
LabVIEW Graphical Programming

Fourth Edition

Gary W. Johnson
Richard Jennings

McGraw-Hill

New York Chicago San Francisco Lisbon London
Madrid Mexico City Milan New Delhi San Juan
Seoul Singapore Sydney Toronto

The McGraw-Hill Companies

McGraw-Hill books are available at special quantity discounts to use as premiums and sales promotions, or for use in corporate training programs. For more information, please write to the Director of Special Sales, Professional Publishing, McGraw-Hill, Two Penn Plaza, New York, NY 10121-2298. Or contact your local bookstore.

LabVIEW Graphical Programming, Fourth Edition

Copyright © 2006 by The McGraw-Hill Companies. All rights reserved. Printed in the United States of America. Except as permitted under the Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of publisher, with the exception that the program listings may be entered, stored, and executed in a computer system, but they may not be reproduced for publication.

1234567890 DOC DOC 019876

ISBN 0-07-145146-3

Sponsoring Editor

Wendy Rinaldi

Editorial Supervisor

Jody McKenzie

Project Manager

Samik Roy Chowdhury
(Sam)

Acquisitions Coordinator

Alexander McDonald

Copy Editor

Patti Scott

Proofreader

Prachi Ghildiyal

Indexer

WordCo Indexing
Services, Inc.

Production Supervisor

James Kussow

Composition

International Typesetting
and Composition

Illustration

International Typesetting
and Composition

Art Director, Cover

Margaret Webster-Shapiro

Information has been obtained by McGraw-Hill from sources believed to be reliable. However, because of the possibility of human or mechanical error by our sources, McGraw-Hill, or others, McGraw-Hill does not guarantee the accuracy, adequacy, or completeness of any information and is not responsible for any errors or omissions or the results obtained from the use of such information.

Contents

Preface	xi
Acknowledgments	xv
Chapter 1 Roots	1
LabVIEW and Automation	2
Virtual instruments: LabVIEW's foundation	4
Why use LabVIEW?	7
The Origin of LabVIEW	8
Introduction	9
A vision emerges	9
All the world's an instrument	11
A hard-core UNIX guy won over by the Macintosh	12
Putting it all together with pictures	12
Favoring the underdog platform for system design	15
Ramping up development	15
Stretching the limits of tools and machine	16
Facing reality on estimated development times	18
Shipping the first version	19
Apple catches up with the potential offered by LabVIEW	19
LabVIEW 2: A first-rate instrument control product becomes a world-class programming system	22
The port to Windows and Sun	23
LabVIEW 3	24
LabVIEW 4	25
LabVIEW branches to BridgeVIEW	26
LabVIEW 5	26
The LabVIEW RT branch	28
LabVIEW 6	29
LabVIEW 7	29
LabVIEW 8	31
Crystal Ball Department	32
LabVIEW influences other software products	32
LabVIEW Handles Big Jobs	34
Chapter 2 Getting Started	37
About the Diagrams in This Book	37
Sequencing and Data Flow	38

iv Contents

LabVIEW under the Hood	39
The parts of a VI	40
How VIs are compiled	41
Multitasking, multithreaded LabVIEW	41
The LabVIEW Environment	43
Front panel	44
Controls	45
Property nodes	45
Block diagram	48
SubVIs	48
Icons	49
Polymorphic VIs	50
Data	50
Clusters	50
Typedefs	52
Arrays	53
Debugging	54
See what the subVIs are up to	54
Peeking at data	55
One step at a time	55
Execution highlighting	57
Setting breakpoints	57
Suspend when called	58
Calling Other Code	58
CINs	58
Dynamic link libraries	59
Programming by Plagiarizing	59
Bibliography	60
Chapter 3 Controlling Program Flow	61
Sequences	61
Data Dependency	62
Adding Common Threads	63
Looping	64
While LOOPS	65
For Loops	66
Shift registers	67
Uninitialized shift registers	69
Globals	71
Global and local variables	71
Built-in global variables—and their hazards	73
Local variables	75
Events	81
Notify and Filter events	81
Mechanical actions	85
Dynamic events	85
Design Patterns	87
Initialize and then loop	87
Independent parallel loops	89
Client-server	90
Client-server (with autonomous VIs)	92
State machines	94
Queued message handler	98
Event-driven applications	100
Bibliography	102

