LabVIEW Graphical Programming

Collection Editor:

Serhat Beyenir

LabVIEW Graphical Programming

Collection Editor:

Serhat Beyenir

Authors:

National Instruments
Malan Shiralkar

Online:

< http://cnx.org/content/col11408/1.1/ >

CONNEXIONS

Rice University, Houston, Texas

Collection structure revised: February 6, 2012
PDF generated: February 6, 2012

Table of Contents

Chapter 1

Introduction

1.1 LabVIEW[1]

LabVIEW programs are called **virtual instruments**, or **VIs**, because their appearance and operation imitate physical instruments, such as oscilloscopes and multimeters. LabVIEW contains a comprehensive set of tools for acquiring analyzing, displaying, and storing data, as well as tools to help you troubleshoot your code.

LabVIEW VIs contain three components-the **front panel**, the **block diagram**, and the **icon** and **connector pane**. This lesson describes the front panel and the block diagram; refer to Modular Programming (Section 2.1) for more information about the icon and the connector pane.

In LabVIEW, you build a user interface, or front panel, with controls and indicators. Controls are knobs, push buttons, dials, and other input devices. Indicators are graphs, LEDs, and other displays. After you build the user interface, you add code using VIs and structures to control the front panel objects. The block diagram contains this code. In some ways, the block diagram resembles a flowchart.

Use LabVIEW to communicate with hardware such as data acquisition, vision, and motion control devices, and GPIB, PXI, VXI, RS-232, and RS-484 devices. LabVIEW also has built-in features for connecting your application to the Web using the LabVIEW Web Server and software standards such as TCP/IP networking and ActiveX.

Using LabVIEW, you can create test and measurement, data acquisitions, instrument control, datalogging, measurement analysis, and report generation applications. You also can create stand-alone executables and shared libraries, like DLLs, because LabVIEW is a true 32-bit compiler.

[1] This content is available online at <http://cnx.org/content/m12192/1.2/>.

1.2 LabVIEW Environment[2]

When you launch LabVIEW, the navigation dialog box (Figure 1.1) appears that includes introductory material and common commands.

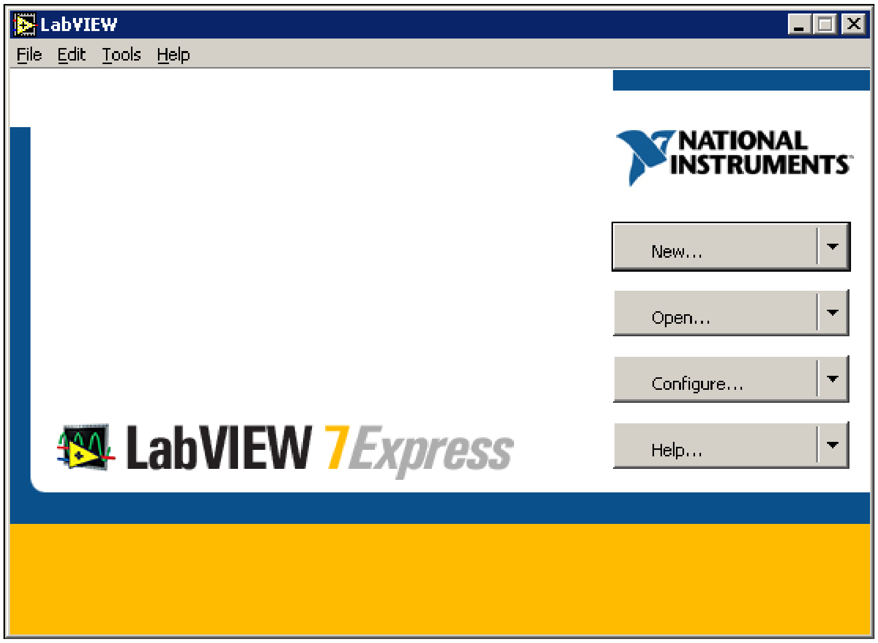


Figure 1.1: LabVIEW Dialog Box

The LabVIEW dialog box includes the following components:

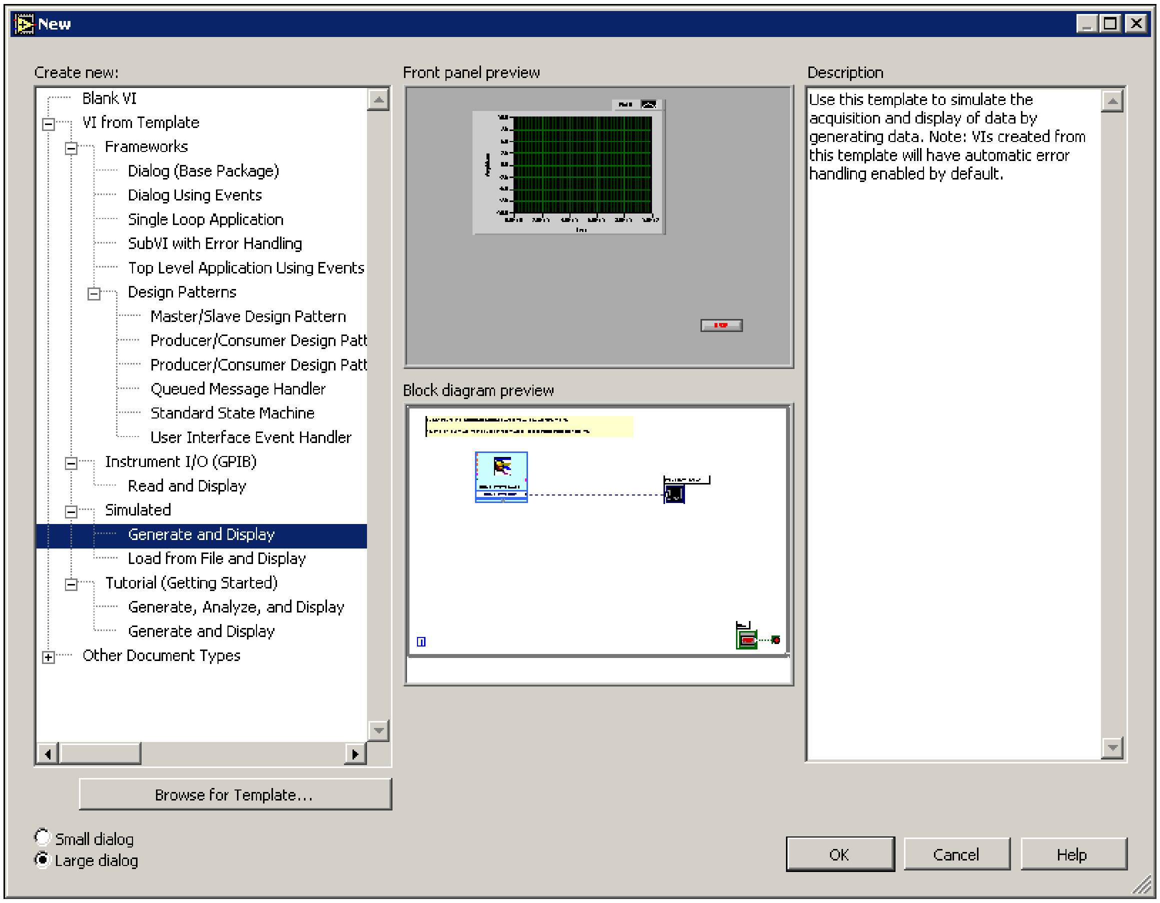
- A menu with standard items such as File≫Exit.
- A set of buttons for creating and opening VIs, configuring data acquisition devices, and finding helpful information.
    - Click the New button to create a new VI. Click the arrow on the New button to choose to open a blank VI or to open the New dialog box.
    - Click the Open button to open an existing VI. Click the arrow on the Open button to open recent files.
    - Click the Configure button to configure your data acquisition devices. Click the arrow on the Configure button to configure LabVIEW.
    - Click the Help button to launch the *LabVIEW Help*. Click the arrow on the Help button for other Help options, including the NI Example Finder.

1.2.1 Creating and Saving a VI

When you click the New button in the LabVIEW dialog box, the New dialog box appears. You also can select File≫New to display this dialog box. When you select a template in the Create new list, previews of the

[2] This content is available online at <http://cnx.org/content/m12193/1.3/>.

VI appear in the `Front panel preview` and the `Block diagram preview` sections, and a description of the template appears in the `Description` section. Figure 1.2 shows the `New` dialog box and the `SubVI with Error Handling VI` template.



Description: Use this template to simulate the acquisition and display of data by generating data. Note: VIs created from this template will have automatic error handling enabled by default.

○ Small dialog
● Large dialog

Figure 1.2: New Dialog Box

Click the `OK` button to open the template. You also can double-click the name of the template VI in the `Create new` list to open the template. If no template is available for the task you want to create, you can start with a blank VI and create a VI to accomplish the specific task. In the `LabVIEW` dialog box, click the arrow on the `New` button and select `Blank VI` from the shortcut menu or press the `Ctrl-N` keys to open a blank VI.

NOTE: You also can open a blank VI by selecting `Blank VI` from the `Create new` list in the `New` dialog box or by selecting `File≫New VI`.

1.2.1.1 Open/Templates

Use the `New` dialog box to create different components in LabVIEW to help you build an application. You can start with a blank VI to write a VI from scratch, or start with a template to simplify the programming.

The `New` dialog box includes the following components:

Create new - Displays templates you can use to start building VIs and other LabVIEW documents. Select from the following templates and click the `OK` button to start building a VI or other LabVIEW document.

- **Blank VI -** Opens a blank front panel and blank block diagram.
- **VI from Template -** Opens a front panel and block diagram with components you need to build different types of VIs.
- **Other Document Types -** Opens the tools you use to build other LabVIEW objects.

Browse for Template - Displays the `Browse` dialog box so you can navigate to a VI, control, or template. If you previously have browsed for and selected a template from this dialog box, use the pull-down menu of the `Browse` button to select a template to reopen it.

Front panel preview - Displays the front panel for the VI template you selected in the `Create new` list.

Block diagram preview - Displays the block diagram for the VI template you selected in the `Create new` list.

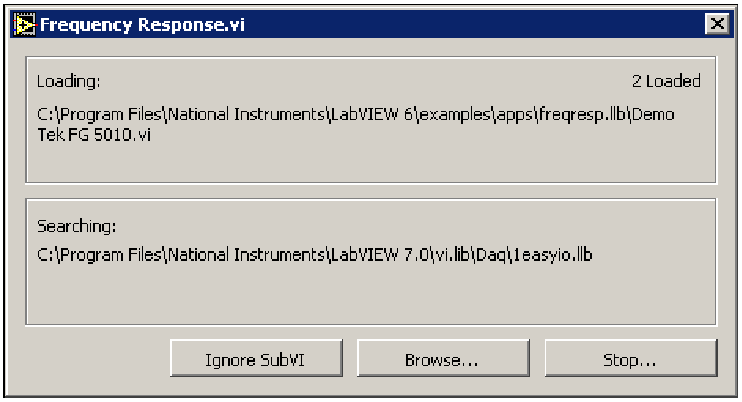
Description - Displays a description of the template you selected in the `Create new` list if the template includes a description.

1.2.1.2 Opening an Existing VI

You load a VI into memory by selecting `File`≫`Open`. In the `Choose the VI to open` dialog box that appears, navigate to the VI you want to open.

The VIs you edit in this course are located in the `C:\Exercises\LabVIEW Basics I` directory.

As the VI loads, a status dialog box similar to Figure 1.3 might appear.



Frequency Response.vi

Loading: 2 Loaded

C:\Program Files\National Instruments\LabVIEW 6\examples\apps\freqresp.llb\Demo Tek FG 5010.vi

Searching:

C:\Program Files\National Instruments\LabVIEW 7.0\vi.lib\Daq\1easyio.llb

Figure 1.3

The `Loading` section lists the subVIs of the VI as they are loaded into memory. `Number Loaded` is the number of subVIs loaded into memory so far. You can cancel the load at any time by clicking the `Stop` button.

If LabVIEW cannot immediately locate a subVI, it begins searching through all directories specified by the `VI Search Path`. You can edit the `VI Search Path` by selecting `Tools`≫`Options` and selecting `Paths` from the top pull-down menu. The `Searching` section lists directories or VIs as LabVIEW searches through them. You can have LabVIEW ignore a subVI by clicking the `Ignore SubVI` button, or you can click the `Browse` button to search for the missing subVI.

1.2.1.3 Saving VIs

Select `Save`, `Save As`, `Save All`, or `Save with Options` from the `File` menu to save VIs as individual files or group several VIs together and save them in a VI library. VI library files end with the extension `.llb`. National Instruments recommends that you save VIs as individual files, organized in directories, especially if multiple developers are working on the same project.

LabVIEW uses the native file dialog boxes so they act similar to other applications on the computer. You can disable this feature by selecting `Tools`≫`Options` and selecting `Miscellaneous` from the top pull-down menu. If you disable native file dialogs, LabVIEW uses its own platform-independent file dialog boxes with some convenient features, such as providing a list of recent paths and reducing the steps necessary to save VIs in VI libraries.

1.2.1.4 Moving VIs Across Platforms

You can transfer VIs from one platform to another, such as from Mac OS to Windows. LabVIEW automatically translates and recompiles the VIs on the new platform.

Because VIs are files, you can use any file transfer method or utility to move VIs between platforms. You can port VIs over networks using FTP, Z or XModem protocols, or similar utilities. Such network transfers eliminate the need for additional file translation software. If you port VIs using magnetic media, such as floppy disks or a moveable external hard drive, you need a generic file transfer utility program, such as the following:

- (**Windows**) MacDisk and TransferPro transfer Mac OS files to the PC format and vice versa.
- (**Mac OS**) DOS Mounter, MacLink, and Apple File Exchange convert PC files to the Mac OS format and vice versa.
- (**Sun**) PC File System (PCFS) converts PC files to the Sun format and vice versa.

> NOTE: Certain operating system-specific VIs are not portable between platforms, such as DDE (Dynamic Data Exchange) VIs, ActiveX VIs, and AppleEvents.

Refer to the *Porting and Localizing LabVIEW VIs* Application Note, available by selecting `Help`≫`Search the LabVIEW Bookshelf`, for more information about porting VIs.

1.2.2 Menus

The menus at the top of a VI window contain items common to other applications, such as `Open`, `Save`, `Copy`, and `Paste`, and other items specific to LabVIEW. Some menu items also list shortcut key combinations.

(**Mac OS**) The menus appear at the top of the screen.

(**Windows** and **UNIX**) The menus display only the most recently used items by default. Click the arrows at the bottom of a menu to display all items. You can display all menu items by default by selecting `Tools`≫`Options` and selecting `Miscellaneous` from the top pull-down menu.

> NOTE: Some menu items are unavailable while a VI is in run mode.

- The `File` menu contains items used for basic file operations, such as opening, closing, saving, and printing files.
- The `Edit` menu contains items that allow you to search for and modify LabVIEW files and their components.
- The `Operate` menu contains items you use to control the operation of VIs.

- The Tools menu contains items for configuring LabVIEW, your projects, and your VIs.
- The Browse menu contains items that allow you to view aspects of the current VI and its hierarchy.
- The Window menu contains items that allow you to configure the appearance of the current windows and palettes. You also can access the Error List window and view the contents of the clipboard.
- The Help menu contains items to explain and define LabVIEW features and other components, provide full LabVIEW documentation, and access National Instruments technical support.

1.2.3 Front Panel and Block Diagram Windows

When you open a blank VI, an untitled front panel window appears. This window displays the front panel and is one of the two LabVIEW windows you use to build a VI. The other window contains the block diagram. The illustration in Figure 1.4 shows a front panel and its corresponding block diagram with front panel and block diagram components.

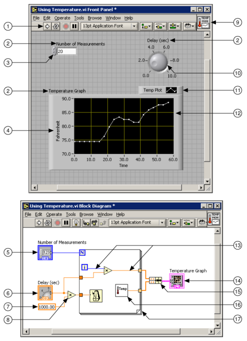


Figure 1.4: 1. Toolbar, 2. Owned Label, 3. Numeric Control, 4. Free Label, 5. Numeric Control Terminal, 6. Knob Terminal, 7. Numeric Constant, 8. Multiply Function, 9. Icon, 10. Knob Control, 11. Plot Legend, 12. XY Graph, 13. Wire Data Path, 14. XY Graph Terminal, 15. Bundle Function, 16. SubVI, 17. For Loop Structure

1.2.4 Front Panel Toolbar

Use the toolbar buttons to run and edit a VI. The toolbar in Figure 1.5 appears on the front panel.

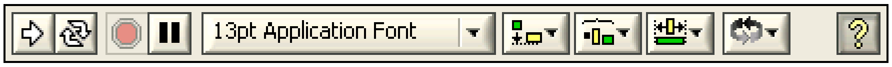


Figure 1.5

⇨ Click the Run button to run a VI. LabVIEW compiles the VI, if necessary. You can run a VI if the Run button appears as a solid white arrow. The solid white arrow, shown above, also indicates you can use the VI as a subVI if you create a connector pane for the VI.

While the VI runs, the Run button appears as shown at left if the VI is a top-level VI, meaning it has no callers and therefore is not a subVI.

If the VI that is running is a subVI, the Run button appears as shown at left.

The Run button appears broken, shown at left, when the VI you are creating or editing contains errors. If the Run button still appears broken after you finish wiring the block diagram, the VI is broken and cannot run. Click this button to display the Error list window, which lists all errors and warnings.

Click the Run Continuously button, shown at left, to run the VI until you abort or pause execution. You also can click the button again to disable continuous running.

While the VI runs, the Abort Execution button, shown at left, appears. Click this button to stop the VI immediately if there is no other way to stop the VI. If more than one running top-level VI uses the VI, the button is dimmed.

> NOTE: Avoid using the Abort Execution button to stop a VI. Either let the VI complete its data flow or design a method to stop the VI programmatically. By doing so, the VI is at a known state. For example, place a button on the front panel that stops the VI when you click it.

Click the Pause button, shown at left, to pause a running VI. When you click the Pause button, LabVIEW highlights on the block diagram the location where you paused execution, and the Pause button appears red. Click the button again to continue running the VI.

13pt Application Font ▾ Select the Text Settings pull-down menu, shown at left, to change the font settings for the selected portions of the VI, including size, style, and color.

Select the Align Objects pull-down menu, shown at left, to align objects along axes, including vertical, top edge, left, and so on.

Select the Distribute Objects pull-down menu, shown at left, to space objects evenly, including gaps, compression, and so on.

Select the Resize Objects pull-down menu, shown at left, to resize multiple front panel objects to the same size.

Select the Reorder pull-down menu, shown at left, when you have objects that overlap each other and you want to define which one is in front or back of another. Select one of the objects with the Positioning tool and then select from Move Forward, Move Backward, Move To Front, and Move To Back.

Select the Show Context Help Window button, shown at left, to toggle the display of the Context Help window.

Type appears to remind you that a new value is available to replace an old value. The Enter button disappears when you click it, press the Enter key, or click the front panel or block diagram workspace.

1.2.5 Block Diagram Toolbar

When you run a VI, buttons appear on the block diagram toolbar that you can use to debug the VI. The toolbar in Figure 1.6 appears on the block diagram.

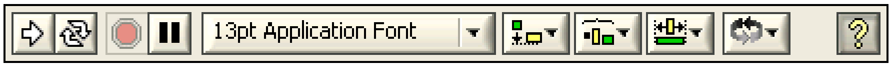


Figure 1.6

Click the `Highlight Execution` button, shown at left, to display an animation of the block diagram execution when you click the `Run` button. see the flow of data through the block diagram. Click the button again to disable execution highlighting.

Click the `Step Into` button, shown at left, to open a node and pause. When you click the `Step Into` button again, it executes the first action and pauses at the next action of the subVI or structure. You also can press the `Ctrl` and down arrow keys. Single-stepping through a VI steps through the VI node by node. Each node blinks to denote when it is ready to execute. By stepping into the node, you are ready to single-step inside the node.

Click the `Step Over` button, shown at left, to execute a node and pause at the next node. You also can press the `Ctrl` and right arrow keys. By stepping over the node, you execute the node without single-stepping through the node.

Click the `Step Out` button, shown at left, to finish executing the current node and pause. When the VI finishes executing, the `Step Out` button becomes dimmed. You also can press the `Ctrl` and up arrow keys. By stepping out of a node, you complete single-stepping through the node and go to the next node.

The `Warning` button, shown at left, appears if a VI includes a warning and you placed a checkmark in the `Show Warnings` checkbox in the `Error List` window. A warning indicates there is a potential problem with the block diagram, but it does not stop the VI from running.

1.2.6 Palettes

LabVIEW has graphical, floating palettes to help you create and run VIs. The three palettes include the `Tools`, `Controls`, and `Functions` palettes. You can place these palettes anywhere on the screen.

1.2.6.1 Tools Palette

You can create, modify, and debug VIs using the tools located on the floating Tools (Figure 1.7) palette. The `Tools` palette is available on both the front panel and the block diagram. A tool is a special operating mode of the mouse cursor. The cursor corresponds to the icon of the tool selected in the `Tools` palette. Use the tools to operate and modify front panel and block diagram objects.

Select `Window`≫`Show Tools Palette` to display the `Tools` palette.

NOTE: Press the `Shift` key and right-click to display a temporary version of the `Tools` palette at the location of the cursor.

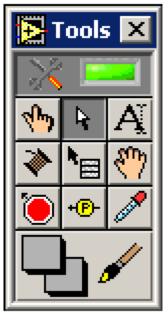


Figure 1.7: Tools Palette

If automatic tool selection is enabled and you move the cursor over objects on the front panel or block diagram, LabVIEW automatically selects the corresponding tool from the Tools palette. You can disable automatic tool selection and select a tool manually by clicking the tool you want on the Tools palette.

If you want to use the Tab key to cycle through the four most common tools on the Tools palette, click the Automatic Tool Selection button, shown at left, on the Tools palette to disable automatic tool selection. Press the Shift-Tab keys or click the Automatic Tool Selection button to enable automatic tool selection again. You also can manually select a tool on the Tools palette to disable automatic tool selection. Press the Tab or Shift-Tab keys or click the Automatic Tool Selection button on the Tools palette to enable automatic tool selection again. If automatic tool selection is disabled, you can press the spacebar to switch to the next most useful tool.

Use the Operating tool, shown at left, to change the values of a control or select the text within a control. The Operating tool changes to the icon shown at left when it moves over a text control, such as a numeric or string control.

Use the Positioning tool, shown at left, to select, move, or resize objects. The Positioning tool changes to resizing handles when it moves over the edge of a resizable object.

Use the Labeling tool, shown at left, to edit text and create free labels. The Labeling tool changes to the following icon when you create free labels.

Use the Wiring tool, shown at left, to wire objects together on the block diagram.

Use the Object Shortcut Menu, shown at left, tool to access an object shortcut menu with the left mouse button.

Use the Scrolling tool, shown at left, to scroll through windows without using scrollbars.

Use the Breakpoint tool, shown at left, to set breakpoints on VIs, functions, nodes, wires, and structures to pause execution at that location.

Use the Probe tool, shown at left, to create probes on wires on the block diagram. Use the Probe tool to check intermediate values in a VI that produces questionable or unexpected results.

Use the Color Copy tool, shown at left, to copy colors for pasting with the Coloring tool.

Use the Coloring tool, shown at left, to color an object. It also displays the current foreground and background color settings.

1.2.6.2 Controls and Functions Palettes

The Controls and Functions palettes contain subpalettes of objects you can use to create a VI. When you click a subpalette icon, the entire palette changes to the subpalette you selected. To use an object on the palettes, click the object and place it on the front panel or block diagram.

The Controls palette, shown in Figure 1.8, is available only on the front panel. The Controls palette contains the controls and indicators you use to build the front panel. Refer to the Front Panel (Section 1.3) section for more information about the using the Controls palette on the front panel. The controls and indicators located on the Controls palette depend on the palette view currently selected.

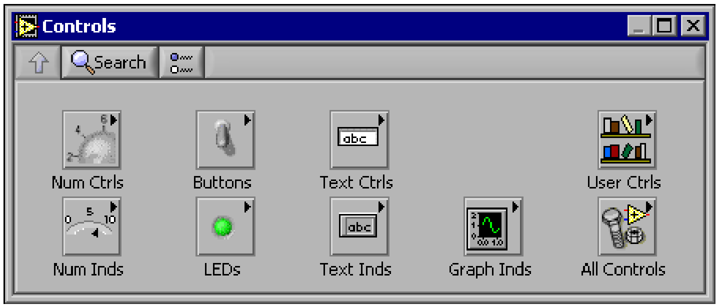


Figure 1.8: Controls Palette

The Functions palette, shown in the Figure 1.9, is available only on the block diagram. The Functions palette contains the VIs and functions you use to build the block diagram. Refer to the Block Diagram (Section 1.4) section of this lesson for more information about using the Functions palette on the block diagram. The VIs and functions located on the Functions palette depend on the palette view currently selected. The VIs and functions are located on subpalettes based on the types of VIs and functions.

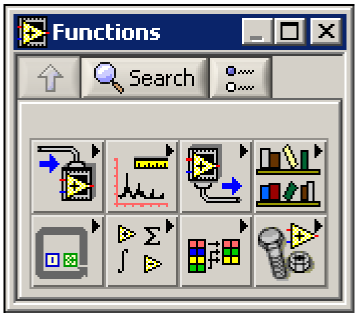


Figure 1.9: Functions Palette

1.2.6.3 Changing Palette Views

Use the `Options` button on the `Controls` or `Functions` palette toolbar to change to another palette view or format:

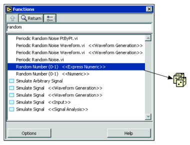
1. Click the `Options` button, shown at left, on the `Controls` or `Functions` palette toolbar to display the `Controls/Functions Palettes` page of the `Options` dialog box.
2. Select a palette view from the `Palette View` pull-down menu.
3. Select a format from the `Format` pull-down menu, such as `Standard`, `All Icons`, `All Text`, or `Icons and Text`.
4. Click the `OK` button. The `Controls` and `Functions` palettes change to the palette view and format you selected.

1.2.6.4 Searching for Controls, VIs, and Functions

Use the following navigation buttons on the `Controls` and `Functions` palettes to navigate and search for controls, VIs, and functions:

Up to Owning Palette - Navigates 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 on the palettes. For example, if you want to find the `Random Number` function, click the `Search` button on the `Functions` palette toolbar and start typing `Random Number` in the text box at the top of the palette. LabVIEW lists all matching items that either start with or contain the text you typed. You can click one of the search results and drag it to the block diagram, as shown in Figure 1.10.


Functions
Return
Periodic Random Noise PtByPt.vi
Periodic Random Noise Waveform.vi <<Waveform Generation>>
Periodic Random Noise Waveform.vi <<Waveform Generation>>
Periodic Random Noise.vi
Random Number (0-1) <<Express Numeric>>
Random Number (0-1) <<Numeric>>
Simulate Arbitrary Signal
Simulate Signal <<Waveform Generation>>
Simulate Signal <<Waveform Generation>>
Simulate Signal <<Input>>
Simulate Signal <<Signal Analysis>>
Options
Help

random

Figure 1.10

Double-click the search result to highlight its location on the palette. You then can click the `Up to Owning Palette` button to view the hierarchy of where the object resides.

1.2.7 Shortcut Menus

The most often-used menu is the object shortcut menu. All LabVIEW objects and empty space on the front panel and block diagram have associated shortcut menus. Use the shortcut menu items to change the look or behavior of front panel and block diagram objects. To access the shortcut menu, right-click the object, front panel, or block diagram. The shortcut menu for a meter is shown in Figure 1.11.

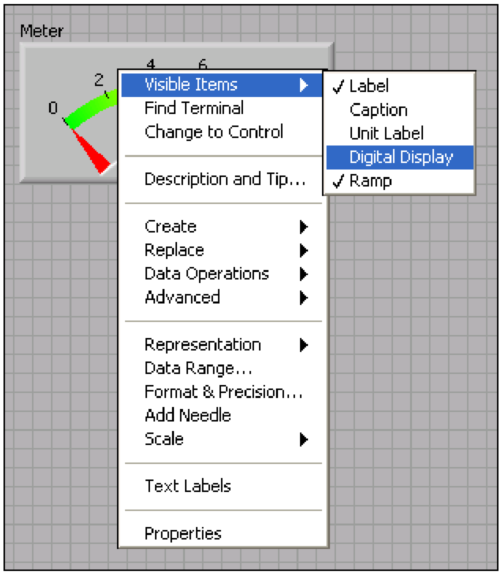
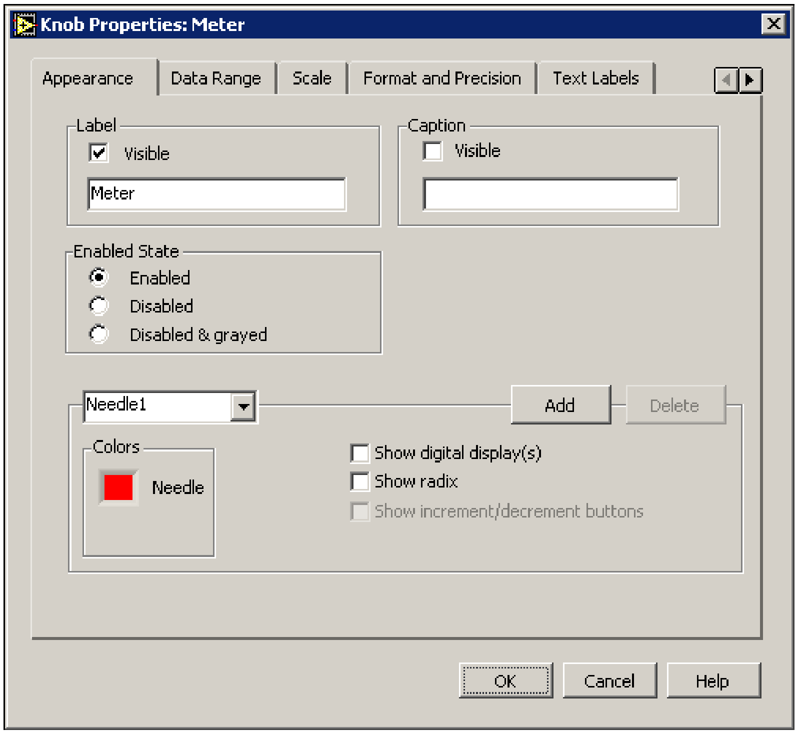


Figure 1.11: Meter Shortcut Menu

1.2.7.1 Property Dialog Boxes

Front panel objects also have property dialog boxes that you can use to change the look or behavior of front panel objects. Right-click a front panel object and select `Properties` from the shortcut menu to access the property dialog box for an object. Figure 1.12 shows the property dialog box for the meter in the previous figure. The options available on the property dialog box for an object are similar to the options available on the shortcut menu for that object.



