APEX WAVES

COMPREHENSIVE SERVICES

We offer competitive repair and calibration services, as well as easily accessible documentation and free downloadable resources.

SELL YOUR SURPLUS

We buy new, used, decommissioned, and surplus parts from every NI series. We work out the best solution to suit your individual needs.

〜 Sell For Cash 〜 Get Credit 〜 Receive a Trade-In Deal

OBSOLETE NI HARDWARE IN STOCK & READY TO SHIP

We stock **New, New Surplus, Refurbished,** and **Reconditioned** NI Hardware.

Bridging the gap between the manufacturer and your legacy test system.

📞 **1-800-915-6216**

🌐 **www.apexwaves.com**

✉️ **sales@apexwaves.com**

All trademarks, brands, and brand names are the property of their respective owners.
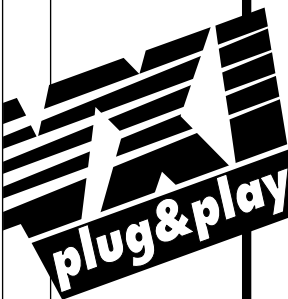
Request a Quote ✉ CLICK HERE *PCI-MXI-2*

Getting Started with Your VXI/VME-PCI8000 Series and the NI-VXI™ Software for Microsoft Operating Systems

MXI

VXI plug&play

Internet Support

support@natinst.com
E-mail: info@natinst.com
FTP Site: ftp.natinst.com
Web Address: http://www.natinst.com

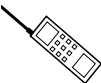**Bulletin Board Support**

BBS United States: (512) 794-5422
BBS United Kingdom: 01635 551422
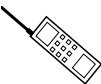BBS France: 01 48 65 15 59

Fax-on-Demand Support

(512) 418-1111

Telephone Support (U.S.)

Tel: (512) 795-8248
Fax: (512) 794-5678

International Offices

Australia 02 9874 4100, 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 09 527 2321, France 01 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,
Mexico 5 520 2635, Netherlands 0348 433466, Norway 32 84 84 00, Singapore 2265886,
Spain 91 640 0085, Sweden 08 730 49 70, Switzerland 056 200 51 51, Taiwan 02 377 1200,
U.K. 01635 523545

National Instruments Corporate Headquarters

6504 Bridge Point Parkway   Austin, TX 78730-5039   Tel: (512) 794-0100

Important Information

Warranty

The National Instruments MXIbus boards and accessories are warranted against defects in materials and workmanship for a period of one year 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

LabVIEW®, MANTIS™, MITE™, NI-VXI™, TIC™, and VXIpc™ 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 Class A 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. Classification requirements are the same for the Federal Communications Commission (FCC) and the Canadian Department of Communications (DOC). This equipment has been tested and found to comply with the following two regulatory agencies:

Federal Communications Commission

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. 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.

Notices 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.*

This device complies with the FCC rules only if used with shielded interface cables of suitable quality and construction. National Instruments used such cables to test this device and provides them for sale to the user. The use of inferior or nonshielded interface cables 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: *Interference to Home Electronic Entertainment Equipment Handbook*. This booklet is available from the U.S. Government Printing Office, Washington, DC 20402.

Canadian Department of Communications

This Class A digital apparatus meets all requirements of the Canadian Interference-Causing Equipment Regulations.

Cet appareil numérique de la classe A respecte toutes les exigences du Règlement sur le matériel brouilleur du Canada.

About This Manual

Chapter 1
Introduction and Quick Start

Chapter 2
PCI-MXI-2 Configuration and Installation

Chapter 3
VXI-MXI-2 Configuration and Installation

Chapter 4
VME-MXI-2 Configuration and Installation

Chapter 5
NI-VXI Software Installation

Chapter 6
NI-VXI Configuration Utility

Chapter 7
Using the NI-VXI Software

Appendix A
Specifications

Appendix B
NI-VXI Software Overview

Appendix C
EEPROM Configuration

Appendix D
Common Questions

Appendix E
Customer Communication

Glossary

Index

Figures

Tables

This manual contains instructions for installing and configuring the National Instruments VXI-PCI8000 or VME-PCI8000 Series interface kit for Microsoft operating systems. The VXI-PCI8000 kit includes a VXI-MXI-2 module, which plugs into a VXI mainframe and links your PCI-based computer to the VXIbus. The VME-PCI8000 kit comes with a VME-MXI-2 that plugs into a VME chassis and links your PCI-based computer to the VMEbus. Both kits include the PCI-MXI-2 interface board, which links your computer to the MXIbus, and the NI-VXI bus interface software. The NI-VXI software is fully VXI*plug&play* compliant.

This manual uses the term *VXI/VME-PCI8000* when information applies to either kit and the term *VXI/VME-MXI-2* when information applies to either the VXI-MXI-2 or the VME-MXI-2. This manual uses the term *Windows 95/NT/3.1* when information applies to all three Windows operating systems.

Organization of This Manual

This manual is organized as follows:

- Chapter 1, *Introduction and Quick Start*, describes the VXI/VME-PCI8000 interface kits, lists what you need to get started, introduces the concepts of MXI-2, and includes a brief description of the hardware and software. This chapter also contains a *Quick Start* section, which has the basic information you need to install the VXI/VME-PCI8000 interface kit with a simple configuration, along with a *Default Settings* section, which lists the hardware and software default settings for easy reference.

- Chapter 2, *PCI-MXI-2 Configuration and Installation*, contains the instructions to configure and install the PCI-MXI-2 module.

- Chapter 3, *VXI-MXI-2 Configuration and Installation*, contains the instructions to configure and install the VXI-MXI-2 module. This chapter applies only if you ordered the VXI-PCI8000 interface kit.

- Chapter 4, *VME-MXI-2 Configuration and Installation*, contains the instructions to configure and install the VME-MXI-2 module. This chapter applies only if you ordered the VME-PCI8000 interface kit.

- Chapter 5, *NI-VXI Software Installation*, contains the instructions to install the NI-VXI software.

- Chapter 6, *NI-VXI Configuration Utility*, contains instructions for using the VXI Resource Editor utility of the NI-VXI software to configure the PCI-MXI-2 and the VXI-MXI-2 or VME-MXI-2.

- Chapter 7, *Using the NI-VXI Software*, discusses programming information for you to consider when developing applications that use the NI-VXI driver.

- Appendix A, *Specifications*, lists various module specifications of the PCI-MXI-2, VXI-MXI-2, and VME-MXI-2 such as physical dimensions and power requirements.

- Appendix B, *NI-VXI Software Overview*, lists and describes the main programs and files that make up the NI-VXI software.

- Appendix C, *EEPROM Configuration*, describes how to control the operation of the PCI-MXI-2 onboard EEPROM and how to fix an invalid EEPROM setting.

- Appendix D, *Common Questions*, addresses common questions you may have about using the NI-VXI bus interface software on the PCI-MXI-2 platform.

- Appendix E, *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.

- The *Index* contains an alphabetical list of key terms and topics in this manual, including the page where you can find each one.

Conventions Used in This Manual

The following conventions are used in this manual:

bold   Bold text denotes parameter names, menus, menu items, or dialog box buttons or options.

bold italic   Bold italic text denotes a note, caution, or warning.

`bold monospace`   Bold text in this font denotes the messages and responses that the computer automatically prints to the screen.

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

`monospace`   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, paths, directories, programs, subprograms, subroutines, device names, functions, variables, filenames, and extensions, and for statements and comments taken from program code.

♦   A diamond is used to denote hardware-specific or operating system-dependent material.

< >   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>.

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

How to Use This Documentation Set
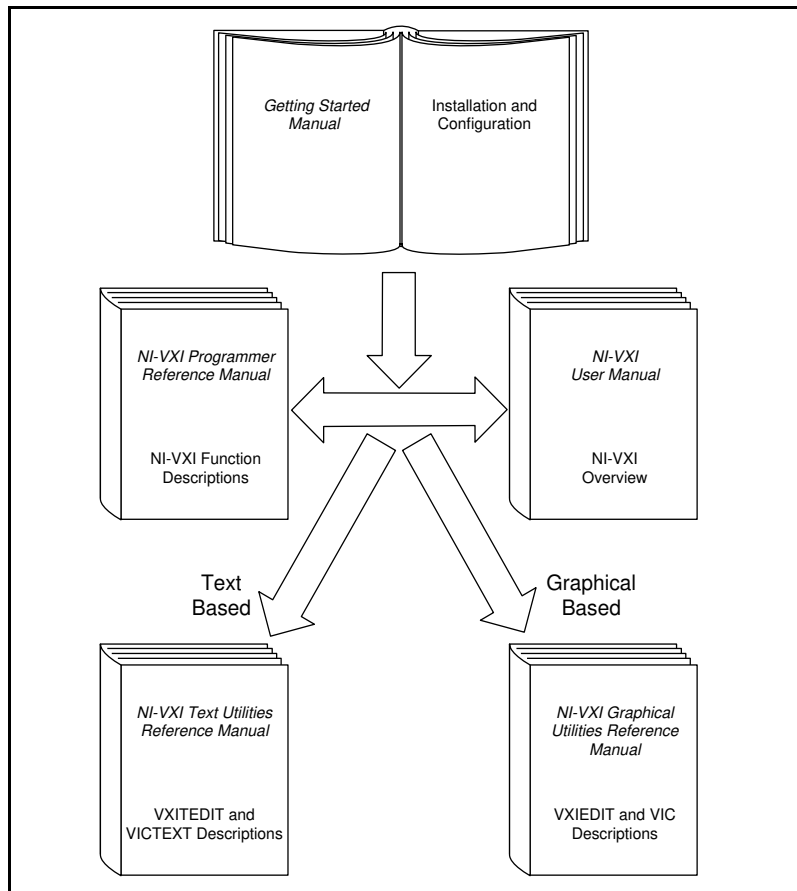
Begin by reading this getting started manual to guide you through the installation and configuration of the hardware and software. You should install and configure the components of the VXI/VME-PCI8000 kit in the order in which this manual describes them. Be sure to review the *Quick Start* and *Default Settings* sections in Chapter 1. The material in those sections may be all you need to get up and running with your VXI/VME-PCI8000 kit.

When you are familiar with the material in this getting started manual, you can begin to use the *NI-VXI User Manual*. This manual presents the concepts of VXI and prepares you for detailed explanations of the

NI-VXI functions. Study the descriptions of each function given in the *NI-VXI Programmer Reference Manual* to fully understand the purpose and syntax of each function.

Refer to the *NI-VXI Graphical Utilities Reference Manual* and the *NI-VXI Text Utilities Reference Manual* to learn more about the NI-VXI utilities.

Related Documentation

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

- ANSI/IEEE Standard 1014-1987, *IEEE Standard for a Versatile Backplane Bus: VMEbus*
- ANSI/IEEE Standard 1155-1993, *IEEE VMEbus Extensions for Instrumentation: VXIbus*
- ANSI/VITA 1-1994, *VME64*
- *Multisystem Extension Interface Bus Specification*, Version 2.0, National Instruments Corporation
- *PCI Local Bus Specification*, Revision 2.0, PCI Special Interest Group
- *VME-MXI-2 User Manual*, National Instruments Corporation
- *VXI-MXI-2 User Manual*, National Instruments Corporation
- VXI-6, *VXIbus Mainframe Extender Specification*, Rev. 1.0, VXIbus Consortium

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

Introduction and Quick Start

This chapter describes the VXI/VME-PCI8000 interface kits, lists what you need to get started, introduces the concepts of MXI-2, and includes a brief description of the hardware and software.

This chapter also contains a *Quick Start* section, which has the basic information you need to install the VXI/VME-PCI8000 interface kit with a simple configuration, along with a *Default Settings* section, which lists the hardware and software default settings for easy reference. You may find that these sections contain as much information as you need to get started with your VXI/VME-PCI8000 interface kit.

This manual uses the term *VXI/VME-PCI8000* when information applies to either the VXI-PCI8000 kit, which contains a VXI-MXI-2 module, or the VME-PCI8000 kit, which contains a VME-MXI-2 module. Similarly, the term *VXI/VME-MXI-2* means that information applies to either the VXI-MXI-2 or the VME-MXI-2. This manual also uses the term *Windows 95/NT/3.1* when information applies to all three Windows operating systems.

The following flowchart shows where to turn in this manual for more details on configuring and using the hardware and software.

How to Use This Manual

Chapter 1 → **Gather What You Need to Get Started**

Chapter 2 → **Configure and Install the PCI-MXI-2**

Using VXI or VME? — VXI / VME

Chapter 3 → **Configure and Install the VXI-MXI-2**

Chapter 4 → **Configure and Install the VME-MXI-2**

Chapter 5 → **Install the NI-VXI Software**

Chapter 6 → **Run the Software Configuration Utility**

Chapter 7 → **Review Programming Considerations**

Software and Utilities Reference → **Write Application Program**

VXI/VME-PCI8000 Kit Overview

The VXI/VME-PCI8000 interface kits link any computer with a PCI bus (hereafter referred to as a PCI-based computer) directly to the VXIbus or VMEbus using the high-speed Multisystem eXtension Interface bus (MXI-2).

A PCI-based computer equipped with a VXI-PCI8000 interface can function as a VXI Commander and Resource Manager. A PCI-based computer equipped with a VME-PCI8000 can function as a VMEbus master and/or slave device. The VXI/VME-PCI8000 makes your PCI-based computer behave as though it were plugged directly into the VXI/VME backplane as an embedded CPU VXI/VME module.

The software included with the kits is for *x*86/Pentium-based computers.

What You Need to Get Started

❑   A PCI-based computer

❑   VXIbus or VMEbus mainframe

❑   PCI-MXI-2 interface board

❑   VXI-MXI-2 or VME-MXI-2 interface module

❑   MXI-2 cable

❑   NI-VXI software media for the PCI-MXI-2

MXI-2 Description

MXI-2 is the second generation of the National Instruments MXIbus product line. The MXIbus is a general-purpose, 32-bit, multimaster system bus on a cable. MXI-2 expands the number of signals on a standard MXI cable by including VXI triggers, all VXI interrupts, CLK10, and all of the utility bus signals (SYSFAIL*, SYSRESET*, and ACFAIL*).

Because MXI-2 incorporates all of these new signals into a single connector, the triggers, interrupts, and utility signals can be extended not only to other mainframes but also to the local CPU in all MXI-2

products using a single cable. Thus, MXI-2 lets CPU interface boards such as the PCI-MXI-2 perform as though they were plugged directly into the VXI/VME backplane.

In addition, MXI-2 boosts data throughput performance past previous-generation MXIbus products by defining new high-performance protocols. MXI-2 is a superset of MXI. All accesses initiated by MXI devices will work with MXI-2 devices. However, MXI-2 defines synchronous MXI block data transfers which surpass previous block data throughput benchmarks. The new synchronous MXI block protocol increases MXI-2 throughput to a maximum of 33 MB/s between two MXI-2 devices. All National Instruments MXI-2 boards are capable of initiating and responding to synchronous MXI block cycles.

☞ **Note:**   *In the remainder of this manual, the term* **MXIbus** *refers to* **MXI-2.**

Hardware Description

The PCI-MXI-2 is a half-size, PCI-compatible plug-in circuit board that plugs into one of the expansion slots in your PCI-based computer. It links your PCI-based computer directly to the MXIbus and vice versa. Because the PCI-MXI-2 uses the same communication register set that other VXIbus message-based devices use, other MXIbus devices view the PCI-MXI-2 as a VXIbus device. The PCI-MXI-2 can also function as the MXIbus System Controller and can terminate the MXIbus signals directly on the PCI-MXI-2. In addition, you can have up to 16 MB of onboard DRAM on the PCI-MXI-2 that can be shared with the MXIbus and VXI/VMEbus and used as a dedicated data buffer.

The VXI-MXI-2 module is an extended-class, register-based VXIbus device with optional VXIbus Slot 0 capability so that it can reside in any slot in a C-size or D-size chassis.

☞ **Note:**   *D-size VXI mainframes have connections for a P3 connector. The VXI-MXI-2, however, does not have this connector and, if configured as a Slot 0 controller, cannot provide the necessary control for VXI devices that need P3 support.*

The VXI-MXI-2 uses address mapping to convert MXIbus cycles into VXIbus cycles and vice versa. By connecting to the PCI-MXI-2 board, the VXI-MXI-2 links the PCI bus to the VXIbus. The VXI-MXI-2 can

automatically determine whether it is located in VXI Slot 0 and/or if it is the MXIbus System Controller.

The VME-MXI-2 module is a single-slot, double-height VMEbus device with optional VMEbus System Controller functions. It uses address mapping to convert MXIbus cycles into VMEbus cycles and vice versa, just like the VXI-MXI-2. By connecting to the PCI-MXI-2 board, it links the PCI bus to the VMEbus. The VME-MXI-2 can automatically determine if it is located in the first slot of a VMEbus chassis and if it is the MXIbus System Controller.

Also, the VXI-MXI-2 and VME-MXI-2 automatically terminate the MXIbus if installed as the first or last device in the MXIbus. If installed in the middle of the MXIbus, both the VXI-MXI-2 and VME-MXI-2 automatically disable MXIbus termination. In addition, you can have up to 64 MB of onboard DRAM on the VXI-MXI-2 and VME-MXI-2 modules that can either be shared with the VXI/VMEbus and MXIbus or used as a dedicated data buffer.

The PCI-MXI-2, VXI-MXI-2, and VME-MXI-2 products achieve high-performance block transfer rates by integrating the MITE custom ASIC, a sophisticated dual-channel DMA controller with standard interfaces for VXI, VME, MXI, and PCI. By using MITE DMA to transfer data and commands to and from devices, the MITE frees up a computer's microprocessor to perform other tasks such as data analysis and presentation. In addition to DMA, the MITE incorporates both the new Synchronous MXI protocol and VME64 MBLT (8-byte block transfers in which both the address bus and data bus are used to transfer data) directly into the ASIC to perform the fastest transfer operation to instruments.

Software Description

The NI-VXI bus interface software for the PCI-MXI-2 includes a Resource Manager, graphical and text-based versions of an interactive VXI resource editor program, a comprehensive library of software routines for VXI/VME programming, and graphical and text-based versions of an interactive control program for interacting with VXI/VME. You can use this software to seamlessly program multiple-mainframe configurations and have software compatibility across a variety of VXI/VME controller platforms.

Software Configurations

There are four software configurations described in this manual:

- NI-VXI for DOS/Windows 3.1—you can use this version of the software to develop and run 16-bit DOS/Windows 3.1 applications. You can also use this software under Windows 95 if you intend to use 16-bit applications only.

- NI-VXI Upgrade for Windows 95—this is a compatibility release that extends your NI-VXI for DOS/Windows 3.1 to allow 32-bit applications running in Windows 95 to use the 16-bit driver. In this configuration you can run both 16-bit and 32-bit applications; however, the core of the driver is 16-bit.

- NI-VXI for Windows 95—this is a fully 32-bit native Plug and Play driver for Windows 95. You can run *only* 32-bit applications with this driver. You cannot use this driver in conjunction with either NI-VXI for DOS/Windows 3.1 or the NI-VXI Upgrade for Windows 95 to run 16-bit applications. Applications developed using this driver will run with NI-VXI for Windows NT without the need to recompile.

- NI-VXI for Windows NT—this is a 32-bit driver designed for Windows NT. You can use this version to develop and run 32-bit applications for Windows 95/NT.

Optional Software

Your VXI/VME-PCI8000 kit includes the NI-VXI bus interface software. In addition, you can use the National Instruments LabVIEW and LabWindows®/CVI application programs and instrument drivers to ease your programming task. These standardized programs match the modular virtual instrument capability of VXI and can reduce your VXI/VMEbus software development time. These programs are fully VXI*plug&play* compliant and feature extensive libraries of VXI instrument drivers written to take full advantage of direct VXI control.

LabVIEW is a complete programming environment that departs from the sequential nature of traditional programming languages and features a graphical programming environment.

LabWindows/CVI is an interactive C development environment for building test and measurement and instrument control systems. It includes interactive code-generation tools and a graphical editor for building custom user interfaces.

LabVIEW and LabWindows/CVI include all the tools needed for instrument control, data acquisition, analysis, and presentation. When you order the LabVIEW VXI Development System for Windows or the LabWindows/CVI VXI Development System for Windows, you also get more than 500 complete instrument drivers, which are modular, source-code programs that handle the communication with your instrument to speed your application development.

Quick Start

You can use this *Quick Start* section as a guide to quickly configure and operate your VXI or VME system using the PCI-MXI-2 and the VXI-MXI-2 or VME-MXI-2.

☞ **Note:**   ***Do not use this Quick Start information if you are installing NI-VXI for Windows 95 on top of either DOS/Windows 3.1 or the Windows 95 Upgrade. Instead, please refer to Chapter 5,*** *NI-VXI Software Installation***.**

The *Quick Start* summary assumes that you intend to perform a basic configuration as follows:

- You have one PCI-MXI-2 interface module, which you will install in your PCI-based computer as the Resource Manager (logical address 0).

- You have either one C-size VXI-MXI-2 or one 6U, B-size VME-MXI-2, which you will install in a VXI or VME chassis, respectively, and connect to the PCI-MXI-2.

- You will be using the NI-VXI software for initialization, configuration, and device interaction.

♦ **Windows 95 users**—We recommend that you install the NI-VXI software for Windows 95 first, and then install the hardware.

- You will use the default hardware and software settings.

    – The PCI-MXI-2 is the main controller, the VXI/VME Resource Manager, and a message-based device.

    – Your system contains only one VXI or VME chassis.

    – There is no shared memory used on the PCI-based computer, the PCI-MXI-2, or the VXI/VME-MXI-2.

Refer to the end of this chapter for a complete listing of the hardware and software default settings. If you need more information, or if you want to try a different configuration, please refer to the appropriate

hardware or software chapters in this manual, which describe the installation and configuration steps in greater detail.

The VXI/VME-PCI8000 kit works with DOS or Windows 95/NT/3.1, but the operating systems have different installation and configuration requirements. Be sure to observe any platform-specific instructions in the following information.

Hardware Installation

To guard against electrostatic discharge, touch the antistatic plastic package to a metal part of your computer before removing the PCI-MXI-2 from the package. Install the PCI-MXI-2 in an available PCI slot in your PCI-based computer.

By default, the PCI-MXI-2 automatically detects whether it should be the system controller on the MXIbus. Verify that the correct cable end labeled *Connect This End To Device Closest To MXIbus Controller In This Daisy Chain* is attached securely to the PCI-MXI-2. The cable must be connected in this manner so that the PCI-MXI-2 can correctly detect whether it should be the system controller on the MXIbus. For more information, refer to Chapter 2, *PCI-MXI-2 Configuration and Installation*.

You received either a VXI-MXI-2 or a VME-MXI-2 in your VXI/VME-PCI8000 kit. To guard against electrostatic discharge, touch the antistatic plastic package to a metal part of your computer before removing the VXI-MXI-2 or VME-MXI-2 from the package. Install the VXI-MXI-2 in the first slot of a VXI chassis, or install the VME-MXI-2 in the first slot of a VME chassis.

The VXI/VME-MXI-2 default configuration automatically detects whether it should be the VXI/VMEbus system controller. The VXI/VMEbus system controllers operate certain VXI/VMEbus lines as required for VXI/VME systems. Verify that any other VXI/VME devices with system controller capability that are located in the same chassis are not configured as system controller. Having more than one device configured as system controller will damage the VXI/VME system.

For VXI systems that include VME devices, ensure that the VME devices are not configured in the upper 16 KB (starting from 0xC000) of the A16 address space. This region is reserved for VXI device configuration registers, which are used for initializing, configuring, and

interacting with VXI devices. The PCI-MXI-2 and VME-MXI-2 also use this region for this purpose.

Also ensure that no VXI devices in your system are configured for either logical addresses 0 or 1. These are the default configurations for the PCI-MXI-2 and the VXI-MXI-2, respectively.

For more details on the VXI-MXI-2 or VME-MXI-2 hardware, refer to either Chapter 3, *VXI-MXI-2 Configuration and Installation*, or Chapter 4, *VME-MXI-2 Configuration and Installation*.

Platform-Specific Instructions

This version of the NI-VXI software works with Windows 95/NT/3.1 or DOS. Refer to the following section that is appropriate for your system.

Windows Users

1. Insert the NI-VXI disk labeled *Disk 1* into the floppy drive and run the Setup program to install the software.

2. **Windows 3.*x* users only**—Locate the NI-VXI group and run the **VXIinit** item. This utility initializes the PCI-MXI-2 hardware.

3. Execute the **Resman** item, which is located within the same NI-VXI group.

DOS Users

While the PCI-MXI-2 default configuration can get Windows users up and running without any changes, DOS users must reconfigure the PCI-MXI-2 to operate with applications that will use the NI-VXI software for DOS. After installing the NI-VXI software for DOS, you must use the VXI Resource Editor program, either VXIedit or VXItedit, to make these necessary changes.

1. Insert the NI-VXI disk labeled *Disk 1* into the floppy drive and run the INSTALL program to install the software.

2. Run the VXIedit or VXItedit utility.

3. Select the **PCI-MXI-2 Configuration Editor** from the options list.

4. Relocate the PCI-MXI-2 driver window to below 1 MB. Notice that the VXIedit or VXItedit utility warns you that the driver window is located above 1 MB. While this default setting is acceptable for Windows users, DOS users must enter a memory address below the 1 MB boundary to relocate the PCI-MXI-2 registers temporarily.

Select an unused section of the Upper Memory region (usually 0xC800 to 0xE800) to be used. Notice that this memory cannot be used by another device (such as an Ethernet card) or memory manager (such as `EMM386.EXE`). This placement is valid only while `VXIedit` or `VXItedit` is running.

5. To permanently place the board at the address, use the **Bus Configuration Editor** in `VXIedit` or `VXItedit`. Within this editor, enable the **Place below 1 MB** checkbox for the driver window. If this change causes any problems when rebooting the computer, refer to the *Fixing an Invalid EEPROM Configuration* section in Appendix C, *EEPROM Configuration*.

6. Update your configuration in `VXIedit` or `VXItedit` by selecting the **Update Current Configuration** option from the **PCI-MXI-2 Configuration Editor** main menu.

7. Reboot your computer by either using the reset button or turning off and on the machine. You cannot use the <Control-Alt-Delete> sequence for this reboot.

8. Execute `VXIinit` from the DOS prompt. This utility initializes the PCI-MXI-2 hardware. `VXIinit` also shows where the **PCI Configuration Manager** has placed your PCI-MXI-2. If this region conflicts with another board in your system, or if you experience any problems with your system, refer to the *User Window and Driver Window* section in Chapter 6, *NI-VXI Configuration Utility*. Notice that if you are using a memory manager (such as `EMM386.EXE`), you must exclude the region assigned to your PCI-MXI-2. This region may shift if you insert additional boards into your PCI system or move your PCI-MXI-2 to another slot.

9. Execute `RESMAN` to configure your VXI/VME system.

VME Users

When used with a VXI-MXI-2, `RESMAN` identifies and configures the VXI devices, including the VXI-MXI-2. When used with a VME-MXI-2, `RESMAN` configures the VME-MXI-2 to allow the PCI-MXI-2 to access devices in the VME chassis. `RESMAN` does not configure VME devices. The VME specification does not specify the initialization and configuration procedures that the VXI specification requires.

However, it is recommended that you enter the information about your VME devices into the `VXIedit` or `VXItedit` utility. `RESMAN` can then

properly configure the various device-specific VME address spaces and VME interrupt lines. For more information on configuring non-VXI devices in your VXI system, refer to the description of the **Non-VXI Device Configuration Editor** in Chapter 3, *VXI Resource Editor: VXIedit*, in the *NI-VXI Graphical Utilities Reference Manual*. For more details about installing the NI-VXI software, refer to Chapter 5, *NI-VXI Software Installation*, in this manual.

Device Interaction

After RESMAN has detected and configured all VXI/VME devices, you can view specific information on each device in your system by using the VXIedit or VXItedit utilities. These utilities include a **Resource Manager Display**, which contains a description for each device, including each VXI device's logical address.

You can interact with your VXI/VME devices by using the VIC or VICtext utilities. These utilities let you interactively control your VXI/VME devices without having to use a conventional programming language, LabVIEW for Windows 95/NT/3.1, or LabWindows/CVI.

Try the following in VIC or VICtext:

At the prompt:

ROOT>>

Type:

ROOT>>help vxiinreg

This help file shows you the syntax for this command, which reads VXI device configuration registers. The first argument is a logical address, and the second is the offset of the VXI device configuration register to be read.

Type:

ROOT>>vxiinreg 1,0

This should return a value, such as:

Return Status (0): SUCCESS.

value = 0x4ff6

If the value ends with `ff6`, you have successfully read the National Instruments manufacturer ID from the ID register for the **VXI/VME-MXI-2**.

You may now want to read the configuration registers from other VXI devices in your system using the command `vxiinreg`. This command accesses only the upper 16 KB of A16 space. Try reading the registers from one of the devices listed in the **Resource Manager Display** of either `VXIedit` or `VXItedit`. In this way, you can verify that your PCI-MXI-2 can access each of the devices in your VXI system successfully.

You can also access VXI and VME devices that are configured in A16, A24, and A32 address space by using the `vxiin` or `vxiout` commands. For more information regarding `VIC` operation and commands, refer to the *NI-VXI Graphical Utilities Reference Manual*. For more information regarding `VICtext` operation and commands, refer to the *NI-VXI Text Utilities Reference Manual*.

Default Settings

This section summarizes the hardware and software default settings for the VXI/VME-PCI8000 kit. If you need more information about a particular setting, or if you want to try a different configuration, please refer to the appropriate hardware or software chapters in this manual. The manual flowchart at the beginning of this chapter directs you to where to find the information you need.

PCI-MXI-2

This section summarizes the hardware and software default settings for the PCI-MXI-2.

Table 1-1. PCI-MXI-2 Hardware Default Settings

Hardware Component	Default Setting
U17 Switch 1 (FOV)	OFF: PCI-MXI-2 boots off the user-configured half of the EEPROM.
U17 Switch 2 (TST)	OFF: Factory configuration of the EEPROM is protected.

