UNIVERSITY OF NORTH CAROLINA AT CHARLOTTE

Department of Electrical and Computer Engineering
ECGR 4161/5196 Introduction to Robotics

Experiment No. 1 – Introduction to LabView

Overview: The purpose of this experiment is to familiarize the student with LabView and to introduce the robot that they will be working with this semester.

LabView is a programming language that uses a graphical interface to create programs. Each program created using LabView is called a VI. A subVI is similar to a function call and is simply a program within another program. There are two windows the front panel and block diagram. The front panel displays the controls, graphs, and indicators of the program. The block diagram shows the mechanisms of the program such as the control structures like case statements, for loops, while loops, math operations, subVIs, and the wiring done for all components.

The robot is already assembled and ready to be programmed. No major modifications to the robot, in terms of mechanical additions or to the electronic components, are necessary. Minor additions may need to be made in later labs. The robot consists of a single board RIO (sbRIO) FPGA board with pins on top in order to easily attach new sensors to the robot whenever needed. The chassis holds the motors of the robot and hides the multitude of wires inside.

The sensor that is attached to the front of the robot is a type of sensor called an ultrasonic sensor. The name of this sensor is the Parallax PING ultrasonic range sensor. This sensor, shown below in Figure 1, sends out an ultrasonic pulse. The distance is measured by the sensor by measuring the time for the echo to return. The Parallax PING ultrasonic range sensor incorporates an activity status LED and has one I/O pin.

Figure 1. Parallax PING Ultrasonic Range Sensor [1]

Lab-Session — Measurements Lab

Introduction to LabView

Note: You many refer to National Instruments tutorial for LabView by going to their website, shown on the references page, and searching Getting started with the LabView robotics starter kit "DaNI". The first item shown in the list should be the correct link to the webpage with the tutorial. Simply download the zip file or the word or presentation file depending on which page you are on.

1. Launch LabView Robotics and then launch the "Hardware Setup Wizard." The wizard is located on the "Getting Started" tab.
2. Follow the instructions provided by the wizard to configure and test the hardware. Be sure to inform the wizard that we are using the LabView Robotics starter kit by checking the checkbox.
   (Note: It may take several attempts to locate the device.)
   a. By following the wizard, the ultrasonic sensor will also be tested and calibrated.
3. Once the hardware setup is complete and it has been configured, make sure that you have selected to create a new robotics project in LabView and then click the Exit button. This will then launch the LabView Project Wizard.
4. Once in the LabView Project Wizard, choose the "Robotics Starter Kit" as the project type and then click "Next."
5. Name the project "MyFIrstProject" and create a folder that will hold the file.
6. A working example will popup once the LabView project is generated. Run the program by clicking on the run arrow at the top of the application. This will download the VI onto the sbRIO and start the execution. Be sure to follow the instructions on the front panel.
   Note: you may see a dialog box like this:

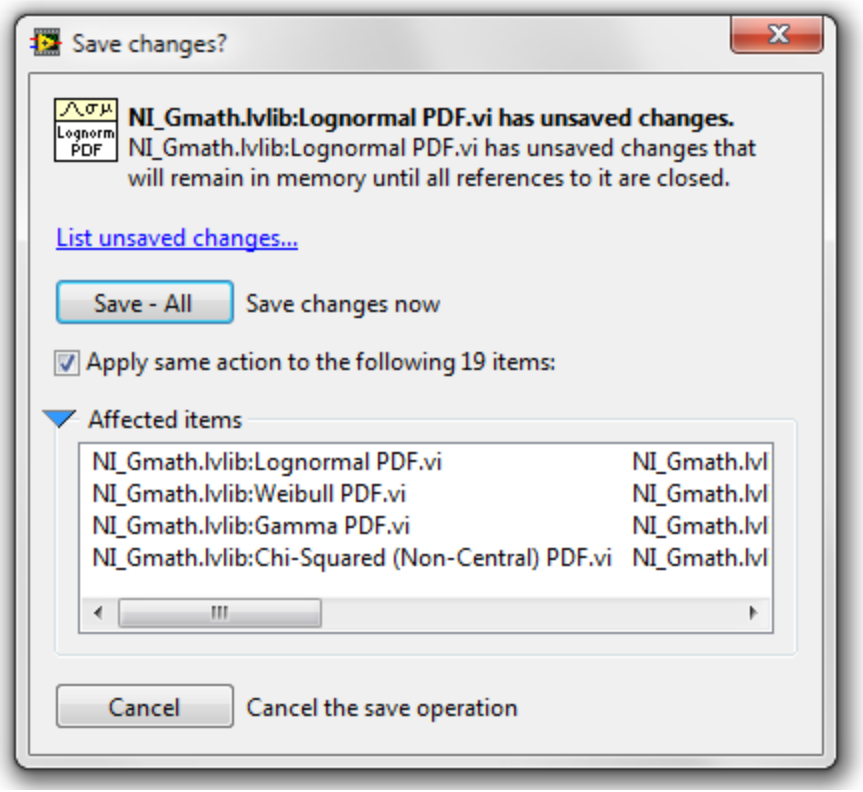

Figure 2. Dialog Box to save changes [1]

Simply choose save all.

Creating your First VI

1. To create your first application, right click on "Starter Kit sbRIO" and choose New>>VI.
2. Save the newly created VI as "MyFirstApp". Once this is done, a new element will show up on the project tree.
3. Choose Window>>Tile Left and Right to see the Block Diagram and Front Panel side by side.
4. Begin programming on the Block Diagram by placing a Timed Loop from the LabView functions palette. To do this:
    a. Right click anywhere on the Block Diagram and go into the palettes by hovering over the palette that you wish to expand. To get the Timed Loop go into Structures>>Timed Structures palette. Figure 3 shows the path you need to take.

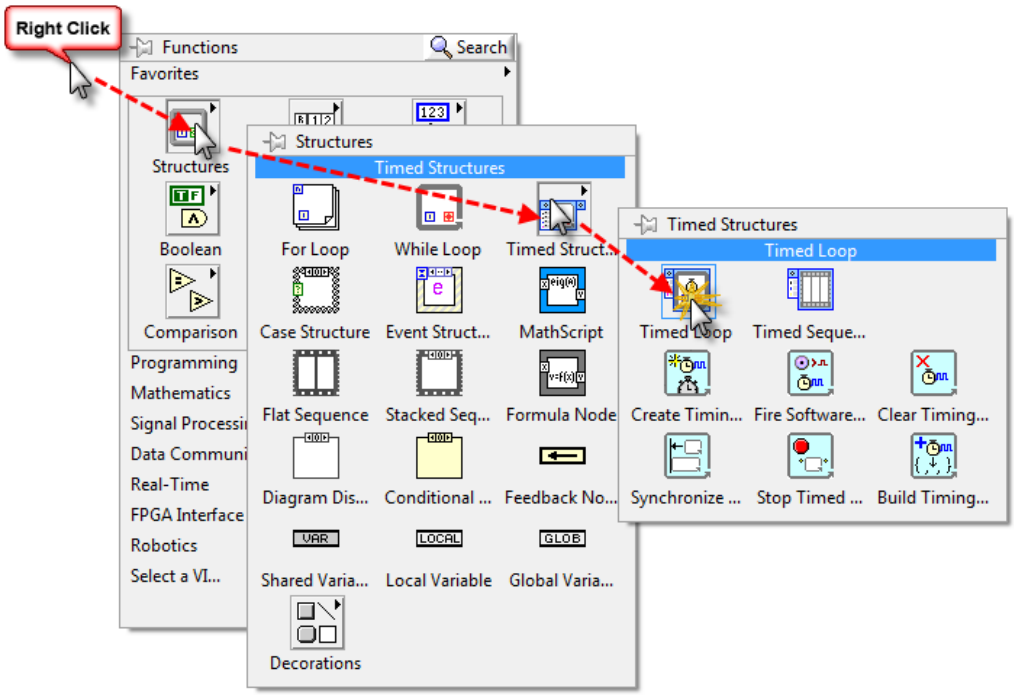

Figure 3. Maneuvering the Function Palette [1]

