Programmable Logic Controllers

Third Edition

Frank D. Petruzella

McGraw-Hill

Chapter 11

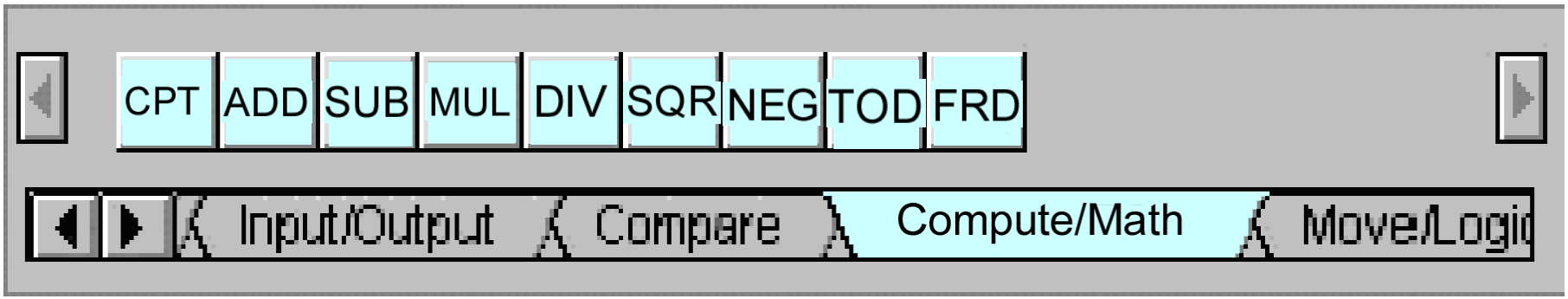
Math Instructions

Math Instructions

PLC math instructions allow you to perform arithmetic functions on values stored in memory words.

For example, assume you are using a counter to keep track of the numbers of parts manufactured and you would like to display how many more must be produced in order to reach a certain quota. This would require the data in the accumulated value of the counter to be subtracted from the quota required.

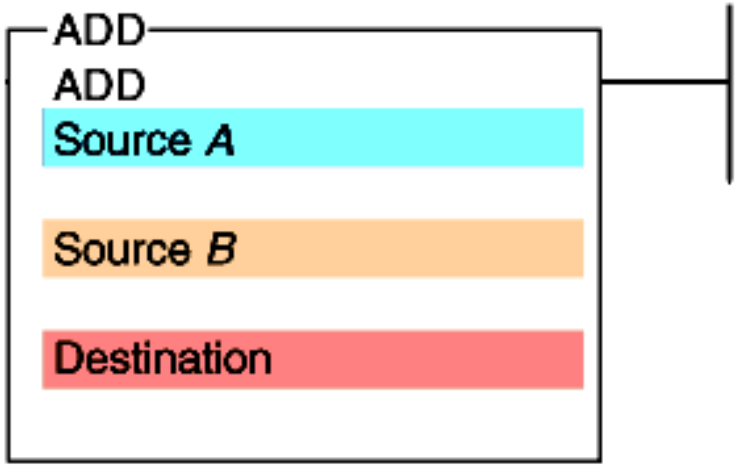
SLC 500 Math Instructions



COMMAND	NAME	DESCRIPTION
FRD	From BCD	Converts a BCD value in the math register or the source to an integer and stores it in the destination.

ADD Instruction

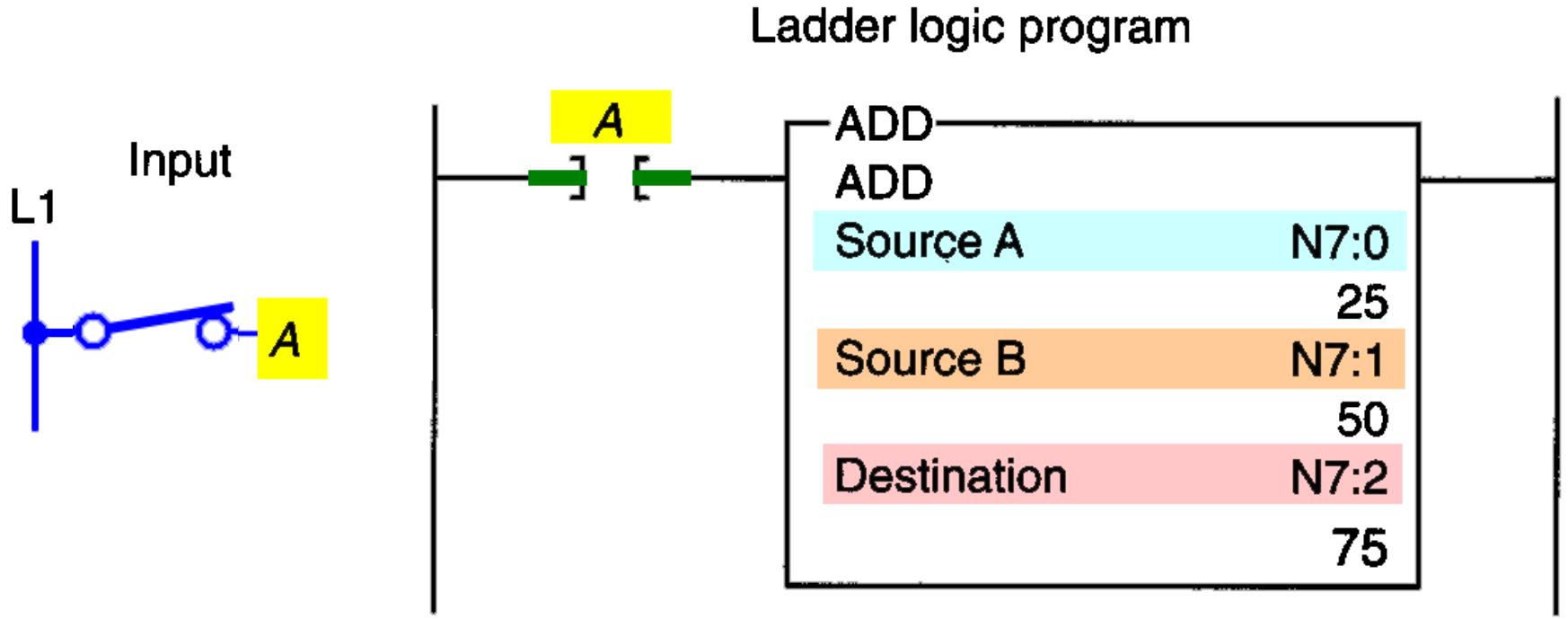
The ADD instruction is an output instruction that performs the addition of two values stored in the referenced memory locations.



This instruction adds the value stored at Source A to the value stored at Source B and stores the answer at the Destination.

ADD Instruction

When the rung is true, the value stored at the source A address, N7:0 (25), is added to the value stored at the source address N7:1 (50), and the answer (75) is stored at the destination address, N7:2.