Table 1-1. PCI-MXI-2 Hardware Default Settings (Continued)

Hardware Component	Default Setting
U17 Switch 3 (POS)	OFF: *Do not alter this setting*.
U17 Switch 4 (CT)	ON: *Do not alter this setting*.
DRAM SIMM Installed	Per customer order

Table 1-2. PCI-MXI-2 Logical Address Configuration Editor Default Settings

Editor Field	Default Setting
Logical Address	0
Device Type	MBD
Address Space	A16
VXI Shared RAM Size	0 KB
Shared RAM Pool	0 KB
Lower Half Window Byte Swapping	Disabled
Lower Half Window Memory Select	System Memory
Upper Half Window Byte Swapping	Disabled
Upper Half Window Memory Select	System Memory
Map Upper and Lower Halves to Same Address	Disabled
Resource Manager Delay	5 s

Table 1-3. PCI-MXI-2 Device Configuration Editor Default Settings

Editor Field	Default Setting
Default Controller (LA-1)	First Remote Controller
System IRQ Level	1

Table 1-3. PCI-MXI-2 Device Configuration Editor Default Settings (Continued)

Editor Field	Default Setting
Servant Area Size	0
Number of Handlers	1
Number of Interrupters	0
Protocol Register	0xFF0
Read Protocol Response	0x8448

Table 1-4. PCI-MXI-2 Bus Configuration Editor Default Settings

Editor Field	Default Setting
MXI System Controller	Auto
MXI Bus Timeout	1 ms
MXI CLK10	Receive
MXI Transfer Limit	Unlimited
VXImove uses Synchronous MXI	Enabled
MXI-2 Auto Retry	Enabled
A24/A32 Write Posting	Disabled
User Window Size	64 KB
User Window Base	Auto
User Window Below 1 MB	Disabled
Driver Window Size	32 KB
Driver Window Base	Auto
Driver Window Below 1 MB	Disabled
Expansion ROM	Enabled

VXI/VME-MXI-2

This section summarizes the hardware and software default settings for the VXI-MXI-2 and VME-MXI-2.

Table 1-5. VXI-MXI-2 Hardware Default Settings

Hardware Component	Default Setting
Logical Address (U43)	1
VXIbus Slot 0/Non-Slot 0 (W2)	Automatic detection
VXIbus Local Bus (S8, S9)	Both OFF: Single VXI-MXI-2
VXIbus CLK10 Routing (W3)	From onboard oscillator
External Trigger Termination (S2)	OFF: Unterminated
SMB CLK10 Direction (S3)	OUT: Drive CLK10 signal
SMB CLK10 Termination (S4)	Ignored; effective only when S3 is set to IN.
Polarity of External SMB CLK10 (S5)	Inverted
MXIbus CLK10 Signal (S7)	Receive CLK10 from MXIbus
MXIbus Termination (U35 switches 1 and 2)	Automatic MXIbus termination: switch 2 set to NO; switch 1 ignored.
Configuration EEPROM (U35 switches 3 and 4)	User-modifiable; factory settings protected: both switches set to NO.
DRAM SIMMs Installed	Per customer order
SIMM Size Configuration (S6)	OFF if SIMMS are 4 M x 32 or larger; ON if smaller than 4 M x 32.

Table 1-6. VME-MXI-2 Hardware Default Settings

Hardware Component	Default Setting
A16 Base Address (U20)	Hex C040
VME-MXI-2 Intermodule Signaling (W2)	No user-defined pin selected
MXIbus Termination (U21 switches 3 and 4)	Automatic MXIbus termination: switch 3 OFF; switch 4 ignored
Configuration EEPROM (U21 switches 1 and 2)	User-modifiable; factory settings protected: both switches OFF
DRAM SIMMs Installed	Per customer order
SIMM Size Configuration (S2)	OFF if SIMMS are 4 M x 32 or larger; ON if smaller than 4 M x 32

Table 1-7. VXI/VME-MXI-2 Configuration Editor Default Settings

Editor Field	Default Setting
Logical Address	1 (set by hardware switch)
LA Selection	Set by hardware switch
Address Space	A24 *
Requested Memory	16 KB *
A16 Write Posting	Disabled
A24/A32 Write Posting	Disabled
Interlocked Mode	Disabled
VXI/VME System Controller	Auto
VXI/VME Bus Timeout Value	125 μs
VXI/VME Auto Retry	Disabled

Table 1-7. VXI/VME-MXI-2 Configuration Editor Default Settings (Continued)

Editor Field	Default Setting
VXI/VME Transfer Limit	256
VXI/VME Arbiter Type	Priority
VXI/VME Request Level	3
VXI/VME Fair Requester	Enabled
VXI/VME Arbiter Timeout	Enabled
MXI System Controller	Auto
MXI Bus Timeout Value	1 ms
MXI Auto Retry	Disabled
MXI Transfer Limit	Unlimited
MXI Parity Checking	Enabled
MXI Fair Requester	Disabled
MXI CLK10 Signal Direction	Set by hardware switch (VXI-MXI-2 only)
* Assumes no DRAM is installed. If DRAM is installed, the **Address Space** would be A32, and **Requested Memory** would match the amount of DRAM.	

PCI-MXI-2 Configuration and Installation

This chapter contains the instructions to configure and install the PCI-MXI-2 module.

Warning: *Electrostatic discharge can damage several components on your PCI-MXI-2 module. To avoid such damage in handling the module, touch the antistatic plastic package to a metal part of your computer chassis before removing the PCI-MXI-2 from the package.*

Configure the PCI-MXI-2

This section describes how to configure the following options on the PCI-MXI-2.

- Configuration EEPROM
- Onboard DRAM

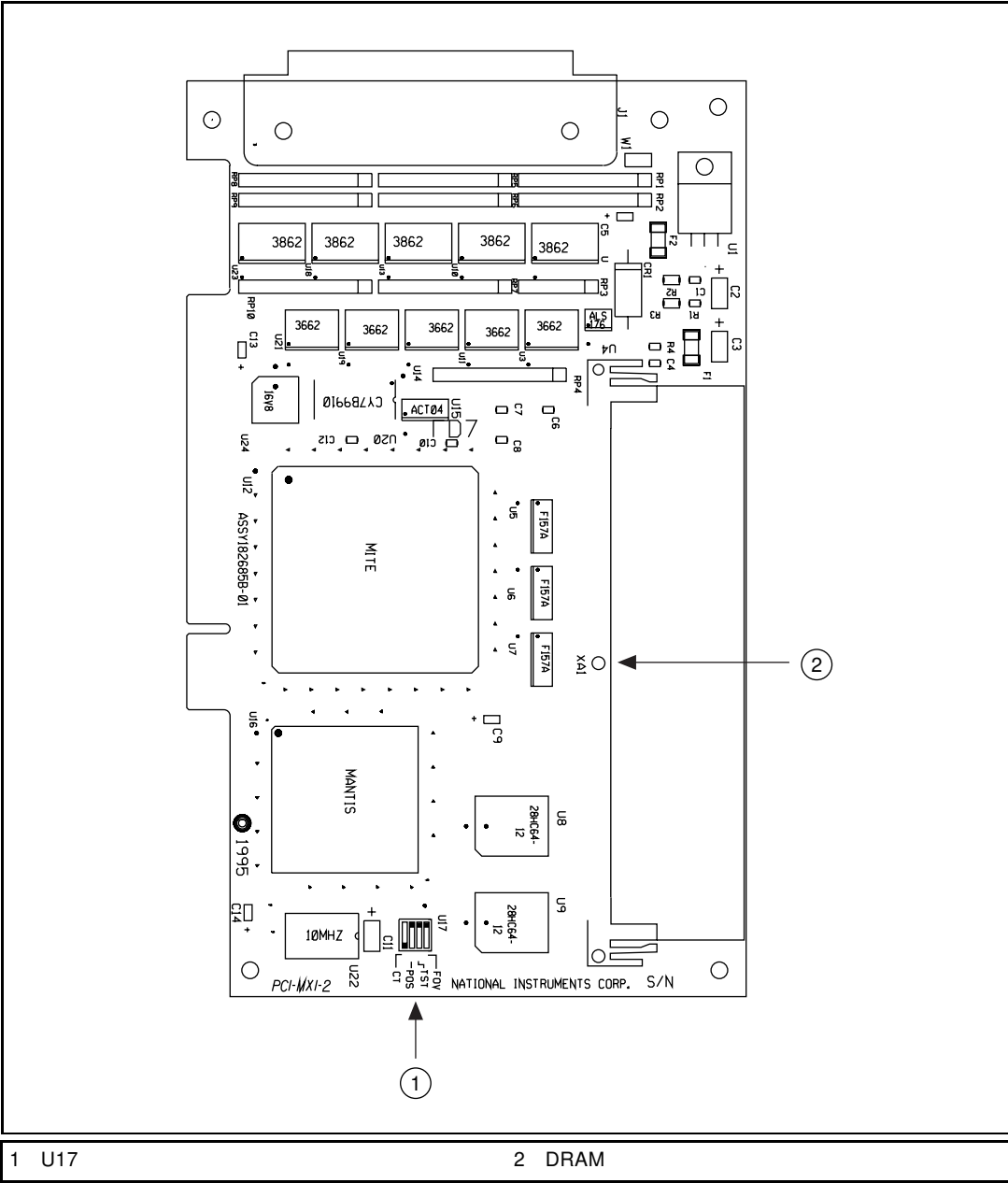Figure 2-1 shows the PCI-MXI-2. The drawing shows the location and factory-default settings on the module.

| 1 | U17 | 2 | DRAM |

Figure 2-1. PCI-MXI-2 Parts Locator Diagram

Configuration EEPROM

The PCI-MXI-2 has an onboard EEPROM, which stores default register values that are loaded at power-on. The EEPROM is divided into two halves—a factory-configuration half, and a user-configuration half—so you can modify the user-configurable half, while the factory-configured half stores a back-up of the default user settings. The factory configuration is a minimal configuration that allows you to boot your PCI-MXI-2 regardless of the changes made to the user configuration.

For information on configuring the onboard EEPROM, refer to Appendix C, *EEPROM Configuration*.

Onboard DRAM

The PCI-MXI-2 can accommodate one DRAM SIMM. Table 2-1 lists the SIMMS you can use. You can use 32-bit or 36-bit SIMMS since DRAM parity is not required. The PCI-MXI-2 can hold up to 16 MB of onboard memory. The PCI-MXI-2 supports DRAM speeds of 80 ns or faster. The maximum size for the DRAM SIMMs is 1 in.

Table 2-1. PCI-MXI-2 DRAM Configurations

SIMMs	Total DRAM	National Instruments Option?
—	0	—
256K x 32 or 256K x 36	1 MB	—
1M x 32 or 1M x 36	4 MB	YES
4M x 32 or 4M x 36	16 MB	YES

Install the PCI-MXI-2

This section contains general installation instructions for the PCI-MXI-2. Consult your computer user manual or technical reference manual for specific instructions and warnings.

1. Plug in your PCI-based computer before installing the PCI-MXI-2. The power cord grounds the computer and protects it from electrical damage while you are installing the module.

Warning: ***To protect both yourself and the computer from electrical hazards, the computer should remain off until you are finished installing the PCI-MXI-2 module.***

2. Remove the top cover or access port to the PCI bus.

3. Select any available PCI expansion slot.

4. Locate the metal bracket that covers the cut-out in the back panel of the chassis for the slot you have selected. Remove and save the bracket-retaining screw and the bracket cover.

5. Touch the metal part of the power supply case inside the computer to discharge any static electricity that might be on your clothes or body.

6. Line up the PCI-MXI-2 with the MXI-2 connector near the cut-out on the back panel. Slowly push down on the top of the PCI-MXI-2 until its card-edge connector is resting on the expansion slot receptacle. Using slow, evenly distributed pressure, press the PCI-MXI-2 straight down until it seats in the expansion slot.

7. Reinstall the bracket-retaining screw to secure the PCI-MXI-2 to the back panel rail.

8. Check the installation.

9. Replace the computer cover.

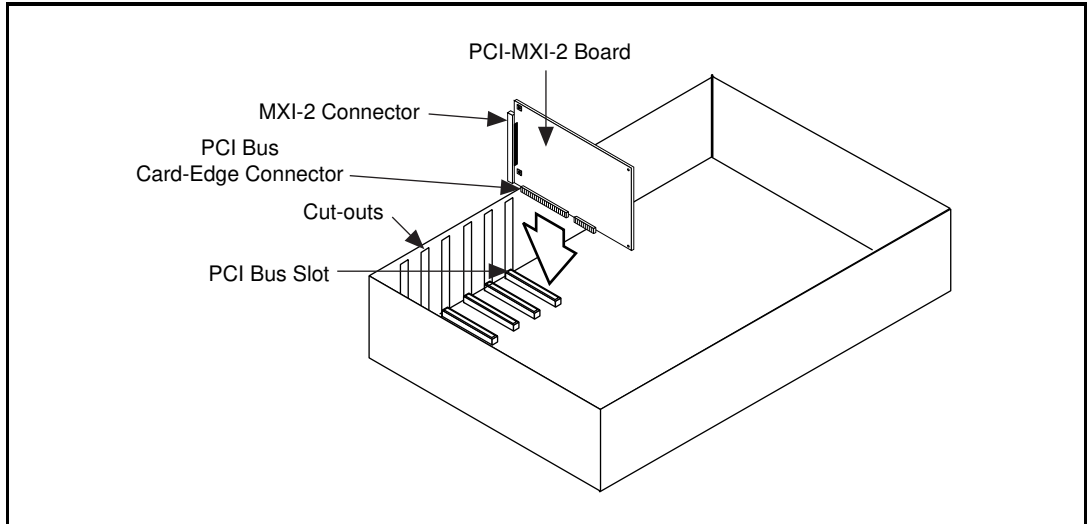Figure 2-2 shows how to install the PCI-MXI-2.

Figure 2-2. PCI-MXI-2 Installed in a Computer

VXI-MXI-2 Configuration and Installation

This chapter contains the instructions to configure and install the VXI-MXI-2 module. This chapter applies only if you ordered the VXI-PCI8000 interface kit. If you ordered the VME-PCI8000 kit, skip this chapter and refer to Chapter 4, *VME-MXI-2 Configuration and Installation*.

Warning: *Electrostatic discharge can damage several components on your VXI-MXI-2 module. To avoid such damage in handling the module, touch the antistatic plastic package to a metal part of your VXI chassis before removing the VXI-MXI-2 from the package.*

Configure the VXI-MXI-2

This section describes how to configure the following options on the VXI-MXI-2.

- VXIbus logical address
- VXIbus Slot 0/Non-Slot 0
- VXIbus local bus
- VXIbus CLK10 routing
- Trigger input termination
- MXIbus termination
- Configuration EEPROM
- Onboard DRAM

Figure 3-1 shows the VXI-MXI-2 as it would appear when facing the right side cover. The drawing shows the location and factory-default settings of most of the configuration switches and jumpers on the module. Notice that switch S6 (called out as number 8 in the figure) is accessible only by removing the front cover.
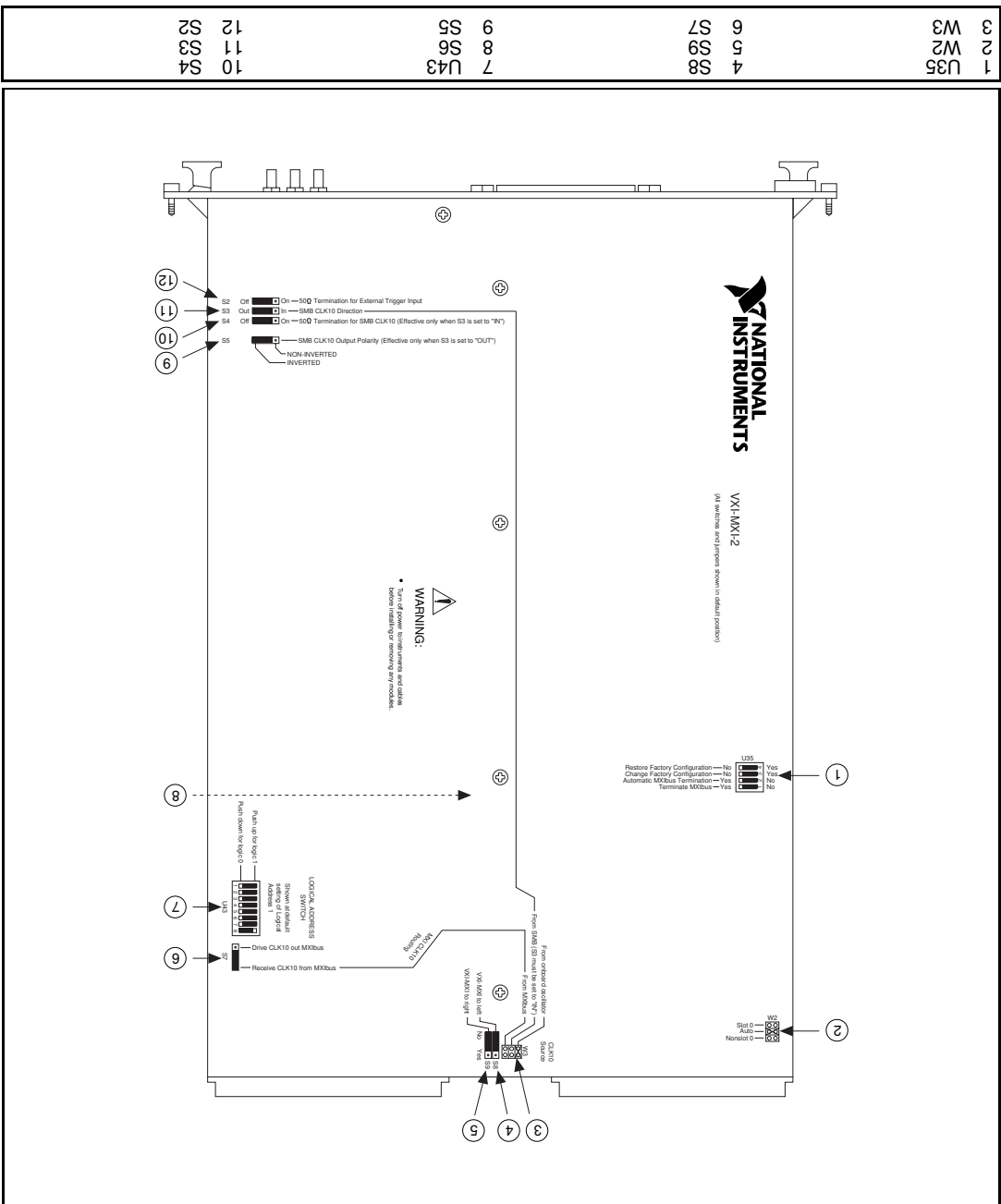
Front Panel Features

The VXI-MXI-2 has the following front panel features.

- Three front panel LEDs

  - **SYSFAIL** LED indicates that the VMEbus SYSFAIL line is asserted.

  - **MXI** LED indicates when the VXI-MXI-2 is accessed from the MXIbus.

  - **VXI** LED indicates when the VXI-MXI-2 is accessed from the VXIbus.

- MXIbus connector

- Three SMB connectors

  - External clock

  - Trigger output

  - Trigger input

- System reset push-button

Removing the Metal Enclosure

The VXI-MXI-2 is housed in a metal enclosure to improve EMC performance and to provide easy handling. Because the enclosure includes cutouts to facilitate changes to the switch and jumper settings, it should not be necessary to remove it under normal circumstances.

However, it is necessary to remove the enclosure if you want to change the amount of DRAM installed on the VXI-MXI-2. Switch S6, which is directly related to the amount of DRAM you want to install, is also accessible only by removing the enclosure. If you will be making this change, remove the four screws on the top, the four screws on the bottom, and the five screws on the right side cover of the enclosure. Refer to the *Onboard DRAM* section later in this chapter for details about changing DRAM.

VXIbus Logical Address

Each device in a VXIbus/MXIbus system is assigned a unique number between 0 and 254. This 8-bit number, called the *logical address*, defines the base address for the VXI configuration registers located on the device. With unique logical addresses, each VXIbus device in the system is assigned 64 bytes of configuration space in the upper 16 KB of A16 space.

Logical address 0 is reserved for the Resource Manager in the VXIbus system. Because the VXI-MXI-2 cannot act as a Resource Manager, do not configure the VXI-MXI-2 with a logical address of 0.

Some VXIbus devices have *dynamically configurable* logical addresses. These devices have an initial logical address of hex FF or 255, which indicates that they can be dynamically configured. While the VXI-MXI-2 does support dynamic configuration of VXI devices within its mainframe, it is itself a *statically configured* device and is preset at the factory with a VXI logical address of 1.

Ensure that no other statically configurable VXIbus devices have a logical address of 1. If they do, change the logical address setting of either the VXI-MXI-2 or the other device so that every device in the system has a unique associated logical address.

You can change the logical address of the VXI-MXI-2 by changing the setting of the 8-bit DIP switch labeled *LOGICAL ADDRESS SWITCH* (location designator U43) on the panel. The down position of the DIP switch corresponds to a logic value of 0 and the up position corresponds to a logic value of 1. Verify that the VXI-MXI-2 does not have the same logical address as any other statically configured VXIbus device in your system. Remember that logical addresses hex 0 and FF are not allowed for the VXI-MXI-2. Also, when setting logical addresses, keep in mind the grouping requirements set by the system hierarchy. See VXI-6, *VXIbus Mainframe Extender Specification*, for more information on setting logical addresses on a multimainframe hierarchy.

Figure 3-2 shows switch settings for logical address hex 1 and C0.

LOGICAL ADDRESS
SWITCH
Shown at default
setting of Logical
Address 1

Push up for logic 1
Push down for logic 0

1 2 3 4 5 6 7 8

U43

a. Switch Set to Logical Address 1 (Default)

LOGICAL ADDRESS
SWITCH
Shown at default
setting of Logical
Address 1

Push up for logic 1
Push down for logic 0

1 2 3 4 5 6 7 8

U43

b. Switch Set to Logical Address Hex C0
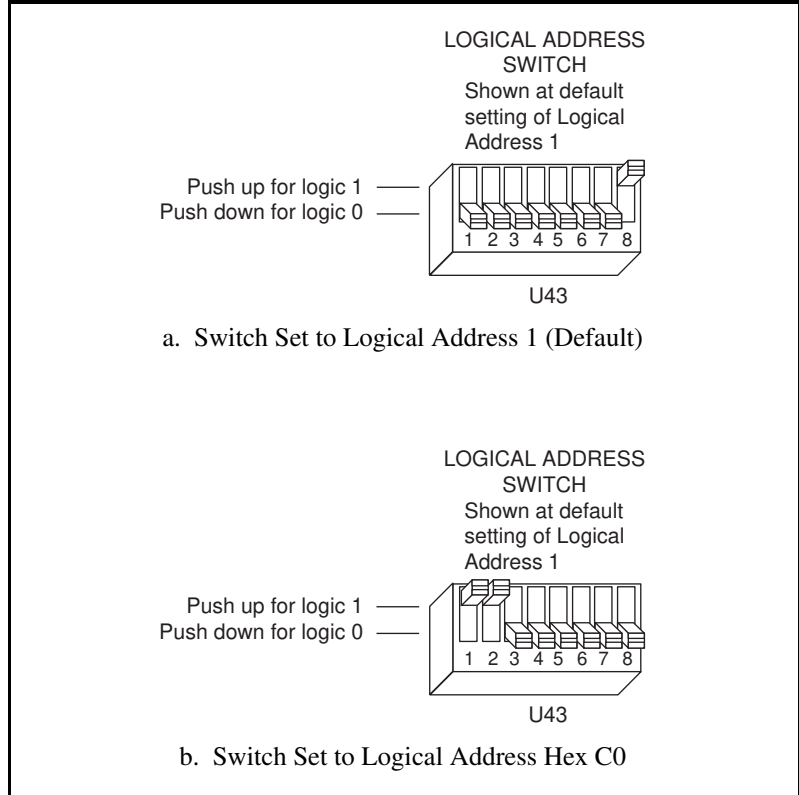
Figure 3-2. Logical Address Selection

VXIbus Slot 0/Non-Slot 0

The VXI-MXI-2 is configured at the factory to automatically detect if it is installed in Slot 0 of a VXIbus mainframe. With automatic Slot 0 detection, you can install the VXI-MXI-2 into any VXIbus slot.

You can manually configure the VXI-MXI-2 for either Slot 0 or Non-Slot 0 operation by defeating the automatic-detection circuitry. Use the three-position jumper W2 to select automatic Slot 0 detection, Slot 0, or Non-Slot 0 operation. Figure 3-3 shows these three settings.

Warning: *Do not install a device configured for Slot 0 into another slot without first reconfiguring it to either Non-Slot 0 or automatic configuration. Neglecting to do this could damage the device, the VXIbus backplane, or both.*

a. Automatic Slot 0 Detection (Default)

b. Manual Slot 0 Configuration

c. Manual Nonslot 0 Configuration
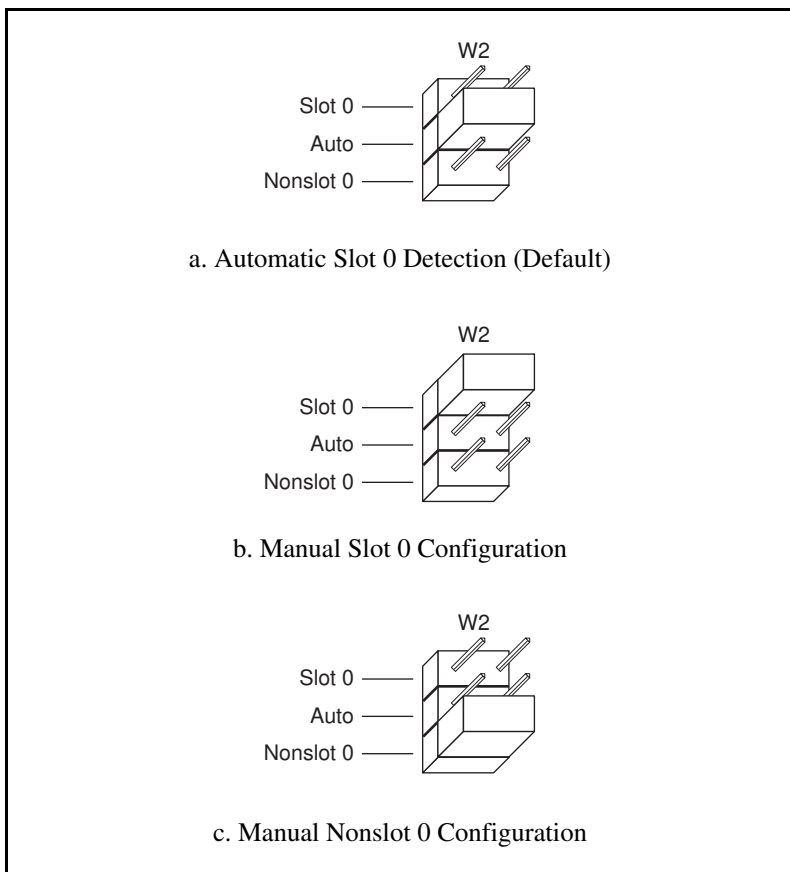
Figure 3-3. VXIbus Slot Configuration

When the VXI-MXI-2 is installed in Slot 0, it becomes the VMEbus System Controller. In this role, it has VMEbus Data Transfer Bus Arbiter circuitry that accepts bus requests on all four VMEbus request levels, prioritizes the requests, and grants the bus to the highest priority requester. As VMEbus System Controller, the VXI-MXI-2 also drives the 16 MHz VMEbus system clock by an onboard 16 MHz oscillator.
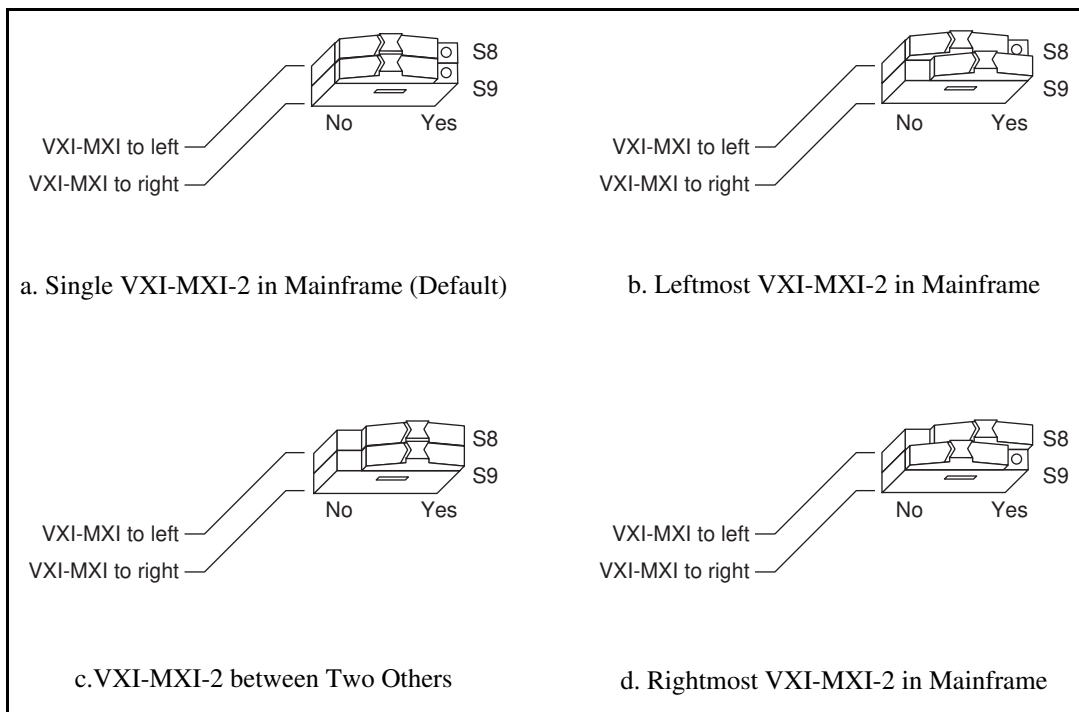
As required by the VXIbus specification, the VXI-MXI-2 drives the 10 MHz signal CLK10 on a differential ECL output when installed in Slot 0. When not installed in Slot 0, the VXI-MXI-2 only receives the CLK10 signal.

VXIbus Local Bus

If you will be installing more than one VXI-MXI-2 in a single VXIbus mainframe, you must configure the boards to use the local bus. The VXI-MXI-2 uses the local bus to pass a signal to the other VXI-MXI-2 modules in the mainframe to disable the VMEbus bus timeout unit (BTO) during cycles that map to the MXIbus. Because the local bus is used, you need to install all VXI-MXI-2 modules for a single mainframe in adjacent slots.

You will use two switches on the VXI-MXI-2 to select its position in relation to any other VXI-MXI-2 module in the mainframe. Use switch S9 when there is a VXI-MXI-2 to the right (higher numbered slot). Use S8 when there is a VXI-MXI-2 to the left (lower numbered slot).

Figure 3-4 shows four configuration settings for a VXI-MXI-2. Figure 3-4a illustrates the default setting, which is for a single VXI-MXI-2 in a mainframe. Use the setting in Figure 3-4b for the VXI-MXI-2 located to the left of all others. Figure 3-4c shows the setting to use if the VXI-MXI-2 is between two others. Use the setting of Figure 3-4d for the VXI-MXI-2 located to the right of all others.

a. Single VXI-MXI-2 in Mainframe (Default)

b. Leftmost VXI-MXI-2 in Mainframe

c. VXI-MXI-2 between Two Others

d. Rightmost VXI-MXI-2 in Mainframe

Figure 3-4. VXIbus Local Bus Configuration

VXIbus CLK10 Routing

When the VXI-MXI-2 is installed in Slot 0 of your mainframe, it supplies the VXIbus CLK10 signal. The VXI-MXI-2 can use three different sources to generate this signal: an onboard oscillator, the external CLK SMB connector, and the MXIbus CLK10 signal. Use the three-position jumper W3 to select these options, as shown in Figure 3-5.

