# Getting Started with CompactRIO™ and LabVIEW™

**Note** If you are a new user of LabVIEW or are unfamiliar with LabVIEW, refer to the *Getting Started with LabVIEW* manual for information about LabVIEW and LabVIEW terminology.

This tutorial demonstrates how to develop a CompactRIO application in LabVIEW using the RIO Scan Interface. The application uses a CompactRIO Reconfigurable Embedded system with LabVIEW to make a simple process-control VI. While developing this application, you will learn the concepts and techniques necessary to develop CompactRIO applications with the Scan Interface. The Scan Interface enables you to use C Series modules directly from LabVIEW Real-Time.

The tutorial also includes optional sections wherein you can program the CompactRIO system using the LabVIEW FPGA Interface.

## Contents



NATIONAL INSTRUMENTS™

# 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 tip, which alerts you to advisory information. |
| | This icon denotes a note, which alerts you to important information. |
| **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, controls and indicators on the front panel, dialog boxes, sections of dialog boxes, menu names, and palette 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` | Monospace text 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. |

# Required Components

This section lists the software and hardware used in the tutorial. This section also lists documents you may find helpful while completing the tutorial.

## Required Software

The following software is required for this tutorial.

- ❑ NI LabVIEW 2009 or later
- ❑ NI LabVIEW Real-Time Module 2009 or later
- ❑ NI LabVIEW FPGA Module 2009 or later (optional)
- ❑ NI-RIO 3.2 or later

## Required Hardware

The following hardware is required for this tutorial.

**Tip** Even if you do not have the hardware used in this tutorial, you can follow the steps and do offline configuration to learn concepts about using CompactRIO with LabVIEW.

- ❑ Power supply for the controller
- ❑ Ethernet connection and cable
- ❑ One analog input (AI) module: NI 9201, NI 9205, NI 9206, NI 9215, or NI 9221
- ❑ One analog output (AO) module: NI 9263, NI 9264, or NI 9269
- ❑ One digital input (DI) module: NI 9401, NI 9411, NI 9421, or NI 9423
- ❑ One digital output (DO) module: NI 9401, NI 9472, or NI 9474

### For Scan Interface Mode

- ❑ CompactRIO controller and chassis that support the RIO Scan Interface
  - cRIO-9073 or cRIO-9074 Integrated Real-Time Controller and Chassis

  or
  - cRIO-9012, cRIO-9014, cRIO-9022, or cRIO-9024 Intelligent Real-Time Embedded Controller; and cRIO-9103, cRIO-9104, or cRIO-911*x* Reconfigurable Embedded Chassis

### For FPGA Interface Mode

- ❑ CompactRIO controller and chassis

## Related Documentation

The following documents contain information that you may find helpful as you read this tutorial:

- Operating instructions for the controller and modules (shipped with the hardware and available at `ni.com/manuals`).
- *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**.
- *Getting Started with LabVIEW*—Use this document as a tutorial to familiarize yourself with the LabVIEW graphical programming environment and the basic LabVIEW features you use to build data acquisition and instrument control applications. Access the *Getting Started with LabVIEW* PDF by selecting **Start»All Programs» National Instruments»LabVIEW»LabVIEW Manuals» LV_Getting_Started.pdf**.
- *Getting Started with the LabVIEW Real-Time Module*—Use this document to learn how to develop a real-time project and VIs, from setting up RT targets to building, debugging, and deploying real-time applications. Access the *Getting Started with the LabVIEW Real-Time Module* PDF by selecting **Start»All Programs»National Instruments»LabVIEW» LabVIEW Manuals»RT_Getting_Started.pdf**.
- *CompactRIO Reference and Procedures (Scan Interface)*—Use this help file to learn about using the CompactRIO system in Scan Interface programming mode. To access this help file from LabVIEW, select **Help»Search the LabVIEW Help**, then expand **Real-Time Module** on the **Contents** tab and select **CompactRIO Reference and Procedures (Scan Interface)**.
- *CompactRIO Reference and Procedures (FPGA Interface)*—Use this help file to learn about using the CompactRIO system in FPGA Interface programming mode. To access this help file from LabVIEW, select **Help»Search the LabVIEW Help**, then expand **FPGA Module** on the **Contents** tab and select **CompactRIO Reference and Procedures (FPGA Interface)**.
- *FPGA Module*—Use this help file to learn about using the LabVIEW FPGA Module. To access this help file from LabVIEW, select **Help»Search the LabVIEW Help**, then expand **FPGA Module** on the **Contents** tab.

# Overview of the Application in this Tutorial

In this tutorial, you will create VIs that can be used in control and industrial applications where you need to measure an input and produce an output. The input and output can be analog or digital. You will configure the DO module for pulse-width modulation and the DI module for period measurement. You will configure the AI module to receive a voltage input from the AO module.

This tutorial shows how to create the VIs using both programming modes, Scan Interface mode and FPGA Interface mode.

# Setting Up the Hardware

Complete the following steps to set up the hardware for the application in this tutorial.

1. Install the controller on the chassis if you are not using an integrated controller and chassis. Refer to the controller operating instructions for information about installing the controller.
2. Install the DO module in slot 1 of the chassis, the DI module in slot 2, the AO module in slot 3, and the AI module in slot 4.
3. Wire the modules as follows.
   a. Connect DO0 on the DO module to DI0 on the DI module.
   b. Connect an external power supply to the DO module if it requires one. Refer to the module operating instructions for information about power requirements.
   c. Connect AO0 on the AO module to AI0 on the AI module.

**Note** Refer to the module operating instructions for information about wiring and for specifications. If the voltage ranges or other attributes of the inputs and outputs make the modules unsuitable for wiring together, skip the module wiring and go to step 4. You can still complete and learn from the tutorial, but you will not be able to test the VI as described in the *Deploying, Testing, and Using the VI in Scan Interface Mode* section.

4. Connect the controller to a power supply and an Ethernet network on the same subnet as the development computer. Refer to the controller operating instructions for information about wiring the controller to the power supply and Ethernet network.

Complete the following steps to configure the controller and install software on it.

1. Launch Measurement & Automation Explorer (MAX) on the development computer.
2. Select the controller under **Remote Systems** in the **Configuration** pane. If you do not see the controller, you may need to disable the firewall on the development computer.

**Note** The default network behavior of an unconfigured controller depends on the revision of the controller. Refer to the controller operating instructions for more information about network configuration.

3. Verify that the **Serial Number** in the **Identification** section matches the serial number on the device.
4. If you do not want to format the disk on the controller, eliminating all installed software and files, power on the controller and skip to step 9.
5. Set the **Safe Mode** switch on the controller to the **On** position.
6. Power on the controller. If it is already powered on, press the **Reset** button on the controller to reboot it.
7. Right-click the controller under **Remote Systems** in the **Configuration** pane and select **Format Disk**. Click **Yes** on the dialog box that appears.
8. When MAX finishes formatting the disk, set the **Safe Mode** switch to the **Off** position and press the **Reset** button on the controller to reboot it.
9. Select the **Obtain an IP address automatically** radio button to assign an IP address or select the **Use the following IP address** radio button to specify a static IP address in the **IP Settings** section.
10. Type a descriptive name for the system in the **Name** field.
11. Click **Apply** above the **Network Settings** tab and let MAX reboot the system.
12. When the new system name appears under **Remote Systems**, expand the controller item in the tree, right-click **Software**, and select **Add/Remove Software**.
13. If the controller you are using supports Scan Interface programming mode, select a recommended software set that includes NI Scan Engine support and install it on the controller. If the controller does not support Scan Interface mode, select a standard recommended software set and install it on the controller. Click **Help** if you need information about installing recommended software sets.
14. After MAX finishes installing software on the controller, close MAX.

