

Introduction to Programmable Logic Controllers – Part II

Module 5: Comparison Instructions and Math Instructions

This module addresses comparison and math instructions. After studying this module, the student will

- Know the common comparison and math instructions used in PLCs;
- Understand how these instructions work;
- Be able to use them in PLC ladder logic programs.

The comparison instructions are controlling instructions, which means they can be used any place in a ladder logic program, except at the most right position of a rung. A comparison instruction compares values of data. Its logic status depends on the data being compared. The comparison instructions include the following:

- Equal (EQU)
- Not Equal (NEQ)
- Less Than (LES)
- Less Than or Equal (LEQ)
- Greater Than (GRT)
- Greater Than or Equal (GEQ)
- Limit (LIM)

The formats of all but the Limit comparison instruction are given in Figures 5.1 to 5.6. As shown in the figures, each of these instructions involves two parameters, Source A and Source B. In these six instructions, one parameter can be a constant for comparing, while the other parameter can be a memory address of a word where the value for comparing is stored. (Each word contains sixteen bits of memory. For example, the Accumulate value of a counter instruction is stored in a word). Or both parameters can be memory addresses. However, constants can't be used for both parameters in the same instruction.

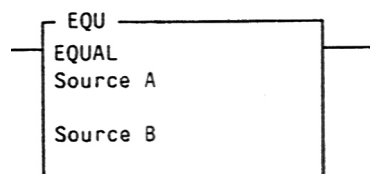


Figure 5.1 An Equal Instruction

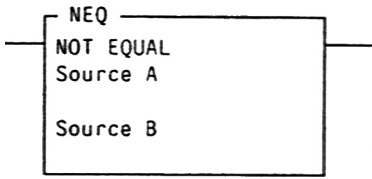


Figure 5.2 A Not Equal Instruction

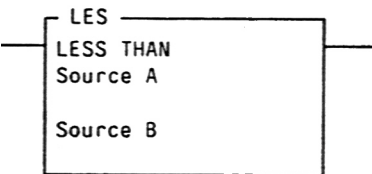


Figure 5.3 A Less Than Instruction

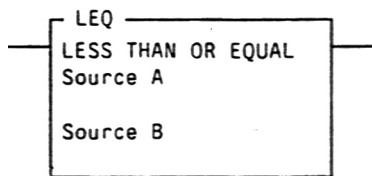


Figure 5.4 A Less Than or Equal Instruction

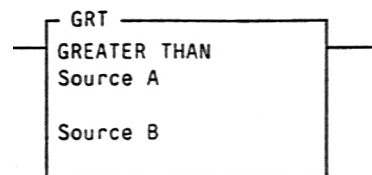


Figure 5.5 A Greater Than Instruction

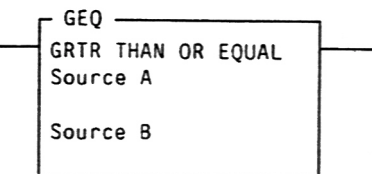


Figure 5.6 A Greater Than or Equal Instruction

The following table shows the conditions that cause these instructions to be true or false.

Instruction	Instruction is logically TRUE	Instruction is logically FALSE
Equal	If the value at source A is equal to the value at source B	If the value at source A is not equal to the value at source B
Not Equal	If the value at source A is not equal to the value at source B	If the value at source A is equal to the value at source B
Less Than	If the value at source A is less than the value at source B	If the value at source A is greater than or equal to the value at source B
Less Than or Equal	If the value at source A is less than or equal to the value at source B	If the value at source A is greater than the value at source B
Greater Than	If the value at source A is greater than the value at source B	If the value at source A is less than or equal to the value at source B
Greater Than or Equal	If the value at source A is greater than or equal to the value at source B	If the value at source A is less than the value at source B

Figure 5.7 shows the format of a limit instruction. This instruction checks a value within or outside of a specified range. There are three parameters in this instruction. They are the Low Limit, the Test, and the High Limit. These parameters can be constants or the memory address of a word with the restrictions of:

- If the Test parameter is a constant, both the Low Limit and High Limit parameters must be word addresses.
- If the Test parameter is a word address, both the Low Limit and High Limit parameters can be constants; both of them can be word addresses; or one can be a constant and the other can be a word address.