Notice that Figure 3-5b and Figure 3-5c also show switches S3 and S7, respectively. You must configure these switches as shown when using the corresponding CLK10 source setting of W3.
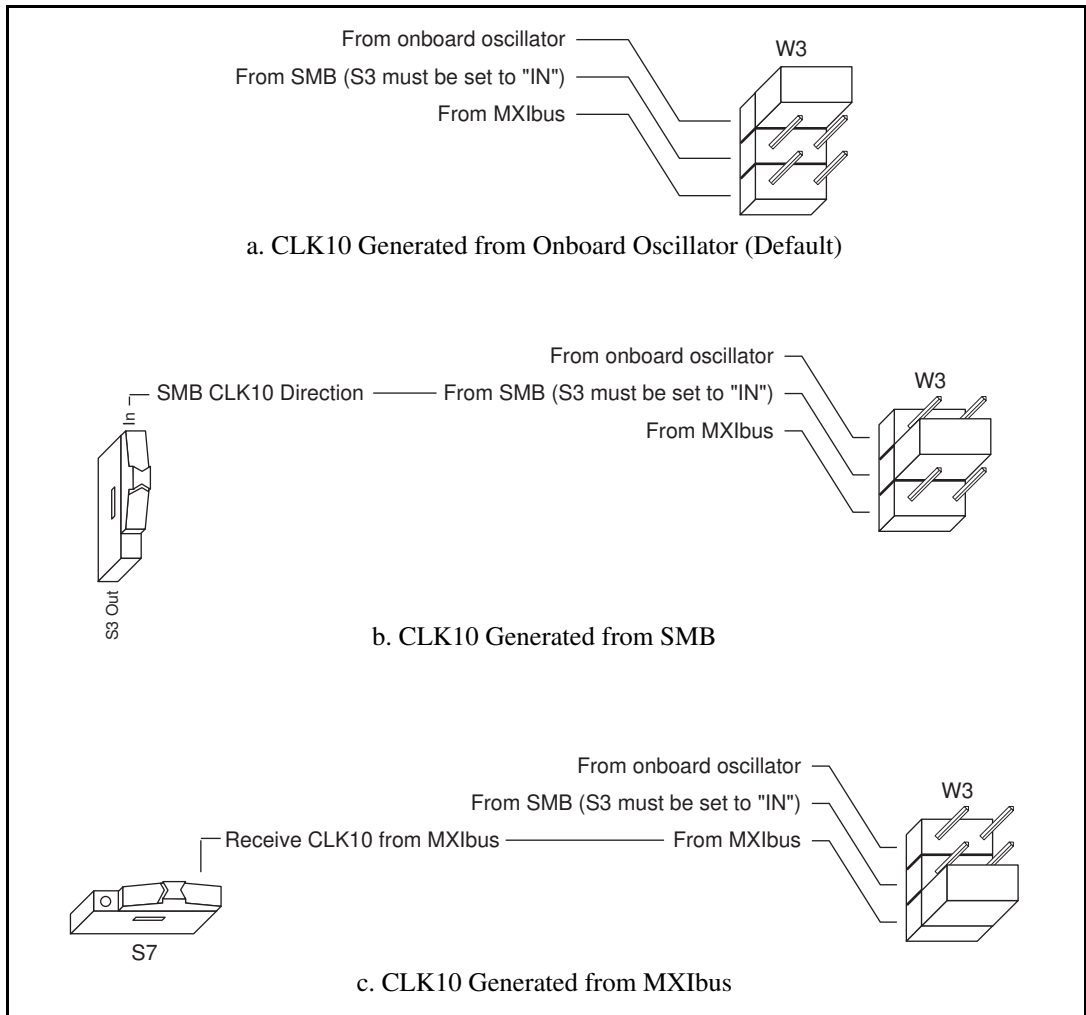
From onboard oscillator ———

From SMB (S3 must be set to "IN") ———

From MXIbus ———

W3

a. CLK10 Generated from Onboard Oscillator (Default)

SMB CLK10 Direction ——— From SMB (S3 must be set to "IN")

From onboard oscillator ———

From MXIbus ———

W3

In

S3 Out

b. CLK10 Generated from SMB

From onboard oscillator ———

From SMB (S3 must be set to "IN") ———

Receive CLK10 from MXIbus ——————— From MXIbus ———

W3

S7

c. CLK10 Generated from MXIbus

Figure 3-5. VXIbus CLK10 Routing

The VXI-MXI-2 can also be configured to drive the external CLK SMB signal from the VXIbus CLK10 signal. Switch S3 controls whether the VXI-MXI-2 drives or receives the external CLK SMB signal. If you change the S3 setting to drive CLK10 out the external CLK10 SMB connector, do not set the W3 jumper to receive the SMB CLK10 signal; instead use the settings shown in either Figure 3-5a or Figure 3-5c as appropriate.

When switch S3 is set so that the VXI-MXI-2 receives the SMB CLK10 signal, you have the option to add a 50 Ω termination to the signal by setting switch S4. S4 is unused—its setting does not matter—when S3 is configured to drive the external CLK SMB signal.

You can use an additional switch, S5, to control the polarity of the external CLK SMB signal when S3 is configured to drive it. S5 is unused—its setting does not matter—when S3 is configured to receive the external CLK SMB signal.

Figure 3-6 shows four configuration settings for the VXI-MXI-2. Figure 3-6a shows the default configuration, which is for driving the inverted external CLK SMB. Use the settings of Figure 3-6b to drive the noninverted external CLK SMB signal. Figure 3-6c illustrates the setting for receiving the external CLK SMB signal. Finally, you can configure the switches as shown in Figure 3-6d to receive the external CLK SMB signal with a 50 Ω termination.

☞ **Note:**   *The settings of any switches shown with this pattern (▨) have no bearing on the configuration described in any of the following figures.*

S2 Off — On − 50 Ω Termination for External Trigger Input

S3 Out — In − SMB CLK10 Direction

S4 Off — On − 50 Ω Termination for SMB CLK10 (Effective only when S3 is set to "IN")

S5 — − SMB CLK10 Output Polarity (Effective only when S3 is set to "OUT")

NON-INVERTED
INVERTED

a. Drive Inverted External CLK SMB (Default)

S2 Off — On − 50 Ω Termination for External Trigger Input

S3 Out — In − SMB CLK10 Direction

S4 Off — On − 50 Ω Termination for SMB CLK10 (Effective only when S3 is set to "IN")

S5 — − SMB CLK10 Output Polarity (Effective only when S3 is set to "OUT")

NON-INVERTED
INVERTED

b. Drive Non-Inverted External CLK SMB

S2 Off — On − 50 Ω Termination for External Trigger Input

S3 Out — In − SMB CLK10 Direction

S4 Off — On − 50 Ω Termination for SMB CLK10 (Effective only when S3 is set to "IN")

S5 — − SMB CLK10 Output Polarity (Effective only when S3 is set to "OUT")

NON-INVERTED
INVERTED

c. Receive External CLK SMB

S2 Off — On − 50 Ω Termination for External Trigger Input

S3 Out — In − SMB CLK10 Direction

S4 Off — On − 50 Ω Termination for SMB CLK10 (Effective only when S3 is set to "IN")

S5 — − SMB CLK10 Output Polarity (Effective only when S3 is set to "OUT")

NON-INVERTED
INVERTED

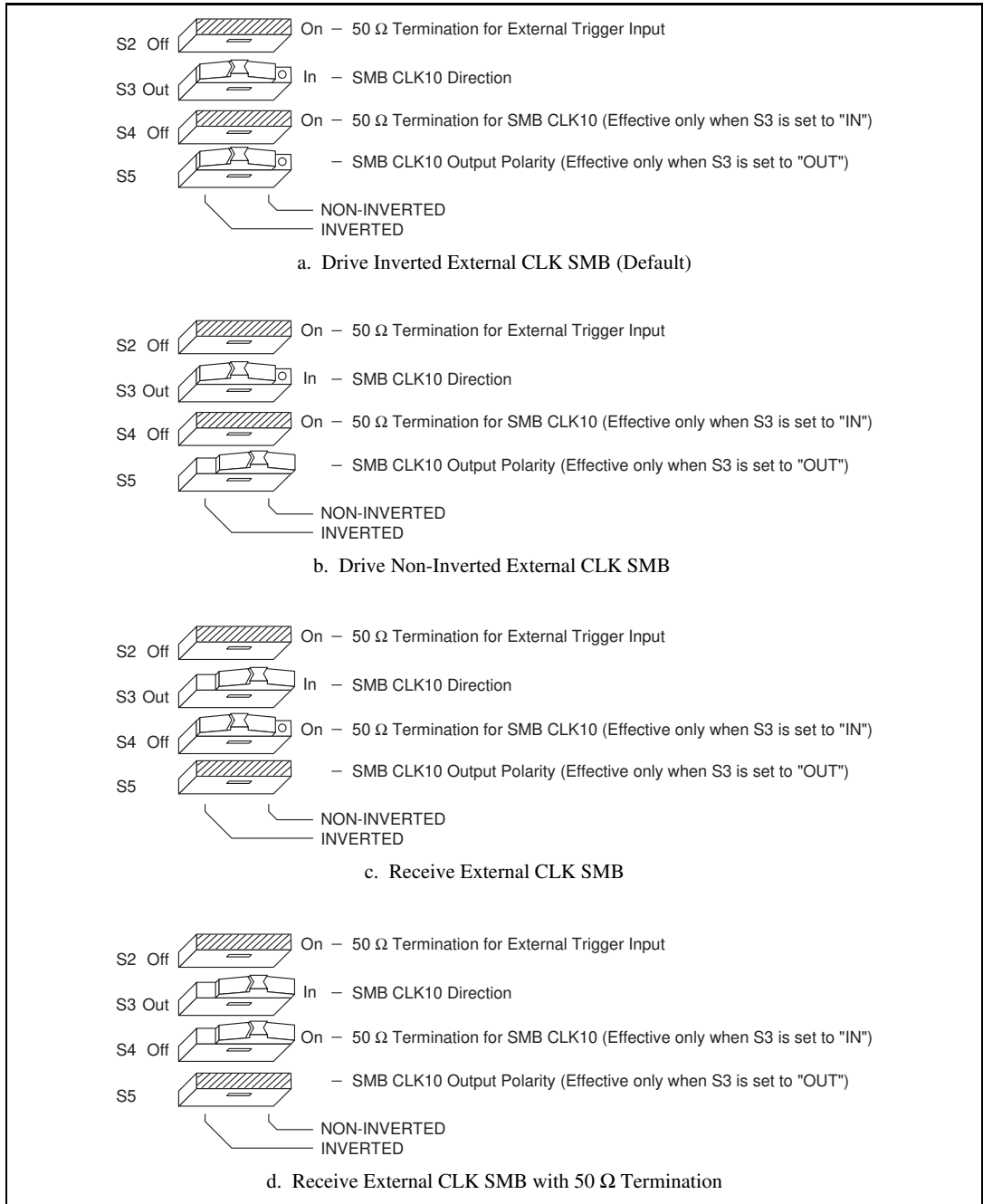d. Receive External CLK SMB with 50 Ω Termination

Figure 3-6. SMB CLK10 Settings

The VXI-MXI-2 can also drive or receive the MXIbus CLK10 signal. Switch S7 controls whether the VXI-MXI-2 drives MXIbus CLK10 from the VXIbus CLK10 or receives MXIbus CLK10. As shown earlier in Figure 3-5c, if W3 is configured to use the MXIbus CLK10 to generate the VXIbus CLK10 signal, switch S7 must be configured to receive MXIbus CLK10. This is shown again in Figure 3-7a below. If you change the S7 setting to drive CLK10 out the MXIbus, do not set the W3 jumper to receive the MXIbus CLK10; instead use the settings shown in Figure 3-5a or Figure 3-5b as appropriate.

Warning: *Do not configure more than one MXIbus device to drive the MXIbus CLK10. Having a second device driving MXIbus CLK10 could damage the device.*

Figure 3-7 shows the configuration settings for receiving and driving MXIbus CLK10, respectively.

— Drive CLK10 out MXIbus

— Receive CLK10 from MXIbus

a. Receive CLK10 from MXIbus (Default)

— Drive CLK10 out MXIbus

— Receive CLK10 from MXIbus

b. Drive MXIbus CLK10 from VXIbus CLK10
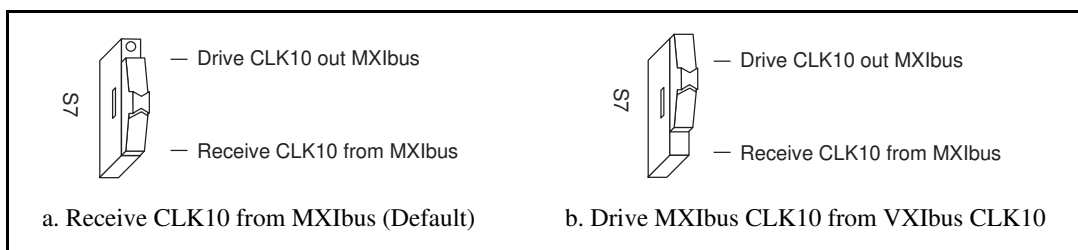
Figure 3-7. Receiving or Driving MXIbus CLK10

Trigger Input Termination

You can use switch S2 to terminate the external trigger input SMB with 50 Ω. Figure 3-8a shows the default setting for a nonterminated trigger input SMB. Use the setting of Figure 3-8b to terminate the trigger input SMB. Switch S2 is located above switches S3, S4, and S5, which have no effect on this configuration.

S2 Off   On − 50 Ω Termination for External Trigger Input

S3 Out   In − SMB CLK10 Direction

S4 Off   On − 50 Ω Termination for SMB CLK10 (Effective only when S3 is set to "IN")

S5   − SMB CLK10 Output Polarity (Effective only when S3 is set to "OUT")

                           NON-INVERTED
                           INVERTED

a. Non-Terminated External Trigger Input SMB (Default)

S2 Off   On − 50 Ω Termination for External Trigger Input

S3 Out   In − SMB CLK10 Direction

S4 Off   On − 50 Ω Termination for SMB CLK10 (Effective only when S3 is set to "IN")

S5   − SMB CLK10 Output Polarity (Effective only when S3 is set to "OUT")

                           NON-INVERTED
                           INVERTED

b. Terminated External Trigger Input SMB with 50 Ω

Figure 3-8. SMB Trigger Input Termination

MXIbus Termination

The first and last MXIbus devices connected to the MXIbus—whether it is a single MXI-2 cable or daisy-chained MXI-2 cables—must terminate the MXIbus. Any MXIbus devices in the middle of a MXIbus daisy chain must *not* terminate the MXIbus.

The VXI-MXI-2 automatically senses whether it is at either end of the MXIbus cable to terminate the MXIbus. You can manually control MXIbus termination by defeating the automatic circuitry. Use switches 1 and 2 of the four-position switch at location U35 to control whether MXIbus termination is automatic (Figure 3-9a), on (Figure 3-9b), or off (Figure 3-9c). The settings of switches 3 and 4 have no effect on MXIbus termination.

Use switch 2 of U35 to select whether you want the VXI-MXI-2 to automatically control termination of the MXIbus. Switch 1 of U35 lets you manually control whether to terminate the MXIbus when automatic termination is turned off. Switch 1 has no effect when switch 2 is set for automatic MXIbus termination; you must turn off automatic termination if you want to manually control termination.
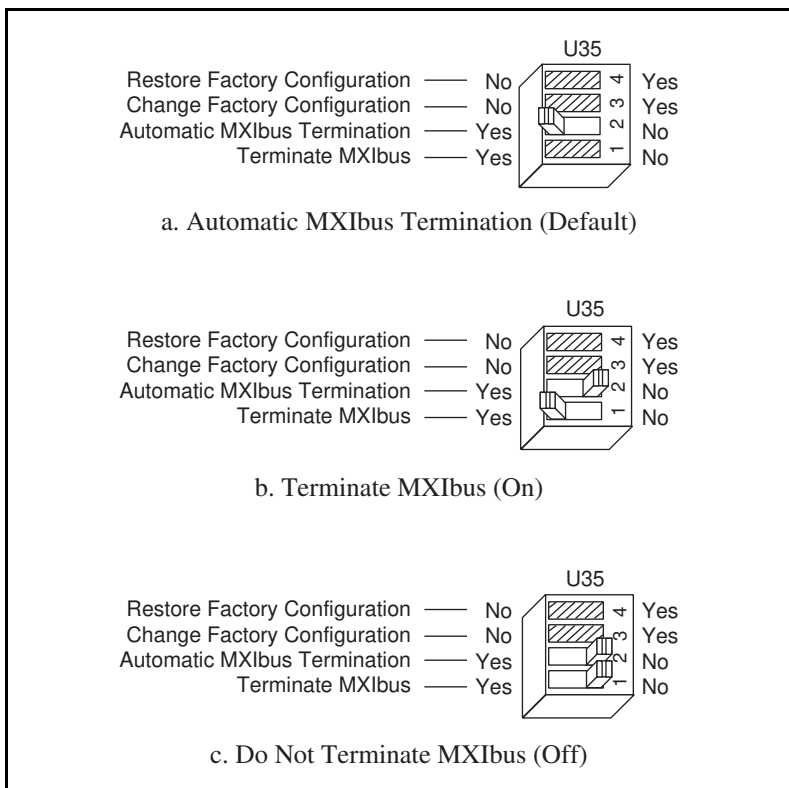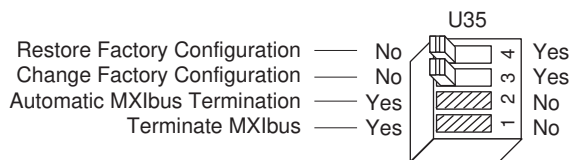

a. Automatic MXIbus Termination (Default)

b. Terminate MXIbus (On)

c. Do Not Terminate MXIbus (Off)

Figure 3-9. MXIbus Termination

Configuration EEPROM

The VXI-MXI-2 has an onboard EEPROM, which stores default register values that are loaded at power-on. The EEPROM is divided into two halves—a factory-configuration half, and a user-configuration half. Both halves were factory configured with the same configuration values so you can modify the user-configurable half, while the factory-configured half stores a back-up of the default user settings.

Use switches 3 and 4 of the four-position switch at location U35 to control the operation of the EEPROM. The Restore Factory Configuration switch (switch 4) causes the VXI-MXI-2 to boot off the factory-configured half instead of the user-modified settings. This is useful in the event that the user-configured half of the EEPROM becomes corrupted in such a way that the VXI-MXI-2 boots to an unusable state.

The Change Factory Configuration switch (switch 3 of U35) lets you change the factory-default configuration settings by permitting writes to the factory settings section of the EEPROM. This switch serves as a safety measure and should not be needed under normal circumstances. When this switch is off (its default setting) the factory configuration of the EEPROM is protected, so any writes to the factory area will be ignored. The factory area is protected regardless of the setting of switch 4 of U35.

Figure 3-10 shows the configuration settings for EEPROM operation. The settings of switches 1 and 2 have no effect on EEPROM configuration.

U35

Restore Factory Configuration —— No   4   Yes
Change Factory Configuration —— No   3   Yes
Automatic MXIbus Termination —— Yes   2   No
Terminate MXIbus —— Yes   1   No

a. Boot from User Configuration (Factory Configuration Protected) (Default)

U35

Restore Factory Configuration —— No   4   Yes
Change Factory Configuration —— No   3   Yes
Automatic MXIbus Termination —— Yes   2   No
Terminate MXIbus —— Yes   1   No

b. Boot from Factory Configuration (Factory Configuration Protected)

U35

Restore Factory Configuration —— No   4   Yes
Change Factory Configuration —— No   3   Yes
Automatic MXIbus Termination —— Yes   2   No
Terminate MXIbus —— Yes   1   No

c. Boot from User Configuration (Factory Configuration Unprotected)

U35

Restore Factory Configuration —— No   4   Yes
Change Factory Configuration —— No   3   Yes
Automatic MXIbus Termination —— Yes   2   No
Terminate MXIbus —— Yes   1   No
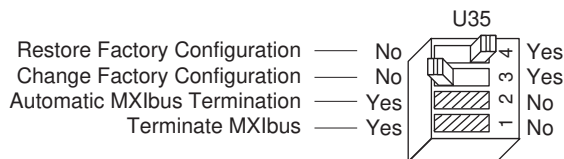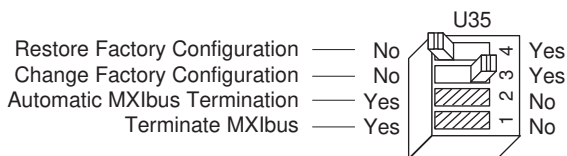
d. Boot from Factory Configuration (Factory Configuration Unprotected)

Figure 3-10. EEPROM Operation

Onboard DRAM

The VXI-MXI-2 can accommodate up to two 1.35 in. DRAM SIMMs. Table 3-1 lists the SIMMs you can use. You can use 32-bit or 36-bit SIMMs since DRAM parity is not required. Because the VXI-MXI-2 supports only one organization at a time, all SIMMs installed must be of the same type. Use Bank 0 first when installing the SIMMs. This allows you to install up to 64 MB. The VXI-MXI-2 supports DRAM speeds of 80 ns or faster.

Switch S6 is used to select the size of each SIMM. The SIMM sockets and S6 are accessible only by removing the right-side cover. To access these components, remove the four screws on the top, the four screws on the bottom, and the five screws on the right-side cover of the metal enclosure. If the SIMMs are 4 M x 32 or larger, S6 should be in the OFF setting as shown in Figure 3-11a. For SIMMs *smaller* than 4 M x 32, use the ON setting as shown in Figure 3-11b.

SIMM SIZE
(• = 4M x 32 AND LARGER)   S6

a. 4 M x 32 and Larger

SIMM SIZE
(• = 4M x 32 AND LARGER)   S6

b. Smaller than 4 M x 32
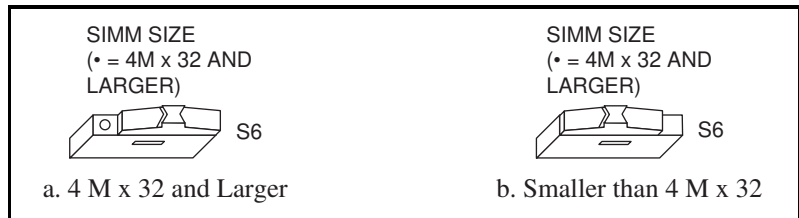
Figure 3-11. SIMM Size Configuration

Refer to Table 3-1 for how to adjust the switch (ON or OFF) for all supported DRAM configurations. Many of the DRAM options are available from National Instruments.

Table 3-1. VXI-MXI-2 DRAM Configurations

Bank 0	Bank 1	Total DRAM	National Instruments Option?	Switch Setting of S6
—	—	0	—	—
256 K x 32 or 256 K x 36	—	1 MB	—	ON
256 K x 32 or 256 K x 36	256 K x 32 or 256 K x 36	2 MB	—	ON
512 K x 32 or 512 K x 36	—	2 MB	—	ON
512 K x 32 or 512 K x 36	512 K x 32 or 512 K x 36	4 MB	—	ON
1 M x 32 or 1 M x 36	—	4 MB	YES	ON
1 M x 32 or 1 M x 36	1 M x 32 or 1 M x 36	8 MB	—	ON
2 M x 32 or 2 M x 36	—	8 MB	YES	ON
2 M x 32 or 2 M x 36	2 M x 32 or 2 M x 36	16 MB	—	ON
4 M x 32 or 4 M x 36	—	16 MB	YES	OFF
4 M x 32 or 4 M x 36	4 M x 32 or 4 M x 36	32 MB	—	OFF
8 M x 32 or 8 M x 36	—	32 MB	YES	OFF
8 M x 32 or 8 M x 36	8 M x 32 or 8 M x 36	64 MB	YES	OFF

Install the VXI-MXI-2

This section contains general installation instructions for the VXI-MXI-2. Consult your VXIbus mainframe user manual or technical reference manual for specific instructions and warnings.

1. Plug in your mainframe before installing the VXI-MXI-2. The power cord grounds the mainframe and protects it from electrical damage while you are installing the module.

Warning: *To protect both yourself and the mainframe from electrical hazards, the mainframe should remain off until you are finished installing the VXI-MXI-2 module.*

2. Remove or open any doors or covers blocking access to the mainframe slots.

3. If you are installing the VXI-MXI-2 into a D-size mainframe, install a support designed for installing C-size boards in D-size mainframes. The VXI-MXI-2 has no P3 connector and cannot provide P3 Slot 0 control to VXI devices requiring this capability.

Warning: *If the VXI-MXI-2 is not configured for automatic Slot 0 detection, be certain that the slot you select in your VXIbus mainframe matches the VXI-MXI-2 configuration as either a Slot 0 device or a Non-Slot 0 device. If you install your VXI-MXI-2 into a slot that does not correspond with the jumper setting, you risk damage to the VXI-MXI-2, the VXIbus backplane, or both.*

4. Insert the VXI-MXI-2 in the slot you have selected by aligning the top and bottom of the board with the card-edge guides inside the mainframe. Slowly push the VXI-MXI-2 straight into the slot until its plug connectors are resting on the backplane receptacle connectors. Using slow, evenly distributed pressure, press the VXI-MXI-2 straight in until it seats in the expansion slot. The front panel of the VXI-MXI-2 should be even with the front panel of the mainframe.

5. Tighten the retaining screws on the top and bottom edges of the front panel.

6. Check the installation.

7. Connect the cables as described in the following section before restoring power.

8. Replace or close any doors or covers to the mainframe.

Connect the MXIbus Cable

There are two basic types of MXI-2 cables. MXI-2 cables can have either a single connector on each end or a single connector on one end and a double connector on the other end.

Connect the labeled end of the cable to the MXI-2 device that will be the MXIbus System Controller. Connect the other end of the cable to the other device. Be sure to tighten the screw locks to ensure proper pin connection.

Figure 3-12 shows the correct cabling for a VXI system containing a PCI-MXI-2 board in a PCI-based computer cabled to a VXI-MXI-2 module residing in Slot 0 of a VXIbus mainframe. Notice that you can expand your system to include other devices by using an additional MXI-2 cable. However, in such a case the first cable needs to have a double connector on one end. You can use a cable with a single connector on each end to connect the last device on the MXIbus.
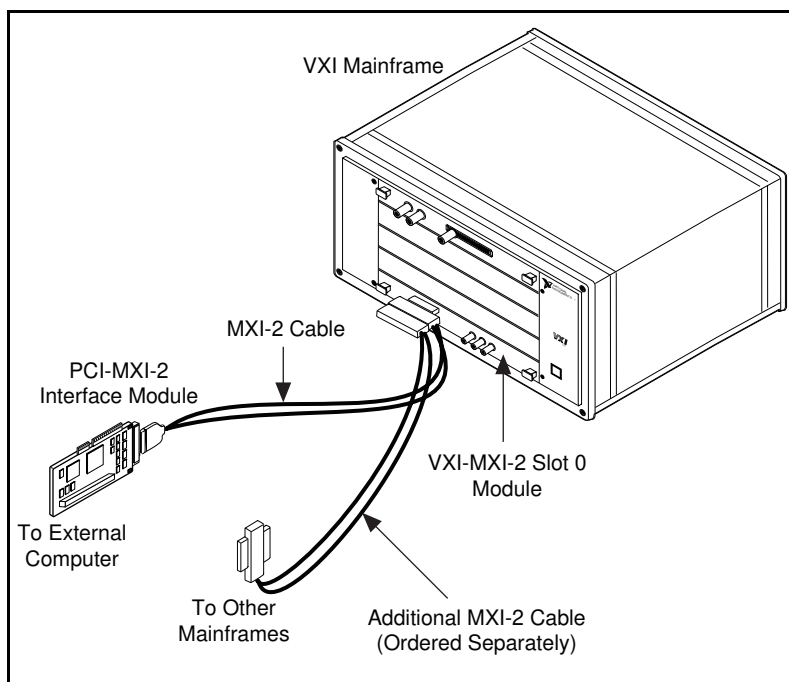
Figure 3-12. MXI-2 Cable Configuration Using a PCI-MXI-2 and a VXI-MXI-2

When you have properly connected the MXI-2 cable, power on the VXIbus mainframe and then the computer.

☞   **Note:**   *Always turn on the mainframe first. Doing so makes it possible for your external computer to access the VXI boards in the mainframe upon startup.*

VME-MXI-2 Configuration and Installation

This chapter contains the instructions to configure and install the VME-MXI-2 module. This chapter applies only if you ordered the VME-PCI8000 interface kit. If you ordered the VXI-PCI8000 kit, you should refer to Chapter 3, *VXI-MXI-2 Configuration and Installation*.

> **Warning:** *Electrostatic discharge can damage several components on your VME-MXI-2 module. To avoid such damage in handling the module, touch the antistatic plastic package to a metal part of your VMEbus chassis before removing the VME-MXI-2 from the package.*

Configure the VME-MXI-2

This section describes how to configure the following options on the VME-MXI-2.

- VMEbus A16 base address
- VME-MXI-2 intermodule signaling
- MXIbus termination
- Configuration EEPROM
- Onboard DRAM

The VME-MXI-2 automatically detects if it is located in the first slot of the chassis to perform the VMEbus System Controller functions. It is not necessary to configure the VME-MXI-2 System Controller option. The module can be installed in any slot of a VMEbus chassis.

Figure 4-1 shows the VME-MXI-2. The drawing shows the location and factory-default settings of the configuration switches and jumpers on the module.

1	S2	3	W2	5	DRAM Bank 1
2	U21	4	U20	6	DRAM Bank 0

Figure 4-1. VME-MXI-2 Parts Locator Diagram

Front Panel Features

The VME-MXI-2 has the following front panel features.

- Three front panel LEDs

    - **SYSFAIL** LED indicates that the VMEbus SYSFAIL line is asserted.

    - **MXI** LED indicates when the VME-MXI-2 is accessed from the MXIbus.

    - **VME** LED indicates when the VME-MXI-2 is accessed from the VMEbus.

