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.

APEX WAVES

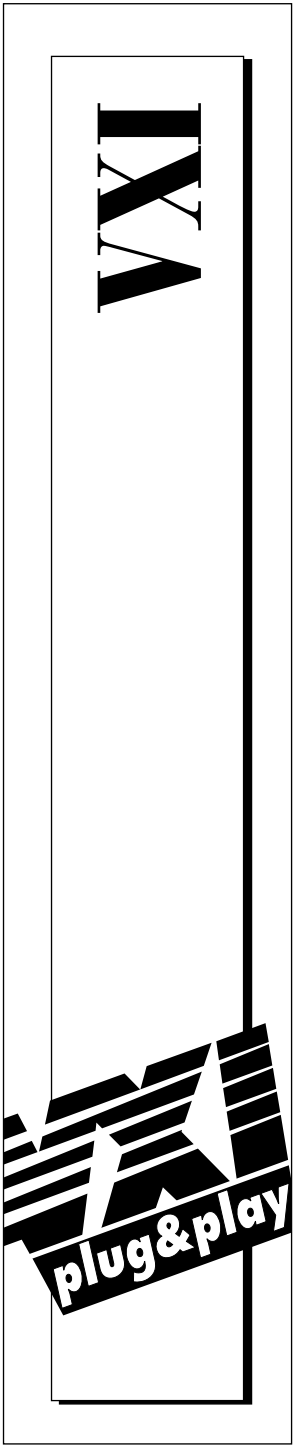
Bridging the gap *between the manufacturer and your legacy test system.*

1-800-915-6216

www.apexwaves.com

sales@apexwaves.com

Request a Quote CLICK HERE ***VXIpc-860***



NI-VXI™ Software Manual for the VXIpc™ 800/700 Series

April 1997 Edition
Part Number 321125E-01

Internet Support

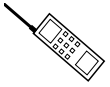
support@natinst.com
E-mail: info@natinst.com
FTP Site: ftp.natinst.com
Web Address: http://www.natinst.com

Bulletin Board Support

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

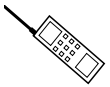
Fax-on-Demand Support

(512) 418-1111



Telephone Support (U.S.)

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



International Offices

Australia 02 9874 4100, Austria 0662 45 79 90 0, Belgium 02 757 00 20,
Canada (Ontario) 905 785 0085, Canada (Québec) 514 694 8521, Denmark 45 76 26 00,
Finland 09 527 2321, France 01 48 14 24 24, Germany 089 741 31 30, Hong Kong 2645 3186,
Israel 03 5734815, Italy 02 413091, Japan 03 5472 2970, Korea 02 596 7456,
Mexico 5 520 2635, Netherlands 0348 433466, Norway 32 84 84 00, Singapore 2265886,
Spain 91 640 0085, Sweden 08 730 49 70, Switzerland 056 200 51 51, Taiwan 02 377 1200,
U.K. 01635 523545

National Instruments Corporate Headquarters

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

Important Information

Warranty

The 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

Trademarks

LabVIEW®, MANTIS™, MITE™, NI-488.2™, NI-VXI™, TIC™, and VXIpc™ are trademarks of National Instruments Corporation.

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

WARNING REGARDING MEDICAL AND CLINICAL USE OF NATIONAL INSTRUMENTS PRODUCTS

National Instruments products are not designed with components and testing intended to ensure a level of reliability suitable for use in treatment and diagnosis of humans. Applications of National Instruments products involving medical or clinical treatment can create a potential for accidental injury caused by product failure, or by errors on the part of the user or application designer. Any use or application of National Instruments products for or involving medical or clinical treatment must be performed by properly trained and qualified medical personnel, and all traditional medical safeguards, equipment, and procedures that are appropriate in the particular situation to prevent serious injury or death should always continue to be used when National Instruments products are being used. National Instruments products are NOT intended to be a substitute for any form of established process, procedure, or equipment used to monitor or safeguard human health and safety in medical or clinical treatment.

Table of Contents

About This Manual

Chapter 1
Introduction

Chapter 2
NI-VXI Software Installation

Chapter 3
NI-VXI Configuration Utility

Chapter 4
Using the NI-VXI Software

Appendix A
NI-VXI Software Overview

Appendix B
Common Questions

Appendix C
Customer Communication

Glossary

Figures

About This Manual

This manual contains instructions for installing and configuring the NI-VXI bus interface software for Microsoft operating systems and the VXIpc 800/700 Series. The NI-VXI software works with all models of the National Instruments VXIpc 800/700 Series embedded computers. The NI-VXI software is fully VXI*plug&play* compliant.

Microsoft operating systems include DOS, Windows 3.1, Windows NT, and Windows 95. This manual uses the term *Windows 95/NT/3.1* when information applies to all three Windows operating systems.

Organization of This Manual

This manual is organized as follows:

- Chapter 1, *Introduction*, describes the NI-VXI software for the VXIpc 800/700 Series, lists what you need to get started, includes a brief description of the NI-VXI software, and lists optional software.
- Chapter 2, *NI-VXI Software Installation*, contains the instructions for installing the NI-VXI software.
- Chapter 3, *NI-VXI Configuration Utility*, contains instructions for using the VXI Resource Editor utility of the NI-VXI software to configure the VXIpc 800/700 Series embedded computer and the VXI-MXI-2 or VME-MXI-2 mainframe extender.
- Chapter 4, *Using the NI-VXI Software*, discusses programming information for you to consider when developing applications that use the NI-VXI driver.
- Appendix A, *NI-VXI Software Overview*, lists and describes the main programs and files that make up the NI-VXI software.
- Appendix B, *Common Questions*, addresses common questions you may have about using the NI-VXI bus interface software on the VXIpc 800/700 platform.

- Appendix C, *Customer Communication*, contains forms you can use to request help from National Instruments or to comment on our products and manuals.
- The *Glossary* contains an alphabetical list and description of terms used in this manual, including abbreviations, acronyms, metric prefixes, mnemonics, and symbols.

Conventions Used in This Manual

The following conventions are used in this manual:

< >	Angle brackets enclose the name of a key on the keyboard (for example, <Page Down>).
♦	The ♦ symbol indicates that the text following it applies only to a specific product, a specific operating system, or a specific software version.
bold	Bold text denotes the names of menus, menu items, dialog box buttons or options, or LEDs.
bold italic	Bold italic text denotes a note, caution, or warning.
italic	Italic text denotes emphasis, a cross reference, or an introduction to a key concept. This font also denotes text from which you supply the appropriate word or value, as in Windows 3.*x*.
`italic monospace`	Italic text in this font denotes that you must supply the appropriate words or values in place of these items.
`monospace`	Text in this font denotes text or characters that are to be literally input from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, programs, subprograms, subroutines, device names, functions, variables, filenames, and extensions.
VXIpc 700 Series	The terms *VXIpc 700 Series* and *VXIpc-700* refer to a series of C-size, single-slot, VXI controllers. Currently, this series consists of the VXIpc-740 and VXIpc-745.
VXIpc 800 Series	The terms *VXIpc 800 Series* and *VXIpc-800* refer to a series of C-size, dual-slot, VXI controllers. Currently, this series consists of the VXIpc-850 and VXIpc-860, with various processors and in different speeds.
VXIpc 800/700 Series	The term *VXIpc 800/700 Series* refers to all models of the VXIpc 800 and VXIpc 700 Series.

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

How to Use This Documentation Set

Begin by reading the *Getting Started with Your VXIpc 800/700 Series* manual for basic instructions on setting up the hardware and software. This is a brief quick start manual that describes how to get started with your kit using the default hardware and software settings.

The *VXIpc 800/700 Series User Manual* contains more details about changing the hardware installation or configuration from the defaults, and using the hardware.

This manual, the *NI-VXI Software Manual for the VXIpc 800/700 Series*, contains more details about changing the NI-VXI software installation or configuration from the defaults, and using the NI-VXI software on the VXIpc 800/700.

When you are familiar with the material in the previous manual, you can begin to use the *NI-VXI User Manual*. This manual presents the concepts of VXI and prepares you for detailed explanations of the NI-VXI functions. Study the descriptions of each function given in the *NI-VXI Programmer Reference Manual* to fully understand the purpose and syntax of each function.

You can also access online help for Windows 95/NT/3.1 in the NI-VXI folder.

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

Related Documentation

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

- ANSI/IEEE Standard 1014-1987, *IEEE Standard for a Versatile Backplane Bus: VMEbus*
- ANSI/IEEE Standard 1155-1993, *IEEE VMEbus Extensions for Instrumentation: VXIbus*

- ANSI/VITA 1-1994, *VME64*
- VXI-6, *VXIbus Mainframe Extender Specification*, Rev. 1.0, VXIbus Consortium

Customer Communication

National Instruments wants to receive your comments on our products and manuals. We are interested in the applications you develop with our products, and we want to help if you have problems with them. To make it easy for you to contact us, this manual contains comment and configuration forms for you to complete. These forms are in Appendix C, *Customer Communication*, at the end of this manual.

Chapter 1

Introduction

This chapter describes the NI-VXI software for the VXIpc 800/700 Series, lists what you need to get started, includes a brief description of the NI-VXI software, and lists optional software. You should already have installed and configured the VXIpc 800/700 Series embedded computer according to the instructions in the *VXIpc 800/700 Series User Manual*.

The VXIpc 800/700 Series includes the two-slot VXIpc 800 Series and the one-slot VXIpc 700 Series. This manual describes these computers together as the VXIpc 800/700 Series when functionality applies to all models.

The VXIpc 800/700 Series controllers are C-size, embedded computers based on the Peripheral Component Interface (PCI) bus and Industry Standard Architecture (ISA). These computers are high-performance, easy-to-use platforms for controlling VXIbus systems, featuring complete VXI functionality through interactive utilities and function calls for C and BASIC.

What You Need to Get Started

- ❑ VXIpc 800/700 Series embedded computer and accessories
- ❑ VXIbus mainframe
- ❑ NI-VXI software media for the VXIpc 800/700 Series

The NI-VXI software is already installed on your VXIpc 800/700 computer. It is also included on disk in case you need to reinstall your software.

Software Description

The NI-VXI bus interface software for the VXIpc 800/700 Series includes a VXI Resource Manager, graphical and text-based versions of an interactive VXI resource editor program, a comprehensive library of software routines for VXI/VME programming, and an interactive control program. You can use this software to seamlessly control multiple-mainframe configurations and have software compatibility across a variety of VXI/VME controller platforms.

Software Configurations

There are four software configurations described in this manual:

- NI-VXI for DOS/Windows 3.1—You can use this version of the software to develop and run 16-bit DOS/Windows 3.1 applications. You can also use this software under Windows 95 if you intend to use 16-bit applications only.
- NI-VXI Upgrade for Windows 95—This is a compatibility release that extends your NI-VXI for DOS/Windows 3.1 to allow 32-bit applications running in Windows 95 to use the 16-bit driver. In this configuration you can run both 16-bit and 32-bit applications; however, the core of the driver is 16-bit. Applications developed using this driver will run with NI-VXI for Windows NT without the need to recompile.

♦ **VXIpc-860 Users**—The VXIpc-860 is designed for use with Windows 95/NT only.

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

Optional Software

Your VXIpc 800/700 kit includes the NI-VXI bus interface software. In addition, you can use the National Instruments LabVIEW and LabWindows®/CVI application programs and instrument drivers to ease your programming task. These standardized programs match the modular virtual instrument capability of VXI and can reduce your VXI/VMEbus software development time. These programs are fully VXI*plug&play* compliant and feature extensive libraries of VXI instrument drivers written to take full advantage of direct VXI control.

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

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

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

Chapter 2

NI-VXI Software Installation

This chapter contains the instructions for installing the NI-VXI software.

☞ **Note:** ***Because the software is already installed on your VXIpc 800/700, the information in this chapter is useful only if you are reinstalling the software.***

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

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

- **VXIpc-860 Users**—The VXIpc-860 is designed for use with Windows 95/NT only.

Using the Windows Setup Program (Windows 3.1)

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

Installing the NI-VXI Software

This section describes how to install the NI-VXI software for the VXIpc 800/700 and Windows 3.1. Please carefully read these directions along with any messages on the screen before making your selections.

