Appendix A

CODE INSTRUMENT OVERVIEW

Introduction

This appendix contains an overview of the functions, applications, and implementations of software modules known as Code Instruments (CIs), and presents comparisons and illustrations of 1260-00C operation with and without CIs.

CIs are a Racal-Dana 1260-00C proprietary feature. They are capable of a wide variety of application-specific translation and control functions. A CI is a set of software routines running on the 1260-00C, but the system sees a CI as a physical Message-Based device. As with physical Message-Based devices, an external controller can communicate directly with the CI via a GPIB secondary address.

CIs perform special functions in the VXI environment. Typical applications of CIs include the following:

- Parsing and interpreting command languages
- Creating virtual (hierarchical) instruments
- Creating a message-based interface for register-based or non-VXI devices

A CI is more than a CPU process that replaces another VXI device's communication path. It has all of the capabilities of a physical Message-Based Commander. These capabilities include the following:

- Having Servant(s) assigned to it
- Having Word Serial communication with its Commander and Servant(s)
- Handling VXI interrupts and signals
- Having communication with the GPIB System Controller through a secondary address
- Having bus mastership for direct access to Register-Based Servants and non-VXI devices
- Having direct access to VXIbus TTL trigger lines

CIs improve the system structure for the following reasons:

GPIB traffic is greatly reduced.
System performance is greatly increased.
The System Controller can treat all devices as if they were Message-Based.
Super instrument structures can be created.

1260-00C Operation Without CIs

Figure A-1 illustrates typical Commander/Servant relationships for 1260-00C operation without CIs. A GPIB System Controller communicates with the local command set and the 1260-00C's Message-Based Servants through GPIB addresses. This is a complete interface solution for Message-Based devices. Although the GPIB and serial controllers are not Commanders of the command parser in the VXI sense, they are its master because it will respond to their commands as if they were its Commander. The 1260-00C maintains independent control paths to the local command set parser from the GPIB, serial controller, and the 1260-00C's Commander.

The 1260-00C has four ports for communicating with other devices. Each port consists of its electrical interface and the associated system software. The 1260-00C communicates with its Commander through the Word Serial Servant port, and with its Servants through the Word Serial Commander port. The GPIB System Controller communicates with the 1260-00C and its Message-Based Servants through the GPIB port, which maps GPIB secondary addresses to VXI logical addresses. A serial controller can access the local command parser through the RS-232 port, and can communicate with the 1260-00C's Message-Based Servants through the Word Serial communication commands.

The System Controller can also directly control Register-Based and non-VXI devices through the VXIbus Access local commands. This solution to the general problem of controlling Register-Based and non-VXI devices is relatively ineffective for high-performance applications because of the low-level functions the System Controller must perform, and the resulting heavy GPIB traffic.

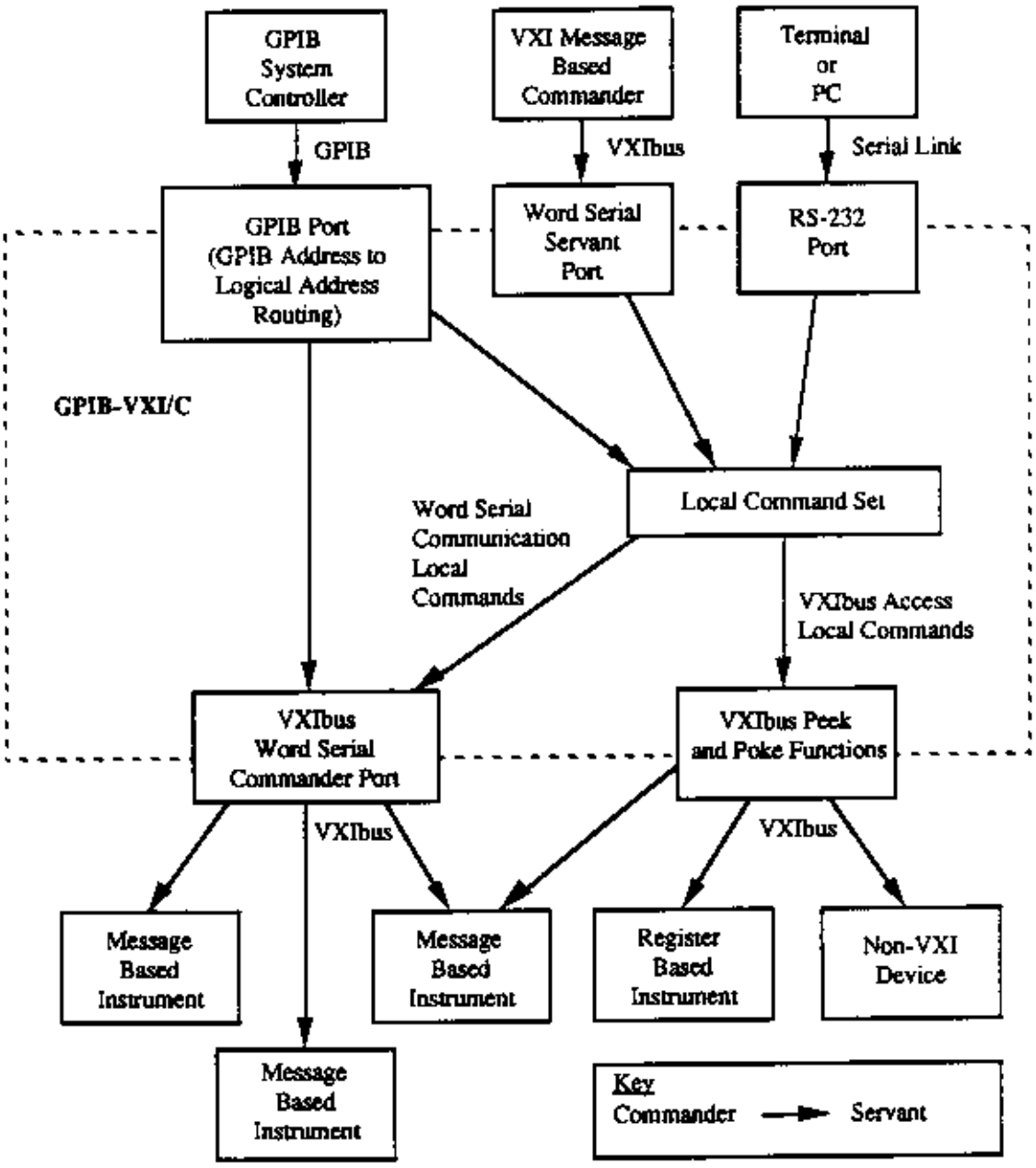


Figure A-1, 1260-00C Operation Without Code Instruments

CI Operation

A CI is a set of software routines that can perform the functions of a physical VXI Message-Based device. These CI capabilities are illustrated in Figure A-2. CIs coexist with the IEEE-488 VXI translation and local command set functions shown in Figure A-1, with the exception that the Word Serial Servant and RS-232 ports support only one command/Servant connection (either to the local command set parser or to a CI, but not both) at one time.

