APEX WAVES

COMPREHENSIVE SERVICES

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

SELL YOUR SURPLUS

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

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

OBSOLETE NI HARDWARE IN STOCK & READY TO SHIP

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

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

📞 **1-800-915-6216**

🌐 **www.apexwaves.com**

✉ **sales@apexwaves.com**

Request a Quote   ✉ **CLICK HERE**   *VXI-MXI*

Getting Started With Your VME-MC2010 and the NI-VXI Software for Microsoft Windows

National Instruments Corporate Headquarters
6504 Bridge Point Parkway
Austin, TX 78730-5039
(512) 794-0100
(800) 433-3488 (toll-free U.S. and Canada)
Technical support fax: (512) 794-5678

Branch Offices:
Australia 03 879 9422, Belgium 02 757 00 20, Canada (Ontario) 519 622 9310, Canada (Québec) 514 694 8521
Denmark 45 76 73 22, Finland 90 5272321, France 1 48 65 33 70, Germany 089 714 50 93, Italy 02 48301892,
Japan 03 3788 1921, Netherlands 01720 45761, Norway 03 846866, Spain 91 896 0675, Sweden 08 984970,
Switzerland 056 27 00 20, U.K. 0635 523545

Limited Warranty

The National Instruments MXIbus boards are warranted against defects in materials and workmanship for a period of one year from the date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace equipment that proves to be defective during the warranty period. This warranty includes parts and labor.

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

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

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

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

Copyright

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

Trademarks

LabVIEW®, LabWindows®, and NI-VXI™ are trademarks of National Instruments Corporation.

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

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:

- The limits for a Class A computing device, in accordance with the specifications in Part 15 of U.S. Federal Communications Commission (FCC) Rules. Operation is subject to the following two conditions:

    – This device may not cause harmful interference.

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

- The limits for radio noise emissions from digital apparatus set out in the Radio Interference Regulations of the Canadian Department of Communications (DOC).

These regulations are designed to provide reasonable protection against interference from the equipment to radio and television reception in commercial areas.

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 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.

- Relocate the equipment with respect to the receiver.

- Reorient 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

Appendix A
Specifications

Appendix B
Customer Communication

Figures

Tables

About This Manual

This manual contains instructions for installing and configuring the National Instruments VME-MC2010 interface kit. This manual is meant to be used with the *NI-VXI C Software Reference Manual for VME* (part number 320389-01), the *NI-VXI DOS Utilities Reference Manual* (part number 320323-01), and the *NI-VXI Text Utilities Reference Manual* (part number 320321-01).

Organization of This Manual

Getting Started with Your VME-MC2010 and the NI-VXI Software for Microsoft Windows is organized as follows:

- Chapter 1, *Introduction*, describes the VME-MC2010 interface kit, contains a picture of the VME-MC2010 interface kit, lists the contents of your kit, and lists optional equipment and software.

- Chapter 2, *Hardware Configuration and Installation*, contains the instructions for installing and configuring the VME-MC2010 interface kit.

- Chapter 3, *NI-VXI Software Configuration and Installation*, lists the programs and files located on the NI-VXI distribution diskettes, contains instructions for installing and configuring the NI-VXI software for Microsoft Windows for VME, and contains a description of the `VXIEDIT` and `VXITEDIT` configuration editors. This chapter also discusses programming information for you to consider when developing Windows applications that use the NI-VXI driver.

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

- Appendix B, *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.

Conventions Used in This Manual

Throughout this manual, the following conventions are used to distinguish elements of text:

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

`monospace`	Text in this font denotes text or characters that are to be literally input from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, programs, commands, variables, filenames, and extensions, and for statements and comments taken from program code.

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

How to Use the Documentation Set

Begin by reading this manual to guide you through the installation and configuration of the hardware and software. The software configuration requires that you first complete the installation and configuration of the hardware.

When you are familiar with the material in this manual, you can begin to use the *NI-VXI C Software Reference Manual for VME*. Chapter 1, *Introduction to NI-VXI for VME*, and Chapter 2, *Introduction to the NI-VXI Functions for VME,* present the concepts of using the NI-VXI software for programming VME systems and prepare you for detailed explanations of the NI-VXI functions. Study the descriptions of each function given in Chapters 3 through 9 to fully understand the purpose and syntax of each function. Refer to the *NI-VXI DOS Utilities Reference Manual* to learn more about the `VXIEDIT` and `VIC` programs. The *NI-VXI Text Utilities Reference Manual* describes the `VXITEDIT` and `VICTEXT` programs.

Related Documentation

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

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

• *NI-VXI Software Reference Manual for C* (for VXI systems, part number 320307-01)

• *The IBM Personal System/2 Quick Reference* manual

• *VME-MXI User Manual* (part number 320330-01)

• VXI-1, *VXIbus System Specification*, Rev. 1.4, VXIbus Consortium (available from National Instruments Corporation, part number 350083-01)

• VXI-6, *VXIbus Mainframe Extender Specification*, Rev. 1.0, VXIbus Consortium (available from National Instruments Corporation, part number 340258-01)

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

Chapter 1
Introduction

This chapter describes the VME-MC2010 interface kit, contains a picture of the VME-MC2010 interface kit, lists the contents of your kit, and lists optional equipment and software.

The VME-MC2010 interface kit links any IBM PS/2 32-bit Micro Channel-based computer or compatible (hereafter referred to as a Micro Channel computer) directly to the VXIbus. A Micro Channel computer equipped with a VME-MC2010 can function as a VME master and/or slave device. The VME-MC2010 makes your computer perform 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. The driver software for the VME boards must, of course, be ported to run on the Micro Channel computer with the VME-MC2010 kit.

Figure 1-1 shows the VME-MC2010 interface kit.

Art not available in PDF version of document.
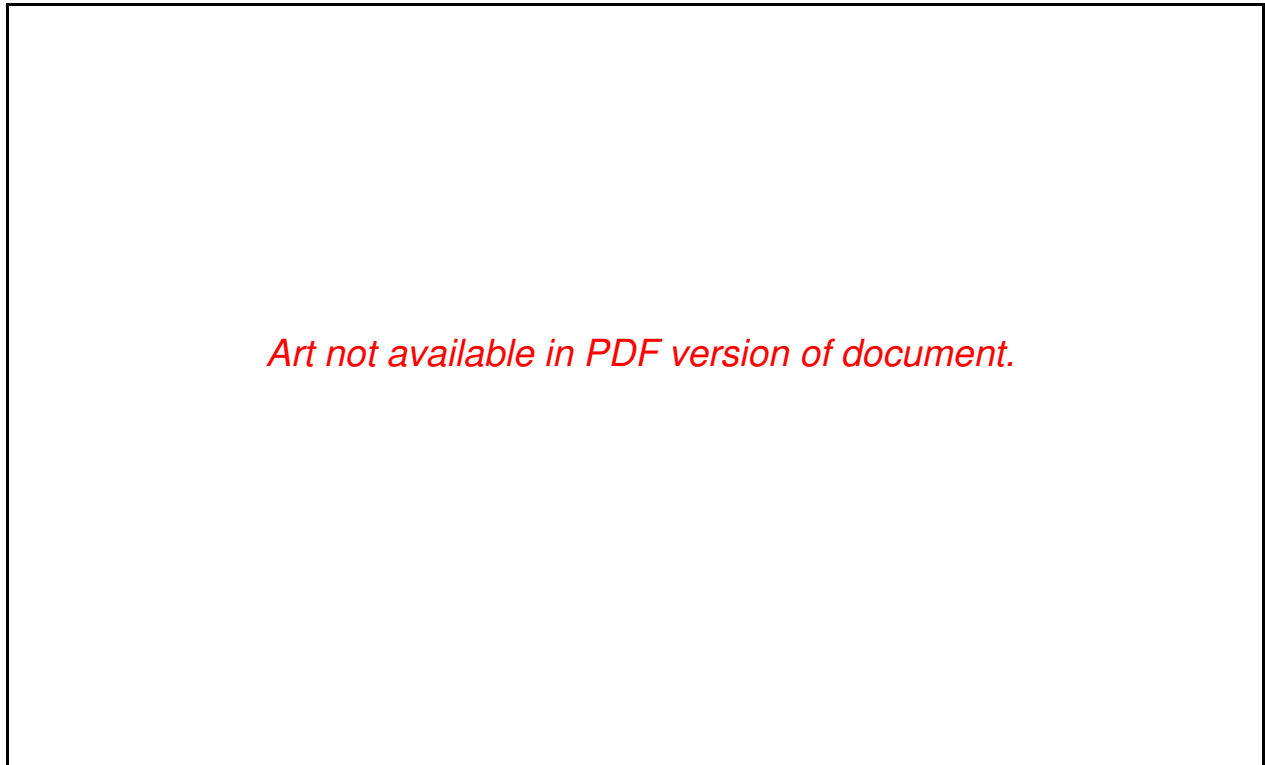
Figure 1-1. VME-MC2010 Interface Kit

What Your Kit Should Contain

Your kit should contain the following components.

