

Lab Week 11: Programmable Logic Controller (PLC)

PLC access:

Access the PLC simulator by navigating to: plcsimulator.net

Make a free account using your purdue email address. The PLC simulator uses Adobe Flash, so it may not work on some internet browsers (e.g. Google Chrome). Try using Firefox or Internet Explorer.

Deliverable:

Complete Section 3 and answer all questions.

0. Learning Objectives








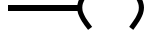
- To learn the basic principles of ladder logic and programmable logic controllers

1. PLC Overview

Programmable Logic Controllers (PLC) are robust inexpensive (few hundred dollars) standalone controllers. PLCs work by analyzing the input “state” and turning on/off the resultant “output”. Inputs are used for measuring information from sensors such as temperature, speed, position, push buttons, etc. Outputs are used for controlling things such as electric motors, linear motors, pneumatic valves, hydraulic valves, light emitting diodes (LEDs), buzzers, etc.

1.1 Terminology:

- Input Relays (contacts): connected to the outside world; physically exist as a switch, button, sensor, etc.
- Output Relays/Coil (contacts): connected to the outside world; physically exist and send on/off signals to solenoids, lights, etc.
- Counters: do not physically exist. Rather, they are simulated counters that are programmed to count pulses. Can count up, down, or up and down between actions.
- Timers: do not physically exist. There are two types of timers: on-delay type and off-delay. On-delay timers receive an input and turns an output “ON” after a defined delay. Off-delay timers receive an input and turns an output “OFF” after a defined delay. Timers can range from 1 ms to 1 s.

| Table 1: Basic Ladder Logic Components | |
|---|---|
| <i>Symbol</i> | <i>Function</i> |
|  | Normally open contact connected in series – open contact at rest |
|  | Normally closed contact connected in series – closed contact at rest |
|  | Normally open contact connected in parallel |
|  | Normally closed contact connected in parallel |
|  | Normally open contact connected in parallel followed by another element |
|  | Normally closed contact connected in parallel followed by another element |
|  | Connect coil (output) to right power rail – inactive output at rest |
|  | Connect coil (output) to right power rail in parallel |

1.2 Ladder Logic

A PLC works by continuously scanning system inputs. There are three basic steps:

1. Check input status: which inputs are on/off?
2. Execute program: execute any coded program, one instruction at a time.
3. Update output status: based on steps 1 and 2, the outputs turn on/off

The PLC continues this process until it receives the signal to end or is forced to stop by an outside stimulus. Most programs require an END or STOP function at the end of a process.

Ladder logic is a programming language based on relay logic. Each program is made up of “rungs” of logic, which results in a series of actions occurring in a predictable manner.

Simulator-specific guidelines:

1. When entering an input or output name, you must hit “enter”. Otherwise, it will not save the name and it will disappear when you run the simulation.
2. Save all programs on the server. This will ensure that you will not lose your work.
3. There is no auto-save feature; save early and often.

2. Assignment

Go to plcsimulator.net and log in with your account information. On the opening page, you will see a new plc program with an empty rung, as seen in Figure 1.

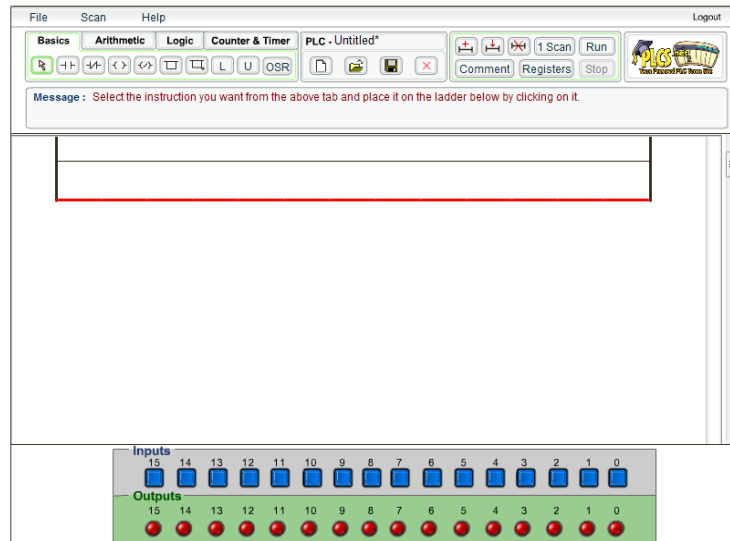


Figure 1. Blank ladder logic program.