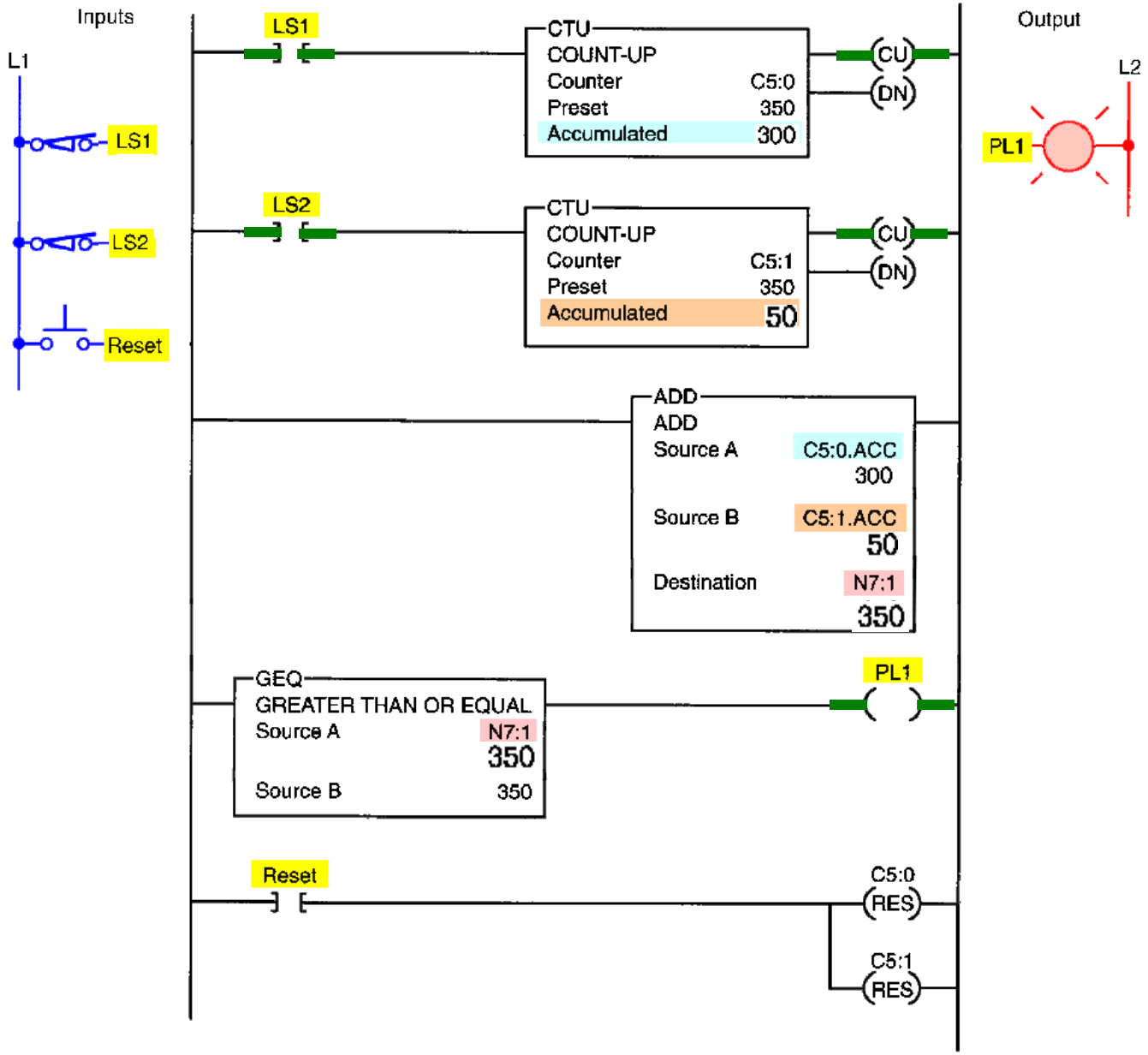


Source A and source B can either be values or addresses that contain values, however source A and source B cannot *both* be constants.

Counter Program That Uses The ADD Instruction

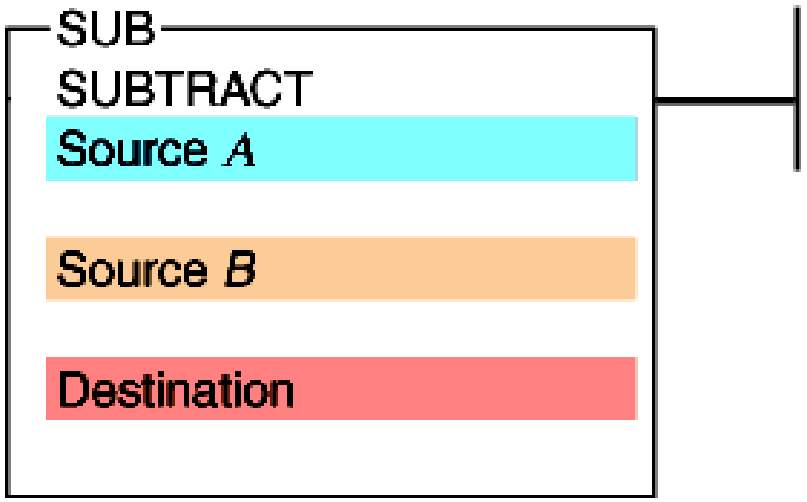
The program of the following slide shows how the ADD instruction can be used to add the accumulated counts of two up-counters. This application requires a light to come on when the sum of the counts from the two counters is equal to or greater than 350. Source A of the ADD instruction is addressed to store the accumulated value of counter C5:0, while source B is addressed to store the accumulated value of counter C5:1. The value at source A is added to the value at source B and the result (answer) is stored at destination address N7:1. Source A of the GREATER THAN OR EQUAL instruction is addressed to store the value of the destination address N7:1, while source B contains the constant value of 350. Therefore the GREATER THAN OR EQUAL instruction will be logic true whenever the accumulated values in the two counters are equal to or greater than the constant value 350. A reset button is provided to reset the accumulated count of both counters to zero.

Counter Program That Uses The ADD Instruction



SUBTRACT Instruction

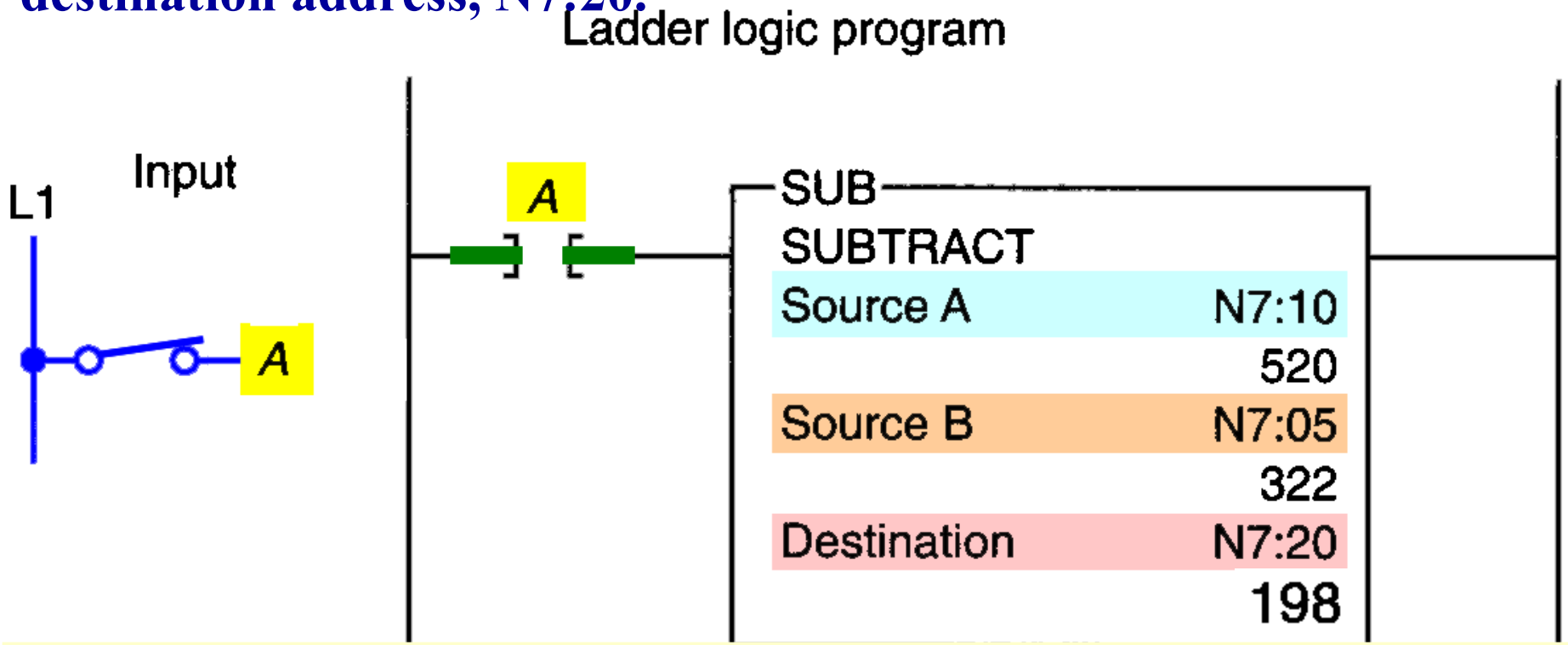
The SUBTRACT instruction is an output instruction that subtracts one value from another and stores the result in the destination address.



When rung conditions are true, the subtract instruction subtracts source B from source A and stores the result in the destination.

SUBTRACT Instruction

When the rung is true, the value stored at the source B address, N7:05 (322), is subtracted from the value stored at the source A address, N7:10 (520), and the answer (198) is stored at the destination address, N7:20.

