LabVIEW™ Upgrade Notes

Version 7.1

These upgrade notes describe the process of upgrading LabVIEW for Windows, Mac OS, and UNIX to version 7.1, issues you might encounter when you upgrade, and new features.

If you are upgrading from LabVIEW 6.1 or earlier to LabVIEW 7.1, refer to the *LabVIEW 7.0 Upgrade Notes* for information about several enhancements to the LabVIEW environment and for information about customizing the LabVIEW environment. National Instruments recommends that all users upgrading from LabVIEW 6.1 or earlier read the *LabVIEW 7.0 Upgrade Notes* in addition to these upgrade notes. Refer to the National Instruments Web site at `ni.com/info` and enter the info code `rd70un` to access the *LabVIEW 7.0 Upgrade Notes*.

Refer to the *Getting Started with LabVIEW* manual for exercises you can complete to familiarize yourself with the new features and enhancements to the LabVIEW environment in LabVIEW 7.0.

For more information…

Refer to the *LabVIEW Help* for more information about LabVIEW 7.1 features. Access the *LabVIEW Help* by selecting **Help»VI, Function, and How-To-Help**. Refer to the *LabVIEW Bookshelf* to search PDF versions of all the LabVIEW manuals and Application Notes. Access the *LabVIEW Bookshelf* by selecting **Help»Search the LabVIEW Bookshelf**.

You must have Adobe Reader 5.0.5 or later with search and accessibility plug-ins to view and use the PDFs. **(Mac OS)** You must use Adobe Reader 6.*x* with search and accessibility plug-ins to view and use the PDFs. Refer to the Adobe Systems Incorporated Web site at `www.adobe.com` to download Adobe Reader.

March 2004
321780F-01

ni.com

 Printed in Ireland.

NATIONAL INSTRUMENTS

Contents

Upgrade Issues

If you are upgrading from LabVIEW 7.0, refer to the *Converting VIs*, the *Upgrading Toolkits, Instrument Drivers, and Add-Ons*, and the *Upgrading from LabVIEW 7.0* sections of this document first.

If you are upgrading from LabVIEW 6.*x*, refer to the *Converting VIs*, the *Upgrading Toolkits, Instrument Drivers, and Add-Ons*, and the *Upgrading from LabVIEW 6.x* sections of this document first.

If you are upgrading from LabVIEW 5.*x*, refer to the *Converting VIs*, the *Upgrading Toolkits, Instrument Drivers, and Add-Ons*, and the *Upgrading from LabVIEW 5.x* sections of this document first.

If you are upgrading from LabVIEW 4.*x*, refer to the *Converting VIs*, the *Upgrading Toolkits, Instrument Drivers, and Add-Ons*, and the *Upgrading from LabVIEW 4.x* sections of this document first.

If you are upgrading from LabVIEW 3.*x*, refer to *Upgrading Toolkits, Instrument Drivers, and Add-Ons* and the *Upgrading from LabVIEW 3.x or Earlier Versions* sections of this document first.

Converting VIs

When you open a VI last saved in LabVIEW 4.0 or later, LabVIEW 7.1 automatically converts and compiles the VI. You must save the VI in LabVIEW 7.1, or the conversion process, which uses extra memory resources, occurs every time you access the VI.

Also, you might experience a large run-time degradation of performance for any VI that has unsaved changes, including a recompile. Refer to the *Memory Issues in Front Panels* section of the *LabVIEW Performance and Memory Management* Application Note for more information about this performance issue.

Note VIs you save in LabVIEW 7.1 do not load in earlier versions of LabVIEW. Select **File»Save with Options** and select the **Save for Previous** option to save VIs so they can run in LabVIEW 7.0. Before saving VIs in LabVIEW 7.1, keep a backup copy of VIs you plan to use in LabVIEW 7.0 or earlier.

You can estimate the amount of memory required to convert VIs by totalling the amount of memory that the VIs and all their subVIs occupy on disk. If the VIs are in VI libraries, add approximately 30 percent of the VI library size because the VIs are compressed. The conversion process might require at least that much memory and an additional 20 MB of memory to run LabVIEW.

If your computer does not have enough memory to convert all the VIs at once, convert the VIs in stages. Examine the hierarchy of VIs you want to convert and begin by loading and saving subVIs in the lower levels of the hierarchy. Then progress gradually to the higher levels of the hierarchy. You also can select **Tools»Advanced»Mass Compile** to convert a directory of VIs. However, mass compiling converts VIs in a directory or VI library in alphabetical order. If the conversion process encounters a high-level VI first, mass compiling requires approximately the same amount of memory as if you opened the high-level VI first.

You can monitor memory usage by selecting **Help»About LabVIEW** to display a summary of the amount of memory you have used.

Upgrading Toolkits, Instrument Drivers, and Add-Ons

After you install LabVIEW 7.1, make sure you have the compatible version of any toolkits and add-ons and reinstall the toolkits and add-ons in the LabVIEW 7.1 directory.

You also must mass compile existing toolkit, instrument driver, and add-on VIs for use in LabVIEW 7.1. Refer to the *Converting VIs* section of this document for more information about mass compiling VIs. LabVIEW 7.1 is compatible with toolkits, instrument drivers, and add-ons designed for LabVIEW 4.0 and later, except for the LabVIEW Application Builder

(Full Development System) If you have the LabVIEW Application Builder, you must upgrade to LabVIEW Application Builder 7.1. The LabVIEW Professional Development System version 7.1 includes Application Builder 7.1. Refer to the *LabVIEW Application Builder User Guide* for more information about installing the LabVIEW Application Builder.

Upgrading Previous Versions of LabVIEW

Upgrading to new versions of LabVIEW does not affect previous versions of LabVIEW on the computer because the new versions install in a different directory. LabVIEW 5.*x* and earlier installs in the `labview` directory. LabVIEW 6.0 and later installs in the `labview` *`x.x`* directory, where *`x.x`* is the version number.

