VXI

Getting Started with the VXIpc™ 770/870/870B Series and the NI-VXI™/NI-VISA™ Software for Linux

May 2002 Edition
Part Number 370550A-01

Worldwide Technical Support and Product Information

ni.com

National Instruments Corporate Headquarters

11500 North Mopac Expressway Austin, Texas 78759-3504 USA Tel: 512 683 0100

Worldwide Offices

Australia 03 9879 5166, Austria 0662 45 79 90 0, Belgium 02 757 00 20, Brazil 011 3262 3599,
Canada (Calgary) 403 274 9391, Canada (Montreal) 514 288 5722, Canada (Ottawa) 613 233 5949,
Canada (Québec) 514 694 8521, Canada (Toronto) 905 785 0085, China (Shanghai) 021 6555 7838,
China (ShenZhen) 0755 3904939, Czech Republic 02 2423 5774, Denmark 45 76 26 00, Finland 09 725 725 11,
France 01 48 14 24 24, Germany 089 741 31 30, Greece 30 1 42 96 427, Hong Kong 2645 3186,
India 91 80 4190000, Israel 03 6393737, Italy 02 413091, Japan 03 5472 2970, Korea 02 3451 3400,
Malaysia 603 9596711, Mexico 001 800 010 0793, Netherlands 0348 433466, New Zealand 09 914 0488,
Norway 32 27 73 00, Poland 0 22 3390 150, Portugal 351 210 311 210, Russia 095 238 7139,
Singapore 6 2265886, Slovenia 386 3 425 4200, South Africa 11 805 8197, Spain 91 640 0085,
Sweden 08 587 895 00, Switzerland 056 200 51 51, Taiwan 02 2528 7227, United Kingdom 01635 523545

For further support information, see the *Technical Support and Professional Services* appendix. To comment on the documentation, send email to techpubs@ni.com.

Important Information

Warranty

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

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

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

National Instruments believes that the information in this document 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

Patents

For patents covering National Instruments products, refer to the appropriate location: **Help»Patents** in your software, the `patents.txt` file on your CD, or `ni.com/patents`.

WARNING REGARDING USE OF NATIONAL INSTRUMENTS PRODUCTS

(1) NATIONAL INSTRUMENTS PRODUCTS ARE NOT DESIGNED WITH COMPONENTS AND TESTING FOR A LEVEL OF RELIABILITY SUITABLE FOR USE IN OR IN CONNECTION WITH SURGICAL IMPLANTS OR AS CRITICAL COMPONENTS IN ANY LIFE SUPPORT SYSTEMS WHOSE FAILURE TO PERFORM CAN REASONABLY BE EXPECTED TO CAUSE SIGNIFICANT INJURY TO A HUMAN.

(2) IN ANY APPLICATION, INCLUDING THE ABOVE, RELIABILITY OF OPERATION OF THE SOFTWARE PRODUCTS CAN BE IMPAIRED BY ADVERSE FACTORS, INCLUDING BUT NOT LIMITED TO FLUCTUATIONS IN ELECTRICAL POWER SUPPLY, COMPUTER HARDWARE MALFUNCTIONS, COMPUTER OPERATING SYSTEM SOFTWARE FITNESS, FITNESS OF COMPILERS AND DEVELOPMENT SOFTWARE USED TO DEVELOP AN APPLICATION, INSTALLATION ERRORS, SOFTWARE AND HARDWARE COMPATIBILITY PROBLEMS, MALFUNCTIONS OR FAILURES OF ELECTRONIC MONITORING OR CONTROL DEVICES, TRANSIENT FAILURES OF ELECTRONIC SYSTEMS (HARDWARE AND/OR SOFTWARE), UNANTICIPATED USES OR MISUSES, OR ERRORS ON THE PART OF THE USER OR APPLICATIONS DESIGNER (ADVERSE FACTORS SUCH AS THESE ARE HEREAFTER COLLECTIVELY TERMED "SYSTEM FAILURES"). ANY APPLICATION WHERE A SYSTEM FAILURE WOULD CREATE A RISK OF HARM TO PROPERTY OR PERSONS (INCLUDING THE RISK OF BODILY INJURY AND DEATH) SHOULD NOT BE RELIANT SOLELY UPON ONE FORM OF ELECTRONIC SYSTEM DUE TO THE RISK OF SYSTEM FAILURE. TO AVOID DAMAGE, INJURY, OR DEATH, THE USER OR APPLICATION DESIGNER MUST TAKE REASONABLY PRUDENT STEPS TO PROTECT AGAINST SYSTEM FAILURES, INCLUDING BUT NOT LIMITED TO BACK-UP OR SHUT DOWN MECHANISMS. BECAUSE EACH END-USER SYSTEM IS CUSTOMIZED AND DIFFERS FROM NATIONAL INSTRUMENTS' TESTING PLATFORMS AND BECAUSE A USER OR APPLICATION DESIGNER MAY USE NATIONAL INSTRUMENTS PRODUCTS IN COMBINATION WITH OTHER PRODUCTS IN A MANNER NOT EVALUATED OR CONTEMPLATED BY NATIONAL INSTRUMENTS, THE USER OR APPLICATION DESIGNER IS ULTIMATELY RESPONSIBLE FOR VERIFYING AND VALIDATING THE SUITABILITY OF NATIONAL INSTRUMENTS PRODUCTS WHENEVER NATIONAL INSTRUMENTS PRODUCTS ARE INCORPORATED IN A SYSTEM OR APPLICATION, INCLUDING, WITHOUT LIMITATION, THE APPROPRIATE DESIGN, PROCESS AND SAFETY LEVEL OF SUCH SYSTEM OR APPLICATION.

Compliance

FCC/Canada Radio Frequency Interference Compliance*

Determining FCC Class

The Federal Communications Commission (FCC) has rules to protect wireless communications from interference. The FCC places digital electronics into two classes. These classes are known as Class A (for use in industrial-commercial locations only) or Class B (for use in residential or commercial locations). Depending on where it is operated, this product could be subject to restrictions in the FCC rules. (In Canada, the Department of Communications (DOC), of Industry Canada, regulates wireless interference in much the same way.)

Digital electronics emit weak signals during normal operation that can affect radio, television, or other wireless products. By examining the product you purchased, you can determine the FCC Class and therefore which of the two FCC/DOC Warnings apply in the following sections. (Some products may not be labeled at all for FCC; if so, the reader should then assume these are Class A devices.)

FCC Class A products only display a simple warning statement of one paragraph in length regarding interference and undesired operation. Most of our products are FCC Class A. The FCC rules have restrictions regarding the locations where FCC Class A products can be operated.

FCC Class B products display either a FCC ID code, starting with the letters **EXN**, or the FCC Class B compliance mark that appears as shown here on the right.

Consult the FCC Web site at `http://www.fcc.gov` for more information.

FCC/DOC Warnings

This equipment generates and uses radio frequency energy and, if not installed and used in strict accordance with the instructions in this manual and the CE Mark Declaration of Conformity**, may cause interference to radio and television reception. Classification requirements are the same for the Federal Communications Commission (FCC) and the Canadian Department of Communications (DOC).

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

Class A

Federal Communications Commission

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

Canadian Department of Communications

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

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

Class B

Federal Communications Commission

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Canadian Department of Communications

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

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

Compliance to EU Directives

Readers in the European Union (EU) must refer to the Manufacturer's Declaration of Conformity (DoC) for information** pertaining to the CE Mark compliance scheme. The Manufacturer includes a DoC for most every hardware product except for those bought for OEMs, if also available from an original manufacturer that also markets in the EU, or where compliance is not required as for electrically benign apparatus or cables.

To obtain the DoC for this product, click **Declaration of Conformity** at `ni.com/hardref.nsf/`. This Web site lists the DoCs by product family. Select the appropriate product family, followed by your product, and a link to the DoC appears in Adobe Acrobat format. Click the Acrobat icon to download or read the DoC.

* Certain exemptions may apply in the USA, see FCC Rules §15.103 **Exempted devices**, and §15.105(c). Also available in sections of CFR 47.

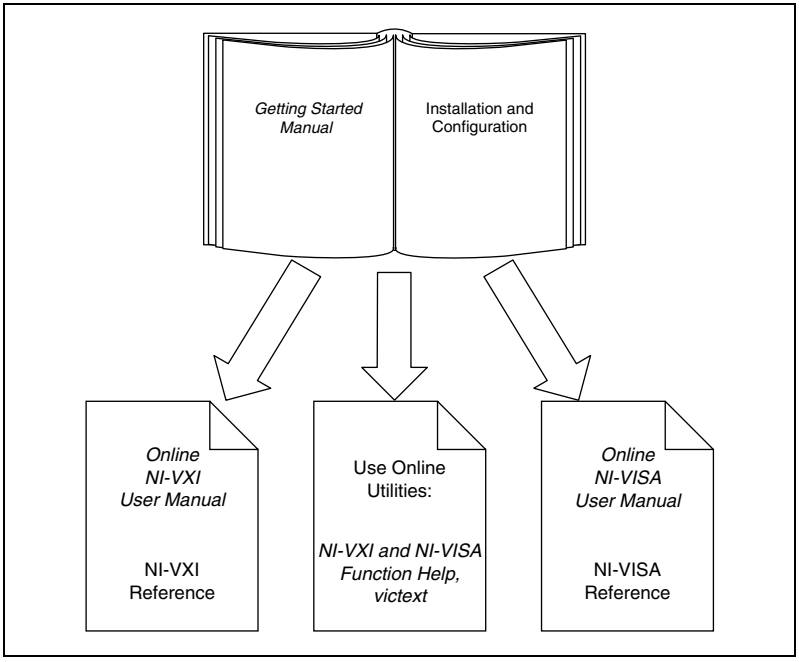
** The CE Mark Declaration of Conformity will contain important supplementary information and instructions for the user or installer.