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

Setup is an interactive, self-guiding program that installs the NI-VXI software and configures your system to use the NI-VXI software with the VXIpc 800/700. Follow these steps to perform the installation.

1. Insert the disk labeled *NI-VXI for VXIpc 800/700 for DOS/Windows*.
2. Select **Run...** from the Windows Program Manager's **File** menu and enter the following code, where *x* is your floppy drive (usually A or B).

   `x:\setup.exe`

   and press <Enter>.
3. Click on the **Next** option at the **Welcome** screen to start the installation.
4. Select the target directory for your installation. Although Setup prompts you to accept a default directory, you can select a different location by clicking on **Browse....**
   - If you do not already have NI-VXI on your computer, Setup prompts you to select the default directory of `C:\NIVXI`.
   - Setup detects if you already have a previous installation of NI-VXI on your computer, and prompts you to either overwrite the previous version or install the new version in a different directory. Only one copy of NI-VXI can be active on your computer at any time. If you install the new version of NI-VXI into a different directory, the old installation is disabled.

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

5. Select the type of installation you want.
   - **Typical** installation includes Windows and DOS drivers, utilities, and development files for all supported compilers.
   - **Compact** installation includes only the driver and utilities necessary to run applications written with NI-VXI.
   - **Custom** installation lets you select the target operating system and development environments according to your needs.

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

6. Select the name of the Program Manager folder that will contain icons for the NI-VXI software for VXIpc 800/700.

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

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

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

9. You can now use VXIedit to configure the hardware in your VXI system. Continue with Chapter 3, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in VXIedit.

Modifying the AUTOEXEC.BAT File

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

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

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

- The LIB variable should include the full path to the subdirectories that contain the C libraries for the compiler you choose to install.
- The INCLUDE variable should include the full path to the subdirectory that contains the NI-VXI include files. By default, the include files reside in the INCLUDE subdirectory of the NI-VXI directory.
- The NIVXIPATH variable should contain the full path to the NI-VXI directory.

Completing the Software Installation

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

After the NI-VXI software is installed, run VXIINIT.EXE and then RESMAN.EXE. You need to run VXIinit to initialize the VXIpc 800/700 before you perform any VXI operations and after each computer reset. RESMAN is the National Instruments Resource Manager, which you must run every time the chassis power is cycled so that your application can access devices in the VXI chassis.

After you run VXIinit and RESMAN, you are ready to use the NI-VXI Resource Editor program VXIedit to interactively configure the hardware in your system. Continue with Chapter 3, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in VXIedit.

Using the Windows Setup Program (Windows 95)

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

System Preparation

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

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

If you have been using your VXIpc 800/700 under Windows 95 with either the NI-VXI software for DOS/Windows 3.1 or the NI-VXI Windows 95 upgrade, you need to remove the Plug and Play information from the Windows 95 Device Manager before installing the new NI-VXI software.

Follow these steps to remove the VXIpc 800/700 information.

1. Double-click on the **System** icon under **Start**»**Settings**»**Control Panel**.
2. Select the **Device Manager** tab from the **System Properties** dialog that appears.
3. Click on the **View devices by type** button and double-click on the **Other Devices** icon.
4. Select the VXIpc 800/700 from the list of devices under **Other Devices**. It will appear under the name **PCI Card** or **PCI Bridge** and will have a circled exclamation point through the **?** (question mark) icon.
5. Click on the **Remove** button.
6. Click **OK** to exit the Device Manager after removing the device information.

Installing the NI-VXI Software

This section describes how to install the NI-VXI software for the VXIpc 800/700 and Windows 95. Please carefully read these directions along with any messages on the screen before making your selections.

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

Setup is an interactive, self-guiding program that installs the NI-VXI software and configures your system to use the NI-VXI software with the VXIpc 800/700.

Follow these steps to perform the installation.

1. Insert the disk labeled *NI-VXI for VXIpc 800/700 Series for Windows 95*.
2. Select **Run...** from the **Start** menu and enter the following text, where *X* is your floppy drive (usually A or B).

   `X:\setup.exe`

   and press <Enter>.

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

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

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

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

5. Select the components you want to install.
   - **NI-VXI for Windows 95** includes only the driver and utilities needed to configure and use your VXI or VME system.
   - **Microsoft C/C++ Development Files** includes the components necessary to develop applications using the Microsoft Visual C++ compiler. Please read the *Modifying the Environment* section if you choose to install these files.
   - **Borland C/C++ Development Files** includes the components necessary to develop applications using the Borland C/C++ compiler. Please read the *Modifying the Environment* section if you choose to install these files.
6. Select the name of the program folder that will contain icons for the NI-VXI software for VXIpc 800/700.

   At this point Setup copies the necessary files to your hard drive and creates program icons.
7. When the installation process completes, you must exit Windows and reboot your computer for the changes to take effect. You can let the Setup program reboot your computer by selecting **the Restart computer** option on the last screen. Click on **Finish** to end the installation.
8. If you backed up the manufacturer and model name files, you should restore them to the `TBL` subdirectory of your NI-VXI directory before running `VXIedit`.

9. You can now use VXIedit to configure the hardware in your VXI system. Continue with Chapter 3, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in VXIedit.

Modifying the Environment

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

Completing the Software Installation

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

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

After you run RESMAN, you are ready to use the NI-VXI Resource Editor program VXIedit to interactively configure the hardware in your system. Continue with Chapter 3, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in VXIedit.

Using the Windows Setup Program (Windows NT)

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

Installing the NI-VXI Software

This section describes how to install the NI-VXI software for the VXIpc 800/700 and Windows NT. Please carefully read these directions along with any messages on the screen before making your selections.

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

Setup is an interactive, self-guiding program that installs the NI-VXI software and configures your system to use the NI-VXI software with the VXIpc 800/700. Follow these steps to perform the installation.

1. Insert the disk labeled *NI-VXI for VXIpc 800/700 Series for Windows NT*.
2. Select **Run...** from the **Start** menu on the taskbar or from the Program Manager **File** menu and enter the following text, where *X* is your floppy drive (usually A or B).

   `X:\setup.exe`

   and press <Enter>.
3. Click on the **Next** button at the **Welcome** screen to start the installation.
4. Select the target directory for your installation. Although Setup prompts you to accept the `C:\NIVXI` directory by default, you can select a different location by clicking on **Browse...**.

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

5. Select the components you want to install.
   - **NI-VXI for Windows NT** includes only the driver and utilities needed to configure and use your VXI or VME system.
   - **Microsoft C/C++ Development Files** includes the components necessary to develop applications using the Microsoft Visual C++ compiler. Please read the *Modifying the Environment* section if you choose to install these files.
   - **Borland C/C++ Development Files** includes the components necessary to develop applications using the Borland C/C++ compiler. Please read the *Modifying the Environment* section if you choose to install these files.
6. Select the name of the program folder that will contain icons for the NI-VXI software for VXIpc 800/700.

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

7. When the installation process completes, you must exit Windows and reboot your computer for the changes to take effect. You can let the Setup program reboot your computer by selecting the **Restart computer** option on the last screen. Click on **Finish** to end the installation.
8. If you backed up the manufacturer and model name files, you should restore them to the `TBL` subdirectory of your NI-VXI directory before running `VXIedit`.
9. You can now use `VXIedit` to configure the hardware in your VXI system. Continue with Chapter 3, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in `VXIedit`.

Modifying the Environment

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

Completing the Software Installation

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

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

After you `RESMAN`, you are ready to use the NI-VXI Resource Editor program `VXIedit` to interactively configure the hardware in your system. Continue with Chapter 3, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in `VXIedit`.

Using the DOS INSTALL Program

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

Running INSTALL

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

Caution: ***In its default configuration, the VXIpc 800/700 requests memory space for use by the NI-VXI driver above the 1 MB DOS boundary. To run your NI-VXI software for the VXIpc 800/700 in DOS you must reconfigure the VXIpc 800/700 to request memory below the 1 MB DOS boundary.***

To change the requested memory space you need to use the `VXIedit` program described in Chapter 3, *NI-VXI Configuration Utility*. Use the **VXIpc Configuration Editor** and access the **Bus Configuration Editor** menu. Change the user and driver windows to be below the 1 MB boundary by enabling the **Place below 1 MB** checkboxes. Please refer to the *User Window and Driver Window* section of Chapter 3, *NI-VXI Configuration Utility*, for more details.

Chapter 3

NI-VXI Configuration Utility

This chapter contains instructions for using the VXI Resource Editor utility of the NI-VXI software to configure the VXIpc 800/700 Series embedded computer and the VXI-MXI-2 or VME-MXI-2 mainframe extender.

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

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

- **VXIpc-860 Users**—The VXIpc-860 is designed for use with Windows 95/NT only.

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

Running the VXIedit Configuration Utility

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

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

Most of the features on the VXIpc 800/700 are configurable through software, using `VXIedit`, rather than through hardware switches or jumpers on the board itself. In addition, the `VXIedit` utility can override some of the hardware settings.

Figure 3-1 shows the main menu of the `VXIedit` resource editor.

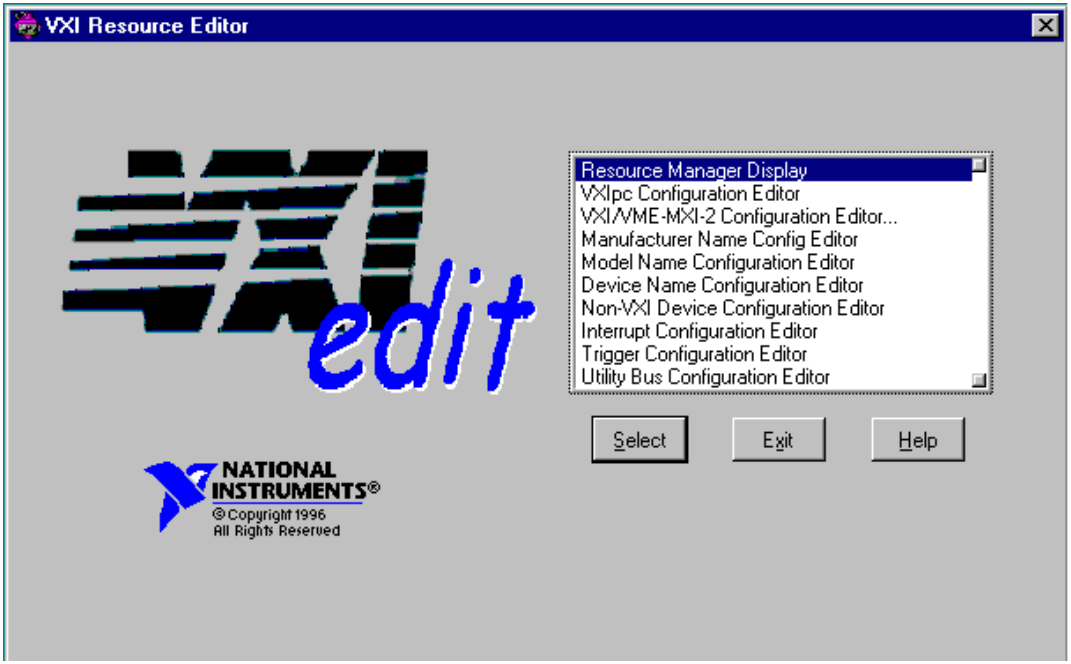


Figure 3-1. VXIedit Main Screen

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

VXIpc Configuration Editor

Figure 3-2 shows the opening screen of the **VXIpc Configuration Editor**. Notice that the screen displays the serial number and hardware revision of the VXIpc 800/700 board in addition to several configuration options.

