Getting Started with Your VXI/VME-AT2010 and the NI-VXI™ Software for Windows

bus

December 1994 Edition

Part Number 320888A-01

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

Branch Offices:
Australia (03) 879 9422, Austria (0662) 435986, Belgium 02/757.00.20, Canada (Ontario) (519) 622-9310,
Canada (Québec) (514) 694-8521, Denmark 45 76 26 00, Finland (90) 527 2321, France (1) 48 14 24 24,
Germany 089/741 31 30, Italy 02/48301892, Japan (03) 3788-1921, Mexico 95 800 010 0793,
Netherlands 03480-33466, Norway 32-84 84 00, Singapore 2265886, Spain (91) 640 0085, Sweden 08-730 49 70,
Switzerland 056/20 51 51, Taiwan 02 377 1200, U.K. 0635 523545

FCC/DOC Radio Frequency Interference Compliance

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

Federal Communications Commission

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

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

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

Canadian Department of Communications

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

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

Instructions to Users

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

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

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

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

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

- Reorient or relocate the receiver's antenna.

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

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

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

Contents

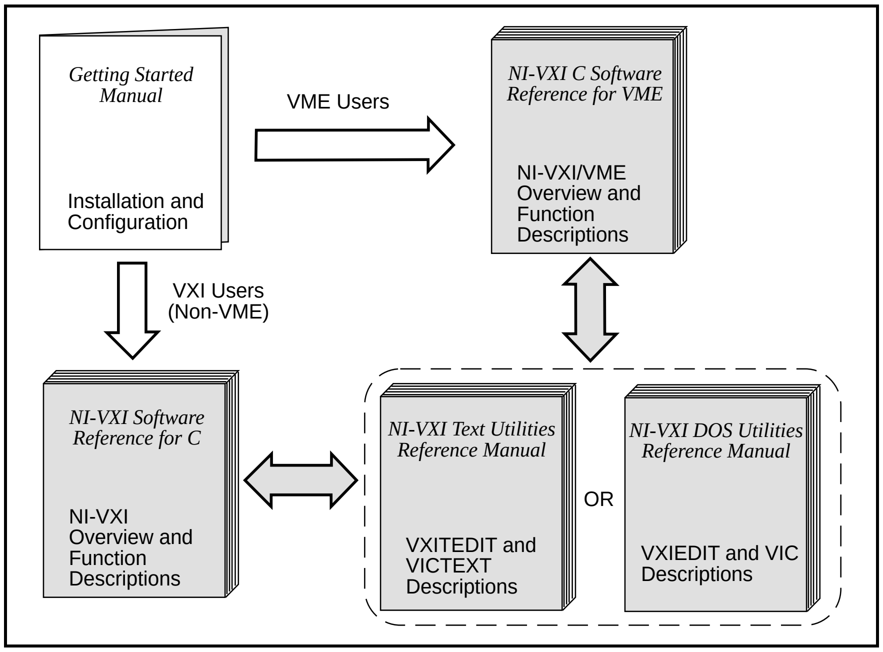
Figures

Tables

About This Manual

This manual contains instructions for installing and configuring the National Instruments VXI-AT2010 or VME-AT2010 interface kit for Windows. The VXI-AT2010 kit includes a VXI-MXI module that plugs into a VXI mainframe and links your PC AT to the VXIbus. The VME-AT2010 kit comes with a VME-MXI that plugs into a VME chassis and links your PC AT to the VMEbus. Both kits include the AT-MXI interface board, which links your PC AT to the MXIbus, and the NI-VXI bus interface software for Windows. This manual uses the term *VXI/VME-AT2010* when information applies to either kit.

How to Use the Documentation Set


Getting Started Manual
Installation and Configuration
VME Users
NI-VXI C Software Reference for VME
NI-VXI/VME Overview and Function Descriptions
VXI Users (Non-VME)
NI-VXI Software Reference for C
NI-VXI Overview and Function Descriptions
NI-VXI Text Utilities Reference Manual
VXITEDIT and VICTEXT Descriptions
OR
NI-VXI DOS Utilities Reference Manual
VXIEDIT and VIC Descriptions

Begin by reading this getting started manual to guide you through the installation and configuration of the hardware and software. Complete the hardware installation and configuration first, then install the software.

Your kit also includes a software reference manual. If you are using the VXI-AT2010 interface kit, you received the *NI-VXI Software Reference Manual for C*. If you are using the VME-AT2010 interface kit, you received the *NI-VXI C Software Reference Manual for VME.* Use these software reference manuals to learn the basics of VXI or VME, and to study the purpose and syntax of each function.

Refer to the *NI-VXI Text Utilities Reference Manual* to learn more about the `VXITEDIT` and `VICTEXT` programs.

Refer to the *NI-VXI DOS Utilities Reference Manual* to learn more about the `VXIEDIT` and `VIC` programs.

Organization of This Manual

Getting Started with Your VXI-AT2010 and the NI-VXI Software for Windows is organized as follows:

- Chapter 1, *Introduction*, describes the VXI/VME-AT2010 interface kit, lists what you need to get started, lists optional equipment and software, and includes a brief description of the hardware and software.
- Chapter 2, *AT-MXI Configuration and Installation*, contains the instructions to configure and install the AT-MXI interface board.
- Chapter 3, *VXI-MXI Configuration and Installation*, contains the instructions to configure and install the VXI-MXI interface module.
- Chapter 4, *VME-MXI Configuration and Installation*, contains the instructions to configure and install the VME-MXI interface module.
- Chapter 5, *NI-VXI Software Installation and Configuration*, contains instructions for installing and configuring the NI-VXI software for Windows, and contains a description of the `VXIEDIT` configuration editor.
- Chapter 6, *Using the NI-VXI Software with Windows*, 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 AT-MXI, VXI-MXI, and VME-MXI such as physical dimensions and power requirements.
- Appendix B, *NI-VXI Software Overview*, describes the programs and files located on the NI-VXI distribution diskettes.
- Appendix C, *Common Questions*, addresses common questions you may have about using the NI-VXI bus interface software on the AT-MXI platform. If you need to troubleshoot specific errors, refer to Appendix D, *Troubleshooting*.
- Appendix D, *Troubleshooting*, addresses specific problems you may encounter when using the NI-VXI bus interface software on the AT-MXI platform. This material covers issues with `VXIINIT`, `RESMAN`, mouse drivers, and networks.
- 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, and symbols.
- The *Index* contains an alphabetical list of key terms and topics used 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 menus, menu items, and dialog box options.
italic	Italic text denotes emphasis, a cross reference, or introduction to a key concept.
bold italic	Bold italic text denotes a note, caution, or warning.
`monospace`	Text in this font denotes characters that are to be literally input from the keyboard, the proper names of disk drives, paths, directories, programs, subprograms, device names, filenames and extensions, and for statements and comments taken from program code.
`bold monospace`	Bold text in this font denotes the messages and responses that the computer automatically prints to the screen.
VXI-AT2010	VXI-AT2010 refers to the interface kit that includes the VXI-MXI module. It is intended for VXI systems.
VME-AT2010	VME-AT2010 refers to the interface kit that includes the VME-MXI module. It is intended for VME systems.
VXI/VME-AT2010	VXI/VME-AT2010 is used in this manual when information applies to either interface kit.

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

Related Documentation

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

- *VME-MXI User Manual*, National Instruments Corporation
- *VXI-MXI User Manual*, National Instruments Corporation
- *IEEE Standard for a Versatile Backplane Bus: VMEbus*, ANSI/IEEE Standard 1014-1987
- *Multisystem Extension Interface Bus Specification*, Version 1.2 (available from National Instruments Corporation)
- VXI-1, *VXIbus System Specification*, Rev. 1.4, VXIbus Consortium
- VXI-6, *VXIbus Mainframe Extender Specification*, Rev. 1.0, VXIbus Consortium (available from National Instruments Corporation)

You might also find helpful information in the Windows manuals that came with your system.

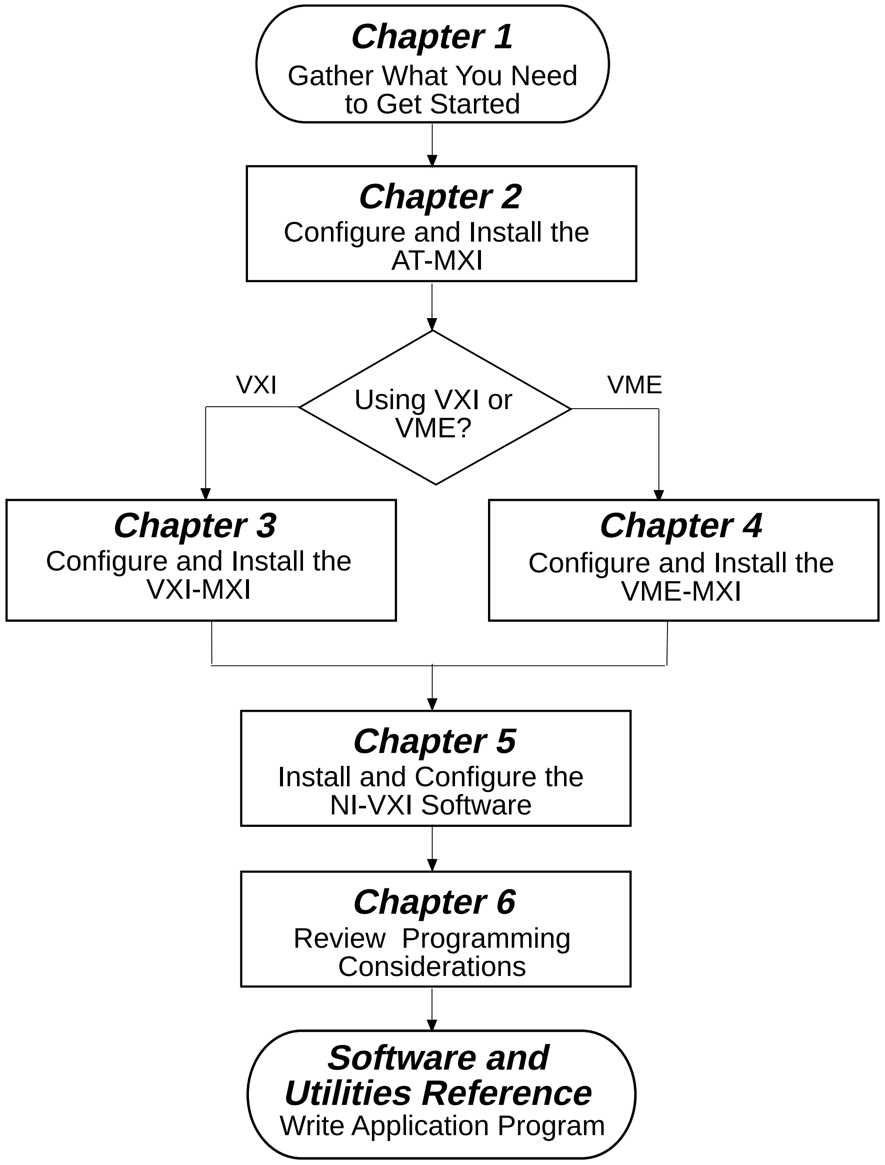
Customer Communication

National Instruments wants to receive your comments on our products and manuals. We are interested in the applications you develop using 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.

Chapter 1
Introduction

This chapter describes the VXI/VME-AT2010 interface kit, lists what you need to get started, lists optional equipment and software, and includes a brief description of the hardware and software.

How to Use This Manual



Chapter 1
Gather What You Need to Get Started
Chapter 2
Configure and Install the AT-MXI
Using VXI or VME?
VXI
VME
Chapter 3
Configure and Install the VXI-MXI
Chapter 4
Configure and Install the VME-MXI
Chapter 5
Install and Configure the NI-VXI Software
Chapter 6
Review Programming Considerations
Software and Utilities Reference
Write Application Program

VXI/VME-AT2010 Kit Overview

The VXI-AT2010 interface kit links any IBM Personal Computer AT or compatible computer (hereafter referred to as the PC AT) directly to the VXIbus. A PC AT equipped with a VXI-AT2010 can function as a VXI Commander and Resource Manager. The VXI-AT2010 makes the PC AT appear as though it were plugged directly into the VXI backplane as an embedded CPU VXI module.

The VME-AT2010 interface kit links any PC AT directly to the VMEbus. A PC AT equipped with a VME-AT2010 can function as a VME master and/or slave device. The VME-AT2010 makes the PC AT appear as though it were an embedded CPU plugged directly into the VME backplane, or as though it has internal VME slots for plug-in boards.

What You Need to Get Started

- ❑ AT-MXI interface board
- ❑ One of the following interface modules:

  Standard VXI-MXI interface module

  Enhanced VXI-MXI interface module with INTX option

  Standard VME-MXI interface module

  Enhanced VME-MXI interface module with INTX option
- ❑ NI-VXI distribution disks for the AT-MXI and Windows

  You received both 3.5 in. disks and 5.25 in. disks in your kit.
- ❑ Windows version 3.1 or higher installed on your computer
- ❑ 2 m Type M1 MXIbus cable

Note: ***The 2 m Type M1 MXIbus cable is included in your kit unless you specified otherwise in your order. You may have ordered your kit without this cable so that you could order a different type or length of MXIbus cable. Refer to the*** *Optional Equipment* ***section.***

Optional Equipment

Contact National Instruments to order any of the following optional equipment.

- VXI-MXI Standard VXIbus Mainframe Extender
- VXI-MXI Enhanced VXIbus Mainframe Extender
- VME-MXI Standard VMEbus Chassis Extender
- VME-MXI Enhanced VMEbus Chassis Extender
- MXIbus Connector Extender
- MXIbus Terminating Pac (external)
- Type M1 MXIbus cable (straight-point to straight-point connectors)
- Type M2 MXIbus cable (straight-point to right-angle daisy-chain connectors)
- Type M3 MXIbus cable (right-angle-point to right-angle daisy-chain connectors)
- Type M4 MXIbus cable (straight-point to reverse-right-angle daisy-chain connectors)
- Type M5 MXIbus cable (right-angle-point to reverse-right-angle daisy-chain connectors)
- Type M6 MXIbus cable (right-angle-point to reverse-right-angle daisy-chain connectors)
- Type MB1 MXIbus Bulkhead cable (right-angle point to wall-mount Bulkhead exit)
- Type MB2 MXIbus Bulkhead cable (straight Bulkhead exit to straight Bulkhead entry)
- Type MB3 MXIbus Bulkhead cable (wall-mount Bulkhead entry to right-angle daisy-chain)
- Type MB4 MXIbus Bulkhead cable (right-angle point to straight Bulkhead entry)
- Type MB5 MXIbus Bulkhead cable (right-angle daisy-chain to straight Bulkhead)

The Type M1, M2, M3, M4, M5, and M6 MXIbus cables are available in 1 m, 2 m, 4 m, 8 m, and 20 m lengths. The Type MB1, MB2, MB3, MB4, and MB5 MXIbus Bulkhead cables are available in 2 m and other lengths.

Hardware Description

The AT-MXI is a full-size, 16-bit circuit board that plugs into a PC AT/EISA slot in your computer. It links your PC AT directly to the MXIbus by using address mapping to translate bus cycles on the PC AT bus to the MXIbus and vice versa. Because the AT-MXI uses the same communication register set that other VXIbus message-based devices use, other MXIbus devices can view the AT-MXI as a VXIbus device. The AT-MXI is also able to function as the MXIbus System Controller and can terminate the MXIbus signals directly on the AT-MXI board.

The VXI-MXI 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 VXIbus chassis. It uses address mapping to convert MXIbus bus cycles into VXIbus bus cycles and vice versa. By connecting to the AT-MXI board, it links your PC AT to the VXIbus.

The VME-MXI interface is a single-slot, 6U VMEbus board, with optional VMEbus System Controller capability, that interfaces the VMEbus to the MXIbus (Multisystem Extension Interface bus). A VMEbus mainframe equipped with a VME-MXI can be transparently connected to other MXIbus devices such as other VMEbus mainframes, MXIbus instruments, or

MXIbus-equipped personal computers. The VME-MXI interface module uses address mapping to transparently translate bus cycles on the VMEbus to the MXIbus and vice versa.

Software Description

The NI-VXI bus interface software for the AT-MXI and Windows 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 the 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.

Optional Software

Your VXI/VME-AT2010 kit includes the NI-VXI bus interface software for Windows. In addition, you can order the LabVIEW or LabWindows®/CVI software from National Instruments. These programs match the modular virtual instrument capability of VXI and can reduce your VXIbus software development time. 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.

Both LabVIEW and LabWindows/CVI include all the tools needed for instrument control, data acquisition, analysis, and presentation. When you order LabVIEW or LabWindows/CVI, you also get more than 300 complete instrument drivers, which are modular, source-code programs that handle the communication with your instrument so that you do not have to learn the programming details.

The LabVIEW for Windows VXI Development System contains the following components:

- LabVIEW for Windows Full Development System
- LabVIEW for Windows VXI Library
- LabVIEW for Windows/Sun VXI Instrument Library

The LabWindows/CVI for Windows, VXI Development System contains the following components:

- LabWindows/CVI for Windows Full Development System
- LabWindows/CVI for Windows VXI Library
- LabWindows/CVI for Windows Instrument Library

Chapter 2
AT-MXI Configuration and Installation

This chapter contains the instructions to configure and install the AT-MXI interface board. When you complete these instructions, continue with either Chapter 3 or Chapter 4 to configure and install the VXI-MXI or the VME-MXI module, respectively.

Warning: ***Several components on your AT-MXI board can be damaged by electrostatic discharge. To avoid such damage in handling the board, touch the antistatic plastic package to a metal part of your computer chassis before removing the board from the package.***

Configure the AT-MXI

You can configure three options on the AT-MXI board:

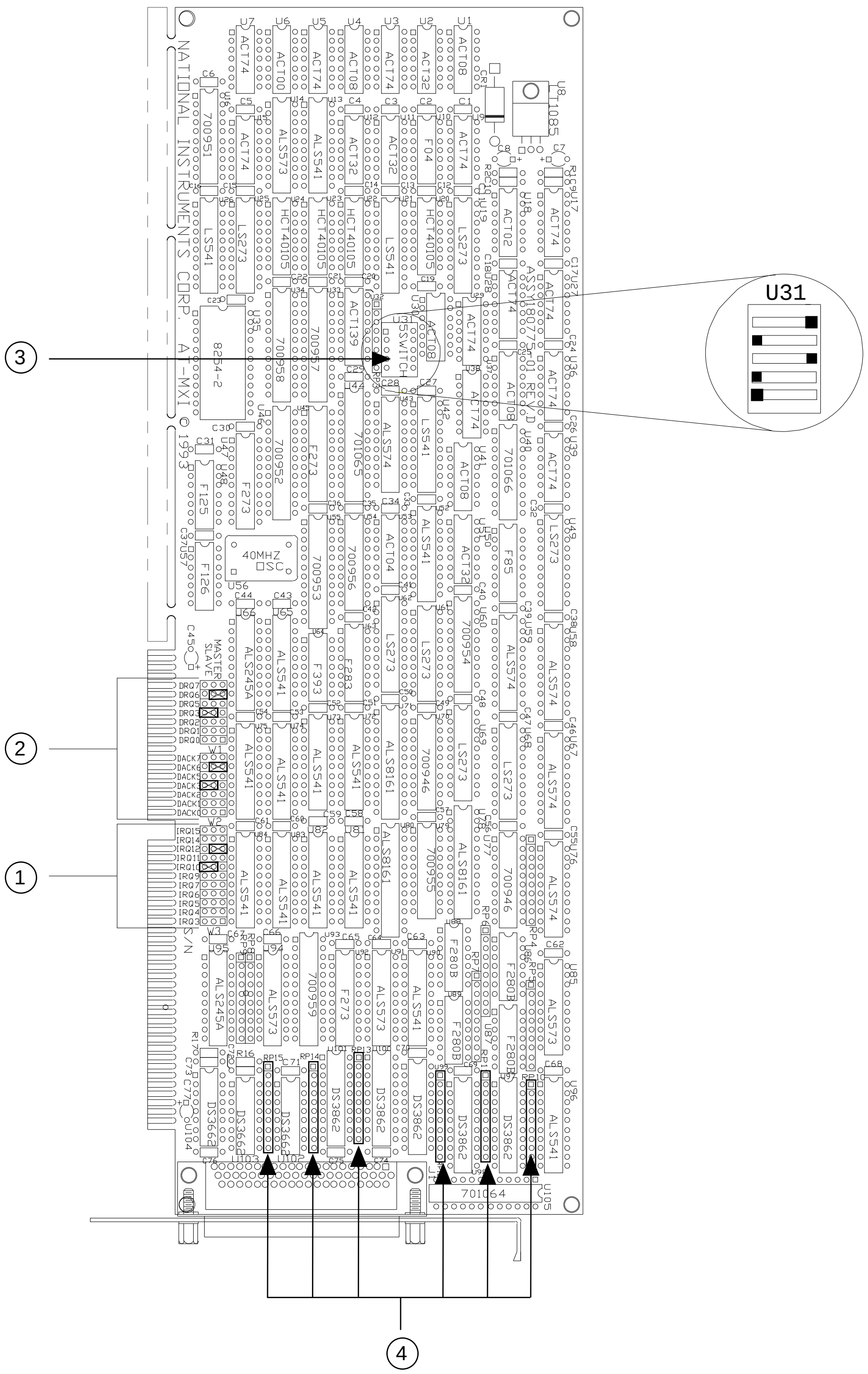
- Base I/O address
- Interrupt levels
- DMA channels

Table 2-1 shows the factory settings and optional configurations for the switches and jumpers on the AT-MXI.

Table 2-1. AT-MXI Default Settings and Optional Configurations

AT-MXI	Default	Optional
Base I/O Address (hex)	340	100 to 3E0, increments of 20 hex
Master DMA Channel	6	0, 1, 2, 3, 5, 6, 7, and Not Used
Slave DMA Channel	3	0, 1, 2, 3, 5, 6, 7, and Not Used
Board Interrupt Level	12	3, 4, 5, 6, 7, 9, 10, 11, 12, 14, and 15
MXIbus Interrupt Level	10	3, 4, 5, 6, 7, 9, 10, 11, 12, 14, 15, and Not Used

The factory-configured settings of the base I/O address, the interrupt levels, and the Direct Memory Access (DMA) channels are suitable for most computer systems. The following sections describe under what conditions you would be required to change the configuration switches and/or jumpers on the AT-MXI and how to make these changes. Figure 2-1 shows the location of the AT-MXI configuration jumpers and switches.



1	Interrupt Level(S)
2	DMA Channel(s)
3	Base I/O Address
4	MXIbus Terminators

Figure 2-1. AT-MXI Parts Locator Diagram

Base I/O Address Selection

The base I/O address of the AT-MXI is the starting address of the AT-MXI configuration registers in PC AT I/O space. The base I/O address is determined by the position of the five switches at location U31, as shown in Figure 2-1. The switches are set at the factory for a base I/O address of 340 hex. Because the AT-MXI requires 32 bytes of consecutive I/O space for its internal registers, the factory configuration uses the I/O address space in the range of 340 to 35F hex.

Note: ***Check to determine that this space is not already used by other I/O interfaces installed in your PC AT computer. If any equipment in your computer uses this I/O address space, you must change either the base I/O address of the AT-MXI or the I/O address space requirements of the other device. All PC AT devices must have a unique partition of the system's I/O address space. If you change the base I/O address of the AT-MXI, you must make a corresponding change to the NI-VXI configuration software as described in the*** *Configuring the Software* ***section of Chapter 5,*** *NI-VXI Software Installation and Configuration*.

Each switch in U31 (1 through 5) corresponds to one of the PC AT address lines (A5 through A9). The first switch (1) corresponds to address line A5, the next switch (2) corresponds to address line A6, and so on. The five least significant bits of the address (A4 through A0) are used by the AT-MXI to select the appropriate AT-MXI register and cannot be changed; therefore, bits A4 through A0 are always zeros when determining the base I/O address.

