➢Chapter 11
Math instruction

❑Math Instructions

- Math instructions, like data manipulation instructions, enable the programmable controller to take on more of the qualities of a conventional computer.
- The PLC's math functions capability allows it to perform arithmetic functions on values stored in memory words or registers.
- For example, assume you are using a counter to keep track of the number of parts manufactured, and you would like to display how many more parts must be produced in order to reach a certain quota.
- This display would require the data in the accumulated value of the counter to be subtracted from the quota required.
- Other applications include combining parts counted, subtracting detected defects, and calculating run rates.
- Depending on what type of processor is used, various math instructions can be programmed.

❑Math Instructions

➢The basic four mathematical functions performed by PLCs are:

✓**Addition —**The capability to add one piece of data to another.

✓**Subtraction —**The capability to subtract one piece of data from another.

✓**Multiplication —**The capability to multiply one piece of data by another.

✓**Division —**The capability to divide one piece of data by another.

❑Math Instructions

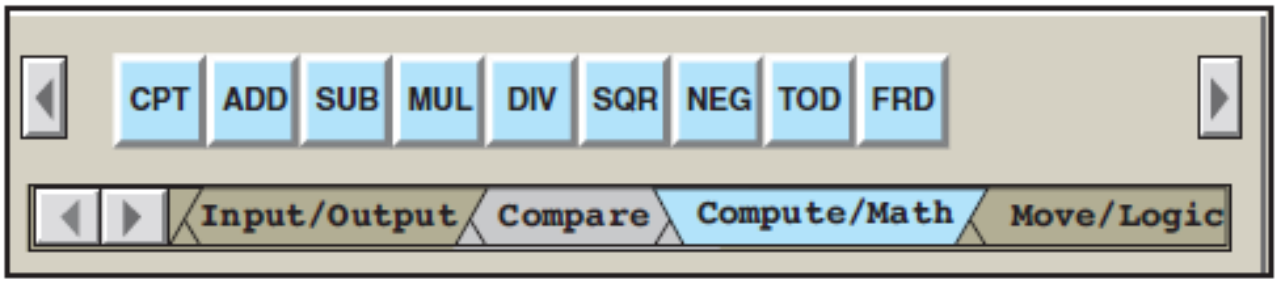
➢The basic four mathematical functions performed by PLCs are:

✓**CPT (Compute)** —Evaluates an expression and stores the result in the destination.

✓**ADD (add)** —Adds source *A to source B and stores* the result in the destination.

✓**SUB (Subtract)** —Subtracts source *B from source A* and stores the result in the destination.

✓**MUL (Multiply)** —Multiplies source *A by source B* and stores the result in the destination.



❑Math Instructions

➢The basic four mathematical functions performed by PLCs are:

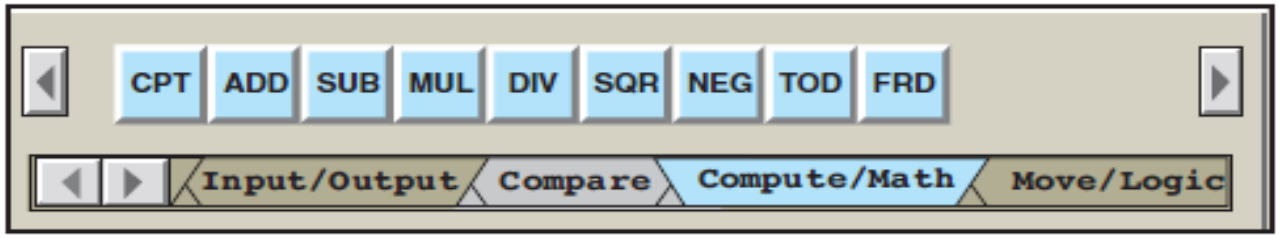
✓**DIV (Divide) —**Divides source *A by source B and* stores the result in the math register.

✓**SQR (Square Root) —**Calculates the square root of the source and places the integer result in the destination.

✓**NEG (Negate) —**Changes the sign of the source and places it in the destination.

✓**TOD (To BCD) —**Converts a 16-bit integer source value to BCD and stores it in the math register or the destination.

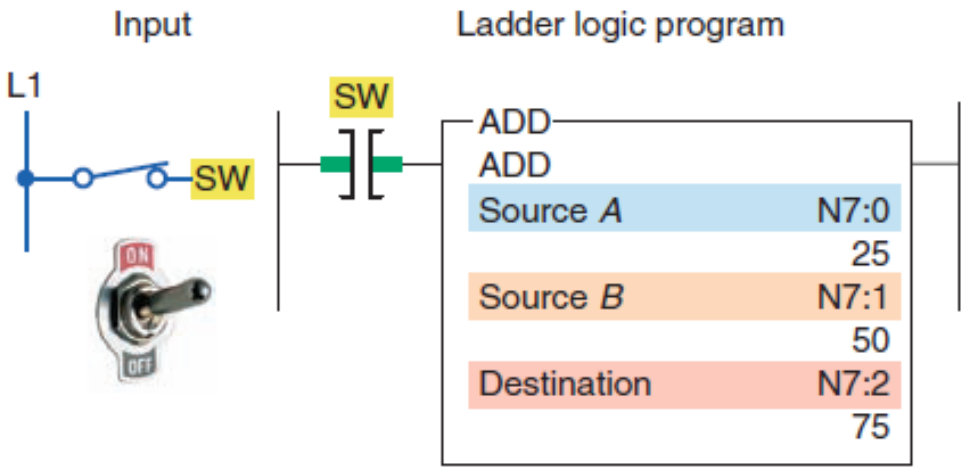
✓**FRD (From BCD) —**Converts a BCD value in the math register or the source to an integer and stores it in the destination.