Scan Interface mode enables you to use C Series modules directly from LabVIEW Real-Time. Modules that you use in Scan Interface mode appear directly under the chassis item in the **Project Explorer** window and I/O channels appear as I/O variables under the modules. To use I/O variables, you drag and drop them to LabVIEW Real-Time VIs. In Scan Interface mode, you do not need to do any LabVIEW FPGA development or program communication between FPGA and Host VIs. You also do not need to wait for VIs to be compiled to the FPGA before deploying and running them. In Scan Interface mode, LabVIEW programs the FPGA on the CompactRIO target to work with the variables.

The following figure shows the **Project Explorer** window with a digital input and other modules added in Scan Interface mode.

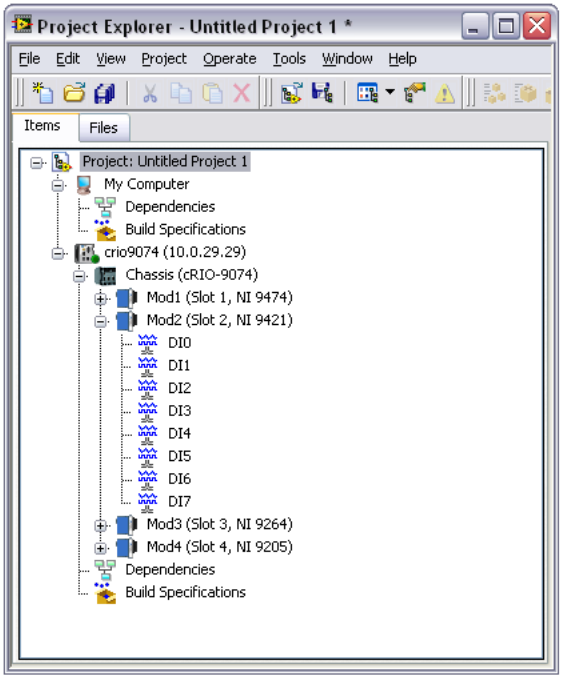


**Figure 1.** Project Explorer Window with Modules in Scan Interface Mode

LabVIEW FPGA Interface mode enables you to use C Series modules from LabVIEW FPGA VIs. Modules that you use in LabVIEW FPGA Interface mode appear directly under the FPGA Target item in the **Project Explorer** window and I/O channels appear as FPGA I/O items under the FPGA Target. To access the I/O channels, you configure FPGA I/O Nodes in FPGA VIs. In LabVIEW FPGA Interface mode, you can use LabVIEW FPGA programming to add more flexibility, customization, and

deterministic timing to your applications. To use the CompactRIO system in LabVIEW FPGA Interface mode, you must either have the LabVIEW FPGA Module installed on the host computer, or have access to a compiled bitfile that you can download to the FPGA. In either case, you use the Open FPGA VI Reference function in a host VI to access the FPGA VI or bitfile.

The following figure shows the **Project Explorer** window with the same modules added in FPGA Interface mode.

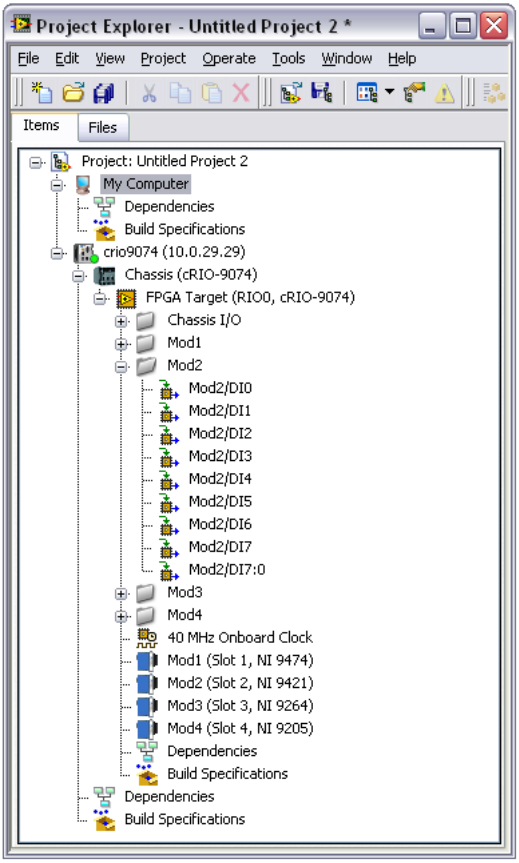


**Figure 2.** Project Explorer Window with Modules in FPGA Interface Mode

If you want to use Scan Interface mode for your application, you can proceed to the *Creating a Project in Scan Interface Mode* section. If you want to use FPGA Interface mode, you can skip to the *Creating a Project in FPGA Interface Mode* section. Alternatively, you can go through the entire tutorial to learn about both modes.

# Creating a Project in Scan Interface Mode

Use a LabVIEW project to manage VIs, targets, and I/O modules on the development computer. Complete the following steps to create a project.

1. Launch LabVIEW.
2. Click the **Empty Project** link in the **Getting Started** window to display the **Project Explorer** window. You can also select **File»New Project** to display the **Project Explorer** window.
3. Select **Help** and make sure that **Show Context Help** is checked. You can refer to the context help throughout the tutorial for information about items on the block diagram.
4. Right-click the top-level project item in the **Project Explorer** window and select **New»Targets and Devices** from the shortcut menu to display the **Add Targets and Devices** dialog box.
5. Make sure that the **Existing target or device** radio button is selected.

**Tip** If you do not have hardware installed, you can select the **New target or device** radio button to display a list of targets and devices that you can create without a physical target or device present. Throughout this tutorial, you can perform similar offline configuration steps to follow along and learn about using CompactRIO and LabVIEW. For more information about offline configuration in FPGA Interface mode, refer to the *Configuring a Project for a CompactRIO Reconfigurable or Integrated System (Scan Interface)* topic of the *LabVIEW Help*.

6. Expand **Real-Time CompactRIO**.
7. Select the CompactRIO controller to add to the project and click **OK**.
8. If you have LabVIEW FPGA installed, the **Select Programming Mode** dialog box appears. Select **Scan Interface** to put the system into Scan Interface mode.
9. Click **Continue**. LabVIEW adds the controller, the chassis, and all the modules to the project.
10. Click **Discover** in the **Discover C Series Modules?** dialog box if it appears. The Project Explorer window should look similar to Figure 1.
11. Select **File»Save Project** and save the project as `Tutorial.lvproj`.

# Creating a VI in Scan Interface Mode

In this section you will create a VI that uses the I/O variables for the modules you added to the project. An I/O variable is a shared variable tied to a physical I/O channel. LabVIEW creates I/O variables for I/O channels of modules that you add to a system in Scan Interface mode. Refer to the *Accessing Scanned I/O Data (ETS, VxWorks, Windows)* topic of the *LabVIEW Help* for more information about I/O variables, Scan Interface mode, and the NI Scan Engine.

## Configuring the Timed Loop

A Timed Loop synchronized to the Scan Engine enables you to read and write coherent sets of precisely timed data using multiple I/O variables. Complete the following steps to configure the Timed Loop.