We will first go through an example to highlight all of the necessary functions for the lab activities. Save this file as “ABE460_Lab11_prob_2_1.plc” on the server. Do this by going to File>Save>...into Server, as seen in Figure 2. Saving it into the server will allow you to access this file whenever you log into this simulator.

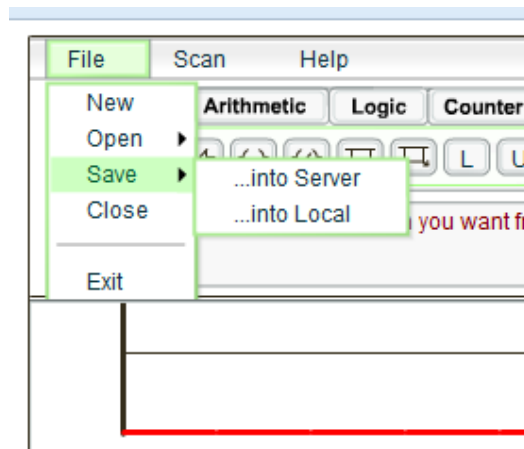


Figure 2. File save location.

2.1 Single input-output pairings

In the far right box, click to add 3 more rungs to your ladder logic program. Your inputs for this exercise will be I/1-4, and your outputs will be O/12-15.

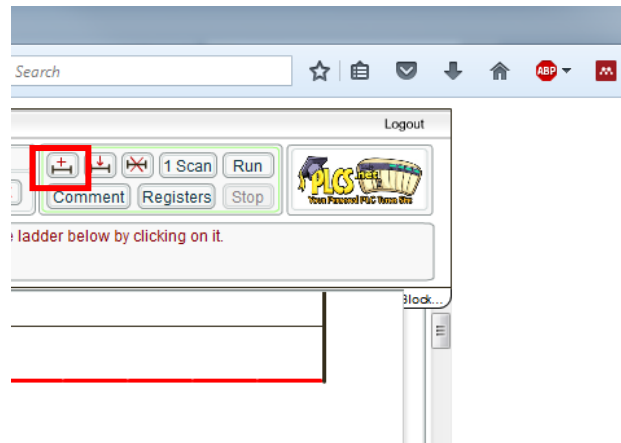


Figure 3. Ladder rung addition.

Your program should look like Figure 4.

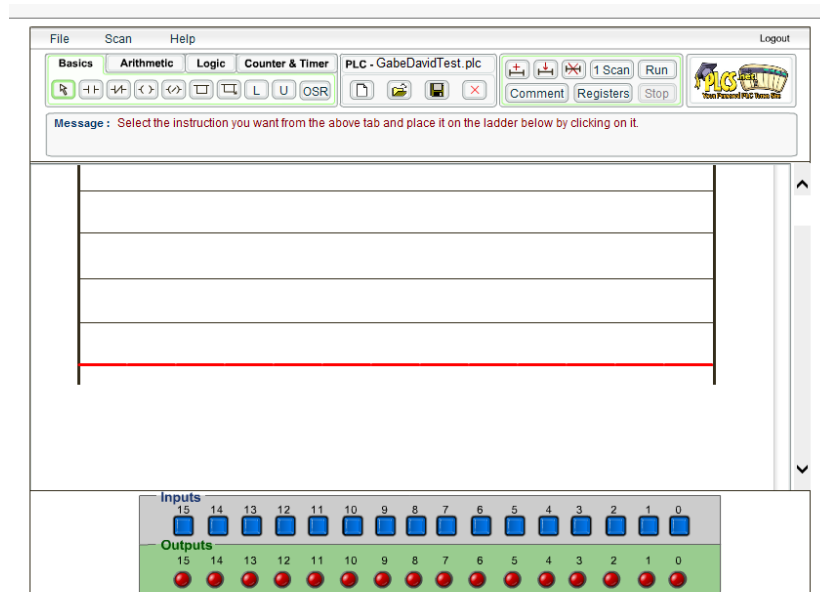


Figure 4. Program with four empty rungs.

Inputs I/1 and I/2 are defined as normally open, inputs I/3 and I/4 are defined as normally closed. Outputs O/13 and O/15 are defined as normally open, while O/12 and O/14 are normally closed.

