NATIONAL INSTRUMENTS™
LabVIEW™

Getting Started with LabVIEW

Evaluation Version 6.0

NATIONAL INSTRUMENTS™

November 2000 Edition
Part Number 350777A-01

Worldwide Technical Support and Product Information

ni.com

National Instruments Corporate Headquarters

11500 North Mopac Expressway Austin, Texas 78759-3504 USA Tel: 512 794 0100

Worldwide Offices

Australia 03 9879 5166, Austria 0662 45 79 90 0, Belgium 02 757 00 20, Brazil 011 284 5011,
Canada (Calgary) 403 274 9391, Canada (Ontario) 905 785 0085, Canada (Québec) 514 694 8521,
China 0755 3904939, Denmark 45 76 26 00, Finland 09 725 725 11, France 01 48 14 24 24,
Germany 089 741 31 30, Greece 30 1 42 96 427, Hong Kong 2645 3186, India 91805275406,
Israel 03 6120092, Italy 02 413091, Japan 03 5472 2970, Korea 02 596 7456, Mexico (D.F.) 5 280 7625,
Mexico (Monterrey) 8 357 7695, Netherlands 0348 433466, New Zealand 09 914 0488, Norway 32 27 73 00,
Poland 0 22 528 94 06, Portugal 351 1 726 9011, Singapore 2265886, Spain 91 640 0085,
Sweden 08 587 895 00, Switzerland 056 200 51 51, Taiwan 02 2528 7227, United Kingdom 01635 523545

For further support information, see the *Technical Support Resources* appendix. To comment on the documentation, send e-mail to techpubs@ni.com

Important Information

Warranty

The media on which you receive National Instruments software are warranted not to fail to execute programming instructions, due to defects in materials and workmanship, for a period of 90 days from date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace software media that do not execute programming instructions if National Instruments receives notice of such defects during the warranty period. National Instruments does not warrant that the operation of the software shall be uninterrupted or error free.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. National Instruments will pay the shipping costs of returning to the owner parts which are covered by warranty.

National Instruments believes that the information in this document is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, National Instruments reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult National Instruments if errors are suspected. In no event shall National Instruments be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, NATIONAL INSTRUMENTS MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF NATIONAL INSTRUMENTS SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. NATIONAL INSTRUMENTS WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF. This limitation of the liability of National Instruments will apply regardless of the form of action, whether in contract or tort, including negligence. Any action against National Instruments must be brought within one year after the cause of action accrues. National Instruments shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failures caused by owner's failure to follow the National Instruments installation, operation, or maintenance instructions; owner's modification of the product; owner's abuse, misuse, or negligent acts; and power failure or surges, fire, flood, accident, actions of third parties, or other events outside reasonable control.

Copyright

Trademarks

WARNING REGARDING USE OF NATIONAL INSTRUMENTS PRODUCTS

(1) NATIONAL INSTRUMENTS PRODUCTS ARE NOT DESIGNED WITH COMPONENTS AND TESTING FOR A LEVEL OF RELIABILITY SUITABLE FOR USE IN OR IN CONNECTION WITH SURGICAL IMPLANTS OR AS CRITICAL COMPONENTS IN ANY LIFE SUPPORT SYSTEMS WHOSE FAILURE TO PERFORM CAN REASONABLY BE EXPECTED TO CAUSE SIGNIFICANT INJURY TO A HUMAN.

(2) IN ANY APPLICATION, INCLUDING THE ABOVE, RELIABILITY OF OPERATION OF THE SOFTWARE PRODUCTS CAN BE IMPAIRED BY ADVERSE FACTORS, INCLUDING BUT NOT LIMITED TO FLUCTUATIONS IN ELECTRICAL POWER SUPPLY, COMPUTER HARDWARE MALFUNCTIONS, COMPUTER OPERATING SYSTEM SOFTWARE FITNESS, FITNESS OF COMPILERS AND DEVELOPMENT SOFTWARE USED TO DEVELOP AN APPLICATION, INSTALLATION ERRORS, SOFTWARE AND HARDWARE COMPATIBILITY PROBLEMS, MALFUNCTIONS OR FAILURES OF ELECTRONIC MONITORING OR CONTROL DEVICES, TRANSIENT FAILURES OF ELECTRONIC SYSTEMS (HARDWARE AND/OR SOFTWARE), UNANTICIPATED USES OR MISUSES, OR ERRORS ON THE PART OF THE USER OR APPLICATIONS DESIGNER (ADVERSE FACTORS SUCH AS THESE ARE HEREAFTER COLLECTIVELY TERMED "SYSTEM FAILURES"). ANY APPLICATION WHERE A SYSTEM FAILURE WOULD CREATE A RISK OF HARM TO PROPERTY OR PERSONS (INCLUDING THE RISK OF BODILY INJURY AND DEATH) SHOULD NOT BE RELIANT SOLELY UPON ONE FORM OF ELECTRONIC SYSTEM DUE TO THE RISK OF SYSTEM FAILURE. TO AVOID DAMAGE, INJURY, OR DEATH, THE USER OR APPLICATION DESIGNER MUST TAKE REASONABLY PRUDENT STEPS TO PROTECT AGAINST SYSTEM FAILURES, INCLUDING BUT NOT LIMITED TO BACK-UP OR SHUT DOWN MECHANISMS. BECAUSE EACH END-USER SYSTEM IS CUSTOMIZED AND DIFFERS FROM NATIONAL INSTRUMENTS' TESTING PLATFORMS AND BECAUSE A USER OR APPLICATION DESIGNER MAY USE NATIONAL INSTRUMENTS PRODUCTS IN COMBINATION WITH OTHER PRODUCTS IN A MANNER NOT EVALUATED OR CONTEMPLATED BY NATIONAL INSTRUMENTS, THE USER OR APPLICATION DESIGNER IS ULTIMATELY RESPONSIBLE FOR VERIFYING AND VALIDATING THE SUITABILITY OF NATIONAL INSTRUMENTS PRODUCTS WHENEVER NATIONAL INSTRUMENTS PRODUCTS ARE INCORPORATED IN A SYSTEM OR APPLICATION, INCLUDING, WITHOUT LIMITATION, THE APPROPRIATE DESIGN, PROCESS AND SAFETY LEVEL OF SUCH SYSTEM OR APPLICATION.

Conventions

The following conventions are used in this manual:

»	The » symbol leads you through nested menu items and dialog box options to a final action. The sequence **File»Page Setup»Options** directs you to pull down the **File** menu, select the **Page Setup** item, and select **Options** from the last dialog box.
	This icon denotes a tip, which alerts you to advisory information.
	This icon denotes a note, which alerts you to important information.
bold	Bold text denotes items that you must select or click on in the software, such as menu items and dialog box options. Bold text also denotes palette and parameter names.
italic	Italic text denotes variables, emphasis, a cross reference, or an introduction to a key concept. This font also denotes text that is a placeholder for a word or value that you must supply.
`monospace`	Text in this font denotes text or characters that you should enter from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, programs, subprograms, subroutines, device names, functions, operations, variables, filenames and extensions, and code excerpts.
Platform	Text in this font denotes a specific platform and indicates that the text following it applies only to that platform.
right-click	**(Macintosh)** Press <Command>-click to perform the same action as a right-click.

Contents

Chapter 1
Introduction to LabVIEW

Chapter 2
Virtual Instruments

Chapter 3
Measurement

Chapter 4
Debugging

Chapter 5
Where to Go from Here

Appendix A
System Requirements

Appendix B
Technical Support Resources

Glossary

1 Introduction to LabVIEW

Refer to Appendix A, *System Requirements* for more information about system configuration requirements. Refer to the *LabVIEW Release Notes* for installation instructions.

