Lab 2: Introduction to LabVIEW 8.5

INTRODUCTION:

This lab is designed as an introduction to using LabVIEW. In this lab you will run through some tutorials to get a basic understanding of some of the LabVIEW functions you may need to use later in the semester.

REQUIRED PARTS AND MATERIALS:

This lab assignment is performed using:

1) PC running LabVIEW 8.5

PRELAB:

1. Read through Chapter 1 of the Getting Started with LabVIEW tutorials. You can access this tutorial file either by viewing the Lab 2 Prelab Supplement or by logging into a DECS PC, running LabVIEW and opening the "Getting Started with LabVIEW" document from the main window. You do not need to actually perform the steps in the tutorial, just read through it to help prepare for doing the tutorials during your lab session.
2. Print the Prelab and Lab2 Grading Sheets. Answer all of the questions in the Prelab Grading Sheet and bring the Lab2 Grading Sheet with you when you come to lab. ***The Prelab Grading Sheet must be turned in to the TA <u>before</u> beginning your lab assignment.***
3. Read through the LABORATORY PROCEDURE before coming to lab. Note: you are not required to print the lab procedure; you can view it on the PC at your lab bench.

LABORATORY PROCEDURE:

This lab will be conducted entirely on the computers available in the lab. Begin by logging on to the computer using your engineering account.

1. Open LabVIEW, select "Getting Started with Labview." Read through this manual and perform the examples indicated in Chapters 1. When you complete the Ch. 1 examples, show the TA your results and <u>ask the TA to sign off on the Grading Sheet</u>.

2. Repeat step 1 for Chapter 2 examples.

3. Repeat step 1 for Chapter 3 examples.

4. Complete the following three additional exercises

Exercise 1:

This exercise contains information which will enhance the users knowledge of **data conversion** methods and an introduction into **array** usage within the LabVIEW software

1. To demonstrate **data conversion** within LabVIEW, we will have our user enter **numeric data** into the VI, and display its Octal, Hexadecimal, and binary values (using array indexing). To start this off, we will first need to introduce one numeric control. This can be found in the Functions palette.

2. After inserting the numeric control we will insert two string indicators that will will be used to display our results. Once you place both the numeric control and string indicators, you should have a VI similar to that shown in Figure 1.

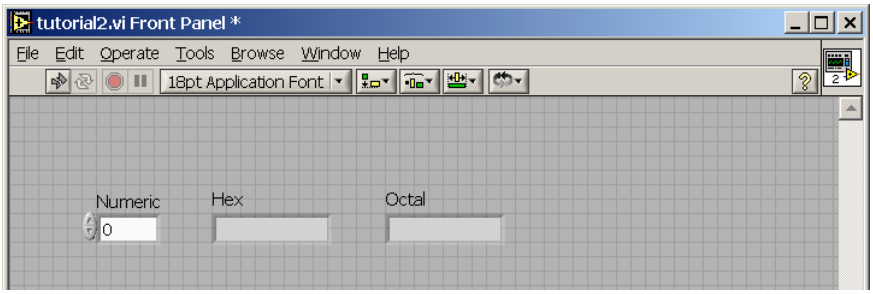

Figure 1

3. For the our data conversion, the objects required for our front panel have been taken care of. If we wish, we can change the labels of the two string indicators at this time to "Hex" and "Octal." The next step is to focus our attention on the **Block Diagram** of our VI. We can do this by pressing **CTRL+E.** From the block diagram we see something similar to what is shown in Figure 2.

