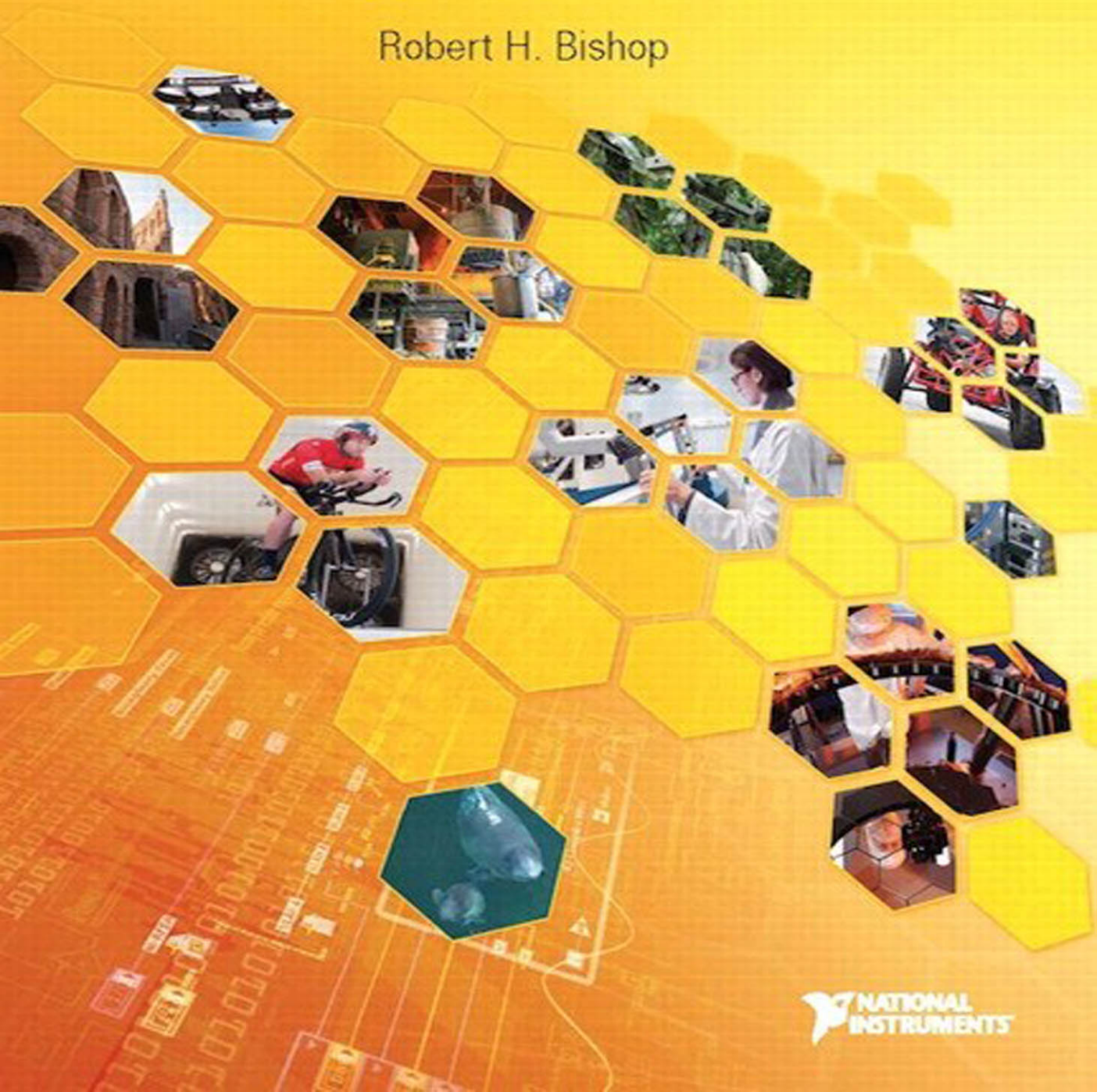


Learning With LabVIEW™

Robert H. Bishop



Learning with LabVIEW™

This page intentionally left blank

Learning with LabVIEW™

Robert H. Bishop

Marquette University

PEARSON

Upper Saddle River Boston Columbus San Francisco New York
Indianapolis London Toronto Sydney Singapore Tokyo Montreal
Dubai Madrid Hong Kong Mexico City Munich Paris Amsterdam Cape Town

Vice President and Editorial Director, ECS: *Marcia J. Horton*
Acquisitions Editor: *Norrin Dias*
Editorial Assistant: *Sandra Rodriguez*
Senior Managing Editor: *Scott Disanno*
Procurement Manager: *Mary Fischer*
Senior Specialist, Program Planning & Support: *Maura Zaldivar-Garcia*
Product Marketing Manager: *Bram van Kempen*
Field Marketing Manager: *Demetrius Hall*
Marketing Assistant: *Jon Bryant*
Cover Design: *National Instruments*
Composition: *Jouve India*

LabVIEW is a trademark of National Instruments.

MATLAB is a registered trademark of The MathWorks, 3 Apple Hill Drive, Natick, MA 01760-2098.

Company and product names mentioned herein are the trademarks or registered trademarks of their respective owners.

Copyright © 2015 by Pearson Education, Inc., Upper Saddle River, New Jersey 07458. All rights reserved. Manufactured in the United States of America. This publication is protected by Copyright and permissions should be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission(s) to use materials from this work, please submit a written request to Pearson Higher Education, Permissions Department, 1 Lake Street, Upper Saddle River, NJ 07458.

The author and publisher of this book have used their best efforts in preparing this book. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The author and publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

Library of Congress Cataloging-in-Publication Data on File.



10 9 8 7 6 5 4 3 2 1

ISBN-13: 978-0-13-402212-3

ISBN-10: 0-13-402212-2

*To my parents, W. Robert Bishop
and Anna Maria DiPietro Bishop*

This page intentionally left blank



CONTENTS

Preface xix

1	LabVIEW Basics	1
1.1	System Configuration Requirements	2
1.2	Installing the <i>LabVIEW Student Edition</i>	2
1.3	The LabVIEW Environment	3
1.4	The Getting Started Screen	5
1.5	Panel and Diagram Windows	9
	1.5.1 Front Panel Toolbar	9
	1.5.2 Block Diagram Toolbar	13
1.6	Shortcut Menus	15
1.7	Pull-Down Menus	16
	1.7.1 File Menu	17
	1.7.2 Edit Menu	17
	1.7.3 View Menu	17
	1.7.4 Project Menu	20
	1.7.5 Operate Menu	20
	1.7.6 Tools Menu	21
	1.7.7 Window Menu	21
	1.7.8 Help Menu	23
1.8	Palettes	24
	1.8.1 Tools Palette	24
	1.8.2 Controls Palette	25
	1.8.3 Functions Palette	27
	1.8.4 Searching the Palettes and Quick Drop	28
1.9	Opening, Loading, and Saving VIs	31

- 1.10 LabVIEW Help Options 33**
 - 1.10.1 Context Help Window 33
 - 1.10.2 LabVIEW Help 36
- 1.11 Building Blocks: Pulse Width Modulation 36**
- 1.12 Relaxed Reading: Revolutionary Cancer Treatment Minimizes Damage to Healthy Tissue 38**
- 1.13 myDAQ Building Blocks 40**
 - 1.13.1 Building a Basic LED Circuit 41
 - 1.13.2 NI ELVISmx Instrument Launcher 42
 - 1.13.3 Testing with the ELVISmx Instrument Panel 42
- 1.14 Summary 43**
 - Exercises 45*
 - Problems 50*
 - Design Problems 52*