1. Right-click the controller item in the **Project Explorer** window and select **New»VI** from the shortcut menu to open a blank VI.
2. Place a Timed Loop on the block diagram of the VI.
3. Double-click the **Input Node** of the Timed Loop to open the **Configure Timed Loop** dialog box.
4. Under **Loop Timing Source**, for **Source Type**, select **Synchronize to Scan Engine**. You can click the **Help** button for information about synchronizing to the Scan Engine.
5. Under **Loop Timing Attributes**, set the **Period** to **5** scans. The **Configure Timed Loop** dialog box should look similar to the following figure.

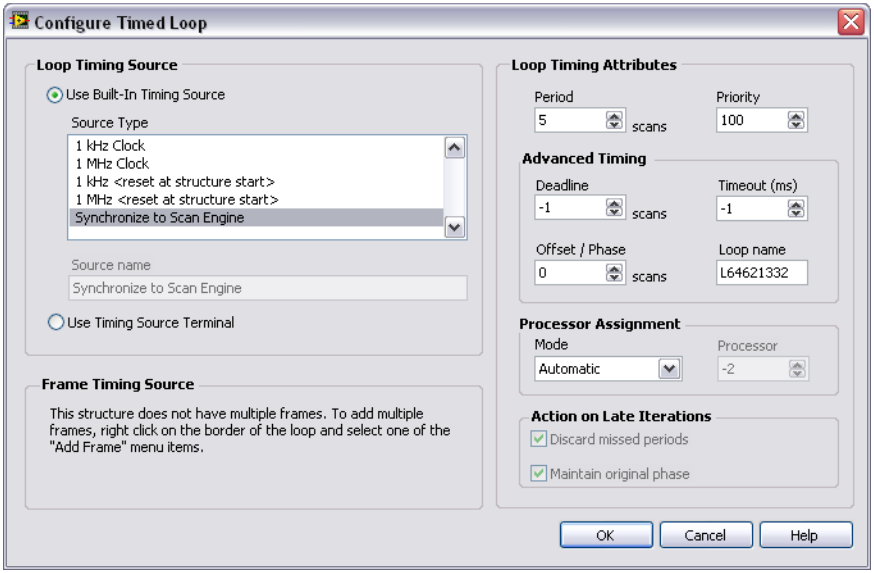


**Figure 3.** Configuring the Timed Loop

6. Click **OK**.
7. On the block diagram, right-click the conditional terminal at the bottom right of the Timed Loop and select **Create Control** from the shortcut menu.

## Configuring the DO Module for Pulse-Width Modulation

Pulse-width modulation varies the duty cycle of a digital voltage output to produce an analog signal range for control applications. You can use pulse-width modulation to provide analog control for digital devices such as DC motors, heaters, and lights. Complete the following steps to configure the DO module for pulse-width modulation.

1. Right-click the DO module in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box.
2. Select **Specialty Digital Configuration** in the **Category** list.
3. Select **Pulse-Width Modulation** for the Specialty Mode.
4. Under **Channels**, make sure that **PWM0** is highlighted.

5. Select **50 Hz (20,000 µs)** for the **Frequency (Period)**. The **C Series Module Properties** dialog box should look like the following figure.

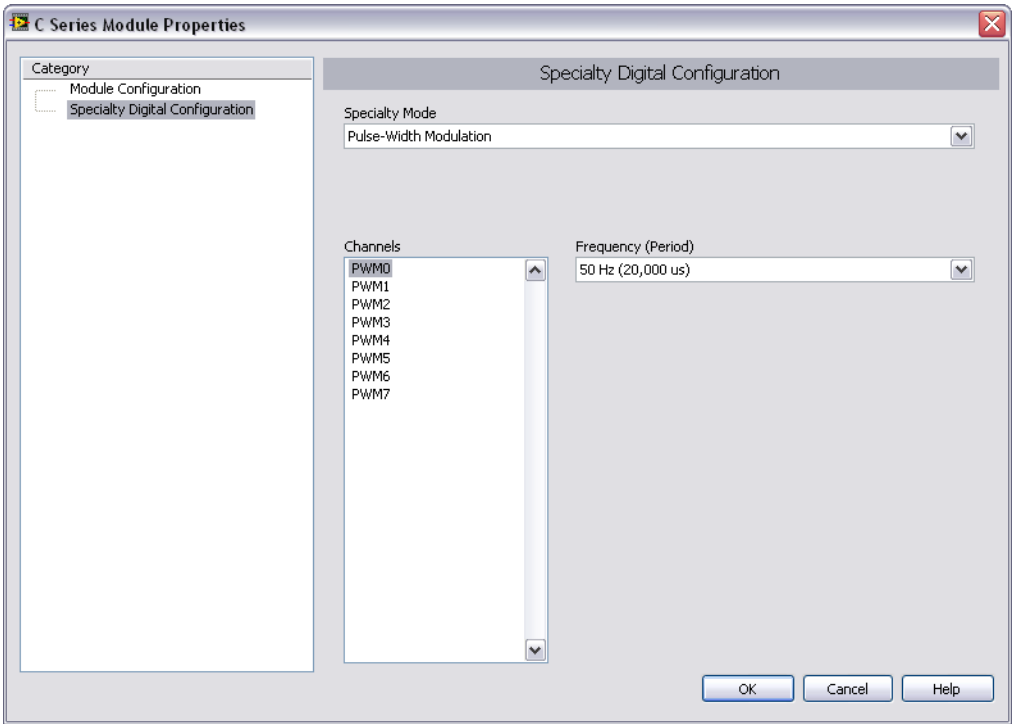


**Figure 4.** Configuring the DO Module for Pulse-Width Modulation

6. Click **OK**.
7. Expand the DO module item in the **Project Explorer** window to see the I/O variable items for the module channels. Note that LabVIEW has changed all the DO I/O variables to PWM I/O variables.

## Adding Pulse-Width Modulation to the VI

Complete the following steps to add pulse-width modulation to the VI.

1. Drag and drop the **PWM0** I/O variable from the **Project Explorer** window to inside the Timed Loop on the block diagram of the VI.
2. On the block diagram, right-click the **PWM0** input of the **PWM0** I/O variable and select **Create»Control** from the shortcut menu to create a control on the front panel.
3. On the front panel, right-click the **PWM0** control and select **Properties** from the shortcut menu to display the **Numeric Properties** dialog box.

4. Under **Label**, change the name of the control from **PWM0** to **PWM Duty Cycle**.
5. Select the **Data Entry** tab and uncheck the **Use Default Limits** checkbox.
6. In the **Minimum** field, enter `0` for the value and select **Coerce** in the **Response to value outside limits** drop-down menu.
7. In the **Maximum** field, enter `100` for the value and select **Coerce** in the **Response to value outside limits** drop-down menu. The **Numeric Properties** dialog box should look like the following figure.