❑Math Instructions

➢Addition Instruction

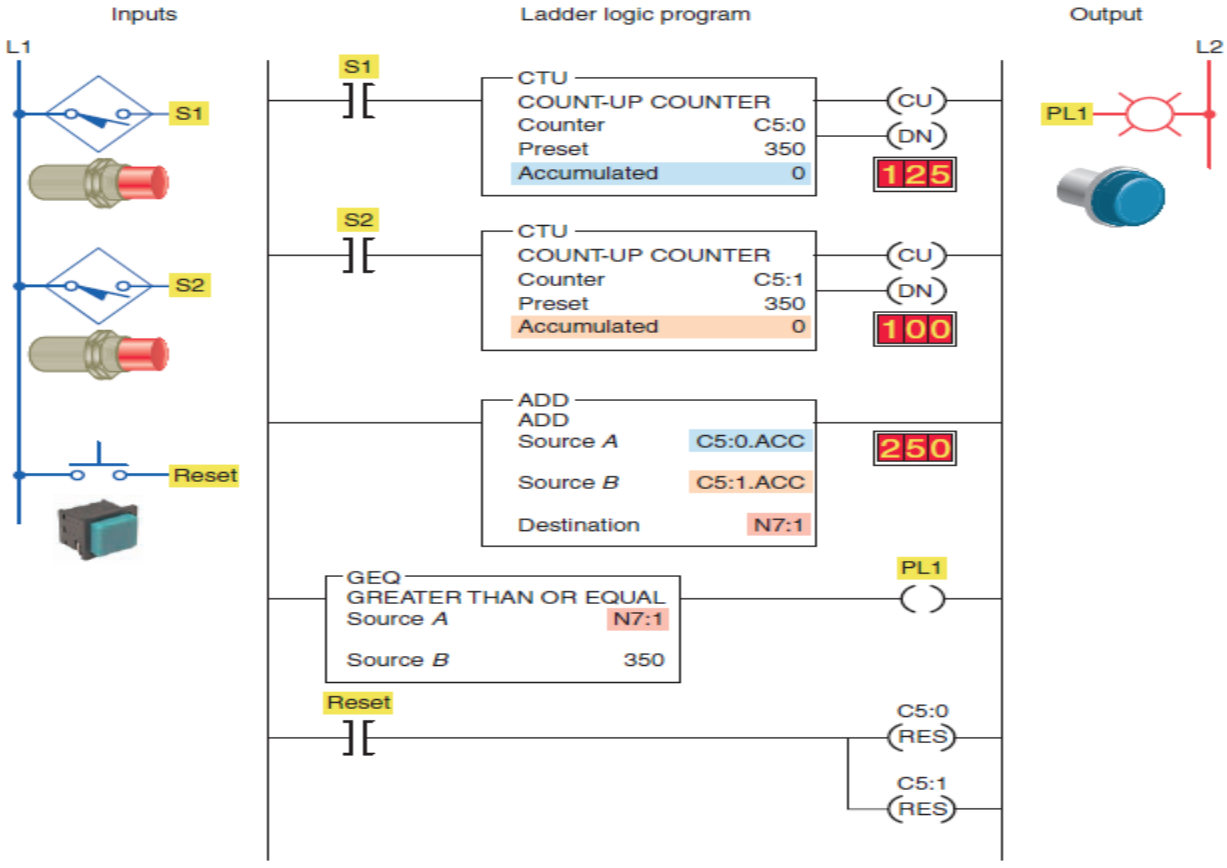
Most math instructions take two input values, perform the specified arithmetic function, and output the result to an assigned memory location.



❑Math Instructions

➢Addition Instruction

The program of Figure illustrates how the ADD instruction can be used to add the accumulated counts of two up-counters



❑Math Instructions

➢Addition Instruction

✓When performing math functions, care must be taken to ensure that values remain in the range that the data table or file can store; otherwise, the overflow bit will be set.

✓The arithmetic status bits for the SLC 500 controller are found in word 0, bits 0 to 3 of the processor status file S2 (Figure).

✓After an instruction is executed, the arithmetic status bits in the status file are updated.

✓The description of each bit can be summarized as follows:

Status Table

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S2:0/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S2:1/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S2:2/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S2:3/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S2:4/	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1
S2:5/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Address S2:0 Table: S2:Status

❑Math Instructions

➢Addition Instruction

✓The description of each bit can be summarized as follows:

Carry (C)—Address S2:0/0, is set to 1 when there is a carry in the ADD instruction or a borrow in the SUB instruction.

