LabVIEW™

Robotics Programming Guide for the *FIRST* Robotics Competition

January 2010
372668F-01

NATIONAL INSTRUMENTS™

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 43 662 457990-0, Belgium 32 (0) 2 757 0020, Brazil 55 11 3262 3599, Canada 800 433 3488, China 86 21 5050 9800, Czech Republic 420 224 235 774, Denmark 45 45 76 26 00, Finland 358 (0) 9 725 72511, France 01 57 66 24 24, Germany 49 89 7413130, India 91 80 41190000, Israel 972 3 6393737, Italy 39 02 41309277, Japan 0120-527196, Korea 82 02 3451 3400, Lebanon 961 (0) 1 33 28 28, Malaysia 1800 887710, Mexico 01 800 010 0793, Netherlands 31 (0) 348 433 466, New Zealand 0800 553 322, Norway 47 (0) 66 90 76 60, Poland 48 22 328 90 10, Portugal 351 210 311 210, Russia 7 495 783 6851, Singapore 1800 226 5886, Slovenia 386 3 425 42 00, South Africa 27 0 11 805 8197, Spain 34 91 640 0085, Sweden 46 (0) 8 587 895 00, Switzerland 41 56 2005151, Taiwan 886 02 2377 2222, Thailand 662 278 6777, Turkey 90 212 279 3031, United Kingdom 44 (0) 1635 523545

To comment on National Instruments documentation, refer to the National Instruments Web site at ni.com/info and enter the info code feedback.

Important Information

Warranty

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

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

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

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

Copyright

Trademarks

Patents

For patents covering National Instruments products/technology, refer to the appropriate location: **Help»Patents** in your software, the `patents.txt` file on your media, or the *National Instruments Patent Notice* at `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

Chapter 3
Configuring the Camera and the CompactRIO Device

Chapter 4
Using the FRC Framework

Chapter 5
Tutorial: Creating an FRC Robot Project

Chapter 6
Tutorial: Creating an FRC Dashboard Project

Chapter 7
Troubleshooting the FRC Robot

Chapter 8
Using the WPI Robotics Library VIs

About This Manual

The *FIRST* Robotics Competition (FRC) software includes three separate programming environments—LabVIEW, Wind River® Workbench, and the Sun™ SPOT Java™ Development Kit for FRC. Use any of these environments to develop the robotics program you want to run on the CompactRIO device. Use LabVIEW to program a robot in the LabVIEW graphical programming environment or to develop a desktop application. Use Wind River Workbench to program a robot in C or C++. Use the Sun SPOT JDK to program a robot in Java.

This manual discusses how to develop a robotics program in LabVIEW. Use this manual to access information about robotics programming concepts. This manual also describes how to configure the CompactRIO device and the Axis camera, as well as how to use the FRC framework.

Refer to the *C/C++ Programming Guide for the FIRST Robotics Competition*, available by navigating to the `WindRiver\docs\extensions\FRC` directory and opening `C Programming Guide for FRC.pdf`, for information about how to develop a robotics program with Wind River Workbench.

Refer to the *C and Java Programming Guide for FRC*, available by navigating to the `FRCJava` directory and opening `C and Java Programming Guide for FRC.pdf`, for information about how to develop a robotics program with the Sun SPOT JDK.

Conventions

The following conventions appear in this manual:

» The **»** symbol leads you through nested menu items and dialog box options to a final action. The sequence **File»Page Setup»Options** directs you to pull down the **File** menu, select the **Page Setup** item, and select **Options** from the last dialog box.

This icon denotes a 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. Italic text also denotes text that is a placeholder for a word or value that you must supply.
`monospace`	Text in this font denotes text or characters that you should enter from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, programs, subprograms, subroutines, device names, functions, operations, variables, filenames, and extensions.
`monospace bold`	Bold text in this font denotes the messages and responses that the computer automatically prints to the screen. This font also emphasizes lines of code that are different from the other examples.
`monospace italic`	Italic text in this font denotes text that is a placeholder for a word or value that you must supply.

Related Documentation

The following documents contain information that you may find helpful as you read this manual. Refer to the FRC Community Web site at `www.usfirst.org/frccontrolsystem` for official information about the FRC competition, including rules and regulations as well as support information. The *FRC Control System Manual* also is available on this Web site under **Documents and Updates**.

Programming in LabVIEW

The following documents contain information that you may find helpful as you use LabVIEW:

- *Getting Started with LabVIEW for the FIRST Robotics Competition*—Use this manual to learn about the LabVIEW graphical programming environment and the basic LabVIEW features you can use to build FRC applications. Access this manual by navigating to the `National Instruments\LabVIEW 8.6\manuals` directory and opening `FRC_Getting_Started.pdf`.
- *LabVIEW Help*—Use the *LabVIEW Help* to access information about LabVIEW programming concepts, step-by-step instructions for using LabVIEW, and reference information about LabVIEW VIs, functions, palettes, menus, tools, properties, methods, events, dialog boxes, and

so on. The *LabVIEW Help* also lists the LabVIEW documentation resources available from National Instruments. Access the *LabVIEW Help* by selecting **Help»Search the LabVIEW Help** in LabVIEW.

- *LabVIEW Quick Reference Card*—Use this card as a reference for information about documentation resources, keyboard shortcuts, data type terminals, and tools for editing, execution, and debugging. Access this manual by navigating to the `National Instruments\LabVIEW 8.6\manuals` directory and opening `LV_Quick_Reference.pdf`.
- *National Instruments FIRST Community*—Refer to the National Instruments *FIRST* Community Web site at `ni.com/first` to access tutorials and examples about using LabVIEW for FRC and to connect with other FRC participants.

Using the CompactRIO Device

The following documents contain information that you may find helpful as you use the CompactRIO device:

- *cRIO-FRC Operating Instructions and Specifications*—Use this manual to learn about installing, configuring, and using the CompactRIO device for the *FIRST* Robotics Competition. Access this manual by navigating to the `National Instruments\CompactRIO\manuals` directory and opening `crio-frc_Operating_Instructions.pdf`.
- *NI 9201/9221 Operating Instructions*—This document describes how to use the National Instruments 9201 and National Instruments 9221 and includes specifications and terminal/pin assignments for the NI 9201/9221. Access this manual by navigating to the `National Instruments\CompactRIO\manuals` directory and opening `NI_9201_9221_Operating_Instructions.pdf`.
- *NI 9403 Operating Instructions and Specifications*—This document describes how to use the National Instruments 9403 and includes specifications and pin assignments for the NI 9403. Access this manual by navigating to the `National Instruments\CompactRIO\manuals` directory and opening `NI_9403_Operating_Instructions.pdf`.
- *NI 9472/9474 Operating Instructions*—This document describes how to use the National Instruments 9472 and National Instruments 9474 and includes specifications and terminal/pin assignments for the NI 9472/9474. Access this manual by navigating to the `National Instruments\CompactRIO\manuals` directory and opening `NI_9472_9474_Operating_Instructions.pdf`.

1

Overview of the *FIRST* Robotics Competition

The objective of the *FIRST* Robotics Competition (FRC) is to build and program a robot to perform certain tasks either autonomously or in response to commands the robot receives from input devices such as joysticks and game controllers.

FRC Terminology

The following terms are useful to know as you build and program a robot for FRC and as you read this manual. Refer to the *FIRST* Community Web site at `www.usfirst.org/frccontrolsystem` for more information about the FRC control system and related terminology.

CompactRIO Device

A CompactRIO device, also referred to as a cRIO, is a small, rugged controller consisting of an embedded real-time processor, an FPGA (field-programmable gate array), and slots for interchangeable I/O modules. For FRC, the CompactRIO device contains five I/O modules: two analog input modules, two digital input/output modules, and one digital output module. Each module interfaces directly with the FPGA on the CompactRIO device.

Refer to the *I/O Modules* section of Chapter 2, *Robot Architecture*, for more information about the I/O modules that the FRC kit provides.

The CompactRIO device serves as the "brain" of each robot and runs the program you develop and deploy from LabVIEW. The CompactRIO device also receives data from the driver station and the various sensors that you connect to the I/O modules. Furthermore, the CompactRIO device hosts two watchdogs. The system watchdog shuts down all motors if the communication network fails. The user watchdog can shut down all motors when programming failures, such as infinite loops, occur.

Driver Station

The driver station is a program that passes data between the host computer; input devices, such as joysticks and game controllers; field management system (FMS); and the robot. The driver station can read analog data and can read and write digital data. The driver station receives data from the input devices and sends the data to the CompactRIO device on the robot. The driver station also passes data it receives from the CompactRIO device to the host computer so you can view the data in LabVIEW.

Using an Ethernet connection, connect the classmate PC that runs the driver station to the Ethernet router. If you develop programs on a host computer, you also must connect the host computer to the Ethernet router.

Note You can develop programs directly on the classmate PC.

Connect the Ethernet router to the CompatRIO device through an Ethernet connection or a wireless access point. The wireless access point must be on the same local subnet as the wireless access point you connect to the CompactRIO device.

During the actual FRC competition, connect the classmate PC to the FMS instead of the Ethernet router. The FMS contains a wireless access point that can communicate with the wireless access points on multiple robots.

Field Management System

FIRST uses an FMS during the FRC competition to monitor robot traffic and to send commands to the driver stations. For example, the FMS designates the current mode of the competition. The FMS can connect through an Ethernet connection to multiple driver stations and through a wireless connection to multiple robots.

Autonomous and Teleop Modes

The FRC competition consists of two parts: Autonomous mode and Teleop mode.

In Autonomous mode, the robot moves without receiving commands from input devices, such as joysticks and game controllers. You develop a program in LabVIEW and deploy that program to the CompactRIO device on the robot. When you run the program, the robot moves and behaves according to instructions in the program.

In Teleop mode, the robot moves in response to commands it receives from input devices. As in Autonomous mode, you develop a program in LabVIEW and deploy that program to the CompactRIO device on the robot. The program must contain instructions that allow the robot to receive data from the input devices and respond accordingly. When you run the program, you then can use the input devices to manipulate the behavior of the robot. If you want the robot to respond to commands it receives from input devices, connect the input devices to the Ethernet router using a USB connection. You can use the joystick that the FRC kit provides. You also can use any other joystick, game controller, or other input device that you choose.

Enabled and Disabled Status

At certain points in the FRC competition, the program on the CompactRIO device must be running, but the robot cannot move. For example, the program continues running between the Autonomous and Teleop parts of the competition, but no motors can move. During these times, the FMS sets the status of the robot to **Disabled** and kills the system watchdog, which disables the PWM, relay, and solenoid outputs of the robot.

When the robot is in the Disabled status, you still can use input devices, such as joysticks and game controllers, to send information to the robot. You also still can read information from sensors and perform image processing on the CompactRIO device. Therefore, you can develop your robotics program to handle the Disabled status such that the program handles initialization and processing tasks, rather than motion tasks, when the robot is disabled. When the FMS sets the status of the robots to **Enabled**, the system watchdog re-enables the PWM, relay, and solenoid outputs. Therefore, you can develop your robotics program to move motors and drive the robot when the robot is enabled.

Ensure the program you run on the CompactRIO device handles the Disabled and Enabled statuses appropriately. Refer to Chapter 4, *Using the FRC Framework*, for more information about the FRC robot project.

Init, Execute, and Stop Derived States

During the competition, the robot can be in one of the following competition states:

- Autonomous Disabled
- Autonomous Enabled
- Teleop Disabled
- Teleop Enabled

For each of these states, the robot can be in one of three derived states: Init, Execute, or Stop. In the Init derived state, you can specify the robot to perform any initialization tasks, such as opening sensor or I/O references. In the Execute derived state, you can specify the robot to move, read and write data, or perform some other behavior. In the Stop derived state, you can specify the robot to perform tasks such as stopping motors and closing references.