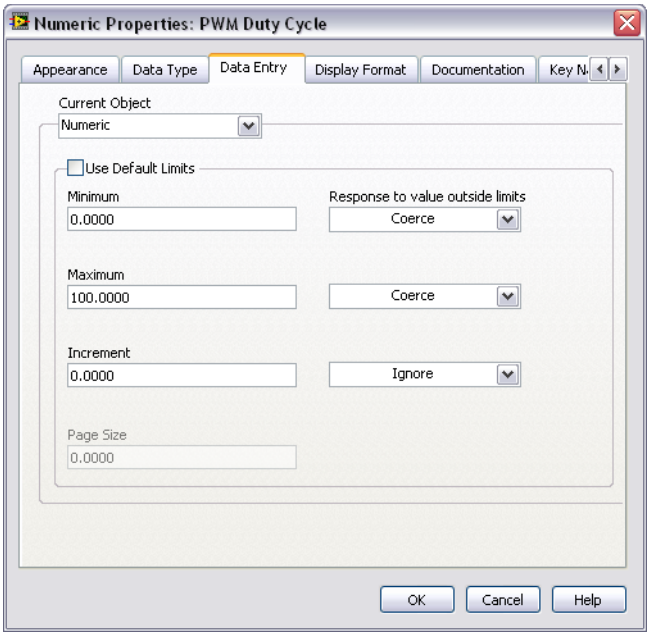


**Figure 5.** Configuring the PWM Duty Cycle

**Note** Coercing values to be between 0 and 100 ensures that duty-cycle values are valid percentages.

8. Click **OK**.
9. Save the VI as `Tutorial.vi`.

## Configuring the DI Module for Period Measurement

Many industrial applications use frequency-output sensors such as flow sensors, torque sensors, timers, and motor-driven actuators. In this application, the DO module is producing a relatively low-frequency signal. With the DI module in Scan Interface mode, you can obtain more accurate

measurements of low frequencies using period measurement and calculating the frequency as the inverse of the period.[1] Complete the following steps to configure the DI module for period measurement.

1. Right-click the DI module in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box.
2. Select **Specialty Digital Configuration** in the **Category** list.
3. Select **Counter** for the Specialty Mode.
4. Under **Channels**, make sure that **CTR0** is highlighted.
5. Select **Period Measurement** for the **Measurement Mode**. The **C Series Module Properties** dialog box should look like the following figure.

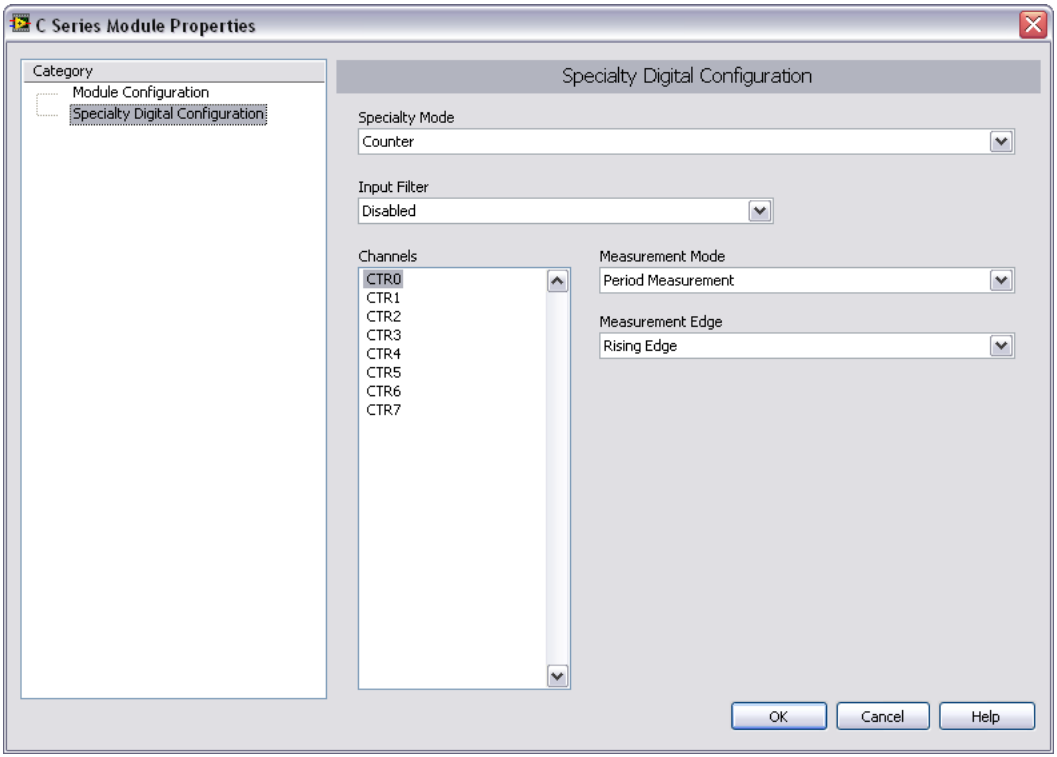


**Figure 6.** Configuring the DI Module for Period Measurement

[1] If you modify the application so that the DO module produces a frequency higher than 1 kHz, you can configure the DI module for direct frequency measurement.

6. Click **OK**.
7. Expand the DI module item in the **Project Explorer** window to see the I/O variable items for the module channels. Note that LabVIEW has changed all the DI I/O variables to CTR I/O variables.

## Adding a Period-Measurement Counter to the VI

Complete the following steps to add a period-measurement counter to the VI and use it to monitor the frequency of the digital input signal.

1. Drag and drop the **CTR0** I/O variable from the **Project Explorer** window to inside the Timed Loop on the block diagram of the VI. Place it to the right of the **PWM0** I/O variable.
2. On the block diagram, right-click the **CTR0** output of the **CTR0** I/O variable and select **Numeric Palette»Reciprocal** to create a numeric function that converts the period to the frequency. Drop the function on the block diagram and wire the **CTR0** output to it.
3. On the block diagram, right-click the output of the **Reciprocal** function and select **Create»Indicator** from the shortcut menu to create an indicator on the front panel.
4. Rename the **1/*x*** indicator `Frequency`.
5. Wire the **error out** terminal of the **PWM0** I/O variable to the **error in** terminal of the **CTR0** I/O variable.
6. Place a Merge Errors VI on the block diagram to the right of the **CTR0** I/O variable.
7. Wire the **error out** terminal of the **CTR0** I/O variable to one of the **error in** terminals of the Merge Errors VI.
8. Save the VI.

## Adding AO and AI to the VI

For this tutorial, you wired an AO channel to an AI channel so that you can learn about using CompactRIO with LabVIEW. In real-world applications, the AO channel might send a voltage to a device in a physical process and the AI channel might receive input from that device or another device. If you have devices you want to connect to the analog modules and control or monitor, you can proceed using this document as a starting point. Complete the following steps to add AO and AI to the VI.

1. Expand the AO module item in the **Project Explorer** window to see the I/O variable items for the module channels.
2. Drag and drop the **AO0** I/O variable from the **Project Explorer** window to inside the Timed Loop on the block diagram of the VI, above or below the **PWM** I/O variable.

3. Expand the AI module item in the **Project Explorer** window to see the I/O variable items for the module channels.
4. Drag and drop the **AI0** I/O variable from the **Project Explorer** window to inside the Timed Loop on the block diagram of the VI. Place it to the right of the **AO0** I/O variable.
5. Right-click the **AO0** input of the **AO0** I/O variable and select **Create» Control** from the shortcut menu to create a control on the front panel.
6. Right-click the **AI0** output of the **AI0** I/O variable and select **Create» Indicator** from the shortcut menu to create an indicator on the front panel.
7. Wire the **error out** terminal of the **AO0** I/O variable to the **error in** terminal of the **AI0** I/O variable.
8. Wire the **error out** terminal of the **AI0** I/O variable to one of the **error in** terminals of the Merge Errors VI.
9. Right-click the **error out** terminal of the Merge Errors VI and select **Create»Indicator** from the shortcut menu to create an indicator on the front panel.