2 Virtual Instruments

57

- 2.1 What Are Virtual Instruments? 58**
- 2.2 Several Worked Examples 60**
- 2.3 The Front Panel 67**
 - 2.3.1 Numeric Controls and Indicators 67
 - 2.3.2 Boolean Controls and Indicators 69
 - 2.3.3 Configuring Controls and Indicators 73
- 2.4 The Block Diagram 73**
 - 2.4.1 VIs and Express VIs 74
 - 2.4.2 Nodes 75
 - 2.4.3 Terminals 76
 - 2.4.4 Wiring 79
- 2.5 Building Your First VI 82**
- 2.6 Data Flow Programming 87**
- 2.7 Building a VI Using Express VIs 89**
- 2.8 Building Blocks: Pulse Width Modulation 96**
- 2.9 Relaxed Reading: Building A Semiautonomous Vehicle Driven By the Visually Impaired 98**
- 2.10 myDAQ Building Blocks 100**
 - 2.10.1 Building a Basic Thermistor Circuit 101
 - 2.10.2 The Thermistor Circuit Test VI 101
 - 2.10.3 Testing the Thermistor Circuit 102

2.11	Summary	103
	<i>Exercises</i>	<i>104</i>
	<i>Problems</i>	<i>110</i>
	<i>Design Problems</i>	<i>113</i>
3	Editing and Debugging Virtual Instruments	117
3.1	Editing Techniques	118
3.1.1	Creating Controls and Indicators on the Block Diagram	118
3.1.2	Selecting Objects	120
3.1.3	Moving Objects	122
3.1.4	Deleting and Duplicating Objects	122
3.1.5	Resizing Objects	123
3.1.6	Labeling Objects	124
3.1.7	Changing Font, Style, and Size of Text	127
3.1.8	Selecting and Deleting Wires	130
3.1.9	Wire Stretching and Broken Wires	131
3.1.10	Aligning, Distributing, and Resizing Objects	134
3.1.11	Coloring Objects	136
3.1.12	Cleaning Up the Block Diagram	144
3.1.13	Routing Wires	146
3.1.14	Reusing Snippets of Code	148
3.2	Debugging Techniques	149
3.2.1	Finding Errors	150
3.2.2	Highlight Execution	151
3.2.3	Single-Stepping Through a VI and Its SubVIs	154
3.2.4	Breakpoints and Probes	155
3.2.5	Navigation Window	159
3.3	Property Nodes	160
3.4	A Few Shortcuts	163
3.5	Building Blocks: Pulse Width Modulation	163
3.6	Relaxed Reading: Using Graphical System Design for Tumor Treatments	166
3.7	myDAQ Building Blocks	168
3.8	Summary	170
	<i>Exercises</i>	<i>172</i>
	<i>Problems</i>	<i>174</i>
	<i>Design Problems</i>	<i>178</i>

4	SubVIs	182
4.1	What Is a SubVI? 183	
4.2	Review of the Basics 184	
4.3	Editing the Icon and Connector 186	
4.3.1	Icons 187	
4.3.2	Connectors 190	
4.3.3	Selecting and Modifying Terminal Patterns 191	
4.3.4	Assigning Terminals to Controls and Indicators 193	
4.4	The Help Window 195	
4.5	Using a VI as a SubVI 199	
4.6	Creating a SubVI from a Selection 202	
4.7	Error Checking and Error Handling 204	
4.7.1	Automatic Error Handling 204	
4.7.2	Manual Error Handling 205	
4.7.3	Error Clusters 205	
4.8	Saving Your SubVI 206	
4.9	The VI Hierarchy Window 207	
4.10	Building Blocks: Pulse Width Modulation 209	
4.11	Relaxed Reading: Mobile Rescue Operations 212	
4.12	myDAQ Building Blocks 213	
4.13	Summary 218	
	<i>Exercises</i> 219	
	<i>Problems</i> 222	
	<i>Design Problems</i> 225	
5	Structures	228
5.1	The For Loop 229	
5.1.1	Numeric Conversion 231	
5.1.2	For Loops with Conditional Terminals 235	
5.2	The While Loop 236	
5.3	Shift Registers and Feedback Nodes 242	
5.3.1	Shift Registers 242	
5.3.2	Using Shift Registers to Remember Data Values from Previous Loop Iterations 244	
5.3.3	Initializing Shift Registers 246	
5.3.4	Feedback Nodes 248	
5.4	Case Structures 251	
5.4.1	Adding and Deleting Cases 254	
5.4.2	Wiring Inputs and Outputs 256	

5.5	Flat Sequence Structures	262
5.5.1	Evaluate and Control Timing in a Sequence Structure	263
5.5.2	Avoid the Overuse of Sequence Structures	265
5.6	The Formula Node	265
5.6.1	Formula Node Input and Output Variables	266
5.6.2	Formula Statements	266
5.7	Diagram Disable Structures	269
5.8	Local Variables	270
5.8.1	Creating Local Variables	270
5.8.2	Use Local Variables With Care	272
5.8.3	Initializing Local Variables	273
5.8.4	Memory and Execution-Speed Considerations	273
5.9	Common Programming Techniques	273
5.9.1	Sequential Programming	273
5.9.2	State Programming and State Machines	275
5.9.3	Parallelism	279
5.10	Some Common Problems in Wiring Structures	280
5.10.1	Failing to Wire a Tunnel in All Cases of a Case Structure	280
5.10.2	Overlapping Tunnels	281
5.10.3	Wiring Underneath Rather Than through a Structure	281
5.11	Building Blocks: Pulse Width Modulation	282
5.12	Relaxed Reading: Refining the Process of Steel Recycling	286
5.13	myDAQ Building Blocks	288
5.14	Summary	290
	<i>Exercises</i>	<i>292</i>
	<i>Problems</i>	<i>296</i>
	<i>Design Problems</i>	<i>301</i>
6	Arrays and Clusters	305
6.1	Arrays	306
6.1.1	Creating Array Controls and Indicators	307
6.1.2	Multidimensional Arrays	309
6.2	Creating Arrays with Loops	310
6.2.1	Creating Two-Dimensional Arrays	313
6.3	Array Functions	314
6.3.1	Array Size	314
6.3.2	Initialize Array	315

6.3.3	Build Array	317
6.3.4	Array Subset	318
6.3.5	Index Array	319
6.4	Polymorphism	327
6.5	Clusters	330
6.6	Creating Cluster Controls and Indicators	331
6.6.1	Cluster Order	333
6.6.2	Using Clusters to Pass Data to and from SubVIs	335
6.7	Cluster Functions	336
6.7.1	The Bundle Function	336
6.7.2	The Unbundle Function	340
6.7.3	Creating Cluster Constants on the Block Diagram	340
6.7.4	Using Polymorphism with Clusters	342
6.8	Matrix Data Type and Matrix Functions	343
6.8.1	Creating Matrix Controls, Indicators, and Constants	344
6.8.2	Matrix Functions	345
6.9	VI Memory Usage	348
6.10	Building Blocks: Pulse Width Modulation	349
6.11	Relaxed Reading: Automatic Laser-Assisted Neuron Growth	352
6.12	myDAQ Building Blocks	355
6.13	Summary	357
	<i>Exercises</i>	<i>359</i>
	<i>Problems</i>	<i>362</i>
	<i>Design Problems</i>	<i>365</i>

7 Charts and Graphs

369

7.1	Waveform Charts	370
7.2	Waveform Graphs	378
7.3	XY Graphs	386
7.4	Customizing Charts and Graphs	389
7.4.1	Axes Scaling	389
7.4.2	The Plot Legend	393
7.4.3	The Graph Palette and Scale Legend	395
7.4.4	Special Chart Customization Features	398
7.4.5	Special Graph Customization Features: Cursor Legend	400