Chapter 4 LabVIEW Data Types	103
Numeric Types	104
Strings	105
Building strings	106
Parsing strings	107
Dealing with unprintables	110
Spreadsheets, strings, and arrays	110
Arrays	114
Initializing arrays	117
Array memory usage and performance	119
Clusters	122
Waveforms	125
Data Type Conversions	127
Conversion and coercion	128
Intricate conversions and type casting	129
Flatten To String (. . . Do what?)	132
Enumerated types (enums)	133
Get Carried Away Department	134
Chapter 5 Timing	137
Where Do Little Timers Come From?	137
Using the Built-in Timing Functions	138
Intervals	139
Timed structures	140
Timing sources	141
Execution and priority	143
Timing guidelines	145
Sending timing data to other applications	146
High-resolution and high-accuracy timing	147
Bibliography	149
Chapter 6 Synchronization	151
Polling	152
Events	153
Occurrences	155
Notifiers	158
Queues	160
Semaphores	161
Me and You, Rendezvous	165
Chapter 7 Files	167
Accessing Files	167
File Types	169
Writing Text Files	170
Reading Text Files	172
Formatting to Text Files	175
Binary Files	176
Writing binary files	178
Reading binary files	179
Writing Datalog Files	181
Reading Datalog Files	183
Datalog file utilities	183

Chapter 8 Building an Application	185
Define the Problem	186
Analyze the user's needs	186
Gather specifications	187
Draw a block diagram	189
Specify the I/O Hardware	191
Prototype the User Interface	192
Panel possibilities	193
First Design and Then Write Your Program	196
Ask a Wizard	197
Top-down or bottom-up?	197
Modularity	198
Choose an architecture: Design patterns	200
The VI hierarchy as a design tool	201
Sketching program structure	202
Pseudocoding	203
Ranges, coercion, and default values	204
Handling errors	207
Putting it all together	213
Testing and Debugging Your Program	214
Tracing execution	214
Checking performance	216
Final Touches	217
VBL epilogue	218
Studying for the LabVIEW Certification Exams	218
CLAD	218
CLD	221
Example 1: Traffic light controller	223
Example 2: Car wash controller	224
Example 3: Security system	227
Bibliography	229
Chapter 9 Documentation	231
VI Descriptions	231
Control Descriptions	232
Custom Online Help	233
Documenting the Diagram	234
VI History	234
Other Ways to Document	235
Printing LabVIEW Panels and Diagrams	235
Putting LabVIEW screen images into other documents	236
Writing Formal Documents	238
Document outline	238
Connector pane picture	239
VI description	239
Terminal descriptions	240
Programming examples	241
Distributing Documents	241
Chapter 10 Instrument Driver Basics	243
Finding Instrument Drivers	243
Driver Basics	245
Communication standards	245
Learn about Your Instrument	249
Determine Which Functions to Program	250

Establish Communications	251
Hardware and wiring	252
Protocols and basic message passing	254
Bibliography	256
Chapter 11 Instrument Driver Development Techniques	257
Plug-and-Play Instrument Drivers	259
General Driver Architectural Concepts	260
Error I/O flow control	261
Modularity by grouping of functions	265
Project organization	266
Initialization	267
Configuration	268
Action and status	270
Data	270
Utility	271
Close	272
Documentation	273
Bibliography	274
Chapter 12 Inputs and Outputs	275
Origins of Signals	275
Transducers and sensors	276
Actuators	278
Categories of signals	279
Connections	284
Grounding and shielding	284
Why use amplifiers or other signal conditioning?	291
Choosing the right I/O subsystem	299
Network everything!	303
Bibliography	304
Chapter 13 Sampling Signals	305
Sampling Theorem	305
Filtering and Averaging	307
About ADCs, DACs, and Multiplexers	309
Digital-to-analog converters	314
Digital codes	315
Triggering and Timing	316
A Little Noise Can Be a Good Thing	317
Throughput	319
Bibliography	321
Chapter 14 Writing a Data Acquisition Program	323
Data Analysis and Storage	325
Postrun analysis	326
Real-time analysis and display	337
Sampling and Throughput	343
Signal bandwidth	343
Oversampling and digital filtering	344
Timing techniques	350
Configuration Management	350
What to configure	351
Configuration editors	352

