LabWindows™/CVI™

Evaluation Guide

NATIONAL INSTRUMENTS™

September 2004 Edition
Part Number 350900C-01

Worldwide Technical Support and Product Information

ni.com

National Instruments Corporate Headquarters

11500 North Mopac Expressway Austin, Texas 78759-3504 USA Tel: 512 683 0100

Worldwide Offices

Australia 1800 300 800, Austria 43 0 662 45 79 90 0, Belgium 32 0 2 757 00 20, Brazil 55 11 3262 3599,
Canada (Calgary) 403 274 9391, Canada (Ottawa) 613 233 5949, Canada (Québec) 450 510 3055,
Canada (Toronto) 905 785 0085, Canada (Vancouver) 604 685 7530, China 86 21 6555 7838,
Czech Republic 420 224 235 774, Denmark 45 45 76 26 00, Finland 385 0 9 725 725 11,
France 33 0 1 48 14 24 24, Germany 49 0 89 741 31 30, India 91 80 51190000, Israel 972 0 3 6393737,
Italy 39 02 413091, Japan 81 3 5472 2970, Korea 82 02 3451 3400, Malaysia 603 9131 0918,
Mexico 01 800 010 0793, Netherlands 31 0 348 433 466, New Zealand 0800 553 322, Norway 47 0 66 90 76 60,
Poland 48 22 3390150, Portugal 351 210 311 210, Russia 7 095 783 68 51, Singapore 65 6226 5886,
Slovenia 386 3 425 4200, South Africa 27 0 11 805 8197, Spain 34 91 640 0085, Sweden 46 0 8 587 895 00,
Switzerland 41 56 200 51 51, Taiwan 886 2 2528 7227, Thailand 662 992 7519,
United Kingdom 44 0 1635 523545

For further support information, refer to the *Technical Support and Professional Services* appendix. To comment on the documentation, send email 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

Patents

For patents covering National Instruments products, refer to the appropriate location: **Help»Patents** in your software, the `patents.txt` file on your CD, or `ni.com/patents`.

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.

Contents

About This Manual

The *LabWindows/CVI Evaluation Guide* provides an introduction to National Instruments, virtual instrumentation, and the LabWindows™/CVI™ development environment. This document contains a step-by-step tutorial that guides you through the process of creating a LabWindows/CVI project that retrieves data from a simulated instrument. In the tutorial, you learn how to create a user interface, generate code, and add an instrument driver to the project.

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 symbol also leads you through the LabWindows/CVI Library Tree to a function panel. For example, **User Interface Library»Pop-up Panels» InstallPopup** directs you to expand the User Interface Library in the Library Tree, expand the Pop-up Panels class, and select InstallPopup.

This icon denotes an activity that you can complete to practice the concepts presented in that section.

This icon denotes a tip, which alerts you to advisory information.

This icon denotes a note, which alerts you to important information.

This icon denotes a caution, which advises you of precautions to take to avoid injury, data loss, or a system crash.

bold

Bold text denotes items that you must select or click in the software, such as menu items and dialog box options. Bold text also denotes 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.

Related Documentation

The following documents contain information that you might find helpful as you read this manual:

- *LabWindows/CVI Help*
- *Getting Started with LabWindows/CVI*
- *LabWindows/CVI Quick Reference*
- *LabWindows/CVI Bookshelf*
- *LabWindows/CVI Instrument Driver Developers Guide*

1

National Instruments, Virtual Instrumentation, and LabWindows/CVI

National Instruments is committed to providing hardware and software for engineers, scientists, and systems integrators who build, maintain, and improve test, measurement, and automation applications. With NI products, you can rapidly develop high-performing virtual instrumentation applications that seamlessly integrate into your test and measurement system.

This chapter provides an introduction to virtual instrumentation and NI LabWindows/CVI.

Virtual Instruments

NI pioneered the area of virtual instrumentation to revolutionize the test and measurement field. A virtual instrument is a combination of hardware and software in a PC with the functionality of a classic stand-alone instrument. With virtual instruments, you acquire and control data much like you do with traditional laboratory instruments. However, virtual instruments can provide more customization, scalability, and modularity than traditional instruments. With powerful NI virtual instrumentation, you get interactive data analysis, presentation, and application distribution capabilities.

LabWindows/CVI

LabWindows/CVI has become the most popular software development environment for C language programmers who develop virtual instrumentation. LabWindows/CVI is a proven ANSI C development environment and compiler with built-in libraries for acquisition, analysis, and presentation. LabWindows/CVI combines the power of ANSI C and the flexibility of Microsoft Windows with easy-to-use tools for building virtual instrumentation systems.

Using LabWindows/CVI to Create Virtual Instruments

Creating virtual instruments is a three-step process when you use NI hardware with LabWindows/CVI.

Step 1—Acquire

Acquire raw data through a hardware interface. LabWindows/CVI provides a high-level interface for acquiring data—whether you use a GPIB or serial instrument, a plug-in data acquisition board, PXI hardware, modular instruments, an image acquisition device, or a motion control device.

Step 2—Analyze

Analyze raw data to make it meaningful to you and your users. Rather than collecting a list of raw signals and later analyzing them with a separate application, use LabWindows/CVI analysis tools to obtain fast and meaningful results. You can employ a variety of signal processing and data analysis tools—including statistical analysis, curve fitting, spectral analysis, and visualization functions—for research and development, engineering and validation, or manufacturing and service applications.

Step 3—Present

Present results in an intuitive way and allow users to interact with the instrumentation system through a graphical user interface (GUI)—the interactive user interface of a virtual instrument. LabWindows/CVI delivers ready-to-use, measurement-specific user interface components with which you can create professional measurement applications. Use three-dimensional, instrument-style controls and indicators—including buttons, knobs, thermometers, tanks, LEDs, slides, numerical displays, strip charts, graphs, trees, and tables—to build panels that represent traditional test and measurement instruments.

Measurement Libraries and Components

Because LabWindows/CVI is a programming environment for measurement applications, it includes a large set of run-time libraries for instrument control, data acquisition, analysis, and user interface design. LabWindows/CVI provides libraries for using the following technologies:

- General Purpose Interface Bus (GPIB)
- RS-232
- Virtual Instrument Software Architecture (VISA)
- Data acquisition
- Data analysis
- Data formatting

- Transport Control Protocol (TCP)
- Windows Dynamic Data Exchange (DDE) communication
- Internet communication
- DIAdem connectivity
- File I/O
- ANSI C

LabWindows/CVI also contains many features that make developing measurement applications much easier than developing in traditional C environments. LabWindows/CVI provides the following tools:

- A development environment that manages projects and source code with complete editing, debugging, and user protection features
- A graphical User Interface Editor, CodeBuilder wizard, and library for building, displaying, and controlling a graphical user interface
- A wizard and library for controlling ActiveX servers
- A wizard and library for creating IVI instrument drivers, which are highly structured VXI*plug&play*-compatible instrument drivers that use an attribute model to enable advanced features, such as state-caching, simulation, and compatibility with generic instrument classes
- An interactive tool for creating and editing NI-DAQmx tasks
- An interactive tool for creating instrument control tasks
- Multithreaded application development and debugging capabilities
- A command line compiler

More Features to Meet Your Needs

NI remains committed to improving and extending the features of LabWindows/CVI. LabWindows/CVI 7.1 incorporates new tools and features to improve virtual instrumentation development and execution.

- **Remote Debugging Capabilities**—With the new remote debugging capabilities, the LabWindows/CVI debugger and the program you are debugging can be on different computers. You can use this remote debugging functionality to debug LabWindows/CVI-built DLLs on a remote LabVIEW Real-Time system.
- **Expanded User Interface Library**—In addition to the existing controls, the User Interface Library includes a new digital graph control and a new splitter control.