To change the base I/O address of the AT-MXI, press the side marked *OFF* to select a binary value of 1 for the corresponding address bit. Press the *ON* side of the switch to select a binary value of 0 for the corresponding address bit. Refer to Table 2-2.

Figure 2-2 shows two possible switch settings for the base I/O address.

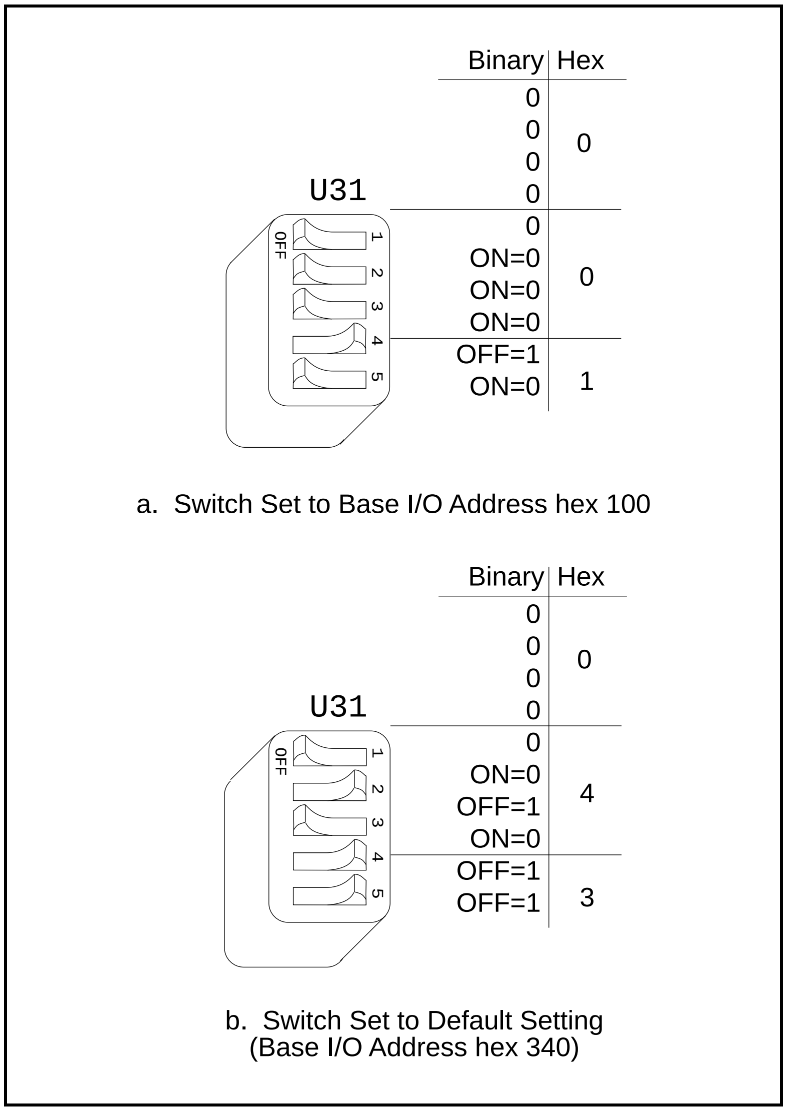


Figure 2-2. Base I/O Address Switch Settings

Table 2-2 lists the 24 possible switch settings, the corresponding base I/O address, and the I/O address space used for that setting. The default settings are shown in ***bold Italic*** text.

Notice that the base address settings that correspond to an I/O address in the range from 0 to FF hex are not listed. These addresses are used by logic on the PC AT motherboard and cannot be used by I/O adapter modules.

Table 2-2. Possible Base I/O Address Settings for the AT-MXI

Switch Setting					**Base I/O Address**	**I/O Ports Used**
A9	**A8**	**A7**	**A6**	**A5**	**(hex)**	**(hex)**
0	1	0	0	0	100	100 - 11F
0	1	0	0	1	120	120 - 13F
0	1	0	1	0	140	140 - 15F
0	1	0	1	1	160	160 - 17F
0	1	1	0	0	180	180 - 19F
0	1	1	0	1	1A0	1A0 - 1BF
0	1	1	1	0	1C0	1C0 - 1DF
0	1	1	1	1	1E0	1E0 - 1FF
1	0	0	0	0	200	200 - 21F
1	0	0	0	1	220	220 - 23F
1	0	0	1	0	240	240 - 25F
1	0	0	1	1	260	260 - 27F
1	0	1	0	0	280	280 - 29F
1	0	1	0	1	2A0	2A0 - 2BF
1	0	1	1	0	2C0	2C0 - 2DF
1	0	1	1	1	2E0	2E0 - 2FF
1	1	0	0	0	300	300 - 31F
1	1	0	0	1	320	320 - 33F
1	***1***	***0***	***1***	***0***	***340***	***340 - 35F***
1	1	0	1	1	360	360 - 37F
1	1	1	0	0	380	380 - 39F
1	1	1	0	1	3A0	3A0 - 3BF
1	1	1	1	0	3C0	3C0 - 3DF
1	1	1	1	1	3E0	3E0 - 3FF

Interrupt Level Selection

The AT-MXI interface board can use either one or two of the eleven interrupt levels of the PC AT I/O channel. Setting up an interrupt level for operation involves two steps. First you select the interrupt level by arranging the jumpers on an array of pins. Next you enable the interrupt level in the NI-VXI configuration software.

Note: ***You must enable the interrupt levels using the NI-VXI configuration software before they can function. If you do not enable the interrupt level, that level will not be driven and can be used by other devices, regardless of the positions of the jumpers.***

Interrupt levels are selected by the position of two jumpers on the 3 by 11 array of pins labeled *W3*, located above the I/O card-edge connector on the AT-MXI (refer to Figure 2-1). Use the jumper *farther* from the card-edge connector to select the PC AT interrupt level that will convey board status and error information. Remember to enable this level, or board interrupt, in the system software for the AT-MXI to function properly. The factory default setting is level 12 and is enabled on this level by the default configuration software.

Use the jumper on the W3 pin array *closer* to the I/O card-edge connector to select which PC AT interrupt level corresponds to the MXIbus interrupt signal *IRQ**. Because the MXIbus interrupt is also one of the conditions covered by the other jumper, normally you do not need to use a separate interrupt level for the MXIbus *IRQ** signal; it is useful only if you want a different interrupt vector or priority for MXIbus interrupts. This jumper is set at the factory to a default level of 10.

Note: ***The AT-MXI does not have the ability to share interrupt levels with other devices. If you select an interrupt level by placing a jumper on a particular level and enable that level in software, no other device in the system can use that level. Make sure that no other devices in your system use the interrupt level(s) selected and enabled for use by the AT-MXI. If they do, change the interrupt level(s) of either the AT-MXI or the other devices. If you change an AT-MXI interrupt level, make a corresponding change to the NI-VXI configuration software as described in the*** *Configuring the Software* ***section of Chapter 5.***

The AT-MXI can use interrupt levels IRQ3, 4, 5, 6, 7, 9, 10, 11, 12, 14, and 15. Be careful when re-assigning interrupt levels on the AT-MXI. Notice that most PC ATs use interrupt level 6 for the diskette drive controller and interrupt level 14 for the hard disk drive controller. Other interrupt levels might be used by standard logic devices on the motherboard so check your computer documentation before changing interrupt levels.

Once you have chosen an interrupt level, place the jumper on the appropriate pins to select that interrupt level. Use the two rows of pins farther from the card-edge connector to select the board interrupt level, and the two rows of pins closer to the card-edge connector to select the MXIbus interrupt level. Figure 2-3(a) shows the factory default interrupt jumper setting of the AT-MXI, with board interrupt level 12 and MXIbus interrupt level 10.

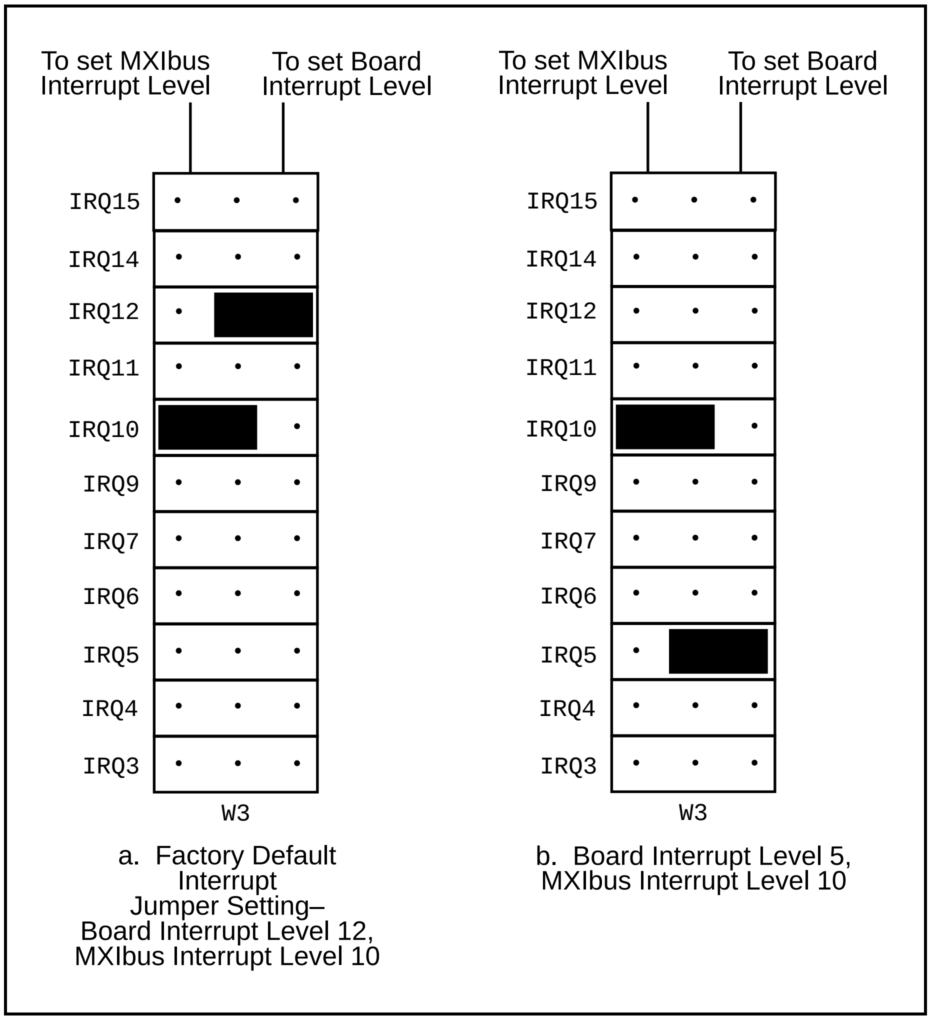

To set MXIbus Interrupt Level
To set Board Interrupt Level
To set MXIbus Interrupt Level
To set Board Interrupt Level
IRQ15
IRQ14
IRQ12
IRQ11
IRQ10
IRQ9
IRQ7
IRQ6
IRQ5
IRQ4
IRQ3
W3
a. Factory Default Interrupt Jumper Setting– Board Interrupt Level 12, MXIbus Interrupt Level 10
W3
b. Board Interrupt Level 5, MXIbus Interrupt Level 10

Figure 2-3. Board and MXIbus Interrupt Jumper Settings

To change to another interrupt level, remove the appropriate jumper from its current position and place it on the new posts. Figure 2-3(b) shows the board interrupt level changed to IRQ5.

DMA Channel Selection

The AT-MXI interface board can use one, two, or none of the seven DMA channels of the PC AT I/O channel. Setting up a DMA channel for operation involves two steps. First you select the DMA channel by arranging the jumpers on an array of pins. Next you enable the DMA channel in the NI-VXI configuration software.

Note: ***You must enable the DMA channels using the NI-VXI configuration software before they can function. If you do not enable the DMA channel, that channel will not be driven and can be used by other devices, regardless of the positions of the jumpers.***

Select the DMA channels on the two 3 by 7 arrays of pins labeled *W1* and *W2*, located above the I/O card-edge connector on the AT-MXI (refer to Figure 2-1). Use the W1 array to select the DMA request line(s), and use the W2 array to select the DMA acknowledge line(s). You must position two jumpers to select a single DMA channel. The DMA ACKnowledge (DACK*n*) and DMA ReQuest (DRQ*n*) lines selected must have the same numeric suffix for proper operation. Therefore, make sure that the jumper positions on the W1 array are identical to the jumper positions on the W2 array.

Master Mode Versus Slave Mode

The AT-MXI can function as both a MXIbus master and a MXIbus slave. As a MXIbus master, the AT-MXI circuitry determines whether a PC AT cycle is to be mapped to a MXIbus cycle intended for some external MXIbus device such as a VXIbus mainframe. As a MXIbus slave, the AT-MXI circuitry determines whether an external device is attempting to access PC AT memory or I/O resources.

Note: ***When allocating DMA channels for use by the AT-MXI, keep in mind that master-mode and slave-mode operation are two distinct asynchronous functions and require different DMA channels.***

Although the master-mode DMA jumpers are set at the factory to use DMA channel 6, the software statically disables the master-mode DMA channel with a setting of `NONE` and cannot be changed. While the AT-MXI itself is capable of using DMA to perform high-speed block-mode transfers to or from external MXIbus devices, the processor is able to transfer data at significantly faster rates by using the `movs` (move string) instruction. DMA transfers offer superior performance only in asynchronous function calls, which begin an operation and return immediately, freeing the processor to do other useful work. However, because DMA transfers would be applicable only for the `VXImove` function, which is *not* asynchronous, the software is configured to ignore this option, making the hardware setting irrelevant. The field remains in the `VXIEDIT` and `VXITEDIT` software configuration programs for backward compatibility only.

The slave-mode DMA jumpers are set at the factory to use DMA channel 3. Although the default software configuration enables this setting, several factors effectively disable the slave-mode DMA channel. As a result, DMA channel 3 can be used by other devices if you do not need to use slave-mode DMA with your AT-MXI.

The slave-mode DMA channel is necessary only for setting up shared access to PC AT resources such as PC AT memory from external MXIbus devices. If you intend to communicate with VXI devices using a shared memory protocol that uses the PC AT memory, you must select and enable a slave-mode DMA channel. However, because the AT-MXI does not share resources in its default software configuration, you need to change some software settings before you can use this feature. Refer to the *Logical Address Configuration* section of Chapter 5, *NI-VXI Software Installation and Configuration*, to see the default settings of the fields that affect the slave-mode DMA channel. The default settings are as follows:

- The **Address Space** field is set to A16 only. To share RAM you need to change this field to A16/A24.
- The **Slave Mem Window Size** field is disabled. To share RAM you need to assign a value in the allowable range.
- The **Slave I/O Window Size** field is disabled. To share I/O space you need to assign a value in the allowable range.
- The **Slave DMA Channel** field is set to use channel 3. If you modified this field to the NONE setting, slave accesses to shared RAM in VXI space and PC I/O space would not succeed. You could assign a NONE setting to this field if you intend to keep the three settings described above in their default settings. This change would free up the DMA channel for use by other devices in the system. On the other hand, if you ever need to switch back and forth between A16 and A24 space, it would be easier to have a slave DMA channel already reserved for the AT-MXI, and just change the appropriate settings in `VXIEDIT` and `VXITEDIT`. Remember that the hardware and software settings must match if you intend to use a slave DMA channel.

Note: ***Seldom, if ever, can the AT-MXI share DMA channels with other devices. If you have selected a DMA channel by placing jumpers on that channel's request and acknowledge lines and enabled the channel in software, no other devices in your system should use that channel. If DMA channels conflict, change the DMA channel(s) used by either the AT-MXI or the other device(s). If you change an AT-MXI DMA channel, make a corresponding change to the NI-VXI configuration software as described in the*** *Configuring the Software* ***section of Chapter 5.***

The AT-MXI can use DMA Channels 0, 1, 2, 3, 5, 6, and 7. Be careful when reassigning DMA channels on the AT-MXI. Notice that most PC ATs use DMA Channel 2 for the disk controller interface. It is possible that various standard logic devices on the motherboard may use other DMA channels, so check your computer documentation before changing DMA channels.

Notice that the PC AT makes a distinction between 8-bit and 16-bit DMA channels. The 8-bit channels are 0, 1, 2, and 3. The 16-bit channels are 5, 6, and 7. The master-mode DMA channel setting must match the data width of the intended block transfers. It is preferable to use one of the 16-bit channels for the master-mode interface because a 16-bit DMA channel can transfer twice the amount of data in the same number of cycles. The slave-mode DMA channel is used only to request the PC AT bus for an alternate PC AT bus master cycle. It can use any available 8-bit or 16-bit channel regardless of the intended data width of the transfers.

Use the two rows of pins farther from the card-edge connector to select the master-mode DMA channel and the two rows closer to the card-edge connector to select the slave-mode DMA channel. Remember that the jumper positions should be identical on both the W1 and W2 arrays. Figure 2-4(a) shows the factory default DMA channel setting of the AT-MXI, with master-mode DMA Channel 6 and slave-mode DMA Channel 3.

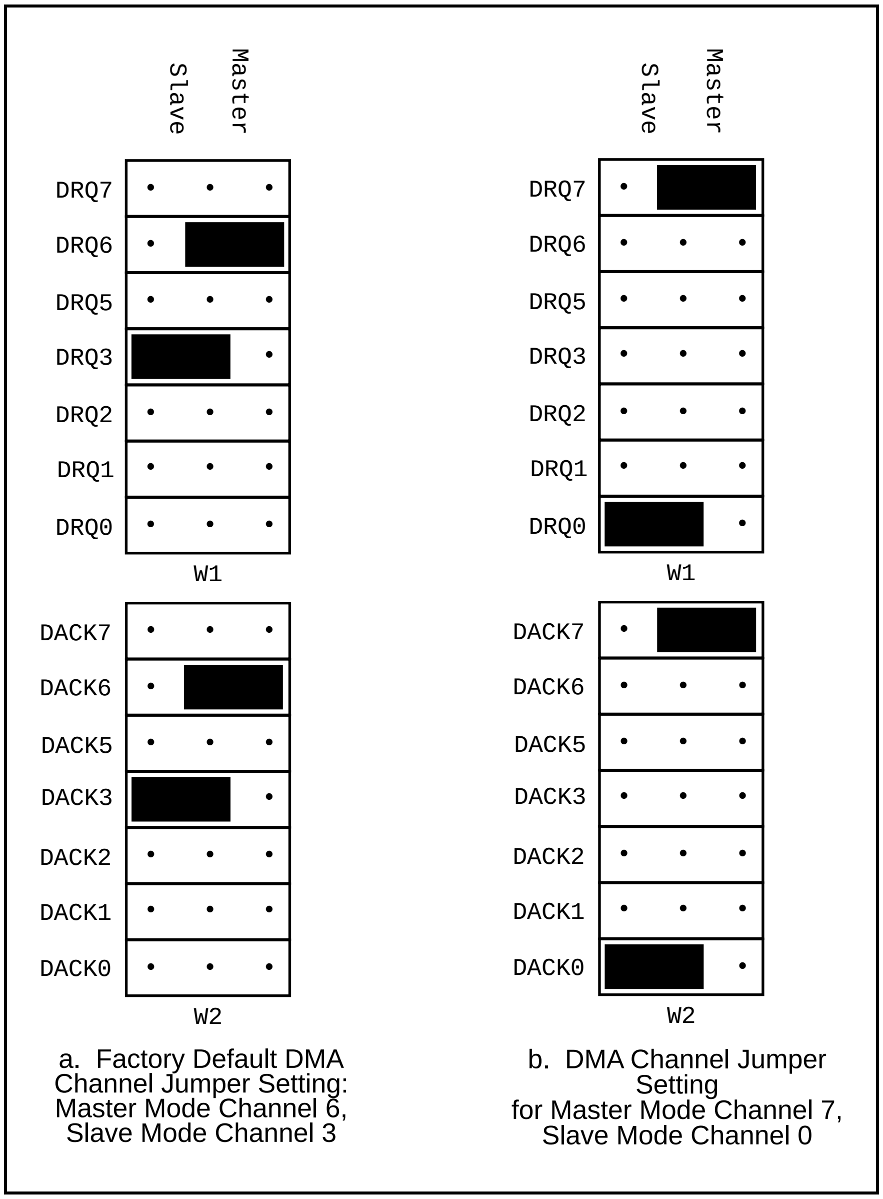

Slave
Master
DRQ7
DRQ6
DRQ5
DRQ3
DRQ2
DRQ1
DRQ0
W1
DACK7
DACK6
DACK5
DACK3
DACK2
DACK1
DACK0
W2
a. Factory Default DMA Channel Jumper Setting: Master Mode Channel 6, Slave Mode Channel 3
b. DMA Channel Jumper Setting for Master Mode Channel 7, Slave Mode Channel 0

Figure 2-4. DMA Channel Settings

To select a new DMA channel, move both the DRQ and DACK jumpers onto the appropriate pins. Remember to make a corresponding change in the system software to enable the new DMA channels. Figure 2-4(b) displays the jumper position for selecting master-mode DMA Channel 7 and slave-mode DMA Channel 0.

Install the AT-MXI

In the space provided here, record the settings of the base I/O address, the DMA channel(s), and the interrupt level(s) for future reference. You will need this information when you install the software.

AT-MXI	Setting
Base I/O Address	
Board Interrupt Level	
MXIbus Interrupt Level	
Master DMA Channel	
Slave DMA Channel	

Before attempting to install the AT-MXI, notice that some MXIbus cable connector hoods are slightly wider than most standard connector hoods and might interfere with other cables installed in adjacent PC AT slots. Normally, this will only be a problem if the cable connector hoods for the adjacent slots are also oversized. When choosing a PC AT slot in which to install the AT-MXI, verify that the MXIbus cable connector will not interfere with cables and connectors in other PC AT slots. If necessary, reposition the boards in the system to prevent cabling conflicts. It may also help to install the AT-MXI in one of the end slots so that you will have to contend with the cable connectors of only one other board. If you are still not able to connect the MXIbus cable to the AT-MXI, you can purchase a MXIbus connector extender to extend the AT-MXI connector out from the AT slot.

If you cannot configure the AT-MXI to co-exist in an existing PC AT system by repositioning the boards, you can use one of the MXIbus cable options with a standard connector hood on the cable end that attaches to the AT-MXI. The standard connector hood is narrower than the MXIbus dual-connector arrangement and provides an easier fit for many system configurations. However, this approach requires that the AT-MXI be the first device in the MXIbus daisy-chain because a cable with a single connector end cannot accept another MXIbus cable to propagate the bus. Remember that the first device in the MXIbus daisy-chain must also be configured as the MXIbus System Controller.

The following instructions are general installation instructions. Consult the user or technical reference manual of your computer for specific instructions and warnings.

1. Plug in your PC AT computer before installing the AT-MXI. The plug grounds the computer and protects it from electrical damage while you are setting up.

   Warning: ***To protect both yourself and the computer from electrical hazards, the computer should remain off until you are finished installing the board.***

2. Remove the top cover or access port to the PC AT I/O bus.

3. Select any available 16-bit full-length PC AT expansion slot. The 16-bit expansion slots have two card-edge receptacle connectors.

4. Locate the metal bracket that covers the cut-out in the back panel of the PC AT 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 AT-MXI with the MXIbus connector near the cut-out on the back panel and the other card edge lined up with the respective slot guide. Slowly push down on the front of the AT-MXI until its card edge connector is resting on the expansion slot receptacle. Using slow, evenly distributed pressure, press the AT-MXI straight down until it seats in the expansion slot.

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

8. Check the installation.

9. Replace the computer cover.

Figure 2-5 shows how to install an AT-MXI into a computer.

