EET165 Lecture #9

1) **Review:** Answer any questions from last week.

2) **Integer Variables:** N7 is a file that can hold 16-bit integer values. An integer is a value that is a whole number only, it can't hold a fractional part of a number. These values will become more important as we move on, but for now just know that you can store integers in N7:XXX where the XXX represents which value you are working with.

3) **Float Variables:** F8 is a file that can hold float values. A float is a value that is a whole number and a fractional part of a number. We do not have access to this file on our test machines. But you should be aware of this file. The PLC stores the floats in F8:XXX where the XXX represents which value you are working with.

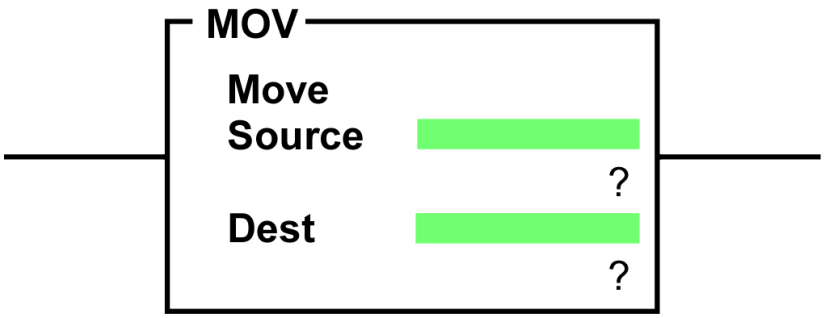
4) **Data Manipulation:** There are many different instructions that will be used to move, copy, and alter stored data. Some of these instructions are considered inputs and some are considered outputs. The outputs may be controlled by an input, or they may run autonomously.

5) **Outputs:** The following data manipulation instructions are outputs that may or may not have an input to control them.

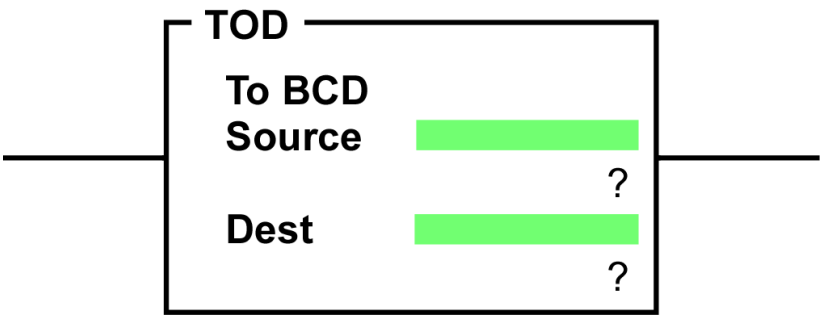
    a. **Source and Destination:** The source value on the MOV, TOD, FRD, and MVM commands can be any of the following:
        - **i.** Integer Table value (called a register) N7:
        - **ii.** Float Table value (Called a register) F8:
        - **iii.** Input card I:1.0
        - **iv.** Constant value (in base 10, base16, or BCD depending on the command)
        - **v.** An instruction value (timer accumulator, timer preset, counter accumulator, counter preset, etc.)

    The destination value on the MOV command can be any of the following:
        - **i.** Integer Table value (called a register) N7:
        - **ii.** Float Table value (Called a register) F8:
        - **iii.** Input card I:1.0
        - **iv.** An instruction value (timer accumulator, timer preset, counter accumulator, counter preset, etc.)

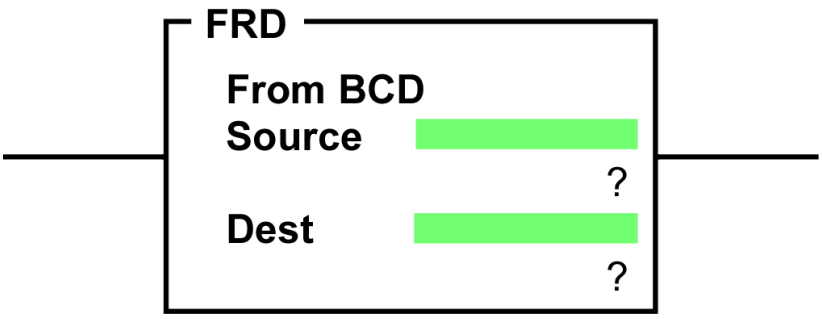
b. **MOV:** This instruction requires you to input two values, the source and the destination. It copies the data from Source to the Destination. The Source is left untouched. Under the source, the PLC displays the current source value. Under the destination, the PLC displays the current destination value. You cannot control these values, they are just displayed. As long as the rung remains true, the instruction moves the data each scan. If you only want the move to happen one time, you should put an OSR instruction in front of it.