Contents

About This Manual

Use this manual to get started with the VXIpc embedded computers and the NI-VXI/NI-VISA software for Linux. This manual summarizes the setup instructions and default settings for the hardware and software.

How To Use the Documentation Set



Begin by reading this manual for basic instructions on setting up the hardware and software. This manual describes how to get started with your kit using the default hardware and software settings and describes optional settings you can configure using the NI-VXI/NI-VISA software.

You received a VXIpc user manual with your kit. The user manual contains more details about changing the hardware installation or configuration from the defaults and using the hardware.

When you are familiar with the material in the previous manuals, you can begin to use the *NI-VXI User Manual* or, for VISA users, the *NI-VISA User Manual.* These manuals present the concepts of VXI and describe how to use NI-VXI and NI-VISA. The NI-VXI online help, the NI-VISA online help, the *NI-VXI Programmer Reference Manual* and the *NI-VISA Programmer Reference Manual* contain detailed explanations of NI-VXI and NI-VISA functions. Study the descriptions of each function to fully understand the purpose and syntax. Use the Acrobat Reader program, version 3 or later, to open, view, and navigate through these manuals online.

We recommend the VISA API for new applications. Refer to the *NI-VISA User Manual* to learn about VISA and how to use it in your system. The NI-VISA online help describes the attributes, events, and operations you can use in NI-VISA. The user manual is available in the `VXIpnp/linux/NIvisa/manuals` directory (where *`VXIpnp`* refers to the actual location where you have installed the NI-VISA software). Use the Acrobat Reader program, version 3 or later, to open this file.

Conventions

The following conventions appear in this manual:

♦ — The ♦ symbol indicates that the following text applies only to a specific product, a specific operating system, or a specific software version.

 — This icon denotes a note, which alerts you to important information.

 — This icon denotes a caution, which advises you of precautions to take to avoid injury, data loss, or a system crash.

bold — Bold text denotes items that you must select or click on in the software, such as menu items and dialog box options. Bold text also denotes parameter names.

italic — Italic text denotes variables, emphasis, a cross reference, an introduction to a key concept, or Word Serial commands and queries. This font also denotes text that is a placeholder for a word or value that you must supply.

`monospace` — Text in this font denotes text or characters that you should enter 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, operations, variables, filenames and extensions, and code excerpts.

monospace bold	Bold text in this font denotes the messages and responses that the computer automatically prints to the screen. This font also emphasizes lines of code that are different from the other examples.
monospace italic	Italic text in this font denotes text that is a placeholder for a word or value that you must supply.

Related Documentation

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

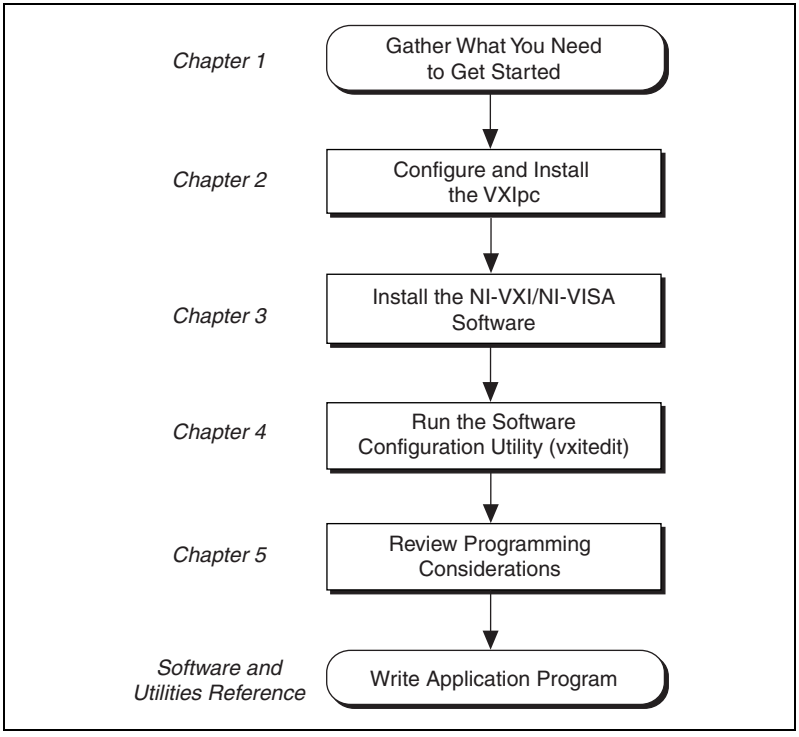
- ANSI/IEEE Standard 1014-1987, *IEEE Standard for a Versatile Backplane Bus: VMEbus*
- ANSI/IEEE Standard 1155-1993, *IEEE VMEbus Extensions for Instrumentation: VXIbus*
- ANSI/VITA 1-1994, *VME64*
- VXI-6, *VXIbus Mainframe Extender Specification*, Rev. 1.0, VXIbus Consortium

1 Introduction

This chapter describes the VXIpc embedded VXI computers and the NI-VXI/NI-VISA software, lists what you need to get started, and gives an overview of the directory structure on your hard drive.

How to Use This Manual

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



What You Need to Get Started

- ❑ VXIpc 770/870/870B Series embedded controller (hereafter described together as the VXIpc controller)
- ❑ VXIbus mainframe
- ❑ Keyboard and included adapter cable
- ❑ Monitor with VGA connector
- ❑ National Instruments software media for the VXIpc embedded controller
- ❑ Support for modules, included with all major Linux distributions
- ❑ Approximately 3 MB of hard disk space
- ❑ Linux kernel 2.2.*x* or 2.4.*x*

You meet the Linux kernel requirements if you are using one of the following distributions, among others:

- RedHat Linux 5.0 or later
- SuSE Linux 6.0 or later
- Caldera OpenLinux 2.2 or later
- Debian Linux 2.0 or later
- Slackware 4.0 or later

Hardware Description

The VXIpc controllers are C-size PCI-based embedded computers. These computers are high-performance, easy-to-use platforms for controlling VXIbus systems, featuring complete VXI functionality through interactive utilities and C function calls. These embedded computers can take advantage of the VXI high-performance backplane capabilities and give you direct control of VXI registers, memory, interrupts, and triggers.

For in-depth details on the VXIpc 770/870/870B Series hardware, including a description of the differences between the various models in their respective series, refer to the *VXIpc 870 Series User Manual* or the *VXIpc 770/870B Series User Manual*.

Software Description

The NI-VXI bus interface software for the VXIpc embedded controller includes a VXI Resource Manager, an interactive configuration program, libraries of software routines for programming, and an interactive VXIbus control program. You can use this software to seamlessly program multiple-mainframe configurations and ensure software compatibility across a variety of controller platforms.

If you decide to change the NI-VXI software configuration from its default settings, refer to Chapter 4, *NI-VXI Configuration Utility*. This chapter describes each field in the VXIpc Configuration Editor and the VXI/VME-MXI-2 Configuration Editor of the vxiedit software utility. Refer to the *NI-VXI Graphical Utilities Reference Manual* for more information about `vic` and the other configuration editors in `vxiedit`. Refer also to the *NI-VXI User Manual*, the *NI-VXI Programmer Reference Manual*, and the NI-VXI online help for thorough details about NI-VXI and the groups of NI-VXI function calls.

NI-VISA is a standard I/O Application Programming Interface (API) for instrumentation programming.

NI-VISA can control VXI/VME, PXI, GPIB, or Serial instruments, making the appropriate driver calls depending on the type of instrument being used. NI-VISA uses the same operations to communicate with instruments regardless of the interface type. For example, the NI-VISA command to write an ASCII string to a message-based instrument is the same whether the instrument is Serial, GPIB, or VXI. As a result, NI-VISA gives you interface independence. This makes it easier to switch bus interfaces and means that users who must program instruments for multiple interfaces need learn only one API.

Another advantage of NI-VISA is that it is an object-oriented API that will easily adapt to new instrumentation interfaces as they evolve, making application migration to the new interfaces easy.

VISA is the industry standard for developing instrument drivers. Most current drivers written by National Instruments use NI-VISA and support Windows, Solaris 2, HP-UX, VxWorks, Linux, and Macintosh, as long as the appropriate *system*-level drivers are available for that platform.

Optional Software

Your VXIpc kit includes the NI-VXI/NI-VISA bus interface software. In addition, you can use National Instruments LabVIEW to ease your programming task. This standardized program matches the modular virtual instrument capability of VXI and can reduce your VXI/VMEbus software development time.

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

2

Configuration and Installation

This chapter contains basic instructions for setting up the VXIpc embedded controller and the NI-VXI/NI-VISA software.