5. Simply click and drag as if you are selecting a region in order to place the loop on the diagram. When you release the mouse, the loop will be drawn. (This also applies to for loops and while loops.)
6. A stop button will need to be added so that you can stop the application. This is done by creating a front panel control and connecting the value of the control to the termination condition of the loop (The stop symbol in the corner of the loop).
7. To create this control, right click on the stop symbol and choose Create>>Control. Once the control is created on the Block Diagram, a stop button will appear on the Front Panel that corresponds to the newly created control.
8. This also works the other way. To prove this we will now create a chart. Right click on the front panel and select Graph>>Waveform Chart. (Figure 4 shows how this is done.) You can resize the chart to any dimension and a corresponding terminal has been automatically added to the Block Diagram.

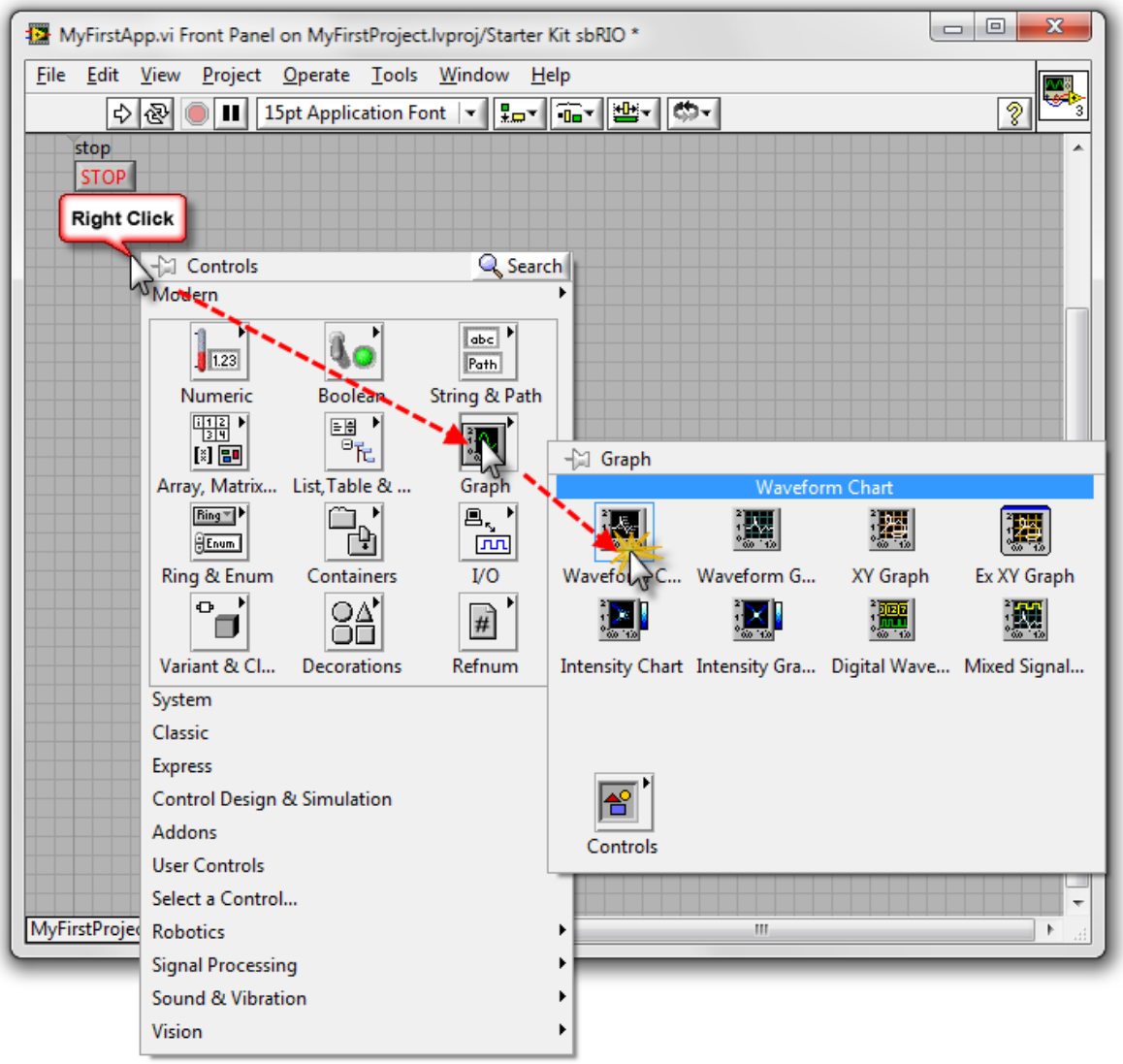

Figure 4. Placing a chart on the Front Panel [1]