- MXIbus connector
- System reset push-button

VMEbus A16 Base Address

The VME-MXI-2 requires 64 B (bytes) of A16 space for its configuration registers. It uses the *logical address* scheme of the VXIbus specification, in which each device is assigned an 8-bit value called the logical address. This logical address allocates 64 B of space to the device within the upper quarter of A16 space. The VME-MXI-2 cannot be configured to locate its registers in the lower three quarters of A16 space. The A16 base address of the VME-MXI-2 will be address lines 15 and 14 high with address lines 13 through 6 matching the logical address of the VME-MXI-2, and address lines 5 through 0 low. In other words, the A16 base address of the VME-MXI-2 module's 64 B register set is as calculated below:

*base address = C000 hex + (logical address) * 40 hex*

The factory-default logical address for the VME-MXI-2 is 1, which locates the registers in the range C040 hex to C07F hex. You can change the logical address of the VME-MXI-2 by changing the setting of the 8-bit DIP switch at location designator U20. The ON position of the DIP switch corresponds to a logic value of 0, and the OFF position corresponds to a logic value of 1. Allowable logical addresses for the VME-MXI-2 range from 1 to 254 (hex FE). Verify that no other devices in your system use the A16 address space for the VME-MXI-2. If possible, configure all other VMEbus A16 devices to be located within the lower three quarters of A16 space. Also, when setting base addresses, keep in mind the grouping requirements set by the system hierarchy. See VXI-6, *VXIbus Mainframe Extender Specification*, for

more information on setting base addresses on a multimainframe hierarchy.

Figure 4-2 shows switch settings for A16 base address hex C040 and F000.

a. Switch Set to A16 Base Address hex C040 (Default)

b. Switch Set to A16 Base Address Hex F000
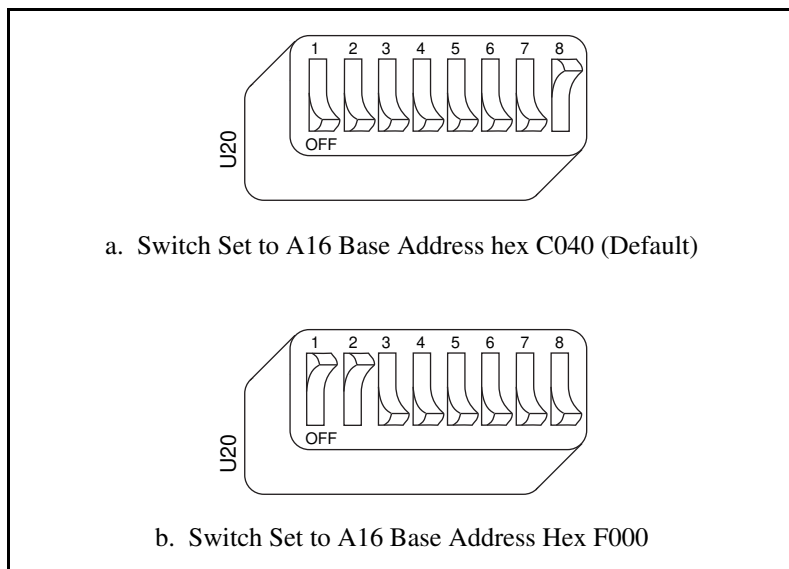
Figure 4-2. Base Address Selection

VME-MXI-2 Intermodule Signaling

If you will be installing more than one VME-MXI-2 in a single VMEbus chassis, you must select a user-defined pin for use by the VME-MXI-2. The VME-MXI-2 modules use this signal to disable the bus timeout unit(s) on the other VME-MXI-2 modules during VMEbus accesses that map to the MXIbus. This is done because the MXIbus bus timeout unit should be the sole timer of any MXIbus access. Since bus timeout units on other VMEbus modules cannot monitor this signal, they should be permanently disabled. If it is not possible to disable a module's bus timeout unit, it should be configured to the highest setting to give MXIbus accesses as much time as possible.

You can choose from three user-defined pins on J2/P2. The pin you select must be bused on the VMEbus backplane between all slots that will have a VME-MXI-2 installed. Use jumper W2 to select pin A5, C5, or C30 of J2/P2, as shown in Figure 4-3.

Notice that a fourth position is also available on the jumper. This is the factory-default setting, which does not connect the VME-MXI-2 to any user-defined pin. You would use this option only if you are installing a single VME-MXI-2 in a chassis.

Figure 4-3 shows the four intermodule signaling settings.

a. User-Defined Pin A5 Selected   b. User-Defined Pin C5 Selected

c. User-Defined Pin C30 Selected   d. No User-Defined Pin Selected (Default)
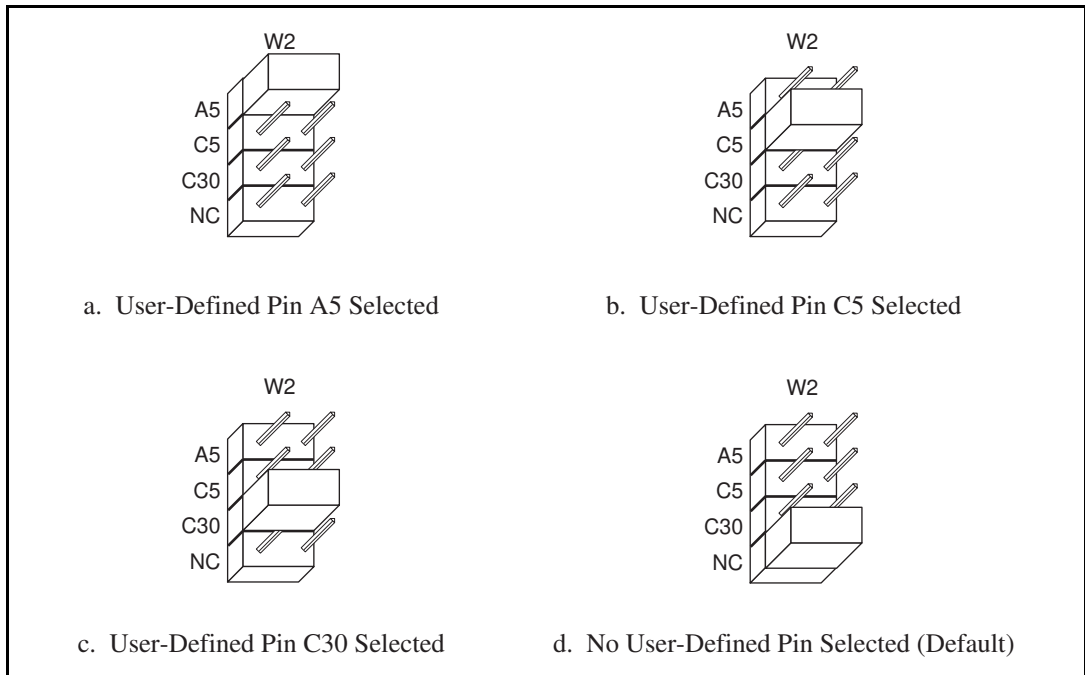
Figure 4-3. VME-MXI-2 Intermodule Signaling Settings

MXIbus Termination

The first and last MXIbus devices connected to the MXIbus—whether it is a single MXI-2 cable or daisy-chained MXI-2 cables—must terminate the MXIbus. Any MXIbus devices in the middle of a MXIbus daisy chain must *not* terminate the MXIbus.

The VME-MXI-2 automatically senses if it is at either end of the MXIbus cable to terminate the MXIbus. You can manually control MXIbus termination by defeating the automatic circuitry. Use switches 3 and 4 of the four-position switch at location U21 to control whether MXIbus termination is automatic (Figure 4-4a), on

(Figure 4-4b), or off (Figure 4-4c). The settings of switches 1 and 2 have no effect on MXIbus termination.

Use switch 3 to select whether you want the VME-MXI-2 to automatically control termination of the MXIbus. Switch 4 lets you manually control whether to terminate the MXIbus when automatic termination is turned off. Switch 4 has no effect when switch 3 is set for automatic MXIbus termination; you must turn off automatic termination if you want to manually control termination.

a. Automatic MXIbus Termination (Default)

b. Terminate MXIbus (On)

c. Do Not Terminate MXIbus (Off)
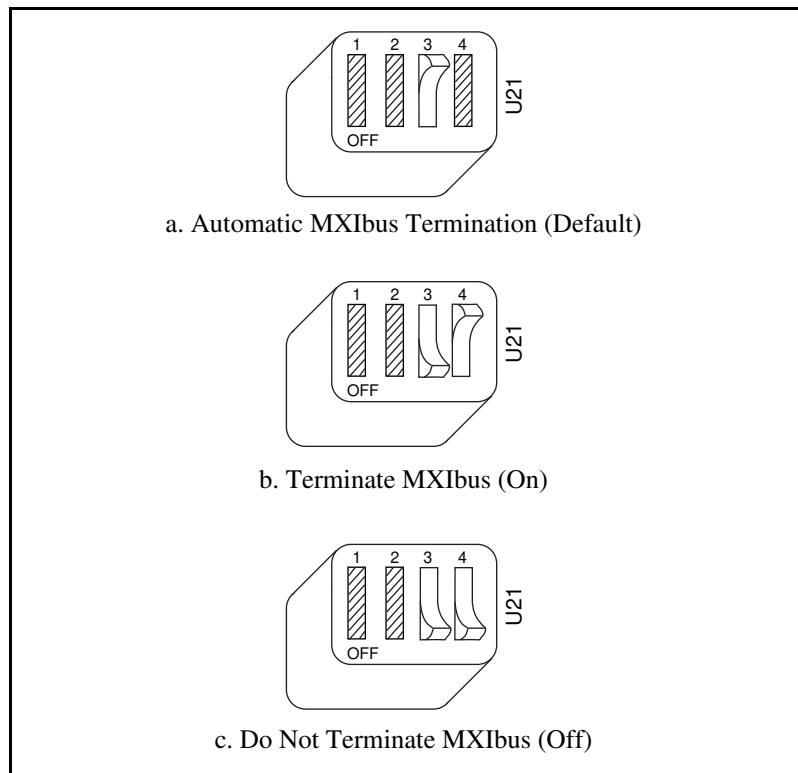
Figure 4-4. MXIbus Termination

Configuration EEPROM

The VME-MXI-2 has an onboard EEPROM, which stores default register values that are loaded at power-on. The EEPROM is divided into two halves—a factory-configuration half, and a user-configuration half. Both halves were factory configured with the same configuration values so you can modify the user-configurable half, while the factory-configured half stores a back-up of the default user settings.

Use switches 1 and 2 of the four-position switch at location U21 to control the operation of the EEPROM. The Restore Factory Configuration switch (switch 1) causes the VME-MXI-2 to boot off the factory-configured half instead of the user-modified settings. This is useful in the event that the user-configured half of the EEPROM becomes corrupted in such a way that the VME-MXI-2 boots to an unusable state.

The Change Factory Configuration switch (switch 2 of U21) lets you change the factory-default configuration settings by permitting writes to the factory settings section of the EEPROM. This switch serves as a safety measure and should not be needed under normal circumstances. When this switch is off (its default setting) the factory configuration of the EEPROM is protected so any writes to the factory area will be ignored. The factory area is protected regardless of the setting of switch 1 of U21.

Figure 4-5 shows the configuration settings for EEPROM operation. The settings of switches 3 and 4 have no effect on EEPROM configuration.

a. Boot from User Configuration (Factory Configuration Protected) (Default)

b. Boot from Factory Configuration (Factory Configuration Protected)

c. Boot from User Configuration (Factory Configuration Unprotected)

d. Boot from Factory Configuration (Factory Configuration Unprotected)
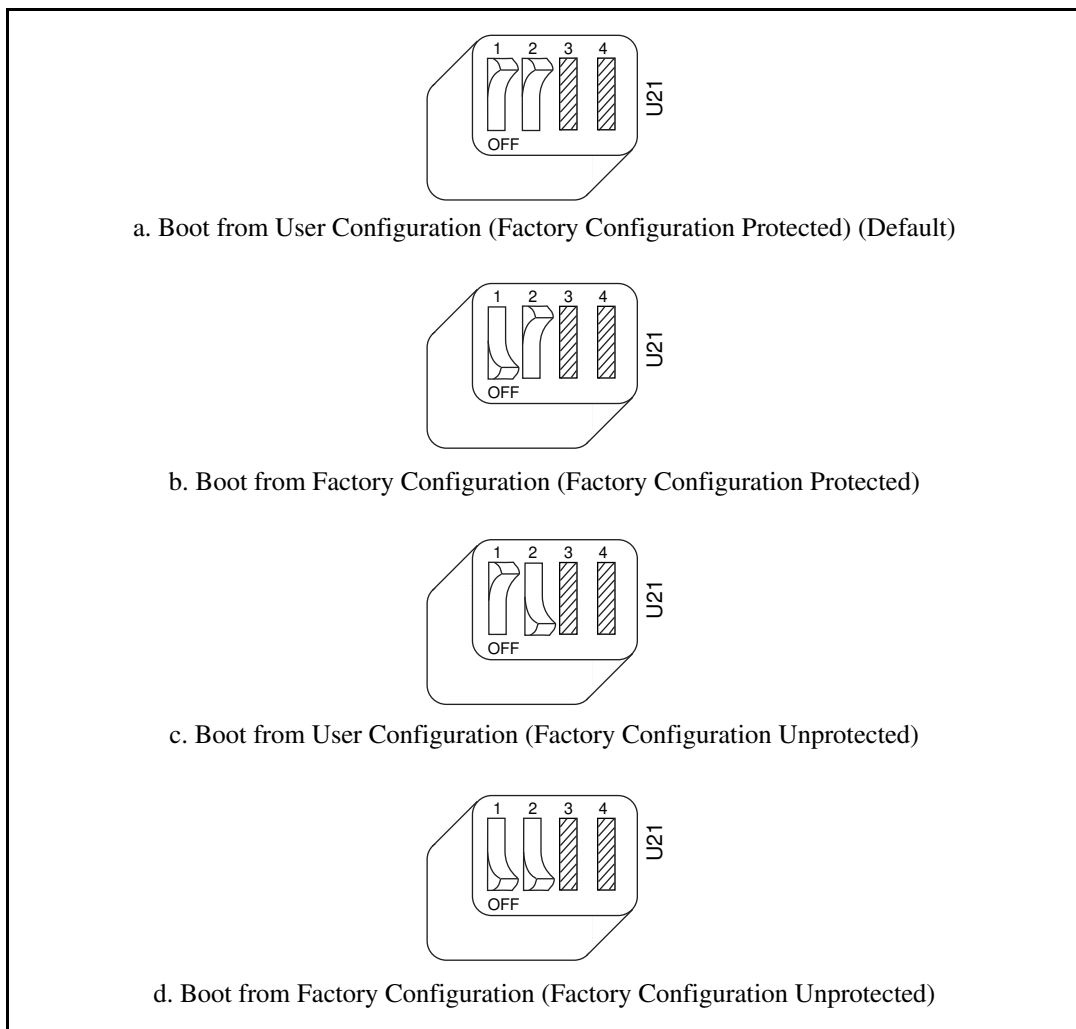
Figure 4-5. EEPROM Operation

Onboard DRAM

The VME-MXI-2 can accommodate up to two 1.35 in. DRAM SIMMs. Table 4-1 lists the SIMMs you can use. You can use 32-bit or 36-bit SIMMs since DRAM parity is not required. Because the VME-MXI-2 supports only one organization at a time, all SIMMs installed must be of the same type. Use Bank 0 first when installing SIMMs. This allows you to install up to 64 MB. The VME-MXI-2 supports DRAM speeds of 80 ns or faster.

Switch S2 is used to select the size of each SIMM. If the SIMMs are 4 M x 32 or larger, S2 should be in the OFF setting as shown in Figure 4-6a. For SIMMs *smaller* than 4 M x 32, use the ON setting as shown in Figure 4-6b.
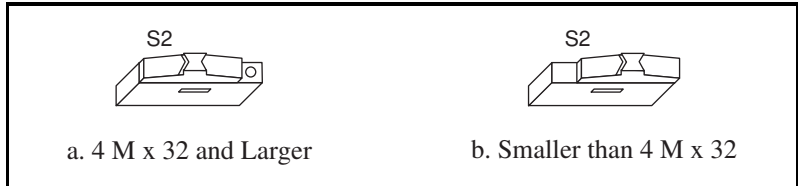
Figure 4-6. SIMM Size Configuration

Refer to Table 4-1 for how to adjust the switch (ON or OFF) for all supported DRAM configurations. Many of the DRAM options are available from National Instruments.

Table 4-1. VME-MXI-2 DRAM Configurations

Bank 0	Bank 1	Total DRAM	National Instruments Option?	Switch Setting of S6
—	—	0	—	—
256 K x 32 or 256 K x 36	—	1 MB	—	ON
256 K x 32 or 256 K x 36	256 K x 32 or 256 K x 36	2 MB	—	ON
512 K x 32 or 512 K x 36	—	2 MB	—	ON
512 K x 32 or 512 K x 36	512 K x 32 or 512 K x 36	4 MB	—	ON
1 M x 32 or 1 M x 36	—	4 MB	YES	ON
1 M x 32 or 1 M x 36	1 M x 32 or 1 M x 36	8 MB	—	ON

Table 4-1. VME-MXI-2 DRAM Configurations (Continued)

Bank 0	Bank 1	Total DRAM	National Instruments Option?	Switch Setting of S6
2 M x 32 or 2 M x 36	—	8 MB	YES	ON
2 M x 32 or 2 M x 36	2 M x 32 or 2 M x 36	16 MB	—	ON
4 M x 32 or 4 M x 36	—	16 MB	YES	OFF
4 M x 32 or 4 M x 36	4 M x 32 or 4 M x 36	32 MB	—	OFF
8 M x 32 or 8 M x 36	—	32 MB	YES	OFF
8 M x 32 or 8 M x 36	8 M x 32 or 8 M x 36	64 MB	YES	OFF

Install the VME-MXI-2

This section contains general installation instructions for the VME-MXI-2. Consult your VMEbus mainframe user manual or technical reference manual for specific instructions and warnings.

1. Plug in your mainframe before installing the VME-MXI-2. The power cord grounds the mainframe and protects it from electrical damage while you are installing the module.

Warning: *To protect both yourself and the mainframe from electrical hazards, the mainframe should remain off until you are finished installing the VME-MXI-2 module.*

2. Remove or open any doors or covers blocking access to the mainframe slots.

3. Insert the VME-MXI-2 in the slot you have selected by aligning the top and bottom of the board with the card-edge guides inside the mainframe. Slowly push the VME-MXI-2 straight into the slot until its plug connectors are resting on the backplane receptacle connectors. Using slow, evenly distributed pressure, press the VME-MXI-2 straight in until it seats in the expansion slot. The front panel of the VME-MXI-2 should be even with the front panel of the mainframe.

4. Tighten the retaining screws on the top and bottom edges of the front panel.

5. Check the installation.

6. Connect the cables as described in the following section before restoring power.

7. Replace or close any doors or covers to the mainframe.

Connect the MXIbus Cable

There are two basic types of MXI-2 cables. MXI-2 cables can have either a single connector on each end or a single connector on one end and a double connector on the other end.

Connect the labeled end of the cable to the MXI-2 device that will be the MXIbus System Controller. Connect the other end of the cable to the other device. Be sure to tighten the screw locks to ensure proper pin connection.

Figure 4-7 shows the correct cabling for a VME system containing a PCI-MXI-2 board in a PCI-based computer cabled to a VME-MXI-2 module residing in Slot 1 of a VMEbus mainframe. Notice that you can expand your system to include other devices by using an additional MXI-2 cable. However, in such a case the first cable needs to have a double connector on one end. You can then use a cable with a single connector on each end to connect the last device on the MXIbus.
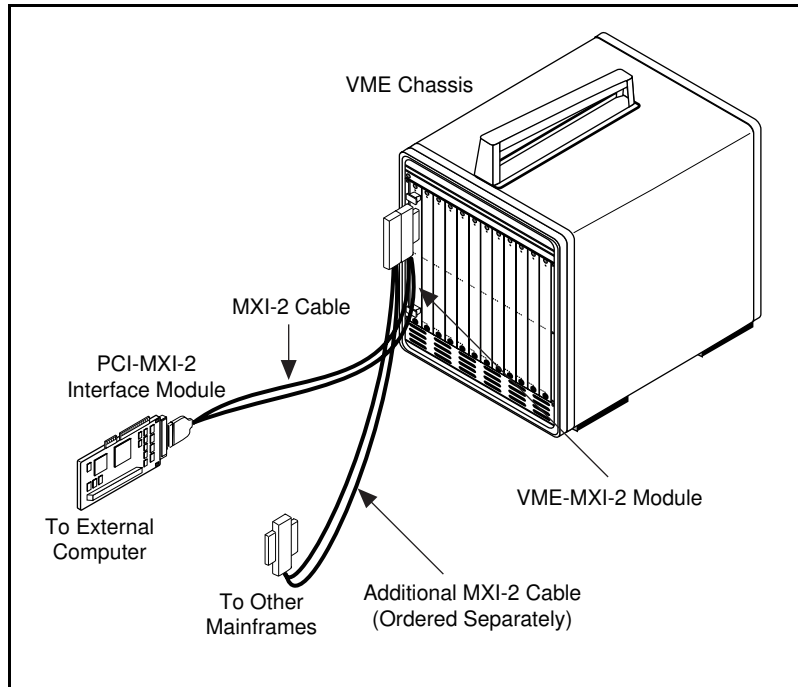
Figure 4-7. MXI-2 Cable Configuration Using a PCI-MXI-2 and a VME-MXI-2

When you have properly connected the MXI-2 cable, power on the VMEbus mainframe and then the computer.

☞ **Note:** *Always turn on the mainframe first. Doing so makes it possible for your external computer to access the VME boards in the mainframe upon startup.*

Chapter

5

NI-VXI Software Installation

This chapter contains the instructions to install the NI-VXI software.

There are two methods for installing your NI-VXI software: the Windows Setup program, and a DOS INSTALL program. You can use the Windows Setup program to install NI-VXI for Windows 95/NT/3.1 and DOS. The DOS INSTALL program can install only the NI-VXI software for DOS. Refer to the section appropriate for the Microsoft operating system you are using.

Some of the new utilities rely on the LabWindows/CVI Run-Time Engine. This software is installed, if necessary, during the NI-VXI installation.

Be sure you have up to 5 MB of free space available to accommodate the NI-VXI software.

Using the Windows Setup Program (Windows 3.1)

You can use the Setup program that came with your NI-VXI software to install the entire NI-VXI software package, a software update, or to reinstall software in the event that your files were accidentally erased. Follow these steps to install all or part of the NI-VXI software.

Installing the NI-VXI Software

This section describes how to install the NI-VXI software for the PCI-MXI-2 and Windows 3.1. Please carefully read these directions along with any messages on the screen before making your selections.

You can quit the Setup program at any time by pressing the **Cancel** button. If you do not have a mouse, pressing the <*Esc*> key performs the same action.

Setup is an interactive, self-guiding program that installs the NI-VXI software and configures your system to use the NI-VXI software with the PCI-MXI-2. Follow these steps to perform the installation.

1. Insert the disk labeled *NI-VXI Software for PCI-MXI-2*.

2. Select **Run...** from the Windows Program Manager's **File** menu and enter the following text, where *X* is your floppy drive (usually A or B).

    `X:\setup.exe`

    and press <Enter>.

3. Click on the **Next** button at the **Welcome** screen to start the installation.

4. Select the target directory for your installation. Although Setup prompts you to accept a default directory, you can select a different location by clicking on **Browse...**.

    -   If you do not already have NI-VXI on your computer, Setup prompts you to select the default directory of `C:\NIVXI`.

    -   Setup detects if you already have a previous installation of NI-VXI on your computer, and prompts you to either overwrite the previous version or install the new version in a different directory. Only one copy of NI-VXI can be active on your computer at any time. If you install the new version of NI-VXI into a different directory, the old installation is disabled.

Caution: *If you have a previous version of the NI-VXI software installed, Setup does not automatically create a backup of the software files. If there are files you want to preserve, you should exit the installation program now and make a backup before continuing.*

5. Select the type of installation you want.

    -   **Typical** installation includes Windows and DOS drivers, utilities, and development files for all supported compilers and environments.

    -   **Compact** installation includes only the driver and utilities necessary to run applications written with NI-VXI.

    -   **Custom** installation lets you select the target operating system and development environments according to your needs.

Note: *If you install the driver files, you must include the Common Windows/DOS Driver Files. Similarly, when installing development files, include Common Development Files in your custom installation.*

6. Select the name of the Program Manager folder that will contain icons for the NI-VXI software for PCI-MXI-2.

At this point Setup copies the necessary files to your hard drive and creates Program Manager icons.

7. Your `AUTOEXEC.BAT` file needs to be modified to use NI-VXI. You can either let Setup modify the file, create a different file, or do nothing. If you decide to not let Setup modify your `AUTOEXEC.BAT` file, you should make the necessary changes manually. Refer to the following section, *Modifying the AUTOEXEC.BAT File*, for complete details.

8. When the installation process completes, you must exit Windows and reboot your computer for the changes to take effect. You can let the Setup program reboot your computer by selecting the **Restart computer** option on the last screen. Click on **Finish** to end the installation.

9. You can now use `VXIedit` to configure the hardware in your VXI system. Continue with Chapter 6, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in `VXIedit`.

Modifying the AUTOEXEC.BAT File

If you choose to let Setup modify your `AUTOEXEC.BAT` file, it updates the setting of environment variables `PATH`, `LIB`, and `INCLUDE` to include the relevant subdirectories of the NI-VXI directory. When Setup modifies the file, it saves the old file as `AUTOEXEC.OLD` in the NI-VXI directory. The previously specified directories in `PATH`, `LIB`, and `INCLUDE` remain unchanged. Setup also adds a new environment variable `NIVXIPATH`, and appends a command line to execute `VXIINIT.EXE` automatically.

If you choose not to let Setup modify your `AUTOEXEC.BAT` file, it creates a file called `AUTOEXEC.VXI` in the NI-VXI directory. Refer to the `AUTOEXEC.VXI` file for suggestions on how to change the following lines manually.

*   The `PATH` variable should include the full path to the subdirectory where the NI-VXI utilities and `NIVXI.DLL` are located, in addition to whatever other directories you have already specified in `PATH`. The path must be specified so that Windows can locate the executable code when the library needs to be loaded. Normally, these files reside in the root of the NI-VXI directory, and also the `WIN` subdirectory.

*   The `LIB` variable should include the full path to the subdirectories that contain the C libraries for the compiler you choose to install.

- The INCLUDE variable should include the full path to the subdirectory that contains the NI-VXI include files. By default, the include files reside in the INCLUDE subdirectory of the NI-VXI directory.

- The NIVXIPATH variable should contain the full path to the NI-VXI directory.

Completing the Software Installation

After you execute Setup, you should exit Windows and reboot your machine to make your system aware of the NI-VXI directory.

After the NI-VXI software is installed, run VXIINIT.EXE and then RESMAN.EXE. You need to run VXIinit to initialize the PCI-MXI-2 before you perform any VXI operations and after each computer reset. RESMAN is the National Instruments Resource Manager, which you must run every time the chassis power is cycled so that your application can access devices in the VXI/VME chassis.

After you run VXIinit and RESMAN, you are ready to use the NI-VXI Resource Editor program VXIedit to interactively configure the hardware in your system. Continue with Chapter 6, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in VXIedit.

Using the Windows Setup Program (Windows 95)

You can use the Setup program that came with your NI-VXI software to install the entire NI-VXI software package, a software update, or to reinstall software in the event that your files were accidentally erased. Follow these steps to install all or part of the NI-VXI software.

System Preparation

If you are currently using either the NI-VXI software for DOS/Windows 3.1 or the NI-VXI Windows 95 Upgrade, Setup will remove it before installing the new software. You cannot have both the 16-bit and the 32-bit versions of NI-VXI installed at the same time.

☞ **Note:** *If you plan to run both 16-bit and 32-bit applications, you should use the NI-VXI Windows 95 Upgrade version instead.*

If you have been using your PCI-MXI-2 under Windows 95 with either the NI-VXI software for DOS/Windows 3.1 or the NI-VXI Windows 95

Upgrade, you need to remove the Plug and Play information from the Windows 95 Device Manager before installing the new NI-VXI software.

Follow these steps to remove the PCI-MXI-2 information.

1. Double-click on the **System** icon under **Start**»**Settings**»**Control Panel**.

2. Select the **Device Manager** tab from the **System Properties** dialog that appears.

3. Click on the **View devices by type** button and double-click on the **Other Devices** icon.

4. Select the PCI-MXI-2 from the list of devices under **Other Devices**. It will appear under the name **PCI Card** and will have a circled exclamation point through the **?** (question mark) icon.

5. Click on the **Remove** button.

6. Click **OK** to exit the Device Manager after removing the device information.

Installing the NI-VXI Software

This section describes how to install the NI-VXI software for the PCI-MXI-2 and Windows 95. Please carefully read these directions along with any messages on the screen before making your selections.

You can quit the Setup program at any time by pressing the **Cancel** button.

Setup is an interactive, self-guiding program that installs the NI-VXI software and configures your system to use the NI-VXI software with the PCI-MXI-2. Follow these steps to perform the installation.

1. Insert the disk labeled *NI-VXI Software for PCI-MXI-2*.

2. Select **Run...** from the **Start** menu and enter the following text, where *X* is your floppy drive (usually A or B).

   X:\setup.exe

   and press <Enter>.

3. Click on the **Next** button at the **Welcome** screen to start the installation.

☞ **Note:**   *If Setup detects a DOS/Windows 3.1 or Windows 95 Upgrade version, it will warn you that this version will be deleted.*

4. Select the target directory for your installation. Although Setup prompts you to accept the `C:\NIVXI` directory by default, you can select a different location by clicking on **Browse...**.

Caution: *If you have a previous version of the NI-VXI software installed, Setup can convert the configuration settings to the new format. However, manufacturer and model name files will not be preserved. If you want to preserve these files, you should exit the installation program now and make a backup before continuing.*

5. Select the components you want to install.

    •   **NI-VXI for Windows 95** includes only the driver and utilities needed to configure and use your VXI or VME system.

    •   **Microsoft C/C++ Development Files** includes the components necessary to develop applications using the Microsoft Visual C++ compiler. Please read the *Modifying the Environment* section if you choose to install these files.

    •   **Borland C/C++ Development Files** includes the components necessary to develop applications using the Borland C/C++ compiler. Please read the *Modifying the Environment* section if you choose to install these files.

6. Select the name of the program folder that will contain icons for the NI-VXI software for PCI-MXI-2.

    At this point Setup copies the necessary files to your hard drive and creates program icons.

7. When the installation process completes, you must exit Windows and reboot your computer for the changes to take effect. You can let the Setup program reboot your computer by selecting the **Restart computer** option on the last screen. Click on **Finish** to end the installation.

8. If you backed up the manufacturer and model name files, you should restore them to the `TBL` subdirectory of your NI-VXI directory before running `VXIedit`.

9. You can now use `VXIedit` to configure the hardware in your VXI system. Continue with Chapter 6, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in `VXIedit`.