10. Click the **Clean Up Diagram** button on the toolbar. The block diagram should look similar to the following figure.

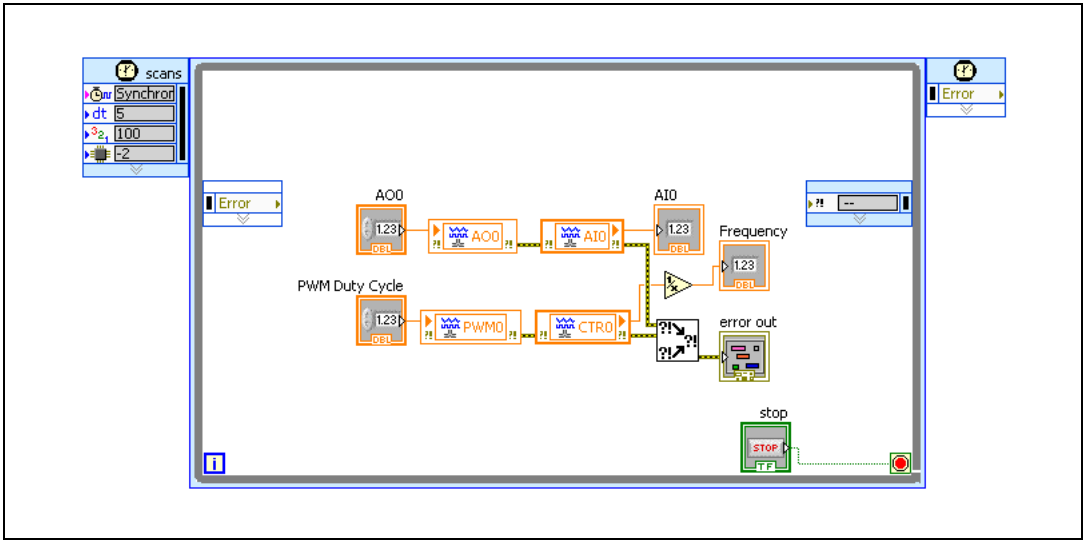


**Figure 7.** Block Diagram of Tutorial.vi

11. Save the VI.
12. Save the project.

# Deploying, Testing, and Using the VI in Scan Interface Mode

Complete the following steps to deploy, test, and use the VI.

1. Run the VI. LabVIEW deploys the VI, and all modules and I/O variables the VI uses, to the controller.
2. On the front panel, change the value of the **AO0** control and verify that the value of the **AI0** indicator changes.
3. Change the value of the **PWM Duty Cycle** control. Verify that the value of the **Frequency** indicator remains constant at 50 Hz.
4. Stop the VI.

# Optional: Modifying the Scan Interface Application Using the FPGA Interface

Some applications require that measurement parameters adapt to changing characteristics of the measured signals. For example, consider a case in which signals change very slowly except for sudden bursts of high-speed activity. In order to reduce the amount of data logged, the application must quickly recognize the need for a higher sampling rate, and reduce the rate when the burst of activity is over. By measuring and analyzing certain aspects of the signals, the application can adapt to changing circumstances and respond appropriately. Although this is only one example, there are many applications that require intelligence—the ability to make decisions based on various conditions—and adaptability. LabVIEW FPGA programming can provide intelligence and adaptability by adding analysis algorithms to applications.

The analog input in the project now consists of only single-point data. In your application, you may want to acquire and analyze waveform data instead. For example, if an alarm condition occurs, you may want to monitor a physical process in more detail or observe changes in data over a period of time. In order to do waveform acquisition of analog input data, you need to use LabVIEW FPGA.

## Putting the Chassis and the AI Module into FPGA Interface Mode

In order to program the AI module using LabVIEW FPGA, you must put the chassis and the AI module into LabVIEW FPGA Interface mode. You can still use the other modules in Scan Interface mode. Complete the following steps to put the chassis and the AI module into FPGA Interface mode.

1. Right-click the chassis item in the **Project Explorer** window and select **New»FPGA Target** from the shortcut menu to display the **Deploy CompactRIO Chassis Settings?** dialog box.
2. Click **Deploy now**.

**Tip** Alternatively, you can right-click the chassis item in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **CompactRIO Chassis Properties** dialog box. After you select the programming mode using that dialog box, you must deploy settings to the chassis.

3. In the **Project Explorer** window, drag and drop the AI module item onto the **FPGA Target** item.

## Creating and Configuring the DMA FIFO

A FIFO is a first-in-first-out memory buffer. A DMA (direct memory access) FIFO enables an FPGA VI to transfer data directly to a host VI. Complete the following steps to create and configure a DMA FIFO.

1. Right-click the **FPGA Target** item in the **Project Explorer** window and select **New»FIFO** to display the **FIFO Properties** dialog box.
2. For the **Type**, select **Target to Host - DMA**.

**Note** You must configure the FIFO for the type of data that the I/O module returns. The FIFO in this tutorial is configured for an NI 9205. The NI 9205 returns signed, calibrated fixed-point data by default. The word length is 26 bits and the integer word length is 5 bits. Other AI modules may return different types of data. Refer to the *CompactRIO Reference and Procedures (FPGA Interface)* help file for information about the data that other AI modules return.

3. For the **Data Type**, select **FXP**.
4. For the **Word length**, select **26 bits**.
5. For the **Integer word length**, select **5 bits**. The **FIFO Properties** dialog box should look like the following figure.

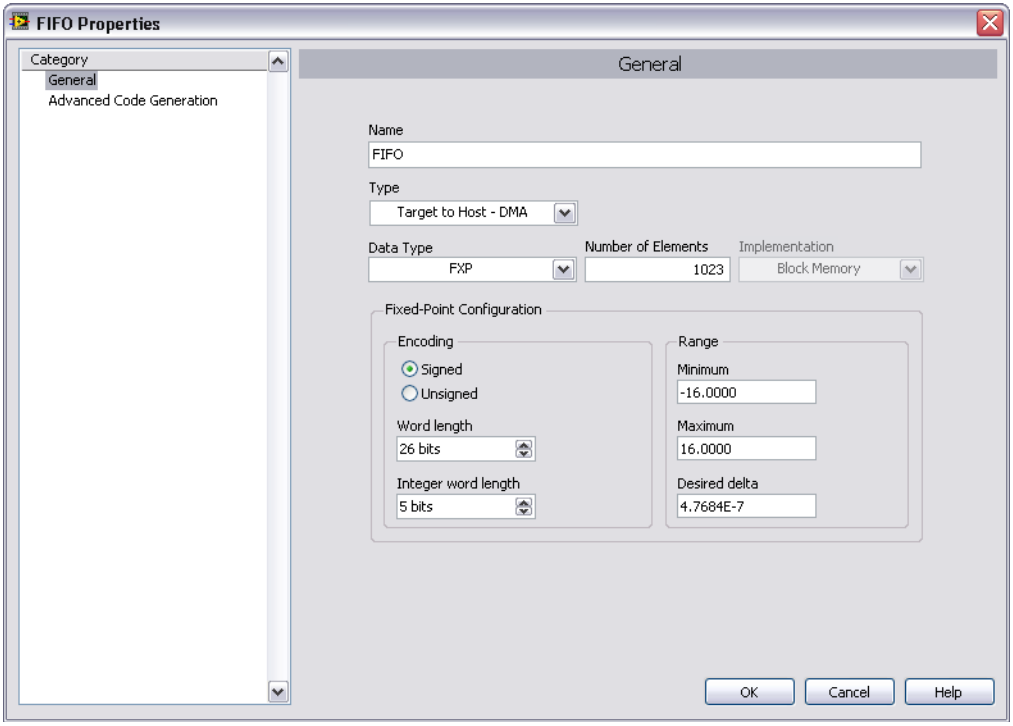


**Figure 8.** Configuring the DMA FIFO

6. Click **OK**.

**Tip** For more information about DMA FIFOs, go to the *LabVIEW Help* and search for "DMA FIFO".

## Adding Example VIs to the Project

The NI-RIO software is installed with example VIs that can save you a lot of work. Complete the following steps to add the example VIs to the project.