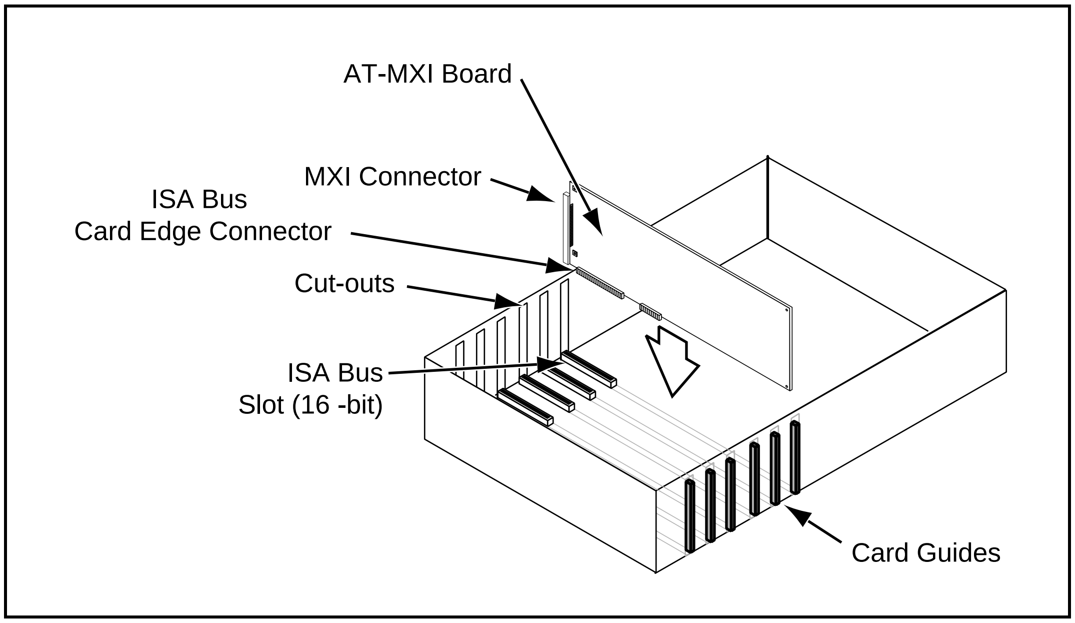


Figure 2-5. AT-MXI Installed in a Computer

Chapter 3
VXI-MXI Configuration and Installation

This chapter contains the instructions to configure and install the VXI-MXI module. This chapter applies only if you ordered the VXI-AT2010 interface kit. If you ordered the VME-AT2010 kit, skip this chapter and refer to Chapter 4, *VME-MXI Configuration and Installation*.

Warning: ***Several components on your VXI-MXI module can be damaged by electrostatic discharge. To avoid such damage in handling the module, touch the antistatic plastic package to a metal part of your computer chassis before removing the VXI-MXI from the package.***

Configure the VXI-MXI

You received either a standard or enhanced VXI-MXI module with your VXI-AT2010 interface kit. The enhanced version includes the Interrupt and Timing Extension (INTX) daughter card option, which provides additional functionality for a multiple-mainframe VXI configuration. This section describes how to configure the following options, which apply to either version of the VXI-MXI:

- Slot 0/Non-Slot 0
- VMEbus Bus Timeout
- VXI Logical Address
- VMEbus Request Level

Do not change the default settings of other jumpers and switches on the VXI-MXI module unless you plan to install more than one VXI-MXI in the same mainframe. If this is the case, refer to the *VXI-MXI User Manual* that came with your additional VXI-MXI interface for more information. The user manual also contains specific information about using the INTX daughter card option.

Figure 3-1 shows the location and factory default settings of the configuration switches and jumpers for a VXI-MXI without the INTX daughter card option. Figure 3-2 shows the factory default settings for a VXI-MXI with INTX.

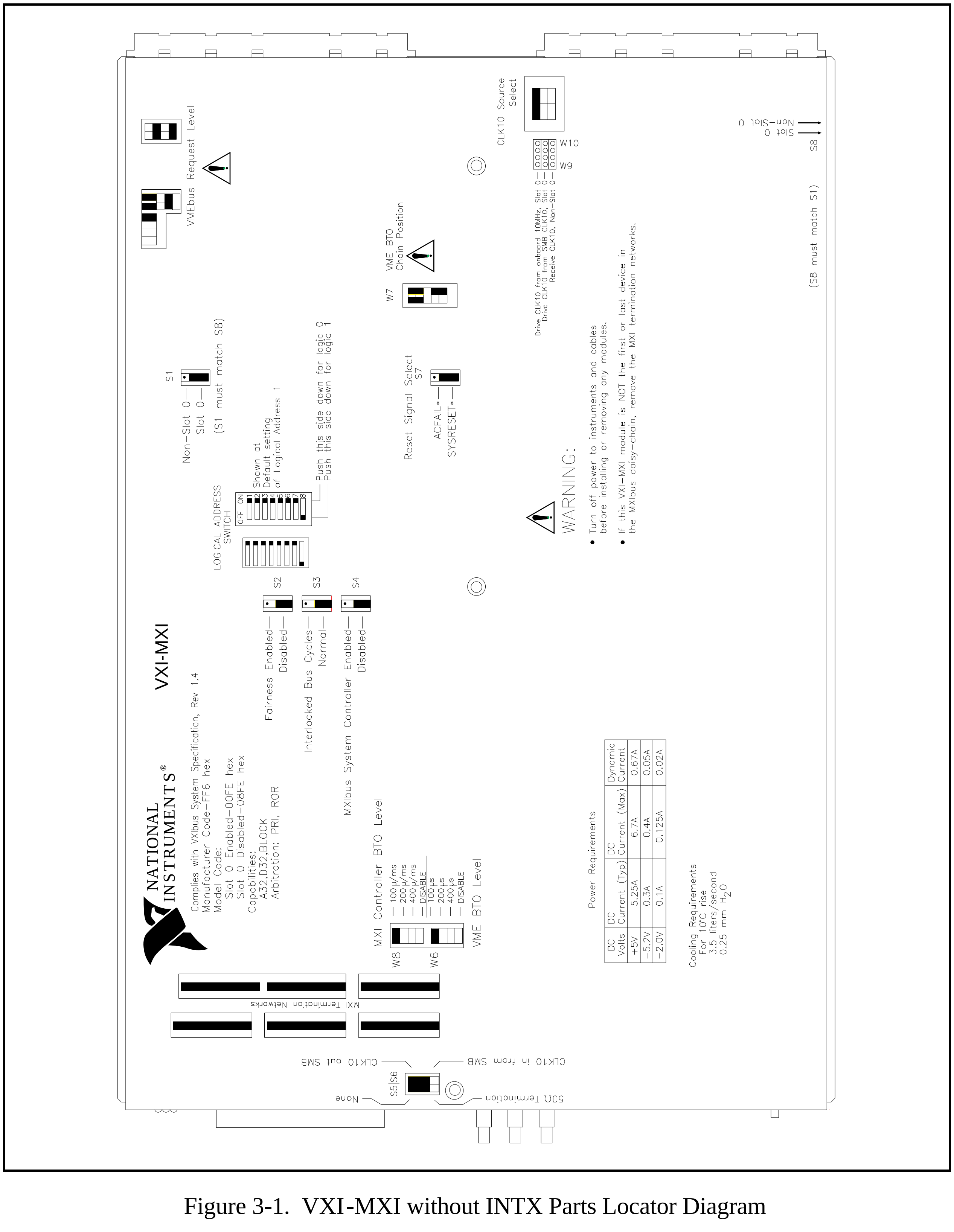


Figure 3-1. VXI-MXI without INTX Parts Locator Diagram

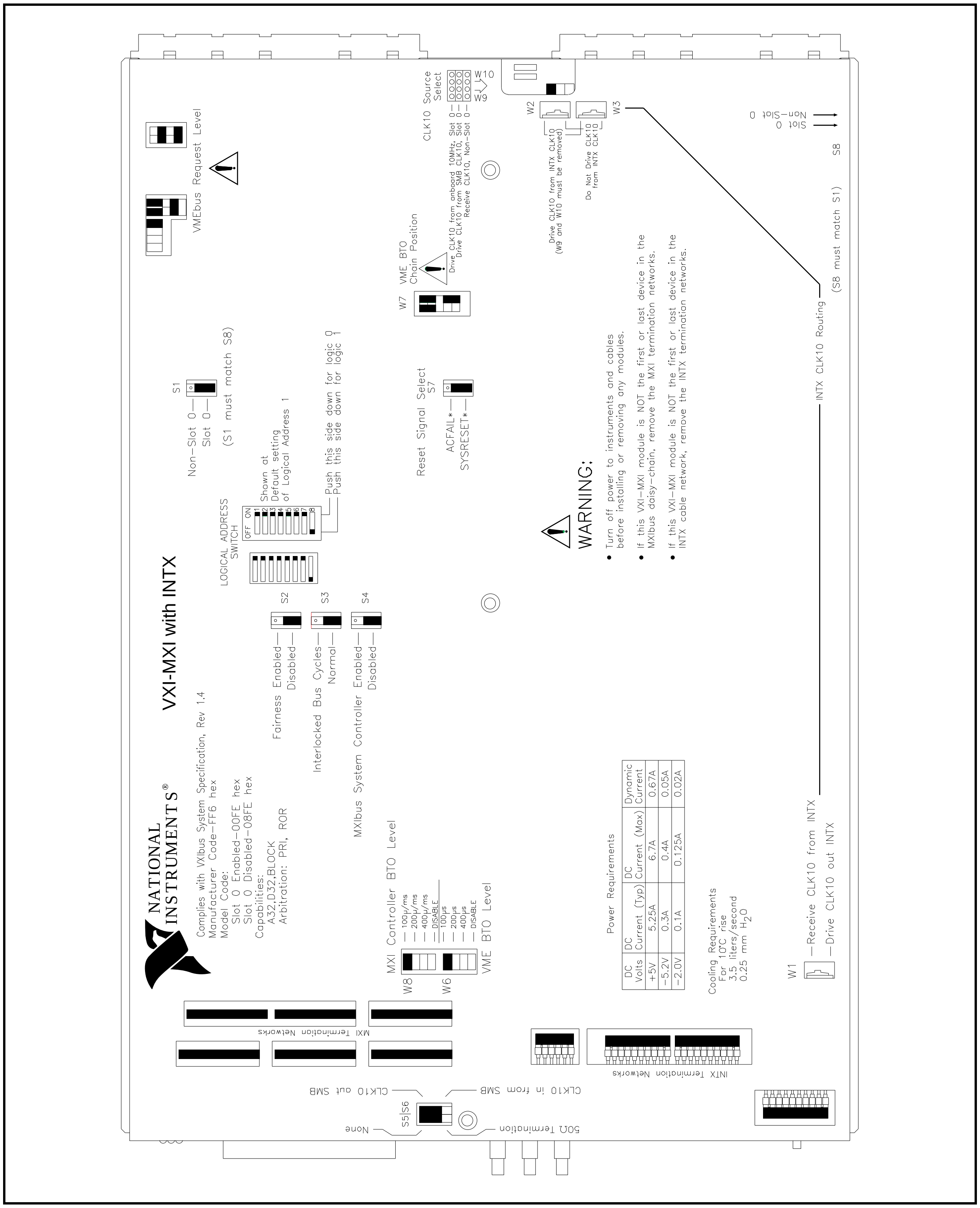

VXI-MXI with INTX
NATIONAL INSTRUMENTS®
Complies with VXIbus System Specification, Rev 1.4
Manufacturer Code–FF6 hex
Model Code:
Slot 0 Enabled–00FE hex
Slot 0 Disabled–08FE hex
Capabilities:
A32,D32,BLOCK
Arbitration: PRI, ROR
VMEbus Request Level
Non-Slot 0
Slot 0
S1
(S1 must match S8)
LOGICAL ADDRESS SWITCH
OFF ON
Shown at Default setting of Logical Address 1
Push this side down for logic 0
Push this side down for logic 1
Fairness Enabled
Disabled
S2
Interlocked Bus Cycles
Normal
S3
MXIbus System Controller Enabled
Disabled
S4
MXI Controller BTO Level
100μs/ms
200μs/ms
400μs/ms
DISABLE
100μs
200μs
400μs
DISABLE
VME BTO Level
W8
W6
MXI Termination Networks
CLK10 out SMB
CLK10 in from SMB
None
50Ω Termination
S5
S6
Reset Signal Select
S7
ACFAIL*
SYSRESET*
VME BTO Chain Position
W7
CLK10 Source Select
Drive CLK10 from onboard 10MHz, Slot 0
Drive CLK10 from SMB CLK10, Slot 0
Receive CLK10, Non-Slot 0
W10
W9
W2
W3
Drive CLK10 from INTX CLK10 (W9 and W10 must be removed)
Do Not Drive CLK10 from INTX CLK10
WARNING:
• Turn off power to instruments and cables before installing or removing any modules.
• If this VXI-MXI module is NOT the first or last device in the MXIbus daisy-chain, remove the MXI termination networks.
• If this VXI-MXI module is NOT the first or last device in the INTX cable network, remove the INTX termination networks.
Power Requirements
DC Volts | DC Current (Typ) | DC Current (Max) | Dynamic Current
+5V | 5.25A | 6.7A | 0.67A
–5.2V | 0.3A | 0.4A | 0.05A
–2.0V | 0.1A | 0.125A | 0.02A
Cooling Requirements
For 10°C rise
3.5 liters/second
0.25 mm H2O
INTX Termination Networks
W1
Receive CLK10 from INTX
Drive CLK10 out INTX
INTX CLK10 Routing
(S8 must match S1)
S8
Slot 0
Non-Slot 0

Figure 3-2. VXI-MXI with INTX Parts Locator Diagram

Front Panel Features

The VXI-MXI has the following front panel features:

- Three front panel LEDs
  - **FAILED** LED indicates that the VMEbus SYSFAIL line is asserted.
  - **VXI ACCESS** LED indicates when the VXI-MXI is accessed from the VXIbus.
  - **MXI ACCESS** LED indicates when the VXI-MXI is accessed from the MXIbus.
- MXIbus connector
- Three SMB connectors
  - Trigger input
  - Trigger output
  - External clock
- System reset pushbutton
- INTX connector (if you have a VXI-MXI with the INTX daughter card option)

Removing the Metal Enclosure

The VXI-MXI 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.

Should you find it necessary to open the enclosure, remove the three screws on the top, the three screws on the bottom, and the three screws on the right side panel of the enclosure.

VXIbus Slot 0

The VXI-MXI is shipped from the factory configured to be installed in Slot 0 of the VXIbus mainframe. If another device is already in Slot 0, you must decide which device will be the Slot 0 device and reconfigure the other device for Non-Slot 0 use.

Warning: ***Do not install a device configured for Slot 0 into another slot without first reconfiguring it for Non-Slot 0 use. Doing so could result in damage to the Non-Slot 0 device, the VXIbus backplane, or both.***

Figure 3-3 shows the default configuration settings for the VXI-MXI installed as the Slot 0 device.

To configure the VXI-MXI as a Non-Slot 0 device, change slide switches S1 and S8 and jumper blocks W7 (labeled *VME BTO Chain Position* on the right side panel), and W9 and W10 (labeled *CLK10 Source Select* on the right side panel) as shown in Figure 3-4.

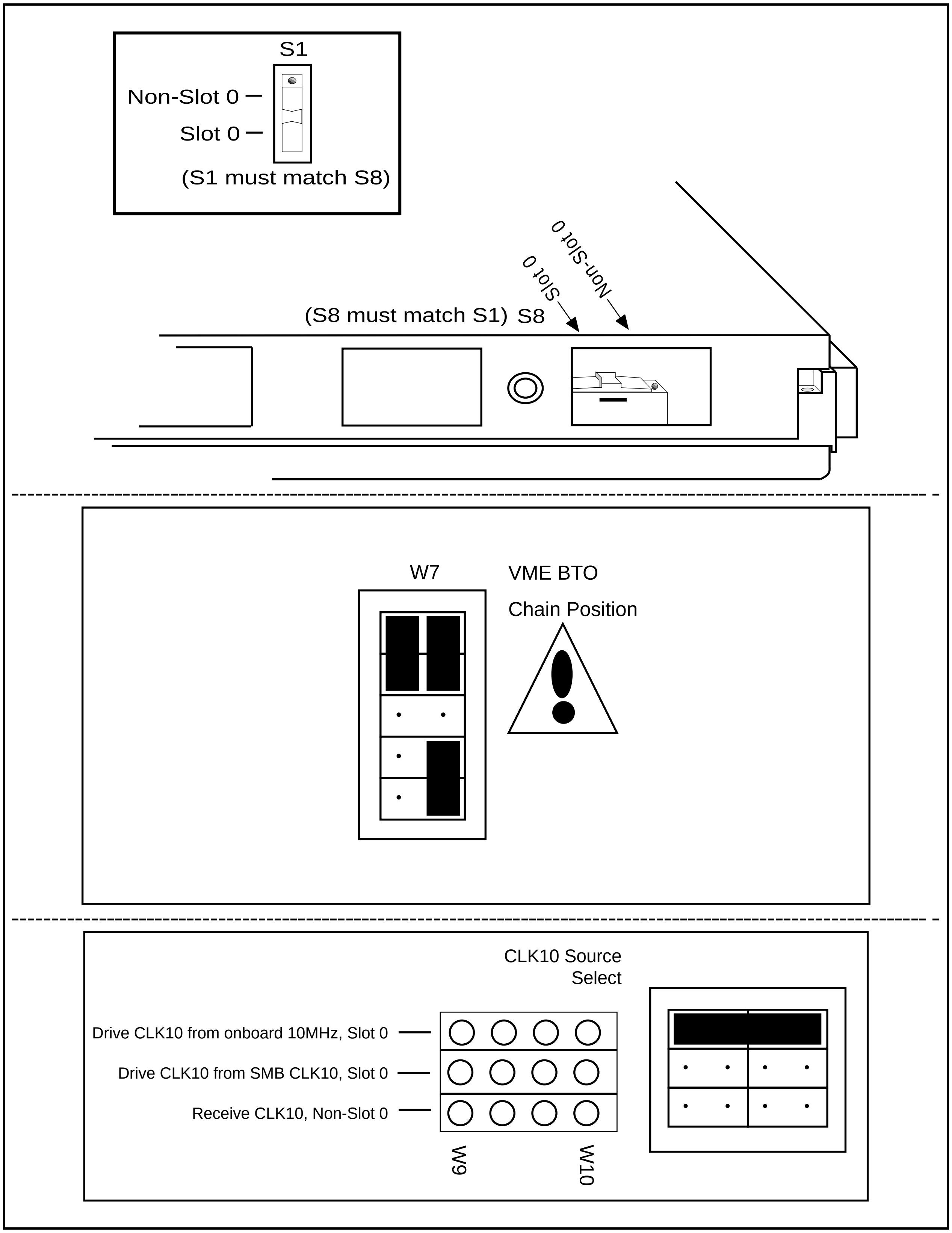


Figure 3-3. VXIbus Slot 0 Configuration

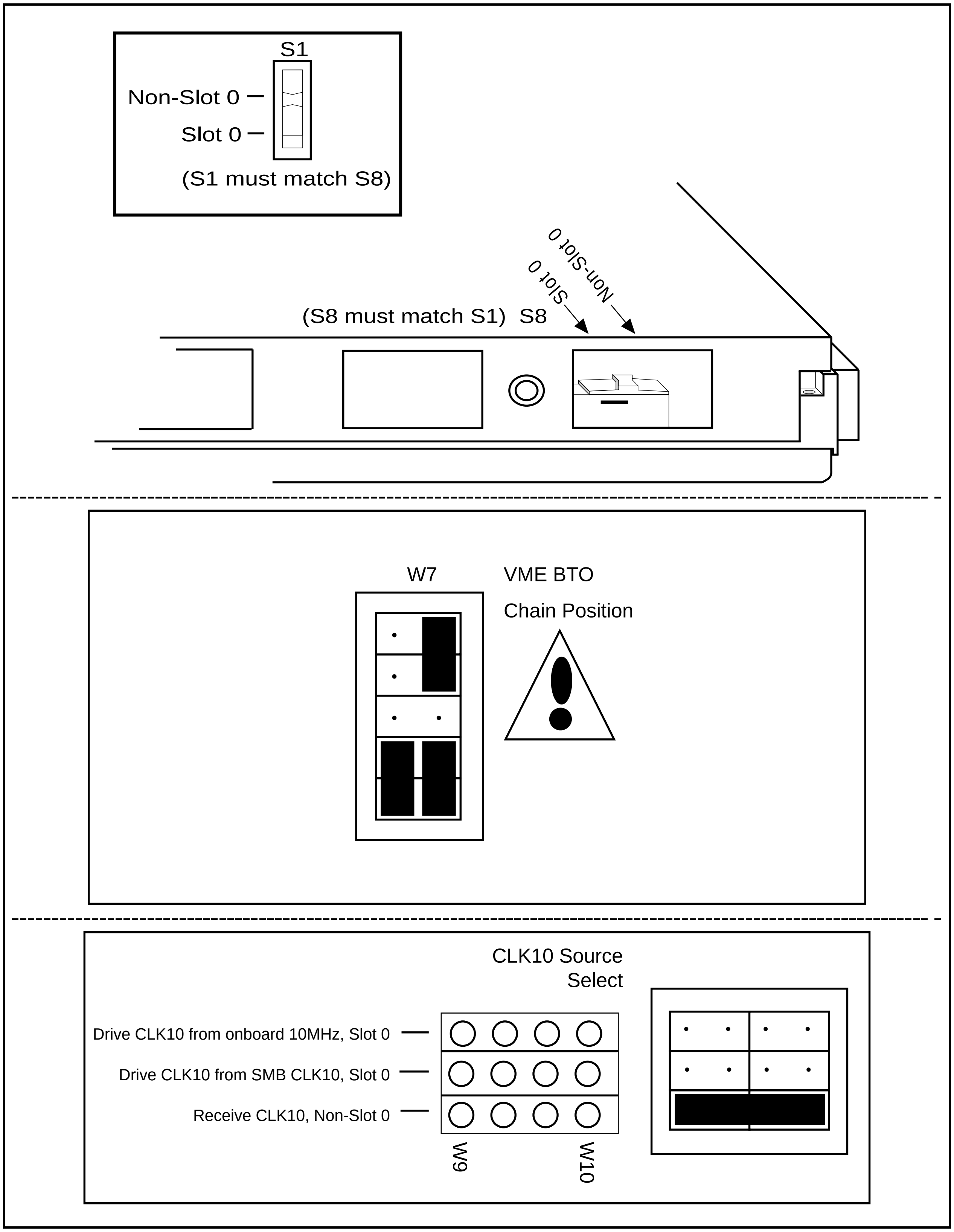


Figure 3-4. VXIbus Non-Slot 0 Configuration

When the VXI-MXI is installed in Slot 0, it becomes the VMEbus System Controller (set by slide switches S1 and S8). As a VMEbus System Controller, it has VMEbus Data Transfer Bus Arbiter (PRI 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 also drives the 16 MHz VMEbus system clock by an onboard 16 MHz oscillator with a 50% ±5% duty cycle.

The VXI-MXI also performs VMEbus BTO functions as described in the following section. The setting of the VME BTO Chain Position jumper block determines how to control these functions. As required by the VXIbus specification for a Slot 0 device, the VXI-MXI drives the 10 MHz signal, CLK10, on a differential ECL output. This is controlled by the CLK10 Source Select jumpers at locations W9 and W10. The Slot 0 setting of the CLK10 Source Select jumpers cause the VXI-MXI to drive CLK10 on the backplane. When configured for Non-Slot 0, the VXI-MXI instead receives the CLK10 signal.

Warning: ***Configuring more than one VXIbus device to drive the CLK10 lines can damage the VXIbus backplane or the CLK10 drivers on the VXIbus devices.***

VMEbus BTO

The VMEbus Bus Timeout (BTO) is a watchdog timer for transfers on the VMEbus Data Transfer Bus. After a specified amount of time (usually user-configurable), the BTO circuitry terminates a VMEbus cycle if no slave has responded. The VXI-MXI must provide the VMEbus BTO to function properly because, when a MXIbus cycle is involved, the 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 it to its maximum setting to give the MXIbus cycles as much time as possible to complete.