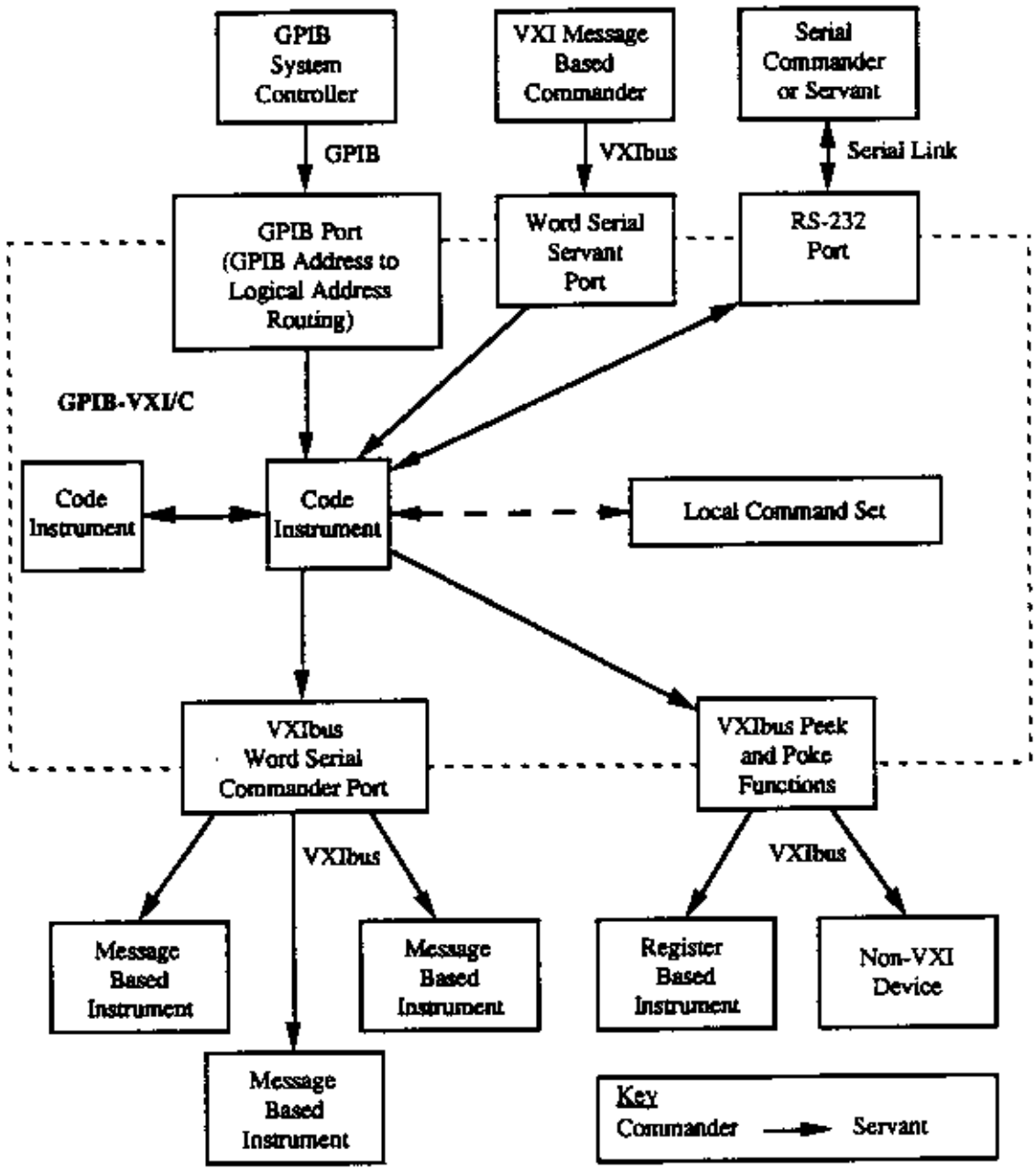


Figure A-2, Code Instrument Operation

CI Characteristics

A CI has all of the abilities of a physical Message-Based Commander. A CI can do the following:

- Set up its own set of configuration registers
- Have Servants assigned to it
- Engage in Word Serial communication with its Commander and Servants
- Receive VXI interrupts
- Send and receive VXI signals
- Directly access the A16 or A24 registers/memory of a VXI or non-VXI device
- Communicate with the GPIB System Controller through a secondary address
- Perform memory-to-memory DMA operations using 68070 DMA channel
- Source or accept a trigger on any one TTL trigger line

As with physical devices, a CI must be an immediate Servant of the 1260-00C in order to have a GPIB address assigned to it. In addition to these VXIbus device capabilities, CIs can also communicate directly with the local command set parser and the serial port.

The 1260-00C emulates the physical capabilities of a Message-Based device for each CI. Because a CI can be a Commander or a Servant, you can construct multi-level hierarchies of CIs and physical Message-Based devices. The only restriction is a CI cannot be mapped out of the hierarchy of devices within the 1260-00C. In other words, a CI can be any of the following:

- The Commander of any number of CIs and/or physical VXI devices
- A top-level Commander
- A Servant of another CI
- A Servant of the 1260-00C's Commander

A CI cannot be the Servant of a physical VXI device that is not the GPIB-00C's Commander.

A CI appears to the GPIB and to other CIs to be a physical device, because it performs all of the functions that a physical Message-Based device performs. If the CI takes control of the physical Word Serial registers on the 1260-00C, it becomes a physical VXI device. Most applications do not require a CI to take control of the physical Word Serial registers, because most CIs function as Commanders to drive other Servants in the system rather than as Servants to higher level Commanders.

A non-VXI device does not have VXI configuration registers, so it does not appear in the VXI device hierarchy. A CI that provides a Message-Based interface for a non-VXI device (together with that non-VXI device) is viewed by the system as a single device. Typical examples of non-VXI devices include:

- VME cards (CPU, Register-Based, memory, and so on)

- CDS 73A-852 adapter module

Downloaded CIs and EPROMed CIs

CIs in the form of binary code can be downloaded into the 1260-00C's RAM. The downloaded modules are called Downloaded CIs, or DCIs. The CI Configuration local commands download and initialize CIs. You can use the DCI form to develop CIs without programming EPROMs, or to create disk-loadable CI applications.