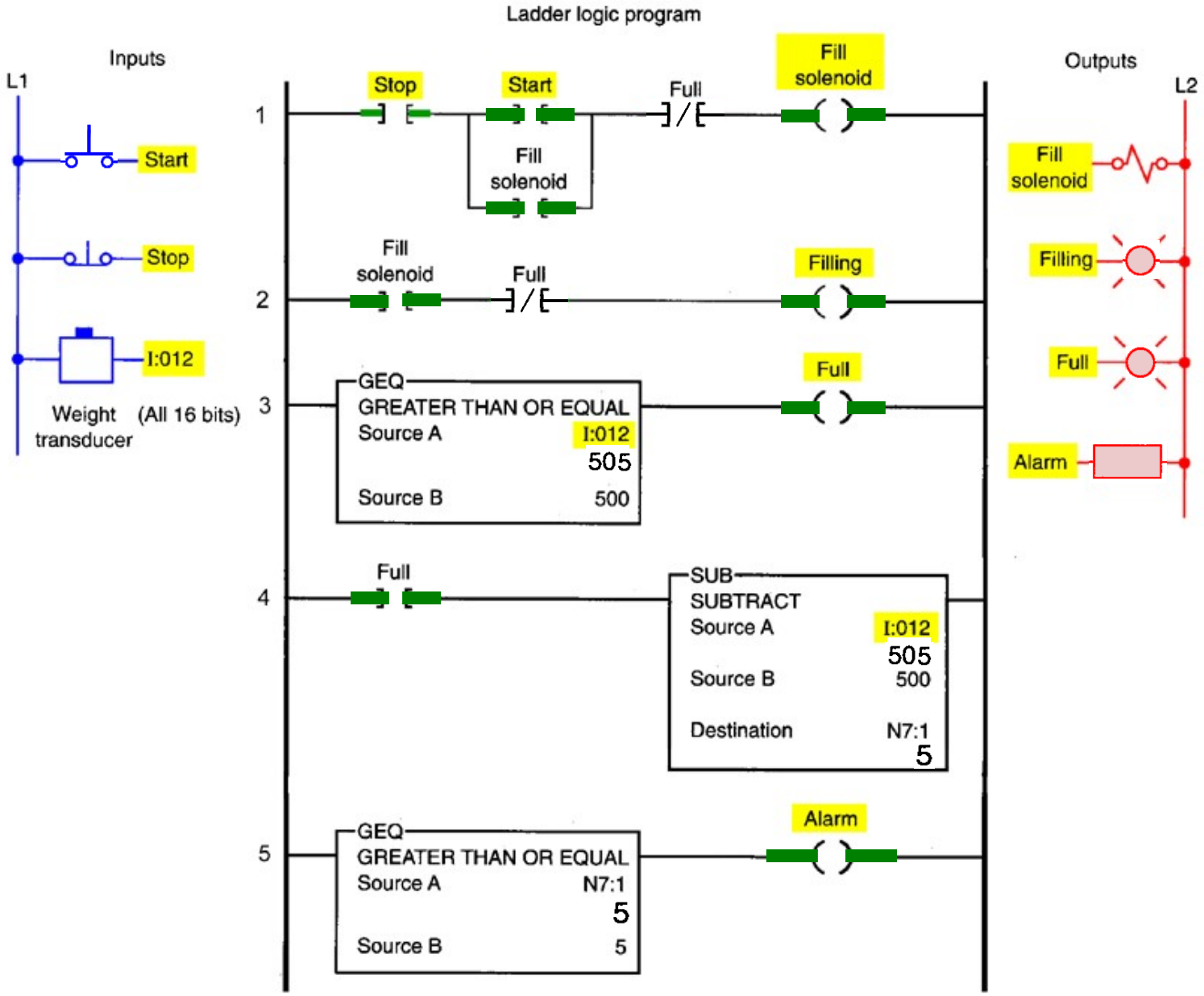


Source A and source B can either be values or addresses that contain values, however source A and source B *cannot* both be constants.

Overfill Alarm Program

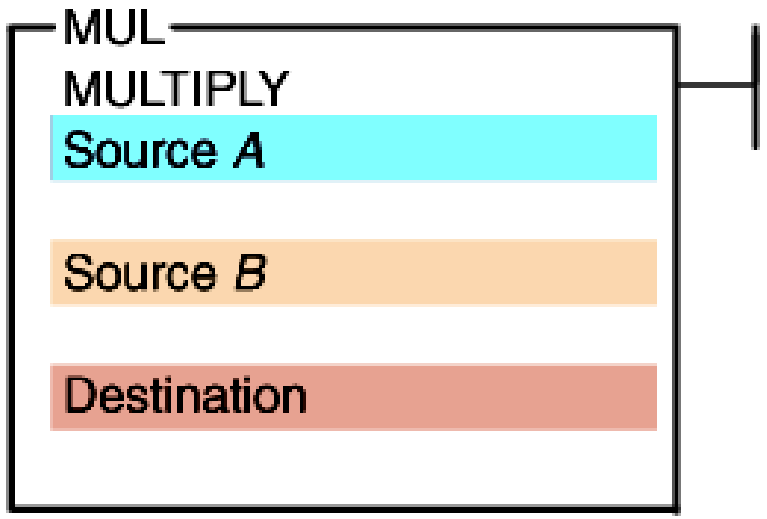
The program of the following side shows how the SUBTRACT function can be used to indicate a vessel overfill condition. This application requires an alarm to sound when a supply system leaks 5 lb or more of raw material into the vessel after a preset weight of 500 lb has been reached. When the start button is pressed, the fill solenoid (rung 1) and filling indicating light (rung 2) are turned on and raw material is allowed to flow into the vessel. The vessel has its weight monitored continuously by the PLC program (rung 3) as it fills. When the weight reaches 500 lb, the fill solenoid is de-energized and the flow is cut off. At the same time, the filling pilot light indicator is turned off and the full pilot light indicator (rung 3) is turned on. Should the fill solenoid leak 5 lb or more of raw material into the vessel, the alarm (rung 5) will energize and stay energized until the overflow level is reduced below the 5 lb overflow limit.

Overfill Alarm Program



MULTIPLY Instruction

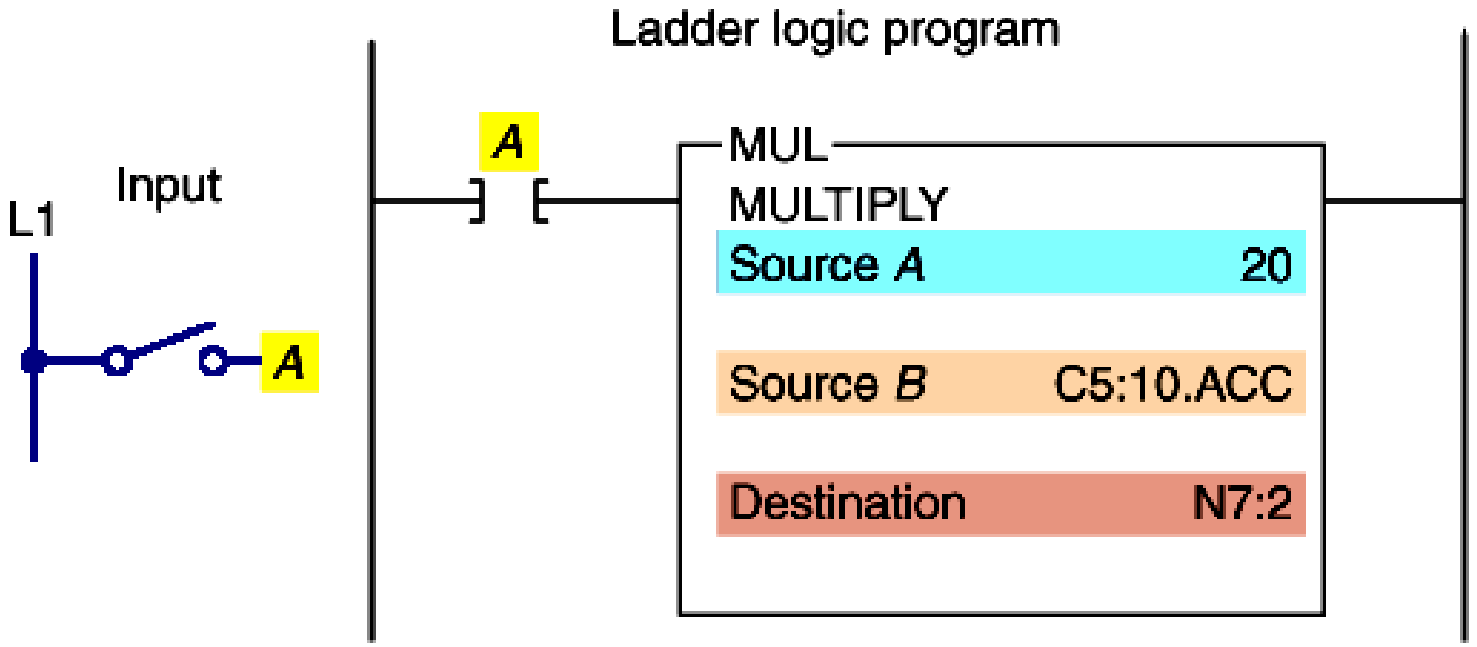
The MULTIPLY instruction is an output instruction that multiplies two values and stores the result in the destination address.



