Release Notes

LabVIEW® for Windows

Version 4.1

Welcome to LabVIEW for Windows

These release notes introduce you to the contents of LabVIEW for Windows, 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 5
Differences between LabVIEW for Windows 3.1, Windows 95, and Windows NT 6
Data Acquisition and GPIB Installation Notes 8
Installing Windows 3.1 GPIB Driver Files 9
Manual Clarifications and Additions 9

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.

HiQ®, NI-DAQ® and LabVIEW® are trademarks of National Instruments Corporation.
Product and company names listed are trademarks or trade names of their respective companies.

 © 1997 National Instruments Corporation. All rights reserved.

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 Windows is distributed primarily on a CD, which contains versions for Windows 3.1, Windows 95, and Windows NT. This CD also includes the complete instrument driver library that was available when LabVIEW 4.1 released. If you do not have a CD-ROM drive, the Windows 3.1 and Windows 95 versions are available on floppy disks.

The new *LabVIEW Online Tutorial* requires that the LabVIEW 4.1 distribution CD be in your CD-ROM drive. Minimum requirements are 640x480 pixel resolution, 256-color display, and the Microsoft Video for Windows driver.

The floppy disk versions for Windows 3.1 and Windows 95 are available separately as a Base Development System and a Full Development System. The Base Development System consists of a set of disks for installing LabVIEW and the associated VIs. The Full Development System includes the Base Development System disks, as well as a second set of disks for installing the advanced analysis VIs, libraries for developing CINs.

The Windows 3.1 version of LabVIEW 4.1 runs under Windows 3.1 and Windows for Workgroups 3.1 in 386 enhanced mode. It is recommended that you have a minimum of 8 MB of RAM. LabVIEW can run on an 80386-based PC, but we strongly recommend a computer with an 80486 CPU. In addition, LabVIEW for Windows requires a coprocessor.

The Windows 95 version of LabVIEW 4.1 runs under any system that supports Windows 95. You should have a minimum of 8 to 12 MB of RAM for this version to run effectively.

The Windows NT version of LabVIEW 4.1 runs only under Windows NT version 3.5.1 or greater. You should have a minimum of 12 to 16 MB of RAM for this version to run effectively. LabVIEW for

Windows NT only runs under Windows NT 80*x*86 computers. It does not run on other processors, such as the DEC Alpha, MIPS, or PowerPC, because LabVIEW uses 80386 instructions. The DEC Alpha, MIPS, and PowerPC 80*x*86 emulators must emulate 80386 instructions in order to run LabVIEW.

☞ **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

These instructions apply to Windows 3.1, Windows 95, and Windows NT systems.

1. If you are installing the Windows NT version of LabVIEW, you must log on to Windows NT as an administrator or as a user with administrator privileges.
2. If you are installing from a floppy disk, insert Disk 1 and run the `setup.exe` program. If you are installing LabVIEW for Windows 3.1 from a CD, run the `X:\WIN31\INSTALL\DISK1\SETUP` on the CD. If you are installing LabVIEW for Windows 95 or Windows NT, run the `X:\WIN95-NT\INSTALL\DISK1\SETUP` on the CD.
3. 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.
4. After you choose an installation, follow the instructions that appear on the screen.
5. The installer asks you whether you want to create a Windows program group with icons for LabVIEW, and the driver configuration programs.
6. In Windows 3.1, you should use the GPIB drivers that came with your board or device.

The LabVIEW 4.1 CD contains all of the latest versions of our GPIB drivers, located in the `drivers` directory.
In Windows 95 and Windows NT, the LabVIEW installer calls the National Instruments GPIB installer, which installs the GPIB drivers and modifies the registry. For more information on installation and configuration of GPIB drivers, refer to the *Data Acquisition and GPIB Installation Notes* section of this document.

In Windows 95 and Windows NT, if you use data acquisition (DAQ) devices, the LabVIEW installer calls the NI-DAQ installer, which installs the correct DAQ driver software. In Windows 3.1, you will need to run the NI-DAQ installer yourself. This installer is located in the `drivers` directory. For more information on installation and configuration of DAQ drivers, refer to the *Data Acquisition and GPIB Installation Notes* section of this document.