VXI 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.

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 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.

The AT-MXI is designated as the VXIbus Resource Manager (RM). As defined by the VXIbus specification, the RM in a VXI system has a logical address of 0.

Ensure that no other statically configurable VXIbus devices have logical addresses of either 0 or 1. If they do, change the logical address settings of the other devices so that every device in the system has a unique associated logical address.

Do not change the logical address of the VXI-MXI unless you are connecting multiple VXI-MXIs to the MXIbus. In this case, refer to the *VXI-MXI User Manual* that came with your additional VXI-MXIs for more information. You can change the logical address of the VXI-MXI by changing the setting of the 8-bit DIP switch labeled *LOGICAL ADDRESS SWITCH* on the right side panel. The ON position on the DIP switch corresponds to a logic value of 0, and the OFF position corresponds to a logic value of 1. Verify that the VXI-MXI 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.

Figure 3-5 shows switch settings for logical address hex 1 and C0.

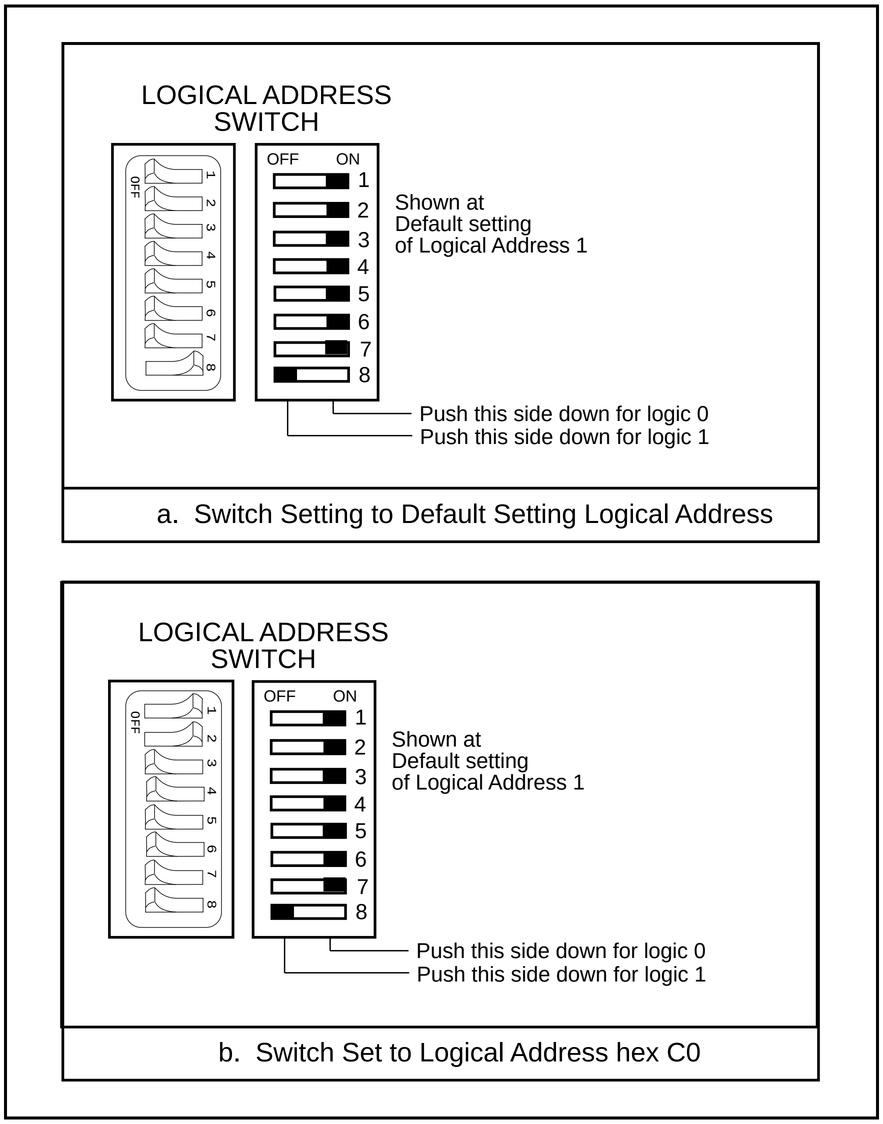

LOGICAL ADDRESS SWITCH
OFF
OFF ON
1 2 3 4 5 6 7 8
Shown at Default setting of Logical Address 1
Push this side down for logic 0
Push this side down for logic 1
a. Switch Setting to Default Setting Logical Address
LOGICAL ADDRESS SWITCH
OFF
OFF ON
1 2 3 4 5 6 7 8
Shown at Default setting of Logical Address 1
Push this side down for logic 0
Push this side down for logic 1
b. Switch Set to Logical Address hex C0

Figure 3-5. Logical Address Selection

VMEbus Request Level Selection

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

The VXI-MXI uses VMEbus request level 3 in its factory default setting, as required by the VXIbus specification. This is suitable for most VXIbus systems. However, you can change the VXI-MXI to use any of the other three request levels (0, 1, or 2) by changing the jumper configuration on the jumper blocks labeled *VMEbus Request Level* on the right side panel. You may want to change request levels to change the priority of the VXI-MXI request signal. For more information, refer to the VMEbus specification.

To change the VMEbus request level of the VXI-MXI, rearrange the jumpers on the pin arrays as shown in Figure 3-6.

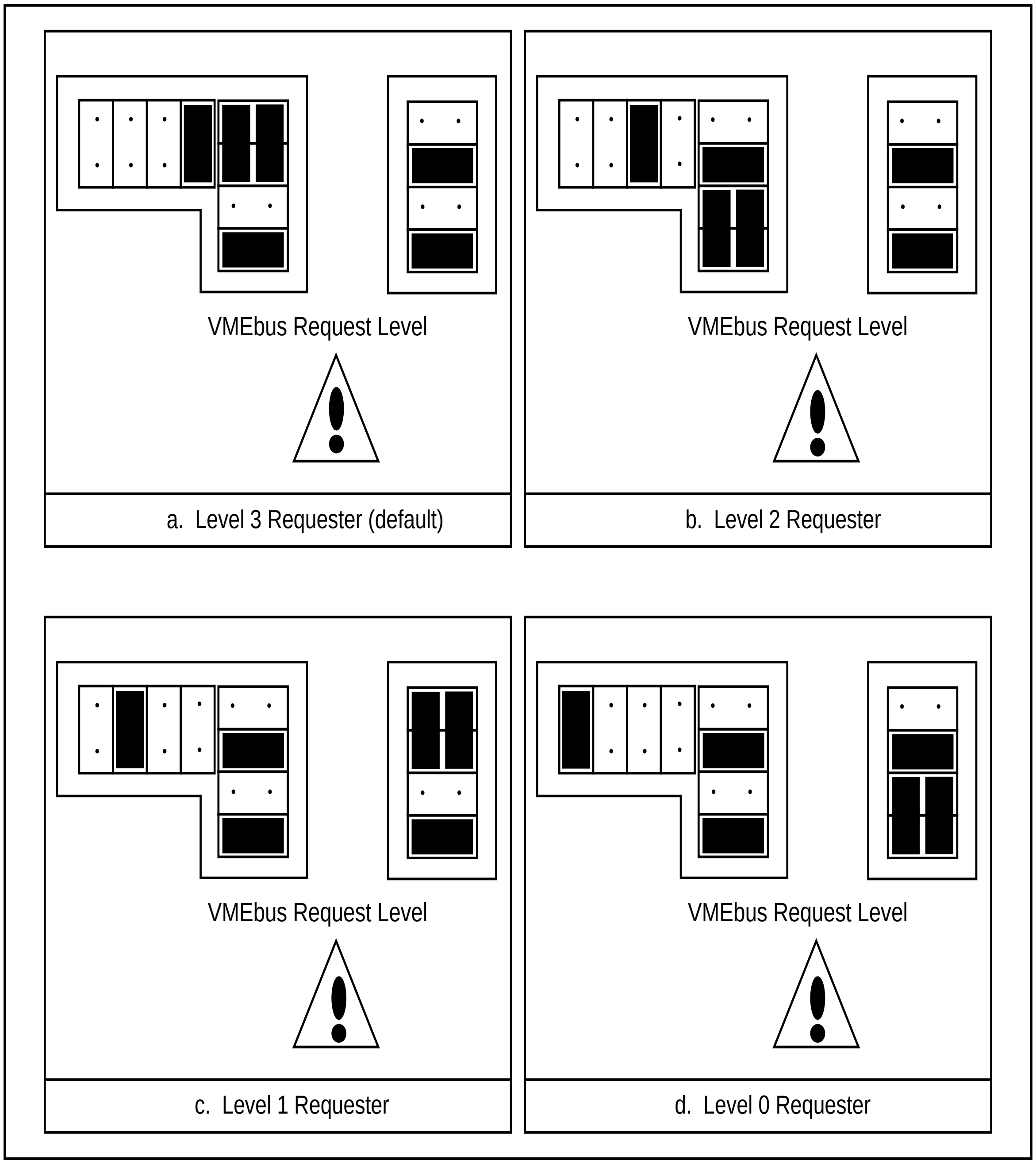


Figure 3-6. VXI-MXI VMEbus Requester Jumper Settings

Install the VXI-MXI

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

1. Plug in your mainframe before installing the VXI-MXI. The plug grounds the mainframe and protects it from electrical damage while you are installing boards.

   Warning: ***To protect both yourself and the mainframe from electrical hazards, the mainframe should remain off until you are finished installing the board.***

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

3. If the VXI-MXI will be installed in a D-size mainframe, install a support designed for installing C-size boards in D-size mainframes.

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

4. Insert the VXI-MXI 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 straight into the slot until its plug connectors are resting on the backplane receptacle connectors. Using slow, evenly distributed pressure, press the VXI-MXI straight in until it seats in the expansion slot. The front panel of the VXI-MXI 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 MXIbus cables. MXIbus cables can have either a single connector on each end, or a single connector on one cable end and a double connector on the other end. Your VXI-AT2010 kit comes standard with a cable with single connectors on each end.

Nonpolarized Cables

The cable with a single connector on each cable end is nonpolarized and may be installed with either end connected to either device. Be sure to tighten the screw locks to ensure proper pin connection. See Figure 3-7.

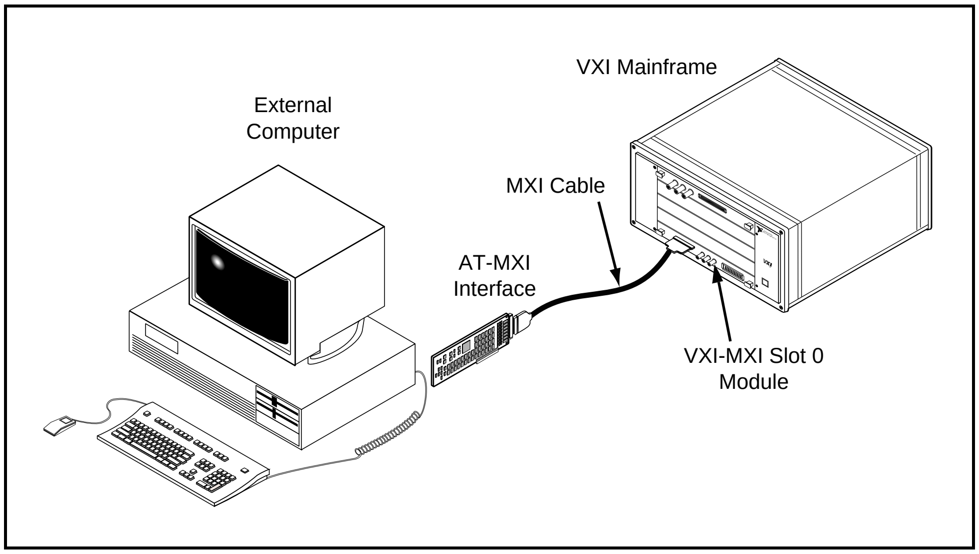


Figure 3-7. MXIbus Nonpolarized Cable Configuration

When you have properly connected the MXIbus cable, power on the VXI mainframe and then the PC AT computer. If you are using polarized cables, refer to the next section.

Note: ***Always turn on the mainframe first. Doing so makes it possible for the PC AT to access the VXI boards in the mainframe as soon as the PC AT starts up.***

Polarized Cables

If you are using a MXIbus cable with a single connector on one cable end and a double connector on the other end, it is a polarized cable that you must install correctly for the system to function properly. Connect the end with the *single* connector to the AT-MXI and the end of the cable with the *double* connector to the VXI-MXI. Be sure to tighten the screw locks to ensure proper pin connection. See Figure 3-8.

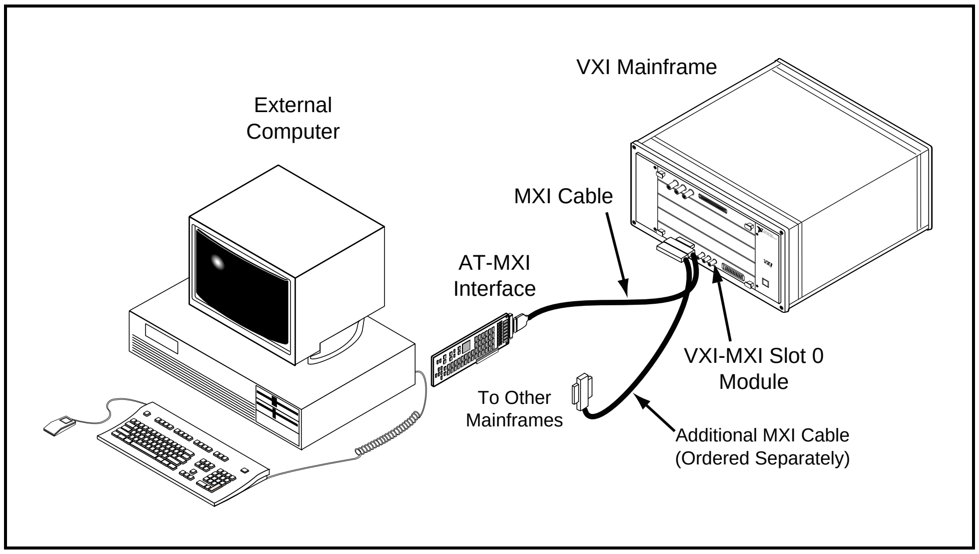


Figure 3-8. MXIbus Polarized Cable Configuration

When you have properly connected the MXIbus cable, power on the VXI mainframe and then the PC AT computer.

Note: ***Always turn on the mainframe first. Doing so makes it possible for the PC AT to access the VXI boards in the mainframe as soon as the PC AT starts up.***

Chapter 4
VME-MXI Configuration and Installation

This chapter contains the instructions to configure and install the VME-MXI module. This chapter applies only if you ordered the VME-AT2010 interface kit. If you ordered the VXI-AT2010 kit, go back to Chapter 3, *VXI-MXI Configuration and Installation*.

Warning: ***Several components on your VME-MXI module can be damaged by electrostatic discharge. To avoid such damage in handling the module, touch the antistatic plastic package to a metal part of your computer chassis before removing the VME-MXI from the package.***

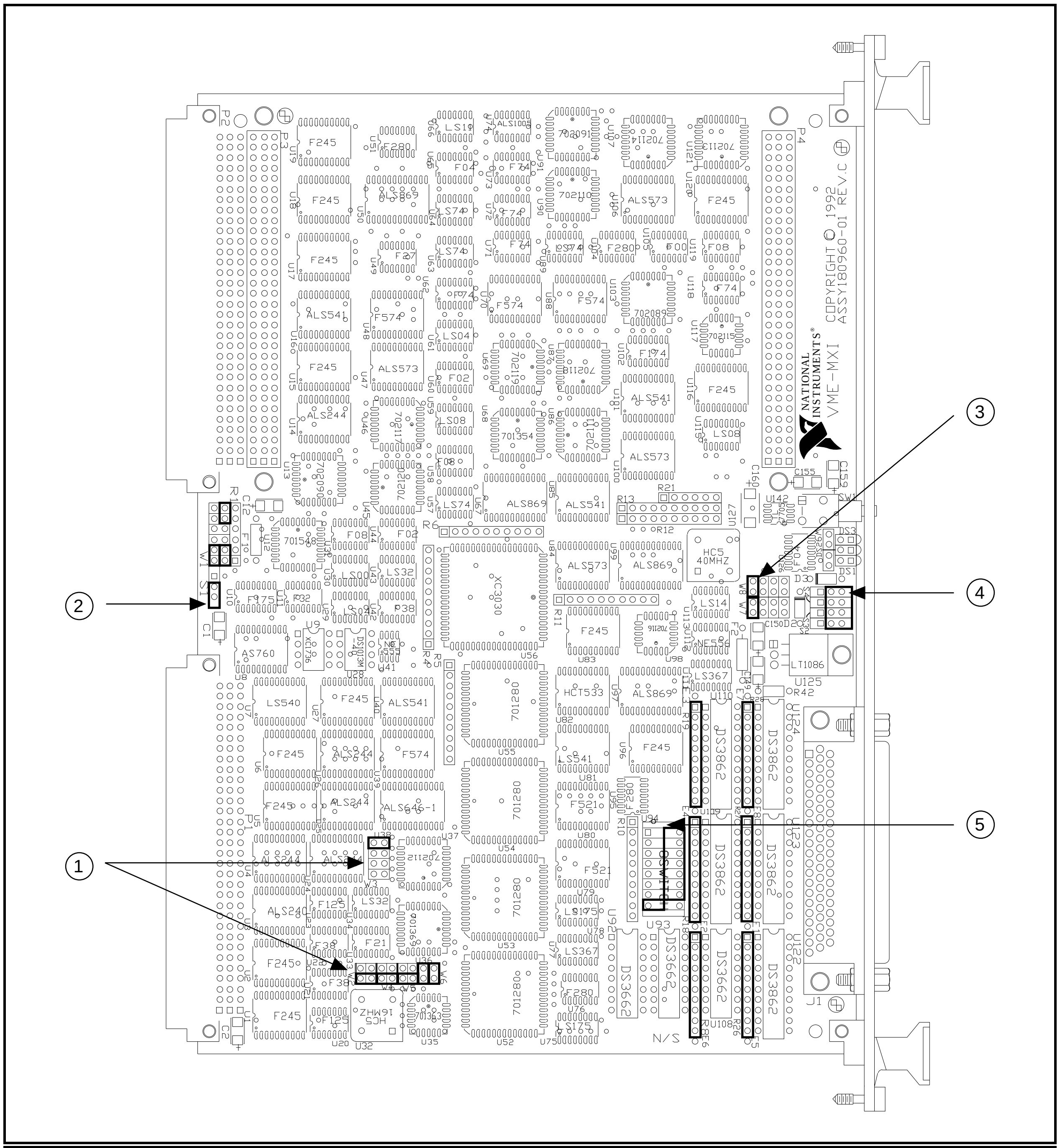
Configure the VME-MXI

You received either a standard or enhanced VME-MXI with your VME-AT2010 interface kit. The enhanced version includes the Interrupt and Timing Extension (INTX) daughter card option, which provides additional functionality for a multiple-chassis VME configuration. This section describes how to configure the following options, which apply to either version of the VME-MXI:

- VMEbus System Controller Selection
- VMEbus Request Level
- VMEbus Timeout Value

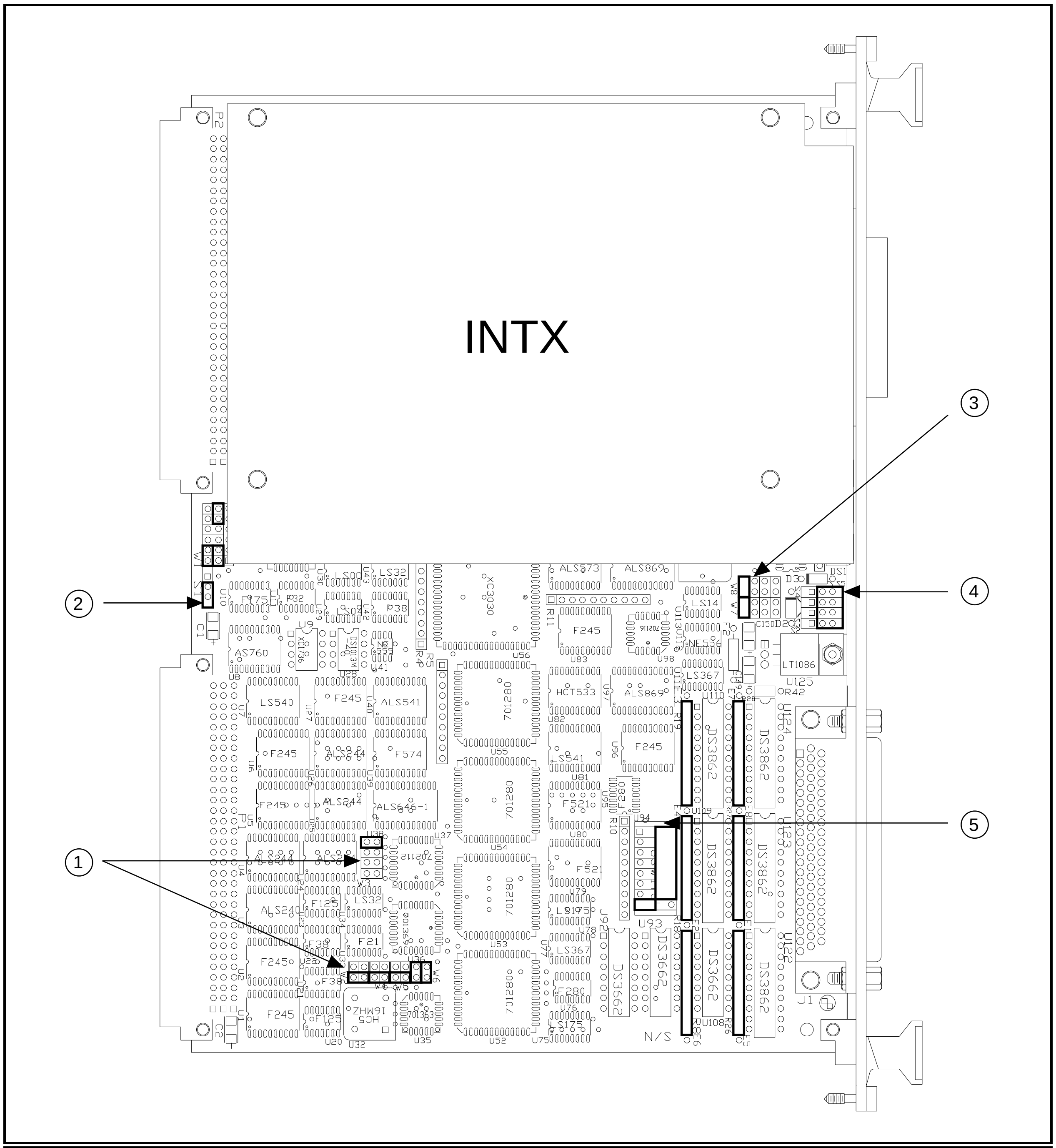
Do not change the default settings of other jumpers and switches on the VME-MXI unless you plan to install more than one VME-MXI in the same chassis or have multiple VME chassis linked via MXI. If this is the case, refer to the *VME-MXI User Manual* that came with your additional VME-MXI interface for more information. The user manual also contains specific information about using the INTX daughter card option.

Figure 4-1 shows the location and factory default settings of the configuration switches and jumpers for a VME-MXI without the INTX daughter card option. Figure 4-2 shows the factory default settings for a VME-MXI with INTX.



