NATIONAL INSTRUMENTS™
The Software is the Instrument®

Release Notes

LabVIEW® for Macintosh

Version 4.1

Welcome to LabVIEW for Macintosh

These release notes introduce you to the contents of LabVIEW for Macintosh, describe the system requirements for the LabVIEW software, and contain installation instructions and updated documentation information.

Contents

How to Proceed 1
Required System Configuration 2
Installing LabVIEW 3
Where to Go from Here 4
Data Acquisition and GPIB Installation Notes 5
Manual Clarifications and Additions 5

How to Proceed

If you are upgrading from a previous version of LabVIEW, carefully read the *LabVIEW Upgrade Notes* that are included with your upgrade package. You should read the upgrade notes before continuing with this installation because there are several things you should consider before converting your VIs to this version of LabVIEW.

Scan the *Required System Configuration* section and then follow the instructions in the *Installing LabVIEW* section of these release notes. If you use data acquisition (DAQ) or GPIB products, read the *Data*

Acquisition and GPIB Installation Notes section. In addition, you should read the *Manual Clarifications and Additions* section before using LabVIEW 4.1.

Required System Configuration

LabVIEW for Macintosh is available either as a CD containing both 680x0 and Power Macintosh software, or as a set of floppies for the 680x0 Macintosh. The Power Macintosh software is only available on the CD version. In addition to the LabVIEW application, the CD also contains an Instrument Drivers directory that includes all the LabVIEW instrument drivers available when LabVIEW 4.1 was released.

The floppy disk version of LabVIEW for the 680x0 based Macintosh is available as a Full Development System, which contains a set of disks for installing LabVIEW and the associated VIs. In addition, you also can install the VXI VI Library.

LabVIEW for the 680x0-based Macintosh requires a math coprocessor. You cannot use a 68000-based computer, such as the Macintosh SE, with LabVIEW because the 68000 does not use a coprocessor. If you are using a Macintosh IIsi or a Macintosh LC, you may not have a coprocessor in your computer. Also, computers using the 680LC40 processor (Centris 610s and some low-end Centris 650s) do not have the math coprocessor functions, and thus cannot run LabVIEW.

LabVIEW requires a minimum of 6 MB of available RAM, in addition to the RAM requirements for your system software and any other applications that you want to run simultaneously. We strongly recommend you have at least 8 MB of RAM. You may need more memory, depending on the size of the application you design in LabVIEW and the amount of data that your application manipulates.

LabVIEW requires System 7 and will not run under System 6 or any previous versions of the Macintosh operating system.

For more accurate timing you should install the Apple QuickTime extension. When you use QuickTime, timing accuracy should increase from 162/3 ms resolution to approximately 1 ms resolution. System response varies depending on background applications, other extensions, networking activity, and disk caching.

☞ **Note:** ***You need approximately 50 MB of disk storage space for a minimal installation of LabVIEW and 70 MB for a full installation.***

Installing LabVIEW

If you are upgrading from an earlier version of LabVIEW, carefully read the *LabVIEW Upgrade Notes* that are included with your upgrade package. You should read the upgrade notes before continuing with this installation.

Installation Procedure

1. If you are installing from a floppy disk, insert Disk 1 and double-click on the LabVIEW Installer icon. If you are installing from a CD on a 680x0-based Macintosh, double-click on the LabVIEW Installer icon in the 680x0 Macintosh directory. If you are installing from a CD on a Power Macintosh, double-click on the LabVIEW Installer icon in the Power Macintosh directory.
2. The installer gives you the option of performing a full installation or a minimal installation. If you do not have sufficient disk space (approximately 70 MB), you can choose the minimal installation and use your LabVIEW CD to access the remaining components.
3. After you select the **Install** button, you are prompted to select a destination directory. You should use the **New Folder** button to create a LabVIEW directory, and then select the **Install In Current Folder** button to install the LabVIEW files in that directory.
4. ***(Floppy Installation Only)*** If you are installing the full development system from floppy disks, you have additional sets of disks that contain the VXI Library VIs. If you do not need VXI support, you can ignore this set of disks. The first disk of this set contains an installer for these VIs. The installer, VXI Library, is similar to the LabVIEW installer. Insert Disk 1 of the VXI VI disks and double-click on the installer icon.

After you have completely installed LabVIEW, it is ready to run. If you plan to use DAQ or GPIB devices with LabVIEW, you must restart your computer so that the new drivers are loaded.

