

ARTISAN®
TECHNOLOGY GROUP

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

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

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

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

Custom engineering
so your equipment works exactly as you specify.

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

Expert team | Trust guarantee | 100% satisfaction

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

Find the *National Instruments GPIB-1014* at our website: *Click HERE*

GPIB-1014
User Manual

March 1997 Edition

Part Number 320030-01

© Copyright 1985, 1997 National Instruments Corporation.
All Rights Reserved.

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-1014 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.

The media on which you receive National Instruments software are warranted not to fail to execute programming instructions, due to defects in materials and workmanship, for a period of 90 days from date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace software media that do not execute programming instructions if National Instruments receives notice of such defects during the warranty period. National Instruments does not warrant that the operation of the software shall be uninterrupted or error free.

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

Under the copyright laws, this publication may not be reproduced or transmitted in any form, electronic or mechanical, including photocopying, recording, storing in an information retrieval system, or translating, in whole or in part, without the prior written consent of National Instruments Corporation.

Trademarks

NI-488M™ is a trademark of National Instruments Corporation.

Product and company names listed are trademarks or trade names of their respective companies.

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.

FCC/DOC Radio Frequency Interference Compliance

This equipment generates and uses radio frequency energy and, if not installed and used in strict accordance with the instructions in this manual, may cause interference to radio and television reception. This equipment has been tested and found to comply with the following two regulatory agencies:

Federal Communications Commission

This device complies with Part 15 of the Federal Communications Commission (FCC) Rules for a Class A digital device. Operation is subject to the following two conditions:

1. This device may not cause harmful interference in commercial environments.

2. This device must accept any interference received, including interference that may cause undesired operation.

Canadian Department of Communications

This device complies with the limits for radio noise emissions from digital apparatus set out in the Radio Interference Regulations of the Canadian Department of Communications (DOC).

Le présent appareil numérique n'émiet pas de bruits radioélectriques dépassant les limites applicables aux appareils numériques de classe A prescrites dans le réglement sur le brouillage radioélectrique édicté par le ministére des communications du Canada.

Instructions to Users

These regulations are designed to provide reasonable protection against harmful interference from the equipment to radio reception in commercial areas. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user will be required to correct the interference at his own expense.

There is no guarantee that interference will not occur in a particular installation. However, the chances of interference are much less if the equipment is installed and used according to this instruction manual.

If the equipment does cause interference to radio or television reception, which can be determined by turning the equipment on and off, one or more of the following suggestions may reduce or eliminate the problem.

- Operate the equipment and the receiver on different branches of your AC electrical system.

- Move the equipment away from the receiver with which it is interfering.

- Reorient or relocate the receiver's antenna.

- Be sure that the equipment is plugged into a grounded outlet and that the grounding has not been defeated with a cheater plug.

Notice to user: Changes or modifications not expressly approved by National Instruments could void the user's authority to operate the equipment under the FCC Rules.

If necessary, consult National Instruments or an experienced radio/television technician for additional suggestions. The following booklet prepared by the FCC may also be helpful: *How to Identify and Resolve Radio-TV Interference Problems.* This booklet is available from the U.S. Government Printing Office, Washington, DC 20402, Stock Number 004-000-00345-4.

Contents

About This Manual xiii
Organization of This Manual xiii
Conventions Used in This Manual xiv
Related Documentation xiv
Customer Communication xv

Chapter 1
Introduction 1-1
What Your Kit Should Contain 1-3
Optional Equipment 1-3
Unpacking 1-4

Chapter 2
General Description 2-1
Electrical Characteristics 2-1
VMEbus Characteristics 2-2
VMEbus Slave-Addressing 2-2
VMEbus Slave-Data 2-3
VMEbus Master-Direct Memory Access 2-5
Interrupter 2-6
Data Transfer Bus (DTB) Requester 2-7
VMEbus Modules Not Provided 2-7
Diagnostic Aids 2-7
Data Transfer Features 2-7
Programmed I/O Transfers 2-8
GPIB-1014 Functional Description 2-8

Chapter 3
Configuration and Installation 3-1
Configuration 3-1
Access Mode 3-3
Base Address 3-3
Set Base Address Using Jumper Block W1 3-4
Set Base Address Using Compare Address Lines 3-4
DMA Address Modifier Code Output 3-5
Other Configuration Parameters 3-7
Installation 3-7
Verification of System Compatibility 3-7
Cabling 3-10
Verification Testing 3-10

Chapter 4
Register Bit Descriptions 4-1
Register Map 4-1
Register Sizes 4-2

Register Description 4-2
Register Description Format 4-3
Interface Registers 4-3
Data In Register (DIR) 4-6
Command/Data Out Register (CDOR) 4-7
Interrupt Status Register 1 (ISR1) 4-8
Interrupt Mask Register 1 (IMR1) 4-8
Interrupt Status Register 2 (ISR2) 4-14
Interrupt Mask Register 2 (IMR2) 4-14
Serial Poll Status Register (SPSR) 4-19
Serial Poll Mode Register (SPMR) 4-19
Address Status Register (ADSR) 4-20
Address Mode Register (ADMR) 4-22
Command Pass Through Register (CPTR) 4-25
Auxiliary Mode Register (AUXMR) 4-27
Hidden Registers 4-33
Internal Counter Register (ICR) 4-34
Parallel Poll Register (PPR) 4-35
Auxiliary Register A (AUXRA) 4-37
Auxiliary Register B (AUXRB) 4-39
Auxiliary Register E (AUXRE) 4-41
Address Register 0 (ADR0) 4-42
Address Register (ADR) 4-43
Address Register 1 (ADR1) 4-44
End of String Register (EOSR) 4-45
DMA Registers 4-46
Address Registers 4-48
Transfer Count Registers 4-48
Function Code Registers 4-50
Device Control Register 4-51
Operation Control Register 4-53
Sequence Control Register 4-55
Channel Control Register 4-56
Channel Status Register 4-58
Channel Error Register 4-60
Channel Priority Register 4-61
Interrupt Vector Registers 4-62
General Control Register 4-63
Configuration Registers 4-64
Configuration Register 1 (CFG1) 4-64
Configuration Register 2 (CFG2) 4-66

Chapter 5
Programming Considerations 5-1
Initialization 5-1
The GPIB-1014 as GPIB Controller 5-3
Becoming Controller-In-Charge (CIC) and Active Controller 5-3
Sending Remote Multiline Messages (Commands) 5-4
Going from Active to Standby Controller 5-4

Going from Standby to Active Controller 5-4
Going from Active to Idle Controller 5-5
The GPIB-1014 as GPIB Talker and Listener 5-6
Programmed Implementation of Talker and Listener 5-6
Addressed Implementation of the Talker and Listener 5-6
Address Mode 1 5-6
Address Mode 2 5-6
Address Mode 3 5-7
Sending/Receiving Messages 5-8
Using Direct Memory Access 5-8
DMA Transfers without the Carry Cycle 5-10
DMA Transfers with the Carry Cycle 5-13
Polling During DMAs 5-17
Sending END or EOS 5-17
Terminating the Transfer and Checking the Result 5-17
Terminating on END or EOS 5-19
Using Programmed I/O 5-19
Sending and Receiving Data 5-19
Sending END or EOS 5-20
Terminating on END or EOS 5-20
Interrupts 5-20
Serial Polls 5-22
Conducting a Serial Poll 5-22
Responding to a Serial Poll 5-22
Parallel Polls 5-22
Conducting a Parallel Poll 5-23
Responding to a Parallel Poll 5-24

Chapter 6
Theory of Operation 6-1
VMEbus Interface 6-1
Data Lines 6-1
Slave Read and Write Transfers 6-1
DMA Transfers 6-1
Control Signals 6-2
Address 6-2
Control Equations of Transceivers 6-3
Address Decoding 6-3
Clock and Reset Circuitry 6-4
Configuration Registers 6-5
Configuration Register 1 6-5
Configuration Register 2 6-6
Timing State Machine 6-6
Slave Cycles 6-6
DMA Cycles 6-7
DMA Gating and Control 6-7
Interrupter 6-9
DTB Requester and Controller 6-9
GPIB Synchronization and Interrupt Control 6-12

68450 DMAC 6-14
DMAC Channel Operation 6-15
Initialization and Transfer Phases 6-15
Device (TLC)/DMAC Communication 6-15
DMA Requests 6-16
Data Transfers 6-16
Operands and Addressing 6-17
Address Register Operation 6-17
Transfer Count Register Operation 6-17
Initiation and Control of Channel Operation 6-18
Initiating the Operation 6-18
The Continue Mode of Operation 6-18
Halt 6-18
Software Abort 6-18
Interrupt Enable 6-19
Block Termination 6-19
Multiple Block Operations 6-19
Continued Operations 6-19
Array Chaining Operations 6-19
Linked Chaining Operations 6-20
Error Conditions 6-21
GPIB Interface 6-23
Test and Troubleshooting 6-24
DMA Stand-Alone Testing 6-24
GPIB Interface Testing 6-24

Chapter 7
Diagnostic and Troubleshooting Test Procedures 7-1
Interpreting Test Procedures 7-1
GPIB-1014 Hardware Installation Tests 7-2

Appendix A
Hardware Specifications A-1

Appendix B
Parts List and Schematic Diagrams B-1

Appendix C
Sample Programs C-1

Appendix D
Multiline Interface Messages D-1

Appendix E
Operation of the GPIB E-1
Types of Messages E-1
Talkers, Listeners, and Controllers E-1
The Controller-In-Charge and System Controller E-2
GPIB Signals and Lines E-2

Data Lines E-2
Handshake Lines E-2
NRFD (not ready for data) E-2
NDAC (not data accepted) E-3
DAV (data valid) E-3
Interface Management Lines E-3
ATN (attention) E-3
IFC (interface clear) E-3
REN (remote enable) E-3
SRQ (service request) E-3
EOI (end or identify) E-3
Physical and Electrical Characteristics E-4
Configuration Requirements E-6
Related Documents E-7

Appendix F
Mnemonics Key F-1

Appendix G
Customer Communication G-1

Glossary Glossary-1

Index Index-1

Figures

Figure 1-1. GPIB-1014 Interface Board 1-2

Figure 2-1. GPIB-1014 with a VMEbus Computer 2-9
Figure 2-2. GPIB-1014 in a Multiprocessor Application 2-10
Figure 2-3. GPIB-1014 Block Diagram 2-11

Figure 3-1. Parts Locator Diagram 3-2
Figure 3-2. Access Mode After RESET 3-3
Figure 3-3. Configuration for GPIB-1014 Base Address 2000 (hex) 3-4
Figure 3-4. Default Settings of AM Code Jumpers W3, W4, and W5 3-5

Figure 4-1. Interface Registers 4-4
Figure 4-2. Writing to the Hidden Registers 4-5
Figure 4-3. DMA Register Memory Map 4-47

Figure 5-1. DMA Transfer without Carry Cycle 5-10
Figure 5-2. DMA Transfer with Carry Cycle 5-13

Figure 6-1. DTB Requester and Controller Flip-Flop Operations 6-10
Figure 6-2. Array Format for Array Chaining Modes 6-20

Figure 6-3. Array Format for Linked Chaining Modes 6-21

Figure E-1. The GPIB Connector and Signal Assignments E-4
Figure E-2. Linear Configuration E-5
Figure E-3. Star Configuration E-6

Tables

Table 2-1. GPIB-1014 Signals 2-1
Table 2-2. µPD7210 Internal GPIB Interface Registers 2-3
Table 2-3. 68450 Internal DMA Registers 2-4
Table 2-4. GPIB-1014 Configuration Registers 2-5
Table 2-5. GPIB-1014 IEEE 488 Interface Capabilities 2-13
Table 2-6. GPIB-1014 IEEE 1014 Interrupter Compliance Levels 2-15

Table 3-1. Programming Values for Default Settings of W3, W4, and W5 3-6
Table 3-2. Setting the Address Modifier Code Bits (AM5-AM0) 3-6
Table 3-3. GPIB-1014 Pin Assignment on VMEbus Connector P1 3-8
Table 3-4. GPIB-1014 Pin Assignment on VMEbus Connector P2 3-9

Table 4-1. GPIB-1014 Register Map 4-1
Table 4-2. Clues to Understanding Mnemonics 4-3
Table 4-3. Multiline GPIB Commands Recognized by the µPD7210 4-25
Table 4-4. Auxiliary Command Summary 4-28
Table 4-5. Auxiliary Commands: Detail Description 4-29
Table 4-6. Examples for Configuring the PPR 4-36
Table 4-7. DMAC DMA Channel Register Set 4-46

Table 6-1. Control Equations of Transceivers 6-3

Table A-1. Electrical Characteristics A-1
Table A-2. Environmental Characteristics A-1
Table A-3. Physical Characteristics A-2

About This Manual

The *GPIB-1014 User Manual* describes the mechanical and electrical aspects of the GPIB-1014, the data transfer features, and contains information concerning its operation and programming.

Organization of This Manual

The *GPIB-1014 User Manual* is organized as follows:

- Chapter 1, *Introduction*, describes the GPIB-1014, lists the contents and optional equipment for your GPIB-1014 kit, and explains how to unpack the GPIB-1014 kit.
- Chapter 2, *General Description*, contains the electrical specifications for the GPIB-1014, the data transfer features, and describes the characteristics of key interface board components.
- Chapter 3, *Configuration and Installation*, describes the steps needed to configure and install the GPIB-1014 hardware.
- Chapter 4, *Register Bit Descriptions*, contains a description of the register map, a list of interface registers, and a description of the DMA registers.
- Chapter 5, *Programming Considerations*, explains the initialization process, sending/receiving messages, and the serial/parallel poll process.
- Chapter 6, *Theory of Operation*, contains a functional overview of the GPIB-1014 board and explains the operation of each functional block making up the GPIB-1014.
- Chapter 7, *Diagnostic and Troubleshooting Test Procedures*, contains test procedures for determining if the GPIB-1014 is installed and operating correctly.
- Appendix A, *Hardware Specifications*, specifies the electrical, environmental, and physical characteristics of the GPIB-1014 board and the condition under which it should be operated.
- Appendix B, *Parts List and Schematic Diagrams*, contains the parts list and schematic diagrams for the GPIB-1014.
- Appendix C, *Sample Programs*, contains listings of routines in 68000 assembly language code that implement the essential elements of the major utility functions.
- Appendix D, *Multiline Interface Messages*, lists the multiline interface messages and describes the mnemonics and messages that correspond to the interface functions. These functions include initializing the bus, addressing and unaddressing devices, and setting device modes for local or remote programming. The multiline interface messages are IEEE 488-defined commands that are sent and received with ATN TRUE.
- Appendix E, *Operation of the GPIB*, describes the operation of the GPIB.

- Appendix F, *Mnemonics Key*, 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.

- Appendix G, *Customer Communication,* contains forms for you to complete to facilitate communication with National Instruments concerning our products.

- The *Glossary* contains an alphabetical list and description of terms used in this manual, including abbreviations, acronyms, metric prefixes, and symbols.

- The *Index* contains an alphabetical list of key terms and topics used in this manual, including the pages where each one can be found.

Conventions Used in This Manual

The following conventions are used to distinguish elements of text throughout this manual:

italic	Italic text denotes emphasis, a cross reference, or an introduction to a key concept.
IEEE 488	IEEE 488 is used throughout this manual to refer to the ANSI/IEEE Standard 488.1-1987, which defines the GPIB.
IEEE 1014	IEEE 1014 is used throughout this manual to refer to the ANSI/IEEE Standard 1014-1987, which defines the GPIB.

Related Documentation

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

- ANSI/IEEE Standard 488.1-1987, *IEEE Standard Digital Interface for Programmable Instrumentation*.

- ANSI/IEEE Standard 1014-1987, *IEEE Standard for a Versatile Backplane Bus: VMEbus*.

- *µPD7210 GPIB-IFC User Manual*, NEC Electronics U.S.A., Inc., One Natick Executive Park, Natick, MA 01760.

- *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)*

- *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

This chapter describes the GPIB-1014, lists the contents and oiptional equipment for your GPIB-1014 kit, and explains how to unpack the GPIB-1014 kit.

The GPIB-1014 is a high-performance IEEE 488 interface for the VMEbus. This interface permits IEEE 488 compatible engineering, scientific, or medical instruments to be controlled from a VMEbus-based computer. The GPIB-1014 has the following features:

- Complete IEEE 488 Talker/Listener/Controller (TLC) capability using the NEC μPD7210 GPIB TLC chip
- DMA transfers
  - Data rates up to 500 kbytes/sec
  - Unlimited data block lengths
  - Full 24-bit addressing
  - GPIB synchronization detection
  - 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
- VME laboratories certified
- IEEE 1014 (VMEbus) standard compliance
- Comprehensive software support

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

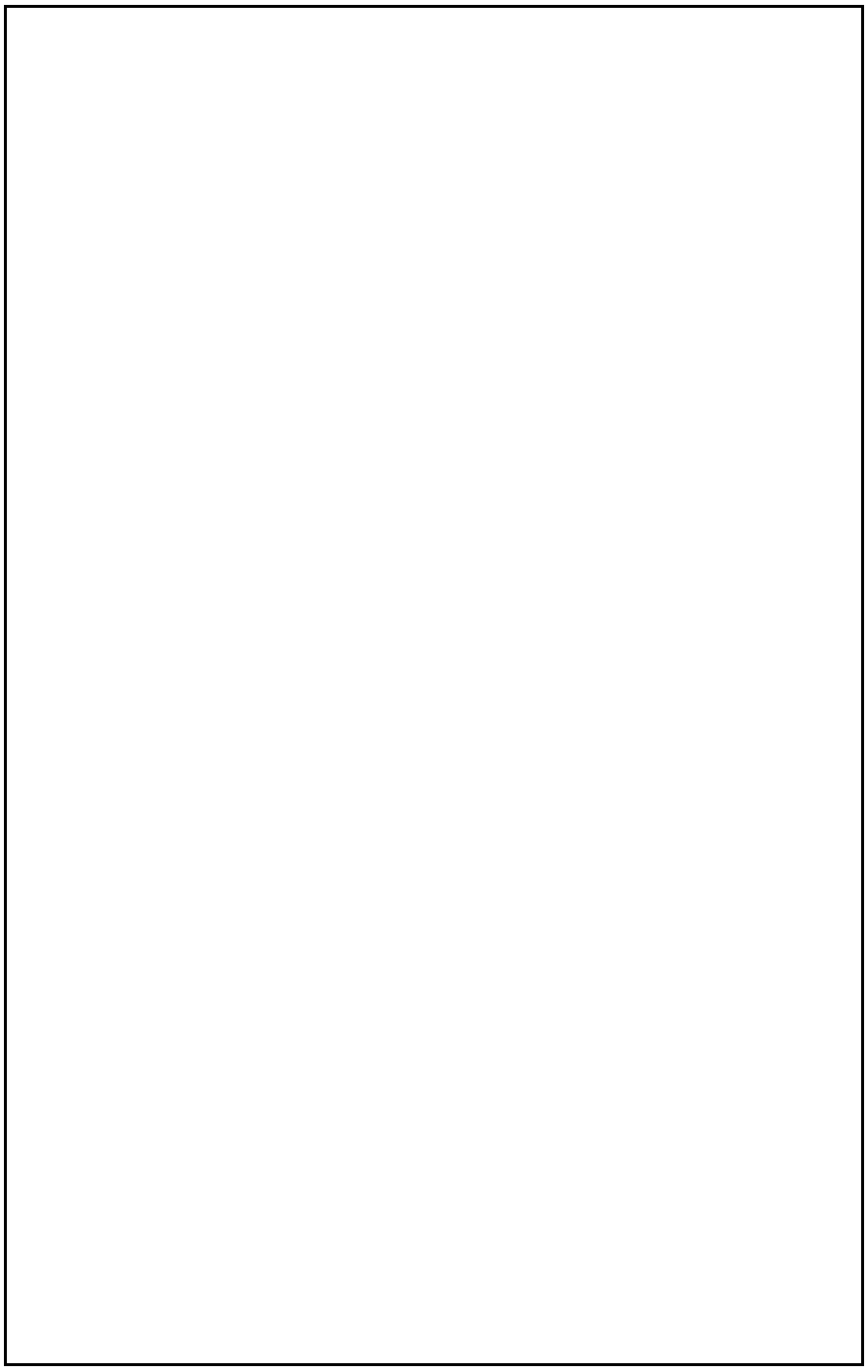


Figure 1-1. GPIB-1014 Interface Board

The GPIB-1014 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-1014 kit should contain the following components:

Kit Component	Part Number
One of these GPIB-1014 boards:	
• GPIB-1014-1	776059-01
• GPIB-1014-2	776060-01
• GPIB-1014-EH (EH=Ejector Handles)	776059-51
• GPIB-1014-1S (no P2 signals)	776059-21
• GPIB-1014-1S-EH (no P2 signals; EH=Ejector Handles)	776059-61
One *GPIB-1014 User Manual*	320030-01

Optional Equipment

Equipment	Part Number
Single-Connector Scrambler Interface Card	180170-01
Dual-Connector Scrambler Interface Card	180170-02
Scrambler Card to P2 Cable Assembly	180173-01
Single-Shielded Cables:	
GPIB Type X1 Cable - 1 m	763001-01
GPIB Type X1 Cable - 2 m	763001-02
GPIB Type X1 Cable - 4 m	763001-03
GPIB Monitor/Analyzer:	
GPIB-400	776074-01
GPIB-410	776104-01

Unpacking

Follow these steps when unpacking your GPIB-1014.

1. Verify that the pieces contained in the package you received match the kit parts list given earlier in this chapter. Do not remove the board from its plastic bag at this point.

2. Your GPIB-1014 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.

3. 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 electrical specifications for the GPIB-1014, the data transfer features, and describes the characteristics of key interface board components.

Electrical Characteristics

All integrated circuit drivers and receivers used on the GPIB-1014 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-1014 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-1014 Signals

Bus Signals	Driver Device Part Number	Receiver Device Part Number
D00-D15	F245	F245
A23-A16	AS573	–
A15-A09	AS573	LS2521
A8	AS573	F1241
A07-A01	F245	F245
AM5-AM3, AM0	F241	LS2521
AM2	F241	LS240
DS0*, DS1*, AS*, WRITE*	F241	F1241
LWORD*	F241	LS2521
IACK*	F241	F1241
SYSCLK	–	LS240
BG0IN*-BG3IN*	–	LS241
BG0OUT*-BG3OUT*	F241	–
DTACK*	AS756	LS240

(continues)

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

Bus Signals	Driver Device Part Number	Receiver Device Part Number
BR0*-BR3*	AS756	LS241
BBSY*	AS756	LS240
IACKIN*	–	LS240
IACKOUT*	F1241	–
IRQ1*-IRQ7*	74145	–
BERR*	–	F1241
SYSFAIL*	AS756	–
SYSRESET*	–	LS240

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-1014 requires regulated +5 VDC power from the VMEbus. Current load is typically 1.6 A (2.0 A maximum).

VMEbus Characteristics

The following paragraphs describe each of the VMEbus modules on the GPIB-1014: 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-1014 occupies 512 bytes (256 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 a short nonprivileged access (AM code = 29). The board responds to short 16-bit addresses. The GPIB-1014 compares address lines A15 through A9 with its base address

to generate its board select signal. It then decodes the lowest eight lines, A8 through A1, to address the following items:

- The 68450 DMA Controller (DMAC)
- The µPD7210 GPIB Talker/Listener/Controller (TLC)
- Two 8-bit, write-only Configuration Registers

You can configure the base address of the board through the hardware jumper set W1 located on the interface board. Except for the models GPIB-1014-1S and GPIB-1014-1S-EH, the base address can also be set using strapped address lines located on the VMEbus P2 connector. See Chapter 3, *Configuration and Installation*, on how to set the base address of the board.

VMEbus Slave-Data

The GPIB-1014 can function as a VMEbus slave, decoding short I/O addresses and commands from a VMEbus master. The µPD7210 and the two Configuration Registers 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 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 D16 & D08(EO).

The GPIB Interface Registers associated with the µPD7210 are addressed relative to the base address of the board, as shown in Table 2-2. The DMA registers internal to the 68450 are shown in Table 2-3. The two Configuration Registers of the GPIB-1014 are shown in Table 2-4.

Table 2-2. µPD7210 Internal GPIB Interface Registers

Address (Base + Hex Offset)	Mode	Register		Size
111	R	Data In	(DIR)	8 bits
111	W	Command/Data Out	(CDOR)	8 bits
113	R	Interrupt Status 1	(ISR1)	8 bits
113	W	Interrupt Mask 1	(IMR1)	8 bits
115	R	Interrupt Status 2	(ISR2)	8 bits
115	W	Interrupt Mask 2	(IMR2)	8 bits
117	R	Serial Poll Status	(SPSR)	8 bits
117	W	Serial Poll Mode	(SPMR)	8 bits
119	R	Address Status	(ADSR)	8 bits
119	W	Address Mode	(ADMR)	8 bits
11B	R	Command Pass Through	(CPTR)	8 bits
11B	W	Auxiliary Mode	(AUXMR)	8 bits
11D	R	Address 0	(ADR0)	8 bits
11D	W	Address	(ADR)	8 bits
11F	R	Address 1	(ADR1)	8 bits
11F	W	End of String	(EOSR)	8 bits

Table 2-3. 68450 Internal DMA Registers

Address (Base + Hex Offset)	**Mode**	**Register**		**Channel**	**Size**
0A	R/W	Memory Transfer Counter	(MTCR0)	0	16 bits
0C	R/W	Memory Address Register	(MAR0)	0	32 bits
29	R/W	Memory Function Code	(MFCR0)	0	8 bits
14	R/W	Device Address Register	(DAR0)	0	32 bits
31	R/W	Device Function Code	(DFCR0)	0	8 bits
1A	R/W	Base Transfer Counter	(BTCR0)	0	16 bits
1C	R/W	Base Address Register	(BAR0)	0	32 bits
39	R/W	Base Function Code	(BFCR0)	0	8 bits
00	R/W	Channel Status	(CSR0)	0	8 bits
01	R	Channel Error	(CER0)	0	8 bits
04	R/W	Device Control	(DCR0)	0	8 bits
05	R/W	Operation Control	(OCR0)	0	8 bits
06	R/W	Sequence Control	(SCR0)	0	8 bits
07	R/W	Channel Control	(CCR0)	0	8 bits
2D	R/W	Channel Priority	(CPR0)	0	8 bits
25	R/W	Normal Interrupt Vector	(NIVR0)	0	8 bits
27	R/W	Error Interrupt Vector	(EIVR0)	0	8 bits
4A	R/W	Memory Transfer Counter	(MTCR1)	1	16 bits
4C	R/W	Memory Address Register	(MAR1)	1	32 bits
69	R/W	Memory Function Code	(MFCR1)	1	8 bits
54	R/W	Device Address Register	(DAR1)	1	32 bits
71	R/W	Device Function Code	(DFCR1)	1	8 bits
5A	R/W	Base Transfer Counter	(BTCR1)	1	16 bit
5C	R/W	Base Address Register	(BAR1)	1	32 bits
79	R/W	Base Function Code	(BFCR1)	1	8 bits
40	R/W	Channel Status	(CSR1)	1	8 bits
41	R	Channel Error	(CER1)	1	8 bits
44	R/W	Device Control	(DCR1)	1	8 bits
45	R/W	Operation Control	(OCR1)	1	8 bits
46	R/W	Sequence Control	(SCR1)	1	8 bits
47	R/W	Channel Control	(CCR1)	1	8 bits
6D	R/W	Channel Priority	(CPR1)	1	8 bits
65	R/W	Normal Interrupt Vector	(NIVR1)	1	8 bits
67	R/W	Error Interrupt Vector	(EIVR1)	1	8 bits
8A	R/W	Memory Transfer Counter	(MTCR2)	2	16 bits
8C	R/W	Memory Address Register	(MAR2)	2	32 bits
A9	R/W	Memory Function Code	(MFCR2)	2	8 bits
94	R/W	Device Address Register	(DAR2)	2	32 bits
B1	R/W	Device Function Code	(DFCR2)	2	8 bits
9A	R/W	Base Transfer Counter	(BTCR2)	2	16 bits
9C	R/W	Base Address Register	(BAR2)	2	32 bits
B9	R/W	Base Function Code	(BFCR2)	2	8 bits
80	R/W	Channel Status	(CSR2)	2	8 bits
81	R	Channel Error	(CER2)	2	8 bits
84	R/W	Device Control	(DCR2)	2	8 bits

(continues)

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

Address (Base + Hex Offset)	Mode	Register		Channel	Size
85	R/W	Operation Control	(OCR2)	2	8 bits
86	R/W	Sequence Control	(SCR2)	2	8 bits
87	R/W	Channel Control	(CCR2)	2	8 bits
AD	R/W	Channel Priority	(CPR2)	2	8 bits
A5	R/W	Normal Interrupt Vector	(NIVR2)	2	8 bits
A7	R/W	Error Interrupt Vector	(EIVR2)	2	8 bits
CA	R/W	Memory Transfer Counter	(MTCR3)	3	16 bits
CC	R/W	Memory Address Register	(MAR3)	3	32 bits
E9	R/W	Memory Function Code	(MFCR3)	3	8 bits
D4	R/W	Device Address Register	(DAR3)	3	32 bits
F1	R/W	Device Function Code	(DFCR3)	3	8 bits
DA	R/W	Base Transfer Counter	(BTCR3)	3	16 bits
DC	R/W	Base Address Register	(BAR3)	3	32 bits
F9	R/W	Base Function Code	(BFCR3)	3	8 bits
C0	R/W	Channel Status	(CSR3)	3	8 bits
C1	R	Channel Error	(CER3)	3	8 bits
C4	R/W	Device Control	(DCR3)	3	8 bits
C5	R/W	Operation Control	(OCR3)	3	8 bits
C6	R/W	Sequence Control	(SCR3)	3	8 bits
C7	R/W	Channel Control	(CCR3)	3	8 bits
ED	R/W	Channel Priority	(CPR3)	3	8 bits
E5	R/W	Normal Interrupt Vector	(NIVR3)	3	8 bits
E7	R/W	Error Interrupt Vector	(EIVR3)	3	8 bits
FF	R/W	General Control Register	(GCR)	all	8 bits

Table 2-4. GPIB-1014 Configuration Registers

Address (Base + Hex Offset)	Mode	Register	Size
101	W	Configuration Register 1	8 bits
105	W	Configuration Register 2	8 bits

VMEbus Master-Direct Memory Access

The GPIB-1014 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 the packing of data in VMEbus memory. In addition to GPIB-to-VMEbus memory DMA transfers, the board can also perform 8- or 16-bit memory-to-memory DMA transfers.

Memory addresses generated by the GPIB-1014 are 24 bits wide and the VMEbus Address Modifier Lines (AM5 through AM0) are fully programmable using function code registers located in the 68450 and three hardware jumpers (W3, W4, and W5). (See Chapter 3 for instructions on setting the hardware jumpers. See Chapter 4 for a description of the DMAC Function Code Registers.) The 24-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-1014 has A24 / D08(EO) & D16 master capability. The board does not use Unaligned Transfer (UAT), Block Transfer (BLT), or 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.