You can use this material as a guide to quickly configure and operate your VXI system using the VXIpc controller. This chapter assumes you intend to perform a basic configuration as follows:

- You have one VXIbus chassis in which you will use the VXIpc embedded controller as the Resource Manager (logical address 0).
- You will use the NI-VXI software for initialization, configuration, and device interaction.
- You will use the default hardware and software settings.

Step 1. Configure the Hardware

The default hardware settings are acceptable for most typical applications. Refer to Appendix A, *Default Settings*, for a complete listing of the hardware and software default settings.

Your VXIpc user manual fully describes the configuration and installation of your embedded controller. Refer to your VXIpc user manual if you want to try a different hardware configuration, or if you would like more information on a particular setting.

Step 2. Install the Hardware

1. To prevent electrostatic discharge, touch the antistatic plastic package to a metal part of your VXIbus chassis before removing the VXIpc module from the package.

Caution To protect both yourself and the mainframe from electrical hazards, leave the mainframe off until you finish installing the VXIpc module.

2. Plug in your chassis, but leave the power turned off.

3. Install the VXIpc controller in the first slot (Slot 0) of the VXI chassis. In its default configuration, the VXIpc automatically detects whether it should be the VXIbus system controller. The VXIbus system controllers operate certain VXIbus lines as required for VXI systems. Verify that no other VXI devices with system controller capability that are located in the same chassis are configured as system controller.

Caution Having more than one device configured as system controller will damage the VXI system.

4. For VXI systems that include VME devices, ensure that the VME devices are not configured in the upper 16 KB (starting from 0xC000) of the A16 address space. This region is reserved for VXI device configuration registers, which are used for initializing, configuring, and interacting with VXI devices.
5. Also ensure that no VXI devices in your system are configured for logical address 0, which is the default configuration for the VXIpc controller.
6. To complete your installation, attach cables for any devices you want to connect to your system. Refer to your VXIpc user manual if you are uncertain about any of these connections.
7. Turn on power to the VXI chassis.

Step 3. Install the Operating System

You must install a version of Linux consistent with the requirements specified in the *What You Need to Get Started* section of Chapter 1, *Introduction*.

Note On the VXIpc-770, you can use a USB CD-ROM drive to install Linux or a USB floppy drive to initiate a network-based installation.

3

NI-VXI/NI-VISA Software Installation

This chapter describes how to install and uninstall the NI-VXI/NI-VISA software for Linux.

Installing the NI-VXI/NI-VISA Software for Linux

Before you begin, you may need to install Linux on your VXIpc. Refer to the Linux documentation for instructions. After your computer is booted into Linux, you are ready to install the NI-VXI/NI-VISA software.

To install NI-VXI/NI-VISA for the VXIpc for Linux, perform the following steps:

1. Insert the *NI-VXI/NI-VISA for Linux* CD.
2. Login to your system as `root`.
3. Mount the CD-ROM.
4. To change the current directory to the mounted CD-ROM, type the following command:

   ```
   cd /mnt/cdrom
   ```

5. To run the installation script, type the following command:

   ```
   ./INSTALL
   ```

The `INSTALL` script places NI-VXI and NI-VISA in their default locations. The script uses `rpm` to install the packages on systems that support it or extracts the files directly on other systems. The script also optionally installs support for NI-VXI in LabVIEW.

You also can install the RPM files without going through the `INSTALL` script by using `rpm`, `glint`, or `gnorpm` on RedHat or other RPM-based systems. For example, to install NI-VXI in `/opt` on a RedHat 5.*x* system, type the following command:

```
rpm --prefix=/opt/nivxi -Uvh nivxi-vxipc-1.6-1.i386.rpm
```

where 1.6-1 is the version you are installing. Note that this version number will be different if you are installing a later version.

Note If you use rpm rather than the INSTALL script, you must repeat this step for each package you want to install.

If you install the software to a location other than the default, set the appropriate environment variable: NIVXIPATH for NI-VXI or VXIPNPPATH for NI-VISA. Refer to the *NI-VXI/NI-VISA Software Location* section for details.

Refer to the README file on the CD-ROM for additional important information and instructions.

Removing the NI-VXI Driver for Linux

To uninstall the driver, you must meet the following requirements:

- You must have superuser privileges.
- The driver must not be in use.

Typing rpm -e nivxi-vxipc nivisa removes the NI-VXI/NI-VISA software.

NI-VXI/NI-VISA Software Location

The NI-VXI software is configured to be loaded in the /usr/local/nivxi directory. If you have installed the software in another directory, set the NIVXIPATH environment variable to your directory. For example, if you have installed NI-VXI in /usr2/nivxi, type the following command:

- in csh:

```
setenv NIVXIPATH /usr2/nivxi
```

- in bash or ksh:

```
export NIVXIPATH=/usr2/nivxi
```

The NI-VISA software is configured to be loaded in the `/usr/local` directory. If you have installed the software in another directory, set the `VXIPNPPATH` environment variable to your directory. For example, if you have installed NI-VISA in `/usr2/vpp`, type the following command:

- in `csh`:

  ```
  setenv VXIPNPPATH /usr2/vpp
  ```

- in `bash` or `ksh`:

  ```
  export VXIPNPPATH=/usr2/vpp
  ```

Place these lines in your `.cshrc` or `.login` (C shell) or `.profile` (Bourne or Korn shells) so they will execute automatically the next time you log in.

Completing the Software Installation

After the software is installed, run Resman, 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.

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 4, *NI-VXI Configuration Utility*, for instructions on using the configuration editors in `vxiedit`.

4

NI-VXI Configuration Utility

This chapter contains instructions for using the VXI Text Resource Editor utility of the NI-VXI software to configure the VXIpc embedded computer.

`vxiedit` 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.

Note A text-based version, `vxitedit`, is also available as an alternative. 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`, type `vxiedit` at the command prompt. You can run `vxiedit` from any directory, but make sure that both the `PATH` and `NIVXIPATH` environment variables have the destination directory of the NI-VXI software added to them. `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 `/usr/local/nivxi`.

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

Figure 4-1 shows the main menu of the `vxiedit` resource editor.

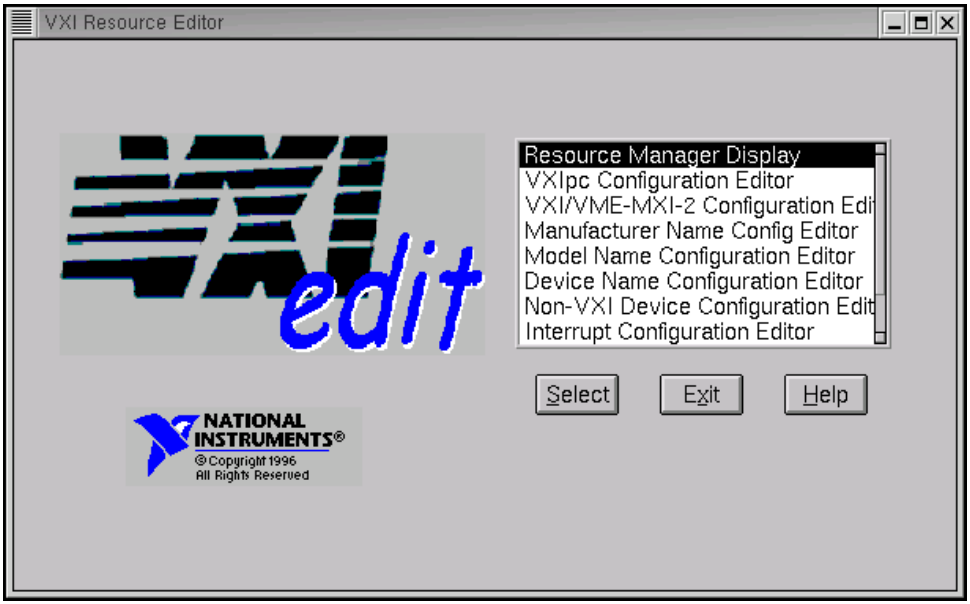


Figure 4-1. vxiedit Main Screen

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 the *NI-VXI Graphical Utilities Reference Manual*.

VXIpc Configuration Editor

Figure 4-2 shows the opening screen of the VXIpc Configuration Editor. Notice that the screen displays the serial number and hardware revision of the VXIpc in addition to several configuration options.

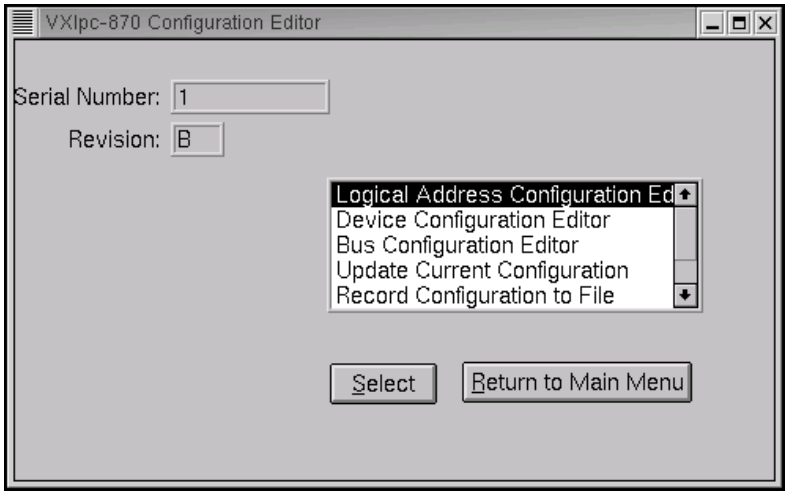


Figure 4-2. VXIpc Configuration Editor

The first three options under the VXIpc Configuration Editor are as follows:

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

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

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 controller EEPROM and files used by NI-VXI. 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.