Overflow (O)—Address S2:0/1, is set to 1 when the result is too large to fit in the destination register.

Zero (Z)—Address S2:0/2, is set to 1 when the result of the subtract instruction is zero.

Sign (S)—Address S2:0/3, is set to 1 when the result is a negative number.

Status Table

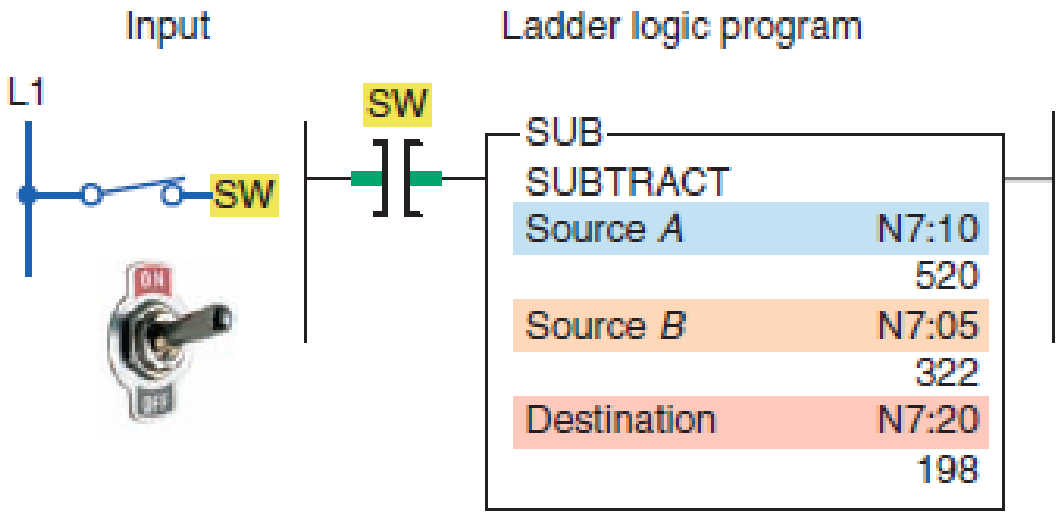
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S2:0/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S2:1/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S2:2/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S2:3/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S2:4/	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1
S2:5/	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Address S2:0 Table: S2:Status

❑Math Instructions

➢Subtraction Instruction

✓The *SUB (subtract) instruction is an output instruction that* subtracts one value from another and stores the result in the destination address