The title of the screen will reflect the model of the controller that you have. For instance, if you have a VXIpc-850, the title will read **VXIpc-850 Configuration Editor**, as shown in the following screen**.**

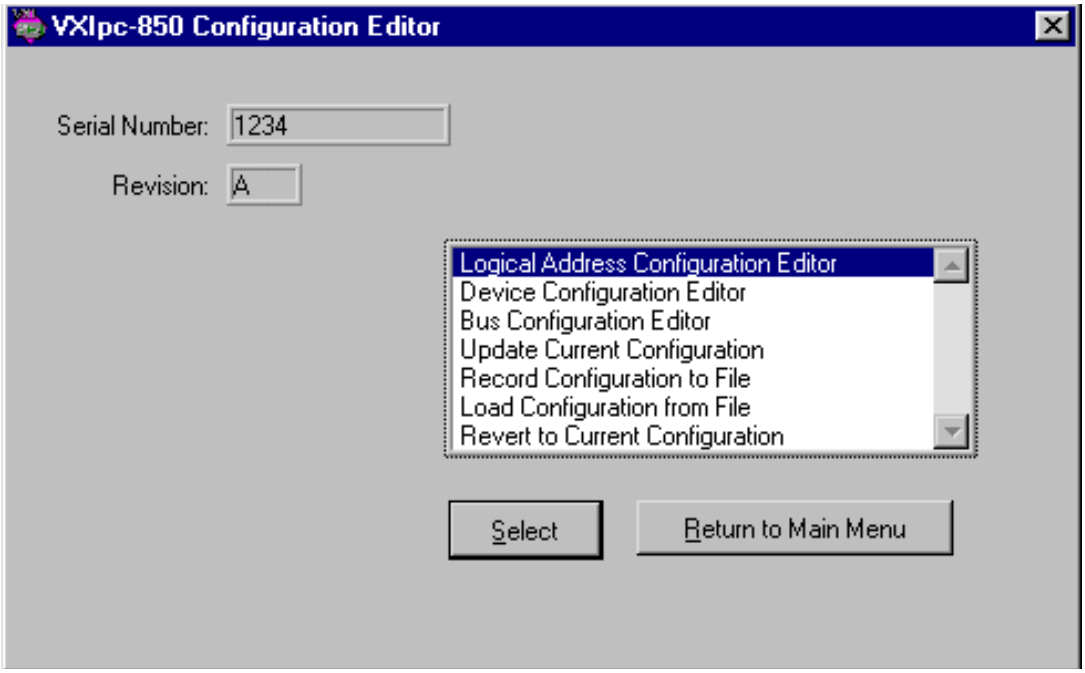


Figure 3-2. VXIpc Configuration Editor

The first three options under the **VXIpc Configuration Editor** are:

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

When making changes to the VXIpc 800/700 through these editors, remember that the changes do not take effect until you commit them by selecting the **Update Current Configuration** option.

- **DOS Users—**If the VXIpc 800/700 driver window is located above 1 MB, the program will prompt you for a memory location below 1 MB. The memory location is a 32 KB window that allows `VXIedit` to access the VXIpc 800/700 registers. This is due to a DOS limitation that restricts DOS programs from accessing memory above 1 MB.

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

Update Current Configuration

Use this option to write the configuration settings to the VXIpc 800/700 EEPROM and files used by NI-VXI. It configures the VXIpc 800/700 to be consistent with the configuration EEPROM. Notice that some of the configuration settings cannot take effect until you reset the machine, either by using the reset button or by turning the power off and on again.

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

Record Configuration to File

With this option you can save your configuration settings to a file. Notice that this option does *not* write the configuration settings to the VXIpc 800/700 configuration EEPROM.

If you want to update the VXIpc 800/700 configuration settings, use the **Update Current Configuration** option instead.

Load Configuration from File

You can use this option to load your configuration settings from a file. This action only updates the configuration settings in your editor. This does *not* write the configuration settings to the VXIpc 800/700 configuration EEPROM. To update the configuration use the **Update Current Configuration** option for the changes to take effect.

Revert to Current Configuration

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

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

Logical Address Configuration Editor

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

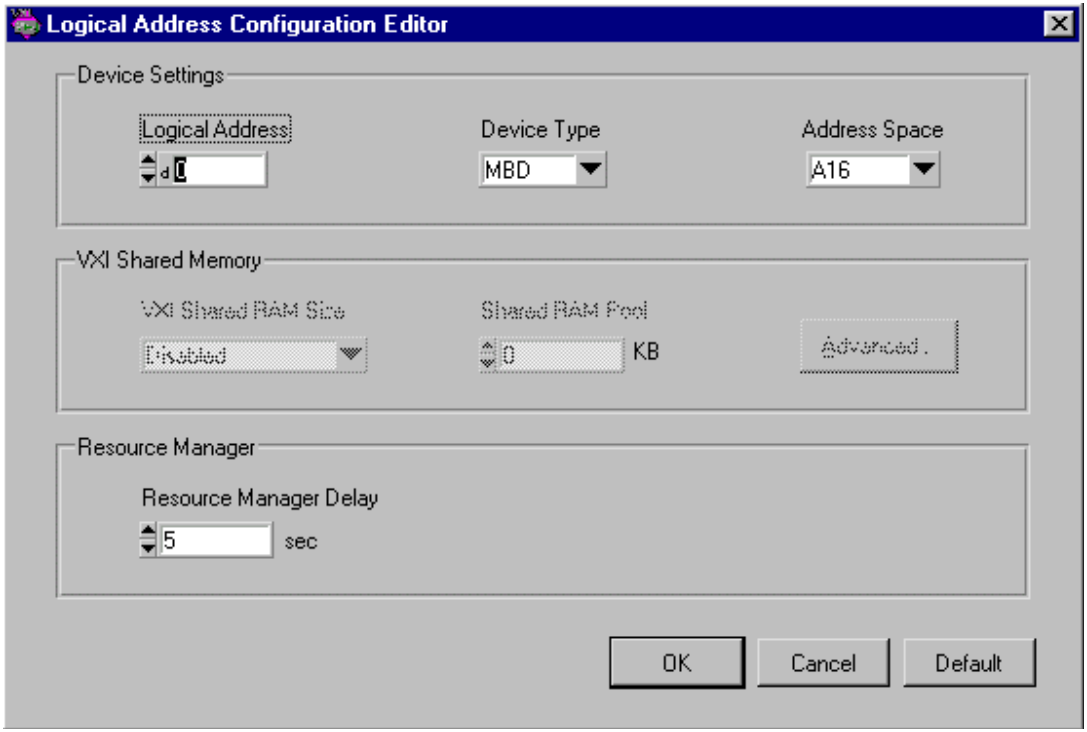


Figure 3-3. Logical Address Configuration Editor

Device Settings

The **Device Settings** group contains the controls to set the logical address, device type, and address space of the VXIpc 800/700.

Logical Address

This parameter sets the logical address of the VXIpc 800/700. The following table shows the allowable range of values and the default value.

Logical Address Range	Default Value
0 to 254	0

Device Type

This field indicates the classification of the VXIpc 800/700. The default value is **MBD**, designating a message-based device. The following table shows the available options.

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

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

Address Space

This field indicates the addressing mode(s) of the device's operational registers. The VXIpc 800/700 can be configured in one of three ways. The default addressing mode is for **A16** space only. Your other options are **A16/A24** and **A16/A32**.

Notice that the options under the **VXI Shared Memory** group are disabled when the **Address Space** control is set to **A16**, as shown in Figure 3-3. Only if you select **A16/A24** or **A16/A32** are the following controls relevant:

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

VXI Shared Memory

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

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

VXI Shared RAM Size

This field indicates the amount of RAM (in bytes) that are shared in either A24 or A32 space. This determines the *total* shared RAM size. Setting this field to **All System RAM** will detect how much memory you have installed in your VXIpc 800/700 and request the same amount of A24 or A32 space.

☞ **Note:** ***If you have more than 8 MB installed in your VXIpc 800/700, the* All System RAM *setting is allowed only if you use A32 memory space.***

Shared RAM Pool

This field indicates the size of memory in kilobytes that is allocated on Windows startup. This is physically contiguous memory that can be dual-ported on the VXIbus.

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

If you make a change to this setting, you must restart Windows 95/NT/3.1 to enable the change.

Memory Range	Default Value
0 to 65535 KB	0 KB

Caution: ***This memory pool is allocated at Windows 95/NT/3.1 startup, and Windows is denied access to this memory. Take into account the memory requirements of Windows and your applications and the amount of RAM in your system before setting this option.***

Advanced Shared RAM Settings

Click on the **Advanced** button to reach additional configuration options for the VXI shared RAM, as shown in Figure 3-4. These options are intended for more advanced users.

The VXI shared RAM is divided into two halves, or *windows*. You can select the byte order and mapping scheme for each half independently.

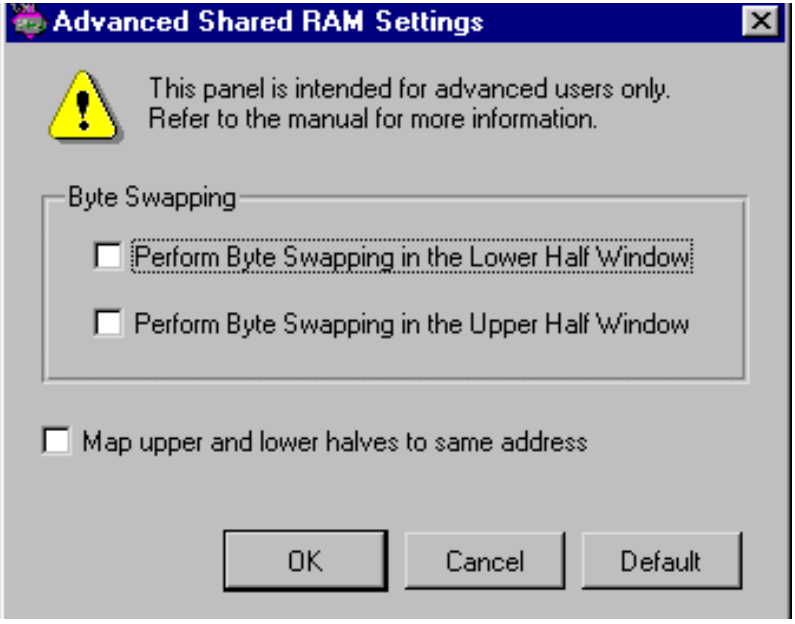


Figure 3-4. Advanced Shared RAM Settings

Upper/Lower Half Window Byte Swapping

This field indicates whether byte swapping should be performed for slave accesses to this half of the VXI shared RAM space. For example, if the native byte order of the shared RAM is Intel (Little Endian), and you want to present data to the VXIbus in Motorola (Big Endian) byte order, you will need to enable byte swapping by clicking on the checkbox for the appropriate window half. Byte swapping is disabled for both windows by default.

Upper/Lower Half Window Address Mapping

This field determines if the upper/lower half windows map to the same address or different addresses in system memory.

When this checkbox is checked, the buffer in system RAM can be dual-ported to the VXIbus in both Little Endian and Big Endian byte order. The setting of the **Byte Swapping** checkboxes for each half window determines whether the byte order is Little Endian or Big Endian. When this option is disabled, which is the default value, each half window maps to a unique local address on the VXIpc 800/700.

Resource Manager

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

Resource Manager Delay

☞ **Note:** ***This field is effective only when the VXIpc 800/700 is at its default logical address of 0. This logical address is required for the Resource Manager.***

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

RM Delay Range	Default Value
0 to 65535 s	5

Device Configuration Editor

Figure 3-5 shows the **Device Configuration Editor**. The following paragraphs describe the options you can select for each of the fields.