1. In the **Project Explorer** window, select **Help»Find Examples** to open the NI Example Finder.
2. Double-click **Toolkits and Modules»FPGA»CompactRIO» NI Scan Engine»Getting Started»Using Scan Interface with FPGA Interface.lvproj**.
3. In the new **Project Explorer** window that opens, expand the **RT CompactRIO Target** item, then the chassis item, then the **FPGA Target** item.

4. Using <Ctrl>-click, drag and drop **`Using Scan Interface with FPGA Interface (FPGA).vi`** onto the **FPGA Target** item in the **Project Explorer - Tutorial.lvproj** window.
5. Using <Ctrl>-click, drag and drop **`Using Scan Interface with FPGA Interface (Getting Started).vi`** onto the controller item in the **Project Explorer - Tutorial.lvproj** window. The **Project Explorer** window should look similar to the following figure.

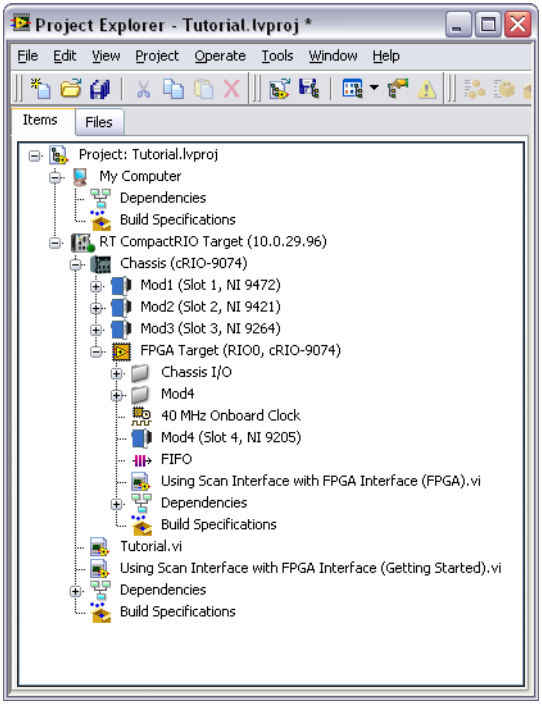


**Figure 9.** Project Explorer Window with Example VIs Added

6. Close the **Project Explorer - Using Scan Interface with FPGA Interface** window and the NI Example Finder.

## Replacing the Timed Loop in the Example Host VI and Running the VI

The host VI communicates with the FPGA VI. You can run the host VI on a Real-Time (RT) target, such as a CompactRIO controller, or on a Windows PC. The example host VI contains a Timed Loop that you must replace with the Timed Loop you created and configured earlier in this tutorial. Complete the following steps to replace the Timed Loop.

1. Open **`Using Scan Interface with FPGA Interface (FPGA).vi`** from the **Project Explorer** window. Verify that there are no broken wires on the block diagram.
2. Open **`Using Scan Interface with FPGA Interface (Getting Started).vi`** from the **Project Explorer** window.

**Note** The Timed Loop at the top of the block diagram is similar to the Timed Loop in `Tutorial.vi`. Note that wiring the **error out** terminal of the **Start** Read/Write Control to the two loops ensures that the FPGA starts running before the loops start running.

3. Delete the Timed Loop and replace it with the Timed Loop from `Tutorial.vi`.
4. Rewire the **error out** terminal of the **Start** Read/Write Control to the **error in** terminal of the **Input Node** of the new Timed Loop.
5. In the Timed Loop, delete the **AI0** I/O variable and indicator, and delete the resulting broken wires. The AI data acquisition is done in the FPGA VI, not in the host VI.
6. Right-click the **Open FPGA VI Reference** function and select **Configure Open FPGA VI Reference**.
7. Verify that the Open FPGA VI Reference is configured to open `FPGA Target\Using Scan Interface with FPGA Interface (FPGA).vi` and click **OK**.
8. Save the VI.
9. Save the project.
10. In the **Project Explorer** window, right-click **`Using Scan Interface with FPGA Interface (FPGA).vi`** and select **Compile** from the shortcut menu to compile the FPGA VI. Compiling can take from a few minutes to a few hours.

**Note** When you compile the VI, the LabVIEW FPGA Compile Server adds to the FPGA VI all the logic necessary to communicate with modules you are using in Scan Interface mode. When you run the host VI, the Open FPGA VI Reference and the **Start** Read/Write control wired to the Timed Loop ensure that the FPGA VI is running before the I/O variables in the Timed Loop start returning data.

11. When the FPGA VI is compiled, run `Using Scan Interface with FPGA Interface (Getting Started).vi`.
12. Stop the VI.

**Tip** The code in `Using Scan Interface with FPGA Interface (FPGA).vi` was generated using the FPGA Wizard. Refer to the *FPGA Wizard* book in the *LabVIEW Help* for information about using the FPGA Wizard to generate code for projects.

# Creating a Project in FPGA Interface Mode

In this section you will create a project and VIs similar to the ones you can create in the *Creating a Project in Scan Interface Mode* section, but you will create them in LabVIEW FPGA Interface mode.

Use a LabVIEW project to manage VIs, targets, and I/O modules on the development computer. Complete the following steps to create a project using the LabVIEW FPGA Project Wizard.

1. Launch LabVIEW.
2. On the **Getting Started** window, under **Targets**, select **FPGA Project**.
3. Click **Go**.
4. On the **Create New LabVIEW FPGA Project** page that appears, select **CompactRIO Reconfigurable Embedded System** and click **Next**.
5. Make sure **Discover existing system** is selected and click **Next**.

**Tip** If you do not have hardware installed, you can select **Create new system**. Throughout this tutorial, you can perform similar offline configuration steps to follow along and learn about using CompactRIO and LabVIEW. For more information about offline configuration in FPGA Interface mode, refer to the *Configuring a Project for a CompactRIO Reconfigurable or Integrated System (FPGA Interface)* topic of the *LabVIEW Help*.

6. Select the controller you are using and click **Next**.
7. Uncheck **Launch FPGA Wizard when finished** and click **Finish**. LabVIEW adds the controller, the chassis, the FPGA target, and all the modules to the project. The **Project Explorer** window that appears should look similar to the following figure if you expand the project items.

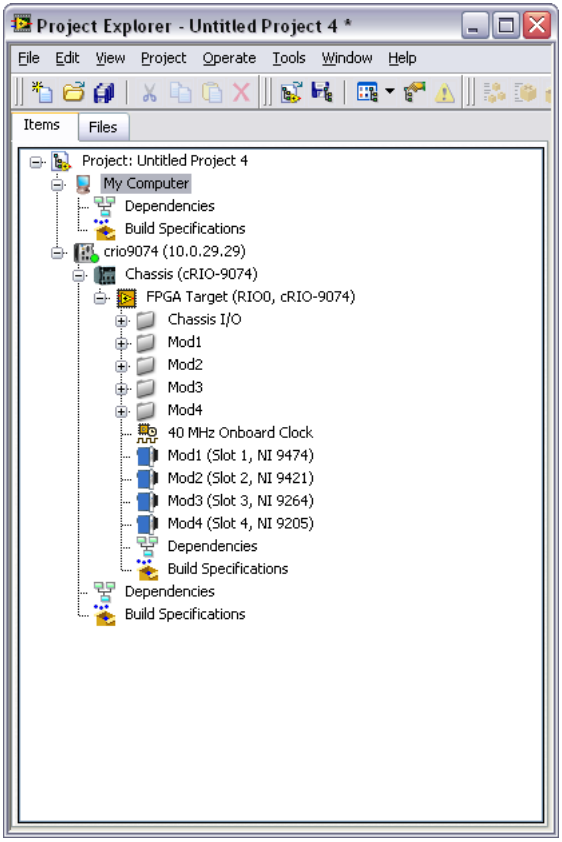


**Figure 10.** Project Explorer Window for New FPGA Project

8. Select **Help** and make sure that **Show Context Help** is checked. You can refer to the context help throughout the tutorial for information about items on the block diagram.
9. Select **File»Save Project** and save the project as `Getting Started.lvproj`.