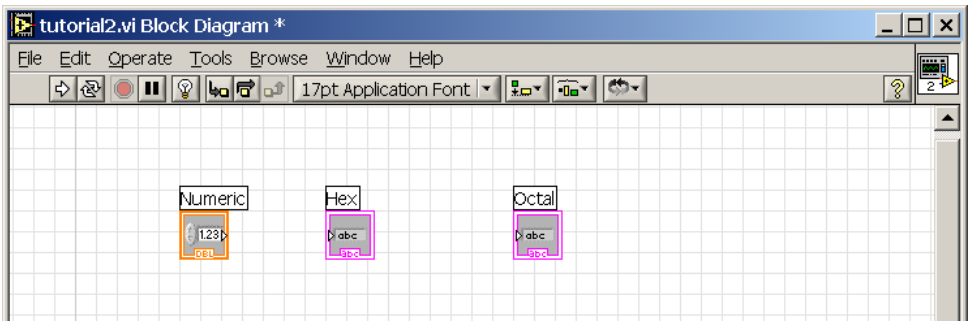

Figure 2

4. From here we need to make all of our connections. For our Hexadecimal conversion, we need to introduce the Numeric to Hexadecimal String object. This can be found in the String -> String/Number Conversion section of the functions palette. We also need to add in our Numeric to Octal String object. This is found in the same location. Lastly, we need to specify the width of our conversion. We do this by instantiating a numeric constant of 2 for our Hex conversions and 3 for our Octal conversion. Once these objects have been instantiated, and connected, your block diagram should look similar to Figure 3.

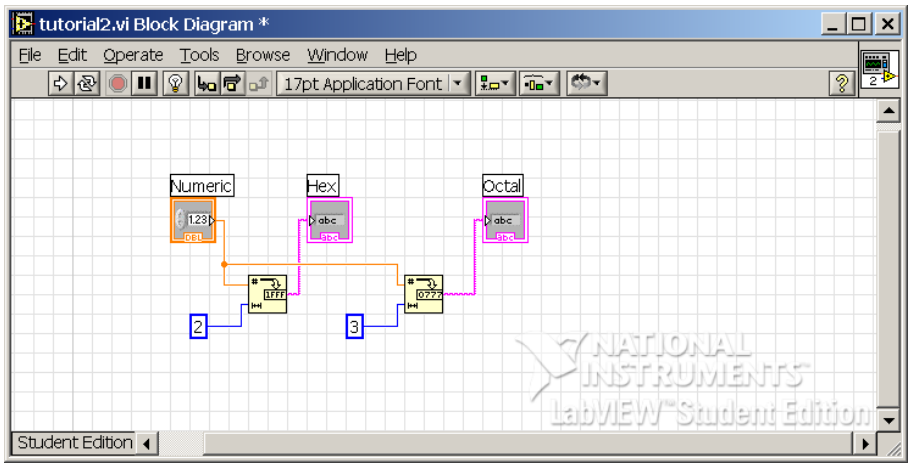

Figure 3

5. We can now test our data conversion by jumping back to our front panel and click on the Run Continuously button. Enter some values into the Numeric control to verify that your data conversions are working.

6. The next step will be to display our numeric input with an LED array. To accomplish this, we need to first add 8 LEDs to our front panel. Since it is annoying to have labels on each individual LED, you may choose not to display the labels by right clicking on the LED and deselecting Label from the Visible Items. You should end up with something similar to Figure 4.

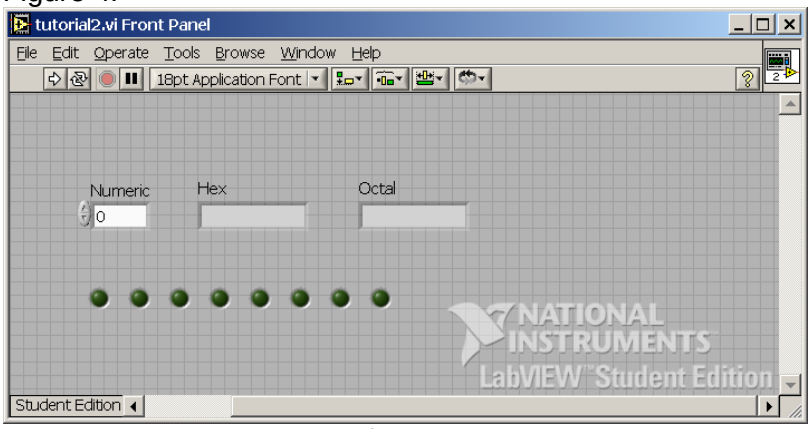

Figure 4

7. Switch over to the block diagram view. We now need to add in several objects which will allow for array indexing in order to turn on the various LEDs. Under the Numeric -> Conversion section of the Functions palette select the Number to Boolean Array object and insert it. This simple object converts our numeric input into a Boolean array that we can then index. The next object we need will allow us to access the individual items of the array. In the Functions palette, select the Array -> Index Array object and insert 8 of them, one for each LED. A numeric constant must also be introduced for each Index Array object in order to specify the indexed item. Once you have inserted all of the objects, they should be connected as shown in Figure 5.