On each rung, pair the following inputs and outputs:

1. I/1 and O/15
2. I/2 and O/14
3. I/3 and O/13
4. I/4 and O/12

Your program should look like Figure 5.

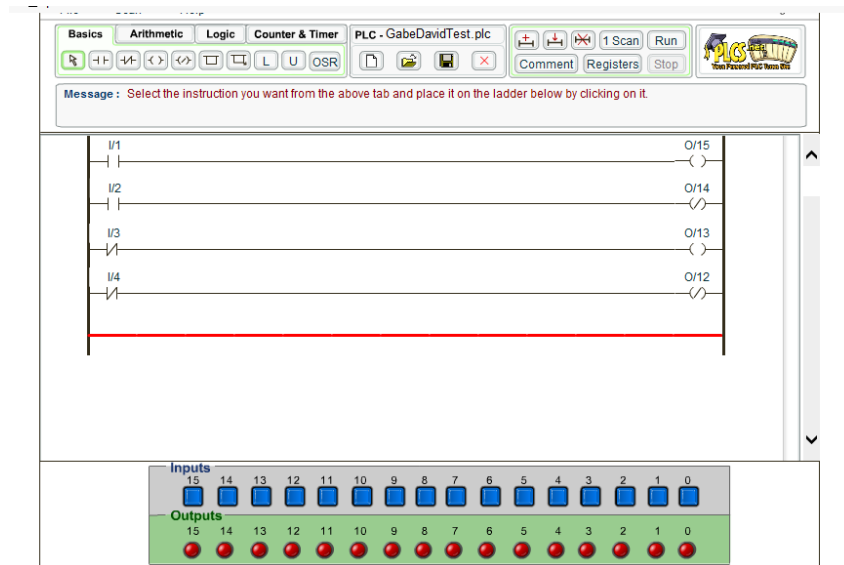


Figure 5. Completed program for section 3.1.

Click "Run" in the top right corner of the screen. **In your report, provide an image of your completed program.**

2.2 Relay Outputs

In this exercise, you will have one input and four outputs. Using the exercise from 2.1, change the program to look like the Figure 6.

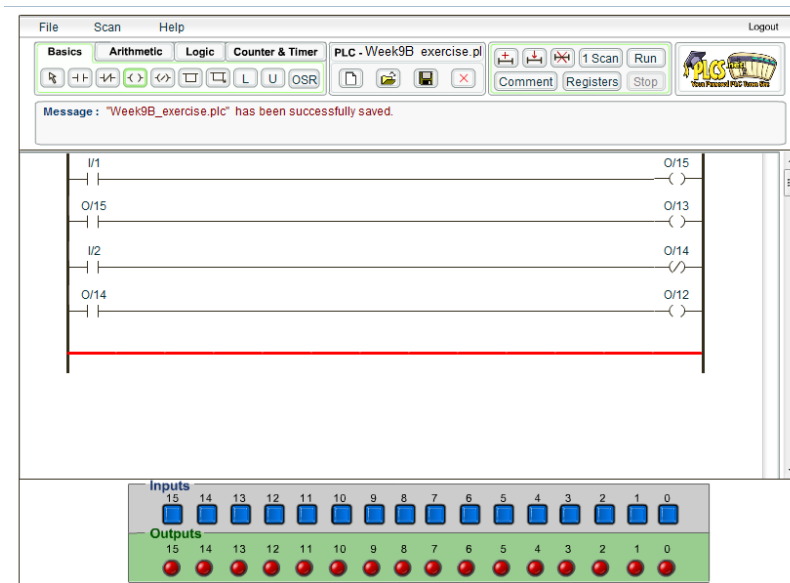


Figure 6. Ladder logic program for exercise 2.2.

Click “Run” in the top right corner of the screen. **In your report, describe what happens when inputs 1 and 2 are activated.**

2.3 Latching/Unlatching Circuits

In this exercise, you will explore how to latch and unlatch a system. Your inputs and outputs are defined as follows:

1. I/1: Start Button
2. I/2: Stop Button
3. O/4: Motor

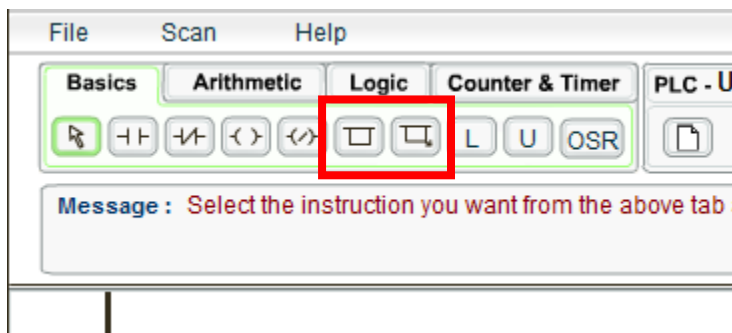
In this scenario, you want to set up a system where a motor turns on and stays on after the “start” button is pressed at least once, but turns off once the “stop” button is pressed.