The 1260-00C also has an interface for installing CIs in onboard EPROMs, including a mechanism for automatically initializing them at system start-up. CIs stored in the EPROMs are called EPROMed CIs, or ECIs. You can use the ECI form to create stand-alone CI applications.

Resident CIs

A Resident CI (RCI) that communicates with a CDS 73A-852 adapter is supplied by Racal-Dana as part of the 1260-00C firmware. The 852 adapter is a non-VXI device that requires a special code module somewhere in the system with a Message-Based interface. Appendix B, Using the DMAmove and CDS-852 Adapter Code Instruments, contains information about installing and using the 852 adapter CI.

Summary

With these capabilities, a CI can emulate or replace any existing VXI or VME device, or extend a device's native capabilities to new levels of functionality, as a disk-loadable or stand-alone solution. To summarize, CIs improve the system structure for the following reasons:

- GPIB traffic is greatly reduced.
- Register-Based and non-VXI devices can be treated as if they were Message-Based.
    - The GPIB Controller sees one type of instrument (an IEEE-488 instrument).
    - Standard IEEE-488 communication is possible with all types of VXI/non-VXI instruments.
- GPIB control of a VXIbus system can be implemented uniformly at a high level.
- Application software is simplified due to uniformity of control.
- System performance is greatly increased.
    - Direct access results in a tight coupling with its Servants.
    - Distributed processing removes burden from outside controller.
    - Access to VXIbus bandwidth is accomplished without GPIB overhead.

Appendix B

USING THE DMAmove AND CDS-852 ADAPTER CODE INSTRUMENTS

Introduction

This appendix contains instructions for installing and using the Racal-Dana supplied Code Instruments (CIs). Two CIs come standard in the firmware of the 1260-00C. The first CI is called the DMAmove CI, and is used for dedicating one of the 1260-00C GPIB addresses for use as a high-speed memory port. The second CI is used for controlling one or more Colorado Data Systems (CDS) 73A-852 adapter module.

The main purpose of the 1260-00C is to convert GPIB protocols to VXI protocols. And many features within the VXI environment are not possible with GPIB because of the memory-mapping architecture of VXI. The DMAmove CI gives you very fast direct access to VXI A16 and A24 memory, and to local 1260-00C memory. The 68070 DMA channel 2 is used within the DMAmove CI to move data around more quickly than the VXI Word Serial protocol or individual peeks and pokes.

The 73A-852 is a non-VXI device with communication registers located in A24 space rather than in A16 space. To communicate with the 852 adapter as a Message-Based device, the 73A-852 requires special adapter software. The 1260-00C performs the Message-Based-to-852 communication translation with a CI. The 1260-00C firmware release includes one CDS-852 Position Independent CI. This CI implements the configuration and translation functions required to communicate with up to 12 CDS-852 adapter modules via the GPIB.

Using EPROMed Code Instruments

This section discusses how to install, execute, and delete an EPROMed Code Instrument.

Installing An EPROMed Code Instrument

An EPROMed Code Instrument (the DMAmove and CDS-852 CIs are examples) can be installed either by configuring the non-volatile configuration parameters, or by using the 1260-00C local command set command `ECIboot?`. This section explains how to use the non-volatile configuration editor to permanently install an EPROMed Code Instrument. See Section 4 CI Configuration Commands and Queries, Local Command Set for information about the command `ECIboot?`.

Enter the non-volatile configuration mode as described in Section 3 Non-Volatile Configuration. The following menu is displayed:

1260-VXI Non-Volatile Configuration Main Menu

==========

1). Read In Non-Volatile Configuration
2). Print Configuration Information
3). Change Configuration Information
4). Set Configuration to Factory Settings
5). Write Back (Save) Changes
6). Quit Configuration
Choice (1-6):

Enter 3 to change the configuration information. The following menu then displays:

1260-VXI Non-Volatile Configuration Changer

==========

0). Edit Local Register Configuration
1). Edit pSOS Configuration
2). Edit VXI Interrupt Handler Logical Address
3). Edit Resource Manager A24/A32 Assign Bases
4). Edit Servant Area and DC Configuration
5). Edit FAILED Device Handling Mode
6). Edit GPIB Configuration
7). Edit Default CI Configuration
8). Edit Resident CI Base Locations
9). Edit CI User Configuration Variables
Q). Quit Editor
Choice (0-9,Q):

Enter 1 to edit pSOS configuration. The following prompt then appears:

------pSOS Configuration------
Enter Dynamic RAM Region 1 Size (default 0x70000):
Enter <CR> to keep the present value and continue to the next entry:
Enter Maximum Number of Processes (default 0x20):

The following formula calculates the maximum number of processes:

Number of processes = 10h + (number of GPIB address links) + (2* number of CIs)

If fewer than six CIs are installed and no other GPIB address links exist, the default value of 32 (0x20) is adequate. Increasing the number of processes affects the throughput of the 1260-00C. Enter the number of processes in hexadecimal.

The next prompt is displayed:

```
Enter Maximum Number of Exchanges (default 0x20):
```

The following formula calculates the maximum number of exchanges:

Number of exchanges = 10h + (number of CIs)

The default value of 32 (0x20) is adequate even if all 12 CIs are installed. Enter <CR> to select the default value.

The last prompt appears:

```
Enter Maximum Number of Message Buffers (default 0x180):
```

The following formula calculates the maximum number of message buffers:

Number of message buffers = 100h + (25* number of CIs)

If fewer than six CIs are installed, the default value of 384 (180h) is adequate. Increasing the number of message buffers affects the throughput of the 1260-00C. Enter the number of message buffers in hexadecimal.

When the edit menu reappears, enter 8 to edit the resident CI base locations.

For the DMAmove Code Instrument, only one CI should be created. Configure a single CI base location as shown below. For the CDS-852 adapter, configure as many CI base locations as there are 852 adapters to be controlled by the 1260-00C. For example, to control four 73A-852s, configure CI base locations 0 through 3. The addresses for the two CIs are as follows:

Code Instrument	Address (Base Location)
CDS852 CI	F7E000
DMAmove CI	F7C000

Type Y to respond yes to the `Debug mode On for Resident CI 0xXX` prompt. This enables debug statement printing to the terminal.