Each time the competition state changes, the robot program transitions first to the Stop derived state of the current state and then to the Init derived state of the new state. For example, if the robot is in the Execute derived state of the Autonomous Disabled state and the FMS sets the competition state to **Teleop Disabled**, the robot transitions first to the Stop derived state of the Autonomous Disabled state and then to the Init derived state of the Teleop Disabled state.

Estop State

The emergency stop, or Estop, state is an emergency or disqualification state that shuts down the robot immediately. Only the FMS can set the robot to this state. If the robot reaches the Estop state, you must reboot the CompactRIO device to restart execution.

2

Robot Architecture

The *FIRST* Robotics Competition (FRC) controller system consists of the following subsystems:

- **Driver Console**—Establishes communication between the CompactRIO device on the robot and the host computer. This subsystem includes the classmate PC that runs the driver station program; one or more input devices, such as joysticks and game controllers; and a host computer.
- **Wireless Communication**—Establishes wireless communication between the driver station and the robot. This subsystem includes wireless access points for the driver station and the CompactRIO device.
- **Robot**—Consists of all parts that make up the moving robot. This subsystem includes the CompactRIO device with five I/O modules, analog and pneumatic breakouts, a digital sidecar, and an Axis camera.

This chapter discusses how to use LabVIEW to interface with the robot subsystem. Refer to the *FRC Control System Manual*, accessible on the FRC Community Web site at `www.usfirst.org/frccontrolsystem`, for more information about the entire FRC controller system.

CompactRIO Device

A CompactRIO device, also referred to as a cRIO, is a small, rugged controller consisting of an embedded real-time processor, an FPGA (field-programmable gate array), and slots for interchangeable I/O modules. For FRC, the CompactRIO device contains five I/O modules: two analog input modules, two digital input/output modules, and one digital output module. Each module interfaces directly with the FPGA on the CompactRIO device.

Real-Time Operating System

Most LabVIEW applications run on a general-purpose operating system (OS) like Windows, Mac OS, or Linux. Some applications require real-time performance that general-purpose operating systems cannot guarantee.

Real-time systems are deterministic. Determinism is the characteristic of a system that describes how consistently the system responds to external events or performs operations within a given time limit. Jitter is a measure of the extent to which execution timing fails to meet deterministic expectations. Most real-time applications require timing behavior to consistently execute within a small window of acceptable jitter.

You can run real-time programs on a real-time target such as a CompactRIO device. The CompactRIO device runs the real-time operating system (RTOS) of Wind River VxWorks. In LabVIEW, deploy each application you want to run on the CompactRIO device from the **RT CompactRIO Target** directory of the **Project Explorer** window. Refer to the *Deploying the FRC Robot Project* section of Chapter 4, *Using the FRC Framework*, for more information about deploying an application to the CompactRIO device.

FPGA

The CompactRIO device includes a built-in, high-performance FPGA. An FPGA is an embedded chip that you can reconfigure for different applications. Each I/O module on the CompactRIO device interfaces directly with the FPGA.

Use the FRC FPGAVersion VI to determine the version and revision of the FPGA on the CompactRIO device. The version of the FPGA corresponds to the year of the FRC competition.

I/O Modules

For FRC, the CompactRIO device contains five I/O modules:

- 2 — NI 9201 analog input
- 2 — NI 9403 digital input/output
- 1 — NI 9472 digital output

Table 2-1 lists the slots on the CompactRIO device and the modules corresponding to those slots.

Table 2-1. Slots for I/O Modules on the CompactRIO Device

Slot	I/O Module
1	NI 9201
2	NI 9201
3	—
4	NI 9403
5	—
6	NI 9403
7	—
8	NI 9472

When you use the WPI Robotics Library VIs, you must specify the module and channel you want to use. For example, you can specify a module value of 1 to use the NI 9201 in slot 1 of the CompactRIO device. You then can select a value for the channel on the NI 9201 that you want to use.

NI 9201

The NI 9201 provides connections for eight analog input channels. You connect an analog breakout to each of the NI 9201 modules. You then can connect any analog sensors, such as accelerometers and gyros, to the analog breakouts to acquire analog data.

Note You must insert the NI 9201 analog modules before the CompactRIO device boots for accurate calibration of the modules. Swapping the modules after the CompactRIO device boots applies the calibration for the original module to the replacement module. Inserting the modules after the CompactRIO device boots applies factory default calibration values as the analog module constants.

Use the Accelerometer VIs and the Gyro VIs to acquire analog data from those sensors. Refer to the *FIRST* Robotics Competition VIs on `ni.com` for more information about these VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

Use the AnalogChannel VIs to acquire analog data from other analog input devices. Refer to the *FIRST* Robotics Competition VIs on ni.com for more information about the AnalogChannel VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

Refer to the *NI 9201/9221 Operating Instructions* document, accessible by navigating to the `National Instruments\CompactRIO\manuals` directory and opening `NI_9201_9221_Operating_Instructions.pdf`, for more information about the NI 9201.

NI 9403

The NI 9403 provides connections for 32 digital input or digital output channels. You use a DSUB cable to connect a digital sidecar to each of the NI 9403 modules. You then can connect digital sensors, such as counters, encoders, and ultrasonic sensors, to the digital sidecars to acquire digital data.

Use the Counter, Encoder, and Ultrasonic VIs to acquire digital data from those sensors. Refer to the *FIRST* Robotics Competition VIs on ni.com for more information about these VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

Use the DigitalInput VIs and the DigitalOutput VIs to send or receive digital data from other digital input and output devices. Refer to the *FIRST* Robotics Competition VIs on ni.com for more information about these VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

Refer to the *NI 9403 Operating Instructions and Specifications* document, accessible by navigating to the `National Instruments\CompactRIO\manuals` directory and opening `NI_9403_Operating_Instructions.pdf`, for more information about the NI 9403.

NI 9472

The NI 9472 provides connections for eight digital output channels. Whereas the NI 9403 can output a maximum voltage of 5.2 V, the NI 9472 can output a maximum voltage of 30 V. If you use the pneumatic breakout with the NI 9472, the NI 9472 can output a maximum voltage of 12 V.

You can use this module to control a solenoid. Connect a pneumatic breakout to the NI 9472 and connect the solenoid to the pneumatic breakout. Then use the Solenoid VIs to specify whether the NI 9472 sends a 12 V signal to the solenoid.

Refer to the *FIRST* Robotics Competition VIs on ni.com for more information about the Solenoid VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

Refer to the *NI 9472/9474 Operating Instructions* document, accessible by navigating to the `National Instruments\CompactRIO\manuals` directory and opening `NI_9472_9474_Operating_Instructions.pdf`, for more information about the NI 9472.

Axis Camera

The robot subsystem includes an Axis camera that you can use to perform image acquisition. You can acquire images and process them directly on the CompactRIO device.

Use the Camera VIs to specify settings for the Axis camera and to acquire images with the camera. Use the *FIRST* Vision VIs to process the images you acquire.

Refer to the *FIRST* Robotics Competition VIs on ni.com for more information about the Camera VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

The CompactRIO device is headless, so you cannot view the images being acquired. However, as you develop the program you want to run on the CompactRIO device, you can send the image data from the CompactRIO device to the host computer and view the images on the host computer. Use the Get Image From Controller VI to retrieve a specific image on the host computer from the image data that the CompactRIO device sends.

Refer to the *FIRST* Robotics Competition VIs on ni.com for more information about the Get Image From Controller VI. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

You also can use live front panel debugging to view the images on the host computer. Click the **Run** button of the VI you want to deploy to the CompactRIO device to perform live front panel debugging.

Refer to the *Deploying the Program Using the Run Button* section of Chapter 4, *Using the FRC Framework*, for more information about live front panel debugging.

Note You can send image data to the host computer or perform live front panel debugging only during development, not during the FRC competition.

Before you can use the Axis camera, you must configure it with the correct username and password. Use the Setup Axis Camera Tool, available by selecting **Tools»Setup Axis Camera**, to configure the Axis camera. Refer to the *Configuring the Axis Camera* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the camera.

FRC Software LabVIEW Components

The FRC software includes LabVIEW, the LabVIEW Real-Time Module, the NI Vision Development Module, NI-RIO, as well as FRC-specific VIs. Use this software to interface directly with the various I/O modules on the CompactRIO device and manipulate the behavior of the robot you build. You can use the FRC software to develop a program that runs on the host computer or that you can deploy and run on the CompactRIO device.

FIRST Robotics Competition VIs

The *FIRST* Robotics Competition VIs are divided into two palettes, ***FIRST* Vision** and **WPI Robotics Library**. Use the *FIRST* Vision VIs to process images you acquire with the Axis camera. Use the WPI Robotics Library VIs to interface with the CompactRIO device and perform tasks such as sending and receiving data from sensors and driving motors. The WPI Robotics Library VIs are analogous to the WPI Robotics Library, a library of C/C++ functions developed by Worcester Polytechnic Institute (WPI) for robotics applications.

Refer to the *FIRST* Robotics Competition VIs on `ni.com` for information about the WPI Robotics Library VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

FRC Configuration Tools

The FRC software also includes configuration tools you can use to configure the Axis camera and the CompactRIO device.

Setup Axis Camera Tool

Use the Setup Axis Camera Tool, available by selecting **Tools»Setup Axis Camera**, to configure the Axis camera with the correct username and password.

Refer to the *Configuring the Axis Camera* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the Axis camera.

CompactRIO Imaging Tool

Use the CompactRIO Imaging Tool, available by selecting **Tools» CompactRIO Imaging Tool** in LabVIEW, to configure the CompactRIO device with a start-up image. You can use this tool to switch between the LabVIEW, Wind River Workbench, and the Sun SPOT JDK programming environments when developing the program you run on the CompactRIO device. You also can restore an image on the CompactRIO device or update the CompactRIO device with a new name or IP address.

Refer to the *Configuring the CompactRIO Device* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the CompactRIO device.

3

Configuring the Camera and the CompactRIO Device

The CompactRIO device in the *FIRST* Robotics Competition (FRC) kit contains an initial image that allows you to steer a robot with a joystick. You can use this image to verify that the motors, motor controllers, and joystick work as expected. After you verify this behavior, you must configure the Axis camera and the CompactRIO device for use in the FRC competition.

Configuring the Axis Camera

When you configure the Axis camera, you set the username and password of the camera to `FRC` so the Camera VIs can communicate with the camera. You need to configure the camera only once. You do not need to reconfigure the Axis camera during the FRC competition.

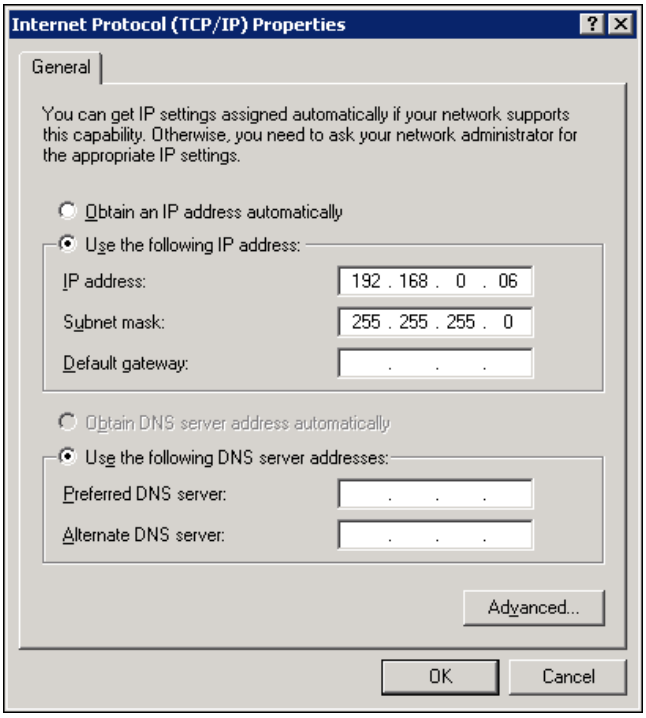
To configure the Axis camera, you must set the static IP address of the computer, connect the computer to the camera, and then run the Setup Axis Camera Tool.