☞ **Note:** ***The function material from the*** *Code Interface Reference Manual* ***and the*** *VXI VI Reference Manual* ***are available online in Adobe Acrobat format on the LabVIEW CD. For more information on installing and viewing these online manuals, please refer to the*** `manuals\readme.txt` ***on the LabVIEW CD.***

☞ **Note:** ***If you are upgrading from a previous version of LabVIEW, you should read the*** *What Is New in LabVIEW* ***section of the*** *LabVIEW Upgrade Notes. If you have one of the add-on packages, such as the Test Executive Toolkit or the Picture Control Toolkit, you may want to install those files at this time.***

Where to Go from Here

The following resources can help get you up to speed with LabVIEW 4.1 more quickly.

- If you are a new LabVIEW user, work through the *LabVIEW Tutorial*. This manual teaches basic LabVIEW concepts and tasks.
- The `examples` directory contains a VI called `readme.vi`. With this VI, you can look at the available examples. when you select a VI, you can see the documentation that was entered for that VI by choosing **Window»Show VI Info...**. to open a VI, choose **File»Open...**.
- If you are going to perform data acquisition, read the *LabVIEW Data Acquisition Basics Manual*, which contains important information about using the DAQ VIs and examples you can find in LabVIEW. For reference information on particular DAQ VIs, refer to the *LabVIEW Function and VI Reference Manual*. The *What Is New in LabVIEW* section, in the *LabVIEW Upgrade Notes*, also contains information about changes in DAQ VIs. You should read this section even if you have used DAQ VIs in previous releases of LabVIEW, because a number of new features have been added.
- The DAQ Navigator, found in the **Help** menu, is a good way to figure out where to get started in your application. The navigator asks you a series of questions, which direct you to information on your task.
- The DAQ examples folder contains a VI library called `run_me.llb` that has a Getting Started example VI for Analog Input, Analog Output, Digital I/O, and Counters.

The `run_me.llb` examples, DAQ Navigator, and the Getting Started sections in Chapter 2 provide an excellent starting place for data acquisition.

Data Acquisition and GPIB Installation Notes

The LabVIEW installation program installs two control panels and an extension in your system folder. NI-GPIB contains the driver code that communicates with your GPIB devices. NI-DAQ contains driver code that communicates with your DAQ devices. The NI-DMA/DSP extension contains DSP and DMA drivers that are used by DAQ, GPIB, and DSP drivers.

Manual Clarifications and Additions

This section contains information that was not included in the LabVIEW documentation and corrections to the LabVIEW manuals. For information on new features to this particular version of LabVIEW, please read the *LabVIEW Upgrade Notes*.

General Interface Changes

The following changes were made to the LabVIEW interface, but were not explained in the printed or online documentation.

Print Margins

You now can set margins on printouts, both globally for LabVIEW and for specific VIs. Margins can be set in inches or millimeters. To set margins for all LabVIEW printouts, use the Printing page of the Preference dialog box. To set margins for a single VI, use the Execution page of the VI Setup dialog box, which can be accessed from the connector pane of the front panel. Settings for a single VI will override the global setting. Margins can be specified arbitrarily small, but will be limited by the device settings on actual printouts.

Printing VIs

The **Functions»Advanced»Printing** palette contains VIs that you can use to print VIs programmatically. This palette consists of the following VIs:

- `Print Panel.vi`—Use this VI to print a VI front panel in the same format as if you selected **File»Print Window** or you enabled programmatic printing. You can specify whether you want the entire front panel or only the visible part of the front panel.
- `Print Documentation.vi`—Use this to print the components of a VI as though you selected **File»Print Documentation**. You can specify whether the VI should display the Print Documentation dialog box so that you can select the format for the printout. If you choose not to display the dialog box, the printout will use the same settings as the last VI printed or the default settings if you have not printed any VIs.
- `Get Panel Image.vi`—Use this VI to programmatically retrieve a front panel image in a format that you could pass to one of the VIs in the Picture Control Toolkit or write to disk. You can specify whether you want the entire front panel or only the visible part of the front panel. You can also specify the color depth for the data. The VI returns arrays of data and color table information in the rectangle describing the image.

