                        |
CSEND1:   btst     #CO,ISR2             | Wait till CDOR is empty
          beq      CSEND1               |
                                        |
          cmpw     #count,d0            | Have all commands been sent?
          beq      CSEND3               | Yes
                                        |
          addw     #1,count             | No--Increment counter and write
          movb     (a0)+,CDOR           | the next command
                                        |
          btst     #ERR,ISR1            | If there are no Listeners, return -1
          bne      CSEND2               | in d0 register
                                        |
          bra      CSEND1               |
                                        |
CSEND2:   movw     #-1,d0               |
                                        |
CSEND3:   rts                           |
```

```
                                        |
                                        | * * * * * * * * * * * * * * * *
                                        | * COMMAND - CMD   *
                                        | * * * * * * * * * * * * * * * *
                                        |
                                        | Summary:
                                        |   - Send GPIB interface or command messages
                                        |
                                        | Assumptions on entry:
                                        |   - The GPIB-1014 is Controller-In-Charge
                                        |   - The commands to be sent are in cmdbuf
                                        |   - The variable cmdct contains the number of
                                        |     commands to be sent, which must be less than 256
                                        |   - Interruption of any data transfer in progress is
                                        |     acceptable
                                        |
                                        | Actions:
                                        |   - Issue TCA command to assert ATN in case the
                                        |     GPIB-1014 is at standby
                                        |   - Load the d0 register with the address of cmdbuf
                                        |   - Load a0 with the number of commands
                                        |   - Call CSEND to transmit the bytes
                                        |
                                        | Status on return:
                                        |   - GPIB-1014 is Active Controller
                                        |   - GPIB devices are programmed as implied by
                                        |     command bytes
                                        |
                                        |
          68000    Code                 |          Comments
--------------------------------------------------------------------------------------------------------------------------------
                                        |
CMD:    movb    #TCA,AUXMR              | Take control in case at standby
                                        |
        movl    #cmdbuf,a0              | Set up registers for CSEND call
        movw    #cmdct,d0               |
                                        |
        bsr     CSEND                   | Call CSEND to send commands
                                        |
                rts                     |
```

```
                                        |
                                        | ************************
                                        | *  PASS CONTROL - PASSC    *
                                        | ************************
                                        |
                                        | Summary:
                                        |     - Passes GPIB Controller-In-Charge status to another
                                        |     device
                                        |
                                        | Assumptions on entry:
                                        |   - The GPIB-1014 is Controller-In-Charge
                                        |   - The primary GPIB address of the new controller is placed
                                        |     in tctadr
                                        |
                                        | Actions:
                                        |   - Send TCA command to take control in case the GPIB-1014
                                        |     is at standby
                                        |   - Set up the command buffer and command count
                                        |   - Call CMD to send the command bytes
                                        |
                                        | Status on return:
                                        |   - The GPIB-1014 is Idle Controller
                                        |
                                        |
          68000    Code                 |          Comments
-------------------------------------------------------------------------------------------------------------------------------
                                        |
PASSC:    movb  #TCA,AUXMR              | Take control in case at standby
                                        |
          movb  #UNT,cmdbuf             | Set up the command buffer
          movb  #UNL,cmdbuf+1           |
          movb  #tctadr,cmdbuf+2        |
          movb  #TCT,cmdbuf+3           | The GPIB-1014 automatically releases
          movw  #4,cmdct                | control when TCT is sent
                                        |
          bsr   CMD                     | Call CMD to send commands
                                        |
          rts                           |
                                        |
```

Appendix D
Multiline Interface Messages

This appendix contains an interface message reference list, which describes the mnemonics and messages that correspond to the interface functions. These multiline interface messages are sent and received with ATN TRUE.

For more information on these messages, refer to the ANSI/IEEE Std 488-1978, *IEEE Standard Digital Interface for Programmable Instrumentation*.