Setting the Static IP Address of the Computer

Complete the following steps to set the static IP address of the computer.

1. Select **Start»Control Panel»Network Connections»Local Area Connection** to display the **Local Area Connection Status** dialog box.
2. Click the **Properties** button to display the **Local Area Connection Properties** dialog box.
3. On the **General** page, select **Internet Protocol (TCP/IP)** from the **This connection uses the following items** list.
4. Click the **Properties** button to display the **Internet Protocol (TCP/IP) Properties** dialog box.
5. Select the **Use the following IP address** option.
6. In the **IP address** text box, enter `192.168.0.06`.

7. The **Subnet mask** text box defaults to `255.255.255.0`. Use this default value.



8. Click the **OK** button twice to close the **Internet Protocol (TCP/IP) Properties** and **Local Area Connection Properties** dialog boxes.
9. Click the **Close** button to close the **Local Area Connection Status** dialog box.

After you set the static IP address of the computer, use an Ethernet crossover cable to connect the computer to the camera. Then configure the camera with the Setup Axis Camera Tool.

Running the Setup Axis Camera Tool

Complete the following steps to configure the camera with the Setup Axis Camera Tool.

1. Select **Start»All Programs»National Instruments»LabVIEW 8.6» Setup Axis Camera** to display the **Setup Axis Camera** dialog box. You also can display this dialog box by selecting **Tools»Setup Axis Camera** in LabVIEW.
2. Wait for the dialog box to finish configuration. The **Setup Axis Camera** dialog box sets the username and password of the camera to `FRC` so the Camera VIs can communicate with the camera. If the configuration is unsuccessful, you might need to verify the camera connection and the IP settings.
3. Click the **Close** button to close the **Setup Axis Camera** dialog box.

After you configure the camera, use an Ethernet crossover cable to connect the camera to Ethernet port 2 of the CompactRIO device. The program you run on the CompactRIO device then can communicate with the camera.

You also can configure the username and password of the Axis camera manually. Refer to the *FRC Control System Manual*, accessible on the FRC Community Web site at `www.usfirst.org/frccontrolsystem`, for more information about configuring the camera manually.

Configuring the CompactRIO Device

To configure the CompactRIO device, you must set the static IP address of the computer, connect the computer to the CompactRIO device, and then run the CompactRIO Imaging Tool.

Setting the Static IP Address of the Computer

Complete the following steps to set the static IP address of the computer.

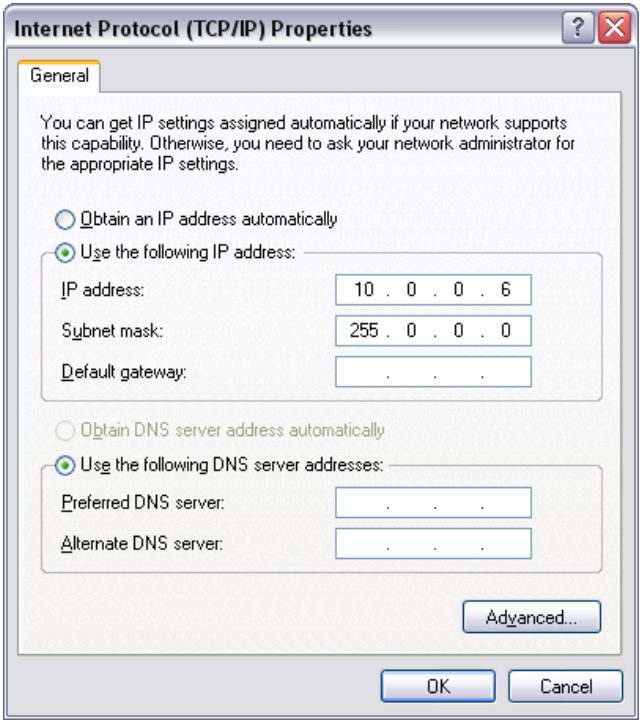
1. Select **Start»Control Panel»Network Connections»Local Area Connection** to display the **Local Area Connection Status** dialog box.
2. Click the **Properties** button to display the **Local Area Connection Properties** dialog box.
3. On the **General** page, select **Internet Protocol (TCP/IP)** from the **This connection uses the following items** list.
4. Click the **Properties** button to display the **Internet Protocol (TCP/IP) Properties** dialog box.

5. Select the **Use the following IP address** option.
6. In the **IP address** text box, enter `10.`*`xx`*`.`*`yy`*`.6`, where *yy* corresponds to the last two digits of the team number and *xx* corresponds to the first or first two digits of the team number. Table 3-1 lists examples of the static addresses corresponding to different team numbers.

Table 3-1. Static IP Addresses Corresponding to Team Numbers

Team Number	Static IP Address
64	`10.0.64.6`
512	`10.5.12.6`
1024	`10.10.24.6`

7. The **Subnet mask** text box defaults to **`255.0.0.0`**. Use this default value.



8. Click the **OK** button twice to close the **Internet Protocol (TCP/IP) Properties** and **Local Area Connection Properties** dialog boxes.
9. Click the **Close** button to close the **Local Area Connection Status** dialog box.

After you set the static IP address of the computer, use an Ethernet crossover cable to connect the computer to Ethernet port 1 of the CompactRIO device. Then configure the CompactRIO device with the CompactRIO Imaging Tool.

Considerations Before Running the CompactRIO Imaging Tool

Before configuring the CompactRIO device with the CompactRIO Imaging Tool, you must ensure that the hardware and software are configured properly.

Do not use the CompactRIO Imaging Tool on the CompactRIO device over a wireless connection. If the connection is lost, the data that the CompactRIO Imaging Tool writes to the CompactRIO device is corrupted.

Do not use Measurement and Automation Explorer (MAX) to install additional software on the CompactRIO device. MAX overwrites the FRC VIs on the CompactRIO device, which makes the CompactRIO device unusable for the FRC competition. If you use MAX to install additional software on the CompactRIO device, you must use the CompactRIO Imaging Tool to restore the device to a usable state.

Before running the CompactRIO Imaging Tool, ensure the SAFE MODE switch on the CompactRIO device is turned off. For routine use, do not use the CompactRIO Imaging Tool when the CompactRIO device is in SAFE MODE.

Note Severe corruptions of the software or settings on the CompactRIO device result in the device no longer functioning. If the CompactRIO device is corrupted or if the IP Address is set incorrectly, the device boots only in SAFE MODE. When this occurs, switch the device into SAFE MODE. The CompactRIO Imaging Tool offers to reformat the disk. After the disk has been reformatted, switch the CompactRIO device out of SAFE MODE, reboot, and run the CompactRIO Imaging Tool normally.

Running the CompactRIO Imaging Tool

Complete the following steps to configure the CompactRIO device with the CompactRIO Imaging Tool.

1. Select **Start»All Programs»National Instruments»LabVIEW 8.6» FRC cRIO Imaging Tool** to launch the **CompactRIO Imaging Tool** dialog box. You also can display this dialog box by selecting **Tools» CompactRIO Imaging Tool** in LabVIEW.
2. Select the CompactRIO device you want to configure from the **Select CompactRIO Device** table. This table lists all CompactRIO devices connected to the host computer.
3. In the **Choose Development Environment** section, specify whether you want to run and debug LabVIEW, C/C++, or Java programs that you run on the CompactRIO device.
4. Place a checkmark in the **Format Controller** checkbox. Use the **Format Controller** section to restore an image on the CompactRIO device or update the CompactRIO device with a new name or team ID.
5. From the **Select Image** list, select the most recent `FRC_2010_xx.zip` file to download the FRC_2010_xx image to the CompactRIO device. The FRC_2010_xx image consists of a LabVIEW, C/C++, and Java program.
6. Enter the name you want to use to identify the CompactRIO device in the **Device name** text box.
7. Enter your team number in the **Team ID** field. The CompactRIO Imaging Tool sets the IP address of the CompactRIO device to `10.xx.yy.2`, where *yy* corresponds to the last two digits of the team number and *xx* corresponds to the first or first two digits of the team number.
8. Click the **Apply** button to apply the changes you made and download the FRC_2010_xx image to the CompactRIO device. Do not turn off power to the CompactRIO device or interfere with the network connection while the CompactRIO Imaging Tool downloads the image to the CompactRIO device.
9. Turn the CompactRIO device off and then turn it back on to load the new image.

If you modify the program that you run on the CompactRIO device and want to revert the changes, you can use the CompactRIO Imaging Tool to restore the FRC_2010_xx image on the CompactRIO device.

If you modify the program you run on the CompactRIO device and want to switch to the program in the other development environment, select the new development environment from the **Choose Development Environment** section and click the **Apply** button. Switching development environments does not reformat or download a new image to the CompactRIO device.

Using the Real-Time System Manager to Manage CompactRIO Device Resources

You can use the Real-Time System Manager to determine the processor load on the CompactRIO device. You may want to do this if the CompactRIO device is operating slowly. The Real-Time System Manager displays details about VIs running on a real-time (RT) target and provides a dynamic display of the memory and CPU resources for the target. You also can stop VIs and start idle VIs on the RT target using the Real-Time System Manager. From the **Project Explorer** window, select **Tools»Real-Time Module»System Manager** to launch the Real-Time System Manager.

Note When you run the Real-Time System Manager, it takes up a significant amount of the memory on the CompactRIO device.

You also can determine the processor load of the CompactRIO device by using the command `memShow` in the **Terminal** tab in Wind River Workbench or in Microsoft Hyperterminal.

Refer to the *Getting Started with the LabVIEW Real-Time Module* document, available by navigating to the `National Instruments\LabVIEW 8.6\manuals` directory and opening `RT_Getting_Started.pdf`, for more information about the Real-Time System Manager.

Refer to the *Configuring the CompactRIO Device* section of this chapter for more information about configuring the CompactRIO device.

4

Using the FRC Framework

The *FIRST* Robotics Competition (FRC) framework is a set of VIs, organized in LabVIEW projects, that you can use as a template when building a robotics application for FRC. The FRC framework consists of two project templates. Use the FRC robot project template to develop the program you want to deploy to and run on the CompactRIO device. Use the FRC dashboard project template to develop the program with which you want to view data on the host computer.

FRC Robot Project

The FRC robot project starts communication between the CompactRIO device and the driver station, initializes the user watchdog and the Axis camera, and specifies how to handle each competition mode and status of the robot.

Complete the following steps to create an FRC robot project.

1. Click the **FRC cRIO Robot Project** link in the **Getting Started** window to display the **Create New FRC Robot Project** dialog box.
2. In the **Project name** text box, enter the name you want to use to identify the new FRC robot project.
3. In the **Project folder** text box, enter the location on the host machine to which you want to save the project files and VIs.
4. In the **cRIO IP address** text box, enter the IP address of the CompactRIO device to which you want to deploy the project. The IP address of the CompactRIO device must be in the form `10.`*`xx`*`.`*`yy`*`.02`, where *yy* corresponds to the last two digits of the team number and *xx* corresponds to the remaining first or first two digits of the team number. You can use the CompactRIO Imaging Tool to set the IP address of the CompactRIO device.
5. Click the **Finish** button to close the **Create New FRC Robot Project** dialog box and create the new FRC robot project. LabVIEW displays the new FRC robot project in the **Project Explorer** window, shown as follows.