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-1014 during an interrupt acknowledge cycle of the correct priority. The GPIB-1014 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, because it releases its interrupt line when the Channel Status Register is written with the proper value. In VMEbus terminology, the GPIB-1014 has D08(O) / RORA Interrupter capability.

Data Transfer Bus (DTB) Requester

The GPIB-1014 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. To maximize the capabilities of the DTB, the board can be programmed to become a Release On Request (ROR) DTB master. Unless programmed, the GPIB-1014 is a Release When Done (RWD) master.

VMEbus Modules Not Provided

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

Data Transfer Features

The GPIB-1014 can be used to transfer data to and from the GPIB using Direct Memory Access (DMA) and programmed I/O. The overall throughput is dependent upon the following parameters:

- The number of GPIB commands sent
- The amount of time spent setting up the GPIB-1014 DMA transfers
- The size of the DMA data buffers (number of bytes transferred in one DMA operation)
- Operating system overhead
- Interrupt service time
- VMEbus memory response time (DS* low to DTACK* low)

- GPIB Listener response time (DAV* low to NDAC* high)
- GPIB Talker response time (NRFD* high to DAV* low)
- GPIB-1014 transfer mode: Cycle Steal with hold, programmable timeout
- T1 timing: high-speed

Transfer rates of 250 to 350 kbytes/sec can be expected in typical systems, and rates up to 500 kbytes/sec can be achieved under optimum conditions.

Programmed I/O Transfers

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

GPIB-1014 Functional Description

In the simplest terms, the GPIB-1014 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 terminology, the GPIB-1014 implements GPIB interface functions for communicating with other GPIB devices and device functions for communicating with the central processor and memory. Expressed in VMEbus terminology, the GPIB-1014 is an interface to the outside world.

Figures 2-1 and 2-2 show typical applications for the GPIB-1014. In Figure 2-1, the GPIB-1014 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-1014 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.

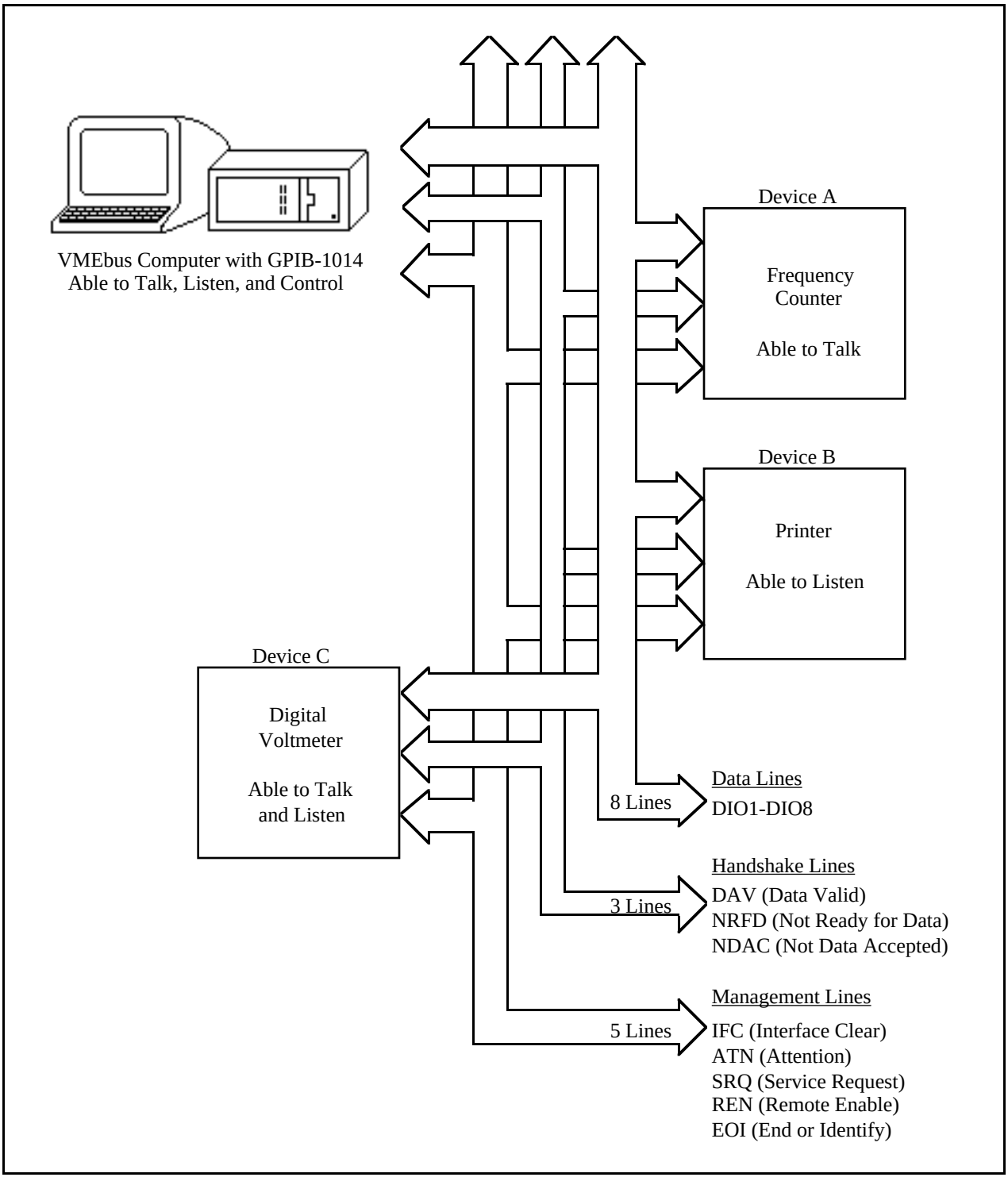


Figure 2-1. GPIB-1014 with a VMEbus Computer

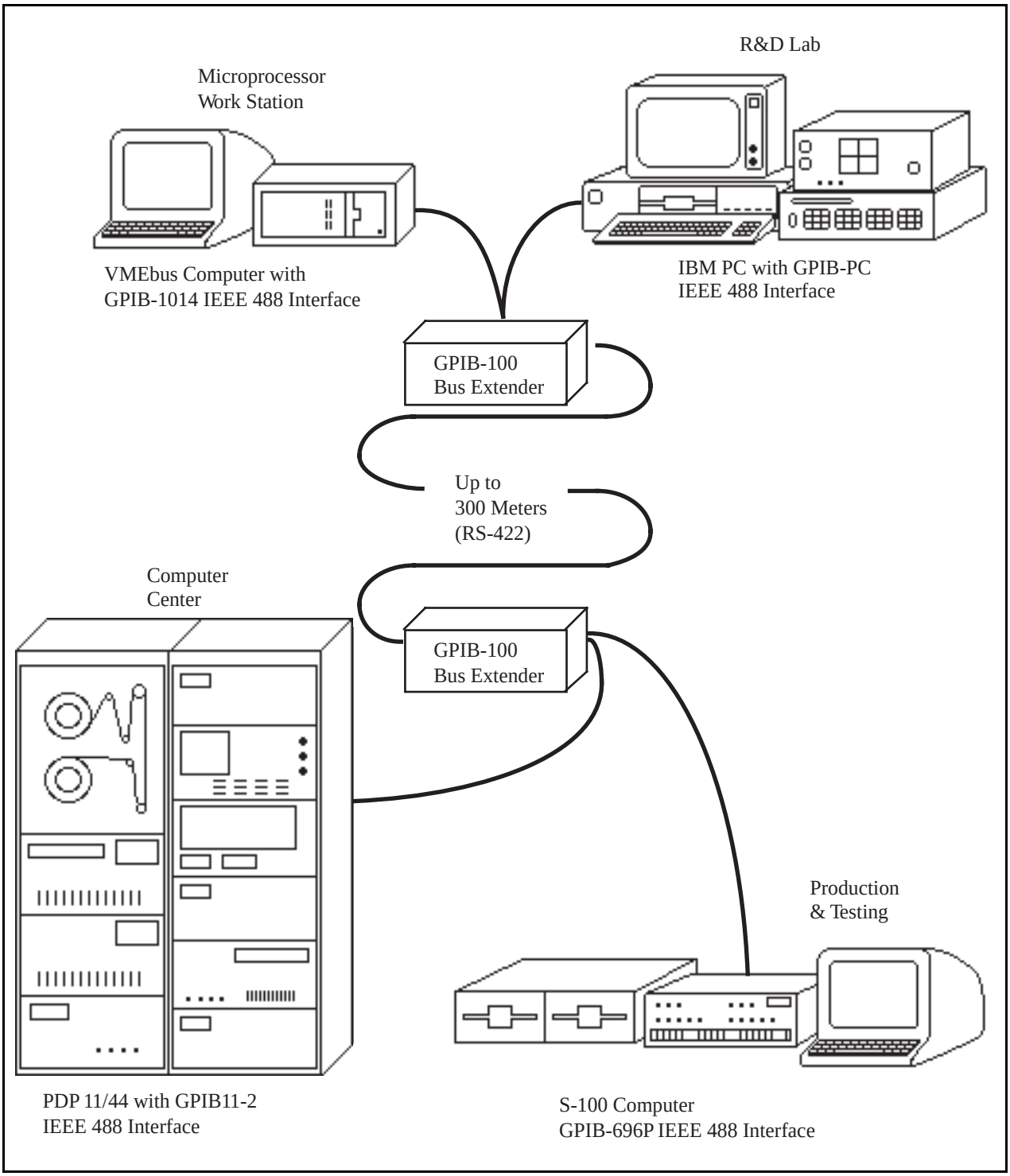


Figure 2-2. GPIB-1014 in a Multiprocessor Application

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

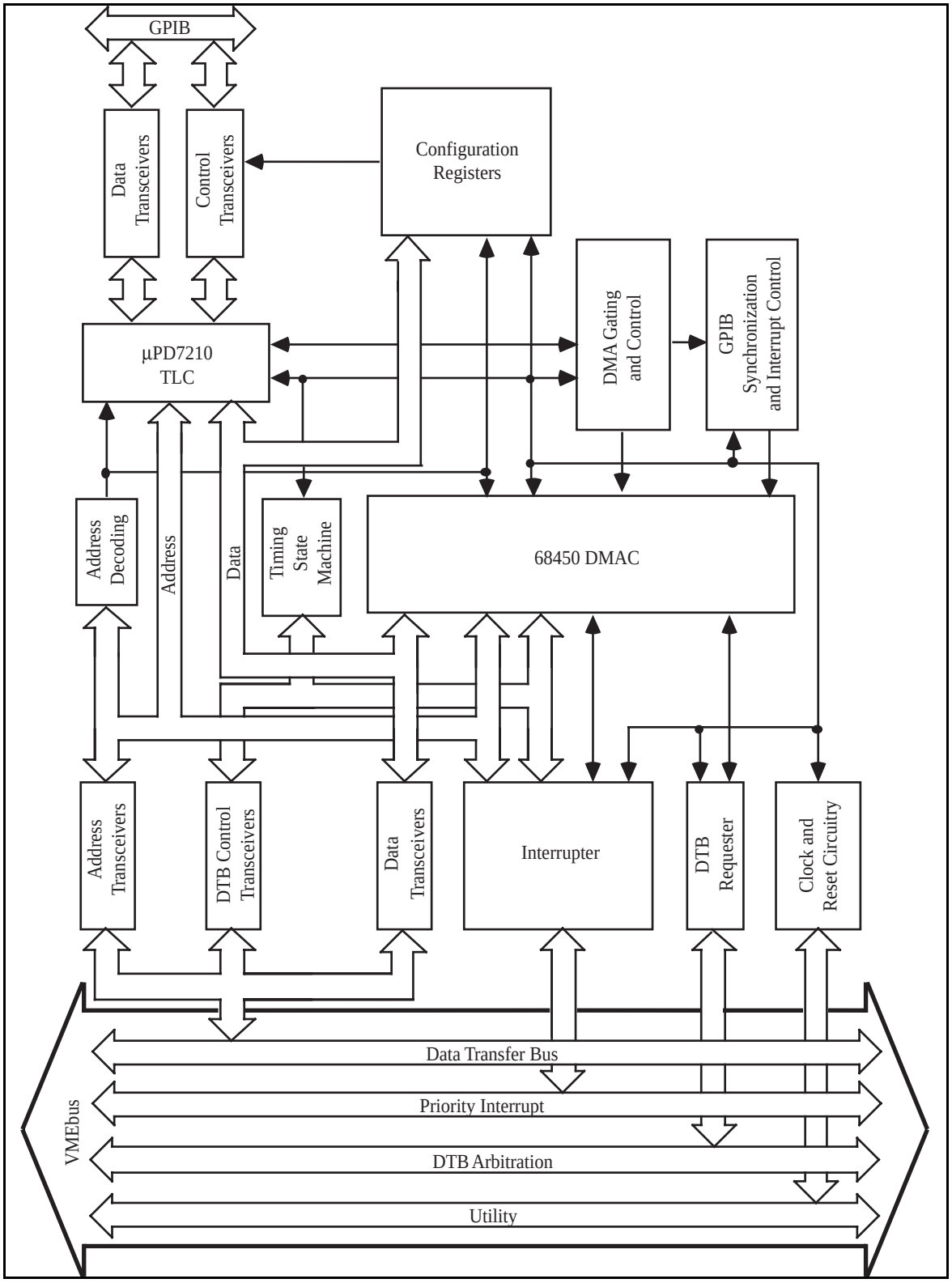


Figure 2-3. GPIB-1014 Block Diagram

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, plus other logic circuitry that converts internal signals to bus-compatible signals.
- Address Decoder — Recognizes when the VMEbus master addresses one of the GPIB-1014 registers and generates the appropriate strobe to begin the data transfer.
- Clock and Reset Circuitry — Monitors the VMEbus utility signals to generate the 8-MHz clock used by the TLC and DMAC and to detect System Reset, Power Failure, and Bus Error conditions.
- Configuration Registers — Programmably configures some of the operating parameters of the GPIB-1014.
- Timing State Machine — Controls the timing of DMA transfers and accesses to the GPIB-1014 from the VMEbus.
- DMA Gating and Control — Controls the DMA request/acknowledge interface between the DMAC and the TLC.
- Interrupter — Implements the VMEbus priority interrupt protocol, allowing the GPIB-1014 to request and respond to an interrupt acknowledge cycle. All interrupt conditions are also detectable by polling.
- DTB Control Transceivers — Performs the necessary VMEbus protocol to request, obtain, and release control of the VME system bus. Once configured for a DMA transfer, the GPIB-1014 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 the Interrupter 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-1014.

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

Table 2-5. GPIB-1014 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
TE5	Complete Extended Talker capability (continued) 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	Complete Extended Listener capability Basic Listener Listen Only mode Unaddressed on MSA*TPAS 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

(continues)

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

Capability Code	Description
DC1	Complete Device Clear capability with software interpretation
DT1	Complete Device Trigger capability with software interpretation
C1, C2, C3, C4, 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-1014 has complete Source and Acceptor Handshake capability.

- The GPIB-1014 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-1014 has full capabilities for requesting service from another Controller. The ability to place the GPIB-1014 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 indicated by the IEEE 488 standard, are included in the GPIB-1014. These include the capability to do the following functions:

- 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-1014 IEEE 1014 compliance levels.

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

Compliance Notation	Description
Bus Slave Compliance Levels	
D8(O)	8-bit data path to TLC and two Configuration Registers
D16 & 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-D07
RORA	Releases its interrupt request line when an onboard register is accessed
Bus Master Compliance Levels	
D8(EO)	8-bit data path for GPIB DMA transfers
D16 & D8(EO)	8- or 16-bit memory-to-memory DMA transfers
A24	24-bit memory address path
DTB Requester Compliance Level	
ROR	Programmable Release on Request Feature

Chapter 3
Configuration and Installation

This chapter describes the steps needed to configure and install the GPIB-1014 hardware.

Configuration

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

- Access Mode (jumper W2)
- Base Address (jumper block W1)
- DMA Address Modifier (AM) Code Output (jumpers W3, W4, and W5)

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

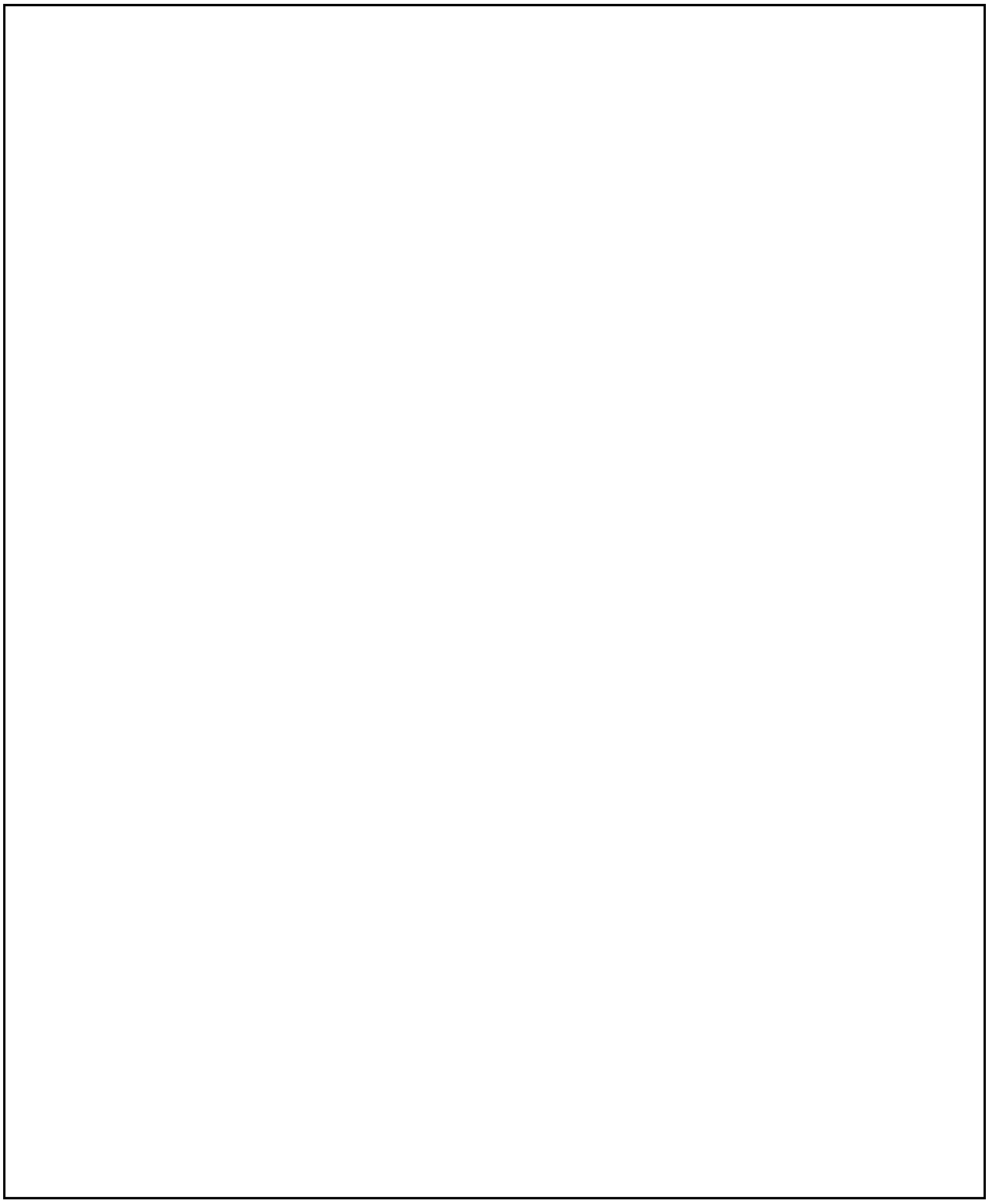


Figure 3-1. Parts Locator Diagram

Access Mode

The GPIB-1014 can be configured to respond to Supervisor (privileged) or User (non-privileged) access. Hardware jumper W2 is used to select the access mode that is automatically in effect upon a power-up or a system reset. The access mode then can 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 jumper to the side labeled *S* for Supervisor mode, or to the side labeled *U* for User mode. Supervisor mode is the default setting configured at the factory. If the board is configured for Supervisor mode, it will initially respond 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 AM codes of 29 and 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.)

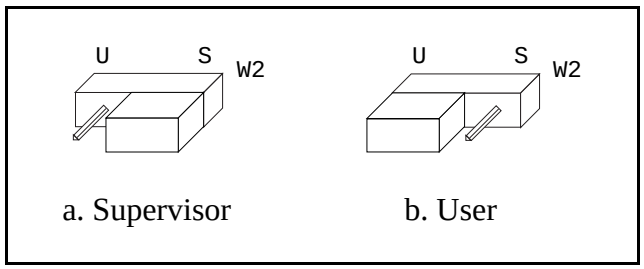


Figure 3-2. Access Mode After RESET

Base Address

The GPIB-1014 occupies a total of 512 bytes of 16-bit I/O space. The base address is selected with either hardware jumper block W1 on the interface board or *compare address* lines located on the P2 connector. The 1S and 1S-EH versions of the GPIB-1014 do not have compare address lines located on the P2 connector.

Note: If the base address is configured using the onboard jumpers, the lines must not be strapped on the P2 connector. Likewise, if the base address is configured on the P2 connector, the jumpers provided on the interface board must be removed.

Set Base Address Using Jumper Block W1

Move the jumper to the side labeled *1* to select a logical one for the corresponding address bit, or to the side labeled *0* to select a logical zero. Figure 3-3 shows the configuration for a base address 2000 (hex), which is the default address configured at the factory.

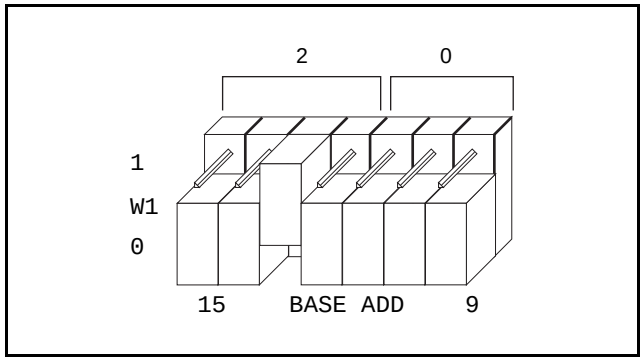


Figure 3-3. Configuration for GPIB-1014 Base Address 2000 (hex) (Default Setting)

Set Base Address Using Compare Address Lines

Another method of setting the base address is to use the compare address lines located on the P2 connector. The jumpers on jumper block W1 must be removed and TTL-compatible voltages must be applied to pins CA9 through CA15 on the P2 connector when using this method. Refer to Table 3-4 for the pin assignment numbers of the P2 connector. A scrambler card and interface cable assembly (described under *Cabling* later in this chapter), available from National Instruments, are necessary when setting the base address of the board using this method.

DMA Address Modifier Code Output

During a DMA cycle, the GPIB-1014 sends out a 6-bit Address Modifier (AM) code to the VMEbus lines AM5 through AM0. The correct code is obtained by both programming the DMAC and setting jumpers W3, W4, and W5. Figure 3-4 shows the default settings of W3, W4, and W5.

Note: Because jumper W5 is not located near jumpers W3 and W4, Figure 3-4 outlines and labels the components on the GPIB-1014 interface board that are found between jumper W5 and the other jumpers.

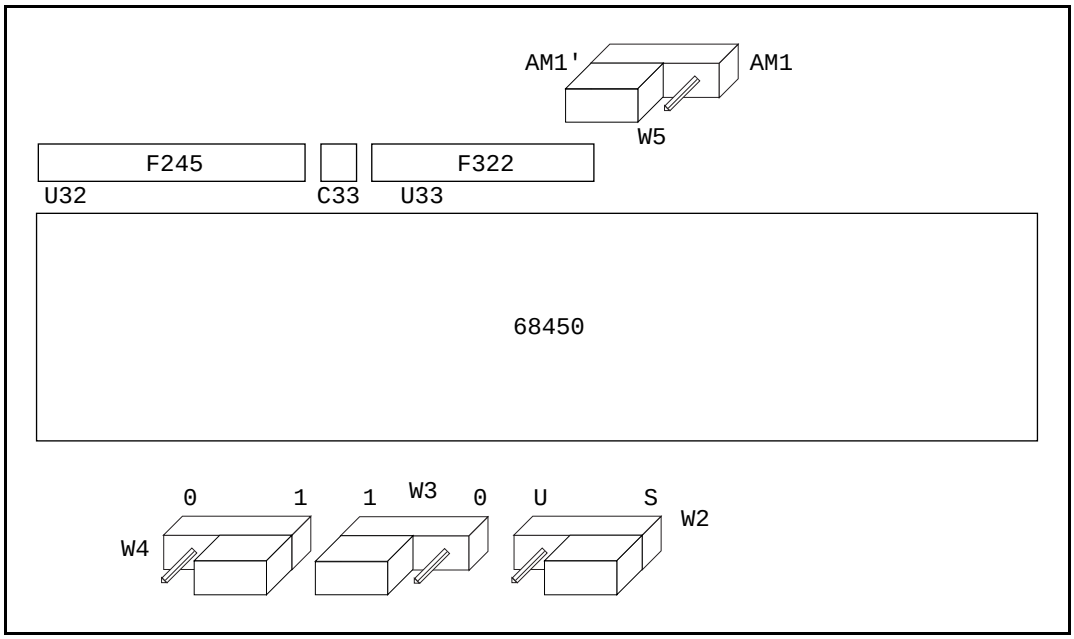


Figure 3-4. Default Settings of AM Code Jumpers W3, W4, and W5

Rev. D and earlier versions of the GPIB-1014 do not have jumpers W3, W4, and W5. If all of the jumpers on later versions of the board remain in their factory default settings, the address modifier codes generated are equivalent to those generated by the earlier versions.

The GPIB-1014 can produce eight AM codes from the default settings of W3, W4, and W5. These eight codes are the most commonly used. Table 3-1 lists these AM 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 W3, W4, and W5

AM Codes	FCR Bits M2 through M0
29	000
2A	001
2D	100
2E	101
39	010
3A	011
3D	110
3E	111

If it is necessary to produce a code other than those listed in Table 3-1, you can produce any arbitrary AM code by changing jumpers W3, W4, and W5 along with programming the DMAC.

Table 3-2 shows how each of the six AM code bits is affected by the jumpers and the values of the DMAC's FCR bits.

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

AM Bits	Controlled By	Value	Default
AM(5)	Jumper W3	0 or 1	1
AM(4)	Bit M1 in FCR	0 or 1	None
AM(3)	Jumper W4	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)	Jumper W5	AM(1) or AM(1)'	AM(1)'

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

1. Set jumper W3 to 0.
2. Set jumper W4 to 0.
3. Set jumper W5 to AM(1).
4. Write the pattern 00000111 to the FCR of the DMAC.

Other Configuration Parameters

All other configuration parameters for the GPIB-1014 are software-selectable, including the following:

- Selecting one of seven interrupt request lines (IRQ1* through IRQ7*)
- Selecting one of four Bus Request lines (BR0* through BR3*)
- Enabling the Release On Request (ROR) feature
- Selecting the automatic carry cycle feature
- Enabling the GPIB-1014 to be the GPIB System Controller
- Selecting the color of front panel SYSFAIL LED indicator (Red/Green)
- Selecting the DMA transfer mode

Software control of these parameters gives you the flexibility to tailor operation and performance to meet the specific requirements of separate tasks within your system.

Installation

The GPIB-1014 is a dual-height board that interfaces to the VMEbus P1 and P2 connectors. There are two options for the GPIB I/O from the GPIB-1014 interface board. The following paragraphs describe the GPIB-1014 interface to the VMEbus backplane and to the IEEE 488 bus.

Verification of System Compatibility

The GPIB-1014 does not use all of the signals 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-1014 will be installed. This is to ensure that the two are compatible (that is, the GPIB-1014 has all the necessary signals needed by the system and vice versa).

Note: For models GPIB-1014-1S and GPIB-1014-1S-EH, there are no signals connected to the P2 connector except for power (+5 V) and ground signals.

Table 3-3. GPIB-1014 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	D10		C19	A19	
C4	D11		C20	A18	
C5	D12		C21	A17	
C6	D13		C22	A16	
C7	D14		C23	A15	
C8	D15		C24	A14	
C9	GND		C25	A13	
C10	SYSFAIL*		C26	A12	
C11	BERR*		C27	A11	
C12	SYSRESET*		C28	A10	
C13	LWORD*		C29	A09	
C14	AM5		C30	A08	
C15	A23		C31		+12V
C16	A22		C32	+5V	

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

Pin No.	Signal Used	Signal Not Used	Notes	Pin No.	Signal Used	Signal Not Used	Notes
A1	CA15		1,4	A17		User I/O	
A2	CA13		1,4	A18		User I/O	
A3	CA11		1,4	A19		User I/O	
A4	CA9		1,4	A20		User I/O	
A5		User I/O		A21	DIO5*		2,4
A6		User I/O		A22	DIO6*		2,4
A7		User I/O		A23	DIO7*		2,4
A8		User I/O		A24	DIO8*		2,4
A9		User I/O		A25	REN*		2,4
A10		User I/O		A26	GND		2,4
A11		User I/O		A27	GND		2,4
A12		User I/O		A28	GND		2,4
A13		User I/O		A29	GND		2,4
A14		User I/O		A30	GND		2,4
A15		User I/O		A31	GND		2,4
A16		User I/O		A32	GND		2,4
B1		+5V		B17		D19	
B2		GND		B18		D20	
B3			3	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	CA14		1,4	C17		User I/O	
C2	CA12		1,4	C18		User I/O	
C3	CA10		1,4	C19		User I/O	
C4		User I/O		C20		User I/O	
C5		User I/O		C21	DIO1*		2,4
C6		User I/O		C22	DIO2*		2,4
C7		User I/O		C23	DIO3*		2,4
C8		User I/O		C24	DIO4*		2,4
C9		User I/O		C25	EOI*		2,4
C10		User I/O		C26	DAV*		2,4
C11		User I/O		C27	NRFD*		2,4
C12		User I/O		C28	NDAC*		2,4
C13		User I/O		C29	IFC*		2,4
C14		User I/O		C30	SRQ*		2,4
C15		User I/O		C31	ATN*		2,4
C16		User I/O		C32		User I/O	

Notes

1. Compare Address lines. Can be used to set base address of GPIB-1014.
2. GPIB signals used for GPIB I/O via P2.
3. Reserved pin. Defined by the VMEbus specification.
4. For models GPIB-1014-1S and GPIB-1014-1S-EH. These signals are not connected to the P2 connector.

Cabling

Two options are available for GPIB I/O from the GPIB-1014:

- A Front Panel Plug-In Connector
- A VMEbus P2 Connector

The Model GPIB-1014-1 interface board has a standard 24-pin IEEE 488 connector on the front panel of the board. A standard GPIB cable can plug in directly to this connector. Two GPIB cables cannot be connected side by side in this configuration; that is, if more than one Model GPIB-1014-1 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-1014-1 interface boards).