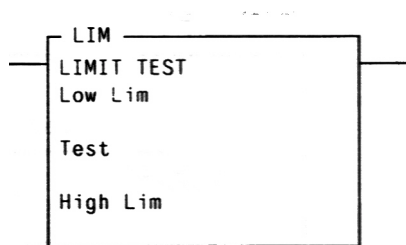


Figure 5.7 A Limit Instruction

In a Limit instruction, the Low Limit value can be either less than or greater than the High Limit value. The conditions that cause this instruction to be true or false are given in the following table.

	Instruction is logically TRUE	Instruction is logically FALSE
When Low Limit value is less than High Limit Value	If Low Limit value $\leq$ Test value $\leq$ High Limit value	If Test value $<$ Low Limit value or Test value $>$ High Limit value
When Low Limit value is greater than High Limit Value	If Test value $\leq$ Low Limit value or Test value $\geq$ High Limit value	If Low Limit value $<$ Test value $<$ High Limit value

Figure 5.8 is a ladder logic program that uses one timer instruction to control three devices, which could be motors, lights, or similar devices. After a start button is pushed, the first device is turned-on for 5 seconds. Then this device is turned-off and the second device is turned-on for 10 seconds. After that, the second device is turned-off and the third device is turned-on for 8 seconds. In the ladder logic program, the three devices are connected to the output terminals O:3/1, O:3/2, and O:3/3. Pushing a start button connected to the input terminal I:2/1 enables the timer T4:2. This timer latches itself by the Exam-if-Closed instruction T4:1/TT. When the timer is timing and its Accumulate value is less than 500 (the time is between 0 and 5 seconds), the second rung is true and the output instruction O:3/1 is energized. When the timer's Accumulate value is between 500 and 1500, the third rung is true and the output instruction O:3/2 is energized. The fourth rung is true and O:3/3 is turned on when the timer's Accumulated value is between 1500 and 2300. The timer is done when its Accumulated value reaches 2300 and T4:1/TT goes to low. This causes the timer to deactivate and its Accumulate value to be reset, which turns O:3/3 off.

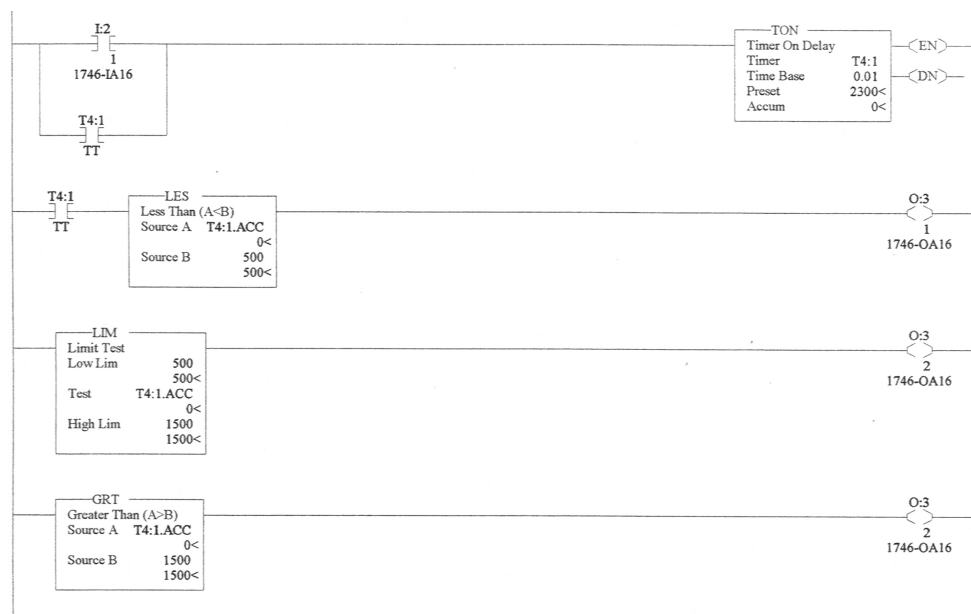


Figure 5.8 A Program Using Comparison Instructions

Math instructions are controlled instructions. In a ladder logic program, when its rung is true, the mathematical operation is performed. Commonly used math instructions include:

- Add (ADD)
- Subtract (SUB)
- Multiply (MUL)
- Divide (DIV)

As shown in Figures 5.9 to 5.12, each of the math instructions has three parameters: Source A, Source B, and Destination. One of the Source A and Source B parameters can be a constant for the mathematical operation and the other can be a memory address of a word where the value for the mathematical operation is stored. Both of them can also be word addresses however, Source A and B cannot both be constants. Destination is a memory address of a word where the mathematical operation result is stored.

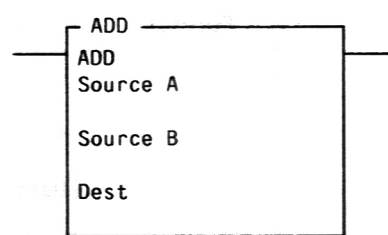


Figure 5.9 An Add Instruction

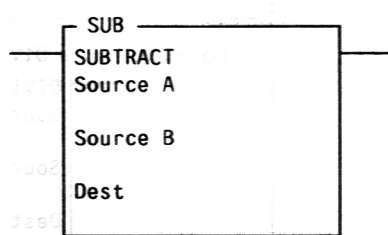


Figure 5.10 A Subtract Instruction

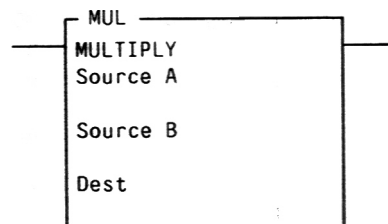


Figure 5.11 A Multiply Instruction

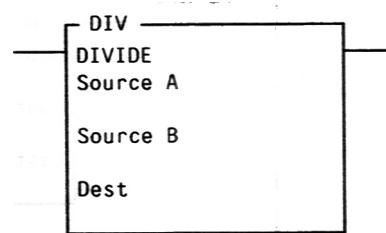


Figure 5.12 A Divide Instruction

Figure 5.13 is a program that counts the numbers of boxes and calculates the number of products. Two kinds of products are sent down a conveyor: Product A, ten of them in each box, and product B, twenty of them in each box. There are two sensors that count the boxes. The first sensor, which is connected to input terminal I:2/3, counts the boxes of product A. The other sensor, which is connected to input terminal I:2/4, counts the boxes of product B. When a box passes by a sensor, that sensor triggers and activates a corresponding counter instruction for counting. The Accumulate value of counter C5:1 is the number of product A boxes and the Accumulate value of C5:2 is the number of product B boxes. When C5:1 is counting, it activates the Multiply instruction in the third rung. This instruction multiplies the C5:1 Accumulate value by 10 and stores the result, which is the number of product A, in memory location B3:10. When C5:2 is counting, it activates the Multiply instruction in the fourth rung. This instruction multiplies the C5:2 Accumulate value by 20 and stores the result, which is the number of product B, in memory location B3:11. When either C5:1 or C5:2 is counting, the add instruction in the fifth rung adds the values in B3:10 and B3:11 together and stores the result, which is the total number of products A and B, in the memory location B3:12. The input I:2/6 is a reset switch. When this switch is closed, both Accumulate values of counter C5:1 and C5:2 are reset to zero. This switch also activates the math instructions in rungs three to five. That causes B3:10, B3:11, and B3:12 to become zero.