Component	Part Number
MC-MXI Interface Board	181075-01
Standard VME-MXI Interface Module or Enhanced VME-MXI Interface Module with INTX	180960-01 180960-02
NI-VXI Distribution Disks for the MC-MXI and Microsoft Windows	422825-52 422839-52 422906-52
2 m Type M1 MXIbus Cable *	180758-02
Getting Started with Your VME-MC2010 and the NI-VXI Software for Microsoft Windows	320430-01
NI-VXI C Software Reference Manual for VME	320389-01
NI-VXI DOS Utilities Reference Manual	320323-01
NI-VXI Text Utilities Reference Manual	320321-01

  * 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 on the following page.

Make sure each of these items is in your kit. If any item is missing, contact National Instruments.

Optional Equipment

Equipment	Part Number
Type M1 MXIbus Cables Straight Point-to-Point Connectors: – 1 m – 2 m – 4 m – 8 m – 20 m	 180758-01 180758-02 180758-04 180758-08 180758-20
Type M2 MXIbus Cables Straight Point-to-Right Angle Daisy-Chain Connectors: – 1 m – 2 m – 4 m – 8 m – 20 m	 180760-01 180760-02 180760-04 180760-08 180760-20
Type M3 MXIbus Cables Right Angle Point-to-Right Angle Daisy-Chain Connectors: – 1 m – 2 m – 4 m – 8 m – 20 m	 180761-01 180761-02 180761-04 180761-08 180761-20
MXIbus Terminating Pac (External)	180780-01
VXI-MXI Standard VXIbus Mainframe Extender Kit	776310-01
VXI-MXI Enhanced VXIbus Mainframe Extender Kit	776310-02
VME-MXI Standard VMEbus Chassis Extender Kit	776434-01
VME-MXI Enhanced VMEbus Chassis Extender Kit	776434-02

Optional Software

If you want to use LabVIEW with your VME-MC2010, you must order both LabVIEW for Windows (either the Base Package or the Full Development System) and the LabVIEW for Windows VXI VI Library.

Software	Part Number
LabVIEW for Windows, Full Development System (includes Base Package and Advanced Analysis Library)	776670-01
LabVIEW for Windows, Base Package	776671-01
LabVIEW for Windows, Advanced Analysis Library	776672-01
LabVIEW for Windows VXI VI Library	776674-01

If you want to use LabWindows with your VME-MC2010, you must order both LabWindows for DOS (either the Base Package or the Full Development System) and the LabWindows VXI Library for the MC-MXI.

Software	Part Number
LabWindows for DOS, Full Development System (includes Base Package and Advanced Analysis Library)	776475-01
LabWindows for DOS, Base Package	776473-01
LabWindows for DOS, Advanced Analysis Library	776474-01
LabWindows VXI Library for MC-MXI	776382-01

Chapter 2
Hardware Configuration and Installation

This chapter contains the instructions to configure and install the VME-MC2010 interface kit. The instructions are given in the order that you should perform them. A summary of the steps is as follows:

1. Unpack the MC-MXI and VME-MXI hardware.

2. Install the MC-MXI adapter file.

3. Install the MC-MXI hardware.

4. Configure the MC-MXI hardware.

5. Configure the VME-MXI hardware.

6. Install the VME-MXI hardware.

7. Connect the MXIbus cable.

Step 1. Unpack the MC-MXI and VME-MXI Hardware

Follow these steps when unpacking your MC-MXI board and VME-MXI module:

1. Before attempting to configure or install the MC-MXI and VME-MXI, inspect the shipping container and its contents for damage. If damage appears to have been caused in shipment, file a claim with the carrier. Retain the packing material for possible inspection and/or for reshipment.

2. Verify that the pieces contained in the package you received match the kit parts list. *Do not* remove the boards from their plastic bags at this point.

3. Your MC-MXI board and VME-MXI module are shipped packaged in antistatic plastic bags to prevent electrostatic damage. Some of the circuitry on the MC-MXI and VME-MXI use CMOS technology and can be damaged by electrostatic discharge. Before removing the boards from their antistatic bags, touch the bags to a metal part of your computer chassis.

4. As you remove the MC-MXI and VME-MXI from their bags, be sure to handle them only by their edges. Avoid touching any of the IC components or connectors. Inspect them for loose components or any other sign of damage. Notify National Instruments if either board appears damaged in any way. *Do not* install equipment that appears to be damaged.

Step 2. Install the MC-MXI Adapter File

To install the MC-MXI adapter file you need the following components:

- NI-VXI Distribution Disks for the MC-MXI and Microsoft Windows

  Note: These diskettes will be referred to as the NI-VXI distribution diskettes throughout the remainder of this manual.

- Backup copy of the *IBM Personal System/2 Reference Diskette*

  Note: This diskette will be referred to as the IBM reference diskette throughout the remainder of this manual.

If you do not have a backup copy of the IBM reference diskette, you can make one by following the instructions in the *IBM Personal System/2 Quick Reference* manual supplied with your IBM Personal System/2 computer.

1. Insert the backup copy of the IBM reference diskette into the diskette drive and restart the computer.

2. From the reference program Main Menu, select *Copy an option diskette* to copy the configuration file from your NI-VXI distribution diskette supplied with your MC-MXI hardware.

3. When the reference program requests the option diskette, insert the NI-VXI distribution diskette and copy the files.

Step 3. Install the MC-MXI Hardware

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

1. Shut off power to the computer.

2. Remove the top cover or access port to the Micro Channel I/O bus.

3. Select a 32-bit full length expansion slot. These slots have a longer card edge expansion connector than those found on 16-bit slots. Most Micro Channel computers have an assortment of both 16-bit and 32-bit expansion slots. A 32-bit board, such as the MC-MXI, cannot be installed in a 16-bit expansion slot.

4. Locate the metal bracket that covers the cut-out in the back panel of the Micro Channel chassis for the slot you have selected. Loosen the bracket retaining thumbscrew and remove the bracket by lifting it out of the top of the slot.

5. Line up the MC-MXI with the MXIbus connector near the cut-out on the back panel and the plastic card guide lined up with the respective slot guide. Slowly push down on the front of the MC-MXI until its card edge connector is resting on the expansion slot receptacle. Using slow, evenly distributed pressure, press the MC-MXI straight down until it seats in the expansion slot.

6. Tighten the bracket retaining thumbscrew to secure the MC-MXI to the back panel rail.

7. Check the installation, and replace the computer cover.

8. Reconnect power to the computer.

Step 4. Configure the MC-MXI Hardware

Restart your computer with the IBM reference diskette and with the MC-MXI board properly installed. A prompt from the reference program will ask if you want to automatically configure the computer. Respond by typing a *y* for yes. The reference program automatically configures the hardware by assigning values for the memory base address, the I/O base address, and interrupt and arbitration levels of the interface board. To see the values assigned by the reference program, select *Set configuration* from the Main Menu and then *View configuration* from the subsequent Set Configuration Menu of the IBM reference diskette.

To change these settings, select *Set configuration* from the Main Menu, then select *Change configuration* from the subsequent Set Configuration Menu of the IBM reference diskette. The following sections describe the MC-MXI configuration settings in more detail.

Memory Base Addresses

The MC-MXI can use one of two different addressing schemes to map Micro Channel memory cycles to the VMEbus. One option is to use a single 64-kilobyte Micro Channel memory window to map all VMEbus transfer types. The other option is a multiwindow approach in which you can map all three VMEbus address spaces individually via a 32832-kilobyte area of Micro Channel memory space. The size of this multiwindow area is a combination of two 16-megabyte windows plus one 64-kilobyte window.

Memory Window for MC-MXI DOS

DOS applications (such as `VIC`) can only access memory resources within the first 1-megabyte block of Micro Channel memory space. Therefore you must use the 64-kilobyte memory window area in order to run any DOS NI-VXI applications. You can select the *NONE* option if you are going to run only Windows NI-VXI applications.

Memory Window for MC-MXI Windows

Microsoft Windows applications (such as the NI-VXI software driver) can access the entire memory space of the Micro Channel bus, including the multiwindow areas provided by the MC-MXI. This is beneficial because using the larger memory windows increases the efficiency of some of the VXI function calls.

Notes: If you are running Windows in Standard Mode, be sure to specify a DOS memory window because the multiwindow area applies only to Enhanced Mode.

If you select the *NONE* option, Enhanced Mode programs will use the single DOS window instead of the multiwindow approach. This should normally not be done.