7. If you are installing the Windows 3.1 version, the installer reminds you to install the Advanced Analysis VIs if you have them. If you have the LabVIEW for Windows Base Development System, you can just ignore this message and run LabVIEW. If you have the Full Development System, complete the following steps.

   a. If you are installing the Full Development System from floppy disks, you have a second set of disks that contain the Advanced Analysis VIs. The first disk of this set contains an installer for these VIs. The installer, called `SETUP.EXE`, is similar to the LabVIEW installer. Insert Disk 1 of the analysis VI disks and run the `SETUP.EXE` program from the Program Manager as described in Step 2. Follow the instructions on the screen. If you are installing the Full Development System from a CD in Windows 95 or Windows NT, the LabVIEW installer installs the Advanced Analysis VIs automatically. However, if you are installing the system from a CD in Windows 3.1, you must run the analysis setup program, located in `x:\Analysis` directory.

   b. ***(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, called `SETUP.EXE`, is similar to the LabVIEW installer. Insert Disk 1 of the VXI VI disks and run the `SETUP.EXE` program from the Program Manager as described in Step 2. Follow the instructions on the screen. Also included in the VXI Development System is a set of disks

containing the VXI Instrument Libraries. You can install the instrument drivers that are needed for your application at this time.

8. ***(CD Installation Only)*** The LabVIEW 4.1 CD contains all of the latest versions of our VXI Instrument Libraries. In Windows 95 and Windows NT, the VXI Instrument Libraries are automatically installed. Windows 3.1 users should install them by launching `DRIVERS\VXI\WIN16\DISK1\SETUP.EXE`. If you do not need VXI support, you can skip this installation.

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 Online Tutorial* and the *LabVIEW QuickStart Guide*. The online tutorial introduces you to the LabVIEW environment and the *LabVIEW QuickStart Guide* covers the basic steps of building a LabVIEW application. After completing the *LabVIEW Online Tutorial* and the *LabVIEW QuickStart Guide*, read and perform the exercises in the *LabVIEW Tutorial Manual* for more information on LabVIEW features.
- If you are using data acquisition (DAQ) and want to find examples, go through the DAQ Solution Wizard by starting LabVIEW and selecting the **DAQ Solution Wizard** button. To find any other type

of examples, go through the Search Examples online help file by starting LabVIEW and selecting the **Search Examples** button.

- 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 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 and the Getting Started sections in Chapter 2 provide an excellent starting place for data acquisition.

Differences between LabVIEW for Windows 3.1, Windows 95, and Windows NT

LabVIEW is very similar between all three platforms. Unless your application calls CINs or DLLs, uses OLE, or communicates with hardware that is not supported by one of the platforms, you should be able to take VIs to other platforms without any problems or having to make any modifications. Following is a complete list of the differences between the platforms.

File Name Support

LabVIEW for Windows 95 and Windows NT both support long filenames. In addition to the VFAT file system, LabVIEW for Windows NT also supports the NT File System (NTFS). Windows 3.1 only supports filenames with 8 characters plus an optional 3 characters for the extension. If you use Windows 3.1 or need to transfer files to Windows 3.1, we recommend you use VI Libraries (`.llbs`), which are a type of LabVIEW file that can contain multiple VIs inside of it, allowing you to use long names for your VIs even under Windows 3.1.

CIN Support

LabVIEW supports creation of CINs under all three platforms. However, under Windows 3.1 you are limited to creating CINs using a compiler called Watcom/C, the only compiler that can create the kind of 32-bit code LabVIEW can support in Windows 3.1. Windows 3.1 cannot load CINs from Windows 95 or Windows NT unless the CINs were created using Watcom.

Windows 95 and Windows NT support CINs created using Visual C++, the Win32 SDK Microsoft C command line compiler (CL386.exe), and Symantec C. Those CINs are not limited in that they can call DLLs and system calls. In addition, Windows 95 and Windows NT have backward support for Watcom CINs designed for Windows 3.1, but Windows 95 and NT cannot support Watcom CINs that make system calls, load DLLs, or do low level register based I/O.

DLL Support

LabVIEW for Windows 3.1 can call 16-bit DLLs. LabVIEW for Windows 95 and Windows NT are Win32 applications and can only call 32-bit DLLs. If you have existing 16-bit Windows 3.1 DLLs that you want to call under Windows 95 or NT, you will need to recompile them to be 32-bit or get new versions. If you cannot get a new version of a DLL, it is possible under Windows 95 to write a 32-bit DLL (called a *thunking* DLL) that can be used to call a 16-bit DLL; however, it is a fairly difficult process and National Instruments recommends against taking that approach if possible.

Low Level Register I/O

LabVIEW for Windows 3.1 and Windows 95 have a set of VIs called In Port and Out Port that can be used to read a write hardware registers. Windows NT applications cannot directly manipulate hardware. Instead, if you need to communicate with a hardware device you must write a Windows NT driver.