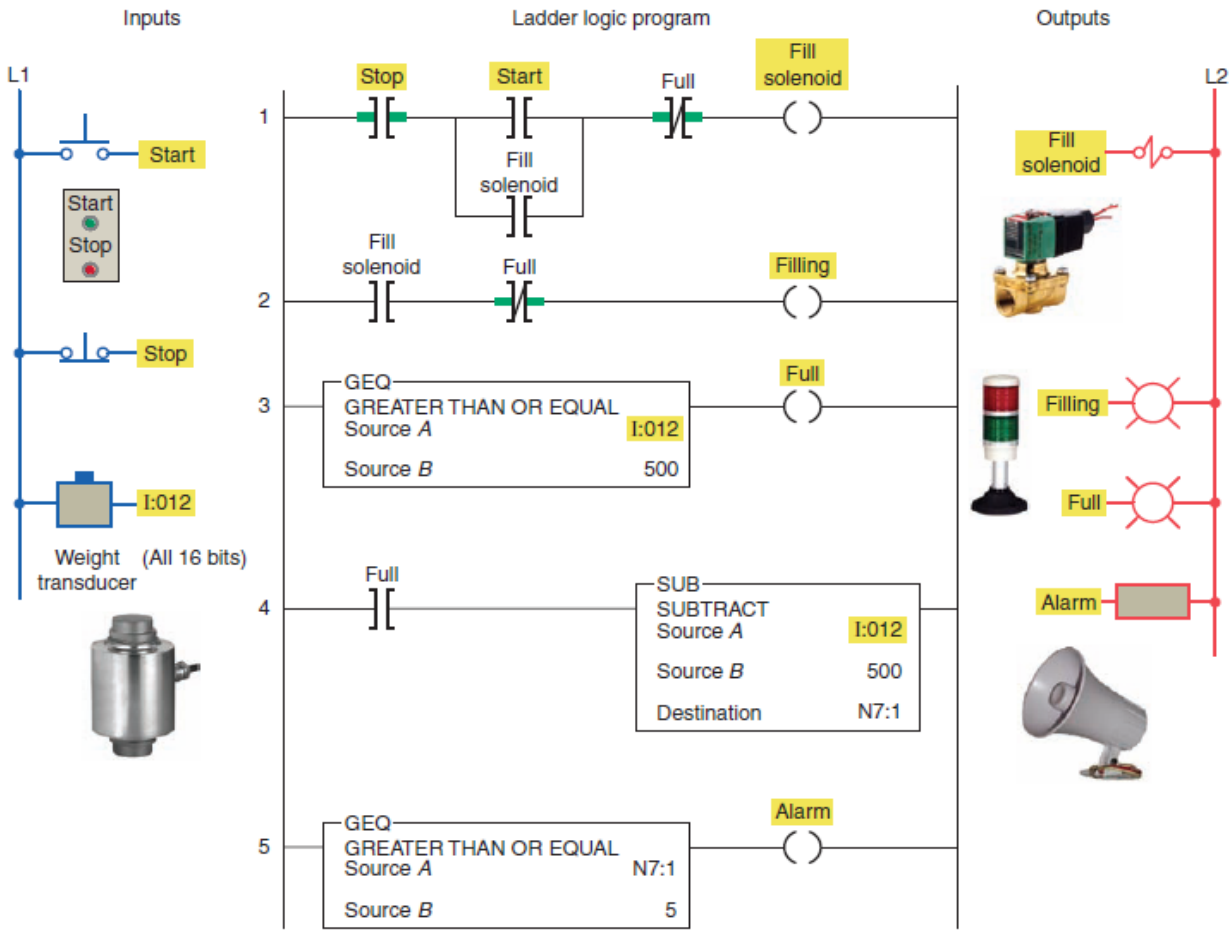


❑Math Instructions

➢Subtraction Instruction

✓The program of Figure shows how the SUB function can be used to indicate a vessel overfill condition.

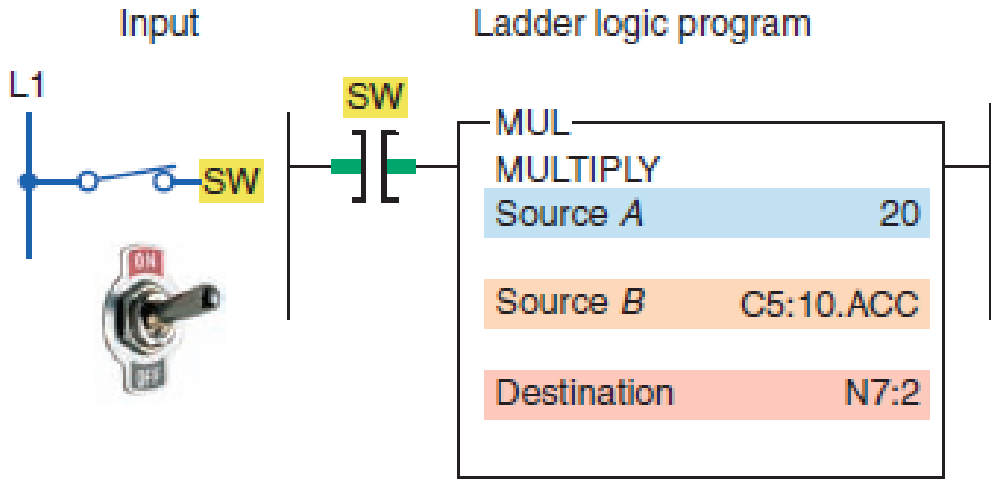
✓This application require an alarm to sound when a supply system leaks 5 lb or more of raw material into the vessel after a preset weight of 500 lb has been reached



❑Math Instructions

➢Multiplication Instruction

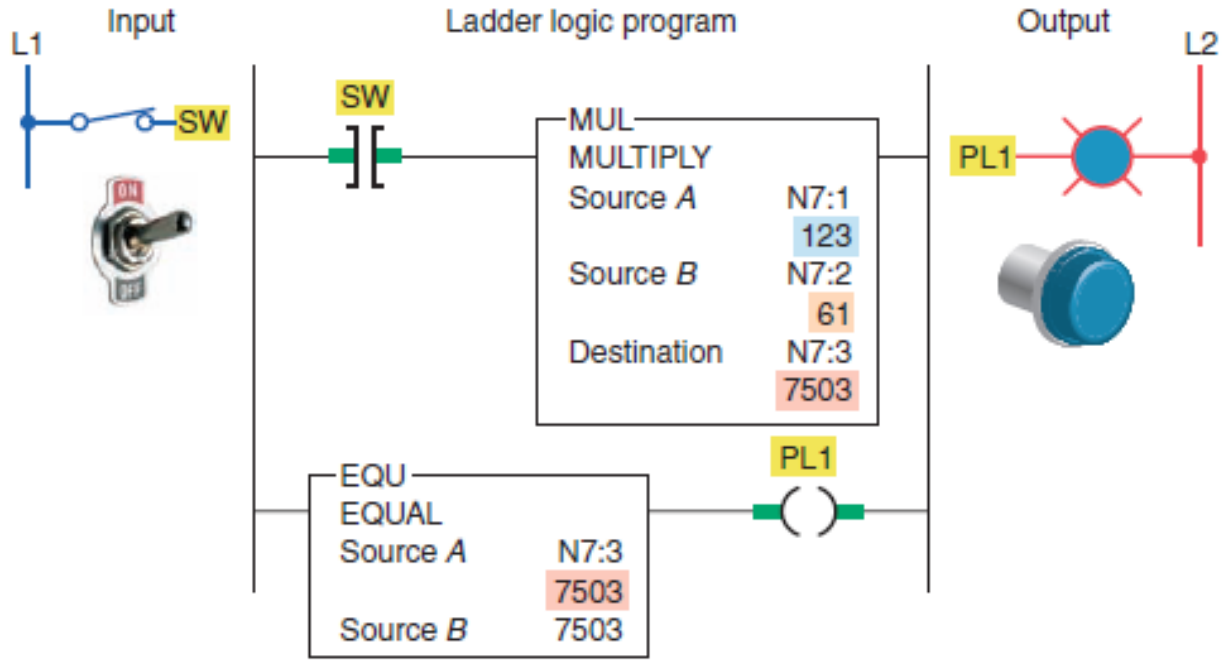
✓The *multiply (MUL) instruction is an output instruction* that multiplies two values and stores the result in the destination address



