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 EXM-GPIB* at our website: *Click HERE*

Getting Started with Your EXM-GPIB and the NI-488.2™ Software for Windows

April 1994 Edition

Part Number 370891A-01

© Copyright 1993, 1994 National Instruments Corporation.
All Rights Reserved.

National Instruments Corporate Headquarters
6504 Bridge Point Parkway
Austin, TX 78730-5039
(512) 794-0100
Technical support fax: (800) 328-2203
(512) 794-5678

Branch Offices:
Australia (03) 879 9422, Austria (0662) 435986, Belgium 02/757.00.20,
Canada (Ontario) (519) 622-9310, 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, Italy 02/48301892, Japan (03) 3788-1921,
Netherlands 03480-33466, Norway 32-848400, Spain (91) 640 0085,
Sweden 08-730 49 70, Switzerland 056/20 51 51, U.K. 0635 523545

Limited Warranty

The EXM-GPIB 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-488® and NI-488.2™ are trademarks 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'émet pas de bruits radioélectriques dépassant les limites applicables aux appareils numériques de classe A prescrites dans le règlement sur le brouillage radioélectrique édicté par le ministère 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 ix
Organization of This Manual ix
Conventions Used in This Manual x
Related Documentation xi
Customer Communication xi

Chapter 1
Introduction 1-1
How to Use This Manual 1-1
What You Need to Get Started 1-2
Software Description 1-2
Hardware Description 1-2

Chapter 2
Hardware Installation and Configuration 2-1
Setting the Shield Ground Configuration (Optional) 2-1
Install the Hardware 2-2
Configure the Hardware 2-2

Chapter 3
Software Installation and Configuration 3-1
NI-488.2 Software Components 3-1
Install the Software 3-1
Configure the Software with wibconf 3-2

Chapter 4
Installation Verification and Troubleshooting 4-1
Run the Hardware Diagnostic Program 4-1
Run the Software Diagnostic Program 4-1
Troubleshooting wibtest Error Messages 4-2
Presence Test of Driver 4-2
Presence Test of GPIB Board 4-2
Incorrect Interrupt Level 4-3
GPIB Cables Connected 4-3

Chapter 5
Using Your NI-488.2 Software 5-1
Introduction to wibic 5-1
Writing Windows Programs That Use the GPIB 5-1
The winsamp Sample 5-2
Programming Considerations 5-2

Appendix A
Hardware Specifications A-1

Appendix B
DLL Direct Entry NI-488 Functions and NI-488.2 Routines B-1

Appendix C
Customer Communication C-1

Glossary G-1

Figure

Figure 2-1. Ground Configuration Jumper Settings 2-1

Tables

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

Table B-1. Direct Entry NI-488.2 Style Routines in C B-3
Table B-2. Direct Entry NI-488 Style Functions in C B-6
Table B-3. Declarations for NI-488.2 Routines in Visual Basic B-9
Table B-4. Declarations for NI-488 Functions in Visual Basic B-12

About This Manual

This manual contains instructions for installing and configuring the National Instruments EXM-GPIB interface board and the NI-488.2 software for Windows. The hardware is intended for use on RadiSys embedded PC systems, including VME, VXI, or EMC systems equipped with 16-bit EXM slots. The software is intended for use with Windows version 3.0 or higher. This manual is meant to be used with the *NI-488.2 Software Reference Manual for MS-DOS* (part number 320282-01). References to EXM-GPIB apply equally to EXM-27.

Organization of This Manual

This manual is organized as follows:

- Chapter 1, *Introduction*, explains how to use this manual, lists what you need to get started, and includes a brief description of the NI-488.2 software and the EXM-GPIB board.
- Chapter 2, *Hardware Configuration and Installation*, contains instructions for configuring and installing your EXM-GPIB board.
- Chapter 3, *Software Installation and Configuration*, contains instructions for installing and configuring your NI-488.2 software.
- Chapter 4, *Installation Verification and Troubleshooting*, describes how to verify the installation and troubleshoot problems.
- Chapter 5, *Using Your NI-488.2 Software,* describes the `wibic` utility, explains how to write a Windows GPIB application program, explains the sample program `winsamp`, and lists some programming considerations.
- Appendix A, *Hardware Specifications*, describes the physical characteristics of the EXM-GPIB board and the recommended operating conditions.
- Appendix B, *DLL Entry NI-488 Functions and NI-488.2 Routines*, explains and gives examples of how to use the DLL Direct Entry NI-488 functions and NI-488.2 routines to access the `gpib.dll` file. Following the examples are tables that list all NI-488.2 routines and NI-488 functions, including their calling syntax and ordinal entry values.
- Appendix C, *Customer Communication*, contains forms you can use to request help from National Instruments or to comment on our products and manuals.

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

Conventions Used in This Manual

The following conventions are used in this manual.

bold	Bold text denotes menus, menu items, dialog buttons, or options.
italic	Italic text denotes emphasis, a cross reference, or an introduction to a key concept.
bold italic	Bold italic text denotes a note, caution or warning.
`monospace`	Lowercase text in this font denotes text or characters that are to be literally input from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, directories, programs, subprograms, subroutines, device names, functions, variables, and filenames.
`bold monospace`	Bold lowercase text in this font denotes the messages and responses that the computer automatically prints to the screen.
<>	Angle brackets enclose the name of a key on the keyboard–for example, <PageDown>.
-	A hyphen between two or more key names enclosed in angle brackets denotes that you should simultaneously press the named keys–for example, <Control-Alt-Delete>.
IEEE 488 and IEEE 488.2	*IEEE 488* and *IEEE 488.2* are used throughout this manual to refer to the ANSI/IEEE Standard 488.1-1987 and the ANSI/IEEE Standard 488.2-1987, respectively, which define the GPIB.
<Enter>	Key names are capitalized.

Abbreviations, acronyms, metric prefixes, mnemonics, symbols, and terms are listed in the *Glossary*.

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 488.2-1987, *IEEE Standard Codes, Formats, Protocols, and Common Commands.*
- *EPC-7 Hardware Reference*, RadiSys Corp.

