How To: Ladder Logic via MATLAB

SD Team 519

I. Introduction

A. What is a PLC?

A Programmable Logic Controller (PLC) is an industrial computer that is designed for automation processes, minimizing manual labor and increasing efficiency. These digital devices are used to monitor inputs, make programmed decisions through software applications, then control outputs. PLCs are rugged, making them widely used in large facilities, because they can withstand extremely harsh conditions. Additionally, they speed up processes by constantly scanning inputs, processing logic, and generating outputs within milliseconds.

B. Why are PLCs important?

Programmable Logic Controllers (PLCs) play a vital role in industrial automation, yet the FAMU-FSU College of Engineering does not cover their applications. With the rise of automation and smart manufacturing, there is a growing demand for engineers with exposure to PLCs. They are widely used in transportation, robotics, and manufacturing. Integrating PLCs into the curriculum will equip students with Ladder Logic programming skills, thus enhancing their technical abilities and broadening their interest in industrial automation. This curriculum design ensures students will obtain hands-on experiences while simulating real world applications and preparing them for technological advancements, overall making them more competitive in the job market.

C. Purpose

The Introduction to Mechatronics course at FAMU-FSU College of Engineering plans to incorporate PLCs by adding three labs for students to complete by the end of the semester. The labs will cover I/O control, sensor-actuator integration, and Ladder Logic programming – the most common language. This tutorial will allow students to incorporate this new concept of Ladder Logic into MATLAB, simulating manufacturing processes digitally and fully grasping the logic behind making PLCs operate successfully.

II. Nomenclature, Symbols, and Terms

Input	Receives signals from devices such as sensors and switches; placed on the left side of rungs
Output	Controls devices such as actuators, motors, lights, and solenoids; placed on the right side of rungs
Output Energize (OTE)	Output is on when the rung conditions are true and off when rung conditions are false
Output Latch (OTL)	Latches an output, turning it on when the rung conditions are true and keeps it on if the rung becomes false
Output Unlatch (OTU)	Unlatches an output, turning it off when the rung conditions are true
Rung	Each line on the ladder logic that holds a minimum of one output
Examine If Closed (XIC)	A Boolean variable where true conditions hold a value of 1 and false conditions hold a value of 0
Examine If Open (XIO)	A Boolean variable where true conditions hold a value of 0 and false conditions hold a value of 1
Normally Closed (NC)	Allows the current only when the input condition is false Example: When the input is off, the contact closes. When the input is on, the contact opens, breaking the circuit.
Normally Open (NO)	Allows the current only when the input condition is true Example: When the input is on, the contact closes. When the input is off, the contact remains open.

