

LabWindows™ /CVI™

Evaluation Guide

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 (Montreal) 514 288 5722, Canada (Ottawa) 613 233 5949,
Canada (Québec) 514 694 8521, Canada (Toronto) 905 785 0085, Canada (Vancouver) 514 685 7530,
China 86 21 6555 7838, Czech Republic 420 2 2423 5774, 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, Greece 30 2 10 42 96 427,
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 001 800 010 0793, Netherlands 31 0 348 433 466,
New Zealand 1800 300 800, Norway 47 0 66 90 76 60, Poland 48 0 22 3390 150, Portugal 351 210 311 210,
Russia 7 095 238 7139, 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 THEREOF 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

Under the copyright laws, this publication may not be reproduced or transmitted in any form, electronic or mechanical, including photocopying, recording, storing in an information retrieval system, or translating, in whole or in part, without the prior written consent of National Instruments Corporation.

Trademarks

CodeBuilder™, CVI™, DataSocket™, IMAQ™, IVI™, LabVIEW™, Measurement Studio™, National Instruments™, NI™, ni.com™, NI-DAQ™, NI Developer Suite™, NI Developer Zone™, and TestStand™ are trademarks of National Instruments Corporation.

Product and company names mentioned herein are trademarks or trade names of their respective companies.

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

Conventions	vii
Related Documentation.....	viii

Chapter 1

National Instruments, Virtual Instrumentation, and LabWindows/CVI

NI Platform	1-1
Virtual Instrumentation	1-2
Step 1—Acquire	1-2
Step 2—Analyze.....	1-2
Step 3—Present	1-2
LabWindows/CVI.....	1-3
A History of Innovation.....	1-3
Measurement Libraries and Components	1-4
Data Acquisition and Instrument Control.....	1-6
Data Analysis.....	1-6
Presentation	1-6
Networking	1-7
More Features to Meet Your Needs.....	1-7
Related Software Packages	1-8
NI Developer Suite	1-8
Measurement Studio.....	1-8
TestStand	1-8
Vision and Image Processing Software.....	1-8
IVI Driver Toolset	1-9
LabWindows/CVI Enterprise Connectivity Toolset	1-9
LabWindows/CVI Signal Processing Toolset	1-9
PID Control Toolset	1-10

Chapter 2

Getting Started with LabWindows/CVI

About this Evaluation Package	2-2
Minimum System Requirements	2-2
Installation Instructions.....	2-3
Exploring LabWindows/CVI.....	2-3
Context-Sensitive Help	2-5

Chapter 3

Creating a LabWindows/CVI Project

Creating a New LabWindows/CVI Project	3-1
Creating a Graphical User Interface	3-1
The User Interface Editor.....	3-2
Generating Your Program Code with CodeBuilder	3-4
Reviewing the Source Code	3-6
The main Function	3-6
The AcquireData Function.....	3-7
The QuitCallback Function.....	3-7
Running the Project.....	3-8
Adding an Instrument Driver to the Project	3-8
Loading the Instrument Driver.....	3-8
Initializing the Instrument.....	3-9
Reading Data from the Instrument.....	3-11
Displaying the Waveform on a Graph Control	3-13
Reviewing the Program	3-14
Running the Completed Project.....	3-15

Chapter 4

Where to Go from Here

Tutorials and Related Documentation	4-1
Programmer and Reference Information	4-1
LabWindows/CVI Examples.....	4-2
Customer Education	4-2

Appendix A

Technical Support and Professional Services

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 your 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 Pop-up Panels, 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 also is used for the proper names of disk drives, paths, directories, programs, subprograms, subroutines, device names, functions, operations, variables, filenames and extensions, and code excerpts.

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*

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. NI provides high performance, seamless integration, and rapid application development of virtual instruments at a lower cost than traditional measurement instruments.

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

NI Platform

NI pioneered the field of virtual instrumentation, which combines powerful software, advanced hardware, and off-the-shelf personal computers, to revolutionize the test and measurement field. This revolution has become the NI platform.

For C-language programmers who develop virtual instrumentation, LabWindows/CVI has become the most popular software development environment. With LabWindows/CVI, you can take advantage of the functionality targeted at test, measurement, and automation applications designed to speed application development. LabWindows/CVI combines the power of ANSI C and the flexibility of Microsoft Windows with easy-to-use tools for building virtual instrumentation systems.

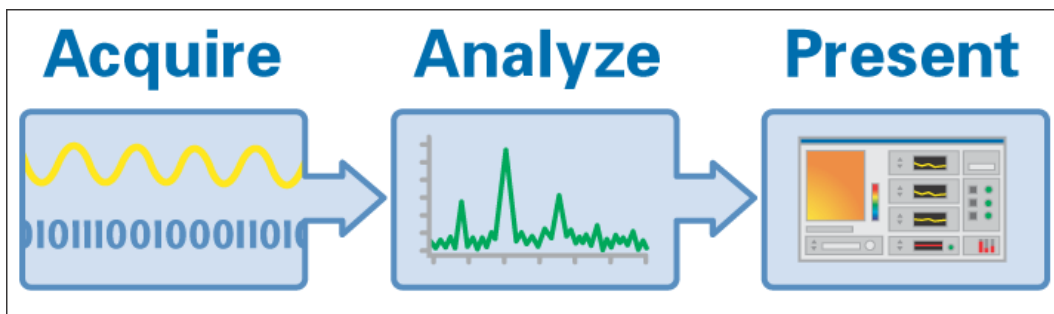
You can use the unique interactive code generation utilities in LabWindows/CVI to unleash the power of ANSI C and speed development in the following areas:

- Creating and controlling graphical user interfaces
- Controlling instruments and plug-in data acquisition hardware
- Developing and debugging ANSI C programs for instrumentation

Virtual Instrumentation

A virtual instrument is a combination of hardware and software in a PC with the functionality of a classic stand-alone instrument. With a virtual instrument, you acquire and control data much like you do with traditional laboratory instruments. With powerful NI virtual instrumentation, you get interactive data analysis, presentation, and application distribution capabilities.

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 hardware connections through an extensive list of interfaces—PCI, PXI, GPIB, VXI, serial, and even Ethernet. Whether you collect data through a plug-in board or through a benchtop instrument, NI has the interface to collect the correct signals and measurements.

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 analysis application, use in-place analysis tools to obtain fast and meaningful results. LabWindows/CVI offers the Advanced Analysis Library for measurement analysis.

Step 3—Present

Present results in an intuitive way and allow users to interact with the instrumentation system through a graphical user interface (GUI). A GUI is the interactive user interface of a virtual instrument. Use three-dimensional, instrument-style controls and indicators—buttons, knobs, thermometers, tanks, LEDs, slides, numerical displays, strip charts, graphs, trees, and tables—to build panels that represent traditional test and measurement instruments.