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 controller configuration EEPROM.

If you want to update the VXIpc controller 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 controller 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

The Logical Address Configuration Editor has options for the device's logical address, device type, address space, VXI shared memory, and the resource manager delay. Figure 4-3 shows the Logical Address Configuration Editor.

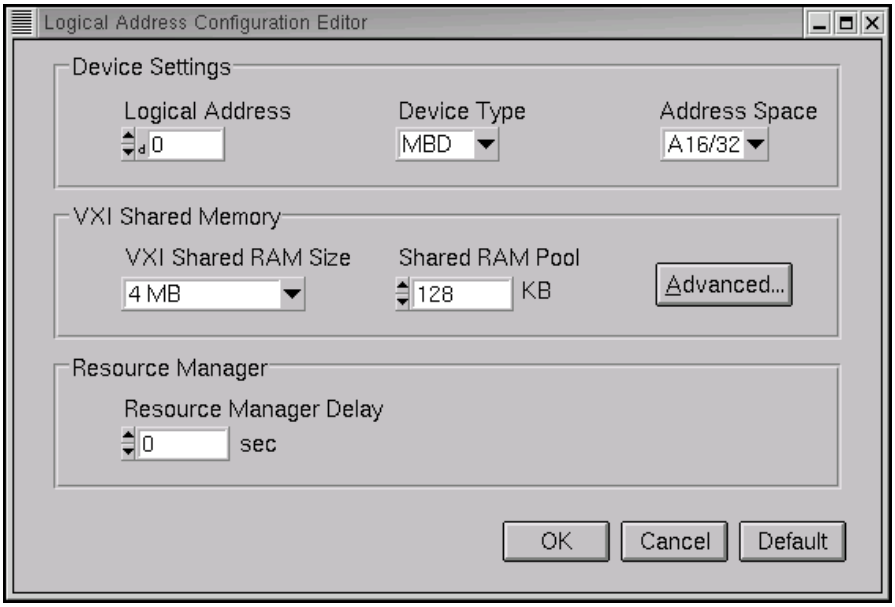


Figure 4-3. VXIpc Logical Address Configuration Editor

The following sections describe the options you can select for each of the fields.

Logical Address

This parameter sets the logical address of the VXIpc controller. 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 controller. 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. You can configure the VXIpc controller 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 options relating to VXI shared memory are disabled when the **Address Space** option is set to A16. Only if you select A16/A24 or A16/A32 are the following options relevant:

- **VXI Shared RAM Size**
- **Shared RAM Pool**
- **Upper/Lower Half Window Byte Swapping**
- **Upper/Lower Half Window Address Mapping**

VXI Shared RAM Size

This field indicates the amount of RAM (in bytes) that is shared in either A24 or A32 space. This determines the *total* shared RAM size. Setting this field to **All of System RAM** detects how much memory you have installed in your VXIpc controller and requests the same amount of A24 or A32 space. However, if you have more than 8 MB installed in your VXIpc, you need to change the **Address Space** field to use A32 space.

Shared RAM Pool

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

The shared RAM pool is used by `VXImemAlloc()` function calls. 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 the computer to enable the change.

Memory Range	Default Value
0 to 65,535 KB	0 KB

Advanced Shared RAM Settings

Clicking the **Advanced** button displays the dialog box in Figure 4-4. The VXI shared RAM is divided into two halves, or *windows*. You can select the byte order and mapping scheme for each half independently. These configuration options are intended for more advanced users.

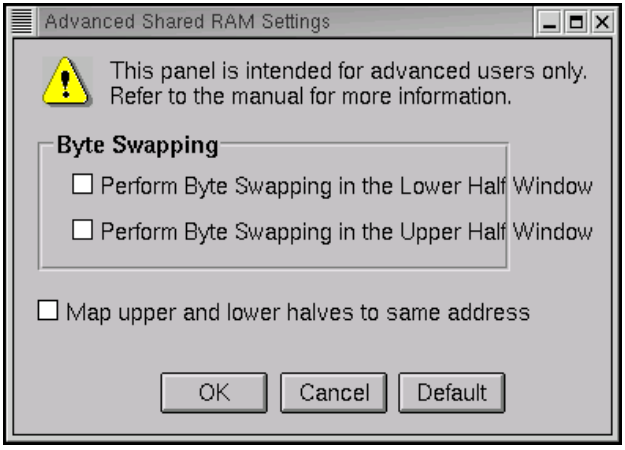


Figure 4-4. Advanced Shared RAM Settings

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 need to enable byte swapping 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.

The default setting maps each half window to a unique local address on the VXIpc controller. If you change this setting, the buffer in system RAM is dual-ported to the VXIbus in both Little Endian and Big Endian byte order. The setting of the **Byte Swapping** option for each half window determines whether the byte order is Little Endian or Big Endian.

Resource Manager Delay

Note This field is effective only when the VXIpc controller 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 65,535 s	0

Device Configuration Editor

Firgure 4-5 shows the Device Configuration Editor. The Device Configuration Editor configures options for remote controller communication and local device settings.

Figure 4-5. VXIpc Device Configuration Editor

System IRQ Level

Remote controllers can report events such as triggers and DMA to the VXIpc 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

Note When the system IRQ line is disabled, the remote controller functionality is not available. Enable the system IRQ line if you are using a multi-mainframe system.

The VXI IRQ designated as system IRQ line cannot be disabled using the `disablevxiint` or `disablevxitosignalint` functions. The VXIpc controller always acknowledges it automatically when it is the Resource Manager.

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 controller 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 controller's 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 is configured as a message-based device.

Servant Area Range	Default Value
0 to 255	0

Note If the VXIpc controller 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 controller supports.

Interrupt Handlers	Default Value
0 to 7	1

Number of Interrupters

This field gives the number of interrupters that the VXIpc controller 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 controller 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 controller from the Resource Manager (if the VXIpc is *not* the Resource Manager in your system). This field is meaningful only when the VXIpc 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 4-6 shows the Bus Configuration editor. Use the Bus Configuration Editor to configure VXI bus settings, PCI bus settings, and bus arbitration settings for the VXIpc controller.

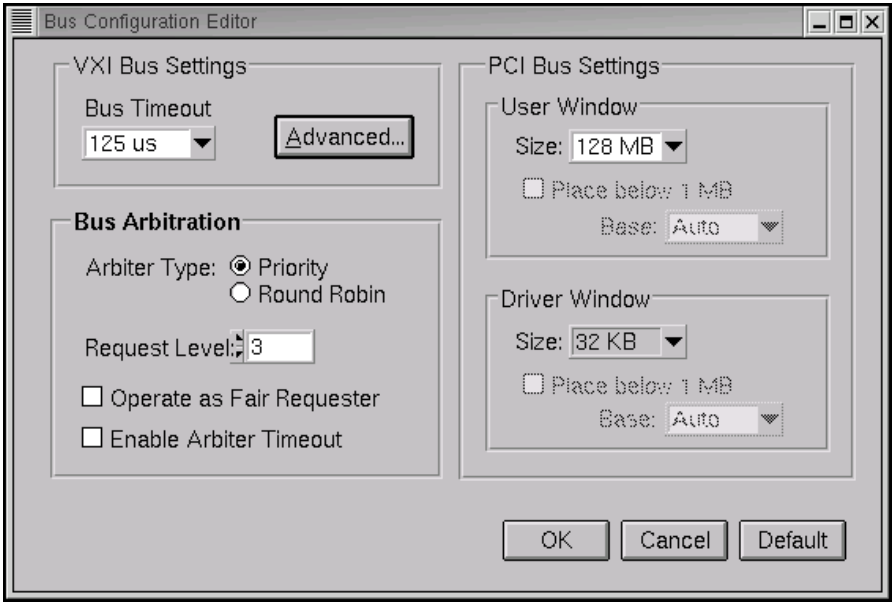


Figure 4-6. VXIpc Bus Configuration Editor

VXI 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 feature is applicable only if the VXIpc controller 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.

Arbiter Type

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

When configured for Priority arbitration, the VXIpc grants the bus to the highest pending bus request level. If you select **Round Robin** arbitration mode, the VXIpc 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 controller uses one of the four VXIbus request levels (0 to 3) to request use of the VXI Data Transfer Bus (DTB). The VXIpc requests use of the DTB whenever a local cycle maps into a VXIbus cycle.

The VXIpc 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 to use any of the other three request levels (0, 1, or 2) by changing the setting of the **Request Level** field. You may want to change request levels to change the priority of the VXIpc controller's request signal. For more information, refer to the VMEbus specification.