Base I/O Address

The MC-MXI uses 6 kilobytes of consecutive I/O space. This space must begin on an even 8K boundary (for example, A000 through B7FF, C000 through D7FF, and so on). The software uses this space to configure and monitor the board's operational status. The base address of this I/O space is set by the automatic configuration utility and can be changed by selecting *Change configuration*.

Interrupt Levels

The MC-MXI interface board can be configured to use either one or two of any of the available interrupt levels. Both interrupt levels are selected during automatic configuration to default to level 10. One of the interrupt conditions selects the level on which any of ten different board conditions can cause interrupts.

The other interrupt condition is the MXIbus IRQ line. Notice that the default configuration sets the board and MXIbus interrupt conditions to operate on the same level, although the MXIbus interrupt condition can be changed to operate at its own level.

Refer to the documentation that came with your computer for more information on interrupt level availability with system-related functions. Notice that all Micro Channel devices can share interrupt levels with other devices.

Arbitration Level Selection

The MC-MXI can use one, two, or none of the available arbitration levels of the Micro Channel. Arbitration levels are selected by the automatic configuration utility and must be unique for every device. The following section describes the operational modes that require the use of Micro Channel arbitration levels.

Master Mode Versus Slave Mode

The MC-MXI can function as both a MXIbus master and a MXIbus slave. As a MXIbus master, the MC-MXI circuitry determines whether a Micro Channel cycle is to be mapped into a MXIbus cycle intended for some external MXIbus device such as a VMEbus backplane. As a MXIbus slave, the MC-MXI circuitry determines whether an external device is attempting to access Micro Channel memory or I/O resources. When allocating arbitration levels for use by the MC-MXI, keep in mind that master-mode and slave-mode operation are two distinct asynchronous functions and require different arbitration levels.

Use the slave-mode arbitration level to set up shared access to Micro Channel resources from external MXIbus devices such as a VME bus master in an external mainframe. If you intend to communicate with VME devices using a shared memory protocol that uses the Micro Channel memory, you must enable an arbitration level for slave-mode operation.

You can use the master-mode arbitration level in conjunction with the Micro Channel DMA Controller to perform high-speed block-mode transfers to or from external MXIbus devices. If a master-mode arbitration level is selected and enabled, any block-mode transfer that maps to or from the MXIbus will use the DMA Controller to perform the block-mode move via that channel.

If the master-mode arbitration level is disabled, block-mode transfers will be performed using the processor's `movs` (move string) command instead of using the DMA Controller. This does not mean that it will take any longer for the data to be transferred. In fact, most Micro Channel machines can transfer data faster using the `movs` instruction than they can using the DMA Controller. It may be preferable to use a DMA channel if you have other useful work to do during a block-mode operation (such as in a multitasking operating system).

Local Bus Fairness

All Micro Channel devices that have the ability of holding the local (Micro Channel) bus for more than one bus cycle must support a programmable fairness option. If fairness is enabled, an arbitrating device that already owns the bus must release it when it is preempted by another requesting device. It may not request the bus again until all other devices have had a chance to use it. The fairness option ensures that all Micro Channel devices get a portion of the bus bandwidth.

If fairness is disabled, a single device can retain ownership of the bus even when other devices are requesting it. Fairness can be disabled to implement a priority scheme but this practice should be used with caution as it can prevent other devices from ever using the bus to transfer data.

Step 5. Configure the VME-MXI Hardware

Figure 2-1 shows the location and factory default settings of the configuration switches and jumpers for a VME-MXI without the INTX daughter card option. The remainder of this chapter describes only those options that are user-configurable. Do not change the default settings of other jumpers and switches on the VME-MXI board unless you plan to install more than one VME-MXIs in the same chassis. If this is the case, refer to the *VME-MXI User Manual* that came with your additional VME-MXI interface for more information.

Art not available in PDF version of document.
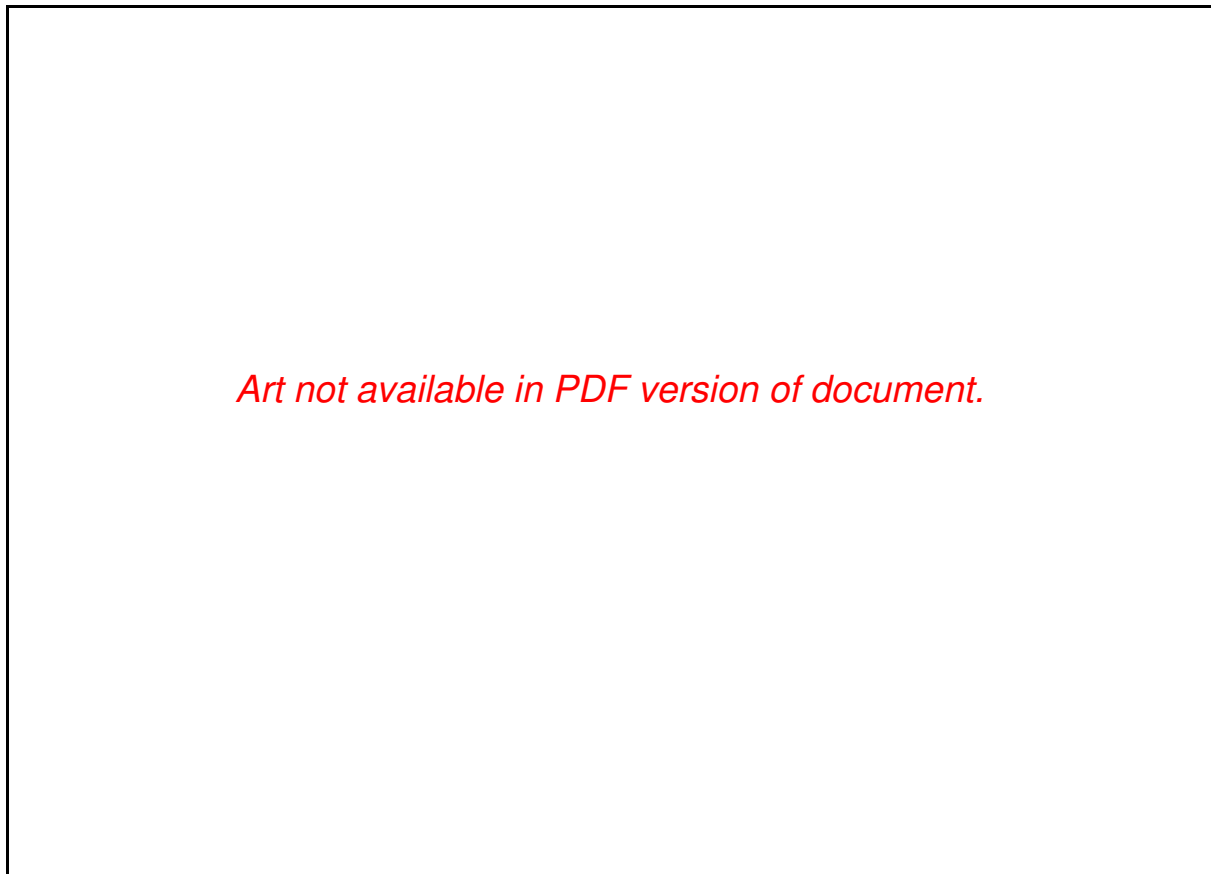
Figure 2-1. VME-MXI Parts Locator Diagram

Figure 2-2 shows the location and factory default settings of the configuration switches and jumpers for a VME-MXI with the INTX daughter card option. The options described in the rest of this chapter apply to VME-MXI modules with or without the INTX option. For more information about configuring modules with the INTX option, refer to the *VME-MXI User Manual*.

