Embedded Data Logging with NI LabVIEW and CompactRIO Hands-On

Customer Exercise

southafrica.ni.com
Exercise 1(Optional): Create a Simple LabVIEW VI

If you are new to LabVIEW this exercise is designed to help you familiarize yourself with the development environment and understand the basic concepts of data-flow programming.

During this exercise you will create a simple piece of LabVIEW code - a VI - that simulates an analogue signal and plots it on a waveform graph. The VI will test the input values against a user-specified limit and raises an alarm if the input value exceeds that limit.

Below are pictures identifying each of the palettes found in LabVIEW to assist you as you complete these exercises.

**Note:** LabVIEW has a built-in Automatic Tool Selection feature that changes the behavior of the cursor depending on what type of object or which part of an object you are currently pointing to.

1. Navigate to Start (or 📦) > All Programs > National Instruments > LabVIEW 2011(32-bit), and select LabVIEW 2011 (32-bit).
2. Once you launch LabVIEW, a splash screen appears. Click on File->New to get to the templates.

3. In the new window, expand From Template. Notice the different categories that appear, corresponding to the types of tasks which you can choose from. You can select Blank VI to start from scratch, and there are also Template VIs to use as a starting point for building your application. Projects and Other Files are more advanced components and will not be described in detail. To get more information on any of the listings in the New Dialog Box, click the Help button in the lower right corner of the window.

4. Navigate to and Select VI» From Template» Tutorial (Getting Started) » Generate and Display and click OK.

Two windows appear. The grey window is the Front Panel, and the white one is the Block Diagram.

Note: The Front Panel contains the parts of your VI used for the User Interface and presenting information, whereas the Block Diagram contains the graphical code that controls the functionality of the VI. You can toggle between the two windows by selecting Window» Show Block Diagram or Window» Show Front Panel. You can also switch between the windows by pressing <Ctrl+E> on your keyboard.
5. Examine the Front Panel and Block Diagram of this Template VI. The Front Panel contains a Waveform Chart and a STOP button as shown in the following figure.
The Block Diagram contains a **Simulate Signal VI**, which is currently configured to simulate a sine wave and plot it to the chart.

6. Switch back to the Front Panel by pressing `<Ctrl+E>`. Since the **Run** button (the white arrow at the top left corner) is solid, you can run this VI as it is. Click the **Run** button and examine the operation of the VI. When you are finished, click the **STOP** button on the **Front Panel** to stop running the VI.

**Note:** As you will see later in the exercise, when the **Run** button in the upper left corner of both the Front Panel and the Block Diagram changes from a solid white arrow to a broken grey arrow, this new icon indicates that the VI is currently not executable. Pressing it will reveal the programmatic errors that are preventing the code from executing.

7. Now we can add some functionality to this basic VI. We will modify the VI to flash an alarm whenever the signal value is above a certain level. Open the **Controls** palette (if it is not open already) by **right-clicking** the anywhere on the grey area of the **Front Panel** window. A small pin icon in the upper left corner of this palette appears. Click this pin to force the palette to remain on your screen.
8. Navigate to the Express palette and click on the Numeric Controls sub-palette. Select a Vertical Pointer Slide to be placed on the Front Panel. To do this, click the Vertical Pointer Slide (you will only see Pointer Slide in the Controls Palette) and drag it onto the Front Panel.

9. Click the Express menu item on the Controls palette to return to the Express Controls palette.

10. Click the LEDs sub-palette, and place a Round LED on the Front Panel.
11. Right-click the **Vertical Pointer Slide** and select **Properties**. A property page will appear. Examine the different properties that you can modify. Make the following changes on the **Appearance** tab and click **OK** to apply the changes.

**Label**: Limit

**Slider 1**: Check **Show digital display(s)**

![Image of property page with options highlighted]
12. Right-click the Round LED labeled **Boolean** and select **Properties**. Examine the different properties that can be modified. On the **Appearance** tab, change the label from Boolean to **Alarm**. Click **OK** to apply your change.

Move the objects on the Front Panel so it resembles the following.
13. Switch to the Block Diagram by pressing <Ctrl+E>. Double-click the **Simulate Signal** Express VI to bring up its properties window. Examine the different properties you can modify. Change the **Amplitude** of the signal to **10**. Click **OK** to apply this change and to close the properties window.

