LabVIEW™

Version 6.0

These upgrade notes describe the process of upgrading LabVIEW for Windows, Macintosh, and UNIX to version 6.0.

For installation instructions and other important information, read the *LabVIEW Release Notes*.

About These Upgrade Notes

This document includes information about upgrade issues you might encounter when you upgrade to LabVIEW 6.0 and information on new features.

For more information…

Refer to the *LabVIEW User Manual* and the *LabVIEW Help* for more information about these new features in LabVIEW 6.0. Access the *LabVIEW Help* by selecting **Help»Contents and Index**.

Contents

Upgrade Issues

If you are upgrading from LabVIEW 5.*x*, read *Converting VIs*, *Upgrading Applibs and Toolsets*, and *Upgrading from LabVIEW 5.x*.

If you are upgrading a version of LabVIEW 4.*x* or earlier version, read *Converting VIs*, *Upgrading Applibs and Toolsets*, and *Upgrading from LabVIEW 4.x*.

Converting VIs

Upgrading your LabVIEW application is an automated process. When you open a VI last saved in LabVIEW 4.0 or later, LabVIEW 6.0 automatically converts and compiles the VI.

You can estimate the amount of memory required to convert VIs by totalling the amount of memory your VIs and all their subVIs occupy on disk. If these 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 3 MB of memory to run LabVIEW.

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

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

Upgrading Applibs and Toolsets

Most existing toolsets work with LabVIEW 6.0 without problems. However, you must mass compile the VIs for use in LabVIEW 6.0. Refer to the *Converting VIs* section, earlier in this document for more information about mass compiling VIs. LabVIEW 6.0 is compatible with toolsets designed for LabVIEW 4.0 and later versions, with the following exceptions.

- **(Full Development System) LabVIEW Application Builder**—You must upgrade to LabVIEW Application Builder 6.0. The Professional

Development System version 6.0 includes the updated Application Builder libraries.

- **(Full Development System) LabVIEW Professional G Developers Toolkit**—If you have the Professional G Developers Toolkit 5.0 or later, you must upgrade to the LabVIEW Professional Development System version 6.0. This upgrade is free to existing users of the Professional G Developers Toolkit 5.1. The Professional Development System version 6.0 includes the new version of the Professional G Developers Toolkit.
- **LabVIEW Test Executive**—If you use LabVIEW Test Executive 5.1 or earlier, you must mass compile these VIs for use in LabVIEW 6.0. Refer to the *Converting VIs* section, earlier in this document for more information about mass compiling VIs.

Upgrading Previous Versions of LabVIEW

The following sections describe upgrade issues specific to different versions of LabVIEW.

Upgrading from LabVIEW 5.x

This section describes the issues you might encounter when you upgrade to LabVIEW 6.0 from LabVIEW 5.*x*.

Converting Datalog Files

When you open a datalog file created in an earlier version of LabVIEW, LabVIEW 6.0 prompts you to convert the file to the LabVIEW 6.0 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 6.0 returns an error and does not open the file.

To automatically convert datalog files when you open them, add the following line to any LabVIEW preference file, such as the `LabVIEW.ini` file.

```
silentDatalogConvert=True
```

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

Compatibility Issues between LabVIEW 5.x Server and LabVIEW 6.0 Client

Attempting to make a connection to the VI Server of a LabVIEW 5.*x* application from a LabVIEW 6.0 client fails because the LabVIEW 5.*x* does not recognize some new aspects of the LabVIEW 6.0 VI Server protocol.

You can connect to the VI Server of a LabVIEW 6.0 application from a LabVIEW 5.*x* client.

Changes with UDP in LabVIEW

The UDP VIs now are the UDP functions are available on the **Functions»Communication»UDP** palette.

Upgrading from LabVIEW 4.x

This section describes changes in the Boolean data format and the VI Control VIs since LabVIEW 4.*x*.

Converting Boolean Data to and from LabVIEW 4.x

The format in which data is stored 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 6.0 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 new Boolean data format affects data manipulation in code interface nodes (CINs), but LabVIEW 6.0 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 5.*x* and 6.0. LabVIEW 6.0 provides a mechanism for reading binary data written in LabVIEW 4.*x* and writing binary data that LabVIEW 4.*x* can read. Five functions—Write File, Read File, Type Cast, Flatten To String, and Unflatten From String—have a **Convert 4.x Data** shortcut menu item. If you 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 6.0 draws a red `4.x` on the function to indicate that it is converting data to or from LabVIEW 4.*x* format. To stop the conversion of data, disable the **Convert 4.x Data** shortcut menu item by selecting it again.

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 6.0 recognizes.