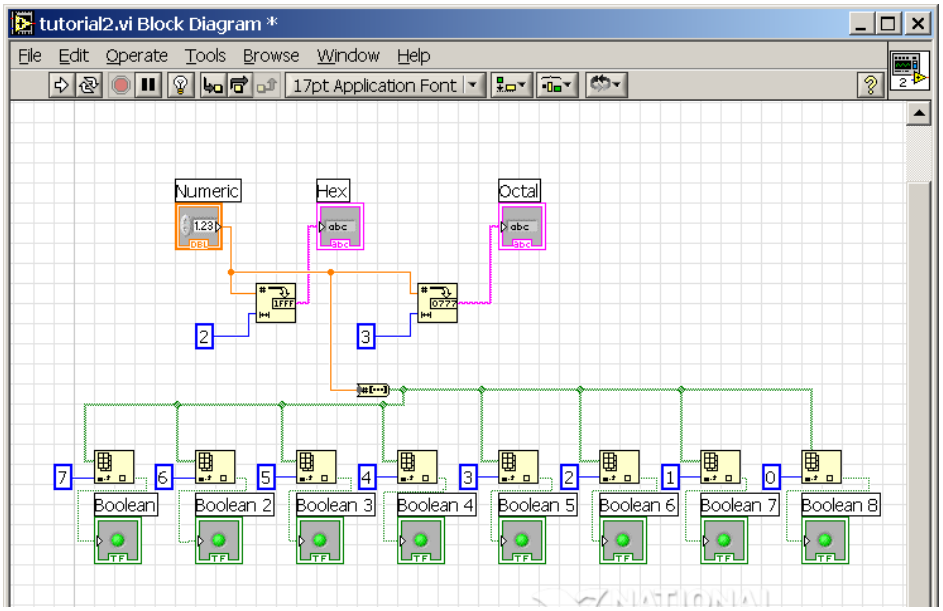

Figure 5

8. Now, feel free to jump back into the Front Panel and select Run Continuously. You will notice that for the numeric values that you enter, your LED array will display the appropriate result similar to what is shown in Figure 6.

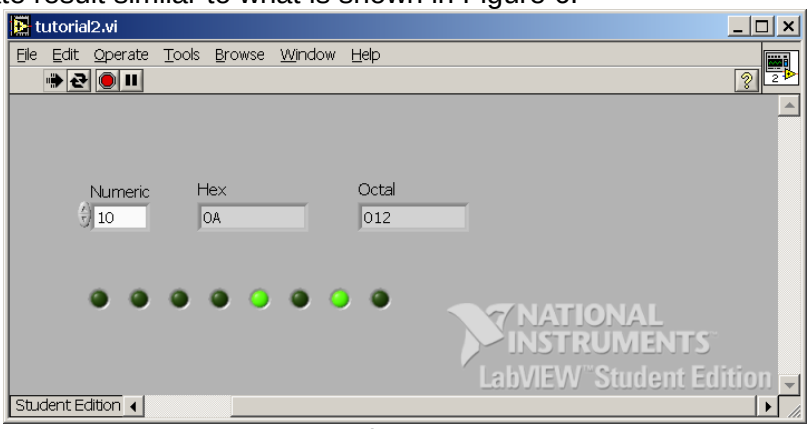

Figure 6

9. ***When you complete this tutorial, show the TA your results before proceeding.***

Exercise 2:

This tutorial contains information which will enhance the users knowledge of **Case and Sequence structures** within the LabVIEW software.

1. In many programming languages, users can take advantage of the simple IF/ELSE and CASE statements. LabView also provides a method for these simple methods known as the **Case Structure**. To demonstrate our Case Structure, we will introduce two objects into our Front Panel from the controls window. We will need introduce a Vertical Toggle Switch and a String Indicator. With these objects, our goal will be display the status of our Toggle Switch in our String Indicator (e.g. “True” or “False”). Your Front Panel should look like that show in Figure 7.