14. Bring up the **Functions** palette by right-clicking on the Block Diagram (Any Where in the white space). Select **Express»Signal Analysis** and place the **Amp & Level** Express VI on the Block Diagram by dragging and dropping as before.
15. When you place an Express VI on the Block Diagram a dialog box appears so that you can configure the function as per your needs. For this function select **RMS** as shown below. Click **OK** to apply your change.

![Configure Amplitude and Level Measurements dialog box]

16. Wire the **output** of the **Simulate Signal VI** to the **Signals** input on the **Amplitude and Level Measurements VI**.

Note: To wire move your mouse towards the wire till mouse icon changes to the wiring tool( ), click on it once and click on the Input of the **Amplitude and Level Measurements**.
Right-click on the **RMS** output and select **Create» Numeric Indicator** from the list. (This will create a numeric indicator on your front panel where you can view the calculated RSM value of the signal.)
17. Bring up the functions palette by right-clicking the Block Diagram. Select **Arithmetic & Comparison** > **Comparison** and place the **Comparison** Express VI on the diagram.

When you place the Comparison Express VI on the Block Diagram, a dialog box appears that lets you configure what type of comparison you will be doing. Make the following selections, then click **OK** to apply these changes and to close the dialog box.

**Compare Condition**: Greater

**Comparison Inputs**: Second signal input
18. You can connect Controls, Functions, and Indicators on the Block Diagram by pointing to an object and clicking on its terminal when the cursor changes to a spool of wire (wiring tool). You can then move the cursor to the terminal of the object you want to connect it to and click again. Following this, connect the **Limit** control to the **Alarm** indicator.

![Connection Diagram]

**Note:** The Run button in the upper left corner of both the Front Panel and the Block Diagram has changed from a solid white arrow, to a broken grey arrow. This new icon indicates that the VI is currently not executable. If you click the Run button when it is solid and white, it runs the VI. Clicking it when it is broken and grey brings up a dialog box that will help you debug the VI.

19. Click the Run button now. The resulting dialog box shows that, in this case, the error results from connecting terminals of two different types. Since the Limit control is a Numeric type and the Alarm indicator is a Boolean type, we cannot wire these two terminals together. Highlight the error by clicking it, and then click **Show Error**. LabVIEW will highlight the location of the error.
20. Notice that the wire between Limit and Alarm is dashed and a red $\times$ is displayed on it.

To delete this broken wire, press $\text{Ctrl+B}$. This keyboard shortcut removes all broken wires from the Block Diagram.

21. Make your Block Diagram resemble the following image by completing these steps:

a. Wire the Limit control to the Operand 2 input of the Comparison function.

b. Connect the wire between the Amplitude & Level Measurement block and the RMS Indicator to the Operand 1 input of the Comparison block.

c. Wire the Result output of the Comparison block to the Alarm indicator.

Your Block Diagram should now resemble the following image.
22. Switch to the Front Panel by pressing <Ctrl+E>.

23. Save the VI in the ..\Desktop\CompactRIO Hands On\cRIO Data logging Folder folder by using the File menu and name it Exercise1.vi

   **Note:** Be sure to save this VI, as you will be using it later in the seminar.

24. **Run** the VI. While the VI is running you can change the **Limit** value. Also notice that when a data point received from the *Simulate Signal VI* is greater than the **Limit** value, the **Alarm** indicator lights up.

   While the VI is still running, switch to the Block Diagram by pressing <Ctrl+E>. Enable **Highlight Execution** by clicking on the light bulb on the tool bar. This will allow you to see the flow of data through your program.

25. Switch to the Front Panel by pressing <Ctrl+E>. When you are finished, stop the VI by clicking the **STOP** button on the Front Panel.
Exercise 2a – CompactRIO I/O Basics

In this section you will learn the following:

- How to build a project to target your CompactRIO
- How to read Analogue Inputs and Outputs.
- How to use specialty Digital Outputs

1. Launch LabVIEW and select File-> New from the LabVIEW Getting Started window. When on the dialog box opens expand the Project folder and click on Empty Project.