Art not available in PDF version of document.
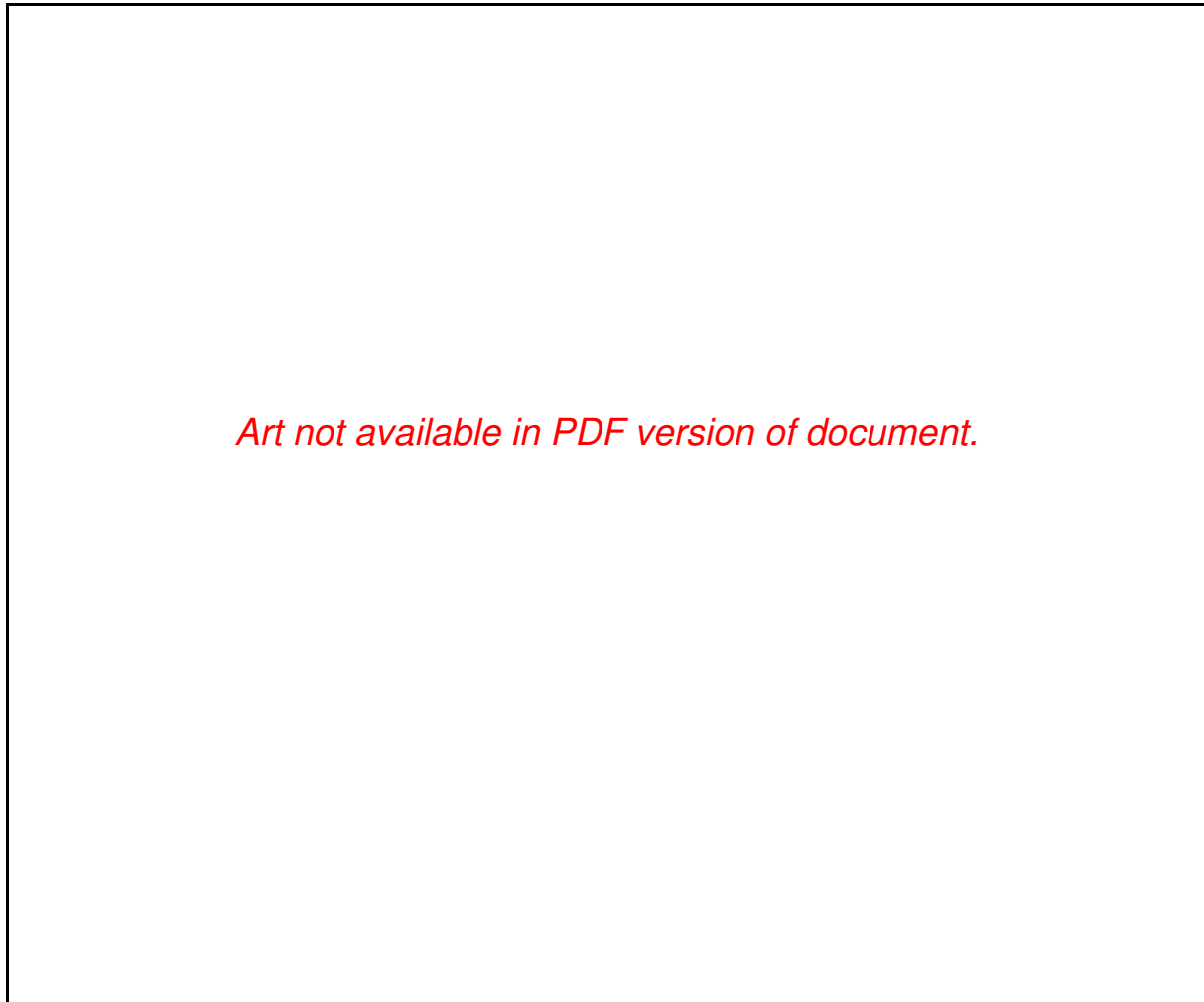
Figure 2-2. VME-MXI with INTX Parts Locator Diagram

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 your VME-MXI includes 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 2-3(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 2-3(b).

Figure 2-3. VMEbus System Controller Selection

VME Address

The MC-MXI and the VME-MXI use VME short (A16) address space for their configuration, communication, and status registers. Because both boards are not only 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 kilobytes of A16 space to each board. The starting address of a device's address space can be calculated using the following formula:

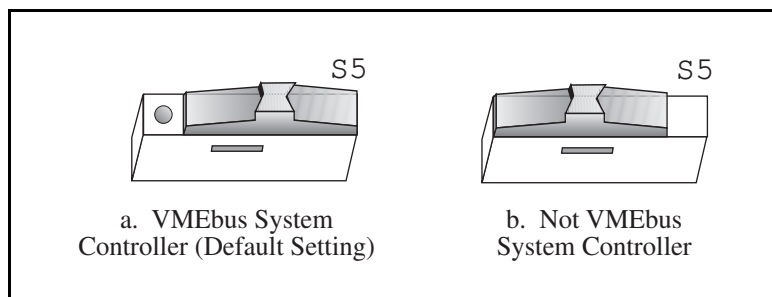$$\text{starting address (A16 space)} = (\textit{logical address} * 40h) + C000h$$

The MC-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 MC-MXI will use A16 address space from C000h through C03Fh. The logical address of the MC-MXI is software configurable, but should not be changed when used with a VME system.

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 MC-MXI or the VME-MXI. If they do, change the other VME devices so that they occupy unique portions of address space. 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 computer with an MC-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 2-4.
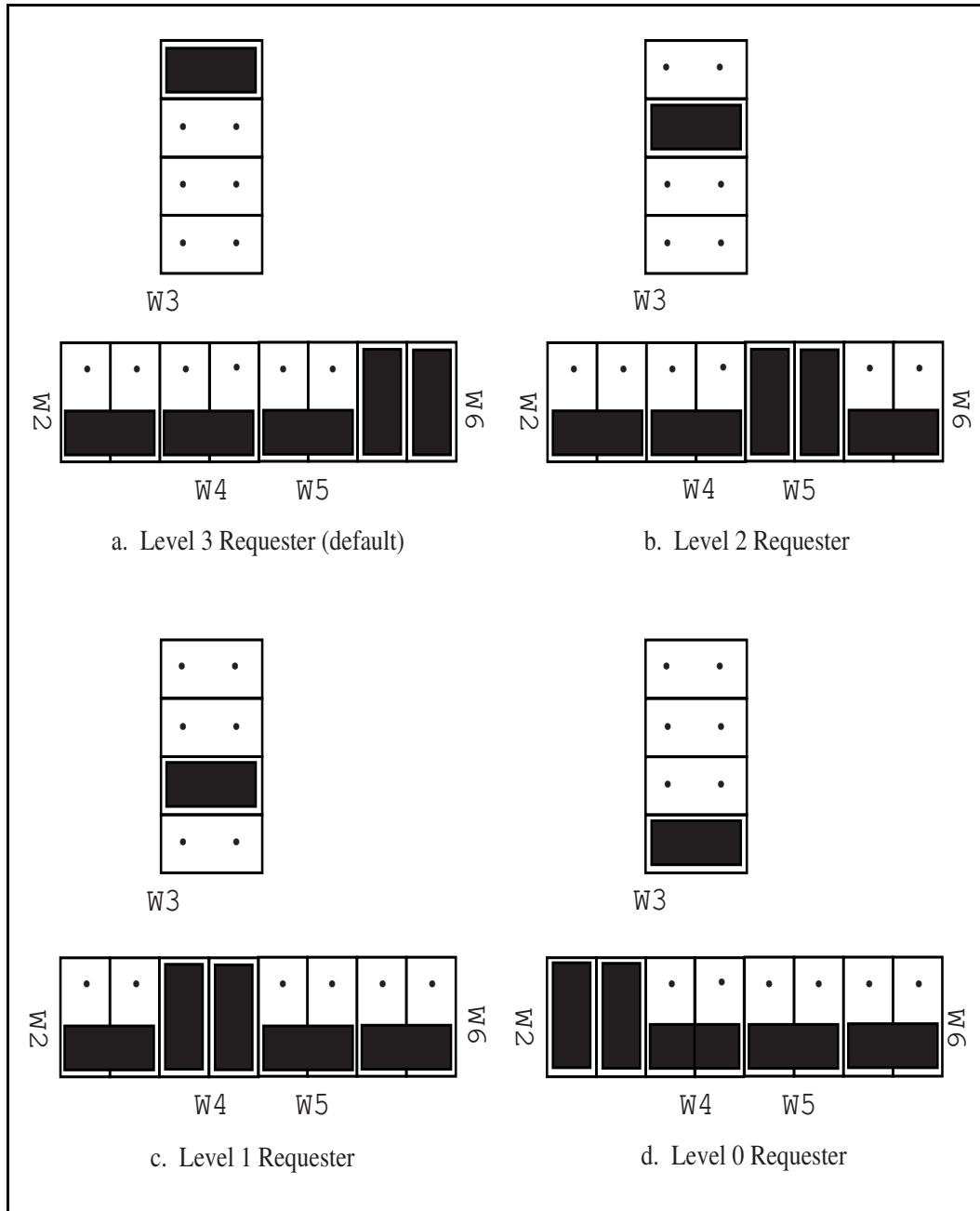
Figure 2-4. VME-MXI VMEbus Requester Jumper Settings

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 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 2-5.
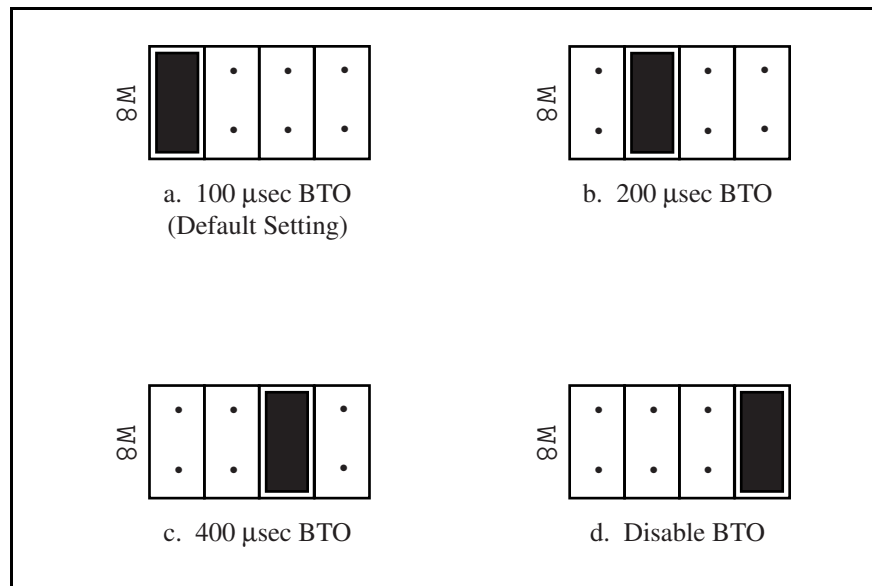
Figure 2-5. VMEbus Timeout Value Selection

Step 6. Install the VME-MXI Hardware

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's 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.