The Model GPIB-1014-2 interface board has no connector on the front panel; therefore, GPIB I/O is performed through the VMEbus P2 connector. A scrambler card and interface cable assembly are available from National Instruments for this configuration. The interface cable assembly is a 96-wire flat ribbon cable used to connect the GPIB I/O signals from the backplane P2 connector to the scrambler card, where they are routed to the IEEE 488 connector. A standard GPIB cable enters the chassis through the rear to plug into the IEEE 488 connector on the scrambler card. The scrambler card is equipped with a 96-pin DIN connector and a 24-pin IEEE 488 connector. A dual connector version of the scrambler card is also available for use with two GPIB-1014-2 Interface Cards.

The Models GPIB-1014-EH, GPIB-1014-1S, and GPIB-1014-1S-EH interface boards use the same cable as the Model GPIB-1014-1.

Verification Testing

A performance verification test can be run to ensure 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 jumper or a front-panel emulator, that can load and read memory and I/O addresses.

The tests presented in Chapter 7 of this manual consist of a series of steps written in a pseudo (processor-independent) language with instructions. The steps generally involve writing data to specific GPIB-1014 device registers followed by reading other GPIB-1014 registers to verify that the programming is correct. These tests exercise virtually all of the major functions of the GPIB-1014, including 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 a National Instruments GPIB-400 or GPIB-410) 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 contains a description of the register map, a list of interface registers, and a description of the DMA registers.

Register Map

The register map for the GPIB-1014 is shown in Table 4-1. This table gives the register name, the register address, the register size in bits, and the register type (read only, write only, or read and write).

Note: For the sake of brevity, only Channel 0 addresses for the DMA Register Group are listed in Table 4-1. See Table 2-3 for a complete listing of the addresses for all four channels of the DMA Register Group.

Table 4-1. GPIB-1014 Register Map

Register Name	Address (Hex)	Type	Size
GPIB Interface 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

(continues)

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

Register Name	Address (Hex)	Type	Size
DMA Register Group:			
Address Registers			
Memory Address Register	Base address + 0C	Read/Write	32-bit
Base Address Register	Base address + 1C	Read/Write	32-bit
Device Address Register	Base address + 14	Read/Write	32-bit
Transfer Count Registers			
Memory Transfer Counter	Base address + 0A	Read/Write	16-bit
Base Transfer Counter	Base address + 1A	Read/Write	16-bit
Function Code Registers			
Memory Function Code	Base address + 29	Read/Write	8-bit
Base Function Code	Base address + 39	Read/Write	8-bit
Device Function Code	Base address + 31	Read/Write	8-bit
Device Control Register	Base address + 04	Read/Write	8-bit
Operation Control Register	Base address + 05	Read/Write	8-bit
Sequence Control Register	Base address + 06	Read/Write	8-bit
Channel Control Register	Base address + 07	Read/Write	8-bit
Channel Status Register	Base address + 00	Read/Write	8-bit
Channel Error Register	Base address + 01	Read Only	8-bit
Channel Priority Register	Base address + 2D	Read/Write	8-bit
General Control Register	Base address + FF	Read/Write	8-bit
Interrupt Vector Registers			
Normal Interrupt Vector	Base address + 25	Read/Write	8-bit
Error Interrupt Vector	Base address + 27	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

Register Sizes

VMEbus computers support three transfer sizes for read and write operations: 8-, 16-, or 32-bit. Table 4-1 shows the size of each GPIB-1014 register. For example, reading the Memory Transfer Counter Register requires a 16-bit read operation at the indicated address, whereas writing to the End Of String Register requires an 8-bit write operation at the indicated address.

Register Description

Table 4-1 divides the GPIB-1014 registers into three different register groups. A detailed bit description of each of the registers making up these groups is included later in this chapter.

The GPIB Interface Register Group consists of 21 registers located in the NEC µPD7210 TLC chip. The DMA Register Group consists of 18 registers located in the 68450 DMAC chip. The Configuration Register Group contains two registers used to configure some of the board operating parameters.

Register Description Format

The remainder of this chapter discusses each of the GPIB-1014 registers in the order shown in Table 4-1. Each register group is introduced, followed by a detailed bit description of each register.

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, and should be ignored. If the register is written to, these bit locations should be cleared.

Mnemonics are assigned to messages, states, registers and bits. Most mnemonics contain a clue to their meaning. Table 4-2 contains a list of clues to look for.

Table 4-2. Clues to Understanding Mnemonics

Clue	Mnemonic Probably Stands For:
Ends in IE	Interrupt Enable bit
Ends in EN	Enable bit
4 letters, ends in S	Interface function as defined in the IEEE 488 standard
Ends in R, R0, R1, R2	GPIB Program Register
3 letters, uppercase	Remote GPIB message
3 letters, lowercase	Local GPIB message

Appendix F contains an alphabetical list of mnemonics.

Interface Registers

Twenty-one GPIB Interface registers, eight of which are readable and 13 of which are writable, are located within the NEC μPD7210 Talker/Listener/Controller (TLC) integrated circuit. Each of the 21 interface registers is addressed relative to the GPIB-1014 VMEbus base address.

Figures 4-1 and 4-2 show the register and bit mnemonics of each TLC internal register, its read/write accessibility, and its relative address. Figure 4-1 shows the regular GPIB-1014 GPIB Interface registers. Figure 4-2 shows the hidden GPIB interface registers and illustrates the method of writing to those registers via the Auxiliary Mode Register. Following Figures 4-1 and 4-2 are detailed function descriptions of all 21 interface registers.

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 bit is not used.

Figure 4-1. Interface Registers

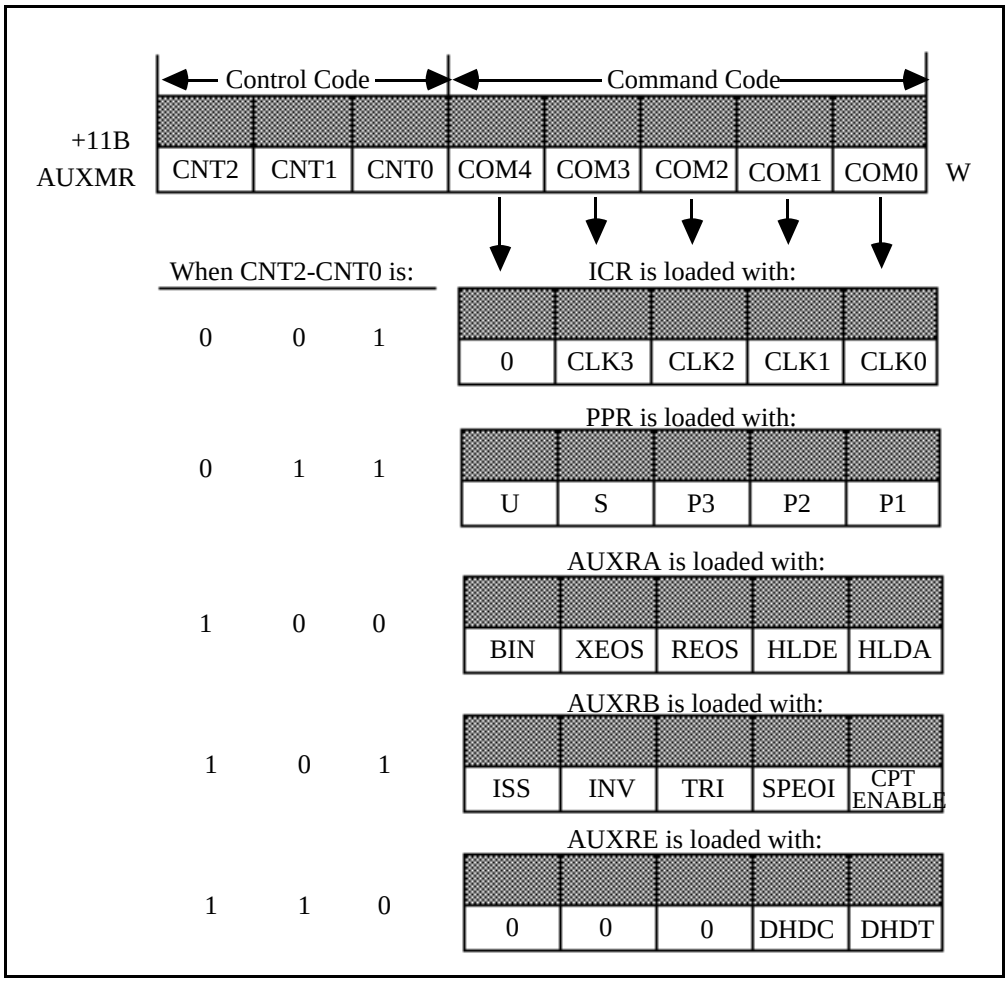
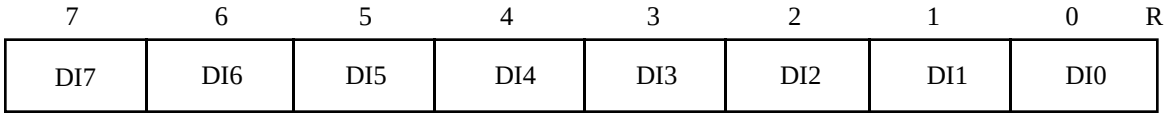


Figure 4-2. Writing to the Hidden Registers

Data In Register (DIR)

VMEbus Address: Base Address + 111 (hex)

Attributes: Read Only, Internal to TLC



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 to the VMEbus memory or by an I/O read from a VMEbus master.

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: Base 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: Base Address + 113 (hex)

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

Interrupt Mask Register 1 (IMR1)

VMEbus Address: Base Address + 113 (hex)

Attributes: Write Only, Internal to TLC

7	6	5	4	3	2	1	0	R
CPT	APT	DET	END RX	DEC	ERR	DO	DI	
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 & (TADS + LADS)] & undefined & ACDS & (CPT ENABLE) + UDPCF & SCG & ACDS & 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 & (TADS + LADS)] & undefined & ACDS & CPT ENABLE

UDPCF is cleared on:

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

The CPT bit flags the occurrence of a GPIB command not recognized by the TLC, and all following GPIB secondary commands 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; 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 & ADM0 & (TPAS + LPAS) & SCG & 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

Bit	Mnemonic	Description

SCG: GPIB Secondary Command Group
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 Device Execute Trigger Bit
5w DET IE 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 End Received Bit
4w END IE End Received Interrupt Enable Bit

END RX is set by:
LACS & (EOI + EOS & REOS) & ACDS
END RX is cleared by:
pon + (read ISR1)
Notes
LACS: GPIB Listener Active State
EOI: GPIB End Of Identify Signal

Bit	Mnemonic	Description
		EOS: GPIB END Of String message 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 3w	DEC 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 2w	ERR ERR IE	Error Bit Error Interrupt Enable Bit ERR is set by: TACS & SDYS & DAC & RFD + SIDS & (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 1w	DO DO IE	Data Out Bit Data Out Interrupt Enable Bit DO is set as: (TACS & 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 0w	DI DI IE	Data In Bit Data In Interrupt Enable Bit Bit DI is set by: LACS & ACDS & continuous mode DI is cleared by: pon + (read ISR1) + (Finish Handshake) & (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: Base Address + 115 (hex)

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

Interrupt Mask Register 2 (IMR2)

VMEbus Address: Base 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 that 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 & CPT IE) + (APT & APT IE) + (DET & DET IE) + ERR & ERR IE) + (END RX & END IE) + (DEC & DEC IE) + (DO & DO IE) + (DI & DI IE) + (SRQI & SRQI IE) + (REMC & REMC IE) + (CO & CO IE) + (LOKC & LOKC IE) + (ADSC & 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 & SRQ & -(RQS & DAV)) becomes true or (CIC & SRQ & RQS & 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 & 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)] & -(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: Base Address + 117 (hex)

Attributes: Read Only, Internal to TLC

Serial Poll Mode Register (SPMR)

VMEbus Address: Base Address + 117 (hex)

Attributes: Write Only, Internal to TLC

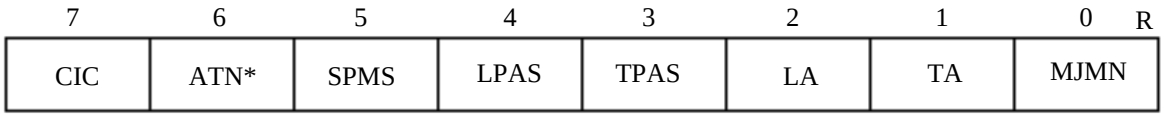
7	6	5	4	3	2	1	0	
S8	PEND	S6	S5	S4	S3	S2	S1	R
S8	rsv	S6	S5	S4	S3	S2	S1	W

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 serial 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 the Negative Poll Response State (NPRS) & 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 serial 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: Base 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 that 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 able 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 represents 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. If the TLC is addressed to talk, it is automatically unaddressed to listen. 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 can 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: Base 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 Bits 1 through 0 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. These are set to configure the µPD7210 to match the transceivers chosen for hardware implementation of the GPIB interface.
3-2w	0	Reserved Bits Write zeros to these bits.

Bit	Mnemonic	Description
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 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.

Bit	Mnemonic	Description
		ADM0 and ADM1 must be cleared when either of the two programmable bits ton or lon is set. For more information on the different addressing modes supported by the GPIB-1014, refer to the *Addressed Implementation of Talker and Listener* section in Chapter 5.

Command Pass Through Register (CPTR)

VMEbus Address: Base 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 4-3. 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 4-3. 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)

Table 4-3. Multiline GPIB Commands Recognized by the µPD7210 (continued)

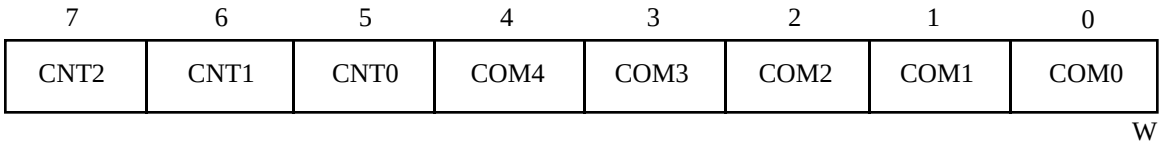
Hex Number	Message	Description
15	PPU	Parallel Poll Unconfigure
18	SPE	Serial Poll Enable
19	SPD	Serial Poll Disable
20-3E	MLA	My Listen Address
3F	UNL	Unlisten
40-5E	MTA	My Talk Address
5F	UNT	Untalk
60-6F	MSA, PPE	My Secondary Address or Parallel Poll Enable
70-7E	MSA, PPD	My Secondary Address or Parallel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

The CPTR is read during a TLC-initiated Parallel Poll operation to retrieve 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: Base Address + 11B (hex)

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



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 4-4 shows the control and command codes used.

Bit	Mnemonic	Description
7-5w	CNT[2-0]	Control Code Bits 2 through 0 These bits indicate 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, 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 indicate the command code of the special function if the control code is 000. Table 4-4 is a summary of the implemented special functions. Table 4-5 explains the details of each special function. If the control code is not 000, these bits are written to one of the hidden registers (indicated by the control code in CNT[2-0]).

Table 4-4. 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 4-5 shows the functions that are executed when the AUXMR Control Code (CNT2 through CNT0) is loaded with 000 (binary) and the Command Code (COM4 through COM0) is loaded.

Table 4-5. 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 4-5. Auxiliary Commands: Detail Description (continued)

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-1014. 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 4-5. Auxiliary Commands: Detail Description (continued)

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.
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.

(continues)

Table 4-5. Auxiliary Commands: Detail Description (continued)

Command Code (COM4-COM0) 4 3 2 1 0	**Description**
1 1 0 1 1 (continued)	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-1014 is SC. In order 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.
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-1014 is SC. In order 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

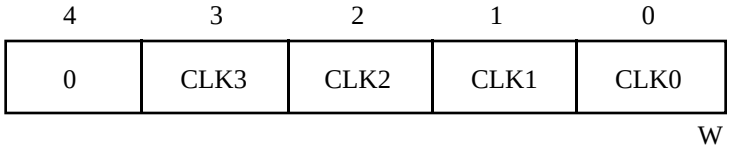
The hidden registers are loaded through the Auxiliary Mode Register (AUXMR). AUXMR[7-5] is loaded with the hidden register number, and AUXMR[4-0] is loaded with the data to be transferred to the hidden register. The hidden registers cannot be read, and in some cases the contents are setable only; that is, they can be cleared or reset to initialized conditions only by issuing the Chip Reset auxiliary command, by a Power On Reset, or by LMR (CR0[2]w). Figure 4-2 earlier in this chapter shows the five hidden registers and illustrates how they are loaded with data from the AUXMR.

Internal Counter Register (ICR)

VMEbus Address: Base Address + 11B (hex)

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

Attributes: Write Only, Internal to TLC
Accessed through AUXMR



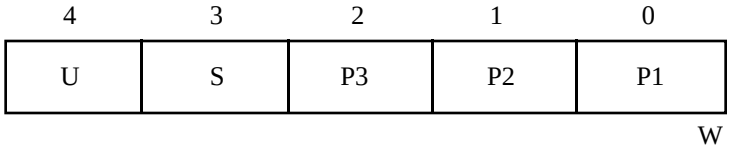
Bit	Mnemonic	Description
4w	0	Reserved Bit Write zero to this bit.
3-0w	CLK[3-0]	Clock Bits 3 through 0 The contents of the ICR are used to divide internal counters that generate TLC state change delay times used by the IEEE 488 specification. The most familiar of these delay times, T1, is the minimum 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. For proper operation, ICR should be set to eight because the TLC is clocked at 8 MHz.

Parallel Poll Register (PPR)

VMEbus Address: Base Address + 11B (hex)

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

Attributes: Write Only, Internal to TLC
Accessed through AUXMR



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. It is only necessary to 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-1014 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-1014 is parallel polled (and S=ist).

Table 4-6 contains some examples of configuring the Parallel Poll Register.

Table 4-6. Examples for Configuring the PPR

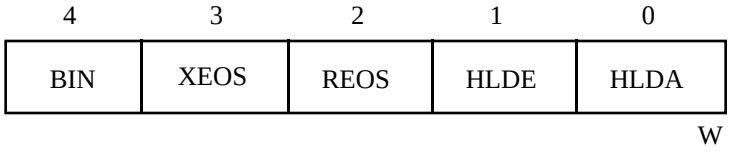
Written to the AUXMR								**Result**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
0	1	1	1	0	0	0	0	Unconfigures PPR
0	1	1	0	0	0	0	0	0 0 0 0 0 is written to the PPR. The GPIB-1014D participates in a Parallel Poll, asserting the DIO1 line if ist=0. Otherwise, the GPIB-1014D does not participate.
0	1	1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

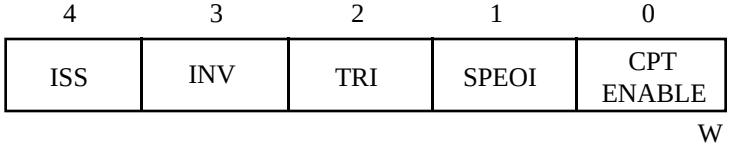
In continuous mode, the rdy message is generated when in ANRS until the end of the data block is detected. A Holdoff is generated at the end of a data block. The FH auxiliary command must be issued to release the Holdoff. The continuous mode is useful for monitoring the data block transfer without actually 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: Base 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 message. 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-1014, 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

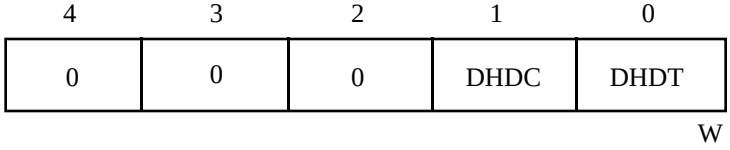
Bit	Mnemonic	Description
2w	TRI	Three-State Timing Bit The TRI bit determines the TLC GPIB Source Handshake Timing, T1. TRI can be set to enable high-speed data transfers (T1 ≥ 500 nsec) when tri-state GPIB drivers are used. (The GPIB-1014D uses tri-state GPIB drivers except during Parallel Poll responses, 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: Base Address + 11B (hex)

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

Attributes: Write Only, Internal to TLC
Accessed through AUXMR



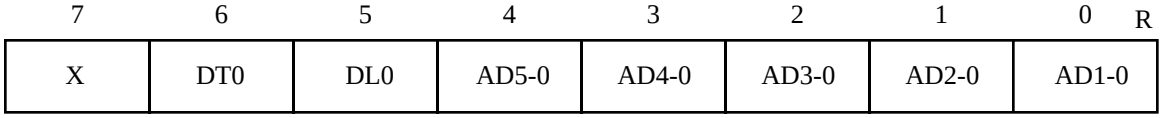
Writing to Auxiliary Register E (AUXRE) is done via the AUXMR. Writing the binary value 110 into the Control Code (CNT[2-0]) and a bit pattern into the the lower five bits of the AUXMR (COM[4-0]) causes the two lowest order bits to be written to AUXRE. The 2-bit code, DHDC and DHDT, determines how the 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: Base Address + 11D (hex)

Attributes: Read Only, Internal to TLC



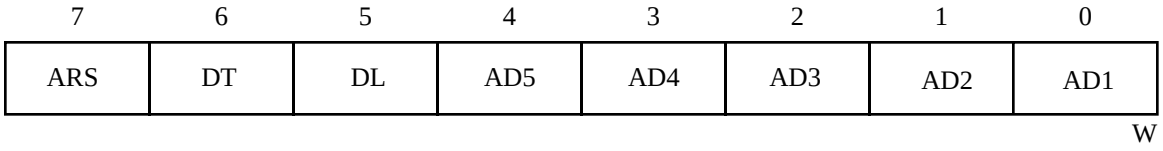
Address Register 0 (ADR0) reflects the internal GPIB address status of the TLC as configured using the ADMR. In addressing mode 2, ADR0 indicates the address and enable bits for the primary GPIB address of the TLC. In dual primary addressing (modes 1 and 3) ADR0 indicates the TLC major primary GPIB address. 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 five 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: Base Address + 11D (hex)

Attributes: Write Only, Internal to TLC



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 must 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 indicate 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: Base 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

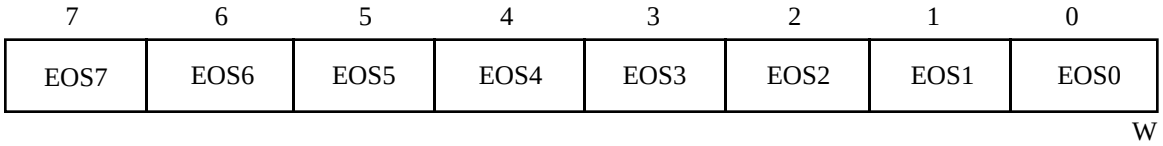
Address Register 1 (ADR1) indicates the status of the GPIB address and enable bits for the secondary address of the TLC if mode 2 addressing is used, or the minor primary address of the TLC if dual-primary addressing is used (modes 1 and 3). If mode 1 addressing is used and only a single-primary address is needed, both the talk and listen addresses disable in this register. If mode 2 addressing is used, 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.
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: Base Address + 11F hex

Attributes: Write Only, Internal to TLC



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) can 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, 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 uses four independent DMA channels. The DMAC can support single address (flyby) transfers or dual address (flowthrough) transfers. The GPIB-1014 uses two channels (Channel 0 and 1) for 8-bit flyby DMA transfers between VMEbus memory and the GPIB. All four channels are available for 8- or 16-bit flowthrough memory-to-memory DMA transfers. The DMAC supports unchained, continue, array chained, or link 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 TLC.

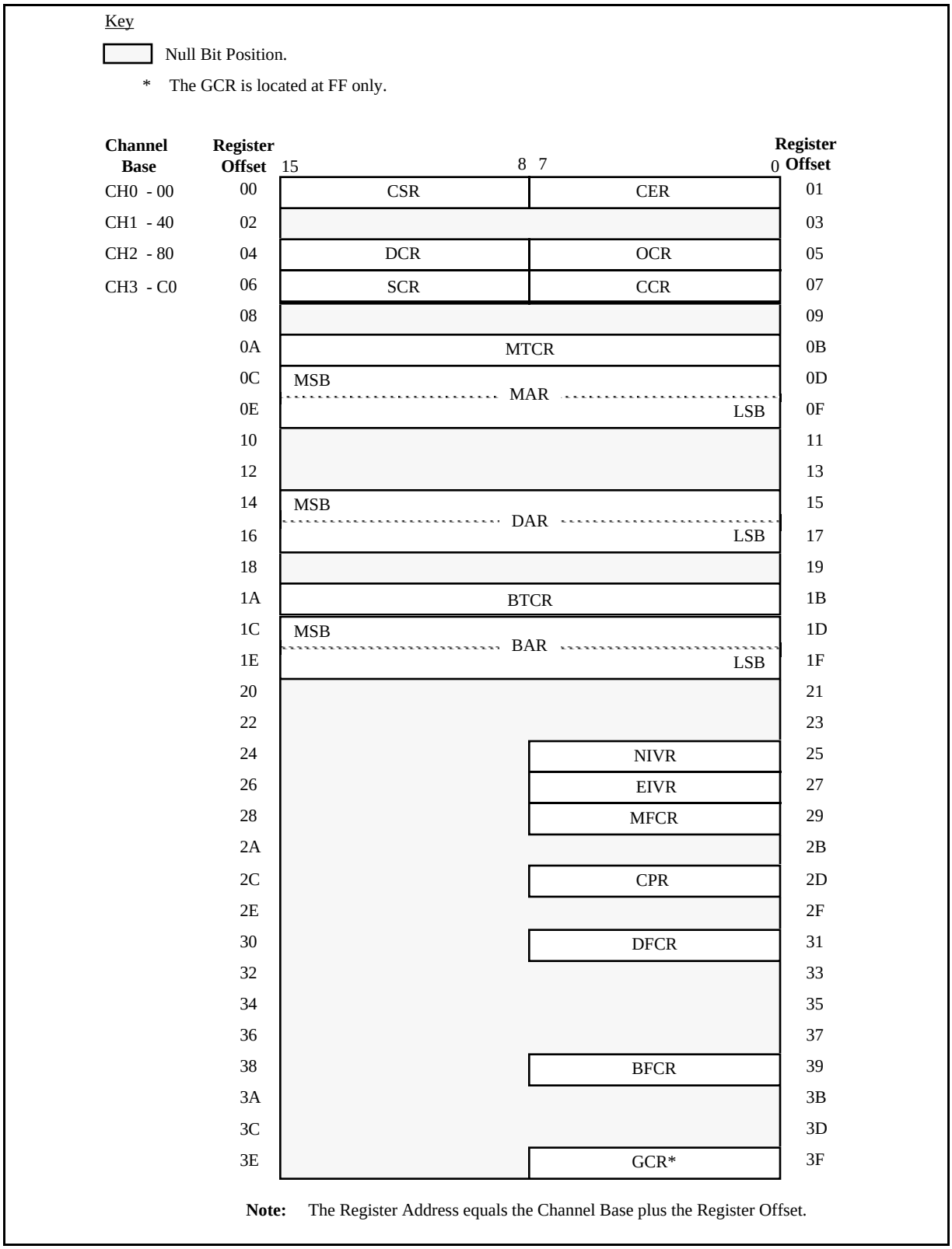
The DMAC contains a large number of internal configuration and status registers. These registers define and control the activity of the DMAC in processing a channel operation. The registers are addressed relative to the base address of the board. Locations not used in the board address space are reserved.
The registers set associated with each DMA channel is shown in Table 4-7.

Table 4-7. DMAC DMA Channel Register Set

Register Name	Mnemonic	Size
Memory Address Register	MAR	32 bits
Memory Transfer Counter Register	MTCR	16 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-1014 as outlined in Table 2-2. Figure 4-3 shows the DMA registers in order of their register offset.

Figure 4-3 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 4-3. DMA Register Memory Map

The following paragraphs describe the channel configuration and status registers. More information on the 68450 can be found in the *Motorola Semiconductor Technical Data MC68450 Advance Information Direct Memory Access Controller (DMAC)* or the *Hitachi Microcomputer System 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. Due to packaging limitations, only the least significant 24 bits are connected to the address output pins.

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 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 the BAR is automatically transferred to the 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 which 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 transfers, the array chained mode of operation is used. For more information on this mode, see *Sending/Receiving Messages* in Chapter 5 and *Block Termination* in Chapter 6.

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 that 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 device address register 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 memory must use flyby mode, memory-to-memory DMA transfers can be accomplished using any of the four available full-function DMA channels (channels 0 to 3). This not only makes the GPIB-1014 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.

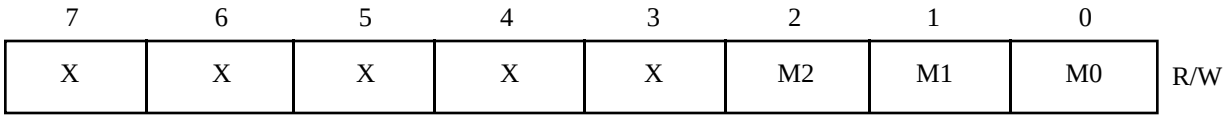
Transfer Count Registers

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 BTCR holds the size of the next block to be transferred which is then transferred into the MTCR when the last block is finished. In array chaining mode of operation, the BTCR holds the number of memory blocks to be transferred. Linked chaining mode of operation does not use the BTCR.

Function Code Registers

VMEbus Address: Base Address + 29 (hex) for Memory Function Code
Base Address + 31 (hex) for Device Function Code
Base Address + 39 (hex) for Base Function Code

Attributes: Read/Write, Internal to DMAC



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. 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 jumpers W3, W4, and W5 determine the 6-bit Address Modifier (AM) code on the VMEbus. The AM codes are used to identify the type of cycle specified by the DMAC when the GPIB-1014 is the bus master.

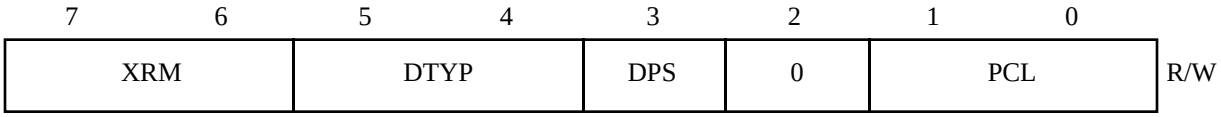
Table 3-1 shows how to program bits M2 through M0 to produce AM codes supported by the default settings of W3, W4, and W5. Tables 3-2 shows how to program bits M2 through M0 to produce any arbitrary AM code.