2. Save the project as CompactRIO Exercise 2.lvproj in \Desktop\CompactRIO Hands On\cRIO Data logging Folder.

3. Right-click on the top-level project item and select New » Targets and Devices...
4. Leave the setting on **Existing Targets or devices** and expand **Real-Time CompactRIO**, select your cRIO (named **cRIOHandsOn** in the following pictures) and click OK.
5. A pop-up window will then appear. You can either chose to run in Scan mode or regular FPGA mode. Select **Scan Interface** and click **Continue**. LabVIEW will automatically detect your I/O modules and add them to the project.

If it brings up a dialogue box requesting you to discover the C series modules, select **Discover**. LabVIEW will discover all C Series I/O modules currently installed in your CompactRIO chassis and add them to a LabVIEW project.

6. Configure the NI Scan Engine from the cRIO properties page. Right-click your cRIO and select **Properties**...
7. Select **Scan Engine** from the Category field. Configure the **Scan Period** for **100ms** and click **OK**. This setting tells the NI Scan Engine to update I/O every 100 milliseconds.

![Scan Engine Configuration Window]

8. Right-click on your cRIO once more and select **New » VI...**
9. Press <Ctrl-S> to bring up the save dialogue box. Save your new VI as **Scan Mode.vi** under \Desktop\CompactRIO Hands On\cRIO Data logging Folder.

10. Go back to the Project Explorer. Expand the **Chassis** and the **NI 9211** module (Mod1 in this picture below). Click and drag **AIO** from this module of the project into the block diagram of your new VI.

11. Right click on the output of the **AIO** terminal and select **Create » Indicator**.

12. Place a **Wait** node from the Functions palette under **Programming » Timing » Wait (ms)**.
Set this wait to a period of 200ms by right clicking on the input and selecting Create » Constant. Set the constant to 200.
13. Select a **While Loop** from the Functions palette under **Programming » Structures » While Loop**. Drag the While Loop around all the objects that you have on your block diagram. Right click on the input of the **Condition terminal** in the right-hand corner of the **While Loop** and select **Create Control**.

14. Switch to view the **front panel** by pressing **<Ctrl-E>**. Right-click on the **Numeric Indicator (A10)** and select **Replace** then select **Thermometer** from the Numeric Controls sub-palette.

15. Rearrange the front panel to resemble the picture below, or otherwise to your liking. Save the VI and then run it, by clicking the white run arrow in the top-left corner.

**Note:** If prompted by a Conflict Resolution dialog box, just click 'OK' to deploy the new Scan Engine settings.

16. **Blow on the Thermocouple** and notice the temperature change is displayed.

17. Stop the VI by clicking your stop button.
Exercise 2b– Synchronized loop with Digital I/O

1. View the Project Explorer and right-click on the NI 9472 module, and select Properties from the menu that appears.

2. From the Category section on the left select Specialty Digital Configuration and then choose Pulse-Width Modulation from the Specialty Mode drop-down list.

3. From the Frequency drop-down menu select 50Hz (20,000 us). Then click OK.

4. From the project window Expand Mod 2 and drag PWM0 into your Timed Loop on the block Diagram
5. Add a Dial to your front panel. To do this, open the **Controls** palette (if it is not open already) by right-clicking the **Front Panel** window, and select **Express » Numeric Controls » Dial**.

6. Change the dial so the range of values it can change between varies from 0 to 20. Do this by double clicking on the minimum and maximum value and entering a number manually.

Your front panel should resemble the image shown below.
7. Switch to the block diagram by pressing <Ctrl-E> and Drag the PWM0 from Mod2 to the block diagram.

Wire the Dial to the input terminal of the PWM0 variable.

8. (Optional) Click the Clean Up Diagram button or press <Ctrl U> to tidy your block diagram.
9. Switch back to your front panel, save <Ctrl-S> and run the VI.

If LabVIEW brings up a conflict resolution dialogue box, showing the Reconfiguration of the NI 9472 module, then click Apply.

Look at the DO0 LED on the NI 9472 module; note the brightness gradient created by the PWM signal when you adjust the Dial on your Front Panel.

When you are ready, stop the VI by returning the front panel Dial back to its 0 position, then hit the stop button on your front panel.