What Is LabVIEW?

LabVIEW is a graphical programming language that uses icons instead of lines of text to create applications. In contrast to text-based programming languages, where instructions determine program execution, LabVIEW uses dataflow programming, where data determine execution.

In LabVIEW, you build a user interface by using a set of tools and objects. The user interface is known as the front panel. You then add code using graphical representations of functions to control the front panel objects. The block diagram contains this code. If organized properly, the block diagram resembles a flowchart.

You can purchase several add-on software toolsets for developing specialized applications. All the toolsets integrate seamlessly in LabVIEW. Refer to the National Instruments Web site at `ni.com/labview` for more information about these toolsets.

LabVIEW is integrated fully for communication with hardware such as GPIB, VXI, PXI, RS-232, RS-485, and plug-in data acquisition devices. LabVIEW also has built-in features for connecting your application to the Internet using the LabVIEW web server and software standards such as TCP/IP networking and ActiveX.

Using LabVIEW, you can create 32-bit compiled applications that give you the fast execution speeds needed for custom data acquisition, test, measurement, and control solutions. You also can create stand-alone executables and shared libraries, like DLLs, because LabVIEW is a true 32-bit compiler.

LabVIEW contains comprehensive libraries for data collection, analysis, presentation, and storage. LabVIEW also includes traditional program development tools. You can set breakpoints, animate program execution,

and single-step through the program to make debugging and development easier.

LabVIEW also provides numerous mechanisms for connecting to external code or software through DLLs, shared libraries, ActiveX, and more. In addition, numerous add-on tools are available for a variety of application needs.

Why Should I Use LabVIEW?

LabVIEW empowers you to build your own solutions for scientific and engineering systems. LabVIEW gives you the flexibility and performance of a powerful programming language without the associated difficulty and complexity.

LabVIEW gives thousands of successful users a faster way to program instrumentation, data acquisition, and control systems. By using LabVIEW to prototype, design, test, and implement your instrument systems, you can reduce system development time and increase productivity by a factor of 4 to 10.

LabVIEW also gives you the benefits of a large installed user base, years of product feedback, and powerful add-on tools. Finally, National Instruments technical support and Developer Zone ensure successful development of your solutions.

How Does LabVIEW Work?

LabVIEW programs are called virtual instruments, or VIs, because their appearance and operation imitate physical instruments, such as oscilloscopes and multimeters. Every VI uses functions that manipulate input from the user interface or other sources and display that information or move it to other files or other computers.

A VI contains the following three components:

- **Front panel**—Serves as the user interface.
- **Block diagram**—Contains the graphical source code of the VI that defines its functionality.
- **Icon and connector pane**—Identifies the VI so that you can use the VI in another VI. A VI within another VI is called a subVI. A subVI corresponds to a subroutine in text-based programming languages.

Front Panel

The front panel is the user interface of the VI. 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.

Block Diagram

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. You cannot delete a terminal from the block diagram. The terminal disappears only after you delete its corresponding object on the front panel.

Every control or indicator on the front panel has a corresponding terminal on the block diagram. Additionally, the block diagram contains functions and structures from built-in LabVIEW VI libraries. Wires connect each of the nodes on the block diagram, including control and indicator terminals, functions, and structures.

Palettes

LabVIEW palettes give you the options you need to create and edit the front panel and block diagram.

Tools Palette

The **Tools** palette is available on the front panel and the block diagram. A tool is a special operating mode of the mouse cursor. When you select a tool, the cursor icon changes to the tool icon. Use the tools to operate and modify front panel and block diagram objects.

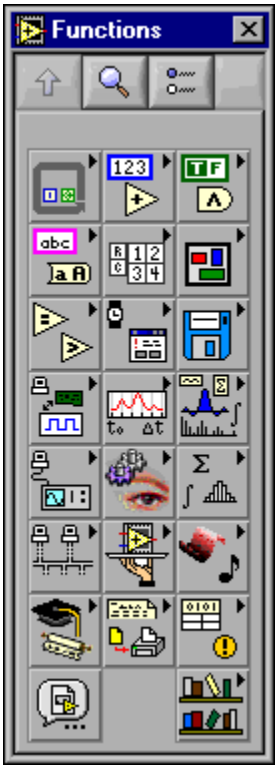
Select **Window»Show Tools Palette** to display the **Tools** palette. You can place the **Tools** palette anywhere on the screen.

Controls Palette

The **Controls** palette is available only on the front panel. The **Controls** palette contains the front panel controls and indicators you use to create the user interface. Select **Window»Show Controls Palette** or right-click the front panel workspace to display the **Controls** palette. You can place the **Controls** palette anywhere on the screen.

Functions Palette

The **Functions** palette is available only on the block diagram. The **Functions** palette contains the objects you use to program your VI, such as arithmetic, instrument I/O, file I/O, and data acquisition operations. Select **Window»Show Functions Palette** or right-click the block diagram workspace to display the **Functions** palette. You can place the **Functions** palette anywhere on the screen.



Navigating the Controls and Functions Palettes

Use the navigation buttons on the **Controls** and **Functions** palettes to navigate and search for controls, VIs, and functions. When you click a subpalette icon, the entire palette changes to the subpalette you selected. You also can right-click a VI icon on the palette and select **Open VI** from the shortcut menu to open the VI.

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

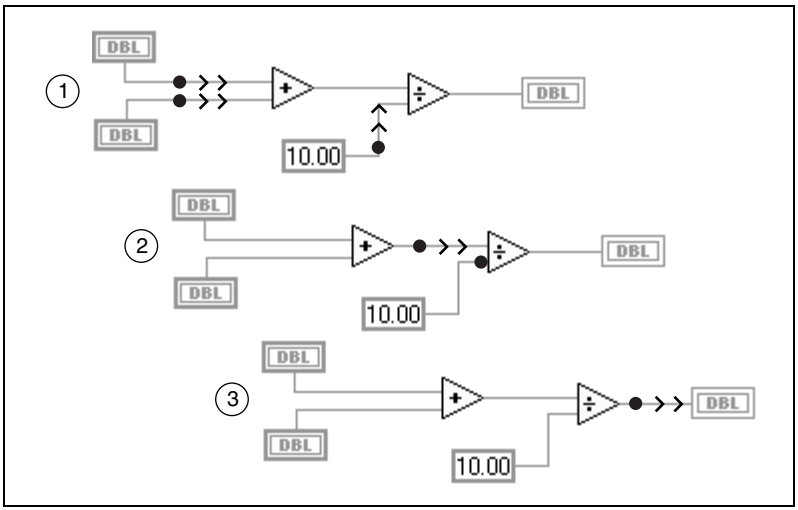
- **Up**—Takes you up one level in the palette hierarchy.

- **Search**—Changes the palette to search mode. In search mode, you can perform text-based searches to locate controls, VIs, or functions in the palettes.

- **Options**—Opens the **Function Browser Options** dialog box, from which you can configure the appearance of the palettes.

Data Flow

LabVIEW follows a dataflow model for running VIs. A block diagram node executes when all its inputs are available. When a node completes execution, it supplies data to its output terminals and passes the output data to the next node in the dataflow path.



Where Do I Start?

If you are new to LabVIEW, use this *Getting Started with LabVIEW* manual and the *LabVIEW Tutorial* to help you get started quickly. The *LabVIEW Tutorial* introduces you to the LabVIEW environment. *Getting Started with LabVIEW* teaches you how to build VIs for data acquisition and instrument control and how to debug them. It also teaches you how to use the Search Examples feature and the DAQ Solution Wizard.

You can complete the activities in this book in approximately 90 minutes.