When rung conditions are true, the multiply instruction multiplies source A by source B and stores the result in the destination.

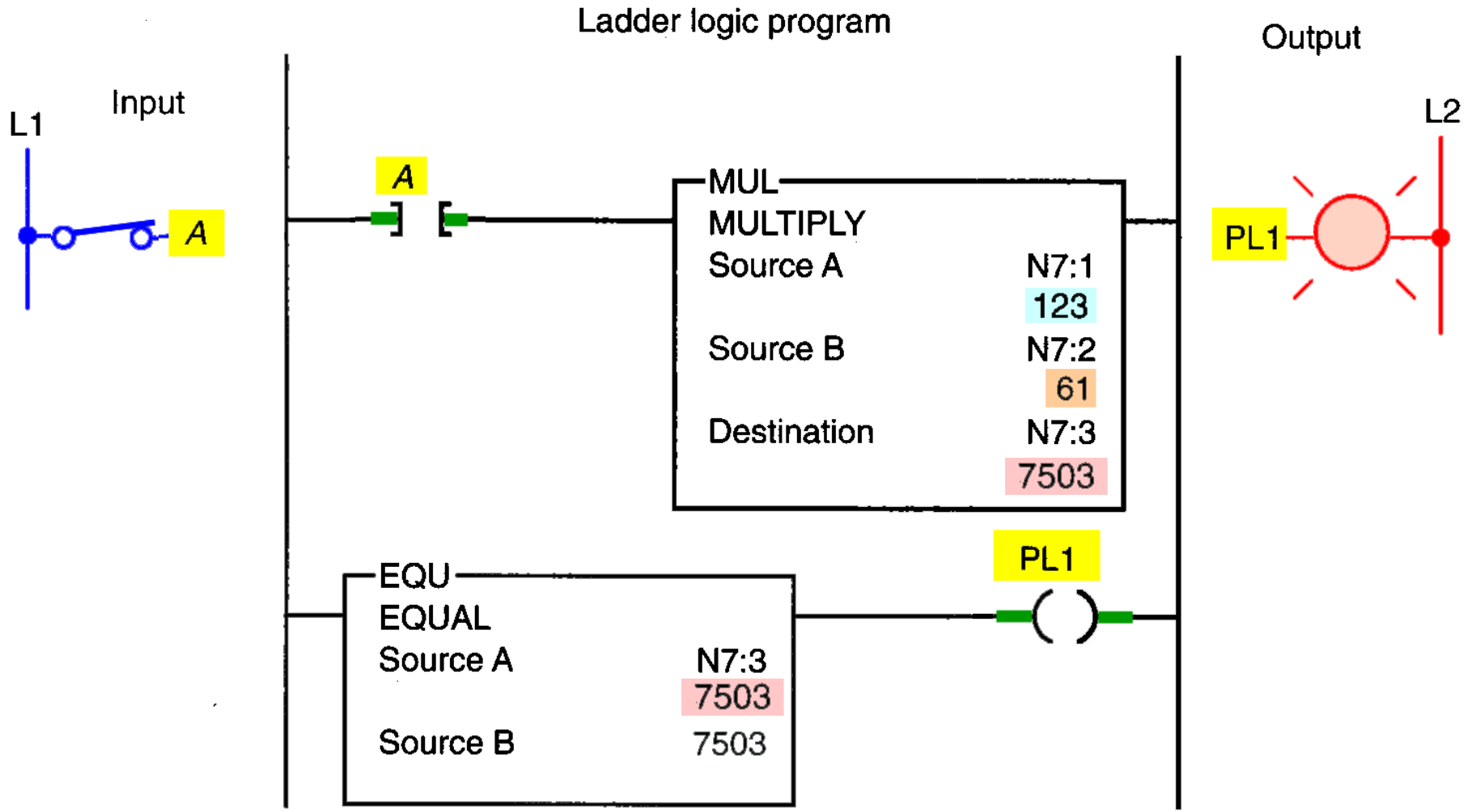
MULTIPLY Instruction

When the rung is true, the data in source A (the constant, 20) will be multiplied by the data in source B (the accumulated value of counter C5:10), with the result being placed in the destination N7:2.



As with previous math instructions, sources A and B can be values (constants) or addresses that contain values.

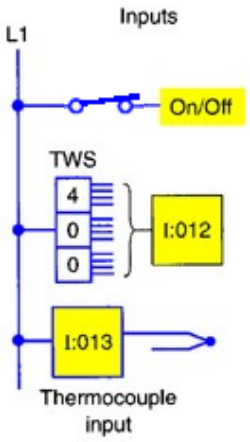
Simple MULTIPLY Program

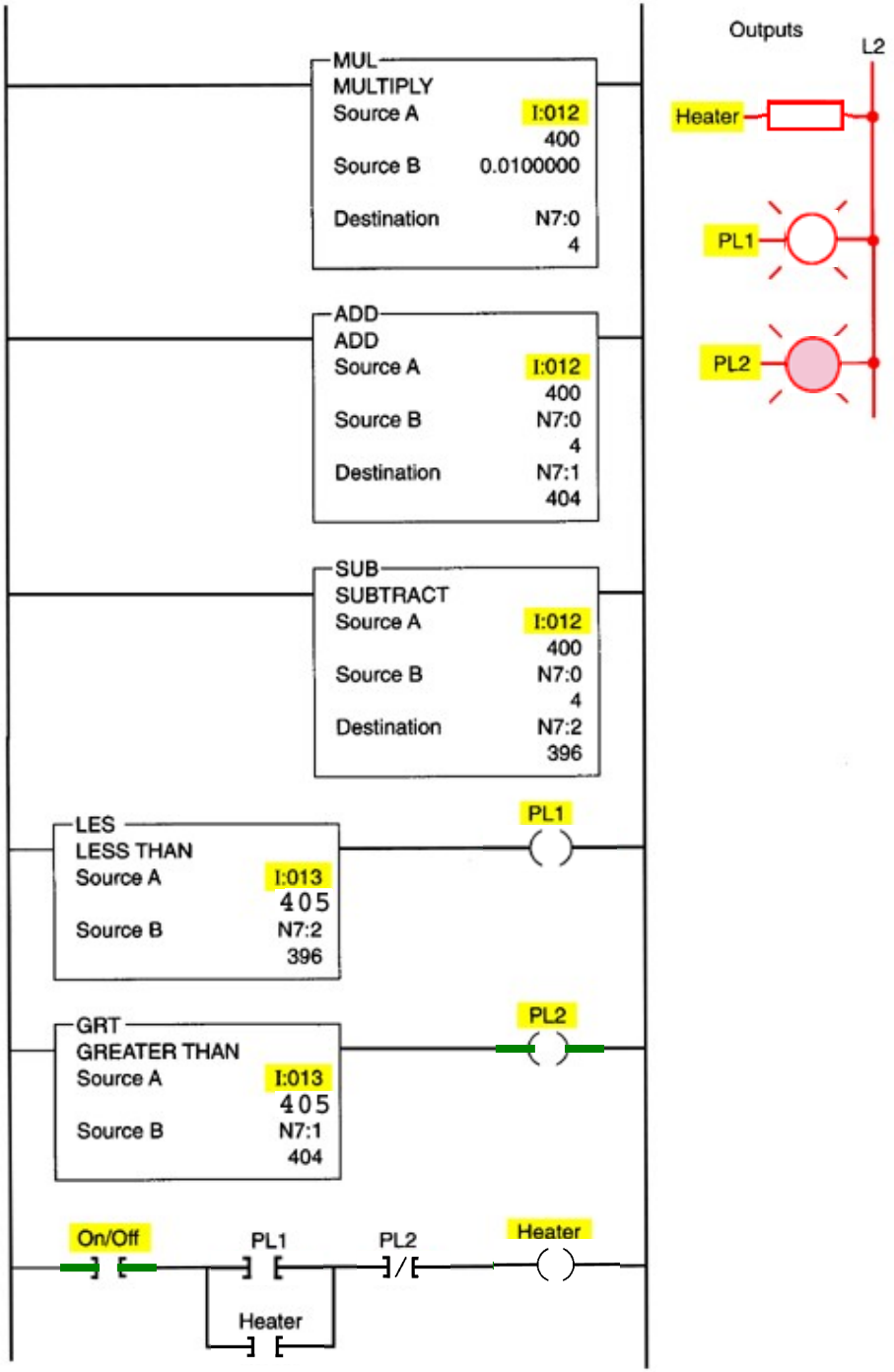


Oven Temperature Control Program

The program of the following side shows how the MULTIPLY instruction is used as part of an oven temperature control program. In this program, the PLC calculates the upper and lower *deadband* or off/on limits about the set point. The upper and lower limits are set automatically at ± 1% regardless of the set-point value. The set-point temperature is adjusted by means of the thumbwheel switch. An analog thermocouple interface module is used to monitor the current temperature of the oven. In this example, the set-point temperature is 400 °F. Therefore, the electric heaters will be turned on when the temperature of the oven drops to less than 396 °F and stay on until the temperature rises above 404 °F. If the set-point is changed to 100 °F, the deadband remains at ± 1%, with the lower limit being 99 °F and the upper limit being 101 °F. The number stored in word N7:1 represents the upper temperature limit, while the number stored in word N7:2 represents the lower limit.