In LabVIEW 6.0, when you load a VI last saved in LabVIEW 4.*x* or previous versions, LabVIEW 6.0 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 function as before. When you decide that your VIs need to use the new LabVIEW 6.0 Boolean data format, disable the **Convert 4.x Data** menu item by selecting it again on each of the functions listed above. Typically, if your VIs do not need to manipulate files that contain Boolean data written in a previous version of LabVIEW or send or receive data that contain Boolean data to or from VIs running in a previous version of LabVIEW, use the new LabVIEW 6.0 Boolean data format. Support for the previous Boolean data format might be discontinued in future versions of LabVIEW.

Converting Datalog Files

Refer to *Converting Datalog Files* in the *Upgrading from LabVIEW 5.x* section earlier in this document for more information about converting datalog files when upgrading.

VI Control VIs

The VI Control VIs (`vi.lib\utility\victl.llb`) have been removed from the default palette set and now exist as compatibility VIs. Their functionality has been included in the VI Server functions Open VI Reference, Call By Reference, Property Node, and Invoke Node on the **Functions»Application Control** palette.

Some of the error codes passed from the VI Control VIs have changed in LabVIEW 6.0. In previous versions of LabVIEW, the VI Control VIs passed the error codes 7 and 1000. The VI Control VIs in LabVIEW 6.0 pass the codes 1004 and 1003. If a VI built in LabVIEW 4.*x* checks for error codes 7 and 1000, you need to make modifications so the VI works in LabVIEW 6.0.

DDE VIs

The DDE VIs (`vilib\platform\dde.llb`) have been removed from the default palette set and now exist as compatibility VIs.

Upgrading from LabVIEW 3.x or Earlier Versions

For information on upgrading from LabVIEW 3.x or earlier, refer to the National Instruments web site, `www.ni.com`

LabVIEW 6.0 Features

This section describes features and improvements in LabVIEW 6.0.

Note Refer to the *LabVIEW 5.0 Upgrade Notes*, part number 321780A-01, available on the National Instruments web site for feature information in previous versions of LabVIEW.

Waveform Data Type

The VIs, functions, and front panel objects you use to build VIs that acquire, analyze, and display measurements accept or return the waveform data type by default. An array of waveform data types represents multiple waveforms.

Note Previous version of LabVIEW represented waveforms as numeric arrays and clusters. If you built VIs that create and access waveforms with LabVIEW 5.*x* or earlier you can continue to manipulate waveforms as arrays and clusters or convert the VIs to use the waveform data type.

The waveform data type contains data associated with a single waveform, including data values and timing information.

The waveform data type passes the waveform components to the VIs and functions you use to build measurement applications. Use the waveform VIs and functions to extract and edit the components of the waveform.

Waveform Data Type Components

The waveform data type is a special cluster of components that includes waveform information only. Each component of the waveform is an element in the cluster, and a data type represents each component. Use the waveform functions to access and manipulate individual components.

Start Time (T_0)

The timestamp is the start time of the first point in the waveform. The waveform graph uses the timestamp when plotting data. The default is 0, which represents the LabVIEW system time of 12:00 a.m., January 1, 1904, GMT.

Delta t (dt)

Delta *t* is the time between successive data points in the waveform.

Waveform Data (Y)

The waveform data is the 1D array of numbers that represents a waveform. Each element in the array is a data value representing the amplitude of the waveform. Generally, the number of data values in an array corresponds directly to the number of scans taken from a data acquisition device.

Waveform Attributes

Waveform Attributes provide information about the waveform data, such the channel or device from which the data came.

Waveform Data on the Front Panel

On the front panel, waveform data can be represented by the **Waveform** control located on the **Controls»I/O** palette.

Use the **Waveform** control to manipulate the **t_0**, **dt**, **Y,** and **attribute** components of the waveform data type or display those components as an indicator. To access the attributes, right-click the **Waveform** control and select **Visible Items»Attributes** from the shortcut menu.

Controls in the **Controls»Waveforms** palette also can display waveforms.

Waveform VIs and Functions

The **Functions»Waveform** palette includes VIs and functions that let you create and manipulate waveforms and manipulate the individual components of a waveform. Many of the VIs available on the **Functions»Data Acquisition** and **Functions»Analyze** palettes return and accept the waveform data type.

I/O Name Controls

Use the I/O Name controls on the **Controls»I/O** or **Controls»Classic»I/O** palettes to select DAQ channel names, VISA resource names, and IVI logical names. Configure these names in Measurement & Automation Explorer (MAX). Use the name controls with the I/O VIs to communicate with an instrument or a data acquisition device. Select **Tools»Measurement & Automation Explorer** for more information on configuring DAQ channel names, VISA resource names, and IVI logical names using MAX.

I/O Name constants are available on the **Functions»Instrument** palette and **Functions»DAQ** palette.

MAX is available on Windows only. **(Macintosh)** Use the DAQ Channel Wizard to configure DAQ channel names. **(Macintosh and UNIX)** Use the VISA configuration utilities to configure VISA resource names.