Support for Template VIs and Controls

You can save commonly used VIs and controls as templates. To create a template VI, save a VI with a `.vit` extension (or `.ctt` extension for typedefs). You can also use the **Stationery Pad** checkbox in the Get Info dialog box in the Finder to change a VI to a template. When you open a template VI or control, the new file you create is named automatically using your template name and a number corresponding to the number of times it has been opened. When you finish editing the VI and try to save it, LabVIEW prompts you to enter a new name for the file.

If you want to modify a template, you should open it, make your changes, and then save over the `.vit` (or `.ctt`) file that you originally created.

Changes to MPW CIN Tools

The MPW script `cinmake` (“:`contools:MPW:cinmake`”) has been modified to use the MrC compiler by default to compile the CIN for the PowerPC platform. A new command line option (“`-C`”) has been added to allow you to choose the older C compiler for the 68K platform.

Load Memory Usage Option *(PowerMac Only)*

A new preference has been added that allows you to control whether labVIEW should compact memory as it loads VIs. It can be set in the Performance and Disk page of the Preferences dialog box. By default, this preference is on, and the preference is presented as **Less memory fragmentation/Slower loading**.

This feature was added because the Modern Memory Manager is faster for most operations, but allows memory to become fragmented more easily.

Compacting memory while loading VIs reduces memory fragmentation but slows down the loading of VIs. Note that switching the preference to **Faster loading/Increased memory fragmentation** can cause your application to run out of memory sooner than it would otherwise, but can dramatically speed up loading for some applications, especially those with many re-entrant VIs.

Better Support for Toolkits

Files that are installed in `vi.lib\addons` will automatically show up at the top level of the **Control** and **Functions** palettes. This feature can be used by new toolkits to make them more accessible after installation. If you already have toolkits that installed files elsewhere, you can move them to the `addons` directory for easier access. If you want to add your own VIs to the palettes, we recommend placing them in `user.lib` or adding them to a custom palette set.

Debugging and Modal VIs

When LabVIEW stops a VI from executing (typically, when the user clicks on the pause button or because the program encounters a breakpoint) all VIs whose window style is “Dialog” temporarily change their window style to be non-modal. When all of the VIs have finished executing, LabVIEW returns the window style to modal.

VI Control Changes

The Call Instrument VI, which is located in **Advanced**»**VI Control**, has had several enhancements to allow it to perform more error checking and to execute calls more quickly.

- The flattened data types component of the **requested outputs** input was previously ignored. The Call Instrument VI now checks the requested types against the types of the subVI that you are calling and returns an error if they are incompatible.
- In previous versions of LabVIEW, if you did not specify any specific outputs for the Call Instrument VI, all outputs were returned. This feature did not allow for any type checking and often introduced runtime errors if you tried to unflatten the wrong data. Secondly, returning all of the outputs of a VI consumes considerable system time and memory, especially with VIs that contain large numbers of indicators. The Call Instrument VI defaults to returning only the outputs that you request. If you do not request any outputs, none are returned. If you want LabVIEW to return all VI outputs, wire a `True` Boolean to the **Return All Outputs** input of the VI.
- If you do not specify a path, the Call Instrument VI does not attempt to load and release the VI, which significantly increases execution speed. In this case, you should preload the VI yourself using the Preload Instrument VI, located in **Advanced**»**VI Control**, before calling the Call Instrument VI.
- You can use the new Get Panel Size VI to find out the size and location of a VI's front panel. The VI must be in memory, but its front panel does not have to be open. One way you might use this is in combination with the Resize Panel VI to position a front panel before calling the Open Panel VI to display it.
- Both the Get Panel Size and Resize Panel VIs have a Boolean input, **panel bounds**, that lets you specify whether you want the bounds to reflect the size of the front panel or the window. The size of the front panel only includes the visible part of the panel and does not include the window's title bar, the scrollbars, the toolbar, or the menu bar.
- The Open Panel VI has a new input that lets you specify whether you want it to reflect VI Setup settings (for instance, modality, hidden components, and so on). This input defaults to `True`. You might set it to `False` if you are opening a VI that you do not plan to immediately run, so that you can edit the VI or look at its block diagram.

Alternate Palette Menu Views

The *LabVIEW User Manual* and *LabVIEW Tutorial* describe the default **Controls** and **Functions** palettes. In addition to these palettes, there are three alternate, customized views that tailor the setup for users of LabVIEW for data acquisition, test and measurement, and basic use. These views are called the DAQ, T & M, and Basic views. If you are interested in customizing LabVIEW to one of these views, see Chapter 8, *Customizing Your LabVIEW Environment*, in the *LabVIEW User Manual* for more information.