LabWindows/CVI

LabWindows/CVI is a proven ANSI C development environment and compiler with built-in libraries for acquisition, analysis, and presentation. LabWindows/CVI provides a comprehensive drag-and-drop User Interface Editor and automated code generation tools that you can use to interactively test code before adding it to your project. With LabWindows/CVI, you can create localized user interfaces, work with ActiveX components, and create multithreaded applications. LabWindows/CVI includes functions to create and control GUIs, functions to control instruments and plug-in data acquisition hardware from your PC, and tools to develop and debug ANSI C programs.

A History of Innovation

- **LabWindows/CVI 7.0** (2003)—Integrated workspace, advanced instrument and data acquisition control, expanded User Interface Library
- **LabWindows/CVI 6.0** (2001)—ActiveX Container, additional user protection
- **LabWindows/CVI 5.5** (1999)—Multithreaded libraries/debugging
- **LabWindows/CVI 5.0** (1998)—New instrument driver technologies
- **LabWindows/CVI 4.0** (1996)—External C/C++ compiler support
- **LabWindows/CVI 3.1** (1995)—Automatic program generation
- **LabWindows/CVI 3.0** (1994)—Multi-platform for Windows and Sun
- **LabWindows for DOS 2.0** (1991)—Graphical user interface tools, memory extender
- **LabWindows for DOS 1.0** (1989)—Introduction

This time line represents a commitment to innovation and development that has allowed our customers to develop measurement and automation applications using an optimized ANSI C tool. LabWindows/CVI first emerged as a DOS-based tool for adding data acquisition and instrument control to test applications. NI added graphical user interface tools and extended memory capabilities in LabWindows for DOS 2.0. Following versions of LabWindows/CVI expanded the platform to Windows and Sun while enhancing user productivity with innovative code generation tools. LabWindows/CVI was enhanced to provide external compiler support for multiple development environments. NI then greatly extended the power of LabWindows/CVI by adding multithreaded capabilities, enhanced debugging functionality, and additional instrument connectivity with *VXIplug&play* and IVI support. LabWindows/CVI 6.0 offered new features by completing the IVI class controls, adding a new 3D appearance to the user interface controls, and incorporating ActiveX capabilities. LabWindows/CVI has again revolutionized ANSI C productivity through an entirely integrated workspace; interactive, code generating I/O assistants; state-of-the-art hardware support; new user interface controls; and enhanced debugging capabilities.

Measurement Libraries and Components

You can interact with the LabWindows/CVI analysis, I/O, presentation, and other measurement-specific libraries and components through function panels. Figure 1-1 shows the User Interface Library as displayed in the Library Tree.

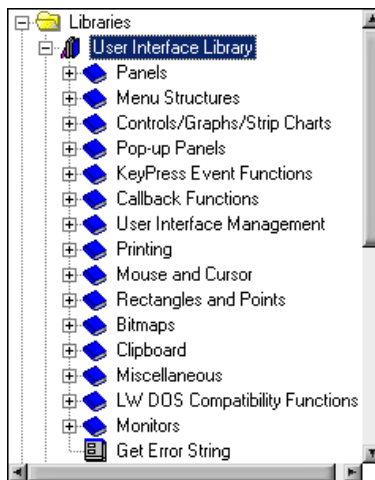


Figure 1-1. User Interface Library in Library Tree

Function panels are graphical representations of LabWindows/CVI functions and their parameters. You can use function panels to interactively test your function calls and paste them into your program. Figure 1-2 shows the function panel for `SetCtrlAttribute`.

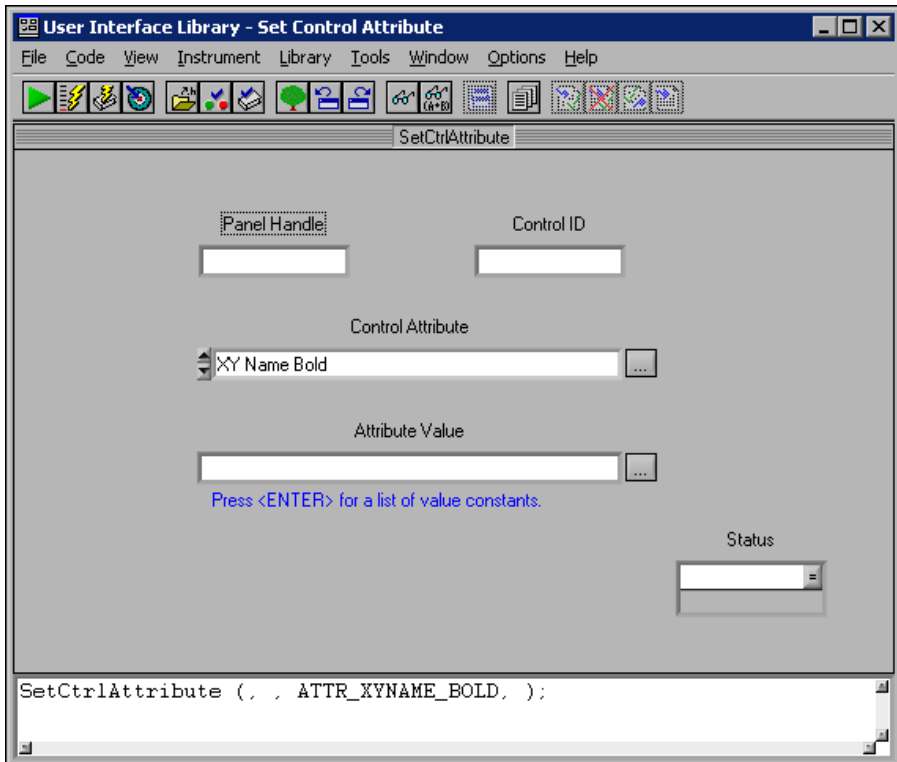


Figure 1-2. SetCtrlAttribute Function Panel

Function panels have help for the function, which you can access by right-clicking the function panel. Figure 1-3 shows the function panel help for `SetCtrlAttribute`.