DataSocket Improvements

Use National Instruments DataSocket technology to share live data with other VIs and other applications, such as National Instruments ComponentWorks, on the Web or on your local computer. DataSocket pulls together established communication protocols for measurement and automation in much the same way a Web browser pulls together different Internet technologies.

DataSocket technology provides access to several input and output mechanisms from the front panel through the **DataSocket Connection** dialog box or from the block diagram with the DataSocket Read and Write functions. You publish (write) or subscribe to (read) data by specifying a URL, in much the same way you specify URLs in a Web browser.

For example, if you want to share the data in a thermometer indicator on the front panel with other computers on the Web, publish the thermometer data by specifying a URL in the **DataSocket Connection** dialog box. Users on other computers subscribe to the data by placing a thermometer on their front panel and selecting the URL in the **DataSocket Connection** dialog box.

This palette also includes the Variant functions. Refer to the *Variant Data* section later in this document for more information about variant data in LabVIEW.

User Interface Features

The following describes the new user interface and ease of use features.

Menu Reorganization

The menus have been reorganized for LabVIEW 6.0 to place related features together and to make the menus more intuitive.
selecting **Help»Contents and Index**.

Changes to the File Menu

- **Print**, **Project»Documentation Tool**, and **Print Documentation** have been combined into the new **Print** dialog box, available by selecting **File»Print**.
- Selecting **File»New** opens the **New** dialog box, which allows you to create new VIs, global variables, controls, run-time menus, polymorphic VIs, build scripts, VI templates, global variable templates, and control templates. You also can use the **New** dialog box to create a new file based on an existing template.
- **Edit Template** has been removed from the **File** menu.

- Selecting **File»Open** opens the **Open** dialog box, which allows you to open any LabVIEW file. When you open a template (`.vit` or `.ctt` file), LabVIEW retains the template file extension when you save the file. Use the **Start from template** option in the **File»New** dialog box to create a new VI, global variable, or control based on an existing template.
- **Save A Copy As** has been removed. The **Save As** dialog box includes a checkbox that allows you to save a copy without updating the callers.
- **VI Setup** and **VI Info** have been moved from the **Window** menu and combined into the new **VI Properties** dialog box, available by selecting **File»VI Properties**.
- **Edit VI Library**, **Mass Compile**, **Convert CVI FP File**, and **Update VXIplug&play Drivers** have been moved to the **Tools** menu.

Changes to the Edit Menu

- **Edit Control** has been renamed **Customize Control**.
- **Edit Menu** has been renamed **Run-Time Menu**.
- **Find** and **Show Search Results** have been added from the **Project** menu.
- **Preferences** has been renamed **Options** and moved to the **Tools** menu.
- **User Name** has been moved to the **Tools** menu.
- **Clear Password Cache** has been removed. The **Miscellaneous** page of the **Options** dialog box includes a **Clear Password Cache** button.

Project Menu Renamed the Tools Menu

The **Project** menu has been renamed the **Tools** menu.The contents of the **Tools** menu have been modified as follows:

- **(Windows) Measurement & Automation Explorer** accesses the Measurement & Automation Explorer (MAX). Use MAX to configure the instruments and data acquisition hardware connected to your computer.
- An **Instrumentation** pull-right menu has been added and contains the **Instrument Driver Network** and **Import CVI Instrument Driver** items.
- The **DAQ Wizards** pull-right menu has been renamed **Data Acquisition** and contains the **DAQ Channel Viewer** and **DAQ Solution Wizard** items.

  (Macintosh) The **Data Acquisition** menu also includes the **DAQ Channel Wizard**.
- **Documentation Tool** has been moved to the **Print** dialog box, available by selecting **File»Print**.

- **File Manager** has been renamed **VI Library Manager**.
- **VI Revision History** has been moved from the **Window** menu.
- **Show VI Hierarchy**, **This VI's Callers**, **This VI's SubVIs**, **Unopened SubVIs**, and **Unopened Type Defs** have been moved to the **Browse** menu.
- **Find** and **Show Search Results** have been moved to the **Edit** menu.
- **Show Profile Window** has been renamed **Profile VIs** and moved to the **Advanced** pull-right menu.
- **Edit VI Library** has been moved from the **File** menu.
- An **Advanced** pull-right menu has been added and contains **Mass Compile**, **Export Strings**, **Import Strings**, **Import ActiveX Controls**, and **Profile VIs**.
- A **Compare** pull-right menu has been added that groups all the compare functions. This item is available only to users who have the Professional Development System configuration of LabVIEW.
- For users with the Application Builder, the **Build Application** item has been renamed **Build Application or Shared Library (DLL)**.

Addition of the Browse Menu

This is a new menu. The **Browse** menu contains items that allow you to view aspects of the current VI and its hierarchy.