OLE and Drag and Drop Support

Drag and Drop and OLE Automation VIs are only supported under Windows 95 and NT. The OLE Automation VIs let you call OLE interfaces in other applications, such as Excel.

Native File Dialog Support

LabVIEW can use the standard file dialog when you perform operations such as opening and saving VIs. LabVIEW also has a dialog that can manipulate libraries more directly than the standard file dialog. You can choose one of these dialog options in **Edit»Preferences**.

Data Acquisition and GPIB Installation Notes

When you install LabVIEW, the installer places the application and most of the related files in a directory you specify. The default name of this directory is `LABVIEW`. If you install DAQ or GPIB VIs, the installer places additional files, described in the following sections.

☞ **Note:** ***All National Instruments GPIB interfaces and DAQ devices come with the drivers and other software you need to use them. LabVIEW also comes with the drivers and other software you need to use National Instruments hardware. While the drivers shipped with LabVIEW are the same NI-488.2 and NI-DAQ drivers National Instruments ships with its GPIB and DAQ hardware, the version numbers may differ. Always use the driver with the higher version number. You can find out what version of NI-DAQ you are using with LabVIEW by running the Get Device Information VI. For instance, the LabVIEW 4.1 installer installs version 5.0 of NI-DAQ, the driver software for DAQ devices. If you already own a device with a version of NI-DAQ earlier than 5.0, do not install that driver.***

You must configure your DAQ hardware before you can use the LabVIEW DAQ VIs. To configure your DAQ hardware, you need to run the NI-DAQ Configuration Utility. For further information on how to configure your DAQ hardware with the NI-DAQ Configuration Utility, please refer to the help files that are installed with LabVIEW.

Windows 3.1 DAQ Configuration

For Windows 3.1 configuration, run `NICFG16.EXE,` which is found in the Windows `SYSTEM` directory.

For instructions on how to configure your particular DAQ device, refer to the `NIDAQCFG.HLP` file. Follow the links for configuring the devices you have.

Further information about the NI-DAQ driver can be found in a text file called `README.WRI` in the `LABVIEW` directory.

Windows 95 and Windows NT 4.0 DAQ Configuration

For Windows 95 and Windows NT 4.0 DAQ configuration, run the NI-DAQ Configuration Utility, which can be launched from the **Start** button. To launch it, click on the **Start** button, and go to **Programs»LabVIEW»NI-DAQ Configuration Utility**.

For instructions on how to configure your particular DAQ device, refer to the NI-DAQ Configuration Help file. To view this help file, click on the Start Button, and go to **Programs»LabVIEW»NI-DAQ Configuration Help**. Follow the links for *Configuring Devices in Windows 95*.

Further information about the NI-DAQ driver can be found in the NI-DAQ Read Me File. To view this file, click on the Start Button, and go to **Programs»LabVIEW»NI-DAQ Read Me File**. This will launch WordPad automatically so you can read the document.

Installing Windows 3.1 GPIB Driver Files

Under Windows 3.1, you can use the driver that came with your device. The LabVIEW CD comes with the latest version of our Windows 3.1 drivers, which is found in the `drivers` directory.

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.

Startup Screen

When you launch LabVIEW, you are greeted with a navigation dialog box where introductory material, common commands, and Quick Tips are easily accessible. If you prefer to bypass the navigation dialog, you can disable it using a checkbox at the bottom of the dialog box. To reenable it, use the Preferences dialog box.

When all VIs are closed, a similar dialog box appears. The **Small Dialog** button switches to a simpler version of the dialog box—with only New, Open, and Exit buttons—which is similar to previous versions of LabVIEW.

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). 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.

OLE Automation VIs for Communication with HiQ

LabVIEW includes OLE Automation VIs for communication with HiQ. HiQ is a National Instruments application for interactive analysis, three-dimensional data visualization, and automated report generation with LabVIEW. Using this OLE automation interface link between LabVIEW 4.1 and HiQ 3.1, you can generate technical reports automatically from LabVIEW. Simply design your report template interactively using HiQ, then automatically generate your publication-quality technical reports directly from LabVIEW using the new HiQ VIs. Your LabVIEW data, analysis, and graphs, are organized into HiQ Notebooks, which you can print automatically from LabVIEW. New VIs include Launch HiQ, Open Notebook, Set Data (Text, Script, Integer, Real, Complex, Vector, Matrix), Get Data (Text, Script, Integer, Real, Complex, Vector, Matrix), Run Script, Save Notebook, Print Notebook, Close Notebook, and Exit HiQ. For more information on these VIs, see the HiQ Palette (**Functions»HiQ**) or the *LabSuite Online Help* (`HiQ.hlp`) file.