- **New Function Libraries**—LabWindows/CVI now includes the Internet Library and the DIAdem Connectivity Library for increased Internet functionality and enhanced offline analysis support.
- **Enhanced Workspace Integration**—You now can edit and operate function panels from within the Window Confinement Region of the Workspace window and view the Variable and Watch windows from within the Debugging Region of the Workspace window.

Related Software Packages

NI offers additional packages for targeted applications with LabWindows/CVI. Visit the Product Catalog at `ni.com` for more information about the following software packages:

- **NI Developer Suite**—The NI Developer Suite Professional Test Edition includes the following components:
  - LabWindows/CVI
  - LabVIEW for graphically developing test, measurement, and automation applications
  - Measurement Studio for developing test, measurement, and automation applications in Visual C++, Visual Basic .NET, and Visual C#
  - TestStand for managing test execution, sequencing, collecting data, and generating reports
  - A comprehensive set of LabWindows/CVI, LabVIEW, and Measurement Studio add-on tools for database communication, signal processing, and code distribution
  - A one-year subscription to Standard Service, which provides quarterly software updates automatically and gives you one-on-one access to NI Applications Engineers for your technical support questions
- **Measurement Studio**—Measurement Studio is an integrated suite of native test, measurement, and control tools and class libraries for Microsoft Visual Studio .NET. Measurement Studio dramatically reduces application development time with wizards, simplified data networking, and .NET user interface controls. The LabWindows/CVI Full Development System includes Measurement Studio.
- **TestStand**—TestStand is a ready-to-run test executive for organizing, controlling, and executing automated prototype, validation, or manufacturing test systems. TestStand is completely customizable, so you can modify and enhance it to match your specific needs. TestStand comes complete with integrated LabWindows/CVI tools, including the flexible LabWindows/CVI Adapter.

- **Vision Development Module**—The Vision Development Module includes IMAQ Vision, a library of vision functions, and IMAQ Vision Builder, an interactive environment for prototyping vision applications. Use vision and image processing software to build machine vision and scientific imaging applications.
- **IVI Driver Toolset**—The IVI Driver Toolset provides tools to build hardware and protocol (GPIB, VXI, and RS-232) independent test programs. This package provides a full library of IVI class and instrument-specific drivers, advanced instrument simulation capabilities, advanced debugging capabilities, and a set of executable soft front panels.

Tip Visit `ni.com/idnet` to access the online Instrument Driver Network—the industry's largest source of instrument drivers, featuring drivers for over 2,500 instruments from over 150 vendors. Here you can download and submit drivers for controlling instruments from LabWindows/CVI. Browse through instrument drivers by instrument type, manufacturer, or development language.

- **DIAdem**—DIAdem is an interactive tool for mathematical and visual data analysis, report generation, task automation, and data management. DIAdem imports data from files and industry-standard databases and can optimally handle datasets with more than one billion parts.
- **LabWindows/CVI Enterprise Connectivity Toolset**—The LabWindows/CVI Enterprise Connectivity Toolset provides enterprise connectivity tools that help you track progress from research and development to the production, testing, and servicing of products. This toolset includes the LabWindows/CVI SQL Toolkit for Structured Query Language (SQL) database operations and the LabWindows/CVI SPC Toolkit for statistical process control (SPC) quality control.
- **LabWindows/CVI Signal Processing Toolset**—The LabWindows/CVI Signal Processing Toolset provides the following four toolkits for digital filter design, joint time-frequency analysis, wavelet and filter bank design, and super-resolution, model-based spectral analysis:
  - **Digital Filter Design Toolkit**—Use this toolkit for signal conditioning, control system design, and digital signal processing.
  - **Joint Time-Frequency Analysis Toolkit**—Use this toolkit to simultaneously examine the time and frequency domain representations of a signal.
  - **Wavelet & Filter Bank Design Toolkit**—Use this toolkit for wavelet design and implementation.
  - **Super-Resolution Spectral Analysis Toolkit**—Use this toolkit for spectral analysis. This toolkit provides a model-based alternative to the Fast Fourier Transform (FFT) method of spectral analysis.
- **PID Control Toolset**—The PID Control Toolset adds sophisticated control algorithms to LabWindows/CVI. With this package, you can build data acquisition and control systems for your own control application.

2

Getting Started with LabWindows/CVI

This chapter introduces the LabWindows/CVI Evaluation Package and provides system requirements and installation information.

The LabWindows/CVI CD includes both the LabWindows/CVI Evaluation Package and the licensed version of LabWindows/CVI. When you launch LabWindows/CVI during the evaluation period and before you have purchased a valid license, the dialog box shown in Figure 2-1 appears.

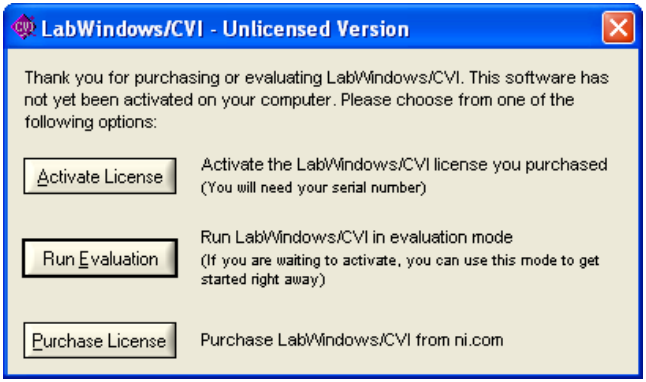


Figure 2-1. Unlicensed Version Dialog Box

Click **Run Evaluation** to run the software in evaluation mode.

To purchase LabWindows/CVI, click **Purchase License**. You also can contact an NI representative at 1-800-433-3488 or your local sales office (`ni.com/global`) to purchase LabWindows/CVI. To activate LabWindows/CVI after you have purchased a license, click **Activate License** and follow the activation instructions.

About this Evaluation Package

The LabWindows/CVI Full Development System includes data acquisition, instrument control, user interface, advanced analysis, digital signal processing, and networking tools. With LabWindows/CVI, you can acquire data with NI data acquisition (DAQ) boards; control serial, GPIB, PXI, USB, Ethernet, and VXI instruments and controllers; analyze the data you acquired from a device; develop custom user interfaces to present data; and share live data between different applications over the Internet.

The LabWindows/CVI Evaluation Package contains the LabWindows/CVI Full Development System with the following restrictions:

- 30-day expiration
- 10-minute run-time timeout
- No external compiler support
- Disabled Create Distribution Kit feature
- Disabled Create Object File feature
- Disabled generation of import libraries from header files
- No IVI Wizard support (disabled Create IVI Instrument Driver feature)
- No Instrument Driver Only support
- No LabVIEW Real-Time support

Caution After you activate LabWindows/CVI, you must rebuild any projects you created in evaluation mode. If you do not rebuild a project you created in evaluation mode, the project will continue to have a 10-minute run-time timeout.

Minimum System Requirements

To run LabWindows/CVI, you must have the following items:

- Personal computer using a Pentium 600 or higher microprocessor
- Windows 2000/NT Service Pack 6/XP
- 800 × 600 resolution (or higher) video adapter
- Minimum of 128 MB of RAM, 256 MB recommended
- 150 MB free hard disk space for full installation
- Microsoft-compatible mouse
- Microsoft Internet Explorer 5.0 or later

Installation Instructions

Complete the following steps to install the LabWindows/CVI Evaluation package.

1. Insert the CD in the CD drive. If the CD does not run automatically, open Windows Explorer, right-click the CD drive icon, and select **AutoPlay**.
2. Select **Install LabWindows/CVI** on the National Instruments LabWindows/CVI screen. Continue to follow the instructions on the screen.
3. Install driver software if you plan to use LabWindows/CVI with NI hardware.
4. Install the NI hardware. Refer to the hardware installation guide for installation information.
5. Configure the NI hardware with NI Measurement & Automation Explorer (MAX).

