CALIFORNIA STATE UNIVERSITY
LONG BEACH

Control Systems Laboratory
EE 370L
Fall 2019
v1.2

Lab #2

Intro to LabVIEW

prepared by
Hossein Jula Ph.D.
Arya Daroui

1. Introduction

LabVIEW is a graphical programming language that is common among control system engineers in industry. It has many of the common features found in other programming languages, but instead of using lines of text, it uses block diagrams and GUIs. It also comes preloaded with a suite of blocks that make it easy to analyze many different phenomena in control systems. In this lab, we will become familiar with LabVIEWs basic features and some of these preloaded blocks.

2. Procedure

2.1 Setup

1. Launch LabVIEW by searching for NI LabVIEW 2018 in the start menu
2. Click on `Create Project`
3. Choose the `Blank VI Template`

You should see three windows open:

- Front Panel
- Block Diagram
- Controls/Functions (Can also be opened from `View` in the menu bar)

The Front Panel window manages the users inputs and outputs to your project. The Block Diagram window manages the internal logic of your program. The Controls and Functions windows let you select the I/O and blocks for the Front Panel and Block Diagram windows, respectively. You will notice some of the blocks will be labeled as the following:

- *Controls*: user inputs on the front panel
- *Constants*: internal input in the block diagram that cannot be changed during runtime
- *Indicators*: user-facing outputs on the front panel

2.2 Hello world

In classic programming fashion, we will begin our venture into this new language with a simple Hello World style program. Your objective is to develop a program that inputs a string, outputs the strings length, and outputs the string in uppercase. Because this is your first LabVIEW project, we will walk through the first steps.

1. From the Controls menu, go to

   `Modern > String & Path > String Control`

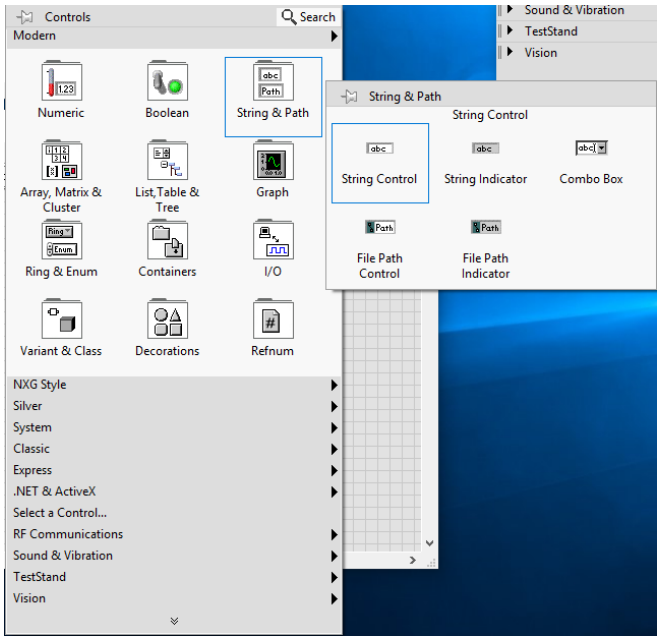

Figure 1. Selecting a String Control

   and drag it to the front panel, as shown in Figure 1. This will serve as your string input.

2. You should notice that a corresponding block has been made in the Block Diagram window. Copy-paste an additional `String Control` block. Now, right-click on the new block and select Change to `Indicator`, as shown in Figure 2. This creates a corresponding `String Indicator` in the Front Panel window.

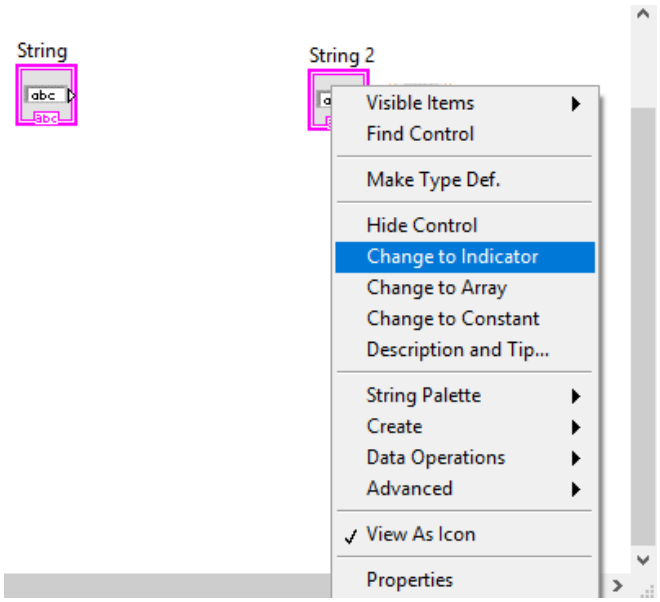

Figure 2. Changing a Control block to an Indicator

3. From the Functions window, find and place a block that

will convert a string to uppercase, and a block that will give string length. Use this as an opportunity to explore the many, many different blocks that LabVIEW comes with. Hint: You can use the search feature; the blocks will look like they do in Figure 3.

4. Hover over the `String Length` block; you should see little blue dot on the output (right) side. Right-click on the blue dot and select `Create > Indicator`. This is an alternative to manually searching for the indicator block.

5. Now, you should have everything in its place. Rename your blocks, controls, and indicators, and wire them all up. Your front panel and block diagram should look similar to Figure 3.