Label
☑ Visible
Meter

Caption
□ Visible

Enabled State
● Enabled
○ Disabled
○ Disabled & grayed

Needle1

Colors
🟥 Needle

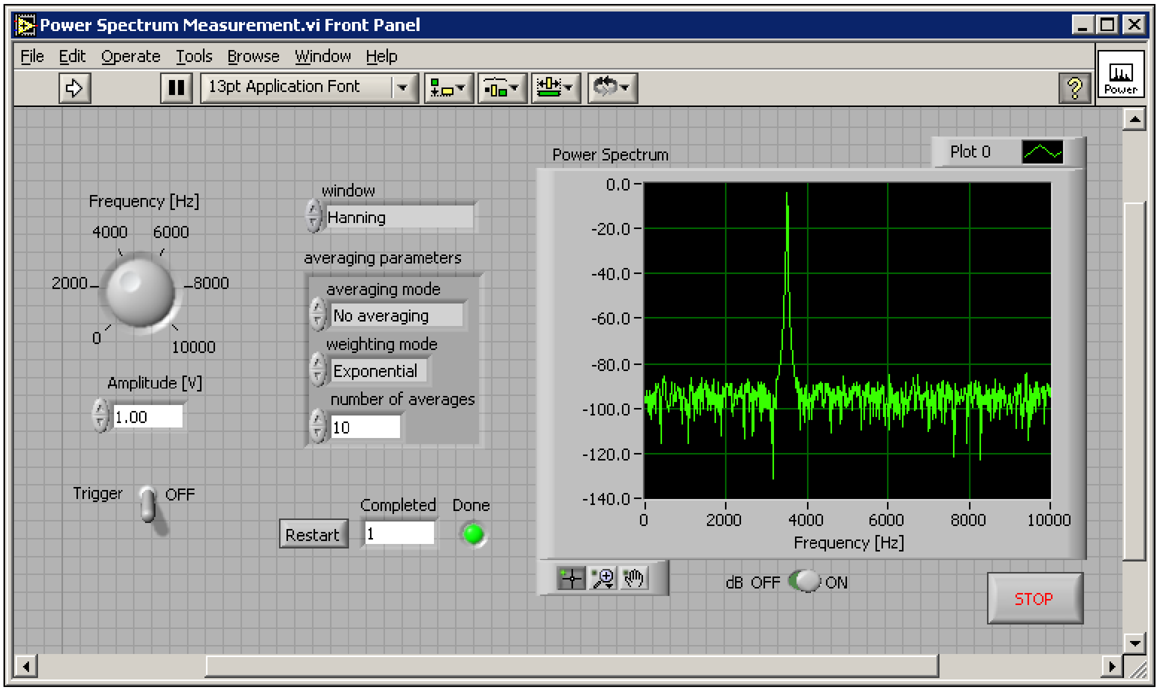
□ Show digital display(s)
□ Show radix
□ Show increment/decrement buttons

Figure 1.12: Meter Property Dialog Box

1.3 Front Panel[3]

The front panel is the user interface of the VI. Figure 1.13 shows an example of a front panel.

[3] This content is available online at <http://cnx.org/content/m12194/1.1/>.



Amplitude [V] 1.00

window Hanning

averaging parameters

averaging mode No averaging

weighting mode Exponential

number of averages 10

Trigger OFF

Completed 1

Done

dB OFF ON

Figure 1.13: Example of a Front Panel

1.3.1 Controls and Indicators

You build the front panel with controls and indicators, which are the interactive input and output terminals of the VI, respectively. Controls are knobs, push buttons, dials, and other input devices. Indicators are graphs, LEDs, and other displays. Controls simulate instrument input devices and supply data to the block diagram of the VI. Indicators simulate instrument output devices and display data the block diagram acquires or generates.

1.3.1.1 Controls Palette

The `Controls` palette is available only on the front panel. The `Controls` palette contains the controls and indicators you use to create the front panel. Select `Window`≫`Show Controls Palette` or right-click the front panel workspace to display the `Controls` palette. Tack down the `Controls` palette by clicking the thumbtack on the top left corner of the palette. By default, the `Controls` palette starts in the Express view.

The Express palette view includes subpalettes on the top level of the `Controls` and `Functions` palettes that contain Express VIs and other objects you need to build common measurement applications. The `All Controls` and `All Functions` subpalettes contain the complete set of built-in controls, indicators, VIs, and functions.

The `Advanced` palette view includes subpalettes on the top level of the `Controls` and `Functions` palettes that contain the complete set of built-in controls, indicators, VIs, and functions. The `Express` subpalettes contain Express VIs and other objects you need to build common measurement applications.

NOTE: In the `Express` palette view, toolsets and modules do not install subpalettes on the top level of the `Controls` and `Functions` palettes. Instead, toolsets and modules install on the `All Controls` and `All Functions` subpalettes. In the `Advanced` palette view, toolsets and modules install subpalettes on the top level.

Click the `Options` button on the `Controls` or `Functions` palette to change to another palette view or format.

1.3.1.1.1 Numeric Controls and Indicators

The two most commonly used numeric objects are the numeric control and the numeric indicator, as shown in Figure 1.14.

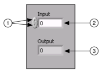


Figure 1.14: 1. Increment and Decrement Buttons, 2. Numeric Control, 3. Numeric Indicator

To enter or change values in a numeric control, click the `increment` and `decrement` buttons with the `Operating` tool or double-click the number with either the `Labeling` tool or the `Operating` tool, type a new number, and press the `Enter` key.

1.3.1.1.2 Boolean Controls and Indicators

Use Boolean controls and indicators to enter and display Boolean (`True` or `False`) values. Boolean objects simulate switches, push buttons, and LEDs. The most common Boolean objects are the vertical toggle switch and the round LED, as shown in Figure 1.15.

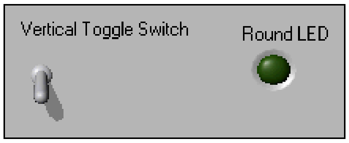


Figure 1.15

1.4 Block Diagram[4]

After you build the front panel, you add code using graphical representations of functions to control the front panel objects. The block diagram contains this graphical source code. Front panel objects appear as terminals, on the block diagram. Block diagram objects include terminals, subVIs, functions, constants, structures, and wires, which transfer data among other block diagram objects.

The VI in Figure 1.16 shows several primary block diagram objects-nodes, terminals, and wires.

[4] This content is available online at <http://cnx.org/content/m12195/1.2/>.

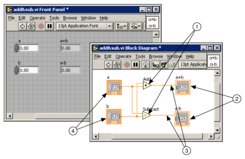


Figure 1.16: 1. Nodes, 2. Indicator Terminals, 3. Wires, 4. Control Terminals

1.4.1 Functions Palette

The `Functions` palette is available only on the block diagram. The `Functions` palette contains the VIs and functions you use to build the block diagram. Select `Window≫Show Functions Palette` or right-click the block diagram workspace to display the `Functions` palette. Tack down the `Functions` palette by clicking the thumbtack on the top left corner of the palette. By default, the `Functions` palette starts in the Express view.

1.4.2 Express VIs, VIs, and Functions

LabVIEW uses colored icons to distinguish between Express VIs, VIs, and functions on the block diagram. By default, icons for Express VIs appear on the block diagram as expandable nodes with icons surrounded by a blue field. Icons for VIs have white backgrounds, and icons for functions have pale yellow backgrounds.

By default, most functions and VIs on the block diagram appear as icons that are not expandable, unlike Express VIs.

1.4.2.1 Express VIs

Use Express VIs for common measurement tasks. Express VIs are nodes that require minimal wiring because you configure them with dialog boxes. You can save the configuration of an Express VI as a subVI. Refer to *Building the Block Diagram*, of the *LabVIEW User Manual* for more information about creating subVIs from Express VIs.

1.4.2.2 VIs

When you place a VI on the block diagram, LabVIEW considers the VI to be a subVI. When you double-click a subVI, its front panel and block diagram appear, rather than a dialog box in which you can configure options. The front panel includes controls and indicators. The block diagram includes wires, front panel icons, functions, possibly subVIs, and other LabVIEW objects.

The upper right corner of the front panel and block diagram displays the icon for the VI. This is the icon that appears when you place the VI on the block diagram.

You can create a VI to use as a subVI. Refer to Modular Programming (Section 2.1) for more information about creating VIs and configuring them as subVIs.

1.4.2.3 Functions

Functions are the fundamental operating elements of LabVIEW. Functions do not have front panels or block diagrams but do have connector panes. Double-clicking a function only selects the function.

1.4.3 Nodes

Nodes are objects on the block diagram that have inputs and/or outputs and perform operations when a VI runs. They are analogous to statements, operators, functions, and subroutines in text-based programming languages. Nodes can be functions, subVIs, or structures. Structures are process control elements, such as `Case` structures, `For Loops`, or `While Loops`. The `Add` and `Subtract` functions in Figure 1.16 are function nodes.

1.4.3.1 Expandable Nodes versus Icons

You can display VIs and Express VIs as icons or as expandable nodes. Expandable nodes appear as icons surrounded by a colored field. SubVIs appear with a yellow field, and Express VIs appear with a blue field. Use icons, such as the `Basic Function Generator` VI icon if you want to conserve space on the block diagram. Use expandable nodes, such as the `Basic Function Generator` VI expandable node to make wiring easier and to aid in documenting block diagrams. By default, subVIs appear as icons on the block diagram, and Express VIs appear as expandable nodes.

To display a subVI or Express VI as an expandable node, right-click the subVI or Express VI and select `View As Icon` from the shortcut menu to remove the checkmark.

You can resize the expandable node to make wiring even easier, but it also takes a large amount of space on the block diagram. Complete the following steps to resize a node on the block diagram.

1. Move the `Positioning` tool over the node. Resizing handles appear at the top and bottom of the node.
2. Move the cursor over a resizing handle to change the cursor to the resizing cursor.
3. Use the resizing cursor to drag the border of the node down to display additional terminals.
4. Release the mouse button.

To cancel a resizing operation, drag the node border past the block diagram window before you release the mouse button.

Figure 1.17 shows the `Basic Function Generator` VI as a resized expandable node.

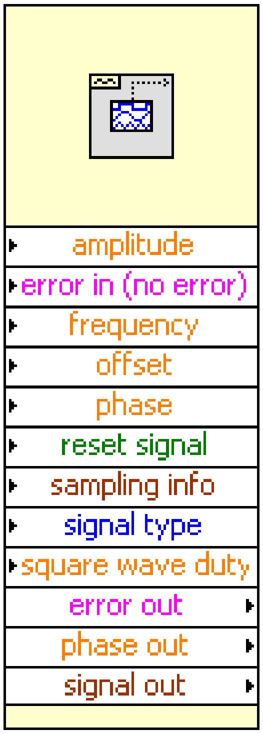


Figure 1.17

NOTE: If you display a subVI or Express VI as an expandable node, you cannot display the terminals for that node and you cannot enable database access for that node.

1.4.3.2 Terminals

Front panel objects appear as terminals on the block diagram. The terminals represent the data type of the control or indicator. You can configure front panel controls or indicators to appear as icon or data type terminals on the block diagram. By default, front panel objects appear as icon terminals. For example, a knob icon terminal, shown at left, represents a knob on the front panel. The DBL at the bottom of the terminal represents a data type of double-precision, floating-point numeric. To display a terminal as a data type on the block diagram, right-click the terminal and select `View As Icon` from the shortcut menu to remove the checkmark. A DBL data type terminal, shown above at left, represents a double-precision, floating-point numeric control or indicator.

Terminals are entry and exit ports that exchange information between the front panel and block diagram. Terminals are analogous to parameters and constants in text-based programming languages. Types of terminals include control or indicator terminals and node terminals. Control and indicator terminals belong to front panel controls and indicators. Data you enter into the front panel controls (*a* and *b* in Figure 1.16) enter the block diagram through the control terminals. The data then enter the `Add` and `Subtract` functions. When the `Add` and `Subtract` functions complete their internal calculations, they produce new data values. The data flow to the indicator terminals, where they exit the block diagram, reenter the front panel, and appear in front panel indicators ($a + b$ and $a - b$ in Figure 1.16). The terminals in Figure 1.16 belong to four front panel controls and indicators. The connector panes of the `Add` and `Subtract` functions, shown in this media, have three node terminals. To display the terminals of the function on the block diagram, right-click the function node and select `Visible Items`≫`Terminals` from the shortcut menu.

1.4.3.3 Wires

You transfer data among block diagram objects through wires. Wires are analogous to variables in text-based programming languages. In Figure 1.16, wires connect the control and indicator terminals to the `Add` and `Subtract` functions. Each wire has a single data source, but you can wire it to many VIs and functions that read the data. Wires are different colors, styles, and thicknesses, depending on their data types. A broken wire appears as a dashed black line with a red X in the middle. The examples in Table 1.1 are the most common wire types.

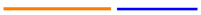
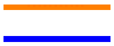
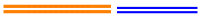
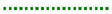
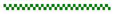
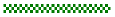
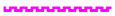
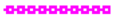
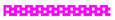
Wire Type	Scalar	1D Array	2D Array	Color
Numeric				Orange (floating-point), Blue (integer)
Boolean				Green
String				Pink

Table 1.1

In LabVIEW, you use wires to connect multiple terminals together to pass data in a VI. The wires must be connected to inputs and outputs that are compatible with the data that is transferred with the wire. For example, you cannot wire an array output to a numeric input. In addition the direction of the wires must be correct. The wires must be connected to only one input and at least one output. For example, you cannot wire two indicators together. The components that determine wiring compatibility include the data type of the control and/or indicator and the data type of the terminal.

1.4.3.3.1 Data Types

Data types indicate what objects, inputs, and outputs you can wire together. For example, a switch has a green border so you can wire a switch to any input with a green label on an Express VI. A knob has an orange border so you can wire a knob to any input with an orange label. However, you cannot wire an orange knob to an input with a green label. Notice the wires are the same color as the terminal.

The dynamic data type stores the information generated or acquired by an Express VI. The dynamic data type appears as a dark blue terminal, shown at left. Most Express VIs accept and/or return the dynamic data type. You can wire the dynamic data type to any indicator or input that accepts numeric, waveform, or Boolean data. Wire the dynamic data type to an indicator that can best present the data. Indicators include a graph, chart, or numeric indicator.

Most other VIs and functions in LabVIEW do not accept the dynamic data type. To use a built-in VI or function to analyze or process the data the dynamic data type includes, you must convert the dynamic data type.

`Use the Convert from Dynamic Data` Express VI, shown at left, to convert the dynamic data type to numeric, Boolean, waveform, and array data types for use with other VIs and functions. When you place the `Convert from Dynamic Data` Express VI on the block diagram, the `Configure Convert from`

Dynamic Data dialog box appears. The Configure Convert from Dynamic Data dialog box displays options that let you specify how you want to format the data that the Convert from Dynamic Data Express VI returns.

When you wire a dynamic data type to an array indicator, LabVIEW automatically places the Convert from Dynamic Data Express VI on the block diagram. Double-click the Convert from Dynamic Data Express VI to open the Configure Convert from Dynamic Data dialog box to control how the data appears in the array.

Use the Convert to Dynamic Data Express VI to convert numeric, Boolean, waveform, and array data types to the dynamic data type for use with Express VIs. When you place the Convert to Dynamic Data Express VI on the block diagram, the Configure Convert to Dynamic Data dialog box appears. Use this dialog box to select the kind of data to convert to the dynamic data type.

1.4.3.3.2 Automatically Wiring Objects

LabVIEW automatically wires objects as you place them on the block diagram. You also can automatically wire objects already on the block diagram. LabVIEW connects the terminals that best match and leaves terminals that do not match unconnected.

As you move a selected object close to other objects on the block diagram, LabVIEW draws temporary wires to show you valid connections. When you release the mouse button to place the object on the block diagram, LabVIEW automatically connects the wires.

Toggle automatic wiring by pressing the spacebar while you move an object using the Positioning tool. You can adjust the automatic wiring settings by selecting Tools≫Options and selecting Block Diagram from the top pull-down menu.

1.4.3.3.3 Manually Wiring Objects

When you pass the Wiring tool over a terminal, a tip strip appears with the name of the terminal. In addition, the terminal blinks in the Context Help window and on the icon to help you verify that you are wiring to the correct terminal.

1.5 Acquiring a Signal VI[5]

Exercise 1.5.1

In the following exercise, you will build a VI that generates a signal and displays that signal in a graph. LabVIEW provides templates containing information from which you can build a VI. These templates help you get started with LabVIEW.

Complete the following steps to create a VI that generates a signal and displays it on the front panel.

1. Launch LabVIEW.
2. In the LabVIEW dialog box that appears, shown in Figure 1.18, click the New button to display the New dialog box.

[5] This content is available online at <http://cnx.org/content/m12196/1.3/>.

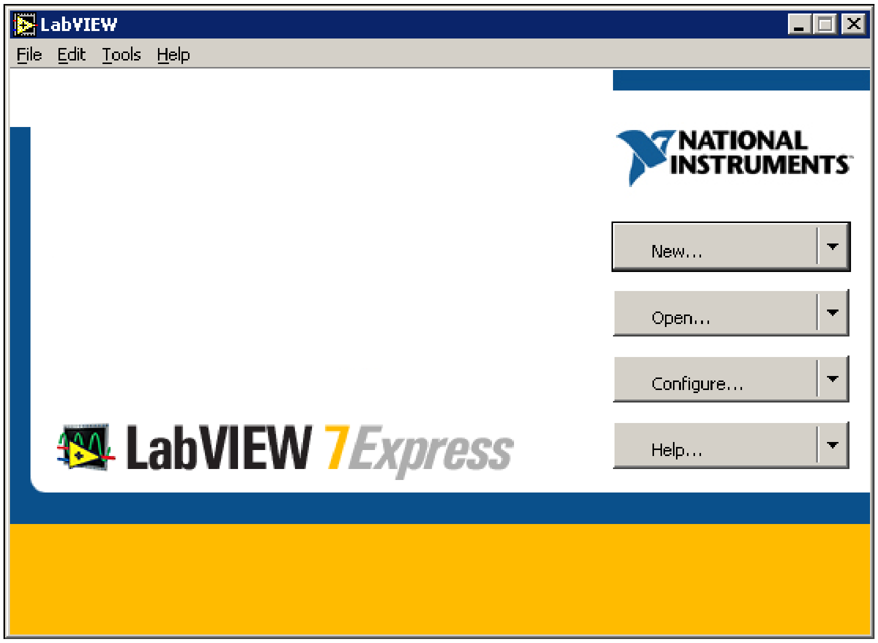
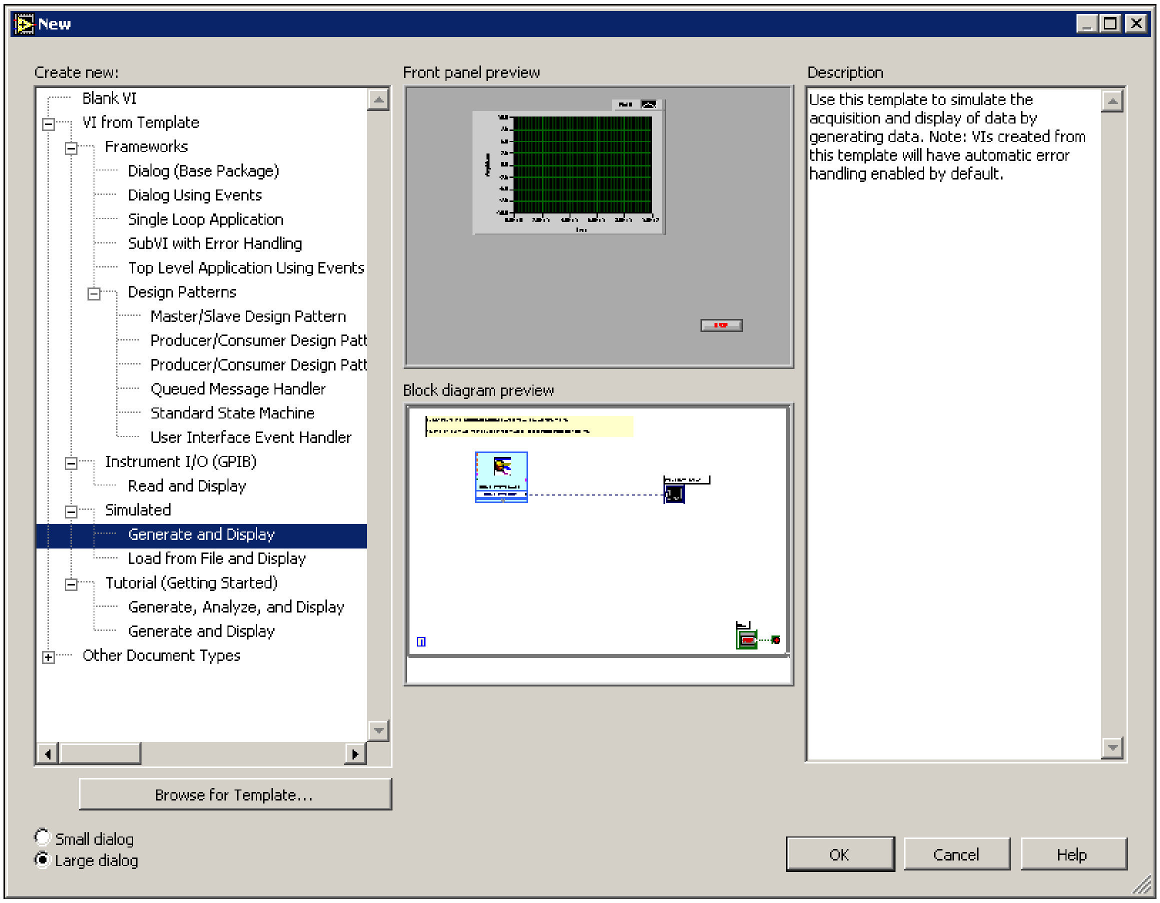


Figure 1.18

3. Select `VI from Template`≫`Tutorial (Getting Started)`≫`Generate and Display` in the `Create new` list. This template VI generates and displays a signal. Notice that previews of the template VI appear in the `Front panel preview` and the `Block diagram preview` sections. The Figure 1.19 shows the `New` dialog box and the `Generate and Display template` VI.



Use this template to simulate the acquisition and display of data by generating data. Note: VIs created from this template will have automatic error handling enabled by default.

○ Small dialog
● Large dialog

Figure 1.19

4. Click the OK button to open the template. You also can double-click the name of the template VI in the Create new list to open the template.
5. Examine the front panel of the VI. The user interface, or front panel, appears with a gray background and includes controls and indicators. The title bar of the front panel indicates that this window is the front panel for the Generate and Display VI.

   NOTE: If the front panel is not visible, you can display the front panel by selecting Window≫Show Front Panel.
6. Examine the block diagram of the VI. The block diagram appears with a white background and includes VIs and structures that control the front panel objects. The title bar of the block diagram indicates that this window is the block diagram for the Generate and Display VI.

   NOTE: If the block diagram is not visible, you can display the block diagram by selecting Window≫Show Block Diagram.
7. ⇨ On the front panel toolbar, click the Run button, shown at left. Notice that a sine wave appears on the graph.
8. STOP Stop the VI by clicking the Stop button, shown at left, on the front panel.

1.5.1 Adding a Control to the Front Panel

Controls on the front panel simulate the input devices on a physical instrument and supply data to the block diagram of the VI. Many physical instruments have knobs you can turn to change an input value. Complete the following steps to add a knob control to the front panel.

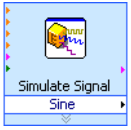
TIP: Throughout these exercises, you can undo recent edits by selecting `Edit≫Undo` or pressing the `Ctrl-Z` keys.

1. If the `Controls` palette is not visible on the front panel, select `Window≫Show Controls Palette` to display it.
2. Move the cursor over the icons on the `Controls` palette to locate the `Numeric Controls` palette. Notice that when you move the cursor over icons on the `Controls` palette, the name of that subpalette appears in the gray space above all the icons on the palette. When you idle the cursor over any icon on any palette, the full name of the subpalette, control, or indicator appears.
3. Click the `Numeric Controls` icon to access the `Numeric Controls` palette.
4. Select the knob control on the `Numeric Controls` palette and place it on the front panel to the left of the waveform graph. You will use this knob in a later exercise to control the amplitude of a signal.
5. Select `File≫Save As` and save this VI as `Acquiring a Signal.vi` in the `C:\Exercises\LabVIEW Basics I` directory.

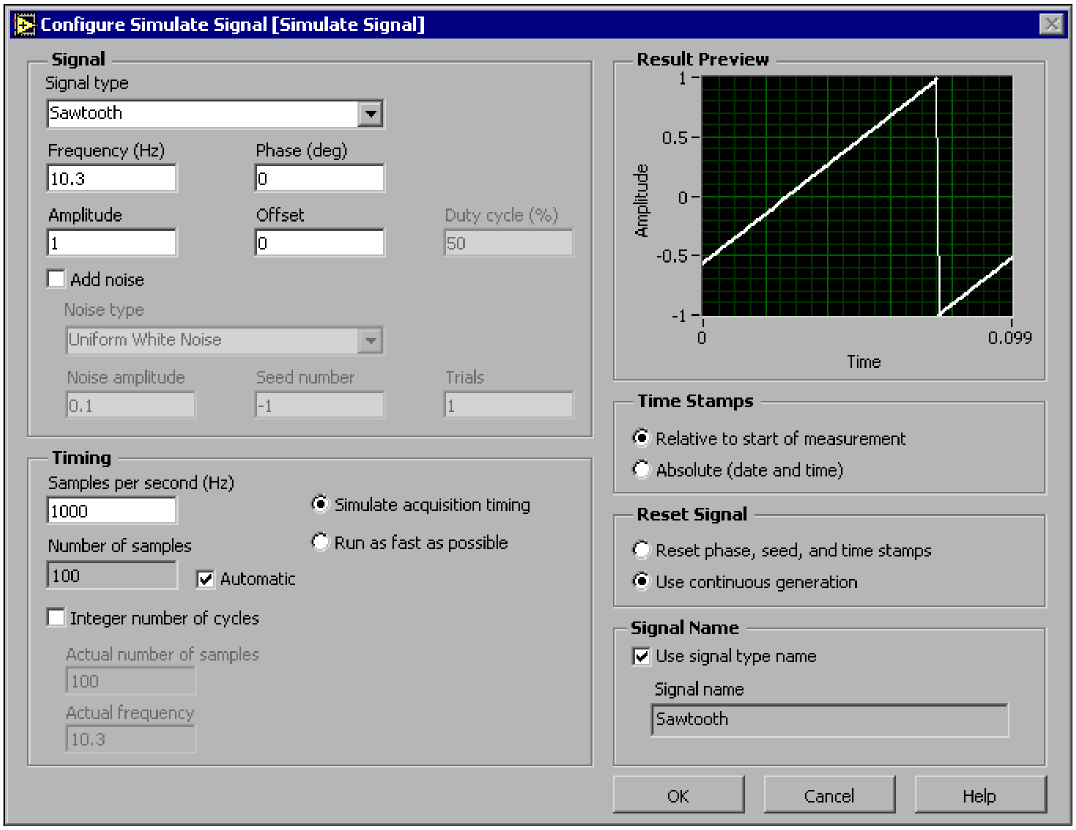
   NOTE: Save all the VIs you edit or create in this course in the `C:\Exercises\LabVIEW Basics I` directory.

1.5.2 Changing the Signal Type

The block diagram has a blue icon labeled `Simulate Signal`. This icon represents the `Simulate Signal` Express VI. The `Simulate Signal` Express VI simulates a sine wave by default. Complete the following steps to change this signal to a sawtooth wave.

1. Display the block diagram by selecting `Window≫Show Block Diagram` or by clicking the block diagram. 

   Notice the `Simulate Signal` Express VI, shown in this media. An Express VI is a component of the block diagram that you can configure to perform common measurement tasks. The `Simulate Signal` Express VI simulates a signal based on the configuration that you specify.
2. Right-click the `Simulate Signal` Express VI and select `Properties` from the shortcut menu to display the `Configure Simulate Signal` dialog box.
3. Select `Sawtooth` from the `Signal type` pull-down menu. Notice that the waveform on the graph in the `Result Preview` section changes to a sawtooth wave. The `Configure Simulate Signal` dialog box should appear similar to Figure 1.20.



Configure Simulate Signal [Simulate Signal]

Signal

Signal type: Sawtooth

Frequency (Hz): 10.3

Phase (deg): 0

Amplitude: 1

Offset: 0

Duty cycle (%): 50

□ Add noise

Noise type: Uniform White Noise

Noise amplitude: 0.1

Seed number: -1

Trials: 1

Timing

Samples per second (Hz): 1000

● Simulate acquisition timing

○ Run as fast as possible

Number of samples: 100 ☑ Automatic

□ Integer number of cycles

Actual number of samples: 100

Actual frequency: 10.3

Result Preview

Time Stamps

● Relative to start of measurement

○ Absolute (date and time)

Reset Signal

○ Reset phase, seed, and time stamps

● Use continuous generation

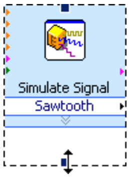
Signal Name

☑ Use signal type name

Signal name: Sawtooth

Figure 1.20

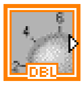
4. Click the `OK` button to apply the current configuration and close the `Configure Simulate Signal` dialog box.
5. Move the cursor over the down arrows at the bottom of the `Simulate Signal` Express VI.

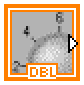


6. When a double-headed arrow appears, shown at left, click and drag the border of the Express VI until the `Amplitude` input appears. Notice how you expanded the `Simulate Signal` Express VI to display a new input. Because the `Amplitude` input appears on the block diagram, you can configure the amplitude of the sawtooth wave on the block diagram. In Figure 1.20, notice how `Amplitude` is an option in the `Configure Simulate Signal` dialog box. When inputs, such as `Amplitude`, appear on the block diagram and in the configuration dialog box, you can configure the inputs in either location.

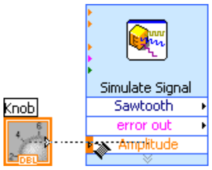
1.5.3 Wiring Objects on the Block Diagram

To use the knob control to change the amplitude of the signal, you must connect the two objects on the block diagram. Complete the following steps to wire the knob to the `Amplitude` input on the `Simulate Signal` Express VI.

1. 

Move the cursor over the Knob terminal, shown at left, until the Positioning tool appears. Notice how the cursor becomes an arrow, or the Positioning tool, shown in this media. Use the Positioning tool to select, position, and resize objects.
2.
Click the Knob terminal to select it, then drag the terminal to the left of the Simulate Signal Express VI. Make sure the Knob terminal is inside the loop, shown in this media. The terminals are representations of front panel controls and indicators. Terminals are entry and exit ports that exchange information between the front panel and block diagram.
3. Deselect the Knob terminal by clicking a blank space on the block diagram.
4. 

Move the cursor over the arrow of the Knob terminal, shown in this media. Notice how the cursor becomes a wire spool, or the Wiring tool, shown at left. Use the Wiring tool to wire objects together on the block diagram.

   NOTE: The cursor does not switch to another tool while an object is selected.
5. 

When the Wiring tool appears, click the arrow and then click the Amplitude input of the Simulate Signal Express VI, shown in this media, to wire the two objects together. Notice that a wire appears and connects the two objects. Data flows along this wire from the terminal to the Express VI.
6. Select File≫Save to save this VI.

1.5.4 Running the VI

Running a VI executes your solution. Complete the following steps to run the Acquiring a Signal VI.

1. Display the front panel by selecting Window≫Show Front Panel or by clicking the front panel.

   TIP: Press the Ctrl-E keys to switch from the front panel to the block diagram or from the block diagram to the front panel.
2. Click the Run button.
3. Move the cursor over the knob control. Notice how the cursor becomes a hand, or the Operating tool, shown at left. Use the Operating tool to change the value of a control or select the text within a control.
4. Using the Operating tool, turn the knob to adjust the amplitude of the sawtooth wave. Notice how the amplitude of the sawtooth wave changes as you turn the knob. Also notice that the y-axis on the graph autoscales to account for the change in amplitude. To indicate

that the VI is running, the Run button changes to a darkened arrow, shown at left. You cannot edit the front panel or block diagram while the VI runs.