Step 7. 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-MC2010 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.

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 MC-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.

When the MXIbus cable is properly connected, turn on the VME chassis and then the Micro Channel computer. After all devices are powered on and running properly, you may run the Resource Manager software utility to configure your system.

Chapter 3
NI-VXI Software Installation and Configuration

This chapter lists all the programs and files located on the NI-VXI distribution diskettes, contains the instructions for installing the NI-VXI software, and describes how to use the `VXIEDIT` and `VXITEDIT` configuration editors to configure the NI-VXI software. This chapter also discusses programming information for you to consider when developing applications that use the NI-VXI driver.

The NI-VXI software package is a comprehensive set of utility programs and driver software routines that you can use to configure your MXIbus-to-VME link and program your VME system. The NI-VXI software package was actually developed for the VXI-MC2010 kit for VXI 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 only apply to VXI systems and are not used in VME systems. Refer to the remainder of this chapter and to the *NI-VXI C Software Reference Manual for VME* to determine which of the NI-VXI software package utility programs and driver functions are used for VME and which apply only to VXI.

NI-VXI Software Overview

The following files make up the NI-VXI software.

Main Programs and Files

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

- `@0DFF.ADF` is the adapter description file for the MC-MXI interface board.

- `VXIINIT.EXE` is the MC-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 MC-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 multimainframe 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 VME (and/or 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 computer can access the VMEbus. This program also displays the system configuration information generated by `RESMAN` 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, although the latter is not recommended. 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 table file is used only in a hybrid VME/VXI system, because VXI devices have manufacturer IDs, but VME devices do not.

- `MODEL.TBL` contains the database of model names, manufacturer names, and the model codes numbers. This table file is used only in a hybrid VME/VXI system.

- `DEVICE.TBL` contains the database of device names, manufacturer names, model names, and frame and slot associations for devices in the system. This table file is used only in a hybrid VME/VXI system.

- `NONVXI.TBL` contains the data base for all non-VXI (VME) devices in the system. This file is used for 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 used only in a hybrid VME/VXI system.

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

- `CREG.TBL` contains system information used by the Resource Manager. This table file is used only in a hybrid VME/VXI system.

- `REGS.TBL` contains the register names of VXI devices. This table file is used only in a hybrid VME/VXI system.

- `WSCMDS.TBL` contains the VXI Word Serial command values. This table file is used only in a hybrid VME/VXI system.

- `VXIBUS.CFG` contains the MC-MXI MXIbus configuration information.

- `VXILA.CFG` contains the MC-MXI logical address configuration information.

- `VXIDEV.CFG` contains the MC-MXI device-specific configuration information.

- `VXIMF.CFG` 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 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.

Installing the Software

The NI-VXI distribution disks that came with your MC-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.

1. The NI-VXI software requires approximately 3 megabytes of free space on your hard disk. Create the necessary free space on the hard disk before starting the installation.

2. Run `INSTALL.EXE` on distribution disk 1. You can run this program from the Windows Program Manager's **File|Run** menu, 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 will install all the necessary files (for a complete or partial installation, as user-specified) on the hard disk. In addition, it will optionally modify the `AUTOEXEC.BAT`, `PROGMAN.INI`, `SYSTEM.INI`, and `WIN.INI` initialization files accordingly. 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. However, if you do install both C libraries, you must let `INSTALL` know which one to make the default library.

- Source drive: This is the floppy drive in which you inserted the distribution disk.

- 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 reflect the location of the NI-VXI library. If you choose to let `INSTALL` do this, you must also enter the drive where `AUTOEXEC.BAT` can be found.

- Modifying Windows initialization files: You can choose whether `INSTALL` modifies the `PROGMAN.INI`, `SYSTEM.INI`, and `WIN.INI` files to reflect the location of the NI-VXI library. If you choose to let `INSTALL` do this, you must also enter the directory in which the Windows files can be found.

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

Note: `INSTALL` will *not* create a backup for you. If there are files you want to preserve, be sure to exit the installation program at this time and make a backup before continuing.

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. In either case, `INSTALL` will prompt you to be certain you really want to quit at that time. If you choose to run the `INSTALL` program again, it starts over and recopies all the necessary files. Of course, if you made any changes from the default configuration, you will need to specify those changes once again.

As mentioned previously, `INSTALL` *optionally* modifies the `AUTOEXEC.BAT`, `PROGMAN.INI`, `SYSTEM.INI`, and `WIN.INI` files. For each file that you *do* modify, it changes the old file's extension to `.BAK` and keeps it for safety purposes. For each file that you do *not* modify, it creates a file in the NI-VXI directory with the changes it proposes you make to the files. The files that it creates in this situation have the same filename but with the extension `.VXI`. For example, if you choose not to let `INSTALL` modify `AUTOEXEC.BAT`, it will create a file in the NI-VXI directory called `AUTOEXEC.VXI`, with changes you will need to make to `AUTOEXEC.BAT`.

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. 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, refer to the `AUTOEXEC.VXI` file in your NI-VXI directory 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.