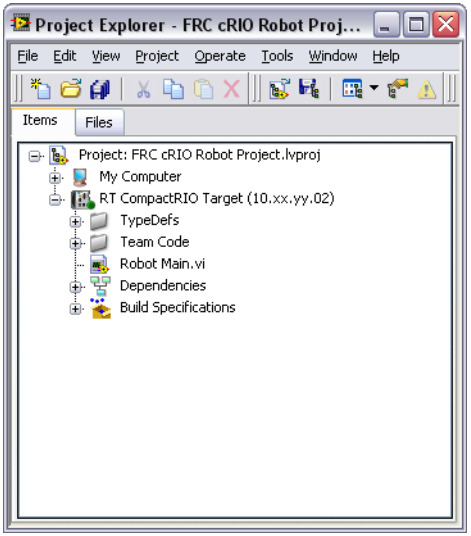


Figure 4-1. FRC Robot Projects

Notice that the FRC robot project contains two targets: **My Computer** and **RT CompactRIO Target**. Files under **My Computer** are those you want to use and run on the host computer. Files under **RT CompactRIO Target** are those you want to deploy to and run on the CompactRIO device.

The **RT CompactRIO Target** contains one top-level VI, the Robot Main VI, which is the master VI for the robot and is the top-level VI for the robotics program you run on the CompactRIO device. This target also contains a **TypeDefs** folder and a **Team Code** folder. These folders contain type definitions and subVIs that the Robot Main VI calls.

Robot Main VI

The Robot Main VI establishes communication with the driver station, acquires and processes images, and performs Autonomous or Teleop tasks depending on the competition mode.

In the **Project Explorer** window, double-click the **Robot Main.vi** item to open the Robot Main VI. Select **Window»Show Block Diagram** or press the <Ctrl-E> keys to view the block diagram. The block diagram contains a single While Loop and a number of subVIs. The While Loop determines the behavior of the robot according to the competition mode and state. When the competition state is Autonomous Enabled, the Robot Main VI calls the Autonomous Iterative VI. When the competition state is Teleop Enabled, the Robot Main VI calls the Teleop VI. If the robot is in Disabled status, the Robot Main VI calls the Disabled VI.

The While Loop also contains the Build DashBoard Data VI, which you can use to send I/O data from the CompactRIO device to the host computer. Refer to the *Build DashBoard Data VI* section of this chapter for more information about the Build DashBoard Data VI.

The other subVIs in the Robot Main VI perform tasks such as initializing data, performing time-based operations, and processing images.

After the Robot Main VI calls the Begin subVI, it uses the Start Communication VI to establish communication with the driver station. The Robot Main VI runs until the **Abort Execution** button on the VI toolbar is pressed or until the **Finish** button on the front panel is pressed. If the **Abort Execution** button is pressed, the Robot Main VI aborts immediately and does not perform any cleanup tasks. If the **Finish** button is pressed, the Robot Main VI calls the Finish VI, which you can configure to close device references, save collected data to file, or perform any other cleanup tasks.

You can modify the code within each of the subVIs in the **Team Code** folder of the FRC robot project.

Team Code VIs

The **Team Code** folder of the FRC robot project contains subVIs that the Robot Main VI calls to perform the initialization, execution, and cleanup tasks for the Autonomous and Teleop portions of the FRC competition.

Begin VI

The Begin VI initializes data for use throughout the Robot Main VI. You can create references to motors and sensors you want to use, load settings from file, and perform other initialization tasks.

By default, the Begin VI configures settings for the Autonomous program you want to run, opens a reference to the Axis camera, and configures options for the FRC dashboard project on the host computer. You can use the WPI Robotics Library VIs and other LabVIEW VIs to configure other initialization tasks.

Specify whether you want to use the Autonomous Independent VI or the Autonomous Iterative VI on the block diagram of the Begin VI. If you use the Autonomous Independent VI, you do not need to make any changes from the default values. If you use the Autonomous Iterative VI, you must select **Iterative** from the **Autonomous Style** control and change the **VI Refnum** control from a reference to the Autonomous Independent VI to a reference to the Autonomous Iterative VI. The Autonomous Independent VI is recommended over the Autonomous Iterative VI.

Create all references you want to reuse in the Begin VI before you develop the programs you want to run on the robot. Complete the following steps to create a reference.

1. In the **Project Explorer** window of the FRC robot project, within the **Team Code** folder under the **RT CompactRIO Target**, double-click the **Begin.vi** item to open the Begin VI.
2. Select **Window»Show Block Diagram** or press the <Ctrl-E> keys to view the block diagram.

   Notice that references have already been set for Camera, Watchdog, Left and Right Motors, and Joystick 1.
3. Add a Gyro Open VI to the block diagram below the Joystick 1 reference.
4. Right-click the **Analog Input** terminal of the Gyro Open VI and select **Create»Constant** from the shortcut menu.
5. Add a Gyro RefNum Registry Set VI to the block diagram to the right of the Gyro Open VI.
6. Right-click the **refnum name** input of the Gyro RefNum Registry Set VI and select **Create»Constant** from the shortcut menu.
7. Type the name you want to assign to this reference, such as `Gyro 1`.
8. Wire the **GyroDevRef** output of the Gyro Open VI to the **GyroDevRef** input of Gyro RefNum Registry Set VI.
9. Wire the **error out** output of the Gyro Open VI to the **error in** input of the Gyro RefNum Registry Set VI.

After you set references in the Begin VI, use the RefNum Registry Get VIs to reuse the references in other VIs that you create.

Autonomous Independent VI

The Autonomous Independent VI is one of two VIs you can choose to run during the Autonomous portion of the FRC competition. This VI runs for the duration of the Autonomous portion of the competition. You do not need to program the Autonomous Independent VI to stop after a certain time because the Robot Main VI terminates this VI when the competition mode changes to Teleop.

By default, the block diagram of the Autonomous Independent VI contains a Case structure. The Case structure specifies whether to run the default autonomous code for the robot. The False case of the Case structure does nothing. The True case contains the code that controls the robot when the robot is in Autonomous mode. You can delete the default autonomous code and use the WPI Robotics Library VIs or other LabVIEW VIs to specify the program you want to run during the Autonomous portion of the FRC competition.

Autonomous Iterative VI

The Autonomous Iterative VI is one of two VIs you can choose to run during the Autonomous portion of the FRC competition. This VI iterates and performs some action each time a packet arrives from the driver station.

The block diagram of the Autonomous Iterative VI consists of two Case structures. The outer Case structure specifies whether to use the Autonomous Independent VI or the Autonomous Iterative VI, as specified on the block diagram of the Begin VI, during the Autonomous portion of the FRC competition. If you choose to use the Autonomous Independent VI, the Autonomous Iterative VI does nothing by default.

If you choose to use the Autonomous Iterative VI during the Autonomous portion of the FRC competition, the inner Case structure executes according to the derived state of the robot. You can use the Init case of the inner Case structure to initialize any motors or sensors for the Autonomous portion of the competition. Alternatively, you can initialize this data in the Begin VI so that the data is available during the Teleop portion of the competition as well. The Execute case of the inner Case structure performs the action you want to take place each time a packet arrives from the driver station. You can use the Autonomous Elapsed Seconds information, wired to the left border of the Case structure, to determine the action to perform. The Stop case of the inner Case structure performs cleanup tasks, such as closing device references that are needed only during the Autonomous portion of the competition. You can use the Stop case to restore settings to the values they held after the Begin VI ran.

Disabled VI

The Disabled VI runs whenever the robot is in Disabled status. You can use this VI to calibrate sensors or the Axis camera before the FRC competition or between the Autonomous and Teleop portions of the competition.

The block diagram of the Disabled VI contains a Case structure that executes depending on the derived state of the robot. You can use the Init case of the Case structure to perform initialization tasks that apply only when the robot is in Disabled status. Alternatively, you can initialize this data in the Begin VI so that the data is available when the robot is in Enabled status as well. The Execute case of the Case structure includes any actions, such as calibrating sensors, that you want the robot to perform when the motors are disabled. The Stop case of the Case structure performs cleanup tasks, such as closing device references that are needed only when the robot is in Disabled status. You can use the Stop case to restore settings to the values they held after the Begin VI ran.

Teleop VI

The Teleop VI iterates and performs some action each time a packet specifying the Teleop mode arrives from the driver station. This VI only handles the event of the competition mode being set to **Teleop**. Use the Periodic Tasks VI to program the actual behavior of the robot during the Teleop portion of the competition.

The block diagram of the Teleop VI contains a Case structure that executes depending on the derived state of the robot. You can use the Init case of the Case structure to initialize any motors or sensors for the Teleop portion of the competition. Alternatively, you can initialize this data in the Begin VI so that the data is available during the Autonomous portion of the competition as well. By default, the Init case initializes the robot to read the value of joystick 1. The Execute case of the Case structure performs the action you want to take place each time a Teleop packet arrives from the driver station. For example, you might want to read values from a joystick, update the robot motors, or update setpoints for the periodic operations specified in the Periodic Tasks VI. The Stop case of the Case structure performs cleanup tasks, such as closing device references that are needed only during the Teleop portion of the competition. You can use the Stop case to restore settings to the values they held after the Begin VI ran.

The Teleop VI uses one or more of the references you set in the Begin VI to specify device references of the correct data types for use with the WPI Robotics Library VIs. Refer to the *Begin VI* section of this chapter for information about creating references.

Robot Global Data VI

Use the Robot Global Data VI to access and pass data among several VIs. In particular, you can use this global VI to store device reference information for the various motors and sensors of your robot.

Because global VIs only pass data and perform no computations, global VIs contain a front panel but no block diagram. By default, the Robot Global Data VI contains an **Enable Vision** front panel control.

On the block diagram of the Vision Processing VI, the **Enable Vision** global variable is wired to the Case structure that determines whether to start or stop image acquisition. The value of the **Enable Vision** control in the Robot Global Data VI determines which case of the Case structure in the Vision Processing VI to execute.

Vision Processing VI

The Vision Processing VI acquires images from the Axis camera and performs image processing. This VI runs continuously while the Robot Main VI is running.

The block diagram of the Vision Processing VI consists of a While Loop, which itself contains a Case structure. The Case structure determines whether to start or stop acquiring image data from the Axis camera and whether to process the images.

Notice the **Enable Vision** global variable wired to the Case structure. This global variable is an instance of the Robot Global Data VI. The value of the **Enable Vision** control is set in the Robot Global Data VI and then passed to the Case structure.

If the **Enable Vision** global variable is TRUE, the True case of each Case structure executes. Therefore, the Vision Processing VI acquires image data from the Axis camera, retrieves specific images from this data, and processes each image. You can use the *FIRST* Vision VIs to process the image data. If the **Enable Vision** global variable is FALSE, the Vision Processing VI neither acquires nor processes any images from the Axis camera.

You might use image processing to help determine the behavior of a robot. For example, you can use the *FIRST* Vision VIs to determine the color of an image you acquire. Depending on the color, you then can move the motors of the robot in an appropriate direction.

If you do not want to perform image acquisition continuously, you can configure the **Enable Vision** control in the Robot Global Data VI to change value depending on the competition mode, for example.

Periodic Tasks VI

The Periodic Tasks VI performs periodic tasks. For example, you might include PID VIs to perform time-based control operations. This VI runs continuously while the Robot Main VI is running.

By default, the block diagram of the Periodic Tasks VI consists of two While Loops. Each While Loop contains a Wait (ms) function that specifies the length of time to wait between each iteration of the loop. You can change the value of the **milliseconds to wait** input to specify the frequency of each periodic task.

Periodic loops often operate with setpoints from other loops. Use a global variable such as the Robot Global Data VI to share data among loops.

The Periodic Tasks VI uses one or more of the references you set in the Begin VI to specify device references of the correct data types for use with the WPI Robotics Library VIs. Refer to the *Begin VI* section of this chapter for information about creating references.

Build DashBoard Data VI

The Build DashBoard Data VI sends I/O data from the modules on the CompactRIO device through the driver station to the host computer. You can use the **Dashboard Enables** input of this VI to specify what data you want to send. By default, the **Dashboard Enables** input specifies to send raw data for the first analog module, the first digital module, and the solenoid module to the host computer. The Build DashBoard Data VI also updates data values in the Dashboard Datatype type definition, which you then can reuse in other VIs.