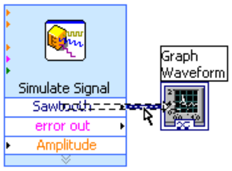
5.

Click the Stop button, shown at left, to stop the VI.

NOTE: Although Abort Execution button looks like a stop button, the Abort Execution button does not always properly close the VI. National Instruments recommends stopping your VIs using the Stop button on the front panel. Use the Abort Execution button only when errors prevent you from terminating the application using the Stop button.

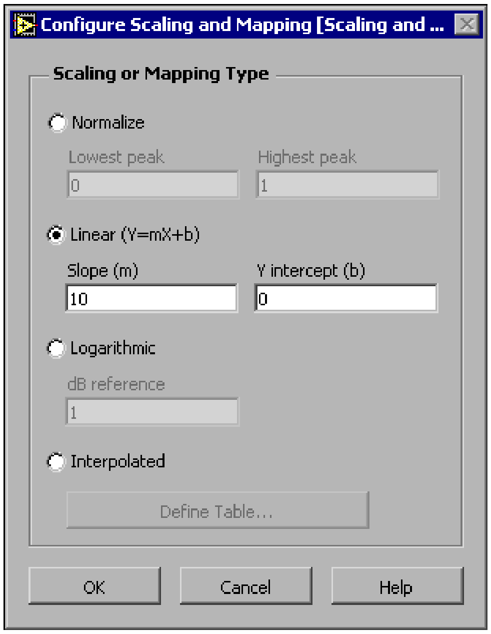
1.5.5 Modifying the Signal

Complete the following steps to add scaling to the signal and display the results in the graph on the front panel.

1. 

On the block diagram, use the Positioning tool to double-click the wire that connects the Simulate Signal Express VI to the Waveform Graph terminal shown in this media.
2. Press the Delete key to delete this wire.
3. If the Functions palette is not visible on the block diagram, select Window≫Show Functions Palette to display it.
4.

Select the Scaling and Mapping Express VI, shown at left, on the Arithmetic & Comparison palette and place it on the block diagram inside the loop between the Simulate Signal Express VI and the Waveform Graph terminal. If there is no room between the Express VI and the terminal, move the Waveform Graph terminal to the right. Notice that the Configure Scaling and Mapping dialog box automatically opens when you place the Express VI on the block diagram.
5. Define the value of the scaling factor by entering 10 in the Slope (m) text box. The Configure Scaling and Mapping dialog box should appear similar to Figure 1.21.



Configure Scaling and Mapping [Scaling and ...

Scaling or Mapping Type

○ Normalize

Lowest peak: 0

Highest peak: 1

● Linear (Y=mX+b)

Slope (m): 10

Y intercept (b): 0

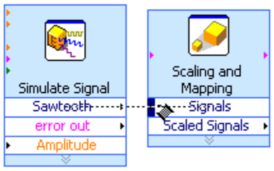
○ Logarithmic

dB reference: 1

○ Interpolated

Figure 1.21

6. Click the OK button to apply the current configuration and close the `Configure Scaling and Mapping` dialog box.
7. Move the cursor over the arrow on the `Sawtooth` output of the `Simulate Signal` Express VI.



8. When the Wiring tool appears, click the arrow and then click the arrow on the `Signals` input of the `Scaling and Mapping` Express VI, shown in this media, to wire the two objects together.
9. Using the `Wiring` tool, wire the `Scaled Signals` output of the `Scaling and Mapping` Express VI to the `Waveform Graph` terminal. Notice the wires connecting the Express VIs and terminals. The arrows on the Express VIs and terminals indicate the direction that the data flows along these wires. The block diagram should appear similar to Figure 1.22.

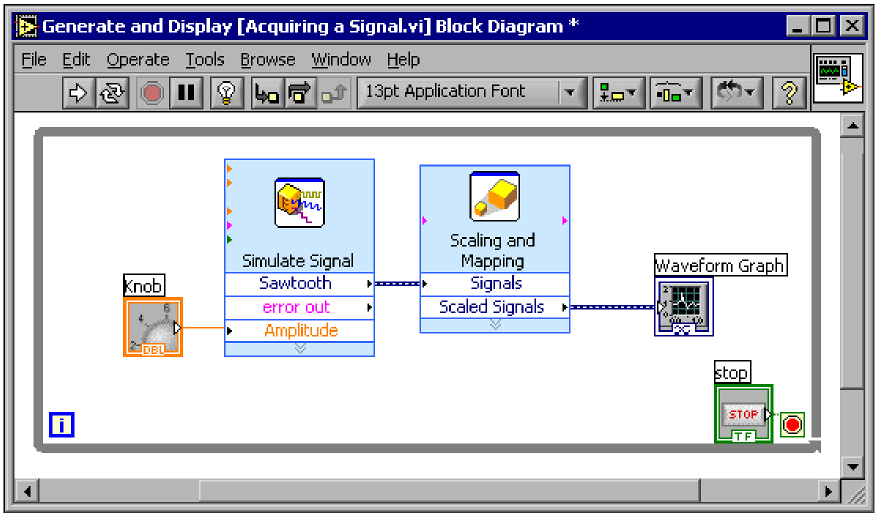


Figure 1.22

NOTE: The terminals in the block diagram are set to display as icons. To display a terminal as a data type on the block diagram, right-click the terminal and select `View As Icon` from the shortcut menu to remove the checkmark.

10. Select `File`≫`Save` to save this VI.

1.5.6 Displaying Two Signals on the Graph

To compare the signal generated by the `Simulate Signal` Express VI and the signal modified by the `Scaling and Mapping` Express VI on the same graph, use the `Merge Signals` function. Complete the following steps to display two signals on the same graph.

1. Move the cursor over the arrow on the `Sawtooth` output of the `Simulate Signal` Express VI.
2. Using the `Wiring` tool, wire the `Sawtooth` output to the `Waveform Graph` terminal. The `Merge Signals` function, shown in this media, appears where the two wires connect. This function takes the two separate signals and combines them so that both can be displayed on the same graph. The block diagram should appear similar to Figure 1.23.

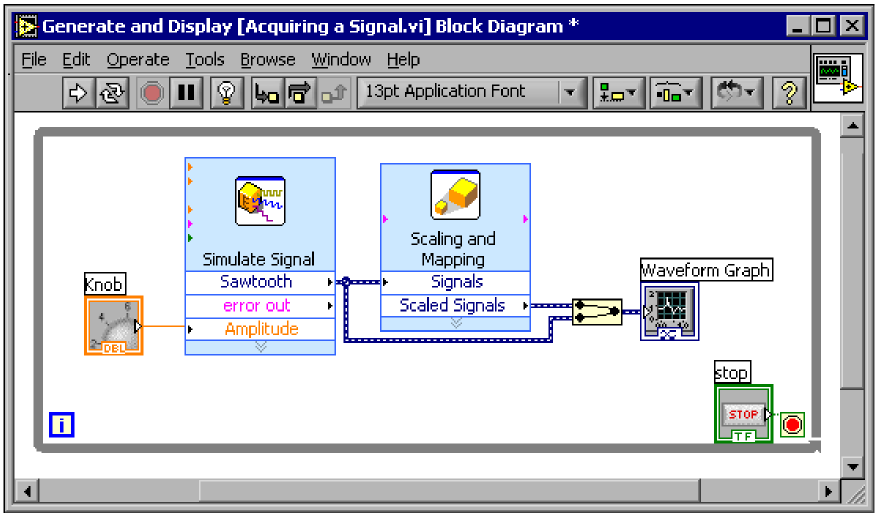


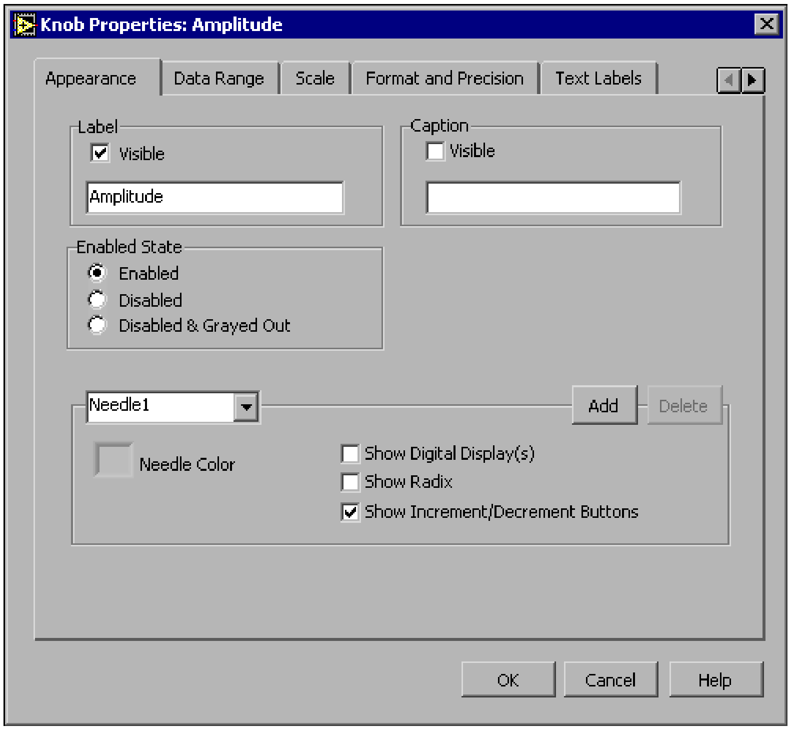
Figure 1.23

3. Select File≫Save to save this VI. You also can press the Ctrl-S keys to save a VI.
4. Return to the front panel, run the VI, and turn the knob control. Notice that the graph plots the sawtooth wave and the scaled signal. Also notice that the maximum value on the y-axis automatically changes to be 10 times the knob value. This scaling occurs because you set the slope to 10 in the Scaling and Mapping Express VI.
5. Click the Stop button.

1.5.7 Customizing the Knob

The knob control changes the amplitude of the sawtooth wave so labeling it Amplitude accurately describes the function of the knob. Complete the following steps to customize the appearance of a control on the front panel.

1. Right-click the knob and select Properties from the shortcut menu to display the Knob Properties dialog box.
2. In the Label section on the Appearance tab, delete the label Knob, and type Amplitude in the text box. The Knob Properties dialog box should appear similar to Figure 1.24.



Label
☑ Visible
Amplitude

Caption
□ Visible

Enabled State
● Enabled
○ Disabled
○ Disabled & Grayed Out

Needle1
Needle Color
□ Show Digital Display(s)
□ Show Radix
☑ Show Increment/Decrement Buttons

Figure 1.24

3. Click the `Scale` tab and, in the `Scale Range` section, change the maximum value to 5.0. Notice how the knob on the front panel instantly updates to reflect these changes.
4. Click the `OK` button to apply the current configuration and close the `Knob Properties` dialog box.
5. Save this VI.

   TIP: As you build VIs, you can experiment with different properties and configurations. You also can add and delete objects. Remember, you can undo recent edits by selecting `Edit`≫`Undo` or pressing the `Ctrl-Z` keys.

6. Experiment with other properties of the knob by using the `Knob Properties` dialog box. For example, try changing the colors for the `Marker Text Color` by clicking the color box located on the `Scale` tab.
7. Click the `Cancel` button to avoid applying the changes you made while experimenting. If you want to keep the changes you made, click the `OK` button.

1.5.8 Customizing the Waveform Graph

The `waveform graph` indicator displays the two signals. To indicate which plot is the scaled signal and which is the simulated signal, you customize the plots. Complete the following steps to customize the appearance of an indicator on the front panel.

1. Move the cursor over the top of the `plot legend` on the `waveform graph`. Notice that while there are two plots on the graph, the `plot legend` displays only one plot.

2. When a double-headed arrow appears, shown in Figure 1.25, click and drag the border of the plot legend until the second plot name appears.

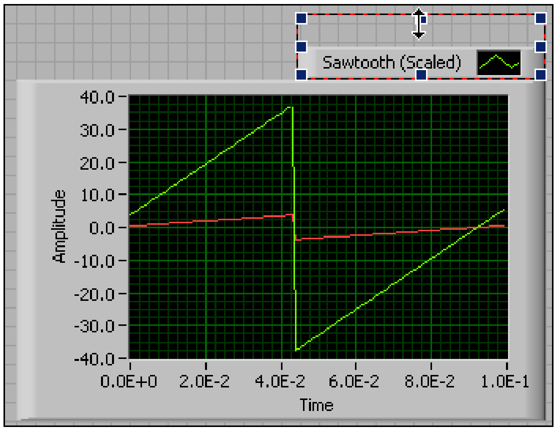


Figure 1.25

3. Right-click the waveform graph and select Properties from the shortcut menu to display the Graph Properties dialog box.
4. On the Plots tab, select Sawtooth from the pull-down menu. Click the Line Color color box to display the color picker. Select a new line color.
5. Select Sawtooth (Scaled) from the pull-down menu.
6. Place a checkmark in the Don't use waveform names for plot names checkbox.
7. In the Name text box, delete the current label and change the name of this plot to Scaled Sawtooth.
8. Click the OK button to apply the current configuration and close the Graph Properties dialog box. Notice how the plot color on the front panel changes.
9. Experiment with other properties of the graph by using the Graph Properties dialog box. For example, try disabling the autoscale feature located on the Scales tab.
10. Click the Cancel button to avoid applying the changes you made while experimenting. If you want to keep the changes you made, click the OK button.
11. Save and close this VI.

1.6 LabVIEW Documentation Resources[6]

[6] This content is available online at <http://cnx.org/content/m12199/1.1/>.

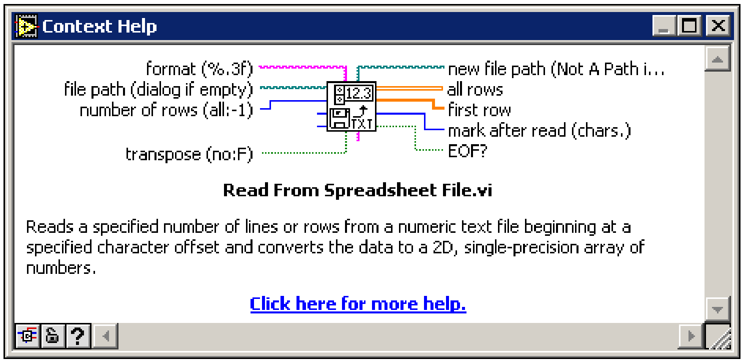
1.6.1 LabVIEW Documentation Resources

Use the `Context Help` window, the *LabVIEW Help*, and the NI Example Finder to help you build and edit VIs. Refer to the *LabVIEW Help* and manuals for more information about LabVIEW.

1.6.1.1 Context Help Window

The `Context Help` window displays basic information about LabVIEW objects when you move the cursor over each object. The `Context Help` window is visible by default. To toggle display of the `Context Help` window, select `Help`≫`Show Context Help`, press the `Ctrl-H` keys, or click the `Show Context Help Window` button, shown in this media, on the toolbar.

When you move the cursor over front panel and block diagram objects, the `Context Help` window displays the icon for subVIs, functions, constants, controls, and indicators, with wires attached to each terminal. When you move the cursor over dialog box options, the `Context Help` window displays descriptions of those options. In the window, required connections are bold, recommended connections are plain text, and optional connections are dimmed or do not appear. Figure 1.26 shows an example `Context Help` window.



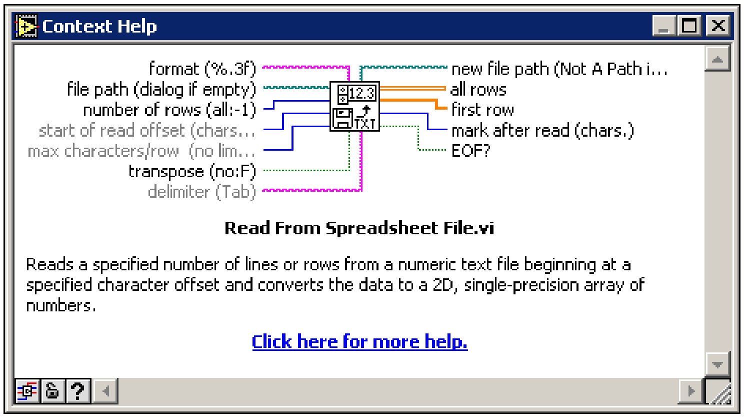
Read From Spreadsheet File.vi

Reads a specified number of lines or rows from a numeric text file beginning at a specified character offset and converts the data to a 2D, single-precision array of numbers.

Click here for more help.

Figure 1.26: Context Help Window

Click the `Hide Optional Terminals and Full Path` button located on the lower left corner of the `Context Help` window to display the optional terminals of a connector pane and to display the full path to a VI. Optional terminals are shown by wire stubs, informing you that other connections exist. The detailed mode displays all terminals, as shown in Figure 1.27.



Read From Spreadsheet File.vi

Reads a specified number of lines or rows from a numeric text file beginning at a specified character offset and converts the data to a 2D, single-precision array of numbers.

Click here for more help.

Figure 1.27: Detailed Context Help Window

Click the `Lock Context Help` button to lock the current contents of the `Context Help` window. When the contents are locked, moving the cursor over another object does not change the contents of the window. To unlock the window, click the button again. You also can access this option from the `Help` menu.

If a corresponding *LabVIEW Help* topic exists for an object the `Context Help` window describes, a blue `Click here for more help.` link appears in the `Context Help` window. Also, the `More Help` button, shown at left, is enabled. Click the link or the button to display the *LabVIEW Help* for more information about the object.

1.6.1.2 LabVIEW Help

You can access the *LabVIEW Help* either by clicking the `More Help` button in the `Context Help` window, selecting `Help≫VI, Function, & How-To Help`, or clicking the blue `Click here for more help.` link in the `Context Help` window.

The *LabVIEW Help* contains detailed descriptions of most palettes, menus, tools, VIs, and functions. The *LabVIEW Help* also includes step-by-step instructions for using LabVIEW features. The *LabVIEW Help* includes links to the following resources:

- *LabVIEW Bookshelf*, which includes PDF versions of all the LabVIEW manuals and Application Notes.
- Technical support resources on the National Instruments Web site[7] , such as the NI Developer Zone[8] , the Knowledge Base[9] , and the Product Manuals Library[10] .

1.6.1.3 NI Example Finder

The `New` dialog box contains many LabVIEW template VIs that you can use to start building VIs. However, these template VIs are only a subset of the hundreds of example VIs included with LabVIEW. You can modify any example VI to fit an application, or you can copy and paste from an example into a VI that you create.

In addition to the example VIs that ship with LabVIEW, you also can access hundreds of example VIs on the *NI Developer Zone* at ni.com/zone[11] . To search all examples using LabVIEW VIs, use the `NI Example Finder`. The `NI Example Finder` is the gateway to all installed examples and the examples located on the *NI Developer Zone*.

To launch the `NI Example Finder`, select `Help≫Find Examples` from the front panel or block diagram menu bar. You also can launch the `NI Example Finder` by clicking the arrow on the `Open` button on the LabVIEW dialog box and selecting `Examples` from the shortcut menu.

1.7 Reduce Samples VI[12]

Exercise 1.7.1

In the following exercises, you will open a blank VI and add Express VIs and structures to the block diagram to build a new VI. When you complete the exercise, the front panel of the VI will appear similar to the Figure 1.28.

[7] http://ni.com

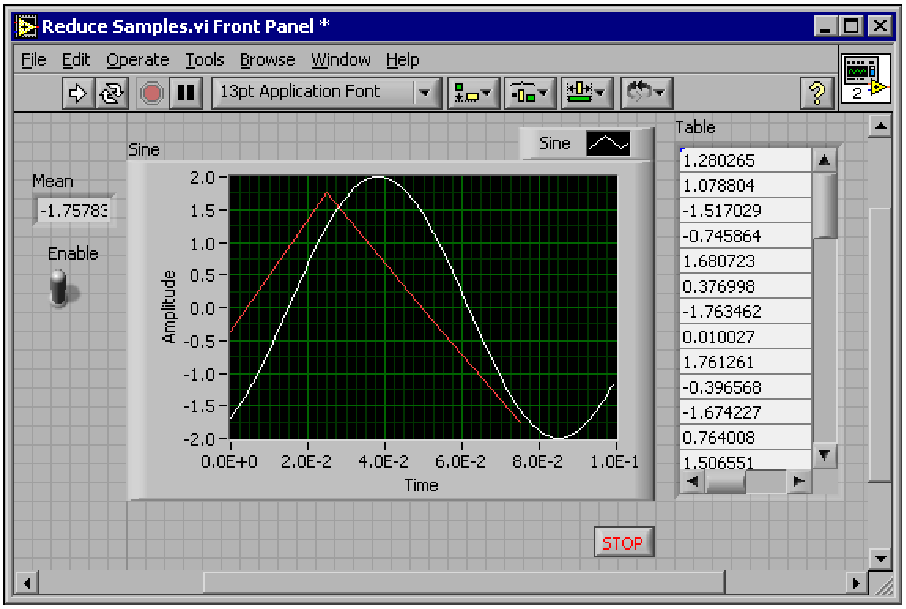
[8] http://zone.ni.com/zone/jsp/zone.jsp

[9] http://digital.ni.com/public.nsf/MainPage?OpenForm&node=133020_US

[10] http://digital.ni.com/manuals.nsf/MainPage?ReadForm&node=132020_US

[11] http://www.ni.com/zone

[12] This content is available online at <http://cnx.org/content/m12200/1.2/>.



Mean -1.75783

Table
1.280265
1.078804
-1.517029
-0.745864
1.680723
0.376998
-1.763462
0.010027
1.761261
-0.396568
-1.674227
0.764008
1.506551

Figure 1.28

1.7.1 Opening a Blank VI

If no template is available for the task you want to create, you can start with a blank VI and add Express VIs to accomplish the specific task. Complete the following steps to open a blank VI.

1. In the LabVIEW dialog box, click the arrow on the New button and select Blank VI from the shortcut menu or press the Ctrl-N keys to open a blank VI. Notice that a blank front panel and block diagram appear.
2. If the Functions palette is not visible, right-click any blank space on the block diagram to display the Functions palette. Click the thumbtack, shown in this media, in the upper left corner of the Functions palette to place the palette on the screen.

NOTE: You can right-click a blank space on the block diagram or the front panel to display the Functions or Controls palettes.

1.7.2 Adding an Express VI that Simulates a Signal

Complete the following steps to find the Express VI you want to use and then add it to the block diagram.

1. If the Context Help window is not visible, press the Ctrl-H keys to open the Context Help window. You also can press the Show Context Help Window button, shown in this media, to open the Context Help window.

2. Select the Input palette on the Functions palette and move the cursor over the Express VIs on the Input palette. Notice that the Context Help window displays information about the function of each Express VI.
3. From the information provided in the Context Help window, find the Express VI that can simulate a sine wave signal.
4. Select the Express VI and place it on the block diagram. The Configure Simulate Signal dialog box appears.
5. Idle the cursor over the various options in the Configure Simulate Signal dialog box, such as Frequency (Hz), Amplitude, and Samples per second (Hz). Read the information that appears in the Context Help window.
6. Configure the Simulate Signal Express VI to generate a sine wave with a frequency of 10.7 and amplitude of 2.
7. Notice how the signal displayed in the Result Preview window changes to reflect the configured sine wave.
8. Close the Configure Simulate Signal dialog box by clicking the OK button.
9. Move the cursor over the Simulate Signal Express VI and read the information that appears in the Context Help window. Notice that the Context Help window now displays the configuration of the Simulate Signal Express VI.
10. Save this VI as Reduce Samples.vi in the C:\Exercises\LabVIEW Basics I directory.

1.7.3 Modifying the Signal

Complete the following steps to use the *LabVIEW Help* to search for the Express VI that reduces the number of samples in a signal.

1. Select Help≫VI, Function, & How-To Help to open the *LabVIEW Help*.
2. Click the Search tab and type sample compression in the Type in the word(s) to search for text box. Notice that this word choice reflects what you want this Express VI to do-compress, or reduce, the number of samples in a signal.
3. To begin the search, press the Enter key or click the List Topics button.
4. Double-click the Sample Compression topic to display the topic that describes the Sample Compression Express VI.
5. After you read the description of the Express VI, click the Place on the block diagram button to select the Express VI.
6. Move the cursor to the block diagram. Notice how LabVIEW attaches the Sample Compression Express VI to the cursor.
7. Place the Sample Compression Express VI on the block diagram to the right of the Simulate Signal Express VI.
8. Configure the Sample Compression Express VI to reduce the signal by a factor of 25 using the mean of these values.
9. Close the Configure Sample Compression dialog box.
10. Using the Wiring tool, wire the Sine output in the Simulate Signal Express VI to the Signals input in the Sample Compression Express VI.

1.7.4 Customizing the Front Panel

In a previous exercise (Section 1.5), you added controls and indicators to the front panel using the Controls palette. You also can add controls and indicators from the block diagram. Complete the following steps to create controls and indicators as shown in Figure 1.29.

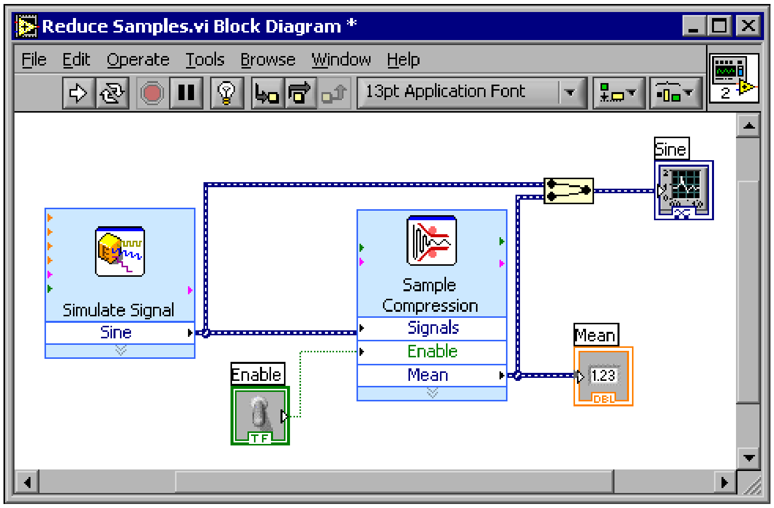


Figure 1.29

1. Right-click the Mean output in the Sample Compression Express VI and select Create≫Numeric Indicator from the shortcut menu to create a numeric indicator.
2. Right-click the Mean output of the Sample Compression Express VI and select Insert Input/Output from the shortcut menu to insert the Enable input.
3. Right-click the Enable input and select Create≫Control from the shortcut menu to create the Enableswitch.
4. Right-click the wire linking the Sine output in the Simulate Signal Express VI to the Signals input in the Signal Compression Express VI and select Create≫Graph Indicator from the shortcut menu. Notice that you can create controls and indicators from the block diagram. When you create controls and indicators using this method, LabVIEW automatically creates terminals that are labeled and formatted correctly.
5. Using the Wiring tool, wire the Mean output in the Sample Compression Express VI to the Sine graph indicator terminal created in the previous step. Notice that the Merge Signals function appears.
6. Arrange the objects on the block diagram so that they appear similar to Figure 1.29.

   TIP: You can right-click any wire and select Clean Up Wire from the shortcut menu to automatically route an existing wire.

7. Display the front panel. Notice that the controls and indicators you added automatically appear on the front panel with labels that correspond to their function.
8. Save this VI.

1.7.5 Configuring the VI to Run Continuously Until the User Stops It

In the current state, the VI runs once, generates one signal, then stops executing. To run the VI until a condition is met, you can add a While Loop to the block diagram. Complete the following steps to add a While Loop.

1. Display the front panel and run the VI. Notice how the VI runs once and then stops. Also notice how there is no Stop button.

2. Display the block diagram and select the While Loop on the `Functions≫Execution Control` palette.
3. Move the cursor to the upper left corner of the block diagram. Place the top left corner of the `While Loop` here.
4. Click and drag the cursor diagonally to enclose **all** the Express VIs and wires, as shown in Figure 1.30.

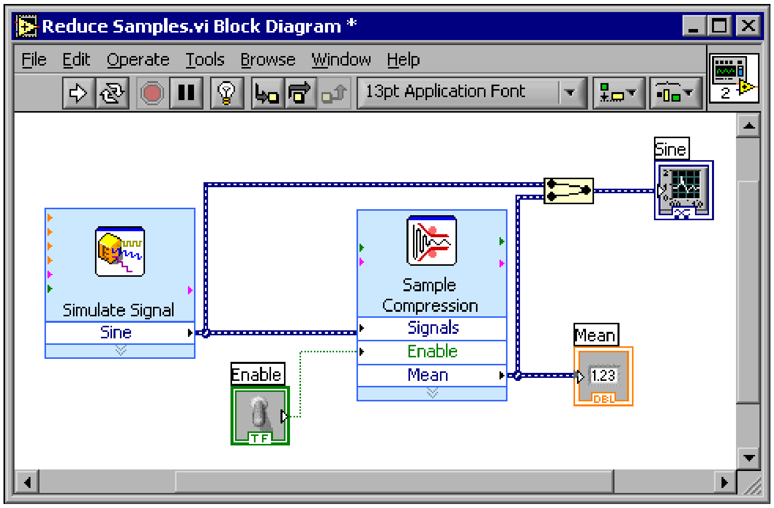
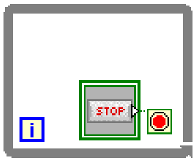


Figure 1.30



Notice that the `While Loop`, shown in this media, appears with a `Stop` button wired to the condition terminal. This `While Loop` is configured to stop when the user clicks the `Stop` button.

5. Display the front panel and run the VI. Notice that the VI now runs until you click the `Stop` button. A `While Loop` executes the functions inside the loop until the user presses the `Stop` button. Refer to Repetition and Loops (Section 3.1) for more information about `While Loops`.

1.7.6 Controlling the Speed of Execution

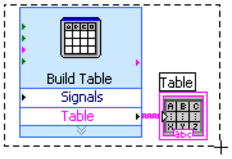
To plot the points on the waveform graph more slowly, you can add a time delay to the block diagram. Complete the following steps to control the speed at which the VI executes.

1. On the block diagram, select the `Time Delay` Express VI on the `Functions≫Execution Control` palette and place it inside the loop.
2. Type .250 in the `Time delay (seconds)` text box. This time delay specifies how fast the loop runs. With a .250 second time delay, the loop iterates once every quarter of a second.
3. Close the `Configure Time Delay` dialog box.
4. Save this VI.
5. Display the front panel and run the VI.

6. Click the `Enable` switch and notice the change on the graph. Notice how if the `Enable` switch is on, the graph displays the reduced signal. If the `Enable` switch is off, the graph does not display the reduced signal.
7. Click the `Stop` button to stop the VI.

1.7.7 Using a Table to Display Data

Complete the following steps to display a collection of mean values in a table on the front panel.

1. On the front panel, select the Express Table indicator on the `Controls`≫`Text Indicators` palette and place it on the front panel to the right of the waveform graph.
2. Display the block diagram. Notice that the `Table` terminal appears wired to the `Build Table` Express VI automatically.
3. 