LabVIEW Tutorial

Use this tutorial to learn basic LabVIEW concepts. The tutorial guides you through several activities to familiarize you with graphical programming. Access the *LabVIEW Tutorial* by selecting **Help»Contents and Index** or by clicking the **LabVIEW Tutorial** button in the **LabVIEW** dialog box, shown in the following illustration.

You can complete the *LabVIEW Tutorial* in approximately 15 minutes.

If you are already running LabVIEW, either launch LabVIEW or close all open VIs including the Demo VI to access the **LabVIEW** dialog box.

After you finish the *LabVIEW Tutorial*, continue with the activities in this manual to learn how to build LabVIEW programs for instrument I/O, data acquisition, and control.

You should complete the activities in this book in the order in which they appear because subsequent activities build on the sample programs you create. Approximate completion times are given for each activity and each section within an activity.

2

Virtual Instruments

This chapter teaches you step by step how to create an application in LabVIEW. It also guides you through the Search Examples feature to help you find examples in LabVIEW.

You will learn to do the following:

- Create a new program in LabVIEW. The VI you build generates data, analyzes it, then writes it to a file.
- Use the Search Examples feature to find and run an example.

Search for Examples

After completing the *LabVIEW Tutorial*, you are ready to run some examples in LabVIEW.

You can complete this activity in approximately 5 minutes.

1. In the **LabVIEW** dialog box, click the **Search Examples** button to open the *Search Examples* online help, which lists LabVIEW examples.

   If you are already running LabVIEW, either launch LabVIEW or close all open VIs including the Demo VI to access the **LabVIEW** dialog box.
2. Select **Analysis** in the **Demonstrations** category.
3. Click **Temperature System Demo** to open this demonstration VI.

4. Click the **Run** button on the front panel toolbar to see how the VI runs.
5. Click the sliders, knobs, and other controls to see how they affect the data.
6. Stop the program by clicking the **Acquisition** switch to the **off** position.

Build a Virtual Instrument

This activity demonstrates how you can plot, analyze, and save data in LabVIEW.

You can complete this activity in approximately 30 minutes.

Create a User Interface

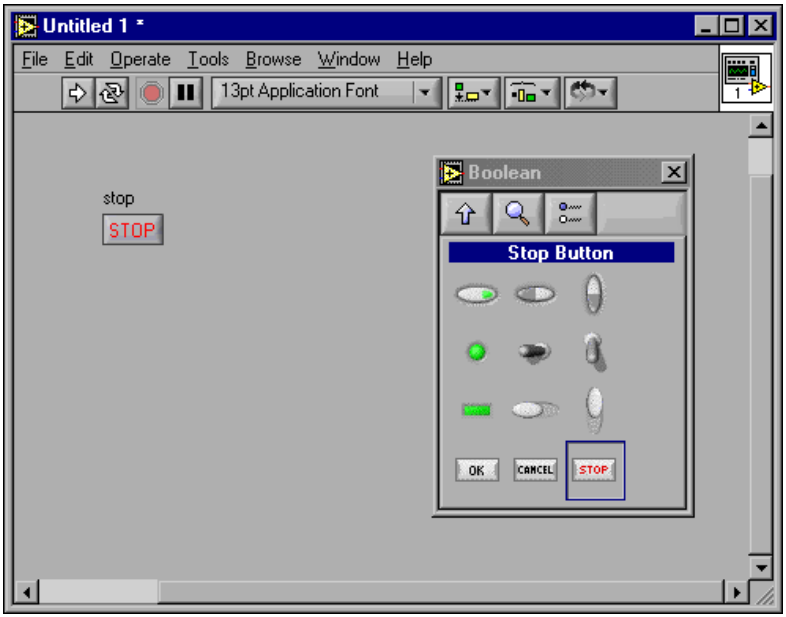
You can create a graphical user interface on the front panel using controls and indicators on the **Controls** palette.

You can complete this section in approximately 5 minutes.

1. Create a new VI by selecting **New VI** in the **LabVIEW** dialog box.

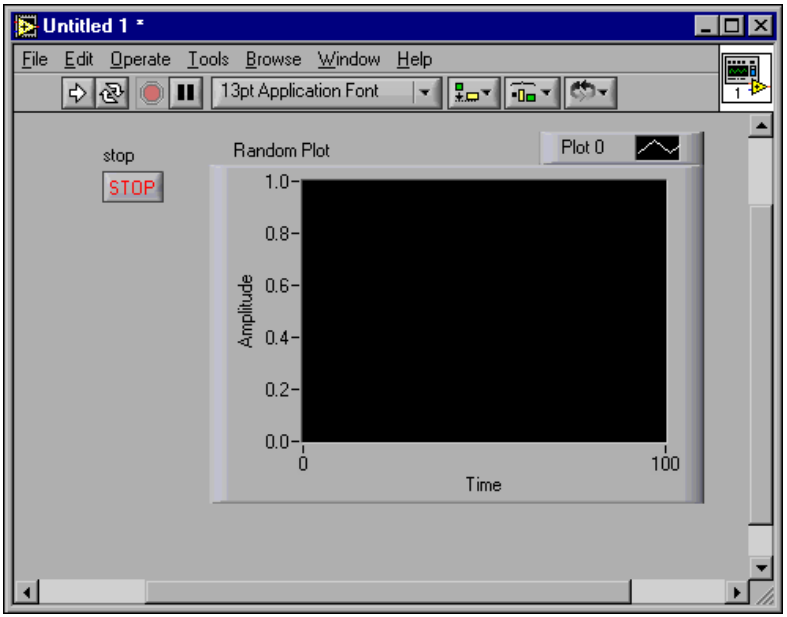
   If you are already running LabVIEW, either launch LabVIEW or close all open VIs including the Demo VI to access the **LabVIEW** dialog box.
2. Create a stop button by selecting **Controls»Boolean»Stop Button** on the **Controls** palette, as shown in the following illustration.
3. Right-click the object and select **Visible Items»Label** from the shortcut menu to show or hide the text label for an object.

4. Use the Positioning tool on the **Tools** palette to rearrange or resize objects. Select **Window»Show Tools Palette** to show the **Tools** palette.



5. Use the **Up** navigation button on the **Controls** palette to navigate back to the main **Controls** palette. Create a waveform chart by selecting **Controls»Graph»Waveform Chart**. This chart plots data one point at a time. Use the Labeling tool on the **Tools** palette to label the waveform chart `Random Plot`.

6. Select **Window»Show Tools Palette** to show the **Tools** palette. Use the Operating tool on the **Tools** palette to change the scale of the waveform chart. Double-click **-10.0** on the Y axis of the Random Plot indicator and type `0.0` to change the scale. Double-click **10.0** on the Y axis of the Random Plot indicator and type `1.0` to change the scale. The front panel should now match the following illustration.



Build the Block Diagram

So far, you have defined how the user interface will look. Now you can create the source code on the block diagram. For this VI, you generate random numbers ranging from zero to one and plot them on a chart.