Table 1: Definitions

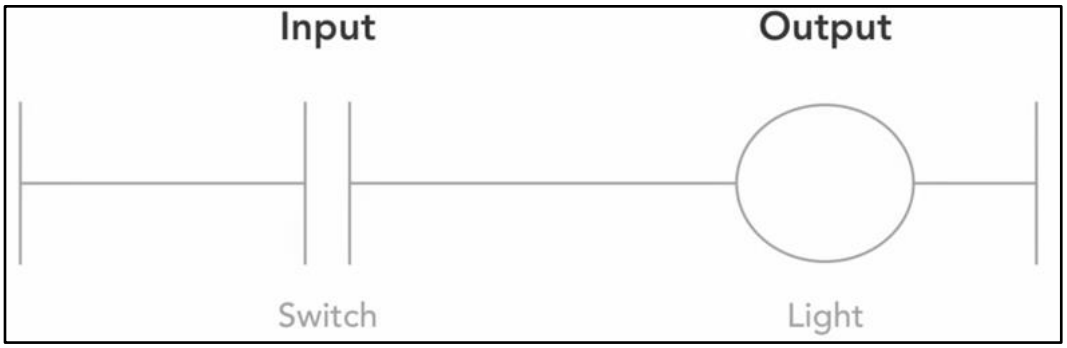

Figure 1: Input and Output Symbols

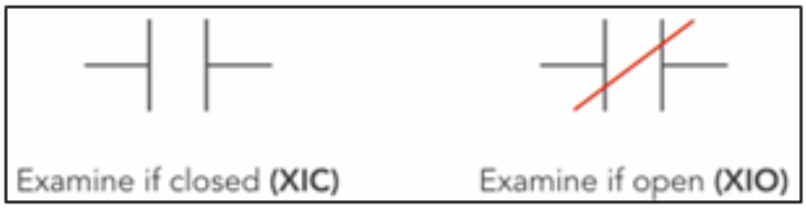

Figure 2: Types of Inputs

Figure 3: Types of Relays

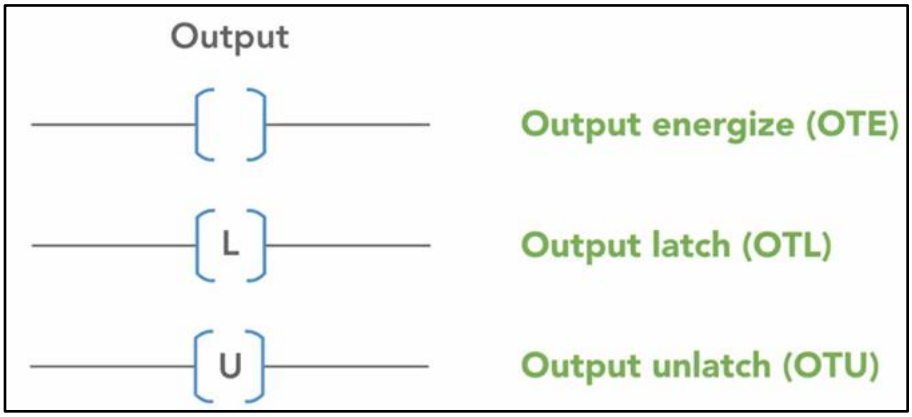

Figure 4: Types of Outputs

III. Launching the Ladder Logic Simulator via MATLAB Simulink

1. How to open the simulation
    a. Open MATLAB
    b. Download the library: *Simulink PLC Coder*
    c. Make sure to download the *edit_simulink_plc* and *run_simulink_plc* files from Github (This will be provided for students!)
        - *edit_simulink_plc*: This function is useful for editing the Ladder Logic diagram
        - *run_simulink_plc*: This function is useful for running the simulation to extract data
    d. Run the Demo file with this line of code: *edit_simulink_plc("Demo");*
        - Notice that the Demo file will not open until a file is generated
    e. The **Demo** and **library** window will pop up
        - The library window is named "Library.studio5000_plclib"
        - There will be many icons in the library which signify the preset symbols used to create ladder logic diagrams
2. Creating the PLC simulation
    a. An icon you will be using in every ladder logic diagram is the "PLC Controller Suite". This simulates the actual PLC device.

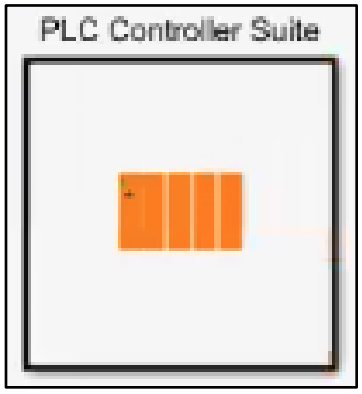

Figure 5: PLC Controller Suite Icon

- Other icons you will be using are the inputs (i.e., XIC, XIO) and the outputs (i.e., OTE, OTL, OTU)

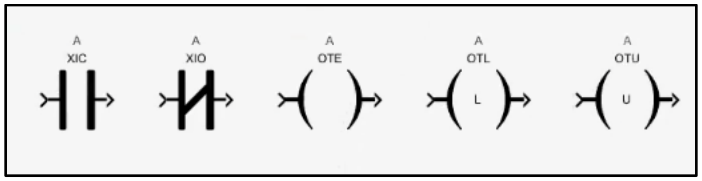

Figure 6: Input and Output Icons

b. Start your ladder logic diagram! Click and drag the “PLC Controller Suite” into the Demo Simulink file
    - It may take a second to load
c. Double click on the “PLC Controller Suite”. You should view this “Logic” window
d. Double click on the blue square labeled “Task”
e. Double click on the orange square labeled “Program”
    - This will open the ladder logic workspace for the PLC we have just created!