If the `Build Table` Express VI and the `Table` terminal are not selected already, click an open space on the block diagram to the left of the `Build Table` Express VI and the `Table` terminal. Drag the cursor diagonally until the selection rectangle encloses the `Build Table` Express VI and the `Table` terminal, shown in this media. A moving dashed outline called a marquee highlights the `Build Table` Express VI, the `Table` terminal, and the wire joining the two.
4. Drag the objects into the `While Loop` to the right of the `Mean` terminal. Notice that the `While Loop` automatically resizes to enclose the `Build Table` Express VI and the `Table` terminal.
5. Using the `Wiring` tool, wire the `Mean` terminal of the `Sample Compression` Express VI to the `Signals` input of the `Build Table` Express VI. The block diagram should appear similar to Figure 1.31.

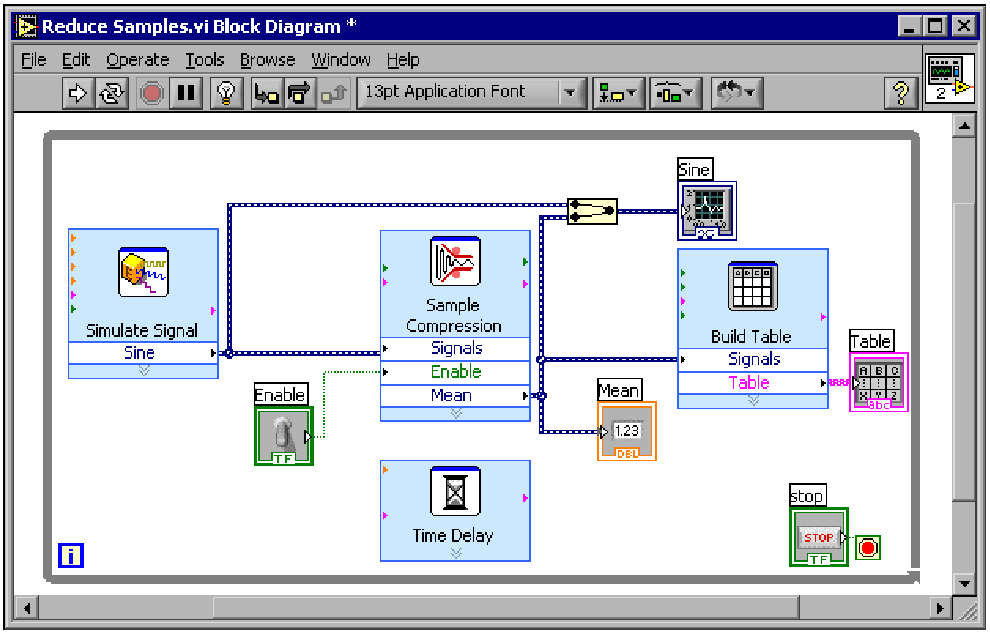


Figure 1.31

6. Display the front panel and run the VI.
7. Click the `Enable` switch. The table displays the mean values of every 25 samples of the sine wave. Notice if the `Enable` switch is off, the table does not record the mean values.
8. Stop the VI.
9. Experiment with properties of the table by using the `Table Properties` dialog box. For example, try changing the number of columns to one.
10. Save and close this VI.

1.8 Debugging Techniques[13]

If a VI does not run, it is a broken, or nonexecutable, VI. The `Run` button often appears broken, shown at left, when you create or edit a VI. If it is still broken when you finish wiring the block diagram, the VI is broken and will not run. Generally, this means that a required input is not wired, or a wire is broken.

1.8.1 Finding Errors

Click the broken `Run` button or select `Windows`≫`Show Error List` to display the `Error list` window, which lists all the errors. Double-click an error description to display the relevant block diagram or front panel and highlight the object that contains the error.

[13] This content is available online at <http://cnx.org/content/m12201/1.1/>.

1.8.2 Execution Highlighting

View an animation of the execution of the block diagram by clicking the `Highlight Execution` button, shown in this media. Execution highlighting shows the flow of data on the block diagram from one node to another using bubbles that move along the wires. Use execution highlighting in conjunction with single-stepping to see how data move from node to node through a VI.

> NOTE: Execution highlighting greatly reduces the speed at which the VI runs.

1.8.3 Single-Stepping

Single-step through a VI to view each action of the VI on the block diagram as the VI runs. The single-stepping buttons affect execution only in a VI or subVI in single-step mode. Enter single-step mode by clicking the `Step Over` or `Step Into` button. Move the cursor over the `Step Over`, `Step Into`, or `Step Out` button to view a tip strip that describes the next step if you click that button. You can single-step through subVIs or run them normally.

If you single-step through a VI with execution highlighting on, an execution glyph, shown in this media, appears on the icons of the subVIs that are currently running.

1.8.4 Probes

Use the `Probe` tool, shown in this media, to check intermediate values on a wire as a VFI runs. When execution pauses at a node because of single-stepping or a breakpoint, you also can probe the wire that just executed to see the value that flowed through that wire.

You also can create a custom probe to specify which indicator you use to view the probed data. For example, if you are viewing numeric data, you can choose to see that data in a chart within the probe. To create a custom probe, right-click a wire and select `Custom Probe≫New` from the shortcut menu.

1.8.5 Breakpoints

Use the `Breakpoint` tool, shown in this media, to place a breakpoint on a VI, node, or wire on the block diagram and pause execution at that locatiFon. When you set a breakpoint on a wire, execution pauses after data pass through the wire. Place a breakpoint on the block diagram workspace to pause execution after all nodes on the block diagram execute. When a VI pauses at a breakpoint, LabVIEW brings the block diagram to the front and uses a marquee to highlight the node or wire that contains the breakpoint. LabVIEW highlights breakpoints with red borders for nodes and block diagrams and red bullets for wires. When you move the cursor over an existing breakpoint, the black area of the `Breakpoint` tool cursor appears white. Use the `Breakpoint` tool to click an existing breakpoint to remove it.

1.9 Debug Exercise (Main) VI[14]

Exercise 1.9.1
Complete the following steps to load a broken VI and correct the error. Use single-stepping and execution highlighting to step through the VI.

[14] This content is available online at <http://cnx.org/content/m12202/1.1/>.

1.9.1 Front Panel

1. Select File≫Open and navigate to C:\Exercises\LabVIEW Basics I to open the Debug Exercise (Main) VI. The front panel shown in Figure 1.32 appears.

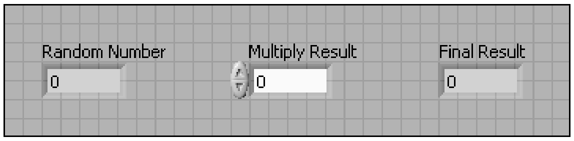


Figure 1.32

Notice the Run button on the toolbar appears broken, shown in this media, indicating that the VI is broken and cannot run.

1.9.2 Block Diagram

1. Select Window≫Show Block Diagram to display the block diagram shown in Figure 1.33.

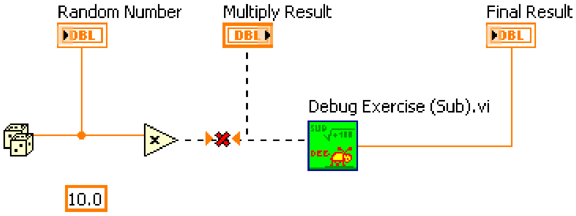


Figure 1.33

The Random Number (0-1) function, located on the Functions≫Arithmetic & Comparison≫Express Numeric palette, produces a random number between 0 and 1. The Multiply function, located on the Functions≫Arithmetic & Comparison≫Express Numeric palette, multiplies the random number by 10.0. 10.0 The numeric constant, located on the Functions≫Arithmetic & Comparison≫Express Numeric palette, is the number to multiply by the random number. The Debug Exercise (Sub) VI, located in the C:\Exercises\ LabVIEW Basics I directory, adds 100.0 and calculates the square root of the value.

2. Find and fix each error.

   a. Click the broken Run button to display the Error list window, which lists all the errors.

b. Select an error description in the `Error list` window. The `Details` section describes the error and in some cases recommends how to correct the error.
c. Click the `Help` button to display a topic in the *LabVIEW Help* that describes the error in detail and includes step-by-step instructions for correcting the error.
d. Click the `Show Error` button or double-click the error description to highlight the area on the block diagram that contains the error.
e. Use the `Error list` window to fix each error.

3. Select `File`≫`Save` to save the VI.
4. Display the front panel by clicking it or by selecting `Window`≫`Show Front Panel`.

1.9.3 Run the VI

1. Click the `Run` button to run the VI several times.
2. Select `Window`≫`Show Block Diagram` to display the block diagram.
3. Animate the flow of data through the block diagram.

   a. Click the `Highlight Execution` button, shown in this media, on the toolbar to enable execution highlighting.

   b. Click the `Step Into` button, shown in this media, to start single-stepping. Execution highlighting shows the movement of data on the block diagram from one node to another using bubbles that move along the wires. Nodes blink to indicate they are ready to execute.

   c. Click the `Step Over` button, shown in this media, after each node to step through the entire block diagram. Each time you click the `Step Over` button, the current node executes and pauses at the next node. Data appear on the front panel as you step through the VI. The VI generates a random number and multiplies it by 10.0. The subVI adds 100.0 and takes the square root of the result.

   d. When a blinking border surrounds the entire block diagram, click the `Step Out` button, shown in this media, to stop single-stepping through the `Debug Exercise (Main)` VI.

4. Single-step through the VI and its subVI.

   a. Click the `Step Into` button to start single-stepping.

   b. When the `Debug Exercise (Sub)` VI blinks, click the `Step Into` button. Notice the run button on the subVI.

   c. Display the `Debug Exercise (Main)` VI block diagram by clicking it. A green glyph, shown in this media, appears on the subVI icon on the `Debug Exercise (Main)` VI block diagram, indicating that the subVI is running.

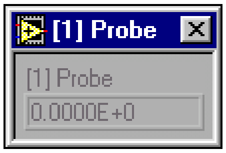
   d. Display the `Debug Exercise (Sub)` VI block diagram by clicking it.

   e. Click the `Step Out` button twice to finish single-stepping through the subVI block diagram. The `Debug Exercise (Main)` VI block diagram is active.

   f. Click the `Step Out` button to stop single-stepping.

5. Use a probe to check intermediate values on a wire as a VI runs.

   a. Use the Probe tool, shown this media, to click any wire. A window similar to Figure 1.34 appears.



[1] Probe

[1] Probe

0.0000E+0

Figure 1.34

   LabVIEW numbers the Probe window automatically and displays the same number in a glyph on the wire you clicked.

   b. Single-step through the VI again. The Probe window displays data passed along the wire.

6. Place breakpoints on the block diagram to pause execution at that location.

   a. Use the Breakpoint tool, shown in this media, to click nodes or wires. Place a breakpoint on the block diagram to pause execution after all nodes on the block diagram execute.

   b. Click the Run button to run the VI. When you reach a breakpoint during execution, the VI pauses and the Pause button on the toolbar appears red.

   c. Click the Continue button, shown this media, to continue running to the next breakpoint or until the VI finishes running.

   d. Use the Breakpoint tool to click the breakpoints you set and remove them.

7. Click the Highlight Execution button to disable execution highlighting.
8. Select File≫Close to close the VI and all open windows.

1.10 Summary, Tips, and Tricks on Introduction to LabVIEW[15]

[15] This content is available online at <http://cnx.org/content/m12203/1.2/>.

1.10.1 Summary

- Virtual instruments (VIs) contain three main components-the front panel, the block diagram, and the icon and connector pane.
- The front panel is the user interface of a VI and specifies the inputs and displays the outputs of the VI.
- The block diagram contains the graphical source code composed of nodes, terminals, and wires.
- Use the `Tools` palette to create, modify, and debug VIs. Press the `Shift` key and right-click to display a temporary version of the `Tools` palette at the location of the cursor.
- Use the `Controls` palette to place controls and indicators on the front panel. Right-click an open space on the front panel to display the `Controls` palette.
- Use the `Functions` palette to place VIs and functions on the block diagram. Right-click an open space on the block diagram to display the `Functions` palette.
- Use the `Search` button on the `Controls` and `Functions` palettes to search for controls, VIs, and functions.
- All LabVIEW objects and empty space on the front panel and block diagram have associated shortcut menus, which you access by right-clicking an object, the front panel, or the block diagram.
- Use the `Help` menu to display the `Context Help` window and the *LabVIEW Help*, which describes most palettes, menus, tools, VIs, functions, and features.
- Select `Help≫Search the LabVIEW Bookshelf` to display the *LabVIEW Bookshelf*, which you can use to search PDF versions of all the LabVIEW manuals and *Application Notes*.
- You build the front panel with controls and indicators, which are the interactive input and output terminals of the VI, respectively.
- Control terminals have a thicker border than indicator terminals. To change a control to an indicator or to change an indicator to a control, right-click the object and select `Change to Indicator` or `Change to Control` from the shortcut menu.
- The block diagram is composed of nodes, terminals, and wires.
- The broken `Run` button appears on the toolbar to indicate the VI is broken. Click the broken `Run` button to display the `Error list` window, which lists all the errors.
- Use execution highlighting, single-stepping, probes, and breakpoints to debug VIs by animating the flow of data through the block diagram.

1.10.2 Tips and Tricks

1.10.2.1 Operating

- Frequently used menu options have equivalent keyboard shortcuts. For example, to save a VI, you can select `File≫Save` or press the `Ctrl-S` keys. Common keyboard shortcuts include the following:

Ctrl-R - Runs a VI.
Ctrl-E - Toggles between the front panel and block diagram.
Ctrl-H - Toggles display of the `Context Help` window.
Ctrl-B - Removes all broken wires.
Ctrl-F - Finds VIs, globals, functions, text, or other objects loaded in memory or in a specified list of VIs.

- To increment or decrement numeric controls faster, use the `Operating` or `Labeling` tools to place the cursor in the control and press the `Shift` key while pressing the up or down arrow keys.
- You can disable the debugging tools to reduce memory requirements and to increase performance slightly. Select `File≫VI Properties`, select `Execution` from the top pull-down menu, and remove the checkmark from the `Allow Debugging` checkbox.

1.10.2.2 Wiring

- Click the `Show Context Help Window` button on the toolbar to display the `Context Help` window. Use the `Context Help` window to determine which terminals are required. Required terminals are bold, recommended connections are plain text, and optional connections are dimmed.
- Press the spacebar to toggle the wire direction.
- To move objects one pixel, press the arrow keys. To move objects several pixels, press the `Shift` key while you press the arrow keys.
- To cancel a wire you started, press the `Esc` key, right-click, or click the terminal where you started the wire.
- Use the tip strips that appear as you move the `Wiring` tool over terminals.
- Display the connector pane by right-clicking the node and selecting `Visible Items≫Terminals` from the shortcut menu.
- You can bend a wire by clicking to tack the wire down and moving the cursor in a perpendicular direction. To tack down a wire and break it, double-click.

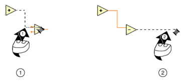


Figure 1.35: 1. Tack Down a Wire by Clicking, 2. Tack and Break the Wire by Double-clicking

1.10.2.3 Editing

- Use the following shortcuts to create constants, controls, and indicators:
  - Right-click a function terminal and select `Create≫Constant`, `Create≫Control`, or `Create≫Indicator` from the shortcut menu.
  - Drag controls and indicators from the front panel to the block diagram to create a constant.
  - Drag constants from the block diagram to the front panel to create a control.
- To duplicate an object, press the `Ctrl` key while using the Positioning tool to click and drag a selection.
- To restrict an object's direction of movement horizontally or vertically, use the `Positioning` tool to select the object and press the `Shift` key while you move the object.
- To keep an object proportional to its original size as you resize it, press the `Shift` key while you drag the resizing handles or circles.
- To resize an object as you place it on the front panel, press the `Ctrl` key while you click to place the object and drag the resizing handles or circles.
- To replace nodes, right-click the node and select `Replace` from the shortcut menu.
- To display the block diagram of a subVI from the calling VI, press the `Ctrl` key and use the `Operating` or `Positioning` tool to double-click the subVI on the block diagram.
- To display the front panel of a subVI from the calling VI, use the `Operating or Positioning` tool to double-click the subVI on the block diagram. You also can select `Browse≫This VI's SubVIs`.
- After you type a label, press the `Enter` key to end text entry.
- To add items quickly to ring controls and `Case` structures, press the `Shift-Enter` keys after each item. Pressing `Shift-Enter` accepts the item and positions the cursor to add the next item. Refer to Making Decisions in a VI[16] for more information about `Case` structures.

[16] "Case Structures" <http://cnx.org/content/m12245/latest/>

- To copy the color of one object and transfer it to a second object without using a color picker, use the `Color Copy` tool to click the object whose color you want to copy. Use the `Coloring` tool to click the object to which you want to apply the color. You also can copy the color of one object by using the `Coloring` tool and pressing the `Ctrl` key.
- Select `Edit`≫`Undo` if you make a mistake.
- To create more blank space on the block diagram, press the `Ctrl` key while you use the `Positioning` tool to draw a rectangle on the block diagram.

1.10.2.4 Debugging

- When single-stepping, use the following keyboard shortcuts:

Ctrl-down arrow - Steps into a node.
Ctrl-right arrow - Steps over a node.
Ctrl-up arrow - Steps out of a node.

Chapter 2

Modular Programming

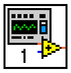
2.1 Modular Programming[1]

The power of LabVIEW lies in the hierarchical nature of the VI. After you create a VI, you can use it on the block diagram of another VI. There is no limit on the number of layers in the hierarchy. Using modular programming helps you manage changes and debug the block diagram quickly.

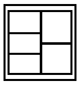
A VI within another VI is called a subVI. A subVI corresponds to a subroutine in text-based programming languages. When you double-click a subVI, a front panel and block diagram appear, rather than a dialog box in which you can configure options. The front panel includes controls and indicators that might look familiar. The block diagram includes wires, front panel icons, functions, possibly subVIs, and other LabVIEW objects that also might look familiar.

The upper right corner of the front panel and block diagram displays the icon for the VI. This icon is the same as the icon that appears when you place the VI on the block diagram.

2.1.1 Icon and Connector Pane



After you build a front panel and block diagram, build the icon and the connector pane so you can use the VI as a subVI. Every VI displays an icon, such as the one shown in this media, in the upper right corner of the front panel and block diagram windows. An icon is a graphical representation of a VI. It can contain text, images, or a combination of both. If you use a VI as a subVI, the icon identifies the subVI on the block diagram of the VI. You can double-click the icon to customize or edit it.



You also need to build a connector pane, shown in this media, to use the VI as a subVI. The connector pane is a set of terminals that correspond to the controls and indicators of that VI, similar to the parameter list of a function call in text-based programming languages. The connector pane defines the inputs and outputs you can wire to the VI so you can use it as a subVI. A connector pane receives data at its input terminals and passes the data to the block diagram code through the front panel controls and receives the results at its output terminals from the front panel indicators.

As you create VIs, you might find that you perform a certain operation frequently. Consider using subVIs or loops to perform that operation repetitively. For example, the block diagram in Figure 2.1 contains two identical operations.

[1] This content is available online at <http://cnx.org/content/m12204/1.2/>.

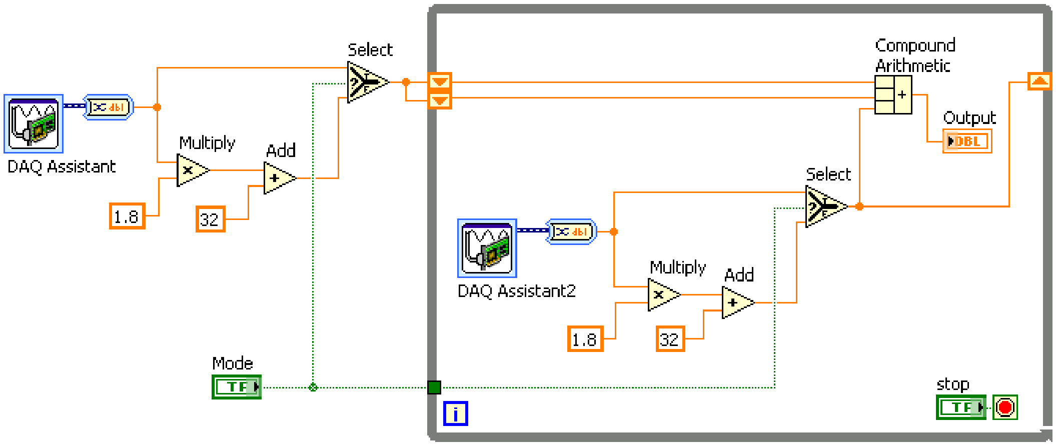


Figure 2.1

You can create a subVI that performs that operation and call the subVI twice. The example in Figure 2.2 calls the `Temperature` VI as a subVI twice on its block diagram and functions the same as the previous block diagram.You also can reuse the subVI in other VIs. Refer to Repetition and Loops (Section 3.1) for more information about using loops to combine common operations.

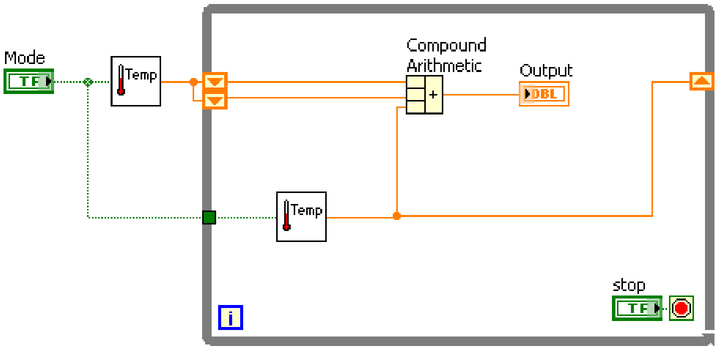


Figure 2.2

Refer to the *LabVIEW Basics II: Development Course Manual* for more information about application development. The following pseudo-code and block diagrams demonstrate the analogy between subVIs and subroutines.

Function Code	**Calling Program Code**

continued on next page

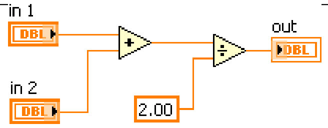
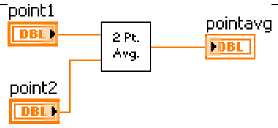
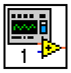
`main { average (point1, point2, pointavg) }`	`function average (in1, in2, out) { out = (in1 + in2)/2.0; }`
SubVI Block Diagram  	CallingVI Block Diagram  point1 DBL point2 DBL 2 Pt. Avg. pointavg DBL

Table 2.1

2.2 Icons and Connector Panes[2]

After you build a VI front panel and block diagram, build the icon and the connector pane so you can use the VI as a subVI.

2.2.1 Creating an Icon



Every VI displays an icon, shown in this media, in the upper right corner of the front panel and block diagram windows. An icon is a graphical representation of a VI. It can contain text, images, or a combination of both. If you use a VI as a subVI, the icon identifies the subVI on the block diagram of the VI.

The default icon contains a number that indicates how many new VIs you have opened since launching LabVIEW. Create custom icons to replace the default icon by right-clicking the icon in the upper right corner of the front panel or block diagram and selecting `Edit Icon` from the shortcut menu or double-clicking the icon in the upper right corner of the front panel. You also can edit icons by selecting `File≫VI Properties` selecting `General` from the `Category` pull-down menu, and clicking the `Edit Icon` button.

Use the tools on the left side of the `Icon Editor` dialog box to create the icon design in the editing area. The normal size image of the icon appears in the appropriate box to the right of the editing area, as shown in the dialog box in Figure 2.3.

[2] This content is available online at <http://cnx.org/content/m12205/1.2/>.

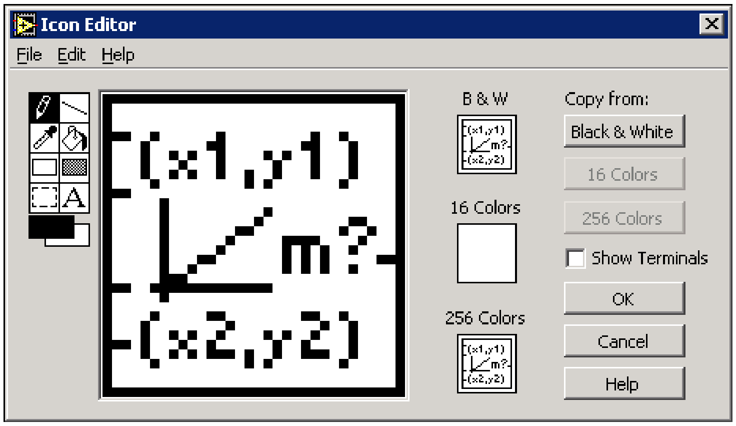


Figure 2.3

Depending on the type of monitor you use, you can design a separate icon for monochrome, 16-color, and 256-color mode. LabVIEW uses the monochrome icon for printing unless you have a color printer.

Use the `Edit` menu to cut, copy, and paste images from and to the icon. When you select a portion of the icon and paste an image, LabVIEW resizes the image to fit into the selection area. You also can drag a graphic from anywhere in your file system and drop it in the upper right corner of the front panel or block diagram. LabVIEW converts the graphic to a 32 × 32 pixel icon.

Use the `Copy from` option on the right side of the `Icon Editor` dialog box to copy from a color icon to a black-and-white icon and vice versa. After you select a `Copy from` option, click the `OK` button to complete the change.

> NOTE: If you do not draw a complete border around a VI icon, the icon background appears transparent. When you select the icon on the block diagram, a selection marquee appears around each individual graphic element in the icon.

Use the tools on the left side of the `Icon Editor` dialog box to create the icon design in the editing area. The normal size image of the icon appears in the appropriate box to the right of the editing area. The following tasks can be performed with these tools:

- Use the `Pencil` tool to draw and erase pixel by pixel.
- Use the `Line` tool to draw straight lines. To draw horizontal, vertical, and diagonal lines, press the <`Shift`> key while you use this tool to drag the cursor.
- Use the `Color Copy` tool to copy the foreground color from an element in the icon.
- Use the `Fill` tool to fill an outlined area with the foreground color.
- Use the `Rectangle` tool to draw a rectangular border in the foreground color. Double-click this tool to frame the icon in the foreground color.
- Use the `Filled Rectangle` tool to draw a rectangle with a foreground color frame and filled with the background color. Double-click this tool to frame the icon in the foreground color and fill it with the background color.

- Use the Select tool to select an area of the icon to cut, copy, move, or make other changes. Double-click this tool and press the <Delete> key to delete the entire icon.
- Use the Text tool to enter text into the icon. Double-click this tool to select a different font. (Windows) The Small Fonts option works well in icons.
- Use the Foreground/Background tool to display the current foreground and background colors. Click each rectangle to display a color palette from which you can select new colors.
- Use the options on the right side of the editing area to perform the following tasks:

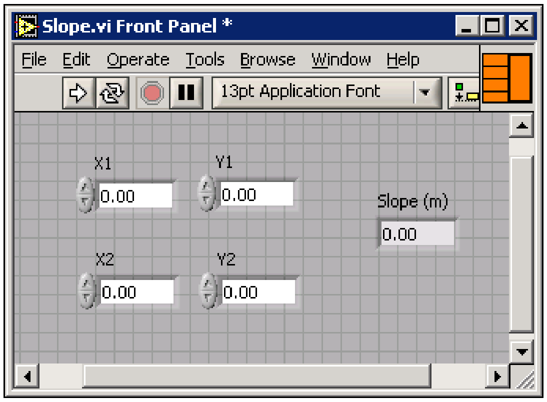
  Show Terminals - Displays the terminal pattern of the connector pane.
  OK - Saves the drawing as the icon and returns to the front panel.
  Cancel - Returns to the front panel without saving any changes.

- The menu bar in the Icon Editor dialog box contains more editing options such as Undo, Redo, Cut, Copy, Paste ,and Clear .

2.2.2 Setting Up the Connector Pane

To use a VI as a subVI, you need to build a **connector pane**, shown in this media. The connector pane is a set of terminals that corresponds to the controls and indicators of that VI, similar to the parameter list of a function call in text-based programming languages. The connector pane defines the inputs and outputs you can wire to the VI so you can use it as a subVI.

Define connections by assigning a front panel control or indicator to each of the connector pane terminals. To define a connector pane, right-click the icon in the upper right corner of the front panel window and select Show Connector from the shortcut menu. The connector pane replaces the icon. Each rectangle on the connector pane represents a terminal. Use the rectangles to assign inputs and outputs. The number of terminals LabVIEW displays on the connector pane depends on the number of controls and indicators on the front panel. The following front panel has four controls and one indicator, so LabVIEW displays four input terminals and one output terminal on the connector pane.



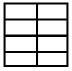
X1 0.00
Y1 0.00
X2 0.00
Y2 0.00
Slope (m) 0.00

Figure 2.4

2.2.3 Selecting and Modifying Terminal Patterns

Select a different terminal pattern for a VI by right-clicking the connector pane and selecting `Patterns` from the shortcut menu. Select a connector pane pattern with extra terminals. You can leave the extra terminals unconnected until you need them. This flexibility enables you to make changes with minimal effect on the hierarchy of the VIs. You also can have more front panel controls or indicators than terminals.

A solid border highlights the pattern currently associated with the icon. The maximum number of terminals available for a subVI is 28.

 The most commonly used pattern is shown in this media. This pattern is used as a standard to assist in simplifying wiring. The top inputs and outputs are commonly used for passing references and the bottom inputs and outputs are used for error handling. Refer to the section on Clusters[3] for more information about error handling.

> NOTE: Try not to assign more than 16 terminals to a VI. Too many terminals can reduce the readability and usability of the VI.

To change the spatial arrangement of the connector pane patterns, right-click the connector pane and select `Flip Horizontal, Flip Vertical, or Rotate 90 Degrees` from the shortcut menu.

2.2.4 Assigning Terminals to Controls and Indicators

After you select a pattern to use for the connector pane, you must define connections by assigning a front panel control or indicator to each of the connector pane terminals. When you link controls and indicators to the connector pane, place inputs on the left and outputs on the right to prevent complicated, unclear wiring patterns in your VIs.

To assign a terminal to a front panel control or indicator, click a terminal of the connector pane, then click the front panel control or indicator you want to assign to that terminal. Click an open space on the front panel. The terminal changes to the data type color of the control to indicate that you connected the terminal.

You also can select the control or indicator first and then select the terminal.

> NOTE: Although you use the `Wiring` tool to assign terminals on the connector pane to front panel controls and indicators, no wires are drawn between the connector pane and these controls and indicators.

2.3 Convert C to F VI[4]

Exercise 2.3.1

Complete the following steps to create a VI that takes a number representing degrees Celsius and converts it to a number representing degrees Fahrenheit.

2.3.1 Front Panel

1. Open a blank VI and begin building the following front panel.

[3] "Error Clusters" <http://cnx.org/content/m12231/latest/>

[4] This content is available online at <http://cnx.org/content/m12207/1.2/>.

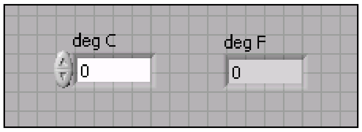


Figure 2.5

2. (Optional) Select `Window≫Tile Left and Right` to display the front panel and block diagram side by side or `Window≫Tile Up and Down` to display the front panel and block diagram stacked.
3. Create a numeric control. You will use this control to enter the value for degrees Celsius.
   a. Select `Controls≫Numeric Controls` to display the `Numeric Controls` palette. If the `Controls` palette is not visible, right-click an open space on the front panel workspace to display it.
   b. Select the `Numeric Control`. Move the control to the front panel and click to place the control.
   c. Type `deg C` in the label of the control and press the <`Enter`> key or click the `Enter` button, shown in this media, on the toolbar. If you do not type the name immediately, LabVIEW uses a default label.