Multiline Interface Messages

Hex	Oct	Dec	ASCII	Msg	Hex	Oct	Dec	ASCII	Msg
00	000	0	NUL		20	040	32	SP	MLA0
01	001	1	SOH	GTL	21	041	33	!	MLA1
02	002	2	STX		22	042	34	"	MLA2
03	003	3	ETX		23	043	35	#	MLA3
04	004	4	EOT	SDC	24	044	36	$	MLA4
05	005	5	ENQ	PPC	25	045	37	%	MLA5
06	006	6	ACK		26	046	38	&	MLA6
07	007	7	BEL		27	047	39	'	MLA7
08	010	8	BS	GET	28	050	40	(	MLA8
09	011	9	HT	TCT	29	051	41	)	MLA9
0A	012	10	LF		2A	052	42	*	MLA10
0B	013	11	VT		2B	053	43	+	MLA11
0C	014	12	FF		2C	054	44	,	MLA12
0D	015	13	CR		2D	055	45	-	MLA13
0E	016	14	SO		2E	056	46	.	MLA14
0F	017	15	SI		2F	057	47	/	MLA15
10	020	16	DLE		30	060	48	0	MLA16
11	021	17	DC1	LLO	31	061	49	1	MLA17
12	022	18	DC2		32	062	50	2	MLA18
13	023	19	DC3		33	063	51	3	MLA19
14	024	20	DC4	DCL	34	064	52	4	MLA20
15	025	21	NAK	PPU	35	065	53	5	MLA21
16	026	22	SYN		36	066	54	6	MLA22
17	027	23	ETB		37	067	55	7	MLA23
18	030	24	CAN	SPE	38	070	56	8	MLA24
19	031	25	EM	SPD	39	071	57	9	MLA25
1A	032	26	SUB		3A	072	58	:	MLA26
1B	033	27	ESC		3B	073	59	;	MLA27
1C	034	28	FS		3C	074	60	<	MLA28
1D	035	29	GS		3D	075	61	=	MLA29
1E	036	30	RS		3E	076	62	>	MLA30
1F	037	31	US		3F	077	63	?	UNL

Message Definitions

DCL Device Clear
GET Group Execute Trigger
GTL Go To Local
LLO Local Lockout
MLA My Listen Address
MSA My Secondary Address
MTA My Talk Address
PPC Parallel Poll Configure
PPD Parallel Poll Disable

Multiline Interface Messages

Hex	Oct	Dec	ASCII	Msg	Hex	Oct	Dec	ASCII	Msg
40	100	64	@	MTA0	60	140	96	`	MSA0,PPE
41	101	65	A	MTA1	61	141	97	a	MSA1,PPE
42	102	66	B	MTA2	62	142	98	b	MSA2,PPE
43	103	67	C	MTA3	63	143	99	c	MSA3,PPE
44	104	68	D	MTA4	64	144	100	d	MSA4,PPE
45	105	69	E	MTA5	65	145	101	e	MSA5,PPE
46	106	70	F	MTA6	66	146	102	f	MSA6,PPE
47	107	71	G	MTA7	67	147	103	g	MSA7,PPE
48	110	72	H	MTA8	68	150	104	h	MSA8,PPE
49	111	73	I	MTA9	69	151	105	i	MSA9,PPE
4A	112	74	J	MTA10	6A	152	106	j	MSA10,PPE
4B	113	75	K	MTA11	6B	153	107	k	MSA11,PPE
4C	114	76	L	MTA12	6C	154	108	l	MSA12,PPE
4D	115	77	M	MTA13	6D	155	109	m	MSA13,PPE
4E	116	78	N	MTA14	6E	156	110	n	MSA14,PPE
4F	117	79	O	MTA15	6F	157	111	o	MSA15,PPE
50	120	80	P	MTA16	70	160	112	p	MSA16,PPD
51	121	81	Q	MTA17	71	161	113	q	MSA17,PPD
52	122	82	R	MTA18	72	162	114	r	MSA18,PPD
53	123	83	S	MTA19	73	163	115	s	MSA19,PPD
54	124	84	T	MTA20	74	164	116	t	MSA20,PPD
55	125	85	U	MTA21	75	165	117	u	MSA21,PPD
56	126	86	V	MTA22	76	166	118	v	MSA22,PPD
57	127	87	W	MTA23	77	167	119	w	MSA23,PPD
58	130	88	X	MTA24	78	170	120	x	MSA24,PPD
59	131	89	Y	MTA25	79	171	121	y	MSA25,PPD
5A	132	90	Z	MTA26	7A	172	122	z	MSA26,PPD
5B	133	91	[	MTA27	7B	173	123	{	MSA27,PPD
5C	134	92	\	MTA28	7C	174	124	|	MSA28,PPD
5D	135	93	]	MTA29	7D	175	125	}	MSA29,PPD
5E	136	94	^	MTA30	7E	176	126	~	MSA30,PPD
5F	137	95	_	UNT	7F	177	127	DEL	

PPE Parallel Poll Enable
PPU Parallel Poll Unconfigure
SDC Selected Device Clear
SPD Serial Poll Disable
SPE Serial Poll Enable
TCT Take Control
UNL Unlisten
UNT Untalk

Appendix E
Operation of the GPIB

Communication among interconnected GPIB devices is achieved by passing messages through the interface system.

Types of Messages

The GPIB carries device-dependent messages and interface messages.

- Device-dependent messages, often called *data* or *data messages*, contain device-specific information such as programming instructions, measurement results, machine status, and data files.

- Interface messages manage the bus itself. They are usually called *commands* or *command messages*. Interface messages perform such tasks as initializing the bus, addressing and unaddressing devices, and setting device modes for remote or local programming.

The term *command* as used here should not be confused with some device instructions which can also be called commands. Such device-specific instructions are actually data messages.

Talkers, Listeners, and Controllers

A Talker sends data messages to one or more Listeners. The Controller manages the flow of information on the GPIB by sending commands to all devices.

Devices can be Listeners, Talkers, and/or Controllers. A digital voltmeter, for example, is a Talker and may be a Listener as well.

The GPIB is a bus like an ordinary computer bus, except that the computer has its circuit cards interconnected via a backplane bus, whereas the GPIB has standalone devices interconnected via a cable bus.

The role of the GPIB Controller can also be compared to the role of the CPU of a computer, but a better analogy is to the switching center of a city telephone system.