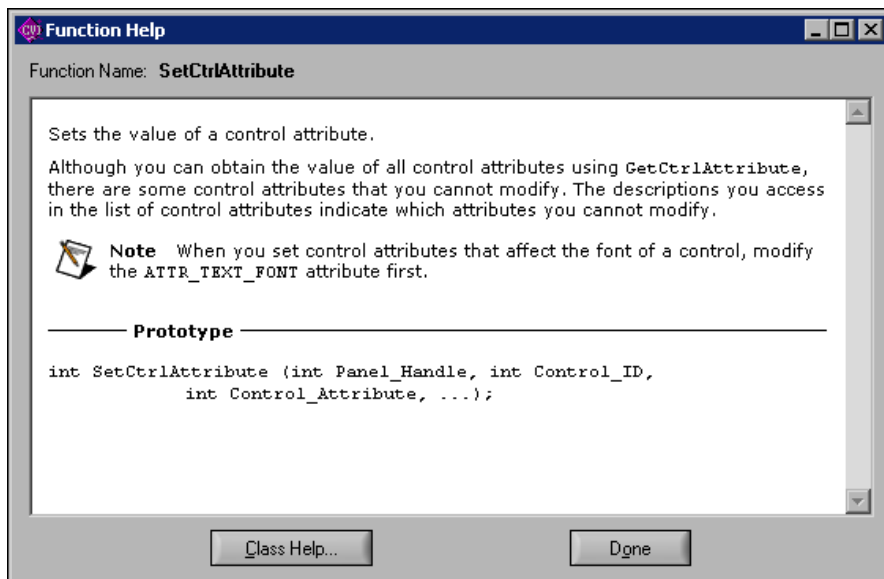


Figure 1-3. Function Panel Help for `SetCtrlAttribute`

Data Acquisition and Instrument Control

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

Data Analysis

LabWindows/CVI delivers robust analysis functions you can use to convert raw data into meaningful information. You can employ a variety of signal processing and data analysis tools—such as curve fitting, spectral analysis, statistics, and visualization—for research and development, engineering and validation, or manufacturing and service applications.

Presentation

LabWindows/CVI delivers ready-to-use, measurement-specific user interface components with which you can create professional measurement applications that mimic stand-alone instruments, such as scopes and temperature loggers. Use three-dimensional controls to create revolutionary application interfaces.

Networking

Whether you exchange data between applications or over the Internet, LabWindows/CVI makes sharing measurement data easy. With NI DataSocket, a programming tool designed for publishing and subscribing to live data in measurement and automation applications, you can publish or retrieve data, control remote applications, or distribute your system with one or more client applications on a network without worrying about data formats and network protocols.

Connect to data sources through the following interfaces from applications you develop with LabWindows/CVI.

- DataSocket Server Transfer Protocol (DSTP)
- OLE for Process Control (OPC)
- Hypertext Transfer Protocol (HTTP)
- File Transfer Protocol (FTP)
- File
- Logos

More Features to Meet Your Needs

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

- **Integrated Workspace Environment**—The Workspace window contains many of the different components of the LabWindows/CVI environment. In the Workspace window, you can conveniently navigate through all workspace projects and view libraries and instruments you have loaded in LabWindows/CVI. Use the Output Window Region of the Workspace window to view search matches, debug messages, and error messages.
- **State-of-the-Art Hardware Support**—LabWindows/CVI provides enhanced hardware support, including the integrated Instrument I/O Assistant and DAQ Assistant and support for the NI-DAQmx Library.
- **Expanded User Interface Library**—In addition to the traditional user interface controls, the User Interface Library includes a new, full-featured tree control and an enhanced graph control.
- **Improved Function Panel Usability**—LabWindows/CVI now provides function panel customization capabilities, HTML-based function panel help, and the option to convert function panels to XML documents.
- **Compiler/Debugger Enhancements**—With the enhanced debugger, you can change the point of execution while you are debugging and quick-edit the value of a variable in tooltips. The LabWindows/CVI compiler also supports 64-bit integers.

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

For one package that includes all of the tools you need to develop an automated test application, consider NI Developer Suite. NI Developer Suite is a software subscription program that packages our most popular software and includes free updates for a year. The NI Developer Suite Professional Test Edition includes LabWindows/CVI, LabVIEW, and Measurement Studio for developing test routines and TestStand for managing test execution, sequencing, collecting data, and generating reports. The test edition also includes a comprehensive set of LabVIEW, Measurement Studio, and LabWindows/CVI add-on tools for Internet connectivity, database communication, signal processing, and code distribution.

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. New code designers interactively define reusable acquisition tasks and automatically generate code. Advanced analysis libraries and rich object-oriented hardware APIs, such as data acquisition and instrument control APIs, enable you to develop sophisticated measurement applications. The highly extensible C++ and .NET class libraries in Measurement Studio deliver unparalleled flexibility to scientists and engineers. Measurement Studio is included with the LabWindows/CVI Full Development System.

TestStand

TestStand is a ready-to-run test executive for organizing, controlling, and executing your automated prototype, validation, or manufacturing test systems. TestStand is completely customizable, so you can modify and enhance it to match your specific needs. Built on a high-speed, multithreaded, parallel execution engine, TestStand delivers high performance to meet your rigorous test throughput requirements. TestStand comes complete with integrated LabWindows/CVI tools, including the LabWindows/CVI flexible adapter.

Vision and Image Processing Software

NI Vision and Image Processing Software 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, so you can preserve your investment in test software as your instruments and instrumentation platforms change. 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. IVI instrument drivers are designed to handle the more rigorous requirements of production test systems for speed, flexibility, and long-term reusability.



Tip Visit the Instrument Driver Network at ni.com/idnet to find additional instrument drivers. Browse through instrument drivers by instrument type, manufacturer, or development language. Or, search for a specific instrument driver using the specifications most important to you—from a specific model name or manufacturer to IVI drivers or drivers supported by NI.

LabWindows/CVI Enterprise Connectivity Toolset

Enterprise connectivity tools help you track progress from research and development to the production, testing, and servicing of products. The Enterprise Connectivity Toolset provides integrated tools for Structured Query Language (SQL) database operations, statistical process control (SPC) quality control, and Internet-enabling technology to give you the access and networking you need for business operations.

The following toolkits are included in the LabWindows/CVI Enterprise Connectivity Toolset.

- LabWindows/CVI Internet Developers Toolkit
- LabWindows/CVI SQL Toolkit
- LabWindows/CVI SPC Toolkit

LabWindows/CVI Signal Processing Toolset

The Signal Processing Toolset provides four toolkits for digital filter design, joint time-frequency analysis, wavelet and filter bank design, and super-resolution, model-based spectral analysis.

The following toolkits are included in the Signal Processing Toolset.

- **Digital Filter Design Toolkit**—Use for signal conditioning, control systems, and digital signal processing.
- **Joint Time-Frequency Analysis Toolkit**—Use to simultaneously examine the time and frequency domain representations of a signal.
- **Wavelet & Filter Bank Toolkit**—Use for wavelet design and implementation.
- **Super-Resolution Spectral Analysis Toolkit**—Use the stand-alone application that provides a model-based alternative to the 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. By combining the PID Control Toolset with the math and logic functions in LabWindows/CVI, you can quickly develop process control applications.

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.