Exploring LabWindows/CVI

If you are already programming with C, LabWindows/CVI complements existing efforts and streamlines future development. Because LabWindows/CVI is built on an open software architecture, you can reuse existing programs within the LabWindows/CVI environment. You can incorporate standard ANSI C code, object files, and DLLs into your applications.

As you develop applications in LabWindows/CVI, you work within the Workspace window. The Workspace window contains five areas.

- **Project Tree**—Contains files for each project in the workspace
- **Library Tree**—Provides access to function panels for the functions in LabWindows/CVI libraries and loaded instruments
- **Window Confinement Region**—Contains open Source windows, User Interface windows, Function Tree Editor windows, Function Panel Editor windows, and function panels
- **Debugging Region**—Contains the Variables, Watch, and Memory windows
- **Output Region**—Contains various error, output, and results windows

You can complete this activity in 5 minutes.

1. Launch LabWindows/CVI. When you open LabWindows/CVI, you see the Workspace window.

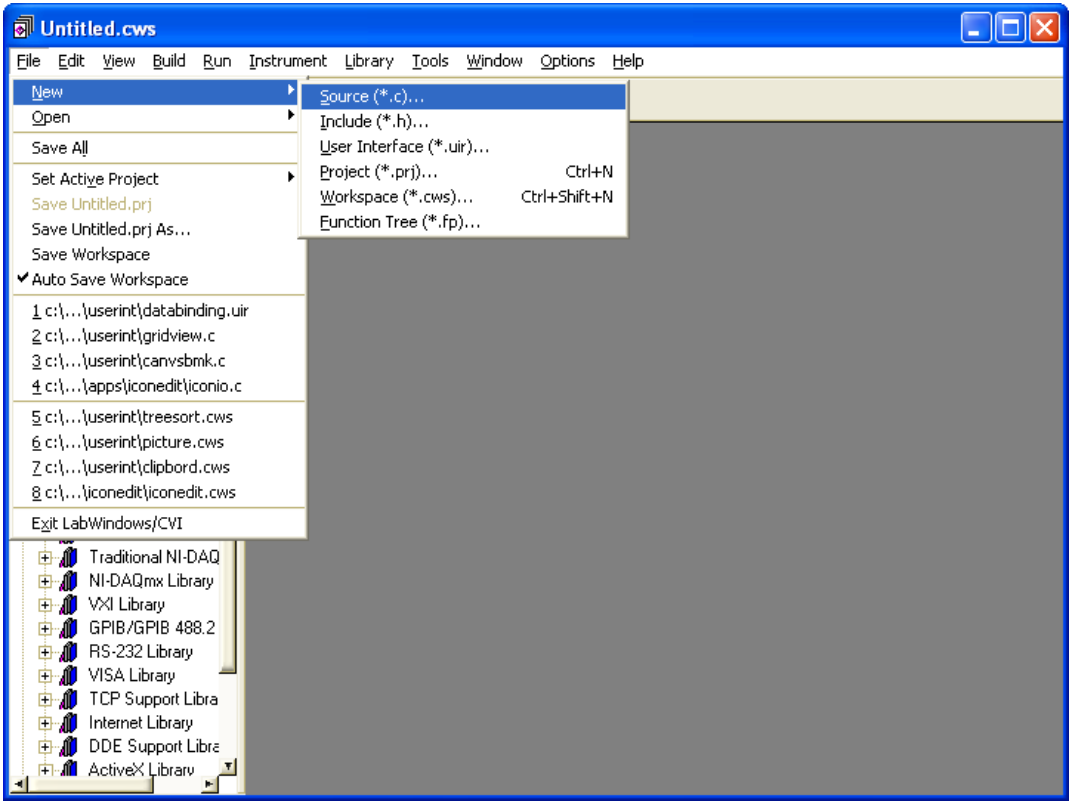


Figure 2-2. LabWindows/CVI Workspace Window

2. Explore the main windows in LabWindows/CVI. To open the Source window, select **File»New»Source (*.c)**. To open the User Interface Editor, select **File»New»User Interface (*.uir)**. To open the Function Tree Editor, select **File»New»Function Tree (*.fp)**.

For more information about the LabWindows/CVI environment, refer to *Using LabWindows/CVI* from the **Contents** of the *LabWindows/CVI Help*.

Context-Sensitive Help

NI designed LabWindows/CVI to deliver the appropriate information when you need it. In addition to traditional help files, you can find context-sensitive help throughout LabWindows/CVI. Use the following resources to learn more about LabWindows/CVI:

- **Help file**—Select **Help»Contents** in LabWindows/CVI to access the *LabWindows/CVI Help*.
- **Menu help**—Press <F1> on menus or right-click menu items and select **Menu Help** to learn more about the options and features available in that menu.
- **Dialog box help**—Press <F1> or click the **Help** button in dialog boxes to access help about the options and features available in that dialog box.
- **Source window keyword help**—Press <F1> on a function name in the Source window to access help for that function. If you use the **Edit»Show Prototype** option to view the function prototype tooltip of the current function in the Source window, you can click the question mark button to the left of the function prototype or press <F1> to view help for the currently highlighted function or parameter.
- **Function panel help**—Right-click the panel or any parameter for help about that function or parameter.
- **User Interface Editor help**—Right-click any control and select **Control Help** or select a control and press <F1> to open help for a GUI control. You also can access help in Edit Control dialog boxes. Click the question mark button in the Edit Control dialog box title bar and then select a control to view help.

3

Creating a LabWindows/CVI Project

This chapter guides you through the process of creating a project that retrieves data from a simulated instrument. You will learn how to create a user interface, generate code, and add an instrument driver to the project.

Creating a New LabWindows/CVI Project

You can complete this activity in 2 minutes.

To create a project, complete the following steps:

1. Select **File»New»Project (*.prj)**.
2. If prompted, click **Yes** to unload the current project and select **Create Project in New Workspace** in the New Project Options dialog box.
3. Save the project as `AcquireWaveform.prj`.

Creating a Graphical User Interface

LabWindows/CVI provides a collection of highly configurable, instrument-style user interface controls and indicators so that application users can interact with and monitor the measurement or control system. Choose from the following controls to create a user interface:

- Knobs, meters, gauges, dials
- Binary switches, LEDs
- Slides, tanks, thermometers
- Real-time 2D graphs, strip charts, digital graphs
- Trees, tables
- ActiveX controls
- Timers

User Interface Editor

The User Interface Editor is an interactive drag-and-drop environment you can use to design GUIs. You can select controls—buttons, toggles, slides, graphs, LEDs, and more—from the **Create** menu and position them on the GUI. You then can use dialog boxes to set control attributes, such as labels, colors, and hot key connections. In the following sections, you build a GUI that acquires and displays a waveform.

You can complete the following activity in 10 minutes.

Figure 3-1 shows the completed GUI.

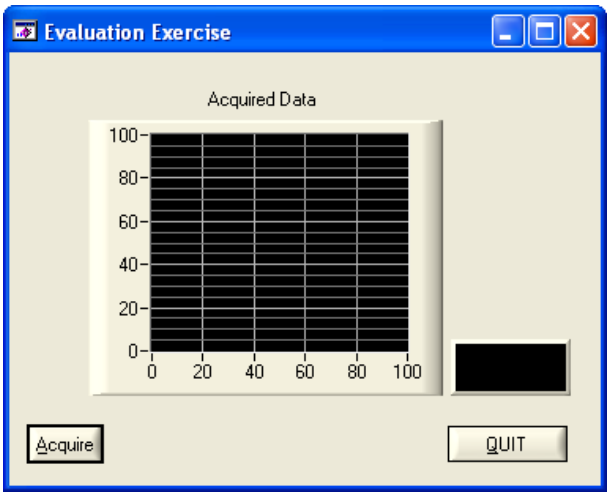


Figure 3-1. Evaluation Exercise GUI

Creating a New User Interface Resource (.uir) File

1. Select **File»New»User Interface (*.uir)**. The User Interface Editor opens an untitled file with an empty panel.
2. Double-click the panel. Enter `Evaluation Exercise` as the **Panel Title** and click **OK**.
3. Save the `.uir` file as `AcquireWaveform.uir`.