The switching center (Controller) monitors the communications network (GPIB). When the center (Controller) notices that a party (device) wants to make a call (send a data message), it connects the caller (Talker) to the receiver (Listener).

The Controller addresses a Talker and a Listener before the Talker can send its message to the Listener. After the message is transmitted, the Controller may unaddress both devices.

Some bus configurations do not require a Controller. For example, one device may always be a Talker (called a Talk-only device) and there may be one or more Listen-only devices.

A Controller is necessary when the active or addressed Talker or Listener must be changed. The Controller function is usually handled by a computer.

With the GPIB interface board and its software your personal computer plays all three roles.

- Controller - to manage the GPIB
- Talker - to send data
- Listener - to receive data

The Controller-In-Charge and System Controller

Although there can be multiple Controllers on the GPIB, only one Controller at a time is active or Controller-In-Charge (CIC). Active control can be passed from the current CIC to an idle Controller. Only one device on the bus, the System Controller, can make itself the CIC. The GPIB interface board is usually the System Controller.

GPIB Signals and Lines

The interface system consists of 16 signal lines and 8 ground return or shield drain lines.

The 16 signal lines are divided into the following three groups.

- Eight data lines
- Three handshake lines
- Five interface management lines

Data Lines

The eight data lines, DI01 through DI08, carry both data and command messages. All commands and most data use the 7-bit ASCII or ISO code set, in which case the eighth bit, DI08, is unused or used for parity.

Handshake Lines

Three lines asynchronously control the transfer of message bytes among devices. The process is called a three-wire interlocked handshake, and it guarantees that message bytes on the data lines are sent and received without transmission error.

NRFD (not ready for data)

NRFD indicates when a device is ready or not ready to receive a message byte. The line is driven by all devices when receiving commands and by Listeners when receiving data messages.

NDAC (not data accepted)