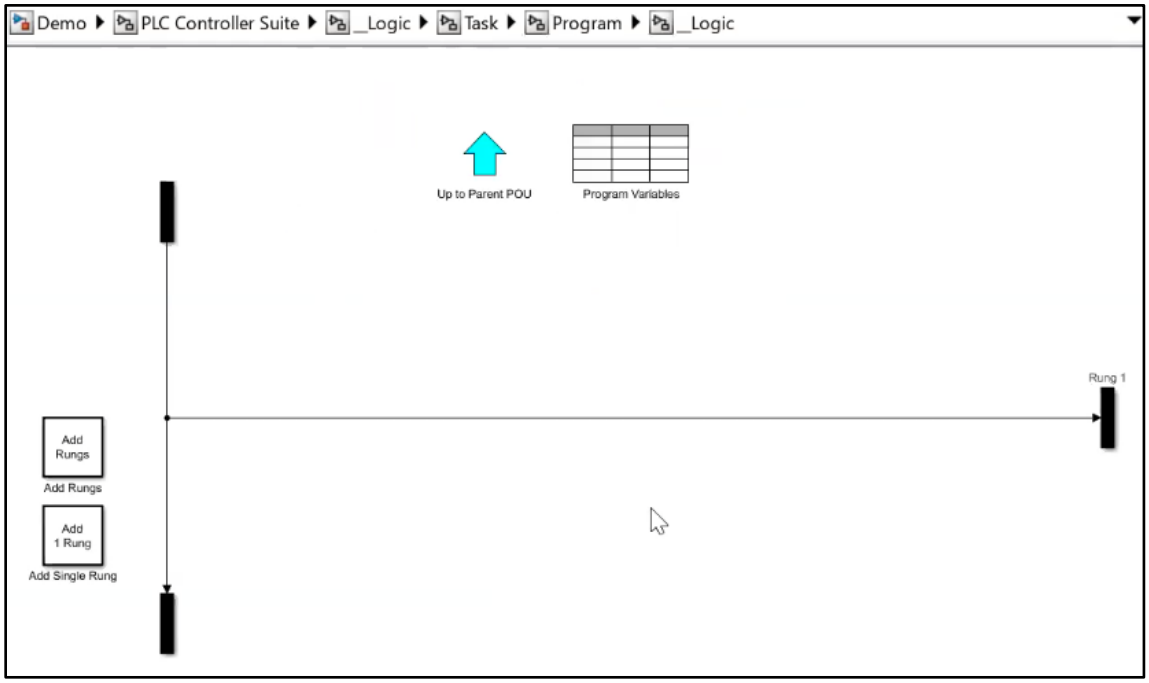

Figure 7: Ladder Logic Workspace

3. Adding custom inputs to the PLC
    a. Click and drag a desired input (XIC, XIO) from the library (Studio5000) window onto the left side of the rung (called Rung 1)
    b. Double click on the input symbol. A popup window will ask to name the “Operand Tag”. This is the variable name for the symbol you are using
        - Example: “Button1”
4. Adding custom outputs to the PLC
    a. Click and drag a desired output (OTE, OTL, OTU) and drag to the right side of the same rung
    b. Double click on the output symbol and name the variable
        - Example: “LED1”

5. Adding rungs
    a. Method 1: Click the “Add Rungs” box located to the left and type the desired number
    b. Method 2: Click the “Add 1 Rung” box located on the left
6. Adding multiple inputs and outputs
    a. Add a second rung. It will be called “Rung 2”
    b. Add another input onto the second rung underneath the first one and name it
        - Example: “Button2”
    c. To create an OR statement, where both XIC inputs are in parallel, click on the rung after the new input on the bottom and delete it
    d. Click on “Junction” in the library, drag it over to Rung 1 where the original input and output are. Make sure to place the junction after input 1 and before output 1
        - The junction will merge 2 inputs and connect to 1 output