- **Show VI Hierarchy** has been moved from the **Project** menu.
- **This VI's Callers** has been moved from the **Project** menu.
- **This VI's SubVIs** has been moved from the **Project** menu.
- **Unopened SubVIs** has been moved from the **Project** menu.
- **Unopened Type Defs** has been moved from the **Project** menu.

Changes to Window Menu

- **Show VI Info** has been moved to the **File** menu and renamed **VI Properties**.
- **Show VI History** has been renamed **VI Revision History** and moved to the **Tools** menu.

Changes to Help Menu

- **Show Help** has been renamed **Show Context Help**.
- **Lock Help** has been renamed **Lock Context Help**.
- **Simple Help** has been removed from the **Help** menu. To toggle the **Context Help** window between simple and complete help, click the

simple help button on the lower left corner of the **Context Help** window.

- **Online Reference** has been renamed **Contents and Index**.
- **Online Help for (the current) VI** has been renamed **Help for This VI**.
- **Search Examples** has been renamed **Examples**.
- **LabVIEW Manuals** has been renamed **View Printed Manuals**.
- **Online Tutorial** has been removed from the **Help** menu. You can access the *LabVIEW Tutorial* by clicking the **LabVIEW Tutorial** button on the LabVIEW navigation dialog box.

Reorganization of Shortcut Menus

The shortcut menus have been reorganized for LabVIEW 6.0. Commonly used features are organized on the top level of the menus, and advanced features are located in the **Advanced** pull-right menu.

- **Show** has been renamed **Visible Items**.
- **Change to Control** and **Change to Indicator** have been added to the constant shortcut menu.
- **Change to Constant** has been added to the front panel terminal shortcut menu.
- Several items have been removed from the **Data Operations** pull-right menu and added to the new **Advanced** pull-right menu.
  - **Key Navigation** has been moved to the **Advanced** pull-right menu.
  - **Synchronous Display** has been moved to the **Advanced** pull-right menu.
  - **Online Help** has been renamed **Help** and moved to the **Advanced** pull-right menu for all terminals that have an **Advanced** pull-right. **Help** is located at the top level of the shortcut menu for terminals that do not have an **Advanced** pull-right menu.
  - **Customize** has been added to the **Advanced** pull-right menu.
- The **Description** item under **Data Operations** has been renamed **Description and Tip** and is now located at the top level of the shortcut menu.
- **Set Breakpoint** and **Clear Breakpoint** have been added to the top level of the shortcut menu.

Cluster and Array Shortcut Menu Options

Several items have been added to the shortcut menus for clusters and arrays.

- **Advanced»Hide Control** hides the cluster or array control.
- **Advanced»Enabled State** dims, disables, and enables controls.
- **Advanced»Show Hidden Element** shows any hidden item in a cluster.
- **Austosizing** resizes a cluster to **Size to Fit**, **Arrange Horizontally**, or **Arrange Vertically**. When you auto-arrange a cluster horizontally or vertically, the items within the cluster do not overlap.
- **Advanced»Insert Element Before** and **Delete Element** inserts and deletes array elements in a 1D array.
- **Advanced»Insert Row Before**, **Insert Column Before**, **Delete Row**, and **Delete Column** inserts and deletes rows and columns of array elements in a 2D array.

Better Navigation of Controls and Functions Palettes

A navigation bar on the **Controls** and **Functions** palettes allows you to navigate the palettes similarly to how you use the navigation buttons in a web browser. When you click a subpalette button, the entire palette changes to the selected subpalette. The navigation bar also includes a search feature to help you find palette items faster.

The **Controls** and **Functions** palettes contain the following navigation buttons:

- **Up**—takes you up one level in the palette hierarchy.
- **Search**—changes the palette to search mode. In search mode, you can perform text-based searches to locate controls, VIs, or functions in the palettes.
- **Options**—opens the **Palette Options** dialog box, from which you can configure the appearance of your palettes.

Searching for Controls, VIs, and Functions on the Palettes

Use the **Search** button to perform text-based searches for any function, VI, or control on the **Controls** and **Functions** palettes. You can search by the beginning of the name (**Begins With**) or keywords (**Contains**). Double-click the control, VI, or function to go to its location on the palette.

Automatic Wiring

You can use the automatic wiring feature to easily wire objects as you place them on the block diagram. You also can automatically wire objects already on the block diagram. When you use automatic wiring, LabVIEW connects the terminals that best match and leaves non-matching terminals unconnected.

With automatic wiring, as you move a selected object close to other objects on the block diagram, LabVIEW draws temporary wires to show you valid connections. When the wires shown are correct, you can drop the object and LabVIEW automatically connects the wires.