1	VMEbus Request Level
2	Reset Signal Select
3	VMEbus Timeout Value
4	VMEbus System Controller
5	MXIbus Logical Address

Figure 4-1. VME-MXI without INTX Parts Locator Diagram



1	VMEbus Request
2	Reset Signal Select
3	VMEbus Timeout Value
4	VMEbus System Controller
5	MXIbus Logical Address

Figure 4-2. VME-MXI with INTX Parts Locator Diagram

Front Panel Features

The VME-MXI has the following front panel features:

- Three front panel LEDs
  - **FAILED** LED indicates that the VMEbus SYSFAIL line is asserted.
  - **VME ACCESS** LED indicates when the VME-MXI is accessed from the VMEbus.
  - **MXI ACCESS** LED indicates when the VME-MXI is accessed from the MXIbus.
- MXIbus connector
- System reset pushbutton
- INTX connector (if you have a VME-MXI with the INTX daughter card option)

VMEbus System Controller

The VME-MXI is shipped from the factory configured to be installed in Slot 1 of the VMEbus chassis as the VMEbus System Controller. If another device is already in Slot 1, you must decide which device will be the VMEbus System Controller and reconfigure the other device as non-VMEbus System Controller.

When the VME-MXI is the VMEbus System Controller, it has the VMEbus Data Transfer Bus Arbiter capability (PRI ARBITER) and it drives the 16 MHz VMEbus system clock. The VMEbus Data Transfer Bus Arbiter circuitry accepts bus requests on all four VMEbus request levels, prioritizes the requests, and grants the bus to the highest priority requester. The VMEbus system clock is driven by an onboard 16 MHz oscillator with a 50% ±5% duty cycle.

If you want to install the VME-MXI into a VXIbus mainframe, you can install it in any slot except Slot 0. The VME-MXI has VXIbus configuration registers, which makes it compatible with the VXIbus specification. However, the VME-MXI does not have the CLK10 and MODID circuitry required by a Slot 0 VXIbus device. Therefore, configure the VME-MXI as a non-VMEbus System Controller and install it in any other slot.

Figure 4-3(a) shows the default configuration setting for the VME-MXI installed as the VMEbus System Controller. To configure the VME-MXI as a non-VMEbus System Controller, change slide switch S5 as depicted in Figure 4-3(b).

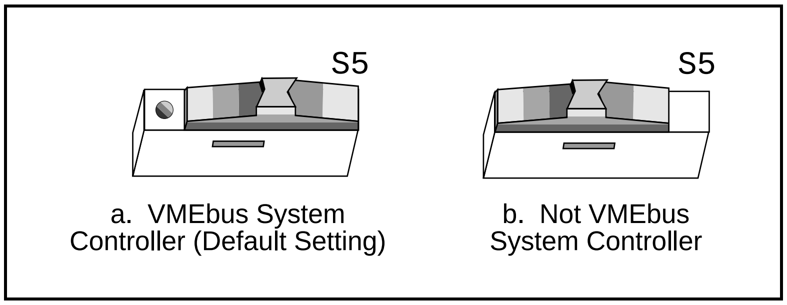

S5
S5
a. VMEbus System Controller (Default Setting)
b. Not VMEbus System Controller

Figure 4-3. VMEbus System Controller Selection

VME Address

The AT-MXI and the VME-MXI use VME short (A16) address space for their configuration, communication, and status registers. Because both boards not only are VMEbus-compatible but are also compatible with the VXIbus specification, each board has an associated logical address. The logical address assigns a unique 64-byte block of address space in the upper 16 KB of A16 space to each board. The starting address of a device's address space can be calculated using the following formula:

starting address (A16 space) = (*logical address* * 40h) + C000h

The AT-MXI is responsible for configuring the system upon power-up or reset and is referred to as the System Resource Manager. Because the System Resource Manager is defined to have a logical address of 0, the AT-MXI will use A16 address space from C000h through C03Fh.

The VME-MXI is factory configured with a logical address of 1, meaning that it occupies A16 space in the range of C040h through C07Fh. The logical address of the VME-MXI is configurable via an onboard switch, but should not be changed unless you are using multiple VME-MXIs in your system. If this is the case, refer to the *VME-MXI User Manual* that came with your additional VME-MXIs for more information.

Ensure that no other VMEbus devices in your system occupy address space used by the AT-MXI or the VME-MXI. If they do, change the other VME devices so that they occupy unique portions of address space. It is recommended that any A16 devices have their operational registers in the lower three-quarters of A16 space (below C000h), leaving the upper quarter for logical addresses. This is required if the VME-MXI is part of a VXIbus system. In addition, if you are installing the VME-MXI in a VXIbus system, make sure that no other VXIbus devices are statically configured at either Logical Address 0 or 1.

VMEbus Request Level Selection

The VME-MXI uses one of the four VMEbus request levels to request use of the VME Data Transfer Bus (DTB). The VME-MXI requests use of the DTB whenever an external MXIbus device, such as a PC AT computer with an AT-MXI interface, attempts a transfer that maps into the VMEbus chassis.

The VME-MXI is factory configured to use VMEbus request level 3. This is the highest priority request level, and is suitable for most VMEbus systems. However, you can change the VME-MXI to use any of the other three request levels (0, 1, or 2) by changing the jumper configuration of the three pin arrays at locations W2, W3, W4, W5, and W6. You may want to change request levels to change the priority of the VME-MXI request signal. For more information, refer to the VMEbus specification.

To change the VMEbus request level of the VME-MXI, rearrange the jumpers on the pin arrays as shown in Figure 4-4.

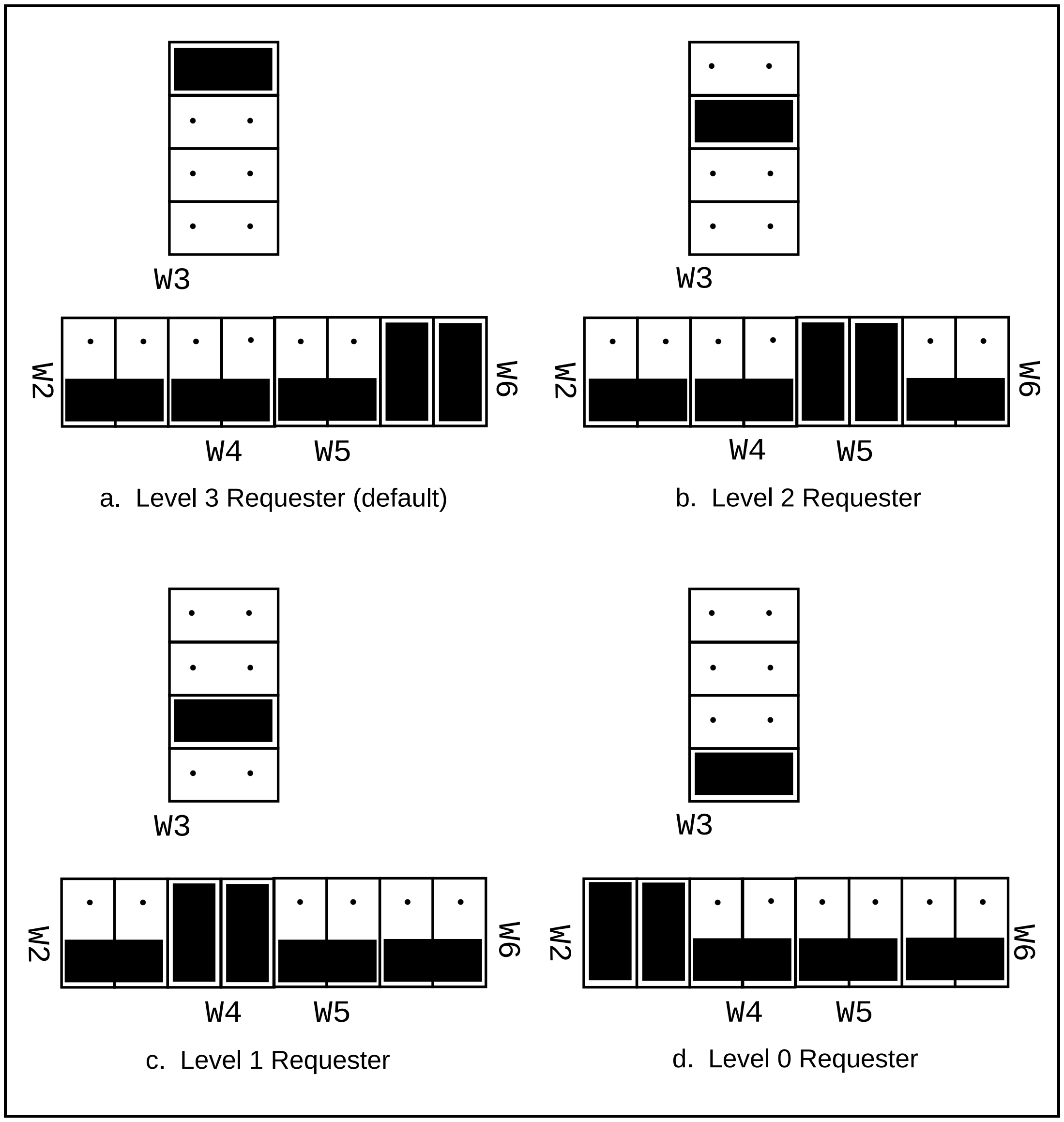


Figure 4-4. VME-MXI VMEbus Requester Jumper Settings

VMEbus Timeout Value

The VME-MXI contains the VMEbus Bus Timeout Unit (BTO) circuitry for the VMEbus. The BTO monitors the current bus cycle and asserts the bus error (BERR*) signal if either data strobe (DS1* or DS0*) remains active for a given amount of time.

The VMEbus system should have one, and only one, device that functions as the BTO Monitor on the VMEbus. The VME-MXI is factory configured to be the VMEbus System Controller (installed in Slot 1) and to provide the VMEbus BTO function. It is recommended that you use the VME-MXI as the VMEbus System Controller. If, however, you use another device as the VMEbus System Controller, it is recommended that you still retain the VME-MXI as the BTO Monitor, if possible. In this case, disable the BTO function on the System Controller and leave it enabled on the VME-MXI. If it is not possible to disable the BTO function on the System Controller, you can disable the BTO function on the VME-MXI instead. However, be sure that the BTO timeout value on your System Controller is sufficient to allow transfers to complete across the MXIbus. This is especially important for multiple-chassis systems or for systems with more than one VME-MXI.

You can either disable the VMEbus BTO value or set it to 100, 200, or 400 µsec by rearranging the jumper selection at location W8, as shown in Figure 4-5.

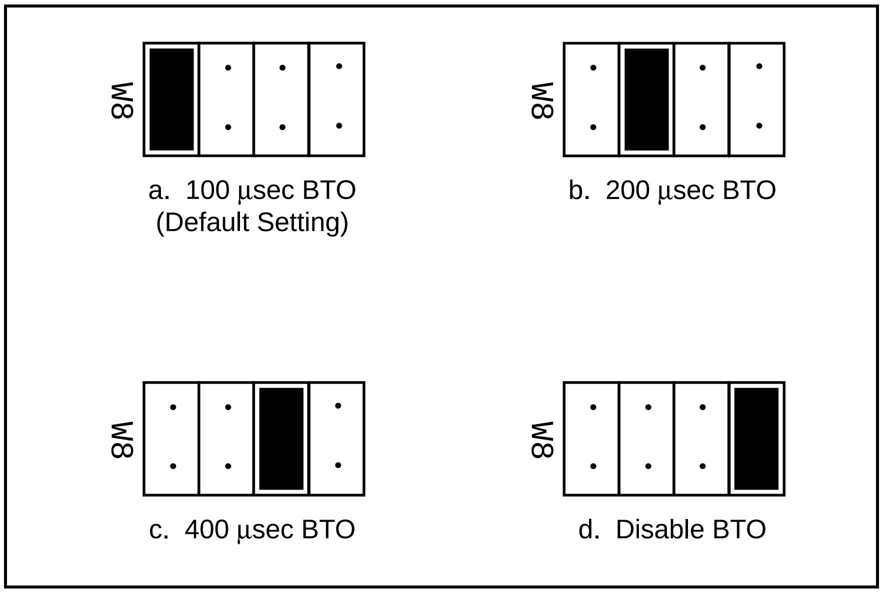


Figure 4-5. VMEbus Timeout Value Selection

Install the VME-MXI

Given below are general installation instructions for the VME-MXI. Consult the user manual or technical reference manual of your VMEbus chassis for specific instructions and warnings.

1. Plug in your VMEbus chassis before installing the VME-MXI. The plug grounds the chassis and protects it from electrical damage while you are installing boards.

   Warning: ***To protect both yourself and the chassis from electrical hazards, the chassis should remain off until you are finished installing the board.***

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

3. Select the slot in the chassis that is correct for your system (see warning below) and insert the VME-MXI in the slot by aligning the top and bottom of the card with the card edge guides inside the chassis. Slowly push the VME-MXI straight into the slot until its plug connectors are resting on the backplane receptacle connectors. Using slow evenly distributed pressure, press the VME-MXI straight in until it seats in the expansion slot. The front panel of the VME-MXI should be even with the front panel of the chassis.

   Warning: ***The VME-MXI can be installed into your VMEbus chassis either in Slot 1 when configured as the VMEbus System Controller, or it can be installed in any other slot except Slot 1 when configured as non-System Controller. Installing your VME-MXI into a slot that does not correspond with the jumper settings may result in damage to the VME-MXI, the VMEbus backplane, or both.***

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 chassis.

Connect the MXIbus Cable

There are two basic types of MXIbus cables. MXIbus cables can have either a single connector on each end or a single connector on one cable end and a double connector on the other end. Your VME-AT2010 kit comes standard with a cable with single connectors on each end.

Nonpolarized Cables

The cable with a single connector on each cable end is nonpolarized and can be installed with either end connected to either device. Be sure to tighten the screw locks to ensure proper pin connection. See Figure 4-6.

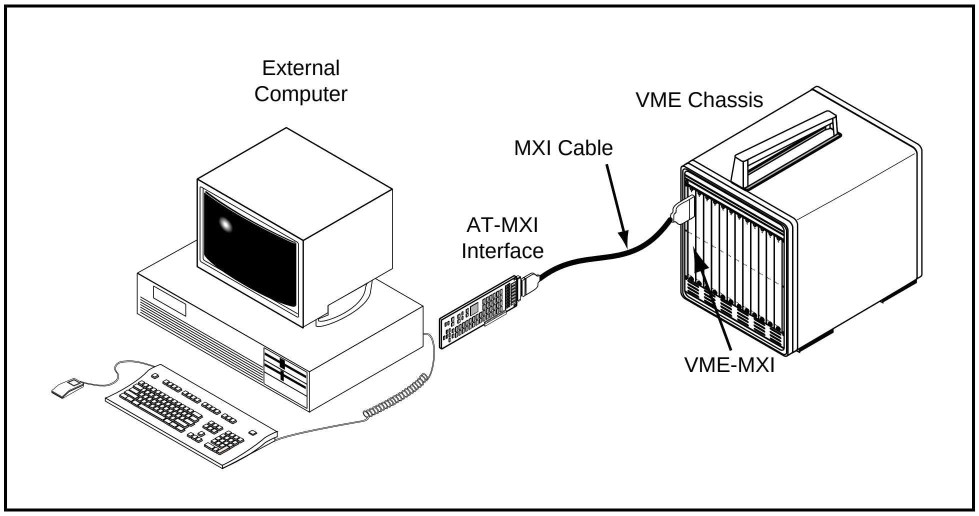


Figure 4-6. MXIbus Nonpolarized Cable Configuration

When you have properly connected the MXIbus cable, power on the VME chassis and then the PC AT computer. After all devices are powered on and running properly, you may run the Resource Manager software utility to configure your system.

Polarized Cables

If you are using a MXIbus cable with a single connector on one cable end and a double connector on the other end, it is a polarized cable that must be installed correctly for the system to function properly. Connect the end with the *single* connector to the AT-MXI and the end of the cable with the *double* connector to the VME-MXI. Be sure to tighten the screw locks to ensure proper pin connection. See Figure 4-7.

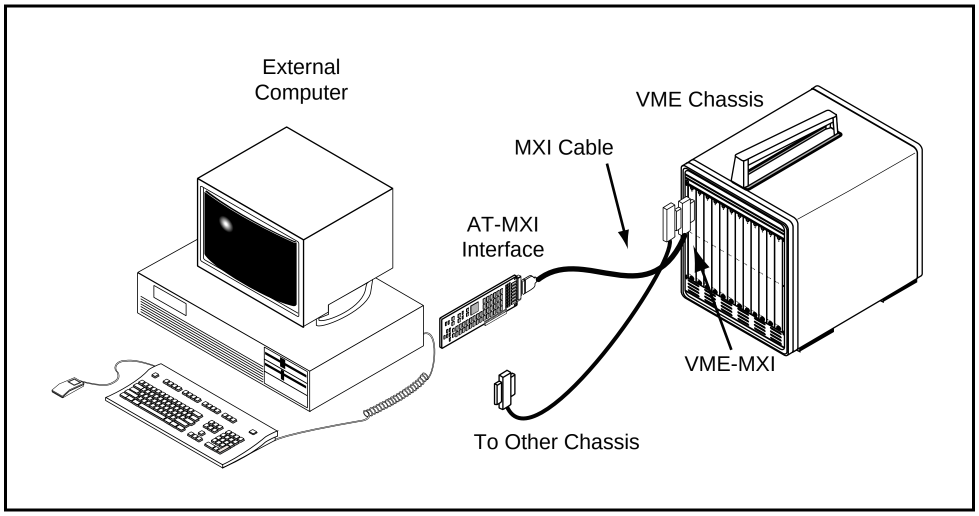


Figure 4-7. MXIbus Polarized Cable Configuration

When you have properly connected the MXIbus cable, power on the VME chassis and then the PC AT computer. After all devices are powered on and running properly, you may run the Resource Manager software utility to configure your system.

Chapter 5
NI-VXI Software Installation and Configuration

This chapter contains instructions for installing and configuring the NI-VXI software for Windows, and contains a description of the `VXIEDIT` configuration editor.

The NI-VXI software package was actually developed for the VXI-AT2010 kit for VXI systems, but you can also use it with the VME-AT2010 kit for VME systems. Because VXI is a superset of VME, and because the VME-MXI interface board for the VMEbus is compatible with the VXI-MXI interface board for the VXIbus, the NI-VXI software package contains comprehensive tools for VME systems. Some of the utility programs and driver software routines in the NI-VXI software package apply only to VXI systems and are not used in VME systems. Refer to the *NI-VXI C Software Reference Manual for VME* for detailed information about using the NI-VXI software with VME systems.

Installing the Software

The NI-VXI distribution disks that came with your AT-MXI contain the NI-VXI files as well as an `INSTALL` program. The `INSTALL` program is used 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.

Note: ***The NI-VXI software requires approximately 3 MB of free space on your hard disk. Create the necessary free space on the hard disk before starting the installation.***

You can quit the `INSTALL` program at any time by choosing *Exit* from the initial screen, or by pressing the **Cancel** button once installation has begun. If you do not have a mouse, pressing the <*Esc*> key performs the same action. If you choose to run the `INSTALL` program again, it starts over and recopies all the necessary files. If you made any changes from the default configuration, you will need to specify those changes again.

Running INSTALL

Run `INSTALL.EXE` on distribution disk 1. You can run this program from the Windows Program Manager's **File** menu, **Run** option, or from the DOS prompt (which loads Windows and then runs `INSTALL`). However, it *cannot* be run from the Windows DOS Shell. `INSTALL` is a graphical, interactive, self-guiding program, which installs all the necessary files (for a complete or partial installation, as user-specified) on the hard disk.

The program prompts you to enter the following information.

- Environments to install: You can choose whether to install NI-VXI for DOS, Windows, or both. The default is to install both.
- Languages to install: You can choose whether to install the Microsoft C, Borland C, and/or QuickBASIC libraries. The default is to install all of them, but if you are using only one compiler, you can select only that compiler.
- Destination directory: This is the complete pathname of the directory where you want to install the software.
- Modifying DOS initialization files: You can choose whether `INSTALL` modifies the `AUTOEXEC.BAT` file to include the location of the NI-VXI library. If you choose to modify `AUTOEXEC.BAT`, you must also enter the drive where it can be found. For more information about this option, refer to the section *Modifying the AUTOEXEC.BAT File* later in this chapter.
- Modifying Windows initialization files: You can choose whether `INSTALL` modifies the `PROGMAN.INI`, `SYSTEM.INI`, and `WIN.INI` files to include the location of the NI-VXI library. If you choose to modify these files, you must also enter the directory in which the files can be found. For more information about this option, refer to the sections *Modifying the PROGMAN.INI File*, *Modifying the SYSTEM.INI File*, and *Modifying the WIN.INI File* later in this chapter.

If you choose to let `INSTALL` modify any or all of the initialization files, it changes the extension of the old file to `.BAK` and saves it for safety purposes. If you choose not to let `INSTALL` modify the initialization files, it creates new files with the extension `.VXI`, which contain the changes it proposes you make to the files. You can use the `.VXI` files as a guide to make the changes yourself.

Note: ***The*** `NIVXIPATH` ***variables in the*** `SYSTEM.INI`, `WIN.INI`***, and*** `AUTOEXEC.BAT` ***files should point to the same subdirectory.***

If `INSTALL` detects a previous installation of the NI-VXI software, it prompts you to determine whether to delete or overwrite the previous version, or to provide an alternate path in which to install the new version.

Caution: ***If you have a previous version of the NI-VXI software installed,*** `INSTALL` ***does not automatically create a backup of the software files. If there are files you want to preserve, be sure to exit the installation program at this time and make a backup before continuing.***

Modifying the AUTOEXEC.BAT File

If you choose to let `INSTALL` 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 `INSTALL` modifies the file, it saves the old file as `AUTOEXEC.BAK` and keeps it for safety purposes. The previously specified directories in `PATH`, `LIB`, and `INCLUDE` remain unchanged. `INSTALL` also adds a new environment variable `NIVXIPATH`, and appends a command line to execute `VXIINIT.EXE` automatically.

If you choose not to let `INSTALL` 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.

Modifying the PROGMAN.INI File

If you choose to let `INSTALL` modify your `PROGMAN.INI` file, it specifies the `NIVXI.GRP` folder file as shown below.

```
[Settings]
Order= x <etc>

[Groups]
Groupx= <NI-VXI directory>\WIN\NIVXI.GRP
```

where <*etc*> represents the numbers already on this line, and where *x* is a new number between 1 and 99 and is the same on both of the preceding lines.

When `INSTALL` modifies the file, it saves the old file as `PROGMAN.BAK` for safety purposes. If you do not let `INSTALL` modify your `PROGMAN.INI` file, it creates a file called `PROGMAN.VXI` in the NI-VXI directory, which you can use as a guide to make the changes yourself.

Modifying the SYSTEM.INI File

If you choose to let `INSTALL` modify your `SYSTEM.INI` file, it adds a line to the `[386Enh]` section to load a device driver that NI-VXI needs for shared memory accesses. The file, which must be loaded at Windows startup, is called `NIVXIPHM.386` and is normally located in the `WIN` subdirectory of the NI-VXI directory as shown below. `INSTALL` also adds a line that keeps Windows from using the VXI space for its own purposes.

```
[386Enh]
DEVICE= <NI-VXI directory>\WIN\NIVXIPHM.386
EMMexclude= A000-EFFF

[NIVXI]
NIVXIPATH= <NI-VXI directory>
Load Data= 6bytes
```

When `INSTALL` modifies the file, it saves the old file as `SYSTEM.BAK` for safety purposes. If you choose not to let `INSTALL` modify your `SYSTEM.INI` file, it creates a file called `SYSTEM.VXI` in the NI-VXI directory, which you can use as a guide to make the changes yourself.

Modifying the WIN.INI File

If you choose to let `INSTALL` modify your `WIN.INI` file, it adds the following lines to the `WIN.INI` file.

```
[NIVXI]
NIVXIPATH= <NI-VXI directory>
```

In this situation, the `NIVXIPATH` variable is used by the `NIVXI.DLL` dynamic link library in addition to the application programs that come with NI-VXI (for example, `RESMAN.EXE`) to locate the NI-VXI configuration and help files.