10. Save and close the project and all open VI's.

If prompted to disconnect, press disconnect and save all the VI's and project
Exercise 3 – Create a Data Logger with CompactRIO

In this section you will learn the following:

- How to build a scalable software architecture
- How to incorporate Data Acquisition into this structure
- How to log data onto the cRIO

1. In LabVIEW select File>>Open Project and navigate to Desktop\CompactRio Hands On\cRio Tour Exercise\cRIODataLogger.lvproj

   ![LabVIEW Project Explorer](image)

   Select this file and click OK.

2. Within the project window that appears you will see that you have two targets; your development machine and the cRIO target. Select File>>New.
3. The tree diagram on the left side of the “New” Dialogue box suggests good starting points for VI development. Expand out the diagram and select 
VI>>From Template>>Frameworks>>Design Patterns>>Standard State Machine
as shown below. Click OK.

4. In the Project Window your new VI will be listed on the tree diagram under My Computer 
click and drag this VI down the project until it is situated under the cRIO named 
cRIOHandsOn.
5. Select **File>>Save**. A dialogue box will appear referring to an unsaved ctl file. Answer **Yes** and then save both files as **“StateMachineCRIo”** under **Desktop\CompactRIo Hands On\cRIo Tour Exercise**.

6. Turn to view the **Block Diagram** of your new VI (**Ctrl-E**). Your will see the simple architecture of a state machine with a case structure within a while loop. The first task for us is to ensure that our architecture will stop correctly. To do this, build the block diagram by undertaking the following steps:
   a. Look at the **Front Panel** of the VI and place down a **Modern>>Boolean>>Stop Button** (*You might need to expand you control pallet by using the two down arrows as shown on the image blow to get to modern palette*).
   b. View the **Block Diagram (Ctrl-E)**. Drag the right-hand edge of the **while loop** out to give you some more space.
c. Hold down Ctrl button and click – hold – and drag the “Stop” enumerated constant to make a copy of it.

**Help**: Enum or Enumerated is a combination of data types. An enum represents a pair of values, a string and a numeric, where enum can be one of a list of values. Enums are useful because it is easier to manipulate numbers on the block diagram than strings.

d. Place down a select function found in the functions palette under

**Programming>>Comparison>>Select**

(You might need to expand you control pallet by using the two down arrows to get to the programming sub-palette).
e. Wire the block diagram to resemble the picture below, using your stop button as the decision maker for the Select function.

7. The State machine currently has only two States (Initialize and Stop - which can be viewed as the cases within the case structure). The next stage is to add more cases, so the State machine can meet our needs.
   a. Right click on the blue enumerated constant showing “Stop” that we have just created and select “OpenType Def.” This will bring up a window showing the control’s prototype.
   b. Right click on the control labeled “States” and select “Edit Items...”
c. Insert “Acquire”, “Decision” and “Log Data” states within the dialogue box and click OK.

d. Save the Control you have edited (Ctrl-S) and close front panel with either the cross in the top right hand corner or by selecting File->Close.
e. On the block diagram of the state machine VI **Right Click** on the border of the Case structure and select **Add Case for Every Value**.

Help : The purpose of this step is to add a case on the case structure for each case that we added in the enum control. We added Acquire, Decision and Log Data states.
f. **Left Click** on the **Enumerated Constant** within the case structure of the **Initialize** State, to bring up the list of defined values. Select the **Acquire**.

![Diagram](image1)

![Diagram](image2)

---

g. Take a copy of the **Enumerated Constant** (ctrl-C), and paste (ctrl-V) it in to each of the newly added states (namely, **Acquire**, **Decision** and **Log Data**).

h. In the **Acquire** state, **Left Click** the newly added **Enumerated Constant** and change the value to **Decision**.

i. Repeat this action for the other states to produce a sequence that flows as follows:

- Initialize > Acquire
- Acquire > Decision
- Decision > Log Data
- Log Data > Acquire
- Stop > Stop

8. The state machine architecture has now been created; next we must add functionality to each state. **First Start with the Initialize state.** *(Make sure that the selected value of the enum is Acquire)*

![Diagram](image3)
9. Now change to view the **Acquire** State.