viii Contents

Configuration compilers	362
Saving and recalling configurations	365
A Low-Speed Data Acquisition Example	370
Medium-Speed Acquisition and Processing	373
Bibliography	375
Chapter 15 LabVIEW RT	377
Real Time Does Not Mean Real Fast	377
RT Hardware	379
Designing Software to Meet Real-Time Requirements	382
Measuring performance	383
Shared resources	388
Multithreading and multitasking	389
Organizing VIs for best real-time performance	391
Context switching adds overhead	393
Scheduling	395
Timed structures	395
Communications	398
Bibliography	399
Chapter 16 LabVIEW FPGA	401
What Is an FPGA?	401
LabVIEW for FPGAs	403
RIO hardware platforms	403
Plug-in cards	403
CompactRIO	404
Timing and synchronization	405
Compact Vision	405
Application Development	406
Compiling	406
Debugging	408
Synchronous execution and the enable chain	408
Clocked execution and the single-cycle Timed Loop	411
Parallelism	413
Pipelining	413
Conclusions	414
Bibliography	415
Chapter 17 LabVIEW Embedded	417
Introduction	417
History	417
LabVIEW Embedded Development Module	419
The technology: What's happening	419
Running LabVIEW Embedded on a new target	421
Porting the LabVIEW runtime library	422
Incorporating the C toolchain	423
The Embedded Project Manager	424
LEP plug-in VIs	425
Target_OnSelect	426
Other plug-in VIs	426
Incorporating I/O drivers	429
LabVIEW Embedded programming best practices	431
Interrupt driven programming	434
LabVIEW Embedded targets	435

Chapter 18 Process Control Applications	437
Process Control Basics	438
Industrial standards	438
Control = manipulating outputs	444
Process signals	447
Control system architectures	449
Working with Smart Controllers	455
Single-loop controllers (SLCs)	461
Other smart I/O subsystems	463
Man-Machine Interfaces	463
Display hierarchy	469
Other interesting display techniques	473
Handling all those front panel items	474
Data Distribution	475
Input scanners as servers	476
Handling output data	477
Display VIs as clients	479
Using network connections	482
Real-time process control databases	484
Simulation for validation	485
Sequential Control	486
Interlocking with logic and tables	486
State machines	487
Initialization problems	489
GrafcetVIEW—a graphical process control package	490
Continuous Control	492
Designing a control strategy	493
Trending	499
Real-time trends	499
Historical trends	502
Statistical process control (SPC)	505
Alarms	506
Using an alarm handler	508
Techniques for operator notification	511
Bibliography	512
Chapter 19 Physics Applications	513
Special Hardware	514
Signal conditioning	514
CAMAC	518
Other I/O hardware	518
Field and Plasma Diagnostics	520
Step-and-measure experiments	520
Plasma potential experiments	527
Handling Fast Pulses	533
Transient digitizers	533
Digital storage oscilloscopes (DSOs)	536
Timing and triggering	537
Capturing many pulses	539
Recovering signals from synchronous experiments	543
Handling Huge Data Sets	546
Reducing the amount of data	546
Optimizing VIs for memory usage	547
Bibliography	553

x Contents

Chapter 20 Data Visualization, Imaging, and Sound	555
Graphing	556
Displaying waveform and cartesian data	558
Bivariate data	563
Multivariate data	565
3D Graphs	570
Intensity Chart	571
Image Acquisition and Processing	572
System requirements for imaging	574
Using IMAQ Vision	577
IMAQ components	577
Sound I/O	586
DAQ for sound I/O	586
Sound I/O functions	587
Sound input	587
Sound output	588
Sound files	588
Bibliography	589
Index	591

Preface

Twenty years have passed since the release of LabVIEW. During this period, it has become the dominant programming language in the world of instrumentation, data acquisition, and control. A product of National Instruments Corporation (Austin, Texas), it is built upon a purely graphical, general-purpose programming language, *G*, with extensive libraries of functions, an integral compiler and debugger, and an application builder for stand-alone applications. The LabVIEW development environment runs on Apple Macintosh computers and IBM PC compatibles with Linux or Microsoft Windows. Programs are portable among the various development platforms. The concept of *virtual instruments* (VIs), pioneered by LabVIEW, permits you to transform a real instrument (such as a voltmeter) into another, software-based instrument (such as a chart recorder), thus increasing the versatility of available hardware. Control panels mimic real panels, right down to the switches and lights. All programming is done via a block diagram, consisting of icons and wires, that is directly compiled to executable code; there is no underlying procedural language or menu-driven system.

Working with research instrumentation, we find LabVIEW indispensable—a flexible, time-saving package without all the frustrating aspects of ordinary programming languages. The one thing LabVIEW had been missing all these years was a useful application-oriented book. The manuals are fine, once you know what you want to accomplish, and the classes offered by National Instruments are highly recommended if you are just starting out. But how do you get past that first blank window? What are the methods for designing an efficient LabVIEW application? What about interface hardware and real-world signal-conditioning problems? In this book, we describe practical problem-solving techniques that aren't in the manual or in the introductory classes—methods you learn only by experience. The principles and techniques discussed in these pages are fundamental to the work of a LabVIEW programmer. This is by no means a rewrite of the manuals or other introductory books, nor is it a substitute for a course in

LabVIEW basics. You are encouraged to consult those sources, as well, in the process of becoming a skilled LabVIEW developer.