9. Go back to the Block Diagram and add some code to communicate to the ultrasonic sensor. This will be done by accessing the FPGA bifile. Place an Open FPGA Reference on the Block Diagram. Right click on the Block Diagram and select FPGA Interface>>Open FPGA Reference. Place it outside of the timed loop to the left.
10. Right click the Open FPGA Reference VI and choose "Configure Open FPGA Reference"
11. On the dialog box, select "Bitfile" and then navigate to where the FPGA bitfile is held in the pathway \Starter Kit Roaming\FPGA Bitfiles\.
12. Right click the resource name input in the upper left corner of the Open FPGA Reference VI and select Create>>Constant. This will indicate which FPGA you will work with.(The figure below will show you how to do this and how the VI will look so far.)

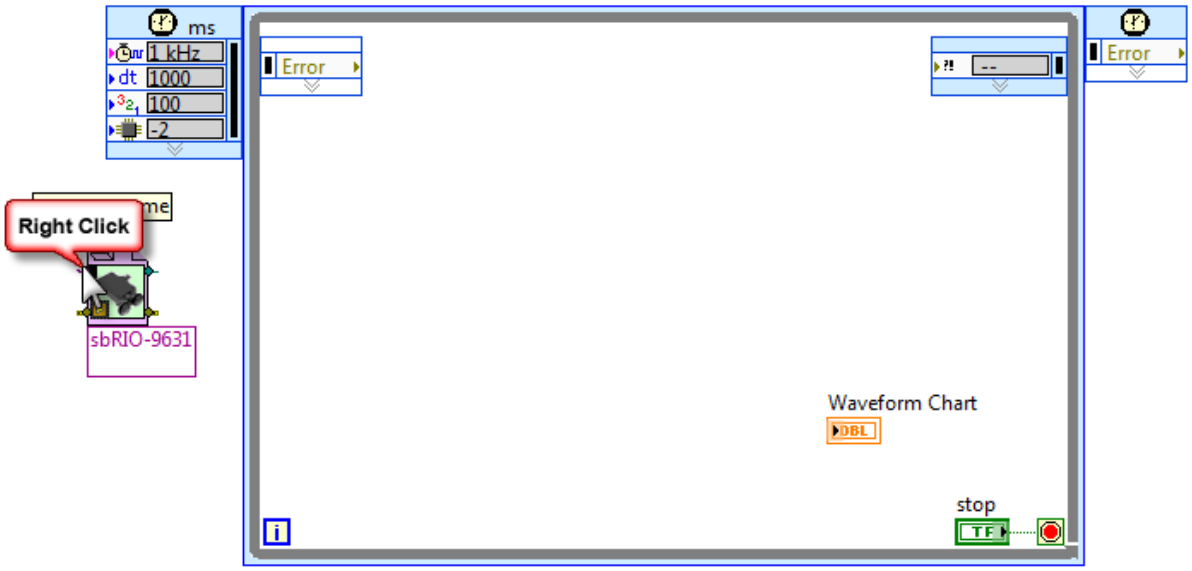

Figure 5. Creating the constant for the FPGA Reference VI [1]

13. For the constant, select "RIO0". (Note: you may just need to type this in.)
14. Now from the FPGA interface palette, select "FPGA Read/Write VI" inside the loop.
15. Connect the FPGA VI Reference "Out terminal" on the Open FPGA Reference VI to the FPGA Reference "In terminal" on the FPGA Read/Write VI. Simply click the two terminals in order connect the two with a wire.
16. From the same palette, place a "Close FPGA VI Reference" on the outside of the loop to the right side. Wire the program as shown below in Figure 6:

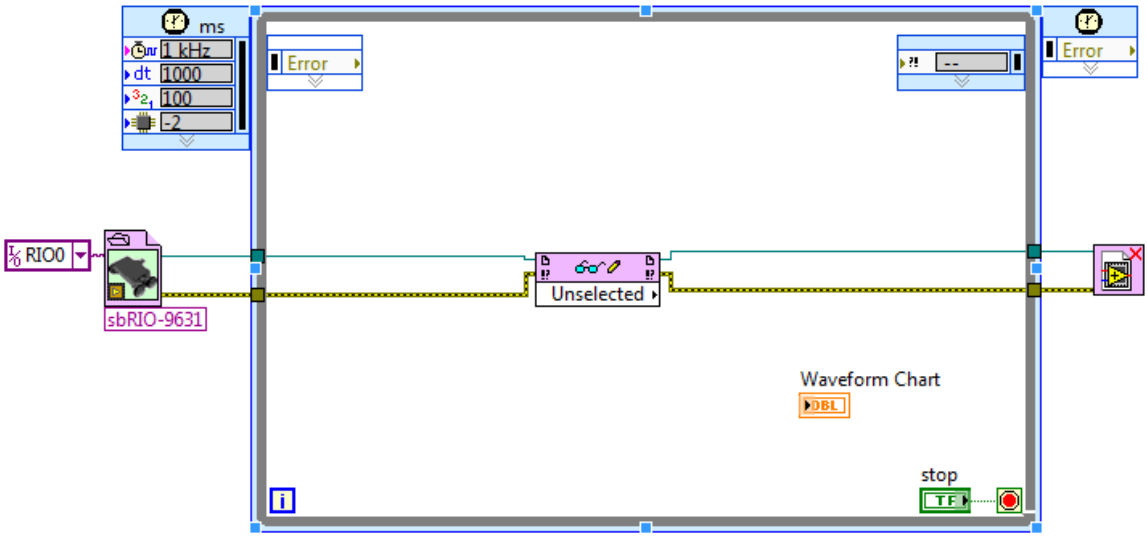

Figure 6. Wiring the elements of the program together [1]

17. Now that the FPGA Read/Write VI is connected to the bitfile, you will be able to select the value that you wish to read. Thus, select “sensor distance (m)” to read the sensor value.
18. Wire the sensor distance output to the Waveform Chart terminal. Note: to clean up your diagram at any time, click on the Block Diagram Clean-Up button or press CTRL+U.
19. Save the VI and click the Run arrow to download the application and execute it. Move your hand back and forth in front of the sensor and observe the corresponding value change on the chart.
20. Press the stop button you created when you are finished playing with your application.

Manipulating the Ultrasonic Sensor

This section will build upon the “myFirstApp” VI created in the last section.

1.) First, add a control to the front panel that will be used to set the position of the servo motor for the ultrasonic sensor. Place a Horizontal Slider control by right clicking the front panel and going through the palette by choosing Numeric>>Horizontal Pointer Slider.

2.) The range of the motion of the motor is around 90 degrees in each direction from the center. Thus, we must change the range on the slider you just placed. Simply double click on the max and min values on the slider and type in the desired number.

3.) Next, switch to the block diagram. We will make the loop run faster in order to have a more responsive behavior. Double click the upper left hand node on the Timed Loop and change the period of execution from 1000ms to 100ms. The figure below shows what value to change.