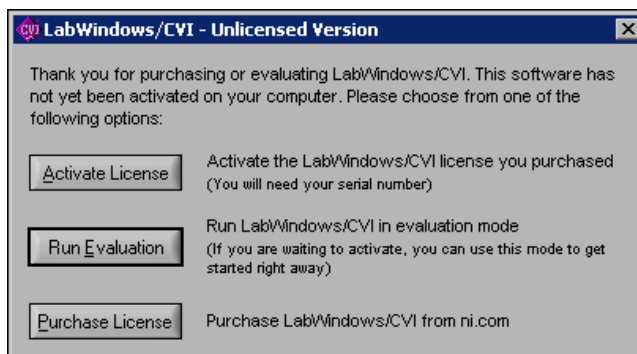


Figure 2-1. Unlicensed Version Dialog Box

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 your copy of LabWindows/CVI after you have purchased a license, click **Activate License** and follow the instructions for activation.

About this Evaluation Package

Evaluate LabWindows/CVI with your test and measurement applications. The Full Development System includes the data acquisition, instrument control, user interface, advanced analysis, digital signal processing, and networking tools you need to build test and measurement programs. With LabWindows/CVI, you can acquire data with NI data acquisition (DAQ) boards; control serial, GPIB, 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 To prevent the loss of work, save all program files created with the LabWindows/CVI Evaluation Package and open them later with a licensed version of LabWindows/CVI.

Minimum System Requirements

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

- Personal computer using a Pentium 600 or higher microprocessor
- Windows 2000/NT SP6/XP/Me/98
- 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

Unless you specify another location during installation, the LabWindows/CVI installation program copies files to `c:\Program Files\National Instruments\CVI70\` after you complete the following steps:

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. On installation startup, the National Instruments LabWindows/CVI 7.0 screen appears. Select **Install LabWindows/CVI**. Continue to follow the instructions on the screen.
3. Install driver software if you plan to use LabWindows/CVI with NI hardware.
4. Install your 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 in your applications.

As you develop applications in LabWindows/CVI, you work within the Workspace window. The Workspace window contains four areas: the Project Tree, the Library Tree, the Window Confinement Region, and the Output Window Region. The Project Tree provides access to files for each project in the workspace. The Library Tree provides access to function panels for the functions in LabWindows/CVI libraries and loaded instruments. The Window Confinement Region contains open Source windows, User Interface windows, and Function Tree Editor windows. The Output Window 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.

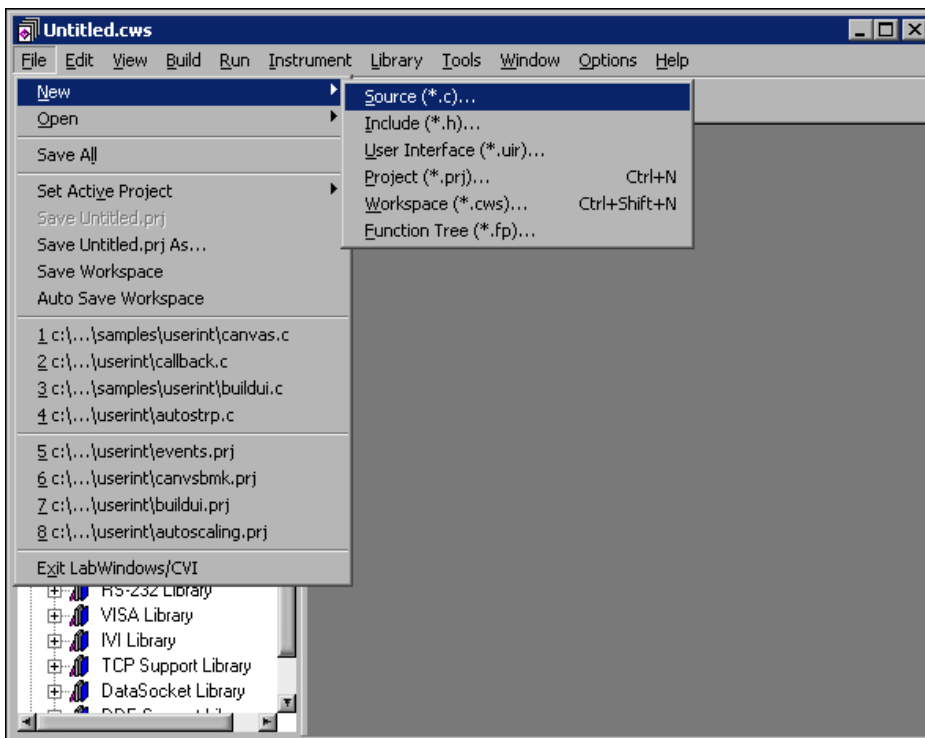


Figure 2-2. LabWindows/CVI Workspace Window

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 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. Look for and take advantage of **Help** buttons in dialog boxes and F1 connectivity in the Source window, User Interface Editor, and menus in all windows.

- **Help file**—Select **Help>Contents** in LabWindows/CVI to access the *LabWindows/CVI Help*.
- **Menus, dialog boxes, and Source window keywords**—Press <F1> on menus, dialog boxes, or function names in the Source window. You also can right-click menu items and select **Menu Help** to learn more about the options and features available in that menu.
- **Function panels**—Right-click the panel or any parameter for help about that function or parameter.
- **User Interface Editor**—Right-click any control and select **Control Help** to open help for GUI controls. You also can access help in Edit Control dialog boxes. Click the question mark button in the Edit Control dialog box titlebar and then select a control to view help.

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 your project.

Creating a New LabWindows/CVI Project



You can complete this activity in 2 minutes.

To create a project file, 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 users of your application can interact with and monitor the measurement or control system. Choose from the following controls to create your user interface.

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

The User Interface Editor

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

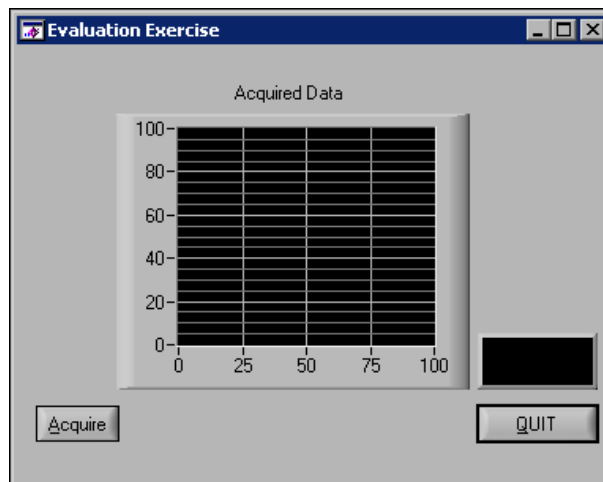


Figure 3-1. Evaluation Exercise GUI

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. This header file is stored in the same directory as the `.uir` file.