When `INSTALL` modifies the file, it saves the old file as `WIN.BAK` for safety purposes. If you choose not to let `INSTALL` modify your `WIN.INI` file, it creates a file called `WIN.VXI` in the NI-VXI directory, which you can use as a guide to make the changes yourself.

Completing the Software Installation

After you execute `INSTALL`, you should exit Windows and reboot your machine (or at least run `AUTOEXEC.BAT`) to make your system aware of the NI-VXI directory.

After the NI-VXI software is installed, run `VXIINIT.EXE` and then `RESMAN.EXE.`

Configuring the NI-VXI Software

Run `VXIINIT` to display the AT-MXI configuration settings. The AT-MXI is configured in hardware as the Resource Manager (Logical Address 0) and a Message-Based device, and cannot be changed. The board's default configuration is to be the MXIbus System Controller, not a MXIbus fair requester, with a base I/O address of 340h, a master window base of D000h, and no VXI Shared RAM. Although the NI-VXI software is designed for VXI systems, you can normally use the default configuration information for your AT-MXI installed in a VME system. To change the current settings, or to view the configuration options, run one of the VXI Resource Editor programs, either `VXIEDIT` or `VXITEDIT`.

You may find the `VXIEDIT` program easier to use because of its structure. `VXIEDIT` runs in DOS or the Windows DOS shell and uses a series of interactive graphical screens to display information for the various editors. You can use `VXITEDIT` to accomplish the same tasks by entering commands at the keyboard. `VXITEDIT` is a Windows application that you can run directly in the Windows environment. You may prefer to use `VXITEDIT` if you do not want to take the extra step to enter DOS to reach `VXIEDIT`. You can execute `VXITEDIT` from DOS, Windows, or the Windows DOS shell.

Note: ***You MUST run*** `VXIEDIT` ***or*** `VXITEDIT` ***and use the Non-VXI Device Editor to configure the VME addresses, interrupt requirements, and other details about any VME devices you want to access from your PC AT. If you do not use*** `VXIEDIT` ***or*** `VXITEDIT`***, the Resource Manager cannot integrate your VME devices into the system.***

You can use `VXIEDIT` or `VXITEDIT` to edit the description of the AT-MXI or any other device that is installed in the system. Using either of these programs, you can easily modify the configuration tables required for VXIbus and MXIbus operation. You can also use either program to configure interrupts, triggers, and the utility bus for single- or multiple-mainframe system configurations.

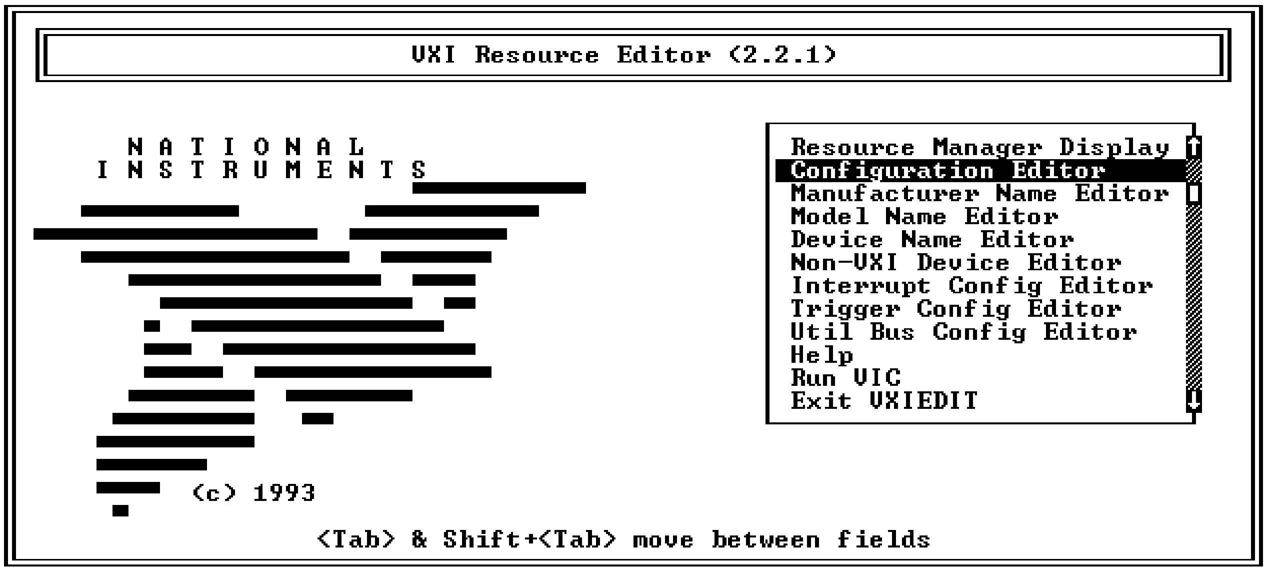
For more details on how to use the `VXIEDIT` program, refer to the *NI-VXI DOS Utilities Reference Manual*. For more details on how to use the `VXITEDIT` program, refer to the *NI-VXI Text Utilities Reference Manual.*

Using VXIEDIT

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

To run `VXIEDIT` from any directory, make sure that the `PATH` environment variable has the destination directory of the NI-VXI software added to it. This program uses the different configuration files (`*.CFG`), table files (`*.TBL`), and help files (`*.HLP`) in its execution. Be sure that the environment variable `NIVXIPATH` is set to the destination directory of the NI-VXI software for proper execution. The default destination directory pathname used by the program is `C:\NIVXI`.

You can run `VXIEDIT` when you are in DOS. If you are in Windows, you must first click on the **DOS Prompt** icon to enter DOS before running `VXIEDIT`. Figure 5-1 shows the main menu of the `VXIEDIT` program.



```
VXI Resource Editor (2.2.1)

  N A T I O N A L
I N S T R U M E N T S

(c) 1993

<Tab> & Shift+<Tab> move between fields
```

Figure 5-1. VXIEDIT Main Menu

You can use all of the editors in VXI systems, but some of them do not apply to VME systems.

- The Manufacturer Name Editor, Model Name Editor, Device Name Editor, and Trigger Configuration Editor are specific to VXI systems and are not used in VME systems.
- The Interrupt Configuration Editor and Utility Bus Configuration Editor can be used in both VME and VXI systems, but are only applicable in multiple-mainframe system configurations.
- The Resource Manager Display, Configuration Editor, and Non-VXI Device Editor are used in both VXI and VME systems.
  - Use the Resource Manager Display to examine information about any of the devices in your system after the Resource Manager program has executed.
  - Use the Configuration Editor to configure details of the AT-MXI hardware configuration.
  - Use the Non-VXI Device Editor to configure the address requirements of VME devices in your system so that the Resource Manager program can automatically set up the MXI link.

The following sections describe how to modify and/or view the configuration information for the AT-MXI board. To configure VME devices into your VXI or VME system, refer to the *VME (Non-VXI) Device Configuration* section later in this chapter.

Configuration Editor

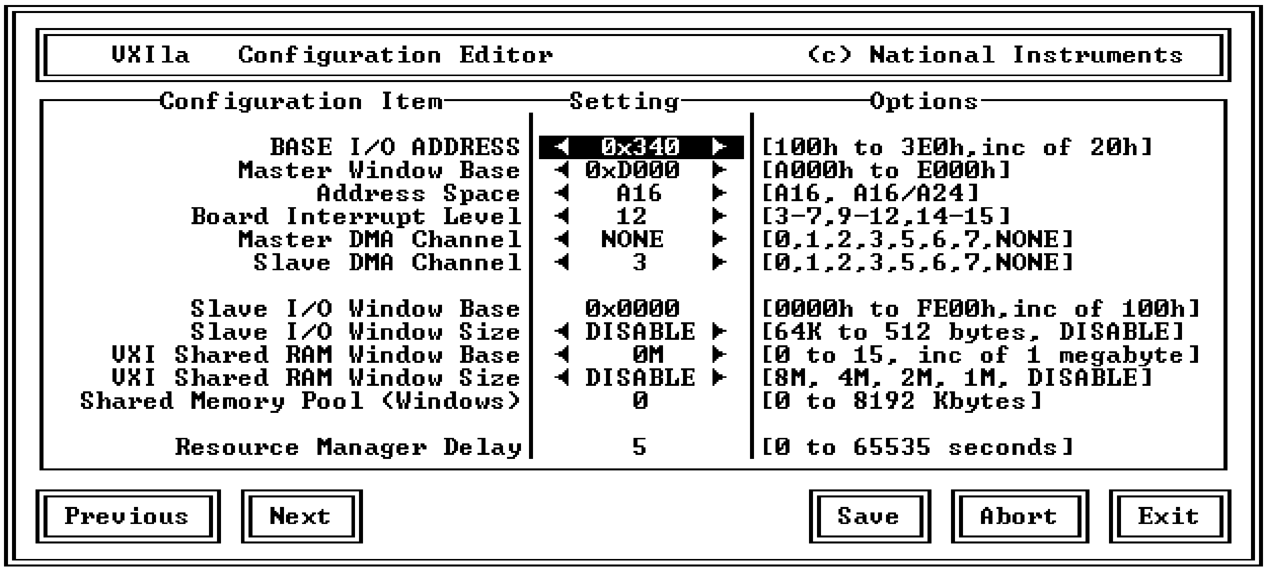
When you select the Configuration Editor from the main menu, the Logical Address Configuration Editor menu (shown in Figure 5-2) is displayed. The following sections describe the various configuration options available under the Logical Address, Bus, and Device Configuration menus.

Select **Next** and **Previous** to move among the three configuration editors. You can change the default settings to any value within the range shown to the right of each item by using the left and right arrow keys.

You can save your changes by selecting **Save**. Saving the changes updates files `VXILA.CFG`, `VXIBUS.CFG`, `VXIDEV.CFG`, and `DVXI.CFG`. These files are used by `VXIINIT.EXE` to update the soft-configured settings, including board interrupts, shared RAM size, VXI register contents, and bus interface. You can exit the program at any time without saving any changes you have made by selecting **Abort**.

Logical Address Configuration

Figure 5-2 shows the Logical Address Configuration Editor (VXIla) menu.



```
VXIla   Configuration Editor                    (c) National Instruments

            Configuration Item          Setting           Options

             BASE I/O ADDRESS    ◄  0x340  ►  [100h to 3E0h,inc of 20h]
           Master Window Base    ◄ 0xD000  ►  [A000h to E000h]
                Address Space    ◄   A16   ►  [A16, A16/A24]
         Board Interrupt Level   ◄   12    ►  [3-7,9-12,14-15]
           Master DMA Channel    ◄  NONE   ►  [0,1,2,3,5,6,7,NONE]
            Slave DMA Channel    ◄    3    ►  [0,1,2,3,5,6,7,NONE]

         Slave I/O Window Base     0x0000     [0000h to FE00h,inc of 100h]
         Slave I/O Window Size   ◄ DISABLE ►  [64K to 512 bytes, DISABLE]
     VXI Shared RAM Window Base  ◄    0M   ►  [0 to 15, inc of 1 megabyte]
     VXI Shared RAM Window Size  ◄ DISABLE ►  [8M, 4M, 2M, 1M, DISABLE]
   Shared Memory Pool (Windows)       0       [0 to 8192 Kbytes]

        Resource Manager Delay        5       [0 to 65535 seconds]

 Previous   Next                              Save   Abort   Exit
```

Figure 5-2. AT-MXI Logical Address Configuration Editor

Table 5-1 describes the characteristics of the options available under the Logical Address Configuration menu. Normally, for VME systems you do not have to modify the default configuration information.

Table 5-1. Logical Address Configuration Characteristics

Characteristic	Description	Default Value
AT-MXI Base I/O Address	Indicates the I/O Address of the AT-MXI configuration registers. This value must correspond to the selected jumper settings on the board. Range = 100h to 3E0h in increments of 20h (hex)	0x340
Master Window Base	A 64 KB memory-mapped I/O window, which the AT-MXI board uses to access the VXIbus or VMEbus. Range = A000h to E000h in increments of 1000h (hex)	0xD000
Address Space	Indicates the address spaces used by the AT-MXI. A16 only A16/A24	A16 only
AT-MXI Board Interrupt Level	Indicates the interrupt level used by the board to interrupt the processor. This value must correspond to the jumper settings on the AT-MXI board. Range = 3, 4, 5, 6, 7, 9, 10, 11, 12, 14, 15	12
Master DMA Channel	Indicates the DMA channel that the AT-MXI uses to perform master mode block transfers onto the VXIbus or VMEbus. This value must correspond to the jumper settings on the AT-MXI board. The NONE option can be selected with any jumper settings. Range = 0, 1, 2, 3, 5, 6, 7, NONE **Note:** ***This field is currently disabled.***	NONE

(continues)

Table 5-1. Logical Address Configuration Characteristics (Continued)

Characteristic	Description	Default Value
Slave DMA Channel	This indicates the slave DMA channel to be used in all slave accesses to the VXI or VME Shared RAM in A24 space or the PC I/O in A16 space. This value must correspond to jumper settings on the AT-MXI board. The NONE option can be selected with any jumper settings. Range = 0, 1, 2, 3, 5, 6, 7, NONE **Note:** ***If the NONE option is selected, slave accesses to VXI or VME Shared RAM and to PC I/O space will not succeed. Accesses to VXI-defined registers on the AT-MXI board, however, remain unaffected.***	3
Slave I/O Window Base	The I/O space in the PC with the AT-MXI can be accessed in A16 space over the VXIbus or VMEbus. This field sets up the base address for these accesses. Range = 0000h to FE00h in increments of 100h (hex) **Note:** ***Never select a base address above C000h. Selecting an address above C000h can conflict with the existing VXI and VME devices in the system. Because the VXIbus specification does not address automatic configuration of the lower 48 KB of A16 space,*** `RESMAN` ***does not handle any device requesting memory in A16 space, including the AT-MXI.*** `RESMAN` ***does not give a warning in the event of an address conflict in A16 space. Also, the opening of A16 windows on frame extenders in the system for A16 accesses depends only on the logical addresses of the devices in the system.***	0x0000

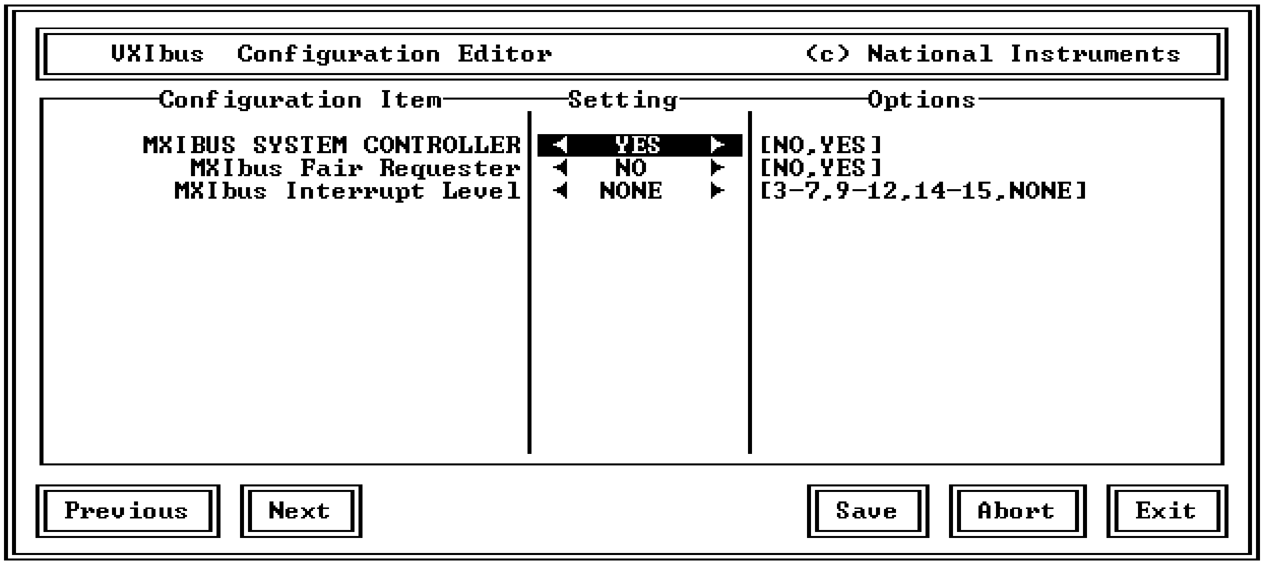
(continues)

Table 5-1. Logical Address Configuration Characteristics (Continued)

Characteristic	Description	Default Value
Slave I/O Window Size	Indicates the size of the window in A16 space for accesses to the PC I/O space over the VXIbus or VMEbus. Range = 64 KB, 32 KB, 16 KB, 8 KB, 4 KB, 2 KB, 1 KB, 512 bytes, DISABLE	DISABLE
Shared RAM Window Base	Indicates the offset in the local RAM above which the RAM is shared over the VXIbus or VMEbus in A24 space. If the **Address Space** selection for the AT-MXI is A16 only, this field is ignored. Range = 0 to 15, in increments of 1 MB	0 MB
Shared RAM Window Size	Indicates the size of VXI or VME Shared RAM in the A24 space. If the **Address Space** selection for the AT-MXI is A16 only, this field is ignored. Range = 8 MB, 4 MB, 2 MB, 1 MB, DISABLE **Note:** ***If the* Address Space *selection is* A16/A24*, you need to select a value other than* DISABLE *for this field. You must also ensure that the Slave DMA channel is set correctly.***	DISABLE
Shared Memory Pool (Windows)	Indicates the size of memory in kilobytes that is allocated on Windows startup to be used exclusively by `VXImemAlloc` function calls from both DOS and Windows applications. Range = 0 to 8192 KB	0 KB
Resource Manager Delay	Time in seconds that the Resource Manager waits before accessing the configuration registers of any other device in the system. Range = 0 s to 65535 s	5 s

Bus Configuration

Figure 5-3 shows the Bus Configuration Editor (VXIbus) menu.



```
VXIbus  Configuration Editor              (c) National Instruments

Configuration Item          Setting          Options

 MXIBUS SYSTEM CONTROLLER   ◄   YES   ►   [NO,YES]
     MXIbus Fair Requester   ◄   NO    ►   [NO,YES]
    MXIbus Interrupt Level   ◄   NONE  ►   [3-7,9-12,14-15,NONE]
```

Figure 5-3. AT-MXI Bus Configuration Editor

Table 5-2 describes the characteristics of the options available under the Bus Configuration menu.

Table 5-2. Bus Configuration Characteristics

Characteristic	**Description**	**Default Value**

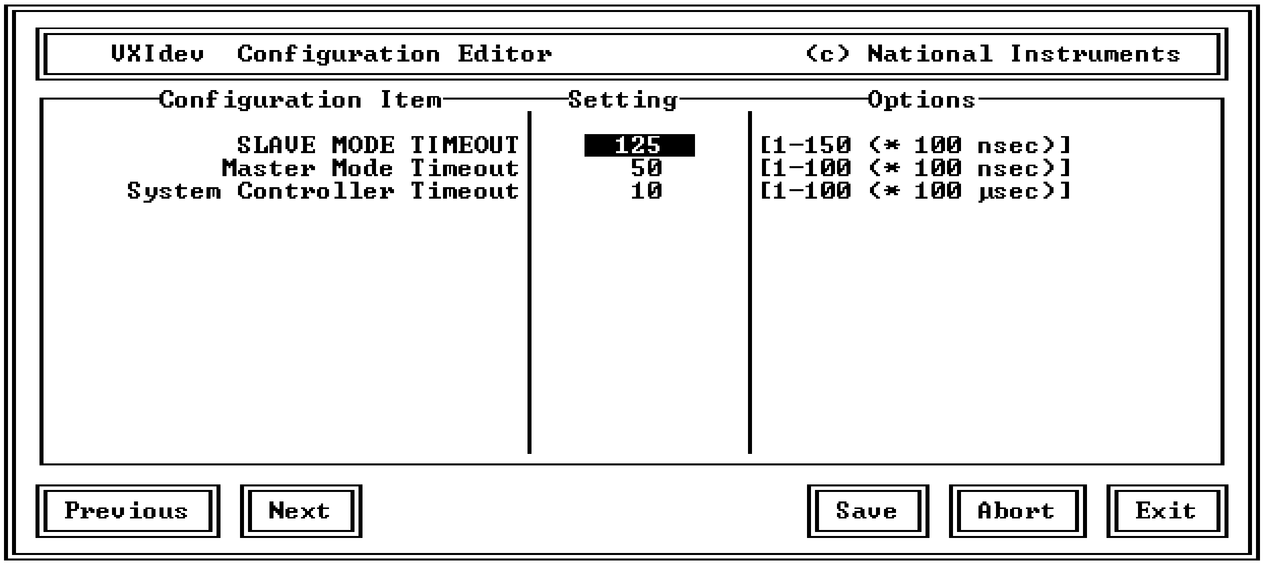
MXIbus System Controller	A MXIbus link must have a single device that is responsible for MXIbus interrupt and bus arbitration as well as bus timeouts. This device is referred to as the MXIbus System Controller and is always the first device in the MXIbus daisy-chain. The AT-MXI is always designated as the MXIbus System Controller when used in this kit and should *not* be changed.
	
	NO = Not MXIbus System Controller
	YES = MXIbus System Controller
		YES

MXIbus Fair Request	MXIbus fair requesters give all MXIbus devices equal opportunity to become bus masters and transfer data, regardless of their position within the MXIbus daisy-chain. MXIbus devices that are not fair requesters have a fixed priority scheme characterized by their relative position within the MXIbus daisy-chain. Devices closest to the MXIbus System Controller have the highest priority. The default configuration programs the AT-MXI to be an unfair requester and should only be changed if additional MXIbus devices are added to the system. Refer to the user manual included with your additional MXIbus devices for more information.
	
	NO = Not a MXIbus Fair Requester
	YES = MXIbus Fair Requester
		NO

MXIbus Interrupt Level	Indicates the interrupt level used by the MXI IRQ to interrupt the processor. This value must correspond to jumper settings on the AT-MXI board; however, the NONE option can be selected with any jumper settings. If the NONE option is selected, the same interrupt level as used for AT-MXI board interrupts is used for this interrupt. Range = 3, 4, 5, 6, 7, 9, 10, 11, 12, 14, 15, or NONE	NONE

Device Configuration

Figure 5-4 shows the Device Configuration Editor (VXIdev) menu.



```
VXIdev  Configuration Editor               (c) National Instruments

Configuration Item          Setting          Options

      SLAVE MODE TIMEOUT      125     [1-150 (* 100 nsec)]
      Master Mode Timeout      50     [1-100 (* 100 nsec)]
System Controller Timeout      10     [1-100 (* 100 µsec)]
```

Figure 5-4. AT-MXI Device Configuration Editor

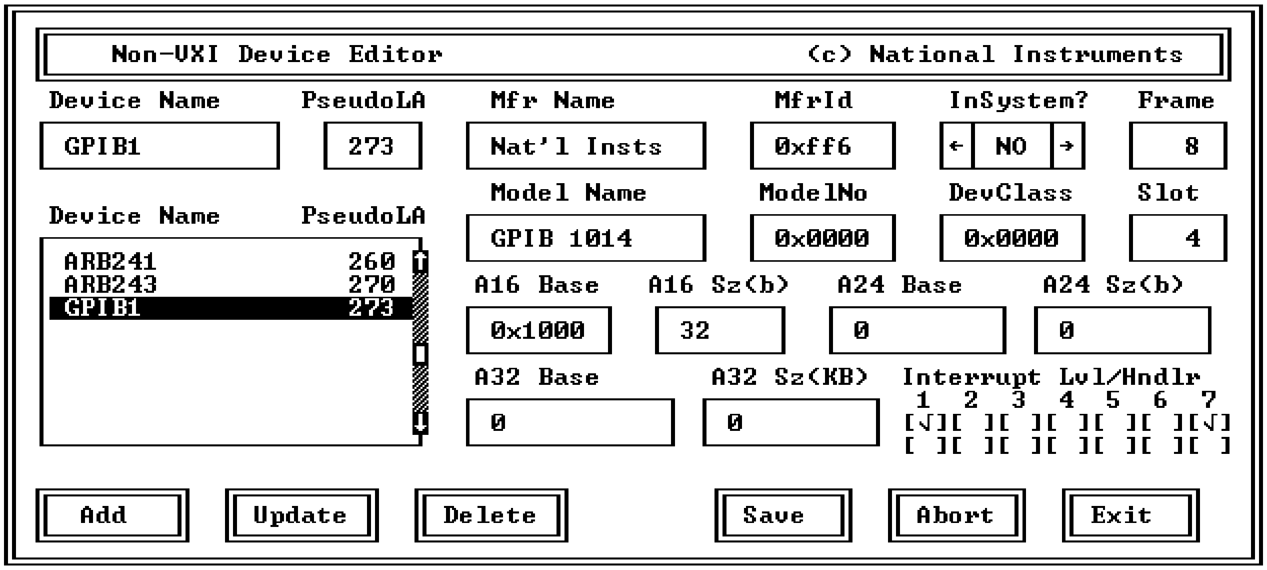
Table 5-3 describes the characteristics of the options available under the Device Configuration menu.