For example, to install the single DMAmove CI and two 852 adapter CIs and enable debug statement printing on the second 852 CI, enter the following sequence for this example:

----- Resident CI Base Location Configuration -----

Enter Number of Base Location to EDIT (0xff = EXIT): 0
Enter Address for Base 0x01 (default = 0x000000): F7C000
Debug mode ON for Resident CI 0x01 (default NO): N
Enter Number of Base Location to EDIT (0xff = EXIT): 4
Enter Address for Base 0x00 (default = 0x000000): F7E000
Debug mode ON for Resident CI 0x00 (default NO): <CR>
Enter Number of Base Location to EDIT (0xff = EXIT): 5
Enter Address for Base 0x01 (default = 0x000000): F7E000
Debug mode ON for Resident CI 0x01 (default NO): Y
Enter Number of Base Location to EDIT (0xff = EXIT): <CR>

When the edit menu reappears, enter Q to exit the configuration editor. When the Non-Volatile Configuration main menu appears, enter 5 to save the configuration changes. When the Non-Volatile Configuration main menu reappears, enter 2 to confirm the configuration information. The CI configuration for the previous example would be displayed as follows:

------ Resident Code Instrument Locations ------

0x00: 00F7C000	0x01: 00000000	0x02: 00000000
0x03: 00000000	0x04: 00F7E000	0x05: 00F7E000
0x06: 00000000	0x07: 00000000	0x08: 00000000
0x09: 00000000	0x0A: 00000000	0x0B: 00000000

When the main menu reappears, enter 6 to quit the configuration mode. The following message appears:

Must Re-initialize pROBE or reboot for pSOS changes to take effect.
Other changes made automatically when configuration saved.
****
DONE WITH CONFIGURATION
Change Start-up mode Dip settings to enter
different mode or push RESET to reconfigure.
****

Executing An EPROMed Code Instrument

If a CI is configured in non-volatile configuration to be executed, the CI will be booted upon the next power cycle of the 1260-00C. The CI booting procedure actually occurs after the Resource Manager has run and the local command set has been initiated on all ports. This guarantees the CI access to all resources of the 1260-00C.

Deleting a CI

To delete a CI, follow the installation procedure and set the CI's base address location to 000000. To delete the CI during runtime, after the CI has already been started up, use the local command set command, `CIDelete?`. See Section 4 CI Configuration Commands and Queries, for further information on the command `CIDelete?`.

The DMAmove Code Instrument

After the non-volatile configuration is complete and the 1260-00C is rebooted, the DMAmove Code Instrument will be up and running. The following message will be printed on the serial port:

```
Racal-Dana' DMAmove Code Instrument Running
```

The following sections describe the runtime capabilities of the DMAmove Code Instrument.

GPIB Address Assignment

The DMAmove CI is assigned Logical Address 160 by default. If a device already exists at Logical Address 160, the DMAmove CI is assigned the next highest available logical address. The GPIB address is assigned to be the upper five bits of the logical address (GPIB address 20), if available. If that GPIB address is taken, it takes the next highest available GPIB address. Use the local command set command `LaSaddr?` to determine the GPIB address and the local command set command `LaSaddr` to change the GPIB address, and communicate directly with the DMAmove CI through this GPIB address.

Capabilities and Operation

The DMAmove CI is a Code Instrument built on top of the function `DMAmove()`, which is available to all Code Instruments. (This CI is provided in source code with development versions of the 1260-00C.) As such, the DMAmove CI has all of the capabilities of the `DMAmove` function plus a few extra device interface type features. The following is the `C` language prototype for the `DMAmove()` function:

```
DMAmove(source, dest, count, mode)
      uint32 source          Local address to transfer from
      uint32 dest            Local address to transfer to
      uint32 count           Number of bytes to transfer
      uint32 mode            Bit vector for mode of transfer
                        Bit 0: Transfer direction
                              0 = Source to destination
                              1 = Destination to source
                        Bit 1: Destination size
                              0 = 16 bit
                              1 = 8 bit
                        Bit 2: Operand size
                              0 = 16 bit
                              1 = 8 bit

                        Bit 3: DMA transfer mode
                              0 = Cycle steal
                              1 = Burst
                        Bit 4: Source address increment
                              0 = Increment source address by operand size
                              1 = Do not increment source address
                        Bit 5: Destination address increment
                              0 = Increment destination address by destination
                                  size
                              1 = Do not increment destination address
```

The interface for the DMAmove CI is almost identical. To control the DMAmove CI, simply send 16 binary bytes of information over the GPIB with EOI on the last byte corresponding to the four 32-bit parameters in the `DMAmove` function prototype. The only exception to this for the DMAmove CI is the value of zero (0) in the `source` parameter specifies to take data from the GPIB as input and the value of zero (0) in the `dest` parameter specifies the source data be transmitted out the GPIB. Do not specify zero in both `source` and `dest` parameters. When writing data with GPIB as the source, simply follow the 16-byte transfer (which has EOI on the last byte) with the continuous data transfer with EOI on the last byte. When reading data from the 1260-00C out to the GPIB, simply follow the 16-byte transfer (which has EOI on the last byte) with a GPIB read. If both `source` and `dest` are non-zero, the transfer will take place without further action. When either `source` or `dest` is non-zero, it specifies a local 1260-00C address as shown in Figure B-1.

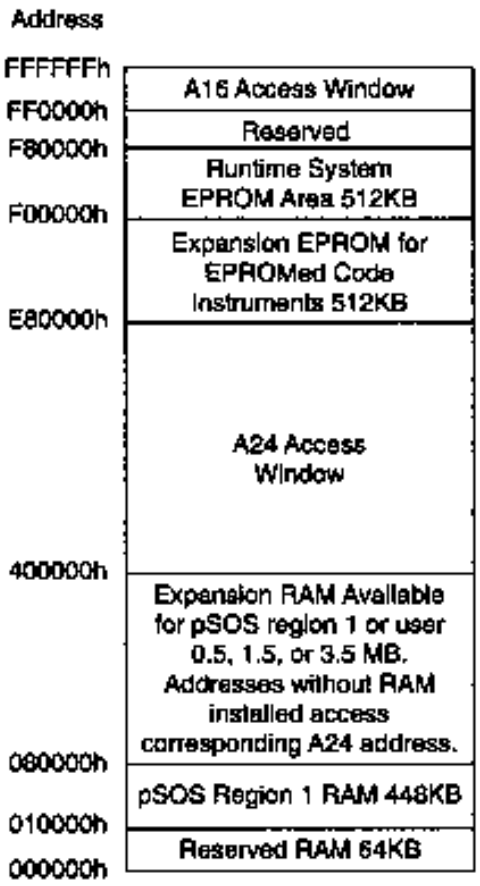


Figure B-1. 1260-00C Local Memory Map

Use the DMAmove CI to perform any of the capabilities of the `DMAmove` function including moving to or from VXI A16 space, VXI A24 space, or local 1260-00C memory. It can transfer 8-bit or 16-bit quantities from either source or destination (they can be different). It can increment or not increment addresses as it counts to give fast access to FIFO registers or block memory.