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

Device Control Register

VMEbus Address: Base Address + 04 (hex)

Attributes: Read/Write, Internal to DMAC



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

Bit	Mnemonic	Description
7-6r/w	XRM	External Request Mode Bits 7 through 6 The External Request Mode bits indicate 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-1014 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 indicate what type of device is on the channel. For the GPIB-1014 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-1014 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-1014 uses the four lines as status inputs. On PCL0*, GPIB signal SRQ* is connected. On PCL2*, signal REN* is connected. If programmed as status inputs, you can determine the state (high or low) of these two GPIB signals by reading the PCS bit in the appropriate CSR. PCL1* is designed to detect an interrupt from the TLC, synchronization of the GPIB handshake, or a bus error during a DMA transfer. PCL1* 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 the *Interrupts* section of Chapter 5. PCL3* is left unconnected. 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

VMEbus Address: Base Address + 05 (hex)

Attributes: Read/Write, Internal to DMAC

7	6	5	4	3	2	1	0	
DIR	0	SIZE		CHN		REQG		R/W

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-1014 GPIB transfers, the size is always byte 00. For memory-to-memory transfers, the size can be byte, word, or long-word. 00 = Byte (8 bits) 01 = Word (16 bits) 10 = Long-word (32 bits) 11 = (undefined, reserved)
3-2r/w	CHN	Chain 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

VMEbus Address: Base Address + 06 (hex)

Attributes: Read/Write, Internal to DMAC

7	6	5	4	3	2	1	0	
0	0	0	0	MAC		DAC		R/W

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

VMEbus Address: Base Address + 07 (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 cannot 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. Generally, this is not used for GPIB-1014 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-1014 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

VMEbus Address: Base Address + 00 (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) bits are set automatically by DMAC. Bits are cleared by writing a one (1) to each register bit or by resetting the DMAC.

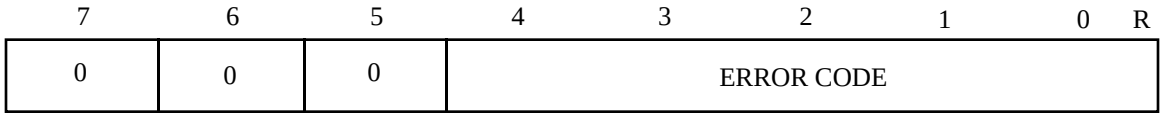
Bit	Mnemonic	Description
7r/w	COC	Channel Operation Complete bit The Channel Operation Complete bit is set if the DMA transfer has completed. This bit is set following the termination, whether successful or not, of any DMA operation. This bit must be cleared to start another channel operation. 0 = Channel 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* in Chapter 6 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-1014 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. 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

VMEbus Address: Base Address + 01 (hex)

Attributes: Read Only, Internal to DMAC



The Channel Error Register (CER) is an error condition 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	0	Reserved Bits
4-0r	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

VMEbus Address: Base Address + 2D (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-1014 application, channel 0 and channel 1 priority are 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 indicate 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 a 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-1014 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-1014 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 Channel Control Register (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.

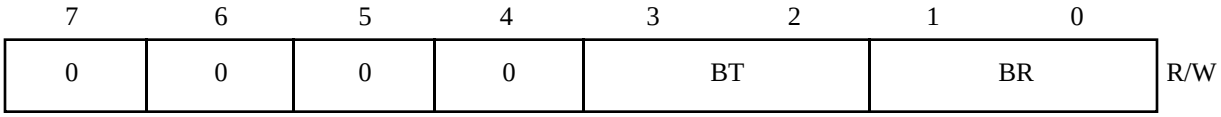
For GPIB DMA transfers, only channel 1 is used for interrupts. This is described in more detail in Chapter 5.

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

VMEbus Address: Base Address + FF (hex)

Attributes: Read/Write, Internal to DMAC



When the transfer mode is cycle steal with hold, 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%

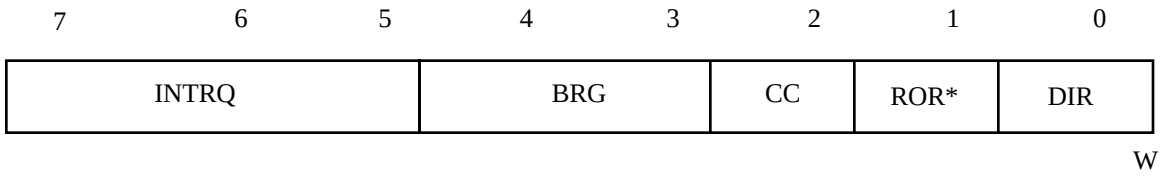
Configuration Registers

The GPIB-1014 contains two 8-bit write-only registers that are used to configure some of the board operating parameters.

Configuration Register 1 (CFG1)

VMEbus Address: Base Address + 101 (hex)

Attributes: Write Only, Internal to DMAC



Configuration Register 1 (CFG1) is an 8-bit write-only register used to configure the GPIB-1014 parameters. The register bits define the following operating parameters.

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

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

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
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.

Bit	Mnemonic	Description
4-3w	BRG	Bus Request/Grant Bits

The Bus Request/Grant bits are used to select which pair of the VMEbus request/grant lines are used by the GPIB-1014 to request and obtain control of the system bus.

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.

2w CC Carry Cycle Bit

The Carry Cycle bit is used to enable or disable the automatic carry cycle feature of the GPIB-1014.

1 = Carry Cycle feature enabled
0 = Carry Cycle feature disabled

The feature is not enabled after power-up or after the board is reset.

1w 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-1014.

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.

0w DIR Direction Bit

This bit is used to indicate the direction of the GPIB DMA data transfer.

0 = Transfer from system memory to GPIB
1 = Transfer from GPIB to system memory

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

Configuration Register 2 (CFG2)

VMEbus Address: Base Address + 105 (hex)

Attributes: Write Only, Internal to DMAC

7	6	5	4	3	2	1	0
0	0	0	0	SFL	SUP	LMR	SC
							W

Configuration Register 2 (CFG2) is an 8-bit write-only register that is used to set the board access mode, set the GPIB-1014 as System Controller, and drive the VMEbus SYSFAIL* line. It also contains a Local Master Reset bit that can be used to reset the GPIB-1014 to a known state.

Bit	Mnemonic	Description
7-4w	0	Reserved Bits Write zeros to these bits.
3w	SFL	System Fail Bit This board-specific bit is used to drive the VMEbus SYSFAIL* line 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 drives the SYSFAIL* line active. Setting this bit to a 1 turns the LED green and releases 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
2w	SUP	Supervisor Bit The Supervisor bit is used to select Supervisor-Only or Supervisor-and-User access to the board. A jumper is provided on the board to automatically select the value of this bit after power-up (or a system reset). 0 = Supervisor-and-User access to the board 1 = Supervisor-Only access to the board The factory default setting of jumper W2 will set this bit to 1 after power-up or reset.

Bit	Mnemonic	Description
1w	LMR	Local Master Reset Bit

The Local Master Reset bit is used to reset the GPIB-1014 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.

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

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

0w	SC	System Controller Bit

The System Controller bit is used to control whether the GPIB-1014 is in the System Controller mode or not.

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

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

Chapter 5
Programming Considerations

This chapter explains the initialization process, sending/receiving messages, and the serial/parallel poll process. Additional information on programming the µPD7210 GPIB interface chip can be obtained from the *µPD7210 GPIB-IFC User Manual* by NEC Electronics U.S.A., Inc. More specific information on programming the 68450 DMAC chip can be obtained from Hitachi and Motorola technical data.

Initialization

On power-up, the VMEbus system typically issues a system reset (SYSRESET*) that drives the GPIB-1014 RESET* signal active. This action clears both Configuration Registers 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-1014 also has another method for initializing the circuitry on the card. If the Local Master Reset (LMR) bit in Configuration Register 2 is set, the RESET signal is driven and the GPIB-1014 is initialized in the same manner.

The NEC µPD7210 Talker/Listener/Controller (TLC) integrated circuit is initialized as follows:

- 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).
- All bits of the Serial Poll Mode Register (SPMR) are cleared.
- The End Or Identify (EOI) bit is cleared.
- All bits of the Auxiliary Registers A, B, and E (AUXRA, AUXRB, and AUXRE) are cleared.
- The Parallel Poll Flag and Request System Control (RSC) local message are cleared.
- The Internal Clock Register (ICR) is set to a count of eight.

- 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 bit is set and after LMR has been cleared. While the TLC internal signal pon is set, all Auxiliary Mode Register (AUXMR) commands are cleared and cannot be executed. All other TLC registers can be programmed while 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 General Control Register (GCR), Device Control Register (DCR), Operation Control Register (OCR), Sequence Control Register (SCR), Channel Control Register (CCR), Channel Status Register (CSR), Channel Priority Register (CPR), and Channel Error Register (CER) are cleared for all channels. This resets the Start (STR), Continue (CNT), Channel Active (ACT) and 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).

A typical programmed initialization sequence for the GPIB-1014 might include the following steps:

1. Set and then clear the Local Master Reset (LMR) bit in Configuration Register 2 (CFG2) to place the GPIB-1014 in a known, quiescent state.

2. Load the two GPIB-1014 Configuration Registers (CFG1 and CFG2) with the appropriate values to configure the desired operation.

3. 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 a VMEbus interrupt is used, the PCL bits in Channel 1 DCR must be set to 01 for status input with interrupt. In addition, bit EINT in Channel 1 CCR must be set for Channel 1 to generate an interrupt. If polling (no interrupt) is used, set PCL bits to 00 for status input without interrupts. Bit EINT should be cleared.

5. If the VMEbus interrupt is used, load the DMAC Channel 1 NIVR and EIVR with the desired GPIB-1014 Status/ID byte.

6. Load the TLC primary GPIB address in Address Register 0 (ADR0) and Address Register 1 (ADR1).

7. Enable or disable the GPIB Talker and Listener functions and addressing mode using the ADMR.

8. Load the Serial Poll response in the SPMR.

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

10. Clear power on (pon) by issuing the Immediate Execute pon auxiliary command to the TLC to bring the TLC online.

11. Execute the desired TLC auxiliary commands.

The registers associated with the 68450 DMAC do not need to be configured further until just prior to a DMA operation.

The GPIB-1014 as GPIB Controller

The GPIB-1014 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). The following section discusses the various transitions between these two modes.

Becoming Controller-In-Charge (CIC) and Active Controller

The TLC can become 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-1014 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 CFG2 is set. To take control, issue the Set IFC auxiliary command, wait for a minimum of 100 μsec, and then issue the Clear IFC auxiliary command. The ensuing 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-1014 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 My Talk Address (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 GPIB talk address.

2. The TLC receives the GPIB TCT message.

3. The current Active Controller sees the completed handshake, goes to idle, 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 clearly determine that the TLC is the GPIB Active Controller and can send remote messages.

Sending Remote Multiline Messages (Commands)

The GPIB-1014 sends commands as Active Controller simply 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 should not be attempted.

The TLC recognizes any commands applicable to itself, such as its own talk or listen address. Thus, to make the TLC a Listener, write its listen address to the CDOR.

Going from Active to Standby Controller

If the TLC is GPIB Active Controller, the Controller Standby State (CSBS) is entered upon reception of the Go To Standby auxiliary command. The ATN line is unasserted as soon as the TLC enters CSBS. Even though the TLC GPIB Controller state machine is in standby, the CIC bit in the ADSR is still set. 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 to standby.

Case 1: The TLC becomes the GPIB Talker when ATN is unasserted. To do this, wait for CO to be set, send the TLC GPIB Talk Address (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), and it can then 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:

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.

Caution: Some instruments are very sensitive to assertions of ATN at what they may consider the "wrong time," and can lose data, abort operations being performed, or, in some severe cases, become completely nonresponsive. Consult your instrument's manual carefully to make sure any such situations can be avoided.

Case 2: The TLC, as a Listener, takes control upon receipt of the Take Control Synchronously auxiliary command. If programmed I/O is used, the Take Control Synchronously auxiliary command should 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 first 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 at reception of the EOS message.

Next, the carry cycle feature of the GPIB-1014 should be used to cause the RFD holdoff on all data mode auxiliary command to be written to AUXRA before the DMAC reads the last data byte from the TLC. Finally, set the CC bit in CFG1, and then start the DMA transfer. When the DMA transfer has finished, either from an END interrupt or a FIN interrupt, 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. Because the Listen in Continuous Mode auxiliary function is active, the Take Control Synchronously auxiliary command may 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 may 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 Cases 2 and 3 above, the END IE bit in IMR1 can also be set to indicate to the program that the TLC (functioning as a GPIB Listener) has received its last byte.

In all cases, a CO bit status of 1 indicates that the GPIB-1014 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 (in order 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 GPIB Talk address 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-1014 as GPIB Talker and Listener

The TLC may be either GPIB Talker or Listener, but not both simultaneously. Either function is deactivated automatically if the other is activated. The TA, LA, and ATN* bits in the ADSR together indicate the specific state of the TLC.

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

The following sections 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) are used to activate the TLC GPIB Talker and Listener functions. If used, ton or lon should 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 using the CO bit and the CDOR. 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. Upon receipt of its major or minor MTA or its major or minor MLA from the GPIB Active Controller, the TLC is addressed as Talker or Listener. If the TLC has received its GPIB Talk Address, the TA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DO bit in ISR1 is set. If the TLC has received its GPIB Listen address, the LA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DI bit in ISR1 is set when the first GPIB data byte is received.

Address Mode 2

Address Mode 2 is used when Talker Extended (TE) or Listener Extended (LE) functions are to be used. TE and LE functions require receipt of two addresses (primary and secondary) before setting TA or LA. The TLC GPIB primary address is indicated by the byte written to ADR0. The secondary address is indicated 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 GPIB Talk address immediately followed by its secondary GPIB Talk address, the TA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DO bit in ISR1 is set. If the TLC has received its Primary GPIB Listen address, the LPAS bit in the ADSR is set. If the TLC has received its primary GPIB Listen address immediately followed by its secondary GPIB Listen address, the LA bit in the ADSR is set, the ADSC bit in ISR2 is set, and the DI bit in ISR1 is set when the first 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 3

Address Mode 3, like Address Mode 2, is used to implement Extended GPIB Talk and Listen address recognition. However, unlike Address Mode 2, Address Mode 3 provides for both major and minor primary addresses, and 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). Write the TLC major GPIB primary address to ADR0 and the TLC minor GPIB primary address to ADR1.

2. Receipt of the TLC major or minor primary GPIB Talk Address (MTA) or major or minor primary GPIB Listen Address (MLA) sets TPAS or LPAS, indicating that the primary address has been received.

3. If the next GPIB command following the primary address is a secondary address, the APT bit is set and a DAC handshake holdoff is activated (the GPIB DAC message is held false).

4. In response to APT, the program must complete the following events:

    - Determine whether the command just received is a listen, talk, major, or minor address by reading the LPAS, TPAS, and MJMN bits of the ADSR.

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

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

6. When 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 occur:

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

    - The GPIB DAC message to be sent true, and the GPIB handshake to be finished.

7. When 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 occur:

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

    - The GPIB DAC message to be sent true, and the handshake to be 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

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-1014). 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 next section.

Using Direct Memory Access

The onboard DMA Controller is the 68450 (DMAC). This chip provides four independent DMA channels, of which two channels (Channel 0 and 1) can be used by the GPIB-1014 to transfer data between the VMEbus memory and the GPIB. The GPIB-1014 supports single-address (flyby mode) DMA transfer to/from the TLC where data bytes transfer directly between VMEbus memory and the TLC.

Although the 68450 supports several different DMA request generation modes, it must be programmed for external, cycle steal DMA request, since the TLC asserts 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, upon receiving a DMA request from the TLC, the DMAC requests use of the VMEbus. Once the VMEbus is granted to the GPIB-1014, the DMAC performs the DMA transfer. Then it immediately releases the bus.
In cycle steal with hold mode, after performing a DMA transfer, the DMAC will hold the VMEbus for a programmable time period waiting for another DMA request. If no DMA request is received from the TLC in the time period specified, the DMAC will relinquish ownership of the VMEbus. If a DMA request is received from the TLC in the time period specified, the DMAC performs the transfer immediately. The hold option is provided to reduce the VMEbus arbitration time during GPIB transfers.

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 a 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-1014 also provides the VMEbus Release On Request feature. This feature, which is enabled in CFG1, causes the board to hold onto the VMEbus as long as no other board requests the VMEbus. 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 while the board is holding the bus, the DMAC is re-granted the bus immediately to start the next DMA transfer. If the board is holding the bus and if some other device requests the bus, the GPIB-1014 will immediately release the bus.
The two DMAC channels used by the GPIB-1014 are channels 0 and 1. The DMAC can be configured to transfer data between the GPIB TLC and the VMEbus system memory with or without the carry cycle feature. This feature is enabled in CFG1. During a DMA transfer from the VMEbus memory to the GPIB, the carry cycle feature is used to make the TLC send the EOI bit active with the last byte of data. During a DMA transfer from the GPIB to the VMEbus memory, the carry cycle feature is used to make the TLC hold off the GPIB handshake sequence after the last byte has been accepted by all GPIB Listeners.

Channel 0 is used to control the transfer of the entire block (n bytes) of data when the carry cycle feature is not selected. If the carry cycle feature is used, Channel 0 is used to control the transfer of the first (n-1) bytes of data. Then Channel 1 is used to transfer one byte of data (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 holdoff handshake). The carry cycle byte is just a TLC auxiliary command that is inserted in the n-byte transfer. There is a slight preparation process you need to do if you use the carry cycle (See *DMA Transfers with a Carry Cycle* later in this chapter.)

After the GPIB-1014 has transferred n bytes of data, it then automatically detects for GPIB handshake synchronization, that is, if all Listeners have accepted the data. This synchronization signal, along with interrupt signals from the TLC and VMEbus signal BERR*, are all routed to a pin (PCL1) on the DMAC. If enabled for interrupt, the assertion of any of these three signals sets the pin and the DMAC asserts an interrupt request line on the VMEbus. Thus, you can set up and start the DMA transfer, then simply wait for a GPIB synchronization interrupt (if interrupts are not enabled, polling is possible). This indicates n bytes have been transferred and the GPIB handshake is synchronized.

DMA Transfers without the Carry Cycle

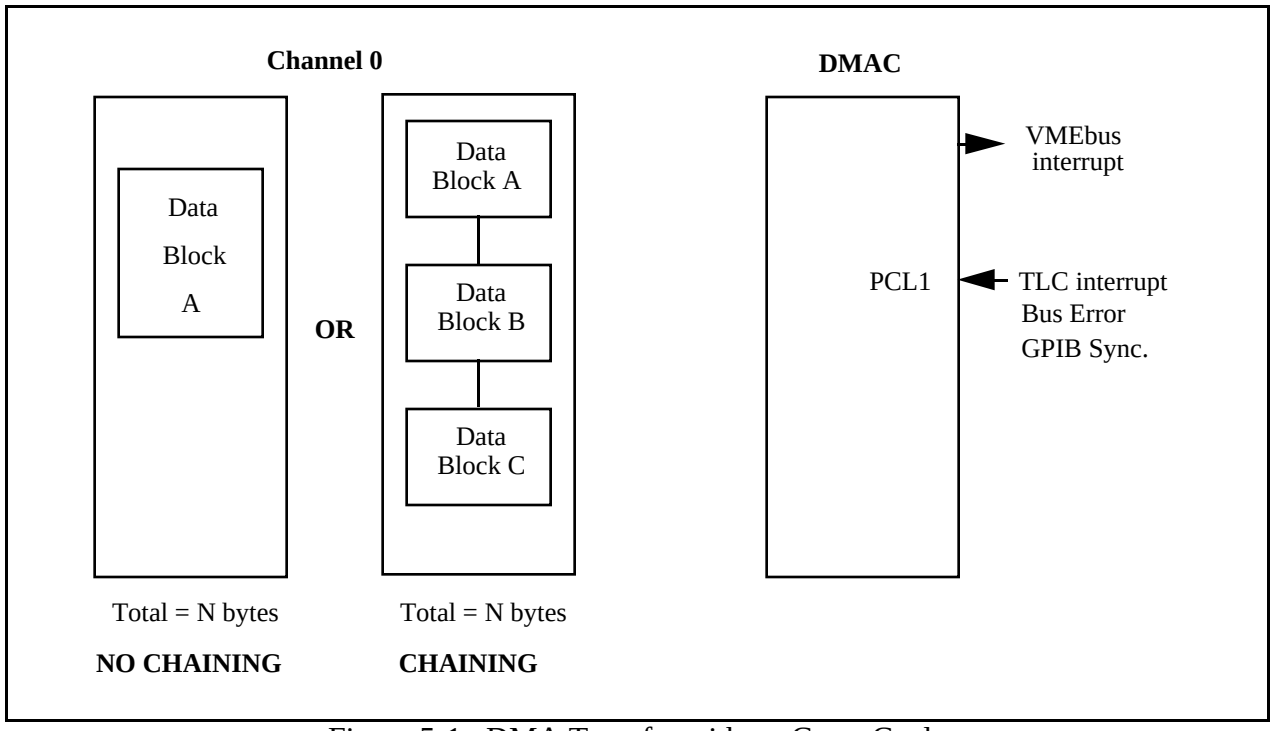


Figure 5-1. DMA Transfer without Carry Cycle

If the carry cycle feature is not needed in a transfer sequence, little programming is needed. Channel 0 is used to transfer the entire block of data, since the carry cycle byte does not need to be inserted before the last data byte. Channel 1 DMA is not used and must not be started. As indicated in Figure 5-1, depending upon the number of data blocks to be transferred, you can use no-chaining or chaining mode. If interrupts are enabled on Channel 1, the TLC can interrupt the processor on the VMEbus system on one of 13 GPIB events. Similarly, if BERR* occurs during a DMA transfer, the processor is also interrupted. When the GPIB handshake is synchronized, the system is also interrupted and you can check the COC and ERR bit in the CSR of Channel 0 to determine the status of the DMA transfer. This is described further in *Terminating the Transfer and Checking the Result* later in this chapter.

A detailed programming sequence for DMA transfers without a carry cycle is as follows:

1. In CFG1, the CC bit must be cleared to 0. The DIR bit should be set to reflect the direction of the DMA transfer (1=GPIB-to-Memory, 0=Memory-to-GPIB). The ROR* bit must also be set if the Release On Request feature is to be enabled. If interrupts are used, the INTRQ bits are set to select the interrupt level. BRG bits must be set to choose one of four VMEbus request lines.

2. Channel 0 must be configured to provide a flyby transfer for the n data bytes between the GPIB and the VME system memory. The sequence is as follows:

    a. The CCR of Channel 0 must be written with the SAB bit set to abort the channel operation in case it is still active.

b. A 0xFF (hex) must be written to the CSR of Channel 0 to clear any leftover error or status bits.

c. The DCR of Channel 0 is loaded with the proper value to select the DMA transfer mode (cycle steal without hold or cycle steal with hold). The DTYP bits should be set to binary 10 (device with ACK*, implicitly addressed), the DPS bit must be set to 0 (8-bit port size), and the PCL bits should be set to binary 000 (status input). If the cycle steal with hold transfer mode is selected, the GCR must be written to select the required timeout. (See the GCR description for recommended values.)

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

e. Write to the SCR of Channel 0. The MAC bits must be set to determine if the MAC counts up or down (binary 01=Up, binary 10=Down). The DAC bits are not used.

f. If no chaining is required, complete the following events:

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

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

- Load the MTCR of Channel 0 with the number of bytes in the buffer to be transferred (must be less than or equal to 64K).

g. If chaining is required, complete the following events:

Note: Chaining modes (array or linked) use an address & transfer count array, which is an array of pointers that point to the data blocks to be transferred.

1. For array or linked chaining, load the BFCR of Channel 0 with the proper data to generate the required Address Modifier Code to access the address & transfer count array. See Tables 3-1 and 3-2 for recommended values.

2. For array or linked chaining, load the BAR of Channel 0 with the starting physical address of the address & transfer count array.

3. For array chaining, load the BTCR of Channel 0 with the number of entries in the address & transfer count array. Linked chaining does not use BTCR.

4. For array or linked chaining, load the MFCR of Channel 0 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.

5. Set up the data blocks and the address & transfer count array in VMEbus memory. Figures 6-1 and 6-2 describe how to set up the array for both chaining modes.

Interrupts are generally not enabled for Channel 0. The CPR, NIVR, DAR, DFCR, and EIVR of Channel 0 are generally not used.

3. Channel 1 must be programmed before starting the transfer in the following manner:

   a. If VMEbus interrupts are used, complete the following events:

      - Set the PCL bits in the DCR of Channel 1 to 01 for status input with interrupt.
      - Set the EINT bit in the CCR of Channel 1 to enable interrupts.
      - Load the NIVR and EIVR of Channel 1 with the proper status/ID byte to return to the VMEbus interrupt handler.

   b. If VMEbus interrupts are not used, complete the following event:

      - Set the PCL bits in the DCR of Channel 1 to binary 00 for status input. You can check the PCT and PCS bits in the CSR of Channel 1 to see if the TLC has requested an interrupt, BERR* has occurred, or if GPIB synchronization has occurred.

      It is not necessary to configure any other Channel 1 registers.

4. Once Channels 0 and 1 have been configured, the DMAC must be started by setting the STR bit in the CCR of Channel 0. Channel 1 is not used.

5. Finally, configure the TLC for 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.

DMA Transfers with the Carry Cycle

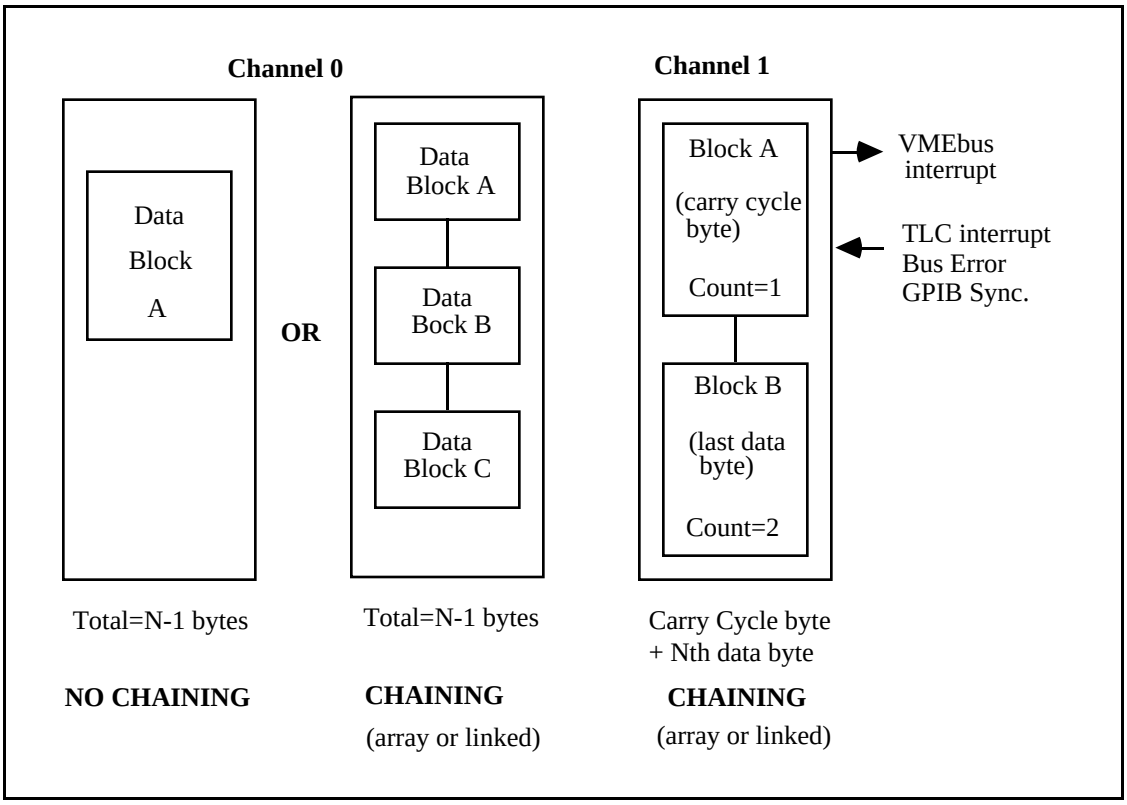


Figure 5-2. DMA Transfer with Carry Cycle

When the carry cycle feature is needed in a transfer, it is transparent to the system CPU and is automatically handled by the GPIB-1014 once the channels have been properly configured. As indicated earlier in this chapter, Channel 1 is now used to transfer the carry cycle byte and the last data byte. Setting up Channel 0 is similar to the steps used to program the DMA explained earlier in this chapter. Channel 1 is set up to transfer two tiny blocks of data. It is easiest to use the array chaining mode on Channel 1; however, linked chaining is also possible. Channel 1 can also be configured to generate VMEbus interrupts. When the GPIB is finally synchronized, you can check the COC and ERR bit in the CSR of Channel 0 to determine the status of the DMA transfer of the first n-1 bytes. The success of the DMA transfer is determined by examining the MTCR of Channel 1. Refer to the *Terminating the Transfer and Checking the Result* section later in this chapter for more details.

A detailed programming sequence for DMA transfers with a carry cycle is as follows:

1. In CFG1, set the CC bit in CFG1 to a 1 to enable the carry cycle feature. Set the DIR bit to reflect the direction of the DMA transfer (1=GPIB-to-Memory, 0=Memory-to-GPIB). Set the ROR* bit if the Release On Request feature is to be enabled. If interrupts are used, the INTRQ bits are set to select the interrupt level. BRG bits must be set to choose one of four VMEbus request lines.

2. Channel 0 must be configured to provide a flyby transfer for the n-1 data bytes between the GPIB and the VME system memory. The sequence is as follows:

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

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

   c. Load the DCR of Channel 0 with the proper value to select the DMA transfer mode, Cycle Steal without Hold or Cycle Steal with Hold. Set the DTYP bits to binary 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 the GCR to select the required timeout. (See the GCR description for recommended values.)

   d. The OCR of Channel 0 is written to. Set the DIR bit to reflect the 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 transfer). Chaining is used to transfer a block of data greater than 64K or to transfer multiple blocks of data.

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

   f. If no chaining is required, complete the following events:
      - Load the MFCR of Channel 0 with the proper data to generate the required Address Modifier Code to access the data buffer. See Tables 3-1 and 3-2 for recommended values.

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

      - Load the MTCR of Channel 0 with the number of bytes (n-1 total) in the buffer to be transferred (must be less than or equal to 64K: 65536 or 0xFFFF hex).

   g. If chaining is required, complete the following events:
      Note: Chaining modes (array or linked) all use an address & transfer count array, which is an array of pointers that point to the data blocks to be transferred.

      - For array or linked chaining, load the BFCR of Channel 0 with the proper data to generate the required Address Modifier Code to access the "address & transfer count" array. See Tables 3-1 and 3-2 for recommended values.

      - For array or linked chaining, load the BAR of Channel 0 with the starting physical address of the address & transfer count array.

      - For array chaining, load the BTCR of Channel 0 with the number of entries in the address & transfer count array. Linked chaining does not use the BTCR.

- For array or linked chaining, load the MFCR of Channel 0 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 & 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.

h. Finally, interrupts are generally not enabled in Channel 0. The CPR, NIVR, DAR, DFCR, and EIVR of Channel 0 are generally not used.

3. Channel 1 is used to control the interrupts from the board and to implement the carry cycle feature. A carry cycle array is constructed in memory. Chaining is used to transfer the array. The carry cycle array is just an address & transfer count array (that 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 the CCR of Channel 1 with the SAB bit set to abort the channel operation in case it is still active.