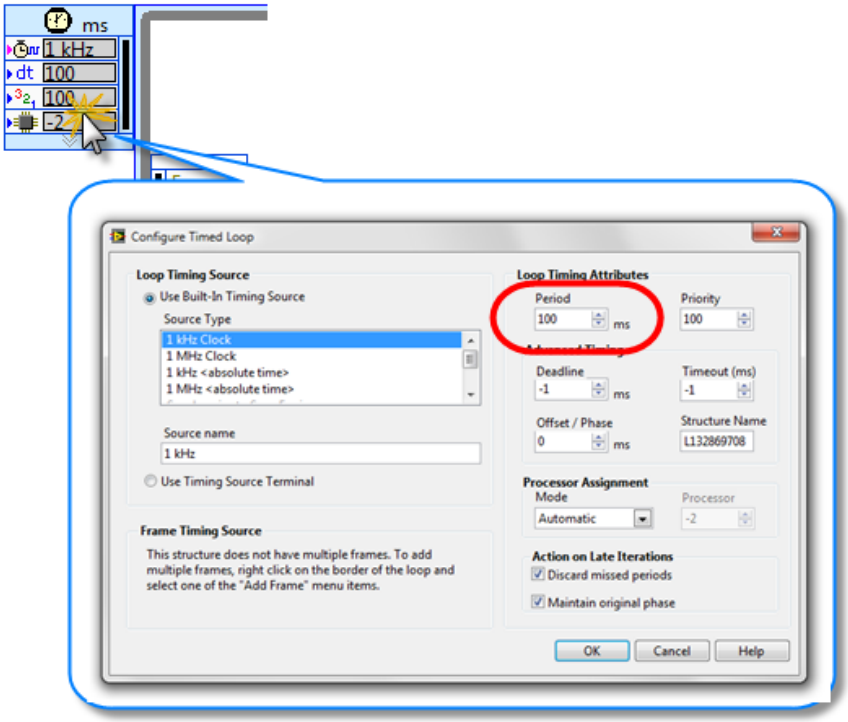

Figure 7. Setting the period for the loop [1]

4.) In order to write to the position controller, the FPGA Read/Write VI will have to be expanded. Resize the node from the bottom to expand the VI.
5.) Now select "sensor motor angle (rad)." This will create an input that will be used to pass the angle of the motor to the FPGA bitfile that implements the driver.
6.) Since the control we have placed on the front panel is in degrees and the driver only accepts angles in radians, we must convert from degrees to radians. Thus, we must use this equation:

$$\frac{\pi}{180} \times degrees = radians$$

Place Pi onto the block diagram but navigating the function palette from Numeric>>Math Constants>>Pi
7.) From the same Numeric palette we can find a Divide and a Multiply block. Place these blocks on the diagram and connect them as shown below. Also to create the constant value of 180, just right click the input to the multiply block and choose Create>>Constant.

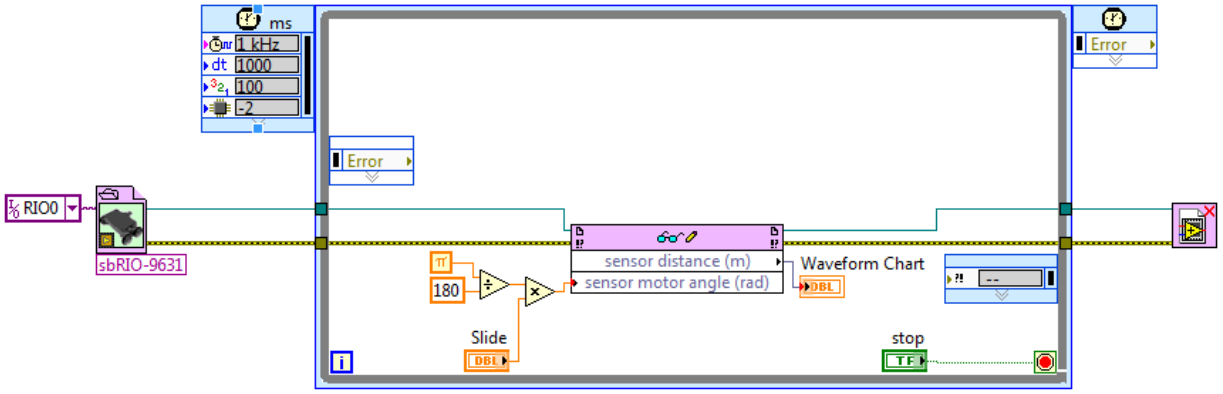

Figure 8. Connections between blocks to implement degree to radian conversion [1]

8.) Save your VI and click the Run button. As you move the slider the orientation of the sensor will change.