NDAC indicates when a device has or has not accepted a message byte. The line is driven by all devices when receiving commands and by Listeners when receiving data messages.

DAV (data valid)

DAV tells when the signals on the data lines are stable (valid) and can be accepted safely by devices. The Controller drives DAV when sending commands and the Talker drives it when sending data messages.

Interface Management Lines

Five lines are used to manage the flow of information across the interface.

ATN (attention)

The Controller drives ATN true when it uses the data lines to send commands and false when it allows a Talker to send data messages.

IFC (interface clear)

The System Controller drives the IFC line to initialize the bus and become CIC.

REN (remote enable)

The System Controller drives the REN line, which is used to place devices in remote or local program mode.

SRQ (service request)

Any device can drive the SRQ line to asynchronously request service from the Controller with the SRQ line.

EOI (end or identify)

The EOI line has two purposes. The Talker uses the EOI line to mark the end of a message string. The Controller uses the EOI line to tell devices to identify their response in a parallel poll.

Physical and Electrical Characteristics

Devices are usually connected with a cable assembly consisting of a shielded 24 conductor cable with both a plug and receptacle connector at each end. This design allows devices to be linked in

either a linear or a star configuration, or a combination of the two. See Figures E-1, E-2, and E-3.

The standard connector is the Amphenol or Cinch Series 57 *Microribbon* or *Amp Champ* type. An adapter cable using a non-standard cable and/or connector is used for special interconnection applications.

The GPIB uses negative logic with standard TTL logic level. When DAV is true, for example, it is a TTL low level (≤ 0.8V), and when DAV is false, it is a TTL high level (≥ 2.0V).

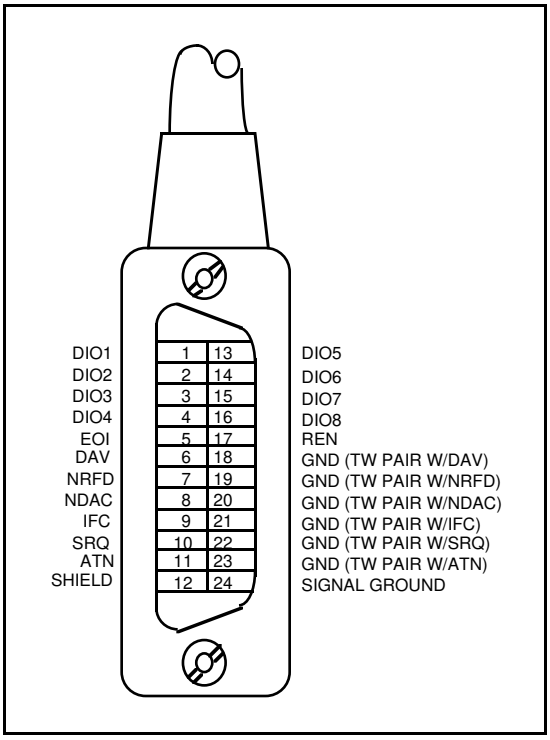


Figure E-1. GPIB Connector and the Signal Assignment

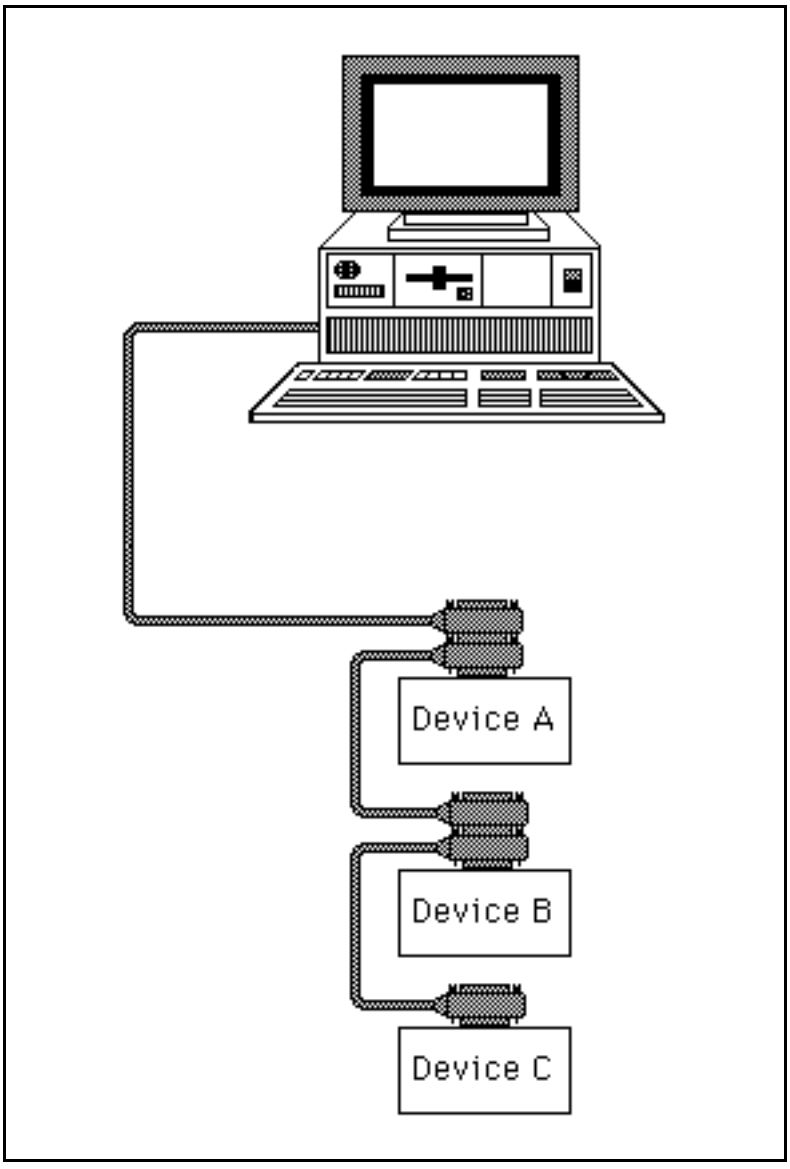


Figure E-2. Linear Configuration

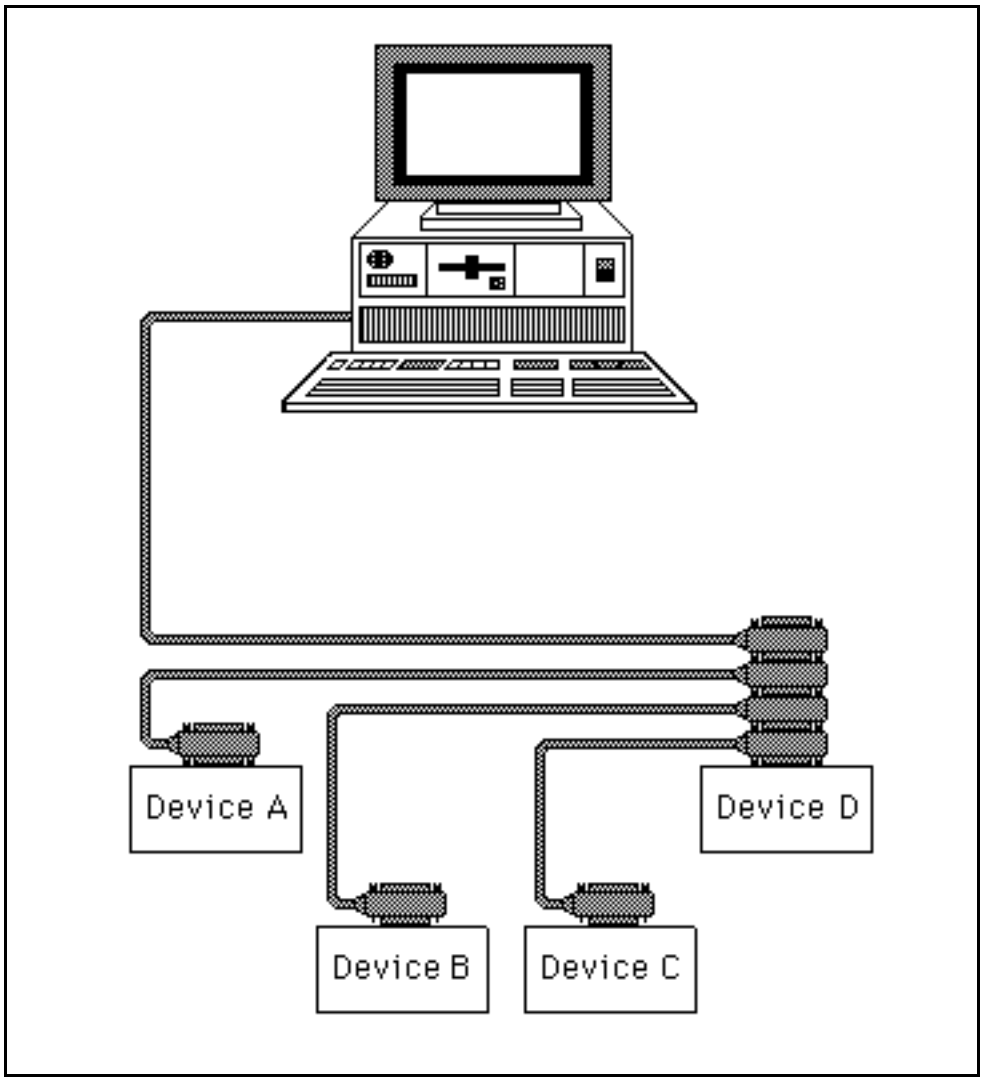


Figure E-3. Star Configuration

Configuration Requirements

To achieve the high data transfer rate that the GPIB was designed for, the physical distance between devices and the number of devices on the bus are limited.