VXI Fair Requester

The VXIpc controller is always a Release On Request requester. However, you can configure whether the VXIpc acts as either a fair or unfair requester on the VXIbus. By default, the VXIpc controller operates as 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 controller when it is acting as the VMEbus arbiter. This feature applies only to a VXI Slot 0 VXIpc. By default, this option is disabled.

If you enable this feature, the timer begins when the arbiter circuit on the VXIpc 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.

User Window and Driver Window

The VXIpc controller 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 controller.

The windows are mapped to PCI base address registers and determine the amount of PCI memory space the VXIpc requests from the PCI system during initialization.

Window Size

The amount of space you can allocate for the user window is system dependent. By changing the value 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 also can disable this option. Disabling the user window causes the VXIpc to request the minimum amount of address space on the PCI bus. With the window disabled, you cannot perform any low-level function calls such as `VXIpeek()`, `VXIpoke()`, and `MapVXIAddress()`.

The default setting for the user window is set at 64 KB. National Instruments recommends that you have a user window of at least this value. If you are going to initiate 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 is system defined and is not user configurable.

Note Neither the user window nor the driver window can be placed below 1 MB with the Linux NI-VXI driver. Therefore, the **Place below 1 MB** option and the **Window Base** option will always be disabled.

Advanced

The **Advanced** button under VXI Bus Settings displays the dialog box in Figure 4-7.

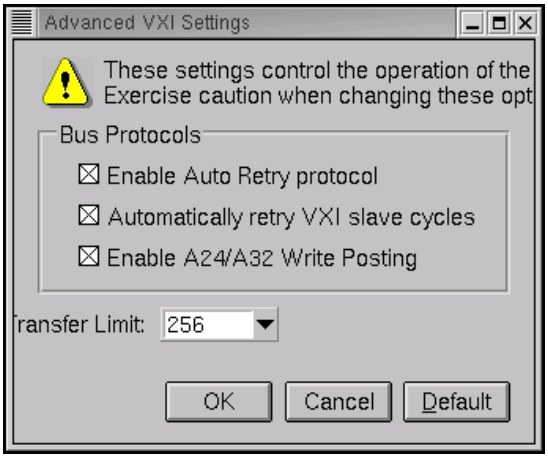


Figure 4-7. Advanced VXI Settings

Automatic VXIbus Retry Protocol

When the **Enable Auto Retry protocol** option is active, the VXIpc controller 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

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

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 automatically retries any PCI cycle when the PCI host responds to a cycle with a retry. The VXIpc 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, you may find it beneficial to disable this feature. With this feature disabled, you can lower the value of the **VXI Bus Timeout** because there is no delay from the inward cycles being retried.

Note The VXIpc 870 has a limit on the number of automatic retries it will perform on any one cycle. If the limit is exceeded and the VXIpc 870 receives another retry, it will pass a retry or BERR (depending on whether the **Enable Auto Retry protocol** option is active or disabled) to the VXIbus even though the **Automatically retry VXI slave cycles** option is active.

A24/A32 Write Posting

The VXIpc controller can increase performance with its capability to post write cycles from the VXIbus. You should post write cycles 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 field affects write cycles to the VXIpc controller via its requested memory space from the VXIbus. When this option is enabled, the VXIpc controller completes a VXIbus write cycle before writing the data from the cycle to the local destination on the VXIpc.

VXI Transfer Limit

You can use the **Transfer Limit** field to set how many data transfers the VXIpc controller 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 to hold the VXIbus long enough to perform 256 transfers (the default value), you can select a smaller value for this field.

VXI/VME-MXI-2 Configuration Editor

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

Note Throughout this section, the term *VXI/VME-MXI-2* denotes that the information applies both 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 4-8.

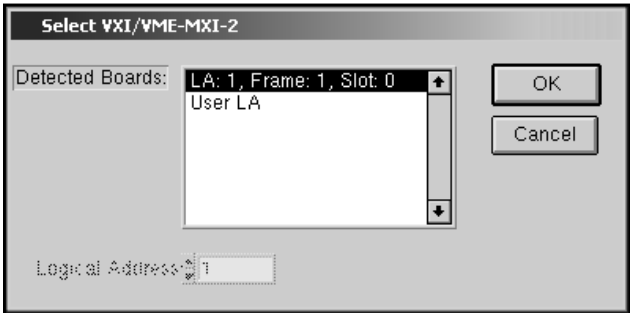


Figure 4-8. VXI/VME-MXI-2 Selection Dialog Box

Select the device you want to configure from the **Detected Boards** pull-down list, or you can select **User LA** and type in the board'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 4-9, that you can use to modify its configuration settings. 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 Figure 4-9.

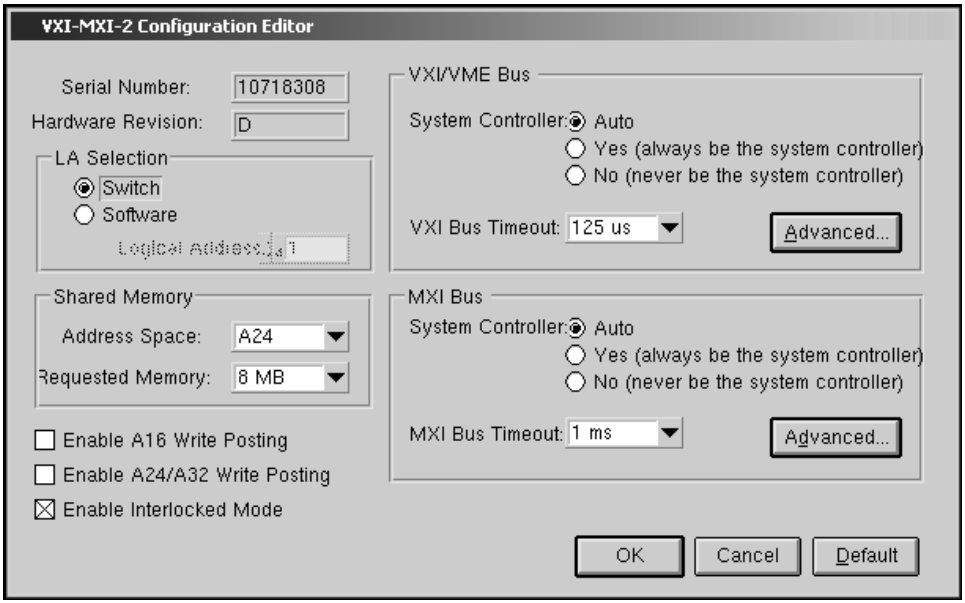


Figure 4-9. VXI/VME-MXI-2 Configuration Editor

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, because 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 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, *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 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 Chapter 4, *VXI-MXI-2 Configuration and Installation*, of the *Getting Started with Your PCI-MXI-2 and the NI-VXI/NI-VISA Software for Linux* 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.

Caution 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 the **Advanced** button to reach additional configuration options for the VXI/VME Bus portion of this editor, as shown in Figure 4-10. These options are intended for more advanced users.

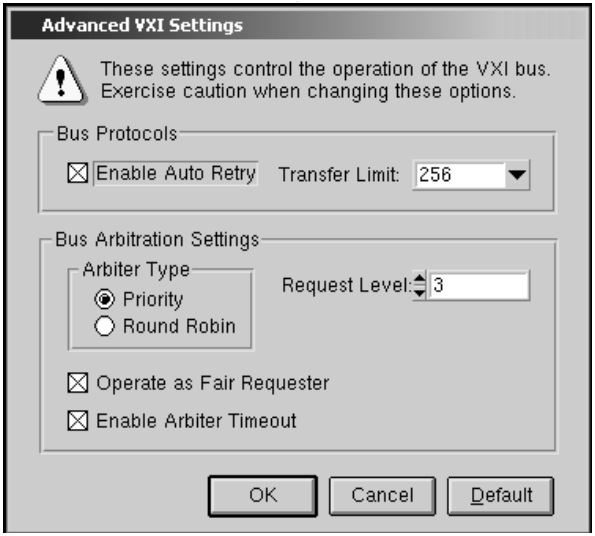


Figure 4-10. Advanced VXI Settings

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. In **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 or PCI-MXI-2B 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 MXI 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 MXI Bus System Controller. The functionality of this control is similar to that of the **VXI Bus Timeout** control described in 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 MXI Bus 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 the **Advanced** button to reach additional configuration options for the MXI Bus portion of this editor, as shown in Figure 4-11. These options are intended for more advanced users.

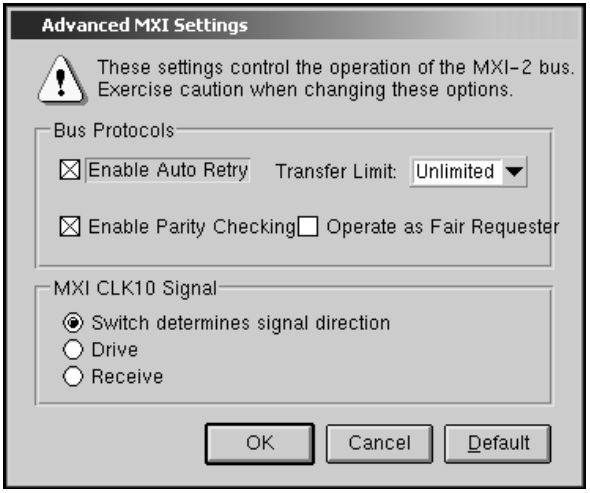


Figure 4-11. Advanced MXI Settings

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 in 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.