❑Math Instructions

➢Multiplication Instruction

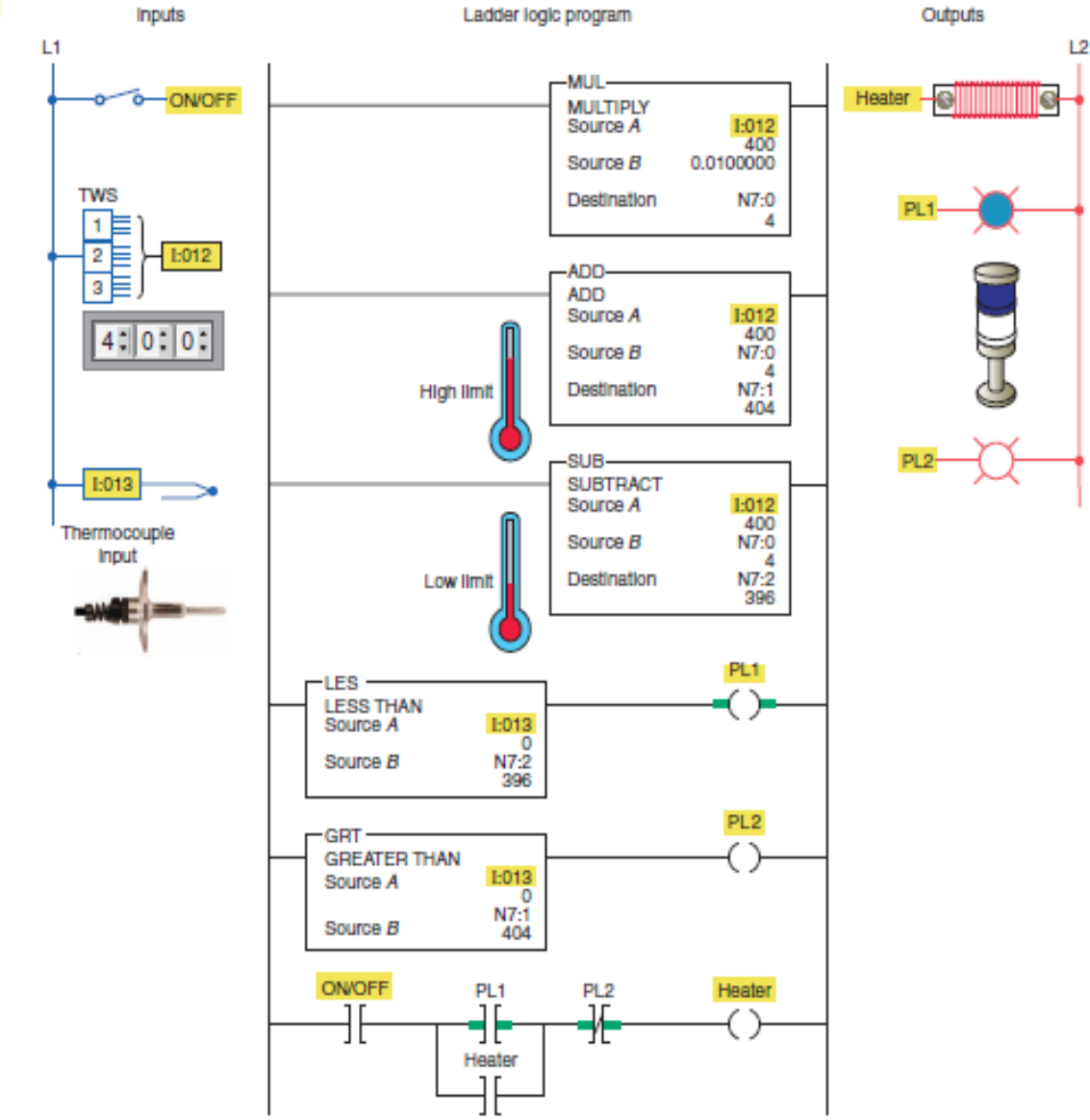
✓The program of Figure is an example of how MUL instruction calculates the product of two sources.