These alternate views contain the same functions as the default view, but the functions are rearranged so that the options that you use more frequently are more accessible. These views also contain some customized controls, which were created using the Control Editor, that are set up to minimize the amount of configuration you have to do yourself for many applications. You can switch between views quickly by choosing **Edit»Select Palette Set**.

DAQ View

The Data Acquisition (DAQ) view emphasizes the DAQ functions by moving them out of the DAQ submenu to the top level of the **Functions** palette and rearranging the DAQ VIs within those palettes. For example, the Signal Conditioning VIs are listed as a submenu from the **Analog Input** palette. Also, the **Charts and Graphs** control palette has custom controls that are set up with standard settings that you might use for DAQ applications.

T & M View

The Test and Measurement (T & M) view emphasizes functions used in test and measurement. The VISA functions appear at the top level of the **Functions** palette, with the GPIB and Serial functions listed below that palette. The **Controls** palette contains a submenu of controls that are frequently used in creating instrument drivers, which also allows you to create instrument drivers more consistently.

Basic View

The new basic control and functions palette view provides a simplified set of palettes, including only the most commonly used functions. This palette can be useful for those who are learning LabVIEW, because it emphasizes only the functions you might need for beginner

applications. To switch to the Basic view, select **Edit»Select Palette Set»Basic**.

LabVIEW Preferences

The **Miscellaneous** page in **Edit**»**Preferences...** has an option for creating constants using a pop-up menu that lets you specify whether the constant name should be visible.

There are preferences that are not a part of the **Preferences...** dialog box. If you want to enable one of these preferences, you can edit your LabVIEW Preferences file to add the option.

- On Windows and the Macintosh, the **Function** palette is hidden when a front panel is active window and **Control** palette is hidden when a block diagram is the active window. If you prefer to have all palettes visible at all times, edit your LabVIEW Preferences file to add a line that sets the `showAllPalettes` preference to `True`.
- By default, LabVIEW looks for the `menus` directory inside of your library directory (the directory that also contains `vi.lib`, `instr.lib`, and `user.lib`). The `menus` directory contains folders of files describing the **Control** and **Function** palette views that are available to you. If you are running off of a network, you may want to define individual `menus` directories for each user. You can add a line to your preference file that sets the `menusDir` preference to an alternative path, one that would be unique for each user's preference file.

Selecting New or Existing Files or Folders with the Macintosh Native File Dialog

Most of the select modes available in the **File Dialog** function map directly into behavior supported by the Macintosh standard file dialogs. However, the Macintosh standard file dialogs do not have a mode where you can select an existing file or type in the name of a new file; the closest thing in the standard dialogs is the **Save** dialog, which automatically tries to "gray-out" files so that you cannot click on them.

The Macintosh standard file dialogs have additional functionality to support select modes `2` (existing or new file) and `5` (existing or new folder). In both modes, you select an existing file or folder just as you would if getting (targeting existing files or folders) a file or folder, by selecting the desired file or folder from the file list box.

To select a new file, press the New... button. This brings up the dialog used to create a new folder or VI library or to specify a new file. Enter the name of the new file in the file name box. Then, press the File button to select the new file.

To select a new folder, press the New... button. This brings up the dialog used to create a new folder or VI Library or to specify a new file. Enter the name of the new folder. Then, press the Folder button to create the new folder. LabVIEW returns you to the file dialog, which will be displaying the contents of the new folder. Press the **Select "<name of new folder>"** button in the file dialog to select the new folder.

Macintosh Alias Resolution

LabVIEW previously automatically resolved aliases in several cases (file dialog, paths, and so on). Due to some problems that this alias resolution feature introduced, this feature has the following limitations.

1. Referencing the alias in a file operation will no longer resolve the alias and automatically remount volumes that had timed out and become unmounted. However, you can select **Function**»**Communication**»**AppleEvent** and choose the AESendFinder Open VI to open the alias.
2. Clicking on an alias to an unmounted volume in the LabVIEW file dialog no longer brings up the authentication dialog box. The alias will not be followed. You should go to the Finder and let the Finder open the alias and mount the volume for you.