The following restrictions are typical.

- A maximum separation of four meters between any two devices and an average separation of two meters over the entire bus.
- A maximum total cable length of 20 meters.
- No more than 15 devices connected to each bus, with at least two-thirds powered on.

Bus extenders are available from National Instruments and other manufacturers for use when these limits must be exceeded.

Related Document

For more information on topics covered in this section, consult *IEEE Standard Digital Interface for Programmable Instrumentation*, IEEE-488 Std. 488-1978.

Appendix F
Mnemonics Key

This appendix contains a mnemonics key that defines the mnemonics (abbreviations) used throughout this manual for functions, remote messages, local messages, states, bits, registers, integrated circuits, system functions, and VMEbus operations and signals.

The mnemonic types in the key that follows are abbreviated to mean the following:

B	Bit
F	Function
IC	Integrated Circuit
GS	GPIB Signal
LM	Local Message
LS	Local Signal
R	Register
RM	Remote Message
SF	System Function
ST	State
VBO	VMEbus Operation
VBS	VMEbus Signal

Mnemonic	Type	Full Name

Numbers

32BIT	B	Extended Addressing (32-bit) Bit

A

ACDS	ST	Acceptor Data State (AH function)
ACG	C	Addressed Command Group
ACRS	ST	Acceptor Ready State
ACT	B	Channel Active Bit
AD[5-1]	B	TLC GPIB Address Bits 5 through 1
AD[5-0 - 1-0]	B	TLC GPIB Address Bits 5 through 1 (Mode 2)
AD[5-1 - 1-1]	B	TLC GPIB Secondary Address Bits 5 through 1 (Mode 2)
ADM[1-0]	B	Address Mode Bits 1 through 0
ADMR	R	Address Mode Register
ADR	R	Address Register
ADR0	R	Address Register 0
ADR1	R	Address Register 1
ADSC	B	Addressed Status Change Bit
ADSC IE	B	Addressed Status Change Interrupt Enable Bit
ADSR	R	Address Status Register
AH	ST	Acceptor Handshake
AIDS	ST	Acceptor Idle State
ANRS	ST	Acceptor Not Ready State
APRS	ST	Affirmative Poll Response State
APT	B	Address Pass Through Bit
APT IE	B	Address Pass Through Interrupt Enable Bit
ARS	B	Address Register Select Bit
ATN	LS	Attention
ATN*	B	Attention Bit
AUXMR	R	Auxiliary Mode Register
AUXRA	R	Auxiliary Register A
AUXRB	R	Auxiliary Register B
AUXRE	R	Auxiliary Register E
AWNS	ST	Acceptor Wait for New cycle State

B

BAR	R	Base Address Register
BFCR	R	Base Function Code Register
BIN	B	Binary
BR	B	Bandwidth Available to DMAC Bits 1 through 0
BRG	B	Bus/Request Grant Bits 4 through 3
BT	B	Burst Transfer Time Bits 3 through 2
BTC	B	Block Termination Complete Bit
BTCR	R	Base Transfer Counter Register

Mnemonic	Type	Full Name

C

C	F	Controller
CACS	ST	Controller Active State (C function)
CADS	ST	Controller Addressed State
CAWS	ST	Controller Active Wait State
CC	B	Carry Cycle Bit
CCR	R	Channel Control Register
CDOR	R	Control/Data Out Register
CDO[7-0]	B	Control/Data Out Bits 7 through 0
CER	R	Channel Error Register
CFG1	R	Configuration Register 1
CFG2	R	Configuration Register 2
CHN	B	Chaining Mode Bits 3 through 2
CIC	B	Controller-In-Charge Bit
CIDS	ST	Controller Idle State
CLK[3-0]	B	Clock Bits 3 through 0
CNT[2-0]	B	Control Code Bits 2 through 0
CO	B	Command Output Bit
CO IE	B	Command Output Interrupt Enable Bit
COC	B	Channel Operation Complete Bit
COM[4 - 0]	B	Command Code Bits 4 through 0
CP	B	Channel Priority Bits 1 through 0
CPPS	ST	Controller Parallel Poll State
CPR	R	Channel Priority Register
CPT	B	Command Pass Through Bit
CPT ENAB	B	Command Pass Through Enable Bit
CPT IE	B	Command Pass Through Interrupt Enable Bit
CPTR	R	Command Pass Through Register
CPT[7-0]	B	Command Pass Through Bits 7 through 0
CPWS	ST	Controller Parallel Poll Wait State
CSBS	ST	Controller Standby State
CSHS	ST	Controller Standby Hold State
CSNS	ST	Controller Service Not Requested State
CSR	R	Channel Status Register