Adding Command Buttons

1. Select **Create»Command Button»Square Command Button**. A button labeled **OK** appears 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** in the **Callback Function** field. The callback function is called 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” appears underlined in the label. The user 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**.

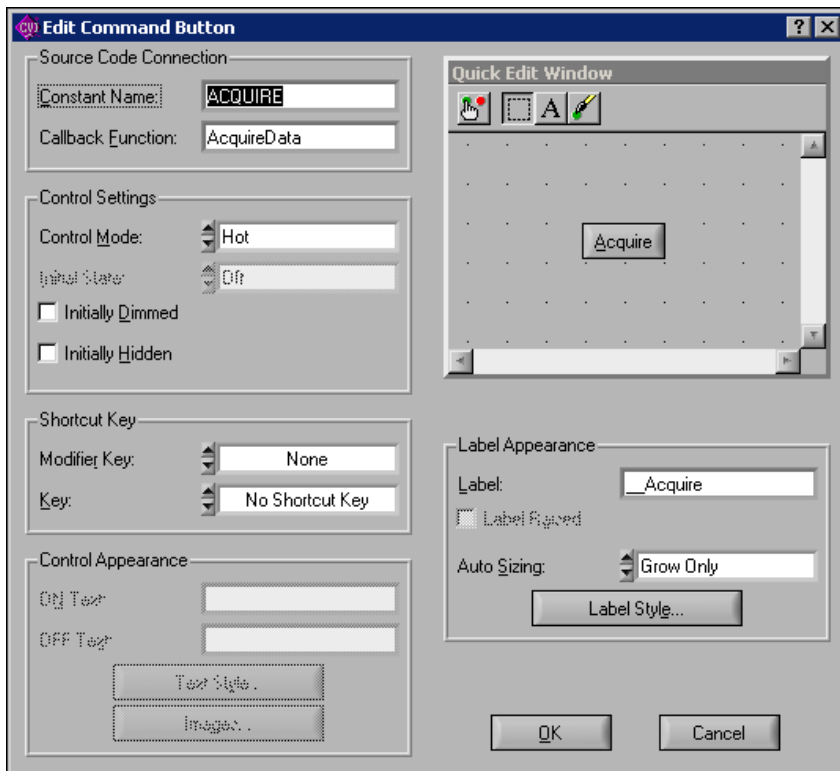


Figure 3-2. Edit Command Button Dialog Box

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 between the command buttons.
3. Double-click the graph control to open the Edit Graph dialog box.
4. Enter `WAVEFORM` as the **Constant Name**.



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` in the **Label** field.
6. Click **OK** to close the dialog box.
7. Save `AcquireWaveform.uir`.

Generating Your 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 control's Edit dialog box, 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 that executes 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 your 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 your 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>). Verify that only `EVENT_COMMIT` is selected and then click **OK**.

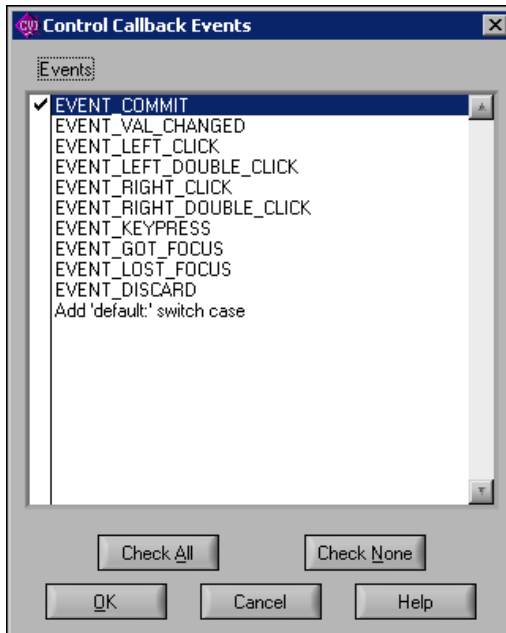


Figure 3-3. Control Callback Events Dialog Box



Tip Click the **Help** button in the dialog boxes to learn more about default events and generating code. 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 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.
7. Click **OK**. CodeBuilder builds the source code for your program and adds the file to the project. A new Source window, `AcquireWaveform.c`, appears containing the new source code.

Reviewing the Source Code

The 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.

```

6 | int main (int argc, char *argv[])
7 | {
8 |     if (InitCVRTE (0, argv, 0) == 0)
9 |         return -1; /* out of memory */
10 |     if ((panelHandle = LoadPanel (0, "AcquireWaveform.uir", PANEL)) < 0)
11 |         return -1;
12 |     DisplayPanel (panelHandle);
13 |     RunUserInterface ();
14 |     DiscardPanel (panelHandle);
15 |     return 0;
16 | }
17 |

```

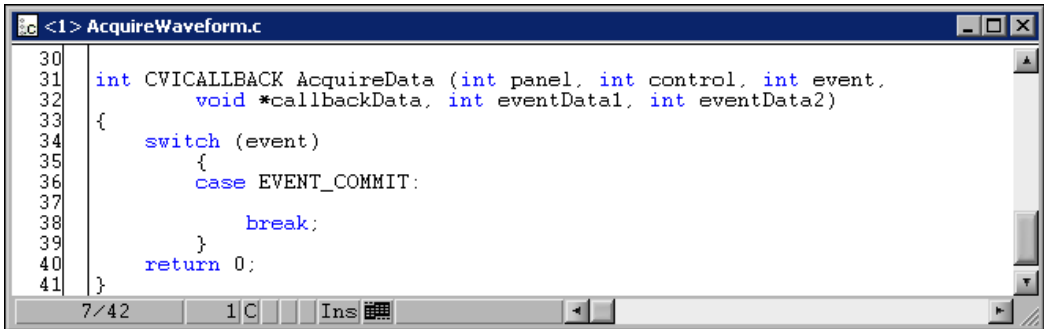
Figure 3-4. main Function

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 you are developing.
- `DiscardPanel` removes the panel from memory and clears it from the screen if visible.