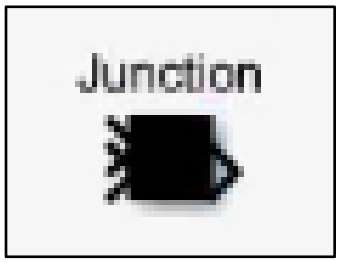

Figure 8: Junction Symbol

    e. Click the outgoing arrow from “Button1” input and drag it to the left side of the junction
    f. Click on the outgoing arrow from “Button2” and drag it to the left side of the junction. This will create a parallel circuit with two inputs, which is an OR statement
        - Assuming the inputs are XIC, if Button1 OR Button2 is “1”, then LED1 will turn on
        - You can delete the Rung 2 black rectangle since there is no longer a second rung

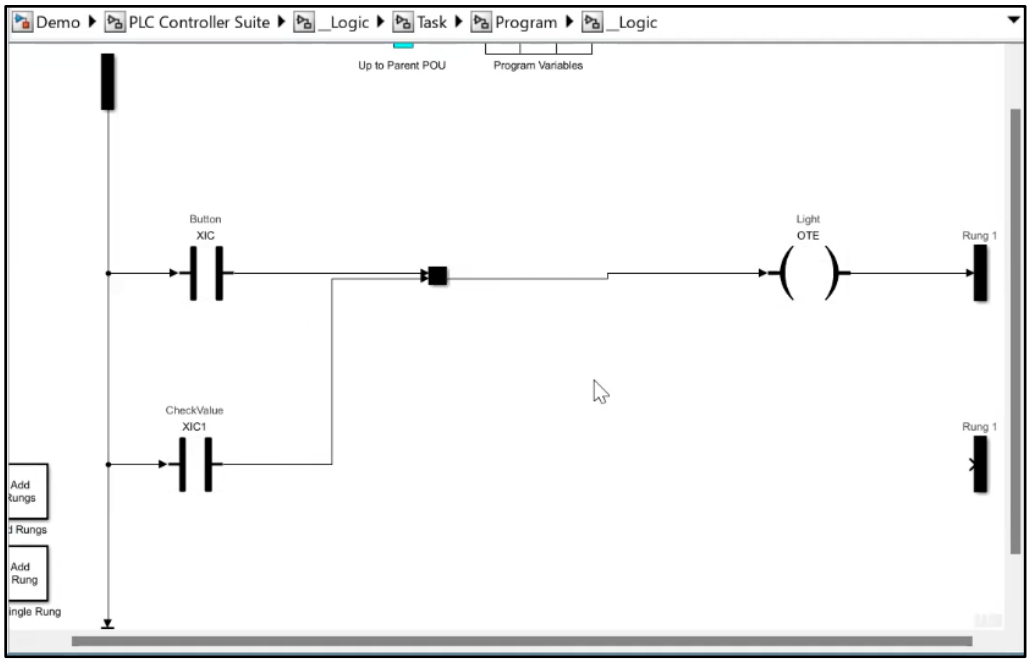

Figure 9: Completed OR Statement via Ladder Logic

g. Once your ladder logic diagram is done, double click on “Program Variables”
h. The popup window for Program Variables will show up on the screen
    - The Name tab signifies the variable names that have been created
    - The Scope tab contains local and external items
    - Local: I/O inside of the PLC
    - External: requires a physical I/O (i.e., Button1, LED1)
    - The Data Type tab will always be BOOL for ladder logic. If it is not, make sure to change it
    - The Initial Value must be changed from false to true (assuming XIC inputs are being used)
i. Hit “Apply” then “OK” to confirm
j. Click “Up to Parent POU” arrow twice until you’re back the main screen where the you will see “Controller Tags” and “Task”
k. Double click on the green “Controller Tags” table. This is where you can view the external variables that have just been created in the Program Variables window. An additional window will pop up named “Block Parameters: Controller Tags”.