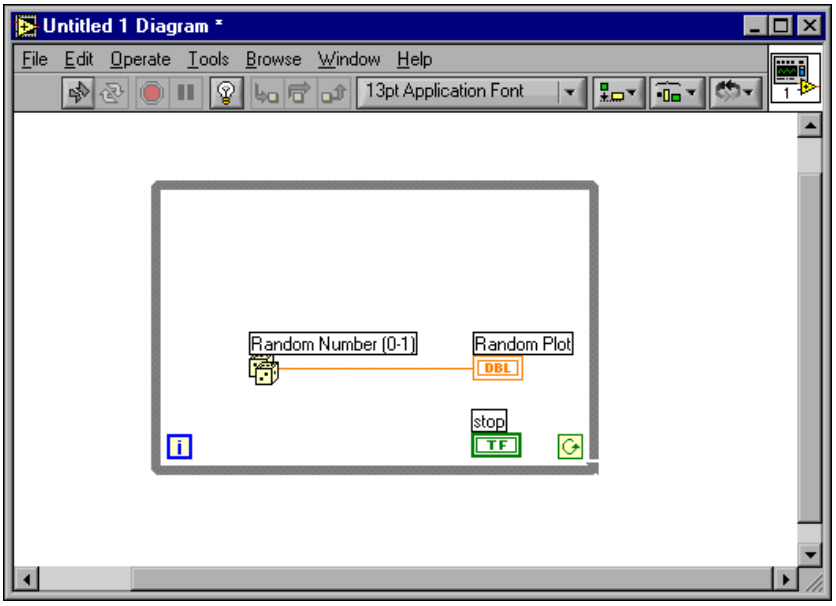
You can complete this section in approximately 5 minutes.

1. View the block diagram by selecting **Window»Show Diagram** or clicking the block diagram window.

Tip Use the keyboard shortcut <Ctrl-E> to switch between the front panel and block diagram.

2. The two terminals on the block diagram correspond to the **Stop** button and the Random Plot waveform chart on the front panel. Select **Window»Show Functions Palette** to show the **Functions** palette.
3. Select **Functions»Numeric»Random Number (0–1)**. As you drag the Random Number (0–1) function close to the Random Plot terminal, LabVIEW automatically draws a wire. When you release the mouse button to place the function, LabVIEW automatically connects the function to the terminal with a wire.

4. Use the **Up** navigation button on the **Functions** palette to navigate back to the main **Functions** palette. Create a While Loop by selecting **Functions»Structures»While Loop**. The While Loop runs all code within its boundaries until the conditional terminal receives a TRUE or FALSE value. The default behavior is **Continue if True**.
5. Place the mouse cursor in the position on the block diagram where you want to anchor the top-left corner of the While Loop. Drag the dotted rectangle diagonally to enclose the Random Number function, the Random Plot chart, and the **Stop** button terminal. The block diagram should now match the following illustration.



Wire and Run the VI

You transfer data between block diagram objects through wires. Wires are different colors, styles, and thicknesses, depending on their data types. A broken wire appears as a dashed black line. When you finish wiring this VI, you can run it from the front panel to see it plot data to the chart.

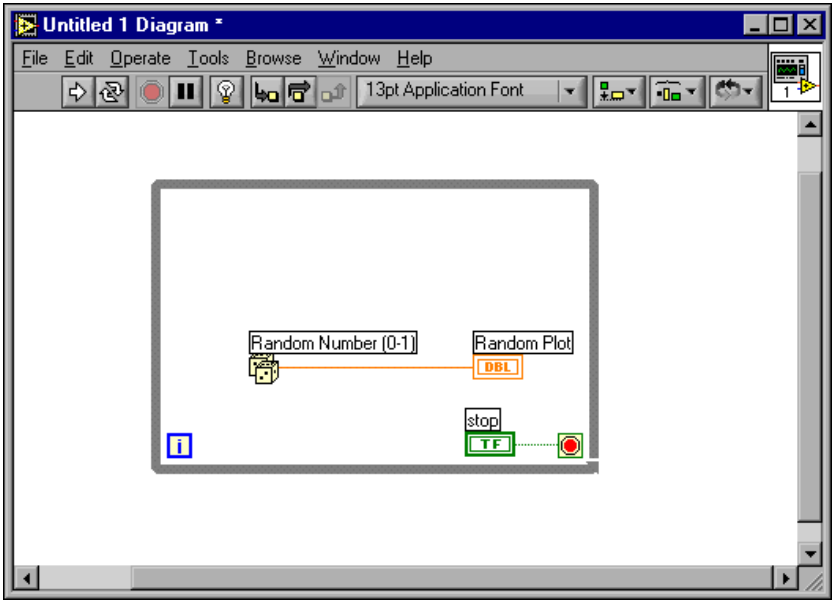
You can complete this section in approximately 5 minutes.

1. Select **Window»Show Tools Palette** to show the **Tools** palette. Select the Wiring tool on the **Tools** palette.

2. Wire the **Stop** button terminal to the While Loop conditional terminal. To do this, click the **Stop** button terminal, move the mouse cursor to the conditional terminal to create a wire and click again to connect the wire.

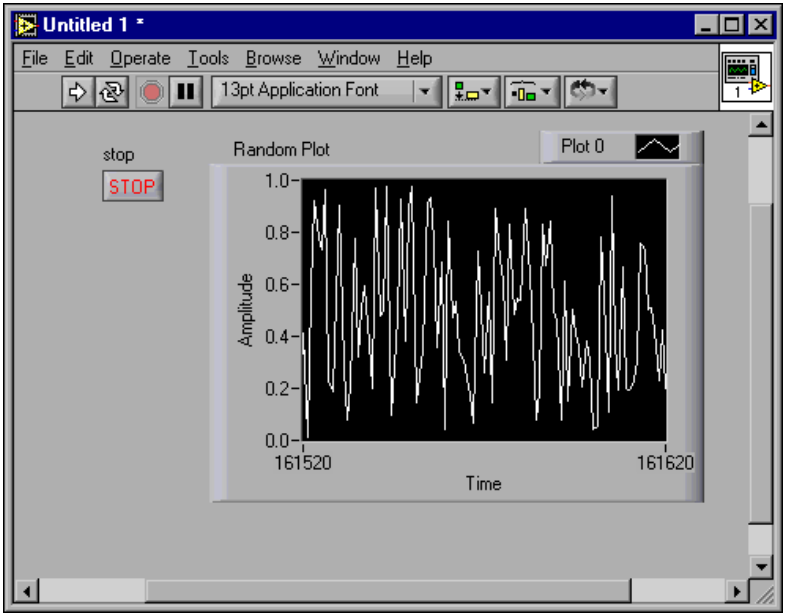
Tip When you position the Wiring tool over a terminal, the terminal blinks, and a tip strip with the name of the terminal appears.

3. Because this VI uses a **Stop** button, you must change the behavior of the While Loop conditional terminal to **Stop if True**. Right-click the conditional terminal or the border of the While Loop and select **Stop if True**. The appearance of the conditional terminal changes as shown at left. The block diagram should now match the following illustration.



4. View the front panel by pressing <Ctrl-E>. Select **Window»Show Tools Palette** to show the **Tools** palette. Select the **Operating** tool on the **Tools** palette.

5. Click the **Run** button on the toolbar to run the VI. The front panel should now resemble the following illustration.



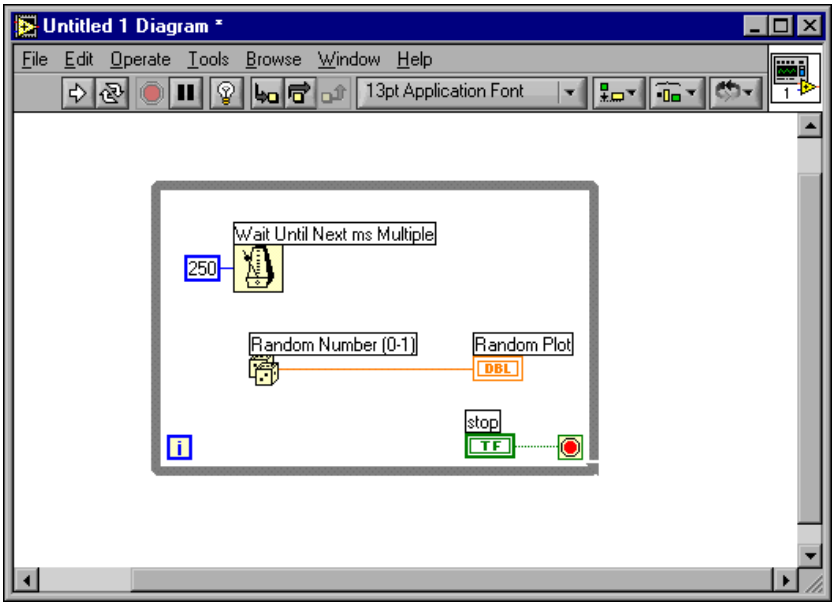
6. Click the **Stop** button to stop the VI.