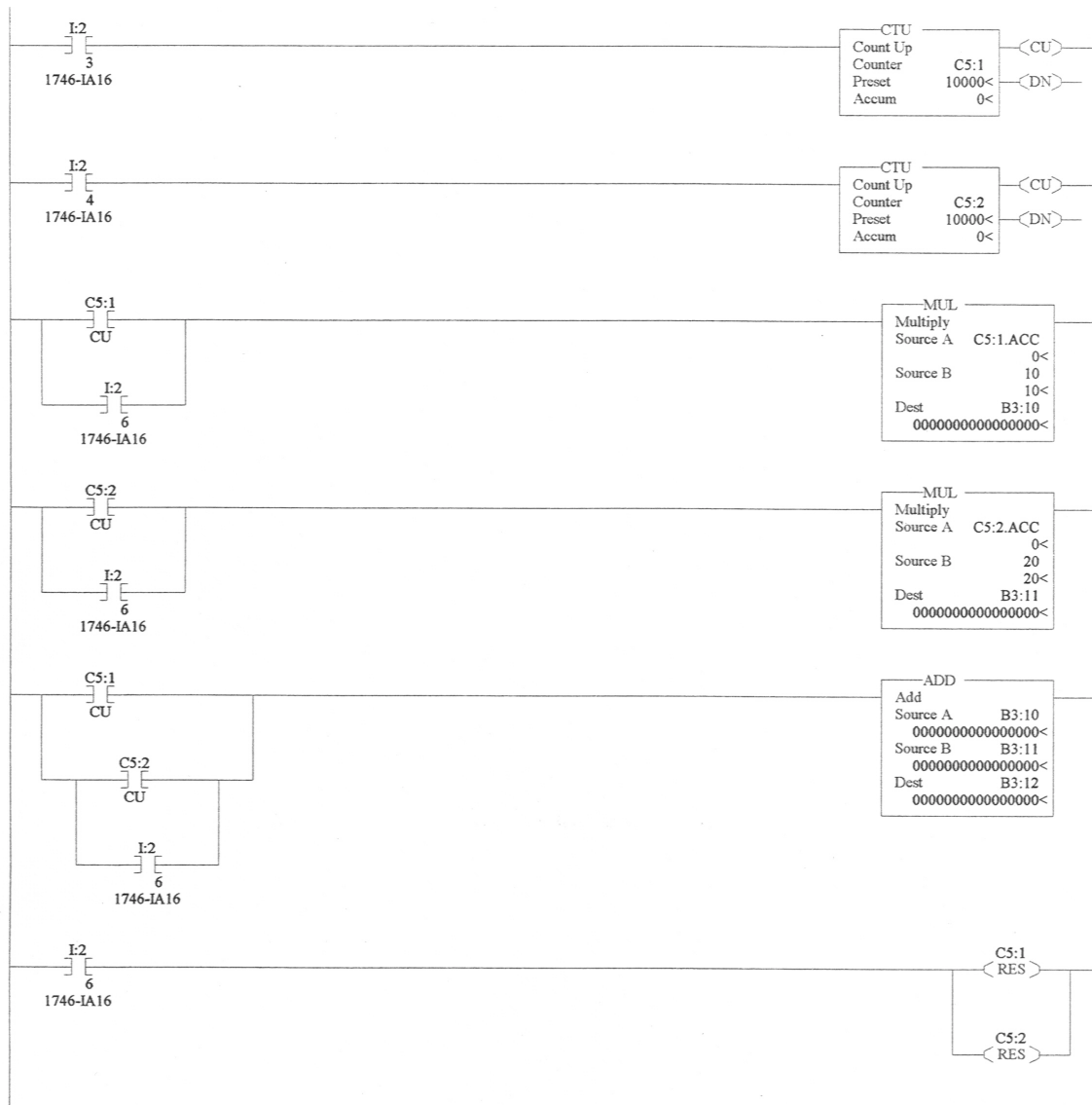


Figure 5.13 A Program Using Math Instructions

Related web sites:

<http://www.plcs.net>

<http://www.sea.siemens.com/step/templates/lesson.mason?plcs:1:1:1>