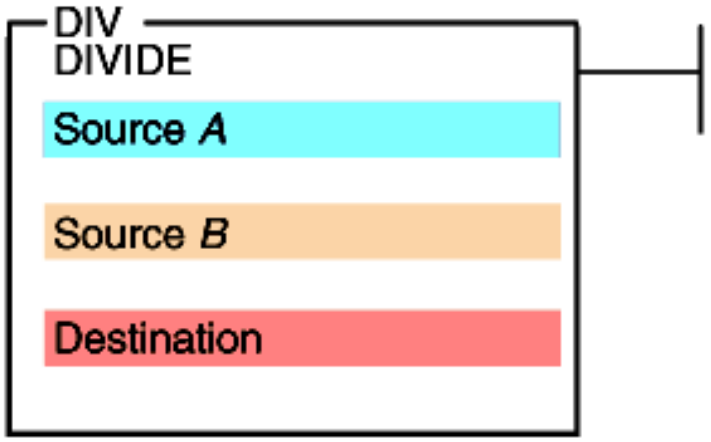
Oven Temperature Control Program





DIVIDE Instruction

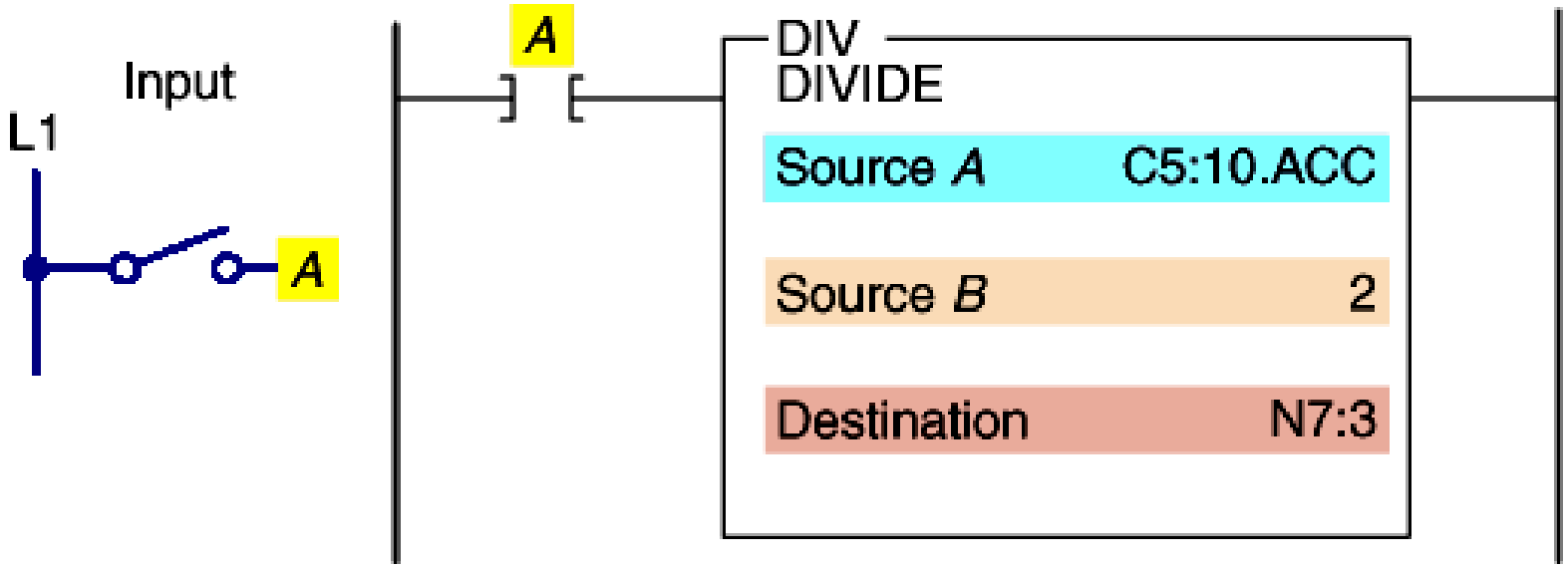
The DIVIDE instruction divides the value in source A by the value in source B and stores the result in the destination and math register.



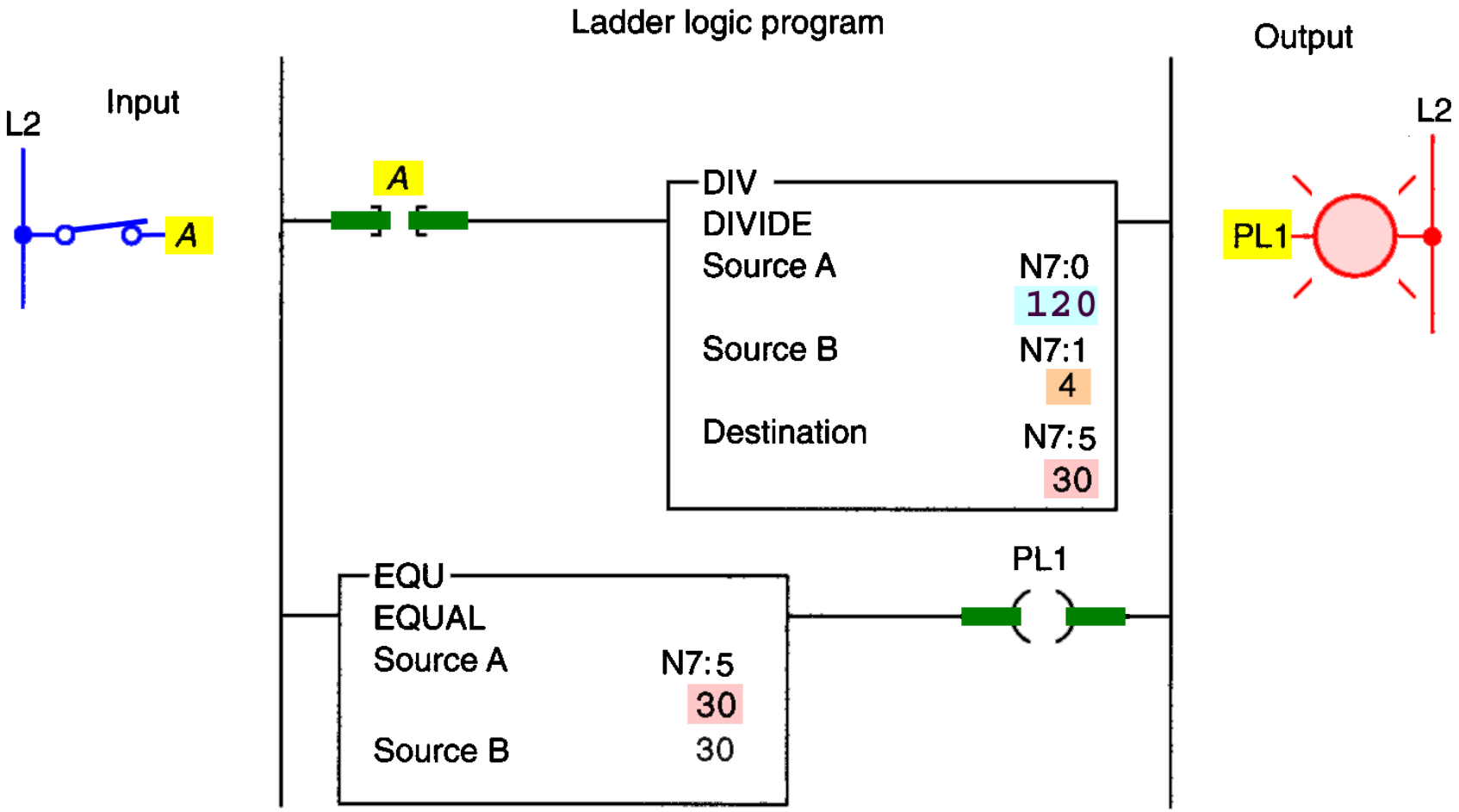
If the reminder is 0.5 or greater, a round-up occurs in the integer destination. The value stored in the math register consists of the unrounded quotient (placed in the most significant word) and the remainder (placed in the least significant word). Some larger PLC's support the use of floating-decimal as well as integer values.