First FPGA Application

1.) Right click the FPGA target in the project and select New>>VI. Save this VI as MyFirstFPGA.vi. Verify that this VI appears under the FPGA target in the project hierarchy in order to make sure that the VI will run on the FPGA instead of the sbRIO.
2.) Draw an empty while loop on the block diagram. The while loop is found in the Structures palette.
3.) In order to add I/O to the diagram, expand the Onboard I/O folder under the FPGA in the project. This folder contains a list of all available analog and digital I/O on the board. We will use the LED output node. To add this node to the diagram, drag the node from the project tree and drop it into the loop.

4.) A shift register will be used here to maintain the state of the LED from one iteration to the next. To add a shift register, right click the loop and choose “Add Shift Register.”
5.) The shift register must be given an initial value for the first iteration of the loop. Thus, create a true constant and wire it to the shift register. Note: Boolean constant can be found in the Boolean palette.

6.) Place a Not block on the diagram. The figure below shows how to wire the blocks.

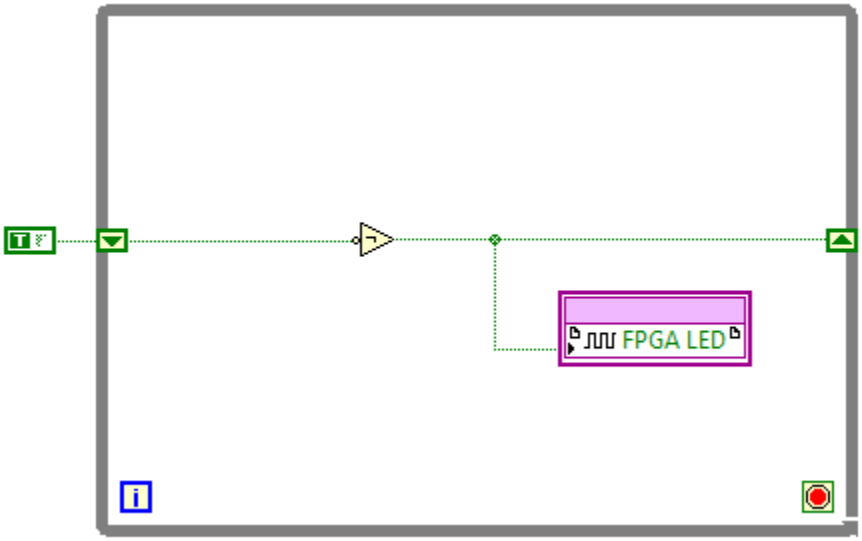

Figure 9. Setting the Boolean constant for the shift register and wiring the Not block [1]

7.) Since the FPGA is capable of controlling execution on a sub-microsecond scale, we must change the timing of the loop. Place a Loop Timer in the loop from the Timing palette.
8.) The default clock rate of the FPGA on the sbRIO is 40MHz. Thus, a “tick” takes 25ns to execute. This is obviously too fast for implementing a blinking LED. Thus, double click the timer, choose mSec from the timer units dropdown box, and then click ok.
9.) Now right click the input to the Loop Timer VI and choose Create>>Control to control the blink rate of the LED.
10.) Right click the stop symbol at the bottom right edge of the loop and create a false constant to make an infinite loop. The completed VI is shown below.

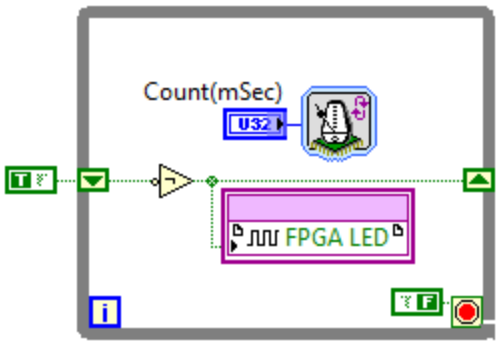

Figure 10. How finished VI should look. [1]

11.) Lastly, save and compile the application by clicking the run button.. Compiling will take several minutes to finish. To minimize the compile window click the lower right button beside Help.

12.) The application will execute once compiling is finished. Set the rate that LED will blink to 1000, which will make the LED change every second. Change the rate to 100 and the LED will blink ten times faster.

References

[1] *NI Code Exchange*. National Instruments. n.d. Web. 1 Dec. 2010. <http://www.ni.com/code>