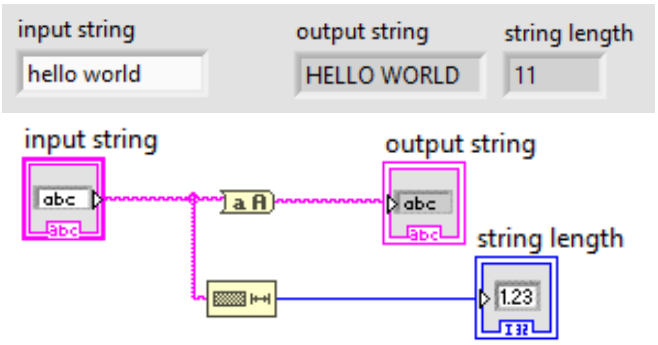

Figure 3. Front panel (above) and block diagram (below)

To test, use the sample input

`hello world`

And then click on the `run` arrow in the toolbar.

2.3 Native wave generation

Now that we know the basics, we will try to take things a step further. If you get stuck, try looking up the documentation on NIs website. Resolving problems by reading through documentation is a valuable skill.

Using a wave generator block, create and plot a sine wave with the adjustable parameters: amplitude A, frequency f, and phase ϕ. Use the sample inputs

$$A = 2,\ f = 10\ \text{Hz},\ \phi = 0°,\ \text{and}$$
$$A = -1,\ f = 5\ \text{Hz},\ \phi = 180°.$$

From the `Functions` window, you can find a wave generator block in

`Signal Processing > Wfm Generation`,

and from the `Controls` Window, you can find the graph indicator in

`Modern > Graph > Waveform Graph`.

Place controls on the front panel to adjust the given wave parameters.

2.4 MathScript

If your internal logic is ever too complicated for a block diagram, you may want to use a MathScript block to program textually; its syntax is similar to MATLAB. You can find the MathScript block from the Functions window in

`Programming > Structures > MathScript`.

Repeat 2.3. using a MathScript block instead of the native wave generation block, but do it two different ways:

1. Using front panel controls and graph indicator to input and output out of the MathScript block.

2. Manually defining the inputs inside the MathScript block and using the MathScript `plot()` function.

2.5 Transfer functions

For the following transfer functions, generate a step response plot, bode plot, and pole-zero plot. Use the sample functions of

$$G_1(s) = \frac{s+1}{s^2+s+1}$$
$$G_2(s) = \frac{s+2}{2s^2+8s+4}$$
$$G_3(s) = \frac{3s^2+2s+1}{4s^3+3s^2+2s+1}$$

From the Functions window, you can find a transfer function block in

`Control & Simulation > Control Design > Model Construction > CD Construct Transfer Function Model`,

and you should look for the necessary plotting blocks in

`Control & Simulation > Control Design`.

Right-click on the transfer function block to create the numerator and denominator controls. Be mindful that LabVIEWs input method for the coefficients of the numerator and denominator of a transfer function are the opposite of MATLABs. That is, in MATLAB, your coefficients for would be in descending order:

```
num = [3 2 1];
den = [4 3 2 1];
```

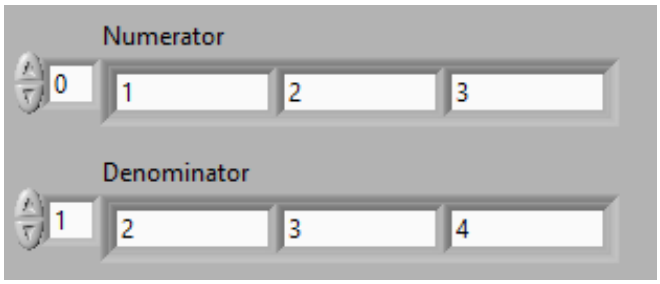

Figure 4. Inputting the coefficients of $G_3(s)$

However, in LabVIEW, you will input in ascending order and use the spin box (arrowed input box on left) to scroll through the coefficients, like in Figure 4 below.

Finally, obtain a table of signal amplitude and time from the step response plot for $G_3(s)$. You can obtain the data by right-clicking on the plot and navigating to Export. Include only 10 datapoints in the table.

2.6 If statements

Using nested `Case Structure` blocks, create a program that will output a string yep for an input x in the range

$$0 < x \leq 10$$

And will output nope when not in the range. Use sample inputs, -1, 0, 5, 10, and 11.

2.7 Loops

Using a loop block as a for loop, create a program to experimentally determine the probability of a random number being selected.

First, create a program to generate a random value $x \in \mathbf{Z}$ where $1 \leq x \leq 5$. Generate the value for n trials using the loop block. Use a `shift register` to count the number of times X, a specific value of x occurs.

For X as the number times a specific value of x occurs in n trials, the experimental probability P of that specific value of X being generated is found through

$$P = X/n$$

Find P with the following sample inputs:

$$x = 1,\ n = 10;$$
$$x = 3,\ n = 50;\ \text{and}$$
$$x = 5,\ n = 1000.$$

Lastly, repeat the above but convert the for loop into a while loop.

3. Report deliverables

For sections 2.2. through 2.7. include:

- Screenshot of the block diagram
- Screenshot of the front panel with sample inputs and corresponding outputs

And for 2.5., include the table.

4. Appendix

4.1 Definitions

Table 1. Definitions

Abbreviation	Explanation
GUI	Graphical user interface
I/O	input-output; input and output
NI	Native Instruments, developer of LabVIEW

4.2 Wire colors

The different wire colors represent different datatypes. You can find a list of the colors and their datatypes at `knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z0000019LsVSAU&l`