ET 401 – Lab 2 – Introduction to ladder-logic programming

MAKE and BREAK commands

In this second lab we shall start to become familiar with ladder-logic programming, which is the most-used programming language for Programmable Logic Controllers. Ladder-logic is a graphical programming language. The basic elements of ladder logic are the MAKE (-||-) and BREAK (-|/-) functions.

If you look at the front cover of the course's textbook, the diagram on the top is a ladder-logic diagram for turning on a motor. Ladder-logic programs can be used for all sorts of automated tasks, where the steps to start something or to run something automatically involve a sequence on ON/OFF steps. In the diagram on the front of the book

the pushbuttons on the left turn the motor and lights on on the right. The logic and sequence followed by starting the motor and turning on the lights is encapsulated in the ladder-logic program, which you right.

Thus the diagram depicts input hardware on the left and output hardware on the right. The ladder-logic can be programmed or changed to do whatever you want the system to do.

The ladder diagram has two sides. The left is reserved for inputs; the right is for outputs.

In the ladder-logic diagram above, notice that there are three different symbols that capture certain activity of
ET 401 – Lab 2 – Introduction to ladder-logic programming

the logic

1. **MAKE command (- || -)**. It’s the symbol with two vertical lines that don’t touch each other, and which looks like the electrical symbol for a capacitor. The MAKE command is an input command.

2. **BREAK command (- | / -)**. It is like the MAKE command but has in addition a slash through it connecting the two vertical lines. BREAK is also an input command.

3. The coil (- ( ) -) is the symbol on the right of the ladder. It represents an output. When the coil is on, anything that refers to it—the motor or lights in this case—turns on.

Much can be done with just these three elements. Understanding them is crucial to understanding more sophisticated ladder-logic programs.

Stepping back and looking the the overall diagram, the input hardware buttons on the left turn on the devices on the right. The logic determines what turns on and what turns off. The logic can be changed to get different behavior, if desired.

The MAKEs, BREAKs, and the coils have addresses or tags that attach them to the hardware. For example the first MAKE command in the ladder is connected with the start button STRT. The coils are logically connected to the hardware output devices that they refer too. Thus is the GRN coil (- ( ) -) is on, the GRN light turns on.

The ladder is made up of rungs. The rungs are pathways that allow logic to flow from the left to the right on any rung. If logic flows through the input devices referred to on the left-hand side of a rung, then the logic reaches the coil and turns it on.

To get some practice with this, let’s start off using a PLC simulator called PLC Fiddle (www.plcfiddle.com). This is a web-based PLC simulator. Go to the PLC Fiddle website to start with. Once there, create a login ID which will allow you to do some things that you can’t do unless you are logged in. In later labs we shall use real hardware, but it helps to use a simulator too, because you may not always have access to real hardware. You can try things out on the simulator and then implement them on the real hardware once you have the simulation worked out.

*In the following exercises, use the same variable names given in your ladder-logic program. Do not change these variable names!*

Other things to pay attention to in the lab:

1. Do not do multiple exercises in the same ladder. Use different ladders for different exercises.

2. Once you create a login/password on the PLC Fiddle website, you can save your work. Here’s how: You go to the “Other” menu, then click Save. This will create a unique URL (web page) for your ladder. Copy this ladder into an Excel worksheet and list in a cell beside it what is in that ladder. Then you will be able to retrieve your simulation at any time.

3. In the lab we may want you to pull up one of the previous exercises, so make sure you do save all your exercises by the procedure mentioned in 2.

**Exercise 1 – 4 MAKES**

Set up the PLC Fiddle simulator with four inputs—I1, I2, I3, I4—and four outputs—O1, O2, O3, O4. Put these in four rungs in a ladder diagram, all alike with one MAKE and one coil on each rung. Try this out and see that it works. Your simulation should look like the figure below.
Run the simulation to ensure that it works as you would expect it to. Get an instructor to sign off this exercise.

**Exercise 2 – 4 BREAKs**

Create a very similar simulation but use BREAKs instead of MAKEs. How does this program operate compared with that in Exercise 1? Demonstrate the operation of this to a lab instructor for sign-off.