Add Timing to the VI

You can add a timing delay to the VI to plot the points more slowly on the waveform chart.

You can complete this section in approximately 5 minutes.

1. View the block diagram by pressing <Ctrl-E>. On the block diagram, select **Functions»Time & Dialog»Wait Until Next ms Multiple** and place the function inside the While Loop.
2. Use the Wiring tool to right-click the **millisecond multiple** terminal on the left side of the Wait Until Next ms Multiple function and select **Create»Constant** from the shortcut menu.
3. Type `250` in the **millisecond multiple** constant to create a 250 ms delay between the generation of points to plot for the chart. The block diagram should now match the following illustration.



4. On the front panel, use the Operating tool to click the **Run** button to run the VI and observe the delay effect. Select **File»Save As** and save this VI as `Random Number Example.vi` in `labview\vi.lib\tutorial.llb`.

Add Analysis and File I/O to the VI

You can average the random data points you have collected and save the data to a spreadsheet file.

You can complete this section in approximately 10 minutes.

1. View the block diagram window of the Random Number Example VI you created. If the VI is not already open, select **File»Open** and browse to the Random Number Example VI in `labview\vi.lib\tutorial.llb`. Select **Window»Show Functions Palette** to show the **Functions** palette.
2. Select **Functions»Mathematics»Probability and Statistics» Mean.vi** and place the VI on the block diagram outside the While Loop.

Note It is important to place the Mean VI outside the While Loop because you want to compute the data mean only after the While Loop completes collecting the data.

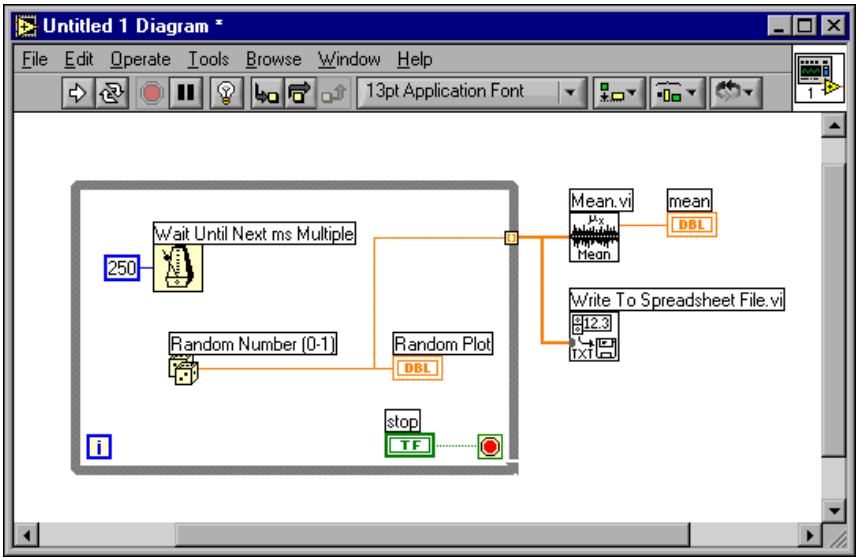
3. Use the Wiring tool to right-click the **mean** terminal on the upper right side of the Mean VI and select **Create»Indicator** from the shortcut menu to create a numeric indicator on the front panel to display the mean of the random data.
4. Select **Functions»File I/O»Write To Spreadsheet File.vi** and place the VI on the block diagram outside the While Loop.

5. Use the Wiring tool to create a wire branch connecting the Random Number function and the **X** input terminal of the Mean VI. Click the existing wire segment between the Random Number function and the **Random Plot** terminal. Connect this wire to the **X** input terminal of the Mean VI. The broken wire segment indicates that you are trying to wire terminals of different data types. You will correct the broken wires in Step 7.

Tip The wire segment blinks when the Wiring tool is positioned over it.

6. Create another wire branch from the branch you created in the previous step. Wire this new branch from the segment outside the loop to the **1D data** input of the Write to Spreadsheet File VI. You use the 1D data input because the While Loop creates a one-dimensional row of data from the generated random numbers. The wires will be broken, but you will correct that in Step 7.

7. The orange square on the While Loop is a data exit terminal called a tunnel. Right-click the tunnel and select **Enable Indexing** from the shortcut menu to allow the While Loop to collect the data and pass it to the Mean VI as a cumulative data set when the loop terminates. The broken wires change to solid orange wires because the terminals are now of the same data type. The block diagram should now match the following illustration.



8. Return to the front panel and click the **Run** button to run the VI. When you click the **Stop** button, you see the mean of the data and the **Choose file to write** dialog box prompts you for the name of the file for saving the random number data points. Type `data.txt` and click the **Save** button.

   Because the Mean VI is outside the While Loop, the VI does not display the mean until you click the **Stop** button.

9. Use any text editor to open `data.txt` and view the data.

Note You can find the solution to this activity in `LabVIEW\vi.lib\tutorial.llb\Random Number Example Solution.vi`.

3

Measurement

This chapter teaches you some basics on how to use LabVIEW to configure, communicate with, and acquire data from special purpose instruments and general purpose data acquisition (DAQ) hardware.

Refer to the National Instruments Developer Zone, at `ni.com/zone`, which offers measurement and automation resources including examples, technical presentations, instrument drivers, and tutorial information. Also, refer to the *LabVIEW Measurements Manual* for more information about how to take and analyze measurements in LabVIEW.

Instrument I/O

This section introduces you to the basic concepts on how to use LabVIEW to acquire data from instruments controlled by GPIB, VXI, RS-232, and other hardware standards. This section also shows you how to run the Demo Scope VI if you do not have instrument I/O hardware installed.

LabVIEW communicates with most instruments through instrument drivers, which are libraries of VIs that control programmable instruments. LabVIEW instrument drivers simplify instrument control and reduce test development time by eliminating the need to learn the low-level programming protocol for each instrument.

Instruments obey a set of commands to respond to remote control and requests for data. When you use LabVIEW instrument drivers, you run intuitive, high-level command VIs, such as the Read DC Voltage VI for a digital multimeter or the Configure Time Axis VI for a digital oscilloscope. The driver VI you call automatically sends the appropriate instrument-specific command strings to the instrument.

The foundation for LabVIEW drivers is the VISA (Virtual Instrument Software Architecture) VI library, a single interface library for controlling GPIB, VXI, RS-232, and other types of instruments. Drivers using VISA are scalable across instrument I/O interfaces.

Refer to Part IV, *Instrument Control in LabVIEW*, of the *LabVIEW Measurements Manual* for more information about instrument control, instrument drivers, and using VISA to communicate with instruments.

Run the Demo Scope VI

If you do not have instrument I/O hardware installed, run the Demo Scope VI. The Demo Scope VI is the demonstration equivalent of a Getting Started VI for an actual instrument driver.

You can complete this activity in approximately 5 minutes.

1. Open the Demo Scope VI instrument driver in `LabVIEW\vi.lib\tutorial.llb`.
2. Run the program to acquire simulated data on one or two channels on the oscilloscope. Change the **Time Base** and **Volts/Division** settings to see the effects.
3. Click the **STOP [F4]** button to stop the VI.
4. View the block diagram. Notice that Initialize is called first, followed by the commands to send to the instrument in the Application Example VI. The Close VI then closes communication with the instrument. When you program with LabVIEW drivers, follow this model to initialize the instrument, then call the functions to control the instrument, and finally close the instrument for communication.