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

   c. Load the DCR of Channel 1 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 recommended values.

   d. Write to the OCR of Channel 1. Set the DIR bit to reflect the 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 or linked chaining), and set the REQG bits to 10 (REQ* line initiates transfer). The array chaining feature is usually used to implement the carry cycle because of its simplicity.

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

   f. For array or linked chaining, load the BFCR of Channel 1 with the proper value to generate the required address modifier code, which then accesses the address & transfer count array. (See Tables 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 & transfer count array.

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

i. For array or linked chaining, load the MFCR of Channel 1 with the proper value to generate the desired address modifier code, which then accesses the data blocks. (See Tables 3-1 and 3-2 for recommended values.)

Note: If you are using the array chaining mode, construct a special carry cycle array in memory. The array must begin at an even address and all addresses in the array must be even. The contents of the array are as follows :

First four bytes = physical address of carry cycle byte

Next two bytes = 0001 (hex)

Next four bytes = physical address of last data byte in the data buffer to be transferred

Last two bytes = 000**2** (hex) (See *Terminating the Transfer and Checking the Result* or *Theory of Operation* on why two bytes are required.)

j. For linked chaining mode, the carry cycle array is similar to the previous carry cycle array, except that you need the link address to the next array entry. Figures 6-1 and 6-2 describe how to set up the array for both chaining modes.

The carry cycle byte is the command that is to be written to the AUXMR of the TLC (send EOI, RFD Holdoff on ALL, and so on). 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 carry cycle array (four bytes). The DMAC uses this carry cycle array and the chaining mode with Channel 1 to insert the TLC auxiliary 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 and the last byte of the data buffer is transferred.

k. Program Channel 1 in the following manner before starting the transfer:

- If VMEbus interrupts are used:

  (1) Set the PCL bits in the DCR of Channel 1 to 01 for status input with interrupt.

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

  (3) Load the NIVR and EIVR of Channel 1 with the proper status/ID byte to return to the VMEbus interrupt handler.

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

It is not necessary to write to the MTCR, MAR, DAR, DFCR, or CPR of Channel 1.

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

5. Finally, configure the TLC for a DMA operation. The sequence is as follows:

    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.

    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.

Polling During DMAs

All the GPIB-1014 registers are accessible during DMA operations while the CPU has control of the bus. If interrupts are not enabled, the CSR of Channel 1 can be read to check that 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 *DMA Transfers With A Carry Cycle* section earlier in this chapter.

Terminating the Transfer and Checking the Result

If either Channel 0 or 1 is improperly programmed, the ERR bit in the CSR of the active channel is set by the DMAC and the CER of the channel indicates a configuration error. The following paragraphs assume the channels are properly programmed to allow normal termination of a GPIB DMA transfer. They describe how to check the CSR of the channel to determine the success of the GPIB DMA transfers.

The termination of the GPIB DMA transfer causes an interrupt if interrupts have been enabled in Channel 1. If interrupts are not used, this condition can be detected by polling. Regardless of whether or not the carry cycle feature is used, when the 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 transition occurs on the Channel 1 PCL line of the DMAC. The PCT bit in the CSR of Channel 1 is set by this transition. This generates an interrupt if Channel 1 is configured to interrupt on a PCL transition. This transition can also be detected by polling the CSR of Channel 1 and waiting for the PCT bit to set. Once detected, the software must service the interrupt condition.

The first action in the interrupt handler must be determining what caused the interrupt. A VMEbus interrupt can be caused by any of the following events:

- An interrupt from the TLC
- A bus or DMAC error that occurred during a DMA transfer
- GPIB handshake synchronization

To determine which condition caused the interrupt, you must first examine the CSR of Channel 1. If the ERR bit of Channel 1 is set, a DMAC error occurred while Channel 1 was transferring data. The CER of Channel 1 indicates the type of error that occurred. 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 examining the CER of Channel 1. See the CER register description in Chapter 4 for more information. If the ERR bit of Channel 1 is zero and its PCT bit is a one (which is the usual case), one of the following conditions has occurred:

- A bus error has occurred while Channel 0 is transferring data.
- The TLC is interrupting.
- Synchronization on the GPIB is detected.

If a Channel 0 bus error has occurred, the ERR bit is set in the CSR of Channel 0 and the CER of Channel 0 indicates a bus error.

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

If the PCT bit in Channel 1 is set and there are no errors on Channel 0 or 1, one of the following events has occurred:

- The TLC is interrupting.
- The DMA operation is complete and GPIB synchronization is detected.

The TLC should 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, the PCL transition was caused by the GPIB becoming synchronized. To ensure that the DMA transfer was completed, the CSR of Channel 0 must always be examined. The COC bit must be set and the ERR bit should be cleared. This check is sufficient if the carry cycle feature was not used.

If the carry cycle feature was used, the CSR of Channel 1 must also be examined. The Channel Operation Complete (COC) bit in Channel 1 must *not* be set, and the MTCR must contain a *1* because the second entry of the carry cycle array requests a 2-byte DMA transfer, but only one byte (the nth data byte) is permitted to transfer. A 2-byte transfer is requested for the following reason: If Channel 1 was permitted to reach terminal count (by requesting a 1-byte transfer), the COC bit would set and an interrupt would occur (the DMAC interrupts if either the COC bit or the PCT bit is set). It is necessary for the COC 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, the MTCR of Channel 1 contains a 1 and the next DMA request from the TLC is not allowed to reach the DMAC through 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 the CCR of Channel 1 with the SAB bit set.

In general, 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 CFG1 to unassert the PCL line of Channel 1. (A write to CFG1 clears the circuitry on the board, which generates the GPIB handshake synchronization status signal.)
6. If a GPIB error occurred during the transfer (ERR bit set in the TLC's ISR1), poll the BTAC and MTCR of the DMAC until you are satisfied that no DMA transfers are taking place (that is, the transfer count is not changing). (This step is necessary because of a 68450 DMAC bug that can cause the VMEbus to hang. If you do not wait for any pending DMA transfer to complete before issuing a SAB, as called out in Step 7 below, the 68450 may incorrectly remove its bus request before receiving a bus grant intended for it.)
7. Write a software abort to Channel 0.
8. Clear the Status Register of Channel 0.
9. Write a software abort to Channel 1 if a carry cycle was used.
10. Clear Status Register of Channel 1 to clear any remaining interrupt.

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 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 the 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 gets 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 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 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-1014 DMA registers are irrelevant.

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.

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.

To determine if the CDOR is empty or if the DIR is full, either poll ISR1 until the DO or DI status first appears, or allow a program interrupt to occur on the respective event. Remember, however, that the status bits and the interrupt signals are cleared when the ISR1 is read, so the absence of a true DO or DI status does not unambiguously indicate that the CDOR is still full or that the DIR is still empty.

Sending END or EOS

Send the GPIB END message by issuing the Send EOI auxiliary command just before writing the last data byte to the CDOR. To send the GPIB EOS message, make 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

If the GPIB-1014 is enabled for interrupts, there are three events that can cause an interrupt on the VMEbus. The first event is an interrupt from the TLC. The second event is a GPIB handshake synchronization that occurs when a DMA transfer is finished and the GPIB is synchronized. The last event is a bus error that occurs during a DMA transfer. Interrupts must be enabled in software to be recognized as described in the following paragraphs.

The Interrupter circuitry of the GPIB-1014 enables the board to interrupt the CPU to request service. The hardware offers programmable selection of the interrupt priority level (level 1 to level 7) via Configuration Register 1. Interrupts can be generated on numerous conditions. These conditions can also be detected by polling if desired. If interrupts are enabled, the GPIB-1014 generates an interrupt on any of the following conditions:

- An interrupt request is received from the GPIB TLC.
- A bus error occurs during a DMA transfer.
- 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.

All interrupt sources from the GPIB-1014 are routed to the onboard DMAC, which actually generates the interrupt request on the VMEbus. The Peripheral Control Line (PCL1) of Channel 1 is used as a status input for the interrupt sources listed earlier. These events are ORed together so that a transition on the PCL1 occurs when *any* of these events occur. If 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 can be programmed to generate an interrupt on a negative transition (high to low) on its PCL. This enables an interrupt, which is requested when the PCT bit of the CSR is set, indicating that a transition has occurred.

The TLC contains its own internal registers, which are used to control and enable interrupts. The interrupt output from the TLC, however, is sensed by the PCL of DMA Channel 1. If an interrupt operation is used, the DMAC Channel 1 must be configured to interrupt on a high-to-low transition of the PCL1.

If the DMAC encounters a bus error during operation, a negative transition is caused on the PCL of Channel 1, thus causing an interrupt.

The GPIB-1014 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 generated on the PCL of Channel 1.

Because 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 the interrupt of the TLC pulls PCL1 low, your program must read ISR1 or ISR2 of the TLC to clear the interrupt and pull PCL1 high. If GPIB synchronization occurs, you must write any value to CFG1 to pull PCL1 high. If a VMEbus error occurs during a DMA transfer, the PCL1 is pulled high automatically when the VMEbus memory releases signal BERR* high. Your program cannot pull PCL high in the last case, but can only clear the PCT bit in the CSR.
Interrupts from the PCL of DMA Channel 1 are enabled by setting the EINT bit in the CCR of Channel 1. An interrupt is generally not enabled on Channel 0. If triggered, all three interrupt events mentioned earlier will cause the set PCT bit to indicate a PCL transition and an interrupt. A bus error that occurs when Channel 1 is active sets both the ERR and PCT bits in the CSR of Channel 1. A bus error that occurs when Channel 0 is active only sets the PCT bit in the CSR of Channel 1 and the ERR bit in the CSR of Channel 0.

After an interrupt is generated, the VMEbus interrupt handler asks the interrupting source for a status/ID byte so that it can branch to the appropriate interrupt service routine. Depending upon the value of the ERR bit, Channel 1 sends 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. The source of the interrupt must be determined by the interrupt service routine. 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 of Channels 0 and 2 are also connected to two GPIB signals so you can directly detect them. The PCL of Channel 0 is connected to the GPIB Service Request signal (active low). 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 the GPIB Remote Enable signal (active low). This can be used to read the status of the REN* line at any time or to generate an interrupt when the REN* line is asserted.

Serial Polls

Conducting a Serial Poll

The TLC, as CIC, can serial poll 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 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.

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 before asserting 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.

While the Serial Poll is in progress (SPAS=1), the CIC normally reads the STB only once; however, it can read STB any number of times. 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. Two general ways in which Parallel Polls are useful are as follows:

- When the GPIB Controller sees SRQ asserted in a system with several devices, it can quickly determine which one needs to be serial 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. This is contrasted with Serial Polls, where the overhead, in the form of addressing and enabling command messages, occurs with each poll.

Conducting a Parallel Poll

The TLC as Active Controller has the capability to conduct a Parallel Poll. When the Execute Parallel Poll auxiliary command is issued and the TLC internal local message rpp is set, 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 upon 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.

Because there are eight data lines, and for each line there can be one response (true or false) for each device (2 lines/device), 16 responses are possible. The line that a device uses and how that device drives the line depends upon how it was configured and whether its local individual status message (ist) 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. However, it can also 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. The following 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, the hardware is configured and installed).

- Remote configuration (Parallel Poll function subset PP1) involves dynamically assigning the response line and sense to devices on the GPIB. This is accomplished using the 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.

Follow the same procedure to disable polling with PPD (for example, when changing responses during reconfiguration).

Responding to a Parallel Poll

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

- Enabling the TLC 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 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 above must be explicitly handled in the software by writing the appropriate values to the U, S, and P3 to 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 to determine the source and value of the local individual status (ist) message. If the ISS bit in the AUXRB is 0, ist is set and cleared via the Set and Clear Parallel Poll auxiliary commands. If ISS is 1, 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 the *Conducting a Serial Poll* section earlier in this chapter, which 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. The GPIB system integrator must decide what ist indicates and what configuration is used. The response can be changed dynamically during program execution by changing the value of ist and, when remote configuration is used, by reconfiguration.

Chapter 6
Theory of Operation

This chapter contains a functional overview of the GPIB-1014 board and explains the operation of each functional block making up the GPIB-1014.

A brief description of the GPIB-1014 interface is given in Chapter 2 along with a functional block diagram (see Figure 2-4). The major elements of the GPIB-1014 are discussed in more detail in this chapter with references to signals and circuits shown in the schematic diagram in Appendix A.

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*. Since in the circuit implementation some positive true signals are derived from the inverted output of flip-flops, these terms are not synonymous with the device signals CLR (clear) and PR (preset).

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-1014. 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 TLC.

Slave Read and Write Transfers

During the slave read and write to the DMAC, both transceivers are enabled. In contrast, only the lower 8-bit transceivers are enabled during accesses to the TLC or to the two onboard 8-bit Configuration Registers. The GPIB-1014 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, which gates the upper data byte to the TLC. This data transceiver is automatically controlled by the DMAC signal HIBYTE*. When the data transfer is on VMEbus data lines D07 through D00, the HIBYTE* is not active and the TLC data bus is connected to the lower eight bits of the VMEbus data bus. When the data transfer is on VMEbus data lines D1 through D09, HIBYTE* is active and the TLC data bus is connected 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-1014 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 on VMEbus memory reads.

Control Signals

An F241 and F1241 buffer pair is used to transceive the data bus control signals AS*, WRITE*, DS0*, and DS1*. The OWN* signal of the DMAC is used to control the direction of the buffer pair. When the GPIB-1014 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* in this chapter). The UDS* and LDS* signals 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). In addition, if DTACK* of BERR* from the last cycle is still asserted, a flip-flop prevents the GPIB-1014 from driving the data bus, DS0* or DS1*.

Address

When the board is a VMEbus slave, information from the VMEbus address bus lines A15 through A9, the Address Modifier Lines AM5 through AM0, and the VMEbus signals IACK* and LWORD* is received and decoded by two LS2521 8-bit comparators. (For more information, see the *Address Decoding* section in this chapter.)

When the board is a VMEbus master, it drives the address lines (A23-A1), the address modifier lines (AM5-AM1), IACK*, and LWORD*. The upper 16 bits of the address are driven by two transparent D-type flip-flop AS573s while the lower eight bits are driven by an F245. The UAS* signal of the DMAC is used to clock the AS573s.

Control Equations of Transceivers

Table 6-1 lists the control equations for the address and data.

Table 6-1. Control Equations of Transceivers

VMEbus Signals	Transceivers	Control Equations
A23 through A16	AS573	OC* = OWN* (output enable) C = UAS (input clock)
A15 through A8	AS573	OC* = OWN* (output enable) C = UAS (input clock)
A7 through A1	F245	EN* = low (output enable) DIR = OWN (transceiver's direction)
HIBYTE transceiver OWN)	F245	EN* = HIBYTE* DIR = (BWR* & OWN*) + (BWR &
AM lines, IACK*, LWORD*	F241	EN* = OWN*
D15 through A8 OWN)	F245	EN* = (ACKEN* & DBEN*) + NEW_CYCLE* + DIACK DIR = (BWR* & OWN*) + (BWR &
D7 through D0 OWN)	F245	EN* = (((ACKEN* & DBEN*) + NEW_CYCLE*) & (BRDEN* + DS*)) + HIBYTE DIR = (BWR* & OWN*) + (BWR &

Address Decoding

During non-DMA operations, the GPIB-1014 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-1014 base address during short I/O transfers.

Two 25LS2521 comparators recognize the short I/O base address of the GPIB-1014. Except for Model 1S, the base address is selectable with jumpers located on the board or with strapped lines routed to the P2 connector. The base address of the GPIB-1014 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 through A9 match the jumper selected base

address. If DS* from the master is also asserted, local signal BRDEN* is asserted. Further decoding is necessary to determine which register is being addressed. Eight data lines (A8 through A1) are latched by an AS573 8-bit register at the start of every slave cycle (that is, when AS* is low and DTACK* from the last cycle is high) to provide the address decoder with constant address information.

When the GPIB-1014 has been addressed (signal BRDEN* 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. Also, DMAC asserts the DTACK* to prevent a bus error. A write to an unused location results in a normal bus cycle but no write occurs. Like a read, DMAC asserts DTACK* to prevent a bus error.

When the GPIB-1014 has been addressed, VMEbus address line A8 is a logic 1, A4 is a logic 1, the lower data strobe DS0* is active, and the TLC select signal TLCCS* is driven active, indicating that the TLC has been addressed. Address lines A3 through A1 are used to select the internal registers of the TLC. These lines are gated with the onboard signal CCBYTE and are connected to the TLC register select lines RS2 through RS0. The TLC functions as an 8-bit slave. It transfers data on the lower eight bits of the VMEbus data bus. The TLC registers are located at consecutive odd addresses and respond only when DS0* is asserted. The TLC responds to 16-bit transfers, but the upper data byte is not used during an I/O write and returns all ones during an I/O read. As described in the *DMA Gating and Control* section, signal CCBYTE forces a write cycle to the Auxiliary Mode Register inside the TLC when it is asserted.

When the GPIB-1014 has been addressed, VMEbus address line A8 is a logic 1, A4 is a logic 0, A2 is a logic 0, and Configuration Register 1 is addressed. When A8 is a logic 1, A4 is a logic 0, A2 is a logic 1, and Configuration Register 2 is addressed. These 8-bit registers 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. A read from either of these registers results in a normal bus cycle, but returns all ones. In addition, a pair of F74 flip-flops is used to keep local signal BRDEN* asserted until both Data Strobes (DS0* and DS1*) and Address Strobe (AS*) from the VMEbus master have gone away and the board has released DTACK* to end the current slave read/write cycle.

Accesses to locations within the 512-byte block that do not specifically address either the DMAC, the TLC, or one of the Configuration Registers result in a bus error because the GPIB-1014 does not respond.

Clock and Reset Circuitry

The clock and reset circuitry on the GPIB-1014 is used to generate the onboard clock and reset signals. The VMEbus signal SYSCLK (16 MHz) is received with a 74LS240 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 and the DMAC. 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 signal to control RD* and WR* to the TLC (see *Timing State Machine* later in this chapter).

The VMEbus signal SYSRESET* is monitored by the GPIB-1014. It is received with an LS240 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-1014 except the two bits in CFG2 (LMR and SYSFAIL*), the Address Decoding circuitry, and the DTACK* generation circuitry. BRDEN* circuitry (refer to the *Address Decoding* section earlier in this chapter). The two bits in CFG2 are reset only by the SYSRESET* signal or by a write to CFG2. The Address Decoding and DTACK* generation circuitry can only be reset by SYSRESET*.

The VMEbus signal BERR* is monitored by the GPIB-1014 while it is bus master. This signal is received by a 74LS241 receiver and is ANDed with the onboard signal OWN*, which is active while the GPIB-1014 is bus master. If BERR* is driven active during a DMA transfer, the DMAC input BEC1 is driven low, 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

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-1014.

Configuration Register 1

A 74LS273 octal D-type flip-flop and an LS74A are used to implement Configuration Register 1 (CFG1). Data is written into each register on the rising edge of the WR* signal generated by the Timing State Machine circuitry. Except for the ROR* bit, all other bits in CFG1 are cleared by the onboard RESET* signal generated by the Clock and Reset circuitry. This write-only register is used for a variety of configuration functions, including the following functions:

- Selecting the VMEbus Interrupt Request Priority for the GPIB-1014. 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 automatic carry cycle feature. Writing a 1 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.

- Enabling the Release On Request feature. Writing a 0 to this bit (ROR*) enables the Release On Request feature while writing a 1 disables the Release On Request feature. This bit is set to 1 during reset or power up. This bit is used by the DTB Requester circuitry.

- Selecting the direction of the DMA transfer. Writing a 1 to this bit (DIR) indicates that the transfer direction is from GPIB to VMEbus memory, while writing a zero indicates that the direction is from VMEbus memory to the GPIB. This bit is used by the GPIB Synchronization and Interrupt Control circuitry.

- Writing any value to this register resets the circuitry which detects GPIB synchronization after a DMA transfer. (For more information, see *GPIB Synchronization and Interrupt Control* later in this chapter).

Configuration Register 2

Four discrete 74LS74A D-type flip-flops are used to implement Configuration Register 2 (CFG2). Data is written into each bit of this register on the rising edge of the WR* signal generated by the Timing State Machine circuitry. The SC bits are cleared by the onboard RESET* signal 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 S/U* bit is set or cleared (to be in supervisor or slave mode) depending upon the way you have set jumper W2. This write-only register is used for a variety of configuration functions, including the following functions:

- Issuing a Local Master Reset (LMR) to the GPIB-1014. The LMR bit is cleared to a 0 on SYSRESET*. Writing to CFG2 with this bit set to a 1 drives the GPIB-1014 local signal RESET*. Writing a 0 to this bit releases RESET.

- Configuring the GPIB-1014 to be System Controller. Writing a 1 to this bit (SC) configures the GPIB-1014 as System Controller. This bit is used by the 75162 GPIB control transceiver to determine whether the GPIB-1014 drives or receives the GPIB signal IFC.

- Selecting Supervisor-only or User access to the GPIB-1014. A jumper on the GPIB-1014 is used to automatically determine the state of this bit (SUP) after a system or local reset. After reset, you can write a 1 to this bit to select Supervisor-only access or a 0 to select User access.

- Controlling the GPIB-1014 LED SYSFAIL* indicator. If this bit (SFL) is a 0, the VMEbus SYSFAIL* line is driven low with an AS756 open collector driver and the LED on the GPIB-1014 turns red. Writing a 1 to this bit releases the VMEbus SYSFAIL* line and turns the LED green. This bit is set to 0 on SYSRESET* and is unaffected by a local master reset (LMR).

Timing State Machine

The Timing State Machine circuitry is designed to control the timing of local signal LOCAL_DTACK* in slave cycles and the RD* and WR* signals, which control the TLC in slave and DMA cycles. A 74LS74A flip-flop, two 74LS393 4-bit counters, a 74S139 decoder, and some miscellaneous logic gates are used to implement a state machine to control this timing.

Slave Cycles

If the GPIB-1014 is functioning as a VMEbus slave, and the DMAC has been addressed, the VMEbus signal DTACK* is driven by the DMAC, and the timing is controlled entirely by the DMAC. If the TLC or one of the Configuration Registers has been addressed, however, the timing is controlled by the state machine.

The timing control begins when one of the signals TLCCS*, CS1*, or CS2* is low, indicating that the TLC, CFG1, or CFG2 is selected. During a read cycle to the TLC, the TLC RD* signal is immediately driven low. The circuitry then counts a minimum delay of 250 nsec, which

corresponds to the read access time of the TLC. Local signal SACK is asserted to drive VMEbus signal DTACK* active to indicate that the data is valid on the VMEbus data lines D07 through D00. The data remains valid and the DTACK* signal remains asserted until DS0* is released by the bus master. When DS0* is released, the board first disables the data transceiver and then releases DTACK*. At the same time, the circuitry delays for a recovery time of 250 nsec. Signal END_RECOVERY is asserted at the end of the 250-nsec period to reset the Timing State Machine. At this point the TLC can perform another read/write cycle.

During a write cycle to the TLC, CFG1, or CFG2, the timing is similar. Once DS0* is asserted, the TLC signal WR* is driven low immediately. The WR* signal is ANDed with both CS1* and CS2* to drive the clock inputs of the two Configuration Registers. Once WR* is driven low, the circuitry counts a data setup time of 250 nsec. SACK then is asserted to drive the VMEbus signal DTACK* low. At the same time, the WR* signal is immediately driven high to latch the data into the TLC or the Configuration Registers. As soon as the master has released DS0*, the board will release DTACK*. At the same time, the circuitry will start a recovery cycle of 250 nsec. Notice that recovery cycle is necessary to prevent another read or write cycle to the TLC.

DMA Cycles

The Timing State Machine also controls the timing of the TLC RD* and WR* signals during a DMA operation. The sequence is similar to a normal TLC access, except that the 74S139 decoder is used to drive the TLC RD* and WR* signals as required to effect the DMA transfer. The DMAC begins a DMA transfer for the TLC by driving either ACK0* or ACK1* low. The Timing State Machine also starts 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 then gates 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 data lines with correct data until the memory drives the DTACK* line. When DMAC has received DTACK* low, it first releases its data strobes (UDS* and LDS*), then acknowledge lines (ACK0* or ACK1*). At this time, the TLC RD* line is released. The timing circuitry then counts a recovery time of 250 nsec while the DMAC and VMEbus memory finish the cycle.

When the DMA transfer is from VMEbus memory to the TLC, the sequence is similar. The timing circuitry starts when ACK0* or ACK1* 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. 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 ACK0* or ACK1*, the timing circuitry counts a recovery time of 250 nsec while the DMAC and memory finish the cycle. Notice the function of the flip-flop in preventing another read/write cycle to the TLC during the recovery period. The flip-flop stays set (Q=1) at all times before the recovery cycle, is cleared (Q=0) at the beginning of the recovery cycle, and is set again at the end of the recovery period.

DMA Gating and Control

The DMA Gating and Control circuitry is designed to control the DMA request/acknowledge interface between the DMAC and the TLC. The circuitry consists of an LS74 flip-flop and miscellaneous logic gates to generate the DMA request signals (DREQ0* and DREQ1*), the carry cycle byte (CCBYTE) strobe, and the DMA Acknowledge Enable signal (DACKEN*). The main function of this circuitry is to gate the DMA request signal from the TLC (DMAREQ) to the proper DMA channel of the DMAC to perform the carry cycle function. Upon RESET, the DMA requests from the TLC are directed to Channel 0. This configuration remains unchanged unless a carry cycle DMA cycle is specified in CFG1 (CC=1). The DMA requests are gated to Channel 0 until the DMAC DONE* signal is active during a DMA transfer, indicating that Channel 0 has completely finished (that is, all the blocks in the chain have been transferred). 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). When the DMAC answers the request on Channel 1 with ACK1* asserted, CCBYTE is asserted to indicate that the carry cycle byte transfer is in progress. During the carry cycle byte transfer, signal DACKEN* is kept high to prevent the circuitry from asserting DMAACK*. In every DMA transfer (except carry cycle byte transfer), DACKEN* causes DMAACK* to go low to acknowledge a DMA request from the TLC. The TLC can request DMA during the carry cycle transfer, but will not be acknowledged until the carry cycle has completed. In addition, since Channel 0 has stopped, DREQ0* can be asserted by the circuitry but has no effect.

The first byte to be transferred from Channel 1 is the carry cycle byte. This byte must be transferred by the DMAC to the TLC auxiliary register. Since the TLC does not allow DMA transfers to the auxiliary register, discrete circuitry must make this cycle appear as a normal write cycle to the TLC. Furthermore, the circuitry must ensure that this byte is always transferred from memory to the TLC even though the DMAC may be configured to transfer the data from the GPIB to VMEbus memory. The circuitry knows that the first byte to be transferred from Channel 1 is always the carry cycle byte, so when ACK1* is driven low at the start of the DMA cycle, the onboard signal CCBYTE becomes active. This signal is ORed with the DMAC WR* signal to ensure that the onboard circuitry performs a read from VMEbus memory and a write to the TLC.

Signal CCBYTE is also used to gate the TLC register select lines RS2 through RS0 to drive them as needed to address the TLC auxiliary register, and to drive the TLC CS* line instead of the DMAACK* line. As mentioned earlier, DACKEN* is kept high during the carry cycle byte to prevent DMAACK* from asserting. The Timing State Machine circuitry and the TLC interpret the cycle as a normal write to the TLC, while the VMEbus memory interprets the cycle as a DMA read from memory. The DMAC signal ACK1* goes high at the end of the transfer. This low-to-high transition is used to set a flip-flop, which gates further DMA requests from the TLC to DMAC Channel 1 and disables generation of the CCBYTE signal. When the TLC issues another DMA request, Channel 1 responds by transferring the last byte of data with a normal DMA transfer cycle. The low to high transition of ACK1* at the end of this cycle clears the flip-flop, which masks any further DMA requests from the TLC. Again, the flip-flop state determines if a carry byte cycle has been completed. It is cleared (Q=0) at all time before and during the carry cycle byte and set (Q=1) at the end of the carry cycle byte. The output of the flip-flop is also used to generate signal ALLDONE, 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 CFG1 is 0, and DMA Gating and Control circuitry directs all DMA requests from the TLC to DMAC Channel 0. DMAC Channel 1 is not used and must not be started.

Interrupter

The GPIB-1014 requests service by using the Interrupter circuitry. The circuitry consists of the following:

- A 74145 open-collector 4-bit decoder
- Two 74F74 flip-flops
- A 74F85 4-bit magnitude comparator
- A 2-tap digital delay line
- Miscellaneous gates

The interrupt priority of the GPIB-1014 is software-selectable via three bits in CFG1. These bits are connected to the 74145 and 74F85. Since all interrupts from the GPIB-1014 are routed through the DMAC, the IREQ* pin of the DMAC is connected to the 74145. Seven of the 74145 outputs are connected directly to the seven VMEbus interrupt request lines IRQ1* through IRQ7*. When the DMAC drives its IREQ* pin low, the VMEbus interrupt request line corresponding to the interrupt priority code in CFG1 is driven low.

When the GPIB-1014 detects an interrupt acknowledge cycle (AS*, DS*, and IACK* all low), the 74F85 compares VMEbus address lines A3 through A1 against the 3-bit interrupt priority code in CFG1. If IACKIN* is asserted and the indicated priority does not match the GPIB-1014 priority, the daisy-chain signal IACKOUT* is asserted. This signal remains asserted until AS* is released. However, if the priority indicated matches the GPIB-1014 priority, the IACK* pin of the DMAC is driven low. The DMAC then finishes the acknowledge cycle by placing the contents of Channel 1 NIVR (EIVR if a DMA error was encountered) on VMEbus data lines D07 through D00 and driving the DTACK* line. An F74 flip-flop keeps the IACK* line of the DMAC continuously asserted until the interrupt handler releases DS*. When IACK* is released, the DMAC then releases the data transceiver and DTACK*. Notice the use of a 2-tap digital delay line in the circuitry. The first delay is used to give the flip-flop enough data to clock setup time and the second is used to prevent fluctuation in the flip-flop output from momentarily asserting signals DIACK* or IACKOUT*.

DTB Requester and Controller

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. If the feature is enabled, the GPIB-1014 interface board, once received and finished using the bus, simply holds control and releases the bus only when there

are some external requests for the bus. While the board is holding the bus and the DMAC requests the bus, the DMAC is immediately granted the bus, thus avoiding bus arbitration time.

The circuitry consists of various components to drive and receive VMEbus signals BBSY*, BR0* through BR3*, BG0IN* through BG3IN*, and BG0OUT* through BG3OUT*. An S139 2- to 4-bit decoder is used to select one of the four VMEbus Request/Grant lines. In addition, the circuitry uses three flip-flops to keep status of the bus arbitration process (that is, bus request pending, bus grant received, and bus release) and uses miscellaneous logic to generate various combinational outputs. The operation and output of each flip-flop (Q1, Q2, and Q3) is shown in Figure 6-1.