c. **TOD:** This instruction requires you to input two values, the source and the destination. This takes a normal base 10 number from the source, it converts the value into BCD and stores it in the destination. The Source is left untouched. Under the source, the PLC displays the current source value. Under the destination, the PLC displays the current destination value. You cannot control these values, they are just displayed. As long as the rung remains true, the instruction moves the data each scan. If you only want the move to happen one time, you should put an OSR instruction in front of it.

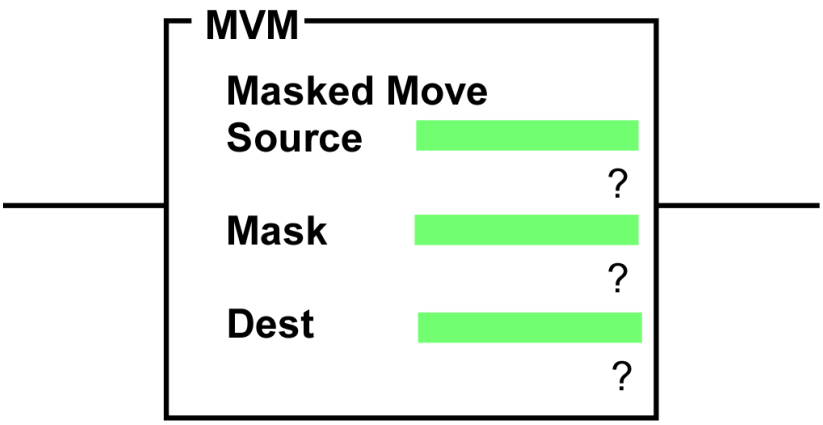


d. **FRD:** This instruction requires you to input two values, the source and the destination. This takes BCD value from the source, it converts the value into Base 10 and stores it in the destination. The Source is left untouched. Under the source, the PLC displays the current source value. Under the destination, the PLC displays the current destination value. You cannot control these values, they are just displayed. As long as the rung remains true, the instruction moves the data each scan. If you only want the move to happen one time, you should put an OSR instruction in front of it.



e. **MVM:** The MVM instruction moves data from the source to the destination and allows portions of the destination data to be blocked or passed. As long as the rung remains true, the instruction moves the data each scan. If you only want the move to happen one time, you should put an OSR instruction in front of it.

The Mask is used to determine which bits are passed and which bits are blocked when the source is moved to the destination. If the mask bit is 1, the value is passed, if the mask bit is 0, the value is blocked.



For example: Assume the source is 1F2A in hex and the mask is 0F0F. the destination is calculated like this:

1F2A (hex) = 0001 1111 0010 1010 (binary)
0F0F (hex) = 0000 1111 0000 1111 (binary)

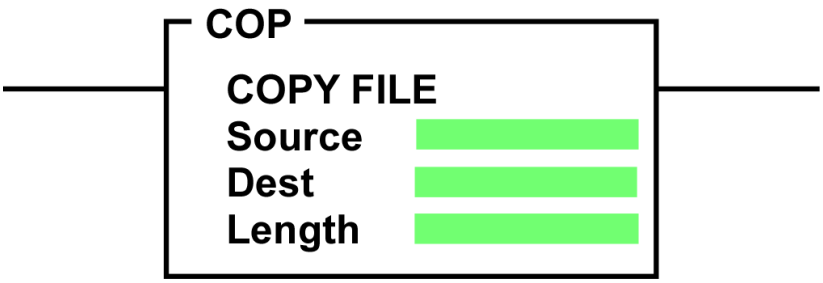
The zeros block the move and the 1's allow the value to move. The destination is:

0000 1111 0000 1010 (binary) = 0F0A (hex)