You can toggle automatic wiring on and off by pressing the space bar while an object is being positioned. By default, automatic wiring is enabled when you select an object from the **Functions** palette, or when you use `<ctrl>`+drag to create a copy of an object already on the block diagram. Automatic wiring is disabled by default when you use the Positioning tool to move an object already on the block diagram.

When automatic wiring is on, the selected object retains its appearance when you drag it. When automatic wiring is off, the selected object appears as a dotted outline when you drag it.

New Dialog Box

Use the **New** dialog box to create any kind of LabVIEW file. You can create new VIs, polymorphic VIs, controls, global variables, run-time menus, VI templates, control templates, and global variable templates. You also can use the **New** dialog box to create new files based on existing templates.

Print Dialog Box

Use the **Print** dialog box to print documentation for VIs, controls, global variables, or templates and to save documentation to HTML, RTF, or TXT files. You also can customize which documentation to print and configure page setup properties.

Better Graph Printing in LabVIEW

LabVIEW 6.0 prints graphs and chart using printer resolution. This mainly affects standard and PostScript printing by printing clearer graphs with distinct lines and images.

VI Properties Dialog Box

Use the **VI Properties** dialog box to configure many aspects of a VI, such as the following:

- How the VI icon appears
- How and where the front panel appears when you call the VI

- How the VI runs when you call it
- What text appears in the **Context Help** window and if you use the VI History options
- If the VI is password protected

To access the **VI Properties** dialog box, right-click the VI icon on the block diagram or front panel and select **VI Properties** from the shortcut menu or select **File»VI Properties**.

The **VI Properties** dialog box combines many of the features formerly in the **VI Setup** dialog box and the **VI Information** dialog box.

Color Picker

The color picker, available by right-clicking an object or background using the Color tool, has been replaced by a wide spectrum color picker. The top window of the color picker contains a gray scale. The middle window contains a spectrum of muted colors that are well-suited to backgrounds and front panel objects. The third window contains a spectrum of colors that are well-suited to highlights. The bottom window contains the placeholders for the 10 most recently selected colors and a placeholder for transparency.

The window to the right of the color placeholders shows the currently selected color. Objects that have foreground and background colors display the foreground color on the left half and the background color on the right half of the color indicator. To color only the foreground or background of an object, hold down the `<f>` or `<b>` keys, respectively, as you select the color.

(Windows and Macintosh) Click the **More** button to access the operating system **Color** dialog box, from which you can select a specific color using hue, saturation, luminosity and red, green, and blue values.

(UNIX) Click the **More** button access a LabVIEW dialog box from which you can select a color by defining red, green, and blue values.

Front Panel Controls and Indicators

The **Controls** palette has been reorganized and updated with new controls and indicators.

3D Controls and Indicators

Many of the front panel objects have been redesigned to have a more modern, three-dimensional appearance. The functionality and behavior of the objects have not changed—including the ability to color and resize the object. Your monitor should be set to display at least 16-bit color for optimal appearance of the controls.

Reorganization of Controls Palette

The Controls palette has been reorganized to accommodate new features in LabVIEW 6.0.

- The new **I/O** palette contains the new DAQ, VISA, and IVI I/O Name controls and the IMAQ Session control.
- The **String & Path** palette contains the string and path controls.
- The **Ring & Enum** palette contains the ring and enum controls, including the Menu Ring.
- The new **List & Table** palette contains the table and listbox controls.
- The original front panel objects are still available from the **Controls»Classic** palette. These objects are best-suited for use with 256-color and 16-color monitors.

Tab Controls

Use tab controls to overlap front panel controls and indicators in a smaller area. A tab control consists of pages and tabs. Place front panel objects on each page of a tab control and use the tab as the selector for displaying different pages. There is no limit to the number of front panel objects you can place on a tab control. The tab control is available from the **Controls»Array & Cluster** and **Controls»Dialog** palettes, and from the corresponding **Controls»Classic** palettes.

Tab controls are useful when you have several front panel objects that are used together or during a specific phase of operation. For example, you may have a VI that requires the user to first configure several settings before a test can start, then allows the user to modify aspects of the test as it progresses, and finally displays and stores specific data. You can use a tab control to show only the controls and indicators for each phase of the VI.

On the block diagram, the tab control is an enumerated control by default. Terminals for controls and indicators placed on the tab control appear as any other block diagram terminal.

Multicolumn Listbox

Use the Multicolumn listbox, available in the **Controls»List and Tables** palette, and the Dialog Multicolumn Listbox, available in the **Controls»Dialog Controls** palettes, to present users with a list of options and information about each option. The multicolumn listbox is like the single-selection and multi-selection listboxes, but the multicolumn listbox includes columns where you can add more information for each option, such as the size of the option and the date it was created. You can set the number of columns and create headers for each column. You can create a multiselection or single selection multicolumn listbox.