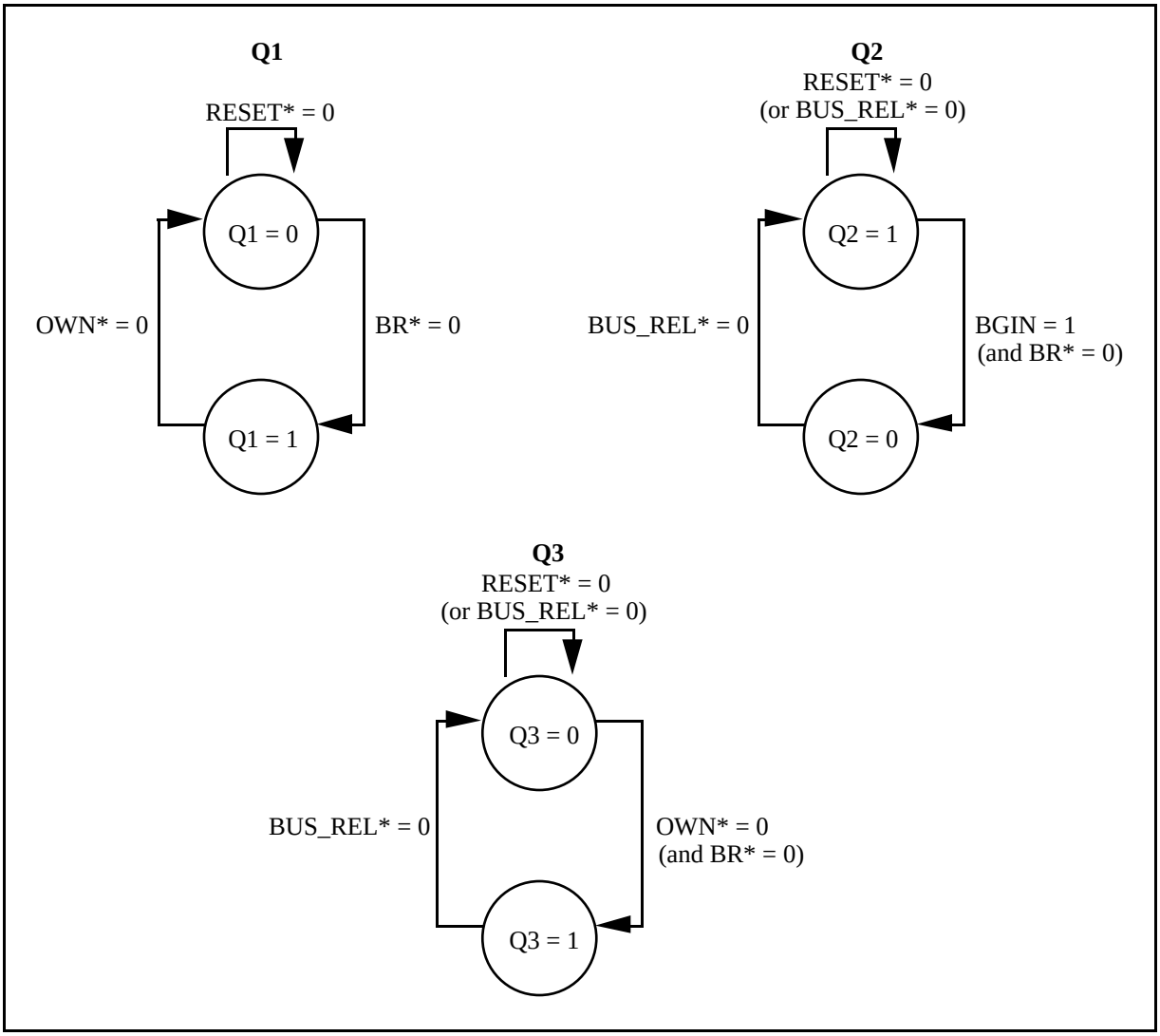


Figure 6-1. DTB Requester and Controller Flip-Flop Operations

In Figure 6-1, BR* is asserted by the DMAC to request the use of the VMEbus. BGIN is asserted by the DTB Requester and Controller circuitry when a correct bus grant signal is

received. OWN* is asserted by the DMAC to indicate that it now has ownership of the bus. BUS_REL* is asserted by the DTB Requester and Controller circuitry to indicate that it is going to release the VMEbus. RESET* is asserted to reset all circuitry on the board.

Based on the output of the flip-flops, there are numerous combinational outputs. The following equations contain these outputs:

Equation 1: LBROUT* = Q3 + Q1'

Equation 2: BG* = (BUS_REL* & Q1 & (MY_BGIN + OWNBUS))'

Equation 3: BGACK* = BBSY* + BGIN'

Equation 4: BGMATCH* = Q2 & BGIN

Equation 5: OWNBUS* = Q3'
BBSY* = Q3'

Equation 6: BUS_REL* = [(ROR*+BRIN) & LBROUT* & OWN* & BR* & DEN* & BGIN']'

In Equation 1, combinational output LBROUT* is asserted if DMAC has a bus request pending and the board does *not* have control of the VMEbus. While the GPIB-1014 is holding the bus and there is a request from the DMAC, LBROUT* (and thus BGIN) will not be asserted. BG*, an input to the DMAC, is asserted to inform the DMAC that it has been granted the bus. Even with its BG* asserted, the DMAC will not assert OWN* and start DMA cycle until BGACK* is released (unasserted). Equation 3 indicates that if the board has been granted the bus (BGIN asserted), BGACK* is released as soon as BBSY* is released by another master on the VMEbus. While the GPIB-1014 is holding the bus, BGACK* will be released at all times. If a pending DMAC bus request is acknowledged with a BGIN, BGMATCH* is asserted to maintain the appropriate BGXOUT* high. When the GPIB-1014 does not request for the bus, BGMATCH* is unasserted to pass BGXIN* directly to BGXOUT* (through the daisy chain). OWNBUS* and BBSY* are both asserted when the GPIB-1014 is using control or is holding control of the bus. If the Release On Request feature is not enabled (ROR*=1), BUS_REL* is asserted to release the bus right after the DMAC has finished with its cycle and released OWN*. If the ROR feature is enabled (ROR*=0) and the board is simply holding the bus (OWN*, BR*, and LBROUT* all released), the board releases the bus as soon as there is an external bus request (BRIN asserted). Finally, BUS_REL* is asserted to release the VMEbus after the board has released the data bus and the arbiter has released its BGx*.

A typical bus arbitration process is described as follows:

1. The DMAC indicates that it needs the bus for a DMA transfer by driving its BR* pin low.

2. Two bits in CFG1 are used to select one of four VMEbus Bus Request/Grant lines. A 74S139 decoder is used to decode these two bits.

3. Because the board does not have control of the bus, LBROUT* is driven low.

4. The outputs of the 74S139 are connected to four 74LS02 gates, along with the LBROUT* signal, to assert one of the four VMEbus bus request lines (BR3* through BR0*).

5. The DTB Requester waits for the appropriate Bus Grant In line (BG3IN* through BG0IN*) to become active, at which time BGIN becomes high.

6. The 74S139 outputs are used to direct the others to the corresponding Bus Grant Out line (BG3OUT* through BG0OUT*).

If BGIN is high and the DMAC has *no* bus request pending, BGMATCH* is not asserted (BUS_REL* is asserted to prevent flip-flop Q2 to change state) and the appropriate Bus Grant Out line is driven low. This line is released when the Bus Grant In line is released. If, however, the DMAC has a bus request pending, flip-flop Q2 changes state and BG* is asserted to inform the DMAC that it has been granted the bus. BGMATCH* is also asserted to maintain the BGXOUT* high. As soon as the VMEbus signal BBSY* is detected high, the DMAC asserts its signal OWN* to start the DMA cycle. (BBSY* and OWNBUS* are both asserted). Notice that BGIN is delayed 25 nsec by a digital delay line before its output is used to create onboard signals BGMATCH* and BG*. This delay prevents fluctuation on Q2 output from momentarily asserting BGMATCH* or BG*.

When the DMAC finishes the DMA transfer and wishes to relinquish the bus, it first unasserts its OWN* signal (at this time both LBROUT* and BR* are high). The DTB Requester then releases the VMEbus BBSY* immediately if the Release On Request feature is *not* enabled (ROR*=1 & BUS_REL*=0). If the DMAC operates in cycle steal mode, it releases OWN* immediately after it has finished the DMA transfer. In contrast, if the DMAC operates cycle steal with hold mode, it keeps asserting OWN* during the hold period and releases OWN* if the onboard TLC does not request DMA within the hold time. If the Release On Request feature is enabled in CFG1 (ROR*=0), however, the DTB Requester circuitry monitors the four VMEbus bus request lines (BR3*-BR0*). If none of these lines is logic zero, indicating that no other device is requesting the bus (BRIN=0), the DTB Requester simply holds the bus by continuing to drive the BBSY* line low, even when the OWN* signal of the DMAC is unasserted. If any of the VMEbus Bus Request lines are driven low (BRIN=1) before the DMAC indicates that it needs the bus again, BUS_REL* is asserted and the circuitry relinquishes the system bus by releasing the BBSY* line. The DTB Requester performs the entire bus arbitration protocol the next time the DMAC requests the bus. If, however, the DMAC requests the bus while the DTB Requester still has control of the bus, the DMAC is immediately granted the bus and the bus arbitration overhead is avoided.

GPIB Synchronization and Interrupt Control

This circuitry is designed to allow the GPIB-1014 to detect GPIB synchronization and control interrupts routed to the PCL input line of DMAC Channel 1. Using the GPIB synchronization circuitry, the GPIB-1014 can 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 that the DMA transfer is complete and does not have to timeout to ensure that the GPIB is synchronized.

The interrupt control circuitry consists of three inputs from the NOR gate and several other miscellaneous gates. The output of the NOR gate is tied to the PCL input line of Channel 1.

This PCL is used to detect interrupts from the GPIB-1014 that are not internal to the DMAC. A negative transition on the PCL sets the PCT bit in the CSR of DMAC Channel 1. If interrupts are enabled in the CCR of Channel 1 (EINT=1), the setting of the PCT bit causes the DMAC to drive its IREQ* line, requesting an interrupt. The three inputs of the NOR gate enable the negative transition to occur if the TLC requests an interrupt, if a bus error occurs while the DMAC is the bus master, or when the GPIB becomes synchronized after the last DMA transfer has finished.

The INT pin of the TLC is connected to one of the inputs of the NOR gate. Each of the TLC interrupt request events or conditions are controlled by mask bits located within the TLC. Each of the 13 events is individually enabled by bits in the mask registers IMR1 and IMR2. If the TLC detects an interrupt condition, it drives its INT pin high. This action causes a negative transition on DMAC Channel 1 PCL, which sets the PCT status bit and generates an interrupt if interrupts are enabled. The interrupt condition can be detected by polling the TLC interrupt status registers. Reading from the appropriate status register releases the TLC INT line and enables 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.

The GPIB synchronization circuitry consists of two LS74A flip-flops and various logic gates. The circuitry detects GPIB synchronization after the last byte in a DMA transfer has been transferred to or from the TLC. This is accomplished by monitoring the DAV* line on the GPIB. The circuitry begins when the ALLDONE signal becomes high, which occurs during the last data byte transfer in the DMA transfer sequence. If the carry cycle was *not* enabled (as indicated by the CC bit in CFG1), the last byte is detected by DONE* and ACK0*, which are both active at the same time. DMAC asserts DONE* when Channel 0 is transferring the last data byte in the last data block in Channel 0 chain. ALLDONE is driven high during this time. If a carry cycle 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 ACK1* asserted) make signal ALLDONE high. After this last data transfer, DMAREQ from TLC is masked (see *DMA Gating and Control* earlier in this chapter) and Channel 1 is still active (see Chapter 5, *Programming Considerations*); thus, Channel 1 cannot reach its terminal count. In both cases, the positive transition (0 to 1) in ALLDONE causes the GPIB synchronization circuitry to begin monitoring the GPIB DAV* line.

The GPIB synchronization is detected differently depending upon the direction of the DMA transfer. The direction of the transfer is specified in the DIR bit in CFG1. If the transfer is from the VMEbus memory to the GPIB (DIR = 0), the ALLDONE signal indicates that the last byte has just been transferred to the TLC. The DAV* line at this time is high. The TLC drives DAV* low as it places the data on the GPIB. As soon as all Listeners have accepted the byte, the TLC releases DAV* to high again. It is this positive transition of the DAV* line that is of interest. A D-type flip-flop, with the DAV* line as the clock input, is set by the DAV* low-to-high transition. This generates a negative transition on the PCL of DMAC Channel 1.

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 signal indicates that the last byte in the DMA transfer has just been transferred from the TLC. 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 the PCL of Channel 1, requesting an interrupt.

For the GPIB-1014 to wait until the GPIB is synchronized to signal that the DMA transfer is complete, it must be configured to wait until the PCL transition occurs to notify the host CPU. Each DMA channel of the DMAC has one interrupt enable, which generates an interrupt not only for the PCL transition, but also if the COC bit is set in the channel CSR. The COC bit (Channel Operation Complete) is set when the channel has finished its DMA transfer. If the channel was operated in the chained mode (array or linked chained), COC will be set after the DMAC has transferred the last data byte in the last data block of the chain. The condition of interest is not just when the DMAC is finished, but when the DMAC is finished and the GPIB is synchronized (that is, two conditions must be satisfied). Channel 1 is used to detect synchronization for both carry cycle and non-carry cycle DMA transfers. Since non-carry cycle DMA transfers do not use Channel 1, synchronization detecting on Channel 1 causes no conflict. Since carry cycle DMA transfers briefly employ Channel 1, their operations require special considerations.

If the carry cycle feature is *not* enabled, Channel 0 transfers everything. It is desirable to disable interrupts on Channel 0, but enable interrupts on Channel 1. A negative transition on the PCL of Channel 1 occurs if the GPIB is synchronized requesting an interrupt (at this point Channel 0 is already complete) . Since Channel 0 can terminate its operation normally, you can simply check the COC and ERR bits in the CSR of Channel 0 to determine the success of the data transfer on Channel 0.

If you do use the carry cycle feature, however, Channel 1 must *not* terminate its operation and set the COC bit, as this termination will cause an interrupt immediately rather than waiting for the GPIB to become synchronized. This is why Channel 1 is configured to operate in chained mode with the length of the second (and also last) block in the chain set to two (one more byte than needed). This value is loaded in the DMAC Channel 1 transfer count register. After one byte is transferred (the last byte in the data buffer), DMA requests are automatically masked. Channel 1 is still active, expecting to transfer one more byte, but never sees another request, so the COC bit is never set. The interrupt is disabled on Channel 0 but enabled on Channel 1. The GPIB becoming synchronized, however, causes a negative transition on the PCL1, requesting an interrupt. Before another DMA transfer with the carry cycle is attempted, Channel 1 must be stopped by issuing a software abort by writing to the CCR with the SAB set.

After the last byte in a DMA transfer has been transferred to or from the TLC, writing any value to CFG1 clears the circuitry, which detects GPIB synchronization. If GPIB synchronization has already been detected, writing to CFG1 pulls the PCL high.

Regardless of whether the carry cycle feature was used, after a DMA transfer has completed, the following actions must be taken in the order shown:

1. Read ISR2 (ISR2 must be read before reading ISR1).

2. Read ISR1.

3. Clear IMR1.

4. Clear IMR2.

5. Write a value to CFG1 to release PCL1 line (perhaps use the same value as the last write to CFG1).

6. Write a software abort to Channel 0.

7. Check, then clear the COC and ERR bits in CSR0.

8. Write a software abort to Channel 1 (if a carry cycle is used).

9. Check, then clear the PCT and ERR bits in CSR1.

The third step can cause a transition on the PCL1 if the VMEbus BERR* line is driven low while the DMAC is bus master, indicating that the cycle cannot be completed. This condition causes the DMAC BEC1* line to go low, which causes the DMAC to terminate the cycle and sets the COC and ERR bits in the CSR of the active channel. Whether or not a carry cycle was used, a bus error also causes a transition on the PCL of Channel 1, requesting an interrupt on Channel 1. The host CPU can detect the error condition by examining the CSR of Channels 0 and 1.

68450 DMAC

The DMAC is a high-performance device and is very flexible. It provides four independent DMA channels, but the GPIB-1014 uses only Channels 0 and 1 to provide GPIB-to-memory DMA flyby transfers. 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 lines. During DMA operations, the GPIB-1014 functions as a full VMEbus master. The DMAC controls the direction of the GPIB-1014 data bus transceivers as needed to effect the transfer. Details of the DMAC operation, with emphasis on GPIB-1014 applications, are given in the following section.

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 completed when the DMAC reports the status of the operation. The following sections 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 specific details on how the DMAC controls the data transfer during DMA operations.

Device (TLC)/DMAC Communication. Communication between the TLC and the DMAC is accomplished mainly by two signals. Each of the four DMA channels has a DMA request input (REQ0* through REQ*3) and a DMA acknowledge output (ACK0* through ACK3*). The TLC requests service by first asserting its DMAREQ line, waits for its DMAACK* pin to assert, and finally waits for its RD* or WR* pins to assert before sending or receiving data. The DMA Gating and Control circuitry gates the DMAREQ of the TLC to REQ0* or REQ1* depending on if carry cycle function is used. Depending on which channel is being serviced, either the ACK0* or the ACK1* line is activated during transfers to or from the TLC. This line is used to tell the TLC its current DMA request is being serviced (the DMAACK* of the TLC is asserted). Flowthrough memory-to-memory transfers must use the automatic request mode. The ACK* line is not activated.

Each channel also has a peripheral control line (PCL0* through PCL3*). The function of this line is quite flexible, and is programmable via the Device Control Register (DCR). The DTYP bits of the DCR define what type of device is on the channel. If the DTYP bits are programmed to be a HMCS6800 device, the PCL definition is ignored, and the PCL is an Enable Clock input. If the DTYP bits are programmed to be a device with READY, the PCL definition is ignored and the PCL is used as a READY input. The PCL is active at all times when it is programmed as a Status input, Interrupt input, Ready input, or Enable input. When programmed as an Abort input, it is only active after the channel has been started. When programmed as a Status input, like in most GPIB-1014 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 (8 MHz), the PCT bit of the CSR is set. The setting of the PCT bit causes an interrupt, if interrupts were previously enabled, by setting the EINT bit in CCR. The GPIB-1014 uses the PCL of Channel 1 (PCL1*) to detect interrupts from the TLC, bus error, and GPIB synchronization conditions. In addition, the board uses the PCLs of Channel 0 (PCL0*) and Channel 2 (PCL2*) to detect the status of two GPIB signals: SRQ* and REN*. If they are used in a GPIB-1014 application, these two PCLs must be programmed as Status input or Interrupt input.

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 can 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-1014 GPIB applications, you must set the REQG bit to 10 to indicate that external REQ line will initiate a transfer. After selecting 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. Do 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.

The GPIB-1014 uses cycle steal mode (with or without hold) for GPIB DMA data transfers. 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 the ACK* line active to transfer the operand. If the XRM bits specify cycle steal without hold, the DMAC relinquishes the bus after each operand transfer. If the XRM bits specify cycle steal with hold, the DMAC retains 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 can be assigned. Each channel can communicate using the following protocols. For GPIB-1014 GPIB DMA data transfers, you must set the device type to 10 for *Device with ACK* for both Channel 0 and Channel 1.

DTYP: Device Type
00 = 68000-compatible device – dual addressing
01 = 6800-compatible device – dual addressing
10 = Device with ACK – single addressing
11 = Device with ACK and READY – single addressing
Dual Address Transfers can be used with 68000-compatible (VMEbus memory) and 6800-compatible devices that must be explicitly addressed. Because the address bus is used to address both the device and the memory, the data cannot be directly transferred to or from the memory (the source) because the device (receiver) also requires addressing. Instead, the data is transferred from the source to the DMAC and held in internal holding register(s). (This may take more than one bus cycle.) After that, one or more transfers between the DMAC and the destination are required to complete the operation.

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. In both cases, data is not temporarily kept in the internal holding register(s) of the DMAC.

Operands and Addressing. Three factors affect how the actual data is handled: device (destination) port size, operand (from source) size, 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 data bits that 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 or the memory width need not equal the operand 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 accesses the parts of the operand in a linear increasing sequence. The step between the addresses of the parts is two. The size of the parts is of course 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 address 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), Device Address Register (DAR), and the Base Address Register (BAR). The MAR is used in all operations since all operations are between memory and a device. This register is either initialized before the channel operation is started, or is loaded during chaining or continue operations, which are discussed later. The DAR is used to

address the device (VMEbus memory) in dual-address transfers. It is initiated before starting the channel operation. The BAR 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 (MTCR) and the Base Transfer Counter (BTCR). The MTCR is used in all operations to count the number of operands transferred in a block. It is decremented 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 BTCR is used for chaining and continue operations. Both the MTCR and the BTCR 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 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 is 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. 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. Illegal operations include selecting any of the operations marked *undefined, reserved*. If the DMA operation is dual-address (for memory-to-memory DMA), the DAR 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 MTCR should have been previously initialized. If the operation is chained, as to transfer multiple blocks of data, the BAR and/or the BTCR should have been previously initialized. For array-chained operation, both the BAR and BTCR should have been previously initialized. For linked chained operation, only the BAR is to be initialized.

The Continue Mode of Operation. The continue bit (CNT) can transfer multiple blocks in unchained operations. The CNT bit is set to continue the current channel operation. If an attempt is made to continue a chained operation, a configuration error is signaled. The BAR and BTCR should have been previously initialized. The continue bit can be set at the same time as the STR bit is set (to start a channel) or it can be set while the channel is still active. The operation timing error bit is signaled if a continuation is otherwise attempted. GPIB-1014 applications generally do not use the continue mode of operation.

Halt. The CCR has a halt bit that can suspend the operation of the channel. If this bit is set, a request can 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 a software abort bit (SAB) that can abort the current operation of the channel. 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 bit (EINT) that the channel can use 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), if desired. While the GPIB-1014 uses Channel 0 and 1, the interrupt is usually enabled in Channel 1 only (see Chapter 5, *Programming Considerations*).

Block Termination

At the end of the transfer of an operand, the DMAC decrements the MTCR. If this counter is decreased to the terminal count, the MTCR 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 has been completed, 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.

Multiple Block Operations. When the MTCR 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 MTCR.

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

1. First, 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 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 is 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 MTCR is loaded with a terminal count, the count error is signaled. The GPIB-1014 usually does not use continue operations for its GPIB DMA transfers.

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 that holds the addresses and transfer counts of the data blocks. The *address & 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 Base Address Register (BAR), and the number of entries in the array is in the Base Transfer Counter (BTCR)–that is, the BAR points to the address & transfer count array. Before starting any block transfers, the DMAC fetches 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 MTCR. After each chaining entry is fetched, the BTCR is decremented by one and the BAR is incremented by six to point to the next entry. When the BTCR reaches a terminal count, the chain is exhausted, and the entry just fetched determines the last block of the the channel operation. After this data block is transferred, the channel operation is completed and the COC bit will be set.

As described in Chapter 5, *Programming Considerations*, the array chaining mode is used for Channel 1 in the GPIB-1014 applications to implement the carry cycle feature. It is also used to transfer multiple blocks of data on Channel 0.

The address & 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 (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 6-2.

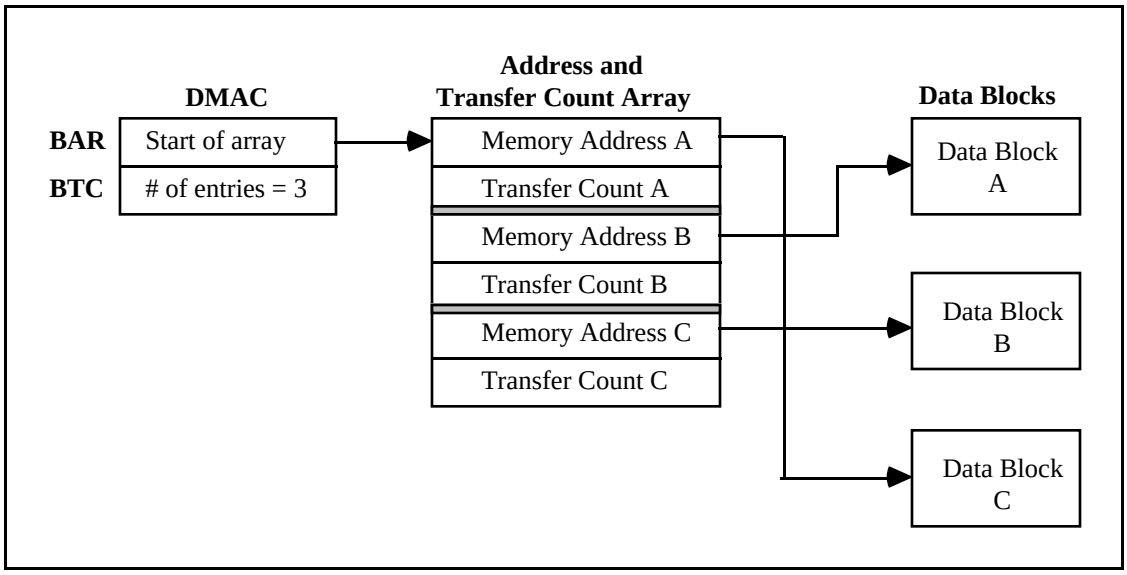


Figure 6-2. Array Format for Array Chaining Modes

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 address, transfer counts, and linked addresses. While array chaining requires a continuous address & transfer count array, the linked address in each entry allows a non-continuous address & transfer count array. Each entry in the chain list is 10 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 link address of the last array entry must be set to zero. Similar to array chaining, the BAR points to the first array entry, but the BTCR is unused. Before starting any block transfers, the DMAC fetches the entry currently pointed to by the BAR. The address information is placed in the MAR, the count information is placed in the MTCR, 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 updated 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 link 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 first array entry must be even. If a terminal count is loaded into the memory transfer counter, the count error is signaled.

While it is not currently implemented in the GPIB-1014 software, the linked chaining method could be used on Channel 0 to transfer large data blocks arbitrarily just like array chaining. Similarly, you can use linked chaining on Channel 1 to implement the carry cycle feature, but array chaining is much simpler. An example of the array format for linked chaining is shown in Figure 6-3.

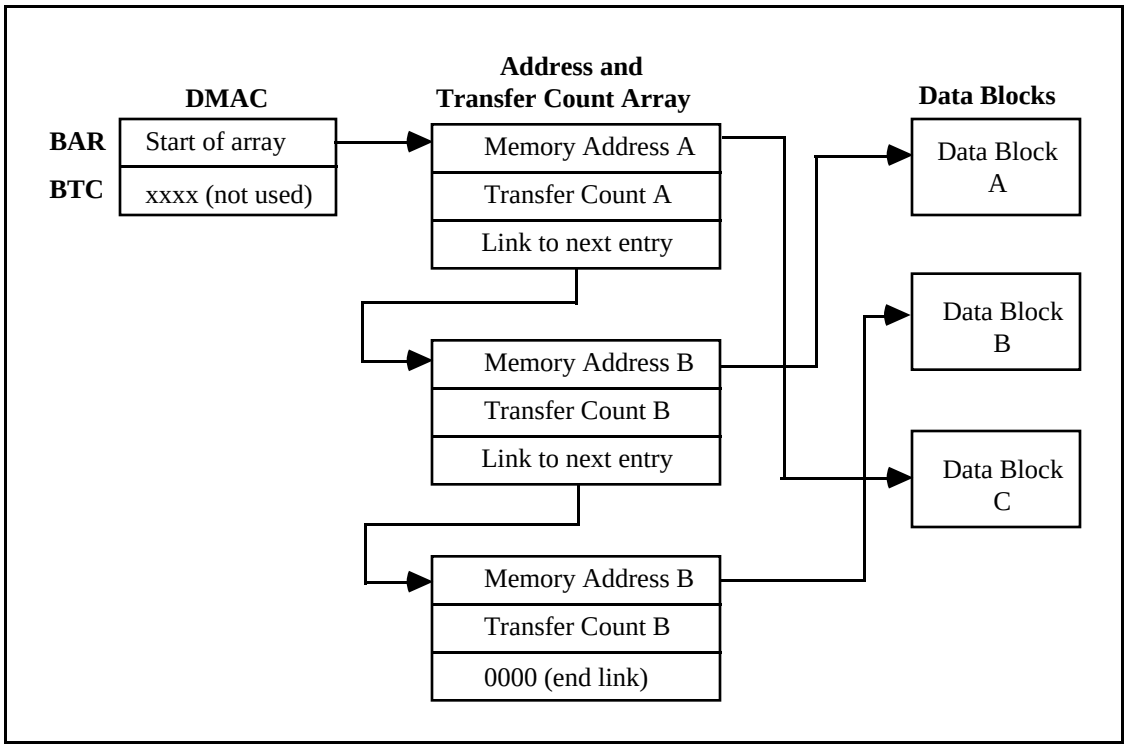


Figure 6-3. Array Format for Linked Chaining Modes

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, and the COC bit is set. The ERR bit of the CSR is also set, and 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.

Sources of errors are as follows:

- Configuration Error — This 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 occurs, 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 signals a configuration error. The undefined configurations are : XPM=01, MAC=11, DAC=11, CHN=01, SIZE=11. In any case, the channel will not start.

- 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 occurs and the channel operation aborts 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 MTCR 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 the channel.

- Count Error — This is signaled if the MTCR or BTCR 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.

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 for 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 Memory Address, Device Address and Base Address Registers, the Memory Transfer and Base Transfer Counters, and the Control, Status, and Error Registers. After the successful completion of any

transfer, the Memory Address and Device Address Registers point to the location of the next operand and the Memory Transfer Counter contains the number of operands yet to be transferred. If an error occurs during a transfer, that transfer has not completed and the registers contain the values they had before the transfer was attempted.

The DMAC logs 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 Interface

The GPIB-1014 is interfaced to the GPIB using an NEC μPD7210 Talker/Listener/Controller (TLC) large scale integrated circuit. Access to the TLC is through a block of 16 VMEbus addresses that access the internal TLC registers.

When the GPIB-1014 is operating as a VMEbus slave, the TLC is enabled (TLCCS* is asserted) when the base address of the GPIB-1014 has been decoded and VMEbus address bit A8 is a logic 1, and bit A4 is a logic 1. 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-1014 is acting as the VMEbus master, and the TLCCS* 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) signal and the TLC internal registers, 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 3-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 can be programmed to do these functions by the VMEbus master or it can 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 standard.

The ADSR is the primary register for monitoring the current status of the TLC; that is, to determine 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.

IMR1 and IMR2 are Interrupt Mask Registers for enabling and disabling the interrupt from the TLC on the occurrence of 13 key GPIB conditions or events. The status of these conditions can be read from the ISR1 and ISR2. 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 that 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-1014 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 operations when the GPIB-1014 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 CFG2, when set to a logic one, programs the 75162 transceiver to drive or receive the GPIB IFC* and REN* control lines. The data lines are always driven in the 3-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-1014 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-1014 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-1014 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 can 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 TLC, the 75160A, and the 75162A are functioning properly. Since most failures (including problems with shorts or open on the printed wire board) prevent the TLC from working at all, this test gives

limited assurance that the TLC and its associated circuitry are working and that the output signals can be manipulated properly.