![Acquire State Diagram]

10. Here we need to define the I/O that we need and pass it into an array to be accessed by all states of our state machine.

   a. Listed under the cRIO chassis in the tree diagram of the project window drag in **PWM0** and **PWM1** from **Module 2 (NI 9472)**, **AI0** from **Module 1 (NI 9211)**

![I/O Diagram]

   b. Switch to view the Front Panel (**Ctrl-E**) and place down a **Horizontal pointer slide** found on the controls palette at **Modern>>Numeric>>Horizontal Pointer Slide**.
c. Change the range of this slide to represent 0-100 by double clicking on the higher limit (10) and changing the value to 100, the scaling will adjust accordingly. Also rename this control to PWM0

Make a copy for PWM1

d. On your front panel, place down **Modern>>Numeric>>Thermometer** and **Modern>>Graph>>Waveform Chart**
e. Name them **Temp** and **Thermometer** respectively. Your front panel should now look similar to the one shown below:

f. Switch back to the Block Diagram (**Ctrl-E**) and wire the new controls to their relevant input and output variables as show below in Acquire Case.
g. On the Block Diagram place down Programming>>Array>>Build Array


h. Expand the Build Array function to accept 3 inputs using click and drag.
i. Wire the output of the Build Array to the right hand wall of the while loop. On the Black colored (undefined data type) tunnel that is created, right click and select replace with shift register.

![Diagram showing the connection and shift register placement.]

j. A shift register has now appeared on the left hand wall, this is for passing information between iterations of the while loop.
k. Wire the output of this shift register to the left hand wall of the case structure.

l. Ensure that the **Enumerated Constant** previously placed in the Acquire state is set to **Decision**.

m. Complete the wiring of your **Acquire** state so that the block diagram looks similar to the one shown below:
n. **Right Click** on the output tunnel and select **Linked Input Tunnel** >> **Create & Wire Unwired Cases**

![Diagram](image1)

o. Your cursor will change to a wiring tool. **Left Click** in the **input tunnels** of the case structure. This will wire the array through in all other cases in the case structure.

![Diagram](image2)

11. Change to visible case in the case structure to the **Log Data** State. Here we shall log data to the hard drive of the CompactRIO in an easily accessible and portable format.
   a. Ensure that the **Enumerated Constant** previously placed in the **Log Data** state is set to **Acquire**.

![Diagram](image3)
b. Place down Programming>>File I/O>>Write To Spreadsheet File

c. Right click on the inputs for file path, append to file? And Delimiter and select Create Constant.
d. Set each of the newly created constants to the following values
   i. File Path - c:\datalog.csv
   ii. Append to file? - True
   iii. Delimiter - ,
e. Wire the data array into the 1D Data input of the Write to Spreadsheet File VI to create a block diagram similar to the one shown below:

12. Change to View the Stop state.
   a. From the Project Window drag and drop variable Module 2>>PWM0 into the stop case.
   b. Right click on the input terminal and select Create Constant. Leave the constant at a default of 0.

13. Save and close the StandardStateMachineRIO.vi
14. From your project open the *StandardStateMachinecRIO.vi*. 

15. **Run the VI.** Try altering the speed of the PWM output with the control on the front panel and to see the LED is changing the color intensity accordingly.

16. (Optional) You can open windows explorer and type in `ftp://10.10.10.11/` and access the compactRIO's hard drive. You will see a file called datalog.csv file (The file you just logged data). Copy this file to the desktop and open in excel or notepad to view the data. (More information on exercise 4)
Exercise 4 – Accessing Logged Data on the cRIO

For the final section, you will use FTP (File Transfer Protocol) to view the files held on the cRIO.

Note: File Transfer Protocol (FTP) is a standard network protocol used to copy files between networked machines.

1. Open windows explorer by selecting Start » Computer
2. Left click in the address bar at the top of the explorer window, and enter the address ftp:\\10.10.10.11 (this is the IP address of your cRIO). This will show you exactly which files have been stored on the cRIO hard drive.

3. Double Left Click the datalog.csv file. If you are prompted by a File Download dialog box, just click Open. This will open your datalog directly into Excel on the host machine.

Note: You have configured the Data Logging application to append all of the acquired data to a single .csv file. However, in LabVIEW, you have complete control over your File IO, enabling you to meet your exact datalogging requirements. You could save data as individual or multiple files, include file headers & additional test information and save data in a variety of common file formats – including .txt, .cvs, .doc, .xls, .lvm, .dat and .tmds.