Multiple X- and Y-Scales on Graphs and Charts

Graphs and charts now support multiple x- and y-scales. Using multiple scales allows you to display multiple plots that do not share a common x- or y-scale.

Graph and Chart Scale Legend

A scale legend has been added to waveform graphs and charts to accommodate multiple scales. You use the scale legend to label scales, as well as configure scale properties.

When you use the Operating tool and press the **Scale Format** buttons, you can configure the format, precision, and mapping mode; toggle the visibility of scales, scale labels, and plots; and format scale labels, grids, grid lines, and grid colors.

Anti-Aliased Line Plots for Graphs and Charts

You can improve the appearance of line plots in your charts and graphs by using anti-aliased lines. When you enable anti-aliased line drawing, your line plots use shades of color to appear smoother. Anti-aliased line drawing does not alter line widths, line styles, point styles, and so on.

Note Anti-aliasing is *not* available on the Waveform Digital Graph.

If the plot legend is not visible, right-click the chart or graph and select **Visible Items»Plot Legend**, then right-click the legend and select **Anti-aliased**.

Note Anti-aliased line drawing is computation-intensive. Using anti-aliased line plots might affect VI performance.

Waveform Digital Graph

Use the Digital Waveform Graph available from the **Controls»Graphs and Charts** palette to display digital data. This is useful when you are working with timing diagrams or logic analyzers.

Color Ramps for Dials, Knobs, Meters, and Gauges

You can add a color ramp to any dial, knob, meter, or gauge on your front panel. Color ramps are useful for visually indicating data ranges, such as a warning range for indicating when a gauge reaches a dangerous value.

General Front Panel and Block Diagram Improvements

Several changes have been made to improve the Front Panel and Block Diagram. The following sections detail the new features.

Grouping and Locking of Front Panel Objects

You can group front panel objects together and lock their position on the front panel. Grouped objects maintain their relative arrangement, while locked objects maintain their location on the front panel, and can not be deleted.

Documenting Front Panel and Block Diagram Objects

You can add custom descriptive information for individual front panel and block diagram objects. The information you enter in the **Description and Tip** dialog box appears as tip strips, in the **Context Help** window description for the object, and in the VI documentation you generate.

Tip strings are brief descriptions that appear on or near an object when you idle the cursor over the object. If a tip has not been entered, no tip string appears.

Note Tips are available only for controls, indicators, or constants. You cannot enter tip information for block diagram functions or VIs.

While Loop Improvements

Controlling the continuation of a While Loop has been simplified for LabVIEW 6.0. By default, a While Loop stops when a FALSE value is passed to its Conditional terminal. To change the behavior of the Conditional terminal, right-click the terminal or the border of the While Loop and select **Stop if True** or **Continue if True**. The default behavior is **Continue if True**.

Changes to Formula Nodes

You now can perform the following new operations in Formula Nodes:

- integer bitwise operations (`&`, `|`, `~`, `<<`, `>>`, and `^`)

Note The exponentiation operation from earlier versions of LabVIEW, `^`, has been replaced by `**` and updates automatically when you recompile VIs for LabVIEW 6.0. The `^` operation is now bitwise exclusive or.

- shortcut assignments (`+=`, `-=`, `*=`, `/=`, `>>=`, `<<=`, `&=`, `\=`, `^=`, `|=`, `%=`, and `**=`)
- pre- and post-increment (`++`) and decrement (`--`) operations

Formula Nodes also now implement type checking to ensure that array indexes are numeric data and that operands to the bit operations are integer data.

Array indices are checked to ensure they are in range. For arrays, an out of bounds value defaults to zero, and an out of bounds assignment defaults to `nop` to indicate no operation occurs.

Formula Nodes also perform automatic type conversion.

C Functionality in Formula Nodes

In addition to executing text-based equations, you can use Formula Nodes to execute many C functions. Using this C functionality, you can execute if statements, while loops, for loops, do loops, switch statements, break and continue statements, array manipulation, and compound statements with the scope rules of C programming.

Expression Nodes

Use the Expression Node, available in the **Functions»Numeric** palette, to calculate expressions, or equations, that contain a single variable. Expression Nodes are useful when an equation has only one variable but is otherwise complicated.

Expression Nodes use the value you pass to the input terminal as the value of the variable. The output terminal returns the value of the calculation.

Polymorphism in Expression Nodes

The input terminal of an Expression Node is the same data type as the control or constant you wire to it. The output terminal is the same data type as the input terminal. The data type of the input can be any non-complex scalar number, array of non-complex scalar numbers, or cluster of non-complex scalar numbers. With arrays and clusters, the expression node applies the equation to each element of an input array or cluster.

Rotated Text Labels

You can use rotated text labels to display rotated and stacked vertical text.