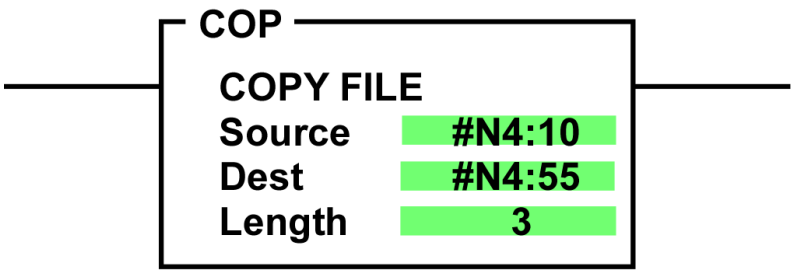
So: 1F2A with a mask of 00FF is 002A
and: 1F2A with a mask of F00F is 100A
and: 1F2A with a mask of FF0F is 1F0A

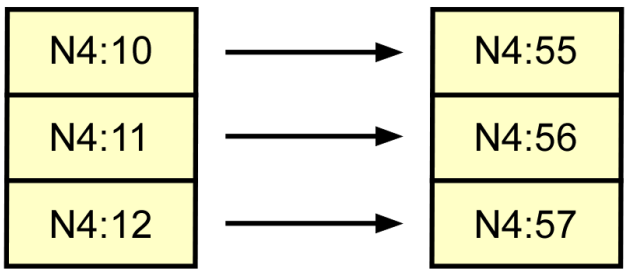
f. **COP:** This instruction requires you to input two values, the source and the destination. It copies the data from Source to the Destination. The Source is left untouched. Under the source, the PLC displays the current source value. Under the destination, the PLC displays the current destination value. You cannot control these values, they are just displayed. As long as the rung remains true, the instruction moves the data each scan. If you only want the move to happen one time, you should put an OSR instruction in front of it. This works exactly like the MOV command. The difference is the copy command can move more than one register. Copy is used to move 2 or more registers at the same time.

The source and destination both have # signs in front of them to let the command know that multiple registers are being moved. There is also a length value that is used to determine how many values are moved.

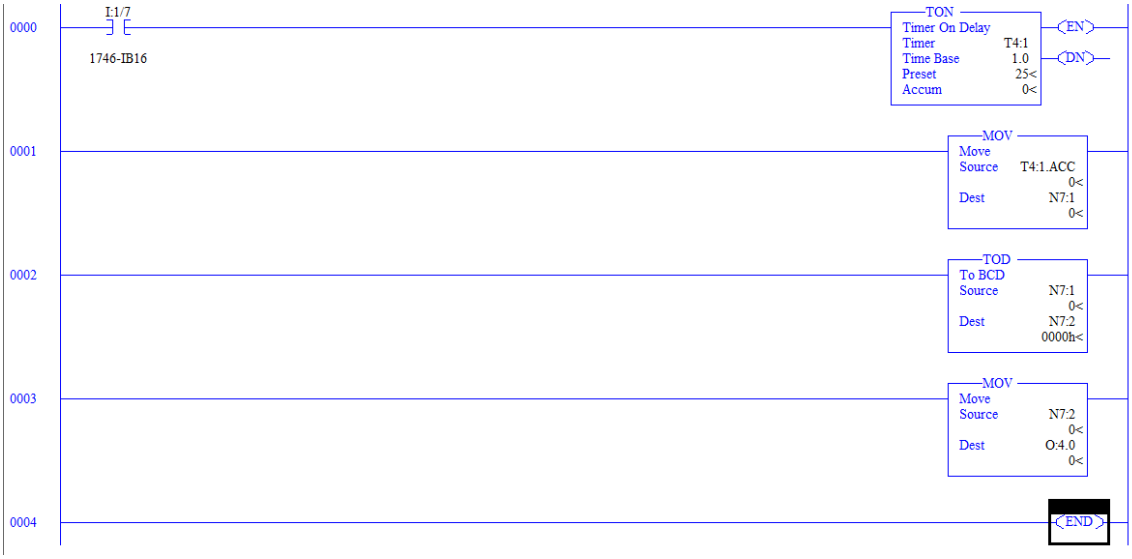


An example is shown below.