   NOTE: You can edit a label at any time by double-clicking the label, using the `Labeling` tool, or right-clicking and selecting `Properties` from the shortcut menu to display the property dialog box.
4. Create a numeric indicator. You will use this indicator to display the value for degrees Fahrenheit.
   a. Select the `Numeric Indicator` located on the `Controls≫Numeric Indicators` palette.
   b. Move the indicator to the front panel and click to place the indicator.
   c. Type `deg F` in the label and press the <`Enter`> key or click the `Enter` button.

2.3.2 Block Diagram

1. Display the block diagram by clicking it or by selecting `Window≫ Show Block Diagram`. LabVIEW creates corresponding control and indicator terminal icons on the block diagram when you place controls and indicators on the front panel. The terminals represent the data type of the control or indicator. You should see two double-precision, floating-point terminals on the block diagram, one indicator, and one control.

   NOTE: Control terminals have a thicker border than indicator terminals.
2. Place the `Multiply` function, located on the `Functions≫Arithmetic & Comparison≫Express Numeric` palette, on the block diagram to the right of the `deg C` indicator. If the `Functions` palette is not visible, right-click an open space on the block diagram workspace to display it.

3. Place the `Add` function, located on the `Functions≫Arithmetic & Comparison≫Express Numeric` palette, on the block diagram to the right of the `Multiply` function.
4. Place a `Numeric Constant`, located on the `Functions≫Arithmetic & Comparison≫Express Numeric` palette, to the lower left of the `Multiply` function. Type `1.80` in the constant. When you first place a numeric constant, it is highlighted so you can type a value. If the constant is no longer highlighted, double-click the constant to activate the `Labeling` tool.
5. Place a `Numeric Constant`, located on the `Functions≫Arithmetic & Comparison≫Express Numeric` palette, to the left of the `Add` function. Type `32.0` in the constant.
6. Use the `Wiring` tool, shown in this media, to wire the icons as shown in Figure 2.6.

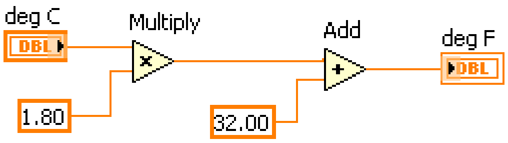


Figure 2.6

- To wire from one terminal to another, use the `Wiring` tool to click the first terminal, move the tool to the second terminal, and click the second terminal. You can start wiring at either terminal.
- You can bend a wire by clicking to tack down the wire and moving the cursor in a perpendicular direction. Press the `spacebar` to toggle the wire direction.
- To identify terminals on the nodes, right-click the `Multiply` and `Add` functions and select `Visible Items≫Terminals` from the shortcut menu to display the connector pane on the block diagram. Return to the icons after wiring by right-clicking the functions and selecting `Visible Items≫Terminals` from the shortcut menu to remove the checkmark.
- When you move the `Wiring` tool over a terminal, the terminal area blinks, indicating that clicking will connect the wire to that terminal and a tip strip appears, displaying the name of the terminal. If the `Context Help` window is open, the terminal area also blinks in the `Context Help` window.
- To cancel a wire you started, press the <`Esc`> key, right-click, or click the terminal where you started the wire.

7. Display the front panel by clicking it or by selecting `Window≫Show Front Panel`.
8. Save the VI as `Convert C to F.vi` in the `C:\Exercises\LabVIEW Basics I` directory.

2.3.3 Run the VI

1. Enter a number in the numeric control and run the VI.

a. Use the `Operating` tool, shown in this media, or the `Labeling` tool to double-click the numeric control and type a new number.
b. Click the `Run` button, shown at left, to run the VI.
c. Try several different numbers and run the VI again.

2.3.4 Icon and Connector Pane

1. Right-click the icon in the upper right corner of the front panel window and select `Edit Icon` from the shortcut menu. The `Icon Editor` dialog box appears.
2. Double-click the `Select` tool, shown in this media, on the left side of the `Icon Editor` dialog box to select the default icon.
3. Press the <`Delete`> key to remove the default icon.
4. Double-click the `Rectangle` tool, shown in this media, to redraw the border.
5. Create the icon in Figure 2.7.

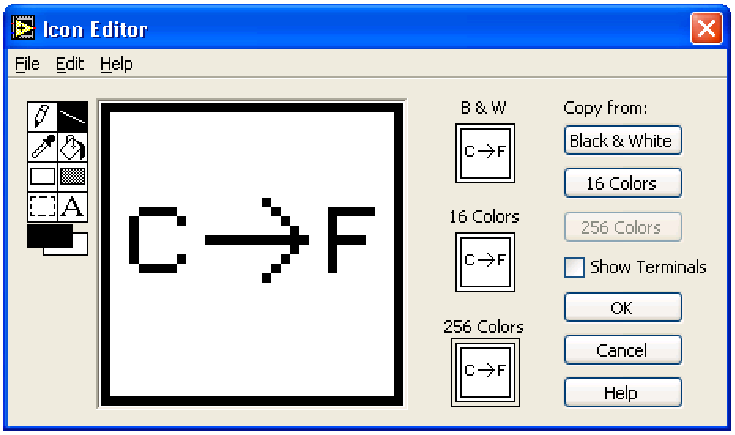
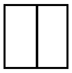


Figure 2.7

a. Double-click the `Text` tool, shown in this media, and change the font to `Small Fonts` .
b. Use the `Text` tool to click the editing area where you will begin typing.
c. Type `C` and `F`. While the text is active, you can move the text by pressing the arrow keys.
d. Use the `Pencil` tool, shown in this media, to create the arrow.

   NOTE: To draw horizontal or vertical straight lines, press the <`Shift`> key while you use the `Pencil` tool to drag the cursor.
e. Use the `Select` tool and the arrow keys to move the text and arrow you created.
f. Select the `B & W` icon and click the `256 Colors` button in the `Copy from` section to create a black and white icon, which LabVIEW uses for printing unless you have a color printer.

g. Select the `16 Colors` icon and click the `256 Colors` button in the `Copy from` section.
h. When you complete the icon, click the `OK` button to close the `Icon Editor` dialog box. The icon appears in the upper right corner of the front panel and block diagram.

6. Right-click the icon on the front panel and select `Show Connector` from the shortcut menu to define the connector pane terminal pattern. LabVIEW selects a default connector pane pattern based on the number of controls and indicators on the front panel. For example, this front panel has two terminals, `deg C` and `deg F` , so LabVIEW selects a connector pane pattern with two terminals, shown in this media.
7. Assign the terminals to the numeric control and numeric indicator.

   a. Select `Help≫Show Context Help` to display the `Context Help` window.
   b. Click the left terminal in the connector pane. The tool automatically changes to the `Wiring` tool, and the terminal turns black.
   c. Click the `deg C` control. A marquee highlights the control on the front panel.
   d. Click an open space on the front panel. The marquee disappears, and the terminal changes to the data type color of the control to indicate that you connected the terminal.
   e. Click the right terminal in the connector pane, and click the `deg F` indicator.
   f. Click an open space on the front panel. Both terminals of the connector pane are orange.
   g. Move the cursor over the connector pane. The `Context Help` window shows that both terminals are connected to double-precision, floating-point values.

8. Save and close the VI. You will use this VI later in the course.

2.4 Using SubVIs[5]

After you build a VI and create its icon and connector pane, you can use the VI as a subVI. To place a subVI on the block diagram, select `Functions≫All Functions≫Select a VI` . Navigate to the VI you want to use as a subVI and and double-click to place it on the block diagram.

You also can place an open VI on the block diagram of another open VI. Use the `Positioning` tool to click the icon in the upper right corner of the front panel or block diagram of the VI you want to use as a subVI and drag the icon to the block diagram of the other VI.

2.4.1 Opening and Editing SubVIs

To display the front panel of a subVI from the calling VI, use the `Operating` or `Positioning` tool to double-click the subVI on the block diagram. You also can select `Browse≫This VI's SubVIs`. To display the block diagram of a subVI from the calling VI, press the <`Ctrl`> key and use the `Operating` or `Positioning` tool to double-click the subVI on the block diagram.

Any changes you make to a subVI affect only the current instance of the subVI until you save the subVI. When you save the subVI, the changes affect all calls to the subVI, not just the current instance.

2.4.2 Setting Required, Recommended, and Optional Inputs and Outputs

In the `Context Help` window, which you can access by selecting `Help≫Show Context Help` required terminals appear bold, recommended terminals appear as plain text, and optional terminals appear

[5] This content is available online at <http://cnx.org/content/m12208/1.1/>.

dimmed. The labels of optional terminals do not appear if you click the `Hide Optional Terminals and Full Path` button in the `Context Help` window.

You can designate which inputs and outputs are required, recommended, and optional to prevent users from forgetting to wire subVI terminals.

Right-click a terminal in the connector pane and select `This Connection Is` from the shortcut menu. A checkmark indicates the terminal setting. Select `Required`, `Recommended`, or `Optional` .

For terminal inputs, required means that the block diagram on which you placed the subVI will be broken if you do not wire the required inputs. Required is not available for terminal outputs. For terminal inputs and outputs, recommended or optional means that the block diagram on which you placed the subVI can execute if you do not wire the recommended or optional terminals. If you do not wire the terminals, the VI does not generate any warnings.

LabVIEW sets inputs and outputs of VIs you create to `Recommended` by default. Set a terminal setting to required only if the VI must have the input or output to run properly. Refer to the Read File function located on the `Functions≫All Functions≫File I/O` palette for examples of required, recommended, and optional inputs and outputs.

2.5 Thermometer VI[6]

Exercise 2.5.1

Complete the following steps to create a VI that reads a measurement from the temperature sensor on the DAQ Signal Accessory and displays the temperature in Celsius or Fahrenheit.

The sensor returns a voltage proportional to temperature. For example, if the temperature is 23 °, the sensor output voltage is 0.23 V. The sensor is connected to Channel 0 of Device 1. Device 1 is the DAQ device. On some systems, the DAQ device may have another device number.

2.5.1 Front Panel

1. In the `LabVIEW` dialog box, click the arrow on the `New` button and select `Blank` VI from the shortcut menu or press the <`Ctrl-N`> keys to open a blank VI.
2. Create the following front panel.

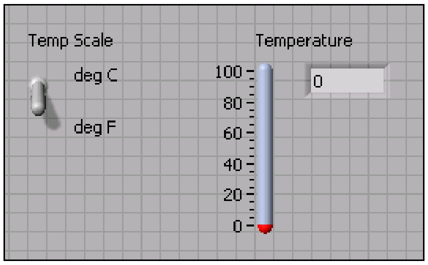


Figure 2.8

a. Place a thermometer, located on the `Controls≫Numeric Indicators` palette, on the front panel.

b. ✓ Type `Temperature` in the label and press the <`Enter`> key or click the `Enter` button on the toolbar, shown at left.

[6] This content is available online at <http://cnx.org/content/m12209/1.1/>.

c. Right-click the thermometer and select `Visible Items≫ Digital Display` from the shortcut menu to show the digital display for the thermometer.
d. Place a vertical toggle switch control, located on the `Controls≫Buttons & Switches` palette, on the front panel.
e. Type `Temp Scale` in the label and press the <`Enter`> key or click the `Enter` button.
f. A Use the `Labeling` tool, shown in this media, to place a free label, `deg C`, next to the `True` position of the switch. If you are using automatic tool selection, double-click the blank area of the front panel to begin typing a free label.
g. Place a free label, `deg F` , next to the `False` position of the switch.

2.5.2 User Documentation

1. Document the VI so a description appears in the `Context Help` window when you move the cursor over the VI icon.

   a. Select `File≫VI Properties` to display the `VI Properties` dialog box.
   b. Select `Documentation` from the `Category` pull-down menu.
   c. Type the following description for the VI in the `VI description` text box: `This VI measures temperature using the temperature sensor on the DAQ Signal Accessory.`
   d. Click the `OK` button.

2. Document the thermometer indicator and switch control so descriptions appear in the `Context Help` window when you move the cursor over the objects and tip strips appear on the front panel or block diagram when you move the cursor over the objects while the VI runs.

   - Right-click the thermometer indicator and select `Description and Tip` from the shortcut menu.
   - Type the following description for the thermometer in the `Description` text box: `Displays the temperature measurement.`
   - Type `temperature` in the `Tip` text box.
   - Click the `OK` button.
   - Right-click the switch control and select `Description and Tip` from the shortcut menu.
   - Type the following description for the vertical switch control in the `Description` text box: `Determines the scale (Fahrenheit or Celsius) to use for the temperature measurement.`
   - Type `scale - C or F` in the `Tip` text box.
   - Click the `OK` button.

3. Select `Help≫Show Context Help` to display the `Context Help` window.
4. Move the cursor over the front panel objects and the VI icon to display the descriptions in the `Context Help` window.

2.5.3 Block Diagram

1. Select `Window≫Show Block Diagram` to display the block diagram (Figure 2.9).

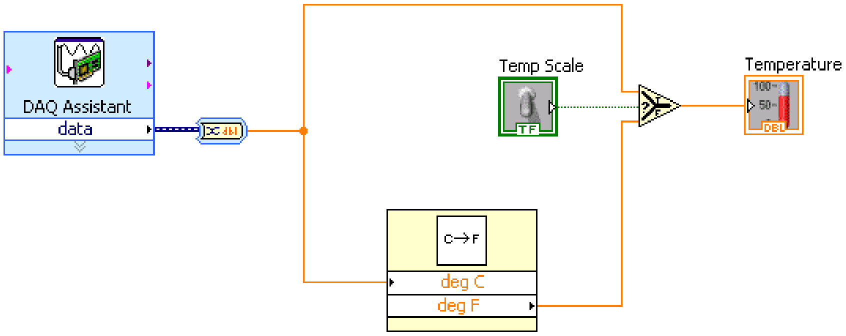


Figure 2.9

2. Place the DAQ Assistant Express VI, located on the Functions≫Input palette, on the block diagram. When you place this Express VI on the block diagram the DAQ Assistant configuration dialog box appears.
   a. Select Analog Input≫Voltage for the type of measurement to make.
   b. Select Dev1≫ai0 (or Dev2≫ai0) for the physical channel and click the Finish button.
   c. You must multiply the temperature by 100 to convert it from voltage to Celsius. On the Settings tab, select Custom Scaling≫Create New. Select a Linear scale. Name the scale Temperature. Enter a slope scale of 100. Click the OK button.
   d. Set min = 0. Set max = 100
   e. Select the Acquire 1 Sample option on the Task Timing tab. Click the OK button.

   NOTE: If you do not have a DAQ device with a temperature sensor connected to your computer, use the (Demo) Read Voltage VI, located in the C:\Exercises LabVIEW Basics I directory.

3. Place the Convert from Dynamic Data Express VI, located on the Functions≫Signal Manipulation palette, on the block diagram. This VI converts the dynamic data type. In the configuration dialog box, select Single scalar in the Resulting data type listbox.

4. Place the Convert C to F VI on the block diagram. Select Functions≫All Functions≫Select a VI, navigate to C:\Exercises\LabVIEW Basics I\Convert C to F.vi. This VI converts the Celsius readings to Fahrenheit.

5. Place the Select function, located on the Functions≫Arithmetic & Comparison≫Express Comparison palette, on the block diagram. This function returns either the Fahrenheit (False) or Celsius (True) temperature value, depending on the value of Temp Scale. Use the Positioning tool to place the icons as shown in the block diagram (Figure 2.9) and use the Wiring tool to wire them together.

   NOTE: To display terminals for a node, right-click the icon and select Visible Items≫Terminals from the shortcut menu.

2.5.4 Front Panel

1. Display the front panel by clicking it or by selecting `Window≫Show Front Panel` .
2. Click the `Run Continuously` button, shown at left, to run the VI continuously.
3. Put your finger on the temperature sensor and notice the temperature increase.
4. Click the `Run Continuously` button again to stop the VI. This allows the VI to finish the current run of the VI.

2.5.5 Icon and Connector Pane

1. Create an icon so you can use the `Thermometer` VI as a subVI. The following icon is an example. If necessary, create a simpler icon to save time.

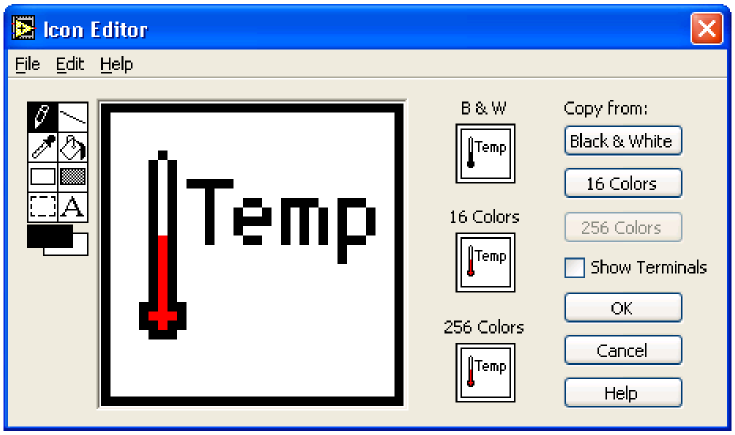


Figure 2.10

a. Right-click the icon in the upper right corner of the front panel and select `Edit` Icon from the shortcut menu. The `Icon Editor` dialog box appears.

b. Double-click the `Select` tool, shown at left, on the left side of the `Icon Editor` dialog box to select the default icon.

c. Press the <`Delete`> key to remove the default icon.

d. Double-click the `Rectangle` tool, shown at this media, to redraw the border.

e. Use the `Pencil` tool, shown in this media, to draw an icon that represents the thermometer.

f. Use the `Foreground and Fill` tools to color the thermometer red.

NOTE: To draw horizontal or vertical straight lines, press the <`Shift`> key while you use the `Pencil` tool to drag the cursor.

g. Double-click the `Text` tool, shown in this media, and change the font to `Small Fonts`.

h. Type `Temp`. Reposition the text if necessary.

i. Select the B & W icon and select 256 Colors in the Copy from section to create a black and white icon, which LabVIEW uses for printing unless you have a color printer.
j. When the icon is complete, click the OK button. The icon appears in the upper right corner of the front panel.

2. Right-click the icon and select Show Connector from the shortcut menu and assign terminals to the switch and the thermometer.
   a. Click the left terminal in the connector pane.
   b. Click the Temp Scale control. The left terminal turns green.
   c. Click the right terminal in the connector pane.
   d. Click the Temperature indicator. The right terminal turns orange.
   e. Click an open space on the front panel.
3. Save the VI as Thermometer.vi in the C:\Exercises\ LabVIEW Basics I directory.You will use this VI later in the course.
4. Close the VI.

2.6 Creating a SubVI from Sections of a VI[7]

You can simplify the block diagram of a VI by converting sections of the block diagram into subVIs. Convert a section of a VI into a subVI by using the Positioning tool to select the section of the block diagram you want to reuse and selecting Edit≫Create SubVI. An icon for the new subVI replaces the selected section of the block diagram. LabVIEW creates controls and indicators for the new subVI and wires the subVI to the existing wires. Figure 2.11 shows how to convert a selection into a subVI.

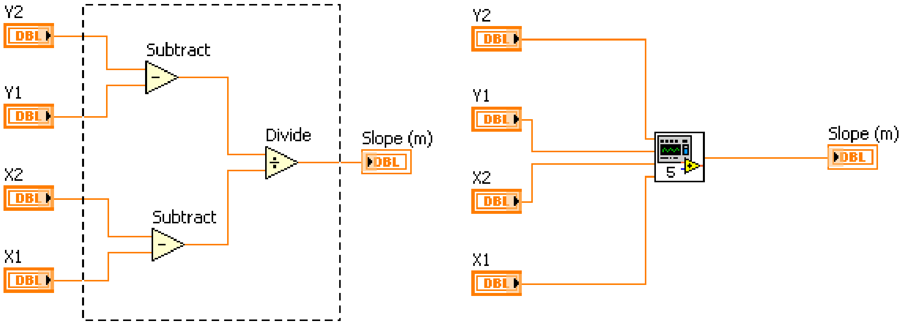


Figure 2.11

The new subVI uses a default pattern for the connector pane and a default icon. Double-click the subVI to edit the connector pane and icon, and to save the subVI.

> NOTE: Do not select more than 28 objects to create a subVI because 28 is the maximum number of connections on a connector pane.

[7] This content is available online at <http://cnx.org/content/m12210/1.1/>.

2.7 Summary, Tips, and Tricks on Modular Programming[8]

- A VI within another VI is called a subVI. Using subVIs helps you manage changes and debug the block diagram quickly.
- After you build a VI front panel and block diagram, build the icon and the connector pane so you can use the VI as a subVI.
- The connector pane is a set of terminals that corresponds to the controls and indicators of that VI. Define connections by assigning a front panel control or indicator to each of the connector pane terminals.
- Create custom icons to replace the default icon by double-clicking the icon in the upper right corner of the front panel to open the `Icon Editor.`
- In the `Icon Editor` dialog box, double-click the Text tool to select a different font.
- You can designate which inputs and outputs are required, recommended, and optional to prevent users from forgetting to wire subVI connections. Right-click a terminal in the connector pane and select `This Connection Is` from the shortcut menu.
- Document a VI by selecting `File≫VI Properties` and selecting `Documentation from the Category` pull-down menu. When you move the cursor over a VI icon, the `Context Help` window displays this description and indicates which terminals are required, recommended, or optional.
- Add descriptions and tip strips to controls and indicators by right-clicking them and selecting `Description and Tip` from the shortcut menu. When you move the cursor over controls and indicators, the `Context Help` window displays this description.
- Convert a section of a VI into a subVI by using the `Positioning` tool to select the section of the block diagram you want to reuse and selecting `Edit≫Create SubVI`.

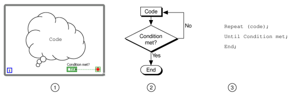
[8] This content is available online at <http://cnx.org/content/m12211/1.1/>.

Chapter 3

Repetition and Loops

3.1 While Loops[1]

A `While Loop` executes a subdiagram until a condition is met. The `While Loop` is similar to a `Do Loop` or a `Repeat-Until Loop` in text-based programming. Figure 3.1 shows a `While Loop` in LabVIEW, a flow chart equivalent of the `While Loop` functionality, and a pseudo code example of the functionality of the `While Loop`.



```
Repeat (code);
Until Condition met;
End;
```

Figure 3.1

The `While Loop` is located on the `Functions`≫`Execution Control` palette. Select the `While Loop` from the palette then use the cursor to drag a selection rectangle around the section of the block diagram you want to repeat. When you release the mouse button, a `While Loop` boundary encloses the section you selected.

Add block diagram objects to the `While Loop` by dragging and dropping them inside the `While Loop`.

NOTE: The `While Loop` always executes at least once.

● The `While Loop` executes the subdiagram until the conditional terminal, an input terminal, receives a specific Boolean value. The default behavior and appearance of the conditional terminal is `Stop If True`, shown in this media. When a conditional terminal is `Stop If True`, the `While Loop` executes its subdiagram until the conditional terminal receives a `True` value.

i The `iteration` terminal, an output terminal, shown in this media, contains the number of completed iterations. The iteration count always starts at zero. During the first iteration, the iteration terminal returns `0`.

In the block diagram in Figure 3.2, the `While Loop` executes until the subVI output is greater than or equal to `10.00` and the `Enable` control is `True`. The `And` function returns `True` only if both inputs are `True`. Otherwise, it returns `False`.

[1] This content is available online at <http://cnx.org/content/m12212/1.2/>.

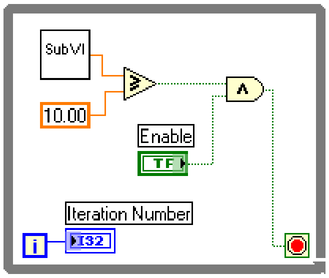


Figure 3.2

In the previous example (Figure 3.2), there is an increased probability of an infinite loop. Generally, the desired behavior is to have one condition met to stop the loop, rather than requiring both conditions to be met.

You can change the behavior and appearance of the conditional terminal by right-clicking the terminal or the border of the `While Loop` and selecting `Continue if True`, shown at left. You also can use the `Operating` tool to click the conditional terminal to change the condition. When a conditional terminal is `Continue if True`, the `While Loop` executes its subdiagram until the conditional terminal receives a `False` value, as shown in Figure 3.3.

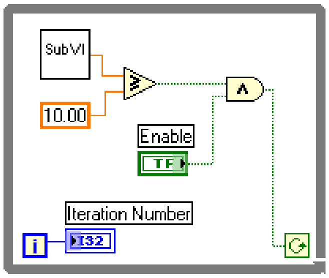


Figure 3.3

The `While Loop` executes until the subVI output is less than `10.00` or the `Enable` control is `False`.

3.1.1 Structure Tunnels

Data can be passed out of or into a `While Loop` through a tunnel. Tunnels feed data into and out of structures. The tunnel appears as a solid block on the border of the `While Loop`. The block is the color of the data type wired to the tunnel. Data pass out of a loop after the loop terminates. When a tunnel passes data into a loop, the loop executes only after data arrive at the tunnel.

In Figure 3.4, the `iteration` terminal is connected to a tunnel. The value in the tunnel does not get passed to the `Iteration Number` indicator until the `While Loop` has finished execution.

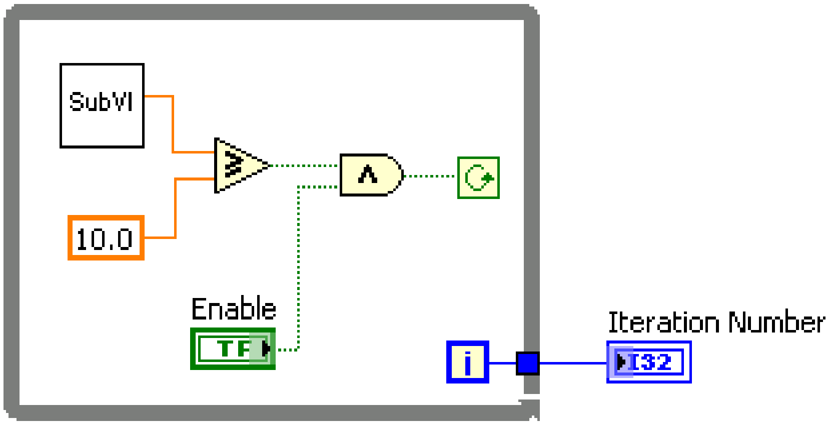
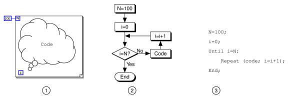


Figure 3.4

Only the last value of the iteration terminal displays in the `Iteration Number` indicator.

3.2 For Loops[2]

A `For Loop` executes a subdiagram a set number of times. Figure 3.5 shows a `For Loop` in LabVIEW, a flow chart equivalent of the `For Loop` functionality, and a pseudo code example of the functionality of the `For Loop`.



```
N=100;
i=0;
Until i=N:
    Repeat (code; i=i+1);
End;
```

Figure 3.5

N The `For Loop` is located on the `Functions`≫`All Functions`≫`Structures` palette. You also can place a `While Loop` on the block diagram, right-click the border of the `While Loop`, and select `Replace with For Loop` from the shortcut menu to change a `While Loop` to a `For Loop`. The value in the `count` terminal (an input terminal), shown in this media, indicates how many times to repeat the subdiagram.

i The `iteration` terminal (an output terminal), shown in this media, contains the number of completed iterations. The iteration count always starts at zero. During the first iteration, the iteration terminal returns `0`.

The `For Loop` differs from the `While Loop` in that the `For Loop` executes a set number of times. A `While Loop` stops executing the subdiagram only if the value at the conditional terminal exists.

The `For Loop` in Figure 3.6 generates a random number every second for 100 seconds and displays the random numbers in a numeric indicator.

[2] This content is available online at <http://cnx.org/content/m12214/1.2/>.

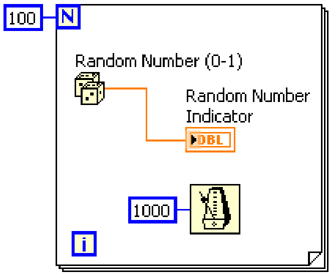


Figure 3.6

3.2.1 Wait Functions

The `Wait Until Next ms Multiple` function, shown in this media, monitors a millisecond counter and waits until the millisecond counter reaches a multiple of the amount you specify. Use this function to synchronize activities. Place this function within a loop to control the loop execution rate.

The `Wait (ms)` function, shown in this media, adds the wait time to the code execution time. This can cause a problem if code execution time is variable.

> NOTE: The `Time Delay` Express VI, located on the `Functions`≫`Execution Control` palette, behaves similar to the `Wait (ms)` function with the addition of built-in error clusters. Refer to Clusters[3] for more information about error clusters.

3.2.2 Wait Functions

LabVIEW can represent numeric data types as signed or unsigned integers (8-bit, 16-bit, or 32-bit), floating-point numeric values (single-, double-, or extended-precision), or complex numeric values (single-, double-, or extended-precision). When you wire two or more numeric inputs of different representations to a function, the function usually returns output in the larger or wider format. The functions coerce the smaller representations to the widest representation before execution, and LabVIEW places a coercion dot on the terminal where the conversion takes place.

For example, the `For Loop` count terminal is a 32-bit signed integer. If you wire a double-precision, floating-point numeric to the count terminal, LabVIEW converts the numeric to a 32-bit signed integer. A coercion dot appears on the count terminal of the first `For Loop`, as shown in Figure 3.7.

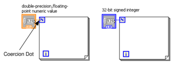


Figure 3.7

[3] "Error Clusters" <http://cnx.org/content/m12231/latest/>

If you wire two different numeric data types to a numeric function that expects the inputs to be the same data type, LabVIEW converts one of the terminals to the same representation as the other terminal. LabVIEW chooses the representation that uses more bits. If the number of bits is the same, LabVIEW chooses unsigned over signed.

In the example in Figure 3.8, a 32-bit signed integer (I32) and a double-precision, floating-point numeric value (DBL) are wired to the `Divide` function. The 32-bit signed integer is coerced since it uses fewer bits than the double-precision, floating-point numeric value.

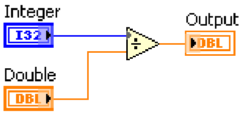


Figure 3.8

To change the representation of a numeric object, right-click the object and select `Representation` from the shortcut menu. Select the data type that best represents the `data.ut` data types.

When LabVIEW converts double-precision, floating-point numeric values to integers, it rounds to the nearest integer. LabVIEW rounds `x.5` to the nearest even integer. For example, LabVIEW rounds 2.5 to 2 and 3.5 to 4.

Refer to the Data Types section of Introduction to LabVIEW, of this manual or to the LabVIEW Help for more information about data types.

3.3 Timed Temperature VI[4]

Exercise 3.3.1
Complete the following steps to build a VI that uses the Thermometer VI to read a temperature once every second for a duration of one minute.

3.3.1 Front Panel

1. Open a blank VI and build the front panel shown in Figure 3.9.

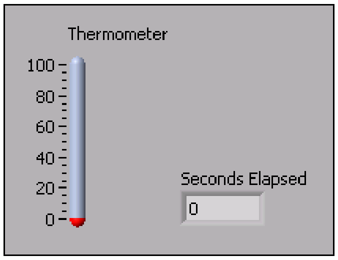


Figure 3.9

[4] This content is available online at <http://cnx.org/content/m12216/1.1/>.

a. Place a `thermometer`, located on the `Controls≫Numeric Indicators` palette, on the front panel. This provides a visual indication of the temperature reading.
b. Place a `numeric indicator`, located on the `Controls≫Numeric Indicators` palette, on the front panel. Label this indicator `Seconds Elapsed`. Right-click the indicator and select `Representation≫I32` from the shortcut menu.