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

DIO1* is the GPIB Data Input/Output bit 1 (LSB). By programming the TLC 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-1014 is installed and operating correctly. The tests are similar to those used by National Instruments to verify correct hardware functioning. The method used is to program specific internal functions by writing to one or more registers, then reading other registers to confirm that the functions were implemented. A user must have available an appropriate mechanism 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. DMA tests 11 through 14 can be done without using an external GPIB device.

Note: Most operations in the test are 8-bits read and write. Some registers inside the DMAC are 8 and 16 bits. To access these, users can use 8- or 16-bit R/W operations to consecutive memory accesses.

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-1014 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, 40 (hex) corresponds to a bit pattern of 01000000.

Equations not followed by a question mark are instructions to the user to load the value shown into the designated register. Equations followed by a question mark are instructions to the user 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 elemental levels of functioning first, and then progress to tests of higher complexity. For this reason, users are advised to perform the tests in the order given.

If the GPIB-1014 does not perform as described in the test procedures, users are advised to 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 jumper 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-1014 is and is not doing with regard to the expected results and contact National Instruments.

GPIB-1014 Hardware Installation Tests

1. Initialize TLC.

105	CFG2 = 0A	Set LMR and turn the LED green
105	CFG2 = 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	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 08	Clear LMR
111	DIR = 0?	(cleared?)
113	ISR1 = 0?	
115	ISR2 = 0?	
117	SPSR = 0?	
119	ADSR = 40?	
11B	CPTR = 0?	(cleared?)
0FF	GCR = 0?	
007	CCR0 = 0?	
047	CCR1 = 0?	
000	CSR0 = 01?	
040	CSR1 = 01?	
065	NIV1 = 0F?	
067	EIV1 = 0F?	

3. Test DMAC Channel Address Registers.

105	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 08	Clear LMR
00C	MAR0 = 5533	
00C	MAR0 = 5533?	
04C	MAR1 = 5533	
04C	MAR1 = 5533?	
01C	BAR1 = 5533	
01C	BAR1 = 5533?	

4. Test Local Master Reset.

105	CFG2 = 08	Clear LMR
065	NIV1 = 55	
065	NIV1 = 55?	
067	EIV1 = 55	
067	EIV1 = 55?	
105	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 08	Clear LMR
065	NIV1 = 0F?	
067	EIV1 = 0F?	

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

105	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 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, LA.

105	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 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, CO.

105	CFG2 = 0B	Set LMR and turn LED green
105	CFG2 = 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 DMA Error.

105	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 08	Clear LMR
007	CCR0 = 80	start channel 0
007	CCR0 = 10	abort channel 0
000	CSR0 = 91?	COC & Error
000	CSR0 = 90	clear bits
000	CSR0 = 01?	bits cleared

9. Test DMAC Interrupt Detection.

105	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 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 memory-to-memory (flowthrough) DMA transfer.

105	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 08	Clear LMR
080	CSR2 = FF	Reset Channel 2 status register
084	DCR2 = 08	Burst mode, 16-bit transfer with 68000 dev
085	OCR2=11	Mem to dev, word xtfr, no chain, internal req
086	SCR2=05	MAC and DAC both count up
08A	MTC2=0005	5 words (10 bytes) transfer
0A9	MFC2=06	
0B1	DFC2=06	
08C	MAR2=daddr	4-byte data address (Ex: 00200000)
094	DAR2=daddr+0A	4-byte data address (Ex: 0020000A)
101	CFG1=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	
087	CCR2=80	Start DMA on Channel 2
080	CSR2=81?	DMA completed without error? Bit 4 = 1 if error
	daddr+0A=FF?	Verify data values at destination locations
	daddr+0B=FE?	
	.....	
	daddr+13(hex)=F6?	

11. Test DMA transfer (flyby) to GPIB, one byte, memory read.

105	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 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	CFG1 = 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 & 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	CFG1 = 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 DMA transfer (flyby) from GPIB to memory, one byte, memory write.

105	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 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	CFG1 = 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	CFG1 = 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 DMA transfer (flyby) to GPIB, one byte, memory read, use the Carry Cycle feature.

    Addresses 3000 to 300E are used for this test, other locations may 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	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 08	Clear LMR
101	CFG1 = 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 DMA transfer (flyby) from GPIB to memory, one byte, memory write, use the Carry Cycle feature.

    Addresses 3000 to 300E are used for this test, other locations may 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	CFG2 = 0A	Set LMR and turn LED green
105	CFG2 = 08	Clear LMR
101	CFG1 = 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

Note: The carry cycle byte (81 = HLDA) was transferred to the TLC automatically by the GPIB-1014 after it transferred the first two bytes and before the last data byte

Appendix A
Hardware Specifications

This appendix specifies the electrical, environmental, and physical characteristics of the GPIB-1014 board and the conditions under which it should be operated.

Table A-1. Electrical Characteristics

Characteristic	Specification
Transfer Rates	
DMA	Over 500 kbytes/sec*
Programmed I/O	Over 80 kbytes/sec*
Power Requirement	
+5 VDC	1.6 A typical 2.0 A maximum
*Actual speed may vary considerably from those shown due to instrumentation capabilities.	

Table A-2. Environmental Characteristics

Characteristic	Specification
Operating Environment	
Component Temperature	0° to 70° C
Relative Humidity	10% to 90%, noncondensing
Storage Environment	
Temperature	-55° to 71° C
Relative Humidity	10% to 90%, noncondensing

Table A-3. Physical Characteristics

Characteristic	Specification
Dimensions	6.3 in. by 9.2 in.
I/O Connector GPIB-1014-1S GPIB-1014-2	 IEEE 488 Standard 24-pin VMEbus P2 connector

Appendix B
Parts List and Schematic Diagrams

This appendix contains the parts list and schematic diagrams for the GPIB-1014.

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.
- Timeout 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 8 MHz, 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 lists the multiline interface messages and describes the mnemonics and messages that correspond to the interface functions. These functions include initializing the bus, addressing and unaddressing devices, and setting device modes for local or remote programming. The multiline interface messages are IEEE 488-defined commands that 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	SPE	Serial Poll Enable
PPU	Parallel Poll Unconfigure	TCT	Take Control
SDC	Selected Device Clear	UNL	Unlisten
SPD	Serial Poll Disable	UNT	Untalk

Appendix E
Operation of the GPIB

This chapter describes the 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 that 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 eight 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.

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).

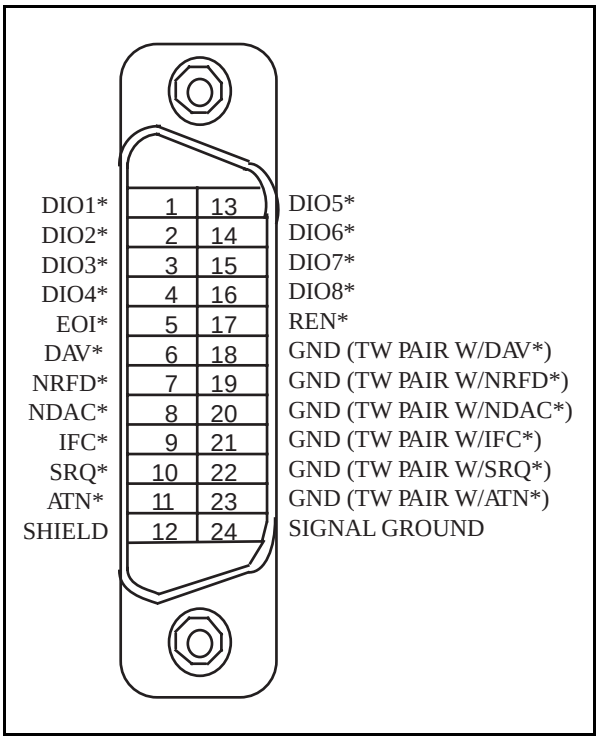


Figure E-1. The GPIB Connector and Signal Assignments

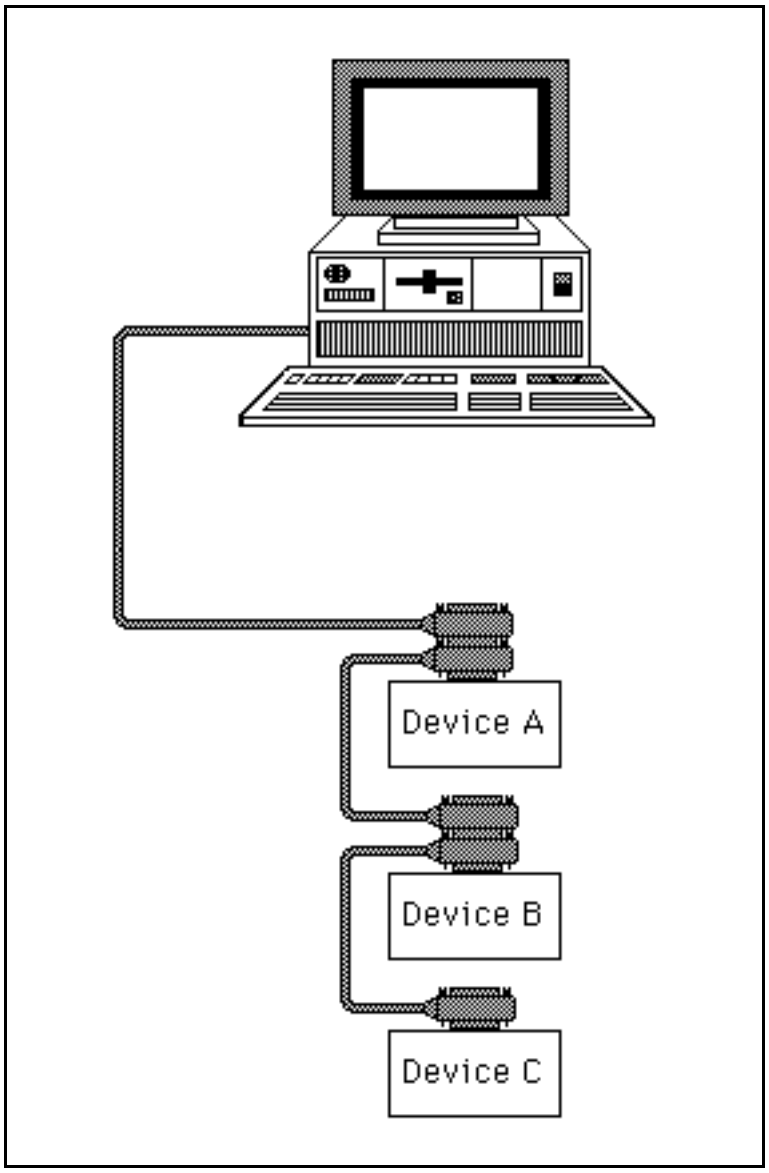


Figure E-2. Linear Configuration

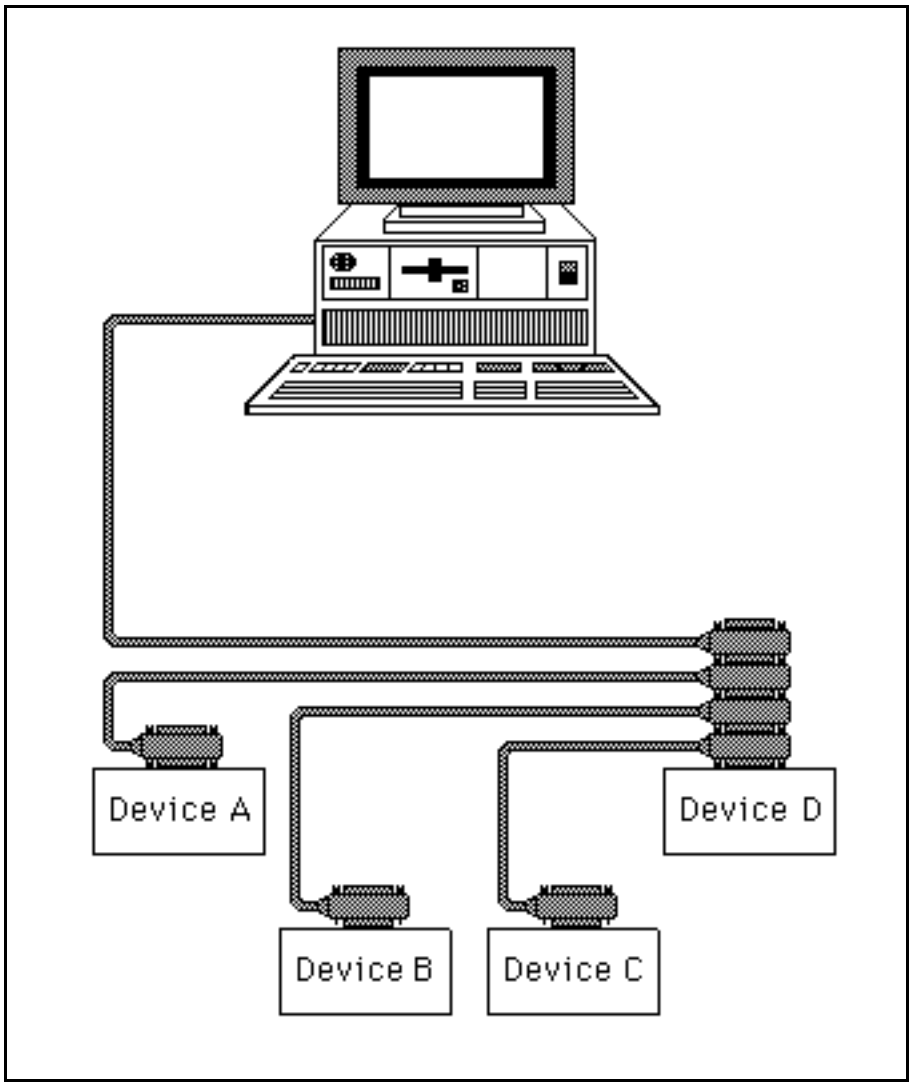


Figure E-3. Star Configuration

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 4 m between any two devices and an average separation of 2 m over the entire bus.
- A maximum total cable length of 20 m.
- 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 Documents

For more information on topics covered in this section, consult the following manuals:

- ANSI/IEEE Std 488-1978, *IEEE Standard Digital Interface for Programmable Instrumentation*
- ANSI/IEEE Std 488.1-1987, *IEEE Standard Digital Interface for Programmable Instrumentation*
- ANSI/IEEE Std 488.2-1987, *IEEE Standard Codes, Formats, Protocols, and Common Commands*
- *MC68450 Direct Memory Access Controller (DMAC)*; Motorola, Inc.

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	Definition
A		
A[01-31]	VBS	Address Lines 1 through 31
ACDS	ST	Acceptor Data State (AH function)
ACFAIL*	VBS	Power Fail Signal
ACG	RM	Addressed Command Group
ACK	LS	DMA Acknowledge Signal
ACRS	ST	Acceptor Ready State
ACT	B	Channel Active Bit
AD[5-1]	B	Talker/Listener/Controller (TLC) GPIB Address Bits 5 through 1
AD[5-0–1-0]	B	Mode 2 Primary TLC GPIB Address Bits 5 through 1
AD[5-1–1-1]	B	Mode 2 Secondary TLC GPIB Address Bits 5 through 1
ADCS	B	Addressed Status Change Bit
ADCS IE	B	Enable Interrupt on Addressed Status Change Bit
ADM[1-0]	B	Address Mode Bits 1 through 0
ADMR	R	Address Mode Register
ADO	VBO	Address Only Cycle
ADR	R	Address Register
ADR0	R	Address Register 0
ADR1	R	Address Register 1
ADSC	B	Address Status Change
ADSC IE	B	Address Status Change Interrupt Enable Bit
ADSR	R	Address Status Register
AH	ST	Acceptor Handshake
AIDS	ST	Acceptor Idle State
AM[0-5]	VBS	Address Modifier Lines
ANRS	ST	Acceptor Not Ready State
APRS	ST	Affirmative Poll Response State
APT	B	Address Pass Through Bit
APT IE	B	Enable Interrupt on Address Pass Through Bit
ARS	B	Address Register Select Bit
AS*	VBS	Address Strobe
ATN	ST	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		
BBSY*	VBS	Bus Busy
BCLR*	VBS	Bus Clear
BERR*	VBS	Bus Error
BG[0-3]IN*	VBS	Bus Grant In Lines

BG[0-3]OUT*	VBS	Bus Grant Out Lines
BIN	B	Binary Bit
BLT	VBO	Block Transfer
BR[0-3]*	VBS	Bus Request Lines

C

C	F	Controller
CACS	ST	Controller Active State (C function)
CADS	ST	Controller Addressed State
CAWS	ST	Controller Active Wait State
CDOR	R	Control/Data Out Register
CDO[7-0]	B	Control/Data Out Bits 7 through 0
CIC	B	Controller-In-Charge Bit
CIDS	ST	Controller Idle State
CLK[3-0]	B	Clock Bits 3 through 0
CNT	B	Continue Bit
CNT[2-0]	B	Control Code Bits 2 through 0
CO	B	Command Out
COC	B	Channel Operation Complete Bit
CO IE	B	Enable Interrupt on Command Output Bit
COM[4-0]	B	Command Code Bits 4 through 0
CPPS	ST	Controller Parallel Poll State
CPT	B	Command Pass Through Bit
CPT ENABLE	B	Command Pass Through Enable Bit
CPT IE	B	Enable Interrupt on Command Pass Through 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
CS*	LS	Chip Select Signal
CSBS	ST	Controller Standby State
CSHS	ST	Controller Standby Hold State
CSNS	ST	Controller Service Not Requested State
CSRS	ST	Controller Service Requested State
CSWS	ST	Controller Synchronous Wait State
CTRS	ST	Controller Transfer State (C function)

D

D[00-15]	VBS	Data Lines 0 through 15
D08(O)	VBO	Single Odd-Byte Transfers
D[16-31]	VBS	Data Lines 16 through 31
DAB	RM	Data Byte
DAC	RM	Data Accepted
dacr	LM	DAC holdoff release
DAV	RM	Data Valid
DC	F	Device Clear
DCAS	ST	Device Clear Active State
DCIS	ST	Device Clear Idle State
DCL	RM	Device Clear
DEC	B	Device Clear Bit
DEC IE	B	Enable Interrupt on Device Clear Bit
DEC RX	B	Device Clear Received

DEN*	LS	Data Enable
DET	B	Device Execute Trigger Bit
DET IE	B	Enable Interrupt on Device Execute Trigger Bit
DHDC	B	DAC Holdoff on DCAS
DHDT	B	Data Accepted Holdoff on Device Trigger Active State Bit
DI	B	Data In Bit
DI [7-0]	B	Data In Bits 7 through 0
DI IE	B	Enable Interrupt on Data In Bit
DIO[1-8]	GS	GPIB Data Lines 1 through 8
DIR	R	Data In Register
DL	B	Disable Listener Bit
DL0	B	Disable Listener 0 Bit
DL1	B	Disable Listener 1 Bit
DMA	SF	Direct Memory Access
DMAI	B	DMA Input Enable Bit
DMAO	B	DMA Out Enable Bit
DO	B	Data Out Bit
DO IE	B	Enable Interrupt on Data Out Bit
DS0*	VBS	Data Strobe Zero
DS1*	VBS	Data Strobe One
DT	F	Device Trigger
DT	B	Disable Talker Bit
DT0	B	Disable Talker 0 Bit
DT1	B	Disable Talker 1 Bit
DTACK*	VBS	Data Transfer Acknowledge
DTAS	ST	Device Trigger Active State
DTIS	ST	Device Trigger Idle State

E

END	RM	End
END IE	B	Enable Interrupt on End Received Bit
END RX	B	End Received Bit
EOI	B	End or Identify Bit
EOI	RM	End or Identify
EOI OE	LM	GPIB EOI Signal Output Enable
EOS	RM	End of String
EOS [7-0]	B	End of String Bits 7 through 0
EOSR	R	End of String Register
ERR	B	Error Bit
ERR	RM	Error
ERR IE	B	Enable Interrupt on Error Bit
EV*	LS	Enable Vector

F

FH	LM	Finish Handshake
FIN	LS	GPIB DMA Transfer Finished

G

GET	RM	Group Execute Trigger
GND	VBS	Ground
GTL	RM	Go To Local
gts	LM	Go to Standby

H

HLDA	B	Holdoff on All Bit
HLDE	B	Holdoff on End Bit

I

IA[1-3]	LS	Interrupt Priority Code Bits
IACK*	VBS	Interrupt Acknowledge Signal
IACKIN*	VBS	Interrupt Acknowledge In
IACKOUT*	VBS	Interrupt Acknowledge Out
IB[1-3]	LS	Interrupt Priority Code Bits
ICR	R	Internal Counter Register
IDTACK*	LS	Interrupt DTACK
IDY	RM	Identify
IFC	RM	Interface Clear
IMR1	R	Interrupt Mask Register 1
IMR2	R	Interrupt Mask Register 2
INT	B	Interrupt Bit
INTA	LS	TLC Interrupt Request Line A
INTB	LS	TLC Interrupt Request Line B
INV	B	Invert Bit
IR	LM	Interrupt Request
IRQ[1-7]*	VBS	Interrupt Request Lines 1 through 7
ISR1	R	Interrupt Status Register 1
ISR2	R	Interrupt Status Register 2
ISS	B	Individual Status Select Bit
ist	LM	Individual Status

L

L	F	Listener
LA	B	Listener Active Bit
LACS	ST	Listener Active State (L function)
LADS	ST	Listener Addressed State (L function)
LAG	RM	Listener Address Group
LD[0-7]	LS	Local Data Bus
LDTACK	LS	Local DTACK
LE	F	Listener Extended
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	Enable Interrupt on Lockout Change Bit
lon	B	Listen Only Bit
lon	LM	Listen Only
LPAS	B	Listener Primary Addressed State Bit
LPAS	ST	Listener Primary Addressed State
lpe	LM	Local Poll Enabled
lpe*	LM	Local Poll Enabled, Active Low
LPIS	ST	Listener Primary Idle State
ltn	LM	Listen
lun	LM	Local Unlisten
LWLS	ST	Local With Lockout State
LWORD*	VBS	Low Word

M

MAKOUT	LS	Internal IACKOUT* From A to B
MCYC	LS	My Cycle
MDTACK*	LS	DMA Acknowledge
MJMN	B	Major-Minor Bit
MLA	RM	My Listen Address
MSA	RM	My Secondary Address
MTA	RM	My Talk Address

N

nba	LM	New Byte Available
NDAC	RM	Not Data Accepted
NPRS	ST	Negative Poll Response State
NRFD	RM	Not Ready for Data
NUL	RM	Null byte

O

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
PCG	RM	Primary Command Group
PE	LM	Pull-up Enable
PEND	B	Pending Bit
pof	LM	Power Off
pon	LM	Power On
PP	F	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	RM	Parallel Poll Response
PPSS	ST	Parallel Poll Standby Active
PPU	RM	Parallel Poll Unconfigure
PUCS	ST	Parallel Poll Unaddressed to Configure State

R

RD*	LS	TLC Read Signal
rdy	LM	Ready for next message
REM	B	Remote Bit
REMC	B	Remote Change Bit
REMC IE	B	Enable Interrupt on Remote Change Bit
REMS	ST	Remote State
REN	RM	Remote Enable
REOS	B	End on End Of String Received Bit
RESET*	LS	Local Reset Signal
RFD	RM	Ready For Data
RL	F	Remote/Local
RMW	VBO	Read-Modify-Write
ROAK	VBO	Release on Register Access
ROR	VBO	Release on Request
RORA	VBO	Release on Register Access
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
RWD	B	Release When Done Bit
RWLS	ST	Remote With Lockout State

S

S	B	Status Bit Polarity (Sense) Bit
S8, S[6-1]	B	Serial Poll Status Bits 8 and 6 through 1
SACS	ST	System Control Active State
SCG	RM	Secondary Command Group
SDC	RM	Selected Device Clear
SDYS	ST	Source Delay State
SEOI	B	Send EOI
SERCLK	VBS	Serial Clock
SERDAT	VBS	Serial Data
SGNS	ST	Source Generate State
SH	F	Source Handshake
SIAS	ST	System Control Interface Clear Active State
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
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 End Or Indentify 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	Enable Interrupt on Service Request Input Bit
SRQS	ST	Service Request State
STB	RM	Status Byte
STRS	ST	Source Transfer State
STRT	LS	Start Cycle Signal
SWNS	ST	Source Wait for New Cycle State

SYSCLK*	VBS	System Clock
SYSFAIL*	VBS	System Fail
SYSRESET*	VBS	System Reset

T

T	F	Talker
TA	B	Talker Active Bit
TACS	ST	Talker Active State (T function)
TADS	ST	Talker Addressed State
TAG	RM	Talk Address Group
tca	LM	Take Control Asynchronously
tcs	LM	Take Control Synchronously
tcse	LM	Take Control Synchronously on End
TCT	TM	Take Control
TDMA	SX	Terminate DMA
TE	F	Extended Talk
TIDS	ST	Talker Idle State
TLC	IC	Talker/Listener/Controller (GPIB Adapter)
TLC CS*	LS	TLC Chip Reset
TLC WR*	LS	TLC Write
ton	B	Talker Only Bit
ton	LM	Talker Only
TPAS	B	Talker Primary Addressed State Bit
TPAS	ST	Talker Primary Addressed State
TPIS	ST	Talker Primary Idle State
TRI	B	Three-State Timing Bit
TRIG	LM	Trigger
TRM[1-0]	B	Transmit/Receive Mode Bits 1 through 0

U

U	B	Unconfigure Bit
UAT	VBO	Unaligned Transfer
UCG	RM	Universal Command Group
UDPCF	LM	Undefined Primary Command Function
UNL	RM	Unlisten command
UNT	RM	Untalk command

V

V[0-7]	LS	Interrupt Vector Bits

W

WR*	LS	TLC Write Signal
WRITE*	VBS	Read/Write Line

X

XEOS	B	Transmit End with End Of String Bit

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-1014 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-1014 User Manual**

Edition Date: **March 1997**

Part Number: **320030-01**

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

Glossary

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

≤	is less than or equal to
≥	is greater than or equal to
°	degrees
A	amperes
ANSI	American National Standards Institute
C	Celsius
FCC	Federal Communications Commission
GPIB	General Purpose Interface Bus
hex	hexadecimal
Hz	hertz
IEEE	Institute of Electrical and Electronic Engineers
in.	inches
K	kilobyte (1,024 bytes)
kbytes	1,000 bytes
m	meters
M	megabytes of memory
Mbytes	1,000,000 bytes
sec	seconds
V	volts
VDC	volts direct current

Index

Numbers

0 (Reserved Bit)
    Channel Error Register, 4-60
    Channel Priority Register, 4-61
    Channel Status Register, 4-59
    Configuration Register 2 (CFG2), 4-66
    Device Control Register, 4-52
    General Control Register, 4-63
    Operation Control Register, 4-53
    Sequence Control Register, 4-55
0 (Reserved Bits)
    Address Mode Register (ADMR), 4-22
    Auxiliary Register E (AUXRE), 4-41
    Internal Counter Register (ICR), 4-34
68450 DMAC. *See* DMAC (68450).

A

abbreviations used in the manual, *vi*
access mode, configuring, 3-3
ACT (Channel Active Bit), 4-59
AD5-0 through AD1-0 (Mode 2 Primary GPIB Address Bits 5-0 through 1-0), 4-42
AD5-1 (TLC GPIB Address Bit 5 through 1), 4-42
AD[5-1 -- 1-1] (Mode 2 Secondary TLC GPIB Address Bits 5-1 through 1-1), 4-44
address
    address decoder, 2-12
    base address configuration
        overview, 3-3
        setting with compare address lines, 3-4
        setting with jumper block W1, 3-4
    compare address lines, 3-3
    decoding, 6-3 to 6-4
    operands and addressing, DMAC channel operation, 6-17
    VMEbus address lines, 6-2
    VMEbus slave-addressing, 2-2 to 2-3
Address Mode Register (ADMR), 4-22 to 4-24
address modifier code output. *See* DMA address modifier code output.
Address Register (ADR), 4-43
Address Register 0 (ADR0), 4-42
Address Register 1 (ADR1), 4-44
Address Registers, DMA
    Base Address Register (BAR), 4-48
    Device Address Register (DAR), 4-48

Memory Address Register (MAR), 4-48
theory of operation, 6-17
Address Status Register (ADSR), 4-20 to 4-21
addressed implementation of Talker and Listener, 5-6 to 5-8
ADM[1-0] (Address Mode Bits 1 through 0), 4-23 to 4-24
ADMR. *See* Address Mode Register (ADMR).
ADR. *See* Address Register (ADR).
ADR0. *See* Address Register 0 (ADR0).
ADR1. *See* Address Register 1 (ADR1).
ADSC (Addressed Status Change Bit), 4-17 to 4-18
ADSC IE (Addressed Status Change Interrupt Enable Bit), 4-17 to 4-18
ADSR. *See* Address Status Register (ADSR).
AM code output. *See* DMA address modifier code output.
ANSI/IEEE Standard 1014-1987, 1-1
APT (Address Pass-Through Bit), 4-9 to 4-10
APT IE (Address Pass-Through Interrupt Enable Bit), 4-9 to 4-10
array chaining operations, 6-19 to 6-20
ARS (Address Register Select Bit), 4-43
ATN* (Attention* Bit), 4-20
ATN (attention) line, E-3
auxiliary commands
detailed description, 4-29 to 4-32
summary table, 4-28
Auxiliary Mode Register (AUXMR)
command summary (table), 4-28
detailed description (table), 4-29 to 4-32
overview, 4-27
Auxiliary Register A (AUXRA), 4-37 to 4-38
Auxiliary Register B (AUXRB), 4-39 to 4-40
Auxiliary Register E (AUXRE), 4-41

B

BAR. *See* Base Address Register (BAR).
base address configuration
selecting base address, 3-3
setting with compare address lines, 3-4
setting with jumper block W1, 3-4
Base Address Register (BAR), 4-48
Base Transfer Counter Register (BTCR), 4-48
BIN (Binary Bit), 4-37
BR (Bandwidth Available to DMAC Bits 1 through 0), 4-63
BRG (Bus Request/Grant Bits), 4-65
BT (Burst Transfer Time Bits 3 through 2), 4-63
BTC (Block Termination Complete Bit), 4-58, 4-62, 6-19
BTCR. *See* Base Transfer Counter Register (BTCR).
bus signals. *See* VMEbus.