Note When you save a `.uir` file, LabWindows/CVI automatically creates a header (`.h`) file with the same base name as the `.uir` file. LabWindows/CVI stores this header file in the same directory as the `.uir` file.

Adding Command Buttons

1. Select **Create»Command Button»Square Command Button**. LabWindows/CVI places a button labeled **OK** on the panel. Position the button in the lower left corner of the panel.
2. To edit the button attributes, double-click the button. In the Edit Command Button dialog box, enter `ACQUIRE` as the **Constant Name** for the command button.
3. Enter `AcquireData` as the **Callback Function** for the command button. The program calls the callback function when a user clicks the button.
4. Enter `__Acquire` as the **Label** for the command button. Because there is a double underscore before the `A` in the **Label** field, `A` is underlined in the label. You can select this command button by pressing <Alt-A>.
5. Make sure the Edit Command Button dialog box matches the one shown in Figure 3-2 and then click **OK**.

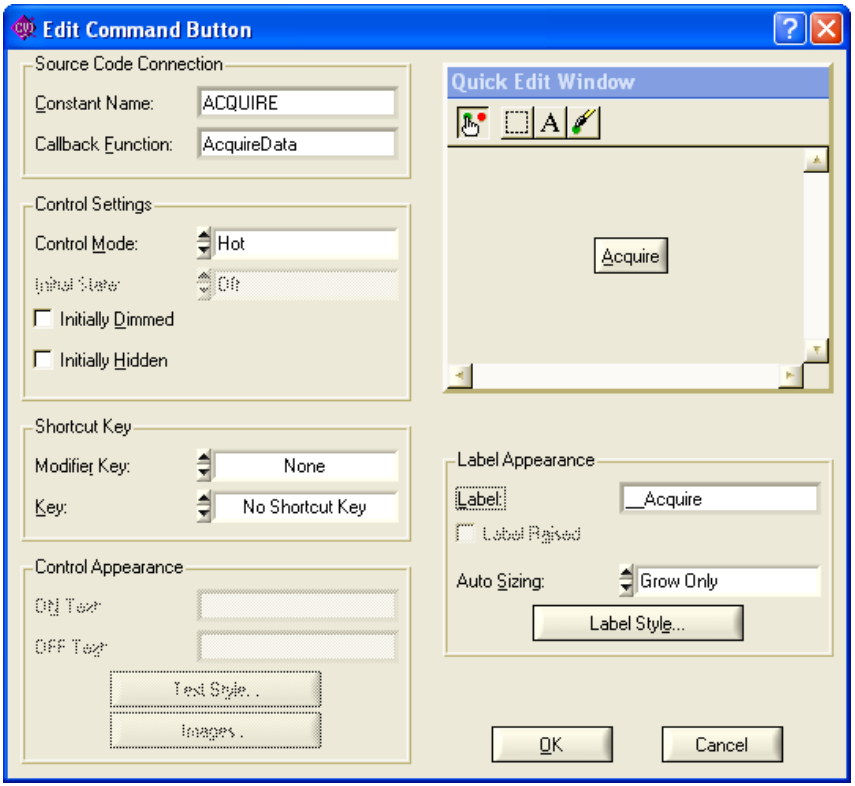


Figure 3-2. Edit Command Button Dialog Box

6. Select **Create»Custom Controls»Quit Button**. Custom controls are controls that you can configure and save between LabWindows/CVI sessions.
7. Position the **QUIT** button in the lower right corner of the panel.
8. Double-click the button to open the Edit Command Button dialog box. Notice the **Constant Name** is `QUITBUTTON` and the **Callback Function** is `QuitCallback`. Do not change these settings. Click **OK** to close the dialog box.

Adding a Graph Control

1. Select **Create»Graph»Graph**.
2. Position the graph control between the command buttons.
3. Double-click the graph control to open the Edit Graph dialog box.
4. Enter `WAVEFORM` as the **Constant Name** for the graph control.

Note Because this graph does not require an action from the user interface, you do not need to assign a callback function to this control. Callback functions are necessary only when the operation of the control initiates an action or acts as an input.

5. Enter `Acquired Data` as the **Label** for the graph control.
6. Click **OK** to close the dialog box.
7. Save `AcquireWaveform.uir`.

Generating Program Code with CodeBuilder

LabWindows/CVI offers an innovative code generation tool called CodeBuilder. CodeBuilder uses information that you specify in control and event dialog boxes to create a code skeleton. The code skeleton includes the `main` function, event callbacks, and an application shutdown callback.

LabWindows/CVI provides event-driven programming through callback functions. You can associate a specific callback function with a specific user interface control in the Edit dialog box for the control, as you did in the previous section. As you set the properties for the control, think about what events you want the control to recognize—a single mouse click, double-click, or keypress—and specify the callback function to execute when those events occur.

You can specify default events for both the panel and controls on the panel. To enable events, select **Code»Preferences»Default Panel Events** and **Default Control Events**. When you generate code, CodeBuilder creates a case statement for each event you enable—you just fill in the code to handle that event.

At this point, you have created a GUI with user interface objects to acquire a waveform, display the waveform, and terminate program execution. Now, use CodeBuilder to create skeleton source code.

You can complete this activity in 5 minutes.

1. Specify the type of events to which the program will respond. Open `AcquireWaveform.uir` if it is not already open. Select **Code»Preferences» Default Control Events**.
2. In the Control Callback Events dialog box, select the events that a control can generate. In this project, the controls respond to one type of event: a commit event (left-click or <Enter>). Select only `EVENT_COMMIT` and then click **OK**.

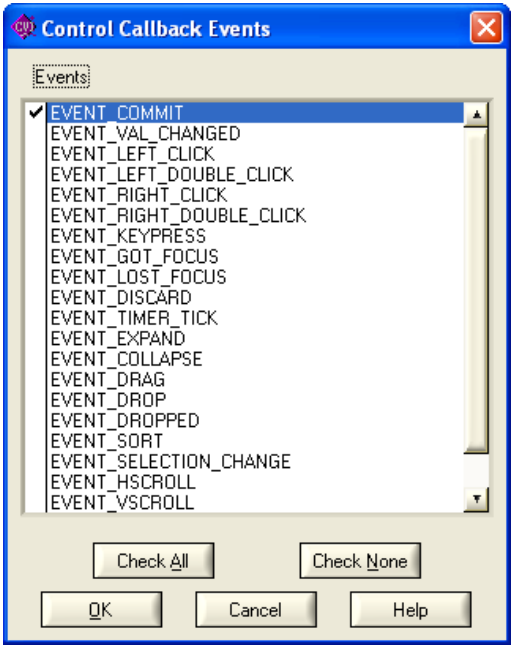


Figure 3-3. Control Callback Events Dialog Box

Tip For more information about events, refer to *Events Overview* from the **Index** of the *LabWindows/CVI Help*.

3. Select **Code»Generate»All Code** to open the Generate All Code dialog box.
4. Select **Add to Current Project** in the Target Files section.