3.3.2 Block Diagram

1. Build the block diagram shown in Figure 3.10.

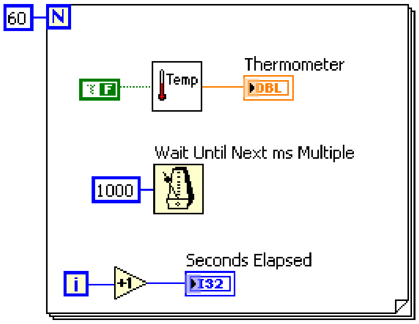


Figure 3.10

- Place a `For Loop`, located on the `Functions≫All Functions≫Structures` palette, on the block diagram. Right-click the `Loop Count` terminal in the upper left corner of the `For Loop` and select `Create Constant` from the shortcut menu. Type `60` in the constant to set the `For Loop` to repeat 60 times.
- Place the `Thermometer` VI on the block diagram. Select `Functions≫All Functions≫Select a VI` and navigate to `C:\Exercises\LabVIEW Basics I\Thermometer.vi`to place the VI. This VI reads the temperature from the DAQ device. Right-click the `Temp Scale` input and select `Create≫Constant` from the shortcut menu. Use a `False` constant for Fahrenheit or a `True` constant for Celsius.

NOTE: If you do not have a DAQ device with a temperature sensor on Channel 0, use the `(Demo) Thermometer` VI instead.

- Place the `Wait Until Next ms Multiple` function, located on the `Functions≫All Functions≫Time & Dialog` palette, on the block diagram. Right-click the input and select `Create≫Constant` from the shortcut menu. Enter a value of `1000` to set the wait to every second.
- Place the `Increment` function, located on the `Functions≫Arithmetic & Comparison≫Express Numeric` palette, on the block diagram. This function adds one to the iteration terminal output.

2. Save this VI as `Timed Temperature.vi` in the `C:\Exercises\LabVIEW Basics I` directory.
3. Run the VI. The first reading might take longer than one second to retrieve if the computer needs to configure the DAQ device.
4. If time permits, complete the following optional and challenge steps, otherwise close the VI.

3.3.3 Optional

1. Build a VI that generates random numbers in a `While Loop` and stops when you click a stop button on the front panel.
2. Save the VI as `General While Loop.vi` in the `C:\Exercises\LabVIEW Basics I` directory.

3.3.4 Challenge

1. Modify the `General While Loop` VI to stop when the stop button is clicked or when the `While Loop` reaches a number of iterations specified by a front panel control.
2. Select `File≫Save As` to save the VI as `Combo While-For Loop.vi` in the `C:\Exercises\LabVIEW Basics I` directory.

3.4 Summary, Tips, and Tricks on Repetition and Loops[5]

- Use structures on the block diagram to repeat blocks of code and to execute code conditionally or in a specific order.
- The `While Loop` executes the subdiagram until the `conditional` terminal receives a specific Boolean value. By default, the `While Loop` executes its subdiagram until the conditional terminal receives a `True` value.
- The `For Loop` executes a subdiagram a set number of times.
- You create loops by using the cursor to drag a selection rectangle around the section of the block diagram you want to repeat or by dragging and dropping block diagram objects inside the loop.
- The `Wait Until Next ms Multiple` function makes sure that each iteration occurs at certain intervals. Use this function to add timing to loops.
- The `Wait (ms)` function waits a set amount of time.
- Coercion dots appear where LabVIEW coerces a numeric representation of one terminal to match the numeric representation of another terminal.
- Use `shift registers` on `For Loops` and `While Loops` to transfer values from one loop iteration to the next.
- Create a `shift register` by right-clicking the left or right border of a loop and selecting `Add Shift Register` from the shortcut menu.
- To configure a `shift register` to carry over values to the next iteration, right-click the left terminal and select `Add Element` from the shortcut menu.
- The `Feedback Node` stores data when the loop completes an iteration, sends that value to the next iteration of the loop, and transfers any data type.
- Use the `Feedback Node` to avoid unnecessarily long wires.

[5] This content is available online at <http://cnx.org/content/m12219/1.1/>.

Chapter 4

Customizing VIs

4.1 Configuring the Appearance of Front Panels[1]

After you build a VI, you can configure the appearance of the front panel so users can more easily operate the VI. For example, you can hide the menu bar and scrollbars to create VIs that look and behave like standard dialog boxes for each platform.

Select `File≫VI Properties` to configure the appearance and behavior of a VI. You also can right-click the VI icon on the front panel or block diagram and select `VI Properties` from the shortcut menu. You cannot access the `VI Properties` dialog box while a VI is running. Refer to the *LabVIEW Basics II: Development Course Manual* for more information about configuring the behavior of VIs.

Use the `Category` pull-down menu at the top of the `VI Properties` dialog box to select from several different option categories, including the following:

General - Displays the current path where a VI is saved, its revision number, revision history, and any changes made since the VI was last saved. You also can use this page to edit the icon or the size of the alignment grid for the VI.

Documentation - Use this page to add a description of the VI and link to a help file topic. Refer to the original Thermometer VI (Section 2.5) for more information about documenting VIs.

Security - Use this page to lock or password-protect a VI.

Window Appearance - Use this page to configure various window settings.

Window Size - Use this page to set the size of the window.

Execution - Use this page to configure how a VI runs. For example, you can configure a VI to run immediately when it opens or to pause when called as a subVI.

Editor Options - Use this page to set the size of the alignment grid for the current VI and to change the style of control or indicator LabVIEW creates when you right-click a terminal and select `Create≫Control` or `Create≫Indicator` from the shortcut menu.

4.1.1 Window Appearance

In the `VI Properties` dialog box, select `Window Appearance` from the `Category` pull-down menu to customize the window appearance for VIs.

These options apply to the VI when it is running. Use these options to change how the user interacts with the application by restricting access to LabVIEW features and by changing the way the window looks and behaves. You can make the VI look and act like a dialog box so the user cannot interact with other windows while the VI window is open. You also can remove the scrollbars and toolbar, and you can set a window to be centered or automatically sized to fit the screen.

[1] This content is available online at <http://cnx.org/content/m12299/1.2/>.

By default, the VI window title is the same as the VI name. You can customize the VI window title to make it more descriptive than the VI filename. This is useful for localized VIs so the VI window title can be translated to the local language. Remove the checkmark from the `Same as VI Name` checkbox to edit `Window title`.

To configure the window appearance, select one of the following window styles. A graphical representation of each style displays on the right when you select the style.

Top-level Application Window - Shows the title bar and menu bar, hides the scrollbars and toolbar, allows the user to close the window, allows run-time shortcut menus, does not allow resizing, and shows the front panel when called.

Dialog - The VI functions as a dialog box in the operating system, so the user cannot interact with other LabVIEW windows while this VI window is open. This option does not prevent you from bringing windows of other applications to the front. (**UNIX**) You cannot make a window stay in front of all other windows. Dialog style windows stay on top, have no menu bar, scrollbars, or toolbar, allow the user to close the window but not resize it, allow run-time shortcut menus, and show the front panel when called. Also, if a Boolean parameter on the front panel is associated with the <`Enter`> or <`Return`> key, LabVIEW highlights the parameter with a dark border.

Default - Same window style used in the LabVIEW development environment.

Custom - Custom window style.

Customize - Displays the Customize Window Appearance dialog box.

4.1.2 Window Size

In the `VI Properties` dialog box, select `Window Size` from the `Category` pull-down menu to customize the window size for VIs. This page includes the following components:

Minimum Panel Size - Sets the minimum size of the front panel. If you allow the user to resize the window on the `Window Appearance` page, the user cannot resize the front panel smaller than the width and height you set on this page.

Size the front panel to the width and height of the entire screen - Automatically resizes the front panel window to fit the screen when you run the VI. The VI does not retain a record of its original size and location, so it stays in the new location if you switch back to edit mode.

Maintain proportions of window for different monitor resolutions - Resizes the VI so it takes up approximately the same amount of screen space when opened on a computer with a different monitor resolution. For example, if you develop a VI on a computer with a monitor resolution of $1,024 \times 768$, you might want to run the VI on a computer with a monitor resolution of 800×600. Use this control in conjunction with scaling one or all the objects on the front panel.

Scale all objects on front panel as the window resizes - Automatically resizes all front panel objects with respect to and in proportion to the size of the front panel window. Text does not resize because the font sizes are fixed. Use this option when you allow the user to resize the front panel window.

4.2 Opening SubVI Front Panels when a VI Runs[2]

A single front panel sometimes is too restrictive to present numerous options or displays. To solve this problem, organize VIs so the top-most VI presents high-level options, and subVIs present related options.

TIP: You also can use tab controls to make the front panel more usable.

[2] This content is available online at <http://cnx.org/content/m12300/1.1/>.

When LabVIEW calls a subVI, ordinarily the subVI runs without opening its front panel. If you want a single instance of the subVI to open its front panel when called, use the `SubVI Node Setup` dialog box. If you want every instance of the subVI to open its front panel when called, use the `VI Properties` dialog box.

4.2.1 Single Instance

If you want a single instance of the subVI to open its front panel when called, right-click the subVI and select `SubVI Node Setup` from the shortcut menu to display the `SubVI Node Setup` dialog box. Place checkmarks in the `Show Front Panel when called` and `Close afterwards if originally closed` checkboxes to open the subVI front panel when called. This dialog box also includes the following components:

Open Front Panel when loaded - Displays the front panel when the subVI loads or when the VI that calls it loads.

Show Front Panel when called - Displays the front panel when the subVI is called.

Close afterwards if originally closed - If `Show Front Panel when called` also contains a checkmark and if the subVI was previously closed, the front panel closes after the subVI runs.

Suspend when called - Suspends a subVI when called and waits for user interaction. This option is the same as selecting `Operate≫Suspend when called`.

4.2.2 Every Instance

If you want every instance of the subVI to open its front panel when called, open the subVI and select `File≫VI Properties`. Select `Window Appearance` from the `Category` pull-down menu, click the `Customize` button, and place checkmarks in the `Show Front Panel When Called` and `Close Afterwards if Originally Closed` checkboxes.

4.3 Pop-up Graph VI and Use Pop-up Graph VI[3]

Exercise 4.3.1

Complete the following steps to build a VI that acquires temperature once every 0.5 seconds for 10 seconds, displays a subVI front panel that shows the acquired data in a graph, and keeps the front panel open until you click a button.

4.3.1 Front Panel

Open a blank VI and build the following front panel.

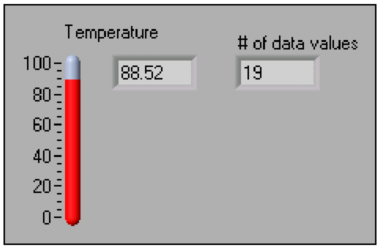


Figure 4.1

[3] This content is available online at <http://cnx.org/content/m12301/1.2/>.

Use the following guidelines to assist you in building the front panel.

- The indicator to the right of the thermometer is a digital display belonging to the thermometer. Right-click the thermometer and select `Visible Items≫Digital Display` from the shortcut menu to display the digital value.
- Change `# of data values` to signed 32-bit integer (I32) representation.

4.3.2 Block Diagram

1. Build the following block diagram.

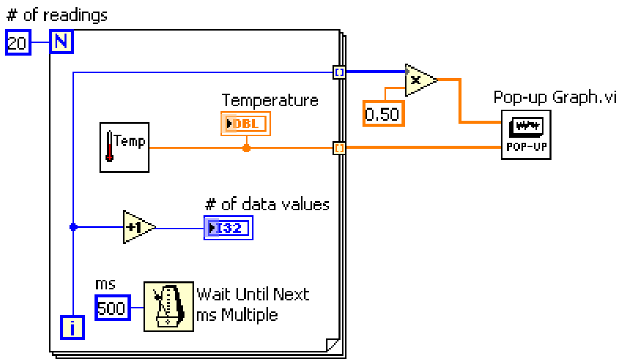
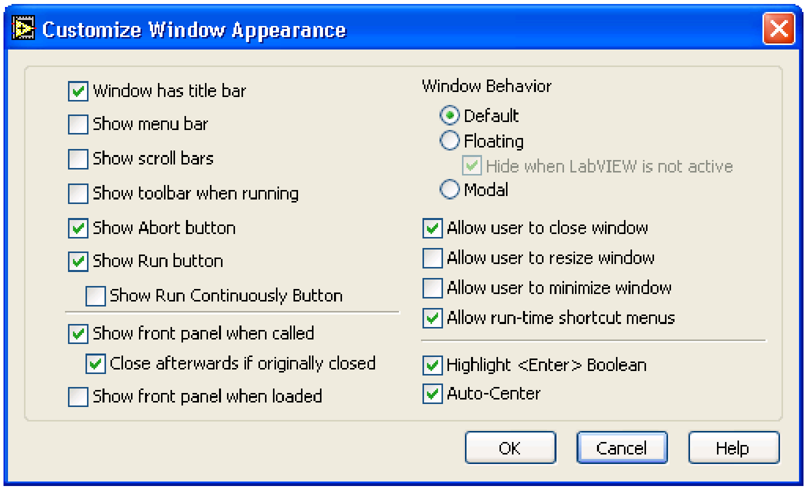


Figure 4.2

   a. Place the `Thermometer` VI from a previous exercise (Section 2.5) on the block diagram. This VI acquires the current temperature value.
   b. Place the `Wait Until Next ms Multiple` function, located on the `Functions≫All Functions≫Time & Dialog` palette, on the block diagram. Right-click the input, select `Create≫Constant`, and type `500` in the constant to cause the `For Loop` to execute every 500 ms.
   c. Place the `Multiply` function, located on the `Functions≫Arithmetic & Comparison≫Express Numeric` palette, on the block diagram. This function multiplies each element of the output array by 0.50 to scale the x values to represent the time interval at which the VI takes the measurements.
   d. Place the `Pop-up Graph` VI, located in the `C:\Exercises\LabVIEW Basics I` directory, on the block diagram. This VI plots the temperature data on an XY graph.
   e. Complete the block diagram as shown in the previous figure.

2. Save the VI as `Use Pop-up Graph.vi` in the `C:\Exercises\LabVIEW Basics I` directory.
3. Configure the subVI to display its front panel when called.
   a. Double-click the `Pop-up Graph` subVI to open its front panel.

b. Select File≫VI Properties.
c. Select Window Appearance from the Category pull-down menu.
d. Click the Customize button. Configure the window appearance as shown in the dialog box (Figure 4.3).



☑ Window has title bar
□ Show menu bar
□ Show scroll bars
□ Show toolbar when running
☑ Show Abort button
☑ Show Run button
□ Show Run Continuously Button
☑ Show front panel when called
☑ Close afterwards if originally closed
□ Show front panel when loaded

Window Behavior
● Default
○ Floating
☑ Hide when LabVIEW is not active
○ Modal
☑ Allow user to close window
□ Allow user to resize window
□ Allow user to minimize window
☑ Allow run-time shortcut menus
☑ Highlight <Enter> Boolean
☑ Auto-Center

Figure 4.3

e. Click the OK button twice and save and close the subVI. If the front panel is not closed, it will not close after the subVI runs.

4. Run the Use Pop-up Graph VI. After the VI acquires 10 seconds of temperature data, the front panel of the Pop-up Graph VI displays and plots the temperature data. Click the DONE button to return to the calling VI.
5. Change the window appearance settings for the Pop-up Graph subVI to the Dialog window style.
6. Save and close the subVI.
7. Run the Use Pop-up Graph VI again. The Pop-up Graph subVI front panel window behaves as a dialog box. For example, the window stays on top of all other windows and uses the system colors.
8. Close all open VIs.

4.4 Keyboard Shortcuts for Controls[4]

While a VI runs, you can press the <Tab> key to change the key focus from one control to the next. The key focus is the same as if you had clicked the control. While a control has the key focus, you can use the keyboard to enter the control value. If the control is a text or numeric control, LabVIEW highlights the text so you can edit it. If the control is Boolean, press the spacebar or the <Enter> key to change its value.

You also can assign keyboard shortcuts to controls so users can navigate the front panel by pressing other keys. Right-click the control and select Advanced≫Key Navigation from the shortcut menu to display the Key Navigation dialog box.

[4] This content is available online at <http://cnx.org/content/m12302/1.1/>.

NOTE: The Advanced≫Key Navigation shortcut menu item is dimmed for indicators because you cannot enter data in an indicator.

Select the shortcut key you want to assign to the control in the Key Assignment section. The front panel control names that appear in the Current Assignments listbox correspond to the owned labels of those controls.

To prevent users from accessing a control by pressing the <Tab> key while the VI runs, place a checkmark in the Skip this control when tabbing checkbox.

4.5 Temperature System VI[5]

Exercise 4.5.1

4.5.1 Front Panel

1. Open the Temperature System VI located in the C:\Exercises\ LabVIEW Basics I directory. The following front panel is already built.

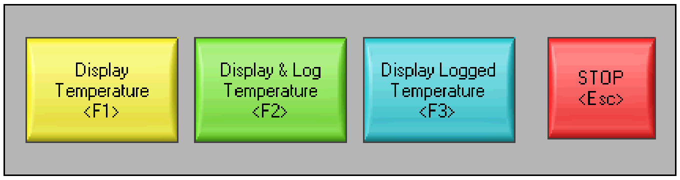

Display Temperature <F1>
Display & Log Temperature <F2>
Display Logged Temperature <F3>
STOP <Esc>

Figure 4.4

   The front panel contains four Boolean buttons (Figure 4.4). The mechanical action of the first three buttons is Latch When Pressed. This setting changes the control value when you click it and retains the new value until the VI reads it once. At this point the control reverts to its default value, even if you keep pressing the mouse button. This action is similar to a circuit breaker and is useful for stopping While Loops or for getting the VI to perform an action only once each time you set the control. The mechanical action of the Stop button is Latch When Released. This setting changes the control value only after you release the mouse button within the graphical boundary of the control. When the VI reads it once, the control reverts to the old value. This action guarantees at least one new value. This action is similar to dialog box buttons and system buttons.
2. Right-click a control and select Advanced≫Key Navigation from the shortcut menu to display the Key Navigation dialog box.
3. In the Key Assignment section, assign the shortcut key shown in the Figure 4.4.
4. Repeat list, item 2, p. 80 and list, item 3, p. 80 for each control.

4.5.2 Block Diagram

1. Examine the block diagram (Figure 4.5) which is already built.

[5] This content is available online at <http://cnx.org/content/m12303/1.1/>.

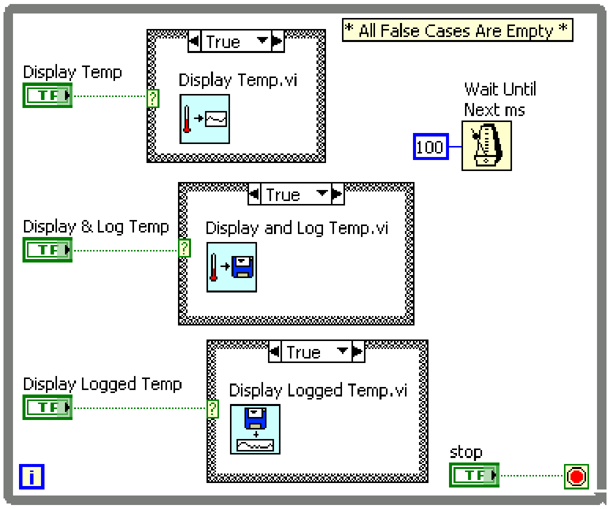


Figure 4.5

- The Display Temp VI simulates a temperature measurement every 500 ms and plots it on a strip chart.
- The Display and Log Temp VI simulates a temperature measurement every 500 ms, plots it on a strip chart, and logs it to a file.
- The Display Logged Temp VI opens a file that you select, reads the logged data, and displays them on a graph.

2. Configure each subVI to display its front panel when called.
   a. Right-click the subVI and select SubVI Node Setup from the shortcut menu.
   b. Place checkmarks in the Show Front Panel when called and Close afterwards if originally closed checkboxes.
   c. Click OK to close the SubVI Node Setup dialog box.
   d. Repeat list, item 1, p. 81 through list, item 3, p. 81 for the remaining two subVIs.
3. Save the VI. Display the front panel and run the VI.
4. Click each button and press the corresponding keyboard shortcuts. The three subVIs return to the Temperature System VI front panel when you press the <Enter> key. Try pressing the <Enter> key to do so.
5. Stop the VI.
6. Configure the Temperature System VI to run automatically when you open the VI.
   a. Select File≫VI Properties.
   b. Select Execution from the Category pull-down menu.
   c. Place a checkmark in the Run When Opened checkbox.
7. Configure the VI so the menu bar and toolbar are not visible while the VI runs.
   a. Select Window Appearance from the Category pull-down menu.
   b. Click the Customize button.

   c. Remove the checkmarks from the `Show Menu Bar` and `Show Toolbar When Running` checkboxes.
   d. Click the `OK` button twice.

8. Save and close all VIs.
9. Open the `Temperature System` VI again. The VI runs automatically when you open it. Click the buttons on the front panel or use the keyboard shortcuts.
10. Stop and close all VIs.

4.6 Editing VI Properties[6]

Sometimes you can select VI properties that make it difficult to edit a VI. For example, you might select the `Run When Opened` option and disable the menu bar and toolbar. If you set the VI to close and exit LabVIEW after it runs, you cannot stop the VI and edit it without it closing and exiting LabVIEW. This VI would be very difficult to edit.

> NOTE: To exit LabVIEW, you can use the Quit LabVIEW function located on the `Functions≫All Functions≫Application Control` palette. This function aborts all running VIs and ends the current session of LabVIEW. The function has one input. If it is wired, the end of the LabVIEW session occurs only if that input is True. If the input is not wired, the end of the session occurs when the node executes.

Before you change VI properties, save a backup of the VI to a new location by selecting `File≫Save with Options` to avoid situations like the previous examples.

Select the `Development Distribution` option to save the VI to a new location along with its entire hierarchy. You also can include the `vi.lib` files in the save. After you save the backup VI, change the VI properties of the original VI. If you encounter a problem, you can return to the backup VI.

> NOTE: If you select the `Remove diagrams` option, you remove the source code of the VI. Select this option only if you never need to edit the VI again. Before you save a VI without the block diagrams, save a backup of the VI with the block diagrams.

If you already saved a development VI with properties that make the VI difficult to edit, refer to the Edit Me VI (Section 4.7) exercise for more information about editing the VI.

4.7 Edit Me VI[7]

Exercise 4.7.1

4.7.1 Front Panel

1. Close any open VIs and open the `Edit Me` VI, located in the `C:\Exercises\LabVIEW Basics I` directory. The following front panel is already built.

[6] This content is available online at <http://cnx.org/content/m12304/1.2/>.

[7] This content is available online at <http://cnx.org/content/m12305/1.2/>.

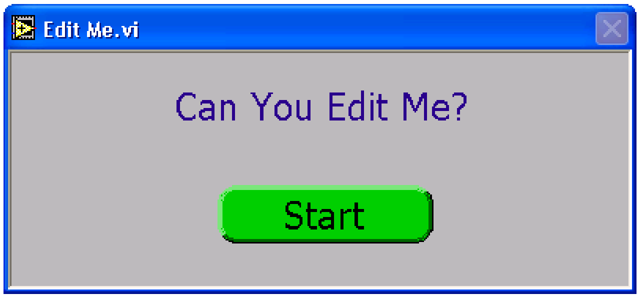


Figure 4.6

The VI is already running when it opens. While the VI runs, you cannot use the menu bar, toolbar, or keyboard shortcuts to edit or abort the VI.

2. Click the `Start` button. After 10 seconds, the VI stops running and quits LabVIEW.
3. Relaunch LabVIEW and open a blank VI.
4. If the VI you want to edit either does not have subVIs or you do not know what it contains, complete list, item 5, p. 83 through list, item 13, p. 83. However, if the VI you want to edit has subVIs, open one of the subVIs and modify the block diagram to break the subVI. For example, place an `Add` function on the block diagram and do not wire the inputs. Open the VI you want to edit. Because its subVI is nonexecutable, the VI that calls it is also nonexecutable. It opens in edit mode and the `Run` button appears broken. Make sure to fix the subVI after you edit the calling VI.
5. Display the block diagram of the new VI.
6. Place the `Edit Me` VI, which is already built, on the block diagram. The front panel for the `Edit Me` VI displays. Although you can display the block diagram of the `Edit Me` VI, you cannot edit it.
7. Select `Operate≫Change to Edit Mode`. A dialog box informs you that the VI is locked.
8. Click the `Unlock` button. You now can edit the VI. You also can unlock a VI by selecting `File≫VI Properties` and selecting `Security` from the `Category` pull-down menu.
9. Select and delete the `Quit LabVIEW` function from the block diagram.
10. Save and close the `Edit Me` VI. Close the new VI and do not save changes.
11. Open the `Edit Me` VI again.
12. After the VI runs, try to edit it.
13. Close the `Edit Me` VI.

4.8 Customizing the Controls and Functions Palettes[8]

You can customize the `Controls` and `Functions` palettes to add VIs and controls to the palettes, hide VIs and functions, or rearrange the built-in palettes to make the VIs and functions you use frequently more accessible.

4.8.1 Adding VIs and Controls to the User Library and the Instrument Library

The simplest method for adding VIs and controls to the `Controls` and `Functions` palettes is to save them in the `labview\user.lib` directory. When you restart LabVIEW, the `Functions≫Express User Libraries`

[8] This content is available online at <http://cnx.org/content/m12306/1.1/>.

and Controls≫Express User Controls palettes contain subpalettes for each directory, VI library (.llb), or menu (.mnu) file in the labview\user.lib directory, and icons for each file in the labview\user.lib directory. After you add files to or remove files from specific directories, LabVIEW automatically updates the palettes when you restart LabVIEW.

The Functions≫All Functions≫Instrument I/O palette corresponds to the labview\instr.lib directory. Save instrument drivers in this directory to add them to the Functions palette.

When you add VIs or controls to the Controls and Functions palettes using this method, you cannot set the name of each subpalette or the exact location of the VIs or controls on the palettes.

4.8.2 Creating and Editing Custom Palette Views

To control the name of each subpalette and the exact location of the VIs and controls you add to the Controls and Functions palettes, you must create a custom palette view. LabVIEW includes two built-in palette views–Express and Advanced. Select Tools≫Advanced≫Edit Palette Views to create or edit custom palette views.

NOTE: You cannot edit a built-in palette view.

Refer to the *LabVIEW User Manual* and the *LabVIEW Help* for more information about palette views.

4.9 Summary, Tips, and Tricks on Customizing VIs[9]

- Select `File≫VI Properties` to configure the appearance and behavior of a VI. You also can right-click the VI icon on the front panel or block diagram and select `VI Properties` from the shortcut menu.
- If you want a single instance of the subVI to open its front panel when called, right-click the subVI and select `SubVI Node Setup` from the shortcut menu. Place checkmarks in the `Show Front Panel when called` and `Close afterwards if originally closed` checkboxes.
- If you want every instance of the subVI to open its front panel when called, select `File≫VI Properties` and select `Window Appearance` from the `Category` pull-down menu. Click the `Customize` button and place checkmarks in the `Show Front Panel When Called` and `Close Afterwards if Originally Closed` checkboxes.
- Assign keyboard shortcuts to controls by right-clicking the control and selecting `Advanced≫Key Navigation` from the shortcut menu.
- Before you change VI properties, save a backup of the VI to a new location by selecting `File≫Save with Options` to avoid making the VI difficult to edit.
- To edit a VI with properties that make the VI difficult to edit:
    - Break one of its subVIs. The VI opens in edit mode because it cannot run with a broken subVI.
    - If the VI has no subVIs, place it on the block diagram of a new VI.
- The simplest method for adding VIs and controls to the `Controls` and `Functions` palettes is to save them in the `user.lib` directory.
- To create or edit a custom palette view, select `Tools≫Advanced≫Edit Palette Views`.
- Change to an icon- or text-only palette view by selecting from the `Format` pull-down menu.

[9] This content is available online at <http://cnx.org/content/m12307/1.1/>.

Chapter 5

Plotting Data

5.1 Waveform Charts[1]

The `waveform chart` is a numeric indicator that displays one or more plots. The `waveform chart` is located on the `Controls`≫`Graph Indicators` palette. `Waveform charts` can display single or multiple plots. Figure 5.1 shows the elements of a multiplot waveform chart. Two plots are displayed: `Raw Data` and `Running Avg`.

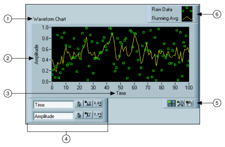


Figure 5.1

Charts use three different modes to scroll data, as shown in the Figure 5.2. Right-click the chart and select `Advanced`≫`Update Mode` from the shortcut menu. Select `Strip Chart`, `Scope Chart`, or `Sweep Chart` . The default mode is `Strip Chart`.

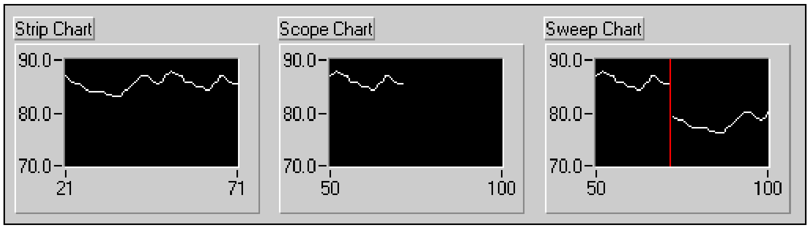


Figure 5.2

A `strip chart` shows running data continuously scrolling from left to right across the chart. A `scope chart` shows one item of data, such as a pulse or wave, scrolling partway across the chart from left to the

[1] This content is available online at <http://cnx.org/content/m12233/1.2/>.

right. A sweep chart is similar to an EKG display. A sweep chart works similarly to a scope except it shows the older data on the right and the newer data on the left separated by a vertical line. The scope chart and sweep chart have retracing displays similar to an oscilloscope. Because there is less overhead in retracing a plot, the scope chart and the sweep chart display plots significantly faster than the strip chart.

5.1.1 Wiring Charts

You can wire a scalar output directly to a waveform chart. The data type in the waveform chart in Figure 5.3 terminal matches the input data type.

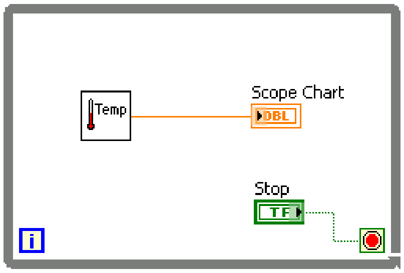


Figure 5.3

Waveform charts can display multiple plots. Bundle multiple plots together using the Bundle function located on the Cluster palette. In Figure 5.4, the Bundle function bundles the outputs of the three VIs to plot on the waveform chart.

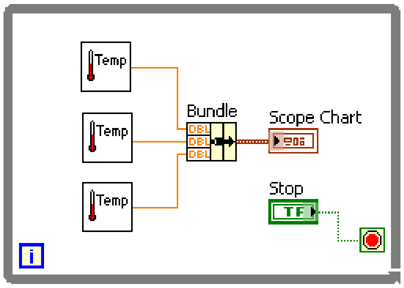


Figure 5.4

The waveform chart terminal changes to match the output of the Bundle function. To add more plots, use the Positioning tool to resize the Bundle function.