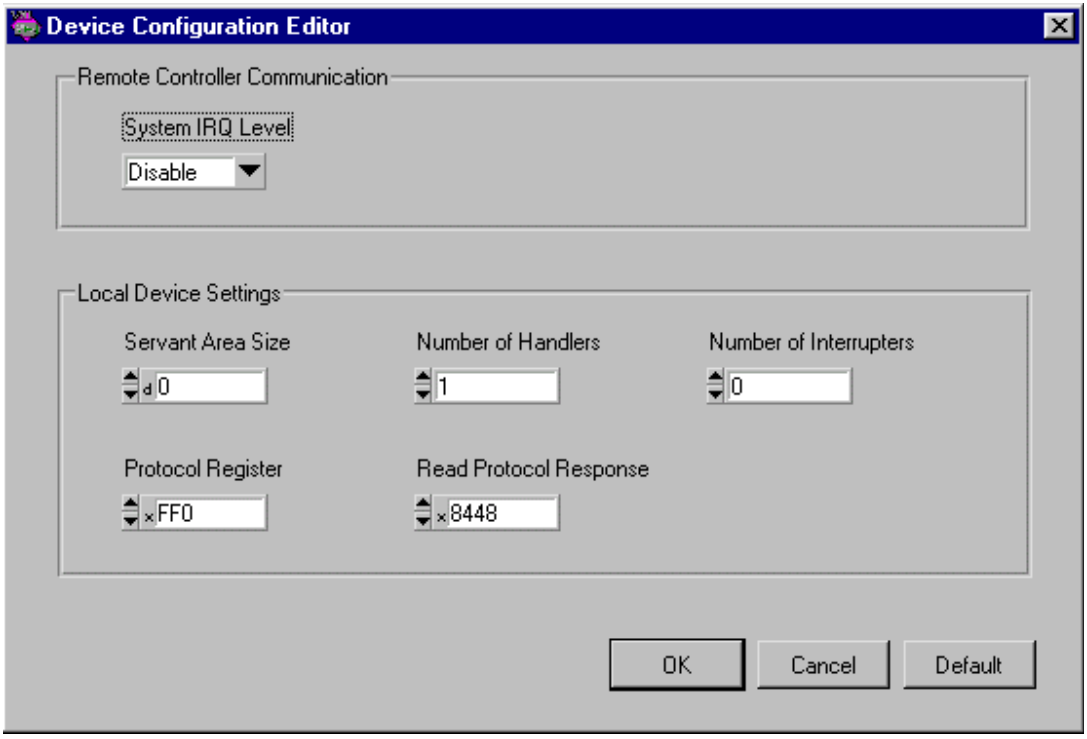


Figure 3-5. Device Configuration Editor

Remote Controller Communication

The only option under the **Remote Controller Communication** portion of the **Device Configuration Editor** is the **System IRQ Level** control.

System IRQ Level

Remote controllers can report events such as triggers and DMA to the VXIpc 800/700 through a VXI IRQ line. This field selects which VXI IRQ level the remote controllers should use to report such events.

Interrupt Request Levels	Default Value
1 to 7 or disabled	Disabled

☞ **Notes:** ***When the system IRQ line is disabled, the remote controller functionality is not available. If you are using a multi-mainframe system you should enable the system IRQ line.***

The VXI IRQ designated as system IRQ line cannot be disabled using the `DisableVXIint` ***or*** `DisableVXItoSignalInt` ***functions. The VXIpc 800/700 will always acknowledge it automatically when it is the Resource Manager.***

Local Device Settings

The **Local Device Settings** group contains the controls to set the servant area size, the number of interrupt handlers and interrupters that the VXIpc 800/700 Series supports, and the contents of the Protocol register.

Servant Area Size

This field designates the servant area size, which is supplied to the Resource Manager in response to the *Read Servant Area* command (if the VXIpc 800/700 is *not* the Resource Manager in your system). The servant area size is an 8-bit value (0 through 255) that indicates the servant area. The servant area begins at the logical address following the VXIpc 800/700 logical address, and includes *N* contiguous logical addresses, where *N* is the value of the servant area size. This field is meaningful only when the VXIpc 800/700 is configured as a message-based device.

Servant Area Range	Default Value
0 to 255	0

☞ **Note:** ***If the VXIpc 800/700 is the Resource Manager (Logical Address 0), this setting is irrelevant.***

Number of Handlers

This field gives the number of interrupt handlers that the VXIpc 800/700 supports.

Interrupt Handlers	Default Value
0 to 7	1

Number of Interrupters

This field gives the number of interrupters that the VXIpc 800/700 supports.

Interrupters	Default Value
0 to 7	0

Protocol Register

This field specifies the contents of the Protocol register, indicating which protocols the device supports. This field is meaningful only when the VXIpc 800/700 is configured as a message-based device. The default value is 0xFF0 (Commander, Signal Register, Master).

Read Protocol Response

This field specifies the response value to a *Read Protocol* command received by the VXIpc 800/700 from the Resource Manager (if the VXIpc 800/700 is *not* the Resource Manager in your system). This field is meaningful only when the VXIpc 800/700 is configured as a message-based device. The default value is 0x8448 (Response Generation, Event Generation, Programmable Handler, Word Serial Trigger, Instrument, Extended Longword Serial, Longword Serial).

Bus Configuration Editor

Figure 3-6 shows the **Bus Configuration Editor**. The following paragraphs describe the options you can select for each of the fields.

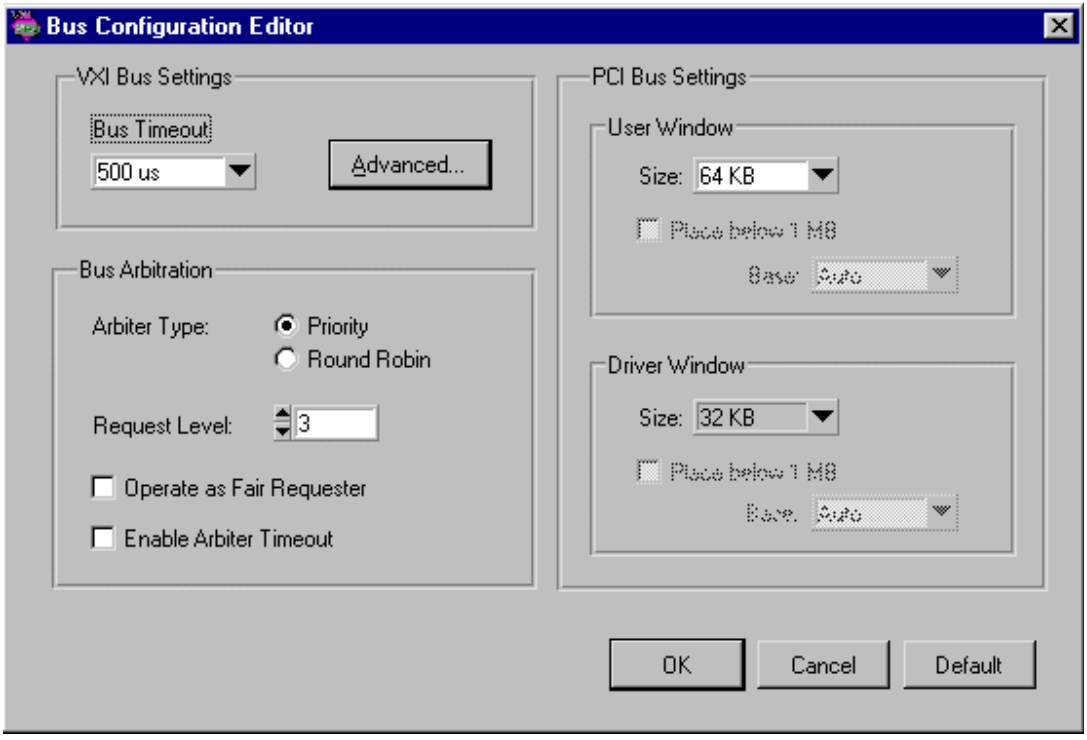


Figure 3-6. Bus Configuration Editor

VXI Bus Settings

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

Bus Timeout

The Bus Timeout (BTO) is a watchdog timer for transfers on the VXIbus. After the specified amount of time has elapsed, the BTO circuitry terminates a VXIbus cycle if no slave has responded. This control is applicable only if the VXIpc 800/700 you are configuring is a VXI Slot 0 device. You should disable the BTO of any other non-Slot 0 devices residing in the mainframe.

The lowest value in the allowable range is 15 µs and the highest is 256 ms. The default value is 500 µs.

Advanced VXI Bus Settings

Click on the **Advanced** button to reach additional configuration options for the VXI bus, as shown in Figure 3-7. These options are intended for more advanced users.

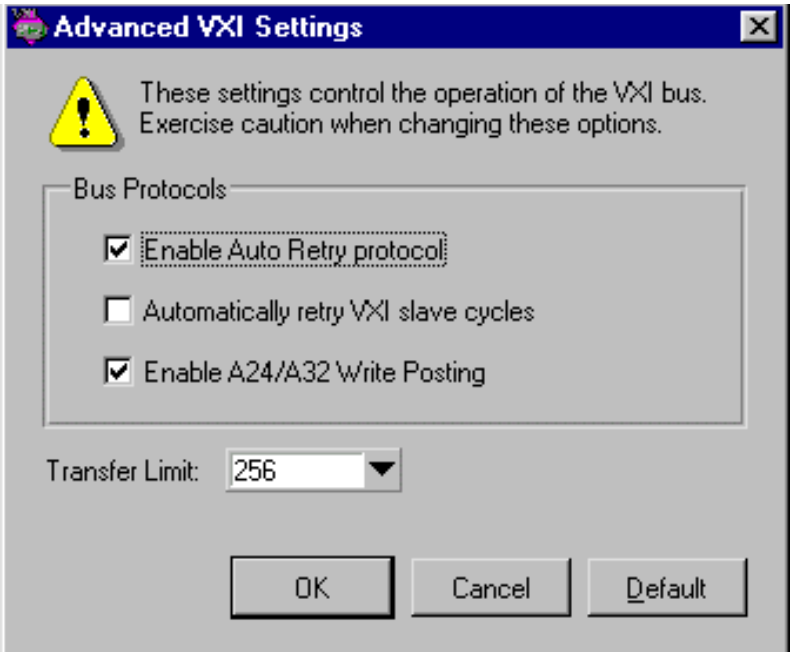


Figure 3-7. Advanced VXI Bus Settings

Automatic Retry Protocol

When the **Enable Auto Retry protocol** checkbox is checked, the VXIpc 800/700 can recognize and send the VXIbus retry protocol. If you disable this option, a retry is mapped to a bus error response. By default this option is enabled.

Automatic VXI Slave Cycle Retry

☞ **Note:** ***This option is not available in the VXIpc 700 Series.***

The VXIpc 800/700 Series has an automatic retry feature for cycles that map from the VXIbus to the PCI bus on the VXIpc 800/700. You can use the **Automatically retry VXI slave cycles** checkbox to enable or disable this option. By default this option is enabled on the VXIpc 800 Series and disabled on the VXIpc 700 Series.

Normally, when a cycle maps from the VXIbus to the PCI bus, any retry response received on the PCI bus is passed to the VXIbus. When this feature is enabled, the VXIpc-800 automatically retries any PCI cycle when the PCI host responds to a cycle with a retry. The VXIpc-800 automatically continues to retry the PCI cycle until it receives either a Disconnect or Target-Abort response, which it then passes to the VXIbus. This behavior is the default because many VXIbus masters do not support VXI retries. If the VXIbus master does support retries, it may be beneficial to disable this feature. With this feature disabled, you can lower the value of the **Bus Timeout** because there will be no delay due to the inward cycles being retried.

☞ **Note:** ***The VXIpc-800 has a limit on the number of automatic retries it will perform on any one cycle. If the limit is exceeded and the VXIpc-800 receives another retry, it will pass a retry or BERR (depending on whether the* Enable Auto Retry protocol *checkbox is checked) to the VXIbus even though the* Automatically retry VXI slave cycles *checkbox is checked.***

A24/A32 Write Posting

The VXIpc 800/700 can increase performance with its capability to post write cycles from the VXIbus. Write cycles should be posted only to addresses that cannot return a BERR signal, because the BERR will not be reported to the originating master. By default, this option is enabled.

The A24/A32 write posting control affects write cycles to the VXIpc 800/700 via its requested memory space from the VXIbus. When this option is enabled, the VXIpc 800/700 completes a VXIbus write cycle before writing the data from the cycle to the local destination on the VXIpc 800/700.

VXI Transfer Limit

You can use the **Transfer Limit** control to set how many data transfers the VXIpc 800/700 will perform on the VXIbus before releasing it to another master device that is requesting use of the bus.

The available options you can choose from are 16, 64, and 256 transfers. If you do not want the VXIpc 800/700 to hold the VXIbus long enough to perform 256 transfers (the default value), you can use this control to select a smaller value.

Bus Arbitration Settings

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

Arbiter Type

You can use the **Arbiter Type** feature to configure the VXIpc 800/700 as either a Priority or Round Robin VMEbus arbiter. This control is applicable only if the VXIpc 800/700 you are configuring is a VXI Slot 0 device. The default value is **Priority**.