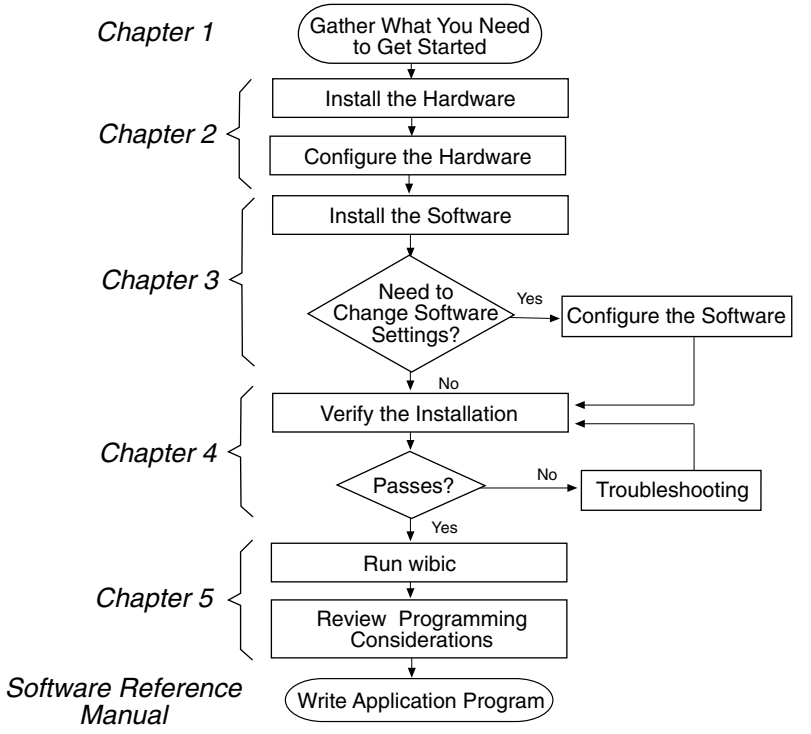
Customer Communication

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 C, *Customer Communication*, at the end of this manual.

Chapter 1
Introduction

This chapter explains how to use this manual, lists what you need to get started, and includes a brief description of the NI-488.2 software and the EXM-GPIB board.

How to Use This Manual



What You Need to Get Started

- ❑ EXM-GPIB board (part number 182355-01)
- ❑ 3.5 in. *NI-488.2 Distribution Disk for EXM-GPIB for Windows Driver and Language Interfaces* (part number 422928-97)
- ❑ Microsoft Windows version 3.0 or higher installed on your computer

Software Description

The NI-488.2 software package for Windows consists of a driver and utilities that transform a 16-bit EXM computer running Windows into a GPIB Controller with complete communications and bus management capabilities. The software includes a Microsoft C language interface that is approximately 85 KB in size.

The NI-488.2 driver is a dynamic link library (DLL) that is loaded whenever a Windows GPIB application starts up. The driver file is configured in accordance with the initialization file `gpib.ini`, which can be modified by using the driver configuration program `wibconf.exe`. The driver and support files are approximately 475 KB in size.

The NI-488.2 driver supports up to four EXM-GPIB boards and is completely compatible with both IEEE 488 and IEEE 488.2 instruments.

Hardware Description

You can use standard GPIB cables to connect the EXM-GPIB with up to 13 instruments. If you want to use more than 13 instruments, you can order a bus extender or expander from National Instruments. Refer to Appendix A, *Hardware Specifications* for more information about the EXM-GPIB hardware specifications and operating conditions.

Chapter 2
Hardware Installation and Configuration

This chapter contains instructions for configuring and installing your EXM-GPIB board.

Setting the Shield Ground Configuration (Optional)

The EXM-GPIB board is set at the factory with the jumper in place to connect the logic ground of the EXM-GPIB board to its shield ground. This configuration minimizes EMI emissions.

Caution: ***The EXM-GPIB board was tested for compliance with FCC standards with the shield ground connected to logic ground. Removing the jumper might cause EMI emissions to exceed any or all of the applicable standards.***

If your application requires that logic ground be disconnected from shield ground, follow these steps:

1. Locate the jumper W1 on your EXM-GPIB board.

2. Remove the jumper and place it across only one of the jumper pins as shown in Figure 2-1.

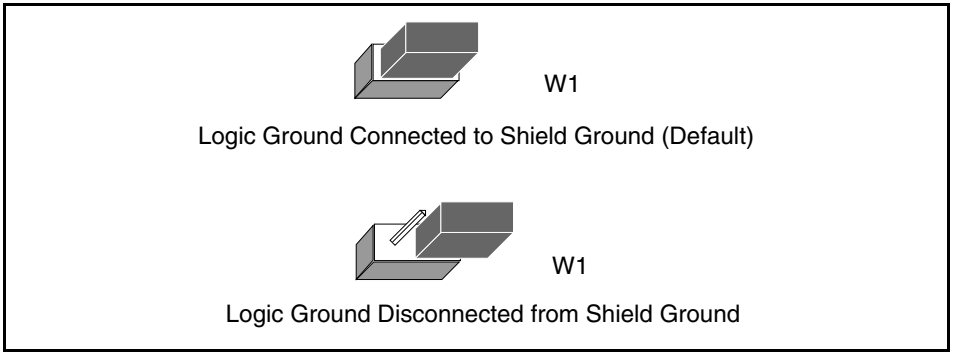


Figure 2-1. Ground Configuration Jumper Settings

3. Record the jumper setting on the *EXM-GPIB Hardware and Software Configuration Form* in Appendix C, *Customer Communication*.

Install the Hardware

You can install the EXM-GPIB board in any unused 16-bit EXM expansion slot (AT-style) in your computer. The EXM-GPIB board does *not* work if installed in an 8-bit expansion slot (PC-style). Consult the user manual or technical reference manual of your computer for specific instructions and warnings.

To install the EXM-GPIB board, perform the following steps:

1. Turn off your computer.

2. Remove the expansion slot cover on the front panel of the computer.

3. Insert the EXM-GPIB board in an unused 16-bit EXM slot with the GPIB receptacle sticking out of the opening on the front panel. It may be a tight fit, but do not force the board into place.

4. Screw the mounting bracket of the EXM-GPIB board into place.

5. Check the installation.

6. Turn on your computer.

7. The boot software asks you if you want to run the RadiSys EPC hardware setup program. Run the hardware setup program and refer to the following section for configuration instructions.

Configure the Hardware

After installing the EXM-GPIB board into an EXM slot of your computer, follow these steps to configure the software settings for the base I/O address, interrupt level, and DMA channel:

1. If you did not automatically enter the EPC hardware setup program after you installed the driver, press <Cntl-Alt-Esc> at the DOS prompt.