Data Acquisition

This section teaches you how to use LabVIEW with general purpose data acquisition (DAQ) hardware. If you use only stand-alone instruments and control them with GPIB, VXI, or serial standards, refer to the *Instrument I/O* section of this chapter.

Refer to Part II, *DAQ Basics*, of the *LabVIEW Measurements Manual* for more information about data acquisition in LabVIEW.

Note Data acquisition and the DAQ wizards are available on Windows and Macintosh only.

You will learn to do the following:

- Use the *DAQ Solution Wizard* to generate solutions for data acquisition applications.
  - Use the *DAQ Channel Wizard* to configure an analog input channel.
  - Generate a solution from the *Solutions Gallery.*
- Add analog input to the VI you created in Chapter 2, *Virtual Instruments*.

Note Refer to your hardware manual or the NI-DAQ Help file for data acquisition hardware installation and configuration instructions.

Using the DAQ Solution Wizard

If you are using DAQ hardware, you must configure analog input, analog output, digital input, or digital output channels. You can launch the DAQ Channel Wizard from the *DAQ Solution Wizard* to configure the channels. Then you can generate a DAQ solution from the Solutions Gallery.

On Windows, you access the DAQ Channel Wizard through the Data Neighborhood in Measurement & Automation Explorer. On Macintosh you can access the DAQ Channel Wizard by selecting **Tools»Data Acquisition»DAQ Channel Wizard**. The DAQ Channel Wizard also can be accessed from the DAQ Solution Wizard.

You can complete this activity in approximately 15 minutes.

Configuring Analog Input Channels

The DAQ Solution Wizard guides you through naming and configuring analog and digital channels using the DAQ Channel Wizard. The DAQ Channel Wizard helps you define the physical quantities you are measuring or generating on each DAQ hardware channel. It queries for information about the physical quantity being measured, the sensor or actuator being used, and the associated DAQ hardware.

You can complete this section in approximately 5 minutes.

1. Click the **DAQ Solutions** button in the **LabVIEW** dialog box to launch the **DAQ Solution Wizard** and get started with analog input quickly and easily.

If you are already running LabVIEW, either launch LabVIEW or close all open VIs including the Demo VI to access the **LabVIEW** dialog box.

2. When the **Welcome to the DAQ Solution Wizard!** dialog box opens, click the **Go to DAQ Channel Wizard** button.

   If a dialog box appears that indicates the Channel Wizard is unavailable while using simulated DAQ, skip to the *Generating a Solution from the Solutions Gallery* section to complete this activity.

3. **(Windows)** Select the Data Neighborhood view in Measurement & Automation Explorer. Right-click Data Neighborhood and select **Create New** from the shortcut menu to configure a new channel. In the **Create New** dialog box select **Virtual Channel** and click the **Finish** button.

   (Macintosh) Select **New Channel** to configure a new channel.

   The DAQ Channel Wizard helps you configure analog and digital channels by name so you can use those names in your program. The DAQ Channel Wizard also conditions those channels so signal conditioning—such as scaling factors and cold-junction compensation—are performed transparently.

4. Select **Analog Input** as the channel type to configure and click the **Next** button. You also can configure analog output and digital input/output in the DAQ Channel Wizard.

5. Type a channel name and channel description in the appropriate text boxes. Click the **Next** button to continue.

6. Select the type of sensor. If the channel is a temperature measurement, click the check box. Click the **Next** button to continue.

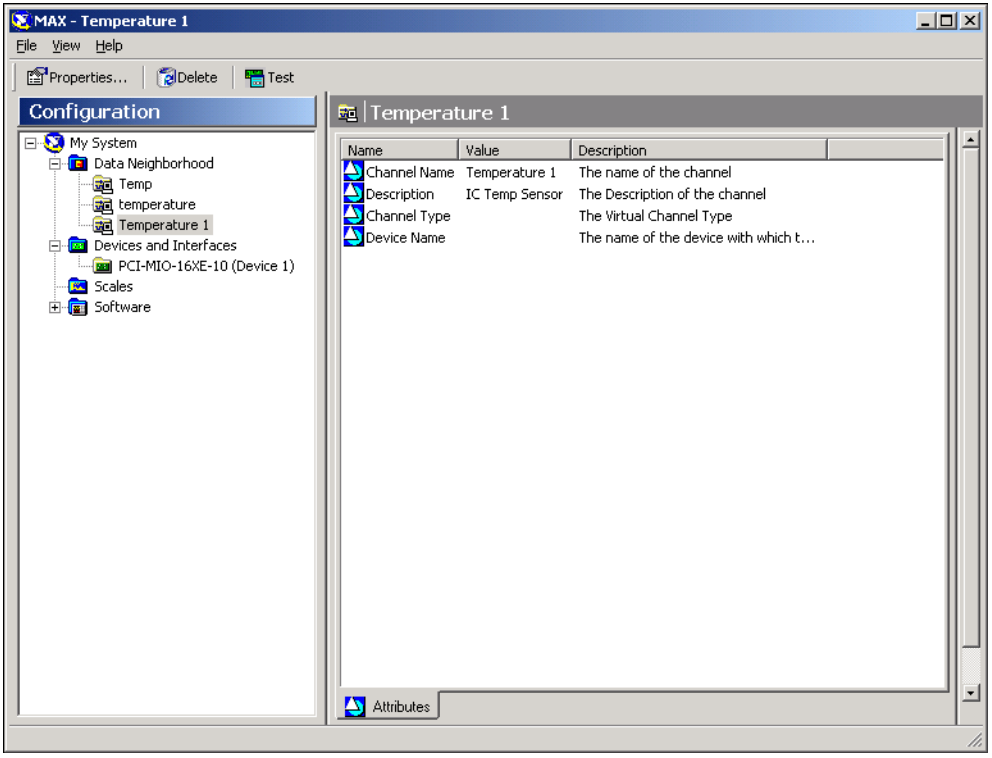
7. Define the physical quantity that you are measuring. Select the units for your measurement and enter the range for the signal in the appropriate boxes. Click the **Next** button to continue.

8. Define how the sensor scales the signal from the physical units to the hardware units. Click the **Next** button to continue.

9. Select the data acquisition device and channel settings. If you have multiple DAQ hardware devices installed, choose the device that will read this signal. Click the **Finish** button to configure the analog input channel.

10. Notice that the new configuration is listed under Data Neighborhood **(Windows)**, as shown below, or in the main DAQ Channel Wizard view **(Macintosh)**. You have finished configuring an analog input channel for your DAQ hardware. Select **File»Close** to close the Measurement &

Automation Explorer **(Windows)**. Select **Quit** to exit the DAQ Channel Wizard **(Macintosh)**.



Generating a Solution from the Solutions Gallery

Once you have configured the channel, you can generate a solution from the Solutions Gallery.

You can complete this section in approximately 10 minutes.

1. When you return to the DAQ Solution Wizard, select **Use channel names specified in DAQ Channel Wizard** and click the **Next** button.
2. Select **Solutions Gallery** and click the **Next** button to open the Solutions Gallery.
3. Choose a solution to generate for the analog input channel. For this example, in the Gallery Categories list, select **Data Logging**.
4. In the Common Solutions list, select **Simple Data Logger**.
5. Click the **Next** button to continue.

6. Select a channel name as the input channel to use for the solution.
7. Click the **Open Solution** button to generate the solution.
8. Click the **Run** button to take data from the configured analog input channel and log the data to a file. A file dialog box prompts you for a file name. Type `data.txt` and click the **Save** button.
9. Click the **STOP** button in the lower-right corner of the front panel to stop the VI. Use any text editor to view `data.txt`.