You can actually have more than one input on a rung in series. You can also create a parallel rung structure using the third element under Other, shown below.

**Exercise 3 – Series & parallel structures**

Create a program with I1 and I2—both MAKEs—aligned in series on the same rung. Have the rung output be O1. Then create a parallel rung structure with I1 in one branch and I2 in the second branch. Have this rung's output be O2. Play with the input values of I1 and I2 and see how this affects the values of O1 and O2. Explain this to a lab instructor for sign-off.

The ladder diagram on the front of the book represents a very basic piece of logic: a latching program for turning a piece of equipment on. Probably the ON button for a motor or other piece of equipment is momentary. i.e. you push it, it makes a contact, but when you quit pushing it, the contact is unmade. We do not want to have a momentary start button that requires the user to sit there and continue to push it to keep the motor on. The logic on the first rung in this ladder diagram enables the employment of a momentary input to produce a persistent effect—i.e. the motor stays on, even when you remove your finger from the start button. This is called latching. Notice that this requires using an output also as an input. On rungs 2 and 3,
MTR is used as an input. Any output in a ladder diagram can also be used an input to a rung too. Indeed, this is how the latch works. On the second parallel path of the first rung, once MTR is on, the MAKE is made. STRT can be unmade (by removing your finger from the STRT button), but MTR stays on because the MAKE on the parallel branch that refers to MTR is made. To turn the motor off, you have to break the logic flow with the STP button. This turns MTR off, which also unmakes the MAKE command in the parallel branch of the first rung (the MAKE referencing MTR).

Exercise 4 – Motor starting circuit

Create the motor starting logic found in the diagram on the front of the book. You have to make a change, because PLC Fiddle does not accommodate normally closed pushbuttons. PLC Fiddle assumes that all inputs representing pushbuttons represent normally open (NO) push buttons. So for this program, use a BREAK command for the STP contact in the first rung. This would represent a NO pushbutton for STP.

Once you've created the logic diagram, test it out. With the motor off, the red light should be illuminated. Once you start the motor, you should be able to turn STRT off (remove your finger from the STRT button), and the motor remains on. When you turn STP on, this is the equivalent of pushing a NO STP button. The motor should turn off and stay off, even when you turn STP off (equivalent to removing your finger from the STP button).

Demonstrate the operation of this to a lab instructor for sign-off.

Exercise 5 – NC button

Although PLC Fiddle assumes all input pushbuttons are NO, you can actually fake a NC pushbutton. Let's let STP be a NC pushbutton. And once it is pushed, it must be reset before the contact is closed again. You can fake this behavior by adding a rung before the first rung. On this rung have a BREAK referencing STP. As this rung's output, have a coil that refers to a new variable called NC. With this, as long as STP is off, NC will be on.

Now replace the BREAK command you used for STP with a MAKE that refers to NC. Test the operation of the latching circuit. With STP off, the motor should turn on with STRT and stay on when STRT is turned off. To turn the motor off, push the STP button. With the motor off and STP still activated, try to STRT the motor. It should not turn on. You should not be able to turn it on until you reset STP, i.e. set it back off again.

Demonstrate the operation of this to a lab instructor for sign-off.

Exercise 6 - Some motor-starting scenarios

Try the following scenarios out on the simulator and describe how they differ from the standard motor-starting ladder on the front of the book. If they are better, describe why. If they are worse or have some built-in flaw, report that too. Have a lab instructor sign-off your explanations.
Exercise 7 – NOR, NAND, XOR, NXOR

Program these four functions into a ladder. Check them that they work as expected.

Exercise 8 – Two rungs with same output

Put together a ladder with one input, I1, and one output, O1. The ladder should have two rungs, both with O1 as output. The top rung has as input a MAKE with I1 as input. The bottom rung has a BREAK with I1 as input. How should this ladder behave? Does it behave as expected? Get instructor to sign off your observations and simulation.

Exercise 9 – Output as input

Create a ladder diagram with one rung with a BREAK as input and a single output O1. The break should also be tagged O1. What does this program do when started?