To use LabVIEW environment settings from a previous version of LabVIEW, copy the LabVIEW preferences file from the `labview` directory in which the previous version is installed. After you install LabVIEW 7.1, copy the LabVIEW preferences file from the previous version into the LabVIEW 7.1 directory. If you replace the LabVIEW 7.1 preferences file with a preferences file from a previous version, you might override preference settings added to LabVIEW since the previous version.

(Windows) LabVIEW stores preferences in the `labview.ini` file.

(Mac OS) LabVIEW stores preferences in the `LabVIEW Preferences` file in the `Library:Preferences` folder in your home directory.

(UNIX) LabVIEW stores preferences in the `.labviewrc` file in your home directory.

To use files from the `user.lib` directory of a previous version of LabVIEW, copy the files from the `labview` directory in which the previous version is installed. After you install LabVIEW 7.1, copy the files to the `user.lib` directory in the LabVIEW 7.1 directory.

(Windows) You also can replace the existing version of LabVIEW with LabVIEW 7.1 by using the Add/Remove Programs applet in the Control Panel to uninstall the existing version of LabVIEW. The uninstaller does not remove any files you created in the `labview` directory.

Caution If you saved your own VIs and controls in existing `.llb` files in the `vi.lib` directory, LabVIEW uninstalls the `.llb` files when you uninstall or reinstall, including any VIs and controls you saved in the `.llb` files. Save your VIs and controls in the `user.lib` directory to add them to the **Functions** and **Controls** palettes.

Run the LabVIEW 7.1 installer and set the default installation directory to the same `labview` directory where you installed the previous version of LabVIEW to replace your existing version of LabVIEW.

Refer to the following sections for upgrade and compatibility issues specific to different versions of LabVIEW.

Upgrading from LabVIEW 7.0

You might encounter the following issues when you upgrade to LabVIEW 7.1 from LabVIEW 7.0.

Platforms Supported

LabVIEW 7.1 includes the following changes in platforms supported:

- LabVIEW 7.1 does not support Windows Me/98.
- LabVIEW 7.1 does not support Mac OS 9.*x* or earlier and the PPC Toolbox.
- LabVIEW 7.1 supports Sun Solaris 7 or later.

Using the Open FP Method

The Open FP method in LabVIEW 7.0 was renamed to Old Open FP, and LabVIEW 7.1 includes a new Open FP method. The method in

LabVIEW 7.1 does not return an error if the front panel is already open. If you have VIs that use the Old Open FP method, replace the method with the new Open FP method or with the LabVIEW 7.1 OpenFP VI in the `labview\vi.lib\_oldvers\_oldvers.llb`.

VIs Removed from the Functions Palette

LabVIEW 7.1 does not install the following VIs:

- **HP34401A Find Range**—Use the HP34401A Config Measurements VI instead.
- **Polynomial Real Zero Counter**—Use the Polynomial Real Zeros Counter VI instead.
- **PPC VIs**—Use the TCP VIs instead.

Changes to the Dot Product VI

In LabVIEW 7.0, the Dot Product VI performed the calculation shown in the following equation.

$$X*Y = \sum_{i=0}^{n-1} x_i y_i$$

In LabVIEW 7.1, the Dot Product VI performs the following calculation:

$$X*Y = \sum_{i=0}^{n-1} x_i y_i*$$

where y_i^* is the complex conjugate of y_i.

Upgrading from LabVIEW 6.*x*

You might encounter the following issues when you upgrade to LabVIEW 7.1 from LabVIEW 6.*x*. Refer to the *Upgrading from LabVIEW 7.0* section of this document for information about other upgrade issues you might encounter.

Refer to the *LabVIEW Upgrade Notes* for each version of LabVIEW between version 6.*x* and 7.1 at `ni.com/manuals` for more information about the new features and changes in each version.

Windows 95 Support

LabVIEW 7.1 does not support Windows 95.

Saving Waveform Data to a File

In LabVIEW 7.0, the waveform data type was updated to use the time stamp data type for the t0 component rather than a double-precision, floating-point number. If you saved data in the waveform data type to a file without including information about the data type in LabVIEW 6.*x*, you might encounter an error if you try to retrieve that data in LabVIEW 7.1. Refer to the National Instruments Web site at `ni.com/info` and enter the info code `exd9zq` for more information about migrating waveform data from LabVIEW 6.*x* to LabVIEW 7.1.

HiQ Support

National Instruments does not support HiQ functionality on Mac OS in LabVIEW 7.1 and will no longer support HiQ functionality on all platforms after LabVIEW 7.1. If an application uses HiQ VIs, consider replacing them with the Analyze and Mathematics VIs. Refer to the *LabVIEW Help* for information about using the Analyze and Mathematics VIs.

Serial Compatibility VIs

The Serial Compatibility VIs do not appear on the **Functions** palette. Use the VISA VIs and functions to build VIs that communicate with VXI devices.

LabVIEW no longer uses the `serpdrv` driver to communicate with the serial driver of the operating system. LabVIEW includes compatible VIs based on VISA. For new applications, use the VISA and Serial VIs and functions to control serial devices. Any VIs built in previous versions of LabVIEW that include Serial VIs continue to work in LabVIEW 7.1.

If you reconfigured the mapping of port numbers to ports, you must specify a mapping to those ports. Use the set serial alias ports VI in the `labview\vi.lib\Instr\_sersup.llb` to specify the serial port mappings. Wire a string array to the **VISA Aliases** input of the VI and enter the port names you use in the input array. Each element in the array should correspond to a port. For example, if you configured port 0 to map to the VISA alias MySerialPort, enter `MySerialPort` as the first element of the **VISA Aliases** input array. You must call the Set Serial Alias Ports VI before you call the VISA Configure Serial Port VI.