Note 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 of whether 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.

Caution 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.

5

Using the NI-VXI/NI-VISA Software

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

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

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

The NI-VXI software was designed for use in VXI systems. Because VXI is a superset of VME, you can also use the NI-VXI functions as a comprehensive set of programming tools for VME systems. Refer to the *NI-VXI User Manual* and the NI-VXI online help for overviews of NI-VXI and detailed descriptions of the NI-VXI functions. The user manual is available in the *NIVXI*/manuals directory (where *NIVXI* refers to the actual location where you have installed the NI-VXI software). Use the Acrobat Reader program to open and navigate through this manual. Notice that the function descriptions indicate whether they apply to VXI only or both VXI and VME. The following function classes apply only to VXI:

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

Refer to the *NI-VISA User Manual* to learn about VISA and how to use it in your system. The NI-VISA online help describes the attributes, events, and operations you can use in NI-VISA. The user manual is available in the *VXIpnp*/linux/NIvisa/Manuals directory (where *VXIpnp* refers to the actual location where you have installed the NI-VISA software). Use the Acrobat Reader program, version 3 or later, to open and navigate through this manual.

Interactive Control of NI-VXI/NI-VISA

The easiest way to learn how to communicate with your instruments is by controlling them interactively. Use the VXI/VME interactive control utility—`vic` and its functionally equivalent text-only version, `victext`—to write to and read from your instruments. This utility displays the status of your VXI/VME transactions and informs you of any errors that occur.

Refer to the online help for instructions on how to use `vic` and `victext` and to learn about the features of each.

Example Programs

The `NIVXI/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 sure that the environment variable `NIVXIPATH` is set correctly as described in Chapter 3, *NI-VXI/NI-VISA Software Installation*. Also refer to your software utilities reference manual for additional examples.

For NI-VISA programming examples, look in *`VXIpnp`*`/linux/NIvisa/Examples`.

Programming Considerations

The following sections contain information for you to consider when developing Linux applications that use the NI-VXI/NI-VISA bus interface software.

Multiple Applications Using the NI-VXI and NI-VISA Libraries

Multiple-application support is another feature in the NI-VXI/NI-VISA libraries. You can have several applications that use the NI-VXI and/or NI-VISA libraries running simultaneously. In addition, you can have multiple instances of the same application running simultaneously.
The NI-VXI/NI-VISA functions perform in the same manner whether you have only one application or several applications (or several instances of a single application) all using the NI-VXI/NI-VISA libraries.

Low-Level Access Functions

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

The functions `viMapAddress()` and `MapVXIAddress()` return a pointer for use with low-level access functions. It is strongly recommended that you use the functions to access the memory instead of directly dereferencing the pointer. Using these functions makes the NI-VXI/NI-VISA software more portable between platforms. Refer to the *Compiling Your C Program for NI-VXI/NI-VISA* section for more information on portability issues, and to your NI-VXI or NI-VISA software reference manual for more information on low-level VXIbus or VMEbus access functions.

Local Resource Access Functions

Notice that sharing the system memory with the VXI/VME system does not mean that the entire range of shared system memory is available to be used for VXI/VME 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.

Caution Use `viMemAlloc()` or `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 `viMemAlloc()` or `VXImemAlloc()` to receive data may cause your computer to crash or behave incorrectly.

System Configuration Functions

The System Configuration functions provide the lowest-level initialization of your VXI controller. For NI-VXI, use the `InitVXIlibrary()` function at the start of each application and the `CloseVXIlibrary()` function at the end of each application. For NI-VISA, use `viOpenDefaultRM()` at the start of each application and the `viClose()` function at the end of each application.

Compiling Your C Program for NI-VXI/NI-VISA

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 programs using the makefile provided to get familiar with how the functions operate. The example programs are broken into multiple files, and each one shows how to use different groups of functions. You can then modify the sample programs to try out different aspects of the NI-VXI software.

The sample programs for the Linux CC compiler are in the `/usr/local/nivxi/examples` directory for NI-VXI and `/usr/local/vxipnp/linux/NIvisa/Examples` directory for NI-VISA.

The easiest way to compile the sample programs is to use the makefile included with the NI-VXI/NI-VISA software. For example, go to the `examples` directory and type `make`.

Symbols

You may need to define a symbol so that the NI-VXI library can work properly with your program. The `VXILINUX` symbol is required for NI-VXI applications, but not for NI-VISA applications. `VXILINUX` designates the application as a Linux NI-VXI application. You can define this symbol using a `#define` statements in the source code, or you can use the `-D` option in your compiler. If you use a `#define` statement, you must define the symbol 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 symbol.

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

```
#define VXILINUX
.
.
.
#include <nivxi.h>
```

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.

A

Default Settings

This appendix summarizes the default settings for the hardware and software in your kit. If you need more information about a particular setting, or if you want to try a different configuration, refer to the *VXIpc 870 Series User Manual* or the *VXIpc 770/870B Series User Manual* for your hardware reference and to Chapter 4, *NI-VXI Configuration Utility*, for your software reference.

Because you can also use `vxitedit` to configure a VXI-MXI-2 or a VME-MXI-2, this appendix also summarizes the software default settings for the VXI/VME-MXI-2

VXIpc 870 Series Controller

This section summarizes the hardware default settings for the VXIpc 870 Series controller.

Table A-1. VXIpc 870 Series Hardware Default Settings

Jumper	Default Setting	Optional Setting
J12	Enable automatic Slot 0 detection	Force Slot 0; Force Non-Slot 0
S1	MITE user configuration	MITE factory configuration
S2	Enable MITE self-configuration	Disable MITE self-configuration
W1, 3, 5, and 7	CPU bus factor	**Note:** Refer to the *CPU Bus Factor for VXIpc 870 Series* section.
W4	100 MHz CPU bus speed	66 MHz CPU bus speed
W6	Normal CMOS operation	Clear CMOS
W8	Flash write enable	Flash Protection

Table A-1. VXIpc 870 Series Hardware Default Settings (Continued)

Jumper	Default Setting	Optional Setting
W10	Enable Ethernet Serial EEPROM configuration	Disable Ethernet Serial EEPROM configuration (uses default power on values)
W11–12	Enable 16-bit SCSI termination	SCSI termination
W15	Voltage monitor only required voltages	Voltage monitor all voltages

Caution Do *not* adjust any jumpers or switches not listed in Table A-1 or that are not documented in this manual unless directed by National Instruments. Other configuration jumpers are shown in the event that National Instruments Technical Support needs to make adjustments to your settings.

CPU Bus Factor for VXIpc 870 Series

The jumper settings in Figure A-1 are provided to allow the user to upgrade as newer CPUs enter the market. Contact National Instruments for jumper settings as new CPUs are released.

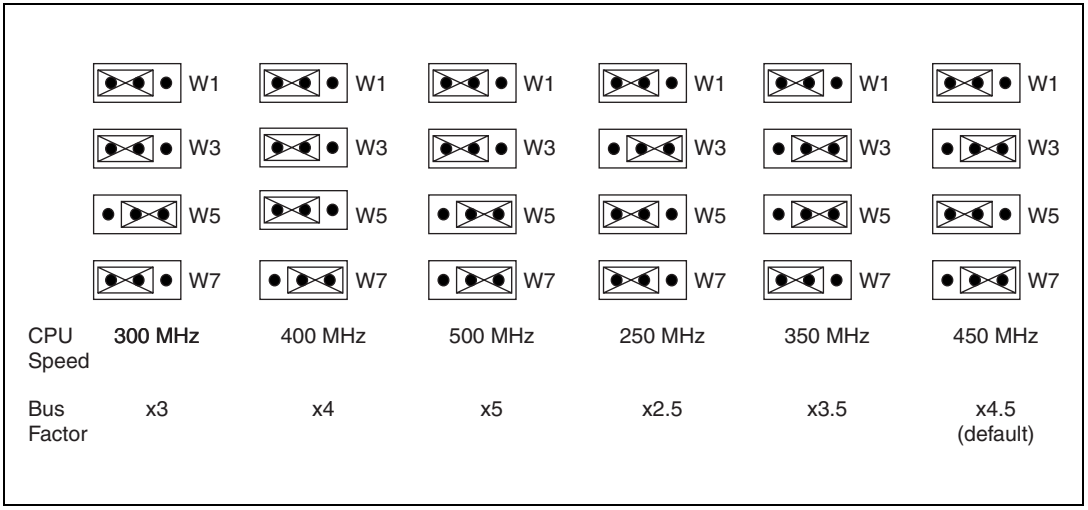


Figure A-1. CPU Bus Factor Jumper Settings

VXIpc 770/870B Series Controller

This section summarizes the hardware default settings for the VXIpc 770/870B Series controller.

Table A-2 lists the factory default settings and options for the onboard jumpers and switches.