2. Select the EXM menu. The following table appears with entries that correspond to configuration registers of the EPC:

Slot	ID	OB1	OB2
0	FF	00	00
1	FF	00	00
2	FF	00	00
3	FF	00	00
4	FF	00	00
5	FF	00	00

3. For every slot in which you have installed an EXM-GPIB board, you must enter the proper configuration values for the three columns ID, OB1, and OB2 as follows:

   ID: Enter 6D, the ID for the EXM-GPIB board. FF means no board is present.

   OB1: Enter a 1-byte hexadecimal number to configure the interrupt level and DMA channel of each board as follows:

7	6	5	4	3	2	1	0
D6	D5	0	0	I2	I1	I0	1

   D6 D5: 00 = No DMA
   10 = DMA Channel 6
   01 = DMA Channel 5

   I2 I1 I0: 000 = No Interrupts
   001 = Interrupt level 3
   011 = Interrupt level 5
   101 = Interrupt level 12

   Bits 5 and 4 are *don't care* bits. Bit 0 must be set to 1.

   OB2: enter a 1-byte hexadecimal number to configure the base I/O address of each board as follows:

7	6	5	4	3	2	1	0
0	0	0	0	B8	B7	B6	B5

   Bits 7 through 4 are *don't care* bits. B8 through B5 make up bits 8 through 5 of the base I/O address for the board; they can be any value between 0 (binary 0000) and 15 (binary 1111).

   The base I/O address is the following 10-bit number:

   1 B8 B7 B6 B5 0 0 0 0 0

   You must select an I/O address that does not conflict with the address of any other device in your system.

Example

Consider a table with the following entries:

Slot	ID	OB1	OB2
0	FF	00	00
1	FF	00	00
2	6D	8B	06
3	6D	47	04
4	FF	00	00
5	FF	00	00

In this table, the entries indicate that two EXM-GPIB boards are installed in slots 2 and 3 of the EPC. The board in slot 2 is `GPIB0`. For `GPIB0`, OB1 is 8B (10001011 in binary), which means it uses DMA channel 6 and interrupt level 12. OB2 is 06 (00000110 in binary), which means B8B7B6B5 is 0110. Therefore, the I/O address of `GPIB0` is 2C0 in hex (1011000000 in binary). The board in slot 3, `GPIB1`, will use DMA channel 5, interrupt level 5, and reside at base address 0x280.

Note: ***The GPIB board in the lowest-numbered slot is designated as*** `GPIB0` ***in the NI-488.2 software. The GPIB board in the next slot is*** `GPIB1`***, and so on***.

4. Save the configuration.

5. Exit the EPC hardware setup program.

6. Reboot your computer.

Chapter 3
Software Installation and Configuration

This chapter contains instructions for installing and configuring your NI-488.2 software.

NI-488.2 Software Components

The NI-488.2 package includes the following components:

- NI-488.2 driver
- An installation program
- Diagnostics
- An interactive GPIB control program
- An interactive configuration utility
- A 16-bit C language interface
- Sample programs that use NI-488 functions and NI-488.2 routines

Install the Software

Complete the following steps to install the NI-488.2 software for Windows.

1. Enter the following command after the DOS prompt to start up Windows:

   ```
   Win
   ```

2. Insert the first installation disk into the floppy drive of your computer.

3. Select **RUN** from the **FILE** menu at the top of the **Program Manager** window.

4. Type `a:\setup` into the dialog box, where `a` is the name of the drive containing the distribution disk.

The interactive Windows setup program takes you through the necessary steps to install the NI-488.2 software. For help during the installation, press the **Help** button. You can exit the setup at any time by pressing the **Exit** button.

Configure the Software with wibconf

`wibconf` is an interactive utility you can use to examine or modify the configuration of the driver. You might want to run `wibconf` to change the settings for device names or timeout values. You can also use `wibconf` to disable DMA transfers and interrupts.

To run `wibconf`, double-click on the **wibconf** icon in the **GPIB** group or enter `wibconf.exe` at the DOS prompt. In Windows, changes made using `wibconf.exe` are recorded in the `gpib.ini` file. The changes are effective immediately.

If you have more than one driver in your system, make sure you are updating the correct `gpib.ini` file. `wibconf.exe` finds the `gpib.ini` file that contains the configuration information by going through the following process:

1. Retrieve the path of the `gpib.ini` file from the command line, if present.

2. Check `C:\Windows` for a `gpib.ini` file.

3. Check `\Windows` for a `gpib.ini` file.

4. Check the current directory for a `gpib.ini` file.

For information about configuration settings in `wibconf`, refer to the *NI-488.2 Software Reference Manual for MS-DOS.*

After the software is installed and configured, you should verify the installation. Refer to Chapter 4, *Installation Verification and Troubleshooting*.

Chapter 4
Installation Verification and Troubleshooting

This chapter describes how to verify the installation and troubleshoot problems.

Run the Hardware Diagnostic Program

To verify and test the hardware installation, run the `ibdiag` hardware diagnostic program that came with your NI-488.2 software. `ibdiag` is a DOS application that can be run either from a computer that was booted under DOS or from the DOS shell that is within Windows. `ibdiag` verifies that your hardware is functioning properly and that the software configuration settings are correct.

Follow these steps to run `ibdiag`:

1. Disconnect any GPIB cables from the EXM-GPIB.
2. Go to the directory where the software is installed (default is `c:\exmgpib`).
3. Enter the following command:

   ```
   ibdiag
   ```

If `ibdiag` completes with no errors, your hardware is functioning properly. If an error does occur, an error message asks if you want to continue testing. Do not continue testing. Type <n> to stop testing and then <q> to quit. Use the EPC setup program to select different hardware settings and run `ibdiag` again. If you still get an error message, record the error message and refer to Appendix C, *Customer Communication*, before contacting National Instruments for technical support.

Run the Software Diagnostic Program

To verify and test the hardware and software installation, run the software diagnostic program `wibtest` that came with your NI-488.2 software. The `wibtest` program is a Windows application that requires no user interaction. Follow these steps to run `wibtest`:

1. Disconnect all GPIB cables from the EXM-GPIB.
2. Double-click on the **wibtest** icon in the **EXM-GPIB** window.

If `wibtest` completes with no errors, you have installed the NI-488.2 software correctly. If `wibtest` responds with an error message, refer to the next section for troubleshooting information.

Troubleshooting wibtest Error Messages

The following sections explain common error messages generated by `wibtest`.