- 7.4.6 Using Graph Annotations 402
- 7.4.7 Exporting Images of Graphs, Charts, and Tables 404
- 7.4.8 Using Context Help 404
- 7.5 Using Math Plots for 2D and 3D Graphs 405**
 - 7.5.1 2D Graphs 405
 - 7.5.2 3D Graphs 407
- 7.6 Building Blocks: Pulse Width Modulation 412**
- 7.7 Relaxed Reading: Environmental Monitoring in the Costa Rican Rain Forest 414**
- 7.8 myDAQ Building Blocks 417**
- 7.9 Summary 419**
 - Exercises 420*
 - Problems 422*
 - Design Problems 424*

8 Data Acquisition

427

- 8.1 Components of a DAQ System 428**
- 8.2 Types of Signals 429**
 - 8.2.1 Digital Signals 431
 - 8.2.2 Analog DC Signals 432
 - 8.2.3 Analog AC Signals 433
 - 8.2.4 Analog Frequency-Domain Signals 434
 - 8.2.5 One Signal—Five Measurement Perspectives 436
- 8.3 Common Transducers and Signal Conditioning 437**
- 8.4 Signal Grounding and Measurements 441**
 - 8.4.1 Signal Source Reference Configuration 441
 - 8.4.2 Measurement System 442
- 8.5 Analog-to-Digital Conversion Considerations 447**
- 8.6 DAQ VI Organization 452**
- 8.7 Choosing Your Data Acquisition Device 453**
 - 8.7.1 X Series Data Acquisition Devices 454
 - 8.7.2 Low Cost Data Acquisition for Students 454
 - 8.7.3 Simulated Data Acquisition 454
 - 8.7.4 Macintosh, Linux, and Mobile Devices 455
- 8.8 DAQ Hardware Configuration 456**
 - 8.8.1 Windows 456
 - 8.8.2 Channels and Tasks 463

- 8.9 Using the DAQ Assistant 466**
 - 8.9.1 DAQmx Task Name Constant 469
- 8.10 Analog Input 473**
 - 8.10.1 Task Timing 473
 - 8.10.2 Task Triggering 474
- 8.11 Analog Output 478**
 - 8.11.1 Task Timing 479
 - 8.11.2 Task Triggering 480
- 8.12 Digital Input and Output 485**
- 8.13 Building Blocks: Pulse Width Modulation 491**
 - 8.13.1 Generating Pulse Width Modulated Signals with Hardware Counters 492
 - 8.13.2 Applications of Pulse Width Modulation 493
- 8.14 Relaxed Reading: Reducing Seismic Risk for an Ancient Roman Amphitheater 493**
- 8.15 myDAQ Building Blocks 495**
- 8.16 Summary 498**
 - Exercises 501*
 - Problems 503*
 - Design Problems 503*

9 Strings and File I/O 505

- 9.1 Strings 506**
 - 9.1.1 Converting Numeric Values to Strings with Build Text Express VI 513
- 9.2 File I/O 514**
 - 9.2.1 Writing Data to a File 518
 - 9.2.2 Reading Data from a File 521
 - 9.2.3 Manipulating Spreadsheet Files 522
 - 9.2.4 File I/O Express VIs 525
 - 9.2.5 Obtaining the Path to the System Directories 530
- 9.3 Building Blocks: Pulse Width Modulation 531**
- 9.4 Relaxed Reading: On the Science of Cycling Speed 533**
- 9.5 myDAQ Building Blocks 535**
- 9.6 Summary 537**
 - Exercises 538*
 - Problems 540*
 - Design Problems 541*

10 NI LabVIEW MathScript RT Module 544

- 10.1 **What is MathScript RT Module?** 545
- 10.2 **Accessing the MathScript Interactive Window** 546
 - 10.2.1 The Command History and Output Windows 548
 - 10.2.2 Viewing Data in a Variety of Formats 549
- 10.3 **MathScript Help** 552
- 10.4 **Syntax** 554
 - 10.4.1 Key MathScript Functions 563
- 10.5 **Defining Functions and Creating Scripts** 564
 - 10.5.1 User-Defined Functions 565
 - 10.5.2 Scripts 568
- 10.6 **Saving, Loading, and Exporting Data Files** 572
 - 10.6.1 Saving and Loading Data Files 572
 - 10.6.2 Exporting Data 574
- 10.7 **MathScript Nodes** 575
 - 10.7.1 Accessing the MathScript Node 577
 - 10.7.2 Entering Scripts into the MathScript Node 577
 - 10.7.3 Input and Output Variables 578
 - 10.7.4 Script Highlighting 582
 - 10.7.5 Debugging Scripts 583
 - 10.7.6 Saving Scripts from within the MathScript Node 590
- 10.8 **Applications of MathScript RT Module** 592
 - 10.8.1 Instrument Your Algorithms 592
 - 10.8.2 Graphical Signal Processing, Analysis, and Mathematics 593
 - 10.8.3 Integrating Measurement Hardware 594
- 10.9 **Building Blocks: Pulse Width Modulation** 595
- 10.10 **Relaxed Reading: Acquiring and Analyzing the Bioacoustic Communication of Killer Whales** 597
- 10.11 **myDAQ Building Blocks** 599
- 10.12 **Summary** 601
 - Exercises* 603
 - Problems* 604
 - Design Problems* 605

11 Analysis 607

- 11.1 **Linear Algebra** 608
 - 11.1.1 Review of Matrices 608
 - 11.1.2 Systems of Algebraic Equations 613
 - 11.1.3 Linear System VIs 615

11.2	Statistics and Curve Fitting	618
11.2.1	Curve Fits Based on Least Squares Methods	619
11.2.2	Fitting a Curve to Data with Normal Distributions	623
11.2.3	The Curve Fitting Express VI	625
11.3	Differential Equations	629
11.4	Finding Zeroes of Functions	637
11.5	Integration and Differentiation	640
11.6	Signal Generation	641
11.6.1	Normalized Frequency	642
11.6.2	Wave, Pattern, and Noise VIs	646
11.6.3	The Simulate Signal Express VI	649
11.7	Signal Processing	651
11.7.1	The Fourier Transform	651
11.7.2	Smoothing Windows	655
11.7.3	The Spectral Measurements Express VI	660
11.7.4	Filtering	663
11.7.5	The Filter Express VI	672
11.8	Building Blocks: Pulse Width Modulation	675
11.9	Relaxed Reading: High-Speed Control System To Test MEMs Microshutters	677
11.10	myDAQ Building Blocks	679
11.11	Summary	682
	<i>Exercises</i>	<i>684</i>
	<i>Problems</i>	<i>685</i>
	<i>Design Problems</i>	<i>686</i>

A Instrument Control 687

A.1	Components of an Instrument Control System	688
A.1.1	What Is GPIB?	688
A.1.2	GPIB Messages	689
A.1.3	GPIB Devices and Configurations	691
A.1.4	Serial Port Communication	693
A.1.5	Other Bus Technologies	695
A.2	Detecting and Configuring Instruments	696
A.2.1	Windows	696
A.2.2	Macintosh OS X	696
A.3	Using the Instrument I/O Assistant	699
A.4	Instrument Drivers	707
A.4.1	Developing Your Own Instrument Driver	713

A.5	Future of Instrument Drivers and Instrument Control	714
A.6	Summary	715
B	LabVIEW Developer Certification	717
B.1	Overview of the NI LabVIEW Certification Structure	718
B.2	Logistics of the CLAD Examination	718
B.3	Benefits of CLAD	719
B.4	Sample CLAD Examination	720
B.5	Detailed Sample CLAD Test Solutions	731
B.6	Additional Study Resources	735
B.7	Summary	736
	<i>Index</i>	<i>738</i>

This page intentionally left blank



P R E F A C E

Learning with LabVIEW™ is the textbook that accompanies the *LabVIEW Student Edition* from National Instruments, Inc. This textbook, as well as the LabVIEW software, has undergone a significant revision from the previous edition. *Learning with LabVIEW* teaches basic programming concepts in a graphical environment and relates them to real-world applications in academia and industry. Understanding and using the intuitive and powerful LabVIEW software is easier than ever before. As you read through the book and work through the examples, we hope you will agree that this book is more of a personal tour guide than a software manual.