Modifying the Environment

The installer does not modify the environment variables under Windows 95. No changes are necessary. However, you may want to add the NI-VXI directory to your LIB and INCLUDE paths if you use makefiles to build your projects, and to the PATH variable if you plan to run the VXI utilities from the command line.

Completing the Software Installation

After you execute Setup, you should restart Windows 95 to make your system load the NI-VXI driver.

After the NI-VXI software is installed, run RESMAN.EXE, which is the National Instruments Resource Manager. You must run RESMAN every time the chassis power is cycled so that your application can access devices in the VXI/VME chassis. Notice that because Windows 95 supports the plug and play architecture, you do *not* need to run VXIINIT.EXE before you do any VXI operation.

After you run RESMAN, you are ready to use the NI-VXI Resource Editor program VXIedit to interactively configure the hardware in your system. Continue with Chapter 6, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in VXIedit.

Using the Windows Setup Program (Windows NT)

You can use the Setup program that came with your NI-VXI software to install the entire NI-VXI software package, a software update, or to reinstall software in the event that your files were accidentally erased. Follow these steps to install all or part of the NI-VXI software.

Installing the NI-VXI Software

This section describes how to install the NI-VXI software for the PCI-MXI-2 and Windows NT. Please carefully read these directions along with any messages on the screen before making your selections.

You can quit the Setup program at any time by pressing the **Cancel** button.

Setup is an interactive, self-guiding program that installs the NI-VXI software and configures your system to use the NI-VXI software with the PCI-MXI-2. Follow these steps to perform the installation.

1. Insert the disk labeled *NI-VXI Software for PCI-MXI-2*.

2. Select **Run...** from the **Start** menu on the taskbar or from the Program Manager **File** menu and enter the following text, where *X* is your floppy drive (usually A or B).

   `X:\setup.exe`

   and press <Enter>.

3. Click on the **Next** button at the **Welcome** screen to start the installation.

4. Select the target directory for your installation. Although Setup prompts you to accept the `C:\NIVXI` directory by default, you can select a different location by clicking on **Browse...**.

Caution: *If you have a previous version of the NI-VXI software installed, Setup can convert the configuration settings to the new format. However, manufacturer and model name files will not be preserved. If you want to preserve these files, you should exit the installation program now and make a backup before continuing.*

5. Select the components you want to install.

   - **NI-VXI for Windows NT** includes only the driver and utilities needed to configure and use your VXI or VME system.

   - **Microsoft C/C++ Development Files** includes the components necessary to develop applications using the Microsoft Visual C++ compiler. Please read the *Modifying the Environment* section if you choose to install these files.

   - **Borland C/C++ Development Files** includes the components necessary to develop applications using the Borland C/C++ compiler. Please read the *Modifying the Environment* section if you choose to install these files.

6. Select the name of the program folder that will contain icons for the NI-VXI software for PCI-MXI-2.

   At this point Setup copies the necessary files to your hard drive and creates program icons.

7. When the installation process completes, you must exit Windows and reboot your computer for the changes to take effect. You can let the Setup program reboot your computer by selecting the **Restart computer** option on the last screen. Click on **Finish** to end the installation.

8. If you backed up the manufacturer and model name files, you should restore them to the `TBL` subdirectory of your NI-VXI directory before running `VXIedit`.

9. You can now use `VXIedit` to configure the hardware in your VXI system. Continue with Chapter 6, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in `VXIedit`.

Modifying the Environment

The installer does not modify the environment variables under Windows NT. No changes are necessary. However, you may want to add the NI-VXI directory to your `LIB` and `INCLUDE` paths if you use makefiles to build your projects, and to the `PATH` variable if you plan to run the VXI utilities from the command line.

Completing the Software Installation

After you execute Setup, you should restart Windows NT to make your system load the NI-VXI driver.

After the NI-VXI software is installed, run `RESMAN.EXE`, which is the National Instruments Resource Manager. You must run `RESMAN` every time the chassis power is cycled so that your application can access devices in the VXI/VME chassis. You do *not* need to run `VXIINIT.EXE` unless `RESMAN` fails.

After you run `RESMAN`, you are ready to use the NI-VXI Resource Editor program `VXIedit` to interactively configure the hardware in your system. Continue with Chapter 6, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in `VXIedit`.

Using the DOS INSTALL Program

If you do not have any Windows operating system installed on your machine, you can install the NI-VXI software for DOS with the INSTALL program. This program functions similarly to the Windows Setup program.

Running INSTALL

Run `INSTALL.EXE`. This program copies the necessary files to your hard disk and performs the necessary steps to install the NI-VXI software for DOS to your system.

Caution: ***In its default configuration, the PCI-MXI-2 requests memory space for use by the NI-VXI driver above the 1 MB DOS boundary. To run your NI-VXI software for the PCI-MXI-2 in DOS you must reconfigure the PCI-MXI-2 to request memory below the 1 MB DOS boundary.***

To change the requested memory space you need to use the VXIedit program described in Chapter 6, *NI-VXI Configuration Utility*. Use the **PCI-MXI-2 Configuration Editor** and access the **Bus Configuration Editor** menu. Change the user and driver windows to be below the 1 MB boundary by enabling the **Place below 1 MB** checkboxes. Please refer to the *User Window and Driver Window* section of Chapter 6 for more details.

NI-VXI Configuration Utility

This chapter contains instructions for using the VXI Resource Editor utility of the NI-VXI software to configure the PCI-MXI-2 and the VXI-MXI-2 or VME-MXI-2.

VXIEDIT.EXE is the VXI resource editor program that you use to configure the system and to edit the manufacturer name and ID numbers, the model names of VXI and non-VXI devices in the system, and the system interrupt configuration information. This program also displays the system configuration information generated by the Resource Manager.

The displays shown in this section are from the Windows 95 version of VXIedit. The Windows NT, Windows 3.1, and DOS versions of VXIedit have the same organization and functionality as the Windows 95 version although the displays may look slightly different. The descriptions and instructions in this chapter apply to all versions of VXIedit.

☞ **Note:** *A text-based version,* VXItedit*, is also available as an alternative. You can run* VXItedit *in either Windows 95/NT/3.1 or DOS. Although this chapter focuses only on the graphical* VXIedit *program, the two programs are functionally equivalent. For information on* VXItedit*, refer to the NI-VXI Text Utilities Reference Manual.*

Running the VXIedit Configuration Utility

To run VXIedit in Windows 95/NT/3.1, double click the **VXIedit** (Windows) icon in the NI-VXI group. To run VXIedit in DOS, type VXIedit at the DOS command prompt. In DOS, you can run VXIedit from any directory, but make sure that both the PATH and NIVXIPATH environment variables have the destination directory of the NI-VXI software added to them. Under Windows 3.1 and DOS, NIVXIPATH is used by the application to find the different configuration files (*.CFG),

table files (`*.TBL`), and help files (`*.HLP`) during its execution. The default pathname used by the program if `NIVXIPATH` is not set is `C:\NIVXI`. Under Windows 95/NT, NI-VXI uses the system registry for all the configuration information.

Most of the features on the PCI-MXI-2, VXI-MXI-2, and VME-MXI-2 are configurable through software, using `VXIedit`, rather than through hardware switches or jumpers on the boards themselves. In addition, the `VXIedit` utility can override some of the hardware settings.

Figure 6-1 shows the main menu of the `VXIedit` resource editor.

Figure 6-1. VXIedit Main Screen

The rest of this chapter describes only the features of the **PCI-MXI-2 Configuration Editor** and the **VXI/VME-MXI-2 Configuration Editor**. For instructions on using the other editors, refer to your software utility reference manual, either the *NI-VXI Graphical Utilities Reference Manual* or the *NI-VXI Text Utilities Reference Manual*.

PCI-MXI-2 Configuration Editor

Figure 6-2 shows the opening screen of the **PCI-MXI-2 Configuration Editor**. Notice that the screen displays the serial number and hardware revision of the PCI-MXI-2 board in addition to several configuration options.
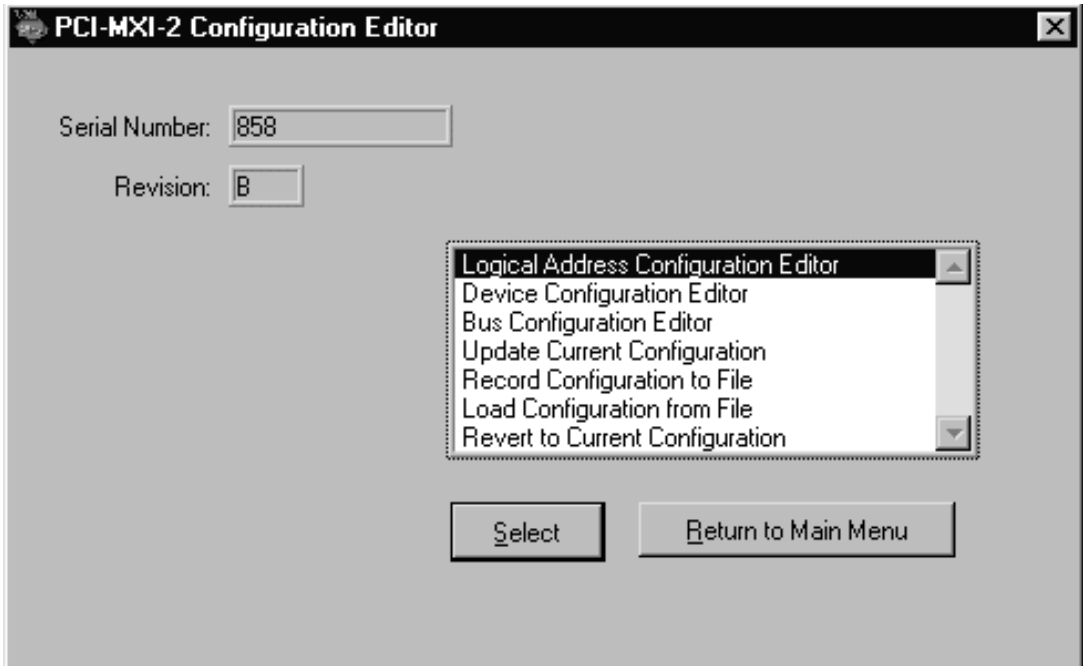
Figure 6-2. PCI-MXI-2 Configuration Editor

The first three options under the **PCI-MXI-2 Configuration Editor** are:

- **Logical Address Configuration Editor**
- **Device Configuration Editor**
- **Bus Configuration Editor**

When making changes to the PCI-MXI-2 through these editors, remember that the changes do not take effect until you commit them by selecting the **Update Current Configuration** option.

♦ **DOS users**—If the PCI-MXI-2 driver window is located above 1 MB, the program will prompt you for a memory location below 1 MB. The memory location is a 32 KB window that allows VXIedit to access the PCI-MXI-2 registers. This is due to a DOS limitation that restricts DOS programs from accessing memory above 1 MB.

Before proceeding to a description of each field in these editors, review the remaining four options of the **PCI-MXI-2 Configuration Editor**. These options directly relate to how you can use the changes you make using the configuration editors, which are described after the options.

Update Current Configuration

Use this option to write the configuration settings to the PCI-MXI-2 EEPROM and files used by NI-VXI. It configures the PCI-MXI-2 to be consistent with the configuration EEPROM. Notice that some of the configuration settings cannot take effect until you reset the machine, either by using the reset button or by turning the power off and on again.

☞ **Note:** *You will not be able to use the <Control-Alt-Del> keystroke combination or the Windows Restart command to perform this reset. Instead you must perform a hardware reset as described above.*

Record Configuration to File

With this option you can save your configuration settings to a file. Notice that this option does *not* write the configuration settings to the PCI-MXI-2 configuration EEPROM.

If you want to update the PCI-MXI-2 configuration settings, use the **Update Current Configuration** option instead.

Load Configuration from File

You can use this option to load your configuration settings from a file. This action only updates the configuration settings in your editor. This does *not* write the configuration settings to the PCI-MXI-2 configuration EEPROM. To update the configuration use the **Update Current Configuration** option for the changes to take effect.

Revert to Current Configuration

If you made changes to the configuration settings without committing those changes (writing to configuration EEPROM), you can revert the configuration settings to the values they had before you made the changes.

☞   **Note:**   *You can successfully revert only if you have NOT yet selected the* **Update Current Configuration** *option.*

Logical Address Configuration Editor

Figure 6-3 shows the **Logical Address Configuration Editor**. Notice that the options are arranged into three groups—**Device Settings**, **VXI Shared Memory**, and **Resource Manager**. The following paragraphs describe the options you can select for each of the fields.
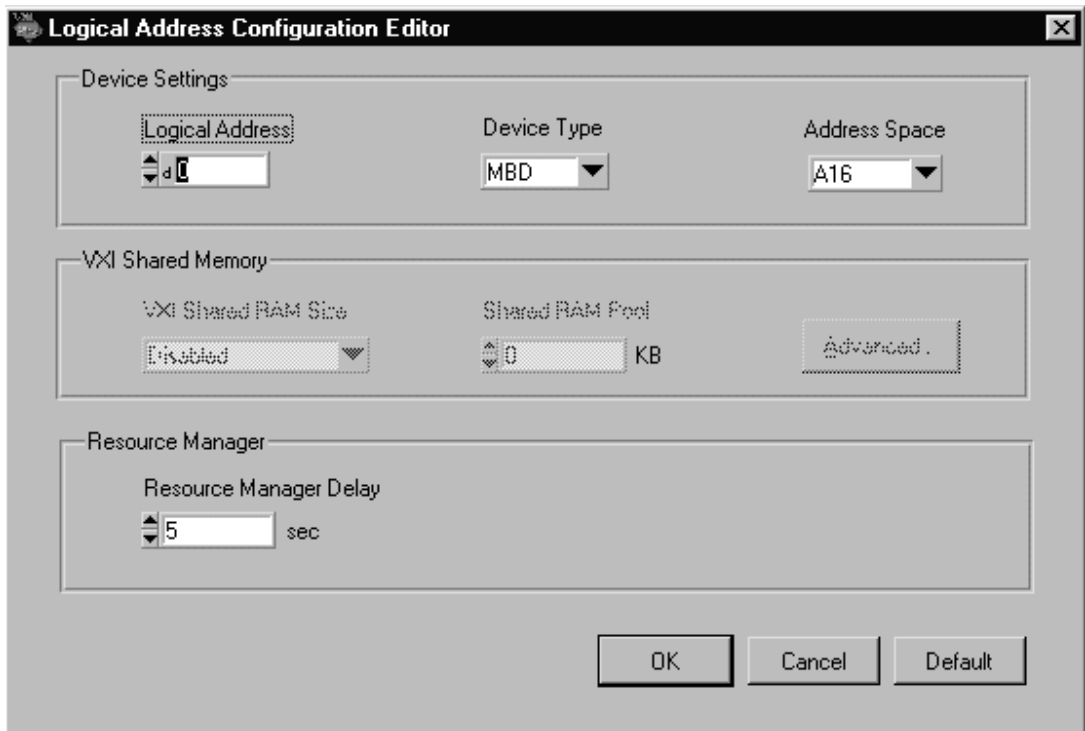
Figure 6-3. PCI-MXI-2 Logical Address Configuration Editor

Device Settings

The **Device Settings** group contains the controls to set the logical address, device type, and address space of the PCI-MXI-2.

Logical Address

This parameter sets the MXI logical address of the PCI-MXI-2. The following table shows the allowable range of values and the default value.

Logical Address Range	Default Value
0 to 254	0

Device Type

This field indicates the classification of the PCI-MXI-2. The default value is **MBD**, designating a message-based device. The following table shows the available options.

Classification	Setting
Extended Device	**EXT**
Message-Based Device	**MBD**
Register-Based Device	**RBD**

The device type affects only the contents of the **Device Class** field in the Device Type register. The functionality of the other registers does not change.

Address Space

This field indicates the addressing mode(s) of the device's operational registers. The PCI-MXI-2 can be configured in one of three ways. The default addressing mode is for **A16** space only. Your other options are **A16/A24** and **A16/A32**.

Notice that several other controls on the configuration editor panel are disabled when the addressing mode is A16, as shown in Figure 6-3.

Only if you select **A16/A24** or **A16/A32** are the following controls relevant:

- **VXI Shared RAM Size**
- **Shared RAM Pool**
- **Advanced**
  - **Byte Swapping**
  - **Memory Select**
  - **Mapping**

VXI Shared Memory

The **VXI Shared Memory** group contains the controls to set the VXI shared RAM size and the shared RAM pool. The **Advanced** button leads to additional options that configure the upper and lower half of the shared RAM area.

☞ **Note:**   *When the* **Address Space** *field is in the default setting of A16 only, all of the options in this group are disabled because they are irrelevant.*

VXI Shared RAM Size

This field indicates the amount of RAM (in bytes) that is shared in either A24 or A32 space. This determines the *total* shared RAM size, which is then divided into two equal halves that you can set up independently of one another.

☞ **Note:**   *When the* **Address Space** *field is in the default setting of A16 only, this field is ignored.*

Shared RAM Pool

This field indicates the size of memory (in kilobytes) that is allocated on Windows startup. This memory is used by the lower/upper half window when the **Memory Select** control (accessible through the **Advanced** popup field) is set to **System memory**.

Memory Range	Default Value
0 to 65535 KB	0 KB

The following table indicates how the **Shared RAM Pool** relates to the **VXI Shared RAM Size** depending on the setting of the **Memory Select** control for the upper and lower half windows.

Lower Half Window	Upper Half Window	Shared RAM Pool (Window)
System memory	System memory	Equal to VXI Shared RAM Size
System memory	Onboard memory	Half the VXI Shared RAM Size
Onboard memory	System memory	Half the VXI Shared RAM Size
Onboard memory	Onboard memory	0

The shared RAM pool is used by `VXImemAlloc()` function calls from both Windows applications and DOS applications running under Windows. For information on the `VXImemAlloc()` function, refer to the *NI-VXI User Manual* and the *NI-VXI Programmer Reference Manual*.

☞ **Note:**   *When the* **Address Space** *field is in the default setting of A16 only, this field is ignored. This field is also ignored if the* **Memory Select** *fields for both the lower and upper half windows are set to* **Onboard memory.**

◢ **Caution:**   *This memory pool is allocated at Windows 95/NT/3.1 startup, and will not be available for Windows. Take into account the memory requirements of Windows and your applications and the amount of memory in your system before setting this option.*

Advanced Shared RAM Settings

Clicking on the **Advanced** button displays a dialog box to configure the destination of MXIbus cycles that map into the PCI-MXI-2 through the A24/A32 shared RAM.
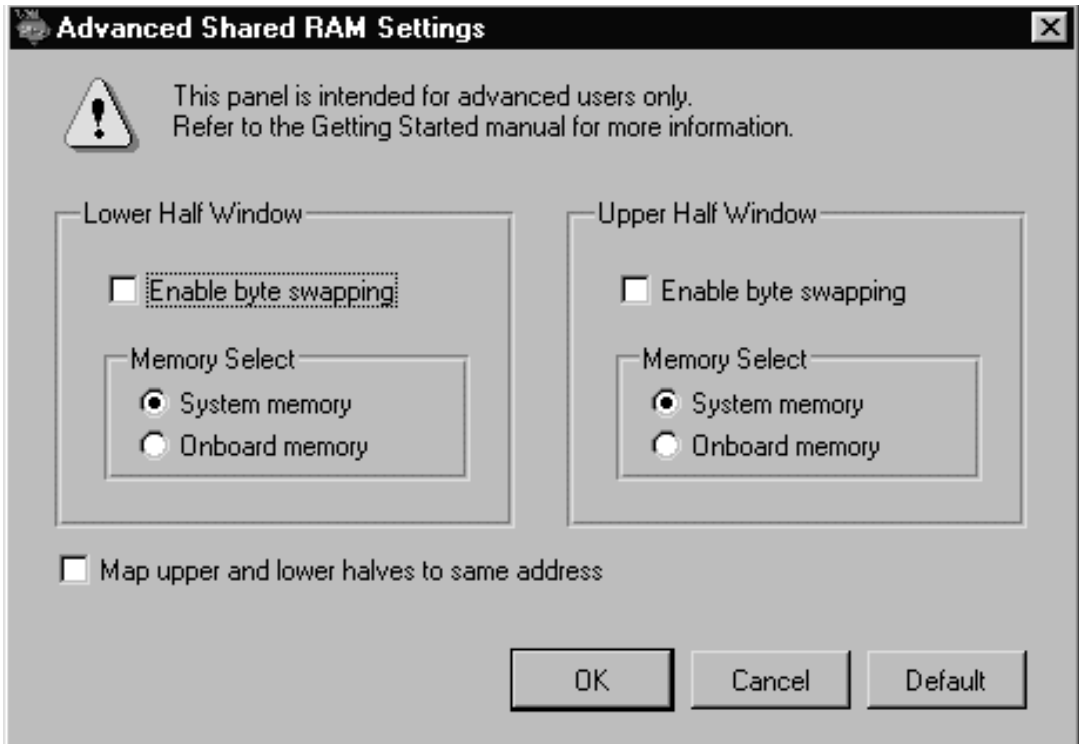
Figure 6-4. Advanced Shared RAM Settings

☞ **Note:**   *When the* **Address Space** *field is in the default setting of A16 only, these fields are ignored, and cannot be accessed.*

The VXI shared RAM is divided into two halves, or *windows*. You can select the byte order for each half independently. You can map each half of the VXI shared RAM independently into system memory on the motherboard or into onboard memory on the PCI-MXI-2.

Because each half is independent of the other, you can choose from any of these mapping options:

- Half the VXI shared RAM mapped to system memory; the other half mapped to PCI-MXI-2 onboard memory
- Both halves mapped to PCI-MXI-2 onboard memory
- Both halves mapped to system memory

Enable Byte Swapping

This checkbox indicates whether byte swapping should be performed for slave accesses to this half of the VXI shared RAM space. For example, if the native byte order of the shared RAM is Intel (Little Endian), and you want to present data to the VXI/VMEbus in Motorola (Big Endian) byte order, you will need to enable byte swapping. The default value is non-swapped. Click on the checkbox if you want to enable byte swapping.

This field is ignored if the **Memory Select** field is set to **Onboard memory**.

Memory Select

This option determines where this half of the VXI shared RAM is mapped. By default, the shared RAM is mapped to **System memory**. If you want to use the RAM on the PCI-MXI-2, choose the **Onboard memory** option.

Window Mapping

You can click on the checkbox at the bottom of the screen if you want to map both halves of the inward window to the same address. When both halves of the inward window are mapped to the same destination with the same byte order, the windows essentially form one continuous window. If the windows are mapped to different destinations, the base of each inward window maps to the base of each destination.

If the windows both map to the shared RAM destination but the byte order is different, the base of each inward window maps to the base of the shared RAM destination. This results in one half of the window accessing the system RAM in Little Endian byte order and the other half accessing it in Big Endian byte order.

Caution: *There is a potential problem when opening up a shared memory region to point to system RAM. The PCI bus may return a retry on any cycle into system RAM. As a consequence, an external VXI device accessing the system RAM may get a VXI retry back. If the external VXI device does not support VXI retry, the VXI device will falsely detect the retry condition as a bus error condition.*

VXI devices that support retries will not have this problem, because they can handle VXI retry conditions correctly by automatically retrying the access. For example, the National Instruments VXI-DAQ boards handle VXI retry conditions properly, and do not exhibit this problem.

Resource Manager

The only option under the **Resource Manager** portion of the **Logical Address Configuration Editor** is the **Resource Manager Delay** control.

Resource Manager Delay

☞ **Note:** *This field is effective only when the PCI-MXI-2 is at its default logical address of 0. The PCI-MXI-2 is the Resource Manager only if its logical address is 0.*

This field specifies the time in seconds that the Resource Manager (RM) waits before accessing any other VXI/VMEbus device's A16 configuration registers.

RM Delay Range	Default Value
0 to 65535 s	5

Device Configuration Editor

Figure 6-5 shows the **Device Configuration Editor**. The following paragraphs describe the options you can select for each of the fields.
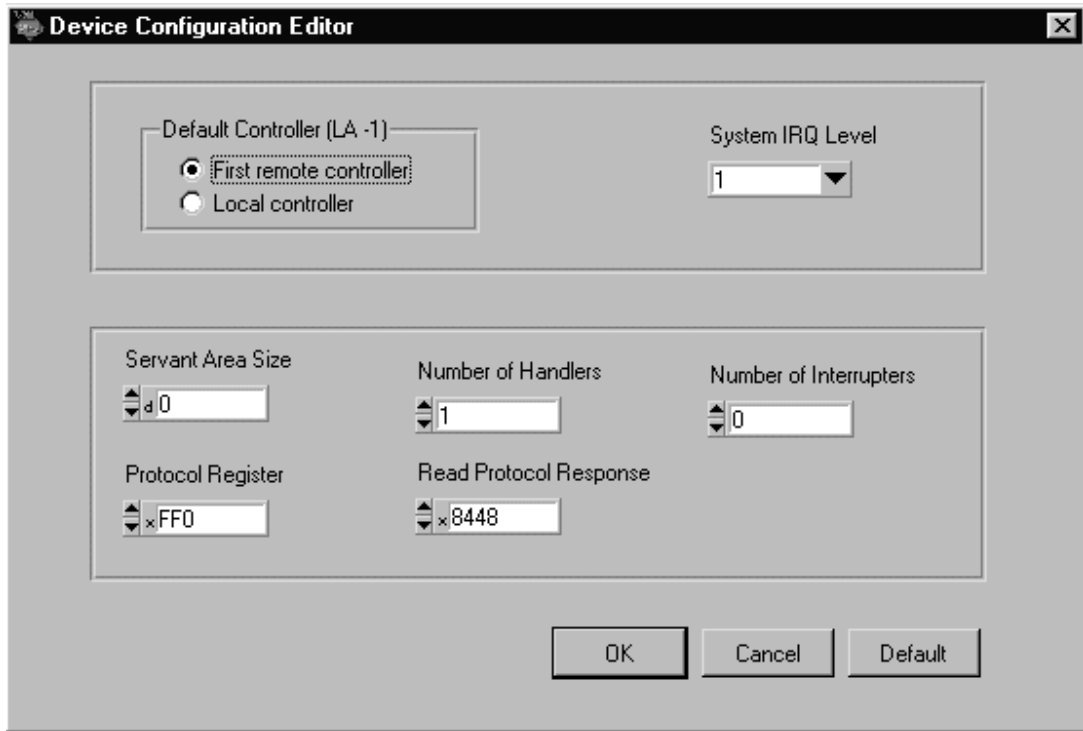
Figure 6-5. PCI-MXI-2 Device Configuration Editor

Default Controller (LA -1)

Many NI-VXI functions use **controller** as a parameter with -1 accepted as a valid value. You use this selection to determine which controller you are referring to when you use -1 in these NI-VXI functions. Review the descriptions of the NI-VXI functions to determine which are applicable for this control.

By default, the **Default Controller (LA -1)** option is set to **First remote controller**, so that any NI-VXI functions that are passed the value of -1 for the **controller** parameter will be executed on the first VXI/VME-MXI-2 in the MXI-2 chain. If you select the **Local controller** option, the NI-VXI functions execute on the PCI-MXI-2.

System IRQ Level

The remote controllers—in this case the VXI/VME-MXI-2—can report events such as triggers and DMA to the PCI-MXI-2 through a VXI IRQ line. This field selects which VXI IRQ level the remote controllers should use to report events to the PCI-MXI-2.

Interrupt Request Levels	Default Value
1 to 7	1

Servant Area Size

This field designates the servant area size, which is supplied to the Resource Manager in response to the *Read Servant Area* command (if the PCI-MXI-2 is *not* the Resource Manager in your system). The servant area size is an 8-bit value (0 through 255) that indicates the servant area. The servant area begins at the logical address following the PCI-MXI-2 logical address, and includes *N* contiguous logical addresses, where *N* is the value of the servant area size. This field is meaningful only when the PCI-MXI-2 is configured as a message-based device.

Servant Area Range	Default Value
0 to 255	0

☞ **Note:**   *If the PCI-MXI-2 is the Resource Manager (Logical Address 0), this setting is irrelevant.*

Number of Handlers

This field gives the number of interrupt handlers that the PCI-MXI-2 supports.

Interrupt Handlers	Default Value
0 to 7	1

Number of Interrupters

This field gives the number of interrupters that the PCI-MXI-2 supports.

Interrupters	Default Value
0 to 7	0

Protocol Register

This field specifies the contents of the Protocol register, indicating which protocols the device supports. This field is meaningful only when the PCI-MXI-2 is configured as a message-based device. The default value is 0x0ff0 (Commander, Signal Register, Master).

Read Protocol Response

This field specifies the response value to a *Read Protocol* command received by the PCI-MXI-2 from the Resource Manager (if the PCI-MXI-2 is *not* the Resource Manager in your system). This field is meaningful only when the PCI-MXI-2 is configured as a message-based device. The default value is 0x8448 (Response Generation, Event Generation, Programmable Handler, Word Serial Trigger, Instrument, Extended Longword Serial, Longword Serial).

Bus Configuration Editor

Figure 6-6 shows the **Bus Configuration Editor**. The following paragraphs describe the options you can select for each of the fields.
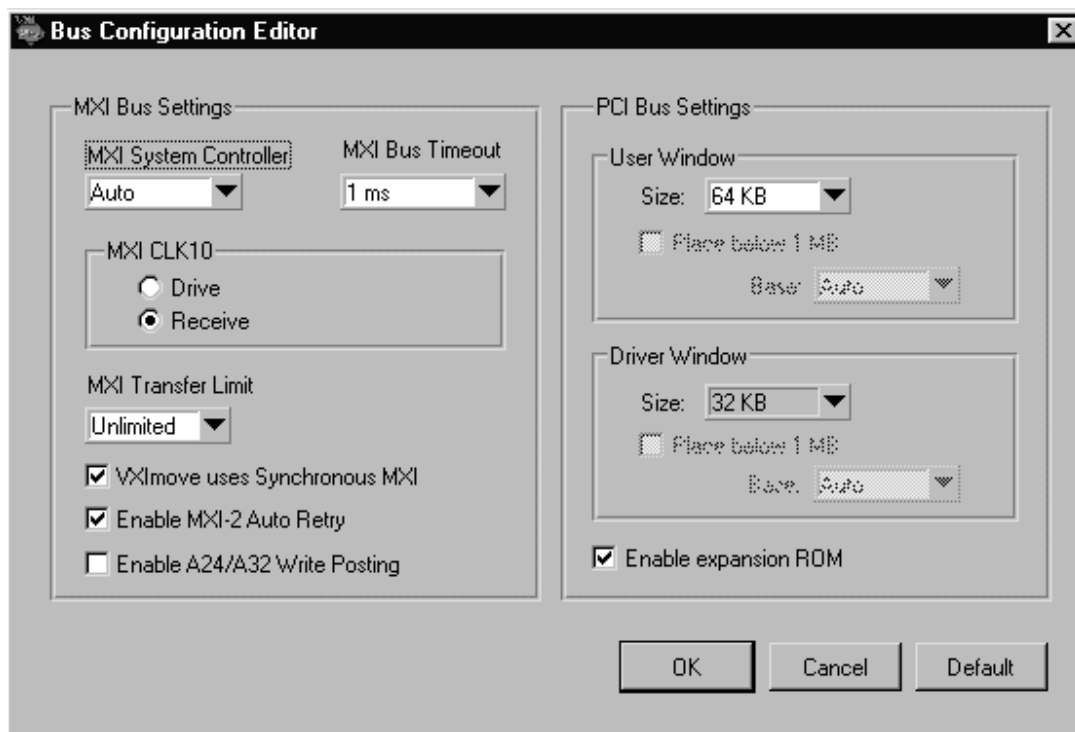
Figure 6-6. PCI-MXI-2 Bus Configuration Editor

MXI Bus Settings

The following paragraphs describe the options for the **MXI Bus** portion of this editor.