Note: ***In the following paragraphs,*** `GPIB`*x* ***refers to board*** `GPIB0,GPIB1,GPIB2,` ***or*** `GPIB3` ***as appropriate.***

Presence Test of Driver

The `wibtest` program tests for the presence of the NI-488.2 driver. It displays the following message if it detects a problem:

`<<< No driver present for GPIBx. >>>`

If this message appears, take one of following actions:

- You can ignore this message when it applies to a nonexistent board. The `wibtest` program always tries to test for four EXM-GPIB boards. In most cases you will have fewer than four boards installed in your computer. This message appears when `wibtest` tries to test a board that does not exist.
- Verify that the files `gpib.dll` and `gpib.ini` are located in your Windows directory (usually `C:\Windows`).

Presence Test of GPIB Board

The following error message appears if `GPIB`*x* is not installed or if the software is not configured properly:

`<<< No board present for GPIBx. >>>`

If this message appears, you could have one of these situations:

- The `Use this GPIB Interface` field in `wibconf` is set to `no` for board `GPIB`*x*. If you want to use this board you need to set this field to `yes`.
- The board is not installed or configured correctly. Check that the board is properly installed. You can run the EPC hardware setup program to verify the board configuration. Refer to *Configure the Hardware* in Chapter 2, *Hardware Installation and Configuration*, for detailed instructions.
- The software and hardware settings do not match. You can run `wibconf` to check the current configuration of the software.

Incorrect Interrupt Level

The `wibtest` program hangs if the EXM-GPIB board under test is configured to use an invalid interrupt level or an interrupt level used by another device in your system. If this occurs, you can either run the EPC hardware setup program and select a different interrupt level or use `wibconf` to disable interrupts. For detailed instructions, refer to the sections *Configure the Hardware* in Chapter 2 and *Configure the Software with wibconf* in Chapter 3.

GPIB Cables Connected

The following error message appears if a GPIB cable is connected to the board when you run `ibtest`.

```
Call(25) 'ibcmd " "' failed, ibsta (0x134) not what was expected (0x8130)

Call(25) 'ibcmd " "' failed, expected ibsta (0x100) to have the ERR bit set.
```

Disconnect all GPIB cables before trying the test again.

Chapter 5
Using Your NI-488.2 Software

This chapter describes the `wibic` utility, explains how to write a Windows GPIB application program, explains the sample program `winsamp`, and lists some programming considerations.

Introduction to wibic

You can use `wibic`, the Windows Interface Bus Interactive Control utility, to enter NI-488.2 functions interactively and display the results of the function calls automatically. Without writing an application, you can use `wibic` to:

- Verify GPIB communication with your device quickly and easily.
- Learn the syntax of the NI-488.2 functions before writing your application.
- Become familiar with the commands of your device.
- Receive data from your GPIB device.

For more information about `wibic`, use the online help feature or refer to the *NI-488.2 Software Reference Manual for MS-DOS.*

Writing Windows Programs That Use the GPIB

There are two methods for writing a Windows application that uses the GPIB. The easiest method is to write an application that uses the standard NI-488 functions and NI-488.2 routines and is linked to one of the Windows 3 language interfaces. The NI-488.2 software includes the Microsoft C language interface. Contact National Instruments for information about ordering interfaces for other languages.

The second method of writing an NI-488.2 for Windows application is to use the DLL direct entry NI-488 functions and NI-488.2 routines. Using direct entry, you do not need to have a special language interface to link with your application. You might want to use direct entry if you are using a language other than C or if you need more flexibility. Refer to Appendix B, *DLL Direct Entry NI-488 Functions and NI-488.2 Routines*, for more information.

The winsamp Sample

There are two primary parts to the sample program `winsamp. winsamp.c` handles most of the details for interfacing with Windows and `gpibsamp.c` makes GPIB calls and then displays the results on the screen.

To compile the sample programs, enter the following commands:

```
cl /c /Gsw /AS winsamp.c
cl /c /Gsw /AS gpibsamp.c
link winsamp gpibsamp,,,gpib slibcew libw,winsamp.def
```

To execute `winsamp`, set it up as a Windows application and select the `winsamp` icon.

Programming Considerations

By following these general rules, any application can use the `gpib.dll`.

- Include the header file `windecl.h` in your source code.
- Make the same GPIB calls that you do under DOS.
- Link `gpib.lib` with your application.

  Note: ***All NI-488.2*** `gpib.dll` ***files for Windows share the same*** `.lib` ***file; therefore, you do not have to relink applications to switch between GPIB boards.***

- Ensure that the correct `gpib.dll` is in the directory in which Windows is installed or in the DOS search path when the application is run. Unlike the `gpib.lib` file, `gpib.dll` files are unique for each National Instruments GPIB board.
- Ensure that `gpib.ini` is in the directory in which Windows is installed when the application is run so that it can be used to properly initialize the `gpib.dll` file. The `gpib.ini` file is also unique for each GPIB board.

For more information about using the NI-488.2 software and each NI-488 function and NI-488.2 routine, refer to the *NI-488.2 Software Reference Manual for MS-DOS*.

Appendix A
Hardware Specifications

This appendix describes the physical characteristics of the EXM-GPIB board and the recommended operating conditions.

Table A-1. Electrical Characteristics

Characteristic	Specification
Transfer Rates	
GPIB Reads	over 1 Mbytes/s*
GPIB Writes	over 1 Mbytes/s*
GPIB Commands	500 kbytes/s*
Power Requirement (from EXM expansion slot)	+5 VDC 0.6 A Typical 0.8 A Maximum
* Actual speed may vary considerably from those shown due to instrumentation capabilities.	

Table A-2. Physical Characteristics

Characteristic	Specification
Dimensions	3.0 in. by 5.9 in.
I/O Connector	IEEE 488 Standard 24-pin

Table A-3. Environmental Characteristics

Characteristic	Specification
Operating Environment	
Component Temperature	0° to 40° C
Relative Humidity	5% to 90%, noncondensing
Storage Environment	
Temperature	-20° to 70° C
Relative Humidity	5% to 90%, noncondensing
EMI	FCC Class A Verified

Appendix B
DLL Direct Entry NI-488 Functions and NI-488.2 Routines

This appendix explains and gives examples of how to use the DLL Direct Entry NI-488 functions and NI-488.2 routines to access the `gpib.dll` file. Following the examples are tables that list all NI-488.2 routines and NI-488 functions, including their calling syntax and ordinal entry values.