The 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.



```

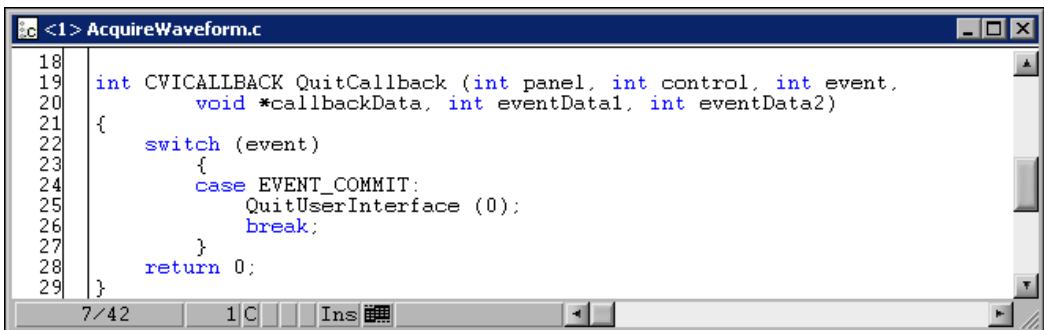
30
31 int CVICALLBACK AcquireData (int panel, int control, int event,
32     void *callbackData, int eventData1, int eventData2)
33 {
34     switch (event)
35     {
36         case EVENT_COMMIT:
37
38             break;
39     }
40     return 0;
41 }

```

Figure 3-5. AcquireData Function

The QuitCallback Function

The QuitCallback function automatically executes when a user clicks the **QUIT** button. This function disables the user interface from sending event information to the callback function and stops execution of the program.



```

18
19 int CVICALLBACK QuitCallback (int panel, int control, int event,
20     void *callbackData, int eventData1, int eventData2)
21 {
22     switch (event)
23     {
24         case EVENT_COMMIT:
25             QuitUserInterface (0);
26             break;
27     }
28     return 0;
29 }

```

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 will use communicates with a simulated instrument and shows you how to use an instrument driver to acquire a waveform from an oscilloscope.

To be complete, your 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.



Note Visit ni.com/idnet to access the online Instrument Driver Network—the industry’s largest source of instrument drivers, featuring drivers for over 2,200 instruments from over 150 vendors. Here you can download and submit drivers for controlling instruments from LabWindows/CVI.

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`.

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

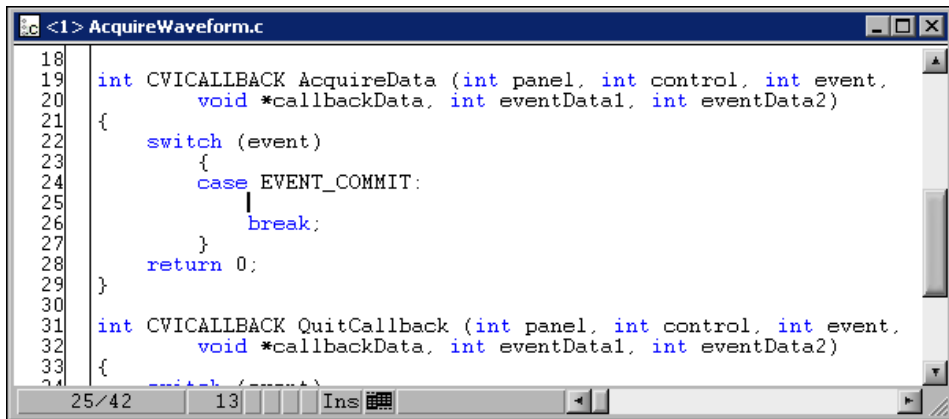


Figure 3-7. Cursor Position in `AcquireWaveform.c`

Initializing the Instrument

- 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.

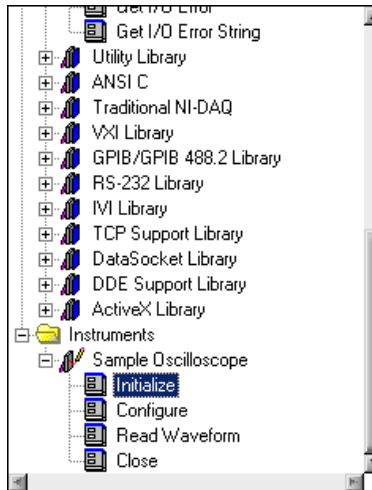


Figure 3-8. Sample Oscilloscope Instrument Driver

2. Double-click **Initialize**. The scope initialization function panel appears, as shown in Figure 3-9.

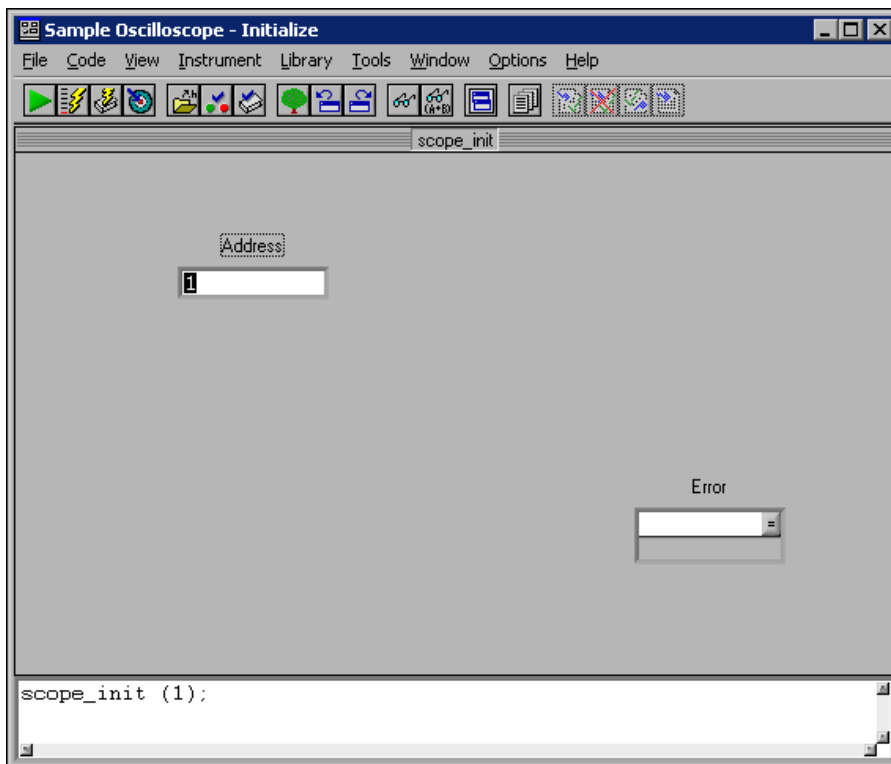


Figure 3-9. Scope Initialization Function Panel

Function panels generate code. 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. Declare the **err** variable. Select **Code»Declare Variable**.
6. In the Declare Variable dialog box, select the **Execute declaration in Interactive Window** and **Add declaration to top of target file** options. Click **OK**.

7. Select **Code»Run Function Panel**. If no errors are detected during execution, the value in the **Error** control is 0.
8. Select **Code»Insert Function Call** to copy the generated code to the Source window.

The function call to initialize the instrument driver now appears in your 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 your 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. The function panel shown in Figure 3-10 appears.