When configured for Priority arbitration, the VXIpc 800/700 grants the bus to the highest pending bus request level. If you select **Round Robin** arbitration mode, the VXIpc 800/700 grants the bus to the next highest bus request level after the level of the previous bus owner. This effectively gives the same priority to each bus request level. Refer to the VMEbus specification for more information on the different types of arbiters.

Request Level

The VXIpc 800/700 uses one of the four VXIbus request levels (0 to 3) to request use of the VXI Data Transfer Bus (DTB). The VXIpc 800/700 requests use of the DTB whenever a local cycle maps into a VXIbus cycle.

The VXIpc 800/700 uses VXIbus request level 3 by default, as required by the VXIbus specification. This setting is suitable for most VXIbus systems. However, you can change the VXIpc 800/700 to use any of the other three request levels (0, 1, or 2) by changing the setting of the **Request Level** control. You may want to change request levels to change the priority of the VXIpc 800/700 request signal. For more information, refer to the VMEbus specification.

Fair Requester

The VXIpc 800/700 is always a Release On Request requester. However, you can configure whether the VXIpc 800/700 acts as either a fair or unfair requester on the VXIbus. By default the **Operate as Fair Requester** checkbox is cleared, signifying an unfair requester. For more information on the different types of requesters, refer to the VMEbus specification.

Arbiter Timeout

An arbitration timeout feature is available on the VXIpc 800/700 when it is acting as the VMEbus arbiter. This feature applies only to a VXI Slot 0 VXIpc 800/700. By default this option is disabled.

If you enable this feature, the timer begins when the arbiter circuit on the VXIpc 800/700 drives one of the BGOUT lines on the backplane. If no device takes over the bus within the timeout limit, the BGOUT is removed and the bus is either idle or granted to another requester.

PCI Bus Settings

Your VXIpc 800/700 uses the MITE custom ASIC to interface to the VXIbus. The MITE chip is located on the PCI bus on the VXIpc 800/700.

The following paragraphs describe the options for the **PCI Bus Settings** portion of the **Bus Configuration Editor**. You can use these options to modify settings relating to how much PCI address space the MITE chip will use and where this space will be located.

User Window and Driver Window

The VXIpc 800/700 driver requires the use of two PCI windows: a user window and a driver window. Calling the `MapVXIAddress()` function allocates regions of the user window to your application. `VXIpeek()` and `VXIpoke()` accesses are performed through this window. NI-VXI uses the driver window to perform high-level functions such as `VXIin()` and `VXIout()`, and to access MITE registers on the VXIpc 800/700.

The windows are mapped to PCI base address registers and determine the amount of PCI memory space the VXIpc 800/700 requests from the PCI system during initialization. You can set the window base, window size and whether the window resides above or below the 1 MB address space boundary.

- **DOS Users**—If you are intending to use DOS applications, note that DOS restricts programs from accessing memory above 1 MB. Verify that the **Place below 1 MB** checkboxes for both the user window and the driver window are enabled.

Window Size

The amount of space you can allocate for the user window is system dependent. By using the up/down arrow of the **Size** parameter, you can select the size of the user window (minimum of 4 KB, maximum of 2 GB). The more you increase the size of the user window, the larger the window you can map in `MapVXIAddress()`.

You can also disable this option. Disabling the user window causes the VXIpc 800/700 to request the minimum amount of address space on the PCI bus. With the window disabled, you will be unable to perform any low-level function calls such as `VXIpeek()`, `VXIpoke()`, and `MapVXIAddress()`. For example, on DOS systems, you may not be able to request more than the 32 KB driver window if the address space between 640 KB and 1 MB is being used by other devices in your computer.

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

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

Below 1 MB

This field determines whether the user or driver window is required to be located below the 1 MB address boundary (a DOS limitation). By default the user and driver windows are above the 1 MB boundary. Unless you are running in DOS, you can keep the windows above the 1 MB boundary, and it is recommended that you keep the default settings. Click on the **Place below 1 MB** checkbox if you need to relocate the windows for DOS applications.

Window Base

You can use the **Base** field to select a region of memory below 1 MB for the user or driver windows. Use this field to assign a new address for these windows. You should choose a region that does not conflict with any other devices you have installed in your system.

- **Windows 95/NT Users**—This option is not available under Windows 95/NT. However, under Windows 95 you can change the base of the driver/user window by using the **Resources** page in the **VXIpc 800/700 Properties** dialog in the Device Manager.

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

VXI/VME-MXI-2 Configuration Editor

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

- **DOS and Windows 3.1 Users**—You must run `VXIinit` before running `RESMAN` or any other VXI operations and after each computer reset.

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

Upon entering the **VXI/VME-MXI-2 Configuration Editor**, the program displays a list of VXI/VME-MXI-2 boards that `RESMAN` detected in your system, as shown in Figure 3-8.

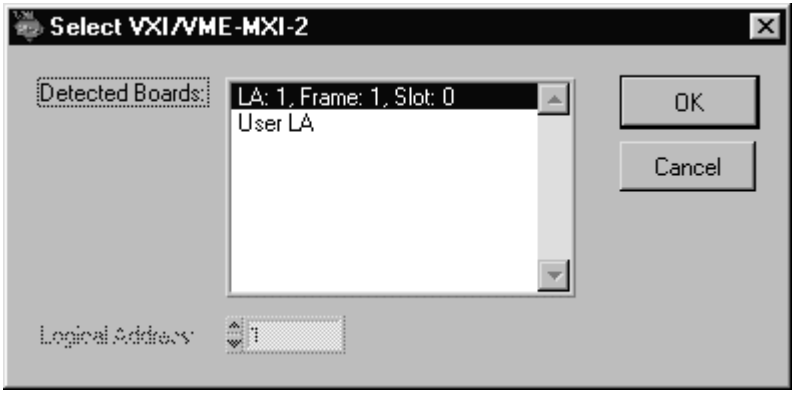


Figure 3-8. VXI/VME-MXI-2 Selection Dialog Box

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

After finding a VXI/VME-MXI-2, the **VXI/VME-MXI-2 Configuration Editor** displays a panel, as shown in Figure 3-9, that you can use to modify the configuration settings of that VXI/VME-MXI-2. The panel displays the current settings of the module. Notice that it also shows the hardware revision and serial number of the VXI/VME-MXI-2.

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

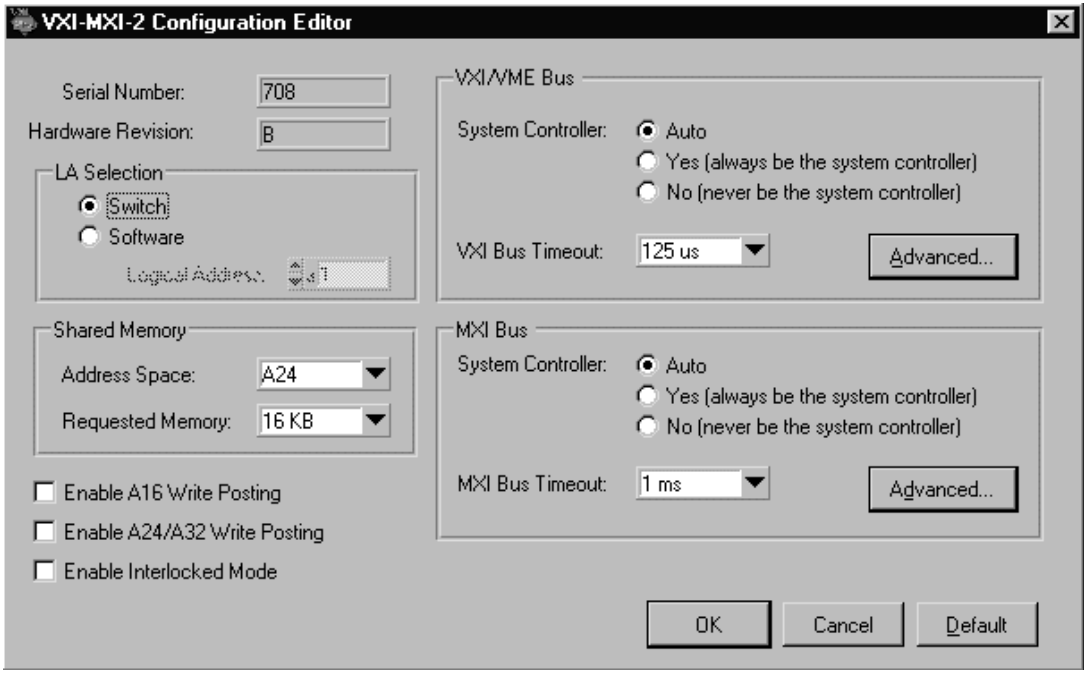


Figure 3-9. VXI/VME-MXI-2 Configuration Editor

LA Selection and Logical Address

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

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

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

Address Space and Requested Memory

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

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

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

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

The DRAM should then be accessed in the upper half of the requested memory space.

A16 Write Post and A24/A32 Write Posting

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

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

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

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

Interlocked Mode

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

In normal operating mode (non-interlocked), multiple masters can operate simultaneously in the VXI/VMEbus/MXIbus system. A deadlock occurs when a MXIbus master requests use of a VXI/VMEbus resource in another VXI/VMEbus mainframe while a VXI/VMEbus master in that mainframe is in the process of requesting a resource across the MXIbus. When this situation occurs, the VXI/VMEbus master must give up its bus ownership to resolve the conflict. The *RETRY* signal is used to terminate the transfer on the VMEbus; however, devices in the VXI/VMEbus mainframe must be able to detect

a *RETRY* caused by a deadlock condition so that they can retry the operation. Any master device that cannot detect the retry protocol will interpret the response as a *BERR* signal instead.

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

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

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

- A VXI/VMEbus mainframe with only slave devices and no masters. Without bus masters, there is no chance for deadlock. You can configure the VXI/VME-MXI-2 devices in this mainframe for normal operating mode.
- A VXI/VMEbus mainframe with both masters and slaves, but the masters communicate only with the slaves in their mainframe. The masters never attempt transfers across the MXIbus, so there is no chance for deadlock when a MXIbus master attempts a transfer into the VXI/VMEbus mainframe. You can configure the VXI/VME-MXI-2 devices in this mainframe for normal operating mode.
- A VXI/VMEbus mainframe in which all masters that perform cycles across the MXIbus support the VME64 RETRY protocol. You can configure the VXI/VME-MXI-2 devices in this mainframe for normal operating mode because all masters that could cause a deadlock will automatically retry the operation.

VXI/VME Bus Options

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

VMEbus System Controller

You can use the **System Controller** control to override the jumper setting on the VXI-MXI-2. The VME-MXI-2 does not have an onboard jumper setting for this option. When the **Auto** setting (the default setting) is active, the onboard jumper setting determines if the VXI-MXI-2 is the VXI Slot 0 device. For more information, refer to your hardware user manual or your MXI-2 getting started manual.

Otherwise, choose either the **Yes** or **No** option. Notice that selecting either of these options overrides the onboard jumper setting on the VXI-MXI-2, so it will not matter how the jumper is set. You would need to run the **VXI/VME-MXI-2 Configuration Editor** again if you decide to change the VMEbus System Controller (VXI Slot 0) setting at a later time.

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

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

VXI/VME Bus Timeout Value

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

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

Advanced VXI Settings

Click on the **Advanced** button to reach additional configuration options for the VXI/VME Bus portion of this editor, as shown in Figure 3-10. These options are intended for more advanced users.