The DLL Direct Entry NI-488 functions and NI-488.2 routines can be used to access the `gpib.dll` file from any language or programming environment that runs under Windows 3 and supports access to standard Windows DLL functions. As with all functions exported by a DLL, these functions conform to the PASCAL calling conventions. A few examples of using these entry points follow. Tables B-1 and B-2 contain a complete list of all of the entry points in C. Tables B-3 and B-4 contain the declarations for the NI-488.2 routines and NI-488 functions in Visual Basic.

For specific information on the variables `ibsta`, `iberr`, and `ibcntl`, refer to Chapter 3 in the *NI-488.2 Software Reference Manual for MS-DOS*. For specific information on a routine or function, refer to Chapters 4 and 5 in the *NI-488.2 Software Reference Manual for MS-DOS*. For information about accessing DLL functions from a given language or environment or using ordinal entry values which some environments do not support, see the documentation provided with that package.

Example 1, accessing the `gpib.dll` file from Turbo Pascal for Windows:

```
(* First, import the DLL functions you plan to use. *)

function DLLibfind(udname: PChar;
                   var ibsta: integer;
                   var iberr: integer;
                   var ibcntl: longint) : integer; far;
external 'GPIB' index 22;

function DLLibsic(ud: integer;
                   var: ibsta: integer;
                   var: iberr: integer;
                   var: ibcntl: longint) : integer; far;
external 'GPIB' index 42;
procedure DLLSendIFC(board: integer;
                   var ibsta: integer;
                   var iberr: integer;
                   var ibcntl: longint) ; far;
external 'GPIB' index 119;

(* Your application can now use the functions. *)
```

```
var BoardHandle: integer;
var ibsta: integer;
var iberr: integer;
var ibcntl: longint;
var temp: integer;

BoardHandle:= DLLibfind('GPIB0', ibsta, iberr, ibcntl);
temp:= DLLibsic(BoardHandle, ibsta, iberr, ibcntl);

                (* or *)

DLLSendIFC(0, ibsta, iberr, ibcntl);
```

Example 2, accessing the `gpib.dll` file from Microsoft Visual Basic:

```
'First declare the DLL functions you plan to use.

Declare function DLLibfind Lib "gpib.dll"
   (ByVal udname$, ibsta%, iberr%, ibcntl&) As Integer
Declare function DLLibsic Lib "gpib.dll"
   (ByVal ud%, ibsta%, iberr%, ibcntl&) As Integer
Declare sub DLLSendIFC Lib "gpib.dll"
   (ByVal board%, ibsta%, iberr%, ibcntl&)

'Your application can now use the functions.

Global BoardHandle As Integer
Global ibsta As Integer
Global iberr As Integer
Global ibcntl As Long

BoardHandle% =
   DLLibfind("GPIB0", ibsta%, iberr%, ibcntl&)

temp% = DLLibsic(BoardHandle%, ibsta%, iberr%, ibcntl&)

   'or

call DLLSendIFC(0, ibsta%, iberr%, ibcntl&)
```

Note: ***All of the routines listed in Table B-1 are of type*** void _far _pascal.

Table B-1. Direct Entry NI-488.2 Style Routines in C

Routine **(ordinal entry value)**	**Syntax**
AllSpoll(100)	DLLAllSpoll (short board, short _far* addresslist, short _far * resultlist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
DevClear(101)	DLLDevClear (short board, short address, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
DevClearList(102)	DLLDevClearList (short board, short _far *addresslist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
EnableLocal(103)	DLLEnableLocal (short board, short _far *addresslist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
EnableRemote(104)	DLLEnableRemote (short board, short _far *addresslist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
FindLstn(105)	DLLFindLstn (short board, short _far *addresslist, short _far *resultlist, short limit, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
FindRQS(106)	DLLFindRQS (short board, short _far *addresslist, short _far *result, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
GenerateREQF(53)	DLLGenerateREQF (short board, short addr, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
GenerateREQT(52)	DLLGenerateREQT (short board,short addr, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
GotoMultAddr(129)	DLLGotoMultAddr (short board, unsigned short type, unsigned short (_far _loadds *addrfunc) (), unsigned short (_far _loadds *spollfunc) (), short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)

(continues)

Table B-1. Direct Entry NI-488.2 Style Routines in C (Continued)

Routine (ordinal entry value)	**Syntax**
PassControl(107)	DLLPassControl (short board, short address, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
PPoll(108)	DLLPPoll (short board, short _far *result, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
PPollConfig(109)	DLLPPollConfig (short board, short address, short dataline, short sense, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
PPollUnconfig(110)	DLLPPollUnconfig (short board, short _far *addresslist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
RcvRespMsg(111)	DLLRcvRespMsg (short board, char _far *data, long count, short termination, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ReadStatusByte(112)	DLLReadStatusByte (short board, short address, short _far *result, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
Receive(113)	DLLReceive (short board, short address, char _far *data, unsigned long count, short termination, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ReceiveSetup(114)	DLLReceiveSetup (short board, short address, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ResetSys(115)	DLLResetSys (short board, short _far *addresslist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
Send(116)	DLLSend (short board, short address, char _far *data, long count, short eotmode, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
SendCmds(117)	DLLSendCmds (short board, char _far *commands, unsigned long count, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
SendDataBytes(118)	DLLSendDataBytes (short board, char _far *data, long count, short eotmode, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)

(continues)

Table B-1. Direct Entry NI-488.2 Style Routines in C (Continued)

Routine **(ordinal entry value)**	**Syntax**
SendIFC(119)	DLLSendIFC (short board, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
SendList(120)	DLLSendList (short board, short _far *addresslist, char _far *data, long count, short eotmode, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
SendLLO(121)	DLLSendLLO (short board, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
SendSetup(122)	DLLSendSetup (short board, short _far *addresslist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
SetRWLS(123)	DLLSetRWLS (short board, short _far *addresslist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
TestSRQ(124)	DLLTestSRQ (short board, short _far *result, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
TestSys(125)	DLLTestSys (short board, short _far *addresslist, short _far *resultlist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
Trigger(126)	DLLTrigger (short board, short address, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
TriggerList(127)	DLLTriggerList (short board, short _far *addresslist, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
WaitSRQ(128)	DLLWaitSRQ (short board, short _far *result, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)

Note: ***All of the functions listed in Table B-2 are of type*** short _far _pascal .