The LabVIEW graphical development environment was built specifically for applications in engineering and science, with built-in functionality designed to reduce development time for design and simulation in signal processing, control, communications, electronics and more. The *LabVIEW Student Edition* delivers all the capabilities of the full version of LabVIEW, widely considered the industry standard for design, test, measurement, automation, and control applications. With LabVIEW, students can design graphical programming solutions to their homework problems and laboratory experiments—an ideal tool for science and engineering applications—that is also fun to use! The *LabVIEW Student Edition* affords students the opportunity for self-paced learning and independent project development.

The goal of this book is to help students learn to use LabVIEW on their own. With that goal in mind, this book is very art-intensive with over 400 figures in all. That means that there are numerous screen captures in each section taken from a typical LabVIEW session. The figures contain additional labels and pointers added to the LabVIEW screen captures to help students understand what they are seeing on their computer screens as they follow along in the book.

The most effective way to use *Learning with LabVIEW* is to have a concurrent LabVIEW session in progress on your computer and to follow along with the steps in the book. A directory of virtual instruments has been developed by the author exclusively for use by students using *Learning with LabVIEW* and is available on www.pearsonhighered.com/bishop. These virtual instruments

complement the material in the book. In most situations, the students are asked to develop the virtual instrument themselves following instructions given in the book, and then compare their solutions with the solutions provided by the author to obtain immediate feedback. In other cases, students are asked to run a specified virtual instrument as a way to demonstrate an important LabVIEW concept.

GAINING PRACTICAL EXPERIENCE AND SOLVING REAL-WORLD PROBLEMS

With higher education emphasizing hands-on laboratory experience, many educational institutions have improved their laboratory facilities in order to increase student exposure to practical problems. College graduates are gaining vital experience in acquiring and analyzing data, constructing computer-based simulations of physical systems, and multipurpose computer programming. LabVIEW offers a powerful, efficient, and easy-to-use development environment, allowing educators to teach their students a wide range of topics with just one open, industry-standard tool. It can also transform the way engineers, scientists, and students around the world design, prototype, and deploy cutting-edge technology. Customers and students at more than 25,000 companies and schools are using LabVIEW and modular hardware from National Instruments to simplify technology development and increase productivity. From testing next-generation gaming systems to creating breakthrough medical devices, the resulting innovative technologies are impacting millions of people worldwide.

The cover of this edition of *Learning with LabVIEW* shows ten interesting application areas that use LabVIEW in the solution process.

1. Marine Mammals
2. Cycling Speed
3. NASA Microshutters
4. Cancer Lab
5. Roman Amphitheater
6. Nucor Steel Plant
7. La Selva Rainforest
8. ARV Drone
9. Tumor Treatment
10. Autonomous Vehicle



1. Marine Mammals

LabVIEW was used to develop a reliable data acquisition system that can collect and analyze ultrasonic signals produced by killer whales to perform

live audio monitoring and movement tracking to study how the marine mammals use echolocation in their natural habitat.

2. Cycling Speed

Using LabVIEW, NI hardware, and off-the-shelf components, a unique wind tunnel was created to conduct continuous loops of development and testing for cyclists and their bikes by evaluating and optimizing their riding positions.

3. NASA Microshutters

The challenge of synchronizing the motion of a magnet moving more than 1 meter per second with the opening and closing of tens of thousands of tiny microelectromechanical system microshutters was addressed for application on the James Webb space telescope to enable the next stepping-stone toward understanding the universe.

4. Cancer Lab

LabVIEW helps advance a unique and more accurate form of cancer treatment where accelerated charged particles can precisely target deep-seated cancer cells opening new perspectives for patients with respect to surgery or traditional radiotherapy.

5. Roman Amphitheater

Building a distributed monitoring system to identify structural damage and reduce seismic risk of the Arena di Verona—one of the largest open-air opera theaters in the world built in the first century—involved creating a data acquisition system to communicate with existing measurement sensors in the theater that continuously monitors signals from sensor devices, tracks changes in a crack and the onset of structural vibrations related to environmental changes.

6. Nucor Steel Plant

Employing LabVIEW programmable automation controllers has led to a tenfold increase in efficiency and drastically reduced the facility automation costs for steel recycling that reduces the amount of energy consumed while improving safety for workers.

7. La Selva Rainforest

Employing a wide range of wireless environmental measurements using a single device that provides robotic control, remote configuration, and data sharing over the Web, researchers in Costa Rica are testing a hypothetical explanation of the exchange of CO₂ and other materials between the forest floor and the atmosphere—the so-called “gap theory.”

8. ARV Drone

A unique LabVIEW application is used to control an unmanned aerial vehicle, acquire the images taken by its cameras, and process the images with NI vision hardware in areas only viewable and accessible by air for fast

detection of casualties in difficult-to-access areas, such as earthquake and collapsed-building sites.

9. Tumor Treatment

Using NI hardware and software, a new medical device was created to treat breast tumors in a less invasive and nearly painless procedure, dramatically reducing the emotional and physical discomfort of patients undergoing tumor treatment.

10. Autonomous Vehicle

A semiautonomous vehicle was developed using LabVIEW software and hardware that allows a blind driver to successfully navigate, control speed, and avoid collision through a secure driving course.

THE LABVIEW STUDENT EDITION SOFTWARE

The *LabVIEW Student Edition* software package is a powerful and flexible instrumentation, analysis, and control software platform for PCs running Microsoft Windows or Apple Macintosh OS X when installed using the MacOS Boot Camp application or a virtual machine environment. The student edition is designed to give students early exposure to the many uses of graphical programming. LabVIEW not only helps reinforce basic scientific, mathematical, and engineering principles, but it encourages students to explore advanced topics as well. Students can run LabVIEW programs designed to teach a specific topic, or they can use their skills to develop their own applications. LabVIEW provides a real-world, hands-on experience that complements the entire learning process.

WHAT'S NEW WITH THE LABVIEW STUDENT EDITION?

The demand for LabVIEW in colleges and universities has led to the development of *LabVIEW Student Edition* based on the industry version of LabVIEW. This is a new and significant software revision that delivers all of the graphical programming capabilities of the full edition. With the student edition, students can design graphical programming solutions for their classroom problems and laboratory experiments on their personal computers. The *LabVIEW Student Edition* features include the following:

- Express VIs that bring interactive, configuration-based application design for acquiring, analyzing, and presenting data.
- Interactive measurement assistants to make creating data acquisition and instrument control applications easier than ever.
- Full LabVIEW advanced analysis capability.

- Full compatibility with all National Instruments data acquisition and instrument control hardware.
- Support for all data types used in the LabVIEW Full Development System.

New LabVIEW software features introduced in this new edition of *Learning with LabVIEW* include:

- Updated editing and manipulations of wires including Owned Labels and Align/Distribute Wires, and you can now wire an error cluster directly to Boolean functions.
- Enhancements in creating subVIs including Create SubVI from Selection that automatically builds the connector pane and front panel of the subVI; and the icon and connector pane that is always visible for easier subVI wiring and icon editing.
- New editing features including new ways to view Control and Function palettes, the ability to change parameters of multiple objects at one time, and you can now right-click a cluster constant and select View Cluster As Icon from the shortcut menu to reduce the size of cluster constants on the block diagram.
- Improved programming constructs including automatically concatenating arrays leaving loops and conditionally processing loop outputs.
- A redesigned Getting Started window to allow ready access to the resources you need to use LabVIEW emphasizing common tasks, such as creating projects and opening existing files, and presents less common tasks, such as downloading additional drivers and products, through submenus.