Refer to the `examples\instr\smplserl.llb` for examples of using the VISA VIs and functions to control serial instruments.

Default Data in Loops

In LabVIEW 6.0 and earlier, For Loops produced undefined data if the loop did not execute. In LabVIEW 6.1 and later, For Loops produce default data

if you wire `0` to the count terminal of the For Loop or if you wire an empty array to the For Loop as an input with auto-indexing enabled. The loop does not execute, and any output tunnel with auto-indexing disabled contains the default value for the tunnel data type.

Remote Front Panel License

The LabVIEW Full Development System and the Application Builder include a remote front panel license that allows one client to view and control a front panel remotely. The LabVIEW Professional Development System includes a remote front panel license that allows five clients to view and control a front panel remotely.

You can upgrade the remote front panel license to support more clients.

Multiple Thread Allocation

LabVIEW 7.1 allocates more threads for executing VIs than in versions earlier than LabVIEW 7.0. Because of this change, you might encounter errors with multiple threads if you incorrectly mark Call Library Function Nodes as reentrant when the DLL you call is not actually reentrant. Refer to Chapter 2, *Shared Libraries (DLLs)*, of the *Using External Code in LabVIEW* manual for more information about reentrancy.

To change how LabVIEW allocates threads, use the `threadconfig.vi` in the `vi.lib\Utility\sysinfo.llb`. You also can disable reentrancy for VIs by removing the checkmark from the **Reentrant execution** checkbox in the **Execution** page of the **VI Properties** dialog box.

Refer to the *Using LabVIEW to Create Multithreaded VIs for Maximum Performance and Reliability* Application Note for more information about multiple thread allocation.

Instrument Drivers

The LabVIEW package no longer includes the LabVIEW Instrument Driver Library CD, which contains instrument drivers. Download instrument drivers from the National Instruments Instrument Driver Network at `ni.com/idnet`. The National Instruments Device Drivers CD includes NI-DAQ, NI-VISA, and other National Instruments drivers.

Units and Conversion Factors

After using the Compound Arithmetic function, you no longer need to use the Convert Unit function to remove the extra unit.

The unit conversion factors in LabVIEW 7.1 more closely match the guidelines published by the National Institute for Standards and Technology (NIST) in the *Guide for the Use of the International System*

of Units (SI). Also, the `calorie` unit is now `calorie (thermal)`, and `horse power` is now `horsepower (electric)`. The abbreviations for these units did not change. The following table details the changes in unit conversion factors between LabVIEW 6.1 and 7.1.

Unit	6.1 Definition	7.1 Definition
astronomical unit (AU)	149,498,845,000 m	149,597,900,000 m
British Thermal Unit (mean)	1055.79 J	1055.87 J
electron volt (eV)	1.602e–19 J	1.60217642e–19 J
foot-candle	10.764 lx	10.7639 lx
horse power versus horse power (electric)	745.7 W	746 W The new conversion is exact.
imperial gallon	4.54596 l	4.54609 l
light year	9.4605 Pm	9.46073 Pm
pound force	4.448 N	4.448222 N
rod	16.5 ft	5.029210 m
slug	32.174 lb	14.59390 kg
unified atomic mass (u)	1.66057e–27 kg	1.66053873e–27 kg

Defer Panel Updates Property

When you set this property to TRUE, LabVIEW redraws any front panel objects with pending changes then defers all new requests for front panel updates. In LabVIEW 6.1 and earlier, LabVIEW waits until the Defer Panel Updates property is FALSE to redraw any front panel objects with pending changes.

In some cases, this change can cause LabVIEW to redraw the changed elements of the front panel an extra time.

Data Ranges for Numeric Controls

In LabVIEW 6.1 and earlier, some numeric controls were set to a minimum value of `0.00`, a maximum value of `0.00`, an increment value of `0.00`, and an out of range action of **Ignore** as default settings. In LabVIEW 7.1, these numeric controls use the default data range values for the data type.

Coercion Dots and Type Definitions

In LabVIEW 6.1 and later, wires include information about type definitions, so you might notice more coercion dots on block diagrams. If you wire a type definition to a VI or function terminal that is not a type definition terminal, a coercion dot appears. A coercion dot also appears if you wire an output terminal that is a type definition to an indicator that is not a type definition. These coercion dots indicate where you are not using type definitions consistently in the VIs.

In this case, coercion dots do not affect run-time performance.

Refer to the *LabVIEW Help* for information about using the Flatten To String function to flatten type definitions.

File Dialog Box Button Label

In LabVIEW 6.1 and earlier, the file dialog box the File Dialog function displays has a button label of **Save** if the user can enter a new filename. Otherwise, the button label is **Open**. In LabVIEW 7.1, the button label is **OK** in all cases unless you change it. Use the **button label** input of the File Dialog function to change the label of the button. If you use the File Dialog function in an existing VI, consider reviewing the behavior of the VI to make sure the default label of **OK** is appropriate to the functionality of the VI.

Control Online Help Function

The **Path to the help file** input of the Control Online Help function is required. You can wire a compiled help filename (`.chm` or `.hlp`) or the full path to a compiled help file to the input. If you wire only a compiled help filename, LabVIEW searches the `labview\help` directory for that file.

Run VI Method

If you set the **Auto Dispose Ref** parameter of the Run VI method to TRUE, LabVIEW disposes the reference even if the method returns an error. This might break a VI if a part of the block diagram depends on the reference.

Displaying the Front Panel When Loaded

In LabVIEW 7.1, if you configure a VI to display its front panel when LabVIEW loads the VI and you load the VI using the VI Server, LabVIEW does not display the front panel. You must use the Open FP method to display the front panel programmatically.

Open VI Reference Function

If you use the Open VI Reference function to create a reference to a template and the template is already in memory, the function returns an error.