CSRS	ST	Controller Service Requested State
CSWS	ST	Controller Synchronous Wait State
CTRS	ST	Controller Transfer State (C function)

D

DAB	RM	Data Byte
DAC	RM	Data Accepted
DAC	B	Device Address Count Bits 1 through 0
dacr		Data Accepted Holdoff Release
DAG		Device Address Group
DAR	R	Device Address Register
DAV	SL	Data Valid
DC		Device Clear
DCAS	ST	Device Clear Active State

Mnemonic	Type	Full Name
DCIS	ST	Device Clear Idle State
DCL	RM	Device Clear
DCR	R	Device Control Register
DEC	B	Device Clear Bit
DEC IE	B	Device Clear Interrupt Enable Bit
DET	B	Device Execute Trigger Bit
DET IE	B	Device Execute Trigger Interrupt Enable Bit
DFCR	R	Device Function Code Register
DHDC	B	Data Accepted Holdoff on Device Clear Active State Bits
DHDT	B	Data Accepted Holdoff on Device Trigger Active State Bits
DI	B	Data In Bit
DI IE	B	Data In Interrupt Enable Bit
DIR	R	Data In Register
DIR	B	Direction Bit
DI[7-0]	B	Data In Bits 7 through 0
DL	B	Disable Listener Bit
DL0	B	Disable Listener 0 Bit
DL1	B	Disable Listener 1 Bit
DMAI	B	DMA Input Enable Bit
DMAO	B	DMA Out Enable Bit
DO	B	Data Out Bit
DO IE	B	Data Out Interrupt Enable Bit
DPS	B	Device Port Size Bit
DT		Device Trigger
DT	B	Disable Talker Bit
DT0	B	Disable Talker 0 Bit
DT1	B	Disable Talker 1 Bit
DTAS	ST	Device Trigger Active State
DTIS	ST	Device Trigger Idle State
DTYP	B	Device Type Bits 5 through 4

E

EINT	B	Interrupt Enable Bit
EIVR	R	Error Interrupt Vector Register
END	RM	End
END IE	B	END Received Interrupt Enable Bit
END RX	B	END Received Bit
EOI	B	END Or Identify Bit
EOI	SL	End Or Identify
EOS	RM	End Of String
EOSR	R	End Of String Register
EOS[7-0]	B	End Of String Bits 7 through 0
ERR	B	Error Bit
ERR	RM	Error
ERR IE	B	Error Interrupt Enable Bit

Mnemonic	Type	Full Name
G		
GCR	R	General Control Register
GET	RM	Group Execute Trigger
GSR	R	GPIB Status Register
GTL	RM	Go To Local
gts	LM	Go To Standby
H		
HLDA	B	Holdoff on All
HLDE	B	Holdoff on END
HLT	B	Halt Bit
I		
ICR	R	Internal Counter Register
IDY	RM	Identify
IFC	RM	Interface Clear
IMR1	R	Interrupt Mask Register 1
IMR2	R	Interrupt Mask Register 2
INT	B	Interrupt Bit
INTRQ	B	Interrupt Request Bits 7 through 5
INV	B	Invert Bit
ISR1	R	Interrupt Status Register 1
ISR2	R	Interrupt Status Register 2
ISS	B	Individual Status Select Bit
ist		Individual Status
L		
L	F	Listen
LA	B	Listener Active Bit
LACS	ST	Listener Active State (L function)
LADS	ST	Listener Addressed State (L function)
LAG	RM	Listen Address Group
LE	F	Extended Listen
LIDS	ST	Listener Idle State
LLO	RM	Local Lockout
LMR	B	Local Master Reset Bit
LOCS	ST	Local State
LOK	B	Lockout Bit
LOKC	B	Lockout Change Bit
LOKC IE	B	Lockout Change Interrupt Enable Bit
lon	B	Listen Only Bit
lon	LM	Listen Only
LPAS	B	Listener Primary Addressed State Bit

Mnemonic	Type	Full Name
LPAS	ST	Listener Primary Addressed State
lpe	LM	Local Poll Enabled
LPIS	ST	Listener Primary Idle State
ltn	LM	Listen
lun	LM	Local Unlisten
LWLS	ST	Local With Lockout State

M

M0	B	Program/Data Access Bit
M1	B	Standard/Short Addressing Bit
M2	B	Supervisor/User Access Bit
MAC	B	Memory Address Count Bits 3 through 2
MAR	R	Memory Address Register
MFCR	R	Memory Function Code Register
MJMN	B	Major-Minor Bit
MLA	RM	My Listen Address
MSA	RM	My Secondary Address
MTA	RM	My Talk Address
MTCR	R	Memory Transfer Counter Register

N

nba	LM	New Byte Available
NDT	B	Normal Device Termination Bit
NIV	R	Normal Interrupt Vector Register
NPRS	ST	Negative Poll Response State
NUL	RM	Null byte

O

OCR	R	Operation Control Register
OSA	RM	Other Secondary Address
OTA	RM	Other Talk Address

P

P[3-1]	B	Parallel Poll Response Bits 3 through 1
PACS	ST	Parallel Poll Addressed to Configure State
PCL	B	Peripheral Control Line Bits 1 through 0
PCG	RM	Primary Command Group