The DMAmove CI reports current status and errors via its status byte and the REQT signal (GPIB SRQ line). The following is a list of the status bytes returned by the DMAmove CI and their corresponding meanings.

Status Value	Meaning
00h	Idle, no operation pending
80h	Operation underway
41h (RSV pending)	DMA timing error
42h (RSV pending)	Bus Error at source
44h (RSV pending)	Bus Error at destination
48h (RSV pending)	Unknown error
50h (RSV pending)	Error in parameter sent

In addition, if Debug mode is specified when the DMAmove CI is booted, diagnostic messages are printed to the serial port. Every access to the DMAmove CI causes messages to be printed. All accesses to or from the GPIB and VXI are logged to the serial port.

The CDS-852 Adapter Code Instrument

After the non-volatile configuration is complete and the 1260-00C is rebooted, the CDS-852 Adapter Code Instrument will be up and running. The following message will be printed on the serial port:

```
Racal-Dana' CDS 73A-852 Code Instrument Adapter Running
```

The following sections describe the runtime capabilities of the CDS-852 Adapter Code Instrument.

Logical Address and A24 Address Assignment

The 852 adapter CIs are assigned logical addresses sequentially, starting with the lowest configured CI base address and Logical Address 80. For example, if the CIs at base address locations 1 and 3 are installed, the CI at location 1 is assigned Logical Address 80, and the CI at location 3 is assigned Logical Address 81.

The default offset where the CI expects to find its 73A-852 registers in VXI/VME A24 space is related to the CI's logical address as follows:

73A-852 A24 offset = CI logical address * 10000h

For example, a CI at Logical Address 80h expects (by default) to find its 73A-852 registers at offset 800000h. The CI's A24 offset can be changed with the CI command `!!L`. The 73A-852 has rotary switches for changing its A24 register locations.

852 Adapter CI Commands

The 852 adapter CI commands are interpreted by the CI itself, and do not directly affect the 852 module. If the adapter CI receives a word serial buffer that does not begin with the `!!` character sequence, it assumes the buffer is for the 73A-852 and writes the buffer to the appropriate A24 register location. The CI command formats were designed to minimize the possibility of conflict with the command sets of the various plug-in instruments that are compatible with the 852 adapter. Because Racal-Dana has not had the opportunity to study the command sets of all CDS plug-in instruments, keep in mind the possibility of conflict as you develop special applications.

The `!!A` and `!!B` commands set the CI read mode for compatibility with the CDS instrument. Some CDS command responses are in ASCII format, while others are in binary format. Refer to the appropriate CDS manual for the response format of a particular command. The `!!A` command sets the adapter CI mode to be compatible with the ASCII response format. If the expected response format is binary, use the `!!B` command to set the CI to the binary read mode.

Two termination conditions can apply to reading data from the 73A-852. Depending upon the 73A-852 configuration and the operation being performed, the 852 may terminate the transmission of data with an end-of-string (EOS) character, or it may assert the END bit (VMEbus data bit 8).

The `!!E, !!T,` and `!!t` commands configure the CI termination conditions. The EOS and END bit termination conditions are independent; that is, you can configure the CI to terminate a read from the 73A-852 on one condition, both conditions, or neither condition. In addition, you can limit the maximum size of binary mode reads with the `!!S` command. Note that with binary responses, there can be no unique EOS character, so use the END or transfer size conditions (not EOS) to terminate binary transfers. The default read mode is ASCII. The default read termination condition is the EOS character <LF> (0Ah) with the END bit set.

The `!!Q` command returns information about the CI settings (always to the serial port). The `!!D` and `!!d` commands control the printing of runtime debug information to the terminal connected to the serial port.

!!A

Purpose: Set the adapter CI read mode to ASCII, for compatibility with the CDS instrument response.

Command

Syntax: `!!A`

or

`!!a`

Action: Sets the adapter CI read mode to ASCII. The maximum ASCII response size allowed is 512 bytes.

!!B

Purpose: Set the adapter CI read mode to binary.

Command

Syntax: !!B

or

!!b

Action: Sets the adapter CI read mode to binary. Read size is limited to 512 bytes, or as configured by the !!S command.

!!D

Purpose: Enable debug message printing to the serial port.

Command

Syntax: !!D

Action: Enables debug message printing to the serial port.

!!d

Purpose: Disable debug message printing to the serial port.

Command

Syntax: !!d

Action: Disables debug message printing to the serial port.

!!E

Purpose: Configure CI read termination on an EOS character.

Command

Syntax: !!E <hex number>

or

!!e <hex number>

<hex number> is the ASCII value corresponding to the desired EOS character (for example, 0Ah for <LF>).

Action: Enables read termination on an EOS with a value of <hex number>. If <hex number> is greater than FFh, the EOS termination condition is disabled.

Examples: Set the EOS character to <CR>

!!E 0D

Disable EOS read termination.

!!E 100

!!L

Purpose: Set the A24 base address where the adapter CI expects to find the 852 adapter.

Command

Syntax: !!L <val>

or

!!l <val>

<val> is a hex value equal to the upper 8 bits of the target adapter's A24 address.

Action: The adapter CI expects to find the target 852 adapter at offset <val> * 10000h. The default (initial) value of <val> is the adapter DCI's logical address.

Example: Set the adapter CI to operate with a 852 adapter at A24 base address 830000h.

!!L 83

!!S

Purpose: Set the maximum size of a binary read.

Command

Syntax: !!S <size>

or

!!s <size>

<size> is a decimal value.

Action: Sets the maximum binary read size to <size> bytes. The default value of the read size is 512 bytes.

!!T

Purpose: Enable read termination when the END bit is set.

Command

Syntax: !!T

Action: Enables read termination when the END bit (bit 8) is set.

!!t

Purpose: Disable read termination on the END bit.

Command

Syntax: !!t

Action: Disables read termination on the END bit (bit 8).

Appendix C
SPECIFICATIONS

Introduction

This appendix lists various module specifications of the 1260-00C, such as physical dimensions and power requirements.