Exponential Representation

In LabVIEW 6.0 and earlier, the ^ operator represented exponentiation in the Formula Node. In LabVIEW 6.1 and later, the operator for exponentiation is `**`—for example, `x**y`. The ^ operator represents the bitwise exclusive or (XOR) operation.

IVI Configuration Store File

The IVI Configuration Store file format now requires that all names be case-sensitive. If you use logical names, driver session names, or virtual names in your program, make sure that the name you use matches the name defined in the IVI Configuration Store file exactly, without any variations in the case of the characters in the name.

Technical Support Form

The LabVIEW installation program does not install `techsup.llb`. Refer to the National Instruments Web site at `ni.com/support` to solve installation, configuration, and application problems and questions.

Upgrading from LabVIEW 5.*x*

You might encounter the following issues when you upgrade to LabVIEW 7.1 from LabVIEW 5.*x*. Refer to the *Upgrading from LabVIEW 6.x* and *Upgrading from LabVIEW 7.0* sections of this document for information about other upgrade issues you might encounter.

Refer to the *LabVIEW Upgrade Notes* for each version of LabVIEW between version 5.*x* and 7.1 and to the *LabVIEW 5.1 Addendum* at `ni.com/manuals` for more information about the new features and changes in each version.

Converting Datalog Files

LabVIEW 7.1 checks the type definition of datalog files to determine if a conversion is necessary. If the datalog file is older than LabVIEW 6.0 or contains waveform datatypes, LabVIEW 7.1 converts the file for reading and appending. In all other cases, reading from the datalog file and appending to the datalog file does not convert the datalog file.

When you open a datalog file created in an earlier version of LabVIEW, LabVIEW 7.1 prompts you to convert the file to the LabVIEW 7.1 format.

If you choose to convert it, LabVIEW replaces the datalog file with data converted to the new format. If you choose not to convert the file, LabVIEW 7.1 returns an error and does not open the file.

Note You should make a backup copy of datalog files before converting if you plan to continue to use old data in LabVIEW 6.1 or earlier. You cannot revert to or read converted datalog files in LabVIEW 6.1 or earlier.

To automatically convert datalog files when you open them, add the following line to the LabVIEW preferences file:

```
silentDatalogConvert=True
```

(Mac OS) Add the following line:

```
silentDatalogConvert:True
```

(UNIX) Add the following line:

```
labview.silentDatalogConvert:True
```

Set the preference to `False` if you do not want to automatically convert datalog files when you open them.

Compatibility Issues between the LabVIEW 5.*x* VI Server and a LabVIEW 7.1 Client

Attempting to make a connection to the VI Server of a LabVIEW 5.*x* application from a LabVIEW 7.1 client fails because the LabVIEW 5.*x* application does not recognize some aspects of the LabVIEW 7.1 VI Server protocol.

You can connect to the VI Server of a LabVIEW 7.1 application from a LabVIEW 5.*x* client.

UDP Functions

Use the built-in UDP functions for network communication. The UDP VIs exist as compatibility VIs in the `vi.lib\_oldvers\_oldvers.llb`.

Upgrading from LabVIEW 4.*x*

You might encounter the following issues when you upgrade to LabVIEW 7.1 from LabVIEW 4.*x*. Refer to the *Upgrading from LabVIEW 5.x*, *Upgrading from LabVIEW 6.x*, and *Upgrading from LabVIEW 7.0* sections of this document for information about other upgrade issues you might encounter.

Refer to the *LabVIEW Upgrade Notes* for each version of LabVIEW between version 4.*x* and 7.1 and to the *LabVIEW 5.1 Addendum* at

ni.com/manuals for more information about the new features and changes in each version.

Converting Boolean Data to and from LabVIEW 4.*x*

The format in which LabVIEW stores Boolean data changed between LabVIEW 4.*x* and LabVIEW 5.*x*. LabVIEW 4.*x* stores Boolean data in two bytes unless the data is in an array, in which case LabVIEW 4.*x* stores each Boolean element in a single bit. LabVIEW 7.1 stores a Boolean value in a single byte, regardless of whether it is in an array. This change enables more block diagram functions to support arrays of Boolean values and makes the behavior of these arrays more consistent with the behavior of arrays of numbers. The LabVIEW 7.1 Boolean data format affects data manipulation in Code Interface Nodes (CINs), but LabVIEW 7.1 provides compatibility for existing CINs.

If you write binary data that includes one or more Boolean values to a file in LabVIEW 4.*x*, its format is different than if you write the same data in LabVIEW 7.1. LabVIEW 7.1 provides a mechanism for reading binary data written in LabVIEW 4.*x* and writing binary data that LabVIEW 4.*x* can read. The Write File, Read File, Type Cast, Flatten To String, and Unflatten From String functions have a **Convert 4.x Data** shortcut menu item. If you right-click the function and select this menu item, the function treats binary data as if it were written for LabVIEW 4.*x*. To produce data formatted for LabVIEW 4.*x*, use the Write File, Flatten To String, or Type Cast function. To read data formatted for LabVIEW 4.*x*, use the Read File, Unflatten From String, or Type Cast function. When you select the **Convert 4.x Data** shortcut menu item, LabVIEW 7.1 draws a red `4.x` on the function to indicate that it is converting data to or from LabVIEW 4.*x* format. To avoid the conversion of data, select the **Convert 4.x Data** shortcut menu item again to remove the checkmark next to it.

If you have several data files with Boolean values, you can create a VI that opens these files and writes the data to a new data file that LabVIEW 7.1 recognizes.

In LabVIEW 7.1, when you load a VI last saved in LabVIEW 4.*x* or earlier, LabVIEW 7.1 automatically sets the **Convert 4.x Data** attribute on the Write File, Read File, Type Cast, Flatten To String, and Unflatten From String functions. These functions continue to operate as before. When you decide that a VI needs to use the LabVIEW 7.1 Boolean data format, select the **Convert 4.x Data** shortcut menu item on each of these functions. Typically, you should use the LabVIEW 7.1 Boolean data format if VIs do not need to manipulate files that contain Boolean data written in LabVIEW 4.*x* or earlier and do not send or receive data that contain Boolean data to or from VIs running in LabVIEW 4.*x* or earlier. Support for

the previous Boolean data format might be discontinued in future versions of LabVIEW.