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>
```

If you choose not to let `INSTALL` modify your `SYSTEM.INI` file, refer to the `SYSTEM.VXI` file in your NI-VXI directory for suggestions on how to make the necessary 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.

If you choose not to let `INSTALL` modify your `WIN.INI` file, refer to the `WIN.VXI` file in your NI-VXI directory for suggestions on how to make the necessary changes yourself.

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

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.

Once the NI-VXI package is installed, run `VXIINIT.EXE` and then `RESMAN.EXE`.

Configuring the NI-VXI Software

Run `VXIINIT` to display the MC-MXI configuration settings. The default configuration is MXIbus System Controller, Resource Manager, Message-Based device. To change the current settings or to view the configuration options, run `VXIEDIT` and select the Configuration Editor from the main menu.

You can use `VXIEDIT` or `VXITEDIT` to edit the description of the MC-MXI or any other device that is installed in the system. You may find the `VXIEDIT` program easier to use because of its structure. `VXIEDIT` 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. You may prefer to use `VXITEDIT` if you do not want to take the extra step to enter DOS to reach `VXIEDIT`.

You can use the Non-VXI Device Editor from the main menu 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. It is recommended 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 can also enter manufacturer name, model name and other identification information for your VME devices. Be sure to configure the `InSystem?` parameter to instruct the Resource Manager that this particular VME device is indeed installed in your system.

Keep in mind that the Resource Manager will only use the configuration information for those devices that you indicated `InSystem`. 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 and device names.

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.

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 can run `VXIEDIT` the same way you would run any DOS program. (One way to run a DOS program from Windows is to click on the *DOS Prompt* icon, then run the program.) After you enter `VXIEDIT`, select the Configuration Editor from the main menu (See Figure 3-1).

Art not available in PDF version of document.
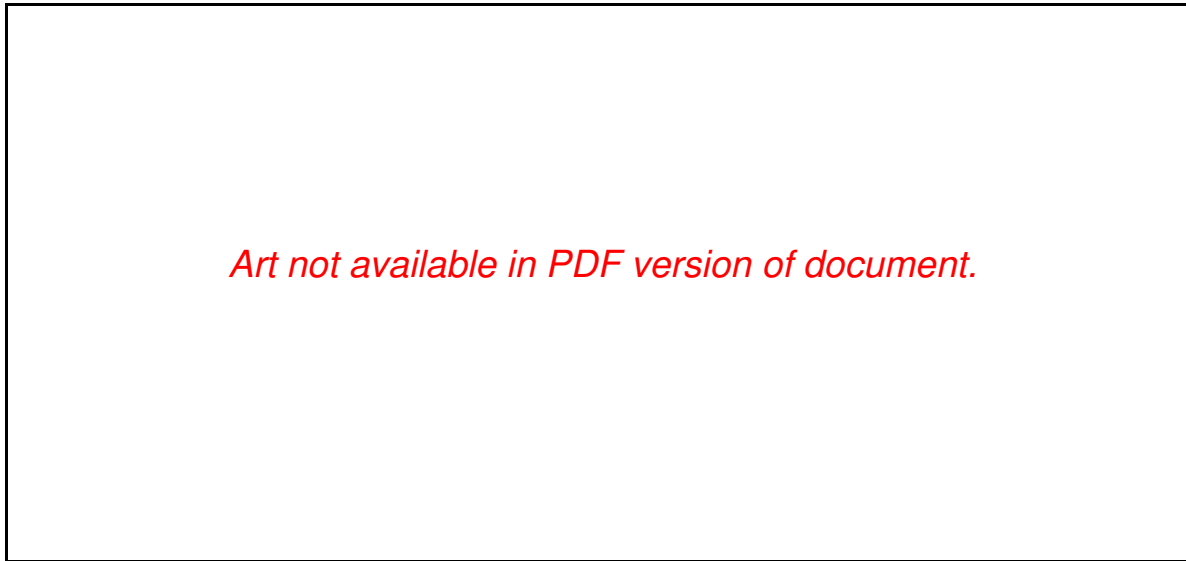
Figure 3-1. VXIEDIT Main Menu

The Configuration Editor, Non-VXI Device Editor, and Resource Manager Display are used in VME systems. Use the Configuration Editor to configure details of the MC-MXI hardware configuration. Use the Non-VXI Device Editor to configure the address requirements of VME devices in your VME system so that the Resource Manager program can automatically set up the MXI link. The Resource Manager Display lets you examine the status of the address space configuration after the Resource Manager program has executed. 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 (VXI is a superset of VME). The Interrupt Configuration Editor and Utility Bus Configuration Editor can be used in VME systems, but only in multimainframe system configurations.

Configuration Editor

When you select the Configuration Editor from the main menu, the Logical Address Configuration Editor menu (shown in Figure 3-2) is displayed. The following sections describe the various configuration options available under the Logical Address Configuration and Bus Configuration menus.

Select *Next* and *Previous* to move between the two 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, 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 3-2 shows the Logical Address Configuration Editor (VXIla) menu.

<div style="border:1px solid black; text-align:center; padding:40px;">

Art not available in PDF version of document.

</div>
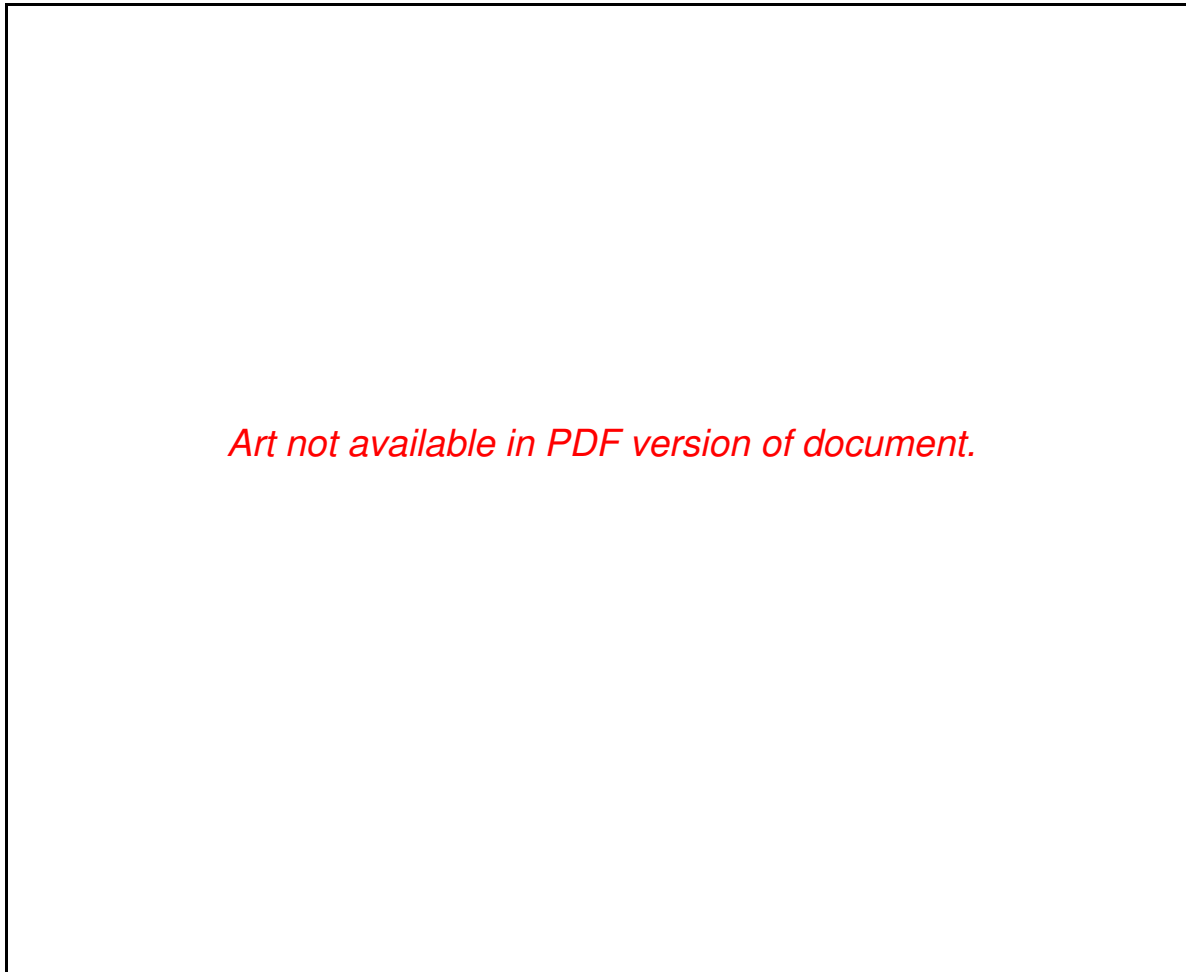
Figure 3-2. MC-MXI Logical Address Configuration Editor

Table 3-1 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 3-1. Logical Address Configuration Characteristics

Characteristic	Description	Default Value
Logical Address	An 8-bit number that uniquely identifies the MC-MXI. It can be configured as a static device with a logical address range from 0 through 254. If it is set to 0, it is the VXIbus Resource Manager. Logical Address 255 configures the MC-MXI as a dynamic device, as specified in the *VXIbus System Specification*. Range = 0 to 255	0
Device Type	Indicates the classification of the MC-MXI: EXT = Extended MBD = Message-Based RBD = Register-Based	MBD
Address Space	Indicates the address spaces used by the MC-MXI: A16/A24 A16/A32 A16 only	A16 only
Shared RAM Window Base	Indicates the size of local base of the Shared RAM. The valid settings cover the entire address space. For A16/A24 the increment is 1 megabyte, and for A16/A32 the increment is 256 megabytes.	DISABLED

(continues)

Table 3-1. Logical Address Configuration Characteristics (continued)

Characteristic	Description	Default Value
Shared RAM Window Size	Indicates the size of VME Shared RAM. The MC-MXI local RAM can be slave-accessed in either A24 or A32 space in D08, D16, and D32 data transfer modes. As an A16-only device, this field is disabled. When the MC-MXI is an A16/A24 device, the possible values are as follows: 4K, 8K, 16K, 32K, 64K, 128K, 256K, 512K, 1M, 2M, 4M, 8M When the MC-MXI is an A16/A32 device, the possible values are as follows: 64K, 128K, 256K, 512K, 1M, 2M, 4M, 8M, 16M, 32M, 64M, 128M, 256M, 512M, 1024M, 2048M	DISABLED
Shared Memory Pool (Windows)	Indicates the size of memory that is allocated on Windows startup to be used exclusively by VXImdmAlloc. This value is not used by the DOS library. Range = 0 to 65535 kilobytes	0 kilobytes
Servant Area Size	Servant area size is supplied when the MC-MXI receives the *Read Servant Area* command from the Resource Manager. The Servant area size is an 8-bit value (0 through 255) that indicates the MC-MXI Servant area. The Servant area begins at the logical address following the MC-MXI's logical address, and includes *N* contiguous logical addresses, where *N* is the value of the Servant area size. Not used in VME systems. Range = 0 to 255	0
Protocol Register	Copy of the Protocol register, indicating which protocols the device supports. (Refer to the *VXIbus System Specification*.) Not used in VME systems.	0x0FF0

(continues)

Table 3-1. Logical Address Configuration Characteristics (continued)

Characteristic	Description	Default Value
Response for Servant-Side Word Serial *Read Protocol* query	The protocols that are supported, which are supplied when the MC-MXI receives the *Read Protocol* query from the Resource Manager. (Refer to the *VXIbus System Specification*.) Not used in VME systems.	 0x8448
Number of Handlers	The number of Interrupt handlers that the application requires the MC-MXI to support (minimum of 1) Range = 0 to 7	 1