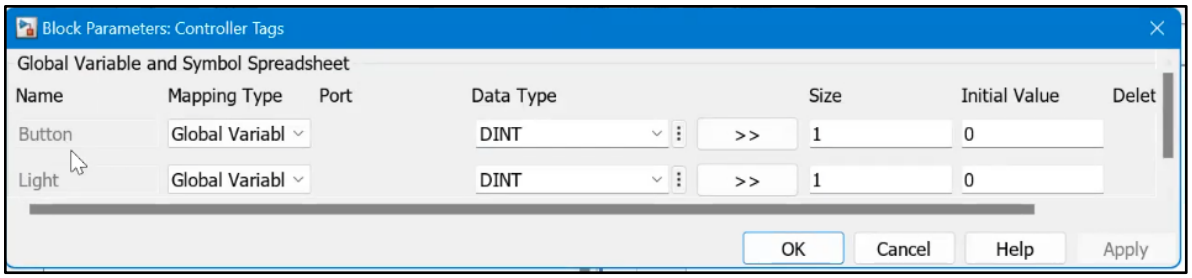

Figure 10: Controller Tags Window

- The Name tab will list the external variables
- The Mapping Type will signify what to define your variables
- Button would be an input and LED would be output
- The Data Type will always be Boolean

l. Hit “Apply” then “OK”

m. Click “Up to Parent POU” once to return to the main Demo workspace. You will see the PLC will have the desired input and output that has just been created

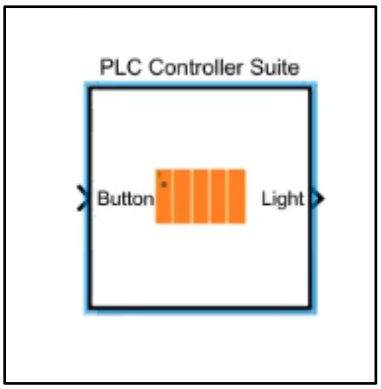

Figure 11: Simulated PLC

n. Once you are done with the Ladder Logic diagram, you can minimize the window and full screen the Demo window

7. Running the PLC Simulation
   a. Under the ‘Simulation’ tab at the top, look for the “Library Browser” icon. This will allow for the user to control the PLC