VI Control VIs

The VI Control VIs do not appear on the **Functions** palette but exist as compatibility VIs in the `vi.lib\utility\victl.llb`. Use the VI Server functions Open VI Reference, Call By Reference, Property Node, and Invoke Node instead of the VI Control VIs.

Some of the error codes the VI Control VIs return are different in LabVIEW 7.1. In previous versions of LabVIEW, the VI Control VIs returned the error codes 7 and 1000. The VI Control VIs in LabVIEW 7.1 return the codes 1004 and 1003. If a VI built in LabVIEW 4.*x* checks for error codes 7 and 1000, you must modify the VI to work in LabVIEW 7.1.

DDE VIs

(Windows) The DDE VIs do not appear on the **Functions** palette but exist as compatibility VIs in the `vi.lib\platform\dde.llb`.

Upgrading from LabVIEW 3.*x* or Earlier Versions

Refer to the National Instruments Web site at `ni.com` for information about using a VI conversion kit to upgrade from LabVIEW 3.*x* or earlier. Refer to the *Upgrading from LabVIEW 4.x*, *Upgrading from LabVIEW 5.x*, *Upgrading from LabVIEW 6.x*, and *Upgrading from LabVIEW 7.0* sections of this document for information about other upgrade issues you might encounter.

Refer to the *LabVIEW Upgrade Notes* for each version of LabVIEW between version 4.*x* and 7.1 and to the *LabVIEW 5.1 Addendum* at `ni.com/manuals` for more information about the new features and changes in each version.

LabVIEW 7.1 Features

Refer to the *LabVIEW Help* for more information about LabVIEW 7.1 features.

Using a Radio Buttons Control

Use the radio buttons control to give users a list of items from which they can select only one item at a time. If you want to give users the option to select none or one item, right-click the control and select **Allow No Selection** from the shortcut menu to place a checkmark next to the menu item. Because the data type of a radio buttons control is an enumerated

type, you can use the radio buttons control to select the cases of a Case structure.

Refer to the Radio Buttons Control VI and the Radio Buttons with Event Structure VI in the `labview\examples\general\controls\booleans.llb` for examples of using a radio buttons control.

Analyze VIs Enhancements

The Analyze VIs include the following enhancements:

- Use of the BLAS/LAPAK algorithm to decrease computation time for linear algebra operations
- Two new palettes for performing polynomial operations
- Polymorphic linear algebra and frequency domain VIs
- New linear algebra and time domain VIs

Polynomial VIs and Rational Polynomial VIs

Use the Polynomial VIs to perform calculations and evaluations with polynomials. Use the Rational Polynomial VIs to perform calculations and evaluations with rational polynomials.

Linear Algebra Palette

The **Linear Algebra** palette no longer contains the **Advanced Linear Algebra**, **Complex Linear Algebra**, and **Advanced Complex Linear Algebra** subpalettes. The VIs that were on these palettes are now polymorphic instances of the VIs on the **Linear Algebra** palette. The following table shows the removed VIs and their new locations.

LabVIEW 7.0 VI	LabVIEW 7.1 VI	Polymorphic Instance
A x Vector	A x B	A x Vector
Complex A x B	A x B	Complex A x B
Complex A x Vector	A x B	Complex A x Vector
Complex Cholesky Factorization	Cholesky Factorization	Complex Cholesky Factorization
Complex Conjugate Transpose Matrix	Transpose Matrix	Complex Conjugate Transpose Matrix
Complex Determinant	Determinant	Complex Determinant
Complex Dot Product	Dot Product	Complex Dot Product

LabVIEW 7.0 VI	LabVIEW 7.1 VI	Polymorphic Instance
Complex EigenValues and Vectors	EigenValues and Vectors	Complex EigenValues and Vectors
Complex Inverse Matrix	Inverse Matrix	Complex Inverse Matrix
Complex LU Factorization	LU Factorization	Complex LU Factorization
Complex Matrix Number	Matrix Number	Complex Matrix Number
Complex Matrix Norm	Matrix Norm	Complex Matrix Norm
Complex Matrix Rank	Matrix Rank	Complex Matrix Rank
Complex Matrix Trace	Matrix Trace	Complex Matrix Trace
Complex PseudoInverse Matrix	PseudoInverse Matrix	Complex PseudoInverse Matrix
Complex Outer Product	Outer Product	Complex Outer Product
Complex QR Factorization	QR Factorization	Complex QR Factorization
Create Special Complex Matrix	Create Special Complex Matrix	Create Special Complex Matrix
Solve Complex Linear Equations	Solve Linear Equations	Solve Complex Linear Equations (single right hand), Solve Complex Linear Equations (multiple right hand)
SVD Factorization	SVD Decomposition	Real SVD Decomposition
Test Complex Positive Definite	Test Positive Definite	Test Complex Positive Definite

The **Linear Algebra** palette contains the following new VIs:

- Use the Back Transform Eigenvectors VI to transform the eigenvectors of a real or complex balanced matrix to those of the original matrix.
- Use the Create Real Matrix From EigenValues VI to generate a real matrix from a set of eigenvalues.
- Use the Generalized Eigenvalues and Vectors VI to compute the generalized right eigenvalues and eigenvectors of a real or complex matrix pair.
- Use the Hessenberg Decomposition VI to perform the Hessenberg decomposition of a real or complex matrix.
- Use the Matrix Balance VI to balance a real or complex general matrix to improve the accuracy of computed eigenvalues and eigenvectors.