5. In general, you must set which panels you want to load and display at program startup. Because there is only one panel in this example, verify that `PANEL` is selected as the panel to load and display at startup and that the **Panel Variable Name** is `panelHandle`.
6. Verify that the `QuitCallback` function is selected as the QuitUserInterface callback. LabWindows/CVI inserts the `QuitUserInterface` function into the `QuitCallback` callback function.
7. Click **OK**. CodeBuilder builds the source code for the program and adds the file to the project. LabWindows/CVI opens a Source window, `AcquireWaveform.c`, which contains the new source code.

Reviewing the Source Code

main Function

The `main` function is the first component you need when you build any application. The `main` function for this project consists of the lines of code shown in Figure 3-4.

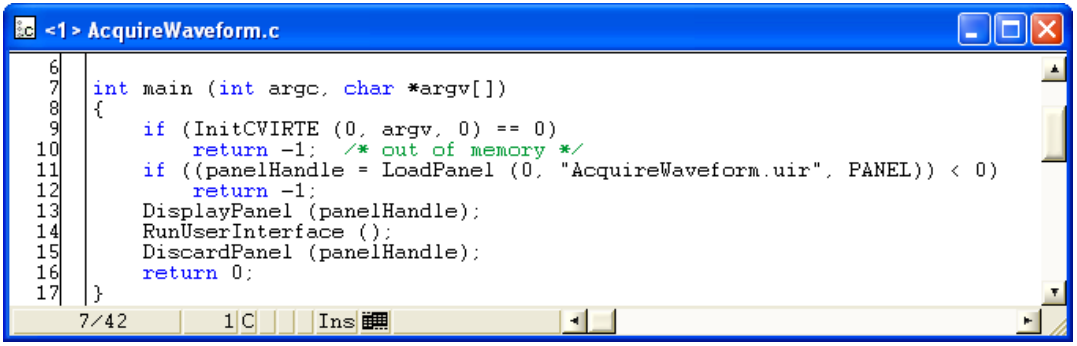


Figure 3-4. main Function

The functions within `main` perform the following actions:

- `LoadPanel` loads the panel from the `.uir` file into memory.
- `DisplayPanel` displays the panel on the screen.
- `RunUserInterface` allows LabWindows/CVI to send events from the user interface to the C program.
- `DiscardPanel` removes the panel from memory and clears it from the screen if visible.

AcquireData Function

The `AcquireData` function automatically executes when a user clicks the **Acquire** button. Currently, the `AcquireData` function contains no functions to execute. In the following sections, you will add data acquisition functions within the `AcquireData` function.

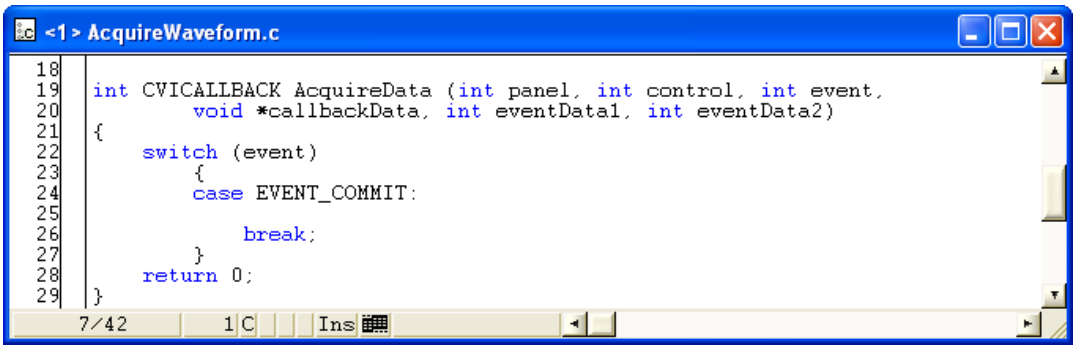


Figure 3-5. AcquireData Function

QuitCallback Function

The `QuitCallback` function automatically executes when a user clicks the **QUIT** button. This function terminates event processing and stops program execution.

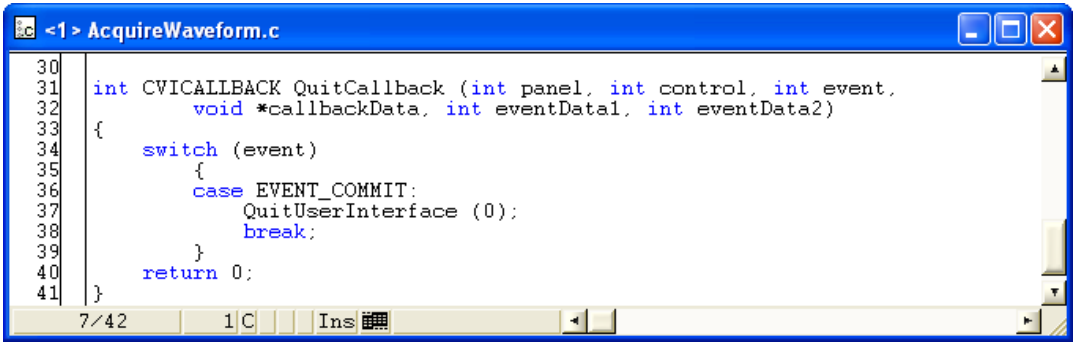


Figure 3-6. QuitCallback Function

Running the Project

To run the project, select **Run»Debug AcquireWaveform_dbg.exe**. Currently, only the **QUIT** button responds to events.

Adding an Instrument Driver to the Project

An instrument driver is a set of functions that controls an instrument or a group of related instruments. The high-level functions in an instrument driver incorporate many low-level operations, including GPIB, VXI, or RS-232 read and write operations; data conversion; and scaling. The sample module you use in this project communicates with a simulated instrument and shows you how to use an instrument driver to acquire a waveform from an oscilloscope.

To be complete, the program must read an array of simulated data from an instrument driver and plot the array on the graph. You must modify the `AcquireData` function in the `AcquireWaveform.c` source file as described in the following steps. Then, when a user clicks the **Acquire** button, the program reads the data from the instrument and plots the data on the graph.

Loading the Instrument Driver

1. Select **Edit»Add Files to Project»Instrument (*.fp)** in the Workspace window.
2. Select the `tutorial\scope.fp` file. Click **Add** and then click **OK** to add the driver to the project.
3. In the Workspace window, open `AcquireWaveform.c`.

4. Place the cursor on the blank line immediately after the `EVENT_COMMIT:` statement in the `AcquireData` function, as shown in Figure 3-7.

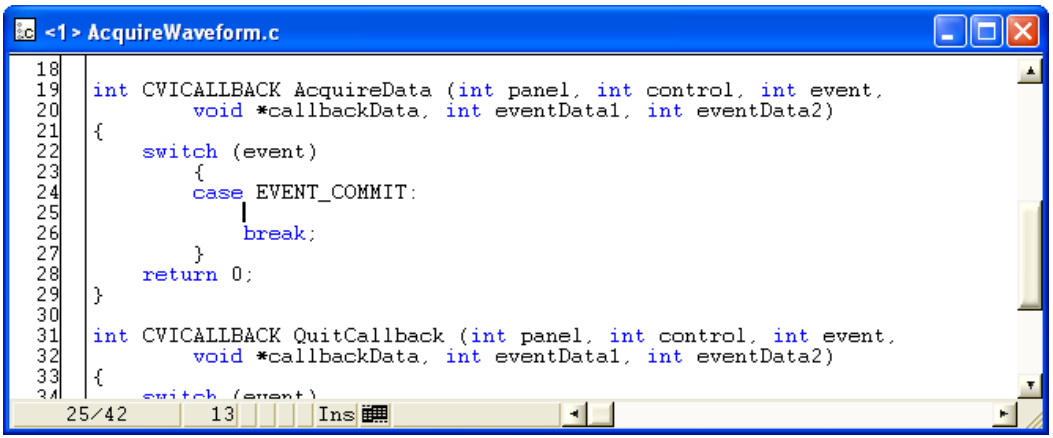


Figure 3-7. Cursor Position in AcquireWaveform.c

Initializing the Instrument

1. Expand **Instruments»Sample Oscilloscope** in the Library Tree. The sample oscilloscope driver contains four functions for communicating with a scope, as shown in Figure 3-8.