Specifications

CPU

Microprocessor	16-MHz 68070
Co-processor (optional)	16-MHz 68881
RAM	4MB (configured to use 512kB

Physical

C-size VXIbus Board

Slot Requirements	1 slot
Local Bus Keying	Class 1, TTL

Front Panel Indicators

- SYSFAIL (red)
- FAILED (red)
- TEST (green)
- ON LINE (green)
- ACCESS (yellow)

Front Panel Connectors

- RS-232
- GPIB
- Trigger Input
- Trigger Output
- External 10-MHz Clock

Reset pushbutton

Power Requirements

Source	Typical	Direct Current (max)	Dynamic Current (max)
+5.0 VDC	2.5 A	4.0 A	Not Available
+12.0 VDC	12.0 mA	15.0 mA	Not Available
-12.0 VCI	12.0 mA	15.0 mA	Not Available
-5.2 VDC	100.0 mA	200.0 mA	Not Available
-2.0 VDC	50.0 mA	100.0 mA	Not Available

Cooling Requirements

Power Dissipation, max 22 W

Operating Environment

Temperature 0° to 55° C

Relative Humidity 0% to 95% non-condensing

Storage Environment

Temperature -40° to 125° C

Relative Humidity 0% to 100% non-condensing

EMI

FCC Class A verified

Functionality

IEEE-488

Capability Code	Description
SH1	Source Handshake
AH1	Acceptor Handshake
T5, TE5	Talker, Extended Talker
L3, LE3	Listener, Extended Listener
SR1	Service Request
DC1	Device Clear
DT1	Device Trigger
RL0	Remote Local
PP0	Parallel Poll

VXIbus Master/Slave

- A16/A24 Addressing
- A32 Addressing (slave only)
- D08(EO)/D16 Data Paths
- Read-Modify-Write

VXIbus

- VXIbus System Specification Compatible
- Multi-Mainframe Resource Manager (defeatable)
- Slot 0 Support (defeatable)
- Message-Based Commander and Servant
- Dynamically Configurable
- Programmable Handler (any three of seven levels)
- Trigger Source/Acceptor (SYNC, SEMI-SYNC, ASYNC, STST protocols)
- External Trigger I/O Support
- External CLK10 I/O Support

Appendix D

CONNECTORS

Introduction

This appendix describes the connectors found on the 1260-00C.

NOTE

The illustrations in this appendix show the mating face of the connectors. An asterisk suffix (*) on a signal name indicates the signal is active low.

RS-232 Connector

Connector Type:

9-pin Subminiature D HD-20

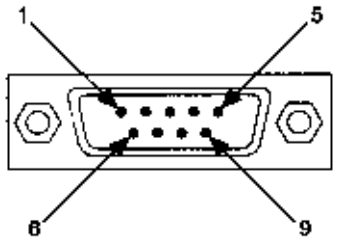


Figure D-1, RS-232 Connector

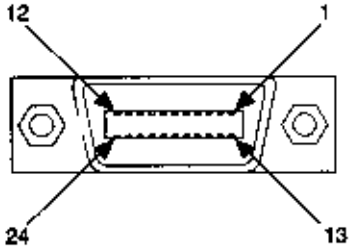
Table D-1, RS-232 Connector Signals

Pin	Signal Name	Signal Description
1	DFI1	Discrete Fault Indicator (leave unconnected)
2	RXD*	Receive Data
3	TXD*	Transmit Data
4	DTR*	Data Terminal Ready
5	GND	Ground
6	DFI2	Discrete Fault Indicator (leave unconnected)
7	RTS*	Ready to Send
8	CTS*	Clear to Send
9	n.c.	Not Connected

WARNING

When building a cable for the RS-232 port, do not connect to pins 1 and 6. Connecting to these pins can result in damage to the 1260-00C.

GPIB



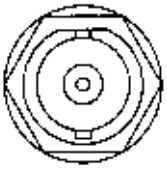
Connector Type: GPIB

Figure D-2, GPIB Connector

Table D-2, GPIB Connector Signals

Pin	Signal Name	Signal Description
1	DIO1*	Data Bit 1
2	DIO2*	Data Bit 2
3	DIO3*	Data Bit 3
4	DIO4*	Data Bit 4
5	EOI*	End or Identify
6	DAV*	Data Valid
7	NRFD*	Not Ready For Data
8	NDAC*	Not Data Accepted
9	IFC*	Interface Clear
10	SRQ*	Service Request
11	ATN*	Attention
12	SHIELD	Chassis Ground
13	DIO5*	Data Bit 5
14	DIO6*	Data Bit 6
15	DIO7*	Data Bit 7
16	DIO8*	Data Bit 8
17	REN*	Remote Enable
18	GND	Logic Ground
19	GND	Logic Ground
20	GND	Logic Ground
21	GND	Logic Ground
22	GND	Logic Ground
23	GND	Logic Ground
24	GND	Logic Ground