Finish VI

The Finish VI performs cleanup tasks before the Robot Main VI stops. This VI runs when you press the **Finish** button on the front panel of the Robot Main VI.

The block diagram of the Finish VI contains a Flat Sequence structure. In the first subdiagram of this structure, you can perform cleanup tasks such as closing device references and saving collected data to file. The second subdiagram of the Flat Sequence structure stops the Finish VI and, in turn, the Robot Main VI.

The Finish VI uses one or more of the references you set in the Begin VI to specify device references of the correct data types for use with the WPI Robotics Library VIs. Refer to the *Begin VI* section of this chapter for information about creating references.

Type Definitions

The **TypeDefs** folder of the FRC robot project contains type definitions for specifying data types that you use with the WPI Robotics Library VIs. For example, the Dashboard Datatype type definition specifies data that corresponds to the two analog modules, two digital modules, and solenoid module of the CompactRIO device. You can add or remove the controls that this type definition contains depending on the data you want to reuse. You also can create your own type definitions to reuse. Refer to Chapter 6, *Tutorial: Creating an FRC Dashboard Project*, for more information about creating and modifying type definitions.

Deploying the FRC Robot Project

After you develop the FRC robot project you want to run, you must deploy the program to the CompactRIO device. You can deploy the program in three ways: using the **Run** button; from the **Project Explorer** window; or as a stand-alone, built application.

Deploying the Program Using the Run Button

Click the **Run** button of the Robot Main VI to deploy the VI to the CompactRIO device. LabVIEW deploys the VI, all items required by the VI, and the target settings to memory on the CompactRIO device.

Note If you click the **Run** button of the Robot Main VI to deploy the VI to the CompactRIO device, the VI might run slowly. Close any subVIs of the Robot Main VI that are open on the host computer to improve performance. Do not close the Robot Main VI.

When you deploy a program with the **Run** button, you maintain a connection between the host computer and the CompactRIO device. The program runs on the CompactRIO device, but you can manipulate the front panel objects of the program from the host computer. You therefore can deploy a program with the **Run** button to perform live front panel debugging.

Note If a program is running on the CompactRIO device and you redeploy that program with the **Run** button, the CompactRIO device stops and restarts the program you deployed. LabVIEW redeploys any VIs that changed or are no longer in memory on the CompactRIO device.

Deploying the Program from the Project Explorer Window

In the **Project Explorer** window, right-click the Robot Main VI and select **Deploy** from the shortcut menu to deploy the VI and any support files for the VI to the target. VIs, libraries, and shared variables are downloaded to memory on the CompactRIO device.

When you deploy a program from the **Project Explorer** window, the program runs only on the CompactRIO device to which it was deployed. Therefore, you cannot perform live front panel debugging.

Building and Deploying a Stand-Alone Application

Build the FRC robot project into a stand-alone application that you then can deploy to the CompactRIO device. You can specify the application to run at startup so the application runs as soon as the CompactRIO is powered on. Deploy stand-alone applications to the CompactRIO device for use in the FRC competition.

Complete the following steps to build the FRC robot project into a stand-alone FRC application and run it on the CompactRIO device at startup.

1. In the **Project Explorer** window, double-click the **FRC Basic Robot Deployment** build specification or the **FRC Robot Boot-up Deployment** build specification under the **Build Specifications** folder to display the **Real-Time Application Properties** dialog box.
2. On the **Information** page, specify a name for the build specification in the **Build specification name** text box.
3. Specify a name for the application in the **Target filename** text box.
4. Specify the location on the host computer to which you want to save the stand-alone application in the **Local destination directory** field.
5. Specify the location on the CompactRIO device to which you want to save the stand-alone application in the **Target destination directory** field.
6. Select **Source Files** from the **Category** list.
7. Verify that the Basic Robot Main VI or the Robot Main VI is in the **Startup VIs** list.
8. Click the **OK** button to close the **Real-Time Application Properties** dialog box.
9. In the **Project Explorer** window, right-click the build specification and select **Build** from the shortcut menu to build the application.
10. Right-click the build specification and select **Run as startup** from the shortcut menu to set the application as the startup application and deploy the application to the CompactRIO device. LabVIEW prompts you to reboot the RT target.
11. Reboot the CompactRIO device to run the application.

Note If you no longer want the application to run on the CompactRIO device at startup, right-click the build specification and select **Unset as startup** from the shortcut menu.

Connecting to the CompactRIO Device

You can connect to the CompactRIO device and access the front panels of VIs in memory on the device. First deploy the VI to the CompactRIO device using the **Run** button, as described in the *Deploying the Program Using the Run Button* section of this chapter. If you stop the VI or close the front panel, the VIs are removed from memory on the CompactRIO device. However, if you only disconnect from the CompactRIO device, the front panel on the host computer appears to stop, but the VI continues running on the CompactRIO device. Right-click the **RT CompactRIO Target** item in the **Project Explorer** window and select **Disconnect** from the shortcut menu to disconnect from the CompactRIO device. If you then close the front panel and reconnect to the CompactRIO device, you re-access the front panels in memory on the device. The front panel of the running VI reappears and displays the current state of the VI. Right-click the **RT CompactRIO Target** item in the **Project Explorer** window and select **Connect** from the shortcut menu to connect to the CompactRIO device.

Note You cannot access the front panels of VIs in memory on a CompactRIO device if a built application is running. You first must stop the running built application or cancel the **Connect** operation.

FRC Dashboard Project

Use the FRC dashboard project on the host computer to view data that the CompactRIO device returns. This project can display images and I/O values that the CompactRIO device sends to the host computer.

Complete the following steps to create an FRC dashboard project.

1. Click the **FRC Dashboard Project** link in the **Getting Started** window to display the **Create New FRC Dashboard Project** dialog box.
2. In the **Project name** text box, enter the name you want to use to identify the new FRC dashboard project.
3. In the **Project folder** text box, enter the location on the host machine to which you want to save the project files and VIs.
4. Click the **Finish** button to close the **Create New FRC Dashboard Project** dialog box and create the new FRC dashboard project. LabVIEW displays the new FRC dashboard project in the **Project Explorer** window.

The FRC dashboard project looks similar to the following figure:

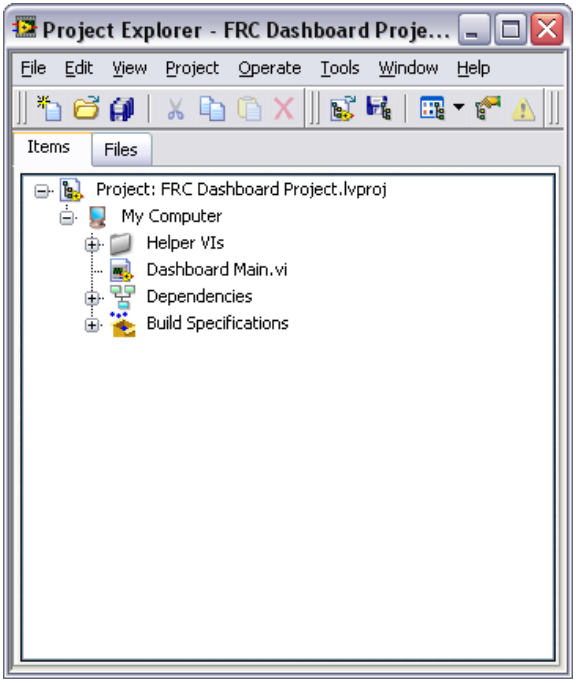


Figure 4-2. FRC Dashboard Project

Whereas the FRC robot project contains two targets, the FRC dashboard project contains only the **My Computer** target. You run the VIs in the FRC dashboard project only on a host computer, such as on a host computer connected to the driver station. You do not deploy these VIs to the CompactRIO device.

The **My Computer** target contains a Dashboard Main top-level VI and a **Helper VIs** folder. The **Helper VIs** folder contains subVIs that the Dashboard Main VI calls. You might not need to modify any of the subVIs in this folder.

The Dashboard Main VI is the master VI in the FRC dashboard project. You can use this VI on the host computer to view image data that the camera connected to the CompactRIO device acquires.

Note You can send image data to the host computer only during development, not during the FRC competition.

You also can use the Dashboard Main VI to read information about the robot, such as error information, robot status, and battery level.

In the **Project Explorer** window, double-click the **Dashboard Main.vi** item to open the Dashboard Main VI. By default, the front panel displays the following information:

- **Image**—Displays the latest image that the camera on the CompactRIO device acquired. The **Video Enable** Boolean control in this section turns image display on or off. Image display might slow performance.
- **Slot 1/Slot 2**—Displays analog input values from the NI 9201 modules in slots 1 and 2 of the CompactRIO device.
- **Slot 4/Slot 6**—Displays digital input or digital output values from the NI 9403 modules in slots 4 and 6 of the CompactRIO device.
- **Slot 8**—Displays digital output values from the NI 9472 module in slot 8 of the CompactRIO device. This module often is used to control a solenoid.
- **Battery level**—Displays the battery level of the robot.
- **Communications**—Displays input and output values from the driver station.
- **User Data**—Displays user-defined data if you place a checkmark in the **Log** checkbox. Use the Set High Priority Dashboard Data VI or the Set Low Priority Dashboard Data VI in the FRC robot project to specify the data you want to send from the CompactRIO device to the Dashboard Main VI.
- **Error Messages**—Displays error messages the CompactRIO device sends to the driver station.
- **Match Information**—Displays the competition mode (Autonomous or Teleop), elapsed time in that mode, and robot status (Enabled or Disabled). The elapsed time starts when the Dashboard Main VI runs and resets when the competition mode changes.
- **Team Logo**—Displays the image saved as `Team Logo.png` or `Team Logo.jpg` in the project directory. You can modify this image to display a logo unique to your FRC team.

Select **Window»Show Block Diagram** or press the <Ctrl-E> keys to view the block diagram of the Dashboard Main VI. The block diagram contains two While Loops.

In the first While Loop, the Dashboard Main VI retrieves specific images on the host computer from the image data that the CompactRIO device sends. By default, the Dashboard Main VI creates a JPEG image called **Host Camera Image**, continuously replaces this image with the most recent image data from the camera on the CompactRIO device, and displays the image in the **Image** indicator on the front panel.

In the second While Loop, the Dashboard Main VI receives data about the robot from the driver station. By default, the Dashboard Main VI connects to the driver station through a UDP connection and acquires status information and I/O data about the robot.

You can use the WPI Robotics Library VIs and other LabVIEW VIs to modify the types of data the Dashboard Main VI displays.

5

Tutorial: Creating an FRC Robot Project

This chapter describes how to create, modify, and deploy a *FIRST* Robotics Competition (FRC) robot project to the CompactRIO device. In this tutorial, you develop a program to perform tank driving with two joysticks and Jaguar motor controllers.

Creating an FRC Robot Project

Complete the following steps to create an FRC robot project.

1. Click the **FRC cRIO Robot Project** link in the **Getting Started** window to display the **Create New FRC Robot Project** dialog box.
2. In the **Project name** text box, enter the name you want to use to identify the new FRC robot project.
3. In the **Project folder** text box, enter the location on the host machine to which you want to save the project files and VIs.
4. In the **cRIO IP address** text box, enter the IP address of the CompactRIO device to which you want to deploy the project. The IP address of the CompactRIO device must be in the form `10.`*`xx`*`.`*`yy`*`.2`, where *yy* corresponds to the last two digits of the team number and *xx* corresponds to the remaining first or first two digits of the team number.

   You can use the CompactRIO Imaging Tool to set the IP address of the CompactRIO device. Refer to the *Running the CompactRIO Imaging Tool* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the CompactRIO device.
5. Click the **Finish** button to close the **Create New FRC Robot Project** dialog box and create the new FRC robot project.

LabVIEW displays the new FRC robot project in the **Project Explorer** window.