MXI System Controller

You can use the **MXI System Controller** control to determine whether the PCI-MXI-2 acts as the MXIbus System Controller. The three options are **Auto**, **Yes**, and **No**.

When the **Auto** setting is active (the default setting), the PCI-MXI-2 automatically can sense from the MXIbus cable whether it should be the controller. This setting requires that the cable is attached properly

before making any NI-VXI function calls or attempting to use the
VXI/VME-MXI-2 Configuration Editor. Refer to the *Connect the
MXIbus Cable* section at the end of either Chapter 3 or Chapter 4 of this
manual.

You can select the **Yes** or **No** options to manually determine whether the
PCI-MXI-2 should be the MXIbus System Controller. You must still be
certain to cable the MXIbus system appropriately when you make either
of these selections.

☞ **Note:**   *Make sure the MXI-2 cable is connected to the PCI-MXI-2 when you power
on or reboot the computer. This is required for the PCI-MXI-2 to
automatically detect whether it will be the MXIbus System Controller.*

MXI Bus Timeout

The MXI Bus Timeout (BTO) is a watchdog timer for transfers on the
MXIbus. The MXI BTO operates only when the PCI-MXI-2 is acting as
the MXIbus System Controller.

After the specified amount of time has elapsed, the BTO circuitry
terminates a MXIbus cycle if no slave has responded. The BTO is also
disabled when the current MXIbus cycle maps to the VXI/VMEbus
through a VXI/VME-MXI-2.

The default timeout value is 1 ms. If the **Enable MXI-2 Auto Retry**
checkbox option is checked, you should use a **MXI Bus Timeout** of
1 ms or greater.

MXI CLK10

The PCI-MXI-2 is capable of either receiving or driving the MXIbus
CLK10 signal.

You can use the **Drive** or **Receive** options of the **MXI CLK10** feature
to control the direction of the MXIbus CLK10 signal. By default all
MXI-2 boards receive MXI CLK10 (the **Receive** option is active);
therefore, you must choose one board on your MXI-2 bus to drive
CLK10 by changing the value of the control to **Drive**. For most
configurations, it is recommended to choose the System Controller as
the CLK10 source for simplicity. The only exception you may want to
make is if you want your triggers synchronous to the VXI 10 MHz
clock.

Warning: *Do not configure more than one MXIbus device to drive MXI CLK10. Having a second device driving MXI CLK10 could damage the device.*

MXI Transfer Limit

Use this feature to control how many data transfers the PCI-MXI-2 will perform on the MXIbus before releasing it to another master device that is requesting use of the bus. The default setting holds the MXIbus for an unlimited period of time.

The other options you can choose from are 256, 64, and 16 transfers. If you do not want the PCI-MXI-2 to hold the MXIbus for an unlimited period of time, you can use this control to select one of these values.

Synchronous MXI

The MXIbus has a special high-speed block protocol for transferring large blocks of data. This protocol, *SYNC-MXI*, locks the MXIbus during the transfer, which prevents anyone else from using the bus. VXImove() uses this protocol to transfer data. By default this option is enabled—the **VXImove uses Synchronous MXI** checkbox is checked.

If a VXI interrupt or signal comes in while a synchronous MXI move is underway, there can be a problem. You can disable SYNC-MXI programmatically in VXImove(). If you have an older NI-VXI application that does not disable SYNC-MXI programmatically, you can deselect the checkbox to force your application to not use SYNC-MXI, if necessary.

MXI-2 Auto Retry

The PCI-MXI-2 has an automatic retry feature for cycles that map from the MXI bus to the PCI bus. By default this option is enabled—the **Enable MXI-2 Auto Retry** checkbox is checked.

Normally, when a cycle maps from the MXI bus to the PCI bus, any retry response received on the PCI bus is passed to the MXI bus. When the MXI-2 auto retry feature is enabled, the PCI-MXI-2 automatically retries any PCI cycle when the PCI host responds to a cycle with a retry. The PCI-MXI-2 automatically continues to retry the PCI cycle until it receives either a *DTACK* or *BERR* response, which it then passes to the MXIbus. This is the default situation because many external masters do not support VXI/MXI retries. If the external master does support retries,

it may be beneficial to disable the MXI-2 auto retry feature. With this feature disabled, you can lower the **MXI Bus Timeout** value because there will be no delay due to the inward cycles being retried.

☞   **Note:**   *The PCI-MXI-2 has a limit on the number of automatic retries it will perform on any one cycle. If the limit is exceeded and the PCI-MXI-2 receives another retry, it will pass a retry back to the MXIbus even though the* **Enable MXI-2 Auto Retry** *checkbox is checked.*

A24/A32 Write Posting

This field determines whether to enable write posting for incoming slave accesses to A24/A32 VXI shared RAM. By default this option is disabled—the **Enable A24/A32 Write Posting** checkbox is cleared.

Enabling write posting will increase the throughput of your inward cycles. However, you should not enable write posting unless the destination of your inward A24/A32 cycles is onboard RAM, because cycles to onboard RAM will always complete successfully.

PCI Bus Settings

The following paragraphs describe the options for the **PCI Bus** portion of this editor.

User Window and Driver Window

The PCI-MXI-2 driver requires the use of two PCI windows: a user window and a driver window. Calling the MapVXIAddress() function allocates regions of the user window to your application. VXIpeek() and VXIpoke() accesses are performed through this window. NI-VXI uses the driver window to perform high-level functions such as VXIin() and VXIout(), and to access registers on the PCI-MXI-2 and VXI/VME-MXI-2.

The windows are mapped to PCI base address registers and determine the amount of PCI memory space the PCI-MXI-2 requests from the PCI system during initialization. You can set the window base, window size and whether the window resides above or below the 1 MB address space boundary.

♦   **DOS users**—If you intend to use DOS applications, notice that DOS restricts programs from accessing memory above 1 MB. Verify that the **Place below 1 MB** checkboxes for both the user window and the driver window are checked.

Window Size

The amount of space you can allocate for the user window is system dependent. You can use the **Size** control to select the size of the user window (minimum of 4 KB, maximum of 2 GB). The more you increase the size of the user window, the larger the window you can map in `MapVXIAddress()`.

You can also disable this option. Disabling the user window causes the PCI-MXI-2 to request the minimum amount of address space on the PCIbus. With the window disabled, you will be unable to perform any low-level function calls such as `VXIpeek()`, `VXIpoke()`, and `MapVXIAddress()`. For example, on DOS systems, you may not be able to request more than the 32 KB driver window if the address space between 640 KB and 1 MB is being used by other devices in your computer.

It is recommended to have a user window of at least 64 KB, and the default setting for the user window is set at this value. If you are going to be initiating transfers to a wide variety of addresses in both A24 and A32, you should increase the size of the user window.

The size of the driver window, however, is system defined and is not user configurable.

1 MB Address Boundary

Caution: *On most PCI-based computers, if the space requested by a plug-in board cannot be allocated, the computer will not boot. Therefore, if the space requested for this configuration causes complications to your computer, refer to the* Fixing an Invalid EEPROM Configuration *section at the end of Appendix C, EEPROM Configuration.*

The **Place below 1 MB** field determines whether the user or driver window is required to be located below the 1 MB address boundary (a DOS limitation). By default the checkboxes are not checked for either the user and driver windows, meaning that they are both above the 1 MB boundary. Unless you are running in DOS, you can keep the windows above the 1 MB boundary, and it is recommended that you keep the default settings.

Window Base

Normally, the **Base** field is set to **Auto** and the PCI plug&play utility automatically configures the user and driver windows to a certain region in the PCI address space. However, if you need to override the PCI plug&play assignment and put the user or driver windows in a different region in PCI address space below the 1 MB address boundary, use this field to assign a new address for these windows.

♦ **Windows 95/NT users**—This option is not available under Windows 95/NT. However, under Windows 95 you can change the base of the driver/user window by using the **Resources** page in the PCI-MXI-2 **Properties** dialog in the Device Manager. Please refer to your Windows 95 documentation for more information.

Notice that you can select the **Base** control only if the **Place below 1 MB** checkbox is checked.

☞ **Note:** *On some PCI-based computers, the BIOS will not adhere to your instructions to place the window base below the 1 MB boundary. Run* VXIinit *and check the display to verify that the address space is below the 1 MB boundary. If your BIOS does not properly locate the board below the 1 MB boundary, you can manually place the board by choosing a base address for window base rather than by using the* **Auto** *feature. Also, the BIOS of some PCI-based computers may incorrectly place the window base at a location that conflicts with other devices. You can use the* **Base** *control to override the BIOS in these cases as well.*

Expansion ROM

Use the **Enable expansion ROM** checkbox to enable or disable the PCI expansion ROM. The expansion ROM is enabled by default. It is recommended to retain the default setting. This option is included in VXIedit in case future versions of the PCI-MXI-2 do not implement a PCI expansion ROM.

VXI/VME-MXI-2 Configuration Editor

Before running the **VXI/VME-MXI-2 Configuration Editor**, you must run VXIinit and RESMAN.

☞ **Note:**   *Throughout this section, the term VXI/VME-MXI-2 denotes that the information applies equally to the VXI-MXI-2 or the VME-MXI-2.*

Upon entering the **VXI/VME-MXI-2 Configuration Editor**, the program displays a list of VXI/VME-MXI-2 boards that RESMAN detected in your system, as shown in Figure 6-7.
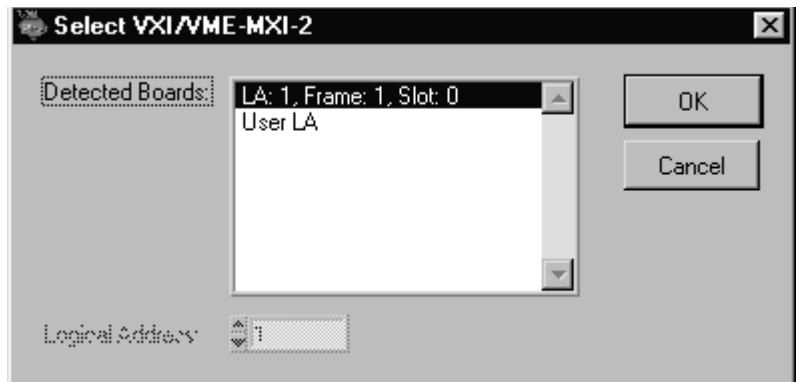
Figure 6-7. VXI/VME-MXI-2 Selection Dialog Box

Select the device you want to configure from the **Detected Boards** list, or you can select the **User LA** control and type in the device's logical address in the **Logical Address** field. Click **OK** to enter the editor or **Cancel** to return to the main menu.

After finding a VXI/VME-MXI-2, the **VXI/VME-MXI-2 Configuration Editor** displays a panel, as shown in Figure 6-8, that you can use to modify the configuration settings of that VXI/VME-MXI-2. The panel displays the current settings of the module. Notice that it also shows the hardware revision and serial number of the VXI/VME-MXI-2.

The title of the screen will reflect the model of the device you have. For instance, if you have a VXI-MXI-2, the title will read **VXI-MXI-2 Configuration Editor** as shown in the following screen.
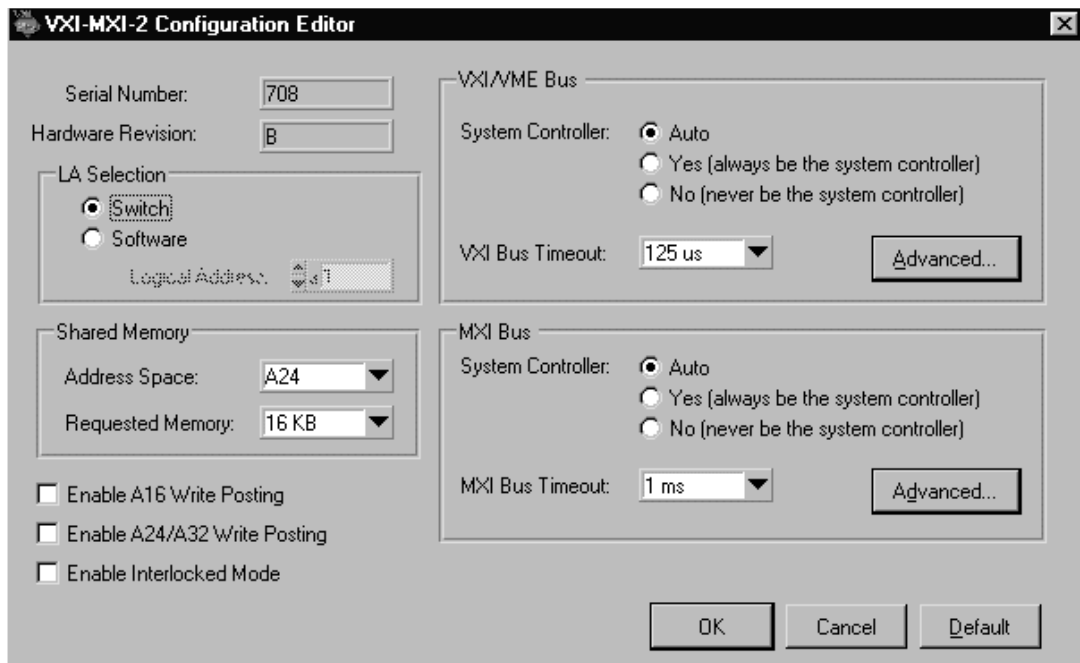
Figure 6-8. VXI/VME-MXI-2 Configuration Editor

LA Selection and Logical Address

You can set or modify the logical address of the VXI/VME-MXI-2
either within the **VXI/VME-MXI-2 Configuration Editor** itself or
with the onboard 8-position DIP switch. To select the configuration
method you prefer, use the **LA Selection** controls.

The default selection is the **Switch** option. Notice that the **Logical
Address** control is inaccessible, since it would have no effect. In this
option you need to change the hardware switch setting on the
VXI/VME-MXI-2 itself if you want to change the logical address.

If you select **Software** for this option, you can then use the **Logical
Address** control to select a logical address within the range of 1 to 254.
If you use this option, the hardware switch setting has no effect and you
must use the **VXI/VME-MXI-2 Configuration Editor** to change the
logical address.

Address Space and Requested Memory

The VXI/VME-MXI-2 requires at least 16 KB of address space in A24 space or at least 64 KB in A32 space. Use the **Address Space** control to select whether you want to use A24 space or A32 space. Use the **Requested Memory** control to set the amount of memory space that the VXI/VME-MXI-2 will request. You can select up to 8 MB in A24 space and up to 2 GB in A32 space. The default setting uses the minimum requirement of 16 KB in A24 space.

These controls are necessary if you change the amount of DRAM installed on the VXI/VME-MXI-2. The amount of memory you set with the **Requested Memory** control should match the amount of DRAM installed on the VXI/VME-MXI-2. If no DRAM is installed, keep the default setting of 16 KB. Notice that the smallest valid amount in A32 space is 64 KB.

Caution: *If you install DRAM into the VXI/VME-MXI-2, do not attempt to use the first 4 KB of memory space. This 4 KB space maps to the registers on the VXI/VME-MXI-2 and does not access onboard DRAM. Accessing this region will cause your VXI/VME-MXI-2 to behave incorrectly.*

If you do not want to lose 4 KB of DRAM you can get around this limitation by setting the **Requested Memory** control to double the amount that is installed on the VXI/VME-MXI-2, because the DRAM is aliased throughout the remainder of the requested memory space. The DRAM should then be accessed in the upper half of the requested memory space.

A16 Write Post and A24/A32 Write Posting

The VXI/VME-MXI-2 can increase performance with its capability to post write cycles from both the MXIbus and the VXI/VMEbus. Write cycles should be posted only to devices that cannot return a *BERR* signal, because the *BERR* will not be reported to the originating master.

Click on the checkbox control(s) if you want to use either A16 or A24/A32 write posting. By default, both options are disabled.

The A16 write posting control affects only write cycles that map through the A16 window from the VXI/VMEbus to the MXIbus and vice versa. A16 write cycles in VXI configuration space are never posted regardless of the setting of this control.

The A24/A32 write posting control affects write cycles that map through the A24 window and A32 window from the VXI/VMEbus to the MXIbus and vice-versa. This control also affects write cycles to the VXI/VME-MXI-2 itself via its requested memory space from both the VXI/VMEbus and the MXIbus. For more information on the A16, A24, and A32 windows, refer to VXI-6, the *VXIbus Mainframe Extender Specification*.

Interlocked Mode

Interlocked arbitration mode is an optional mode of operation in which at any given moment the system can perform as if it were one large VXI/VMEbus mainframe with only one master of the entire system— VXI/VMEbus and MXIbus. This mode of operation prevents deadlocks by interlocking all arbitration in the VXI/VMEbus/MXIbus system. By default, this option is disabled, which puts the VXI/VME-MXI-2 in normal operating mode.

In normal operating mode (non-interlocked), multiple masters can operate simultaneously in the VXI/VMEbus/MXIbus system. A deadlock occurs when a MXIbus master requests use of a VXI/VMEbus resource in another VXI/VMEbus mainframe while a VXI/VMEbus master in that mainframe is in the process of requesting a resource across the MXIbus. When this situation occurs, the VXI/VMEbus master must give up its bus ownership to resolve the conflict. The *RETRY* signal is used to terminate the transfer on the VMEbus; however, devices in the VXI/VMEbus mainframe must be able to detect a *RETRY* caused by a deadlock condition so that they can retry the operation. Any master device that cannot detect the retry protocol will interpret the response as a *BERR* signal instead.

The VXI/VME-MXI-2 is shipped from the factory configured for normal operating mode (non-interlocked). If MXIbus transfers will be occurring both into and out of the mainframe and the VXI/VMEbus modules in your system do not have the capability for handling retry conditions, you may want to configure the VXI/VME-MXI-2 for interlocked arbitration mode by clicking on the Enable checkbox. In this mode, no software provisions for deadlock conditions are required. However, parallel accesses in separate VXI/VMEbus mainframes are no longer possible, and system performance may be lower than in normal operating mode.

In a VXI/VMEbus/MXIbus system, you can configure some VXI/VME-MXI-2 modules for normal operating mode and others for interlocked arbitration mode. The VXI/VMEbus mainframes configured in interlocked arbitration mode will be interlocked with each other and the mainframes configured for normal operating mode can perform transfers in parallel.

This type of system configuration is recommended if you have one of the following situations:

• A VXI/VMEbus mainframe with only slave devices and no masters. Without bus masters, there is no chance for deadlock. You can configure the VXI/VME-MXI-2 devices in this mainframe for normal operating mode.

• A VXI/VMEbus mainframe with both masters and slaves, but the masters communicate only with the slaves in their mainframe. The masters never attempt transfers across the MXIbus, so there is no chance for deadlock when a MXIbus master attempts a transfer into the VXI/VMEbus mainframe. You can configure the VXI/VME-MXI-2 devices in this mainframe for normal operating mode.

• A VXI/VMEbus mainframe in which all masters that perform cycles across the MXIbus support the VME64 RETRY protocol. You can configure the VXI/VME-MXI-2 devices in this mainframe for normal operating mode because all masters that could cause a deadlock will automatically retry the operation.

VXI/VME Bus Options

Use the options in this group to control features of the VXI/VMEbus interface on the VXI/VME-MXI-2.

VMEbus System Controller

You can use the **System Controller** control to override the jumper setting on the VXI-MXI-2. The VME-MXI-2 does not have an onboard jumper setting for this option. When the **Auto** setting (the default setting) is active, the onboard jumper setting determines if the VXI-MXI-2 is the VXI Slot 0 device. For more information, refer to the *VXIbus Slot 0/Non-Slot 0* section in Chapter 3, *VXI-MXI-2 Configuration and Installation*.

Otherwise, choose either the **Yes** or **No** option. Notice that selecting either of these options overrides the onboard jumper setting on the

VXI-MXI-2, so it will not matter how the jumper is set. You would need to run the **VXI/VME-MXI-2 Configuration Editor** again if you decide to change the VMEbus System Controller (VXI Slot 0) setting at a later time.

Warning: *Do not install a VXI/VME-MXI-2 configured for VMEbus System Controller (VXI Slot 0) into another slot without first reconfiguring it to either Non-Slot 0 or automatic configuration. Neglecting to do this could damage the VXI/VME-MXI-2, the VXI/VMEbus backplane, or both.*

This means that you should use either the **No** *option or the* **Auto** *option of this control. For the VXI-MXI-2, you also have the option of changing the hardware jumper setting.*

VXI/VME Bus Timeout Value

The VXI/VMEbus Bus Timeout (BTO) is a watchdog timer for transfers on the VMEbus Data Transfer bus. After the specified amount of time has elapsed, the BTO circuitry terminates a VMEbus cycle if no slave has responded. The VXI/VME-MXI-2 must provide the VXI/VMEbus BTO for proper operation because when a MXIbus cycle is involved, the VXI/VMEbus timeout must be disabled and the MXIbus BTO enabled. You should disable the BTO of any other BTO module residing in the mainframe. If this is not possible, set its **VXI Bus Timeout** control to its maximum setting to give the MXIbus cycles as much time as possible to complete.

The lowest value in the allowable range is 15 µs and the highest value is 256 ms. The default value is 125 µs.

Advanced VXI Settings

Click on the **Advanced** button to reach additional configuration options for the VXI/VME Bus portion of this editor, as shown in Figure 6-9. These options are intended for more advanced users.
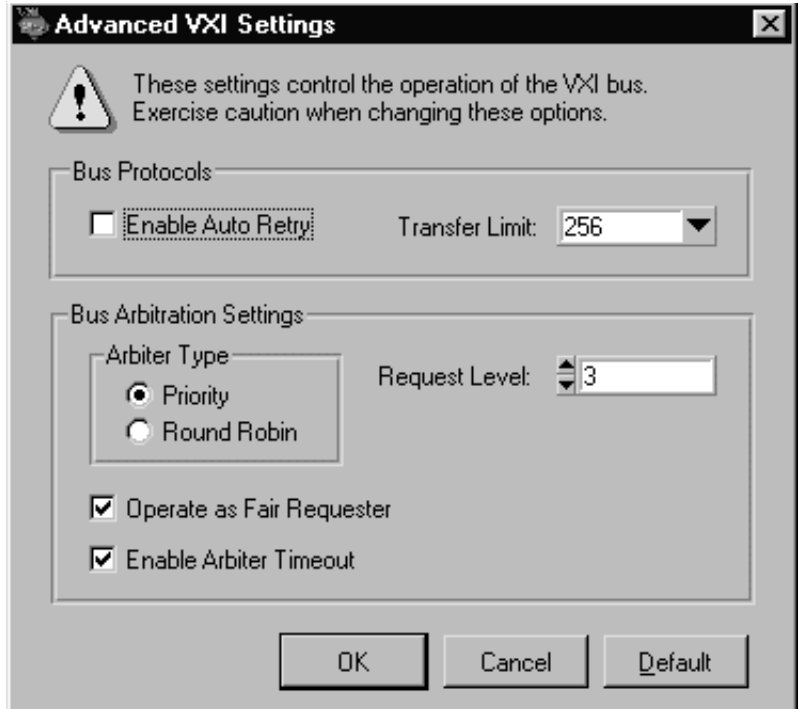
Figure 6-9. Advanced VXI Settings

VXI/VME Auto Retry

The VXI/VME-MXI-2 has an automatic retry feature for cycles that map from the VXI/VMEbus to the MXIbus. By default this option is disabled.

Normally, when a cycle maps from the VXI/VMEbus to the MXIbus, any retry response received on the MXIbus is passed to the VXI/VMEbus. If you enable the **Auto Retry** feature, the VXI/VME-MXI-2 automatically retries any MXI cycle that receives a retry response instead of passing a retry response back to the VXI/VMEbus. The VXI/VME-MXI-2 automatically continues to retry the MXI cycle until it receives either a *DTACK* or *BERR* response, which it then passes to the VXI/VMEbus.

Notice that the VXI/VME-MXI-2 has a limit on the number of automatic retries it will perform on any one cycle. If the limit is exceeded and the VXI/VME-MXI-2 receives another retry, it will pass a retry back to the VXI/VMEbus even though **Auto Retry** is enabled.

Transfer Limit

You can use this feature to control how many data transfers the
VXI/VME-MXI-2 will perform on the VXI/VMEbus before releasing it
to another master device that is requesting use of the bus.

The available options you can choose from are 16, 64, and 256 transfers.
If you do not want the VXI/VME-MXI-2 to hold the VXI/VMEbus long
enough to perform 256 transfers (the default value), you can use this
control to select a smaller value.

Arbiter Type

You can use the **Arbiter Type** feature to configure the
VXI/VME-MXI-2 as either a Priority or Round Robin VMEbus arbiter.
This control is applicable only if the VXI/VME-MXI-2 you are
configuring is a VMEbus System Controller (VXI Slot 0) device. The
default value is **Priority**.

When configured for **Priority** arbitration, the VXI/VME-MXI-2 grants
the bus to the highest pending bus request level. If you select **Round
Robin** arbitration mode, the VXI/VME-MXI-2 grants the bus to the
next highest bus request level after the level of the previous bus owner.
This effectively gives the same priority to each bus request level. Refer
to the VMEbus specification for more information on the different types
of arbiters.

Request Level

The VXI/VME-MXI-2 uses one of the four VMEbus request levels
(0 to 3) to request use of the VME Data Transfer Bus (DTB). The
VXI/VME-MXI-2 requests use of the DTB whenever an external
MXIbus device, such as a PCI-based computer with a PCI-MXI-2
interface, attempts a transfer that maps into the VXI/VMEbus
mainframe.

The VXI/VME-MXI-2 uses VMEbus request level 3 by default, as
required by the VXIbus specification. This is suitable for most VXIbus
systems. However, you can change the VXI/VME-MXI-2 to use any of
the other three request levels (0, 1, or 2) by changing the setting of the
Request Level control. You may want to change request levels to
change the priority of the VXI/VME-MXI-2 request signal. For more
information, refer to the VMEbus specification.

VXI/VME Fair Requester

The VXI/VME-MXI-2 is always a Release On Request requester. However, you can configure whether the VXI/VME-MXI-2 acts as either a fair or unfair requester on the VXI/VMEbus. By default the **Operate as Fair Requester** checkbox is enabled, signifying a fair requester. For more information on the different types of requesters, refer to the VMEbus specification.

Arbiter Timeout

An arbitration timeout feature is available on the VXI/VME-MXI-2 when it is acting as the VMEbus arbiter. This feature applies only to a VXI Slot 0 (VMEbus System Controller) VXI/VME-MXI-2. By default this option is enabled.

The timer begins when the arbiter circuit on the VXI/VME-MXI-2 drives one of the *BGOUT* lines on the backplane. If no device takes over the bus within the timeout limit, the *BGOUT* is removed and the bus is either idle or granted to another requester.

MXI Bus Options

Use the options in this group to control features of the MXIbus interface on the VXI/VME-MXI-2 module.

MXI Bus System Controller

You can use the **System Controller** control to determine whether the VXI/VME-MXI-2 acts as the MXI Bus System Controller. When the **Auto** setting (the default setting) is active, the VXI/VME-MXI-2 automatically can sense from the MXIbus cable whether it should be the controller.

You can select either **Yes** or **No** to manually determine if the VXI/VME-MXI-2 should be the MXI Bus System Controller. You must still be certain to cable the MXIbus system appropriately when you make either of these selections.

MXI Bus Timeout Value

The MXIbus Bus Timeout (BTO) is a watchdog timer for transfers on the MXIbus. The MXIbus BTO unit operates only when the VXI/VME-MXI-2 is acting as the MXIbus System Controller. The

functionality of this control is similar to that of the **VXI Bus Timeout** control described previously under the *VXI/VME Bus Options* section. The options range from 8 μs to 128 ms, with a default value of 1 ms.

After the specified amount of time has elapsed, the BTO circuitry terminates a MXIbus cycle if no slave has responded. The BTO circuitry is automatically deactivated when the VXI/VME-MXI-2 is not acting as the MXIbus System Controller. The BTO is also disabled when the current MXIbus cycle maps to the VXI/VMEbus through a VXI/VME-MXI-2.

Advanced MXI Settings

Click on the **Advanced** button to reach additional configuration options for the MXI Bus portion of this editor, as shown in Figure 6-10. These options are intended for more advanced users.
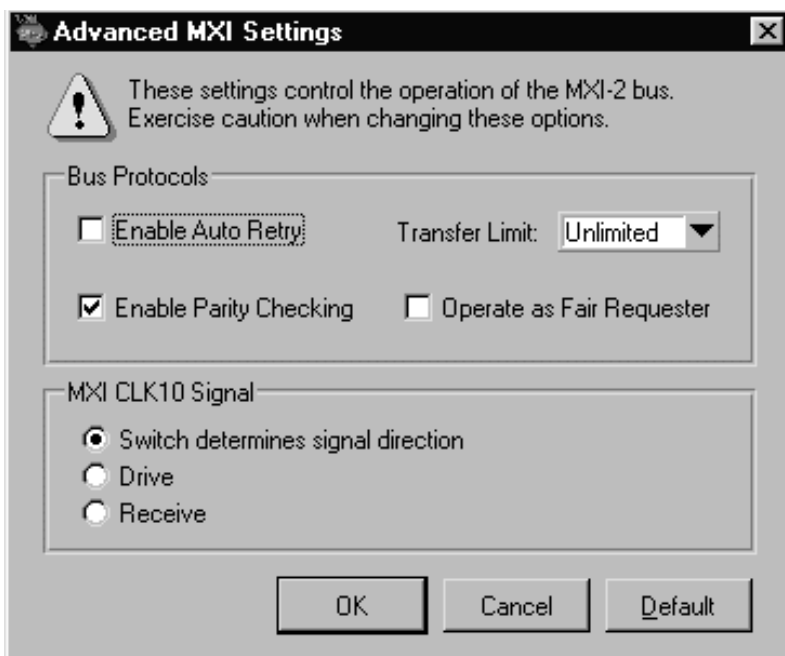
Figure 6-10. Advanced MXI Settings

MXI Auto Retry

The VXI/VME-MXI-2 has an automatic retry feature for cycles that map from the MXIbus to the VXI/VMEbus. This feature works in the same manner as the **Auto Retry** control described previously under the *VXI/VME Bus Options* section. By default, this option is disabled.