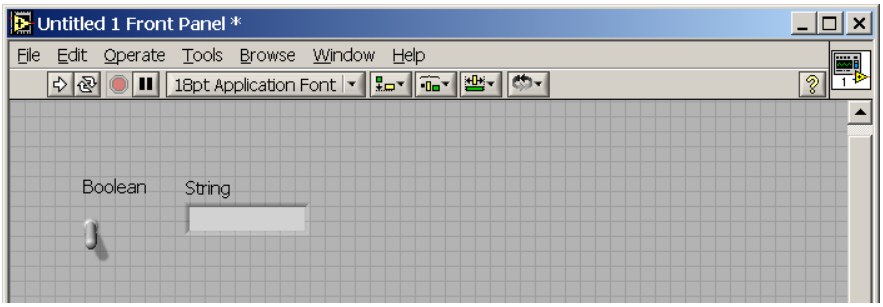

Figure 7

2. From here, we have to make all of our connections and introduce our Case Structure. What we plan on doing is connecting a boolean value (True or False) to our Case Structure, which is visually a “box-like” object. At the top of our Case Structure box, you'll notice that we have there is a small menu which has the “case options.” Depending on which case is selected in this menu depends on the action that is taken when that particular case is active. This all makes more sense when it is seen visually. Please refer to Figure 8(a&b) to see how all of the connections are made.

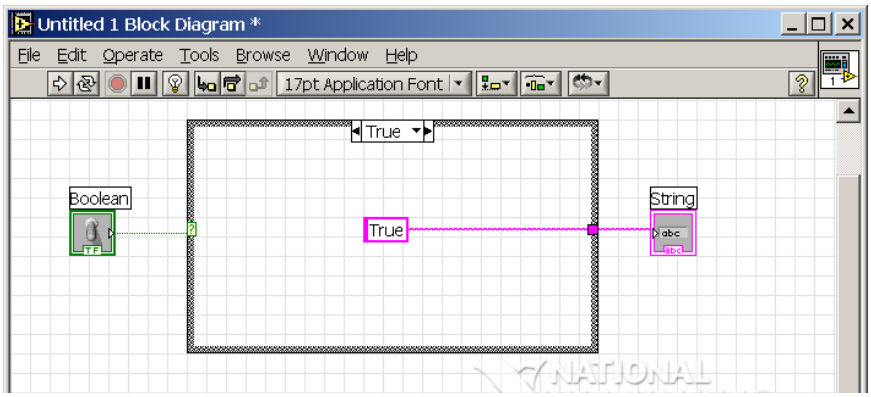

Figure 8 (a)

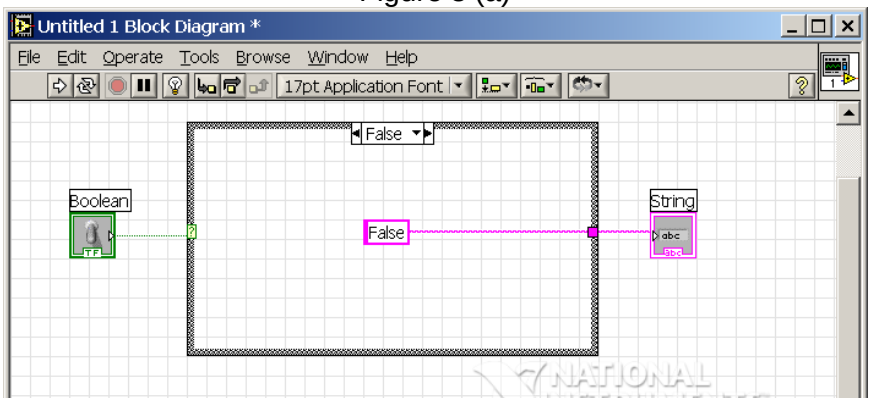

Figure 8(b)

3. You will notice that from the connections made in Step 5, the Boolean toggle switch acts as our input, and the Case Structure lets us define what will happen for the different boolean values. As a quick note, the Case Structure may also have a Combo Box (drop down menu) as an input if multiple case definitions are required. Feel free to jump back to the Front Panel and select Run Continously. Toggle the switch and you'll see that for its two different values, True or False will be displayed in our string indicator. This can be viewed in Figure 9.