Adding VIs to the Project and Help Menus

You can now add VIs to the **Project** and **Help** menus by placing them inside of the project or help directories in the LabVIEW directory. One way you could use this would be to provide quick access to VIs that act as tools in your system. National Instruments uses this feature to make the DAQ Navigator and the Tech Support VIs accessible from the **Help** menu.

Any VI placed at the top level of the project or help directory will be directly appended to the corresponding menu. If you create a subdirectory, a submenu will be appended. If you place a library (LLB) inside of one of these directories, only VIs that are marked as top level will be appended to the menu bar.

DAQ Navigator Tool

The DAQ Navigator tool displays a series of dialogs that ask you questions about the kind of DAQ application you want to create. Based on these questions, the tool highlights and opens useful examples that illustrate how you might write your application, and it also displays references in the manual that you will find useful for further research.

Applibs and Toolkit Users

National Instruments plans to upgrade all of our toolkits for LabVIEW 4.1. In most cases, any existing toolkits will work without problems with LabVIEW 4.1. Version 4.1 is compatible with all toolkits designed for Version 4.0 with the following exception.

LabVIEW Application Builder Libraries

If you have the LabVIEW Application Builder Libraries and want to use LabVIEW 4.1, you must upgrade your Application Builder Libraries as well. The upgrade is free to existing users. Contact your local National Instruments sales office for availability.

Adding Options to the Controls and Functions Menus

Many toolkits such as the Picture Control Toolkit, the SPC toolkit, and the PID toolkit add new VIs and controls to the **Control** and **Function** palettes. In previous versions, toolkits installed VIs in `vi.lib`. National Instruments discourages placing VIs in `vi.lib` anymore, because you can overwrite your files when you install new versions of VIs as you upgrade your software. Instead, toolkit VIs should be placed in `vi.lib\addons`, as described earlier. VIs of your own should be placed in `user.lib` or `instr.lib`, where they will automatically appear in the user libraries or instrument drivers submenus of the **Functions** menu. Or, you can use the **Edit»Edit Control and Function Palettes** menu option to add them anywhere to the menus.

For more information on customizing the **Control** and **Function** palettes, refer to Chapter 8, *Customizing Your LabVIEW Environment*, of the *LabVIEW User Manual*.

LabVIEW User Manual Corrections

In Chapter 11, *Boolean Controls and Indicators*, in the *LabVIEW User Manual*, the manual states:

> You can also remove the Boolean text from both states by selecting **Hide Boolean Text** from the pop-up menu.

This sentence should read as the following:

> You can also remove the Boolean text from both states by toggling the **Boolean Text** option on the **Show** submenu.

In Chapter 11, *Boolean Controls and Indicators*, in the *LabVIEW User Manual*, the manual states:

> If you want to change the font of either the name label or the Boolean text without changing both, select what you want to change with the Labeling tool, and then use the **Text** menu options to make the changes you want.

This sentence should read as the following:

> If you want to change the font of either the name label or the Boolean text without changing both, select what you want to change with the Labeling tool, and then use the **Font** ring options to make the changes you want.

The *Documenting VIs with the Get Info... Option* section of Chapter 7, *Printing and Documentation*, in the *LabVIEW User Manual* should be replaced with the following section.

Documenting VIs with the Show VI Info… Option

Selecting **Windows»Show VI Info...** displays the information dialog box for the current VI. You can use the information dialog box to perform the following functions.

- Enter a description of the VI. The description window has a scrollbar so you can edit or view lengthy descriptions.
- Lock or unlock the VI. You can execute but not edit a locked VI.
- See the current revision number.
- View the path of the VI.
- See how much memory the VI uses. The **Memory Usage** portion of the information box displays the disk and system memory used by the VI. (This figure applies only to the amount of memory the

VI is using and does not reflect the memory used by any of its subVIs.)

The memory usage is divided into space required for the front panel and the block diagram, VI code, and data space. The memory usage can vary widely, especially as you edit and execute the VI. The block diagram usually requires the most memory. When you are not editing the diagram, save the VI and close the block diagram to free space for more VIs. Saving and closing subVI panels also frees memory.

Macintosh CINs and Memory Manager Calls

Almost all existing CINs should continue to work in 4.1 without recompiling. The one exception to this is if you pass LabVIEW buffers to Macintosh system calls you may encounter problems.