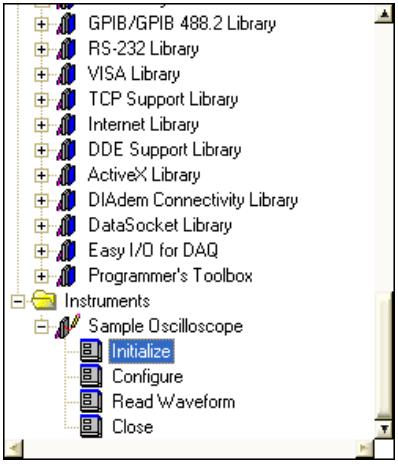


Figure 3-8. Sample Oscilloscope Instrument Driver

2. Double-click **Initialize** to open the Initialize function panel, as shown in Figure 3-9.

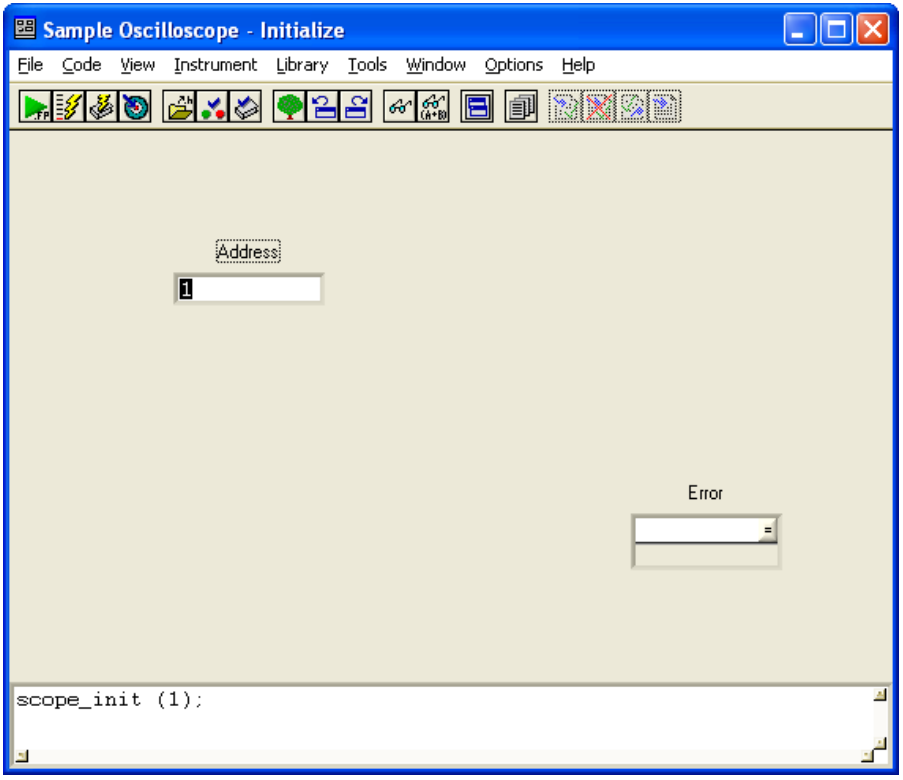


Figure 3-9. Initialize Function Panel

Tip You can use function panels to interact with LabWindows/CVI libraries and loaded instrument drivers. Function panels are graphical representations of LabWindows/CVI functions and their parameters. Use function panels to interactively test function calls and paste them into a program. When you enter values in the function panel controls, LabWindows/CVI builds the function call at the bottom of the panel.

3. Enter `1` in the **Address** control.
4. Enter `err` in the **Error** control.
5. Select **Code»Declare Variable** to declare the `err` variable.
6. In the Declare Variable dialog box, enable the **Execute declaration in Interactive Window** and **Add declaration to top of target file "AcquireWaveform.c"** options. Click **OK**.

7. Select **Code»Run Function Panel**. If LabWindows/CVI does not detect any errors during execution, the **Error** control returns 0.
8. Select **Code»Insert Function Call** to copy the generated code into the Source window.

LabWindows/CVI places the function call to initialize the instrument driver in the source code below the `EVENT_COMMIT:` code, as follows:

```
err = scope_init (1);
```

Reading Data from the Instrument

Perhaps the most important function of an instrument driver is to read data from an instrument and convert the raw data into a format the program can use. For example, a digital oscilloscope returns a waveform as a string of comma-separated ASCII numbers. The instrument driver parses the string, scales the data to volts, and places the data into an array in memory.

1. Select **Instruments»Sample Oscilloscope»Read Waveform** in the Library Tree.

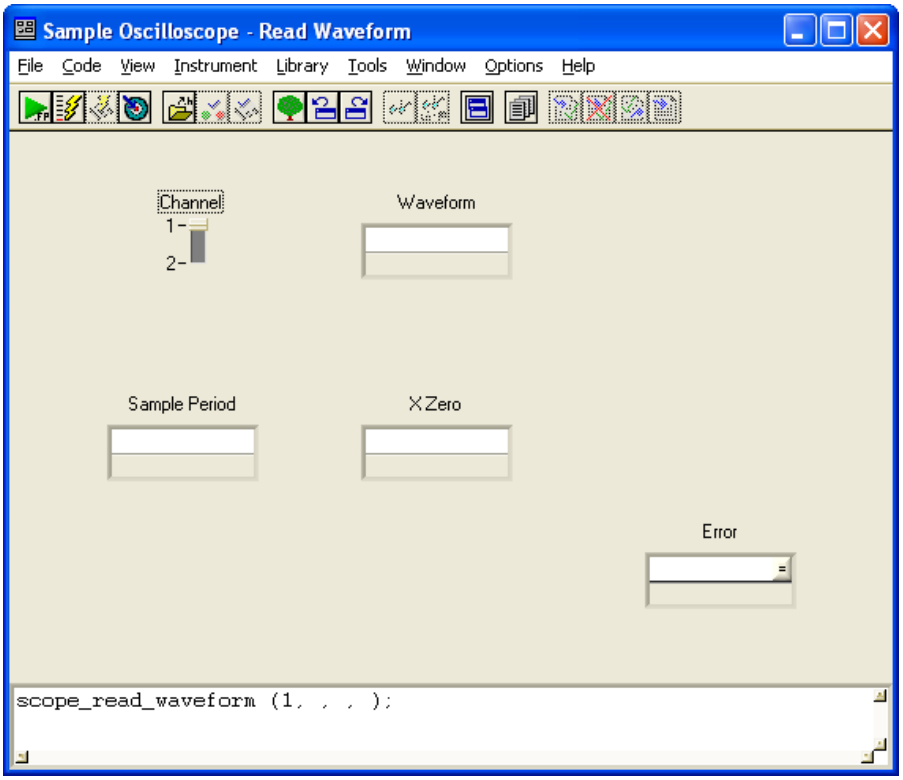


Figure 3-10. Read Waveform Function Panel

2. In the Read Waveform function panel, shown in Figure 3-10, set the **Channel** control to 2. Channel 1 is sine wave data, and Channel 2 is random data.
3. Enter `datapoints` in the **Waveform** control.
4. Select **Code»Declare Variable** to declare the `datapoints` variable in memory. In the Declare Variable dialog box, enter `100` in the **Number of Elements** control.
5. Verify that the **Execute declaration in Interactive Window** and **Add declaration to top of target file "AcquireWaveform.c"** options are selected. Click **OK** to declare the `datapoints` array.
6. Enter `delta_t` in the **Sample Period** control.
7. Select **Code»Declare Variable**. Verify that the **Execute declaration in Interactive Window** and **Add declaration to top of target file "AcquireWaveform.c"** options are selected and click **OK**.