Running the FRC Robot Project

You can deploy the FRC robot project to the CompactRIO device before making any modifications. In the **Project Explorer** window, right-click the **Robot Main.vi** item and select **Run** from the shortcut menu. LabVIEW deploys the Robot Main VI and any support files for the VI to the CompactRIO device. The Robot Main VI then runs on the CompactRIO device. If the robot has a joystick connected to port 1 of the driver station and Jaguar motor controllers controlling the two wheels, you can move the joysticks and observe how the robot responds.

You also can run the FRC robot project on the CompactRIO device and maintain a connection with the host computer to perform live front panel programming and debugging. By maintaining a connection with the host computer, you can monitor indicators and observe how changes to the front panel of VIs affect the behavior of the robot.

Complete the following steps to run the FRC robot project and perform live front panel debugging.

1. In the **Project Explorer** window, double-click the **Robot Main.vi** item to open the Robot Main VI.
2. Click the **Run** button of the Robot Main VI to deploy the VI to the CompactRIO device. LabVIEW deploys the VI, all items required by the VI, and the target settings to memory on the CompactRIO device.
3. Move the joysticks and observe how the robot responds.
4. Click the **Abort** button of the Robot Main VI. Notice that the VI stops. When you deploy a program with the **Run** button, the program runs on the CompactRIO device, but you can manipulate the front panel objects of the program from the host computer.

Note If you redeploy the Robot Main VI with the **Run** button, the CompactRIO device stops and restarts the Robot Main VI. LabVIEW redeploys any VIs that changed or are no longer in memory on the CompactRIO device.

Refer to the *Deploying the FRC Robot Project* section of Chapter 4, *Using the FRC Framework*, for more information about deploying an FRC Robot Project.

Creating a Stand-Alone FRC Application

You can build the FRC robot project into a stand-alone application that you then can deploy to the CompactRIO device. You can specify the application to run at startup so the application runs as soon as the CompactRIO is powered on. You must deploy stand-alone applications to the CompactRIO device for use in the FRC competition.

Use **Build Specifications** in the **Project Explorer** window to create build specifications for source distributions, such as stand-alone applications. A build specification contains all the settings for the build, such as files to include, directories to create, and settings for VIs.

The default FRC robot project includes the **FRC Basic Robot Deployment** build specification. Use the **Real-Time Application Properties** dialog box to define and modify the settings for a build specification. In the **Project Explorer** window, double-click the **FRC Basic Robot Deployment** build specification under **Build Specifications** to display the **Real-Time Application Properties** dialog box.

Use the **Information** page of the **Real-Time Application Properties** dialog box to specify a name, executable filename, local destination, and target destination for a stand-alone real-time application. You also can compose a description of the build specification. On the Information page, notice that the **Build specification name** is `FRC Basic Robot Deployment`. Notice also the **Local destination directory** path. You might need to change this path to an appropriate location on the host computer to which you want to save the application.

Use the **Source Files** page of the **Real-Time Application Properties** dialog box to specify which VIs automatically run when the application launches and which VIs always are deployed to the CompactRIO device. On the **Source files** page, notice that the Basic Robot Main VI is configured by default to run when the application launches.

When you verify the build specification settings, complete the following steps to build the FRC robot project into a stand-alone FRC application and run it on the CompactRIO device at startup.

1. In the **Project Explorer** window, right-click the **FRC Basic Robot Deployment** build specification under **Build Specifications** and select **Build** from the shortcut menu to build the application.
2. Right-click the build specification and select **Run as startup** from the shortcut menu to set the application as the startup application and

deploy the application to the CompactRIO device. LabVIEW prompts you to reboot the real-time target.

3. Reboot the CompactRIO device to run the application.

Note If you no longer want the application to run on the CompactRIO device at startup, right-click the build specification and select **Unset as startup** from the shortcut menu.

4. Move the two joysticks and observe how the motors of the robot respond.

6

Tutorial: Creating an FRC Dashboard Project

This chapter describes how to send data to the Dashboard Main VI and how to modify the *FIRST* Robotics Competition (FRC) dashboard project.

The Dashboard Main VI in the FRC dashboard project displays information about the robot, such as error information, robot status, and battery level. You can use the Dashboard Main VI during robot testing to receive live feedback from the robot. The Dashboard Main VI displays the current input and output values from the CompactRIO device, the robot, and the driver station. This VI also displays user-defined data. Refer to the *FRC Dashboard Project* section of Chapter 4, *Using the FRC Framework*, for more information about the Dashboard Main VI.

The **User Data** section of the Dashboard Main VI displays user-defined messages and information. In this tutorial, you learn how to display gyroscope information from the robot in the **User Data** section of the Dashboard Main VI.

Displaying Gyroscope Data in the Dashboard Main VI

By default, the Dashboard Main VI does not display gyroscope data. You must acquire this data from a gyroscope, send the data to the Dashboard Main VI, and then modify the Dashboard Main VI to display this data.

Sending Gyroscope Data to the Dashboard Main VI

The Gyro Example VI, accessible by clicking the **Gyro Example** link in the **Getting Started** window, acquires information from a gyroscope. You can modify the Gyro Example VI to send the gyroscope information to the Dashboard Main VI.

Complete the following steps to modify the Gyro Example VI to send data to the Dashboard Main VI.

1. Click the **Gyro Example** link in the **Getting Started** window to display the Gyro Example FRC robot project in the **Project Explorer** window.
2. Select **File»Save As** to display the **Save As** dialog box.
3. Select **Copy** and click the **Continue** button to create a copy of the `.lvproj` file on disk.
4. Save the project as `Modified Gyro Example.lvproj` in an easily accessible location.
5. Close the **Project Explorer** window for the Gyro Example FRC robot project. Do not save any changes.
6. Open the Modified Gyro Example project you saved to display the Modified Gyro Example FRC robot project in the **Project Explorer** window.
7. Double-click the **Gyro Example.vi** item in the **Project Explorer** window to open the Gyro Example VI. This VI displays the current angle of the gyroscope you specify.
8. Select **Window»Show Block Diagram** or press the <Ctrl-E> keys to display the block diagram.

Note You might need to resize the While Loop and clear some space below the Quotient & Remainder function on the block diagram of the Gyro Example VI before completing the following steps.

9. Place a Number To Exponential String function below the Quotient & Remainder function.
10. Wire the **x-y*floor(x/y)** output of the Quotient & Remainder function to the **number** input of the Number To Exponential String function. The **x-y*floor(x/y)** output corresponds to the gyroscope angle.
11. Place a String To Byte Array function to the right of the Number To Exponential String function.
12. Wire the **E-format string** output of the Number To Exponential String function to the **string** input of the String To Byte Array function.
13. Place a Set High Priority Dashboard Data VI or a Set Low Priority Dashboard Data VI to the right of the String To Byte Array function. These VIs specify the user data that the Dashboard Main VI receives.
14. Wire the **unsigned byte array** output of the String To Byte Array function to the **User Data (984 bytes)** input of the Set User Data VI.

The affected portion of the block diagram should appear similar to the following figure.

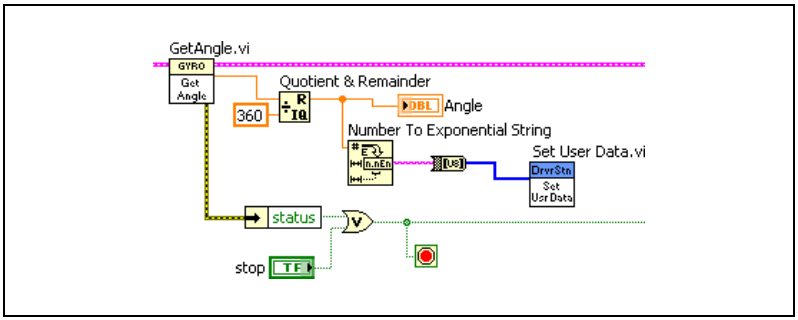


Figure 6-1. Sending Gyroscope Data to the Dashboard Main VI

15. Select **File»Save As** and save the VI as `User Data Gyro Example.vi` in an easily accessible location.

The default Gyro Example VI returns the gyroscope angle as a double-precision, floating-point data type. The modified VI converts this data to an array of bytes, which the Set User Data VI can accept. The Set User Data VI then sends this data to the Dashboard Main VI.

Adding an Indicator to the Dashboard Main VI

The Gyro Example VI now sends the gyroscope data to the Dashboard Main VI. This section describes how to set up the Dashboard Main VI to receive and display the user data. Complete the following steps to display the gyroscope data in the Dashboard Main VI.

1. Select **View»Getting Started Window** to display the **Getting Started** window.
2. Click the **FRC Dashboard Project** link in the **Getting Started** window to display the **Create New FRC Dashboard Project** dialog box.
3. Enter `FRC Dashboard Project User Data Example` in the **Project name** text box of the **Create New FRC Dashboard Project** dialog box.
4. Click the **Finish** button in the **Create New FRC Dashboard Project** dialog box to create a new FRC dashboard project.
5. Double-click the **Dashboard Main.vi** item in the **Project Explorer** window to open the Dashboard Main VI.

6. Place a **Gauge** indicator on the front panel below the **User Data** section of the front panel.
7. Right-click the **Gauge** indicator and select **Properties** from the shortcut menu to display the **Knob Properties** dialog box.
8. On the **Appearance** page, enter `Gyroscope Angle` in the **Label Text** text box to change the name of the indicator.
9. On the **Scale** page, enter `360` as the **Maximum** value in the **Scale Range** section. The **Gyroscope Angle** indicator now displays the heading of the gyroscope within a range of 360 degrees.
10. Click the **OK** button to apply the new value.
11. Hover over the 360 marker on the **Gyroscope Angle** indicator and use the Operating tool to drag the marker to the 0 value. When you drag a marker, the cursor changes to a circular arrow to indicate the tool is over a marker.
12. Right-click the **Gyroscope Angle** indicator and select **Visible Items»Digital Display** from the shortcut menu to add a digital display to the **Gyroscope Angle** indicator.

The affected portion of the front panel should appear similar to the following figure.

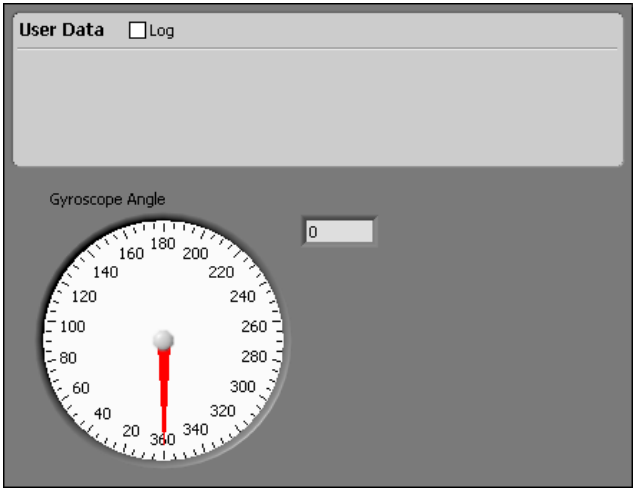


Figure 6-2. Displaying Gyroscope Data with a Gauge Indicator

13. Double-click the **Gyroscope Angle** indicator to display the location of the **Gyroscope Angle** indicator on the block diagram.

Note You might need to resize the While Loop and clear some space to the right of the **Battery level** indicator on the block diagram of the Dashboard Main VI before completing the following steps.

14. Place the **Gyroscope Angle** indicator in the second While Loop of the block diagram to the right of the **Battery level** indicator.
15. Place an Unbundle function between the **Battery level** indicator and the **Gyroscope Angle** indicator.
16. Wire the **Strings** output of the Receive DS Packet VI to the **cluster** input of the Unbundle function. The Receive DS Packet VI receives the gyroscope data that the Gyro Example VI sends.
17. Place a Fract/Exp String To Number function between the Unbundle function and the **Gyroscope Angle** indicator.
18. Wire the **User Data** output of the Unbundle function to the **string** input of the Fract/Exp String To Number function.
19. Wire the **number** output of the Fract/Exp String To Number function to the **Gyroscope Angle** indicator.