PCS	B	Peripheral Control Status Bit
PCT	B	Peripheral Control Transition Bit
PEND	B	Pending
PGREG	R	Page Register
pof	LM	Power Off

Mnemonic	Type	Full Name
pon	LM	Power On
PP		Parallel Poll (scan all status flags)
PPAS	ST	Parallel Poll Active State
PPC	RM	Parallel Poll Configure
PPD	RM	Parallel Poll Disable
PPE	RM	Parallel Poll Enable
PPIS	ST	Parallel Poll Idle State
PPR	R	Parallel Poll Register
PPSS	ST	Parallel Poll Standby Active
PPU	RM	Parallel Poll Unconfigure
PUCS	ST	Parallel Poll Unaddressed to Configure State

R

rdy	LM	Ready For Next Message
REM	B	Remote Bit
REMC	B	Remote Change Bit
REMC IE	B	Remote Change Interrupt Enable Bit
REMS	ST	Remote State
REN	RM	Remote Enable
REOS	B	END on EOS Received Bit
REQG	B	DMA Request Generation Bits 1 through 0
RFD	RM	Ready For Data
RL	F	Remote/Local
ROR	B	Release On Request Bit
rpp	LM	Request Parallel Poll
RQS	RM	Request Service
rsc	LM	Request System Control
rsv	B	Request Service Bit
rsv	LM	Request Service
rtl	LM	Return To Local
RWLS	ST	Remote With Lockout State

S

S	B	Status Bit Polarity (Sense) Bit
S[6-1]	B	Serial Poll Status Bits 6 through 1
S[8]	B	Serial Poll Status Bit 8
SAB	B	Software Abort Bit
SACS	ST	System Control Active State
SC	B	System Controller Bit
SCG		Secondary Command Group
SCR	R	Sequence Control Register
SDC	RM	Selected Device Clear
SDYS	ST	Source Delay State
SFL	B	System Fail Bit
SGNS	ST	Source Generate State
SH	F	Source Handshake
SIAS	ST	System Control Interface Clear Active State

Mnemonic	Type	Full Name
sic	LM	Send Interface Clear
SIDS	ST	Source Idle State
SIIS	ST	System Control Interface Clear Idle State
SINS	ST	System Control Interface Clear Not Active State
SIWS	ST	Source Idle Wait State
SIZE	B	Size Bits 5 through 4
SNAS	ST	System Control Not Active State
SP	F	Serial Poll (scanning flags)
SPAS	ST	Serial Poll Active State (T function)
SPD	RM	Serial Poll Disable
SPE	RM	Serial Poll Enable
SPEOI	B	Send Serial Poll EOI Bit
SPIS	ST	Serial Poll Idle State
SPMR	R	Serial Poll Mode Register
SPMS	B	Serial Poll Mode State Bit
SPMS	ST	Serial Poll Mode State
SPSR	R	Serial Poll Status Register
SR	F	Service Request
SRAS	ST	System Control Remote enable Active State
sre	LM	Send Remote Enable
SRIS	ST	System Control Remote Enable Idle State
SRNS	ST	System Control Remote Enable Not Active State
SRQ	RM	Service Request
SRQI	B	Service Request Input Bit
SRQI IE	B	Service Request Input Interrupt Enable Bit
SRQS	ST	Service Request State
STB	RM	Status Byte
STR	B	Start Bit
STRS	ST	Source Transfer State
SUP	B	Supervisor Bit
SWNS	ST	Source Wait for New Cycle State
SYSFAIL	LM	System Fail

T

T	F	Talk
TA	B	Talker Active Bit
TACS	ST	Talker Active State (T function)
TADS	ST	Talker Addressed State
TAG	RM	Talk Address Group
tca	LM	Talk Control Asynchronously
tcs	LM	Take Control Synchronously
TCT	RM	Take Control
TE	F	Talker Extended
TIDS	ST	Talker Idle State
TLC		Talker/Listener/Controller (GPIB Adapter)
ton	B	Talk Only Bit
ton	LM	Talk Only
TPAS	B	Talker Primary Addressed State Bit
TPAS	ST	Talker Primary Addressed State

Mnemonic	Type	Full Name
TPIS	ST	Talker Primary Idle State
TRI	B	Three-State Timing Bit
TRM[1-0]	B	Transmit/Receive Mode Bits 1 through 0

U

U	B	Unconfigure Bit
UCG	RM	Universal Command Group
UNL	RM	Unlisten command
UNT	RM	Untalk command

X

X	B	Don't Care Bit
XEOS	B	Transmit END with EOS
XRM	B	External Request Mode Bits 7 through 6

Appendix G
Customer Communication

For your convenience, this appendix contains forms to help you gather the information necessary to help us solve technical problems you might have as well as a form you can use to comment on the product documentation. Filling out a copy of the *Technical Support Form* before contacting National Instruments helps us help you better and faster.