5.2 Temperature Monitor VI[2]

Exercise 5.2.1

Complete the following steps to build a VI that measures temperature and displays it on a waveform chart.

5.2.1 Front Panel

1. Open a blank VI and build the front panel shown in Figure 5.5.

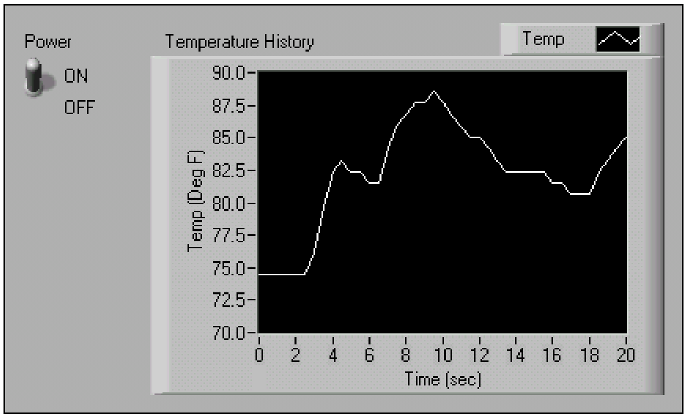


Figure 5.5

a. Place the vertical toggle switch, located on the `Controls≫Buttons & Switches` palette, on the front panel. Label this switch `Power`. You use the switch to stop the acquisition.
b. Place a `waveform chart`, located on the `Controls≫Graph Indicators` palette, on the front panel. Label the chart `Temperature History`. The `waveform chart` displays the temperature in real time.
c. The `waveform chart legend` labels the plot `Plot 0`. Use the `Labeling` tool to triple-click `Plot 0` in the chart legend, and change the label to `Temp`.
d. The temperature sensor measures room temperature. Use the `Labeling` tool to double-click 10.0 in the y-axis and type 90 to rescale the chart. Leave the x-axis in its default state.
e. Change −10.0 in the y-axis to 70.
f. Label the y-axis `Temp (Deg F)` and the x-axis `Time (sec)`.

5.2.2 Block Diagram

1. Select `Window≫Show Block Diagram` to display the block diagram.
2. Enclose the two terminals in a While Loop, as shown in the block diagram (Figure 5.6).

[2] This content is available online at <http://cnx.org/content/m12234/1.2/>.

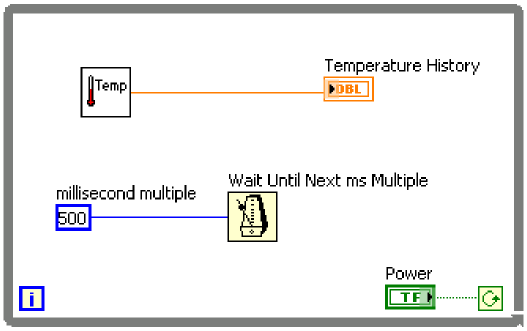


Figure 5.6

3. Right-click the `conditional` terminal and select `Continue if True`.
4. Wire the objects as shown in Figure 5.6.

   a. Temp Place the `Thermometer` VI on the block diagram. Select `Functions≫All Functions≫Select a VI` and navigate to `C:\Exercises\LabVIEW Basics I\Thermometer.vi`. This subVI returns one temperature measurement from the temperature sensor.

   NOTE: Use the `(Demo) Thermometer` VI if you do not have a DAQ device available.

   b. Place the `Wait Until Next ms Multiple` function, located on the `Functions≫All Functions≫Time & Dialog` palette, on the block diagram.

   c. 500 Right-click the `millisecond multiple` input of the `Wait Until Next ms Multiple` function, select `Create≫Constant` from the shortcut menu, type 500, and press the <`Enter`> key. The numeric constant specifies a wait of 500*ms* so the loop executes once every half-second.

   NOTE: To measure temperature in Celsius, wire a Boolean `True` constant located on the `Functions≫Arithmetic & Comparison≫Express Boolean` palette to the `Temp Scale` input of the `Thermometer` VI. Change the scales on charts and graphs in subsequent exercises to a range of 20 to 32 instead of 70 to 90.

5. Save the VI as `Temperature Monitor.vi` in the `C:\Exercises\LabVIEW Basics I` directory.

5.2.3 Run the VI

1. Display the front panel by clicking it or by selecting `Window≫Show Front Panel`.
2. Use the `Operating` tool to click the vertical toggle switch and turn it to the `ON` position.
3. Run the VI. The subdiagram within the `While Loop` border executes until the specified condition is `True`. For example, while the switch is on (`True`), the `Thermometer` VI takes and returns a new measurement and displays it on the `waveform chart`.
4. Click the vertical toggle switch to stop the acquisition. The condition is `False`, and the loop stops executing.

5.2.4 Front Panel

1. Format and customize the x- and y-scales of the waveform chart.
   a. Right-click the chart and select `Properties` from the shortcut menu to display the `Chart Properties` dialog box.
   b. Click the `Format and Precision` tab. Select `Deg F (Y-axis)` in the top pull-down menu. Set the `Digits of precision` to 1.
   c. Click the `Plots` tab and select different styles for the y-axis, as shown in Figure 5.7.

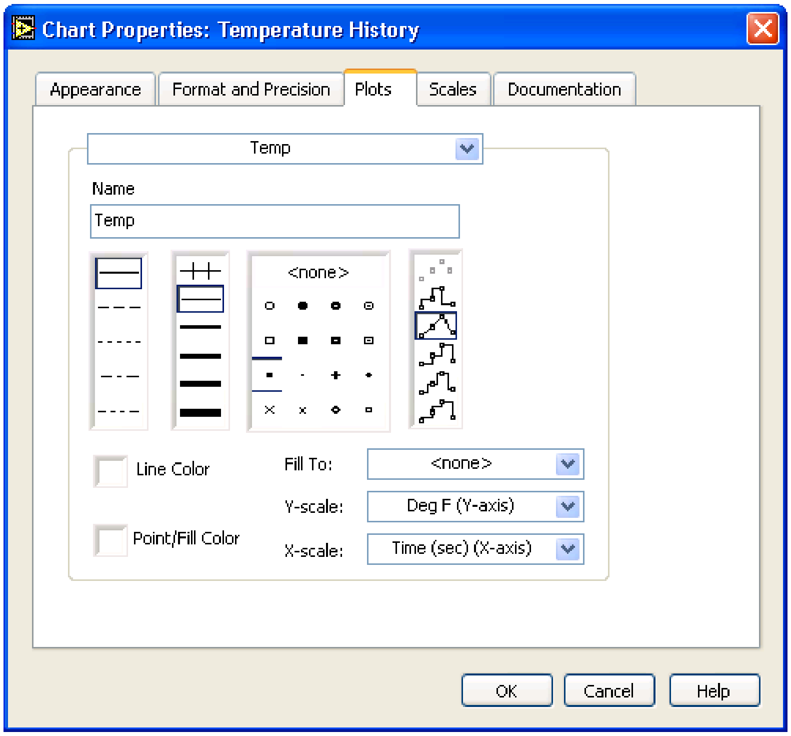


Chart Properties: Temperature History

Temp

Name: Temp

Fill To: <none>

Y-scale: Deg F (Y-axis)

X-scale: Time (sec) (X-axis)

Figure 5.7

   d. Select the `Scales` tab and select the `Time (sec) (X-axis)` in the top pull-down menu. Set the scale options as shown in Figure 5.8. Set the `Multiplier` to 0.50 to account for the `500 ms Wait` function.

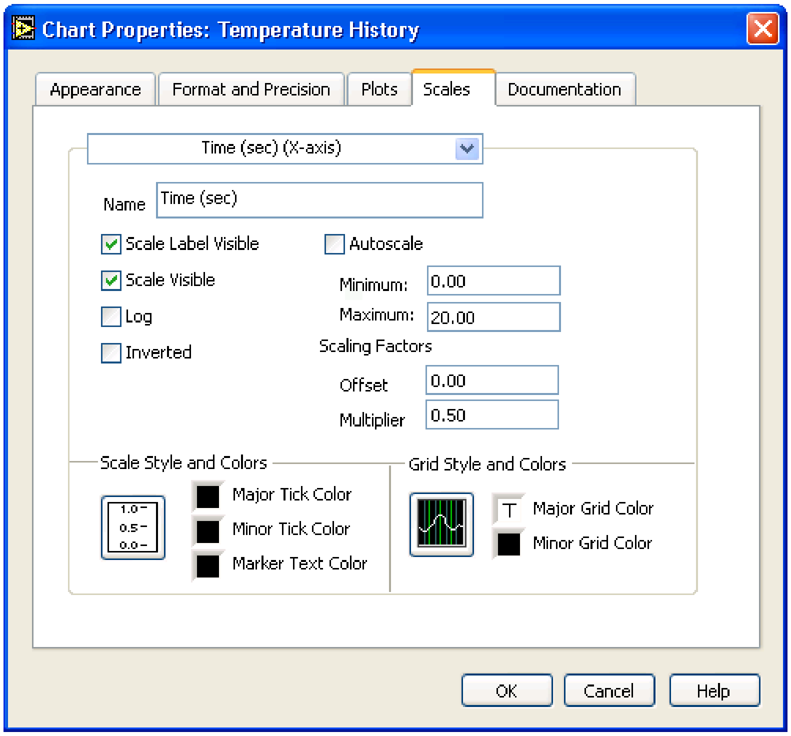


Chart Properties: Temperature History

Time (sec) (X-axis)

Name: Time (sec)

☑ Scale Label Visible

☑ Scale Visible

□ Log

□ Inverted

□ Autoscale

Minimum: 0.00

Maximum: 20.00

Scaling Factors

Offset: 0.00

Multiplier: 0.50

Figure 5.8

e. In the `Scales` tab, select the `Deg F (Y-axis)` in the top pull-down menu. Set the scale options as shown in Figure 5.9.

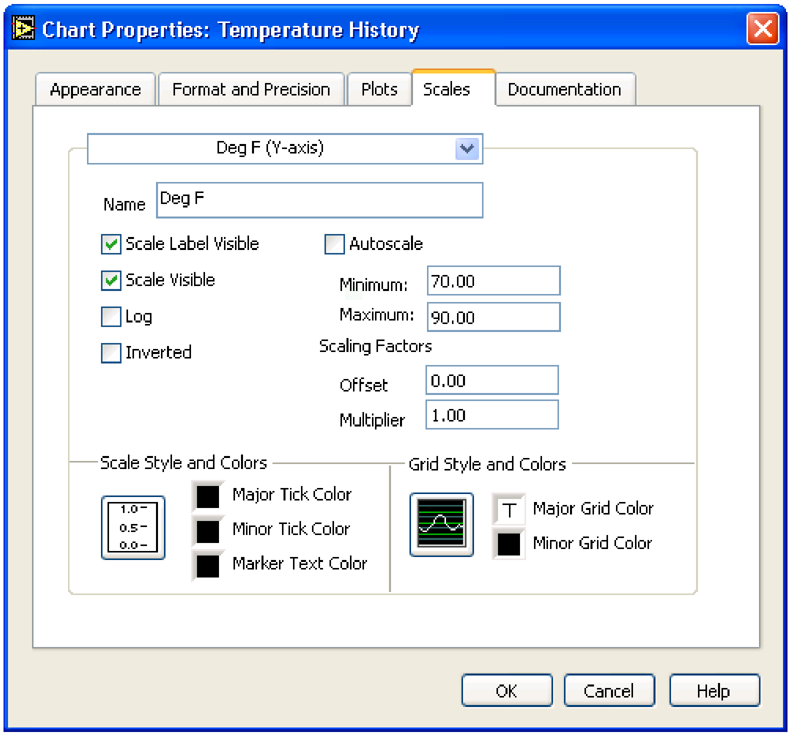


Chart Properties: Temperature History

Appearance | Format and Precision | Plots | Scales | Documentation

Deg F (Y-axis)

Name: Deg F

☑ Scale Label Visible

☑ Scale Visible

□ Log

□ Inverted

□ Autoscale

Minimum: 70.00

Maximum: 90.00

Scaling Factors

Offset: 0.00

Multiplier: 1.00

Scale Style and Colors

Major Tick Color

Minor Tick Color

Marker Text Color

Grid Style and Colors

Major Grid Color

Minor Grid Color

Figure 5.9

   f. Click the OK button to close the dialog box when finished.

2. Right-click the waveform chart and select Data Operations≫Clear Chart from the shortcut menu to clear the display buffer and reset the waveform chart.

   TIP: When a VI is running, you can select Clear Chart from the shortcut menu.

3. Each time you run the VI, you first must turn on the vertical toggle switch and then click the Run button due to the current mechanical action of the switch. Modify the mechanical action of the vertical toggle switch so temperature is plotted on the graph each time you run the VI, without having to first set the toggle switch.

   a. Stop the VI if it is running.
   b. Use the Operating tool to click the vertical toggle switch and turn it to the ON position.
   c. Right-click the switch and select Data Operations≫Make Current Value Default from the shortcut menu. This sets the ON position as the default value.
   d. Right-click the switch and select Mechanical Action≫Latch When Pressed from the shortcut menu. This setting changes the control value when you click it and retains the new value until the VI reads it once. At this point the control reverts to its default value, even if you keep pressing the mouse button. This action is similar to a circuit breaker and is useful for stopping While Loops or for getting the VI to perform an action only once each time you set the control.

5.2.5 Run the VI

1. Run the VI.

2. Use the `Operating` tool to click the vertical switch to stop the acquisition. The switch changes to the `OFF` position and changes back to `ON` after the `conditional` terminal reads the value.
3. Save the VI. You will use this VI in the Temperature Running Average (Section 5.3) VI.

5.3 Temperature Running Average VI[3]

Exercise 5.3.1
Complete the following steps to modify the `Temperature Monitor` VI to average the last three temperature measurements and display the average on a `waveform chart`.

5.3.1 Front Panel

1. Open the Temperature Monitor (Section 5.2) VI.
2. Select `File≫Save As` and rename the VI `Temperature Running Average.vi` in the `C:\Exercises\LabVIEW Basics I` directory.

5.3.2 Block Diagram

1. Display the block diagram.
2. Right-click the right or left border of the `While Loop` and select `Add Shift Register` from the shortcut menu to create a `shift register`.
3. Right-click the left terminal of the `shift register` and select `Add Element` from the shortcut menu to add an element to the `shift register`.
4. Modify the block diagram as in Figure 5.10.

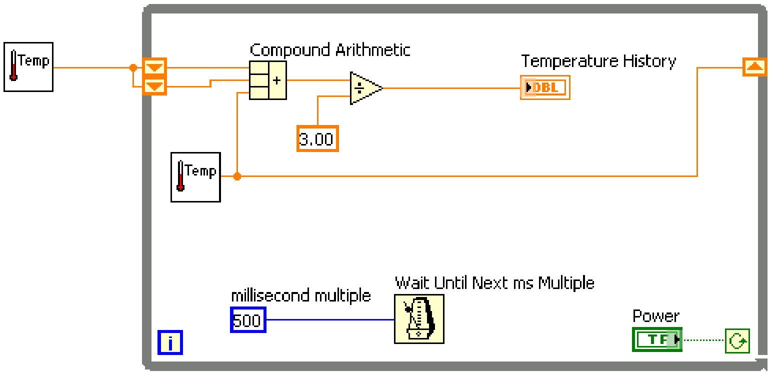


Figure 5.10

a. Temp Press the <`Ctrl`> key while you click the `Thermometer` VI and drag it outside the `While Loop` to create a copy of the subVI. The `Thermometer` VI returns one tempera-

[3] This content is available online at <http://cnx.org/content/m12235/1.2/>.

ture measurement from the temperature sensor and initializes the left `shift registers` before the loop starts.

b. Place the `Compound Arithmetic` function, located on the `Functions≫Arithmetic & Comparison≫Express Numeric` palette, on the block diagram. This function returns the sum of the current temperature and the two previous temperature readings. Use the `Positioning` tool to resize the function to have three left terminals.

c. Place the `Divide` function, located on the `Functions≫Arithmetic & Comparison≫Express Numeric` palette, on the block diagram. This function returns the average of the last three temperature readings.

d. 3.00 Right-click the `y` terminal of the `Divide` function, select `Create≫Constant`, type 3, and press the <`Enter`> key.

5.3.3 Run the VI

1. Run the VI. During each iteration of the `While Loop`, the `Thermometer` VI takes one temperature measurement. The VI adds this value to the last two measurements stored in the left terminals of the `shift register`. The VI divides the result by three to find the average of the three measurements, the current measurement plus the previous two. The VI displays the average on the `waveform chart`. Notice that the VI initializes the `shift register` with a temperature measurement.

5.3.4 Block Diagram

1. Modify the block diagram as shown in Figure 5.11.

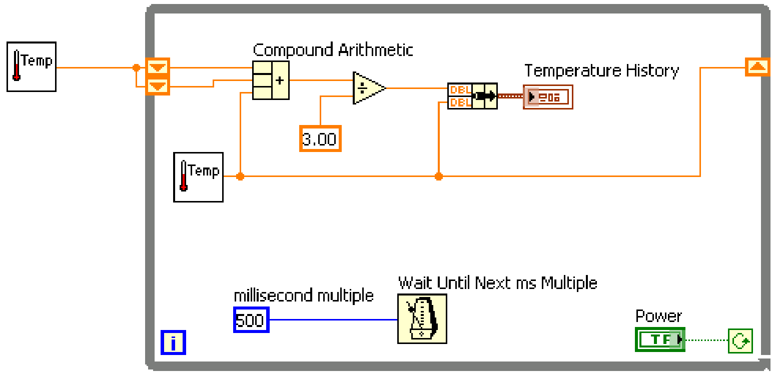


Figure 5.11

a. Place the `Bundle` function, located on the `Functions≫All Functions≫Cluster` palette, on the block diagram. This function bundles the average and current temperature for plotting on the `waveform chart`.

2. Save the VI. You will use this VI later in the course.

5.3.5 Run the VI

1. Run the VI. The VI displays two plots on the `waveform chart`. The plots are overlaid. That is, they share the same vertical scale.
2. If time permits, complete the optional steps. Otherwise, close the VI.

5.3.6 Optional

Customize the waveform chart as shown in Figure 5.12. You can display a plot legend, a scale legend, a graph palette, a digital display, and a scrollbar. By default, a `waveform chart` displays the plot legend.

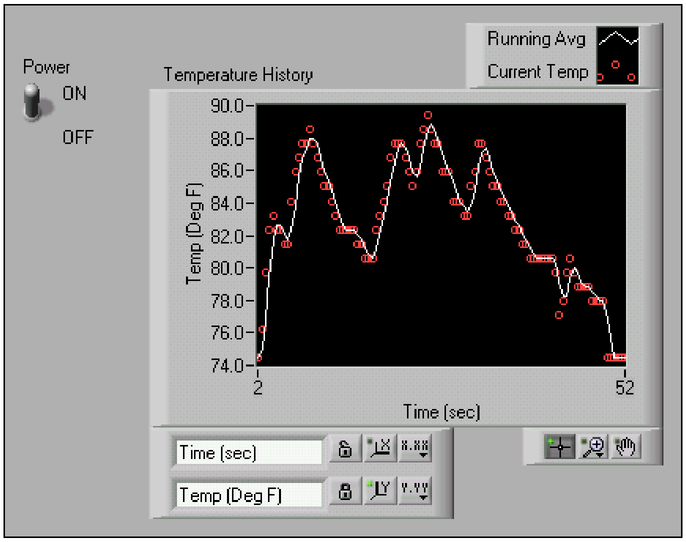


Figure 5.12

1. Customize the y-axis.

   a. A Use the `Labeling` tool to double-click 70.0 in the y-axis, type 75.0, and press the <`Enter`> key.

   b. **b** - Use the `Labeling` tool to double-click the second number from the bottom on the y-axis, type 80.0, and press the <`Enter`> key. This number determines the numerical spacing of the y-axis divisions. For example, if the number above 75.0 is 77.5, it indicates a y-axis division of 2.5, changing the 77.5 to 80.0 reformats the y-axis to multiples of 5.0 (75.0, 80.0, 85.0, and so on).

NOTE: The `waveform chart` size has a direct effect on the display of axis scales. Increase the `waveform chart` size if you encounter problems while customizing the axis.

2. Right-click the waveform chart and select Visible Items≫Scale Legend from the shortcut menu to display the scale legend, as shown in Figure 5.13. You can place the scale legend anywhere on the front panel.

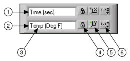


Figure 5.13: 1. X-axis, 2. Y-axis, 3. Scale Labels, 4. Scale Lock Button, 5. Autoscale Button, 6. Scale Format Button

3. Use the scale legend to customize each axis.
   a. Make sure the Lock Autoscale button appears locked and the Autoscale LED is green so the y-axis adjusts the minimum and maximum values to fit the data in the chart.
   b. Click the Scale Format button to change the format, precision, mapping mode, scale visibility, and grid options for each axis.
4. Use the plot legend to customize the plots.
   a. Use the Positioning tool to resize the plot legend to include two plots.
   b. Use the Labeling tool to change Temp to Running Avg and to change Plot 1 to Current Temp. If the text does not fit, use the Positioning tool to resize the plot legend.
   c. Right-click the plot in the plot legend to set the line and point styles and the color of the plot background or traces.
5. Right-click the waveform chart and select Visible Items≫Graph Palette from the shortcut menu to display the graph palette, as shown in Figure 5.14. You can place the graph palette anywhere on the front panel.

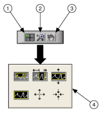


Figure 5.14: 1. Cursor Movement Tool, 2. Zoom Button, 3. Panning Tool, 4. Zoom Pull-down Menu

   Use the Zoom button on the graph palette to zoom in or out of sections of the chart or the whole chart. Use the Panning tool to pick up the plot and move it around on the display. Use the Cursor Movement tool to move the cursor on the graph.
6. Run the VI. While the VI runs, use the buttons in the scale legend and graph palette to modify the waveform chart.

   NOTE: If you modify the axis labels, the display might become larger than the maximum size that the VI can correctly present.
7. Use the Operating tool to click the Power switch and stop the VI.
8. Save and close the VI.

5.4 Waveform and XY Graphs[4]

VIs with graphs usually collect the data in an array and then plot the data to the graph. Figure 5.15 shows the elements of a graph.

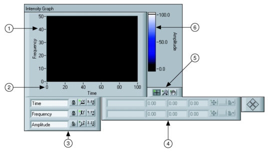


Figure 5.15

The graphs located on the `Controls`≫`Graph Indicators` palette include the `waveform graph` and `XY graph`. The `waveform graph` plots only single-valued functions, as in $y = f(x)$, with points evenly distributed along the x-axis, such as acquired time-varying waveforms. `XY graphs` display any set of points, evenly sampled or not.

Resize the plot legend to display multiple plots. Use multiple plots to save space on the front panel and to make comparisons between plots. `XY` and `waveform graphs` automatically adapt to multiple plots.

5.4.1 Single Plot Waveform Graphs

The `waveform graph` accepts a single array of values and interprets the data as points on the graph and increments the x index by one starting at $x = 0$. The graph also accepts a cluster of an initial x value, a $\Delta(x)$, and an array of y data. Refer to the `Waveform Graph` VI in the `NI Example Finder` for examples of the data types that single-plot `waveform graphs` accept.

5.4.2 Multiplot Waveform Graphs

A multiplot `waveform graph` accepts a 2D array of values, where each row of the array is a single plot. The graph interprets the data as points on the graph and increments the x index by one, starting at $x = 0$. Wire a 2D array data type to the graph, right-click the graph, and select `Transpose Array` from the shortcut menu to handle each column of the array as a plot. Refer to the `(Y) Multi Plot 1 graph` in the `Waveform Graph` VI in the `NI Example Finder` for an example of a graph that accepts this data type.

A multiplot `waveform graph` also accepts a cluster of an x value, a $\Delta(x)$ value, and a 2D array of y data. The graph interprets the y data as points on the graph and increments the x index by $\Delta(x)$, starting at $x = 0$. Refer to the `(Xo, dX, Y) Multi Plot 3 graph` in the `Waveform Graph` VI in the `NI Example Finder` for an example of a graph that accepts this data type.

A multiplot `waveform graph` accepts a cluster of an initial x value, a $\Delta(x)$ value, and an array that contains clusters. Each cluster contains a point array that contains the y data. You use the `Bundle` function to bundle the arrays into clusters, and you use the `Build Array` function to build the resulting clusters into an array. You also can use the `Build Cluster` Array, which creates arrays of clusters that contain inputs you specify. Refer to the `(Xo, dX, Y) Multi Plot 2` graph in the `Waveform Graph` VI in the `NI Example Finder` for an example of a graph that accepts this data type.

[4] This content is available online at <http://cnx.org/content/m12236/1.1/>.

5.4.3 Single Plot XY Graphs

The single-plot XY graph accepts a cluster that contains an x array and a y array. The XY graph also accepts an array of points, where a point is a cluster that contains an x value and a y value. Refer to the XY Graph VI in the NI Example Finder for an example of single-plot XY graph data types.

5.4.4 Multiplot XY Graphs

The multiplot XY graph accepts an array of plots, where a plot is a cluster that contains an x array and a y array. The multiplot XY graph also accepts an array of clusters of plots, where a plot is an array of points. A point is a cluster that contains an x value and a y value. Refer to the XY Graph VI in the NI Example Finder for an example of multiplot XY graph data types.

5.5 Graph Waveform Array VI[5]

Exercise 5.5.1
Complete the following steps to build a VI that generates and plots an array on a waveform graph and modify the VI to graph multiple plots.

5.5.1 Front Panel

1. Open a blank VI and build the front panel shown in Figure 5.16.

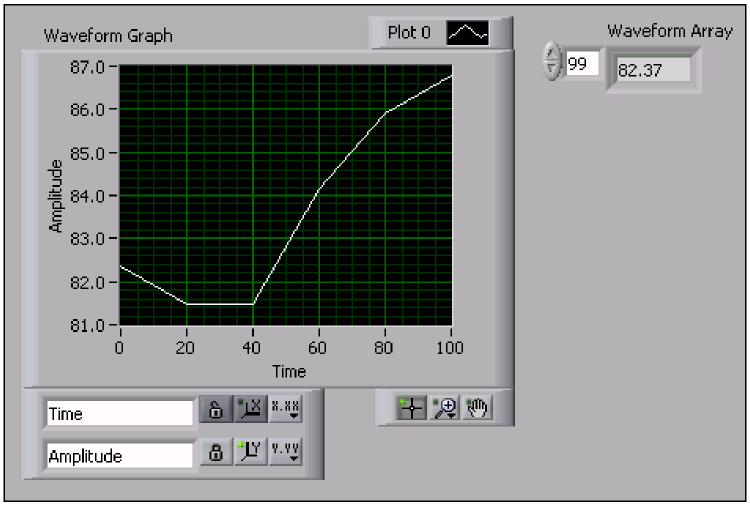


Figure 5.16

a. Place an array, located on the Controls≫All Controls≫Array &Cluster palette, on the front panel.
b. Label the array Waveform Array.
c. Place a numeric indicator, located on the Controls≫Numeric Indicators palette, in the array shell.

[5] This content is available online at <http://cnx.org/content/m12237/1.2/>.

d. Place a waveform graph, located on the Controls≫Graph Indicators palette, on the front panel.

5.5.2 Block Diagram

1. Build the block diagram shown in Figure 5.17.

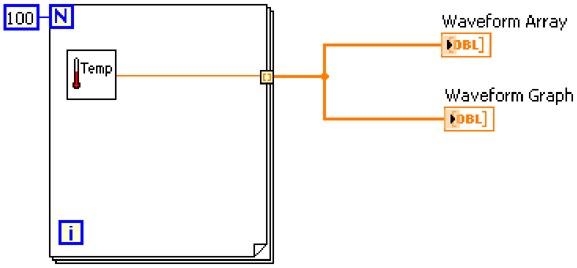


Figure 5.17

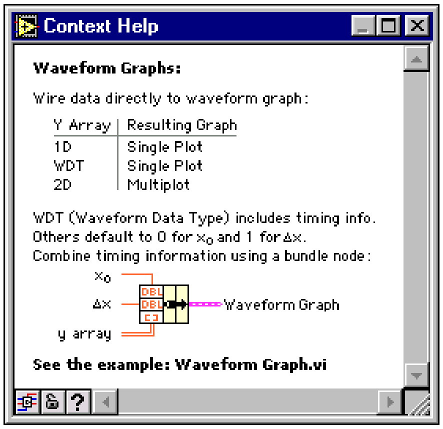
a. Place the Thermometer VI on the block diagram. Select Functions≫All Functions≫Select a VI and navigate to C:\Exercises\ LabVIEW Basics I\Thermometer.vi. This subVI returns one temperature reading during each For Loop iteration.

NOTE: Use the (Demo) Thermometer VI if you do not have a DAQ device available.

b. Place a For Loop, located on the Functions≫All Functions≫Structures palette, on the block diagram. In this exercise, each For Loop iteration generates a temperature value and stores it in the indexed tunnel. Create a constant of 100 for the count terminal.

c. Wire the block diagram as shown in Figure 5.17.

NOTE: When you wire data to charts and graphs, refer to the Context Help window for more information about wiring the objects, including whether to use a Build Array or Bundle function, the order of the input terminals, and so on. In general, use a waveform chart for single scalar points, a waveform graph for an array of y values, and an XY graph for an array of x values and an array of y values. For example, if you move the cursor over a waveform graph terminal on the block diagram, the information shown in Figure 5.18 appears in the Context Help window. Select Help≫Find Examples to launch the NI Example Finder, double-click Fundamentals, double-click Graphs and Charts, and double-click the Waveform Graph VI to open the example. Refer to the lesson on Data Acquisition and Waveforms for more information about the waveform data type.



Waveform Graphs:

Wire data directly to waveform graph:

Y Array	Resulting Graph
1D	Single Plot
WDT	Single Plot
2D	Multiplot

WDT (Waveform Data Type) includes timing info.
Others default to 0 for x_0 and 1 for Δx.
Combine timing information using a bundle node:

See the example: Waveform Graph.vi

Figure 5.18

2. Save the VI as `Graph Waveform Array.vi` in the `C:\Exercises\LabVIEW Basics I` directory.

5.5.3 Run the VI

1. Display the front panel and run the VI. The VI plots the auto-indexed waveform array on the `waveform graph`.
2. Enter the index of any element in the `Waveform Array` index display to view the value of that element. If you enter a number greater than the array size of 100, the display dims.
3. Use the `Positioning` tool to resize `Waveform Array` to view more than one element. The indicator displays elements in ascending index order, beginning with the index you entered.

5.5.4 Block Diagram

In this block diagram, you use the default value of the initial x and $\Delta(x)$ value for the waveform. In cases where the initial x and $\Delta(x)$ value are a specific value, use the `Bundle` function to specify an initial x and $\Delta(x)$ value for a `waveform array`.