To rotate a text label, right-click the label and select **Vertical Arrangement**, and choose from one of the following options from the pull-right menu:

- **None**—label appears as normal text
- **Stacked**—label appears as stacked text
- **Clockwise**—label text is rotated 90 degrees clockwise. This results in vertical text that reads downward.
- **Counterclockwise**—label text is rotated 90 degrees counterclockwise. This results in vertical text that reads upward.

Windows Registry Access VIs

Use the Windows Registry Access VIs, available from the **Functions»Advanced»Windows Registry Access** palette, to create, open, query, enumerate, close, and delete Windows Registry keys. You also can enumerate, read, write, and delete the value of Windows Registry keys.

Caution Improperly modifying the Windows Registry can render your Windows operating system inoperable.

Control References

Control references allow you to pass front panel information to subVIs. Once a control reference has been passed to a subVI, you use Property Nodes and Invode Nodes to read and configure properties and invoke methods of the referenced front panel object.

Note Attribute nodes have been removed from LabVIEW. They have been replaced by Property Nodes that have an implicit control refnum input.

Attribute nodes in earlier versions of LabVIEW are replaced with Property Nodes in 6.0. Because of this, the data name of some attributes have changed. The following lists the new attribute names.

- Caption and Caption Visible properties
- Scroll Position, Selection, and Text Colors properties for string and path controls
- Ring Text Size, Ring Text Colors, and Numeric Text Colors properties for ring controls
- Numeric Text Colors, Ramp Visible, and Dig Display Visible properties for color ramp controls
- Most of graph/chart scale properties
- Most of graph/chart plot properties
- Most of graph cursor properties
- Most of slide scale properties

If you have put these attributes into a cluster and wire the cluster to Unbundle by Name, the bundler might appear broken because of the data name change. You must correct this manually. To correct the problem, right-click the Unbundle by Name node and reselect the appropriate property.

Implicitly Linked Property Nodes

When you create a Property Node from a front panel control, that Property Node is implicitly linked to the control. To find the referenced control right-click the Property Node and select **Find Control**, or double-click the Property Node to find the control.

If you select an implicitly linked Property Node and create a subVI by selecting **Edit»Create SubVI**, a control reference is created and wired to the subVI.

Attribute Nodes created in LabVIEW 5.*x* or earlier mutate to become implicitly linked Property Nodes by default.

Control Refnums

The **Controls»Refnum** palette contains a Control refnum you can use to pass/receive control references to/from subVIs. When you drop a control refnum on the front panel, its type is generic control reference. You configure it by dragging a specific front panel control to it. For example, to create a control refnum for a digital control, drop a Control refnum and drag a digital control onto it.

To create a generically typed control refnum, right-click the Control refnum and choose from the **Select VI Server Class** pull-right menu. To make a generically typed refnum strictly typed, right-click the refnum and select **Include Data Type**.

To make a front panel Control refnum appear as its referenced control, right-click the refnum and select **Show Control**. To revert back to the icon, right-click the refnum and select **Show Icon**.

Error Handling Improvements

Several changes have been made to improve error handling on the block diagram. Many functions and structures that accept Boolean data also can recognize the status Boolean parameter of the error cluster, allowing you to easily wire error clusters on your block diagram.

Error Handling with While Loops

When you wire an error cluster to the Continuation terminal of a While Loop, the terminal changes its functionality to accommodate error handling. The shortcut menu items **Stop if True** and **Continue if True** change to **Stop if Error** and **Continue while Error**.

Error Handling with Case Structures

When you wire an error cluster to the Selector terminal of a Case structure, the subdiagram display window shows two cases, **Error** and **No Error**. The border of the Case structure changes color—red for the **Error** case and green for the **No Error** case.

Additional Error Handling Functions

The Stop and Quit LabVIEW functions in the **Functions»Application Control** palette and the Select function in the **Functions»Comparison** palette also accept error clusters as inputs. When an error cluster is wired to one of these functions and an error occurs, a TRUE value is passed to the function.

The **Functions»Time & Dialog** palette includes the Merge Errors VI, which lets you merge error I/O clusters from different functions. The VI first looks for error clusters where the status is TRUE. The first error found is reported. If the VI finds no errors, it looks for warnings and returns the first warning found. If it does not find a warning, the VI returns an error status FALSE.

Error Handling Using Execution Highlighting

When you use execution highlighting to debug a VI, after each node executes it displays its value with a border that is the same color as its data type. When an error is reported from an error cluster, the Error value displayed has a red border. When no error is reported, the border of the **OK** value is green.

Building Shared Libraries (DLLs)

(Professional Development System) With LabVIEW 6.0, you can build a shared library (DLL) in much the same way you can build a stand-alone application. Select **Tools»Build Application or Shared Library (DLL)**. The **Build Application or Shared Library (DLL)** dialog box appears. This dialog box includes the following tabs:

- **Target**—Use this tab to name and choose where to save your shared library.