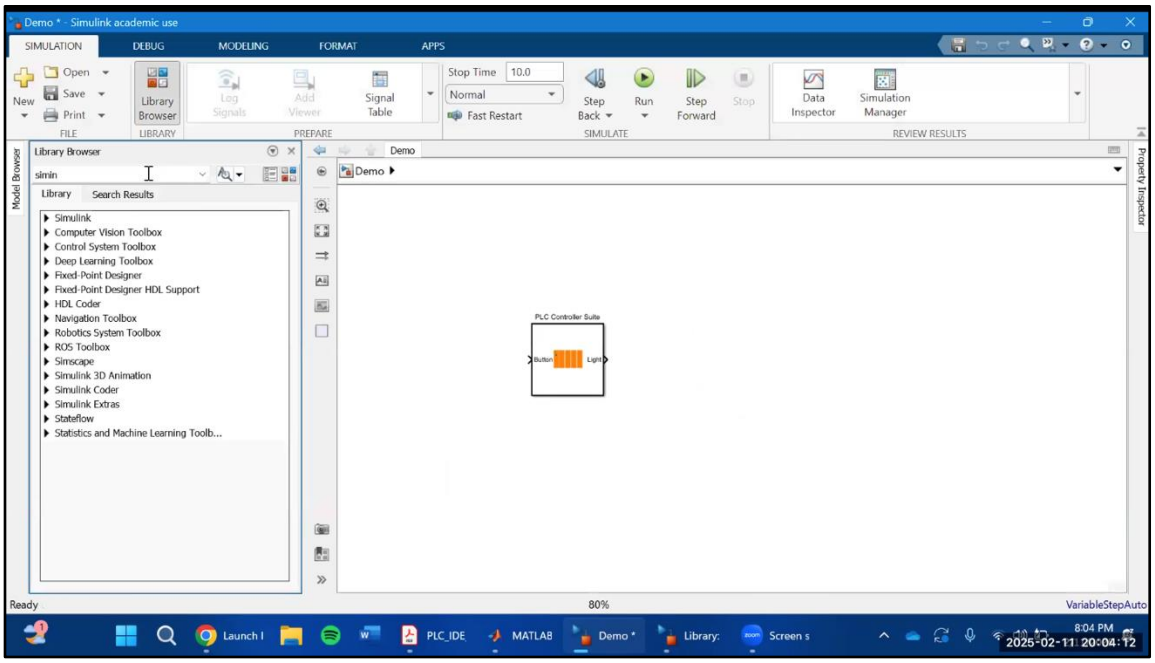

Figure 12: Library Browser and Demo Screen

b. Under the Library Browser window that pops up on the left, type in "Constant" in the search bar. This feeds a constant value into the input, in this case, a button.
c. Click and drag the icon and drag it to the left side of the PLC. Connect the arrow so that it feeds into the button.
d. Double click on the constant symbol. A popup window will show up. Under the "Main" tab, the constant value is assigned to "1", but change if needed.
e. Under the "Signal Attributes" tab, make sure to change the output data type to "Boolean" or "Inherit via Back Propagation". Refer to the troubleshooting section below named "PLC Simulation" for more details to ensure no errors occur.
f. Hit "Apply" then "OK"

8. Visualizing results of the PLC Simulation
   a. Go back to the Library Browser. Type in "Scope" in the search bar. Click and drag the icon to the right side of the PLC. Connect it to the PLC.

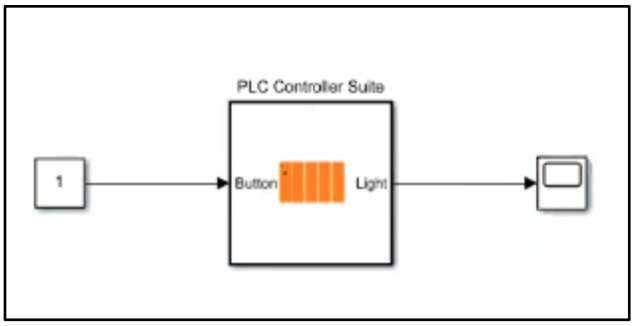

Figure 13: Constant, PLC, and Scope in Demo Workspace

b. You can close out of the library browser tab if desired.
c. Look for the "Run" green circle button under the Simulation tab in MATLAB. Once you hit the drop down, click "Simulation Pacing"
d. Hit the check box next to "Enable pacing to slow down simulation". If you don't check this off, the simulation will run as fast as possible, hindering any data to be collected.
    - You can adjust the Stop Time found at the top next to the Run button. This stop time reflects the duration of the simulation.
e. Double click on the scope icon to view the graph.
f. Hit "Run" on either the scope pop up or the demo window. You have successfully simulated a PLC via Ladder Logic! ☺

IV. Troubleshooting/Debugging/FAQ's

- Important notes
    - Note that Ladder Logic is read left to right and top to bottom
    - To exit out of the Ladder Logic diagram or return back to previous pages, hit the bright blue "Up to Parent POU" arrow. This serves as the "back" button.
- Adding the first Rung
    - If there is no rung initially, click and drag an input into the workspace. Click on the arrow at the left of the input symbol and drag onto the desired line. Repeat on the arrow at the right to complete the rung.
- PLC Simulation
    - An error will pop up if the value for the input (i.e., Button) does not match the Constant value that sends through the PLC. To avoid this, read the steps below:

        - After adding the “Constant” and double clicking it for the “Block Parameters: Constant” window to pop up, go to the “Signal Attributes” tab. There are two methods to avoid errors.
            - Method 1: Change the Output Data Type to Boolean to match the Boolean input
            - Method 2: Change the Output Data Type to “Inherit via back propagation”. This will force the constant to match the input.
        - If there are issues in the scope window (which graphically illustrates the PLC), make sure to adjust the inputs from false to true depending on what type of input is utilized. Keeping it to the default (false) will allow for it to act “OR 0”, even if the constant is “1”.
    - Debugging Tips
        - If you want to see the Ladder Logic being ran in real time with the simulation, go back into the Ladder Logic diagram. Run the simulation.
        - As the simulation is running, the parts turn green. This is helpful to animate what is happening as it runs
            - The symbol will always show as green, but pay attention to if the rung/arrow/line after it is green. If it is, then it shows that it is running through that portion of code!

V. Additional Resources
    - Code from Github: https://github.com/jjv432/SeniorDesign/tree/Demo
    - Setting up Arduino Opta PLC: https://docs.arduino.cc/software/plc-ide/tutorials/plc-ide-setup-license/
    - https://github.com/arduino/ArduinoCore-mbed/blob/main/libraries/STM32H747_System/examples/QSPIFormat/QSPIFormat.ino