This latest edition of *Learning with LabVIEW* also features:

- A myDAQ Building Blocks section in each chapter to provide the opportunity to apply the concepts introduced in the chapter using the myDAQ hardware.
- New relaxed readings that illustrate how students, engineers, and scientists are using LabVIEW to solve real-world problems.
- Information on how to become a certified LabVIEW user for career advancement and employment opportunities.

ORGANIZATION OF LEARNING WITH LABVIEW

This textbook serves as a LabVIEW resource for students. The pace of instruction is intended for both undergraduate and graduate students. The book is comprised of 11 chapters and should be read sequentially when first learning LabVIEW. For more experienced students, the book can be used as a reference

book by using the index to find the desired topics. The 11 chapters are as follows:

CHAPTER 1: LabVIEW Basics—This chapter introduces the LabVIEW environment and helps orient students when they open a virtual instrument. Concepts such as windows, toolbars, menus, and palettes are discussed.

CHAPTER 2: Virtual Instruments—The components of a virtual instrument are introduced in this chapter: front panel, block diagram, and icon/connector pair. This chapter also introduces the concept of controls (inputs) and indicators (outputs) and how to wire objects together in the block diagram. Express VIs are introduced in the chapter.

CHAPTER 3: Editing and Debugging Virtual Instruments—Resizing, coloring, and labeling objects are just some of the editing techniques introduced in this chapter. Students can find errors using execution highlighting, probes, single-stepping, and breakpoints, just to name a few of the available debugging tools.

CHAPTER 4: SubVIs—This chapter emphasizes the importance of reusing code and illustrates how to create a VI icon/connector. It also shows parallels between LabVIEW and text-based programming languages.

CHAPTER 5: Structures—This chapter presents loops, case structures, and flat sequence structures that govern the execution flow in a VI. The Formula Node is introduced as a way to implement complex mathematical equations.

CHAPTER 6: Arrays and Clusters—This chapter shows how data can be grouped, either with elements of the same type (arrays) or elements of a different type (clusters). This chapter also illustrates how to create and manipulate arrays and clusters.

CHAPTER 7: Charts and Graphs—This chapter shows how to display and customize the appearance of single and multiple charts and graphs.

CHAPTER 8: Data Acquisition—The basic characteristics of analog and digital signals are discussed in this chapter, as well as the factors students need to consider when acquiring and generating these signals. This chapter introduces students to the Measurement and Automation Explorer (MAX) and the DAQ Assistant.

CHAPTER 9: Strings and File I/O—This chapter shows how to create and manipulate strings on the front panel and block diagram. This chapter also explains how to write data to and read data from files.

CHAPTER 10: NI LabVIEW MathScript RT Module—This chapter introduces the interactive MathScript environment, which combines a mathematics-oriented text-based language with the intuitive graphical data-flow programming of LabVIEW. Both the interactive MathScript

environment for command line computation and the MathScript Node for integrating textual scripts within the LabVIEW block diagram are discussed.

CHAPTER 11: Analysis—LabVIEW can be used in a variety of ways to support analysis of signals and systems. Several important analysis topics are discussed in this chapter, including how to use LabVIEW for signal generation, signal processing, linear algebra, curve fitting, formula display on the front panel, differential equations, finding roots (zero finder), and integration and differentiation.

APPENDIX A: Instrument Control—The components of an instrument control system using a GPIB or serial interface are presented in this appendix. Students are introduced to the notion of instrument drivers and of using the Measurement and Automation Explorer (MAX) to detect and install instrument drivers. The Instrument I/O Assistant is introduced.

APPENDIX B: LabVIEW Developer Certification—Discusses the certification process to validate your expertise, beginning with the Certified LabVIEW Associate Developer (CLAD), continuing with the Certified LabVIEW Developer (CLD), and culminating with the Certified LabVIEW Architect (CLA). It includes a CLAD introductory-level certification practice test with complete answers, along with information on additional resources to help you prepare for the examination.

The important pedagogical elements in each chapter include the following:

1. A brief table of contents and a short preview of what to expect in the chapter.
2. A list of chapter goals to help focus the chapter discussions.
3. Margin icons that focus attention on a helpful hint or on a cautionary note.



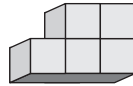
4. An end-of-chapter summary and list of key terms.



KEY TERMS

5. Sections entitled **Building Blocks** near the end of each chapter present the continuous development and modification of a virtual instrument for calculating and generating a pulse-width modulated signal. The student is expected to construct the VIs based on the instructions given in the sections. The same VI is used as the starting point and then improved in each

subsequent chapter as a means for the student to practice with the newly introduced chapter concepts.

**BUILDING BLOCK**

6. At the end of each chapter, we include a **myDAQ Building Blocks** section to provide the opportunity to apply the concepts introduced in the chapter using the myDAQ hardware. Each myDAQ Building Blocks exercise will help you build a project that will involve controlling LEDs and reading temperature from a thermistor to create a temperature monitoring system. As with the Building Blocks, the same VI is used as the starting point and then improved in each subsequent chapter as a means for the student to apply the chapter concepts in a my DAQ hardware environment.



7. Many worked examples are included in each chapter including several new examples introduced in this edition. In most cases, students construct the VIs discussed in the examples by following a series of instructions given in the text. In the early chapters, the instructions for building the VIs are quite specific, but in the later chapters, students are expected to construct the VIs without precise step-by-step instructions. Of course, in all chapters, working versions of the VIs are provided for all examples in the **Learning** directory included as part of the *LabVIEW Student Edition*. Here is a sample of the worked examples:

- Temperature system demonstration.
- Solving a set of linear differential equations.
- Building your first virtual instrument.
- Computing area, diameter, and radius of a circle.
- Computing and graphing the time value of money.
- Studying chaos using the logistic difference equation.
- Acquiring data.
- Writing ASCII data to a file.

8. A section entitled *Relaxed Reading* that describes how LabVIEW is being utilized to solve interesting real-world problems. The material is intended to give students a break from the technical aspects of learning LabVIEW and to stimulate thinking about how LabVIEW can be used in various other situations.
9. End-of-chapter exercises, problems, and design problems reinforce the main topics of the chapter and provide practice with LabVIEW.

ORIGINAL SOURCE MATERIALS

Learning with LabVIEW was developed with the aid of important references provided by National Instruments. The main references are the various LabVIEW help manuals found at the website www.ni.com/manuals. They provide information on LabVIEW programming concepts, step-by-step instructions for using LabVIEW, and reference information about LabVIEW VIs, functions, palettes, menus, and tools. You can access this same material in LabVIEW by selecting **Help»LabVIEW Help** (see Chapter 1 of this book for more details on accessing the LabVIEW help). By design, there is a strong correlation between some of the material contained in the various LabVIEW help manuals and the material presented in this book. Our goal here has been to refine the information content and make it more accessible to students learning LabVIEW on their own.

OPERATING SYSTEMS AND ADDITIONAL SOFTWARE

It is assumed that the reader has a working knowledge of either the Windows or the Mac OS X operating system. If your computer experience is limited, you may first want to spend some time familiarizing yourself with your computer in order to understand the operation of your Mac or PC. You should know how to access pull-down menus, open and save files, download software from the Internet, and use a mouse. You will find previous computer programming experience helpful—but not necessary.