❑Math Instructions

➢Multiplication Instruction

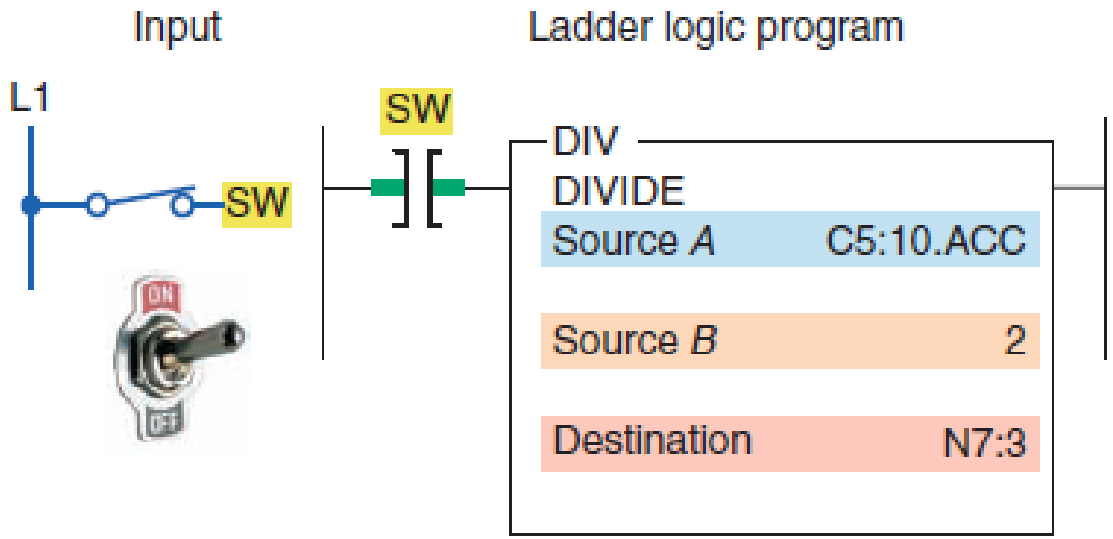
The program of Figure is an example of how the MUL instruction is used as part of an oven temperature control program



❑Math Instructions

➢Division Instruction

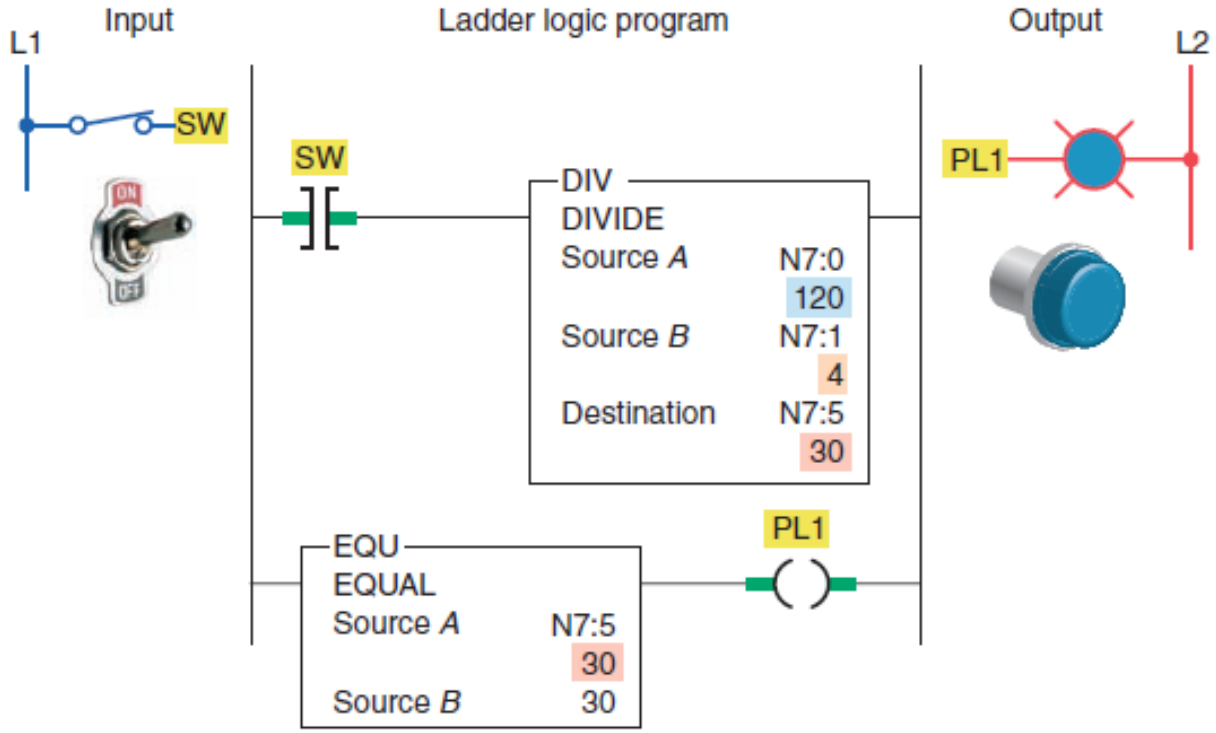
✓The *divide (DIV) instruction divides the value in source A* by the value in source *B and stores the result in the destination* and math register.