   Notice that LabWindows/CVI adds a leading ampersand, &, to the variable name when you declare the variable. LabWindows/CVI automatically prefixes scalar output variables with an ampersand.
8. Enter `x_zero` in the **X Zero** control.
9. Select **Code»Declare Variable**. Verify that the **Execute declaration in Interactive Window** and **Add declaration to top of target file "AcquireWaveform.c"** options are selected and click **OK**.
10. Enter `err` in the **Error** control.
11. Select **File»Save All**. Select **Code»Run Function Panel** to execute the function panel. If the `scope_read_waveform` function executes correctly, the **Error** control returns 0. After the function executes, a row of boxes in the **Waveform** control signifies that the data is in the waveform array.
12. (Optional) You can double-click the row of boxes to display two windows of data returned by the function call—the Variables window shows variable values, and the Array Display window shows the waveform data collected in the array. If you choose to display these windows, close both of them before continuing.
13. From the Read Waveform function panel, select **Code»Insert Function Call** to copy the generated code into the Source window. LabWindows/CVI places the following line of code within the `AcquireData` callback function:

```
err = scope_read_waveform (2, datapoints, &delta_t, &x_zero);
```

Displaying the Waveform on a Graph Control

At this point, you have completed the steps necessary to acquire data. You now need to plot the acquired data array to the graph control on the user interface using the `PlotY` function.

In this exercise, you will use the LabWindows/CVI source code completion options to write code. With the source code completions options, you can view functions, variables, prototypes, constant values, and help while you type in the Source window.

1. Type `PlotY` in the blank line below the `scope_read_waveform` function call and press <Ctrl-Shift-Space> to display the function prototype tooltip. Notice the question mark button to the left of the function prototype. When you click this button or press <F1>, LabWindows/CVI displays help for the currently highlighted function or parameter.
2. Enter the following values for the first four function parameters. LabWindows/CVI highlights the next parameter in the function prototype after you enter a parameter value.

   PanelHandle: `panelHandle`
   ControlID: `PANEL_WAVEFORM`
   YArray: `datapoints`
   NumberofPoints: `100`
3. When you reach the **YDataType** parameter, LabWindows/CVI displays a **…** button to the right of the parameter name. Click this button or press <Ctrl-Shift-Enter> to display a list of constant values for the **YDataType** parameter. Select **VAL_DOUBLE** from the list to insert this value into the code.
4. Enter the following values for the remaining parameters. Select from the list of constant values if LabWindows/CVI displays a **…** button to the right of the parameter name.

   PlotStyle: **VAL_THIN_LINE**
   PointStyle: **VAL_EMPTY_SQUARE**
   LineStyle: **VAL_SOLID**
   PointFrequency: `1`
   Color: **VAL_RED**

Deleting Existing Graph Plots

If the user clicks the **Acquire** button more than once, the data from previous acquisitions remains on the graph with the new data plot. Modify the `AcquireData` callback to include the `DeleteGraphPlot` function, which deletes one or all plots from a graph control.

1. Add a blank line immediately following the `EVENT_COMMIT:` statement in the `AcquireData` callback, type `Delete`, and press <Ctrl-Space> to view a list of potential matches for the function you are typing. Select **DeleteGraphPlot** from the list.

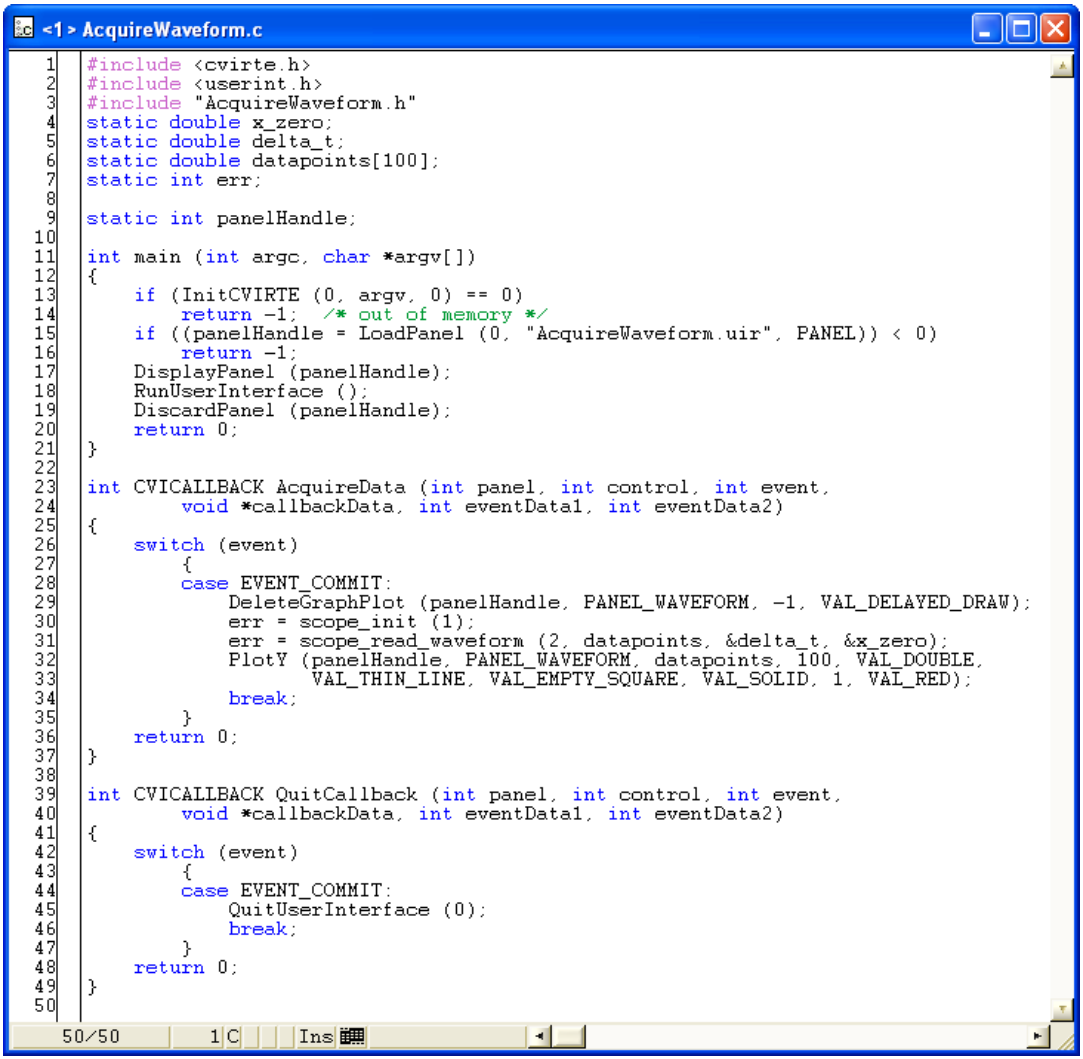
Note You also can use the Show Completions option to view potential matches for variables, structure members, enum members, and macros after you compile the source code.

2. Display the function prototype tooltip if it is not already visible. Enter the following values for the function parameters:

PanelHandle:	`panelHandle`
ControlID:	`PANEL_WAVEFORM`
PlotHandle:	`-1`
Refresh:	**Delayed Draw**

Note If you specify `-1` for the **plotHandle** parameter, LabWindows/CVI deletes all the plots for the specified graph control when you call `DeleteGraphPlot`.