C

cabling, 3-10
capability codes for GPIB-1014, 2-13 to 2-15
CC (Carry Cycle Bit), 4-65
CCR. *See* Channel Control Register (CCR).
CDO[7-0] (Command/Data Out Bits 7 through 0), 4-7
CDOR. *See* Command/Data Out Register (CDOR).
CER. *See* Channel Error Register (CER).
chaining operations
    array chaining operations, 6-19 to 6-20
    linked chaining operations, 6-20 to 6-21
Channel Control Register (CCR), 4-56 to 4-57
Channel Error Register (CER), 4-60
Channel Priority Register (CPR), 4-61
Channel Status Register (CSR), 4-58 to 4-59
Chip Reset command
    codes for, 4-28
    description, 4-29
CHN (Chain Bits 3 through 2), 4-53
CIC (Controller-In-Charge Bit), 4-20
Clear IFC command
    codes for, 4-28
    description, 4-32
Clear Parallel Poll Flag command
    codes for, 4-28
    description, 4-31
Clear REN command
    codes for, 4-28
    description, 4-32
CLK[3-0] (Clock Bits 3 through 0), 4-34
clock and reset circuitry
    definition, 2-12
    theory of operation, 6-4 to 6-5
CNT (Continue Bit), 4-56, 6-18
CNT[2-0] (Control Code Bits 2 through 0), 4-27
CO (Command Out Bit), 4-16
CO IE (Command Out Interrupt Enable Bit), 4-16
COC (Channel Operation Complete Bit), 4-58, 4-62
COM[4-0] (Command Code Bits 4 through 0), 4-27
COMMAND-CMD sample program, C-20
Command/Data Out Register (CDOR), 4-7
Command Pass Through Register (CPTR), 4-25 to 4-26
COMMAND SEND-CSEND sample program, C-19
commands
    auxiliary commands
        detailed description, 4-29 to 4-32
        summary table, 4-28

commands or command messages, E-1
multiline GPIB commands (table), 4-25 to 4-26, D-2 to D-3
compare address lines
location of, 3-3
setting base address, 3-4
compliance levels for GPIB-1014 IEEE 1014 interrupter, 2-15
configuration
access mode, 3-3
base address, 3-3 to 3-4
DMA address modifier code output, 3-5 to 3-7
hardware jumpers, 3-1
jumpers and switches (illustration), 3-2
other configuration parameters, 3-7
requirements, E-6
Supervisor or Non-privileged access, 3-3
Configuration Registers
Configuration Register 1 (CFG1), 4-64 to 4-65
Configuration Register 2 (CFG2), 4-66 to 4-67
definition of, 2-12
GPIB-1014 Configuration registers (chart), 2-5
register map, 4-2
theory of operation, 6-5 to 6-6
control equations of transceivers, 6-3
Controller function
becoming controller-in-charge (CIC) and active controller, 5-3
Controller-In-Charge (CIC) and System Controller, E-2
going from active to idle, 5-5
going from active to standby, 5-4
going from standby to active, 5-5
operation of, E-1 to E-2
sending remote multiline messages (commands), 5-4
CP (Channel Priority Bits 1 through 0), 4-61
CPR. *See* Channel Priority Register (CPR).
CPT (Command Pass-Through Bit), 4-8 to 4-9
CPT[7-0] (Command Pass Through Bits 7 through 0), 4-25 to 4-26
CPT ENABLE (Command Pass Through Enable Bit), 4-40
CPT IE (Command Pass-Through Interrupt Enable Bit), 4-8 to 4-9
CPTR. *See* Command Pass Through Register (CPTR).
CSR. *See* Channel Status Register (CSR).
Customer support, *vii*

D

DAC (Device Address Count Bits 1 through 0), 4-55
DAR. *See* Device Address Register (DAR).
Data In Register (DIR), 4-6
data lines
GPIB signals and lines, E-2

VMEbus, 6-3
data or data messages, E-1
DATA SEND-DSEND sample program, C-14 to C-16
data transfer bus (DTB) requester
description of, 2-7
VMEbus modules not provided, 2-7
data transfer features. *See also* DMA data transfers.
programmed I/O transfers, 2-8
throughput, 2-7 to 2-8
DAV (data valid) signal, E-3
DCL (Device Clear) command, 4-25
DCR. *See* Device Control Register (DCR).
DEC (Device Clear Bit), 4-11
DEC IE (Device Clear Interrupt Enable Bit), 4-11
DET (Device Execute Trigger Bit), 4-10
DET IE (Device Execute Trigger Interrupt Enable Bit), 4-10
Device Address Register (DAR), 4-48
Device Control Register (DCR), 4-51 to 4-52
DHDC (DAC Holdoff on DCAS Bit), 4-41
DHDT (DAC Holdoff on DTAS Bit), 4-41
DI[7-0] (Data In Bits 7 through 0), 4-6
DI (Data In Bit), 4-12 to 4-13
DI IE (Data In Interrupt Enable Bit), 4-12 to 4-13
DIR. *See* Data In Register (DIR).
DIR (Direction Bit), 4-53, 4-65
Disable System Control command
codes for, 4-28
description, 4-32
DL (Disable Listener Bit), 4-43
DL0 (Disable Listener 0 Bit), 4-42
DL1 (Disable Listener 1 Bit), 4-44
DMA address modifier code output
configuration, 3-5 to 3-7
default settings of AM code jumpers W3, W4 and W5, 3-5
programming values for default settings W3, W4 and W5, 3-6
setting AM code bits (AM5-AM0), 3-6
DMA cycles, Timing State Machine, 6-7
DMA data transfers
Dual Address Transfers, 6-16
overview, 5-8 to 5-9
polling during DMAs, 5-17
sending END or EOS, 5-17
Single Addressing Mode, 6-16
terminating on END or EOS, 5-19
terminating transfer and checking results, 5-17 to 5-19
theory of operation, 6-16
VMEbus interface, 6-1
with carry cycle, 5-13 to 5-17
without carry cycle, 5-10 to 5-12

DMA gating and control circuitry, 6-7 to 6-8
DMA registers
    68450 internal DMA registers (chart), 2-5
    Address Registers, 4-48
    Base Address Register (BAR), 4-48
    Base Transfer Counter Register (BTCR), 4-48
    Channel Control Register (CCR), 4-56 to 4-57
    Channel Error Register (CER), 4-60
    Channel Priority Register (CPR), 4-61
    Channel Status Register (CSR), 4-58 to 4-59
    Configuration Register 1 (CFG1), 4-64 to 4-65
    Configuration Register 2 (CFG2), 4-66 to 4-67
    Device Address Register (DAR), 4-48
    Device Control Register (DCR), 4-51 to 4-52
    DMAC DMA channel register set (chart), 4-46
    Function Code Registers, 4-50
    General Control Register (GCR), 4-63
    Interrupt Vector Registers, 4-62
    Memory Address Register (MAR), 4-48
    Memory Transfer Counter Register (MTCR), 4-48
    Operation Control Register (OCR), 4-53 to 4-54
    overview, 4-46 to 4-48
    register map, 4-2
    register memory map, 4-47
    Sequence Control Register (SCR), 4-55
    Transfer Count Registers, 4-48 to 4-49
DMAC (68450)
    definition of, 2-12
    initialization, 5-2
    theory of operation, 6-14 to 6-22
DMAC channel operation
    block termination
        array chaining operations, 6-19 to 6-20
        continued operations, 6-19
        error conditions, 6-21 to 6-22
        linked chaining operations, 6-20 to 6-21
        multiple block operations, 6-19
        overview, 6-19
    initialization and transfer phases
        address register operation, 6-17
        address sequencing, 6-17
        data transfers, 6-16
        device port size, 6-17
        device (TLC)/DMAC communication, 6-15
        DMA requests, 6-16
        operand size, 6-17
        operands and addressing, 6-17
        transfer count register operation, 6-17 to 6-18
    initiation and control of channel operations

continue mode of operation, 6-18
halt, 6-18
initiating the operation, 6-18
interrupt enable, 6-19
software abort, 6-18
overview, 6-15
DMAI (DMA Input Enable Bit), 4-16
DMAO (DMA Out Enable Bit), 4-16
DO (Data Out Bit), 4-12
DO IE (Data Out Interrupt Enable Bit), 4-12
documentation
abbreviations used in the manual, *vi*
related documents, *vi* to *vii*, E-7
don't care bits, 4-3. *See also* X (Don't Care Bit).
DPS (Device Port Size Bit), 4-51
DT (Disable Talker Bit), 4-43
DT0 (Disable Talker 0 Bit), 4-42
DT1 (Disable Talker 1 Bit), 4-44
DTB control transceivers, 2-12
DTB Requester and Controller
flip-flop operations (illustration), 6-10
theory of operation, 6-9 to 6-12
typical bus arbitration process, 6-11
DTYP (Device Type Bits 5 through 4), 4-51

E

EINT (Interrupt Enable Bit), 4-57, 4-62, 6-19
electrical characteristics. *See* physical and electrical characteristics.
END IE (End Received Interrupt Enable Bit), 4-10 to 4-11
End of String Register (EOSR), 4-45
END or EOS, sending/receiving, 5-17, 5-20
END RX (End Received Bit), 4-10 to 4-11
EOI (End or Identify Bit), 4-44
EOI (End or Identify) line, E-3
EOS[7-0] (End of String Bits 7 through 0), 4-45
EOSR. *See* End of String Register (EOSR).
ERR (Error Bit), 4-11 to 4-12, 4-59
ERR IE (Error Interrupt Enable Bit), 4-11 to 4-12
ERROR CODE, Channel Error Register, 4-60
error conditions, DMAC channel operation, 6-21 to 6-22
Execute Parallel Poll command
    codes for, 4-28
    description, 4-32

F

features of GPIB-1014, 1-1
Finish Handshake (FH) command
    codes for, 4-28
    description, 4-30
Function Code Registers, 4-50

G

General Control Register (GCR), 4-63
GET (Group Execute Trigger) command, 4-25
Go To Standby command
    codes for, 4-28
    description, 4-31
GPIB-1014
    block diagram, 2-11
    capabilities, 2-13 to 2-15
    contents of kit, 1-3
    definition, 1-1
    features of, 1-1
    functional description, 2-8 to 2-15
    IEEE 1014 compliance levels, 2-15
    illustration, 1-2
    major components of, 2-12
    in multiprocessor application (illustration), 2-10
    optional equipment, 1-3

parts list and schematic diagrams, B-1 to B-9
theory of operation
68450 DMAC, 6-14 to 6-23
address decoding, 6-3 to 6-4
clock and reset circuitry, 6-4 to 6-5
Configuration registers, 6-5 to 6-6
DMA gating and control, 6-7 to 6-8
DTB requester and controller, 6-9 to 6-12
GPIB interface, 6-23 to 6-24
GPIB synchronization and interrupt control, 6-12 to 6-14
interrupter, 6-9
test and troubleshooting, 6-24
timing state machine, 6-6 to 6-7
VMEbus interface, 6-1 to 6-3
with VMEbus computer (illustration), 2-9
GPIB Controller. *See* Controller function.
GPIB Synchronization and Interrupt Control
definition of, 2-12
theory of operation, 6-12 to 6-14
GPIB TLC. *See* Talker/Listener/Controller (TLC).
GTL (Go To Local) command, 4-25

H

handshake lines
DAV (data valid), E-3
NDAC (not data accepted), E-3
NRFD (not ready for data), E-2
overview, E-2
hidden registers
Auxiliary Register A (AUXRA), 4-37 to 4-38
Auxiliary Register B (AUXRB), 4-39 to 4-40
Auxiliary Register E (AUXRE), 4-41
Internal Counter Register (ICR), 4-34
overview, 4-33
Parallel Poll Register (PPR), 4-35 to 4-36
HLDA (Holdoff on All Bit), 4-38
HLDE (Holdoff on END Bit), 4-38
HLT (Halt Bit), 4-56, 6-18

I

ICR. *See* Internal Counter Register (ICR).
IEEE 488 standard
GPIB-1014 capabilities, 2-13 to 2-15
GPIB-1014 compatibility, 1-1
IEEE 1014 standard

GPIB-1014 compatibility, 1-1
GPIB-1014 compliance levels, 2-15
IFC (interface clear) line, E-3
Immediate Execute Pon command
codes for, 4-28
description, 4-29
IMR1. *See* Interrupt Mask Register 1 (IMR1).
initialization of GPIB-1014, 5-1 to 5-3
INITIALIZE-INIT sample program, C-5 to C-6
installation
cabling, 3-10
hardware installation tests, 7-2 to 7-8
prerequisites for, 3-1
unpacking the GPIB-1014, 1-4
verification of system compatibility, 3-7 to 3-9
verification testing, 3-10
INT (Interrupt Bit), 4-14 to 4-15
INTERFACE CLEAR-IFC sample program, C-7
interface management lines
ATN (attention), E-3
EOI (end or identify), E-3
IFC (interface clear), E-3
overview, E-3
REN (remote enable), E-3
SRQ (service request), E-3
interface registers
Address Mode Register (ADMR), 4-22 to 4-24
Address Register (ADR), 4-43
Address Register 0 (ADR0), 4-42
Address Register 1 (ADR1), 4-44
Address Status Register (ADSR), 4-20 to 4-21
Auxiliary Mode Register (AUXMR), 4-27 to 4-33
Command/Data Out Register (CDOR), 4-7
Command Pass Through Register (CPTR), 4-25 to 4-26
Data In Register (DIR), 4-6
End of String Register (EOSR), 4-45
hidden registers
Auxiliary Register A (AUXRA), 4-37 to 4-38
Auxiliary Register B (AUXRB), 4-39 to 4-40
Auxiliary Register E (AUXRE), 4-41
Internal Counter Register (ICR), 4-34
overview, 4-33
Parallel Poll Register (PPR), 4-35 to 4-36
illustration, 4-4
Interrupt Mask Register 1 (IMR1), 4-8 to 4-13
Interrupt Mask Register 2 (IMR2), 4-14 to 4-18
Interrupt Status Register 1 (ISR1), 4-8 to 4-13
Interrupt Status Register 2 (ISR2), 4-14 to 4-18
overview, 4-3

register map, 4-1
Serial Poll Mode Register (SPMR), 4-19
Serial Poll Status Register (SPSR), 4-19
writing to hidden registers, 4-4
Internal Counter Register (ICR), 4-34
interrupt control. *See* GPIB Synchronization and Interrupt Control.
Interrupt Mask Register 1 (IMR1), 4-8 to 4-13
Interrupt Mask Register 2 (IMR2), 4-14 to 4-18
Interrupt Status Register 1 (ISR1), 4-8 to 4-13
Interrupt Status Register 2 (ISR2), 4-14 to 4-18
Interrupt Vector Registers, 4-62
interrupter. *See also* GPIB Synchronization and Interrupt Control.
definition, 2-12
description of, 2-6
GPIB-1014 IEEE 1014 interrupter compliance levels, 2-15
programming considerations, 5-20 to 5-21
theory of operation, 6-9
INTRQ (Interrupt Request Bits 7 through 5), 4-64
INV (Invert Bit), 4-39
ISR1. *See* Interrupt Status Register 1 (ISR1).
ISR2. *See* Interrupt Status Register 2 (ISR2).
ISS (Individual Status Select Bit), 4-39

J

jumpers and switches
base address, 3-3 to 3-4
configuration, 3-1 to 3-2
DMA address modifier code output
default settings of AM code jumpers W3, W4 and W5, 3-5
programming values for default settings W3, W4 and W5, 3-6
setting AM code bits (AM5-AM0), 3-6
parts locator diagram, 3-2
for Supervisor or Non-privileged access, 3-3

L

LA (Listener Active Bit), 4-21
lines. *See* signals and lines.
linked chaining operations, 6-20 to 6-21
Listen command
codes for, 4-28
description, 4-31
Listen in Continuous Mode command
codes for, 4-28
description, 4-31 to 4-32
LLO (Local Lockout) command, 4-25

LMR (Local Master Reset Bit), 4-67
Local Unlisten command
    codes for, 4-28
    description, 4-32
LOK (Lockout Bit), 4-16
LOKC (Lockout Change Bit), 4-17
LOKC IE (Lockout Change Interrupt Enable Bit), 4-17
lon (Listen Only Bit), 4-22
LPAS (Listener Primary Addressed State Bit), 4-20

M

M0 (Program/Data Access Bit), 4-50
M1 (Standard/Short Addressing Bit), 4-50
M2 (Supervisor/User Access Bit), 4-50
MAC (Memory Address Count Bits 3 through 2), 4-55
MAR. *See* Memory Address Register (MAR).
master-direct memory access, VMEbus, 2-5 to 2-6
Memory Address Register (MAR), 4-48
Memory Transfer Counter Register (MTCR), 4-48
messages
    multiline interface command messages, D-1 to D-2
    types of, E-1
MJMN (Major-Minor Bit), 4-21
MLA (My Listen Address) command, 4-26
mnemonics for registers
    alphabetical list with definitions, F-1 to F-9
    clues to understanding, 4-3
MSA,PPD (My Secondary Address or Parallel Poll Disable) command, 4-26
MSA,PPE (My Secondary Address or Parallel Poll Enable) command, 4-26
MTA (My Talk Address) command, 4-26
MTCR. *See* Memory Transfer Counter Register (MTCR).
multiline GPIB commands (table), 4-25 to 4-26, D-2 to D-3
µPD7210 interface registers, 2-3

N

NDAC (not data accepted) signal, E-3
NDT (Normal Device Termination Bit), 4-58
Non-Valid Secondary Command or Address command
    codes for, 4-28
    description, 4-30
NRFD (not ready for data) signal, E-2

O

OCR. *See* Operation Control Register.
operands and addressing, DMAC channel operation, 6-17
operating environment, A-1
Operation Control Register (OCR), 4-53 to 4-54
optional equipment for GPIB-1014, 1-3

P

P[3-1] (Parallel Poll Response Bits 3 through 1), 4-36
Parallel Poll Register (PPR), 4-35 to 4-36
parallel polls
    conducting, 5-23
    overview, 5-22
    responding to, 5-24
parts list and schematic diagrams, B-1 to B-9
PASS CONTROL-PASSC sample program, C-21
PCL (Peripheral Control Line Bits 1 through 0), 4-52
PCS (Peripheral Control Status Bit), 4-59
PCT (Peripheral Control Transition Bit), 4-59, 4-62
PEND (Pending Bit), 4-19
physical and electrical characteristics
    description of, 2-1 to 2-2, E-4
    GPIB connector and signal assignments (illustration), E-4
    linear configuration (illustration), E-5
    specifications, A-1
    star configuration (illustration), E-6
pin assignments. *See* signals.
power requirement, A-1
PPC (Parallel Poll Configure) command, 4-25
PPR. *See* Parallel Poll Register (PPR).
PPU (Parallel Poll Unconfigure) command, 4-26
programmed I/O transfers
    overview, 2-8
    sending and receiving data, 5-19 to 5-20
    sending END or EOS, 5-20
    terminating on END or EOS, 5-20
programming
    Controller function
        becoming controller-in-charge (CIC) and active controller, 5-3
        going from active to idle, 5-5
        going from active to standby, 5-4
        going from standby to active, 5-4
        sending remote multiline messages (commands), 5-4
    initialization, 5-1 to 5-3
    interrupts, 5-20 to 5-21
    parallel polls, 5-22 to 5-24
    sample programs
        COMMAND-CMD, C-20

COMMAND SEND-CSEND, C-19
DATA SEND-DSEND, C-14 to C-16
GPIB-1014 Sample Functions for Driver, C-2 to C-4
INITIALIZE-INIT, C-5 to C-6
INTERFACE CLEAR-IFC, C-7
overview, C-1
PASS CONTROL-PASSC, C-21
READ, C-12 to C-13
RECEIVE-RCV, C-11
REMOTE ENABLE-REN, C-8
WRITE, C-17 to C-18
sending/receiving messages
DMA transfers with carry cycle, 5-13 to 5-17
DMA transfers without carry cycle, 5-10 to 5-12
overview, 5-8
polling during DMAs, 5-17
sending and receiving data, 5-19 to 5-20
sending END or EOS, 5-17, 5-20
terminating on END or EOS, 5-19, 5-20
terminating transfer and checking results, 5-17 to 5-19
using direct memory access, 5-8 to 5-9
using programmed I/O, 5-19 to 5-20
serial polls, 5-22
Talker and Listener
addressed implementation, 5-6 to 5-8

overview, 5-6
programmed implementation, 5-6

R

READ sample program, C-12 to C-13
RECEIVE-RCV sample program, C-11
receiving messages. *See* sending/receiving messages.
registers
Configuration registers
definition of, 2-12
DMA Configuration registers, 4-64 to 4-67
GPIB-1014 Configuration registers (chart), 2-5
DMA registers
68450 internal DMA registers (chart), 2-5
address registers, 4-48
Base Address Register (BAR), 4-48
Base Transfer Counter Register (MTCR), 4-48
Channel Control Register, 4-56 to 4-57
Channel Error Register, 4-60
Channel Priority Register, 4-61
Channel Status Register, 4-58 to 4-59
Configuration Registers, 4-64 to 4-67
Device Address Register (BAR), 4-48
Device Control Register, 4-51 to 4-52
DMAC DMA channel register set (chart), 4-46
Function Code Registers, 4-50
General Control Register, 4-63
Interrupt Vector Registers, 4-62
Memory Address Register (MAR), 4-48
Memory Transfer Counter Register (MTCR), 4-48
Operation Control Register, 4-53 to 4-54
overview, 4-46 to 4-48
register memory map, 4-47
Sequence Control Register, 4-55
transfer count registers, 4-48 to 4-49
format for description of, 4-3
interface registers
Address Mode Register (ADMR), 4-22 to 4-24
Address Status Register (ADSR), 4-20 to 4-21
Auxiliary Mode Register (AUXMR), 4-27 to 4-33
Command/Data Out Register (CDOR), 4-7
Command Pass Through Register (CPTR), 4-25 to 4-26
Data In Register (DIR), 4-6
hidden registers, 4-33 to 4-41
illustration, 4-4
Interrupt Mask Register 1 (IMR1), 4-8 to 4-13
Interrupt Mask Register 2 (IMR2), 4-14 to 4-18

Interrupt Status Register 1 (ISR1), 4-8 to 4-13
Interrupt Status Register 2 (ISR2), 4-14 to 4-18
µPD7210 internal GPIB interface registers (chart), 2-3
overview, 4-3
Serial Poll Mode Register (SPMR), 4-19
Serial Poll Status Register (SPSR), 4-19
writing to hidden registers, 4-4
mnemonics for, 4-3
register map, 4-1 to 4-2
size of, 4-2
terminology related to, 4-3
REM (Remote Bit), 4-16
REMC (Remote Change Bit), 4-17
REMC IE (Remote Change Interrupt Enable Bit), 4-17
REMOTE ENABLE-REN sample program, C-8
REN (remote enable) line, E-3
REOS (END on EOS Received Bit), 4-37
REQG (DMA Request Generation Bits 1 through 0), 4-54
reset. *See* system reset.
Return to Local (rtl) command
codes for, 4-28
description, 4-30
RFD Holdoff mode, 4-38
RFD (Ready for Data) message, 4-6
ROR* (Release On Request Bit), 4-65
rsv (Request Service Bit), 4-19

S

S (Status Bit Polarity Bit), 4-36
S[6-1] (Serial Poll Status Bits 6 through 1), 4-19
S8 (Serial Poll Status Bit 8), 4-19
SAB (Software Abort Bit), 4-56, 6-18
sample programs
COMMAND-CMD, C-20
COMMAND SEND-CSEND, C-19
DATA SEND-DSEND, C-14 to C-16
GPIB-1014 Sample Functions for Driver, C-2 to C-4
INITIALIZE-INIT, C-5 to C-6
INTERFACE CLEAR-IFC, C-7
overview, C-1
PASS CONTROL-PASSC, C-21
READ, C-12 to C-13
RECEIVE-RCV, C-11
REMOTE ENABLE-REN, C-8
WRITE, C-17 to C-18
SC (System Controller Bit), 4-67
schematic diagrams, B-1 to B-9

SCR. *See* Sequence Control Register.
SDC (Selected Device Clear) command, 4-25
Send EOI (SEOI) command
    codes for, 4-28
    description, 4-30
sending/receiving messages
    using direct memory access
        DMA transfers with carry cycle, 5-13 to 5-17
        DMA transfers without carry cycle, 5-10 to 5-12
        polling during DMAs, 5-17
        sending END or EOS, 5-17
        terminating on END or EOS, 5-19
        terminating transfer and checking results, 5-17 to 5-19
    using programmed I/O, 5-19 to 5-20
        sending and receiving data, 5-19 to 5-20
        sending END or EOS, 5-20
        terminating on END or EOS, 5-20
Sequence Control Register (SCR), 4-55
Serial Poll Mode Register (SPMR), 4-19
Serial Poll Status Register (SPSR), 4-19
serial polls
    conducting, 5-22
    responding to, 5-22
Set IFC command
    codes for, 4-28
    description, 4-32
Set Parallel Poll Flag command
    codes for, 4-28
    description, 4-31
Set REN command
    codes for, 4-28
    description, 4-32
SFL (System Fail Bit), 4-66
SH (Source Handshake), 4-7
signals and lines
    data lines, E-2
    GPIB-1014 pin assignment on VMEbus connector P1, 3-8
    GPIB-1014 pin assignment on VMEbus connector P2, 3-9
    GPIB connector and signal assignments (illustration), E-4
    handshake lines
        DAV (data valid), E-3
        NDAC (not data accepted), E-3
        NRFD (not ready for data), E-2
        overview, E-2
    interface management lines
        ATN (attention), E-3
        EOI (end or identify), E-3
        IFC (interface clear), E-3
        overview, E-3

REN (remote enable), E-3
SRQ (service request), E-3
VMEbus signals
chart of, 2-1 to 2-2
control signals, 6-2
operation, 6-1 to 6-3
SIZE (Size Bits 5 through 4), 4-53
slave-addressing, VMEbus, 2-2 to 2-3
slave cycles, Timing State Machine, 6-6 to 6-7
slave-data, VMEbus, 2-5
slave read and write transfers, 6-1
SPD (Serial Poll Disable) command, 4-26
SPE (Serial Poll Enable) command, 4-26
specifications
electrical characteristics, 2-1 to 2-2
IEEE 488 bus transfer rate, A-1
operating environment, A-1
physical characteristics, A-1
power requirement, A-1
storage environment, A-1
SPEOI (Send Serial Poll EOI Bit), 4-40
SPMR. *See* Serial Poll Mode Register (SPMR).
SPMS (Serial Poll Mode State Bit), 4-20
SPSR. *See* Serial Poll Status Register (SPSR).
SRQ (service request) line, E-3
SRQI (Service Request Input Bit), 4-15
SRQI IE (Service Request Input Interrupt Enable Bit), 4-15
standards for GPIB, 1-1
storage environment, A-1
STR (Start Bit), 4-56
SUP (Supervisor Bit), 4-66
Supervisor or Non-privileged access, configuration, 3-3
switches. *See* jumpers and switches.
Synchronization and Interrupt Control, GPIB. *See* GPIB Synchronization and Interrupt Control.
system reset
clock and reset circuitry, 6-4
during initialization, 5-1

T

TA (Talker Active Bit), 4-21
Take Control Asynchronously (Pulsed) command
codes for, 4-28
description, 4-31
Take Control Synchronously command
codes for, 4-28
description, 4-31
Take Control Synchronously on END command

codes for, 4-28
description, 4-31
Talker/Listener/Controller (TLC). *See also* Controller function; DMAC channel operation.
addressed implementation
Address Mode 1, 5-6
Address Mode 2, 5-6 to 5-7
Address Mode 3, 5-7 to 5-8
definition, 2-12
GPIB interface, 6-23 to 6-24
initialization of, 5-1 to 5-2
operation of, E-1 to E-2
overview, 5-6
programmed implementation, 5-6
sending/receiving messages
DMA transfers with carry cycle, 5-13 to 5-17
DMA transfers without carry cycle, 5-10 to 5-12
polling during DMAs, 5-17
sending and receiving data, 5-19 to 5-20
sending END or EOS, 5-17, 5-20
terminating on END or EOS, 5-19, 5-20
terminating transfer and checking results, 5-17 to 5-19
using direct memory access, 5-8 to 5-19
using programmed I/O, 5-19 to 5-20
VMEbus slave-addressing, 2-2 to 2-3
TCT (Take Control) command, 4-25
Technical support, *vii*
test and troubleshooting. *See* troubleshooting test procedures.
theory of operation
68450 DMAC
block termination, 6-19 to 6-22
DMAC channel operation, 6-15 to 6-18
initialization and transfer phases, 6-15 to 6-18
initiation and control of channel operation, 6-18 to 6-19
overview, 6-14
address decoding, 6-3 to 6-4
clock and reset circuitry, 6-4
Configuration Register 1, 6-5
Configuration Register 2, 6-6
DMA gating and control, 6-7 to 6-8
DTB Requester and Controller, 6-9 to 6-12
GPIB interface, 6-23 to 6-24
GPIB Synchronization and Interrupt Control, 6-12 to 6-14
interrupter, 6-9
test and troubleshooting, 6-24
Timing State Machine
DMA cycles, 6-7
slave cycles, 6-6 to 6-7
VMEbus interface, 6-1 to 6-3
Timing State Machine

definition of, 2-12
theory of operation
DMA cycles, 6-7
overview, 6-6
slave cycles, 6-6 to 6-7
TLC. *See* Talker/Listener/Controller (TLC).
ton (Talk Only Bit), 4-22
TPAS (Talker Primary Addressed State Bit), 4-21
transceivers for GPIB-1014
component designations, 2-2
control equations of transceivers, 6-3
Transfer Count Registers
Base Transfer Counter Register (BTCR), 4-48
Memory Transfer Counter Register (MTCR), 4-48
theory of operation, 6-17 to 6-18
TRI (Three-State Timing Bit), 4-40
Trigger command
codes for, 4-28
description, 4-30
TRM[1-0] (Transmit/Receive Mode Bits 1 through 0), 4-22

troubleshooting test procedures
    DMA stand alone testing, 6-24
    GPIB interface testing, 6-24
    hardware installation tests, 7-2 to 7-8
    interpreting test procedures, 7-1
    overview, 7-1
    verification of GPIB-1014 before installation, 3-10

U

U (Parallel Poll Unconfigure Bit), 4-35
UNL (Unlisten) command, 4-26
unpacking the GPIB-1014, 1-4
UNT (Untalk) command, 4-26

V

Valid Secondary Command or Address command
    codes for, 4-28
    description, 4-30
verification
    of system compatibility, 3-7 to 3-9
    testing, 3-10
VMEbus
    68450 internal DMA registers (chart), 2-4 to 2-5
    address, 6-2
    control equations of transceivers, 6-3
    control signals, 6-2
    data lines, 6-1
    data transfer bus (DTB) requester
        description of, 2-7
        VMEbus modules not provided, 2-7
    definition, 2-12
    DMA transfers, 6-1
    GPIB-1014 Configuration registers (chart), 2-5
    interrupt events, 2-6
    master-direct memory access, 2-5 to 2-6
    µPD7210 interface registers, 2-3
    signals (chart), 2-1 to 2-2
    slave-addressing, 2-2 to 2-3
    slave-data, 2-3
    slave read and write transfers, 6-1

W

WRITE sample program, C-17 to C-18

X

X (Don't Care Bit), 4-42, 4-50
XEOS (Transmit END with EOS Bit), 4-37
XRM (External Request Mode Bits 7 through 6), 4-51

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

Gold-standard solutions

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

We buy equipment

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

Learn more!

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

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

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

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

ARTISAN® TECHNOLOGY GROUP