To increase LabVIEW performance and decrease fragmentation on the Macintosh, data space memory manager calls (for example, DSSetHandleSize, DSNewHandle, CINSetArraySize, and so on) now operate on memory that is allocated by LabVIEW. The handles that LabVIEW creates are not Macintosh handles. Consequently, you will get an error if you try to pass a LabVIEW data handle to Macintosh routine that is going to resize the buffer.

LabVIEW Analysis VI Reference Manual Additions

The following information supplements the *LabVIEW Analysis VI Reference Manual*, the *LabVIEW Function and VI Reference Manual*, and the *Online Help*.

Complex QR Factorization

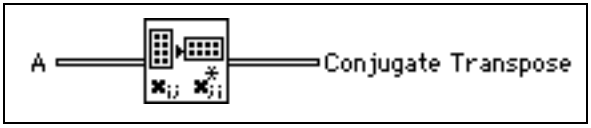
The Complex QR Factorization VI, located in **Analysis»Linear Algebra»Complex Linear Algebra»Advanced Complex Linear Algebra**, only supports the Householder algorithm.

General LS Linear Fit

The General LS Linear Fit VI, which is found in **Analysis»Curve Fitting**, has an additional input, called **covariance selector**. This **covariance selector** input bypasses the covariance computation.

Complex Conjugate Transpose

The Complex Conjugate Transpose VI, located in **Analysis**»**Linear Algebra**, takes the complex conjugate transpose of the input matrix A, forming the output complex matrix Conjugate Transpose.



[CDB] **A** refers to the input matrix to the complex conjugate transpose operation

[CDB] **Conjugate Transpose** is the complex conjugate transpose of the input matrix **A**. The Conjugate Transpose C of a complex matrix **A** is defined as:

$$C = A^{H} \Rightarrow c_{ij} = a^{*}_{ji}$$

LabVIEW Code Interface Reference Manual Additions

The example code in the *Computing the Cross Product of Two Two-Dimensional Arrays* section (found in the *Examples with Variably-Sized Data* section of Chapter 2, *CIN Parameter Passing*) produces the wrong result if the array B is not square. The line at the bottom of this code that reads:

```
*resultElmtp += aElmtp[i*k + l] + bElmtp[l*k + j];
```

should read:

```
*resultElmtp += aElmtp[i*k + l] + bElmtp[l*cols + j];
```

LabVIEW Instrument I/O VI Reference Manual Additions

The following information supplements the *LabVIEW Instrument I/O VI Reference Manual* and the *Online Help*.

VISA Attribute Descriptions

The following attributes are available with LabVIEW 4.1, but were not included in the *LabVIEW Instrument I/O VI Reference Manual*. Each attribute description includes the range, default value, and access privileges. *Local* applies to the current session only. *Global* refers to all sessions to the same VISA resource.

Interface Number of Parent

Specifies the board number of the parent device.

Range: 0 to FFFFh
Default: 0
Access Privilege: Read Only Global

Number of Bytes at Serial Port

Specifies the number of bytes currently available at the serial port used by this session.

Range: 0 to FFFFFFFFh
Default: N/A
Access Privilege: Read Only Global

Serial Baud Rate

Specifies the baud rate of the given communications port.

Range: System Dependent (any 32-bit value, usually from 300 to 38400)
Default: 9600
Access Privilege: Read/Write Global

Serial Data Bits

Specifies the number of data bits contained in each frame.

Range: 5 to 8
Default: 8
Access Privilege: Read/Write Global

Serial End Mode for Reads

Specifies the method used to terminate read operations.

Range: VI_ASRL_END_NONE(0), VI_ASRL_END_LAST_BIT(1), VI_ASRL_END_TERMCHAR(2)
Default: 2
Access Privilege: Read/Write Local

Serial End Mode for Writes

Specifies the method used to terminate write operations.

Range:	VI_ASRL_END_NONE(0), VI_ASRL_END_LAST_BIT(1), VI_ASRL_END_BREAK(3)
Default:	0
Access Privilege:	Read/Write Local

Serial Flow Control

Specifies the flow control method used for both transmitting and receiving data.

Range:	VI_ASRL_FLOW_NONE(0), VI_ASRL_FLOW_XON_XOFF(1), VI_ASRL_FLOW_RTS_CTS(2)
Default:	0
Access Privilege:	Read/Write Global

Serial Parity

Specifies the parity used with every frame that is transmitted or received.