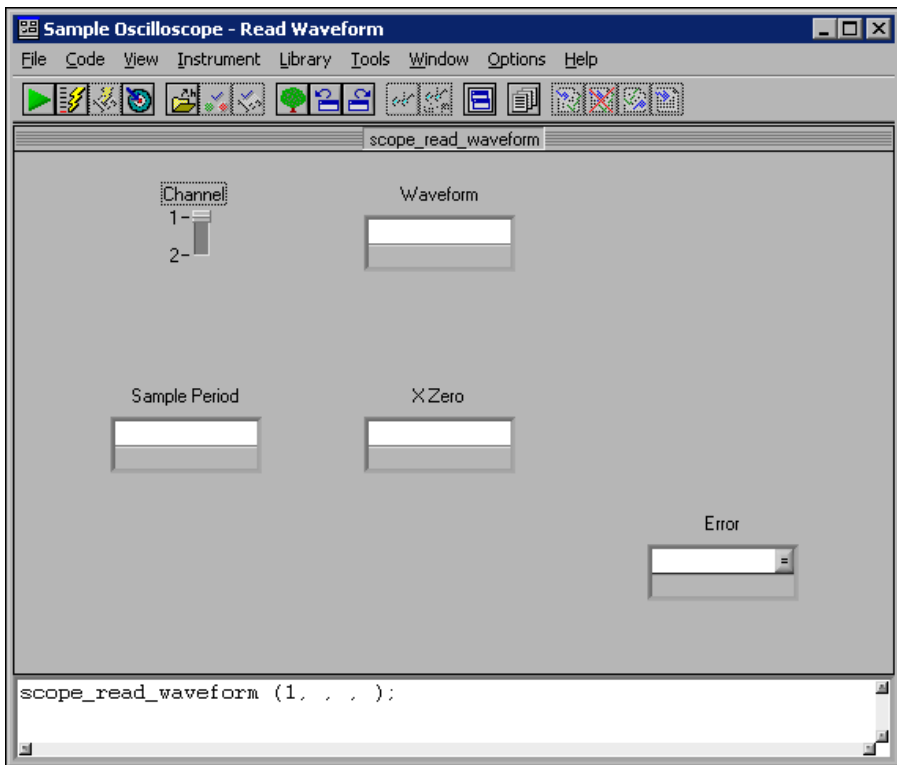


Figure 3-10. Read Waveform Function Panel

2. In the Read Waveform function panel, 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** options are selected.
6. Click **OK** to declare the `datapoints` array.
7. Enter `delta_t` in the **Sample Period** control.
8. Select **Code»Declare Variable**. Verify that the **Execute declaration in Interactive Window** and **Add declaration to top of target file** options are selected and click **OK**.
9. Enter `x_zero` in the **X Zero** control.
10. Select **Code»Declare Variable**. Verify that the **Execute declaration in Interactive Window** and **Add declaration to top of target file** options are selected and click **OK**.
11. Enter `err` in the **Error** control.
12. 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 indicates 0. After the function executes, a row of boxes in the **Waveform** control signifies that the data is in the waveform array.
13. Optional—You can double-click the row of boxes to display two windows of data returned by the function call: one showing variable values and the other showing the waveform data collected in the array. If you choose to review these windows, close both of them before continuing.
14. Select **Code»Insert Function Call** to copy the generated code to the Source window. The following line of code now exists 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. Now you need to plot this data using the `PlotY` function, which can plot the acquired data array to the graph control on the user interface.

1. To open the Plot Y function panel, type `PlotY` in the blank line below the read waveform function call and press `<Ctrl-P>`.
2. Complete the function panel controls to match the following settings:

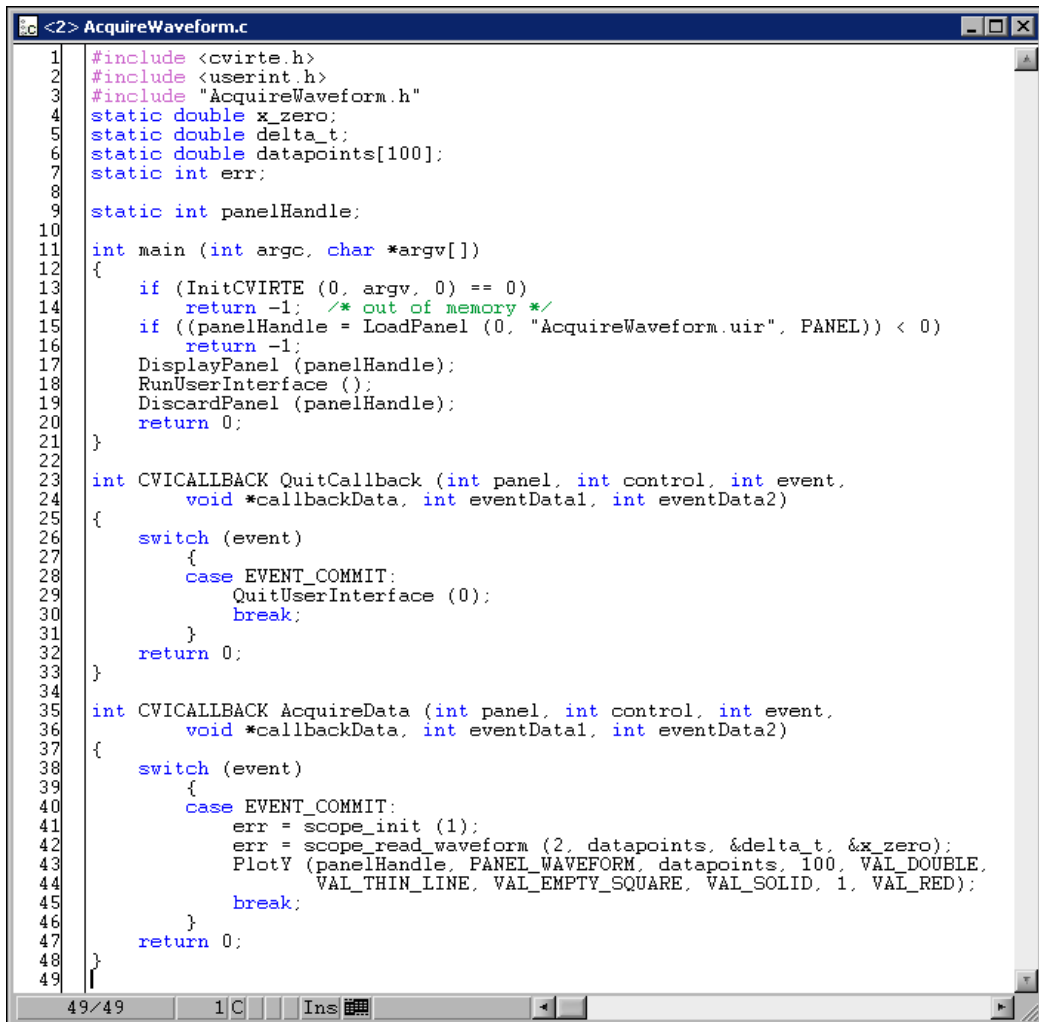
Panel Handle:	<code>panelHandle</code>
Control ID:	<code>PANEL_WAVEFORM</code>
Y Array:	<code>datapoints</code>
Number of Points:	<code>100</code>

Leave the default settings for the remaining controls.