A set of virtual instruments has been developed by the author for this book. You will need to obtain the Learning directory from the companion website to this book at Prentice Hall:

<http://www.pearsonhighered.com/bishop>

For more information, you may also want to visit the NI Student Edition website at

<http://www.ni.com/labviewse>

All of the VI examples in this book were tested by the author on an HP Elite-Book running Windows 7. Obviously, it is not possible to verify each VI on all the available Windows and Macintosh platforms that are compatible with LabVIEW so if you encounter platform-specific difficulties, please let us know.

If you would like information on upgrading to the LabVIEW Professional Version, please write to

National Instruments
att.: Academic Sales
11500 North Mopac Expressway
Austin, TX 78759

or visit the National Instruments website: <http://www.ni.com>

LIMITED WARRANTY

The software and the documentation are provided “as is,” without warranty of any kind, and no other warranties, either expressed or implied, are made with respect to the software. National Instruments does not warrant, guarantee, or make any representations regarding the use, or the results of the use, of the software or the documentation in terms of correctness, accuracy, reliability, or otherwise and does not warrant that the operation of the software will be uninterrupted or error-free. This software is not designed with components and testing for a level of reliability suitable for use in the diagnosis and treatment of humans or as critical components in any life-support systems whose failure to perform can reasonably be expected to cause significant injury to a human. National Instruments expressly disclaims any warranties not stated herein. Neither National Instruments nor Pearson Education shall be liable for any direct or indirect damages. The entire liability of National Instruments and its dealers, distributors, agents, or employees are set forth above. To the maximum extent permitted by applicable law, in no event shall National Instruments or its suppliers be liable for any damages, including any special, direct, indirect, incidental, exemplary, or consequential damages, expenses, lost profits, lost savings, business interruption, lost business information, or any other damages arising out of the use, or inability to use, the software or the documentation even if National Instruments has been advised of the possibility of such damages.

ACKNOWLEDGMENTS

Thanks to all the folks at National Instruments for their assistance and input during the development of *Learning with LabVIEW*. A very special thanks to Gretchen Edelman and Jayme Walton of NI for providing day-to-day support

during the final months of the project. Thanks also go to the following reviewers: Austin B. Asgill, Southern Polytechnic State University; Jeff Doughty, Northeastern University; Buford Furman, San Jose State University; R. Glynn Holt, Boston University; Thomas Koon, Binghamton University; Milivoje Kostic, Northern Illinois University; Jay Porter, Texas A&M University; and Yi Wu, Penn State University. Finally, I wish to express my appreciation to Lynda Bishop for assisting me with the manuscript preparation, for providing valuable comments on the text, and for handling my personal day-to-day activities associated with the entire production.

KEEP IN TOUCH!

The author and the staff at Pearson and at National Instruments would like to establish an open line of communication with the users of the *LabVIEW Student Edition*. We encourage students to e-mail the author with comments and suggestions for this and future editions.

Keep in touch!

ROBERT H. BISHOP
robert.bishop@marquette.edu

GRETCHEN EDELMON
Senior Manager, Academic Courseware Systems
gretchen.edelmon@ni.com

This page intentionally left blank

Learning with LabVIEW™

This page intentionally left blank



CHAPTER 1

LabVIEW Basics

1.1	System Configuration Requirements	2
1.2	Installing the <i>LabVIEW Student Edition</i>	2
1.3	The LabVIEW Environment	3
1.4	The Getting Started Screen	5
1.5	Panel and Diagram Windows	9
1.6	Shortcut Menus	15
1.7	Pull-Down Menus	16
1.8	Palettes	24
1.9	Opening, Loading, and Saving VIs	31
1.10	LabVIEW Help Options	33
1.11	Building Blocks: Pulse Width Modulation	36
1.12	Relaxed Reading: Revolutionary Cancer Treatment Minimizes Damage to Healthy Tissue	38
1.13	myDAQ Building Blocks	40
1.14	Summary	43

Welcome to the *LabVIEW Student Edition*! **LabVIEW** is a powerful and complex programming environment. Once you have mastered the various concepts introduced in this book you will have the ability to develop applications in a graphical programming language and to develop virtual instruments for design, control, and test engineering. This introductory chapter provides a basic overview of LabVIEW and its components.

GOALS

1. Installation of the *LabVIEW Student Edition*.
2. Familiarization with the basic components of LabVIEW.
3. Introduction to front panels and block diagrams, shortcut and pull-down menus, palettes, VI libraries, and online help.

1.1 SYSTEM CONFIGURATION REQUIREMENTS

The *LabVIEW Student Edition* textbook bundle includes the LabVIEW Student Edition software for Windows 8.1/8/7/Vista/XP and Mac OS X, when installed using the MacOS Boot Camp application or a virtual machine environment. This textbook without software is available under the name *Learning with LabVIEW*.

Windows 8.1/8/7/Vista (32 bit and 64 bit), Windows XP SP3 (32 bit)

Processor:	Pentium III/Celeron 866 MHz or equivalent minimum Pentium 4/M or equivalent recommended
RAM:	256 MB minimum; 1 GB recommended
Screen Resolution:	1024 × 768 pixels
Operating System:	Windows 7/Vista/XP
Disk Space:	3.67 GB (includes default drivers from the NI Device Drivers DVD)

Macintosh OS X 10.3 or later

Windows Application:	Mac OS Boot Camp, virtual machine application, or Windows emulator
Processor:	Intel-based processor
RAM:	256 MB minimum; 1 GB recommended
Screen Resolution:	1024 × 768 pixels
Operating System:	Mac OS X (10.3 or later)
Disk Space:	1.2 GB on the Windows partition (for the complete installation excluding drivers)

1.2 INSTALLING THE *LABVIEW STUDENT EDITION*



Disable any automatic virus detection programs before you install. Some virus detection programs interfere with the installation program.

Windows

1. Log on as an administrator or as a user with administrator privileges.
2. Download and install LabVIEW using the detailed instructions found on this webpage: from <http://www.ni.com/white-paper/13413/en/>. If you plan

to use NI data acquisition hardware such as NI myDAQ, be sure to download and install the hardware drivers as described in step 3 of the installation instruction page. Use the serial number printed on the inside cover to activate the software.

3. Detailed instructions for activation are included in the installation instructions as well as on this page: <http://www.ni.com/white-paper/14146/en/>.
4. After installation, check your hard disk for viruses and enable any virus detection programs you disabled.
5. To use the *LabVIEW Help*, the Measurement & Automation Explorer (MAX) interactive help system, and the NI Example Finder, you must have Microsoft Internet Explorer 5.0 or later.

LabVIEW relies on a licensing activation process to obtain an activation code. The Activation Wizard is a part of the NI License Manager that takes you through the steps of enabling software to run on a machine. If you have questions regarding activation, visit the NI website at <http://www.ni.com/labviewse/>.

Macintosh

1. To install the downloaded LabVIEW Student Edition software on your Mac Computer, you will need a Mac Computer that supports the Mac OS Boot Camp application or a virtual machine environment, and a licensed copy of the Microsoft Windows operating system.
2. Boot the Mac OS computer into Windows or launch the Windows virtual machine.
3. Follow the instructions for Windows installation in the section above.



The LabVIEW installation folder might contain the following items:

- *cintools*—Tools for calling C object code from LabVIEW
- *AppLibs*—(LabVIEW Professional Development System). Contains stub for stand-alone applications.

Refer to LabVIEW Help for a complete list of the folders installed in LabVIEW directory structure.