Before you begin, fill out the following chart (be sure to include this chart in your report).

| Input/Output | Is it normally open or normally closed? |
|--------------|---|
| I/1 | |
| I/2 | |
| O/4 | |

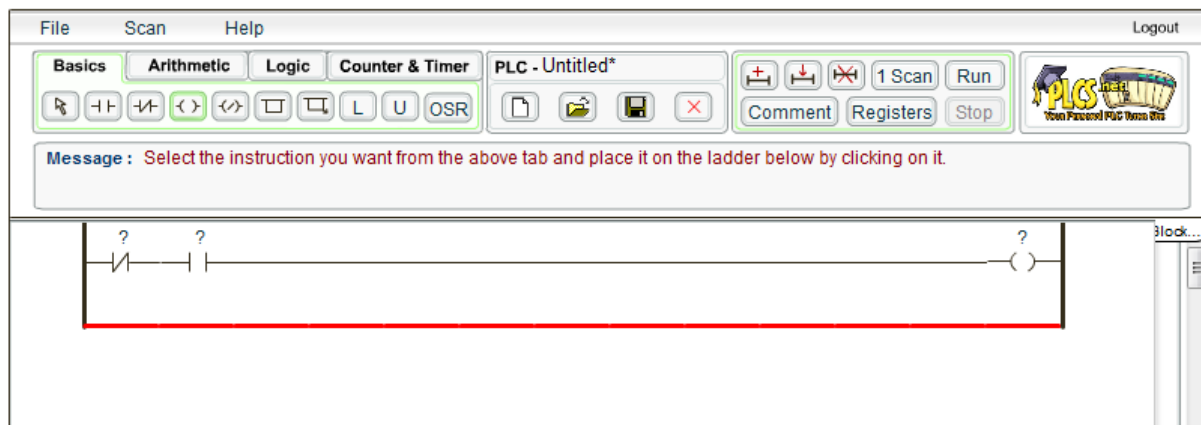
2.3.1 Parallel Latching

Creating parallel rung is a quick way to create a latching circuit. In the “basics” tab of the plc simulator, you will see two branching options:

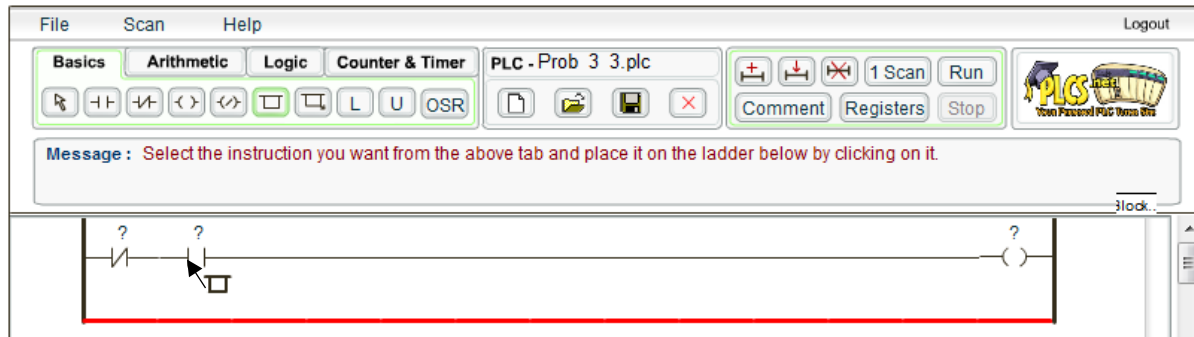


The button on the left is used to create a branch that spans a single operation, while the option on the right creates an extended branch. For our purposes, we will only use the branching option on the left.