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.

Update VXI*plug&play* Drivers Menu Option

VXI*plug&play* installs files in a special `vxipnp` directory on your system. This directory is created if you install the NI-VISA driver. If LabVIEW detects this directory, it adds the **Update VXI*plug&play* Drivers** menu option to the **File** menu, which makes it easy to import these drivers into LabVIEW's **Functions** palette.

When you select **File**»**Update VXI*plug&play* Drivers**, LabVIEW scans the `vxipnp` directory to find new instrument drivers. LabVIEW looks for libraries and CVI Function Panels and displays lists of the new files that were found, asking which function panels you want to convert and which libraries you want to copy to the `instr.lib` directory. For every function panel you select, LabVIEW converts the function panel as if you had selected **File**»**Convert FP File**.

Updated CVI Panel Converter for Easier Handling of Arrays and Strings

The documentation for the CVI Function Panel converter indicates that the main reason you might have problems in converting a file is if the functions return arrays or strings. In these cases, LabVIEW needs to preallocate a buffer large enough to hold the data. With previous versions of LabVIEW, you had to modify a constant on the diagram if the panel had the potential for returning a larger buffer. With LabVIEW 4.1, the buffer size can be specified as an input to the subVI instead of a constant. Consequently, there are very few cases in which you would need to modify the diagrams or panels of a converted function panel.

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. This palette also is used as part of the LabVIEW QuickStart Manual. 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.

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 Tech Support VIs accessible from the **Help** menu. Also, if you have the Application Builder libraries installed, you will see a **Create Distribution Kit** option in the **Project** 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. For subdirectory names in Windows, which does not support mixed case or long file names, you can place a file with the same name as the directory with a `.txt` extension. 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.

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*.

Creating CINs Using Symantec C for Windows 95/NT

You can now use Symantec C for Windows 95 and Windows NT to create CINs. This has been tested with Symantec C, Version 7.2.

☞ **Note:** ***You cannot currently create external subroutines using Symantec C.***

The process for creating CINs using Symantec C is very similar to the process for Visual C++ as described in the *LabVIEW Code Interface Reference Manual*. Using `ntlvsb.mak`, just use `smake` instead of `nmake` and the switch should happen transparently.

Calling LabVIEW Functions from DLLs Created Using Symantec C for Windows 95/NT

If you want to build DLLs that call back into LabVIEW with Symantec C, include `cintools\win32\labview.sym.lib` in your Symantec project or link command.

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.

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

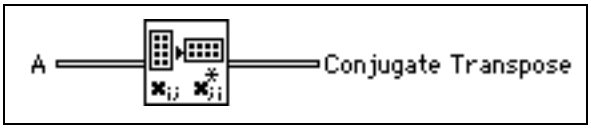
The Complex QR Factorization VI, located in **Analysis**»**Linear Algebra**»**Complex Linear Algebra**»**Advanced Complex Linear Algebra**, only supports the Householder algorithm.

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

U16

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

U32

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

U32

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

U16

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

U16

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

U16

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

U16

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

U16

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 asynchronous in Windows.***

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	√		√	√	

Operations	VISA Classes				
	Instr	GPIB Instr	VXI/ GPIB-VXI MBD Instr	VXI/ GPIB-VXI RBD Instr	ASRL Instr
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	√	√	√		√
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	√		√	√	
Increment Source Count	√		√	√	
Interface Number	√	√	√	√	√
Interface Number of Parent	√		√	√	
Interface Type	√	√	√	√	√
IO Protocol	√	√	√		√
Mainframe Logical Address	√		√	√	
Manufacturer Identification	√		√	√	
Maximum Queue Length	√	√	√	√	√

Attributes	VISA Classes				
	Instr	GPIB Instr	VXI/ GPIB-VXI MBD Instr	VXI/ GPIB-VXI RBD Instr	ASRL Instr
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	√				√
Serial Stop Bits	√				√
Slot	√		√	√	
Suppress End Enable	√	√	√		√
Termination Character	√	√	√		√
Termination Char Enable	√	√	√		√
Timeout Value	√	√	√	√	√
Trigger Identifier	√	√	√	√	
User Data	√	√	√	√	√

Attributes	VISA Classes				
	Instr	**GPIB Instr**	**VXI/ GPIB-VXI MBD Instr**	**VXI/ GPIB-VXI RBD Instr**	**ASRL Instr**
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	√		√	√	

320540D-01
May 1997