External CLK10



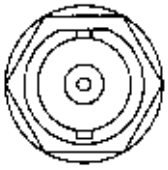
Connector Type: BNC

Figure D-3, EXT CLK Connector

Table D-3, EXT CLK Connector Signals

Pin	Signal Description
Center	CLK10 I/O (TTL, 10 MHz)
Shield	Ground

Trigger Input



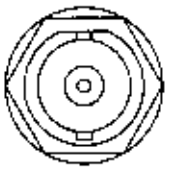
Connector Type: BNC

Figure D-4, TRG IN Connector

Table D-4, TRG IN Connector Signals

Pin	Signal Description
Center	Trigger Input (TTL)
Shield	Ground

Trigger Output



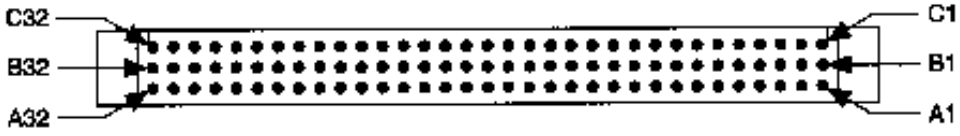
Connector Type: BNC

Figure D-5, TRG OUT Connector

Table D-5, TRG OUT Connector Signals

Pin	Signal Description
Center	Trigger Input (TTL, 50Ω driver)
Shield	Ground

VXIbus P1 and P2



Connector Type: 96-pin DIN

Figure D-6, VXIbus Connector

Table D-6, VXIbus P1 Connector Signals

Pin	Row A Signals	Row B Signals	Row C Signals
1	D00	BBSY*	D08
2	D01	BCLR*	D09
3	D02	Not Connected	D10
4	D03	BG0IN*	D11
5	D04	BG0OUT*	D12
6	D05	BG1IN*	D13
7	D06	BG1OUT*	D14
8	D07	BG2IN*	D15
9	GND	BG2OUT*	GND
10	SYSCLK	BG3IN*	SYSFAIL*
11	GND	BG3OUT*	BERR*
12	DS1*	BR0*	SYSRESET*
13	DS0*	BR1*	LWORD*
14	WRITE*	BR2*	AM5
15	GND	BR3*	A23
16	DTACK*	AM0	A22
17	GND	AM1	A21
18	AS*	AM2	A20
19	GND	AM3	A19
20	IACK*	GND	A18
21	IACKIN*	Not Connected	A17
22	IACKOUT*	Not Connected	A16
23	AM4	GND	A15
24	A07	IRQ7*	A14
25	A06	IRQ6*	A13
26	A05	IRQ5*	A12
27	A04	IRQ4*	A11
28	A03	IRQ3*	A10
29	A02	IRQ2*	A09
30	A01	IRQ1*	A08
31	-12 V	Not Connected	+12 V
32	+5 V	+5 V	+5 V

Table D-7, VXIbus P2 Connector Signals

Pin	Row A Signals	Row B Signals	Row C Signals
1	ECLTRG0	+5 V	CLK10+
2	-2 V	GND	CLK10-
3	ECLTRG1	Not Connected	GND
4	GND	A24	-5.2 V
5	MODID12	A25	LBUSC00
6	MODID11	A26	LBUSC01
7	-5.2 V	A27	GND
8	MODID10	A28	LBUSC02
9	MODID09	A29	LBUSC03
10	GND	A30	GND
11	MODID08	A31	LBUSC04
12	MODID07	GND	LBUSC05
13	-5.2 V	+5 V	-2 V
14	MODID06	Not Connected	LBUSC06
15	MODID05	Not Connected	LBUSC07
16	GND	Not Connected	GND
17	MODID04	Not Connected	LBUSC08
18	MODID03	Not Connected	LBUSC09
19	-5.2 V	Not Connected	-5.2 V
20	MODID02	Not Connected	LBUSC10
21	MODID01	Not Connected	LBUSC11
22	GND	GND	GND
23	TTLTRG0*	Not Connected	TTLTRG1*
24	TTLTRG2*	Not Connected	TTLTRG3*
25	+5 V	Not Connected	GND
26	TTLTRG4*	Not Connected	TTLTRG5*
27	TTLTRG6*	Not Connected	TTLTRG7*
28	GND	Not connected	GND
29	Not Connected	Not Connected	Not Connected
30	MODID00	Not Connected	GND
31	GND	GND	+24 V
32	SUMBUS	+5 V	-24 V

ERROR CODES

Introduction

This appendix lists the local command set error codes, and describes the error associated with each error code.

Table E-1. Error Codes

Error Number	Type	Description
0	Format	Command format error
1	Syntax	Command was not found
2	Syntax	Illegal identifier after <Program Data Separator>
3	Syntax	Missing <Program Data Separator>
4	Syntax	Maximum <Program Mnemonic> length is 12 characters
5	Syntax	Illegal command: Expecting upper or lower case alpha
6	Syntax	Illegal command
7	Syntax	Illegal non-numeric
8	Syntax	Illegal <Decimal Numeric Program Data>
9	Syntax	Illegal <Suffix Program Data>
10	Syntax	Maximum <Suffix Program Data> length is 12 characters
11	Syntax	Illegal <String Program Data>
12	Syntax	Illegal <Arbitrary Block Program Data>
13	Syntax	Illegal <Expression Program Data>
14	Syntax	Illegal <Character Program Data>
15	Syntax	Illegal character on input
16	Syntax	Illegal identifier after command
17	Syntax	Illegal identifier after <Program Separator>
18	Syntax	Missing <Program Separator>
19	Syntax	Too much data
30	Device	No error
31	Device	Logical address is out of range 0 through 254
32	Device	No device is at that logical address
33	Device	GPIB secondary address is out of range 0 through 30
34	Device	VXI interrupt handler number is out of range 1 through 3
35	Device	VXI interrupt level is out of range 0 through 7
36	Device	A16 address is out of range 0000h through FFFEh

Table E-1. Error Codes (continued)

Error Number	Type	Description
37	Device	Address must be even
38	Device	Word write value is out of range 0000h through FFFFh
39	Device	A bus error occurred during the access
40	Device	A24 address is out of range 200000h through E7FFFEh
41	Device	488.2 register is out of range 0 through 255
42	Device	Console mode is disabled: must have one output mode enabled
43	Device	Logical device has no secondary address link
44	Device	Unable to delete secondary address link
45	Device	Unable to create secondary address link
46	Device	Secondary address is already attached to a logical address
47	Device	Device is not a Message-Based device
48	Device	Device is not a servant of this 1260-00C
49	Device	Device does not have commander capability
50	Device	Not Dynamically Configured
51	Device	Commander did not accept *Device Grant* command
52	Device	Servant did not accept BNO or *Identify Commander* command
53	Device	Logical address cannot be this 1260-00C
54	Device	Word Serial command is out of range 0 through FFFFh
55	Device	Logical address is not physical VXI device
56	Device	Unable to create I/O buffer
57	Device	Commander did not accept *Release Device* command
58	Device	Unable to grant CI to physical device
59	Device	Device is already a servant
60	Device	Device is not commander of servant
61	Device	Register offset out of range 0 through 3Eh
62	Device	Timeout waiting for Downloaded Data
63	Device	TTL/ECL trigger line out of range 0 through 9
100	CI	DCI functionality is inactive
101	CI	Logical address conflict
102	CI	Logical address is out of range
103	CI	Block(s) requested are used
104	CI	Block(s) requested do not exist
105	CI	Servant(s) requested do not exist
106	CI	Servant(s) requested are not servants of the 1260-00C or anotherDCI
107	CI	Commander requested does not exist

Table E-1. Error Codes (continued)