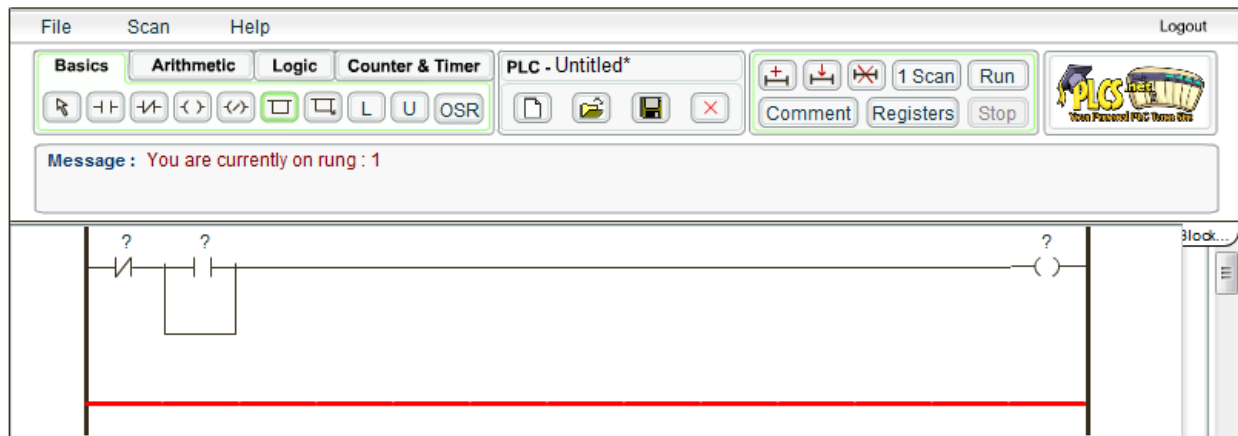
To create a latching system using a parallel branch, first make the following rung in your system, with the appropriate inputs and outputs filled in.



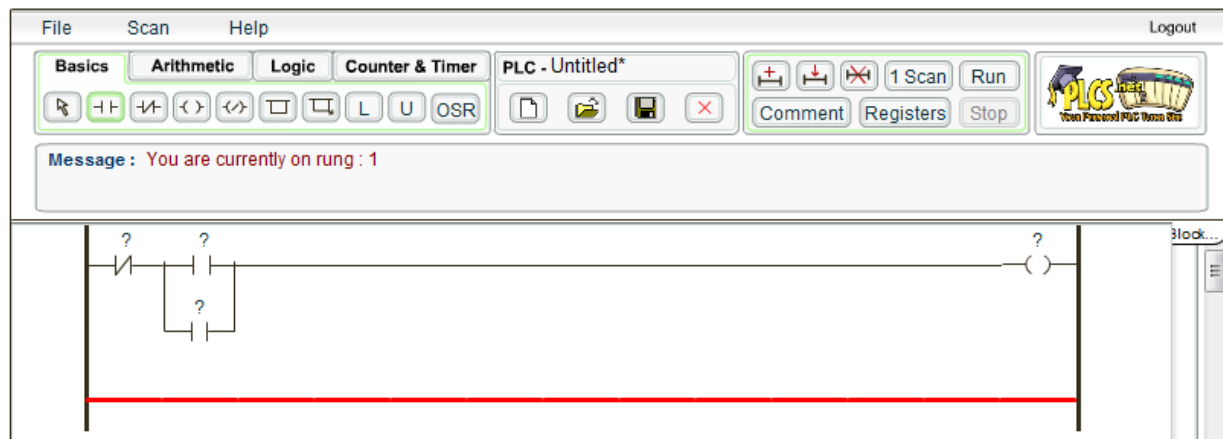
Next, add a branch that runs parallel to your normally-open input. This is done by clicking on the “branch” option, and then clicking on the left part of the normally-open input.



You will have an image that looks like this:



Finally, add a normally-open input onto the branch, resulting in this final set-up:



Hint: The input on the branch should match the normally-open output already placed on the original rung.

3. Deliverables

In your report, provide the following:

1. Screen shot of your final ladder-logic setups for sections 2.1, 2.2, and 2.3.
2. A description of what happens when the various inputs are activated or deactivated.

Answer these questions:

1. For section 2.3, why does the latching circuit “latch”?
2. For section 2.3, what are the advantages of using a latching circuit over a simple on/off switch? (Hint: what would happen after a power failure of a simple on/off switch, like a light switch, was used to power a large sawmill motor?)

No executive summary is required.