- Use the Matrix Characteristic Polynomial VI to compute the characteristic polynomial of a real matrix.
- Use the Matrix Exp VI to compute the exponential of a real or complex matrix.
- Use the Matrix Logarithm VI to compute the natural logarithm of a real or complex matrix.
- Use the Matrix Power VI to compute the *n*th power of a real or complex matrix.
- Use the Matrix Square Root VI to compute the square root of a real or complex matrix.
- Use the QZ Decomposition VI to perform the QZ decomposition of a pair of real or complex square matrixes.
- Use the Schur Decomposition VI to perform the Schur decomposition of a real or complex square matrix.
- Use the SVD Decomposition VI to compute the singular value decomposition of an $m \times n$ real or complex matrix.
- Use the Sylvester Equations VI to solve the Sylvester matrix equation for a real or complex matrix.
- Use the Transpose Matrix VI to transpose a real or complex matrix.

Time Domain Palette

The **Time Domain** palette contains the following new VIs:

- Use the Resample (constant to constant) VI to resample a single input signal or multiple input signals according to the values you specify for the sampling interval and time stamp.
- Use the Resample (constant to variable) VI to resample a single input signal or multiple input signals according to the time instance you specify.

Frequency Domain Palette

In LabVIEW 7.0 and earlier, the **Frequency Domain** palette contained the Real FFT, Inverse Real FFT, Complex FFT, and Inverse Complex FFT VIs. In LabVIEW 7.1, the **Frequency Domain** palette contains polymorphic FFT and Inverse FFT VIs. The FFT VI can process real or complex signals. The Inverse FFT VI can compute the inverse real FFT or the inverse complex FFT.

Storage VIs

(Windows) Use the Storage VIs to read and write waveforms and waveform properties to NI Test Data Exchange Format (`.tdm`) files. Use `.tdm` files to exchange data between NI software, such as LabVIEW and DIAdem.

The Storage VIs combine waveforms and waveform properties to form channels. A channel group organizes a set of channels. A file includes a set of channel groups. In addition to numeric values, the Storage VIs support arrays of strings and arrays of time stamps. A reference number represents files, channel groups, and channels on the block diagram.

You also can use the Storage VIs to query files to obtain channel groups or channels that meet conditions you specify.

Refer to the examples in the `labview\examples\file\storage.llb` for examples of using the Storage VIs.

Append Signals Express VI

Use the Append Signals Express VI to append signals to each other. Refer to the Append Signals VI in the `examples\express` directory for an example of using the Append Signals Express VI.

Bluetooth VIs and Functions

(Windows) Use the Bluetooth VIs and functions to communicate with devices that use the Bluetooth communication protocol. LabVIEW supports only Bluetooth devices that use the Microsoft Bluetooth driver on Windows XP Service Pack 1 or later. Refer to the Microsoft Windows Catalog Web site at `www.microsoft.com` for more information about Bluetooth devices that support the Microsoft Bluetooth driver. Most Bluetooth devices use a proprietary Bluetooth driver by default. To use the device with LabVIEW, you must switch to the Microsoft Bluetooth driver. To switch drivers, you must obtain the Microsoft Bluetooth driver CD from the manufacturer of the device.

Refer to the *Using LabVIEW with Wireless Devices* Application Note for more information about Bluetooth communication in LabVIEW. Refer to the Bluetooth Web site at `www.bluetooth.com` for more information about Bluetooth technology. Refer to the `labview\examples\comm\Bluetooth.llb` for examples of using the Bluetooth VIs and functions.

Saving Graphs, Charts, Tables, and Digital Data Controls as Images

Right-click a graph, chart, table, or digital data control or indicator and select **Data Operations»Export Simplified Image** from the shortcut menu to save an image of the control or indicator to the clipboard or as a `.emf` or `.bmp` file.

Using the Navigation Window

The **Navigation** window displays an overview of the active front panel in edit mode or the active block diagram. Use the **Navigation** window to navigate large front panels or block diagrams. Click an area of the image in the **Navigation** window to display that area in the front panel or block diagram window. You also can click and drag the image in the **Navigation** window to scroll through the front panel or block diagram. Portions of the front panel or block diagram that are not visible appear dimmed in the **Navigation** window.

Select **Window»Show Navigation Window** to display the **Navigation** window or press the <Ctrl-Shift-N> keys. **(Mac OS)** Press the <Command-Shift-N> keys. **(Sun)** Press the <Meta-Shift-N> keys. **(Linux)** Press the <Alt-Shift-N> keys.

Note The **Navigation** window is available only in the LabVIEW Full and Professional Development Systems.

Displaying Buffer Allocations

Select **Tools»Advanced»Show Buffer Allocations** to display the **Show Buffer Allocations** window. Place a checkmark next to the data type(s) for which you want to see buffers and click the **Refresh** button. Black squares appear on the block diagram to indicate where LabVIEW creates buffers to hold the data on the block diagram.

Note The **Show Buffer Allocations** window is available only in the LabVIEW Full and Professional Development Systems.

After you know where LabVIEW creates buffers, you might be able to edit the VI to reduce the amount of memory LabVIEW requires to run the VI. Refer to the *VI Memory Usage* section of the *LabVIEW Performance and Memory Management* Application Note for information about how LabVIEW allocates memory and for tips for using memory efficiently.

If you make a change to a VI that requires LabVIEW to recompile the VI, the black squares disappear because the buffer information might no longer be correct. Click the **Refresh** button on the **Show Buffer Allocations** window to recompile the VI and display the black squares. When you close the **Show Buffer Allocations** window, the black squares disappear.

Xmath Script Node

Use the Xmath Script Node to execute external Xmath scripts. You must have Xmath installed on your computer to use Xmath Script Nodes.

Use Xmath, part of the NI MATRIXx product family, to develop mathematical analysis, visualization, and scripting applications. Use Xmath for advanced data analysis, as a working environment for script development, and as a visualization tool for simulation data. Xmath includes a library of mathematical, system modeling, and analysis functions. Refer to `ni.com/matrixx` for more information about Xmath and NI MATRIXx.