Table B-2. Direct Entry NI-488 Style Functions in C

Functions (**ordinal entry value**)	**Syntax**
ibbna(10)	DLLibbna (short ud, char _far *bname, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibcac(11)	DLLibcac (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibclr(12)	DLLibclr (short ud, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibcmd(13)	DLLibcmd (short ud, char _far *cmd, long cnt, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibcmda(14)	DLLibcmda (short ud, char _far *cmd, long cnt, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibconfig(15)	DLLibconfig (short ud, unsigned short option, unsigned short value, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibdev(16)	DLLibdev (short boardindex, short pad, short sad, short tmo, short eot, short eos, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibdma(18)	DLLibdma (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibeos(19)	DLLibeos (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibeot(20)	DLLibeot (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibevent(21)	DLLibevent (short ud, short event, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibfind(22)	DLLibfind (char _far *udname, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibgts(23)	DLLibgts (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)

(continues)

Table B-2. Direct Entry NI-488 Style Functions in C (Continued)

Functions (ordinal entry value)	**Syntax**
ibist(24)	DLLibist (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
iblines(25)	DLLiblines (short ud, short _far *clines, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibln(26)	DLLiblin (short ud, short pad, short sad, short _far *listen, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibloc(27)	DLLibloc (short ud, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibonl(28)	DLLibonl (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibpad(29)	DLLibpad (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibpct(30)	DLLibpct (short ud, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibppc(32)	DLLibppc (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibrd(33)	DLLibrd (short ud, short _far *rd, unsigned long cnt, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibrda(34)	DLLibrda (short ud, char _far *rd, unsigned long cnt, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibrdf(35)	DLLibrdf (short ud, char _far *flname, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibrdkey†(36)	DLLibrdkey (short ud, char _far *rd, unsigned short cnt, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibrpp(37)	DLLibrpp (short ud, char _far *ppr, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibrsc (38)	DLLibrsc (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)

(continues)

Table B-2. Direct Entry NI-488 Style Functions in C (Continued)

Functions (ordinal entry value)	**Syntax**
ibrsp(39)	DLLibrsp (short ud, char _far *spr, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibrsv(40)	DLLibrsv (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibsad(41)	DLLibsad (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibsic(42)	DLLibsic (short ud, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibsre(43)	DLLibsre (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibstop(44)	DLLibstop (short ud, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibtmo(45)	DLLibtmo (short ud, short v, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibtrg(46)	DLLibtrg (short ud, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibwait(47)	DLLibwait (short ud, short mask, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibwrt(48)	DLLibwrt (short ud, char _far *wrt, unsigned long cnt, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibwrta(49)	DLLibwrta (short ud, char _far *wrt, unsigned long cnt, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibwrtf(50)	DLLibwrtf (short ud, char _far *flname, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)
ibwrtkey†(51)	DLLibwrtkey (short ud, char _far *wrt, unsigned short cnt, short _far *ibsta, short _far *iberr, unsigned long _far *ibcntl)

† ibrdkey and ibwrtkey are OEM functions. Refer to the *NI-488 Hardware Key Functions Reference Guide* (part number 320359-01) for a detailed description of these functions.

Table B-3 lists the declarations of the NI-488.2 routines for Visual Basic.

Note: ***When you call a subroutine that has an array as one or more of its parameters, pass the first element of the array. For example, you would include the*** `addresslist` ***array as follows:***

```
Call DLLFindRQS (board%, addresslist%(0), result%, ibsta%,
iberr%, ibcntl&)
```

Table B-3. Declarations for NI-488.2 Routines in Visual Basic

Functions (**ordinal entry value**)	**Syntax**
`AllSpoll(100)`	Declare sub `DLLAllSpoll` Lib `"gpib.dll"` (ByVal `board%, addresslist` As Integer, `resultlist` As Integer, `ibsta%, iberr%, ibcntl&)`
`DevClear(101)`	Declare sub `DLLDevClear` Lib `"gpib.dll"` (ByVal `board%,` ByVal `address%, ibsta%, iberr%, ibcntl&)`
`DevClearList(102)`	Declare sub `DLLDevClearList` Lib `"gpib.dll"` (ByVal `board%, addresslist` As Integer, `ibsta%, iberr%, ibcntl&)`
`EnableLocal(103)`	Declare sub `DLLEnableLocal` Lib `"gpib.dll"` (ByVal `board%, addresslist` As Integer, `ibsta%, iberr%, ibcntl&)`
`EnableRemote(104)`	Declare sub `DLLEnableRemote` Lib `"gpib.dll"` (ByVal `board%, addresslist` As Integer, `ibsta%, iberr%, ibcntl&)`
`FindLstn(105)`	Declare sub `DLLFindLstn` Lib `"gpib.dll"` (ByVal `board%, addresslist` As Integer, `resultlist` As Integer, ByVal `limit%, ibsta%, iberr%, ibcntl&)`
`FindRQS(106)`	Declare sub `DLLFindRQS` Lib `"gpib.dll"` (ByVal `board%, addresslist` As Integer, `result%, ibsta%, iberr%, ibcntl&)`
`PassControl(107)`	Declare sub `DLLPassControl` Lib `"gpib.dll"` (ByVal `board%,` ByVal `address%, ibsta%, iberr%, ibcntl&)`
`PPoll(108)`	Declare sub `DLLPPoll` Lib `"gpib.dll"` (ByVal `board%, result%, ibsta%, iberr%, ibcntl&)`

(continues)

Table B-3. Declarations for NI-488.2 Routines in Visual Basic (Continued)