Tip You can customize the front panel and block diagram of the VI solution and save your customized solutions.

10. Save any changes you want to keep and close this VI.
11. View the DAQ Solution Wizard window and click the **Back** button to browse other gallery categories and common solution VIs.
12. Click the **Cancel** button to exit the DAQ Solution Wizard.

Adding Analog Input to the VI

The Random Number Example VI you created in Chapter 2, *Virtual Instruments* generates random numbers and plots them to a chart. Now you will replace the random number generator with an analog input VI to acquire data from your DAQ device, plot it on a chart, analyze it, and write it to a file.

You can complete this activity in approximately 5 minutes.

Note If you have not built the Random Number Example VI, you can find the solution VI in `LabVIEW\vi.lib\tutorial.llb\Random Number Example Solution.vi`.

1. Open the Random Number Example VI from `labview\vi.lib\tutorial.llb`.
2. View the block diagram. Right-click the Random Number function, and select **Replace»Data Acquisition»Analog Input» AI Sample Channel** to replace the Random Number function with the AI Sample Channel VI. Be sure to select the AI Sample Channel VI and not the AI Sample Channels VI. Note that the output has changed from a double-precision number to a waveform. This allows timing information to flow to the chart for display.

3. Use the Wiring tool to right-click the **channel (0)** input of the AI Sample Channel VI and select **Create Constant** from the shortcut menu to specify the name of the channel from which you will acquire data.
4. Select the name of a channel from the pull-down menu on the constant.
5. View the front panel. Right-click the graph and select **Y Scale»AutoScale Y** from the shortcut menu.

6. Click the **Run** button to acquire and display data on the chart.
7. Click the **Stop** button to stop the VI.
8. In the file dialog box, enter a name such as `data.txt` and click the **Save** button to save the data to a spreadsheet file.
9. Save this VI as `Acquire Data.vi` in `labview\vi.lib\tutorial.llb`.

Refer to the *LabVIEW Measurements Manual* for information about multiple-point acquisition, waveform generation, digital I/O, and counter/timer applications.

4

Debugging

This chapter teaches you how to use some of the debugging techniques available in LabVIEW.

Use Execution Highlighting

Execution highlighting traces the data flow of a VI during execution.

You can complete this activity in approximately 5 minutes.

1. Open the Random Number Example VI you created earlier from `LabVIEW\vi.lib\tutorial.llb`.
2. 
   View the block diagram and click the **Highlight Execution** button on the toolbar.
3. 
   Run the VI from the block diagram window. The program executes in slow motion with moving bubbles to highlight the flow of execution. It also displays data as it becomes available in the VI.
4. 
   Click the **Abort** button to stop the VI.
5. 
   Click the **Highlight Execution** button again to turn off execution highlighting.

Single-Step with Probes

You can set breakpoints in a VI and single-step into, out of, and over sections of the block diagram. You also can insert probes to examine data values in detail during program execution. This example shows how to insert probes and single-step through a VI to monitor data while the VI runs.

You can complete this activity in approximately 5 minutes.

1. Open the Random Number Example VI you created earlier from `LabVIEW\vi.lib\tutorial.llb`.

2. Right-click the output wire of the Random Number function and select **Probe** from the shortcut menu. A small **number:** window displays the value of the data at that point.

3. Click the **Step Into** button on the toolbar. The While Loop now flashes to indicate that the VI is running in single-step mode.
4. Click the **Step Into** button again. The Random Number function now flashes.

5. Click the **Step Over** button to step over the Random Number function. The Random Number function executes, so the output now appears in the probe.
6. Using **Step Into** and **Step Over** buttons, single-step a few more times through the loop to view each random number as the VI generates them.

7. Click the **Pause** button to run the VI in normal mode.
8. View the front panel and click the **Stop** button to stop the VI. Save the collected data to `data.txt`.

5

Where to Go from Here

This chapter describes how to use the *LabVIEW Help* and the additional resources available to you.

Refer to the *LabVIEW User Manual*, the *LabVIEW Measurements Manual*, or the *LabVIEW Help* for more information about LabVIEW programming.

Note The LabVIEW documentation set is available in Portable Document Format (PDF) on the LabVIEW CD in the `manuals` directory. You must have Adobe Acrobat Reader 3.0 or later installed to view these files.

Online Help

All built-in LabVIEW VIs and functions include a complete description of the VI functionality and parameters. Access context-sensitive VI and function help by selecting **Help»Show Context Help** or pressing <Ctrl-H>.

When you place a VI on the block diagram and pass the mouse cursor over it, the VI description displays in the **Context Help** window. Click the lock button at the bottom of the **Context Help** window to lock the current contents of the window.

Access the *LabVIEW Help* file to search for particular topics by selecting **Help»Contents and Index**. You also can right-click any built-in VI or function on the block diagram and select **Help** from the shortcut menu.

National Instruments' Commitment to You

LabVIEW represents a long-standing commitment by National Instruments to provide tools that simplify the development of instrumentation, data acquisition, and control systems. When you choose LabVIEW as your development environment, you join thousands of scientists and engineers who are taking advantage of the power of graphical programming.

Customer Education

For additional training, National Instruments offers interactive CDs, videos, books, and hands-on LabVIEW courses to help you master LabVIEW quickly and develop successful applications.

Alliance Program

The Alliance Program is a network of third-party developers and consultants who are experts in LabVIEW and other National Instruments products. The National Instruments *Alliance Solutions* directory lists additional libraries and utilities developed by our Alliance members to help you use LabVIEW. In addition, the *Alliance Solutions* directory lists expert LabVIEW consultants who can help you develop custom applications.

A

System Requirements

The following table describes the required system configurations for running LabVIEW on Windows, Macintosh, and UNIX platforms.

Platform	Media and System Requirements	Important Notes
All Platforms	Distributed on CD-ROM.	LabVIEW and the *LabVIEW Help* contain 16-bit color graphics. LabVIEW requires a minimum color palette setting of 256 colors. 16-bit color is recommended. The *LabVIEW Help* requires a minimum color palette setting of 256 colors with a screen resolution of 1024 × 768 pixels. 16-bit color is recommended.
All Windows Versions	Refer to the installation instruction that appear on your screen for information on the size of the LabVIEW system you are installing. LabVIEW runs on any system that supports Windows.	To use the Measurement & Automation Explorer you must have Microsoft Internet Explorer 5.0 or later installed. The *LabVIEW Tutorial* requires a sound card, a video card capable of playing `.avi` files, and a minimum color palette setting of 256 colors with a screen resolution of 1024 × 768 pixels. 16-bit color is recommended.
Windows NT	LabVIEW runs on Windows NT 4.0 Service Pack 3 or later.	To take advantage of ActiveX functionality in LabVIEW 6.0, you must have Windows NT 4.0 Service Pack 3 or later and Microsoft Internet Explorer 4.0 or later installed.
Windows ME	—	For information on using LabVIEW in Windows ME, refer to `ni.com/windowsme`.