This fourth edition is founded on LabVIEW 8, but we've worked closely with National Instruments to ensure its relevance now and through future versions of LabVIEW. Chapter 1, "Roots," starts off with an entertaining history of the development of LabVIEW. New to this edition is coverage of material on National Instruments' certification exams. There are three levels of LabVIEW certification: Certified LabVIEW Associate Developer (CLAD), Certified LabVIEW Developer (CLD), and Certified LabVIEW Architect (CLA). Each exam builds on the knowledge required for the previous exam. We have worked closely with National Instruments to highlight study material in this book for the first two certification exams. Throughout Chapters 2 through 9 you will find **CLAD** or **CLD** icons next to sections covered on the certification exams.

In Chapters 2 through 9, we get down to the principles of programming in G. After a discussion of the principles of dataflow programming, we discuss programming structures, data types, timing, synchronization, and file I/O. Chapter 8, "Building an Application," shows you how to design a LabVIEW application. Here we assume that you are not a formally trained software engineer, but rather a technically skilled person with a job to do (that certainly describes us!). We'll walk through the development of a real application that Gary wrote, starting with selection of hardware, then prototyping, designing, and testing the program. Chapter 9, "Documentation," covers this important but oft-neglected topic. We discuss recommended practice for creating effective documentation as it pertains to the world of LabVIEW. If you know the material in Chapters 2 through 9 you should have no problems with the certification exams. At the end of Chapter 8 we provide three practice exams for the Certified LabVIEW Developer (CLD) Exam. Use the knowledge you gained in Chapters 2 through 9 to complete these practice exams in four hours and you are well on your way to certification.

If you connect your computer to any external instruments, you will want to read Chapters 10 and 11, "Instrument Driver Basics" and "Instrument Driver Development Techniques." We begin with the basics of communications and I/O hardware (GPIB, serial, and VXI), then cover recommended driver development techniques and programming practices, especially the virtual instrument standard architecture (VISA) methods. Instrument drivers can be fairly challenging to write. Since it's one of our specialties, we hope to pass along a few tricks.

The basics of interface hardware, signal conditioning, and analog/digital conversion are discussed in Chapter 12, "Inputs and Outputs," and Chapter 13, "Sampling Signals." Notably, these chapters contain no LabVIEW programming examples whatsoever. The reason is

simple: more than half the “LabVIEW” questions that coworkers ask us turn out to be hardware- and signal-related. Information in this chapter is vital and will be useful no matter what software you may use for measurement and control. Chapter 14, “Writing a Data Acquisition Program,” contains a practical view of data acquisition (DAQ) applications. Some topics may seem at first to be presented backward—but for good reasons. The first topic is data analysis. Why not talk about sampling rates and throughput first? Because the only reason for doing data acquisition is to collect data for analysis. If you are out of touch with the data analysis needs, you will probably write the wrong data acquisition program. Other topics in this chapter are sampling speed, throughput optimization, and configuration management. We finish with some of the real applications that you can use right out of the box.

LabVIEW RT brings the ease of graphical programming to the arcane world of real-time system programming. In Chapter 15, “LabVIEW RT,” we show you how LabVIEW RT works and how to achieve top performance by paying attention to code optimization, scheduling, and communications.

When software-timed real-time applications won’t fit the bill, LabVIEW FPGA is the way to go. LabVIEW FPGA applications are not constrained by processor or operating system overhead. With LabVIEW FPGA you can write massively parallel hardware-timed digital control applications with closed loop rates in the tens of megahertz. Chapter 16, “LabVIEW FPGA,” gives a solid introduction to programming FPGAs with LabVIEW.

Embedded computer systems are all around us—in our cars, VCRs, appliances, test equipment, and a thousand other applications. But until now, LabVIEW has not been a viable development system for those miniaturized computers. Chapter 17, “LabVIEW Embedded,” introduces a new version of LabVIEW capable of targeting any 32-bit microprocessor.

Chapter 18, “Process Control Applications,” covers industrial control and all types of measurement and control situations. We’ll look at human-machine interfaces, sequential and continuous control, trending, alarm handling, and interfacing to industrial controllers, particularly programmable logic controllers (PLCs). We frequently mention a very useful add-on toolkit that you install on top of LabVIEW, called the Datalogging and Supervisory Control Module (formerly available as BridgeVIEW), which adds many important features for industrial automation.