The affected portion of the block diagram should appear similar to the following figure.

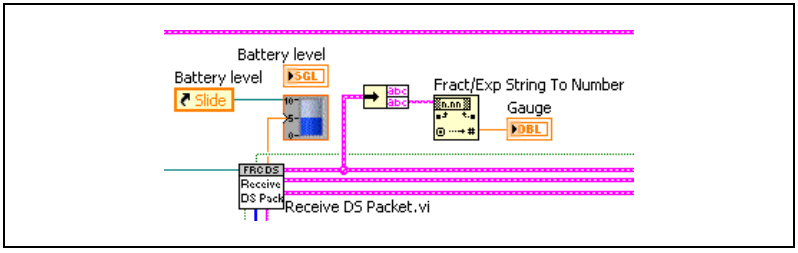


Figure 6-3. Displaying Gyroscope Data in the Dashboard Main VI

20. Select **File»Save As** and save the VI as `User Data Dashboard Main.vi` in an easily accessible location.

The User Data Dashboard Main VI now receives the gyroscope data from the User Data Gyro Example VI and displays the data in the **Gyroscope Angle** indicator on the front panel.

Running the Dashboard Main VI

Complete the following steps to run the User Data Gyro Example and User Data Dashboard Main VIs.

1. Switch to the User Data Gyro Example VI.
2. Verify that you configured the IP address of the CompactRIO device correctly. Refer to the *Configuring the CompactRIO Device* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the IP address of the CompactRIO device.
3. Click the **Run** button to deploy the User Data Gyro Example VI to the CompactRIO device. The User Data Gyro Example VI runs on the CompactRIO device and sends gyroscope data to the Dashboard Main VI.
4. Switch to the User Data Dashboard Main VI.
5. Verify that you configured the IP address of the host computer correctly. Refer to the *Setting the Static IP Address of the Computer* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the IP address of the host computer.
6. Click the **Run** button to run the User Data Dashboard Main VI on the host computer. This VI now reflects the changes in the angle of the gyroscope.
7. Stop and close all VIs and projects.

This tutorial demonstrated how to send gyroscope data to the Dashboard Main VI. Use the techniques in this tutorial to display any user data in the Dashboard Main VI.

Refer to the *FRC Dashboard Project* section of Chapter 4, *Using the FRC Framework*, for more information about the FRC dashboard project.

Displaying Multiple Data Values in the Dashboard Main VI

In the *Adding an Indicator to the Dashboard Main VI* section of this chapter, you learned how to display gyroscope data in the Dashboard Main VI. In that tutorial, you displayed only one value in the Dashboard Main VI. In this section, you learn how to display multiple values in the Dashboard Main VI using a custom control.

Creating a Custom Control

You can use a custom control to convert multiple data values that the Gyro Example VI sends to the Dashboard Main VI into a data type the Dashboard Main VI can accept. Complete the following steps to create a custom control.

1. Select **File»New** to display the **New** window.
2. Select **Other Files»Custom Control** from the **Create New** list and click the **OK** button to create a custom control. A custom control extends the available set of front panel objects for use in another VI.
3. Select **Strict Type Def.** from the **Control Type** pull-down menu on the toolbar. Specifying the control type as **Strict Type Def.** creates a custom control that reflects any data type changes you make to the custom control in all instances of every VI the control. Also, cosmetic changes you make to a strict type definition affect all instances of the strict type definition.
4. Place a cluster shell on the front panel.
5. Add a numeric control and a string control to the cluster.

The custom control should appear similar to the following figure.

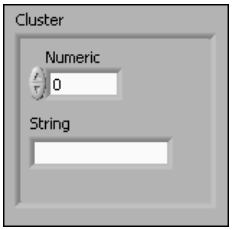


Figure 6-4. Strict Type Definition with Two Data Values

6. Save this control as `myTypeDef.ctl` in an easily accessible location.

Sending Multiple Data Values to the Dashboard Main VI

The Gyro Example VI, accessible by clicking the **Gyro Example** link in the **Getting Started** window, acquires information from a gyroscope. You can modify the Gyro Example VI to send this information to the Dashboard Main VI.

Complete the following steps to modify the Gyro Example VI to send multiple data values to the Dashboard Main VI.

1. Select **View»Getting Started Window** to display the **Getting Started** window.
2. Click the **Gyro Example** link in the **Getting Started** window to display the Gyro Example FRC robot project in the **Project Explorer** window.
3. Select **File»Save As** to display the **Save As** dialog box.
4. Select **Copy** and click the **Continue** button to create a copy of the `.lvproj` file on disk.
5. Save the project as `Multiple Data Gyro Example.lvproj` in an easily accessible location.
6. Close the **Project Explorer** window for the Gyro Example FRC robot project. Do not save any changes.
7. Open the Multiple Data Gyro Example project you saved to display the Multiple Data Gyro Example FRC robot project in the **Project Explorer** window.
8. Double-click the **Gyro Example.vi** item in the **Project Explorer** window to open the Gyro Example VI. This VI displays the current angle of the gyroscope you specify.
9. Select **Window»Show Block Diagram** or press the <Ctrl-E> keys to display the block diagram.

Note You might need to resize the While Loop and clear some space below the Quotient & Remainder function on the block diagram of the Multiple Data Gyro Example VI before completing the following steps.

10. Place a Bundle By Name function below the **Angle** indicator on the block diagram of the Gyro Example VI.
11. Resize the Bundle By Name function to display two elements.
12. Select **Select a VI** on the **Functions** palette. Navigate to the **myTypeDef** control you created and click the **OK** button.
13. Place the **myTypeDef** control below the Quotient & Remainder function.

Note You also can place the **myTypeDef** control on the block diagram by dragging the control icon from the upper right corner of the **Control Editor** window to the block diagram of the Multiple Data Gyro Example VI.

14. Wire the **myTypeDef** control to the **input cluster** input of the Bundle By Name function.
15. Click the second **Numeric** element of the Bundle By Name function and select **String** from the shortcut menu.

The affected portion of the block diagram should appear similar to the following figure.

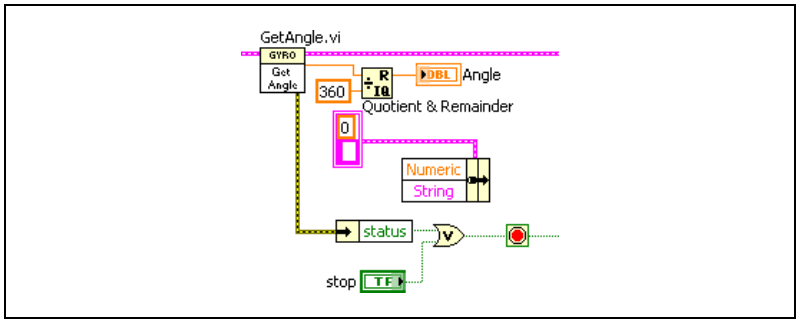


Figure 6-5. Bundling Elements for the myTypeDef Strict Type Definition

16. Wire the **x-y*floor(x/y)** output of the Quotient & Remainder function to the **Numeric** element of the Bundle By Name function.
17. Place a string constant to the left of the Bundle By Name function and enter `Example`.
18. Wire the string constant to the **String** input of the Bundle By Name function. You now can send both the gyroscope angle and this string to the Dashboard Main VI.
19. Place a Flatten To String function to the right of the Bundle By Name function.
20. Wire the **output cluster** output of the Bundle By Name function to the **anything** input of the Flatten To String function.
21. Place a String To Byte Array function to the right of the Flatten To String function.
22. Wire the **data string** output of the Flatten To String function to the **string** input of the String To Byte Array function.

23. Place a Set User Data VI to the right of the String To Byte Array function. This VI specifies the user data that the Dashboard Main VI receives.
24. Wire the **unsigned byte array** output of the String To Byte Array function to the **User Data (984 bytes)** input of the Set User Data VI.

The affected portion of the block diagram should appear similar to the following figure.

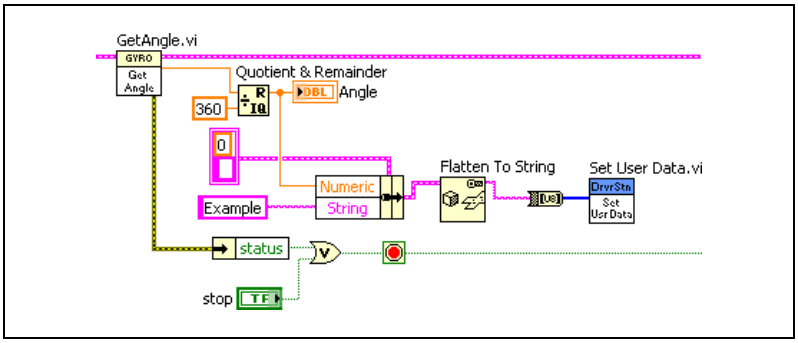


Figure 6-6. Sending Multiple Data Values to the Dashboard Main VI

25. Select **File»Save As** and save the VI as `Multiple Data Gyro Example.vi` in an easily accessible location.

The default Gyro Example VI returns the gyroscope angle as a double-precision, floating-point data type and does not return any string data. The modified VI converts the gyroscope angle and string data to an array of bytes, which the Set User Data VI can accept. The Set User Data VI then sends these multiple data values to the Dashboard Main VI.

Adding Indicators to the Dashboard Main VI

The Multiple Data Gyro Example VI now sends the gyroscope data to the Dashboard Main VI. This section describes how to set up the Dashboard Main VI to receive and display the user data. Complete the following steps to display multiple data values from the Multiple Data Gyro Example VI in the Dashboard Main VI.

1. Select **View»Getting Started Window** to display the **Getting Started** window.
2. Click the **FRC Dashboard Project** link in the **Getting Started** window to display the **Create New FRC Dashboard Project** dialog box.

3. Enter `FRC Dashboard Project Multiple Data Example` in the **Project name** text box of the **Create New FRC Dashboard Project** dialog box.
4. Click the **Finish** button in the **Create New FRC Dashboard Project** dialog box to create a new FRC dashboard project.
5. Double-click the **Dashboard Main.vi** item in the **Project Explorer** window to open the Dashboard Main VI.
6. Place a **Gauge** indicator on the front panel below the **User Data** section of the front panel.
7. Right-click the **Gauge** indicator and select **Properties** from the shortcut menu to display the **Knob Properties** dialog box.
8. On the **Appearance** page, enter `Gyroscope Angle` in the **Label Text** text box to change the name of the indicator.
9. On the **Scale** page, enter `360` as the **Maximum** value in the **Scale Range** section. The **Gyroscope Angle** indicator now displays the heading of the gyroscope within a range of 360 degrees.
10. Click the **OK** button to apply the new value.
11. Hover over the 360 marker on the **Gyroscope Angle** indicator and use the Operating tool to drag the marker to the 0 value. When you drag a marker, the cursor changes to a circular arrow to indicate the tool is over a marker.
12. Right-click the **Gyroscope Angle** indicator and select **Visible Items»Digital Display** from the shortcut menu to add a digital display to the **Gauge** indicator.
13. Place a **String** indicator above the **Gyroscope Angle** indicator.
14. Select **Window»Show Block Diagram** or press the <Ctrl-E> keys to display the block diagram.

Note You might need to expand the second While Loop and clear some space to the right of the **Battery level** indicator on the block diagram of the Dashboard Main VI before completing the following steps.

15. Place the **Gyroscope Angle** and **String** indicators in the second While Loop of the block diagram to the right of the **Battery level** indicator.
16. Place an Unbundle function between the **Battery level** indicator and the **Gyroscope Angle** and **String** indicators.
17. Wire the **Strings** output of the Receive DS Packet VI to the **cluster** input of the Unbundle function.