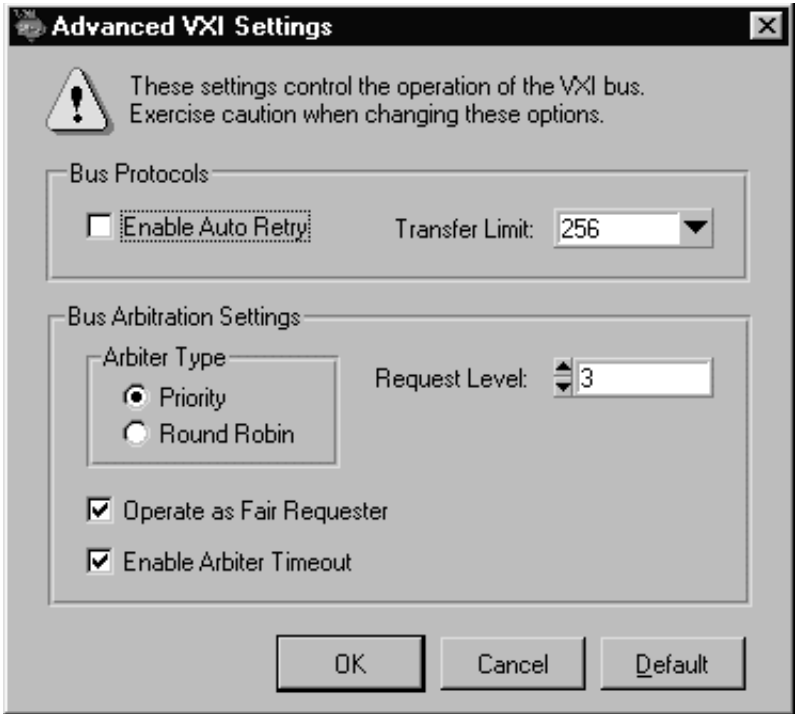


Figure 3-10. Advanced VXI Settings

VXI/VME Auto Retry

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

Normally, when a cycle maps from the VXI/VMEbus to the MXIbus, any retry response received on the MXIbus is passed to the VXI/VMEbus. If you enable the **Auto Retry** feature, the VXI/VME-MXI-2 automatically retries any MXI cycle that receives a retry response instead of passing a retry response back to the

VXI/VMEbus. The VXI/VME-MXI-2 automatically continues to retry the MXI cycle until it receives either a *DTACK* or *BERR* response, which it then passes to the VXI/VMEbus.

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

Transfer Limit

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

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

Arbiter Type

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

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

Request Level

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

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

VXI/VME Fair Requester

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

Arbiter Timeout

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

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

MXI Bus Options

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

MXI Bus System Controller

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

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

MXI Bus Timeout Value

The MXIbus Bus Timeout (BTO) is a watchdog timer for transfers on the MXIbus. The MXIbus BTO unit operates only when the VXI/VME-MXI-2 is acting as the MXIbus System Controller. The functionality of this control is similar to that of the **VXI Bus Timeout** control described previously under the *VXI/VME Bus Options* section. The options range from 8 µs to 128 ms, with a default value of 1 ms.

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

Advanced MXI Settings

Click on the **Advanced** button to reach additional configuration options for the MXI Bus portion of this editor, as shown in Figure 3-11. These options are intended for more advanced users.

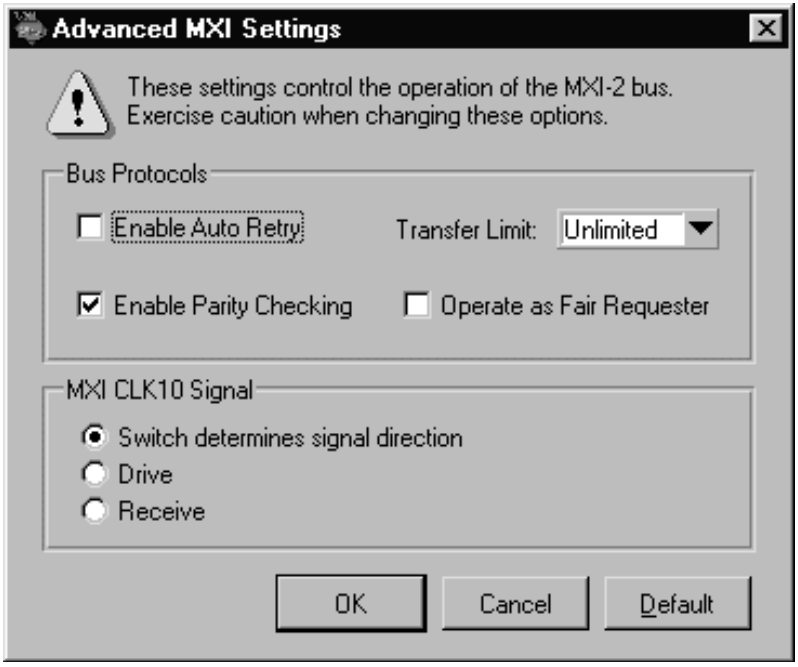


Figure 3-11. Advanced MXI Settings

MXI Auto Retry

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

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

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

Transfer Limit

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

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

Parity Checking

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

MXI Fair Requester

You can use the **Operate as Fair Requester** checkbox control to configure the VXI/VME-MXI-2 as either a fair or unfair requester on the MXIbus. In its default setting (disabled), the VXI/VME-MXI-2 will request the bus at any time. If you enable this option, the VXI/VME-MXI-2 will request the MXIbus only when there are no requests pending from other MXIbus masters. This prevents other MXIbus masters from being starved of bandwidth.

MXI CLK10 Signal

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

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

  You can use the **Drive** or **Receive** options to override the setting of S7 and control the direction of the MXIbus CLK10 signal. When receiving the MXIbus CLK10 signal, configure the W3 jumper setting to use the MXIbus as the source for generating the VXIbus CLK10 (applicable

only if the VXI-MXI-2 is a Slot 0 device). When driving the MXIbus CLK10, the VXIbus CLK10 is used as the source. In this case, change the jumper setting so that it does *not* use the MXIbus CLK10 as the source for the VXIbus CLK10.

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

Chapter 4

Using the NI-VXI Software

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

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

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

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

  Refer to the *NI-VXI User Manual* and the *NI-VXI Programmer Reference Manual* for an overview of NI-VXI and detailed descriptions of the NI-VXI functions. Windows 95/NT/3.1 users can also access online help from the NI-VXI folder.

Interactive Control of NI-VXI

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

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

- **DOS and Windows 3.1 Users**—If `NIVXI.DLL` is loaded in memory, do not attempt to execute `VIC` or any DOS program that uses the NI-VXI library in a DOS shell. Conflicts occur if both the DOS and Windows NI-VXI drivers are active at the same time, and may cause a system failure.

To guard against this conflict, the safest approach is to always exit Windows before attempting to execute any DOS program that uses the NI-VXI library, including `VIC`. You can execute `VIC` from a Windows DOS shell, however, if you ensure that no other Windows application that uses the `NIVXI.DLL` file is executing.

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

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

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

Example Programs

The `EXAMPLES` subdirectory contains various example programs along with a makefile that show how to use various functions in the NI-VXI software and how to develop application programs using these functions. Make certain that the environment variables `LIB` and `INCLUDE` are set correctly as described in Chapter 2, *NI-VXI Software Installation*. Also refer to the *NI-VXI Programmer Reference Manual* for additional examples.

Programming Considerations

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

Memory Model (DOS or Windows 3.1 Only)

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

Multiple Applications Using the NI-VXI Library

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

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

Low-Level Access Functions

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

The function `MapVXIAddress()` returns a pointer for use with low-level access functions. It is strongly recommended to use the `VXIpeek()` and `VXIpoke()` macros to access the memory instead of directly dereferencing the pointer. Using these macros makes the NI-VXI software more portable between platforms, because some platforms (such as the AT-MXI) require checking for retries, which can be handled through the macros. Directly dereferencing the pointers does not give you any speed benefit because the macros reduce to pointer dereferences at compile time for the VXIpc 800/700. Refer to the *Compiling Your C Program* section later in this chapter for more information on portability issues, and to your NI-VXI software reference manual for more information on low-level VXIbus access functions.

Setting User Handlers (DOS or Windows 3.1 Only)

You can set a user handler that will be invoked when certain conditions occur, such as VXI signals and triggers, by using functions such as `SetSignalHandler()` in the NI-VXI library. However, setting a new user handler replaces the existing handler, meaning that the existing

handler will no longer be invoked when the condition occurs. The following example illustrates the point.

```
SetSignalHandler(5, mySignalHandler1)

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

SetSignalHandler(5, mySignalHandler2)

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

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

Local Resource Access Functions

By using `VXIedit` or `VXItedit`, you can set up the VXIpc 800/700 to share the system memory on the VXIpc 800/700 motherboard with the VXI system. Refer to the *NI-VXI Graphical Utilities Reference Manual* or the *NI-VXI Text Utilities Reference Manual* for more information on setting these parameters.

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

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

Another factor to consider is that although you may have selected to share 8 MB, you must also inform Windows 95/NT/3.1 how much memory to set aside for possible `VXImemAlloc()` calls. You can use the **Shared Memory Pool** option in `VXIedit` and `VXItedit` for this purpose. But remember that the memory you put into this pool is no longer available to Windows. If this setting is too large, you may experience memory limitation problems when you run Windows.

Also remember that changes in the size of the pool do not take effect until the next time you start Windows.

System Configuration Functions

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

Compiling Your C Program

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

- **Windows 3.1 Users**—The sample Windows 3.1 program for the Microsoft C compiler is in the `\nivxi\win\msc\examples` directory, and the sample Windows program for the Borland C compiler is in the `\nivxi\win\borlandc\examples` directory.
- **Windows 95/NT Users**—The sample Windows 95/NT program for the Microsoft C compiler is in the `\nivxi\win32\msc\examples` directory, and the sample Windows 95/NT program for the Borland C compiler is in the `\nivxi\win32\borlandc\examples` directory.
- **DOS Users**—The sample DOS program for the Microsoft C compiler is in the `\nivxi\dos\msc\examples` directory, and the sample DOS program for the Borland C compiler is in the `\nivxi\dos\borlandc\examples` directory.

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

Symbols

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

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

- `VXIWIN` designates the application as a Windows 3.1 application.
- `VXINT` designates the application as a Windows 95/NT application.
- `VXIDOS` designates the application as a DOS application. You can use the same NI-VXI header files to compile DOS programs by defining `VXIDOS`.

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

The following symbol is optional.

- `BINARY_COMPATIBLE` makes the application binary compatible with external controllers, such as the National Instruments PCI-MXI-2 external controller. Using this option may cause a slight performance degradation when using low-level VXIbus access functions.

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

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

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

For the Microsoft C compiler:

```
/DVXIWIN /DBINARY_COMPATIBLE
```

For the Borland C compiler:

```
-DVXIWIN; BINARY_COMPATIBLE;
```

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

NI-VXI Software Overview

Appendix A

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

Main Programs and Files

This section lists the main programs and files of the NI-VXI software.

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

- `VXIINIT.EXE` is the VXIpc 800/700 initialization program. You can execute `VXIinit` from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell, although the latter is not recommended. This program initializes the VXIpc 800/700. It may be included in the DOS batch file `AUTOEXEC.BAT` or the Windows Startup folder so that the VXIpc 800/700 is automatically initialized at startup. The configuration settings can be modified using the `VXIEDIT.EXE` or `VXITEDIT.EXE` program.
- `RESMAN.EXE` is the National Instruments multiple-mainframe Resource Manager. You can execute `RESMAN` from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell, although the latter is not recommended. Under DOS and Windows 3.1, `RESMAN.EXE` may be executed only after `VXIINIT.EXE` has been run.
- `VIC.EXE` is an interactive control program that executes functions you enter from the keyboard. `VIC` helps you learn the functions, program your VXI devices, and develop and debug your application program. You can execute `VIC` from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell, although the latter is not recommended. If you do use `VIC` from the Windows 3.1 DOS shell, you must ensure that no other Windows application that uses NI-VXI functions is executing. This program is described in detail in the *NI-VXI Graphical Utilities Reference Manual.*

- VICTEXT.EXE is a text-based interactive control program that is functionally equivalent to VIC.EXE. You can execute VICtext from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell, although the latter is not recommended. If you run VICtext as a Windows application, you can use it at the same time that other Windows applications that use NI-VXI functions are executing. This program is described in detail in the *NI-VXI Text Utilities Reference Manual.*
- VXIEDIT.EXE is the VXI resource editor program. You use VXIedit to configure the system, and to edit various details such as the model names of VXI devices, the manufacturer name and ID numbers, and system interrupt configuration. This program is described in detail in the *NI-VXI Graphical Utilities Reference Manual.*
- VXITEDIT.EXE is the text-based VXI resource editor program that is functionally equivalent to VXIEDIT.EXE. You can execute VXItedit from DOS, Windows 95/NT, the Windows 95/NT DOS shell, or the Windows 3.1 DOS shell. This program is described in detail in the *NI-VXI Text Utilities Reference Manual.*
- README.TXT contains the latest updates and corrections to the manual when appropriate.

Header Files

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

- NIVXI.H is the main header file containing the C prototypes for the NI-VXI functions.
- DATASIZE.H contains data size specifications.
- BUSACC.H contains parameter and return values for the bus access functions.
- DEVINFO.H contains parameter and return values for the device information and system configuration functions.
- VXIINT.H contains parameter and return values for the interrupt and signal functions.
- SYSINT.H contains parameter and return values for the system interrupt functions.
- TRIG.H contains parameter and return values for the trigger functions.

- `WS.H` contains parameter and return values for the Commander and Servant Word Serial functions.
- `NIVXI.INC` is the include file for the Visual Basic for DOS language interface.

Common Questions

Appendix
B

This appendix addresses common questions you may have about using the NI-VXI bus interface software on the VXIpc 800/700 platform.

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

The 32-bit driver components all have version resources. Under Windows 95 and Windows NT 4.0, you can find out version information by right-clicking on any component and selecting the **Properties** option. This will display a property sheet with a version tab. This tab has version information about the product (NI-VXI) and the component (`NIVXINT.DLL`, for example). You can also find out version information if you run the NI-VXI utility program `VIC` or `VICtext`. At the prompt type `ver`, and the utility will display the versions of `VIC/VICtext` and NI-VXI, and the latest VXIpc 800/700 hardware revision that this NI-VXI driver supports.

How can I determine the revision of the VXIpc 800/700 that my NI-VXI software supports?

Running the NI-VXI utility program `VICtext` as described above will display the versions of `VICtext` and NI-VXI, and the hardware revision of the VXIpc 800/700 that the NI-VXI software supports.

How can I determine the serial number and hardware revision of the VXIpc 800/700?

Run the NI-VXI utility program `VXIedit`. Choose the **VXIpc Configuration Editor**. The opening screen displays the serial number and hardware revision of the VXIpc 800/700.

Which NI-VXI utility program must I use to configure the VXIpc 800/700?

Use the VXI Resource Editor program, either `VXIedit` or `VXItedit`, to configure the VXIpc 800/700.

Which NI-VXI utility program must I use to initialize the VXIpc 800/700?

Use the hardware initialization program, `VXIinit`, to initialize the VXIpc 800/700. `VXIinit` is required if you are running DOS or Windows 3.*x*. However, in Windows 95/NT, the board is automatically initialized at system startup, so you do not need to run `VXIinit` unless `RESMAN` fails.

What does `VXIinit` do?

The `VXIinit` program configures the components of the VXIpc 800/700 interface that are not configured by the EEPROM, such as resetting the DMA channels, manually placing the driver windows (if required), and, most importantly, setting up the MITE windows for direct access. In other words, it initializes the VXIpc 800/700. If `RESMAN` fails, it marks the controller as FAILED. `VXIinit` clears this condition so that you can run `RESMAN` again. `VXIinit` also reads the current configuration and displays that information to the user.

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

Use the `RESMAN` program to perform startup Resource Manager operations. `RESMAN` uses the settings in the Configuration Editor of `VXIedit` or `VXIt-edit`. It initializes your VXIbus system and stores the information that it collects to the `RESMAN.TBL` file in the `TBL` subdirectory of the `NIVXI` directory.

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

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

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

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

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

There are several reasons why you might get the **`Configuration EEPROM is Invalid`** message. If you turned off the computer while the configuration update process was still in progress, the VXIpc 800/700 will function normally except when using `VXIedit`. To correct this problem, switch to the factory configuration (switch S9 on the VXIpc 800 Series; jumper W7 on the VXIpc 700 Series), reboot the computer, and update the configuration, or load the configuration from file. For more details, refer to the *How to Fix an Invalid EEPROM Configuration* section in either Chapter 3, *VXIpc 800 Series Configuration and Installation*, or Chapter 4, *VXIpc 700 Series Configuration and Installation*, in the *VXIpc 800/700 Series User Manual*.

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

How do I fix an incorrect driver window base?

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

To correct the problem, switch to the factory configuration as described in the previous topic. If the problem disappears, use `VXIedit` to correct the driver and user window placement. If the problem persists, it may indicate that BIOS is not configuring the MITE VXI interface correctly.

Contact National Instruments for assistance. In Windows 95, you can also use the Device Manager to check for problems by using the **Resources** page in the **Properties** dialog.

How do I resolve a memory conflict?

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

If you are using Windows 3.*x*, you should place the MITE above the 1 MB boundary. To do so, switch to the factory configuration as described in the previous two topics. Run `VXIedit` and in the **Bus Configuration Editor** make sure that the **Place below 1 MB** option is *not* checked. Update the configuration and reboot, either by powering down your system or by using the reset button.

If you are using DOS, you can run `VXIedit` with the ***-o*** command line option to manually select the memory address used by the VXIpc 800/700 while `VXIedit` is running. `VXIedit` will prompt you to select a driver window base every time you try to enter the **VXIpc Configuration Editor**. You can try several different options until you find a region that does not cause a conflict.

If you have installed an adapter in the PCI or ISA expansion slot of the VXIpc-800, that device may conflict with the MITE VXI interface.

Often the memory conflict is not with another device, but rather with the memory management software such as EMM386, QEMM, or 386MAX. These products enable your computer to use parts of the upper memory, typically reserved for hardware adapters, to load device drivers and TSR. Because the VXIpc 800/700 needs that memory to operate, you must instruct your memory management software to exclude the regions used by the board. You can either omit the line that loads the memory manager or add a flag that prohibits the memory manager from using the requested range of upper memory. The most common memory manager is EMM386 because it ships standard with Microsoft DOS and Windows 3.*x*. In many cases, EMM386 is not needed for you configuration and can be omitted by adding `REM` to the

beginning of that line. If any of your applications do need it, you can add the following flag to the end of the line that loads EMM386:

```
x=start-end
```

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

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

What do the LEDs on the front of the VXIpc 800/700 mean?

The LEDs are fully described in Appendix D, *LED Indicators*, in the *VXIpc 800/700 Series User Manual*.

Is something wrong with the VXIpc 800/700 if the red SYSFAIL and FAILED LEDs stay lit after booting the VXIpc 800/700?

If either the **SYSFAIL** or **FAILED** LED remains lit, perform the following steps:

1. Power off the mainframe.
2. Remove all other modules from the mainframe.
3. Make sure that the VXIpc 800/700 jumper settings are set correctly.
4. Make sure that the VXIpc 800/700 is seated properly in the mainframe.
5. Power on the mainframe and observe whether the **SYSFAIL** and **FAILED** LEDs become unlit some time before the operating system boots.

Refer also to Appendix D, *LED Indicators*, in the *VXIpc 800/700 Series User Manual*.

Can I access 32-bit registers in my VXIbus system from the VXIpc 800/700?

Yes. The VXIpc 800/700 uses the 32-bit PCI bus to interface to the VXIbus. In fact, the VXIbus circuitry on the VXIpc 800/700 also supports the new VME64 standard for D64 accesses.

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

CLK10 is a differential ECL signal on the VXIbus backplane. However, the oscillator for the VXIpc 800/700 series and the EXTCLK input on the VXIpc 800 Series front panel use TTL levels. Therefore, you need to supply a TTL level signal for EXTCLK. Our voltage converters convert the signal to differential ECL. You cannot drive CLK10 externally on the VXIpc 700 Series.

What is the accuracy of the CLK10 signal?

The CLK10 generated by the VXIpc 800/700 is ±100 ppm (0.01%) as per the VXIbus specification. If you need a more accurate CLK10 signal on the VXIpc 800 Series, you can use the EXTCLK connector on its front panel.

If I boot the computer without video, and then plug in the video, why is it in black and white?

When the computer first boots, the video chips try to synchronize with the monitor. If the monitor is not there, the video chips cannot synchronize and establish color. You need to have the monitor attached at boot time to get color.

I've installed the SCSI software. Why doesn't the VXIpc 800 Series recognize my SCSI device?

This problem usually occurs when SCSI has not been enabled in the BIOS. Enter your BIOS by pressing <DEL> at boot time. In the **Setup** window, click on the **Chipset** icon. Select the **On Board PCI SCSI** option and enable it.

What type of video interface is present onboard the VXIpc 800/700? What video drivers are included with the VXIpc 800/700? Can I use Super VGA with my VXIpc 800/700? If my application requires a special type of video display, how do I configure my VXIpc 800/700?

The VXIpc 800/700 uses the Trident Microsystems TGUI96*xx* chip family, a combination graphics accelerator and RAMDAC. The TGUI9660 is the first chip used from this family. Succeeding pin-compatible chips will follow. The chips in this family are compatible with the Standard VGA video output MS Windows video driver, as well as the Trident Microsystems video driver. For information on the Trident driver, refer to the information in the `images\manual\video` directory for your operating system.

If your application requires a special type of video display, you can remove the PCI video card included with your VXIpc 800/700. Contact National Instruments for information on how to do this.

What kind of monitor can I use with the VXIpc 800/700?

VXIpc 800/700 computers that use Super VGA video output will work only with monitors having a horizontal scan rate of at least 50 kHz and a vertical scan rate of 60 Hz.

Warning: ***Make sure that your monitor meets this specification. Enabling the Super VGA option on a monitor that does not meet this specification will damage your monitor.***

What can I do if my keyboard connector does not fit into the keyboard port on the VXIpc 800/700?

You can plug keyboards that have a 6-pin Mini DIN PS/2 type connector directly into the VXIpc 800/700. You can use the keyboard adapter cable that is included with every VXIpc 800/700 kit to adapt the larger AT keyboard connector to the 6-pin Mini DIN connector.

How do I connect an external speaker to the VXIpc-800 to get audio capability?

A twisted-pair cable connects the front panel audio connector to the VXIpc 800 motherboard. Connect the external speaker to this front-panel connector. The center pin of the connector provides the audio signal. The shield of the connector is GROUND.

How do I add RAM to the VXIpc 800/700? What is the maximum amount of RAM that I can have on the VXIpc 800/700?

To add RAM to the VXIpc 800/700 series, remove the top cover and add SIMM modules to the empty SIMM sockets. The maximum amount the VXIpc-860 will support is 256 MB, or 64 MB in each socket. The maximum amount the VXIpc-850 will support is 128 MB, or 32 MB in each socket. The maximum amount for the VXIpc 700 Series is 64 MB, or 32 MB in each socket.

Which interrupt levels are free to be used by ISA bus boards? Which area of upper memory (adapter space) is free for use by ISA bus boards or expanded memory manager software programs?

See Appendix B, *VXIpc 800 Series System Resources*, or Appendix C, *VXIpc 700 Series System Resources,* in the *VXIpc 800/700 Series User Manual* for information on the available port I/O register space, upper memory area, interrupts, and DMA channels.

- The VXIpc 700 Series does not support an add-in ISA slot; however, you may still refer to the user manual for interrupt and address map information.

How do I install the VXIpc 800/700 in a slot other than Slot 0?

The VXIpc 800/700 automatically detects whether it is in Slot 0 of a VXIbus mainframe. You do not need to change jumper settings to install the VXIpc 800/700 in a slot other than Slot 0 unless you have defeated the first slot detector (FSD) circuitry by changing the appropriate jumper on the VXIpc 800/700.

Refer to the *VXIbus Slot 0/Non-Slot 0* section in either Chapter 3, *VXIpc 800 Series Configuration and Installation*, or Chapter 4, *VXIpc 700 Series Configuration and Installation*, in the *VXIpc 800/700 Series User Manual* for information on enabling and defeating the FSD circuitry.

How do I check the configuration of the memory, floppy drive, hard drive, time/date, and so on?

Follow these steps to view the BIOS setup parameters:

1. Reboot the VXIpc 800/700.
2. During the memory tests, press the <DEL> key.
3. After hitting the <DEL> key, you will get a graphical utility with icons that can be selected to view and change system parameters.

Can I upgrade my VXIpc 800/700?

You can upgrade the modular CPU card to upgrade the processor only on the VXIpc 800 Series. Contact National Instruments for information.

Customer Communication

Appendix C

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

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

Electronic Services

Bulletin Board Support

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

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

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

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

FTP Support

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

Fax-on-Demand Support

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

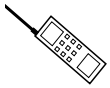
E-Mail Support (currently U.S. only)

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

support@natinst.com

Telephone and Fax Support

National Instruments has branch offices all over the world. Use the list below to find the technical support number for your country. If there is no National Instruments office in your country, contact the source from which you purchased your software to obtain support.

	 Telephone	 Fax
Australia	02 9874 4100	02 9874 4455
Austria	0662 45 79 90 0	0662 45 79 90 19
Belgium	02 757 00 20	02 757 03 11
Canada (Ontario)	905 785 0085	905 785 0086
Canada (Quebec)	514 694 8521	514 694 4399
Denmark	45 76 26 00	45 76 26 02
Finland	09 527 2321	09 502 2930
France	01 48 14 24 24	01 48 14 24 14
Germany	089 741 31 30	089 714 60 35
Hong Kong	2645 3186	2686 8505
Israel	03 5734815	03 5734816
Italy	02 413091	02 41309215
Japan	03 5472 2970	03 5472 2977
Korea	02 596 7456	02 596 7455
Mexico	5 520 2635	5 520 3282
Netherlands	0348 433466	0348 430673
Norway	32 84 84 00	32 84 86 00
Singapore	2265886	2265887
Spain	91 640 0085	91 640 0533
Sweden	08 730 49 70	08 730 43 70
Switzerland	056 200 51 51	056 200 51 55
Taiwan	02 377 1200	02 737 4644
U.K.	01635 523545	01635 523154

Technical Support Form

Photocopy this form and update it each time you make changes to your software or hardware, and use the completed copy of this form as a reference for your current configuration. Completing this form accurately before contacting National Instruments for technical support helps our applications engineers answer your questions more efficiently.

If you are using any National Instruments hardware or software products related to this problem, include the configuration forms from their user manuals. Include additional pages if necessary.

Name _____

Company _____

Address _____

_____

Fax (___)_____ Phone (___) _____

Computer brand _____ Model _____ Processor_____

Operating system (include version number) _____

Clock speed _____MHz RAM _____MB Display adapter _____

Mouse ___yes ___no Other adapters installed _____

Hard disk capacity _____MB Brand _____

Instruments used _____

_____

National Instruments hardware product model _____ Revision _____

Configuration _____

National Instruments software product _____ Version _____

Configuration _____

The problem is: _____

_____

_____

_____

_____

List any error messages: _____

_____

_____

The following steps reproduce the problem:_____

_____

_____

_____

_____

_____

Hardware and Software Configuration Form

Record the settings and revisions of your hardware and software on the line to the right of each item. Complete a new copy of this form each time you revise your software or hardware configuration, and use this form as a reference for your current configuration. Completing this form accurately before contacting National Instruments for technical support helps our applications engineers answer your questions more efficiently.

National Instruments Products

VXIpc 800 Series Hardware Settings

VXIpc 800 Series Model Number _____

Part Number _____

Serial Number _____

Hard Drive Size _____ Video Memory _____

Processor Speed_____

DRAM SIMMs Installed _____

Slot Location _____

W1 Setting: SCSI Termination _____

W2 Setting: CMOS _____

W4 Setting: LPT1 DMA _____

W13 Setting: Slot 0 Detection _____

S1 Setting: Ethernet EEPROM _____

S2 Setting: MITE Self-Configuration _____

S3 Setting: CLK10 Source _____

S4 Setting: Inverted/Non-inverted CLK10 Output _____

S5 Setting: CLK10 SMB _____

S6 Setting: CLK10 Input Termination_____

S7 Setting: External Trigger Input Termination _____

S8 Setting: GPIB IRQ Level _____

S9 Setting: MITE User/Factory Configuration _____

VXIpc 700 Series Hardware Settings

VXIpc 700 Series Model Number _____

Part Number _____

Serial Number _____

Hard Drive Size _____ Video Memory _____

Processor Speed _____

DRAM SIMMs Installed _____

Slot Location _____

W1 Setting: Slot 0 Detection _____

W3 Setting: CMOS _____

W6 Setting: Ethernet EEPROM _____

W7 Setting: MITE User/Factory Configuration _____

W10 Setting: MITE Self-Configuration _____

W12 Setting: External Trigger Input Termination _____

NI-VXI Software Settings

NI-VXI Software Version Number _____

Using VXIedit or VXItedit? _____

VXIpc Configuration Editor Settings (VXIedit)

Logical Address _____

Device Type _____

Address Space _____

VXI Shared RAM Size _____

Shared RAM Pool _____

Byte Swapping for Lower Half Window _____

Byte Swapping for Upper Half Window _____

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

_____

Resource Manager Delay _____

System IRQ Level _____

Servant Area Size _____

Number of Handlers _____

Number of Interrupters _____

Protocol Register _____

Read Protocol Response _____

VXI Bus Timeout _____

Auto Retry Protocol _____

Auto Retry VXI Slave Cycles _____

A24/A32 Write Posting _____

VXI Transfer Limit _____

Arbiter Type _____

Request Level _____

Fair Requester _____

Arbiter Timeout _____

User Window Size _____

User Window Below 1 MB _____

User Window Base _____

Driver Window Size _____

Driver Window Below 1 MB _____

Driver Window Base _____

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

Logical Address _____

LA Selection _____

Address Space _____

Requested Memory _____

A16 Write Posting _____

A24/A32 Write Posting _____

Interlocked or Normal Mode _____

VXI/VME System Controller _____

VXI/VME Bus Timeout Value _____

Auto Retry for Cycles from VXI/VMEbus to MXIbus _____

Transfer Limit on VXI/VMEbus _____

VXI/VME Arbiter Type _____

VXI/VME Request Level _____

VXI/VME Fair Requester _____

VXI/VME Arbiter Timeout _____

MXI System Controller _____

MXI Bus Timeout Value _____

Auto Retry for Cycles from MXIbus to VXI/VMEbus _____

Transfer Limit on MXIbus _____

MXI Parity Checking _____

MXI Fair Requester _____

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

Other Products

Mainframe Make and Model _____

Microprocessor _____

Clock Frequency _____

Type of Video Board Installed _____

Operating System _____

Operating System Version _____

Operating System Mode _____

Programming Language _____

Programming Language Version _____

Other Boards in System _____

Monitor (Manufacturer, Model) _____

Mouse (Manufacturer, Model) _____

Keyboard (Manufacturer, Model) _____

Other Peripherals (Manufacturer, Model) _____

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: *NI-VXI ™ Software Manual for the VXIpc™ 800/700 Series*

Edition Date: April 1997

Part Number: 321125E-01

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

_____

_____

_____

_____

_____

_____

_____

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

_____

_____

_____

_____

_____

_____

_____

Thank you for your help.

Name _____

Title _____

Company _____

Address _____

_____

Phone (___)_____ Fax (___) _____

Mail to: Technical Publications
National Instruments Corporation
6504 Bridge Point Parkway
Austin, TX 78730-5039

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

Glossary

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

A

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

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

A32 space — VXIbus address space equivalent to the VME 4 GB *extended* address space

address — Character code that identifies a specific location (or series of locations) in memory

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
ASIC	application-specific integrated circuit

B

B	bytes
backplane	An assembly, typically a printed circuit board, with 96-pin connectors and signal paths that bus the connector pins. A C-size VXIbus system will have two sets of bused connectors called J1 and J2. A D-size VXIbus system will have three sets of bused connectors called J1, J2, and J3.
BERR*	Bus error signal
BIOS	Basic Input/Output System. BIOS functions are the fundamental level of any PC or compatible computer. BIOS functions embody the basic operations needed for successful use of the computer's hardware resources.
BTO	See *Bus Timeout 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.
byte order	How bytes are arranged within a word or how words are arranged within a longword. Motorola ordering stores the most significant (MSB) byte or word first, followed by the least significant byte (LSB) or word. Intel ordering stores the LSB or word first, followed by the MSB or word.

C

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

D

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.
DMA	Direct Memory Access; a method by which data is transferred between devices and internal memory without intervention of the central processing unit
DRAM	Dynamic RAM (Random Access Memory); storage that the computer must refresh at frequent intervals
driver window	A region of address space that is decoded by the VXIpc 800/700 for use by the NI-VXI software
DTB	See *Data Transfer Bus*.

E

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

embedded controller	An intelligent CPU (controller) interface plugged directly into the VXI backplane, giving it direct access to the VXIbus. It must have all of its required VXI interface capabilities built in.

F

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

G

GPIB	General Purpose Interface Bus (IEEE 488)

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

IEEE	Institute of Electrical and Electronics Engineers
I/O	input/output; the techniques, media, and devices used to achieve communication between machines and users
interrupt	A means for a device to request service from another device
interrupt handler	A VMEbus functional module that detects interrupt requests generated by interrupters and responds to those requests by requesting status and identify information
interrupt level	The relative priority at which a device can interrupt
IRQ*	Interrupt signal

K

KB	kilobytes of memory

L

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

M

master	A functional part of a VME/VXIbus device that initiates data transfers on the backplane. A transfer can be either a read or a write.
MB	megabytes of memory
MBD	Message-Based Device
message-based device	An intelligent device that implements the defined VXIbus registers and communication protocols. These devices are able to use Word Serial Protocol to communicate with one another through communication registers.
MITE	A National Instruments custom ASIC, a sophisticated dual-channel DMA controller that incorporates the Synchronous MXI and VME64 protocols to achieve high-performance block transfer rates

N

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

P

PCI	Peripheral Component Interconnect. The PCI bus is a high-performance 32-bit or 64-bit bus with multiplexed address and data lines.
PCMCIA	Personal Computer Memory Card International Association
POSC	Power On Self Configuration

R

RBD	Register-Based Device
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
retry	An acknowledge by a destination that signifies that the cycle did not complete and should be repeated

S

s	seconds
Servant	A device controlled by a Commander; there are message-based and register-based Servants
Shared Memory Protocol	A communication protocol that uses a block of memory that is accessible to both a client and a server. The memory block operates as a message buffer for communications.
SIMM	Single In-line Memory Module

slave	A functional part of a 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
Slot 0 device	A device configured for installation in Slot 0 of a VXIbus mainframe. This device is unique in the VXIbus system in that it performs the VMEbus System Controller functions, including clock sourcing and arbitration for data transfers across the backplane. Installing such a device into any other slot can damage the device, the VXIbus backplane, or both.
SMB	Sub Miniature Type B connector that features a snap coupling for fast connection

T

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

U

user window	A region of address space reserved by the VXIpc 800/700 Series for use via the NI-VXI low-level function calls. `MapVXIAddress()` uses this address space to allocate regions for use by the `VXIpeek()` and `VXIpoke()` macros.

V

VIC or VICtext	VXI Interactive Control Program, a part of the NI-VXI bus interface software package. Used to program VXI devices, and develop and debug VXI application programs.
VME	Versa Module Eurocard or IEEE 1014
VMEbus System Controller	A device configured for installation in Slot 0 of a VXIbus mainframe or Slot 1 of a VMEbus chassis. This device is unique in the VMEbus system in that it performs the VMEbus System Controller functions, including clock sourcing and arbitration for data transfers across the backplane. Installing such a device into any other slot can damage the device, the VMEbus/VXIbus backplane, or both.

VXIbus	VMEbus Extensions for Instrumentation
VXIedit or VXItedit	VXI Resource Editor program, a part of the NI-VXI bus interface software package. Used to configure the system, edit the manufacturer name and ID numbers, edit the model names of VXI and non-VXI devices in the system, as well as the system interrupt configuration information, and display the system configuration information generated by the Resource Manager.
VXIinit	A program in the NI-VXI bus interface software package that initializes the board interrupts, shared RAM, VXI register configurations, and bus configurations

W

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