LabVIEW has a large following in physics research, so we wrote Chapter 19, “Physics Applications.” Particular situations and solutions in this chapter are electromagnetic field and plasma diagnostics, measuring fast pulses with transient recorders, and handling very large data sets. This last topic, in particular, is of interest to almost all users

because it discusses techniques for optimizing memory usage. (There are tidbits like this all through the book—by all means, read it cover to cover!)

Chapter 20, “Data Visualization, Imaging, and Sound,” shows off some of the data presentation capabilities of LabVIEW. Some third-party products and toolkits (such as IMAQ for imaging) are featured. They enable you to acquire video signals, process and display images, make three-dimensional plots, and record and play sound.

As far as possible, this book is platform-independent, as is LabVIEW itself. Occasional topics arise where functionality is available on only one or two of the computer platforms. The LabVIEW user manual contains a portability guide that you can consult when developing applications that you intend to propagate among various platforms.

Many important resources are available only via the Internet. For your convenience, Internet addresses are interspersed in the text. While writing this book, we found that user-supplied example VIs were hard to obtain, owing to the fact that so many of us work for government laboratories and places that just don’t like to give away their software. Where it was not possible to obtain the actual code, we attempted to reconstruct the important aspects of real applications to give you an idea of how you might solve similar problems.

Third-party LabVIEW products, such as driver and analysis packages, are described where appropriate. They satisfy important niche requirements in the user community at reasonable cost, thus expanding the wide applicability of LabVIEW.

If nothing else, we hope that our enthusiasm for LabVIEW rubs off on you.

Gary W. Johnson and Richard Jennings

Acknowledgments

We would like to thank the engineers, developers, and managers at National Instruments who supplied vital information without which this book would not be possible, particularly Jeff Kodosky, David Gardner, Joel Sumner, Newton Petersen, P. J. Tanzillo, and Kathy Brown. A special thanks goes to Zaki Chasmawala of National Instruments for proof-reading and highlighting the pertinent parts of Chapters 2 through 9 covered by the certification exams.

Credit also goes to our wives, Katharine Decker Johnson and Patty Jennings, whose patience during this project cannot be overstated. And to Elizabeth, Danny, Chris, and David Jennings—thank you for understanding.

Finally, thanks to our editor, Wendy Rinaldi.

*Gary W. Johnson:
To my wife, Katharine*

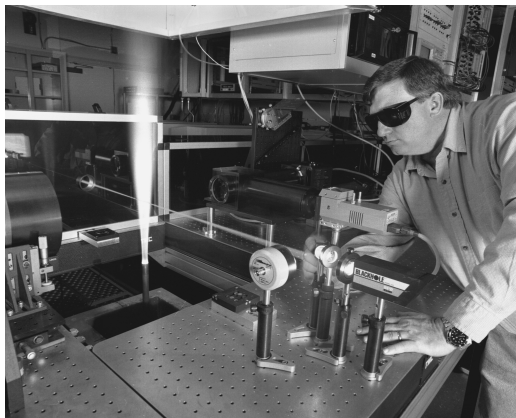
*Richard Jennings:
To my Lord and Savior, Jesus Christ*

ABOUT THE AUTHORS

Gary W. Johnson is an instrumentation engineer at the Lawrence Livermore National Laboratory. He has a BS degree in electrical engineering/bioengineering from the University of Illinois. His professional interests include measurement and control systems, electro-optics, communications, transducers, circuit design, and technical writing. In his spare time, he enjoys woodworking, bicycling, and amateur radio. He and his wife, Katharine, a scientific illustrator, live in Livermore, California, with their twin Afghan hounds, Chloe and Derby.



LabVIEW goes aloft. Gary works on a LabVIEW-based laser wavelength controller for an airborne LIDAR system aboard an Air Force C-135.



Do not look into laser with remaining good eye. Richard wisely wears his laser eye protection while manually aligning laser beams through a piloted jet burner in the Turbulent Combustion Laboratory.

Richard Jennings is president of Jennings Embedded Services, LLC in San Antonio, Texas. His company was founded in 2005 to serve as a hardware and software resource for companies engaged in embedded software and hardware development in emerging embedded markets such as industrial control, wireless, and embedded instrumentation. He is a 15-year veteran hardware and software engineer. Prior to starting Jennings Embedded Services, a National Instruments Certified Alliance partner, Jennings worked as a system integrator at Sandia National Laboratories and at Lawrence Livermore National Laboratories in Livermore, California. He holds an Associate Degree in Laser-ElectroOptics from Texas State Technical Institute in Waco, Texas, and is a Certified LabVIEW Developer. In 2003 he was awarded National Instruments' prestigious Virtual Instrumentation Professional (VIP) award. www.jembedded.com