Refer to the Xmath Script - Lorenz Diff Eq VI in the `labview\examples\scriptnode\Differential Equation.llb` and the Xmath Script - Fractal VI in the `labview\examples\scriptnode\Fractal.llb` for examples of using the Xmath Script Node.

LabVIEW and Hyper-Threading

Hyper-Threading is an advanced feature of high-end versions of the Intel Pentium 4 and later. A Hyper-Threaded computer has a single processor but acts like a computer with multiple processors. In text-based languages, to make an application multithreaded, you have to create multiple threads and write code to communicate among those threads. However, LabVIEW can recognize opportunities for multithreading in VIs, and the execution system handles multithreading communications for you. Refer to the *LabVIEW and Hyper-Threading* Application Note for more information about how LabVIEW executes code on a Hyper-Threaded computer.

Preparing Example VIs to Appear in the NI Example Finder

Select **Tools»Prepare Example VIs for NI Example Finder** to document example VIs you create to appear in the NI Example Finder. After you document the example VIs, build the data (`.bin3`) file that stores the task-based browse paths, keywords, and other information about the example VIs you document. LabVIEW uses the `.bin3` file to display example VIs in the NI Example Finder.

Note The example VIs you created and the `.bin3` file must reside in the `examples`, `instr.lib`, or `user.lib` directory. The example VIs also must reside in a directory structure identical to the directory structure in which they will reside on the user computer. The NI Example Finder does not locate the `.bin3` file if the `.bin3` file resides more than one directory level below the `examples`, `instr.lib`, or `user.lib` directory.

NI Example Finder Enhancements

Use the **Favorites** folder and the **Most Recent** folder on the **Browse** tab of the NI Example Finder to organize and easily access example VIs you use most frequently. You also can specify which example VIs to view based on hardware requirements. Refer to the *NI Example Finder Help* for more information about viewing example VIs by hardware device. Select **Refresh hardware list** from the **Hardware** pull-down menu the first time you launch the NI Example Finder and any time you add or remove a device to the **Hardware** pull-down menu.

Timed Loops

(Windows 2000/XP) A Timed Loop executes an iteration of the loop at the period you specify. Use the Timed Loop when you want to develop VIs with multi-rate timing capabilities, precise timing, feedback on loop execution, timing characteristics that change dynamically, or several levels of execution priority. Refer to the *Using the Timed Loop to Write Multirate Applications in LabVIEW* Application Note for more information about using the Timed Loop. Use the Timed Loop and DAQmx VIs and functions with the Timed Loop to develop VIs that control timing sources.

Refer to the `labview\examples\general\timedloop.llb` for examples of using the Timed Loop VIs and functions.

VI Server and Remote Front Panel Enhancements

Different versions of the LabVIEW browser plug-in can run concurrently, but they communicate with different remote front panel servers. Each browser plug-in can display VIs developed in the same version of LabVIEW. The HTML `OBJECT/EMBED` tag determines which plug-in the browser loads. For the LabVIEW ActiveX control, the `CLASSID` you specify in the `OBJECT` tag determines which plug-in to load. The `CLASSID` for each version of the plug-in is different. For the Netscape browser plug-in, the MIME type you specify in the `EMBED` tag determines which plug-in to load. Refer to the *LabVIEW Help* for more information about the syntax to use in LabVIEW 7.1. Refer to the *LabVIEW Help* in the previous versions of LabVIEW for more information about the syntax used in those versions.

New Properties and Methods

LabVIEW 7.1 includes the following new VI Server properties and methods:

- Allow No Selection property
- Get Diagram Image Scaled method
- Get Panel Image Scaled method
- Is Probe property

LabVIEW Companion Products CD

You can purchase several add-on software toolkits for developing specialized applications. All the toolkits integrate seamlessly in LabVIEW. Refer to the LabVIEW Companion Products CD included with LabVIEW and to the National Instruments Web site at `ni.com/toolkits` for more information about these toolkits.

Documentation Enhancements and Changes

Certain VI and function reference topics in the *LabVIEW Help* contain **Open example** and **Browse related examples** buttons. Click the **Open example** button to open the example VI to which the topic refers. Click the **Browse related examples** button to open the NI Example Finder and display related example VIs.

National Instruments recommends using the following Web browsers to view the *LabVIEW Help*:

- **(Mac OS)** Safari 1.0 or later
- **(UNIX)** Netscape 6.0 or later, or Mozilla 1.2 or later

Documents Not Revised

National Instruments did not revise the following documents for LabVIEW 7.1:

- *Getting Started with LabVIEW*
- *LabVIEW Development Guidelines*
- *LabVIEW Measurements Manual*
- *LabVIEW Quick Reference Card*
- *LabVIEW User Manual*
- *Using External Code in LabVIEW*

National Instruments did not revise the following Application Notes and white papers for LabVIEW 7.1:

- *Integrating the Internet into Your Measurement System—DataSocket Technical Overview*
- *Porting and Localizing LabVIEW VIs*
- *Using Apple Events and the PPC Toolbox to Communicate with LabVIEW Applications on the Macintosh*
- *Using LabVIEW to Create Multithreaded VIs for Maximum Performance and Reliability*

Saving VIs for Use in Previous Versions

In LabVIEW 7.1, you can save VIs for use in LabVIEW 7.0 by selecting **File»Save with Options** and selecting the **Save for Previous** option. If you want to save a VI for use in LabVIEW 6.1, you must open the VI in LabVIEW 7.0 and save it by selecting **File»Save with Options** and selecting the **Save for Previous** option.

Other LabVIEW 7.1 Features and Changes

LabVIEW 7.1 includes the following miscellaneous features and changes.

Application Builder Enhancements

Refer to the *LabVIEW Application Builder User Guide* for information about changes introduced between previous versions of the Application Builder and the current version.