1. Modify the block diagram as shown in Figure 5.19.

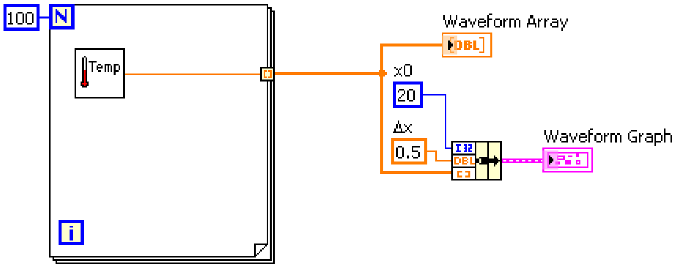


Figure 5.19

a. Place the `Bundle` function, located on the `Functions≫All Functions≫Cluster` palette, on the block diagram. This function assembles the plot elements into a single cluster. The elements include the initial x value (20), the $\Delta(x)$ value (0.5), and the y array of waveform data.
b. Create two numeric constants for the initial x value and $\Delta(x)$ value.
c. Label the $\Delta(x)$ constant by typing `Dx`. Use the `Labeling` tool to select the `D` and select the `Symbol` font from the `Text Settings` pull-down menu on the toolbar. `D` converts to the delta symbol (Δ).
d. Wire the block diagram as shown in Figure 5.19.

2. Save the VI.

5.5.5 Run the VI

1. Display the front panel and run the VI. The graph displays the same 100 points of data with a starting value of 20 and a $\Delta(x)$ of 0.5 for each point on the x-axis. In a timed test, this graph would correspond to 50 seconds worth of data starting at 20 seconds.

   NOTE: Change the initial x and $\Delta(x)$ values in only one location, either the `Bundle` function or in the `Waveform Graph Properties` dialog box.

2. If time permits, complete the optional steps. Otherwise, close the VI.

5.5.6 Optional

1. Right-click the `waveform graph` and select `Visible Items≫Graph Palette` from the shortcut menu to display the graph palette. Click the `Zoom` button to see the data on the graph in more detail.
2. Right-click the graph and select `Visible Items≫Scale Legend` from the shortcut menu to display the scale legend.
3. Return to the block diagram. Create a multiple-plot waveform graph by building a 2D array of the data type normally passed to a single-plot graph. Modify the block diagram as shown in Figure 5.20.

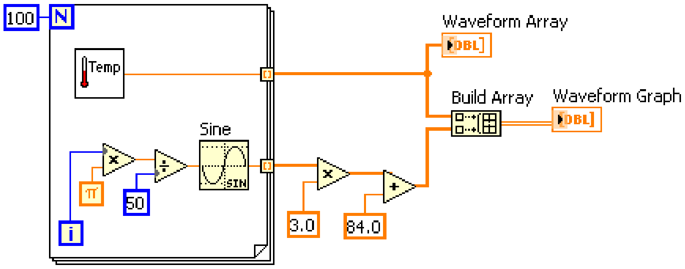


Figure 5.20

a. Place the `Sine` function, located on the `Functions≫Arithmetic & Comparison≫Express Numeric≫Express Trigonometric` palette, on the block diagram. Use this function to build an array of points that represents one cycle of a sine wave.

b. Place the `Build Array` function, located on the `Functions≫All Functions≫Array` palette, on the block diagram. This function creates the data structure to plot two arrays on a `waveform graph`.

c. Place the `pi` constant, located on the `Functions≫Arithmetic & Comparison≫Express Numeric≫Express Numeric Constants` palette, on the block diagram.

d. Wire the block diagram as shown Figure 5.20.

4. Save the VI.
5. Display the front panel and run the VI. The two waveforms plot on the same `waveform graph`.
6. Display the block diagram.
7. Right-click the wire to `Waveform Array` , select `Custom Probes≫Controls≫Graph Indicators` from the shortcut menu, and select a waveform graph to place a graph probe on the wire.
8. Display the front panel and run the VI. The probe shows only the data array. The sine wave is not present because you did not place the probe on the wire to which the sine wave is bundled.
9. Close the `Probe` window.
10. Zoom in on a portion of the graph.

a. Click the `Zoom` button on the graph palette, shown in this media, to display the `Zoom` pull-down menu.

b. Select `Zoom by X Rectangle`, as shown in Figure 5.21.

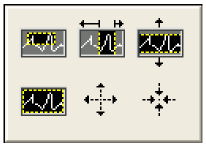


Figure 5.21

c. Click and drag a selection rectangle on the graph. When you release the mouse button, the graph display zooms in on the selected area.

d. You also can select `Zoom by Y Rectangle` or `Zoom by Selected Area`. Experiment with these options.

e. Select `Undo Zoom` from the lower left corner of the pull-down menu to undo a zoom or click the x-axis single fit button and the y-axis single fit button on the scale legend, shown in this media and this media.

11. Use the `Panning` tool, shown in this media, to click and drag the graph display. Click the x-axis and y-axis single fit buttons again to restore the display to its original position.
12. Use the `Cursor Movement` tool, shown in this media, to return the cursor to standard mode.
13. Save and close the VI.

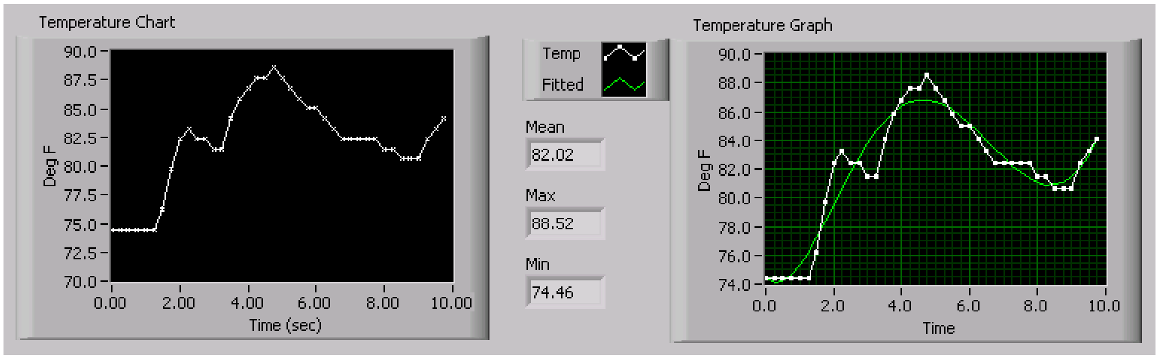
5.6 Temperature Analysis VI[6]

Exercise 5.6.1

Complete the following steps to build a VI that measures temperature every 0.25 seconds for 10 seconds. During the acquisition, the VI displays the measurements in real time on a `waveform chart`. After the acquisition is complete, the VI plots the data on a graph and calculates the minimum, maximum, and average temperatures. The VI displays the best fit of the temperature graph.

5.6.1 Front Panel

1. Open a blank VI and build the front panel shown in Figure 5.22.



Mean
82.02

Max
88.52

Min
74.46

Figure 5.22

a. Set the point style of the `waveform chart` plot to a small x.
b. Hide the plot legend of the `waveform chart`.
c. Change the label of the `waveform chart` to `Temperature Chart`.
d. Change the label of the `waveform graph` to `Temperature Graph`.
e. Right-click `Temperature Chart` and select `Properties` from the shortcut menu. On the `Scales` tab, change the x-axis multiplier to `0.25`. On the `Format and Precision` tab, change the `Digits of Precision` to 2. The data for `Temperature Graph` will be formatted on the block diagram.
f. Resize the plot legend of the `waveform graph` to show two plots.
g. Change the name of `Plot 0` to `Temp` and `Plot 1` to `Fitted`.
h. Set the point style of the `waveform graph` `Temp` plot to a small square.
i. Do not create the `Mean`, `Max`, and`Min`indicators yet. They will be created from the block diagram.

5.6.2 Block Diagram

1. Build the block diagram Figure 5.23.

[6] This content is available online at <http://cnx.org/content/m12238/1.1/>.

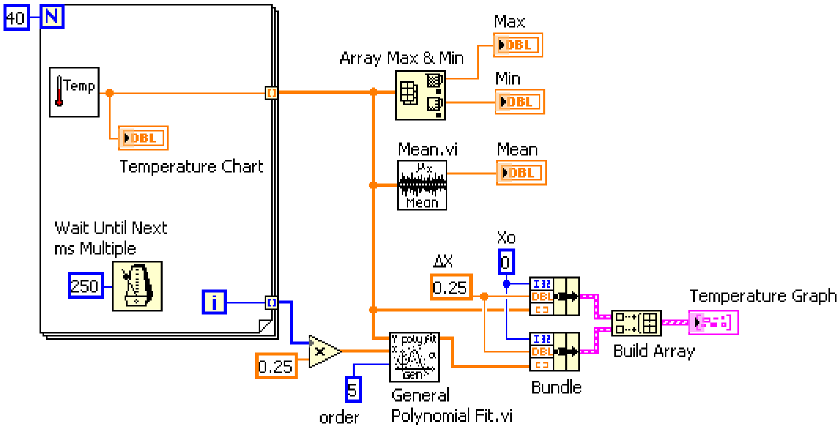


Figure 5.23

a. Place the Thermometer VI on the block diagram. Select Functions≫All Functions≫Select a VI and navigate to C:\Exercises\LabVIEW Basics I\Thermometer.vi. This subVI returns one point of temperature data.

b. Place the Wait Until Next ms Multiple function, located on the Functions≫All Functions≫Time & Dialog palette, on the block diagram. This function causes the For Loop to execute every 0.25*seconds* (250*ms*).

c. Place the Array Max & Min function, located on the Functions≫All Functions≫Array palette, on the block diagram. This function returns the maximum and minimum temperature.

d. Place the Mean VI, located on the Functions≫All Functions≫Analyze≫Mathematics≫Probability and Statistics palette, on the block diagram. This subVI returns the average of the temperature measurements.

e. Right-click the output terminals of the Array Max & Min function and Mean VI and select Create≫Indicator from the shortcut menu to create the Max, Min, and Mean indicators.

f. Place the General Polynomial Fit VI, located on the Functions≫All Functions≫Analyze≫Mathematics≫Curve Fitting palette, on the block diagram. This subVI returns an array that is a polynomial fit to the temperature array.

g. Place the Bundle function, located on the Functions≫All Functions≫Cluster palette, on the block diagram. This function assembles the plot elements into a single cluster. Press the <Ctrl> key while you drag the function to copy it. The elements include the initial x value (0), the $\Delta(x)$ value (0.25), and the y array of temperature data. The $\Delta(x)$ value of `0.25` is required so that the VI plots the temperature array points every 0.25 seconds on the `waveform graph`.

h. Place the `Build Array` function, located on the `Functions≫All Functions≫Array` palette, on the block diagram. This function creates an array of clusters from the temperature cluster and the best fit cluster.

i. Complete the block diagram as shown.

2. Save the VI as `Temperature Analysis.vi` in the `C:\Exercises\LabVIEW Basics I` directory.

5.6.3 Run the VI

1. Display the front panel and run the VI. The graph displays both the temperature data and best fit curve of the temperature waveform.
2. Try different values for the polynomial order constant on the block diagram and run the VI again.
3. Change the appearance of the plots by modifying the plot styles and fill styles.
   a. Right-click the `Temp` plot display in the plot legend and select `Common Plots≫Scatter Plot` from the shortcut menu, the top middle option.
   b. Right-click the `Fitted` plot display in the plot legend, select `Bar Plots` from the shortcut menu, and select the second option in the middle row. The `waveform graph` should appear similar to the front panel in Figure 5.24.

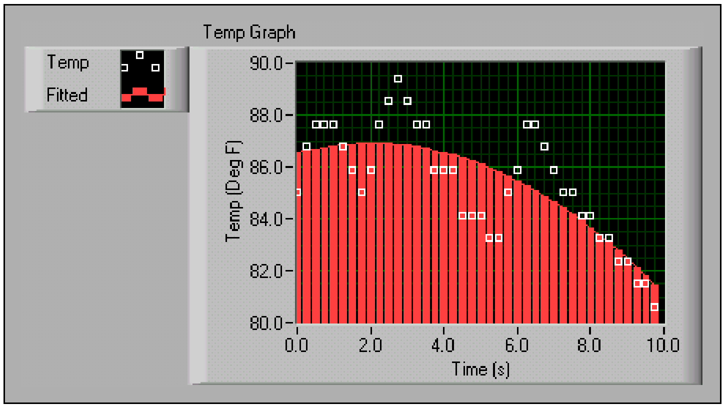


Figure 5.24

4. Save and close the VI.

5.7 Graph Circle VI[7]

Exercise 5.7.1

Complete the following steps to build a VI that plots a circle using independent x and y arrays.

[7] This content is available online at <http://cnx.org/content/m12239/1.1/>.

5.7.1 Front Panel

1. Open a blank VI and build the front panel shown in Figure 5.25.

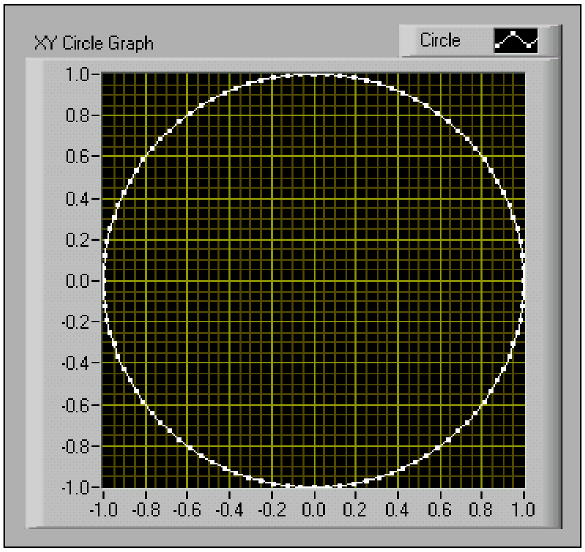


Figure 5.25

a. Place an `XY Graph`, located on the `Controls`≫`All Controls`≫`Graph` palette, on the front panel.
b. Label the graph `XY Circle Graph`.
c. Change `Plot 0` to `Circle` in the plot legend.
d. Right-click the plot in the plot legend, select `Point Style` from the shortcut menu, and select the small square.
e. Change the scale labels and ranges, as shown in Figure 5.25.

5.7.2 Block Diagram

1. Build the block diagram shown in Figure 5.26.

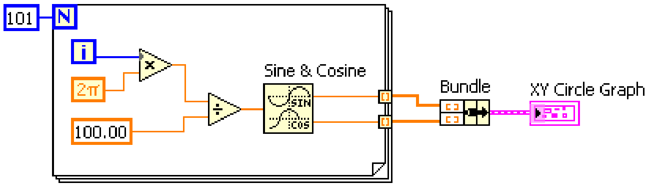


Figure 5.26

a. Place the `Sine & Cosine` function, located on the `Functions≫Arithmetic & Comparison≫Express Numeric≫Express Trigonometric` palette, on the block diagram. This function builds an array of points that represents one cycle of a sine wave and a cosine wave.

b. Place the `Bundle` function, located on the `Functions≫All Functions≫Cluster` palette, on the block diagram. This function assembles the sine array and the cosine array to plot the sine array against the cosine array to produce a circle.

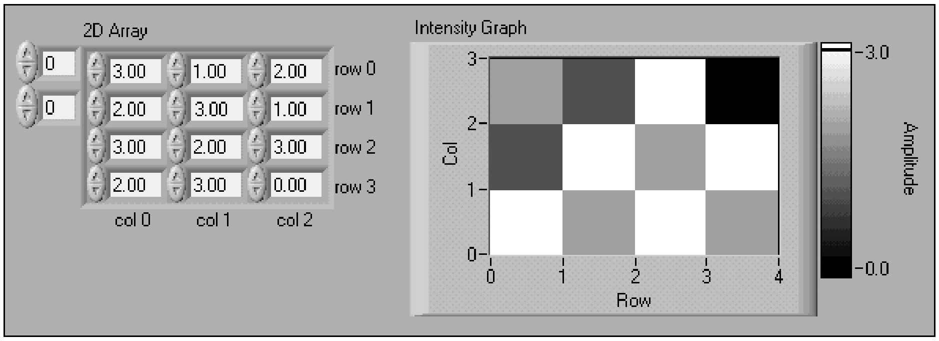
c. Place the `Pi Multiplied by 2` constant, located on the `Functions≫Arithmetic & Comparison≫Express Numeric≫Express Numeric Constants` palette, on the block diagram.

2. Save the VI as `Graph Circle.vi` in the `C:\Exercises\LabVIEW Basics I` directory.
3. Display the front panel and run the VI.
4. Close the VI.

5.8 Intensity Plots[8]

You can use intensity graphs and charts to display patterned data, such as temperature patterns and topographical terrain, where the magnitude represents altitude. Like the waveform graph and chart, the intensity graph features a fixed display while the intensity chart features a scrolling display. The intensity graph and chart accept a 2D array of numbers. Each number in the array represents a specific color. The indexes of the elements in the 2D array set the plot locations for the colors. The intensity graph or chart can display up to 256 discrete colors.

Figure 5.27 shows a 4 × 3 array plotted on an intensity graph. The graph transposes the array elements.



2D Array

0 0	col 0	col 1	col 2
row 0	3.00	1.00	2.00
row 1	2.00	3.00	1.00
row 2	3.00	2.00	3.00
row 3	2.00	3.00	0.00

Figure 5.27

[8] This content is available online at <http://cnx.org/content/m12240/1.2/>.

5.8.1 Intensity Graph and Chart Options

The intensity graphs and charts share many of the optional parts of the waveform graphs and charts, which you can show or hide by right-clicking the graph or chart and selecting `Visible Items` from the shortcut menu. In addition, because the intensity graphs and charts include color as a third dimension, a scale similar to a color ramp control defines the range and mappings of values to colors. Figure 5.28 shows the elements of an intensity graph.

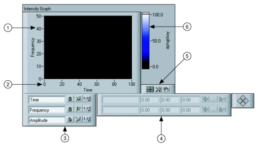


Figure 5.28

Use the `Operating` or `Positioning` tools to right-click the marker next to the color ramp, select `Marker Color` from the shortcut menu to change the color associated with a marker, and select the color you want from the color picker that appears. To add markers to a color ramp, right-click the color ramp and select `Add Marker` from the shortcut menu. To change the value of an arbitrary marker on a color ramp, use the `Operating` tool to drag the marker to the value you want or use the `Labeling` tool to highlight the text of the marker and enter a new value.

5.9 Intensity Graph Example VI[9]

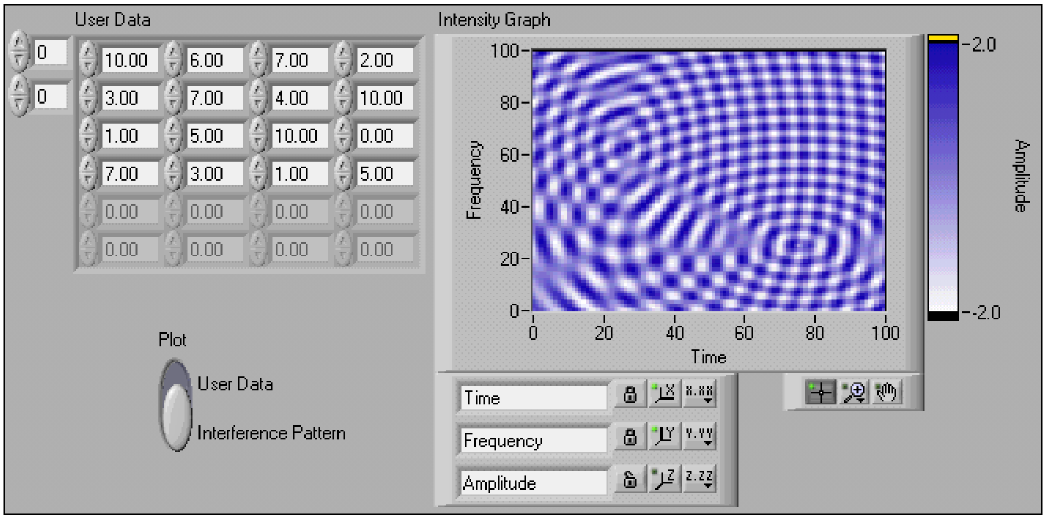
Exercise 5.9.1

In this exercise, you use a VI that displays a wave interference pattern on an intensity graph. You also use the VI to plot a 2D array of data on the graph.

5.9.1 Front Panel

1. Open and run the Intensity Graph Example (Figure 5.29) VI located in the `C:\Exercises\LabVIEW Basics I` directory. By default, the VI plots an interference waveform. A `Property Node` on the block diagram defines the color range used in the intensity graph. You can change the color range by opening the block diagram and modifying the `Color Array` constant.

[9] This content is available online at <http://cnx.org/content/m12241/1.1/>.

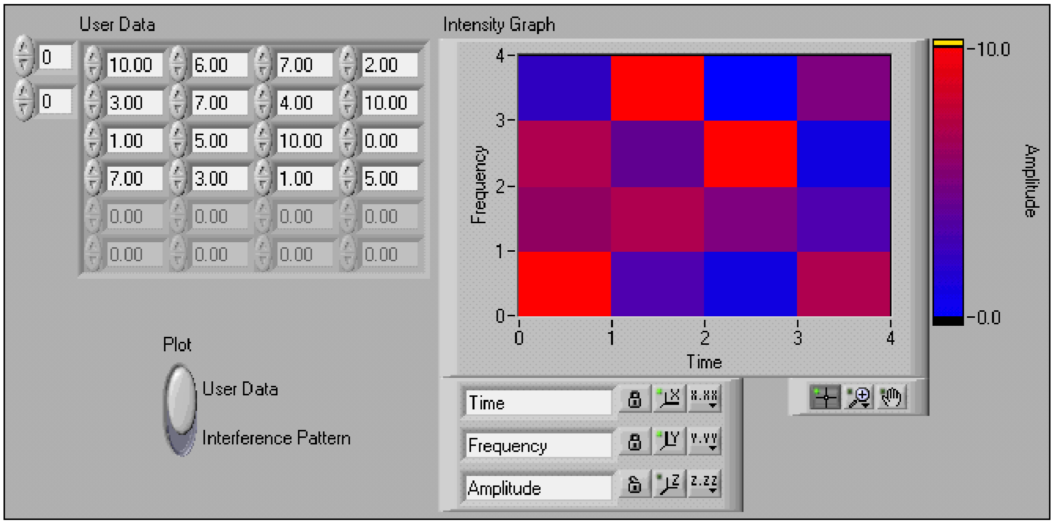


User Data

0				
0	10.00	6.00	7.00	2.00
	3.00	7.00	4.00	10.00
	1.00	5.00	10.00	0.00
	7.00	3.00	1.00	5.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00

Figure 5.29

2. Change the `Plot` switch on the front panel to `User Data` and enter values between 0.0 and 10.0 in the `User Data` array control. Run the VI. Notice how the magnitude of each element is mapped to the intensity graph.



User Data

0				
0	10.00	6.00	7.00	2.00
	3.00	7.00	4.00	10.00
	1.00	5.00	10.00	0.00
	7.00	3.00	1.00	5.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00

Figure 5.30

3. Close the VI. Do not save changes.

5.10 Summary, Tips, and Tricks on Plotting Data[10]

- The waveform chart is a special numeric indicator that displays one or more plots.
- The waveform chart has the following three update modes:
  - A strip chart shows running data continuously scrolling from left to right across the chart.
  - A scope chart shows one item of data, such as a pulse or wave, scrolling partway across the chart from left to the right.
  - A sweep display is similar to an EKG display. A sweep works similarly to a scope except it shows the old data on the right and the new data on the left separated by a vertical line.
- Waveform graphs and XY graphs display data from arrays.
- Right-click a waveform chart or graph or its components to set attributes of the chart and its plots.
- You can display more than one plot on a graph using the Build Array function located on the Functions≫All Functions≫Array palette and the Bundle function located on the Functions≫All Functions≫Cluster palette for charts and XY graphs. The graph becomes a multiplot graph when you wire the array of outputs to the terminal.
- You can use intensity charts and graphs to plot three-dimensional data. The third dimension is represented by different colors corresponding to a color mapping that you define. Intensity charts and graphs are commonly used in conjunction with spectrum analysis, temperature display, and image processing.
- When you wire data to charts and graphs, use the Context Help window to determine how to wire them.

5.11 Additional Exercises for Plotting Data[11]

Exercise 5.11.1

Build a VI that displays two plots, a random plot and a running average of the last four points, on a waveform chart in sweep update mode. Use the following tips:

- Use a For Loop ($n = 200$) instead of a While Loop.
- Use a shift register with three left terminals to average the last four data points.
- Use the Random Number (0-1) function located on the Functions≫Arithmetic & Comparison≫Express Numeric palette to generate the data.
- Use the Bundle function located on the Functions≫All Functions≫Cluster palette to group the random data with the averaged data before plotting.

Save the VI as Random Average.vi in the C:\Exercises\LabVIEW Basics I directory.

Exercise 5.11.2

Build a VI that continuously measures the temperature once per second and displays the temperature on a scope chart. If the temperature goes above or below limits specified with front panel controls, the VI turns on a front panel LED. The chart plots the temperature and the upper and lower temperature limits. You should be able to set the limit from the following front panel.

Save the VI as Temperature Limit.vi in the C:\Exercises\LabVIEW Basics I directory.

Exercise 5.11.3

Modify the VI you created in Exercise 5.11.2 to display the maximum and minimum values of the temperature trace.

TIP: Use shift registers and two Max & Min functions located on the Functions≫All Functions≫Comparison palette.

[10] This content is available online at <http://cnx.org/content/m12242/1.2/>.

[11] This content is available online at <http://cnx.org/content/m12243/1.1/>.

Select `File≫Save As` to save the VI as `Temp Limit (max-min).vi` in the `C:\Exercises\LabVIEW Basics I` directory.

Index of Keywords and Terms

Keywords are listed by the section with that keyword (page numbers are in parentheses). Keywords do not necessarily appear in the text of the page. They are merely associated with that section. *Ex.* apples, § 1.1 (1) **Terms** are referenced by the page they appear on. *Ex.* apples, 1

Attributions

Collection: *LabVIEW Graphical Programming*
Edited by: Serhat Beyenir
URL: http://cnx.org/content/col11408/1.1/
License: http://creativecommons.org/licenses/by/3.0/

Module: "LabVIEW"
By: National Instruments
URL: http://cnx.org/content/m12192/1.2/
Page: 2
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "LabVIEW Environment"
By: National Instruments
URL: http://cnx.org/content/m12193/1.3/
Pages: 3-14
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Front Panel"
By: National Instruments
URL: http://cnx.org/content/m12194/1.1/
Pages: 14-16
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Block Diagram"
By: National Instruments
URL: http://cnx.org/content/m12195/1.2/
Pages: 16-21
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Acquiring a Signal VI"
By: National Instruments
URL: http://cnx.org/content/m12196/1.3/
Pages: 21-32
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "LabVIEW Documentation Resources"
By: National Instruments
URL: http://cnx.org/content/m12199/1.1/
Pages: 33-36
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Reduce Samples VI"
By: Malan Shiralkar
URL: http://cnx.org/content/m12200/1.2/
Pages: 36-42
Copyright: Malan Shiralkar
License: http://creativecommons.org/licenses/by/1.0

Module: "Debugging Techniques"
By: National Instruments
URL: http://cnx.org/content/m12201/1.1/
Pages: 42-43
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Debug Exercise (Main) VI"
By: National Instruments
URL: http://cnx.org/content/m12202/1.1/
Pages: 43-46
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Summary, Tips, and Tricks on Introduction to LabVIEW"
By: National Instruments
URL: http://cnx.org/content/m12203/1.2/
Pages: 47-50
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Modular Programming"
By: National Instruments
URL: http://cnx.org/content/m12204/1.2/
Pages: 51-53
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Icons and Connector Panes"
By: National Instruments
URL: http://cnx.org/content/m12205/1.2/
Pages: 53-56
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Convert C to F VI"
By: National Instruments
URL: http://cnx.org/content/m12207/1.2/
Pages: 56-60
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Using SubVIs"
By: National Instruments
URL: http://cnx.org/content/m12208/1.1/
Pages: 60-61
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Thermometer VI"
By: National Instruments
URL: http://cnx.org/content/m12209/1.1/
Pages: 61-65
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Creating a SubVI from Sections of a VI"
By: National Instruments
URL: http://cnx.org/content/m12210/1.1/
Page: 65
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Summary, Tips, and Tricks on Modular Programming"
By: National Instruments
URL: http://cnx.org/content/m12211/1.1/
Page: 66
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "While Loops"
By: National Instruments
URL: http://cnx.org/content/m12212/1.2/
Pages: 67-69
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "For Loops"
By: National Instruments
URL: http://cnx.org/content/m12214/1.2/
Pages: 69-71
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Timed Temperature VI"
By: National Instruments
URL: http://cnx.org/content/m12216/1.1/
Pages: 71-73
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Summary, Tips, and Tricks on Repetition and Loops"
By: National Instruments
URL: http://cnx.org/content/m12219/1.1/
Page: 74
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Configuring the Appearance of Front Panels"
By: National Instruments
URL: http://cnx.org/content/m12299/1.2/
Pages: 75-76
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Opening SubVI Front Panels when a VI Runs"
By: National Instruments
URL: http://cnx.org/content/m12300/1.1/
Pages: 76-77
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Pop-up Graph VI and Use Pop-up Graph VI"
By: National Instruments
URL: http://cnx.org/content/m12301/1.2/
Pages: 77-79
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Keyboard Shortcuts for Controls"
By: National Instruments
URL: http://cnx.org/content/m12302/1.1/
Pages: 79-80
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Temperature System VI"
By: National Instruments
URL: http://cnx.org/content/m12303/1.1/
Pages: 80-82
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Editing VI Properties"
By: National Instruments
URL: http://cnx.org/content/m12304/1.2/
Page: 82
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Edit Me VI"
By: National Instruments
URL: http://cnx.org/content/m12305/1.2/
Pages: 82-83
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Customizing the Controls and Functions Palettes"
By: National Instruments
URL: http://cnx.org/content/m12306/1.1/
Pages: 83-84
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Summary, Tips, and Tricks on Customizing VIs"
By: National Instruments
URL: http://cnx.org/content/m12307/1.1/
Page: 85
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Waveform Charts"
By: National Instruments
URL: http://cnx.org/content/m12233/1.2/
Pages: 87-88
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Temperature Monitor VI"
By: National Instruments
URL: http://cnx.org/content/m12234/1.2/
Pages: 89-94
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Temperature Running Average VI"
By: National Instruments
URL: http://cnx.org/content/m12235/1.2/
Pages: 94-97
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Waveform and XY Graphs"
By: National Instruments
URL: http://cnx.org/content/m12236/1.1/
Pages: 98-99
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Graph Waveform Array VI"
By: National Instruments
URL: http://cnx.org/content/m12237/1.2/
Pages: 99-103
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Temperature Analysis VI"
By: National Instruments
URL: http://cnx.org/content/m12238/1.1/
Pages: 104-106
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Graph Circle VI"
By: National Instruments
URL: http://cnx.org/content/m12239/1.1/
Pages: 106-108
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Intensity Plots"
By: National Instruments
URL: http://cnx.org/content/m12240/1.2/
Pages: 108-109
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Intensity Graph Example VI"
By: National Instruments
URL: http://cnx.org/content/m12241/1.1/
Pages: 109-110
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Summary, Tips, and Tricks on Plotting Data"
By: National Instruments
URL: http://cnx.org/content/m12242/1.2/
Page: 111
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

Module: "Additional Exercises for Plotting Data"
By: National Instruments
URL: http://cnx.org/content/m12243/1.1/
Pages: 111-112
Copyright: National Instruments
License: http://creativecommons.org/licenses/by/1.0

LabVIEW Graphical Programming
Introduction to LabVIEW