DIVIDE Instruction

When the rung is true, the data in source A (the accumulated value of counter C5:5) will be divided by the data in source B (the constant 2), with the result being placed in the destination N7:3.



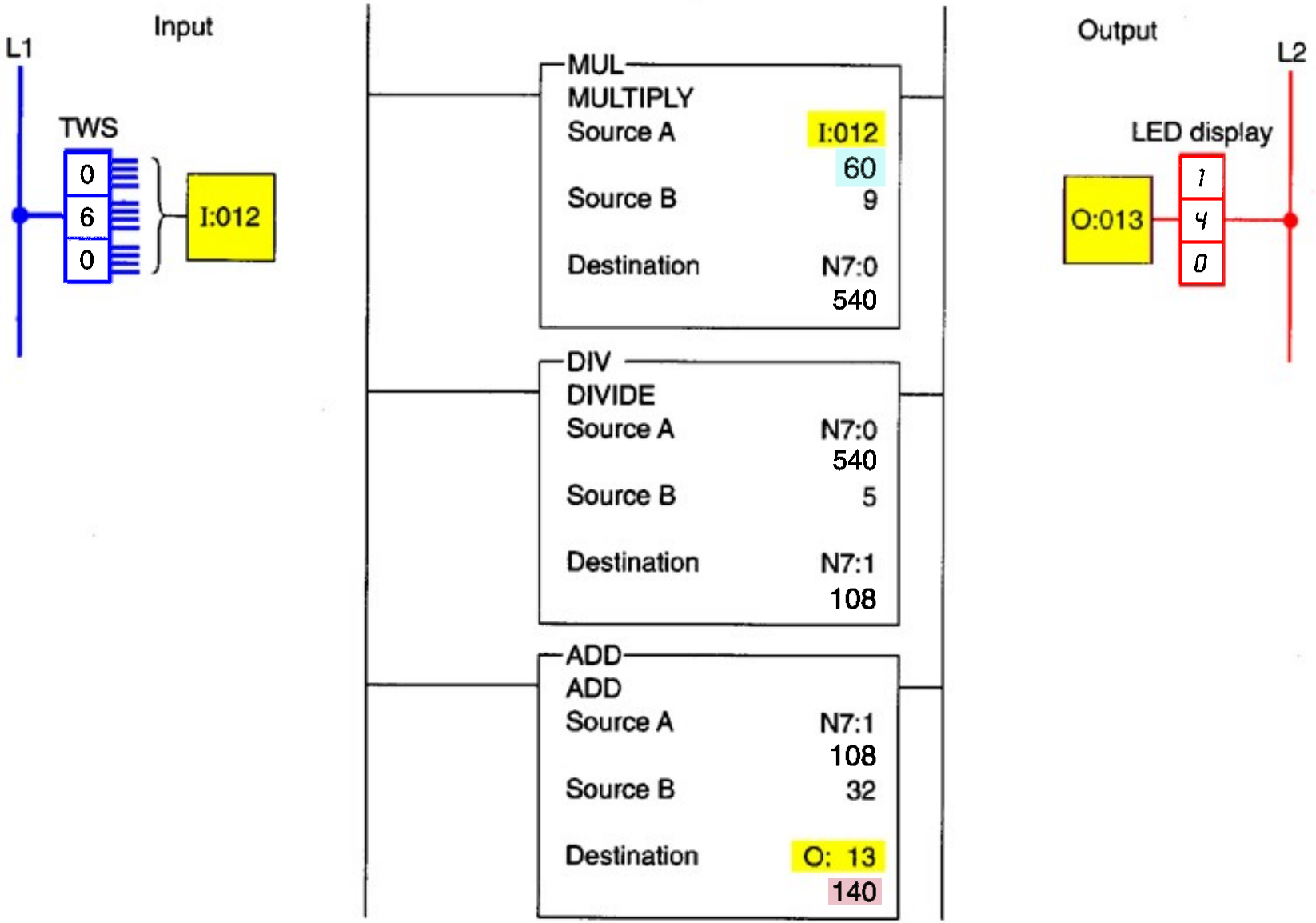
Simple DIVIDE Program



Converting ºC to ºF Program

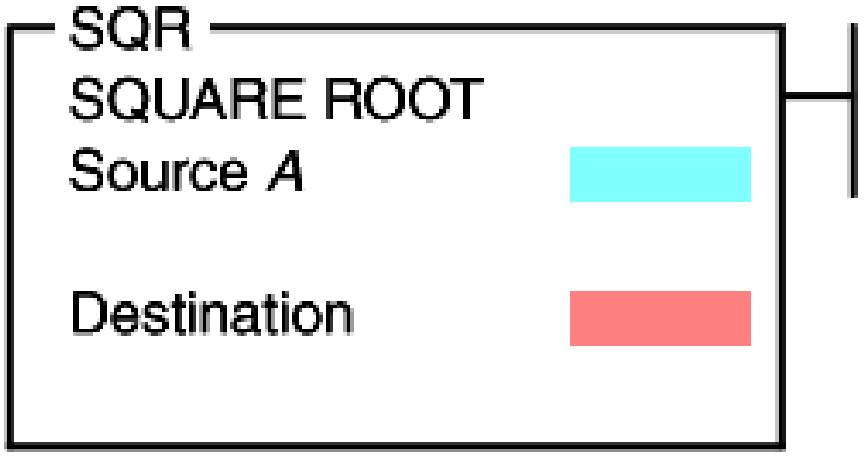
The program of the following side shows how the DIVIDE instruction is used as part of a program to convert Celsius temperature to Fahrenheit. In this application, the thumbwheel switch connected to the input module indicates Celsius temperature. The program is designed to convert the recorded Celsius temperature in the data table to Fahrenheit values for display. The formula: F = (9/5 x C) + 32 forms the basis for the program. In this example, a current temperature reading of 60 °C is assumed. The MULTIPLY instruction multiplies the temperature (60°C) by 9 and stores the product (540) in address N7:0. Next, the DIVIDE instruction divides 5 into the 540 and stores the answer (108) in address N7:1. Finally, the ADD instruction adds 32 to the value of 108 and stores the sum (140) in address O:13. Thus 60°C = 140°F.

Converting ºC to ºF Program



Square Root (SQR) Instruction

The Square Root (SQR) instruction is an output instruction that determines the square root of a number.



When rung conditions are true, the square root instruction calculates the square root of the number stored at source A and places the answer in the destination.

1. Math instructions are all input instructions. (True/False)

2. The rung containing an arithmetic function must be false to produce a result. (True/False)

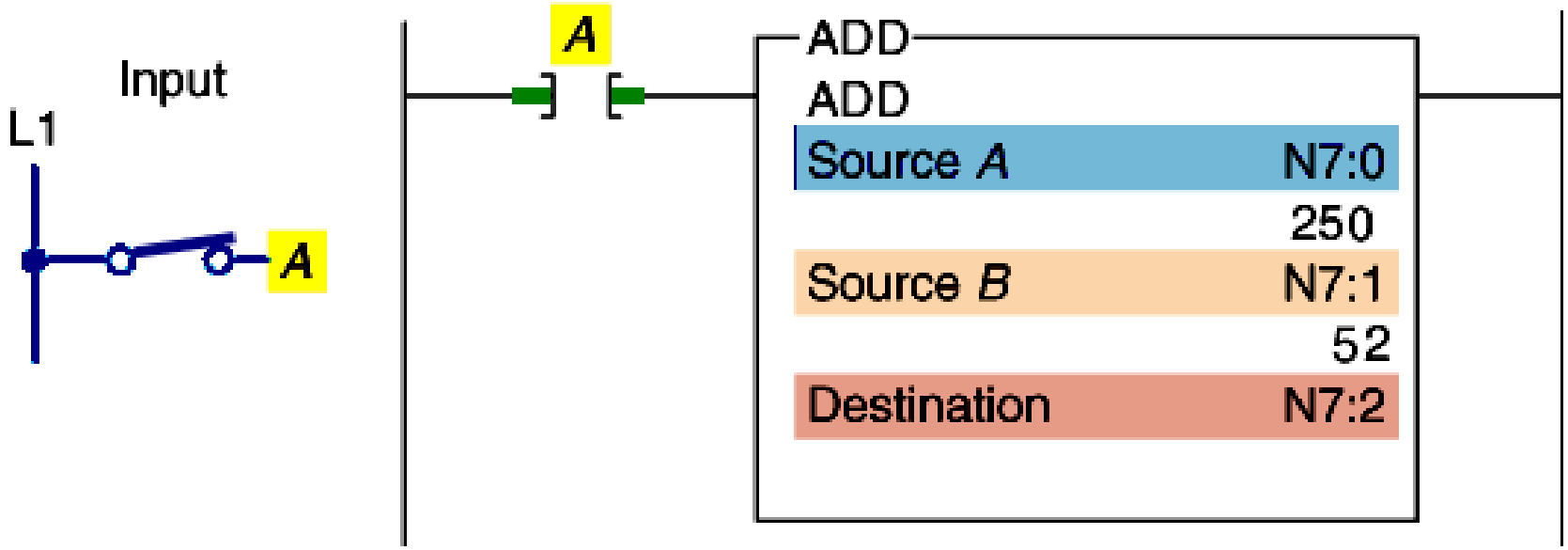
3. The arithmetic functions require the manipulation of single bits. (True/False)