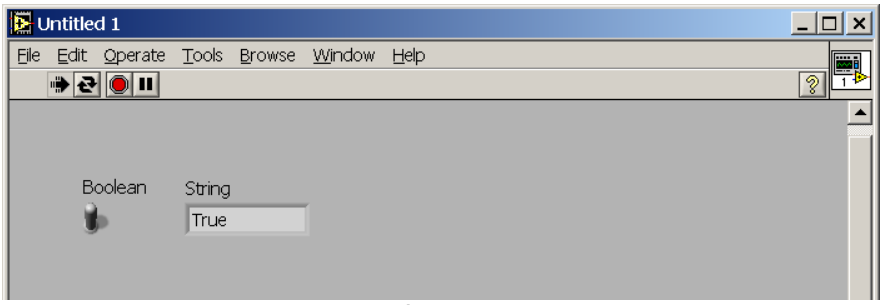

Figure 9

4. The next part of this tutorial will be to implement a Sequence Structure. To accomplish this, we are going to extend our existing VI and add a Sequence Structure within our existing Case Structure and use it to create a 1 second delay on our toggle switch. Go ahead an insert a Flat Sequence Structure within our Case Structure (both the True and False cases), right click on frame, and select “Add Frame After.” This will create a two frame Sequence Structure. In the first frame of our sequence, we are going to insert a time delay. This is done with the Wait object, and a numeric constant. In the second

frame, we will have our True and False values, depending on the case. To better view this representation, please look at Figures 10(a&b).

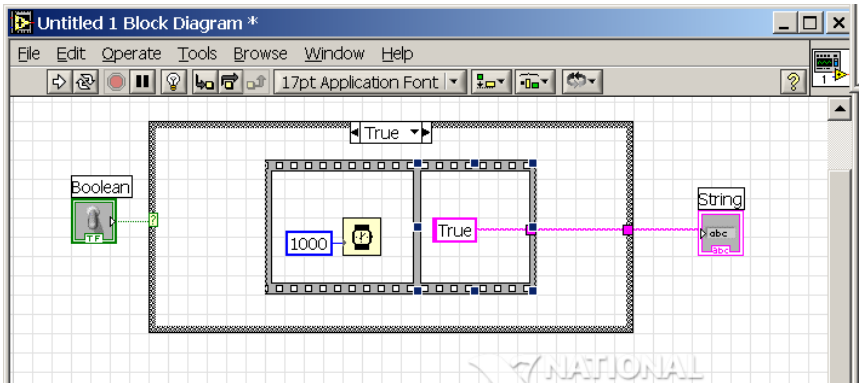

Figure 10(a)

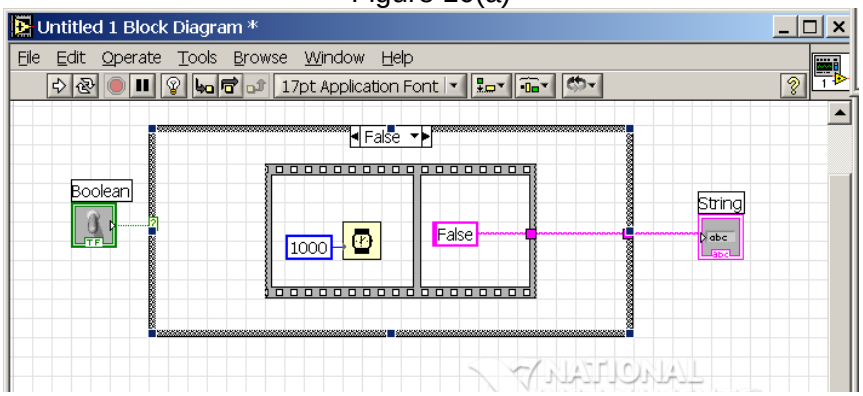

Figure 10(b)

5. Now go back into the front panel and select Run Continuously and toggle our switch. You will now notice that there is a delay in our output after we modify the swtich.