National Instruments provides comprehensive technical assistance around the world. In the U.S. and Canada, applications engineers are available Monday through Friday from 8:00 a.m. to 6:00 p.m. (central time). In other countries, contact the nearest branch office. You may fax questions to us at any time.

Technical Support Phone: (512) 795-8248
Technical Support Fax: (512) 794-5678

Branch Offices	**Phone Number**	**Fax Number**
Australia	03 9879 5166	03 9879 6277
Austria	0662 45 79 90 0	0662 45 79 90 19
Belgium	02 757 00 20	02 757 03 11
Canada (Ontario)	905 785 0085	905 785 0086
Canada (Quebec)	514 694 8521	514 694 4399
Denmark	45 76 26 00	45 76 26 02
Finland	90 527 2321	90 502 2930
France	1 48 14 24 24	1 48 14 24 14
Germany	089 741 31 30	089 714 60 35
Hong Kong	2645 3186	2686 8505
Israel	03 5734815	03 5734816
Italy	02 413091	02 41309215
Japan	03 5472 2970	03 5472 2977
Korea	02 596 7456	02 596 7455
Mexico	5 520 2635	5 520 3282
Netherlands	0348 433466	0348 430673
Norway	32 84 84 00	32 84 86 00
Singapore	2265886	2265887
Spain	91 640 0085	91 640 0533
Sweden	08 730 49 70	08 730 43 70
Switzerland	056 200 51 51	056 200 51 55
Taiwan	02 377 1200	02 737 4644
U.K.	01635 523545	01635 523154

Technical Support Form

Photocopy this form and update it each time you make changes to your software or hardware, and use the completed copy of this form as a reference for your current configuration. Completing this form accurately before contacting National Instruments for technical support helps our applications engineers answer your questions more efficiently.

If you are using any National Instruments hardware or software products related to this problem, include the configuration forms from their user manuals. Include additional pages if necessary.

Name _____

Company _____

Address _____

_____

Fax (___)_____ Phone (___)_____

Computer brand _____ Model _____ Processor _____

Operating system _____

Speed _____MHz RAM _____MB Display adapter _____

Mouse _____yes _____no Other adapters installed _____

Hard disk capacity _____MB Brand _____

Instruments used _____

National Instruments hardware product model _____ Revision _____

Configuration _____

National Instruments software product _____ Version _____

Configuration _____

The problem is _____

_____

_____

_____

_____

List any error messages _____

_____

_____

_____

_____

The following steps will reproduce the problem _____

_____

_____

GPIB-1014D Hardware and Software Configuration Form

Record the settings and revisions of your hardware and software on the line to the right of each item. Update this form each time you revise your software or hardware configuration, and use this form as a reference for your current configuration.

National Instruments Products

- NI-488M Software Version Number on Distribution Medium: _____
- National Instruments board installed (GPIB-1014, GPIB-1014D, GPIB-1014P, or GPIB-1014DP): _____
- GPIB-1014 Revision:_____
- Hardware Settings:

	Base I/O Address	Interrupt Request Line	DMA Channel
1st GPIB-1014	_____	_____	_____
2nd GPIB-1014	_____	_____	_____

- Software Settings:

	Base I/O Address	Interrupt Vector Number	DMA Channel
`gpib0`	_____	_____	_____
`gpib1`	_____	_____	_____

Other Products

- Application Programming Language/Version: _____
- Computer Make and Model: _____
- Microprocessor: _____
- Clock Frequency: _____
- Type of Video Board Installed: _____

- Type of other boards installed and their respective hardware settings:

Board Type	Base I/O Address	Interrupt Level	DMA Channel
_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____

Documentation Comment Form

National Instruments encourages you to comment on the documentation supplied with our products. This information helps us provide quality products to meet your needs.

Title: **GPIB-1014D User Manual**

Edition Date: **March 1997**

Part Number: **320140-01**

Please comment on the completeness, clarity, and organization of the manual.

If you find errors in the manual, please record the page numbers and describe the errors.

Thank you for your help.

Name _____

Title _____

Company _____

Address _____

_____

Phone (_____) _____

Mail to: Technical Publications
National Instruments Corporation
6504 Bridge Point Parkway
Austin, TX 78730-5039

Fax to: Technical Publications
National Instruments Corporation
(512) 794-5678

Index

Numbers

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X