Normally, when a cycle maps from the MXIbus to the VXI/VMEbus, any retry response received on the VXI/VMEbus is passed to the MXIbus. If you enable the **Auto Retry** feature, the VXI/VME-MXI-2 automatically retries any VXI/VME cycle that receives a retry response instead of passing a retry response on to the MXIbus. The VXI/VME-MXI-2 automatically continues to retry the VXI/VME cycle until it receives either a *DTACK* or *BERR* response, which it then passes to the MXIbus.

Notice that the VXI/VME-MXI-2 has a limit on the number of automatic retries it will perform on any one cycle. If the limit is exceeded and the VXI/VME-MXI-2 receives another retry, it will pass a retry back to the MXIbus even though **Auto Retry** is enabled.

Transfer Limit

You can use this feature to control how many data transfers the VXI/VME-MXI-2 will perform on the MXIbus before releasing it to another master device that is requesting use of the bus. The default setting holds the MXIbus for an unlimited period of time.

The other options you can choose from are 16, 64, and 256 transfers. If you do not want the VXI/VME-MXI-2 to hold the MXIbus for an unlimited period of time, you can use this control to select one of these values.

Parity Checking

By default, MXIbus parity checking is enabled and should not be disabled under normal circumstances. MXIbus parity is always generated regardless if checking is enabled or disabled.

MXI Fair Requester

You can use the **Operate as Fair Requester** checkbox control to configure the VXI/VME-MXI-2 as either a fair or unfair requester on the MXIbus. In its default setting (disabled), the VXI/VME-MXI-2 will request the bus at any time. If you enable this option, the

VXI/VME-MXI-2 will request the MXIbus only when there are no requests pending from other MXIbus masters. This prevents other MXIbus masters from being starved of bandwidth.

MXI CLK10 Signal

The VXI-MXI-2 can either receive or drive the MXIbus CLK10 signal. In its default setting, the VXI-MXI-2 uses the switch setting of S7 to determine the signal direction.

♦ **VME users**—This option is not applicable to the VME-MXI-2.

You can use the **Drive** or **Receive** options to override the setting of S7 and control the direction of the MXIbus CLK10 signal. When receiving the MXIbus CLK10 signal, configure the W3 jumper setting to use the MXIbus as the source for generating the VXIbus CLK10 (applicable only if the VXI-MXI-2 is a Slot 0 device). When driving the MXIbus CLK10, the VXIbus CLK10 is used as the source. In this case, change the jumper setting so that it does *not* use the MXIbus CLK10 as the source for the VXIbus CLK10.

Warning: *Do not configure more than one MXIbus device to drive MXI CLK10. Setting up a second device to drive MXI CLK10 could damage the device.*

Using the NI-VXI Software

This chapter discusses programming information for you to consider when developing applications that use the NI-VXI driver.

After installing the driver software, you can begin to develop your VXI application software. Be sure to check the README.DOC file for the latest application development notes.

♦ **DOS and Windows 3.1 users**—You must run the VXIinit initialization program before performing any VXI operations and after each computer reset.

You must also run RESMAN each time the chassis power is cycled so that your application can access devices in the VXI chassis.

The NI-VXI software was designed for use in VXI systems. Because VXI is a superset of VME, you can also use the NI-VXI functions as a comprehensive set of programming tools for VME systems. Refer to the *NI-VXI User Manual* and the *NI-VXI Programmer Reference Manual* for an overview of NI-VXI and detailed descriptions of the NI-VXI functions. Notice that the following function classes apply only to VXI systems:

- Commander Word Serial Protocol functions
- Servant Word Serial Protocol functions
- VXI Signal functions
- VXI Trigger functions

Interactive Control of NI-VXI

The easiest way to learn how to communicate with your instruments is by controlling them interactively. Use the VXI interactive control utility (VIC or VICtext) to write to and read from your instruments. This utility displays the status of your VXI transactions and informs you of any errors that occur.

Refer to the *NI-VXI Graphical Utilities Reference Manual* for instructions on how to use VIC and to learn about its features. If you are using VICtext refer to the *NI-VXI Text Utilities Reference Manual* for information.

Warning: **DOS and Windows 3.1 users—***If* NIVXI.DLL *is loaded in memory, do not attempt to execute* VIC *or any DOS program that uses the NI-VXI library in a DOS shell. Conflicts occur if both the DOS and Windows NI-VXI drivers are active at the same time, and may cause a system failure. To guard against this conflict, the safest approach is to always exit Windows before attempting to execute any DOS program that uses the NI-VXI library, including* VIC*. You can execute* VIC *from a Windows DOS shell, however, if you ensure that no other Windows application that uses the* NIVXI.DLL *file is executing.*

Note: *When compiling NI-VXI applications, you must define one of these macros in your makefile/project:*

- VXIWIN *(if you are developing a Windows 3.1 application),*
- VXIDOS *(if you are developing a DOS application), or*
- VXINT *(if you are developing a Windows 95/NT application).*

Refer to the example programs on your software diskettes for details.

Example Programs

The EXAMPLES subdirectory contains various example programs along with a makefile that show how to use various functions in the NI-VXI software and how to develop application programs using these functions. Make certain that the environment variables LIB and INCLUDE are set correctly as described in Chapter 5, *NI-VXI Software Installation*. Also refer to the *NI-VXI User Manual* and the *NI-VXI Programmer Reference Manual* for additional examples.

Programming Considerations

The following paragraphs contain information for you to consider when developing DOS or Windows 95/NT/3.1 applications that use the NI-VXI bus interface software. This information applies to all four Microsoft operating systems unless otherwise noted.

Memory Model (DOS or Windows 3.1 Only)

The NI-VXI libraries were compiled using the large memory model. All DOS applications must also be compiled for the large memory model. However, Windows 3.1 application programs that link with the NI-VXI library can also use the medium, compact, or small memory models. Because of this ability to use different memory models for your application, you not only can take advantage of the efficiency inherent in small memory model programs, but also run multiple instances of the application as well. (Normally, you cannot run multiple instances of an application if it is a large memory model application.)

Multiple Applications Using the NI-VXI Library

Multiple-application support is another feature in the NI-VXI library. You can have several applications that use the NI-VXI library running simultaneously in Windows 95/NT/3.1. In addition, you can have multiple instances of the same application that uses the NI-VXI library running simultaneously. The NI-VXI functions perform in the same manner whether you have only one application that uses the NI-VXI library or several applications (or several instances of an application) all using the NI-VXI library.

However, you do need to be careful in certain cases as described in the following section.

Low-Level Access Functions

The memory windows used to access the VXIbus are a limited resource. You should follow the protocol of calling the MapVXIAddress() function with Access Only mode first before attempting to perform low-level VXIbus access with VXIpeek() or VXIpoke(). Your application should always call the UnMapVXIAddress() function immediately after the accesses are complete so that you free up the memory window for other applications.

The function `MapVXIAddress()` returns a pointer for use with
low-level access functions. It is strongly recommended to use the
`VXIpeek()` and `VXIpoke()` macros to access the memory instead of
directly dereferencing the pointer. Using these macros makes the
NI-VXI software more portable between platforms, because some
platforms (such as the AT-MXI) require checking for retries, which can
be handled through the macros. Directly dereferencing the pointers does
not give you any speed benefit because the macros reduce to pointer
dereferences at compile time for the PCI-MXI-2. Refer to the
Compiling Your C Program section later in this chapter for more
information on portability issues, and to your NI-VXI software
reference manual for more information on low-level VXIbus or
VMEbus access functions.

Setting User Handlers (DOS or Windows 3.1 Only)

You can set a user handler that will be invoked when certain conditions
occur, such as VXI signals and triggers, by using functions such as
`SetSignalHandler()` in the NI-VXI library. However, setting a new
user handler replaces the existing handler, meaning that the existing
handler will no longer be invoked when the condition occurs. The
following example illustrates the point.

```
SetSignalHandler(5, mySignalHandler1)

/* mySignalHandler1 is now the handler for VXI
signals from logical address 5. */

SetSignalHandler(5, mySignalHandler2)

/* mySignalHandler1 is replaced by mySignalHandler2
as the new handler for VXI signals from logical
address 5. */
```

Caution: **DOS and Windows 3.1 users**—*Avoid running multiple applications that
indiscriminately change user handlers, or try to make sure that when the
applications do change the user handlers, they do so in a coordinated
manner. Setting new handlers while using multiple applications runs the
risk of one application inadvertently overwriting the handler that another
application had set up, causing disruptions and incorrect behavior.*

Local Resource Access Functions

By using `VXIedit` or `VXItedit`, you can set up the PCI-MXI-2 to share either the system memory on the motherboard or the onboard memory on the PCI-MXI-2 with the VXI/VME system. Refer to the *NI-VXI Graphical Utilities Reference Manual* or the *NI-VXI Text Utilities Reference Manual* for more information on setting these parameters.

Notice that sharing the system memory with the VXI/VME system does not mean that the entire range of shared system memory is available to be used for VXI transfers. You need to be cautious in specifying the portion of memory you want to share, as some areas are already used for other purposes.

Warning: *Use* `VXImemAlloc()` *to allocate a buffer in the system memory that is reserved for your use only. Using any range of addresses that was not returned from* `VXImemAlloc()` *to receive data may cause your computer to crash or behave incorrectly.*

Another factor to consider is that although you may have selected to share 8 MB, you must also inform Windows 95/NT/3.1 how much memory to set aside for possible `VXImemAlloc()` calls. You can use the **Shared Memory Pool** option in `VXIedit` and `VXItedit` for this purpose. But remember that the memory you put into this pool is no longer available to Windows. If this setting is too large, you may experience memory limitation problems when you run Windows. Also remember that changes in the size of the pool do not take effect until the next time you start Windows.

The onboard memory on your PCI-MXI-2, on the other hand, is entirely available to you. You can obtain the VXI address of your onboard memory using the `GetDevInfo()` function. When you have the VXI address, you can access that memory using high-level or low-level VXIbus access functions.

System Configuration Functions

The System Configuration functions provide the lowest-level initialization of your NI-VXI software and VXI controller. You must use the `InitVXIlibrary()` function at the start of each application and the `CloseVXIlibrary()` function at the end of each application.

Compiling Your C Program

You can use the sample programs included with the NI-VXI software as a starting point to develop your own C program that uses NI-VXI functions. First, look over and compile the sample program using the makefile provided to get familiar with how the functions operate. The example program is broken into multiple files, and each file shows how to use different groups of functions. You can then modify the sample program to try out different aspects of the NI-VXI software.

♦ **Windows 3.1 users**—The sample Windows 3.1 program for the Microsoft C compiler is in the `\nivxi\win\msc\examples` directory, and the sample Windows program for the Borland C compiler is in the `\nivxi\win\borlandc\examples` directory.

♦ **Windows 95/NT users**—The sample Windows 95/NT program for the Microsoft C compiler is in the `\nivxi\win32\msc\examples` directory, and the sample Windows 95/NT program for the Borland C compiler is in the `\nivxi\win32\borlandc\examples` directory.

♦ **DOS users**—The sample DOS program for the Microsoft C compiler is in the `\nivxi\dos\msc\examples` directory, and the sample DOS program for the Borland C compiler is in the `\nivxi\dos\borlandc\examples` directory.

The easiest way to compile the sample program is to use the makefile included with the NI-VXI software. If you are using the Microsoft C compiler, go to the Microsoft C sample directory and type `nmake` to compile that program. If you are using the Borland C compiler, go to the Borland C sample directory and type `make -f example.mak`.

Symbols

You may need to define some symbols so that the NI-VXI library can work properly with your program. You can define the symbols using `#define` statements in the source code or you can use either the `/D` or the `-D` option in your compiler (both the Microsoft and Borland compilers support the `/D` and `-D` options). If you use `#define` statements, you must define the symbols before including the NI-VXI header file `nivxi.h`. If you use the makefiles to compile the sample program, the makefile already defined the necessary symbols.

One of the following symbols is usually required. You must define it when using the Microsoft C or Borland C compiler.

- VXIWIN designates the application as a Windows 3.1 application.

- VXINT designates the application as a Windows 95/NT application.

- VXIDOS designates the application as a DOS application. You can use the same NI-VXI header files to compile DOS programs by defining VXIDOS.

☞ **Note:** *Because LabWindows/CVI cannot be used to compile DOS programs, the correct symbol is automatically defined. You should not define* VXIWIN *or* VXINT *when using LabWindows/CVI.*

The following symbol is optional.

- BINARY_COMPATIBLE makes the application binary compatible with embedded VXI controllers, such as the National Instruments VXIpc series of embedded controllers. Using this option may cause a slight performance degradation when using low-level VXIbus access functions.

If you define these symbols in your source code, your source code should look something like the following sample code:

```
#define VXIWIN
#define BINARY_COMPATIBLE
.
.
.
#include <nivxi.h>
```

If you define these symbols using the /D or –D compiler options, you should specify the following when invoking the compiler.

For the Microsoft C compiler:

```
/DVXIWIN /DBINARY_COMPATIBLE
```

For the Borland C compiler:

```
–DVXIWIN; BINARY_COMPATIBLE;
```

Refer to the documentation that came with your compiler package for detailed instructions about using the compiler and the various tools (linker, debugger, and so on). Your compiler documentation is an important and useful source of information for writing, compiling, and debugging C programs.

Specifications

This appendix lists various module specifications of the PCI-MXI-2, VXI-MXI-2, and VME-MXI-2 such as physical dimensions and power requirements.

PCI-MXI-2

The following pages list the specifications for the PCI-MXI-2 module.

MXIbus Capability Descriptions

- Master-mode A32, A24, and A16 addressing
- Master-mode block transfers and synchronous block transfers
- Slave-mode A32, A24, and A16 addressing
- Slave-mode block transfers and synchronous block transfers
- Master-mode D32, D16, and D08 data sizes
- Slave-mode D32, D16, and D08 data sizes
- Optional MXIbus System Controller
- Can be a fair MXIbus requester
- Can lock the MXIbus for indivisible transfers
- Can terminate the MXIbus
- MXIbus master retry support
- MXIbus slave retry support
- Interrupt handler for levels 7 to 1
- Interrupt requester for levels 7 to 1
- MXIbus D32, D16, D08(O) interrupt handler
- MXIbus D32, D16, D08(O) interrupter
- Release on Acknowledge or Register Access interrupter
- MXIbus bus timer (programmable limit)
- Automatic MXIbus System Controller detection

PCI Functionality

Characteristic	Specification
PCI Initiator (Master) Capability	Supported
PCI Target (Slave) Capability	Supported
Data Path	32 bits
Card Voltage/Type	5 V only; 32-bit half-size card
Parity Generation/Checking, Error Reporting	Supported
Target Decode Speed	Medium (1 clock)
Target Fast-Back-to-Back Capability	Supported
Resource Locking	Supported as a master and slave
PCI Interrupts	Interrupts passed on INTA# signal
Base Address Registers	BAR 0 dedicated to local registers BAR 1–3 size configurable from 256 B to 4 GB
Expansion ROM	8 KB
PCI Master Performance (Ideal Maximum)	132 MB/s (16 Dwords max.)
PCI Slave Performance (Ideal Maximum)	33 MB/s (to local registers)

Environmental

Characteristic	Specification
Temperature	0° to 55° C operating; -40° to 85° C storage
Relative Humidity	0% to 95% noncondensing, operating; 0% to 95% noncondensing, storage
EMI	FCC Class A Verified

Requirements

Characteristic	Specification
Memory Space	32 KB minimum, programmable

Physical

Characteristic	Specification
Board Dimensions	174.63 by 106.68 mm (6.875 by 4.2 in.)
Connectors	Single fully implemented MXI-2 connector
Slot Requirements	Single PCI slot
MTBF	Contact factory
Weight	0.18 Kg (0.4 lb) typical (no DRAM installed)

Electrical

Source	Typical	Direct Current (Max)
+5 VDC	2.2 A	3.5 A

Performance

MXI Transfer Rate	
Peak	33 MB/s
Sustained	23 MB/s

VXI-MXI-2

The following pages list the specifications for the VXI-MXI-2 module.

MXIbus Capability Descriptions

- Master-mode A32, A24, and A16 addressing
- Master-mode block transfers and synchronous block transfers
- Slave-mode A32, A24, and A16 addressing
- Slave-mode block transfers and synchronous block transfers
- Master-mode D32, D16, and D08 data sizes
- Slave-mode D32, D16, and D08 data sizes
- Optional MXIbus System Controller
- Can be a fair MXIbus requester
- Can lock the MXIbus for indivisible transfers
- Can terminate the MXIbus
- MXIbus master retry support
- MXIbus slave retry support
- Interrupt handler for levels 7 to 1
- Interrupt requester for levels 7 to 1
- MXIbus D32, D16, D08(O) interrupt handler
- MXIbus D32, D16, D08(O) interrupter
- Release on Acknowledge or Register Access interrupter
- MXIbus bus timer (programmable limit)
- Automatic MXIbus System Controller detection
- Automatic MXIbus termination detection

VMEbus Capability Codes

Capability Code	Description
A32, A24, A16 (master)	VMEbus master A32, A24, and A16 addressing
A32, A24, A16 (slave)	VMEbus slave A32, A24, and A16 addressing
D32, D16, D08(EO) (master)	VMEbus master D32, D16, and D08 data sizes
D32, D16, D08(EO) (slave)	VMEbus slave D32, D16, and D08 data sizes
BLT, MBLT (master)	VMEbus master block and D64 transfers
BLT, MBLT (slave)	VMEbus slave block and D64 transfers
RMW (master)	VMEbus master read/modify/write transfers
RMW (slave)	VMEbus slave read/modify/write transfers
RETRY (master)	VMEbus master retry support
RETRY (slave)	VMEbus slave retry support
FSD	First slot detector
SCON	VMEbus System Controller
PRI, RRS	Prioritized or Round Robin Select arbiter
ROR, FAIR	Release on Request and FAIR bus requester
IH(7–1)	Interrupt handler for levels 7–1
I(7–1)	Interrupt requester for levels 7–1
D32, D16, D08(O) (Interrupt Handler)	VMEbus D32, D16, D08(O) interrupt handler
D32, D16, D08(O) (Interrupter)	VMEbus D32, D16, D08(O) interrupter
ROAK, RORA	Release on Acknowledge or Register Access interrupter
BTO(x)	VMEbus bus timer (programmable limit)

Requirements

Characteristic	Specification
VXIbus Configuration Space	64 B
A24 or A32 Space	16 KB minimum (programmable)

Environmental

Characteristic	Specification
Temperature	0° to 55° C operating; -40° to 85° C storage
Relative Humidity	0% to 95% noncondensing, operating; 0% to 95% noncondensing, storage
EMI	FCC Class A Verified

Physical

Characteristic	Specification
Board Dimensions	Fully enclosed, shielded VXI C-size board 233.35 by 340 mm (9.187 by 13.386 in.)
Connectors	Single fully implemented MXI-2 bus connector and three SMB connectors
Slot Requirements	Single VXI C-size slot
Compatibility	Fully compatible with VXI specification
VXI Keying Class	Class 1 TTL
MTBF	Contact factory
Weight	1.027 Kg (2.26 lb) typical (no DRAM installed)

Electrical

	DC Current Ratings	
Source	**Typical**	**Maximum**
+5 VDC	2.5 A	3.5 A
-5.2 VDC	180 mA	225 mA
-2 VDC	80 mA	100 mA

Performance

VME Transfer Rate	
Peak	33 MB/s
Sustained	23 MB/s

VME-MXI-2

The following pages list the specifications for the VME-MXI-2 module.

MXIbus Capability Descriptions

- Master-mode A32, A24 and A16 addressing
- Master-mode block transfers and synchronous block transfers
- Slave-mode A32, A24, and A16 addressing
- Slave-mode block transfers and synchronous block transfers
- Master-mode D32, D16, and D08 data sizes
- Slave-mode D32, D16, and D08 data sizes
- Optional MXIbus System Controller
- Can be a fair MXIbus requester
- Can lock the MXIbus for indivisible transfers
- Can terminate the MXIbus
- MXIbus master retry support
- MXIbus slave retry support
- Interrupt handler for levels 7 to 1
- Interrupt requester for levels 7 to 1
- MXIbus D32, D16, D08(O) interrupt handler
- MXIbus D32, D16, D08(O) interrupter
- Release on Acknowledge or Register Access interrupter
- MXIbus bus timer (programmable limit)
- Automatic MXIbus System Controller detection
- Automatic MXIbus termination detection

VMEbus Capability Codes

Capability Code	Description
A32, A24, A16 (master)	VMEbus master A32, A24, and A16 addressing
A32, A24, A16 (slave)	VMEbus slave A32, A24, and A16 addressing
D32, D16, D08(EO) (master)	VMEbus master D32, D16, and D08 data sizes
D32, D16, D08(EO) (slave)	VMEbus slave D32, D16, and D08 data sizes
BLT, MBLT (master)	VMEbus master block and D64 transfers
BLT, MBLT (slave)	VMEbus slave block and D64 transfers
RMW (master)	VMEbus master read/modify/write transfers
RMW (slave)	VMEbus slave read/modify/write transfers
RETRY (master)	VMEbus master retry support
RETRY (slave)	VMEbus slave retry support
FSD	First slot detector
SCON	VMEbus System Controller
PRI, RRS	Prioritized or Round Robin Select arbiter
ROR, FAIR	Release on Request and FAIR bus requester
IH(7–1)	Interrupt handler for levels 7–1
I(7–1)	Interrupt requester for levels 7–1
D32, D16, D08(O) (Interrupt Handler)	VMEbus D32, D16, D08(O) interrupt handler
D32, D16, D08(O) (Interrupter)	VMEbus D32, D16, D08(O) interrupter
ROAK, RORA	Release on Acknowledge or Register Access interrupter
BTO(x)	VMEbus bus timer (programmable limit)

Requirements

Characteristic	Specification
A16 Space	64 B
A24 or A32 Space	16 KB minimum (programmable)

Environmental

Characteristic	Specification
Temperature	0° to 55° C operating; -40° to 85° C storage
Relative Humidity	0% to 95% noncondensing, operating; 0% to 95% noncondensing, storage
EMI	FCC Class A Verified

Physical

Characteristic	Specification
Board Dimensions	VMEbus double-height board 233.36 by 160 mm (9.187 by 6.2999 in.)
Connectors	Single fully implemented MXI-2 bus connector
Slot Requirements	Single VMEbus double-height slot
Compatibility	Fully compatible with VMEbus specification
MTBF	Contact factory
Weight	0.33 Kg (0.73 lb) typical (no DRAM installed)

Electrical

	DC Current Ratings	
Source	**Typical**	**Maximum**
+5 VDC	2.2 A	3.2 A

Performance

VME Transfer Rate	
Peak	33 MB/s
Sustained	23 MB/s

NI-VXI Software Overview

This appendix lists and describes the main programs and files that make up the NI-VXI software.

Main Programs and Files

This section lists the main programs and files that you can use for controlling your VXI/VME interface.

☞ **Note:** *Any executable not listed in this section is used by the driver and should not be executed by the user directly.*

- VXIINIT.EXE is the PCI-MXI-2 initialization program. You can execute VXIinit from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell, although the latter is not recommended. This program initializes the PCI-MXI-2. It may be included in the DOS batch file AUTOEXEC.BAT so that the PCI-MXI-2 is automatically initialized at startup. The configuration settings can be modified using the VXIEDIT.EXE or VXITEDIT.EXE program.

- RESMAN.EXE is the National Instruments multiple-mainframe Resource Manager. You can execute RESMAN from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell, although the latter is not recommended. Under DOS and Windows 3.1, RESMAN.EXE may be executed only after VXIINIT.EXE has been run.

- VIC.EXE is an interactive control program that executes functions you enter from the keyboard. VIC helps you learn the functions, program your VXI devices, and develop and debug your application program. You can execute VIC from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell, although the latter is not recommended. If you do use VIC from the Windows 3.1 DOS shell, you must ensure that no other Windows application that uses NI-VXI functions is executing. This program is described in detail in the *NI-VXI Graphical Utilities Reference Manual*.

- VICTEXT.EXE is a text-based interactive control program that is functionally equivalent to VIC. You can execute VICtext from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell, although the latter is not recommended. If you run VICtext as a Windows application, you can use it at the same time that other Windows applications that use NI-VXI functions are executing. This program is described in detail in the *NI-VXI Text Utilities Reference Manual.*

- VXIEDIT.EXE is the VXI resource editor program that runs in DOS or Windows 95/NT/3.1. You use the **Non-VXI Device Editor** in VXIedit to identify details about VME devices installed in your system. You must use this editor to instruct your system about the addresses your VME devices occupy. The Resource Manager can then use this configuration information to automatically open the hardware windows so that your PC can access the VMEbus. This program also displays the system configuration information generated by the Resource Manager after it configures the link to the VMEbus. In VXI systems, you also use VXIedit to edit the model names of VXI devices and the manufacturer name and ID numbers. This program is described in detail in the *NI-VXI Graphical Utilities Reference Manual.*

- VXITEDIT.EXE is the text-based VXI resource editor program that is functionally equivalent to VXIedit. You can execute VXItedit from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell. This program is described in detail in the *NI-VXI Text Utilities Reference Manual.*

- README.DOC contains the latest updates and corrections to the manual when appropriate.

Header Files

The C:\NIVXI\INCLUDE directory contains the following include files for the Microsoft C and Borland C language interfaces.

- NIVXI.H is the main header file containing the C prototypes for the NI-VXI functions.

- DATASIZE.H contains data size specifications.

- BUSACC.H contains parameter and return values for the bus access functions.

- DEVINFO.H contains parameter and return values for the device information and system configuration functions.

- `VXIINT.H` contains parameter and return values for the interrupt and signal functions.

- `SYSINT.H` contains parameter and return values for the system interrupt functions.

- `TRIG.H` contains parameter and return values for the trigger functions. This file is useful in VXI systems but is not applicable for VME systems.

- `WS.H` contains parameter and return values for the Commander and Servant Word Serial functions. This file is useful in VXI systems but is not applicable for VME systems.

- `NIVXI.INC` is the include file for the Visual Basic for DOS language interface.

EEPROM Configuration

This appendix describes how to control the operation of the PCI-MXI-2 onboard EEPROM and how to fix an invalid EEPROM setting.

The EEPROM stores default registers values that are loaded at power-on. The EEPROM is divided into two halves so that you can modify one half, while the factory-configured half retains a back-up of the default user settings.

Controlling the EEPROM Operation

Use switch 1 (FOV) of the four-position switch at location U17 to control the operation of the EEPROM. Switch 1 determines whether the PCI-MXI-2 boots off the factory-configured half or the user-configurable half. In its default setting, the PCI-MXI-2 boots off the user-configurable half. This switch is useful in the event that the configuration becomes corrupted in such a way that the PCI-MXI-2 boots to an unusable state.

The TST switch (switch 2 of U17) lets you change the default factory configuration settings by permitting writes to the factory settings section of the EEPROM. This switch serves as a safety measure and should not be needed under normal circumstances. When this switch is off (its default setting) the factory configuration of the EEPROM is protected so any writes to the factory area will be ignored. The factory area is protected regardless of the setting of switch 1 of U17.

Figure C-1 shows the default settings for EEPROM operation.

Caution: *Do not alter the settings of switches 3 and 4 of U17. Leave these switches as shown in Figure C-1 unless specifically directed by National Instruments.*
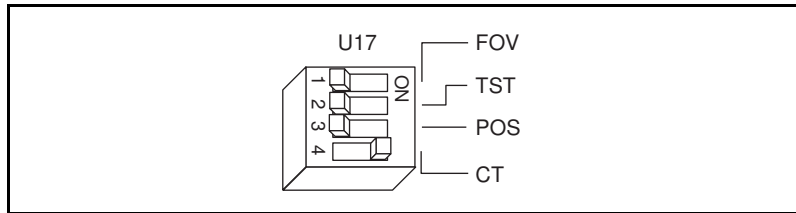
Figure C-1. EEPROM Operation

Fixing an Invalid EEPROM Configuration

Certain EEPROM configurations can cause your PCI computer to lock up while in its boot process. Generally, only the size and location of the memory windows can cause problems with the PCI-MXI-2 locking up your system. For example, many PCI-based computers will not boot if a board in its system requests more memory space than the computer can allocate. If you encounter this situation you should reduce the size of the PCI-MXI-2 user window.

If this situation occurs after changing the configuration on the PCI-MXI-2, follow these steps to reconfigure the PCI-MXI-2.

1.   Turn your computer off.

Warning: *To protect both yourself and the computer from electrical hazards, the computer should remain off while changing the settings on the PCI-MXI-2 module.*

2.   Remove the top cover or access port to the PCI bus.

3.   Change switch 1 (FOV) on U17 to the ON position as shown in Figure C-2 to restore the factory configuration.
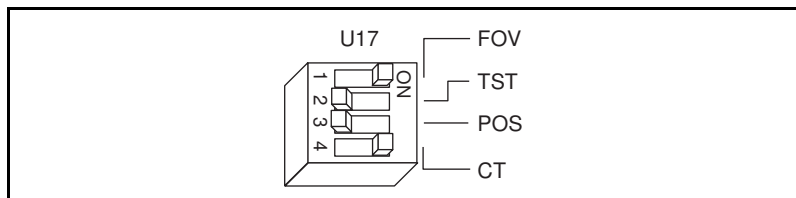
Figure C-2. Restoring the Factory Configuration

Note: *If you have to remove the PCI-MXI-2 module to access switch 1, follow the installation instructions given in Chapter 2,* PCI-MXI-2 Configuration and Installation, *to re-install the PCI-MXI-2 module.*

4. Replace the computer cover.

5. Turn on the computer. The computer should boot this time because the factory-default configuration is being used to initialize the PCI-MXI-2 module.

6. Run VXIedit to re-adjust the PCI-MXI-2 configuration. Refer to Chapter 6, *NI-VXI Configuration Utility*, for instructions on using this utility.

7. After saving the configuration, exit Windows and turn off the computer.

8. Remove the top cover or access port to the PCI bus.

9. Change switch 1 (FOV) on U17 to the OFF position.

10. Replace the computer cover.

11. Turn on the computer. If the computer does not boot with this configuration, you will have to repeat these steps, modifying your configuration until a final configuration is reached.

Common Questions

This appendix addresses common questions you may have about using the NI-VXI bus interface software on the PCI-MXI-2 platform.

How can I determine which version of the NI-VXI software I have installed?