❑Math Instructions

➢Division Instruction

✓The program of Figure is an example of how the DIV instruction calculates the integer value that results from dividing source *A by source B*

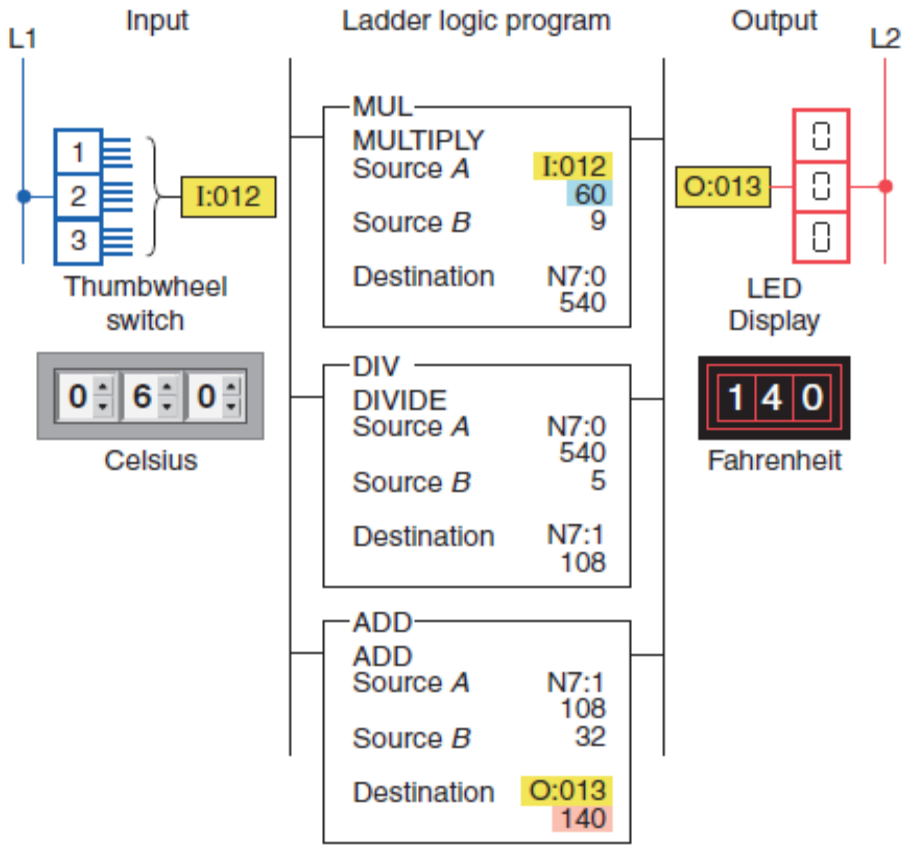


❑Math Instructions

➢Division Instruction

✓The program of Figure is an example of how the DIV function is used as part of a program to convert Celsius temperature to Fahrenheit

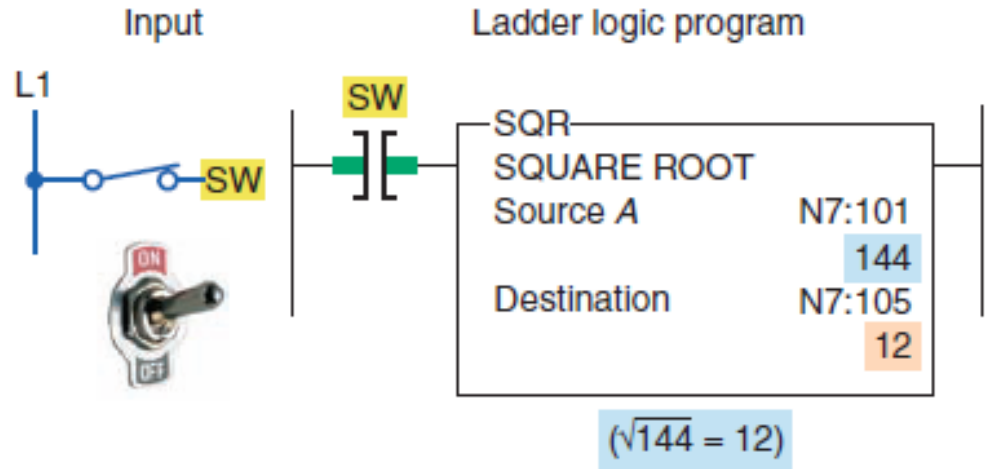
$$F = \left(\frac{9}{5} \times C\right) + 32$$