1.3 THE LABVIEW ENVIRONMENT

LabVIEW is short for **L**aboratory **V**irtual **I**nstrument **E**ngineering **W**orkbench. It is a powerful and flexible graphical development environment created by the folks at National Instruments—a company that creates hardware and software

products that leverage computer technology to help engineers and scientists take measurements, control processes, and analyze and store data. National Instruments was founded over thirty-five years ago in Austin, Texas by James Truchard (known as Dr. T), Jeffrey Kodosky, and William Nowlin. At the time, all three were working on sonar applications for the U.S. Navy at the Applied Research Laboratories at The University of Texas at Austin. Searching for a way to connect test equipment to DEC PDP-11 computers, Dr. T decided to develop an interface bus. He recruited Jeff and Bill to join him in his endeavor, and together they successfully developed LabVIEW and the notion of a “virtual instrument.” In the process they managed to infuse their new company—National Instruments—with an entrepreneurial spirit that still pervades the company today.

Engineers and scientists in research, development, production, test, and service industries as diverse as automotive, semiconductor, aerospace, electronics, chemical, telecommunications, and pharmaceutical have used and continue to use LabVIEW to support their work. LabVIEW is a major player in the area of testing and measurements, industrial automation, and data analysis. For example, SpaceX uses LabVIEW in a variety of mission applications, including the Falcon 9 rocket that carried the Dragon spacecraft into space. The Dragon is a free-flying reusable spacecraft that was the first commercially built vehicle to dock with the International Space Station and demonstrate a cargo resupply mission. This book is intended to help you learn to use LabVIEW as a programming tool and to serve as an introduction to the power of graphical programming and the myriad applications to which it can be applied.

LabVIEW programs are called **Virtual Instruments**, or VIs for short. LabVIEW is different from text-based programming languages (such as Fortran and C) in that LabVIEW uses a graphical programming language, known as the G programming language, to create programs relying on graphic symbols to describe programming actions. LabVIEW uses a terminology familiar to scientists and engineers, and the graphical icons used to construct programs in G are easily identified by visual inspection. You can learn LabVIEW even if you have little programming experience, but you will find knowledge of programming fundamentals helpful. If you have never programmed before (or maybe you have programming experience but have forgotten a few things) you may want to review the basic concepts of programming before diving into the G programming language.

LabVIEW provides an extensive library of virtual instruments and functions to help you in your programming. The MathScript RT Module environment is discussed in detail in Chapter 10. The MathScript RT Module provides a text-based command line environment complementing the LabVIEW graphical programming environment. Users have the capability to make quick calculations or computations at a LabVIEW command line prompt. The MathScript RT Module also enables users to integrate their scripts with LabVIEW block diagrams,

easily mixing graphical programming and powerful user interfaces with text-based scripts. LabVIEW contains application-specific libraries for data acquisition (discussed in Chapter 8), file input/output (discussed in Chapter 9), and data analysis (discussed in Chapter 11). It includes conventional program debugging tools with which you can set breakpoints, single-step through the program, and animate the execution so you can observe the flow of data. Editing and debugging VIs is the topic of Chapter 3. LabVIEW has a set of VIs for data presentation on various types of charts and graphs. Chapter 7 discusses the process of presenting data on charts and graphs.

The LabVIEW system consists of LabVIEW application executable files and many associated files and folders. LabVIEW uses files and directories to store information necessary to create your VIs. Some of the more important files and directories are:

1. The **LabVIEW** executable. Use this to launch LabVIEW.
2. The **vi.lib** directory. This directory contains libraries of VIs such as data acquisition, instrument control, and analysis VIs; it must be in the same directory as the LabVIEW executable. Do not change the name of the **vi.lib** directory, because LabVIEW looks for this directory when it launches. If you change the name, you cannot use many of the controls and library functions.
3. The **examples** directory. This directory contains many sample VIs that demonstrate the functionality of LabVIEW.
4. The **user.lib** directory. This directory is where you can save VIs you have created, and they will appear in the LabVIEW **Functions** palette.
5. The **instr.lib** directory. This directory is where your instrument driver libraries are placed if you want them to appear in the **Functions** palette.
6. The **Learning** directory. This file contains a library of VIs that you will use with the *Learning with LabVIEW* book.



*The files in the **Learning** directory must be downloaded from the site <http://www.pearsonhighered.com/bishop>. You can access the Pearson Higher Education website through the Internet using any standard Web browser.*

1.4 THE GETTING STARTED SCREEN

When you launch LabVIEW by double-clicking on its icon, the Getting Started screen appears as in Figure 1.1. The Getting Started window provides ready access to LabVIEW resources to enhance the ease of use. Creating projects and opening existing files—some of the more common tasks—can be easily

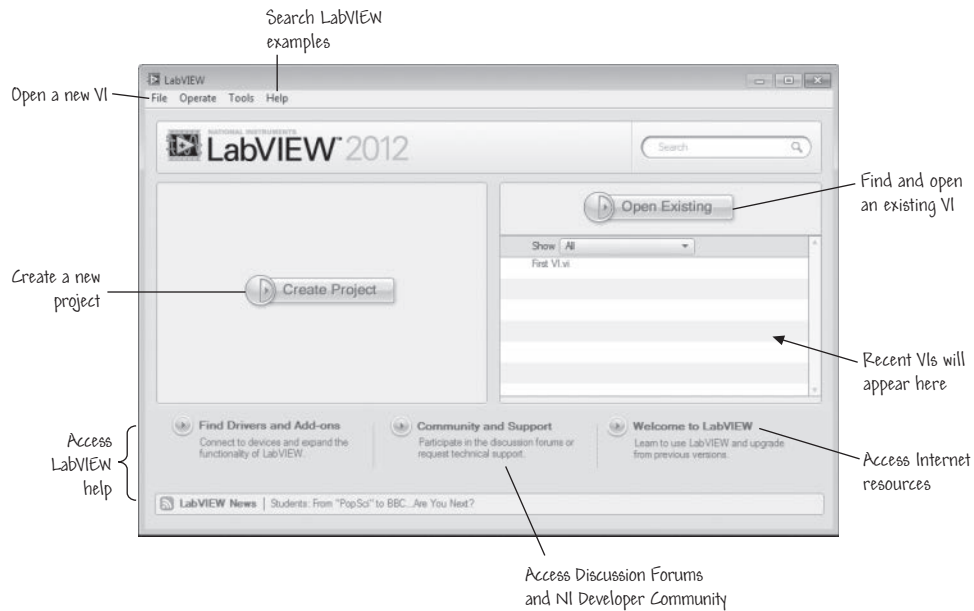


FIGURE 1.1
The Getting Started screen.

accomplished through the use of the handy buttons front and center in the window. Reaching online resources is made much easier by having access to submenus on the Getting Started window that link to NI resources to download additional drivers and products and connect to discussion forums and developer communities. The Getting Started screen contains a navigation dialog box that includes introductory material and common commands. The dialog box includes a menu that allows users to quickly create new projects, select among the most recently opened LabVIEW files, find examples, and search the LabVIEW Help. The upper section of the Getting Started screen contains *file* functions, such as creating a new VI or for opening an existing VI. Across the lower section are *resource* functions. Information and resources are reachable from the Getting Started screen to help you learn about LabVIEW using online manuals and tutorials, with convenient access to Internet links providing news, technical content, example programs, and training information from the National Instruments website. These categories are dynamically populated with a list of articles relating to LabVIEW.

On the Getting Started screen you can:

- Click on **File**»**New** to open the **New** dialog box to create a blank VI or one based on a VI template.
- Click on **Open Existing** to open an existing VI by browsing to find the desired file, or select a VI from the list of recently used VIs.