18. Select **Select a VI** on the **Functions** palette. Navigate to the **myTypeDef** control you created and click the **OK** button.
19. Place the **myTypeDef** above the Unbundle function.
20. Place an Unflatten From String function between the Unbundle function and the **Gyroscope Angle** and **String** indicators.
21. Wire the **myTypeDef** constant to the **type** input of the Unflatten From String function.
22. Wire the **User Data** output of the Unbundle function to the **binary string** input of the Unflatten From String function.
23. Place an Unbundle By Name function between the Unflatten From String function and the **Gyroscope Angle** and **String** indicators. Resize the Unbundle By Name function to display two elements.
24. Wire the **value** output of the Unflatten From String function to the **input cluster** input of the Unbundle By Name function.
25. Click the second **Numeric** element of the Unbundle By Name function and select **String** from the shortcut menu.
26. Wire the **Numeric** element of the Unbundle By Name function to the **Gyroscope Angle** indicator.
27. Wire the **String** element of the Unbundle By Name function to the **String** indicator.

The affected portion of the block diagram should appear similar to the following figure.

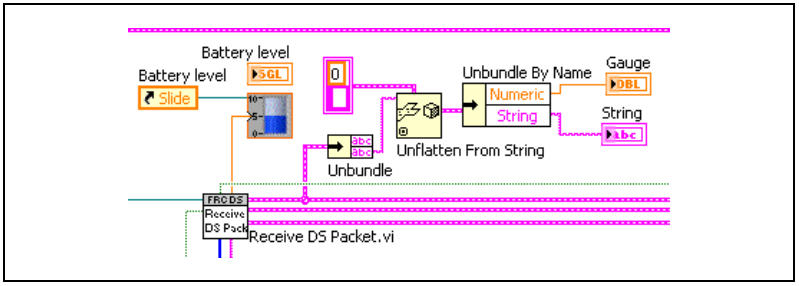


Figure 6-7. Displaying Multiple Data Values in the Dashboard Main VI

28. Select **File»Save As** and save the VI as `Multiple Data Dashboard Main.vi` in an easily accessible location.

The Multiple Data Dashboard Main VI now receives the gyroscope angle and string data from the Multiple Data Gyro Example VI and displays the data in the **User Data** section of the front panel.

Running the Dashboard Main VI

Complete the following steps to run the Multiple Data Gyro Example and Multiple Data Dashboard Main VIs.

1. Switch to the Multiple Data Gyro Example VI.
2. Verify that you configured the IP address of the CompactRIO device correctly. Refer to the *Configuring the CompactRIO Device* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the IP address of the CompactRIO device.
3. Click the **Run** button to deploy the Multiple Data Gyro Example VI to the CompactRIO device. The Multiple Data Gyro Example VI runs on the CompactRIO device and sends both the angle and string data to the Dashboard Main VI.
4. Switch to the Multiple Data Dashboard Main VI.
5. Verify that you configured the IP address of the host computer correctly. Refer to the *Setting the Static IP Address of the Computer* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the IP address of the host computer.
6. Click the **Run** button to run the Multiple Data Dashboard Main VI on the host computer. This VI now reflects the changes in the angle of the gyroscope and shows the **Example** string value.
7. Stop and close all VIs and projects.

This tutorial demonstrated how to send multiple data values to the Dashboard Main VI. Use the techniques in this tutorial to display any user data in the Dashboard Main VI.

Refer to the *FRC Dashboard Project* section of Chapter 4, *Using the FRC Framework*, for more information about the FRC dashboard project.

7

Troubleshooting the FRC Robot

In previous chapters, you learned how to program a robot using the CompactRIO device, the *FIRST* Robotics Competition framework, and the WPI Robotics Library VIs. However, at times, the robot might not work as you expect. Table 7-1 describes common issues you might encounter when working with the robot and solutions to address those issues.

Table 7-1. Common Programming Issues

Issue	Solution
The CompactRIO Imaging Tool does not run on some computers with two Internet connections.	Disable one of the Internet connections and run the CompactRIO Imaging Tool again. For example, if a computer has both a wireless Internet connection and a wired connection, disable the wireless connection and run the CompactRIO Imaging Tool again. Refer to the *Running the CompactRIO Imaging Tool* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the CompactRIO device.
The motors on the FRC robot do not run when I attempt to run them.	Ensure that the user watchdog is enabled and that you configured the Watchdog VIs to feed the user watchdog. You can use the SetEnabled VI to specify whether the user watchdog is enabled. If you enable the user watchdog, you must specify a timeout period and ensure that the program you write feeds the watchdog regularly. The user watchdog is enabled by default. If you do not feed the user watchdog during the period you specify, the watchdog disables the PWM, relay, and solenoid outputs of the CompactRIO device. Refer to the *FIRST* Robotics Competition VIs on `ni.com` for more information about the WPI Robotics Library Watchdog VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**. Additionally, use the Start Communication VI to set up communications between the driver station and the CompactRIO. Ensure that the program that you run on the CompactRIO includes this VI. Otherwise, the system watchdog shuts down the motors. Refer to the *FIRST* Robotics Competition VIs on `ni.com` for more information about the DriverStation VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

Table 7-1. Common Programming Issues (Continued)

Issue	Solution
When I try to deploy a program to the CompactRIO device, the program does not run. Sometimes the FRC robot begins operation although I did not run a program.	Launch the **CompactRIO Imaging Tool** dialog box and examine the **Choose Development Environment** section. Ensure that you select the development environment in which you want to work, then redeploy the program. If you specify a development environment different from the one in which you want to work, the program from the other development environment might run instead and prevent you from accessing the CompactRIO device. Refer to the *Running the CompactRIO Imaging Tool* section of Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about the CompactRIO Imaging Tool.
I cannot access the Axis camera using the *FIRST* Vision VIs.	Ensure the Axis camera is connected to the CompactRIO device using the orange Ethernet crossover cable in the FRC kit. Refer to Chapter 3, *Configuring the Camera and the CompactRIO Device*, for more information about configuring the Camera. Refer to the *FIRST* Robotics Competition VIs on `ni.com` for information about the *FIRST* Vision VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.
When I try to run the CompactRIO Imaging Tool, the **CompactRIO Imaging Tool** dialog box does not list any CompactRIO devices connected to the host computer.	Check the network firewall and ensure that the firewall allows the computer to access the CompactRIO device. You might need to disable the firewall for the CompactRIO Imaging Tool to run with no errors.
When I run the CompactRIO Imaging Tool, the tool stops while downloading an image to the CompactRIO device.	Ensure that the network firewall allows the computer to access the CompactRIO device with both an `0.0.0.0` IP address and an `10.xx.yy.02` IP address. During the imaging process, the CompactRIO Imaging Tool sets the IP address of the CompactRIO device to `0.0.0.0`. You might need to disable the firewall for the CompactRIO Imaging Tool to run with no errors.

8

Using the WPI Robotics Library VIs

Use the WPI Robotics Library VIs to interface with the CompactRIO device and perform tasks such as reading and writing data to sensors and driving motors.

Refer to the *FIRST* Robotics Competition VIs on ni.com for reference information about the WPI Robotics Library VIs. You also can access this information in the *LabVIEW Help*, available by selecting **Help»Search the LabVIEW Help**.

Reference Clusters

Many of the WPI Robotics Library VIs contain input and output reference clusters, such as **CompressorDevRef**, **RelayDevRef**, and so on. Use these reference clusters to pass information about a specific sensor or module between VIs.

For example, the following figure illustrates how to open a reference to an encoder, start the same encoder, stop the encoder, and then close the corresponding reference.

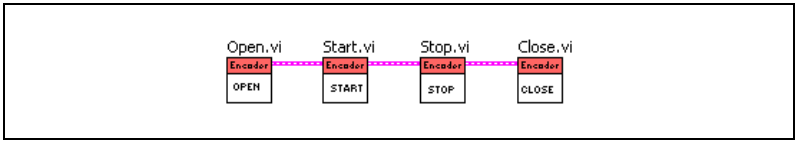


Figure 8-1. Using Reference Clusters with the Encoder VIs

First, use the Open VI on the **Encoder** palette to open a reference to an encoder. Opening a reference for an encoder reserves an available encoder index for that encoder. If all encoder indexes are reserved already, you cannot open a reference for the encoder until you close an existing encoder reference. The Open VI returns an **EncoderDevRef** output reference cluster that identifies the encoder for which you opened the reference. Wire the **EncoderDevRef** output reference cluster to the **EncoderDevRef** input reference cluster of the Start VI to specify this encoder as the one you want to start. Similarly, wire the **EncoderDevRef** output reference cluster of the Start VI to the **EncoderDevRef** input reference of the Stop VI to specify the same

encoder as the one you want to stop. Finally, wire the **EncoderDevRef** output reference cluster of the Stop VI to the **EncoderDevRef** input reference cluster of the Close VI to close the corresponding reference.

Wiring the **EncoderDevRef** reference cluster between VIs establishes a reference to the same encoder for each VI. By using the reference cluster, you do not have to specify the same information about the encoder for each VI.

Caution Do *not* manually specify any information in a reference cluster. Always use a corresponding Open VI for the sensor or module to create the reference cluster that you then can wire to other VIs.

All input and output reference clusters contain at least a **DevStatus** cluster, which contains error information and is similar to the LabVIEW error cluster. Refer to the *Getting Started with LabVIEW for the FIRST Robotics Competition* manual, available by navigating to the `National Instruments\LabVIEW 8.6\manuals` directory and opening `FRC_Getting_Started.pdf`, for more information about the LabVIEW error cluster.

Some reference clusters also contain additional controls or indicators unique to the sensor or module to which they apply. For example, the **EncoderDevRef** reference cluster, shown as follows, contains a **DevStatus** cluster as well four encoder-specific controls: **EncoderIndex**, **CounterIndex**, **DistancePerCount**, and **Decoding Type**.

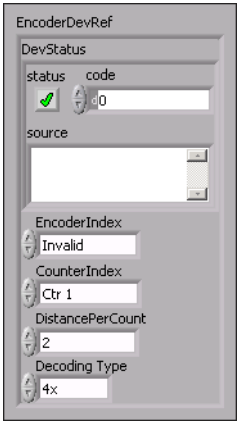


Figure 8-2. EncoderDevRef Input Reference Cluster

EncoderIndex specifies the index of the reserved encoder. Therefore, **EncoderIndex** establishes a reference to a particular encoder.

Error Handling

When you use reference clusters to connect VIs, you also pass error information between those VIs. You can use this error information to troubleshoot the application.

In Figure 8-1, *Using Reference Clusters with the Encoder VIs*, if the Open VI runs normally, the **DevStatus** cluster of the **EncoderDevRef** output reference cluster is empty. The Open VI therefore does not pass any errors to the Start VI, and the **DevStatus** cluster of the **EncoderDevRef** input reference cluster of the Start VI also is empty.

However, suppose an error occurs when the Start VI runs. The Start VI returns an error in the **DevStatus** cluster of the **EncoderDevRef** output reference cluster and passes this information to the **EncoderDevRef** input reference cluster of the Stop VI. Because the Stop VI receives an error, it does not execute and passes the error information to the Close VI, again through the **EncoderDevRef** reference cluster. If you wire an indicator to the **error out** output of the Close VI, **error out** returns the cumulative error information for all VIs preceding and including the Close VI. From this error information, you can determine that an error originated with the Start VI, and you can troubleshoot that error accordingly.

Many of the WPI Robotics Library VIs also contain **error in** and **error out** clusters. You can use these clusters to merge error information from different parts of an application. Each WPI Robotics Library VI merges the error information it receives from the input reference cluster and the **error in** cluster and returns this merged information in both the output reference cluster and the **error out** cluster.