Platform	Media and System Requirements	Important Notes
Power Macintosh	LabVIEW requires System 7.6.1 or later. You need a minimum of 32 MB of RAM and at least 100 MB of disk storage space for the minimal installation of LabVIEW or 250 MB for the full installation.	National Instruments recommends that you have at least 32 MB of RAM. You might need more memory, depending on the size of the application you design in LabVIEW and the amount of data that your application manipulates. For more accurate timing, install the Apple QuickTime extension. When you use QuickTime, timing accuracy should increase from 16.6 ms resolution to approximately 1 ms resolution. System response varies depending on background applications, other extensions, networking activity, and disk caching. The *LabVIEW Tutorial* requires a sound card, a video card capable of playing `.avi` files, and a minimum color palette setting of 256 colors with a screen resolution of 1024 × 768 pixels. 16-bit color is recommended.
All UNIX Versions	LabVIEW requires an X Window System server, such as OpenWindows, HP-VUE, CDE, or X11R6. You need a minimum of 32 MB of RAM with 32 MB of swap space storage. You need between 65 MB to 150 MB of disk storage space depending on the components you install.	LabVIEW uses a directory for storing temporary files. Some of the temporary files are large, so keep several megabytes of disk space available for this temporary directory. The default for the temporary directory is `/tmp`. You can change the temporary directory by selecting **Tools»Options**. If LabVIEW aborts unexpectedly, it might leave files behind in the temporary directory. Remove old files occasionally to avoid depleting your disk space. To save space, install only the VIs you plan to use. LabVIEW does not require a specific graphical user interface (GUI) such as Motif or OpenLook, because LabVIEW uses `Xlib` to create its own GUI.

Platform	Media and System Requirements	Important Notes
Sun	LabVIEW runs on SPARCstations with Solaris 2.5.1 or later.	—
HP-UX	LabVIEW runs on Hewlett-Packard Model 9000 Series 700 computers with HP-UX 10.20 or later.	HP workstations limit the size of a process such as LabVIEW to 64 MB. You may need to increase this setting to accommodate your LabVIEW application. Refer to the *HP-UX* section of the *LabVIEW Release Notes* under the *Installation* section for more information about changing this setting.
Linux	LabVIEW runs on Linux for Intel *x*86 processors with kernel version 2.0.*x* or later. LabVIEW runs on most major Linux distributions, such as RedHat, Calderra, SuSE, and Debian.	Requires GNU C Library Version 2 (`glibc2`, also known as `libc.so.6`). RedHat Linux 5.0 or later includes the `glibc2` run-time library.

B

Technical Support Resources

Web Support

National Instruments Web support is your first stop for help in solving installation, configuration, and application problems and questions. Online problem-solving and diagnostic resources include frequently asked questions, knowledge bases, product-specific troubleshooting wizards, manuals, drivers, software updates, and more. Web support is available through the Technical Support section of `ni.com`

NI Developer Zone

The NI Developer Zone at `ni.com/zone` is the essential resource for building measurement and automation systems. At the NI Developer Zone, you can easily access the latest example programs, system configurators, tutorials, technical news, as well as a community of developers ready to share their own techniques.

Customer Education

National Instruments provides a number of alternatives to satisfy your training needs, from self-paced tutorials, videos, and interactive CDs to instructor-led hands-on courses at locations around the world. Visit the Customer Education section of `ni.com` for online course schedules, syllabi, training centers, and class registration.

System Integration

If you have time constraints, limited in-house technical resources, or other dilemmas, you may prefer to employ consulting or system integration services. You can rely on the expertise available through our worldwide network of Alliance Program members. To find out more about our Alliance system integration solutions, visit the System Integration section of `ni.com`

Worldwide Support

National Instruments has offices located around the world to help address your support needs. You can access our branch office Web sites from the Worldwide Offices section of `ni.com`. Branch office Web sites provide up-to-date contact information, support phone numbers, e-mail addresses, and current events.

If you have searched the technical support resources on our Web site and still cannot find the answers you need, contact your local office or National Instruments corporate. Phone numbers for our worldwide offices are listed at the front of this manual.

Glossary

B

block diagram — Pictorial description or representation of a program or algorithm. The block diagram, consists of executable icons called nodes and wires that carry data between the nodes. The block diagram is the source code for the VI. The block diagram resides in the block diagram window of the VI.

C

conditional terminal — Terminal of a While Loop that contains a Boolean value that determines if the VI performs another iteration.

control — Front panel object for entering data to a VI interactively or to a subVI programmatically, such as a knob, push button, or dial.

Controls palette — Palette that contains front panel controls, indicators, and decorative objects.

D

DAQ Channel Wizard — Utility that guides you through naming and configuring DAQ analog and digital channels. Available in the Data Neighborhood of Measurement & Automation Explorer **(Windows)** or DAQ Channel Wizard **(Macintosh)**.

DAQ Solution Wizard — Utility that guides you through specifying your DAQ application, and it provides a custom DAQ solution.

data flow — Programming system that consists of executable nodes that execute only when they receive all required input data and produce output automatically when they execute. LabVIEW is a dataflow system.

E

Enable Indexing — Option that allows you to build a set of data to be released at the termination of a While Loop. With indexing disabled, a While Loop releases only the final data point generated within the loop.

execution highlighting — Debugging technique that animates VI execution to illustrate the data flow in the VI.

F

front panel — Interactive user interface of a VI. Front panel appearance imitates physical instruments, such as oscilloscopes and multimeters.

function — Built-in execution element, comparable to an operator, function, or statement in a text-based programming language.

Functions palette — Palette that contains VIs, functions, block diagram structures, and constants.

G

General Purpose Interface Bus — GPIB—synonymous with HP-IB. The standard bus used for controlling electronic instruments with a computer. Also called IEEE 488 bus because it is defined by ANSI/IEEE Standards 488-1978, 488.1-1987, and 488.2-1992.

I

IEEE 488.2 — Institute of Electrical and Electronic Engineers Standard 488.2-1987, which defines the GPIB.

indicator — Front panel object that displays output, such as a graph or LED.

L

Labeling tool — Tool to create labels and enter text into text windows.

LabVIEW — Laboratory Virtual Instrument Engineering Workbench. LabVIEW is a graphical programming language that uses icons instead of lines of text to create programs.

M

MB — Megabytes of memory. 1 MB is equal to 1,024 KB.

N

node	Program execution element. Nodes are analogous to statements, operators, functions, and subroutines in text-based programming languages. On a block diagram, nodes include functions, structures, and subVIs.

O

Operating tool	Tool to enter data into controls and operate them.

P

palette	Display of icons that represent possible options.
Positioning tool	Tool to move and resize objects.
probe	Debugging feature for checking intermediate values in a VI.
PXI	PCI eXtensions for Instrumentation. A modular, computer-based instrumentation platform.

R

RS-232	Recommended Standard 232, a serial interface bus standard.
RS-485	Recommended Standard 485, a serial interface bus standard.

S

Solutions Gallery	Option within the DAQ Solution Wizard in which you can select from numerous categories of common DAQ applications.
structure	Program control element, such as a Sequence Structure, Case structure, For Loop, or While Loop.
subpalette	A palette contained in an icon of another palette.
subVI	VI used in the block diagram of another VI. Comparable to a subroutine.

T

terminal	Object or region on a node through which data pass.
tip strip	Small yellow text banners that identify the terminal name and make it easier to identify terminals for wiring.
Tools palette	Palette that contains tools you can use to edit and debug front panel and block diagram objects.
tunnel	Data entry or exit terminal on a structure.

V

VI	*See* virtual instrument.
virtual instrument	Program in LabVIEW that models the appearance and function of a physical instrument.
VISA	Single interface library for controlling GPIB, VXI, RS-232, and other types of instruments.
VXI	VME eXtensions for Instrumentation (bus).

W

waveform chart	Indicator that plots data points at a certain rate.
While Loop	Loop structure that repeats a section of code until a condition is met.
wire	Data path between nodes.
wire branch	Section of wire that contains all the wire segments from junction to junction, terminal to junction, or terminal to terminal if there are no junctions between.
wire junction	Point where three or more wire segments join.
wire segment	Single horizontal or vertical piece of wire.
Wiring tool	Tool to define data paths between terminals.