Table A-2. VXIpc 770/870B Series Hardware Default Settings

Jumper	Default Setting	Optional Setting
W5	1-2 Normal CMOS operation	2-3 Clear CMOS
W6	2-3 16-bit SCSI termination enabled	1-2 SCSI termination disabled
J20	No jumper	Master/Slave/CSEL*
J17	3-4 Automatic slot zero detection	1-2 Non-Slot zero 5-6 Force Slot zero
W2	2-3 Enable MITE self configuration	1-2 Disable MITE self configuration
W1	2-3 MITE user configuration	1-2 MITE factory configuration
S1	1-2 Internal oscillator	2-3 External oscillator
* These pins are generally defined in a figure on the hard drive cover.		

Caution Do *not* adjust any jumpers or switches not listed in Table A-2 or that are not documented in this manual unless directed by National Instruments. Other configuration jumpers are shown in the event that National Instruments technical support needs to make adjustments to your settings.

NI-VXI Settings

This section summarizes the NI-VXI default settings for the VXIpc 770/870B Series controllers. Tables A-3, A-4, and A-5 list the factory default NI-VXI settings. You can change these settings through the `vxiedit` utility.

Table A-3. Logical Address Configuration Editor Default Settings

Editor Field	Default Setting
Logical Address	0
Device Type	MBD
Address Space	A16
VXI Shared RAM Size	0 KB
Shared RAM Pool	0 KB
Lower Half Window Byte Swapping	Disabled (non-swapped)
Upper Half Window Byte Swapping	Disabled (non-swapped)
Map Upper and Lower Halves to Same Address	Disabled
Resource Manager Delay	0 s

Table A-4. Device Configuration Editor Default Settings

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

Table A-5. Bus Configuration Editor Default Settings

Editor Field	Default Setting
Bus Timeout	500
Automatic Retry Protocol	Enabled
Automatic VXI Slave Cycle Retry	Enabled
A24/A32 Slave Write Posting	Disabled
VXI Transfer Limit	256
Arbiter Type	Priority
Request Level	3
Fair Requester	Disabled
Arbiter Timeout	Disabled
User Window Base	Auto
User Window Size	64 KB
User Window Below 1 MB	No
Driver Window Base	Auto
Driver Window Size	32 KB
Driver Window Below 1 MB	No

B

NI-VXI/VISA Software Overview

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

Main Programs and Files

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

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

- Resman is the National Instruments multiple-mainframe Resource Manager.
- `vic` is a graphical interactive control program. This program is described in detail in the *NI-VXI Graphical Utilities Reference Manual*.
- `victext` is a text-based interactive control program with all the power of `vic`. This program is described in detail in the *NI-VXI Text Utilities Reference Manual*.
- `vxiedit` is a graphical VXI resource editor program. This program is described in detail in the *NI-VXI Graphical Utilities Reference Manual*.
- `vxitedit` is the text-based VXI resource editor program. This program is described in detail in the *NI-VXI Text Utilities Reference Manual*.
- `vxiinit` is the driver initialization program. Depending on the version of the driver installed on your system, it may or may not be present.
- The README file contains the latest updates and corrections to the manual when appropriate.

Header Files for NI-VXI

The *NIVXI*/include directory (where *NIVXI* is the actual location where you installed the NI-VXI software package) contains the following include files for the C language interface:

- nivxi.h is the main header file containing the C prototypes for the NI-VXI functions.
- datasize.h contains data size specifications.
- busacc.h contains parameter and return values for the bus access functions.
- devinfo.h contains parameter and return values for the device information and system configuration functions.
- vxiint.h contains parameter and return values for the interrupt and signal functions.
- sysint.h contains parameter and return values for the system interrupt functions.
- trig.h contains parameter and return values for the trigger functions. This file is useful in VXI systems but is not applicable for VME systems.
- ws.h contains parameter and return values for the Commander and Servant Word Serial functions. This file is useful in VXI systems but is not applicable for VME systems.

Header Files for NI-VISA

The *VXIpnp*/linux/include directory (where *VXIpnp* is the actual location where you installed the NI-VISA software package) contains the following include files for the C language interface:

- visa.h is the main header file containing the C prototypes for the NI-VISA functions.
- visatype.h contains VXI*plug&play* data type specifications.
- vpptype.h contains useful definitions for instrument drivers.

C

Common Questions

This appendix addresses common questions you may have about using the NI-VXI software on the VXIpc platform for Linux.

What is the function of the NI-VXI utilities?

The utilities have the following responsibilities:

- Resman—This utility initializes and configures all the other devices in your VXI system.
- `vxiedit` and `vxitedit`—These utilities configure your National Instruments hardware.
- `vic` and `victext`—Use these utilities to interactively communicate with VXI devices over the VXIbus using the NI-VXI API.

What does Resman do?

The Resman utility performs the duties of a VXI Resource Manager as discussed in the VXIbus specification. When you set a National Instruments controller to Logical Address 0, you will at some point need to run Resman to configure your VXI instruments. If your controller uses a different (non-zero) logical address and is a message-based device, you need to start Resman before running it on the Logical Address 0 computer.

When do you need to run Resman?

You need to run Resman whenever you need to configure your VXI instruments. For example, if you power-cycle your VXI chassis, your instruments will be reset, and you will need to run Resman to configure them. You can get into trouble if you run Resman when your devices are not in a reset state. Therefore, if you have to run Resman after running it once, you should reset all of your VXI instruments.

How do I handle VME devices?

Although there is no way to automatically detect VME devices in a system, you can add them easily through the Non-VXI Device Editor in `vxiedit`. After you assign a pseudo-logical address and other resource values, Resman can configure the VME devices into your VXI system.

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

Run the NI-VXI utility program `vic`. Under the **Text** tab, type `ver` in the command field. The utility displays the versions of `vic` and NI-VXI, and the latest VXIpc hardware revision that this NI-VXI driver supports.

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

Run the NI-VXI utility program `vic`. Under the **Text** tab, type `ver` in the command field. The utility displays the versions of `vic` and NI-VXI, and the latest VXIpc hardware revision that this NI-VXI driver supports.

How can I determine the serial number and hardware revision of the VXIpc controller?

Run `vxiedit` and select the **VXIpc Configuration Editor**. The editor window displays the serial number and hardware revision of the VXIpc.

Which NI-VXI utility program must I use to configure the VXIpc controller?

Use the graphical `vxiedit` or the text-mode `vxitedit` utility to configure the VXIpc. You do not need to run `vxiedit` if you are satisfied with the default settings. Refer to Chapter 4, *NI-VXI Configuration Utility*, for complete details on using the configuration editors.

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

Use the Resman utility to perform startup Resource Manager operations. This utility uses the settings configured in `vxiedit`. It initializes your VXI/VMEbus system and stores the information that it collects to the `resman.tbl` file in the `tbl` subdirectory of the `nivxi` directory. You can access this information using the NI-VXI system configuration functions described in detail in Chapter 2, *Function Reference*, of the *NI-VXI Programmer Reference Manual*.

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 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.

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

There are several reasons why you could receive this message. If you turn off the computer while the configuration update process is still in progress, the VXIpc functions normally except when using `vxiedit`. To correct this problem, switch to the factory configuration as described in the configuration and installation chapter of your VXIpc user manual. Reboot the computer and update the configuration, or load the configuration from file.

What do the LEDs on the front of the VXIpc controller mean?

Refer to your VXIpc user manual for a description of the front panel LEDs.

Is something wrong if the red SYSFAIL and FAILED LEDs stay lit after booting the VXIpc controller?

If either the **SYSFAIL** or **FAILED** LED remains lit, refer to your VXIpc user manual for troubleshooting steps.

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

Yes. The VXIpc uses the 32-bit PCI bus to interface to the VXIbus. In fact, its VXIbus circuitry 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 backplane. However, the oscillator for the VXIpc 800/700 and the EXTCLK input on the VXIpc 870 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.

What is the accuracy of the CLK10 signal?

The CLK10 signal generated by all of the VXIpc controllers is ±100 ppm (0.01%) as per the VXIbus specification. If you need a more accurate CLK10 signal on the VXIpc 870 Series, you can use the EXTCLK connector on its front panel.

What kind of monitor can I use with the VXIpc controller?

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

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

How do I connect an external speaker to get audio capability?

A twisted-pair cable connects the front panel audio connector to the VXIpc 870/870B Series motherboards. 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? What is the maximum amount of RAM that I can have?

For information about adding RAM to the VXIpc controller, refer to Appendix A, *Specifications*, in your VXIpc user manual.

Which interrupt levels are free to be used by ISA bus boards on the VXIpc-872 controller? Which area of upper memory (adapter space) is free for use by ISA bus boards or expanded memory manager software programs?

Refer to the appendix on VXIpc system resources in your VXIpc user manual for information on the available port I/O register space, upper memory area, interrupts, and DMA channels.

How do I install the VXIpc controller in a slot other than Slot 0?

The VXIpc controller automatically detects whether it is in Slot 0 of a VXIbus mainframe. You do not need to change jumper settings to install the VXIpc in a slot other than Slot 0 unless you have defeated the first slot

detector (FSD) circuitry by changing the appropriate jumper setting on the VXIpc.

Refer to the configuration and installation chapter of your VXIpc 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?

You can view these parameters in the BIOS setup. To enter the BIOS setup, reboot the VXIpc and press the <Del> key during the memory tests.