Changes to Existing VIs and Functions

LabVIEW 7.1 includes the following changes to existing VIs and functions:

- The Unregister For Events function is located on the **ActiveX** palette.
- The **error message** input of the Error Cluster from Error Code VI is an optional input. Use **error message** to specify an error description for user-defined error codes.
- The behavior of the **ignore previous** input of the Wait on Notification and Wait on Notification from Multiple functions changed to the behavior of LabVIEW 6.1 and earlier versions. If you pass a TRUE value to the **ignore previous** input for the first iteration of a loop, then pass a FALSE value for successive iterations, the function ignores messages sent before the first iteration of the loop.
- The Formula Node supports the `atan2` function.

- The polymorphic Digital Comparison VI has two new instances, Waveform to Waveform and Table to Table. Use the Digital Comparison VI to compare a digital waveform to another digital waveform, a set of digital data to another set of digital data, a digital waveform to a specified value, or a set of digital data to a specified value.
- The Write LabVIEW Measurement File Express VI includes a **reset** input, which you can use to restart the writing to a LabVIEW measurement data file (`.lvm`).
- The Convert to Dynamic Data Express VI includes options to set the **Start Time** of the input signal to **Zero** or **Now**, the current system time.
- The **VI reference** input of the Call by Reference Node accepts wires from strictly typed VI references and static VI references. You also can use the Call by Reference Node to run a VI inside the static VI reference.
- The Savitzky Golai Filter PtByPt VI is now named Savitzky Golay Filter PtByPt VI.
- In LabVIEW 7.0, the (Incomplete) Beta Function VI completed the beta function when **a** = 1. In LabVIEW 7.1, for any nonpositive real value of upper limit **a** ≤ 1, the function is defined for all real nonnegative values of **x** and **y**.
- The following VIs were moved from the `vi.lib\analysis\8numeric.llb` to the `vi.lib\analysis\baseanly.llb`:
  - Normalize Vector VI
  - Normalize Matrix VI
  - Quick Scale 1D VI
  - Quick Scale 2D VI
  - Scale 1D VI
  - Scale 2D VI
  - Unit Vector VI
- The FIR Windowed Coefficients VI has a new input, **option,** which specifies whether to scale the **FIR Windowed Coefficients**. The default value is **not scaled**.
- The MSE VI behavior changed for LabVIEW 7.1. If the **Y values** and **X values** input arrays have different lengths, the MSE VI computes **mse** based on the sequence that contains the fewest elements and then returns a warning.

- The HP34401A Initialize VI has a new parameter, **Access Mode**, which sets advanced access options of the VISA Open VI. The default value is `0`. The **Instr Descriptor** parameter was renamed **VISA Resource**.
- The **Source** parameters were renamed to **Trigger Source** in the HP34401A App. Example VI, HP34401A Config Trigger VI, and HP34401A Getting Started VI.
- The **Instr Descriptor** parameter of the HP34401A Find Meter VI was renamed to **VISA Resource**.
- The Exponential Fit VI has a new parameter, **Standard Deviation**, which is the standard deviation, sigma[i] for data point (x[i],y[i]).
- The **name** input was removed from the Destroy Semaphore VI.

New, Moved, and Renamed Example VIs

The following example VIs were moved or renamed in LabVIEW 7.1:

- The `sndExample.llb` and `sndExample adv.llb` in the `examples\sound` directory have been renamed `sound.llb` and `sound adv.llb`, respectively.
- The TreeView VI is in the `examples\comm\axevent.llb`.

Refer to the **New Examples for LabVIEW 7.1** folder on the **Browse** tab of the NI Example Finder to view descriptions for and launch example VIs added to LabVIEW 7.1.

Miscellaneous

LabVIEW 7.1 includes the following miscellaneous changes:

- The **Skip navigation dialog on launch** checkbox on the **Miscellaneous** page of the **Options** dialog box was renamed to **Skip LabVIEW dialog box on launch**.
- The digital states X and Z have a new line style in the digital waveform graph control.
- When you probe 32-bit unsigned or signed integer or double-precision, floating-point values, you can select the radix on the **Condition** page of the probe. This change applies to the Conditional Double Probe, the Conditional Double Array Probe, the Conditional Signed32 Probe, the Conditional Signed32 Array Probe, the Conditional Unsigned32 Probe, and the Conditional Unsigned32 Array Probe.
- You can right-click a FieldPoint IO point control and select **Allow Undefined Names** from the shortcut menu to allow the control to accept FieldPoint item names you have not configured in Measurement & Automation Explorer (MAX) or another configuration utility.

- The **New and Changed in 7.0** pull-down menu item in the **Options** dialog box was renamed to **New and Changed in 7.x**.
- **(Linux)** When you install the LabVIEW Run-Time Engine on a target computer where you want to run an application that uses the Analyze VIs, enter `yes` when prompted to install the LabVIEW Run-Time Engine Analysis Support package (`labview71-rte-aal-7.1-1.i386.rpm`).
- You can right-click a subpanel control in the LabVIEW Full Development System and use the shortcut menu to configure the subpanel control. In LabVIEW 7.0, you could do so only in the LabVIEW Professional Development System.
- Performance of 2D array operations improved on Windows and Linux.
- The Target Operating System property has a new possible value, RTX.
- The VI Type property has a new possible value, SubSystem. LabVIEW returns the SubSystem value for subVIs for the LabVIEW Simulation Module that you can place only on a simulation diagram.
- The Web Publishing Tool now generates additional HTML code to determine what localized version of the LabVIEW Run-Time Engine to use to view an embedded VI.
- If wiring to a terminal would create a broken wire, the cursor does not change.
- The National Instruments Software License Agreement has changed. Refer to the *National Instruments Software License Agreement* located on the LabVIEW 7.1 installation CD for the licensing requirements.