Number of Interrupters	The number of Interrupters that the application requires the MC-MXI to support. Range = 0 to 7	 0
Resource Manager Delay	Time in seconds to wait after power-up or reset before configuring the VME hardware interface and checking for any VXIbus devices residing in the system. Range = 0 to 65535 sec	 5 sec

Bus Configuration

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

Art not available in PDF version of document.
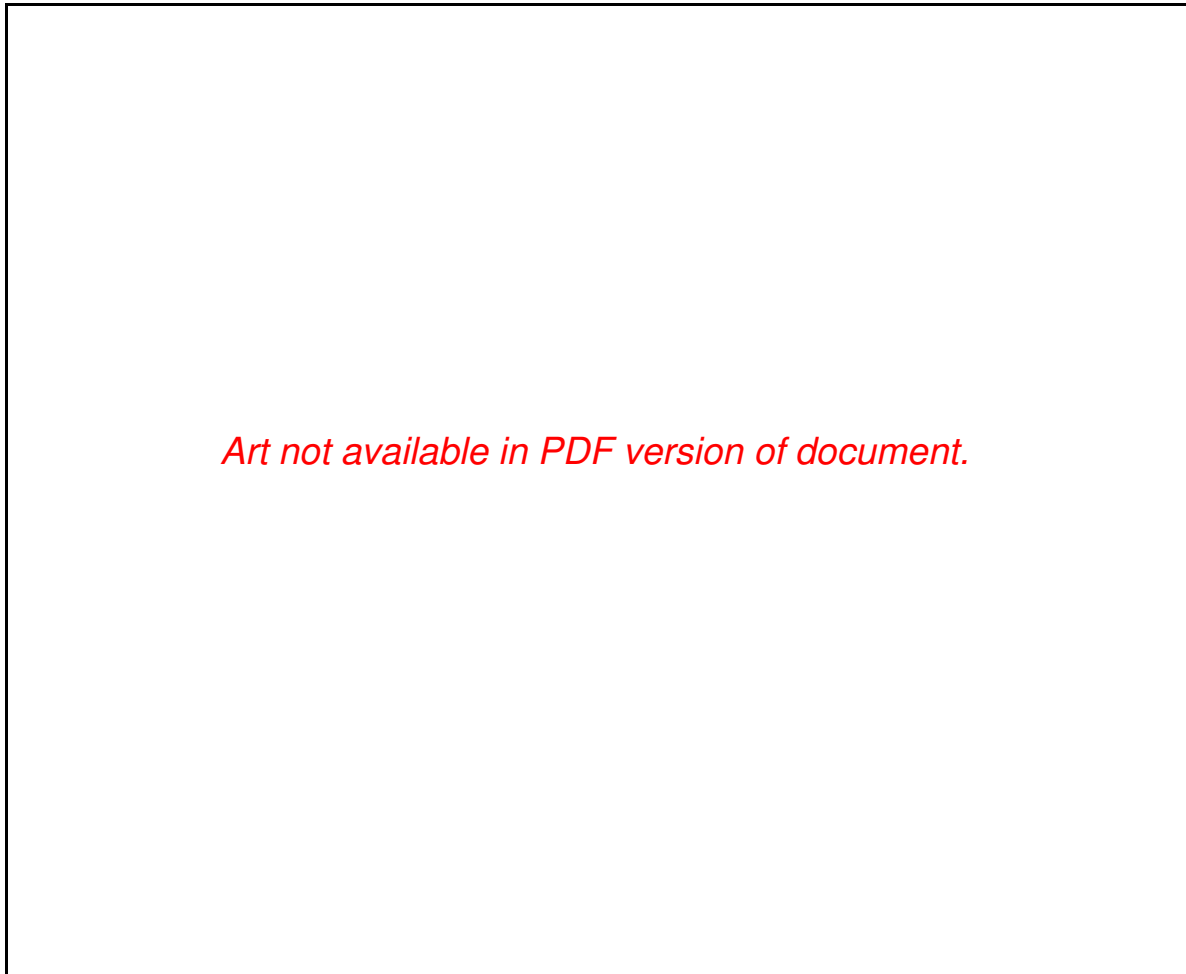
Figure 3-3. MC-MXI Bus Configuration Editor

Table 3-2 describes the characteristics of the options available under the Bus Configuration menu.

Table 3-2. 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 MC-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 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 MC-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

Non-VXI Device Editor

Select the Non-VXI Device Editor from the main menu. Figure 3-4 shows the configuration available from this menu.

Art not available in PDF version of document.
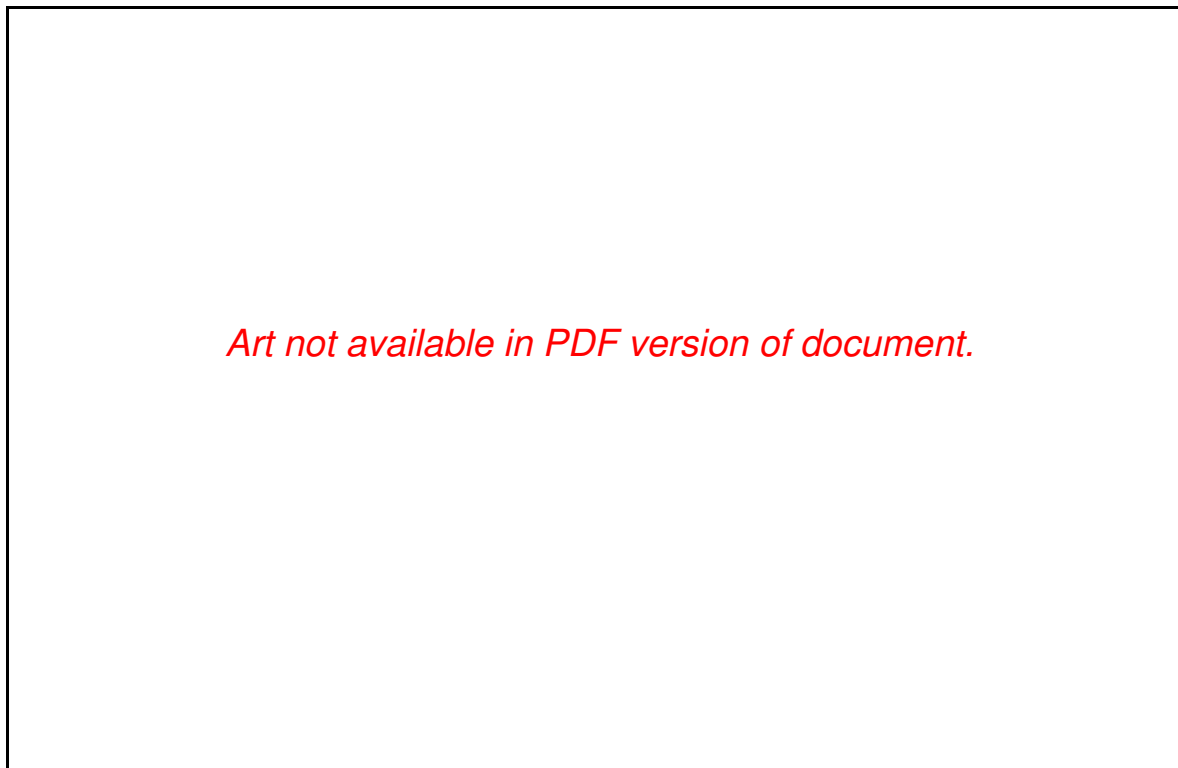
Figure 3-4. Non-VXI Device Editor

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. 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.

You need to assign VME devices pseudo-logical addresses (between 256 and 511 inclusive), as well as device names, manufacturer names, and pseudo-model names. The `Frame` field should contain the logical address of the VME-MXI in the chassis. The default logical address is 1.

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 typing a *y* for yes or an *n* for no), depending on whether that device is actually present in your system.

In the example in Figure 3-4, we have a VME memory device in the system that requires 16 bytes of memory in A16 space. We assign it a pseudo-logical address between 256 and 511. Set its `Frame` field to the logical address of the VME-MXI in the chassis (the default logical address for the VME-MXI in your kit is 1). Next, assign it a device name, manufacturer name, and model name. Finally, set its A16 memory base and size fields according to its memory requirements, and set the `InSystem?` field.

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`, and `DVXI.CFG`. The `VXIDEV.CFG` file, which contains some system-dependent information used by the `VXIINIT` program, should not be modified. 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 *VME-MC2010 Hardware and Software Configuration Form* in Appendix B.

Using VXITEDIT

As an alternative to `VXIEDIT`, you can use `VXITEDIT`, a text-based VXI resource editor. `VXITEDIT` is a Windows application and can be run 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 on the Windows menu. 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`, and `DVXI.CFG`. The `VXIDEV.CFG` file, which contains some system-dependent information used by the `VXIINIT` program, should not be modified. 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 *VME-MC2010 Hardware and Software Configuration Form* in Appendix B.

Developing Your Application Program

To verify that the driver software has been installed successfully, you can execute `VXIINIT` from the root directory. Now you can proceed with development of your VXI application software.

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 VME 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. Conflict occurs 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.

The `EXAMPLES` subdirectories contain various example programs along with makefiles 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 earlier in this chapter.

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.

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

Multi-application support is another feature in the NI-VXI Windows library. You can have several applications that use the NI-VXI library running simultaneously. In addition, you can have multiple instances of an 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. Because the NI-VXI functions execute to their completion without yielding control of the microprocessor, function calls from different applications do not inadvertently conflict with one another.

However, you do need to be careful in cases when you have multiple applications using the low-level VXIbus access functions. You should follow the protocol of calling the `MapVXIAddress` function first before attempting to perform low-level VXIbus access with `VXIpeek` or `VXIpoke`. The `MapVXIAddress` call prevents the hardware interface setting of the MC-MXI from being altered by other applications. Your application should always call the `UnMapVXIAddress` function immediately after the accesses are complete.

Appendix A
Specifications

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

MC-MXI

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

Capability Codes

MXIbus

Capability Code	Description
MA32	Master Mode A32, A24 and A16 addressing
MBLT	Master Mode block transfers
SA32	Slave Mode A32, A24, and A16 addressing
SBLT	Slave Mode block transfers
MD32	Master Mode D32, D16, and D08 data sizes
SD32	Slave Mode D32, 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

Micro Channel

Capability Code	Description
MCMD32	Master D32, D16, and D08 data sizes
MCSD32	Slave D32, D16, and D08 data sizes
MCFAIR	Can be a fair Micro Channel requester
AM	Can function as an MC alternate master
LOCK	Can lock the Micro Channel for indivisible transfers
DMA32	Supports D32, D16 or D08 DMA transfers
INT	Can be a Micro Channel Interrupter

Electrical

Source	Typical	Direct Current (max)
+5 VDC	3.0 A	3.5 A

Environmental

Component Temperature   0° to 70° C (32° to 158° F) operating
-55° to 150° C (-67° to 302° F) storage

Emissions   FCC Class A

Relative Humidity   0% to 95% noncondensing; operating
0% to 100% noncondensing; storage

Safety   Not applicable

Shock and Vibration   Not applicable

Physical

Board Size   Standard full-length Micro Channel board
(11.5 in. by 3.475 in.)

Connectors   Single fully implemented MXIbus connector

Slot Requirements   Single 32-bit Micro Channel slot

Reliability

MTBF   Contact Factory

Requirements

Memory space required   64 kilobytes or 32,832 kilobytes

I/O space required   4 kilobytes

Timing

Master Mode   **Slave Mode**

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

Transfer Type	Transfer Rate
Write	340 nsec
Read	440 nsec
Block Write	310 nsec
Block Read	360 nsec

Other

Daisy-Chain Delay   120 nsec
(Passing GIN to GOUT or GOUT generation from System Controller)

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

Electrical

Source	Typical	Maximum
+5 VDC	5.25 A	7.0 A

Environmental

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

Board size   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

VXI Keying Class   Class 1 TTL