4. For an Allen-Bradley SLC-500, the result of the math instruction is stored at the Source A address. (True/False)

5. For the math operation shown, the value stored at N7:2 would be:

(a) 50 **(b) 72**

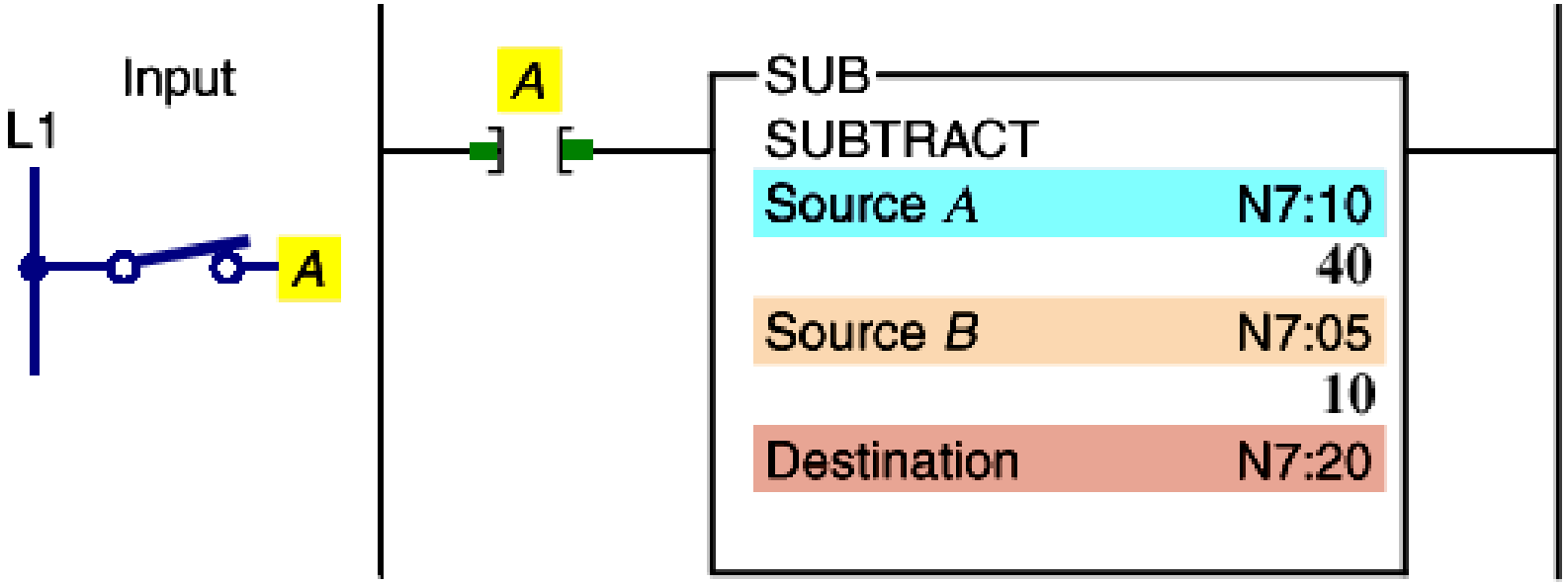
(c) 248 **(d) 302**



6. For the math operation shown, the value stored at N7:2 would be:

(a) 30 **(b) 50**

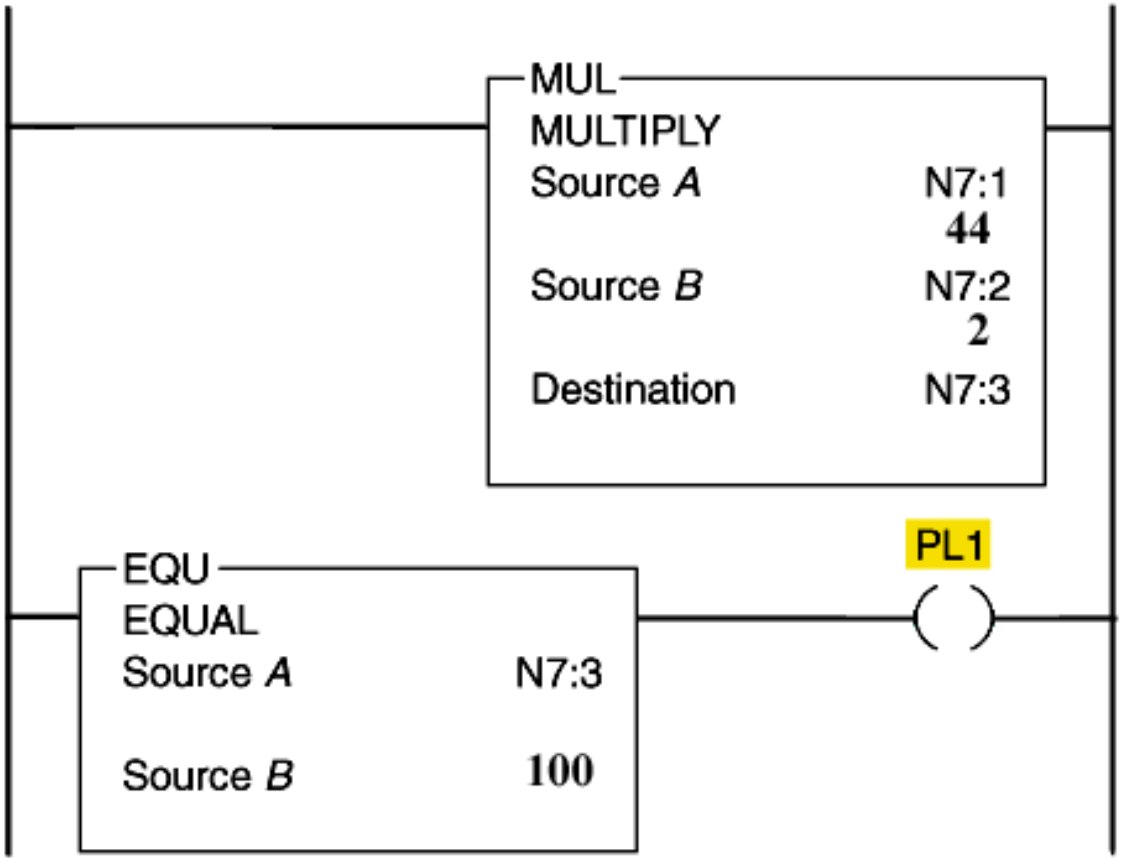
(c) 15 **(d) 25**



7. For the program shown the status of PL1 would be:

(a) 44 **(b) 88**

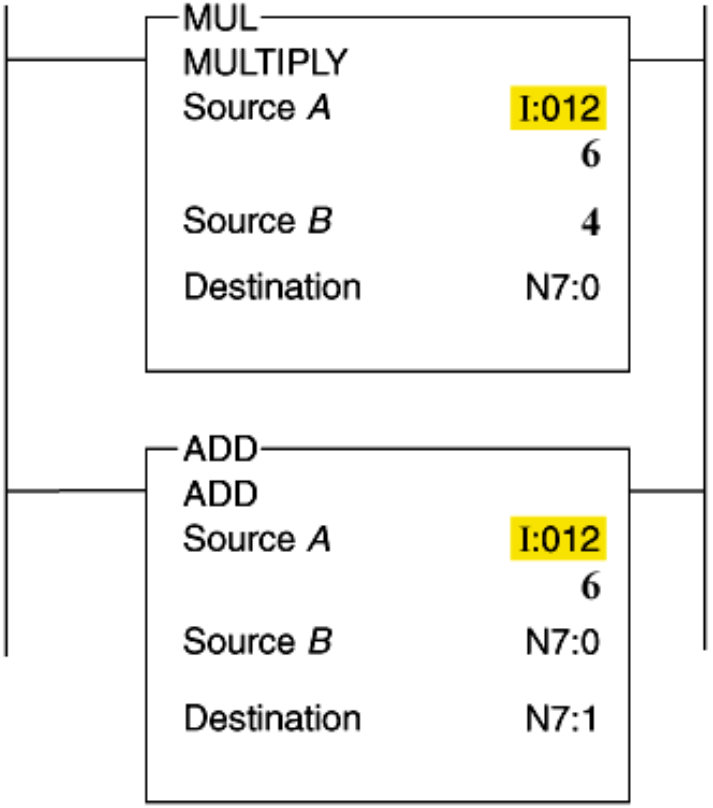
(c) false **(d) true**



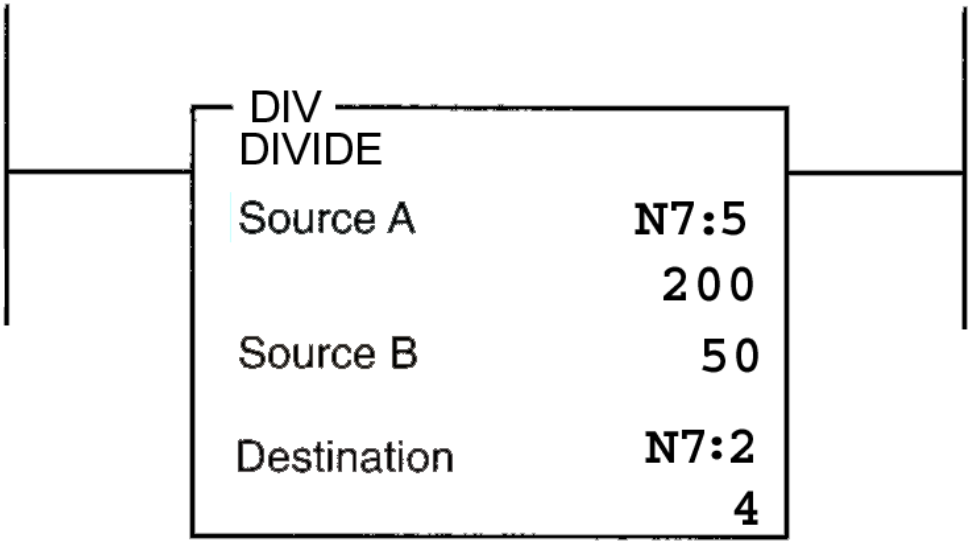
8. For the program shown the value stored in N7:1 would be:

(a) 30 **(b) 24**