Error Number	Type	Description
108	CI	Servant(s) requested do not have the same commander
109	CI	Requested zero blocks
110	CI	Logical address referenced is not a DCI
111	CI	DCI base address is out of range
112	CI	DCI area new base address is not a multiple of 4096
113	CI	DCI area request exceeds available memory
114	CI	Attempted to change DCI area while DCIs were installed
116	CI	DCI was not found
117	CI	Logical address referenced is not a DCI
120	CI	Error encountered while spawning DCI process(es)
121	CI	Error encountered while creating Asynch process exchange
122	CI	No DCI initialization information (need to do `DCISetup?` query)
123	CI	Download timed out
125	CI	Download overflowed requested blocks
126	CI	Servant(s) requested has secondary address link
127	CI	Memory requested for DCI Word Serial structures is unavailable
128	CI	Logical address referenced is not the 1260-00C or local DCI
129	CI	Logical address referenced is not 1260-00C's or CI's servant
130	CI	Stack size requested for worker process exceeds FFFFh words
301	Trigger	No Trigger hardware support for this operation
302	Trigger	Invalid Controller for trigger functions
303	Trigger	Invalid Trigger line, External line, or protocol
304	Trigger	Trigger line not supported
305	Trigger	Trigger protocol not supported
306	Trigger	Wait period exceeded, timeout occurred
307	Trigger	Line already configured, must unconfigure to configure
308	Trigger	Source line not supported
309	Trigger	Destination line not supported for this source
310	Trigger	Invalid configuration mode
311	Trigger	Line already mapped, must unmap to map
312	Trigger	Line has not been configured/mapped for this operation
313	Trigger	Invalid count (TICK = 0 through 32, CNTR = 0 through 65535)
314	Trigger	Invalid/Unsupported mapping signal conditioning mode

Table E-1. Error Codes (continued)

Error Number	Type	Description
315	Trigger	Previous operation is still pending for this line
316	Trigger	Previous acknowledge is still pending for this line
33064	Trigger	Trigger overrun, too many triggers received
33068	Trigger	Trigger unassertion overrun, too many triggers received
33076	Trigger	Trigger pulse stretch overrun, too many triggers received

Glossary

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

Numbers/Symbols

°	degrees
Ω	ohms
%	percent

A

A	ampere

B

Backplane	An assembly, typically a printed circuit board, with 96 pin connectors and signal paths that bus the connector pins. VXIbus systems have either two sets of bused connectors, designated J1 and J2 backplanes, or three sets of bused connectors, designated J 1, J2, and J3 backplanes.
BTO	Bus Timeout Unit
bytes/sec	bytes per second

C

C	Celsius

CI	*See* Code Instrument.
Code Instrument	CI; a proprietary National Instruments software structure that uses software to emulate the capabilities of a Vx' Message-Based device.
Command	Causes the GPIB-VXI/C to take some action.
Commander	A Message-Based device that is also a bus master and can control one or more Servants.
Console response	Returned in the form of readable sentences, which is better suited for interactive command entry.

D

DC	Dynamic Configuration, or Dynamically Configured
DC device	*See* Dynamic configuration device.
DCI	*See* Downloaded CI.
Diagnostics mode	Mode in which you can perform extensive offline diagnostic tests of the GPIB-VXI/C.
Downloaded CI	DCI; a form of CI that is downloaded into the (3PIB-VXIIC's, RAM memory.
Dynamic configuration device	DC device; a device that initially has a logical address of *255.* The RM subsequently assigns it a different, unique logical address.

E

ECI	*See* EPROMed CI.
EEPROM	Electrically Eraseable Programmable Read Only Memory
EPROM	Eraseable Programmable Read Only Memory
EPROMed CI	ECI; a form of CI that is user-installed into EPROMs.

F

FIFO	First-In First-Out

G

GPIB	General Purpose Interface Bus. The industry standard IEEE 488 bus.
GPIB-VXI/C local	Consists of commands and queries. command set

H

Hz	Hertz (cycles per second)

I

IEEE	Institute of Electrical and Electronic Engineers
in.	inches

L

Logical address	An 8-bit number that uniquely identifies each Vxlbus device in a system. It defines a device's A16 register address and indicates Commander/Servant relationships.
LSB	Least Significant Bit

M

m	meter
MB	Megabytes of memory
Message-Based device	An intelligent device that implements the defined Vxlbus registers and communication protocols.
MODID lines	VXI backplane signals used by the Resource Manager (through the use of the Slot 0 device) in order to perform slot associations for logical addresses. There are 13 MODID lines, one for each slot in a full-size mainframe.
Module	Typically consists of a board assembly and its associated mechanical parts, front panel, optional shields, and so on. A module contains everything required to occupy a slot in a mainframe. A module can occupy one or more slots.
MSB	Most Significant Bit

N

Nonvolatile configuration mode	Mode in which you can edit the contents of the nonvolatile EEPROM memory.
NV	Nonvolatile memory

P

Peek	To read the contents.
PH	Programmable Handler
P1	Position Independent
Poke	To write a value.
pROBE	A low-level interactive debugger for use with the pSOS operating system. It is commercially available from Software Components Group, Inc. VXI pROBE is an enhanced version of pROBE supplied on developmental versions of the GPIB-VXI/C.
Program mode response	Has a terse data-only format that is intended for a control program to read and parse.
pSOS	A small, multitasking operating system kernel used on the GPIB-VXI/C. It is commercially available from Software Components Group, Inc.

Q

Query	Similar to a command in that it also causes the GPIB-VXI/C to take some action, but it always returns a response containing data or other information.

R

RCI	*See* Resident CI.
Register-Based Device	A Servant-only device that supports Vxlbus configuration registers. Register-Based devices are typically controlled by Message-Based devices via device-dependent register reads and writes.
Resident CI	RCI; a CI that is supplied by National Instrument and resides in the firmware.
RM	*See* Resource Manager.

Resource Manager	A Message-Based Commander located at Logical Address 0 that provides configuration management services such as address map configuration, Commander/Servant mappings, self-test and diagnostic management.

S

SC	Static Configuration, or Statically Configured
SC Device	See Static configuration device.
sec	seconds
Servant	A device that is controlled by a Commander. Any device can be a Servant.
Static configuration device	SC device; a device that has its logical address set by static means, such as by a DIP switch.
System configuration table	During the execution of the RM and general configuration operations, the GPIB-VXI/C builds up a table of system configuration information. Each device has an entry in the table containing the device's logical address, its Commander's logical address, its secondary address, slot number, device class, manufacturer ID number, model code, memory space requirement, memory base address, and memory size. This table remains after the RM and general configuration operations are complete. It is accessible through the GPIB-VXI/C local command set.

V

V	volts
VME	Versa Module Eurocard or IEEE 1014.
VXIbus	VMEbus Extensions for Instrumentation.
VXI pROBE mode	Mode in which you can use the enhanced pROBE debugger. This mode is available only with the GPIB-VXI/C development firmware option.
VXI system mode	The startup mode for normal operation in a VXI system.

W

W	watts

Word Serial communication	The simplest form of communication required by Message-Based devices. It utilizes the A 16 communication registers to transfer data using a simple polling handshake method.
Word Serial Protocol	The rules and regulations involved in performing Word Serial communication.