6. ***When you complete this tutorial, show the TA your results before proceeding.***

Exercise 3:

Now we will look at some of the aesthetic components of the LabVIEW program.

1. Start by creating a blank VI. First place 4 “Simulate Signal” controls on the block diagram. Each time you place one, configure it to simulate a different waveform (sine, triangle, square, and sawtooth). After these are configured, place 4 “Waveform Graph” controls and connect a different signal to each waveform graph. This allows us to view the signals. Now you should be able to run the VI; do that now and explain what happens when you do. How would you fix this?
2. The VI is only run once, so we need something that allows us to actually view the signals. If we place a while loop around the entire VI, it will allow the waveform graphs to continuously show the input signals. Do this, then run the simulation again. What happens now? Stop the simulation and go back into the properties of the individual signal generators (right-click->properties). You will notice near the bottom of the properties box that the signal generator is generating 100 samples at 1000 Hz. The waveform graph then displays this data and the while loop starts the process over. Changing the frequency of your signal to 20.1 Hz will put the period of the generated waveform and the sample period just out of synch so that you are able to watch the waveform slowly move across the screen. **Note that the time between waves moving

across the screen is not the period. To determine the period you MUST use the timing information on the x-axis.** After doing that, run the simulation again and observe what happens. To fix this final part, open the properties of the waveform viewer. In the scales tab, deselect the autoscale box for the amplitude, then insert plus/minus twice the amplitude of each signal. When you run the simulation again, you should be able to see a clear waveform moving across the screen. Include a delay(ms) block with 10ms delay if necessary

3. What if you wanted to amplify a given signal? With LabVIEW it is quite easy. To start, change all the waveform viewer amplitude scales to plus/minus 10. After that, go to the front panel and select four different numerical controls (i.e. a slider or knob) and place them near the waveform viewers. Go to the block diagram and delete all wires, place the numeric controls inside the while loop, and place four "multiply" controls on the panel. Now, connect the signal to one of the inputs of the multiply control and the numeric control (i.e. slider, dial, knob) to the other input. Finally, connect the output of the multiply control to the waveform viewer. Make sure each numeric control is connected to the appropriate waveform viewer and signal generator. Run the VI and rotate/slide the controls on the front panel to observe a change in the amplitude of the signals. This is how you would amplify a signal in LabVIEW. Additionally, instead of a dial, you could simply connect a numeric constant if you wanted a specific amplification factor.
4. How would you simulate threshold recognition, like turning on a LED if a voltage is higher that X volts? Go to the block diagram and select 4 different comparison numeric controls (less than, equal to, etc.) and place them on the VI. Connect the output of your amplified signal to one input of the comparison control. Then place 4 numeric constant controls on the diagram and type a different number for each. Connect the numeric constant to the other input of the comparison control. Finally, on the front panel, place four LEDs on the diagram, and connect each one to the output of a different comparison control. This allows you to set different thresholds for signals. For example, if you select a greater-than control and connect a constant with a value of 5, the LED will light up any time the signal is greater than 5. Also, you do not need to use a LED, any sort of Boolean operation can be substituted. This is just a simple demonstration.
5. Finally, what sort of different controls are there in the LabVIEW libraries, or what are some of the ways a signal can be displayed? LabVIEW has many different indicators that can be used to display different signals. Go to the front panel and select (at least) 4 different numeric indicators, such as a thermometer or a gauge. Depending on the control, you may have to examine the properties of it to adjust the scaling so that the control can handle the proper range of values. After adjusting the indicators, simply connect the amplified signal to the input of the indicator. Make sure you connect the amplified signal wire and not the compared wire, or else you will only see a 0 or a 1. Finally, run the VI and observe the meters. You can change the values on the meters by adjusting your sliders and/or knobs.
6. ***When you complete this exercise, show the TA your results.***

Turn in your completed lab grading sheet to the TA –one grading sheet per lab team.

This concludes the introduction to LabVIEW. If you have any more questions about LabVIEW, there are many more resources available to you. LabVIEW Help is a great place to start looking for some problems you may be having. The National Instruments website also has some more reference material on LabVIEW.