3. From the Plot Y function panel, select **Code»Insert Function Call** to paste the `PlotY` function call into your source code. A message box appears with a warning that text already exists on the line where LabWindows/CVI will insert the function call. Click **Replace**.
4. The changes you made to the `AcquireData` callback function match the source code shown in the [Reviewing the Program](#) section.
5. Save `AcquireWaveform.c`.

Reviewing the Program

Make sure that your completed source file matches the code block shown in Figure 3-11.



```

1  #include <cvirte.h>
2  #include <userint.h>
3  #include "AcquireWaveform.h"
4  static double x_zero;
5  static double delta_t;
6  static double datapoints[100];
7  static int err;
8
9  static int panelHandle;
10
11 int main (int argc, char *argv[])
12 {
13     if (InitCVMIRTE (0, argv, 0) == 0)
14         return -1; /* out of memory */
15     if ((panelHandle = LoadPanel (0, "AcquireWaveform.uir", PANEL)) < 0)
16         return -1;
17     DisplayPanel (panelHandle);
18     RunUserInterface ();
19     DiscardPanel (panelHandle);
20     return 0;
21 }
22
23 int CVICALLBACK QuitCallback (int panel, int control, int event,
24     void *callbackData, int eventData1, int eventData2)
25 {
26     switch (event)
27     {
28         case EVENT_COMMIT:
29             QuitUserInterface (0);
30             break;
31     }
32     return 0;
33 }
34
35 int CVICALLBACK AcquireData (int panel, int control, int event,
36     void *callbackData, int eventData1, int eventData2)
37 {
38     switch (event)
39     {
40         case EVENT_COMMIT:
41             err = scope_init (1);
42             err = scope_read_waveform (2, datapoints, &delta_t, &x_zero);
43             PlotY (panelHandle, PANEL_WAVEFORM, datapoints, 100, VAL_DOUBLE,
44                 VAL_THIN_LINE, VAL_EMPTY_SQUARE, VAL_SOLID, 1, VAL_RED);
45             break;
46     }
47     return 0;
48 }
49

```

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. A message appears prompting you that your Interactive Execution declarations will be cleared. Select **OK**. During the compile process, LabWindows/CVI recognizes that your program is missing the `scope.h` header file statement and offers to insert it into your source code. Click **Yes** to add this include file to the program.



Note When you click **Acquire** more than once, the data from previous acquisitions remains on display with the new data plot. You can modify the `AcquireData` callback to include the `DeleteGraphPlot` function as shown in the code excerpt in Figure 3-12. `DeleteGraphPlot` deletes all plots from the graph.



Note You can find the `DeleteGraphPlot` function in the User Interface Library.

```

34
35 int CVICALLBACK AcquireData (int panel, int control, int event,
36     void *callbackData, int eventData1, int eventData2)
37 {
38     switch (event)
39     {
40         case EVENT_COMMIT:
41             DeleteGraphPlot (panelHandle, PANEL_WAVEFORM, -1, VAL_DELAYED_DRAW);
42             err = scope_init (1);
43             err = scope_read_waveform (2, datapoints, &delta_t, &x_zero);
44             PlotY (panelHandle, PANEL_WAVEFORM, datapoints, 100, VAL_DOUBLE,
45                 VAL_THIN_LINE, VAL_EMPTY_SQUARE, VAL_SOLID, 1, VAL_RED);
46             break;
47     }
48     return 0;
49 }

```

Figure 3-12. Modified AcquireData Function

Now that you have completed a LabWindows/CVI project, which is shown in Figure 3-13, look through the rest of the libraries for functionality not covered in this manual. Also, explore the samples included in this evaluation version of LabWindows/CVI. You can find a number of samples in the `samples` folder.

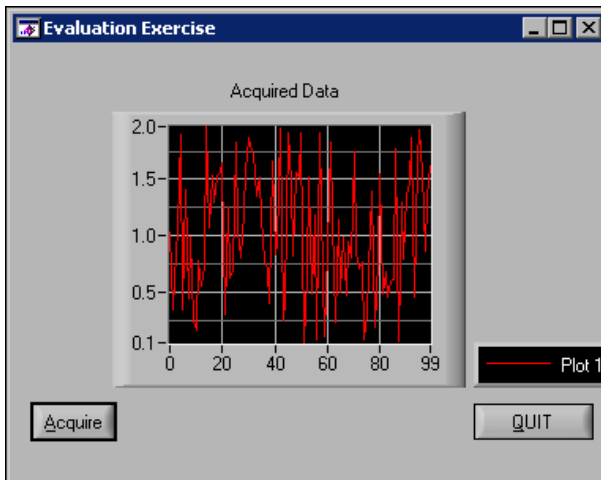


Figure 3-13. Completed Project

Where to Go from Here

LabWindows/CVI is yet another long-standing commitment by NI to provide tools that simplify the development of instrumentation, data acquisition, and control systems. 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.

Tutorials and Related Documentation

Launch the *LabWindows/CVI Bookshelf*, available from **Start»Programs»National Instruments»LabWindows CVI»LabWindows CVI Bookshelf**. 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.adobe.com.

Read the *Getting Started with LabWindows/CVI* manual for a complete tutorial.

Programmer and Reference Information

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 each menu, option, tool, and feature available in LabWindows/CVI.
- *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.

- *Example Programs* provides descriptions of sample programs.
- *Tools Library* provides an overview of additional instrument drivers included in LabWindows/CVI.

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

Also, review the *LabWindows/CVI Quick Reference* card provided with this evaluation package. The card 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»NI Example Finder**. 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

For additional training, NI offers hands-on LabWindows/CVI courses. These courses, taught by a LabWindows/CVI expert instructor, prepare you to unlock the potential of LabWindows/CVI, 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. Visit ni.com/training for information about courses, training, technical workshops, and more.



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 include the following:
 - **Self-Help Resources**—For immediate answers and solutions, visit our extensive library of technical support resources available in English, Japanese, and Spanish at ni.com/support. These resources are available for most products at no cost to registered users and include software drivers and updates, a KnowledgeBase, product manuals, step-by-step troubleshooting wizards, conformity documentation, example code, tutorials and application notes, instrument drivers, discussion forums, a measurement glossary, and so on.
 - **Assisted Support Options**—Contact NI engineers and other measurement and automation professionals by visiting ni.com/support. Our online system helps you define your question and connects you to the experts by phone, discussion forum, or email.
- **Training**—Visit ni.com/training for self-paced tutorials, videos, and interactive CDs. 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.