## Creating the AI/AO Loop in the FPGA VI

The FPGA VI is the VI you download to the FPGA target, which, in this application, is the CompactRIO chassis. You use the FPGA VI to read from and write to the I/O channels of C Series modules. Complete the following steps to create the FPGA VI and add AI and AO to it.

1. Right-click the **FPGA Target** item in the **Project Explorer** window and select **New»VI** from the shortcut menu to open a blank VI.
2. Place a While Loop on the block diagram of the VI.
3. In the **Project Explorer** window, expand the **Mod4** folder item.

4. Drag and drop the **Mod4/AI0** item from the **Project Explorer** window to inside the While Loop on the block diagram to create an FPGA I/O Node for that item.
5. On the block diagram, drag down the bottom edge of the **Mod4/AI0** FPGA I/O Node to add one element.
6. Click the new element and select **Mod3»Mod3/AO0** from the shortcut menu to associate the element with the **Mod3/AO0** item.

**Tip** Analog I/O modules return calibrated fixed-point data by default. If you need to conserve FPGA resources, you can configure the modules to return uncalibrated data. Right-click the module you want to reconfigure in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box. Under **Calibration Mode**, select **Raw**, then click **OK**.

7. Right-click the **Mod3/AO0** element and select **Create»Control** from the shortcut menu to create a control on the front panel.
8. Right-click the **Mod4/AI0** element and select **Create»Indicator** from the shortcut menu to create an indicator on the front panel.
9. Right-click the FPGA I/O Node and select **Show Error Terminals** from the shortcut menu.

**Tip** Showing error terminals uses FPGA resources. After testing and debugging your application, you can hide error terminals to conserve resources.

10. Right-click the **error out** output of the FPGA I/O Node and select **Create»Indicator** from the shortcut menu to create an indicator on the front panel.

**Tip** In this tutorial, you use one FPGA I/O Node for I/O on two C Series modules. If you find that you need to identify the sources of errors in your application, use one FPGA I/O Node and error indicator for each module.

11. Right-click the conditional terminal at the bottom right of the While Loop and select **Create Control** from the shortcut menu to create a **stop** control.
12. Rename the new control `Stop AI/AO`.

13. Click the **Clean Up Diagram** button on the toolbar. The block diagram should look similar to the following figure.

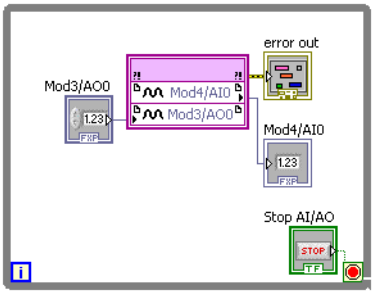


**Figure 11.** Block Diagram of the FPGA VI with AI and AO

14. Save the new VI as `Getting Started (FPGA).vi`.

## Creating the PWM Loop

Pulse-width modulation varies the duty cycle of a digital voltage output to produce an analog signal range for control applications. You can use pulse-width modulation to provide analog control for digital devices such as DC motors, heaters, and lights. Complete the following steps to add pulse-width modulation to the FPGA VI.

1. Place an additional While Loop on the block diagram above the first While Loop.
2. In the **Project Explorer** window, expand the **Mod1** folder item.
3. Drag and drop the **Mod1/DO0** item from the **Project Explorer** window to inside the While Loop on the block diagram.
4. Place a Flat Sequence structure around the **Mod1/DO0** FPGA I/O Node on the block diagram.
5. Right-click the left border of the Flat Sequence structure and select **Add Frame Before** from the shortcut menu.
6. Place a Loop Timer VI in the new frame.
7. On the **Configure Loop Timer** dialog box that appears, make sure **Ticks** and **32 Bit** are selected, then click **OK**.
8. Place two numeric controls on the front panel and label the controls `Low Pulse` and `High Pulse`.
9. Place a Select function to the left of the Flat Sequence structure on the block diagram.
10. Move the **Low Pulse** and **High Pulse** controls to inside the While Loop, to the left of the Select function.

11. Right-click the **Low Pulse** control and select **Representation» U32** from the shortcut menu to change the data type to 32-bit unsigned integer (U32).
12. Right-click the **High Pulse** control and select **Representation» U32** from the shortcut menu to change the data type to 32-bit unsigned integer (U32).
13. Wire the output terminal of the **Low Pulse** control to the **t** input of the Select function.
14. Wire the output terminal of the **High Pulse** control to the **f** input of the Select function.
15. Wire the output terminal of the Select function to the input terminal of the Loop Timer VI.
16. Right-click the left border of the While Loop and select **Add Shift Register** from the shortcut menu. Shift registers transfer values from one loop iteration to the next.
17. Place a **False** Boolean constant outside the While Loop and wire it to the shift register terminal on the left side of the While Loop.
18. Wire the shift register terminal on the left to the **s** input of the Select function.
19. Place a Not function inside the While Loop, below the frame of the Flat Sequence structure that contains the **Mod1/DO0** FPGA I/O Node.
20. Wire the shift register terminal on the left side of the While Loop to the **Mod1/DO0** input of the **Mod1/DO0** FPGA I/O Node.
21. Wire the shift register terminal on the left side of the While Loop to the input terminal of the Not function.
22. Wire the output terminal of the Not function to the shift register terminal on the right side of the While Loop.
23. Right-click the conditional terminal at the bottom right of the While Loop and select **Create Control** from the shortcut menu to create a **stop** control.
24. Name the new **stop** control `Stop PWM`.

25. Click the **Clean Up Diagram** button on the toolbar. The block diagram should look similar to the following figure.

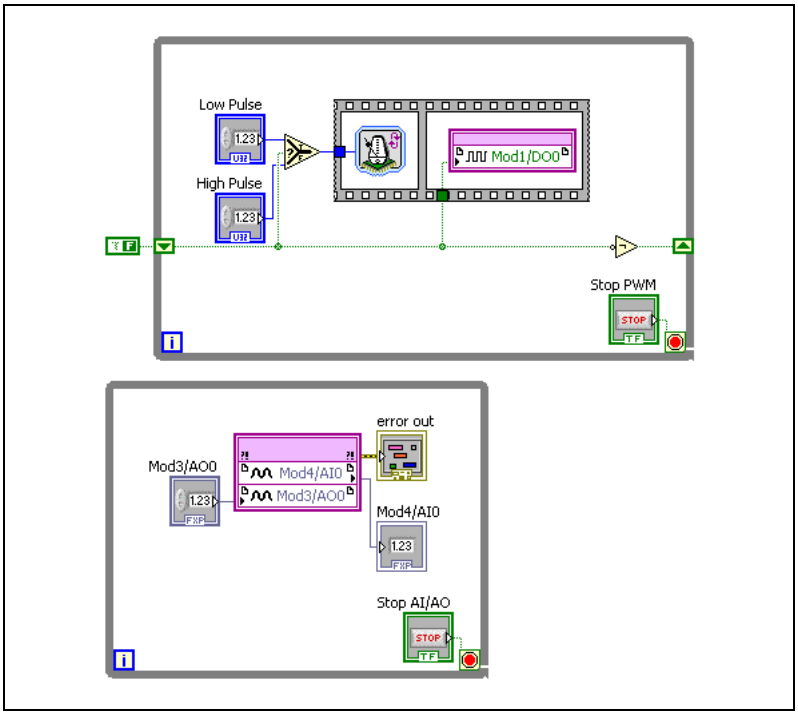


**Figure 12.** Block Diagram of the FPGA VI with Two While Loops

26. Save the VI.
27. Save the project.
28. In the **Project Explorer** window, right-click **`Getting Started (FPGA).vi`** and select **Compile** from the shortcut menu to compile the FPGA VI. Compiling can take from a few minutes to a few hours.