- **Source Files**—Use this tab to add and delete files from your shared library.
- **VI Settings**—Use this tab to edit various window and execution options for the VIs included in your shared library.
- **(Windows) Application Settings**—Use this tab to specify a custom icon for the DLL.
- **(Windows) Installer**—Use this tab to create an installer for your shared library. By default, LabVIEW does not create an installer.

Array Function Enhancements

LabVIEW includes three new functions for manipulating arrays—the Replace Array Subset, Insert Into Array, and Delete From Array functions, available in the **Functions»Arrays** palette. Many of the existing array functions are redesigned in LabVIEW 6.0 to make manipulating arrays easier. New array functionality includes the following:

- Automatic resizing of array functions based on dimension of the input array
- Grouping of inputs based on the array subset they are manipulating

String Function Enhancements

LabVIEW includes functions that let you search for and replace characters or substrings within in a string. The new Replace Substring and Search/Replace String functions, available in the **Functions»String** palette, let you replace an arbitrary portion of a string with a substring or replace a specified substring with another substring, respectively. The Split String function, available in the **Functions»String»Additional String Functions** palette, now lets you search for an arbitrary string, rather than a single character.

File I/O Functions

Use the Format Into File function to format string, numeric, path, and Boolean data as text and write the text to a file. Often you can use this function instead of separately formatting the string with the Format Into String function and writing the resulting string with the Write Characters to File VI or Write File function.

With the Format Into File function you can determine the order in which the data appears in the text file. However, you cannot use this function to append data to a file or overwrite existing data in a file. For these operations, use the Format Into String function with the Write File function.

Use the Scan from File function to scan text in a file for strings, numbers, paths, and Boolean values and then convert the text into a data type. Often you can use this function instead of reading data from a file with the Read File function or Read Characters from File VI and scanning the resulting string with Scan from String function.

The Scan from File function reads all the text in the file. However, you cannot use this function to determine the point in the file where the scanning starts. For this operation, use the Read Characters from File VI with the Scan from String function.

Polymorphic VIs

Polymorphic VIs accept different data types for a single input or output terminal. A polymorphic VI is a collection of subVIs with the same connector pane patterns. Each subVI is an instance of the polymorphic VI. If a terminal in the connector pane of one VI is an input, the corresponding terminal on the connector pane of the other VIs also must be an input or it must be unused.

For example, the Read Key VI, available in the **Functions»File I/O»Configuration File VIs** palette, is polymorphic. Its **default value** terminal accepts a Boolean, numeric double, 32-bit integer, path, string, or unsigned 32-bit integer data type.

Variant Data

The Variant data type stores data that might have been of a different data type when execution occurred, either in LabVIEW or another application. This type of data commonly is used with ActiveX functions and properties.

Use the Variant functions located on the **Functions»Advanced»Data Manipulation»Variant** palette to create and manipulate variant data. You can convert any LabVIEW data type to the variant data type to use variant data in other LabVIEW VIs and functions. For example, if you convert a string to variant data, the variant data type stores the text and indicates that the text is a string.

ActiveX Enhancements

LabVIEW 6.0 includes functionality that allows for easier access to ActiveX properties and methods.

Accessing ActiveX Properties and Methods

Create a new Property Node or Method Node for an ActiveX object from the front panel or block diagram by right-clicking the object and selecting **Create Property** or **Create Method** from the shortcut menu. You do not have to navigate through the **Functions** palette, select a Property Node or

Method Node, and place it on the block diagram each time you want to control an ActiveX object.

Multiple Platform and HTML Functionality in Reports

With LabVIEW 6.0, you can create HTML-based reports on all platforms. On the Windows platforms, you can create and print paper-based reports. You find these VIs in the **Functions»Report Generation** palette.

Note HTML-based report generation functionality is not available in Base version of LabVIEW 6.0.

Localization Enhancements

You can now export strings on the block diagram, such as labels and captions. This is useful if you are localizing VIs. You also can export strings associated with the following block diagram objects: string constants, path constants, ring constants, enumerated constants (font information only), array and cluster constants, and free labels.

Changes to the LabVIEW Documentation Set

National Instruments has revised the LabVIEW 6.0 documentation set from previous versions. Refer to the *LabVIEW Help*, available in the **Help»Contents and Index** menu, for information about specific LabVIEW features and procedures on how to use those features when building VIs. Refer to the *LabVIEW User Manual* for conceptual information about developing in the LabVIEW environment. Refer to the *LabVIEW Measurements Manual* for information about developing data acquisition and instrument control VIs. Refer to the LabVIEW Application Notes available in PDF for developing advanced VIs.

Refer to the *LabVIEW Documentation Resources* section of Chapter 1, *Introduction to LabVIEW*, in the *LabVIEW User Manual* for an overview of the documentation resources available for LabVIEW.

321780C-01 Jul00