Whenever I try to execute any of the NI-VXI utilities, I receive a message similar to this one:

```
error in loading shared libraries: libnivxi.so: cannot
open shared object file: no such file or directory
```

What does this error message mean?

This usually means that the application could not load the NI-VXI library. Check the environment variable `NIVXIPATH` and your `/etc/ld.so.conf` file. `NIVXIPATH` should be set only to the directory where you installed NI-VXI (`/usr/local/nivxi` by default). `/etc/ld.so.conf` should include the directory where you installed the NI-VXI library (`/usr/local/nivxi/lib` by default). Run `ldconfig` to reread this file.

Whenever I try to execute any of the NI-VXI utilities, I receive a message that it could not find a particular file even though the file does exist. What is wrong?

When a NI-VXI utility cannot find a file that it needs, it usually means that one of the environment variables is set incorrectly. Check the environment variable `NIVXIPATH` and your `/etc/ld.so.conf` file. `NIVXIPATH` should be set only to the directory where you installed NI-VXI (`/usr/local/nivxi` by default). `/etc/ld.so.conf` should include the directory where you installed the NI-VXI library (`/usr/local/nivxi/lib` by default). Run `ldconfig` to reread this file.

You can also receive this error message if you do not have full permissions to some of the NI-VXI files and directories. Users who will be using NI-VXI should have full permissions to the `tbl` and `examples` directories. They should also have read-write permissions for all the files contained in those directories.

D

Technical Support and Professional Services

Visit the following sections of the National Instruments Web site at ni.com for technical support and professional services:

- **Support**—Online technical support resources include the following:
  - **Self-Help Resources**—For immediate answers and solutions, visit our extensive library of technical support resources available in English, Japanese, and Spanish at ni.com/support. These resources are available for most products at no cost to registered users and include software drivers and updates, a KnowledgeBase, product manuals, step-by-step troubleshooting wizards, hardware schematics and conformity documentation, example code, tutorials and application notes, instrument drivers, discussion forums, a measurement glossary, and so on.
  - **Assisted Support Options**—Contact NI engineers and other measurement and automation professionals by visiting ni.com/ask. Our online system helps you define your question and connects you to the experts by phone, discussion forum, or email.
- **Training**—Visit ni.com/custed for self-paced tutorials, videos, and interactive CDs. You also can register for instructor-led, hands-on courses at locations around the world.
- **System Integration**—If you have time constraints, limited in-house technical resources, or other project challenges, NI Alliance Program members can help. To learn more, call your local NI office or visit ni.com/alliance.

If you searched ni.com and could not find the answers you need, contact your local office or NI corporate headquarters. Phone numbers for our worldwide offices are listed at the front of this manual. You also can visit the Worldwide Offices section of ni.com/niglobal to access the branch office Web sites, which provide up-to-date contact information, support phone numbers, email addresses, and current events.

Glossary

Prefix	Meanings	Value
p-	pico-	10^{-12}
n-	nano-	10^{-9}
µ-	micro-	10^{-6}
m-	milli-	10^{-3}
k-	kilo-	10^{3}
M-	mega-	10^{6}
G-	giga-	10^{9}
t-	tera-	10^{12}

Symbols

°	Degrees.
Ω	Ohms.
%	Percent.

A

A	Amperes.
A16 space	VXIbus address space equivalent to the VME 64 KB short address space. In VXI, the upper 16 KB of A16 space is allocated for use by VXI devices configuration registers. This 16 KB region is referred to as VXI configuration space.
A24 space	VXIbus address space equivalent to the VME 16 MB *standard* address space.
A32 space	VXIbus address space equivalent to the VME 4 GB *extended* address space.

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

B

B	Bytes.
backplane	An assembly, typically a printed circuit board, with 96-pin connectors and signal paths that bus the connector pins. A C-size VXIbus system will have two sets of bused connectors called J1 and J2. A D-size VXIbus system will have three sets of bused connectors called J1, J2, and J3.
BERR*	Bus error signal.
binary	A numbering system with a base of 2.
BIOS	Basic Input/Output System. BIOS functions are the fundamental level of any PC or compatible computer. BIOS functions embody the basic operations needed for successful use of the computer's hardware resources.

block-mode transfer	An uninterrupted transfer of data elements in which the master sources only the first address at the beginning of the cycle. The slave is then responsible for incrementing the address on subsequent transfers so that the next element is transferred to or from the proper storage location. In VME, the data transfer may have no more than 256 elements; MXI does not have this restriction.
BTO unit	Bus Timeout Unit; a functional module that times the duration of each data transfer and terminates the cycle if the duration is excessive. Without the termination capability of this module, a bus master attempt to access a nonexistent slave could result in an indefinitely long wait for a slave response.
bus master	A device that is capable of requesting the Data Transfer Bus (DTB) for the purpose of accessing a slave device.

C

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

D

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

E

ECL	Emitter-Coupled Logic.
EEPROM	Electronically Erasable Programmable Read Only Memory.
embedded controller	An intelligent CPU (controller) interface plugged directly into the VXI backplane, giving it direct access to the VXIbus. It must have all of its required VXI interface capabilities built in.
EMC	Electromechanical Compliance.

EMI	Electromagnetic Interference.
expansion ROM	An onboard EEPROM that may contain device-specific initialization and system boot functionality.
external controller	In this configuration, a plug-in interface board in a computer is connected to the VXI mainframe via one or more VXIbus extended controllers. The computer then exerts overall control over VXIbus system operations.

F

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

H

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

I

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

K

KB	Kilobytes of memory.

L

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

M

m	Meters.
master	A functional part of a MXI/VME/VXIbus device that initiates data transfers on the backplane. A transfer can be either a read or a write.
master-mode operation	A device is in master mode if it is performing a bus cycle which it initiated.
MB	Megabytes of memory.
MBLT	Eight-byte block transfers in which both the Address bus and the Data bus are used to transfer data.
message-based device	An intelligent device that implements the defined VXIbus registers and communication protocols. These devices are able to use Word Serial Protocol to communicate with one another through communication registers.
MITE	A National Instruments custom ASIC, a sophisticated dual-channel DMA controller that incorporates the Synchronous MXI and VME64 protocols to achieve high-performance block transfer rates.
MODID	Module Identification lines.
MTBF	Mean Time Between Failure.
MXI-2	The second generation of the National Instruments MXIbus product line. MXI-2 expands the number of signals on a standard MXIbus cable by including VXI triggers, all VXI interrupts, CLK10, SYSFAIL*, SYSRESET*, and ACFAIL*.

MXIbus	Multisystem eXtension Interface Bus; a high-performance communication link that interconnects devices using round, flexible cables.
MXIbus System Controller	A functional module that has arbiter, daisy-chain driver, and MXIbus cycle timeout responsibility. Always the first device in the MXIbus daisy-chain.

N

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

O

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

P

PCI	Peripheral Component Interconnect. The PCI bus is a high-performance 32- or 64-bit bus with multiplexed address and data lines.
propagation	The transmission of signal through a computer system.

R

register-based device	A Servant-only device that supports VXIbus configuration registers. Register-based devices are typically controlled by message-based devices via device-dependent register reads and writes.
retry	An acknowledge by a destination that signifies that the cycle did not complete and should be repeated.
Resman	The name of the National Instruments Resource Manager in NI-VXI bus interface software. *See* Resource Manager.

Resource Manager	A message-based Commander located at Logical Address 0, which provides configuration management services such as address map configuration, Commander and Servant mappings, and self-test and diagnostic management.
RPM	RPM Package Manager, a widely-used software distribution tool that you can use to install, upgrade, or remove software from your system.

S

s	Seconds.
Servant	A device controlled by a Commander; there are message-based and register-based Servants.
Shared Memory Protocol	A communication protocol that uses a block of memory that is accessible to both a client and a server. The memory block operates as a message buffer for communications.
SIMM	Single In-line Memory Module.
slave	A functional part of a MXI/VME/VXIbus device that detects data transfer cycles initiated by a VMEbus master and responds to the transfers when the address specifies one of the device's registers.
slave-mode operation	A device is in slave mode if it is responding to a bus cycle.
Slot 0 device	A device configured for installation in Slot 0 of a VXIbus mainframe. This device is unique in the VXIbus system in that it performs the VMEbus System Controller functions, including clock sourcing and arbitration for data transfers across the backplane. Installing such a device into any other slot can damage the device, the VXIbus backplane, or both.
statically configured device	A device whose logical address cannot be set through software; that is, it is not dynamically configurable.
SYSFAIL	A VMEbus signal that is used by a device to indicate an internal failure. A failed device asserts this line. In VXI, a device that fails also clears its PASSed bit in its Status register.

SYSRESET	A VMEbus signal that is used by a device to indicate a system reset or power-up condition.
System RAM	RAM installed on your personal computer and used by the operating system, as contrasted with onboard RAM, which is installed on the PCI-MXI-2 or VXI/VME-MXI-2.

T

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

U

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

V

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

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

VXIedit — VXI Resource Editor program, a part of the NI-VXI bus interface software package. Used to configure the system, edit the manufacturer name and ID numbers, edit the model names of VXI and non-VXI devices in the system, as well as the system interrupt configuration information, and display the system configuration information generated by the Resource Manager.

W

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

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.

Index

Numerics

A

B

C

D

E

F

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W