OBJECTIVE:
After completing this assignment, the student will be able to:

a) Identify and use the OTL and OTU instructions as latch relays.
b) Store and clear information in a PLC program with the OTL and OTU instructions.
c) Identify and apply an ONS (One Shot) instruction

REFERENCES:
Cox, Technician's Guide to Programmable Controllers, Ch 10
Allen-Bradley, Bulletin 1766 Reference Manual, Ch 4

MATERIALS NEEDED:
Allen-Bradley Micro 1400 programmable controller with simulator
RSLogix 500 and RSLinx Software

DISCUSSION:
Latching relays are used to store information. They can be found in counting circuits, relay logic systems, lighting, and alternating pump controls. Expensive mechanical versions are sold with two control coils: one to latch the contacts, the other to unlatch the contacts. Both coils must not be energized at the same time.

In the MicroLogix 1400, the mnemonic name for a latching relay is OTL (Output Latch) and the symbol is an output coil with an "L" inside -(L)-. It is also known as a SET. A Latch is a write type instruction and it writes a 1 into the data table at the address on the instruction when the rung conditions go true. When the rung conditions go false it doesn't write anything. When a bit has been latched it will stay that way, even through a power cycle of the controller, until something tells it to Unlatch. The OTU instruction is used to do this. It is also a write type instruction but it writes a 0 into the data table at the address on the instruction when the rung conditions go true. When the rung conditions go false it doesn't write anything. The mnemonic name for a Unlatching relay is OTU (Output Unlatch) and the symbol is an output coil with a "U" inside -(U)- . It is also known as a RESET; not to be confused with RES instruction in the controller.

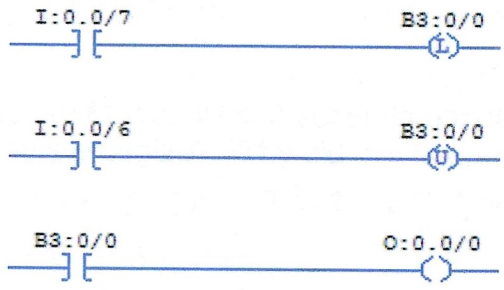
These instructions are frequently used in machine control to give them a "memory". Imagine a piece of equipment doing an indexing function where one conveyor has been stopped at a specific location and a second one is transferring something to it. If there was a power interruption and the machine was operating with held in relays when the power came back on the machine would be lost because all of the held in relays opened up. This would require the machine to have to be manually re-homed wasting a lot of time. If the same thing happened to a machine using latches it would know exactly where it was in its cycle and when an operator pressed a button it would simply pick up where it had left off. These instructions are typically used in pairs, having the same address, but this isn't always the case. When the latch instruction is used after the unlatch instruction then the circuit has latch priority. Conversely if the unlatch instruction is used after the latch instruction then the circuit has unlatch priority. This programming method defines which circuit will win in the event of a tie.

When the conditions before a One Shot instruction (ONS) go true, it sets the right side of the

instruction high for the rest of that processor scan. The next scan through, the right side of the instruction will be off even if the rung condition is still true. It will not fire again until the conditions go false and then back to true, meaning it is firing on the rising edge.

EXPERIMENTAL PROCEDURES:

In RSLogix, create a new program, named LAB06, write the program below, download it and put the controller in run the mode.



Q1. Close, then open toggle switch 7. Does the output stay on?

Yes

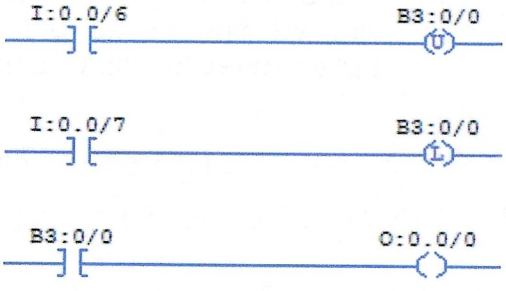
Q2. Close, then open toggle switch 6. What does the output do?

First went on when switch go off output go off

Q3. What happens when the latch output and the unlatch output are energized at the same time? What type of priority is this?

Unlatch priority

Modify your program to look like the one below, download, put it in run mode and go online:



Q4. What happens when toggle switch 6 and toggle switch 7 are both closed? What type of priority is it?

LATCH priority

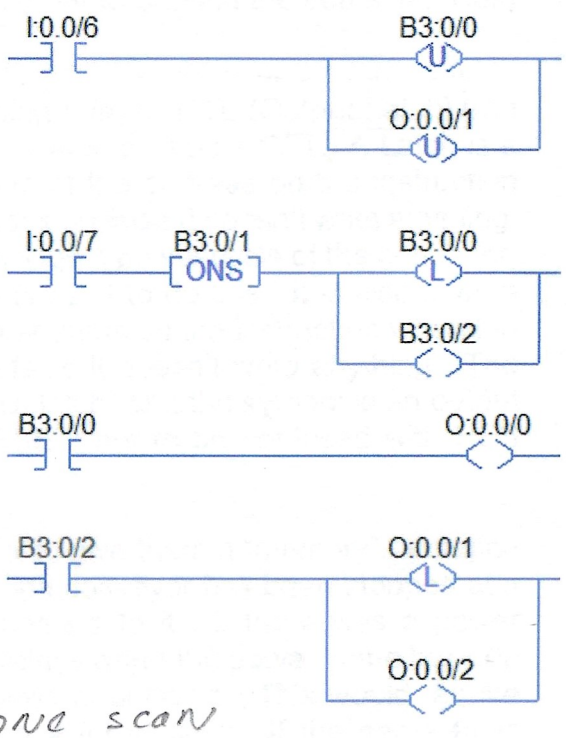
Q5. Open toggle switch 6, then open toggle switch 7. Did Output 0 remain energized?

yes remain energized

Q6. Put the controller in Program Mode. Wait 2 seconds and then put the controller back in Run Mode. Did any outputs energize? Why did this happen?

yes OUTPUT 0 energize The program knew where it was and pick up from there

Modify your program to look like the one below, download it, put it in run mode and go online:



Q7. Turn toggle switch 7 on. What addresses got energized?

I:0/7, B3:0/0, O:0/1, B3:0/0, B3:0/0, O:0/0, O:0/1

Q8. Turn toggle switch 7 off. Did any of the addresses recorded above de-energize?

I:0/7

Q9. If address B3:0/2 wasn't on how did address O:0.0/1 get energized? Close toggle switch 6 and then open it to reset the circuit and try this sequence several times and observe what happens.

The B3:0/2 got energize in one scan This is so fast that I can no see it at that time is when OUTPUT O:0.0/1 get energized