3. Save `AcquireWaveform.c`.
4. Verify that the completed source code matches the code block shown in Figure 3-11.




```
#include <cvirte.h>
#include <userint.h>
#include "AcquireWaveform.h"
static double x_zero;
static double delta_t;
static double datapoints[100];
static int err;

static int panelHandle;

int main (int argc, char *argv[])
{
    if (InitCVIRTE (0, argv, 0) == 0)
        return -1;  /* out of memory */
    if ((panelHandle = LoadPanel (0, "AcquireWaveform.uir", PANEL)) < 0)
        return -1;
    DisplayPanel (panelHandle);
    RunUserInterface ();
    DiscardPanel (panelHandle);
    return 0;
}

int CVICALLBACK AcquireData (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{
    switch (event)
        {
        case EVENT_COMMIT:
            DeleteGraphPlot (panelHandle, PANEL_WAVEFORM, -1, VAL_DELAYED_DRAW);
            err = scope_init (1);
            err = scope_read_waveform (2, datapoints, &delta_t, &x_zero);
            PlotY (panelHandle, PANEL_WAVEFORM, datapoints, 100, VAL_DOUBLE,
                   VAL_THIN_LINE, VAL_EMPTY_SQUARE, VAL_SOLID, 1, VAL_RED);
            break;
        }
    return 0;
}

int CVICALLBACK QuitCallback (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{
    switch (event)
        {
        case EVENT_COMMIT:
            QuitUserInterface (0);
            break;
        }
    return 0;
}
```

Figure 3-11. Completed Source File

Running the Completed Project

You now have a completed project, saved as `AcquireWaveform.prj`. Select **Run»Debug AcquireWaveform_dbg.exe** to execute the code. In the dialog box that prompts you that the action will clear your Interactive Execution declarations, select **OK**. During the compile process, LabWindows/CVI recognizes that the program is missing the `scope.h` header file statement and offers to insert it into the source code. Click **Yes** to add this include file to the program.

Now that you have completed a LabWindows/CVI project, which is shown in Figure 3-12, look through the rest of the libraries for functionality not covered in this manual. Also, explore the samples included in the evaluation version of LabWindows/CVI. You can find a number of example programs in the `samples` folder.

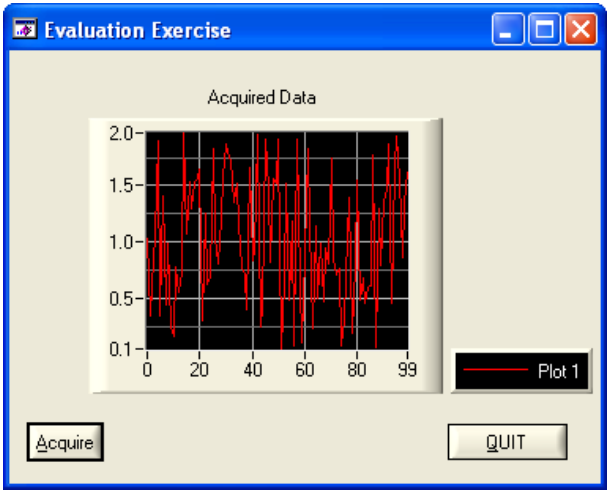


Figure 3-12. Completed Project

4

Where to Go from Here

LabWindows/CVI is yet another long-standing commitment by NI to provide tools that simplify instrumentation, data acquisition, and control system development. When you choose LabWindows/CVI, you join thousands of scientists and engineers who take advantage of the power of this versatile tool for quick and easy test and measurement development.

Visit the LabWindows/CVI Web site at `ni.com/cvi` and NI Developer Zone at `ni.com/zone` for the most up-to-date information. You also can visit the Developer Exchange at `ni.com/exchange` to participate in discussion forums and exchange code with other LabWindows/CVI users around the world.

Tutorials and Documentation

Launch the *LabWindows/CVI Bookshelf* by selecting **Start»Programs»National Instruments»LabWindows CVI»LabWindows CVI Bookshelf** or by selecting **Help»LabWindows/CVI Bookshelf** within the LabWindows/CVI environment. The bookshelf lists all currently available LabWindows/CVI documentation. The bookshelf also describes and links to LabWindows/CVI manuals, application notes, and white papers in PDF format.

Note Use Adobe Acrobat Reader 5.0.5 or later with Search and Accessibility to search the *LabWindows/CVI Bookshelf*. You can download a free version of the Reader from `www.acrobat.com`.

Read the *Getting Started with LabWindows/CVI* manual for a complete tutorial, additional exercises, and quick reference information about configuring various types of hardware for use with LabWindows/CVI.

The *LabWindows/CVI Help* contains complete reference information. The *LabWindows/CVI Help* contains the following sections:

- *Using LabWindows/CVI* provides detailed descriptions and instructions for using the LabWindows/CVI environment, menus, options, tools, and features.
- *Library Reference* provides in-depth function reference for each LabWindows/CVI function. This section includes code examples and overviews of LabWindows/CVI libraries.

- *Programmer Reference* provides information to help you develop programs in LabWindows/CVI. This section includes topics about the LabWindows/CVI compiler, external compilers, user protection, and application distribution.
- *Tools Library* provides an overview of additional instrument drivers included in LabWindows/CVI and function reference for each Tools Library function.

To launch the help file, select **Help»Contents**.

Also, review the *LabWindows/CVI Quick Reference* card provided with this evaluation package. The card provides an overview of LabWindows/CVI, outlines product resources, and lists the LabWindows/CVI libraries.

LabWindows/CVI Examples

LabWindows/CVI installs example programs. Find examples with the NI Example Finder, which you can access by selecting **Help»Find Examples**. You also can visit NI Developer Zone at `ni.com/zone` for new examples.

Tip Examples are an excellent way to get started. Look for an example similar to what you need and modify it according to your specifications.

Customer Education and Certification

For additional training, NI offers hands-on LabWindows/CVI courses. These courses, taught by a LabWindows/CVI expert instructor, provide in-depth training, so you can create sophisticated applications for data acquisition and instrument control. The courses greatly improve your productivity with LabWindows/CVI whether you are a new or intermediate user.

National Instruments also offers the Certified LabWindows/CVI Developer program. The goal of this program credential is to ensure that candidates are equipped with a high level of technical ability with LabWindows/CVI to provide comprehensive measurement and automation solutions for business and industry. To obtain certification, a candidate must successfully complete an examination on LabWindows/CVI programming. The evaluation focuses on the candidate's understanding of programming concepts and style and the candidate's applied use of LabWindows/CVI to solve specific problems.

Visit `ni.com/training` for information about courses, certification, training, technical workshops, and more.

A

Technical Support and Professional Services

Visit the following sections of the National Instruments Web site at ni.com for technical support and professional services:

- **Support**—Online technical support resources at ni.com/support include the following:
  - **Self-Help Resources**—For immediate answers and solutions, visit the award-winning National Instruments Web site for software drivers and updates, a searchable KnowledgeBase, product manuals, step-by-step troubleshooting wizards, thousands of example programs, tutorials, application notes, instrument drivers, and so on.
  - **Free Technical Support**—All registered users receive free Basic Service, which includes access to hundreds of Application Engineers worldwide in the NI Developer Exchange at ni.com/exchange. National Instruments Application Engineers make sure every question receives an answer.
- **Training and Certification**—Visit ni.com/training for self-paced training, eLearning virtual classrooms, interactive CDs, and Certification program information. You also can register for instructor-led, hands-on courses at locations around the world.
- **System Integration**—If you have time constraints, limited in-house technical resources, or other project challenges, NI Alliance Program members can help. To learn more, call your local NI office or visit ni.com/alliance.

If you searched ni.com and could not find the answers you need, contact your local office or NI corporate headquarters. Phone numbers for our worldwide offices are listed at the front of this manual. You also can visit the Worldwide Offices section of ni.com/niglobal to access the branch office Web sites, which provide up-to-date contact information, support phone numbers, email addresses, and current events.