On the lower section of the screen you can:

- **Find Drivers and Add-ons:** Link to resources to search for NI Device Drivers, connect to the NI Instrument Device Driver to install drivers for connected instruments, and search for LabVIEW add-ons to expand functionality.
- **Community and Support:** Access NI discussion forums and developer communities.
- **Welcome to LabVIEW:** Access resources to help with LabVIEW and learn more about new features as they become available.



Throughout this book, use the left mouse button (if you have one) unless we specifically tell you to use the right one.

Searching the LabVIEW Examples

In this exercise you will search through the list of example VIs and demonstrations that are included with the *LabVIEW Student Edition*. Open the LabVIEW application and get to the Getting Started screen. The search begins at the LabVIEW Getting Started screen by selecting **Find Examples**, as shown in Figure 1.2. The NI Example Finder screen displays the numerous examples available with LabVIEW, as illustrated in Figure 1.3. The examples can be

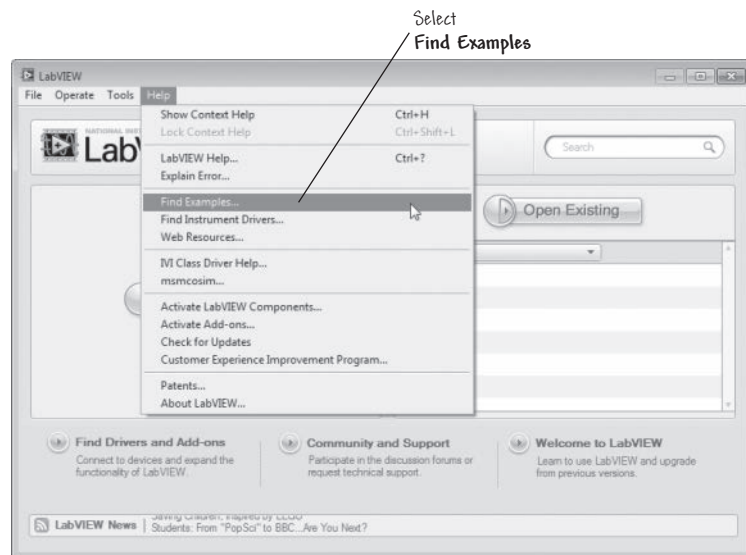


FIGURE 1.2
LabVIEW examples.

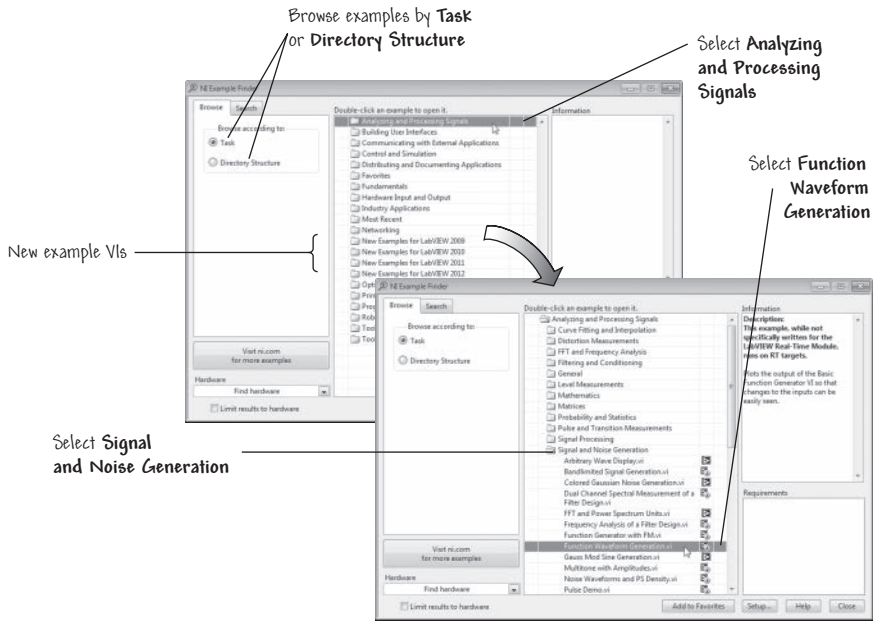


FIGURE 1.3
The NI Example Finder.

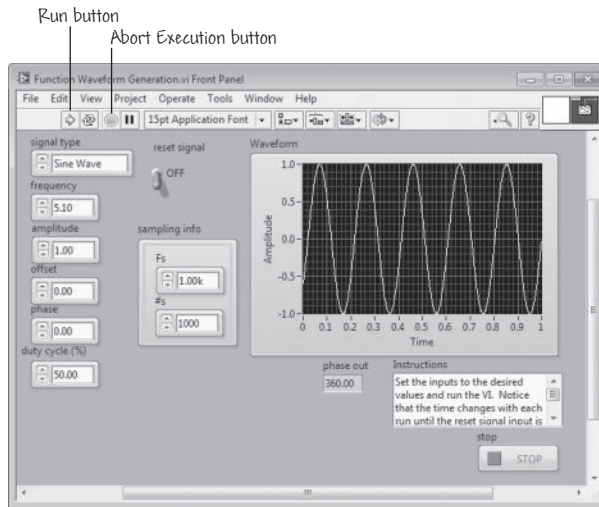


FIGURE 1.4
The Function Waveform Generation front panel.

browsed by Task or by Directory Structure. In Figure 1.3, the examples are sorted by Task. As indicated in Figure 1.3, on the **Browse** tab of the **NI Example Finder** you can find **New Examples for LabVIEW** associated with each new update to LabVIEW. These VIs are examples demonstrating new features of the current version of LabVIEW.

To reach the desired example—in this case, we are searching for the **Function Waveform Generation**—select **Analyzing and Processing Signals**, as shown in Figure 1.3. Selecting **Signal and Noise Generation** and **Function Waveform Generation** opens up the associated virtual instrument (VI) (more on VIs in Chapter 2). Just for fun, you can start the VI running and see what happens. Start the VI by clicking on the **Run** button, as shown in Figure 1.4. Stop the VI by clicking on the **Stop** button. Try it! ◆

1.5 PANEL AND DIAGRAM WINDOWS

An untitled front panel window appears when you select **File»New VI** from the Getting Started screen. The front panel window is the interface to your VI code and is one of the two LabVIEW windows that comprise a virtual instrument. The other window—the block diagram window—contains program code that exists in a graphical form (such as icons, wires, etc.).

Front panels and block diagrams consist of graphical objects that are the G programming elements. Front panels contain various types of controls and indicators (that is, inputs and outputs, respectively). Block diagrams contain terminals corresponding to front panel controls and indicators, as well as constants, functions, subVIs, structures, and wires that carry data from one object to another. Structures are program control elements (such as For Loops and While Loops).

Figure 1.5 shows a front panel and its associated block diagram. You can find the virtual instrument **First VI.vi** shown in Figure 1.5 in the **Chapter 1** folder within the directory **Learning**. That VI can be located by choosing **Open Existing** on the Getting Started screen and navigating to the **Chapter1** folder in the **Learning** directory and then selecting **First VI.vi**. Once you have the VI front panel open, find the **Run** button on the panel toolbar and click on it. Your VI is now running. You can turn the knob and vary the different inputs and watch the output changes reflected in the graph. Give it a try! If you have difficulty getting things to work, then just press ahead with the material in the next sections and come back to this VI when you are ready.

1.5.1 Front Panel Toolbar

A toolbar of command buttons and status indicators that you use for controlling VIs is located on both the front panel and block diagram windows. The front panel toolbar and the block diagram toolbar are different, although they do each contain some of the same buttons and indicators. The toolbar that appears at the top of the front panel window is shown in Figure 1.6.



While the VI is executing, the **Abort Execution** button appears. Although clicking on the abort button terminates the execution of the VI, as a general rule you should avoid terminating the program execution this way and either let the