The 32-bit driver components all have version resources. Under Windows 95 and Windows NT 4.0, you can find out version information by right-clicking on any component and selecting the **Properties** option. This will display a property sheet with a version tab. This tab has version information about the product (NI-VXI) and the component (NIVXINT.DLL, for example). You can also find out version information if you run the NI-VXI utility program VIC or VICtext. At the prompt type ver, and the utility will display the versions of VIC/VICtext and NI-VXI, and the latest PCI-MXI-2 board revision that this NI-VXI driver supports.

How can I determine the revision of the PCI-MXI-2 board that my NI-VXI software supports?

Running the NI-VXI utility program VICtext as described above will display the versions of VICtext and NI-VXI, and the hardware revision of the PCI-MXI-2 that the NI-VXI software supports.

How can I determine the serial number and hardware revision of the PCI-MXI-2 board?

Run the NI-VXI utility program VXIedit. Choose the **PCI-MXI-2 Configuration Editor**. The opening screen displays the serial number and hardware revision of the PCI-MXI-2 board.

How can I determine the serial number and hardware revision of the VXI-MXI-2 or VME-MXI-2?

Run the NI-VXI utility program VXIedit. Choose the **VXI/VME-MXI-2 Configuration Editor**. The opening screen displays the serial number and hardware revision of the VXI-MXI-2 or VME-MXI-2.

Which NI-VXI utility program must I use to configure the PCI-MXI-2?

Use the VXI Resource Editor program, either `VXIedit` or `VXItedit`, to configure the PCI-MXI-2. It is located in the `NIVXI` directory.

Which NI-VXI utility program must I use to initialize the PCI-MXI-2?

Use the hardware initialization program, `VXIinit`, to initialize the PCI-MXI-2. It is located in the `NIVXI` directory. `VXIinit` is required if you are running Windows 3.*x*. However, in Windows 95/NT, the board is automatically initialized at system startup, so you do not need to run `VXIinit` unless `RESMAN` fails.

What does `VXIinit` do?

The `VXIinit` program configures the components of the PCI-MXI-2 interface that are not configured by the EEPROM, such as resetting the DMA channels, manually placing the driver windows (if required), and, most importantly, setting up the MITE windows for direct access. In other words, it initializes the PCI-MXI-2. `VXIinit` also reads the current configuration and displays that to the user.

Which NI-VXI utility program must I use to perform startup Resource Manager operations?

Use the `RESMAN` program to perform startup Resource Manager operations. It is located in the `NIVXI` directory. `RESMAN` uses the settings in the Configuration Editor of `VXIedit` or `VXItedit`. It initializes your VXI/VMEbus system and stores the information that it collects to the `RESMAN.TBL` file in the `TBL` subdirectory of the `NIVXI` directory.

What can I do to make sure that my system is up and running?

The fastest method for testing the system is to run `RESMAN`. This program attempts to access memory in the upper A16 address space of each device in the system. If `RESMAN` does not report any problems, the VXI/MXI communication system is operational.

To test individual devices, you can use the `VIC` or `VICtext` program to interactively issue NI-VXI functions. You can use the `VXIin()` and `VXIout()` functions or the `VXIinReg()` and `VXIoutReg()` functions to test register-based devices by programming their registers. If you have any message-based devices, you can send and receive messages

with the `WSwrt()` and `WSrd()` functions. Notice that `VXIinReg()` and `VXIoutReg()` are for VXI devices only.

Finally, if you are using LabVIEW or LabWindows/CVI and you have instrument drivers for the devices in your chassis, you can use the interactive features of these programs to quickly test the functionality of the devices.

What should I do if I get a `Configuration EEPROM is Invalid` message?

There are several reasons why you might get the **`Configuration EEPROM is Invalid`** message. If you turned off the computer while the configuration update process was still in progress, or if you are using an old version of `VXIedit` (such as version 1.01 of the NI-VXI software for DOS/Windows for the PCI-MXI-2), the board will function normally except when using `VXIedit`. To correct these problems, reboot the computer with the Factory Override (FOV) switch set as described in Appendix C, *EEPROM Configuration* and update the configuration, or load the configuration from file.

Two other reasons you might receive this error message are that the board might have an incorrect base address assigned for the driver window, or there may be a conflict with another adapter or memory management software. The next two questions deal specifically with incorrect driver window base and memory conflict problems.

How do I fix an incorrect driver window base?

Run `VXIinit` and check the driver and user window base and size values near the top of the display. The driver window size must be 32 KB (0x8000 bytes). The user window should be either the default size of 64 KB (0x100000 bytes) or the size you have set up in `VXIedit` if you have changed it. Typically, the base addresses will be either in the 0xC8000 to 0xE800 region, immediately following the end of RAM installed in your computer (for example, if you have 16 MB of RAM the address will be of the form 0x10*hhhhhh*) or at the end of the address space (the address will be of the form 0xF*hhhhhhh*, where *h* is a hexadecimal digit).

To correct the problem, reboot the computer with the Factory Override (FOV) switch set as described in Appendix C, *EEPROM Configuration*. If the problem disappears, use `VXIedit` to correct the driver and user window placement. If the problem persists, it may indicate that your BIOS is not configuring the board correctly. Contact your computer manufacturer

for a BIOS update. In Windows 95, you can also use the Device Manager to check if there are any problems with the board by using the **Resources** page in the PCI-MXI-2 properties dialog. Finally, you can try a different PCI slot, because the BIOS on some computers may assign resources differently depending on the location of the device relative to a PCI bus bridge.

How do I resolve a memory conflict?

A memory conflict is the most likely cause for the `Configuration EEPROM is Invalid` error. In most cases the configuration EEPROM is actually valid, but `VXIedit` cannot read its contents because of the conflict.

If you are using Windows 3.*x*, you should place the board above the 1 MB boundary. To do so, reboot the computer with the Factory Override (FOV) switch set as described in Appendix C, *EEPROM Configuration*. Run `VXIedit` and in the PCI-MXI-2 **Bus Configuration Editor** make sure that the **Place below 1 MB** option is *not* checked. Update the configuration and reboot, either by powering down your system or by using the reset button. You should not use the <Control-Alt-Del> keyboard command because on many systems it does not reset the PCI boards.

If you are using DOS, you can run `VXIedit` with the *-o* command line option to manually select the memory address used by the PCI-MXI-2 while `VXIedit` is running. `VXIedit` will prompt you to select a driver window base every time you try to enter the **PCI-MXI-2 Configuration Editor**. You can try several different options until you find a region that does not cause a conflict.

Usually SCSI and network adapters need the upper memory and can conflict with the PCI-MXI-2. Determine the memory regions used by these devices and configure the PCI-MXI-2 to use a different region. Some BIOSes use the 0xE0000 to 0xEFFFF region to shadow PCI extensions, making the region unavailable to other devices. If your computer's BIOS uses this region, you may be able to turn off the BIOS shadowing feature. Refer to your computer documentation for instructions on how to accomplish this. If you cannot turn off the BIOS shadowing, select a different memory region for the PCI-MXI-2.

Often the memory conflict is not with another device, but rather with the memory management software such as EMM386, QEMM, or 386MAX. These products enable your computer to use parts of the upper memory, typically reserved for hardware adapters, to load device drivers and TSR. Because the PCI-MXI-2 needs that memory to

operate, you must instruct your memory management software to exclude the regions used by the board. You can either omit the line that loads the memory manager or add a flag that prohibits the memory manager from using the requested range of upper memory. The most common memory manager is EMM386 because it ships standard with Microsoft DOS and Windows 3.*x*. In many cases, EMM386 is not needed for your configuration and can be omitted by adding REM to the beginning of that line. If any of your applications do need it, you can add the following flag to the end of the line that loads EMM386:

x=*start-end*

where *start* is the base address of the window and *end* is the base plus the size minus 1. Notice that both the base and the size are in paragraphs (16-byte units). For example, if your driver window is located at 0xC8000 and your 64 KB user window is at 0xE0000, you should specify the following flags at the end of the EMM386 line:

x=0xC800-0xCFFF   x=0xE000-0xEFFF

What should I do if RESMAN hangs?

1. If you are using MXI-2, ensure that the MXI-2 cable is plugged in and that the end labeled *Connect this end to the device closest to the MXIbus System Controller* is connected to the MXIbus system controller (by default the PCI-MXI-2). Because the MXI-2 cables are polarized, it matters which end is connected to which device.

2. Check for bent or broken pins on the MXI-2 connectors.

3. Make sure VXIinit and VXIedit work correctly. If not, refer to the previous question and answer topics that deal with incorrect driver window base and memory conflict problems.

4. If problems persist, run RESMAN with the VXI/VME-MXI-2 located in the leftmost slot of your chassis and no other instruments installed. If this works, try adding instruments until the problem occurs again. Contact National Instruments for further assistance.

What do the LEDs on the front of the VXI-MXI-2 or VME-MXI-2 mean?

The **SYSFAIL** LED shows the state of the VXIbus/VMEbus SYSFAIL line. This line is asserted whenever any device in the chassis has not yet passed its self test, if it has failed its self test, or if it has detected a failure after originally passing its self test. The **MXI** LED indicates that the VXI-MXI-2 or VME-MXI-2 is acting as a slave to another device on the MXIbus, such as when the PCI-MXI-2 communicates with either

the VXI-MXI-2 or VME-MXI-2 or with another device in the chassis. The **VXI (VME)** LED, when lit, indicates that the VXI-MXI-2 or VME-MXI-2 is acting as a slave to another device in the VXI (VME) chassis, such as when a bus master inside the chassis wants to talk to either the VXI-MXI-2 or VME-MXI-2 or another device outside the chassis.

Are the PCI-MXI-2 and the VXI-MXI-2 two devices or one with respect to the VXIbus?

Both the PCI-MXI-2 and the VXI-MXI-2 are unique VXIbus devices with their own logical addresses. However, the MXIbus allows the PCI computer to behave as if it is inside the chassis with the VXI-MXI-2 by transparently converting PCI bus cycles to MXIbus cycles to VXIbus cycles, and vice versa.

I have a system that requires rugged chassis and bulkhead cables. Can I still use MXIbus?

Yes, National Instruments sells MXIbus bulkhead cables. Contact National Instruments for further information.

What kind of signal is CLK10 and what kind of signal do I need for an external CLK10?

CLK10 is a differential ECL signal on the VXIbus backplane. However, the oscillator for the VXI-MXI-2 and the EXTCLK input from the front panel use TTL. Therefore, you need to supply a TTL level signal for EXTCLK and our voltage converters will convert the signal to differential ECL.

CLK10 is not applicable to VME.

What is the accuracy of the CLK10 signal?

The CLK10 generated by the VXI-MXI-2 is 100 ppm (0.01%) as per the VXIbus specification. If you need a more accurate CLK10 signal, you can use the EXTCLK input at the front of the VXI-MXI-2.

CLK10 is not applicable to VME.

Customer Communication

For your convenience, this appendix contains forms to help you gather the information necessary to help us solve your technical problems and a form you can use to comment on the product documentation. When you contact us, we need the information on the Technical Support Form and the configuration form, if your manual contains one, about your system configuration to answer your questions as quickly as possible.

National Instruments has technical assistance through electronic, fax, and telephone systems to quickly provide the information you need. Our electronic services include a bulletin board service, an FTP site, a FaxBack system, and e-mail support. If you have a hardware or software problem, first try the electronic support systems. If the information available on these systems does not answer your questions, we offer fax and telephone support through our technical support centers, which are staffed by applications engineers.

Electronic Services

Bulletin Board Support

National Instruments has BBS and FTP sites dedicated for 24-hour support with a collection of files and documents to answer most common customer questions. From these sites, you can also download the latest instrument drivers, updates, and example programs. For recorded instructions on how to use the bulletin board and FTP services and for BBS automated information, call (512) 795-6990. You can access these services at:

United States: (512) 794-5422
   Up to 14,400 baud, 8 data bits, 1 stop bit, no parity

United Kingdom: 01635 551422
   Up to 9,600 baud, 8 data bits, 1 stop bit, no parity

France: 01 48 65 15 59
   Up to 9,600 baud, 8 data bits, 1 stop bit, no parity

FTP Support

To access our FTP site, log on to our Internet host, `ftp.natinst.com`, as `anonymous` and use your Internet address, such as `joesmith@anywhere.com`, as your password. The support files and documents are located in the `/support` directories.

Fax-On-Demand Support

FaxBack is a 24-hour information retrieval system containing a library of documents on a wide range of technical information. You can access FaxBack from a touch-tone telephone at (512) 418-1111.

E-Mail Support (currently U.S. only)

You can submit technical support questions to the applications engineering team through e-mail at the Internet address listed below. Remember to include your name, address, and phone number so we can contact you with solutions and suggestions.

`support@natinst.com`

Fax and Telephone Support

National Instruments has branch offices all over the world. Use the list below to find the technical support number for your country. If there is no National Instruments office in your country, contact the source from which you purchased your software to obtain support.
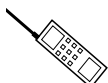
	Telephone	Fax
Australia	02 9874 4100	02 9874 4455
Austria	0662 45 79 90 0	0662 45 79 90 19
Belgium	02 757 00 20	02 757 03 11
Canada (Ontario)	905 785 0085	905 785 0086
Canada (Quebec)	514 694 8521	514 694 4399
Denmark	45 76 26 00	45 76 26 02
Finland	09 527 2321	09 502 2930
France	01 48 14 24 24	01 48 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 (include version number) _____

Clock 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 reproduce the problem:_____

_____

_____

_____

_____

_____

Hardware and Software Configuration Form

Record the settings and revisions of your hardware and software on the line to the right of each item. Complete a new copy of this form each time you revise your software or hardware configuration, and use 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.

National Instruments Products

NI-VXI Software Version Number _____

Using `VXIedit` or `VXItedit`? _____

PCI-MXI-2

Hardware Revision Number _____

Switch U17 Settings _____

DRAM SIMMs Installed _____

PCI-MXI-2 Configuration Editor Settings (VXIedit)

Logical Address _____

Device Type _____

Address Space _____

VXI Shared RAM Size _____

Shared RAM Pool _____

Byte Swapping for Lower Half Window _____

Memory Select for Lower Half Window _____

Byte Swapping for Upper Half Window _____

Memory Select for Upper Half Window _____

Mapping Scheme for Lower and Upper Half Windows of VXI Shared RAM _____

_____

Resource Manager Delay _____

Default Controller (LA -1) _____

System IRQ Level _____

Servant Area Size _____

Number of Handlers _____

Number of Interrupters _____

Protocol Register _____

Read Protocol Response _____

MXI System Controller _____

MXI BTO Value _____

MXI CLK10 _____

MXI Transfer Limit _____

Synchronous MXI _____

MXI-2 Auto Retry _____

A24/A32 Write Posting _____

User Window Size _____

User Window Base _____

User Window Below 1 MB _____

Driver Window Size _____

Driver Window Base _____

Driver Window Below 1 MB _____

Expansion ROM _____

VXI/VME-MXI-2

Using VXI-MXI-2 or VME-MXI-2? _____

Hardware Revision Number _____

VXI-MXI-2 Hardware Configuration

Slot Location _____

VXIbus Logical Address Switch Setting (U43) _____

VXIbus Slot 0/Non-Slot 0 (W2) _____

VXIbus Local Bus (S8, S9) _____

VXIbus CLK10 Routing (W3) _____

SMB CLK10 (S3, S4, S5) _____

Receiving or Driving MXIbus CLK10 (S7) _____

Trigger Input Termination (S2) _____

MXIbus Termination (U35 switches 1 and 2) _____

EEPROM Operation (U35 switches 3 and 4) _____

Onboard DRAM SIMM Size (S6) _____

DRAM SIMMs Installed _____

VME-MXI-2 Hardware Configuration

Slot Location _____

VMEbus A16 Base Address (U20) _____

VME-MXI-2 Intermodule Signaling (W2) _____

MXIbus Termination (U21 switches 3 and 4) _____

EEPROM Operation (U21 switches 3 and 4) _____

Onboard DRAM SIMM Size (S2) _____

DRAM SIMMs Installed _____

VXI/VME-MXI-2 Configuration Editor Settings (VXIedit)

Logical Address _____

LA Selection _____

Address Space _____

Requested Memory _____

A16 Write Posting _____

A24/A32 Write Posting _____

Interlocked or Normal Mode _____

VXI/VME System Controller _____

VXI/VME Bus Timeout Value _____

Auto Retry for Cycles from VXI/VMEbus to MXIbus _____

Transfer Limit on VXI/VMEbus _____

VXI/VME Arbiter Type _____

VXI/VME Request Level _____

VXI/VME Fair Requester _____

VXI/VME Arbiter Timeout _____

MXI System Controller _____

MXI Bus Timeout Value _____

Auto Retry for Cycles from MXIbus to VXI/VMEbus _____

Transfer Limit on MXIbus _____

MXI Parity Checking _____

MXI Fair Requester _____

MXI CLK10 Direction (VXI-MXI-2 only) _____

Other Products

Computer Make and Model _____

Mainframe Make and Model _____

Microprocessor _____

Clock Frequency _____

Type of Video Board Installed _____

Operating System _____

Operating System Version _____

Operating System Mode _____

Programming Language _____

Programming Language Version _____

Other Boards in System _____

Base I/O Address of Other Boards _____

DMA Channels of Other Boards _____

Interrupt Level of Other Boards _____

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 VXI/VME-PCI8000 Series and the NI-VXI™ Software for Microsoft Operating Systems*

Edition Date: December 1996

Part Number: 320961C-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 (___)_____   Fax (___) _____

Mail to: Technical Publications   **Fax to:** Technical Publications
National Instruments Corporation   National Instruments Corporation
6504 Bridge Point Parkway   (512) 794-5678
Austin, TX 78730-5039

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

Symbols

°	degrees
Ω	ohms
%	percent

A

A   amperes

A16 space   VXIbus address space equivalent to the VME 64 KB short address space. In VXI, the upper 16 KB of A16 space is allocated for use by VXI devices configuration registers. This 16 KB region is referred to as VXI configuration space.

A24 space   VXIbus address space equivalent to the VME 16 MB *standard* address space.

A32 space	VXIbus address space equivalent to the VME 4 GB *extended* address space.
ACFAIL	A VMEbus backplane signal that is asserted when a power failure has occurred (either AC line source or power supply malfunction), or if it is necessary to disable the power supply (such as for a high temperature condition).
address	Character code that identifies a specific location (or series of locations) in memory.
address modifier	One of six signals in the VMEbus specification used by VMEbus masters to indicate the address space in which a data transfer is to take place.
address space	A set of 2^n memory locations differentiated from other such sets in VXI/VMEbus systems by six addressing lines known as address modifiers. n is the number of address lines required to uniquely specify a byte location in a given space. Valid numbers for n are 16, 24, and 32. In VME/VXI, because there are six address modifiers, there are 64 possible address spaces.
address window	A portion of address space that can be accessed from the application program.
ANSI	American National Standards Institute
arbitration	A process in which a potential bus master gains control over a particular bus.
asynchronous	Not synchronized; not controlled by time signals.

B

B	bytes
backplane	An assembly, typically a printed circuit board, with 96-pin connectors and signal paths that bus the connector pins. A C-size VXIbus system will have two sets of bused connectors called J1 and J2. A D-size VXIbus system will have three sets of bused connectors called J1, J2, and J3.
BERR*	Bus error signal

binary	A numbering system with a base of 2.
BIOS	Basic Input/Output System. BIOS functions are the fundamental level of any PC or compatible computer. BIOS functions embody the basic operations needed for successful use of the computer's hardware resources.
block-mode transfer	An uninterrupted transfer of data elements in which the master sources only the first address at the beginning of the cycle. The slave is then responsible for incrementing the address on subsequent transfers so that the next element is transferred to or from the proper storage location. In VME, the data transfer may have no more than 256 elements; MXI does not have this restriction.
BTO unit	Bus Timeout Unit; a functional module that times the duration of each data transfer and terminates the cycle if the duration is excessive. Without the termination capability of this module, a bus master attempt to access a nonexistent slave could result in an indefinitely long wait for a slave response.
bus master	A device that is capable of requesting the Data Transfer Bus (DTB) for the purpose of accessing a slave device.

C

C	Celsius
CLK10	A 10 MHz, ± 100 ppm, individually buffered (to each module slot), differential ECL system clock that is sourced from Slot 0 of a VXIbus mainframe and distributed to Slots 1 through 12 on P2. It is distributed to each slot as a single-source, single-destination signal with a matched delay of under 8 ns.
CMOS	Complementary Metal Oxide Semiconductor; a process used in making chips.
Commander	A message-based device which is also a bus master and can control one or more Servants.
configuration registers	A set of registers through which the system can identify a module device type, model, manufacturer, address space, and memory requirements. In order to support automatic system and memory

configuration, the VXIbus specification requires that all VXIbus
devices have a set of such registers.

D

daisy-chain	A method of propagating signals along a bus, in which the devices are prioritized on the basis of their position on the bus.
Data Transfer Bus	DTB; one of four buses on the VMEbus backplane. The DTB is used by a bus master to transfer binary data between itself and a slave device.
DIP	Dual Inline Package
DMA	Direct Memory Access; a method by which data is transferred between devices and internal memory without intervention of the central processing unit.
DRAM	Dynamic RAM
driver window	A region of PCI address space that is decoded by the PCI-MXI-2 for use by the NI-VXI software.
DTACK*	Data Acknowledge signal
DTB	See *Data Transfer Bus*.
dynamic configuration	A method of automatically assigning logical addresses to VXIbus devices at system startup or other configuration times.
dynamically configured device	A device that has its logical address assigned by the Resource Manager. A VXI device initially responds at Logical Address 255 when its MODID line is asserted. A MXIbus device responds at Logical Address 255 during a priority select cycle. The Resource Manager subsequently assigns it a new logical address, which the device responds to until powered down.

E

ECL	Emitter-Coupled Logic
EEPROM	Electronically Erasable Programmable Read Only Memory

embedded controller	An intelligent CPU (controller) interface plugged directly into the VXI backplane, giving it direct access to the VXIbus. It must have all of its required VXI interface capabilities built in.
EMC	Electromechanical Compliance
EMI	Electromagnetic Interference
expansion ROM	An onboard EEPROM that may contain device-specific initialization and system boot functionality.
external controller	In this configuration, a plug-in interface board in a computer is connected to the VXI mainframe via one or more VXIbus extended controllers. The computer then exerts overall control over VXIbus system operations.

F

fair requester	A MXIbus master that will not arbitrate for the MXIbus after releasing it until it detects the bus request signal inactive. This ensures that all requesting devices will be granted use of the bus.

H

hex	Hexadecimal; the numbering system with base 16, using the digits 0 to 9 and letters A to F.
Hz	hertz; cycles per second.

I

IC	Integrated Circuit
IEEE	Institute of Electrical and Electronics Engineers
in.	inches
I/O	input/output; the techniques, media, and devices used to achieve communication between machines and users.
interrupt	A means for a device to request service from another device.

interrupt handler A VMEbus functional module that detects interrupt requests generated by Interrupters and responds to those requests by requesting status and identify information.

interrupt level The relative priority at which a device can interrupt.

IRQ* Interrupt signal

K

KB Kilobytes of memory

L

LED Light Emitting Diode

logical address An 8-bit number that uniquely identifies each VXIbus device in a system. It defines the A16 register address of a device, and indicates Commander and Servant relationships.

M

m meters

master A functional part of a MXI/VME/VXIbus device that initiates data transfers on the backplane. A transfer can be either a read or a write.

master-mode operation A device is in master mode if it is performing a bus cycle which it initiated.

MB Megabytes of memory

MBLT Eight-byte block transfers in which both the Address bus and the Data bus are used to transfer data.

message-based device An intelligent device that implements the defined VXIbus registers and communication protocols. These devices are able to use Word Serial Protocol to communicate with one another through communication registers.

MITE	A National Instruments custom ASIC, a sophisticated dual-channel DMA controller that incorporates the Synchronous MXI and VME64 protocols to achieve high-performance block transfer rates.
MODID	Module Identification lines
MTBF	Mean Time Between Failure
MXI-2	The second generation of the National Instruments MXIbus product line. MXI-2 expands the number of signals on a standard MXIbus cable by including VXI triggers, all VXI interrupts, CLK10, SYSFAIL*, SYSRESET*, and ACFAIL*.
MXIbus	Multisystem eXtension Interface Bus; a high-performance communication link that interconnects devices using round, flexible cables.
MXIbus System Controller	A functional module that has arbiter, daisy-chain driver, and MXIbus cycle timeout responsibility. Always the first device in the MXIbus daisy-chain.

N

NI-VXI	The National Instruments bus interface software for VME/VXIbus systems.
Non-Slot 0 device	A device configured for installation in any slot in a VXIbus mainframe other than Slot 0. Installing such a device into Slot 0 can damage the device, the VXIbus backplane, or both.

O

Onboard RAM	The optional RAM installed into the SIMM slots of the PCI-MXI-2 board or VXI/VME-MXI-2 module.

P

PCI	Peripheral Component Interconnect. The PCI bus is a high-performance 32-bit or 64-bit bus with multiplexed address and data lines.

propagation	The transmission of signal through a computer system.

R

register-based device	A Servant-only device that supports VXIbus configuration registers. Register-based devices are typically controlled by message-based devices via device-dependent register reads and writes.
retry	An acknowledge by a destination that signifies that the cycle did not complete and should be repeated.
RESMAN	The name of the National Instruments Resource Manager in NI-VXI bus interface software. See *Resource Manager*.
Resource Manager	A message-based Commander located at Logical Address 0, which provides configuration management services such as address map configuration, Commander and Servant mappings, and self-test and diagnostic management.

S

s	seconds
Servant	A device controlled by a Commander; there are message-based and register-based Servants.
Shared Memory Protocol	A communication protocol that uses a block of memory that is accessible to both a client and a server. The memory block operates as a message buffer for communications.
SIMM	Single In-line Memory Module
slave	A functional part of a MXI/VME/VXIbus device that detects data transfer cycles initiated by a VMEbus master and responds to the transfers when the address specifies one of the device's registers.
slave-mode operation	A device is in slave mode it if is responding to a bus cycle.
Slot 0 device	A device configured for installation in Slot 0 of a VXIbus mainframe. This device is unique in the VXIbus system in that it performs the VMEbus System Controller functions, including clock sourcing and arbitration for data transfers across the

backplane. Installing such a device into any other slot can damage the device, the VXIbus backplane, or both.

statically configured device	A device whose logical address cannot be set through software; that is, it is not dynamically configurable.
SYSFAIL	A VMEbus signal that is used by a device to indicate an internal failure. A failed device asserts this line. In VXI, a device that fails also clears its PASSed bit in its Status register.
SYSRESET	A VMEbus signal that is used by a device to indicate a system reset or power-up condition.
System RAM	RAM installed on your personal computer and used by the operating system, as contrasted with onboard RAM, which is installed on the PCI-MXI-2 or VXI/VME-MXI-2.

T

trigger	Either TTL or ECL lines used for intermodule communication.
TTL	Transistor-Transistor Logic

U

user window	A region of PCI address space reserved by the PCI-MXI-2 for use via the NI-VXI low-level function calls. MapVXIAddress() uses this address space to allocate regions for use by the VXIpeek() and VXIpoke() macros.

V

V	volts
VDC	volts direct current
VIC or VICtext	VXI Interactive Control Program, a part of the NI-VXI bus interface software package. Used to program VXI devices, and develop and debug VXI application programs.
VME	Versa Module Eurocard or IEEE 1014

VMEbus System Controller	A device configured for installation in Slot 0 of a VXIbus mainframe or Slot 1 of a VMEbus chassis. This device is unique in the VMEbus system in that it performs the VMEbus System Controller functions, including clock sourcing and arbitration for data transfers across the backplane. Installing such a device into any other slot can damage the device, the VMEbus/VXIbus backplane, or both.
VXIbus	VMEbus Extensions for Instrumentation
VXIedit or VXItedit	VXI Resource Editor program, a part of the NI-VXI bus interface software package. Used to configure the system, edit the manufacturer name and ID numbers, edit the model names of VXI and non-VXI devices in the system, as well as the system interrupt configuration information, and display the system configuration information generated by the Resource Manager.
VXIinit	A program in the NI-VXI bus interface software package that initializes the board interrupts, shared RAM, VXI register configurations, and bus configurations.

W

Word Serial Protocol	The simplest required communication protocol supported by message-based devices in a VXIbus system. It utilizes the A16 communication registers to transfer data using a simple polling handshake method.
write posting	A mechanism that signifies that a device will immediately give a successful acknowledge to a write transfer and place the transfer in a local buffer. The device can then independently complete the write cycle to the destination.

Numbers

1 MB AddressBoundary, 6-19

A

A16 base address, VMEbus, 4-3 to 4-4
A16 write posting, VXI/VME-MXI-2, 6-23 to
6-24
A24/A32 write posting
PCI-MXI-2, 6-18
VXI/VME-MXI-2, 6-23 to 6-24
address space configuration
PCI-MXI-2, 6-6 to 6-7
VXI/VME-MXI-2, 6-23
addressing mode for PCI-MXI-2, setting, 6-6 to
6-7
arbiter timeout, setting, 6-29
arbiter type, setting, 6-28
arbitration mode, interlocked, 6-24 to 6-25
AUTOEXEC.BAT file, modifying, 5-3 to 5-4
automatic retry feature
MXI, 6-31
MXI-2, 6-17 to 6-18
VXI/VME, 6-27

B

BINARY_COMPATIBLE symbol, 7-7
BTO. *See* Bus Timeout (BTO) value.
bulkhead cables, D-6
bulletin board support, E-1
Bus Configuration Editor. *See* PCI-MXI-2 Bus
Configuration Editor.
Bus Timeout (BTO) value
MXIbus, 6-16, 6-29 to 6-30
VXI/VME, 6-26

C

C programs
compiling, 7-6 to 7-7
sample programs, 7-6
symbols, 7-6 to 7-7
cables, bulkhead, D-6
CLK10 routing, VXIbus, 3-8 to 3-12
CLK10 generated from MXIbus
(figure), 3-9
CLK10 generated from onboard oscillator
(figure), 3-9
CLK10 generated from SMB (figure), 3-9
drive inverted external CLK SMB
(figure), 3-11
drive non-inverted external CLK SMB
(figure), 3-11
receive external CLK SMB (figure), 3-11
receive external CLK SMB with 50 ohm
termination (figure), 3-11
receiving or driving MXIbus CLK10
(figure), 3-12
CLK10 signal, MXIbus
common questions, D-6
setting, 6-16 to 6-17, 6-32
CloseVXIlibrary function, 7-5
common questions about NI-VXI software, D-1
to D-6
compiling C programs, 7-6 to 7-7
configuration
default settings, 1-12 to 1-17
PCI-MXI-2 board, 2-1 to 2-3
configuration EEPROM, 2-3
default settings
Bus Configuration Editor
(table), 1-14

Q

U

V