Fully compatible with VMEbus specification

Reliability

MTBF   Contact Factory

Requirements

A16 Space   64 bytes

Timing

Master Mode

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

Slave Mode

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

Other

Daisy-Chain Delay   120 nsec max
(Passing GIN to GOUT or GOUT generation from System Controller)

Appendix B
Customer Communication

For your convenience, this appendix contains a form 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.

Technical Support

Because it helps us solve your problems more easily, we encourage you to fax us a completed copy of the *Technical Support Form* and the configuration forms from the user manuals of the National Instruments hardware or software you are using, even if you wish to contact us later by telephone. You can contact us by fax at any time at the following number:

(512) 794-5678

You can use the following numbers between the hours of 8:00 a.m. and 5:30 p.m. (central time) to call the National Instruments applications engineering department:

(512) 794-0100
(800) 433-3488 (toll-free in the U.S. and Canada)

For best service, please complete the *Technical Support Form* and any applicable configuration forms, record any error messages, and be available at your computer when calling for technical support.

Documentation Comments

You can use the *Documentation Comment Form* to express your opinion of our documents. Please mail or fax it according to the instructions printed at the bottom of the form.

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 _____M Display adapter _____

Mouse _____yes _____no Other adapters installed _____

Hard disk capacity _____M 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 _____

_____

_____

_____

VME-MC2010 Hardware and Software Configuration Form

Record the settings and revisions of your hardware and software on the line located to the right of each item. Complete 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 MC-MXI and Microsoft Windows*) _____

- C Programming Language Interface Revision _____

- MC-MXI Logical Address _____

- MC-MXI Device Type _____

- MC-MXI Address Space _____

- MC-MXI Shared RAM Window Base _____

- MC-MXI Shared RAM Window Size _____

- MC-MXI Shared Memory Pool (Windows) _____

- MC-MXI Servant Area Size _____

- MC-MXI Protocol Register _____

- MC-MXI Response for Servant-Side
  Word Serial *Read Protocol* Query _____

- Number of MC-MXI Handlers _____

- Number of MC-MXI Interrupters _____

- MC-MXI Resource Manager Delay _____

- MC-MXI is MXIbus System Controller? _____

- MXIbus (Fair) Controller? _____

- MC-MXI Memory Base Window Address _____

- MC-MXI Base I/O Address _____

- MC-MXI Interrupt Level(s) _____

- MC-MXI Arbitration Level(s) _____

- VME-MXI Installed in Slot 1? _____

- VME-MXI is VMEbus System Controller? _____

- VME-MXI Address _____

- VME-MXI VMEbus Request Level _____

- VMEbus BTO Timeout Value _____

- VME-MXI Hardware Revision _____

- MC-MXI Hardware Revision _____

Other Products

- Computer Make and Model _____

- Microprocessor _____

- Clock Frequency (Bus and Microprocessor) _____

- Type of Video Board Installed _____

- Microsoft Windows Version _____

- Windows Mode (Enhanced, Standard, or Real) _____

- Programming Language Version _____

- Other Boards in System _____

- Base I/O Address of Other Boards _____

- Arbitration Levels of Other Boards _____

- Interrupt Level of Other Boards _____

- VMEbus Chassis Make and Model _____

- Other VME/VXIbus Devices in System _____

- Static Logical Addresses of Other 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 VME-MC2010 and the NI-VXI™ Software for Microsoft Windows**

Edition Date: **January 1993**

Part Number: **320430-01**

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

Glossary

_____

Prefix	Meaning	Value
n-	nano-	10^{-9}
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 kilobyte *short* address space. In VXI, the upper 16 kilobytes of A16 space is allocated for use by VXI devices configuration registers. This 16 kilobyte region is referred to as VXI configuration space.

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

A32 space   VXIbus address space equivalent to the VME 4 gigabyte *extended* address space.

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.

B

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.
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
chassis extender	A device such as the VME-MXI that interfaces a VMEbus chassis to an interconnect bus. It routes bus transactions from the VMEbus to the interconnect bus or vice versa. A chassis extender has a set of registers that defines the routing mechanisms for data transfers, interrupts, and utility bus signals, and has optional VMEbus Slot 1 capability.
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.

D

daisy-chain	A method of propagating signals along a bus, in which the devices are prioritized on the basis of their position on the bus.
Data Transfer Bus	DTB; one of four buses on the VMEbus backplane. The DTB is used by a bus master to transfer binary data between itself and a slave device.
DIP	Dual Inline Package
DMA	Direct Memory Access; a method by which data is transferred between devices and internal memory without intervention of the central processing unit.
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

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

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

IEEE 1014   The VMEbus specification

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

K

K   1,024; kilobytes of memory

L

LED   Light Emitting Diode

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

M

M	1,048,576; 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.
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.

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

sec	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.
statically configured device	A device whose logical address cannot be set through software; that is, it is not dynamically configurable.

T

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

V

V	volts
VDC	volts direct current
VIC	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. Called *VICTEXT* in text-based applications.
VME	Versa Module Eurocard or IEEE 1014
VMEbus System Controller	A device configured for installation in 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 backplane, or both.
VXI	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	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. Called *VXITEDIT* in text-based applications.

W

Word Serial Protocol	The simplest required communication protocol supported by Message-Based devices in the VXIbus system. It utilizes the A16 communication registers to transfer data using a simple polling handshake method.