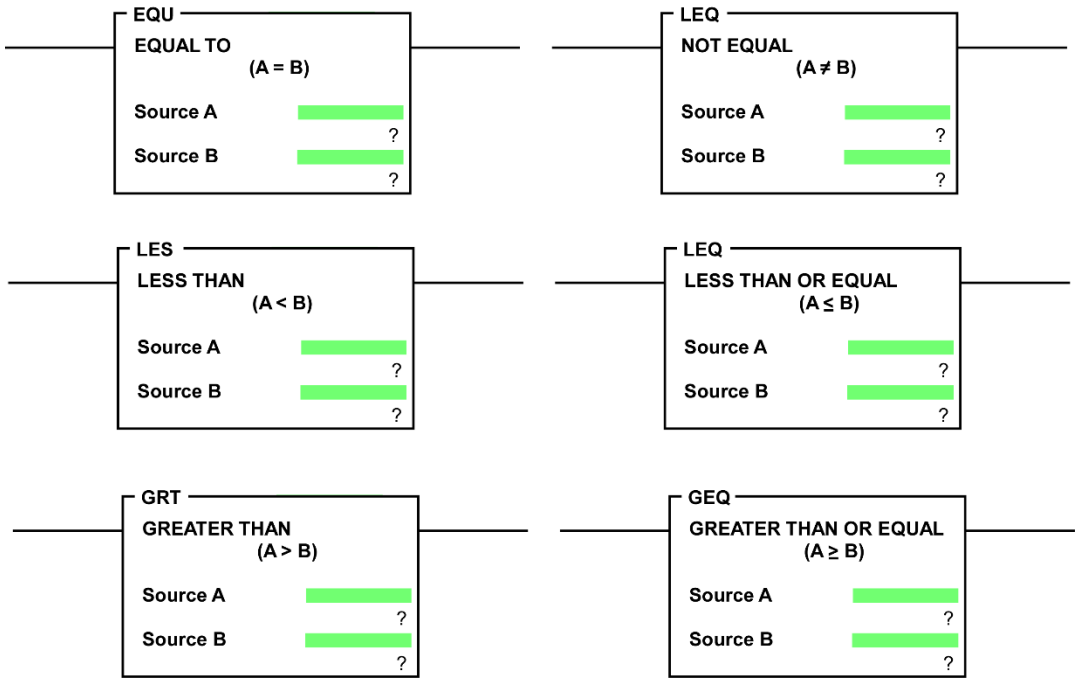


g. **Example:** When I:1/7 is turned on, the timer starts to count to 25. Because there is no input on the next three rungs, they run automatically. The second rung copies the base 10 number in the timer's Accumulator (ACC) into an integer storage register (N7:1). It is recommended that conversions should be done from an integer storage register and not from a value that might be changing. The register is then given to a TOD (To BCD) and the BCD value is stored in a second integer register. The third rung moves the BCD value to the BCD display. This will show the timer counting on the display.

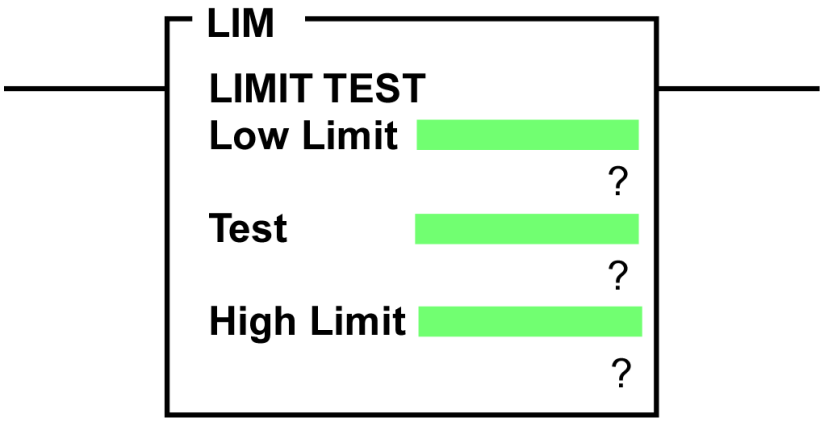


6) **Inputs:** The following data manipulation instructions are inputs that will turn on or turn off an output. If the answer to the question is TRUE, then the output is ON. If the answer to the question is FALSE, the output is OFF.