## Creating a Host VI in FPGA Interface Mode

The host VI communicates with the FPGA VI. You can run the host VI on a Real-Time (RT) target, such as a CompactRIO controller, or on a Windows PC. Complete the following steps to build the host VI.

1. In the **Project Explorer** window, right-click **My Computer** and select **New»VI**.
2. Place a While Loop on the block diagram of the new VI.
3. Place an Open FPGA VI Reference function on the block diagram to the left of the While Loop.
4. Double-click the Open FPGA VI Reference function.

5. On the **Configure Open FPGA VI Reference** dialog box that appears, select the **VI** radio button.
6. On the **Select VI** dialog box that appears, select **`Getting Started (FPGA).vi`** and click **OK**.
7. Click **OK** on the **Configure Open FPGA VI Reference** dialog box.
8. Place a Read/Write Control function inside the While Loop on the block diagram.
9. Wire the **FPGA VI Reference Out** output of the Open FPGA VI Reference function to the **FPGA VI Reference In** input of the Read/Write Control function.
10. Right-click the **FPGA VI Reference Out** output of the Read/Write Control function and select **FPGA Interface Palette»Close FPGA VI Reference**.
11. Place the Close FPGA VI Reference function to the right of the While Loop.
12. Wire the **FPGA VI Reference Out** output of the Read/Write Control function to the **FPGA VI Reference In** input of the Close FPGA VI Reference function.
13. Wire the **error out** output of the Open FPGA VI Reference function to the **error in** input of the Read/Write Control function.
14. Wire the **error out** output of the Read/Write Control function to the **error in** input of the Close FPGA VI Reference function.
15. Click the **Unselected** element of the Read/Write Control function and select **Mod3/AO0**.
16. Expand the Read/Write Control function downward until it shows all of the controls and indicators in `Getting Started (FPGA).vi`: **Mod3/AO0**, **Mod4/AI0**, **Stop AI/AO**, **Low Pulse**, **High Pulse**, **Stop PWM**, and **error out**.
17. Create a control for the **Mod3/AO0** element.
18. Create an indicator for the **Mod4/AI0** element.
19. Create controls for the **Low Pulse** and **High Pulse** elements.
20. Wire the input terminals of the **Stop AI/AO** element and the **Stop PWM** element to the output terminal of the **stop** control of the While Loop.

21. Click the **Clean Up Diagram** button on the toolbar. The block diagram should look similar to the following figure.

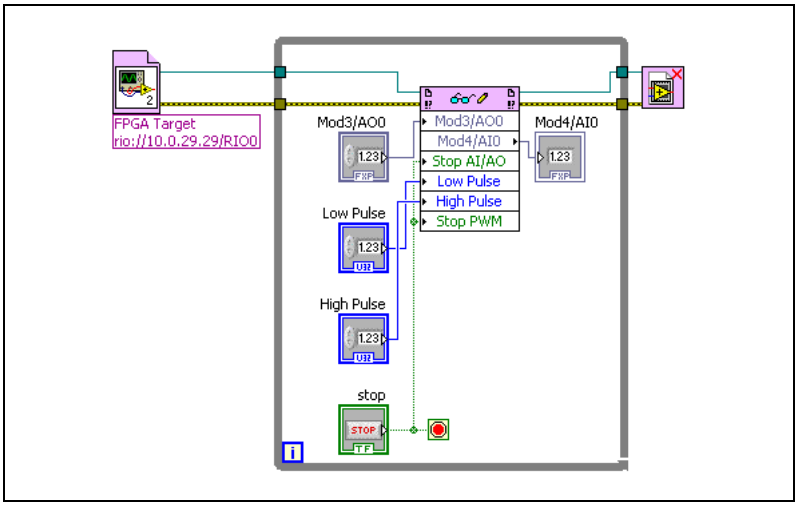


**Figure 13.** Block Diagram of the Host VI

22. Save the VI as `Getting Started (Host).vi`.
23. Save the project.

## Running and Testing the Host VI

You can use the host VI you just created to control your application. Complete the following steps to run and test the host VI.

1. Run the host VI, `Getting Started (Host).vi`.
2. On the front panel, change the value of the **Mod3/AO0** control. The value of the **Mod4/AI0** indicator should change in response.
3. Set the **High Pulse** and **Low Pulse** controls to equal values. This results in a duty cycle of 50%. If the DO module has LEDs, the LED for channel 0 should light.

**Tip** You can use the following formula to calculate the duty cycle:

$$\frac{HighPulse}{HighPulse + LowPulse} = DutyCycle$$

4. Click the **Stop** control to stop the VI.

# What You Have Learned

This tutorial taught the following key concepts about developing a CompactRIO application:

- You can use CompactRIO in either Scan Interface mode or FPGA Interface mode, or you can combine the two modes. Creating and configuring VIs is somewhat simpler in Scan Interface mode, whereas FPGA Interface mode offers more possibilities for customization.
  - If you are using both Scan Interface and FPGA Interface modes, you must use proper data flow and an Open FPGA VI Reference function to make sure that the FPGA VI is running before I/O variables start returning data.
- A typical CompactRIO application consists of a LabVIEW project and one or more VIs.
  - Use the **Project Explorer** window to organize VIs, configure settings for the VIs, configure the CompactRIO devices, and configure the channel aliases and I/O variables.
  - If you are using any modules in FPGA Interface mode, the project must contain an FPGA VI and a host VI. The FPGA VI runs on the FPGA of a CompactRIO chassis. The host VI typically runs on the CompactRIO controller, but it can also run on a Windows PC.
  - Use the FPGA VI to read and write to the CompactRIO I/O channels and to implement logical operations in the FPGA.
  - Use the host VI to communicate with the FPGA VI and to do datalogging and analysis.
- Use error terminals throughout a CompactRIO application. However, if the FPGA VI does not fit on the FPGA, you can disable error checking to decrease the size of the FPGA VI.

# Where to Go for Support

The National Instruments Web site is your complete resource for technical support. At `ni.com/support` you have access to everything from troubleshooting and application development self-help resources to email and phone assistance from NI Application Engineers.

National Instruments corporate headquarters is located at 11500 North Mopac Expressway, Austin, Texas, 78759-3504. National Instruments also has offices located around the world to help address your support needs. For telephone support in the United States, create your service request at `ni.com/support` and follow the calling instructions or dial 512 795 8248. For telephone support outside the United States, contact your local branch office:

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



372596B-01 Aug09