❑Math Instructions

➢Other Word-Level Math Instructions

✓The program of Figure is an example of the *square root (SQR) instruction*

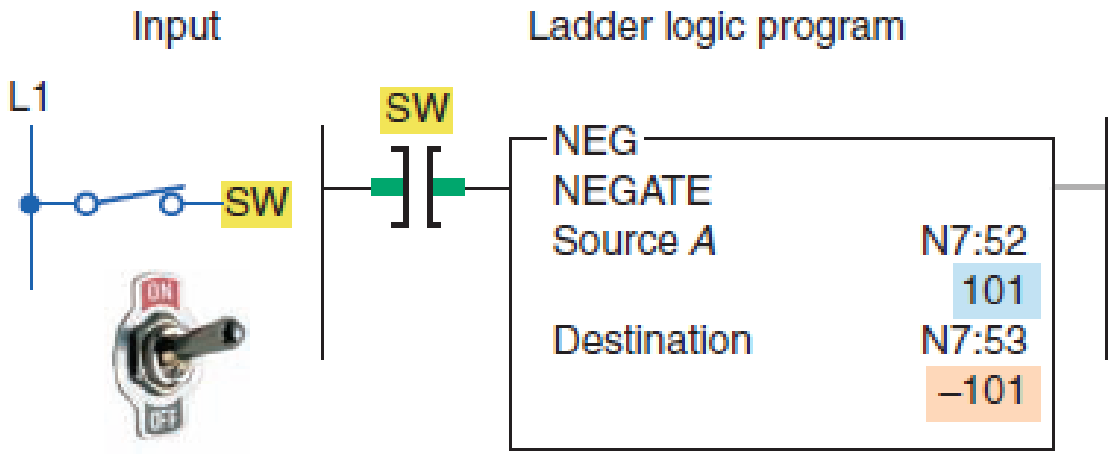


❑Math Instructions

➢Other Word-Level Math Instructions

✓The program of Figure is an example of the *negate (NEG) instruction.*

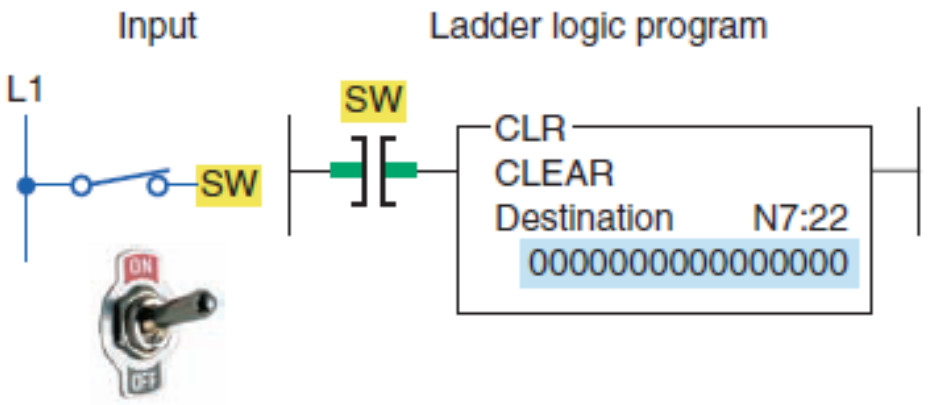
✓*This math function changes* the sign of the source value from positive to negative.



❑Math Instructions

➢Other Word-Level Math Instructions

✓The program of Figure 11-16 is an example of the *clear (CLR) instruction*

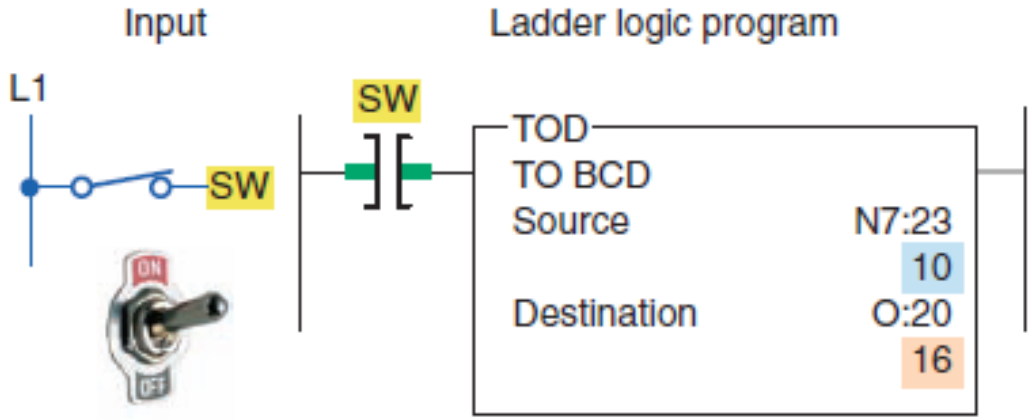


❑Math Instructions

➢Other Word-Level Math Instructions

✓The *convert to BCD (TOD) instruction is used to* convert 16-bit integers into *binary-coded decimal (BCD) values.*

✓ *This instruction could be used when* transferring data from the processor (which stores data in binary format) to an external device, such as an LED display, that functions in BCD format



❑Math Instructions

➢Other Word-Level Math Instructions

✓The *convert from BCD (FRD) instruction is used to convert* binary-coded decimal (BCD) values to integer values.

✓This instruction could be used to convert data from a BCD external source, such as a BCD thumbwheel switch, to the binary format in which the processor operates.

✓The program of Figure is an example of the FRD instruction