Functions (**ordinal entry value**)	**Syntax**
PPollConfig(109)	Declare sub DLLPPollConfig Lib "gpib.dll" (ByVal board%, ByVal address%, ByVal dataline%, ByVal sense%, ibsta%, iberr%, ibcntl&)
PPollUnconfig(110)	Declare sub DLLPPollUnconfig Lib "gpib.dll" (ByVal board%, addresslist As Integer, ibsta%, iberr%, ibcntl&)
RcvRespMsg(111)	Declare sub DLLRcvRespMsg Lib "gpib.dll" (ByVal board%, ByVal data$, ByVal count&, ByVal termination%, ibsta%, iberr%, ibcntl&)
ReadStatusByte(112)	Declare sub DLLReadStatusByte Lib "gpib.dll" (ByVal board%, ByVal address%, result%, ibsta%, iberr%, ibcntl&)
Receive(113)	Declare sub DLLReceive Lib "gpib.dll" (ByVal board%, ByVal address%, ByVal data$, ByVal count&, ByVal termination%, ibsta%,iberr%, ibcntl&)
ReceiveSetup(114)	Declare sub DLLReceiveSetup Lib "gpib.dll" (ByVal board%, ByVal address%, ibsta%, iberr%, ibcntl&)
ResetSys(115)	Declare sub DLLResetSys Lib "gpib.dll" (ByVal board%, addresslist As Integer, ibsta%, iberr%, ibcntl&)
Send(116)	Declare sub DLLSend Lib "gpib.dll" (ByVal board%, ByVal address%, ByVal data$, ByVal count&, ByVal eotmode% ibsta%, iberr%, ibcntl&)
SendCmds(117)	Declare sub DLLSendCmds Lib "gpib.dll" (ByVal board%, ByVal commands$, ByVal count&, ibsta%, iberr%, ibcntl&)
SendDataBytes(118)	Declare sub DLLSendDataBytes Lib "gpib.dll" (ByVal board%, ByVal data$, ByVal count&, ByVal eotmode%, ibsta%, iberr%, ibcntl&)
SendIFC(119)	Declare sub DLLSendIFC Lib "gpib.dll" (ByVal board%, ibsta%, iberr%, ibcntl&)

(continues)

Table B-3. Declarations for NI-488.2 Routines in Visual Basic (Continued)

Functions (**ordinal entry value**)	**Syntax**
SendList(120)	Declare sub DLLSendList Lib "gpib.dll" (ByVal board%, addresslist As Integer, ByVal data$, ByVal count&, ByVal eotmode%, ibsta%, iberr%, ibcntl&)
SendLLO(121)	Declare sub DLLSendLLO Lib "gpib.dll" (ByVal board%, ibsta%, iberr%, ibcntl&)
SendSetup(122)	Declare sub DLLSendSetup Lib "gpib.dll" (ByVal board%, addresslist As Integer, ibsta%, iberr%, ibcntl&)
SetRWLS(123)	Declare sub DLLSetRWLS Lib "gpib.dll" (ByVal board%, addresslist As Integer, ibsta%, iberr%, ibcntl&)
TestSRQ(124)	Declare sub DLLTestSRQ Lib "gpib.dll" (ByVal board%, result%, ibsta%, iberr%, ibcntl&)
TestSys(125)	Declare sub DLLTestSys Lib "gpib.dll" (ByVal board%, addresslist As Integer, resultlist As Integer, ibsta%, iberr%, ibcntl&)
Trigger(126)	Declare sub DLLTrigger Lib "gpib.dll" (ByVal board%, ByVal address%, ibsta%, iberr%, ibcntl&)
TriggerList(127)	Declare sub DLLTriggerList Lib "gpib.dll" (ByVal board%, addresslist As Integer, ibsta%, iberr%, ibcntl&)
WaitSRQ(128)	Declare sub DLLWaitSRQ Lib "gpib.dll" (ByVal board%, result%, ibsta%, iberr%, ibcntl&)
GenerateREQF(53)	Not available in Visual Basic
GenerateREQT(52)	Not available in Visual Basic
GotoMultAddr(129)	Not available in Visual Basic

Table B-4 lists the declarations of the NI-488 functions for Visual Basic.

Table B-4. Declarations for NI-488 Functions in Visual Basic

Functions (**ordinal entry value**)	**Syntax**
ibbna(10)	Declare function DLLibbna Lib "gpib.dll" (ByVal ud%, ByVal bname$, ibsta%, iberr%, ibcntl&) As Integer
ibcac(11)	Declare function DLLibcac Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibclr(12)	Declare function DLLibclr Lib "gpib.dll" (ByVal ud%, ibsta%, iberr%, ibcntl&) As Integer
ibcmd(13)	Declare function DLLibcmd Lib "gpib.dll" (ByVal ud%, ByVal cmd$, ByVal cnt&, ibsta%, iberr%, ibcntl&) As Integer
ibcmda(14)	Declare function DLLibcmda Lib "gpib.dll" (ByVal ud%, ByVal cmd$, ByVal cnt&, ibsta%, iberr%, ibcntl&) As Integer
ibconfig(15)	Declare function DLLibconfig Lib "gpib.dll" (ByVal ud%, ByVal option%, ByVal value%, ibsta%, iberr%, ibcntl&) As Integer
ibdev(16)	Declare function DLLibdev Lib "gpib.dll" (ByVal index%, ByVal pad%, ByVal sad%, ByVal tmo%, ByVal eot%, ByVal eos%, ibsta%, iberr%, ibcntl&) As Integer
ibdma(18)	Declare function DLLibdma Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibeos(19)	Declare function DLLibeos Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibeot(20)	Declare function DLLibeot Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibevent(21)	Declare function DLLibevent Lib "gpib.dll" (ByVal ud%, ByVal event%, ibsta%, iberr%, ibcntl&) As Integer

(continues)

Table B-4. Declarations for NI-488 Functions in Visual Basic (Continued)

Functions (ordinal entry value)	**Syntax**
ibfind(22)	Declare function DLLibfind Lib "gpib.dll" (ByVal udname$, ibsta%, iberr%, ibcntl&) As Integer
ibgts(23)	Declare function DLLibgts Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibist(24)	Declare function DLLibist Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
iblines(25)	Declare function DLLiblines Lib "gpib.dll" (ByVal ud%, clines%, ibsta%, iberr%, ibcntl&) As Integer
ibln(26)	Declare function DLLiblin Lib "gpib.dll" (ByVal ud%, ByVal pad%, ByVal sad%, listen%, ibsta%, iberr%, ibcntl&) As Integer
ibloc(27)	Declare function DLLibloc Lib "gpib.dll" (ByVal ud%, ibsta%, iberr%, ibcntl&) As Integer
ibonl(28)	Declare function DLLibonl Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibpad(29)	Declare function DLLibpad Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibpct(30)	Declare function DLLibpct Lib "gpib.dll" (ByVal ud%, ibsta%, iberr%, ibcntl&) As Integer
ibppc(32)	Declare function DLLibppc Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibrd(33)	Declare function DLLibrd Lib "gpib.dll" (ByVal ud%, ByVal rd$, ByVal cnt&, ibsta%, iberr%, ibcntl&) As Integer
ibrda(34)	Declare function DLLibrda Lib "gpib.dll" (ByVal ud%, ByVal rd$, ByVal cnt&, ibsta%, iberr%, ibcntl&) As Integer

(continues)

Table B-4. Declarations for NI-488 Functions in Visual Basic (Continued)