Range:	VI_ASRL_PAR_NONE(0), VI_ASRL_PAR_ODD(1), VI_ASRL_PAR_EVEN(2), VI_ASRL_PAR_MARK(3), VI_ASRL_PAR_SPACE(4)
Default:	0
Access Privilege:	Read/Write Global

VISA Classes

The following table shows which VISA classes are supported by each VISA operation

☞ **Note:** ***The VISA Read and VISA Write operations are synchronous on the Macintosh.***

Operations	VISA Classes				
	Instr	**GPIB Instr**	**VXI/ GPIB-VXI MBD Instr**	**VXI/ GPIB-VXI RBD Instr**	**ASRL Instr**
VISA Assert Trigger	√	√	√	√	
VISA Clear	√	√	√		
VISA Disable Event	√	√	√	√	√
VISA Discard Events	√	√	√	√	√
VISA Enable Event	√	√	√	√	√
VISA In 8/16/32	√		√	√	
VISA Lock	√	√	√	√	√
VISA Map Address	√		√	√	
VISA Mem Allocate	√		√	√	
VISA Mem Free	√		√	√	
VISA Move In 8/16/32	√		√	√	
VISA Move Out 8/16/32	√		√	√	
VISA Out 8/16/32	√		√	√	
VISA Peek 8/16/32	√		√	√	
VISA Poke 8/16/32	√		√	√	
VISA Read	√	√	√		√

Operations	VISA Classes				
	Instr	GPIB Instr	VXI/ GPIB-VXI MBD Instr	VXI/ GPIB-VXI RBD Instr	ASRL Instr
VISA Read STB	√	√	√		
VISA Wait On Event	√	√	√	√	√
VISA Write	√	√	√		√
VISA Unmap Address	√		√	√	
VISA Unlock	√	√	√	√	√

The following table shows which VISA classes are supported by each VISA attribute.

Attributes	VISA Classes				
	Instr	GPIB Instr	VXI/ GPIB-VXI MBD Instr	VXI/ GPIB-VXI RBD Instr	ASRL Instr
Bytes at Serial Port	√				√
Commander Logical Address	√		√	√	
Fast Data Channel Number	√		√		
Fast Data Channel Mode	√		√		
Fast Data Channel Pairs	√		√		
Fast Data Channel Signals	√		√		
GPIB Primary Address	√	√	√	√	
GPIB Secondary Address	√	√	√	√	
Immediate Servant	√		√	√	
Increment Destination Count	√		√	√	

Attributes	VISA Classes				
	Instr	GPIB Instr	VXI/ GPIB-VXI MBD Instr	VXI/ GPIB-VXI RBD Instr	ASRL Instr
Increment Source Count	√		√	√	
Interface Number	√	√	√	√	√
Interface Number of Parent	√		√	√	
Interface Type	√	√	√	√	√
IO Protocol	√	√	√		√
Mainframe Logical Address	√		√	√	
Manufacturer Identification	√		√	√	
Maximum Queue Length	√	√	√	√	√
Model Code	√		√	√	
Resource Lock State	√	√	√	√	√
Resource Manufacturer Name	√	√	√	√	√
Resource Manufacturer ID	√	√	√	√	√
Resource Name	√	√	√	√	√
Send End Enable	√	√	√		√
Serial Baud Rate	√				√
Serial Data Bits	√				√
Serial End Mode for Reads	√				√
Serial End Mode for Writes	√				√
Serial Flow Control	√				√
Serial Parity	√				√

Attributes	VISA Classes				
	Instr	GPIB Instr	VXI/ GPIB-VXI MBD Instr	VXI/ GPIB-VXI RBD Instr	ASRL Instr
Serial Stop Bits	√				√
Slot	√		√	√	
Suppress End Enable	√	√	√		√
Termination Character	√	√	√		√
Termination Char Enable	√	√	√		√
Timeout Value	√	√	√	√	√
Trigger Identifier	√	√	√	√	
User Data	√	√	√	√	√
Version of Implementation	√	√	√	√	√
Version of Specification	√	√	√	√	√
VXI Logical Address	√		√	√	
VXI Memory Address Space	√		√	√	
VXI Memory Base Address	√		√	√	
VXI Memory Size	√		√	√	
Window Access	√		√	√	
Window Base Address	√		√	√	
Window Size	√		√	√	

320542D-01
May 1997