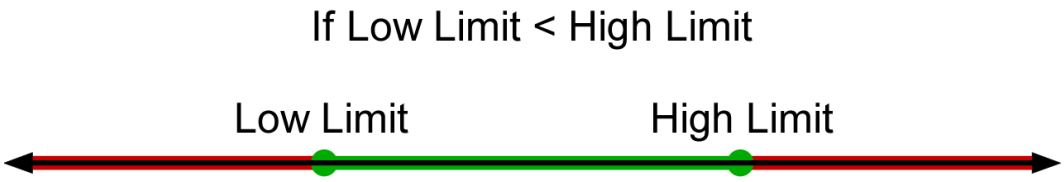
EQU	Source A is equal to Source B
NEQ	Source A is not equal to Source B
LES	Source A is less than Source B
LEQ	Source A is less than or equal to Source B
GRT	Source A is greater than Source B
GEQ	Source A is greater than or equal to Source B



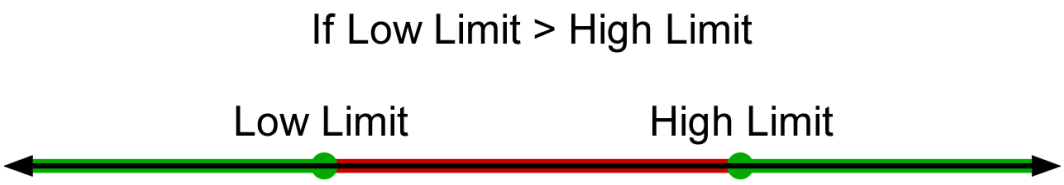
7) **Limits (in range and out of range):** The LIM command will check if a value is inside or outside of a range of numbers depending on how you enter the values. The command needs three values, the upper limit, the lower limit, and the value you want to test. The command is shown below.



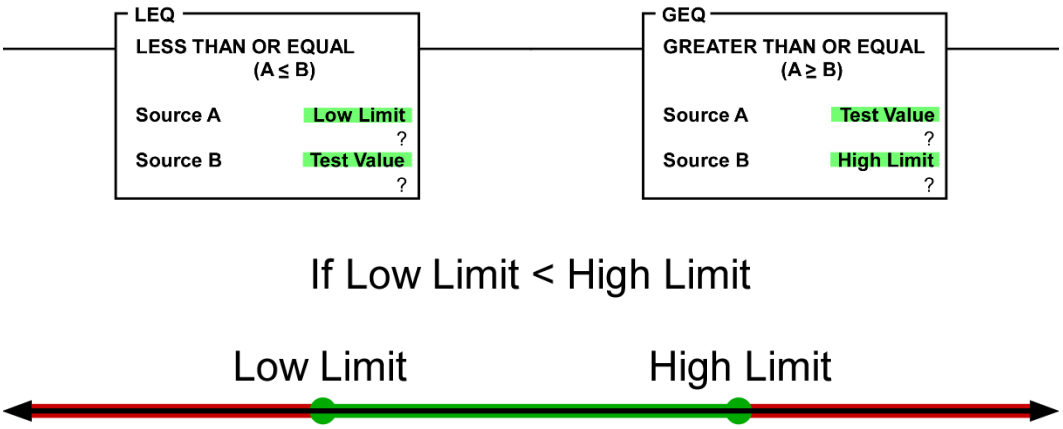
a. **Inside the Range:** When the low limit value is less than the high limit value, the LIM value will return a "true" if the test value is greater than or equal to the low limit and less than or equal to the high limit.



b. **Outside the Range:** When the low limit value is greater than the high limit value, the LIM value will return a "true" if the test value is less than or equal to the low limit or greater than or equal to the high limit.



8) **Inside the Range:** The limit command can be replaced with LEQ and GEQ. This method can be considered better because you can change one or the other limits to be GRT and LES. If the test value is between the Low Limit and the High Limit then both will be true and the output will be true.



9) **Outside the Range:** The limit command can be replaced with LEQ and GEQ. This method can be considered better because you can change one or the other limits to be GRT and LES. If the test value less than the Low Limit or higher than the High Limit then one will be true and the output will be true.