Functions (ordinal entry value)	**Syntax**
ibrdf(35)	Declare function DLLibrdf Lib "gpib.dll" (ByVal ud%, ByVal flname$, ibsta%, iberr%, ibcntl&) As Integer
ibrdkey†(36)	Declare function DLLibrdkey Lib "gpib.dll" (ByVal ud%, ByVal rd$, ByVal cnt%, ibsta%, iberr%, ibcntl&) As Integer
ibrpp(37)	Declare function DLLibrpp Lib "gpib.dll" (ByVal ud%, ppr%, ibsta%, iberr%, ibcntl&) As Integer
ibrsc(38)	Declare function DLLibrsc Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibrsp(39)	Declare function DLLibrsp Lib "gpib.dll" (ByVal ud%, spr%, ibsta%, iberr%, ibcntl&) As Integer
ibrsv(40)	Declare function DLLibrsv Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibsad(41)	Declare function DLLibsad Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibsic(42)	Declare function DLLibsic Lib "gpib.dll" (ByVal ud%, ibsta%, iberr%, ibcntl&) As Integer
ibsre(43)	Declare function DLLibsre Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibstop(44)	Declare function DLLibstop Lib "gpib.dll" (ByVal ud%, ibsta%, iberr%, ibcntl&) As Integer
ibtmo(45)	Declare function DLLibtmo Lib "gpib.dll" (ByVal ud%, ByVal v%, ibsta%, iberr%, ibcntl&) As Integer
ibtrg(46)	Declare function DLLibtrg Lib "gpib.dll" (ByVal ud%, ibsta%, iberr%, ibcntl&) As Integer

(continues)

Table B-4. Declarations for NI-488 Functions in Visual Basic (Continued)

Functions (ordinal entry value)	**Syntax**
ibwait(47)	Declare function DLLibwait Lib "gpib.dll" (ByVal ud%, ByVal mask%, ibsta%, iberr%, ibcntl&) As Integer
ibwrt(48)	Declare function DLLibwrt Lib "gpib.dll" (ByVal ud%, ByVal wrt$, ByVal cnt&, ibsta%, iberr%,ibcntl&) As Integer
ibwrta(49)	Declare function DLLibwrta Lib "gpib.dll" (ByVal ud%, ByVal wrt$, ByVal cnt&, ibsta%, iberr%, ibcntl&) As Integer
ibwrtf(50)	Declare function DLLibwrtf Lib "gpib.dll" (ByVal ud%, ByVal flname$, ibsta%, iberr%, ibcntl&) As Integer
ibwrtkey†(51)	Declare function DLLibwrtkey Lib "gpib.dll" (ByVal ud%, ByVal wrt$, ByVal cnt%, ibsta%, iberr%, ibcntl&) As Integer

† ibrdkey and ibwrtkey are OEM functions. Refer to the *NI-488 Hardware Key Functions Reference Guide* (part number 320359-01) for a detailed description of these functions.

Appendix C
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.

Corporate Headquarters
(512) 795-8248
Technical support fax: (800) 328-2203
(512) 794-5678

Branch Offices	**Phone Number**	**Fax Number**
Australia	(03) 879 9422	(03) 879 9179
Austria	(0662) 435986	(0662) 437010-19
Belgium	02/757.00.20	02/757.03.11
Denmark	45 76 26 00	45 76 71 11
Finland	(90) 527 2321	(90) 502 2930
France	(1) 48 14 24 00	(1) 48 14 24 14
Germany	089/741 31 30	089/714 60 35
Italy	02/48301892	02/48301915
Japan	(03) 3788-1921	(03) 3788-1923
Netherlands	03480-33466	03480-30673
Norway	32-848400	32-848600
Spain	(91) 640 0085	(91) 640 0533
Sweden	08-730 49 70	08-730 43 70
Switzerland	056/20 51 51	056/20 51 55
U.K.	0635 523545	0635 523154

Technical Support Form

Technical support is available at any time by fax. Include the information from your configuration form. Use 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 _____

(continues)

National Instruments software product _____

Version _____

Configuration _____

The problem is _____

_____

_____

_____

_____

_____

_____

_____

List any error messages _____

_____

_____

_____

_____

_____

_____

_____

The following steps will reproduce the problem _____

_____

_____

_____

_____

_____

_____

EXM-GPIB 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

- EXM-GPIB Hardware Revision _____
- NI-488.2 Software Revision Number on Distribution medium

  _____
- Board Settings:

	Base I/O Address	Interrupt Level	DMA Channel
GPIB0	_____	_____	_____
GPIB1	_____	_____	_____
GPIB2	_____	_____	_____
GPIB3	_____	_____	_____

- Shield Ground Connected to Logic (Yes or No) _____

Other Products

- Computer Make and Model _____
- Microprocessor _____
- Clock Frequency (Bus and Microprocessor)_____
- Type of Video Board Installed _____
- Microsoft Windows Version_____
- Windows Mode (Enhanced, Standard, or Real)_____
- Application Programming Language (BASIC, C, Pascal, and so on)_____

(continues)

- Other Boards in System _____
- Base I/O Address of Other Boards _____
- Arbitration Levels of Other Boards _____
- Interrupt Level of Other Boards _____
- VXIbus Mainframe Make and Model _____
- Other VXIbus Devices in System _____
- Static Logical Addresses of Other VXIbus Devices _____

_____

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: **Getting Started with Your EXM-GPIB and the NI-488.2™ Software for Windows**

Edition Date: **April 1994**

Part Number: **370891A-01**

Please comment on the completeness, clarity, and organization of the manual.

(continues)

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, MS 53-02
Austin, TX 78730-5039

Fax to: Technical Publications
National Instruments Corporation
MS 53-02
(512) 794-5678

Glossary

Prefix	Meaning	Value
k-	kilo-	10^3
M-	mega-	10^6

°	degrees
%	percent
A	amperes
AC	alternating current
ANSI	American National Standards Institute
BIOS	Basic Input/Output System
C	Celsius
DLL	dynamic link library
DMA	direct memory access
EMI	electromagnetic interference
EPC	Embedded Personal Computer
GPIB	General Purpose Interface Bus
hex	hexadecimal
Hz	hertz
I/O	input/output
IEEE	Institute of Electrical and Electronic Engineers
in.	inches
KB	kilobytes of memory
MB	megabytes of memory
OEM	original equipment manufacturer
PC	personal computer
RAM	random-access memory
s	seconds
VDC	volts direct current

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