Table 5-3. Device Configuration Characteristics

Characteristic	Description	Default Value
Slave Mode Timeout	This field gives the maximum amount of time a slave access to the VXI or VME Shared RAM may hold the PC bus. The units are in hundreds of nanoseconds.	125 (12.5 µs)
Master Mode Timeout	This field gives the maximum amount of time a master access from the PC to the MXIbus may wait until it has to be retried. The units are in hundreds of nanoseconds.	50 (5.0 µs)
System Controller Timeout	This field gives the maximum amount of time a master access from the PC to the MXIbus may be allowed to continue without being terminated as a bus error. The units are in hundreds of microseconds. **Note:** ***If the* MXIbus System Controller *field in the Bus Configuration Editor is set to* NO*, this field is ignored.***	10 (1.0 ms)

VME (Non-VXI) Device Configuration

To add or modify the address space and interrupt requirements for VME devices in your system, select the Non-VXI Device Editor from the main menu. Figure 5-5 shows the configuration available from this menu.



```
Non-VXI Device Editor                    (c) National Instruments

Device Name       PseudoLA    Mfr Name       MfrId       InSystem?     Frame
GPIB1             273         Nat'l Insts    0xff6       ← NO →            8
                              Model Name     ModelNo     DevClass      Slot
Device Name       PseudoLA    GPIB 1014      0x0000      0x0000            4
ARB241                 260    A16 Base    A16 Sz(b)   A24 Base      A24 Sz(b)
ARB243                 270    0x1000      32          0             0
GPIB1                  273    A32 Base         A32 Sz(KB)   Interrupt Lvl/Hndlr
                                                             1  2  3  4  5  6  7
                              0                0            [√][ ][ ][ ][ ][ ][√]
                                                            [ ][ ][ ][ ][ ][ ][ ]

  Add      Update      Delete          Save      Abort      Exit
```

Figure 5-5. Non-VXI Device Editor

The NI-VXI software needs information about the attributes of the VME devices so that the Resource Manager can integrate them into the system. For help with the Non-VXI Device Editor, select the **Non-VXI Device Editor** field on the main menu and press <RETURN> or <Enter>. Refer also to the *NI-VXI DOS Utilities Reference Manual* for more information about this editor.

When you use `VXIEDIT` to configure your non-VXI (VME) devices, be sure to specify a unique pseudo-logical address for each VME device. The pseudo-logical address can be any value between 256 and 511 inclusive. We recommend that you start with 256 for your first device, and then give each subsequent device the next higher number. Assigning pseudo-logical addresses to your VME devices ensures that the Resource Manager program recognizes your VME devices when it configures your interface hardware.

You also need to enter device names, manufacturer names, model names, and other identification information for your VME devices. The **Frame** field should contain the logical address of the VXI-MXI or VME-MXI in the chassis. The default logical address is 1. A16 VME devices should be configured to occupy memory below C000h in A16 space. This is not a requirement, but any VME devices above C000h can interfere with any VXI devices in your system and make the integration more difficult. Be sure to configure the **InSystem** parameter to instruct the Resource Manager that this particular VME device is indeed installed in your system.

Remember that the Resource Manager will use the configuration information for only those devices that you indicated are in the system. When the Resource Manager executes, it displays information regarding the configuration of the interface hardware and your VME devices by referencing their assigned pseudo-logical addresses. After the Resource Manager has executed and configured your interface hardware, your application can retrieve the Resource Manager information about the VME devices in your system that you entered using the Non-VXI Device Editor. To retrieve this information, your application program will use the pseudo-logical address value that you assigned to your VME device(s) as a parameter in the NI-VXI function calls. For more information, refer to the chapter on system configuration functions in your software reference manual.

You can enter configuration information for each VME device in your system either individually as unique VME devices, or collectively by considering the overall address space requirements of your VME devices as one logical VME device. By using `VXIEDIT` to configure the address space requirements, the Resource Manager can automatically enable the hardware windows through which your computer can access the corresponding VME addresses where your VME devices reside. You can also run `VXIEDIT` at any time to examine the software configurations.

Select **Add** to add a new entry to the alphabetical list of device names. After modifying an entry, select **Update** to replace the original entry. To delete an entry, select the entry in the list of device names and then select **Delete**. Select **Save** to save the information that was changed or select **Abort** to ignore the modifications. **Exit** returns to the main menu. Make sure to set or clear the **InSystem?** field of an entry (by selecting yes or no), depending on whether that device is actually present in your system.

The example in Figure 5-5 shows a VME memory device that is not in the system, but would require 32 bytes of memory in A16 space. The pseudo-logical address is set between 256 and 511. Its **Frame** field is set to the logical address of the VME-MXI in the chassis, which in this case is 8 (the default logical address for the VME-MXI in your kit is 1).

Run the Resource Manager to automatically configure the system. Verify that the Resource Manager display reflects all the changes you have made to accommodate the VME devices in your system.

Exiting VXIEDIT and Reinitializing the Hardware

Select **Exit** or press the <Esc> key to quit the `VXIEDIT` program. If you changed any of the software configuration information, you are prompted to save your changes before exiting the configuration menu. Saving the configuration information updates the configuration files `VXILA.CFG`, `VXIBUS.CFG`, `VXIDEV.CFG`, and `DVXI.CFG`. Run `VXIINIT.EXE` to reinitialize the hardware according to the new settings.

If you changed any of the software configuration settings from the default settings, record the new settings on the *VXI/VME-AT2010 Hardware and Software Configuration Form* in Appendix E.

Using VXITEDIT

As an alternative to `VXIEDIT`, you can use `VXITEDIT`, a text-based VXI resource editor that is functionally equivalent to `VXIEDIT`. `VXITEDIT` is a Windows application and you can run it directly in the Windows environment. `VXITEDIT` uses the `NIVXIPATH` variable set in the `WIN.INI` file to locate the configuration files, table files, and help files. When `NIVXIPATH` is not specified in `WIN.INI`, it assumes the default directory `C:\NIVXI`.

To run `VXITEDIT`, click on the **VXITEDIT** icon in the NI-VXI group folder in the **Program Manager**. Refer to the *NI-VXI Text Utilities Reference Manual* for further information on the use of the `VXITEDIT` program.

Exiting VXITEDIT and Reinitializing the Hardware

To exit `VXITEDIT`, type `exit` or the number `11` when the main menu is displayed. If you changed any information, you are prompted to save your changes before exiting the menu. Saving the configuration information updates the configuration files `VXILA.CFG`, `VXIBUS.CFG`, `VXIDEV.CFG`, and `DVXI.CFG`. Run `VXIINIT.EXE` to reinitialize the hardware according to the new settings.

If you changed any of the software configuration settings from the default settings, record the new settings on the *VXI/VME-AT2010 Hardware and Software Configuration Form* in Appendix E.

Chapter 6
Using NI-VXI with Windows

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. Remember that you must run the `VXIINIT` initialization program before performing any VXI operations and after each computer reset. `VXIINIT` is placed into your `autoexec.bat` file by default. 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 your software reference manual—either the *NI-VXI Software Reference Manual for C*, or the *NI-VXI C Software Reference Manual for VME*—for overviews of NI-VXI and detailed descriptions of the NI-VXI functions. The VME manual does not document the following function classes, which are applicable to VXI systems only:

- 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 program (`VIC` from DOS or `VICTEXT` from Windows) to write to and read from your instruments. Both programs display the status of your VXI transactions and inform you of any errors that occur. Refer to the *NI-VXI DOS Utilities Reference Manual* on how to use `VIC` from DOS and to learn about its features, and to the *NI-VXI Text Utilities Reference Manual* on how to use `VICTEXT` from Windows and to learn about its features.

Caution: ***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 the macro*** `VXIWIN` ***(if you are developing a Windows application) or the macro*** `VXIDOS` ***(if you are developing a DOS application) in your makefile/project. Refer to the example programs 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 the *Installing the Software* section in Chapter 5. Also refer to your software reference manual for additional examples.

Programming Considerations

The following paragraphs contain information for you to consider when developing Windows applications that use the NI-VXI bus interface software.

Memory Model

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 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 can not only 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 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 trying to use the NI-VXI library.

However, you do need to be careful in cases when you have multiple applications using the low-level VXIbus 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 Owner Access first before attempting to perform low-level VXIbus access with `VXIpeek()` or `VXIpoke()`. This will guarantee exclusive use of the window for your application. 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.

Low-Level Access Functions

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. Due to the possible timeout from the ISA bus, it may be necessary to perform retry operations to successfully complete the operation. For more information, refer to the chapter on low-level VXIbus or VMEbus access functions in your software reference manual.

Local Resource Access Functions

By using `VXIEDIT` or `VXITEDIT`, you can allow the AT-MXI to share the PC memory and I/O space with the VXI/VME system. Refer to the *NI-VXI DOS Utilities Reference Manual* or the *NI-VXI Text Utilities Reference Manual* for more information on setting these parameters.

Remember that choosing to share these resources with the VXI/VME system does not mean that the resources are reserved. For example, if you share the lower 8 MB of RAM with the VXI/VME system, you still have DOS and Windows running in that memory. To reserve the memory, you must use `VXImemAlloc()`. See the *NI-VXI C Reference Manual* for more information on this function.

Another factor to consider is that although you may have selected to share 8 MB, you must also inform Windows how much memory to set aside for possible `VXImemAlloc` calls. You can inform Windows using the **Shared Memory Pool (Windows)** option in `VXIEDIT` and `VXITEDIT`. 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.

System Configuration Functions

The System Configuration functions provide the lowest level initialization of your VME interface. 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. Then you can modify the sample program to try out different things.

The sample C program for the Microsoft C compiler is in the `\nivxi\win\msc\examples` directory and the sample C program for the Borland C compiler is in the `\nivxi\win\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.

The following symbols are usually required. You must define them when using the Microsoft C or Borland C compiler.

`VXIWIN` This means that the application is a Windows application. You can use the same NI-VXI header files to compile DOS programs by defining `VXIDOS` instead.

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

`__PROTOTYPES__` This means that you are using an ANSI C-compliant compiler. If you are using a C++ compiler or if your compiler defines the `__STDC__` symbol, this symbol is automatically defined.

The following symbols are optional.

`BINARY_COMPATIBLE`	This makes the application binary compatible with embedded VXI controllers, such as the VXIpc-486 series of embedded controllers. Using this option may cause a slight performance degradation when using low-level VXIbus access functions.
`__I386__`	If you have an 80386 or higher end machine and you have the compiler set up to generate 80386 instructions, you can define this symbol to achieve higher performance when using the 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 __PROTOTYPES__
#define BINARY_COMPATIBLE
#define __I386__
.
.
.
#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 /D__PROTOTYPES__ /DBINARY_COMPATIBLE /D__I386__
```

For the Borland C compiler:

```
-DVXIWIN:__PROTOTYPES__;BINARY_COMPATIBLE;__I386__
```

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

Appendix A Specifications

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

AT-MXI

The following pages list the specifications for the AT-MXI module.

Capability Codes

MXIbus

Capability Code	Description
MA32	Master Mode A32, A24 and A16 addressing
MBLT	Master Mode block transfers
SA24	Slave Mode A24 and A16 addressing
SBLT	Slave Mode block transfers
MD16	Master Mode D16 and D08 data sizes
SD16	Slave Mode D16 and D08 data sizes
SC	Optional MXIbus System Controller
FAIR	Can be a fair MXIbus requester
LOCK	Can lock the MXIbus for indivisible transfers
TERM	Can terminate the MXIbus

ISA Bus

Capability Code	Description
AM	Can function as an AT Alternate Master
LOCK	Can lock the AT bus for indivisible transfers
DMA16	Supports D08 or D16 DMA transfers
INT	Can interrupt on the PC AT bus

Requirements

Characteristic	Specification
Memory Space	64 KB
I/O Space	32 B

Environmental

Characteristic	Specification
Component Temperature	0° to 70° C operating; -55° to 150° C storage
Relative Humidity	0% to 95% noncondensing, operating; 0% to 100% noncondensing, storage
Emissions	FCC Class A
Safety	Not applicable
Shock and Vibration	Not applicable

Physical

Characteristic	Specification
Board Dimensions	Standard full-length AT-height board 339.72 mm by 121.92 mm (13.36 in. by 4.8 in.)
Connectors	Single fully implemented MXIbus connector
Slot Requirements	Single AT (ISA) slot
MTBF	Contact Factory

Electrical

Source	Typical	Direct Current (max)
+5 VDC	3.3 A	4.4 A

Timing

Master Mode

Transfer Type	Transfer Rate
Write	530 ns
Read	430 ns
Block Write	290 ns
Block Read	190 ns

Slave Mode

Transfer Type	Transfer Rate
Write	840 ns
Read	840 ns
Block Write	590 ns
Block Read	590 ns

Other

Transfer Type	Transfer Rate
Daisy-Chain Delay (Passing GIN to GOUT or GOUT generation from System Controller)	120 ns

VXI-MXI

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

Capability Codes

VMEbus

Capability Code	Description
MA32, MA24, MA16	Master Mode A32, A24, and A16 addressing
SA32, SA24, SA16	Slave Mode A32, A24, and A16 addressing
MD32, MD16, MD08(EO)	Master Mode D32, D16, and D08 data sizes
SD32, SD16, SD08(EO)	Slave Mode D32, D16, and D08 data sizes
MBLOCK	Master Mode block transfers
SBLOCK	Slave Mode block transfers
MRMW	Master Mode Read/Modify/Write
SRMW	Slave Mode Read/Modify/Write
PRI	Prioritized arbitration
ROR	Release on Request bus requester
IH	Interrupt Handler
IR	Interrupt Requester
ROAK	Release on Acknowledge interrupter
BTO	Bus Timeout
SC	Optional VMEbus System Controller
IACK	IACK daisy-chain driver

VXIbus

Capability Code	Description
TRIG+1	Supports TTLTRIG0:7 and ECLTRIG0:1 trigger lines and full protocol operations for each. The VXI-MXI may participate in only one protocol operation at a time.

MXIbus

Capability Code	Description
MA32, MA24, MA16	Master Mode A32, A24, and A16 addressing
SA32, SA24, SA16	Slave Mode A32, A24, and A16 addressing
MD32, MD16, MD08(EO)	Master Mode D32, D16, and D08 data sizes
SD32, SD16, SD08(EO)	Slave Mode D32, D16, and D08 data sizes
MBLOCK	Master Mode block transfers
SBLOCK	Slave Mode block transfers
SC	Optional MXIbus System Controller
FAIR	Optional MXIbus fair requester
TERM	Can accept MXIbus termination resistors
IH	Interrupt Handler
IR	Interrupt Requester

Requirements

Characteristic	Specification
A16 Space	64 B

Environmental

Characteristic	Specification
Component Temperature	0° to 70° C operating; -40° to 85° C storage
Relative Humidity	10% to 90% noncondensing, operating; 0% to 95% noncondensing, storage
Airflow	3.5 liters/s for 10° rise
Emissions	FCC Class A
Safety	Not applicable
Shock and Vibration	Not applicable

Physical

Characteristic	Specification
Board Dimensions	Fully enclosed, shielded VXI C-size board 233.35 mm by 340 mm (9.187 in. by 13.386 in.)
Connectors	Single fully implemented MXIbus connector Single INTX connector (on boards equipped with optional INTX daughter card)
Slot Requirements	Single VXI C-size slot
Compatibility	Fully compatible with VXI specification
VXI Keying Class	Class 1 TTL
MTBF	Contact Factory

Electrical

	DC Current Ratings		
Source	**Typical**	**Maximum**	**Dynamic Current**
+5 VDC	5.25 A	6.7 A	0.67 A
-5.2 VDC	300 mA	400 mA	50 mA
-2 VDC	100 mA	125 mA	20 mA

Timing

Master Mode

Transfer Type	Transfer Rate
Write	675.5 ns
Read	625.5 ns
Block Write	320 ns
Block Read	270 ns

Slave Mode

Transfer Type	Transfer Rate
Write	381 ns
Read	381 ns
Block Write	238 ns
Block Read	238 ns

Other	
Transfer Type	**Transfer Rate**
Daisy-Chain Delay (Passing GIN to GOUT or GOUT generation from System Controller)	120 ns

VME-MXI

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

Capability Codes

VMEbus

Capability Code	Description
MA32, MA24, MA16	Master Mode A32, A24, and A16 addressing
SA32, SA24, SA16	Slave Mode A32, A24, and A16 addressing
MD32, MD16, MD08(EO)	Master Mode D32, D16, and D08 data sizes
SD32, SD16, SD08(EO)	Slave Mode D32, D16, and D08 data sizes
MBLOCK	Master Mode block transfers
SBLOCK	Slave Mode block transfers
MRMW	Master Mode Read/Modify/Write
SRMW	Slave Mode Read/Modify/Write
PRI	Prioritized arbitration
ROR	Release on Request bus requester
IH	Interrupt Handler
IR	Interrupt Requester
ROAK	Release on Acknowledge interrupter
BTO	Bus Timeout
SC	Optional VMEbus System Controller
IACK	IACK daisy-chain driver

MXIbus

Capability Code	Description
MA32, MA24, MA16	Master Mode A32, A24, and A16 addressing
SA32, SA24, SA16	Slave Mode A32, A24, and A16 addressing
MD32, MD16, MD08(EO)	Master Mode D32, D16, and D08 data sizes
SD32, SD16, SD08(EO)	Slave Mode D32, D16, and D08 data sizes
MBLOCK	Master Mode block transfers
SBLOCK	Slave Mode block transfers
SC	Optional MXIbus System Controller
FAIR	Optional MXIbus fair requester
TERM	Can accept MXIbus termination resistors
IH	Interrupt Handler
IR	Interrupt Requester

Requirements

Characteristic	Specification
A16 Space	64 B

Environmental

Characteristic	Specification
Component Temperature	0° to 70° C operating; -40° to 85° C storage
Relative Humidity	10% to 90% noncondensing operating; 0% to 95% noncondensing storage
Emissions	FCC Class A
Safety	Not applicable
Shock and Vibration	Not applicable

Physical

Characteristic	Specification
Board Dimensions	233.35 mm by 160 mm (9.187 in. by 6.299 in.)
Connectors	Single fully implemented MXIbus connector Single INTX connector (on boards equipped with optional INTX daughter card)
Slot Requirements	Single slot
Compatibility	Fully compatible with VMEbus specification
MTBF	Contact Factory

Electrical

Source	Typical	Maximum
+5 VDC	5.25 A	7.0 A

Timing

Master Mode

Transfer Type	Transfer Rate
Write	675.5 ns
Read	625.5 ns
Block Write	320 ns
Block Read	270 ns

Slave Mode

Transfer Type	Transfer Rate
Write	381 ns
Read	381 ns
Block Write	238 ns
Block Read	238 ns

Other Transfer Type	Transfer Rate
Daisy-Chain Delay (Passing GIN to GOUT or GOUT generation from System Controller)	120 ns max

Appendix B
NI-VXI Software Overview

This appendix describes the programs and files located on the NI-VXI distribution diskettes.

Main Programs and Files

The main programs and files of the NI-VXI software package are found in the `C:\NIVXI` directory.

- `VXIINIT.EXE` is the AT-MXI initialization program. This is a DOS program that initializes the board interrupts, shared RAM, VXI register configurations, and bus configurations. `VXIINIT.EXE` must be executed from DOS (or the Windows DOS shell). It is typically included in the DOS batch file `AUTOEXEC.BAT` so that the AT-MXI 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, or the Windows DOS shell, although the latter is not recommended. `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 or the Windows DOS shell, although the latter is not recommended. If you do use `VIC` from the Windows 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 DOS Utilities Reference Manual.*

- `VICTEXT.EXE` is a text-based interactive control program that is functionally equivalent to `VIC.EXE`. You can execute `VICTEXT` from DOS, Windows, or the Windows 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 the Windows DOS shell. You use the Non-VXI Device Editor in this program 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 AT 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 this program 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 DOS Utilities Reference Manual.*

- `VXITEDIT.EXE` is the text-based VXI resource editor program that is functionally equivalent to `VXIEDIT.EXE`. You can execute `VXITEDIT` from DOS, Windows, or the Windows 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.

Additional Programs and Files

The `C:\NIVXI\HLP` directory contains various help files used by the `VIC.EXE` and `VXIEDIT.EXE` programs.

The `C:\NIVXI\TBL` directory contains the following files.

- `MFNAMEID.TBL` contains the database of manufacturer names and their ID numbers. This file is primarily used in VXI systems because VXI devices have manufacturer IDs and VME devices do not.

- `MODEL.TBL` contains the database of model names, manufacturer names, and the model codes numbers. This file is primarily used in VXI systems.

- `DEVICE.TBL` contains the database of device names, manufacturer names, model names, and frame and slot associations for devices in the system. This file is primarily used in VXI systems.

- `NONVXI.TBL` contains the data base for all non-VXI (VME) devices in the system. This file is used for VXI/VME systems and describes the address space and interrupt requirements of the VME devices in the system. `RESMAN` uses this file to determine how to open the windows to the VMEbus. Use the Non-VXI Device Editor to configure the address space requirements for your VME devices so `RESMAN` can configure the hardware windows properly.

- `INTCFG.TBL` contains the system interrupt configuration information. In VXI systems, interrupts can be assigned dynamically. In VME systems, however, interrupts are statically configured. This file also contains interrupt mapping information that can be used in a VME system, but only in a multimainframe situation.

- `TRIGCFG.TBL` contains the system trigger configuration information. This table file is not used in VME systems.

- `UTILBUS.TBL` contains the utility bus configuration information. This file can be used in a VME system, but only in a multimainframe situation.

- `CREG.TBL` contains local system information used by the Resource Manager.

- `REGS.TBL` contains the register names of VXI devices. This table file is not used in VME systems.

- WSCMDS.TBL contains the VXI Word Serial command values. This table file is not used in VME systems.
- VXIBUS.CFG contains the AT-MXI MXIbus configuration information.
- VXILA.CFG contains the AT-MXI logical address configuration information.
- VXIDEV.CFG contains the AT-MXI device-specific configuration information.
- VXIMF.CFG, if present, contains information specific to the board manufacturer.
- DVXI.CFG contains information for internal use by the driver.

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

The C:\NIVXI\DOS directory contains the following files.

- RESMAND.EXE is used by RESMAN in the DOS environment. RESMAN calls this file directly.
- VICD.EXE is used by VIC in the DOS environment. VIC calls this file directly.
- VICTEXTD.EXE is used by VICTEXT in the DOS environment. VICTEXT calls this file directly.
- VXIEDITD.EXE is used by VXIEDIT in the DOS environment. VXIEDIT calls this file directly.

- VXITEDID.EXE is used by VXITEDIT in the DOS environment. VXITEDIT calls this file directly.

The C:\NIVXI\DOS directory contains up to three subdirectories.

The C:\NIVXI\DOS\BORLANDC subdirectory is installed if you choose to install both the DOS environment and the Borland C libraries. It contains the following file and subdirectory.

- NIVXIDOS.LIB is the NI-VXI library, which must be included along with other DOS libraries when linking your application program from Borland C. The NI-VXI library uses the large memory model.
- The C:\NIVXI\DOS\BORLANDC\EXAMPLES subdirectory contains various example programs that show how to use the NI-VXI software with Borland C.

The C:\NIVXI\DOS\MSC subdirectory is installed if you choose to install both the DOS environment and the Microsoft C libraries. It contains the following file and subdirectory.

- NIVXIDOS.LIB is the NI-VXI library, which must be included along with other DOS libraries when linking your application program from Microsoft C. The NI-VXI library uses the large memory model.
- The C:\NIVXI\DOS\MSC\EXAMPLES subdirectory contains various example programs that show how to use the NI-VXI software with Microsoft C.

The C:\NIVXI\DOS\QB subdirectory contains the following object, library, and batch files to generate the Microsoft library for QuickBASIC.

- QBNIVXI.LIB contains the NI-VXI function library for the QuickBASIC interface.
- QBNIVXI.OBJ contains the NI-VXI symbols used to generate the QuickBASIC library for all versions.
- MKQLB*.BAT and MKQLB*.LNK are batch and response files used to generate a QuickBASIC library for the desired version.
- MKAPPBC.BAT and MKAPPBC.LNK are batch and response files that contain an example of how to build a BASIC application using NI-VXI functions.

Also included are the following files containing system functions used in the generation of QuickBASIC libraries for version 4.5 (default) and version 7.0 (7F and 7N).

- QB.OBJ
- QBMEM.OBJ
- QB1.LIB
- QB1_7F.LIB
- QB1_7N.LIB

- QB2.LIB
- B1_71F.LIB
- B1_71N.LIB
- B2_71.LIB

The C:\NIVXI\WIN directory contains the following files.

- NIVXI.DLL is the NI-VXI dynamic link library executable file, which contains the executable code for all the NI-VXI functions. This file is loaded when any Windows application that uses the NI-VXI library is executed. The directory where it resides should be added to the PATH variable to ensure that it is loaded properly.

- RESMANW.EXE is used by RESMAN in the Windows environment. RESMAN calls this file directly.

- VICTEXTW.EXE is used by VICTEXT in the Windows environment. VICTEXT calls this file directly.

- VXITEDIW.EXE is used by VXITEDIT in the Windows environment. VXITEDIT calls this file directly.

- NIVXIPHM.386 is a virtual memory driver that the NI-VXI library uses for shared memory under Windows only.

- NIVXI.GRP is a Windows group (folder) which contains icons for easy execution of NI-VXI utilities (such as RESMAN and VICTEXT).

- NIVXI.PIF is a Windows Program Information File that lets you run the DOS NI-VXI utilities in a Windows DOS shell.

The C:\NIVXI\WIN directory contains up to two subdirectories.

The C:\NIVXI\WIN\BORLANDC subdirectory is installed if you choose to install both the Windows environment and the Borland C libraries. It contains the following file and subdirectory.

- NIVXIWIN.LIB is the NI-VXI library, which must be included along with other Windows libraries when linking your application program from Borland C. The NI-VXI library uses the large memory model. However, you are not restricted to using only the large memory model for your application. You can also use the medium, compact, or small memory models.

- The C:\NIVXI\WIN\BORLANDC\EXAMPLES subdirectory contains various example programs that show how to use the NI-VXI software with Borland C.

The `C:\NIVXI\WIN\MSC` subdirectory is installed if you choose to install both the Windows environment and the Microsoft C libraries. It contains the following file and subdirectory.

- `NIVXIWIN.LIB` is the NI-VXI library, which must be included along with other Windows libraries when linking your application program from Microsoft C. The NI-VXI library uses the large memory model. However, you are not restricted to using only the large memory model for your application. You can also use the medium, compact, or small memory models.
- The `C:\NIVXI\WIN\MSC\EXAMPLES` subdirectory contains various example programs that show how to use the NI-VXI software with Microsoft C.

Appendix C
Common Questions

This appendix addresses common questions you may have about using the NI-VXI bus interface software on the AT-MXI platform. If you need to troubleshoot specific errors, refer to Appendix D, *Troubleshooting*.

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

Run the NI-VXI utility program `VICTEXT`. At the prompt type `ver`, and `VICTEXT` will display the versions of `VICTEXT` and NI-VXI, and the latest AT-MXI board revision that this NI-VXI driver supports.

How can I determine what is the revision of the AT-MXI board I have in my system?

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 AT-MXI that the NI-VXI software supports. Because this may not reflect the actual hardware revision of your AT-MXI, the only method currently available to determine the actual hardware revision is to check the label on the board.

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

Use the VXI Resource Editor program, either `VXIEDIT` or `VXITEDIT`, to configure the AT-MXI. It is located in the `NIVXI` directory.

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

Use the hardware initialization program, `VXIINIT`, to initialize the AT-MXI. It is located in the `NIVXI` directory. `VXIINIT` uses the settings in the Configuration Editor of `VXIEDIT` or `VXITEDIT`.

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. Note that `VXIinReg()` and `VXIoutReg()` are for VXI devices only.

Finally, if you are using LabVIEW or LabWindows 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 do the LEDs on the front of the VXI-MXI or VME-MXI 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 ACCESS** LED indicates that the VXI-MXI or VME-MXI is acting as a slave to another device on the MXIbus, such as when the AT-MXI communicates to either the VXI-MXI or VME-MXI or another device in the chassis. The **VXI (VME) ACCESS** LED, when lit, indicates that the VXI-MXI or VME-MXI 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 or VME-MXI or another device outside the chassis.

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

Both the AT-MXI and the VXI-MXI are unique VXIbus devices with their own logical addresses. However, the MXIbus allows the ISA (PC AT) computer to appear as if it is inside the chassis with the VXI-MXI by transparently converting ISA bus cycles to MXIbus cycles to VXIbus cycles, and vice versa.

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

Yes, National Instruments sells MXIbus bulkhead cables. Contact National Instruments and ask for Technical Note 30, *Planning a MXIbus System Using MB Series Bulkhead Cables*, which describes the cables and discusses how to design the system.

Can I access 32-bit registers in my VXIbus/VMEbus system from the AT-MXI?

Because the AT-MXI is designed to work on the ISA bus, which is only 16 bits wide, the AT-MXI cannot access a 32-bit register in a single, D32 access. However, NI-VXI low-level functions accept the request for accessing a 32-bit register by breaking the access into two 16-bit accesses. The high-level functions, however, return an error since the width exceeds the capability of the AT-MXI. If the device supports the reading of its 32-bit registers 16 bits at a time, there should be no problem requesting a 32-bit register I/O from the AT-MXI.

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 backplane. However, the oscillator for the VXI-MXI 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.

What is the accuracy of the CLK10 signal?

The CLK10 generated by the VXI-MXI 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.

Appendix D Troubleshooting

This appendix addresses specific problems you may encounter when using the NI-VXI bus interface software on the AT-MXI platform, including the following issues:

- Running the `VXIINIT` initialization program results in an error message or causes the system to hang.
- Running the `RESMAN` Resource Manager program results in incorrect data or causes the system to hang.
- The mouse behaves erratically after you install the NI-VXI software.
- The network fails to initialize.

If you encounter any of these problems, your system may have a conflict with one of the AT-MXI configuration settings or you may have a problem with the MXIbus cable(s).

If I encounter problems, what should I try first?

First, check your configuration. For proper operation, the AT-MXI requires the allocation of the following system resources:

- Thirty-two bytes of I/O register space.
- At least one interrupt level that cannot be shared with another device.
- A DMA channel (optional).
- A contiguous 64 KB block of memory space in the upper memory region of the PC AT.

The remaining questions in this appendix discuss these system requirements and problems that may be caused by a bad MXI cable.

What does `VXIINIT` mean by `VXI device not found`?

`VXIINIT` first checks to see if the AT-MXI board is present by attempting to access certain registers on the board. If these accesses do not return the expected information, `VXIINIT` assumes that the board is not installed and reports **`VXI device not found`**. If you receive this message, check the following:

1. Verify that the settings in `VXIEDIT` or `VXITEDIT` match the settings on the board.

2. Make sure that the I/O registers of the AT-MXI do not conflict with any other boards in the system. If they do, change the settings of the AT-MXI or the other device.

3. Make sure the AT-MXI is securely seated in the ISA (PC AT) slot or try another slot of the computer.

If the message persists, there is still some reason that the NI-VXI software cannot access the I/O registers of the AT-MXI. If possible, try the AT-MXI in another computer.

What does `VXIINIT` mean by `File` *`filename`* `not found`?

When `VXIINIT` runs, it attempts to access some of the configuration or table files, such as `VXILA.CFG`, `VXIBUS.CFG`, `VXIDEV.CFG`, `MFNAMEID.TBL`, and `MODEL.TBL`. If `VXIINIT` cannot find any of these files, you have the following two options.

- The file(s) may have been deleted accidentally. If the file does not exist in your `TBL` directory after you installed your software, you will need to reinstall NI-VXI from your software distribution media.

- The `NIVXIPATH` variable may be pointing to the wrong directory. The path should be set in the `AUTOEXEC.BAT` file to point to the root directory of the NI-VXI files. By default this directory is `C:\NIVXI`.

What does `VXIINIT` mean by `Error writing configuration to file`?

This is only a warning message, and does not indicate a problem with the configuration. Either the log file could not be created, or only a partial log was written.

`VXIINIT` creates a file `VXIINIT.OUT` in the `TBL` subdirectory. This file is a text file that contains the same configuration information as you see displayed on the screen. The likely cause of this message is that the disk drive is full.

What do I do if `VXIINIT` causes my system to hang?

If your system hangs when you run `VXIINIT`, first make sure that the I/O registers of the AT-MXI do not conflict with any other hardware in the PC AT, as discussed in the previous questions. If you cannot find a conflict with the I/O registers, check for a DMA channel conflict.

In its default configuration, the AT-MXI uses one DMA channel, channel 3, for slave accesses. You can change this channel by using the Logical Address Configuration Editor of the `VXITEDIT` program and modifying the **Slave DMA Channel** field. Try setting this parameter to NONE, saving the new configuration, and running `VXIINIT`. If the system does not hang, the problem was due to a DMA channel conflict.

An application uses slave DMA when it needs to dual-port PC AT memory so that it can be accessed from the VXI/VME chassis. If the PC AT will simply read and write from devices, it does not need slave DMA; you can keep the parameter set to NONE. If your application needs slave accesses to the PC AT, follow these steps to find an available DMA channel that you can set up successfully:

1. Determine if other devices are configured to use DMA channels.

2. Choose a level that is not used in your system and set the AT-MXI to use this level. Refer to the *DMA Channel Selection* section of Chapter 2, *AT-MXI Configuration and Installation*, for instructions on making this hardware change.

3. After you change the AT-MXI slave DMA channel jumper, use the Logical Address Configuration Editor of `VXIEDIT` or `VXITEDIT` to enter the new slave DMA channel.

4. Save the changes and run `VXIINIT`.

As mentioned in the *Master Mode Versus Slave Mode* section of Chapter 2, the NI-VXI software does not use master-mode DMA for block transfers because the `movs` instruction is faster on most PCs. The latest version of the NI-VXI software sets the default value for the **Master DMA Channel** field in the Logical Address Configuration Editor to NONE, and this value cannot be modified. If you are using a version of the NI-VXI software for the AT-MXI prior to Version 3.1, make sure that this value is set to NONE.

What is wrong if my mouse behaves erratically after I run `VXIINIT`?

If you notice your mouse behaving erratically after you run `VXIINIT`, the problem is most likely an interrupt conflict. The AT-MXI requires one interrupt line that it cannot share with any other device in the system. The default AT-MXI interrupt level is 12.

To correct an interrupt conflict, first verify that the AT-MXI board interrupt level (set by a jumper) matches the software setting in the Logical Address Configuration Editor of the `VXIEDIT` or `VXITEDIT` program. Then determine if any other devices are configured to use the AT-MXI interrupt level.

If a conflict exists, change the interrupt level of either the conflicting device or the AT-MXI. If you decide to change the AT-MXI interrupt level, you must change the interrupt jumper hardware setting, enter the new interrupt level in the Logical Address Configuration Editor, save the changes, and reload the driver.

Why would `RESMAN` report a warning/error that a device does not exist?

If the `VXIINIT` program runs successfully but you encounter errors when running `RESMAN`, the Resource Manager program, your system probably contains a device that already occupies all or some of the upper memory area that is allocated for the master window. The master window is a 64-KB block of contiguous memory in the PC AT upper memory space that the AT-MXI uses to access the VXI/VMEbus. The upper memory space of a PC AT is the region from 640 KB (A000h) to 1024 KB (FFFFh). Several system resources, such as system ROM and video memory, reside in this area. By default, the master window occupies a memory region from a base address of D000h to DFFFh.

To solve this problem, first determine what area of upper memory your devices use. Then move either your device or the master window base address to an unused area of upper memory. You can change the master window base address by running the VXI resource editor program, either `VXIEDIT` or `VXITEDIT`, and using the Logical Address Configuration Editor to make the change.

If you have a Super VGA monitor, your system is likely to use video RAM. Generally, any video RAM used by your PC AT occupies all or some of the upper memory from A000h to BFFFh. If you have Ethernet networking cards, be aware that many of them occupy upper memory. Many Ethernet cards have an upper memory base address of D000h or E000h.

Your system may also contain an expanded-memory manager or an expanded-memory emulator that uses a block of upper memory within the range that is allocated for the Master Window.

To solve this problem, verify which expanded-memory manager or emulator you are using, and prevent it from using upper memory within the Master Window address range. The following examples show how to prevent various expanded-memory managers or emulators from using upper memory within the default Master Window address range of D000h to DFFFh.

Note: ***If you have changed your Master Window address range, use it instead of the default range (D000 to DFFF) in the following examples.***

If you are using EMM386, which is included with MS-DOS, prevent it from using the default Master Window address range by editing the `CONFIG.SYS` file and appending the flag `X=D000-DFFF` to the line `DEVICE=EMM386.EXE`.

If you are using QEMM386 from Quarterdeck, prevent it from using the default Master Window address range by editing the `CONFIG.SYS` file and appending the flag `X=D000-DFFF` to the line `DEVICE=QEMM386.SYS`.

If you are using 386MAX from Qualitas, prevent it from using the default Master Window address range by editing the `386MAX.PRO` file located in the `386MAX` directory, and inserting the line `RAM=D000-DFFF`.

If you are using any other expanded-memory manager, refer to its documentation to learn how to prevent it from using the default Master Window address range of D000h to DFFFh.

If these suggestions do not fix the problem, you may have a problem with your MXI cable. Refer to the question *What caused `RESMAN` to suddenly stop detecting my chassis or detect it intermittently?*

What is the problem if `VXIINIT` works fine but `RESMAN` hangs the system?

This problem is most likely caused by a conflict with the master window. See the above question, *Why does `RESMAN` report warning/errors about devices that do not exist in my system?*

What might cause `RESMAN` to suddenly stop detecting my chassis or detect it only intermittently?

If `RESMAN` can detect the AT-MXI but has problems seeing the chassis, the problem may relate to the MXI cable. Perform the following checks to determine the problem with your MXIbus cables.

1. Check the pins on your cable connectors. Sometimes the pins get bent and fail to provide the proper electrical connection.

2. Make sure that each end of the MXIbus cable is plugged into the proper connector. Certain MXIbus cables have a single-point connector on one end and a daisy-chain connector on the other end. With these cables, you must connect the single-point connector to the MXIbus device that is closer in the MXIbus chain to the MXIbus System Controller. The MXIbus System Controller determines who gains control of the MXIbus. By default, the AT-MXI is the MXIbus System Controller (set by the Bus Configuration Editor) and the VXI-MXI is not the System Controller (set by an onboard slide switch). If you are having problems, check that the AT-MXI and the VXI-MXI are set accordingly.

3. If the cabling problem is not apparent, try another MXIbus cable if you have one available. If this solves the problem, contact National Instruments for repair information.

What is wrong if my system hangs after I install the AT-MXI board and try to start Windows?

In this situation, you might have a conflict with the network card in your system. Both network drivers and the AT-MXI use memory-mapped I/O and other hardware resources. Check the following items for conflicts between the AT-MXI and the network card:

- IRQ (interrupt) lines used (default settings for the AT-MXI are IRQ level 10 for the MXIbus interrupt level and level 12 for the board interrupt level)
- Master window base (default for the AT-MXI is 0xD000)
- Base I/O address (default base address for the AT-MXI is 0x340)

If you detect a conflict in any of these items, modify the settings for either the AT-MXI or the network card. You can change the setting for the AT-MXI by changing the jumpers on the board and then making the corresponding change in `VXITEDIT`. Remember that you need to reload the driver if you make any changes to the hardware. See Chapter 2, *AT-MXI Configuration and Installation*, for information on changing the hardware settings and Chapter 5, *NI-VXI Software Installation and Configuration*, for information on using `VXITEDIT` to change the system settings.

What might prevent my network from working properly after I install the AT-MXI?

The problem may be a conflict with the network card. See the previous question *What is wrong if my system hangs after I install the AT-MXI board and try to boot Windows?* for more information.

What information should I have before I call National Instruments?

Make sure that you have looked through this troubleshooting information for the answer to your problem. Appendix C, *Common Questions*, may also contain the answer to your question. When you call National Instruments, make sure you have filled out the *VXI/VME-AT2010 Hardware and Software Configuration Form* in Appendix E. Also, if you are having problems with `RESMAN`, you can get `RESMAN` to create a file of its output by issuing the following command:

```
resman -o
```

from DOS, the Windows Program Manager's **File** menu **Run** option, or a Windows DOS Shell, although the latter is not recommended.

Appendix E
Customer Communication

For your convenience, this appendix contains forms to help you gather the information necessary to help us solve technical problems you might have as well as a form you can use to comment on the product documentation. Filling out a copy of the *Technical Support Form* before contacting National Instruments helps us help you better and faster.

National Instruments provides comprehensive technical assistance around the world. In the U.S. and Canada, applications engineers are available Monday through Friday from 8:00 a.m. to 6:00 p.m. (central time). In other countries, contact the nearest branch office. You may fax questions to us at any time.

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

Branch Offices	**Phone Number**	**Fax Number**
Australia	(03) 879 9422	(03) 879 9179
Austria	(0662) 435986	(0662) 437010-19
Belgium	02/757.00.20	02/757.03.11
Denmark	45 76 26 00	45 76 71 11
Finland	(90) 527 2321	(90) 502 2930
France	(1) 48 14 24 00	(1) 48 14 24 14
Germany	089/741 31 30	089/714 60 35
Italy	02/48301892	02/48301915
Japan	(03) 3788-1921	(03) 3788-1923
Mexico	95 800 010 0793	95 800 010 0793
Netherlands	03480-33466	03480-30673
Norway	32-848400	32-848600
Singapore	2265886	2265887
Spain	(91) 640 0085	(91) 640 0533
Sweden	08-730 49 70	08-730 43 70
Switzerland	056/20 51 51	056/20 51 55
Taiwan	02 377 1200	02 737 4644
U.K.	0635 523545	0635 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. Use additional pages if necessary.

Name ____

Company ____

Address ____

____

Fax (___) ____________________ Phone (___) ____________________

Computer brand ____________ Model ____________ Processor ____________

Operating system ____________________________________

Speed __________MHz RAM __________MB Display adapter __________

Mouse _______yes _______no Other adapters installed ________________

Hard disk capacity __________MB Brand ________________________

Instruments used ____________________________________

National Instruments hardware product model ____________ Revision ____________

Configuration ____________________________________

National Instruments software product ____________ Version ____________

Configuration ____________________________________

The problem is ____________________________________

____

____

____

____

List any error messages ________________________________

____

____

____

____

The following steps will reproduce the problem ____________________

____

____

____

VXI/VME-AT2010 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 Revision Number ____________________
  (Disk Labels: *NI-VXI Distribution Disks for the AT-MXI and Windows*)
- C Programming Language Interface Revision ____________________
- AT-MXI

Hardware Revision	____________________
Base I/O Address	____________________
Master Window Base	____________________
Address Space	____________________
Board Interrupt Level	____________________
Master DMA Channel	____________________
Slave DMA Channel	____________________
Slave I/O Window Base	____________________
Slave I/O Window Size	____________________
Shared RAM Window Base	____________________
Shared RAM Window Size	____________________
Resource Manager Delay	____________________
AT-MXI is MXIbus System Controller?	____________________
AT-MXI is Fair Requester?	____________________
MXIbus Interrupt Level	____________________
Slave Mode Timeout	____________________
Master Mode Timeout	____________________
System Controller Timeout	____________________

- Using VXI Mainframe or VME Chassis? ____________________

• Using VXI-MXI or VME-MXI?	
• VXI-MXI Hardware Revision	
• VME-MXI Hardware Revision	
• Slot Location of VXI-MXI or VME-MXI	
• VXI-MXI Logical Address	
• VXI-MXI VMEbus Request Level	
• VME-MXI is VMEbus System Controller?	
• VME-MXI Address	
• VME-MXI VMEbus Request Level	
• VME-MXI VMEbus BTO Timeout Value	

Other Products

• Computer Make and Model	
• Microprocessor	
• Clock Frequency (Bus and Microprocessor)	
• Total Memory in System	
• Type of Video Board Installed	
• Windows Version	
• Windows Mode (Enhanced or Standard)	
• C Compiler	
• Other Boards in System	
• Base I/O Address of Other Boards	
• DMA Channels of Other Boards	
• Interrupt Level of Other Boards	
• VXIbus Mainframe Make and Model	
• Other VXIbus Devices in System	
• Static Logical Addresses of Other VXIbus Devices	
• VMEbus Chassis Make and Model	
• VMEbus Devices in System	
• Addresses of other VMEbus Devices	

Documentation Comment Form

National Instruments encourages you to comment on the documentation supplied with our products. This information helps us provide quality products to meet your needs.

Title: **Getting Started with Your VXI/VME-AT2010 and the NI-VXI™ Software for Windows**

Edition Date: **December 1994**

Part Number: **320888A-01**

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

____

____

____

____

____

____

____

If you find errors in the manual, please record the page numbers and describe the errors.

____

____

____

____

____

____

____

____

Thank you for your help.

Name ____

Title ____

Company ____

Address ____

____

Phone (_______) ___________________________

Mail to:	Technical Publications National Instruments Corporation 6504 Bridge Point Parkway, MS 53-02 Austin, TX 78730-5039	Fax to:	Technical Publications National Instruments Corporation MS 53-02 (512) 794-5678

Glossary

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

Symbols

°	degrees
%	percent
±	plus or minus

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.
binary	A numbering system with a base of 2.
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

DACK	DMA Acknowledge
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.
DRQ	DMA Request
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
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.
EMI	Electromagnetic Interference
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

F	Fahrenheit
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.
FCC	Federal Communications Commission

G

GIN	Daisy-chain Grant In signal
GOUT	Daisy-chain Grant Out signal
GPIB	General Purpose Interface Bus; the industry-standard IEEE 488 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.
INTX	Interrupt Timing and Extension
IRQ*	Interrupt signal
ISA	Industry Standard Architecture

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

MB	Megabytes of memory
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.
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.
MODID	Module Identification lines
`movs`	Move string command
MTBF	Mean Time Between Failure
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.

P

PC AT	Personal Computer Advanced Technology
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.
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.
RM	See *Resource Manager*.

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.
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.

T

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

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
VXIINIT	A program in the NI-VXI bus interface software package that initializes the board interrupts, shared RAM, VXI register configurations, and bus configurations.
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.

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.

Index