(c) 40 **(d) 12**



9. For the math operation shown, if the value of source A increases to 400:



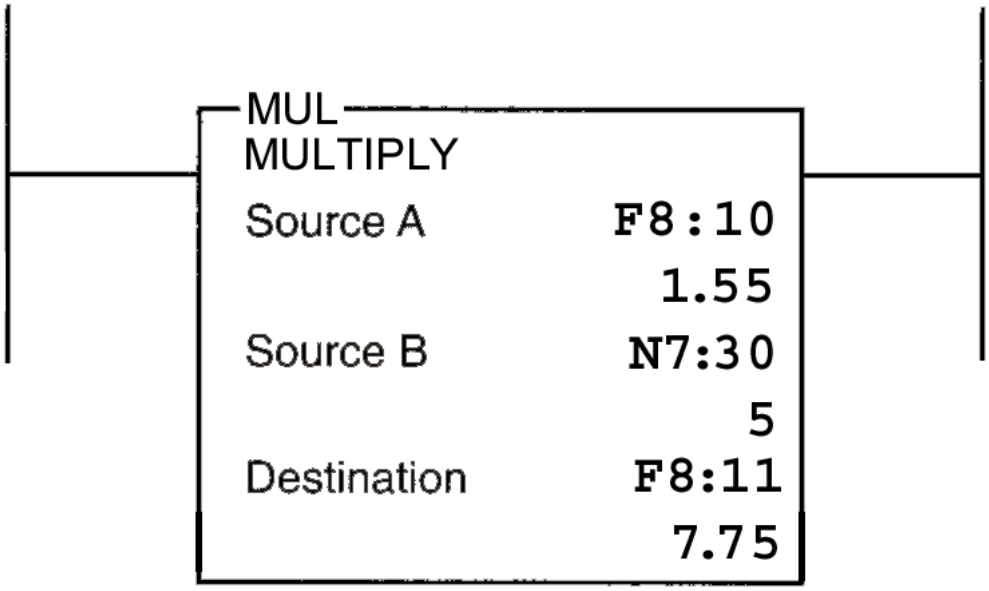
a. the value stored at source B will decrease to 25

b. the value stored at source B will increase to 100

c. the value stored in the destination will increase to 8

d. the value stored in the destination will decrease to 2

10. **For the math operation shown, assume the destination address is changed from the floating decimal to an integer file. As a result the number stored in the destination address would be:**



a. 7.75

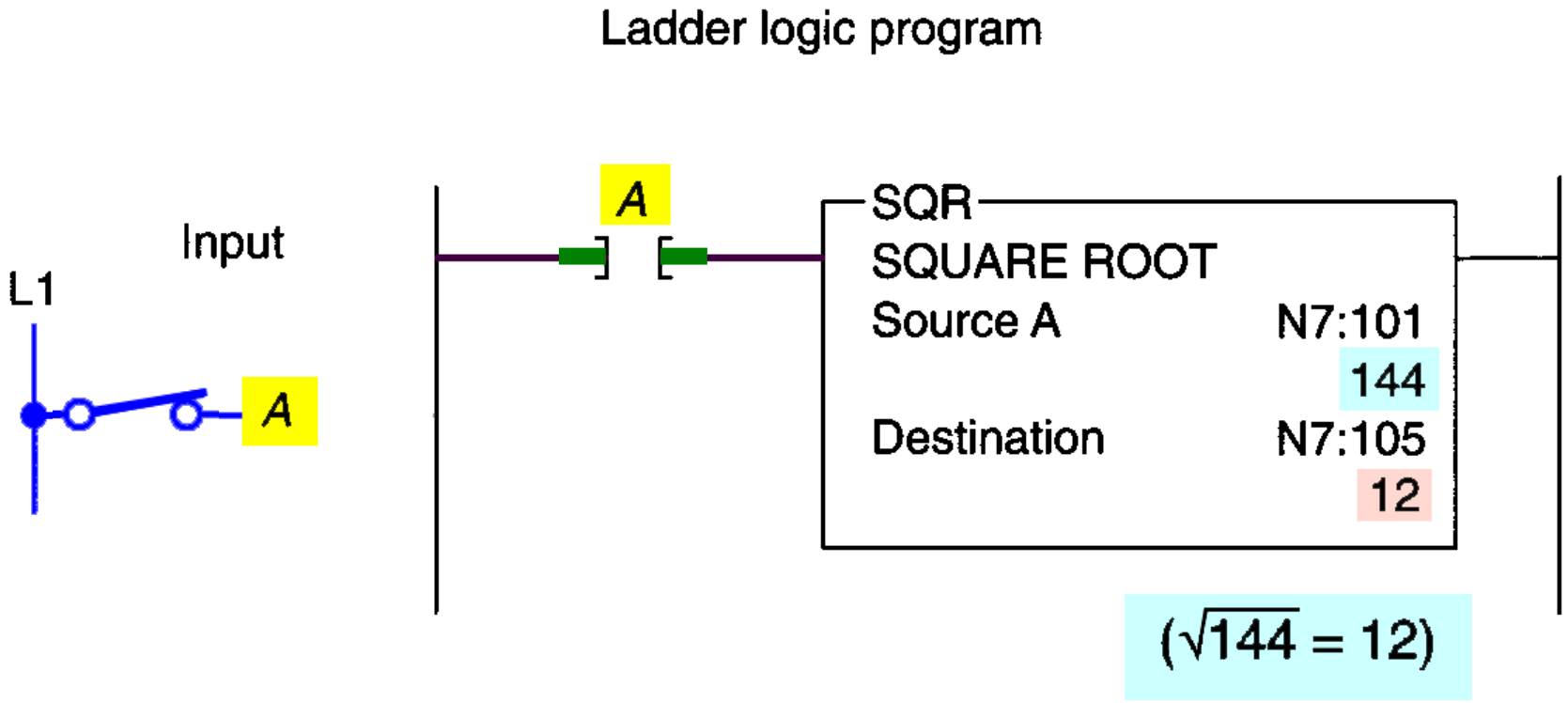
b. 8

c. 7

d. 6.75

Square Root (SQR) Instruction

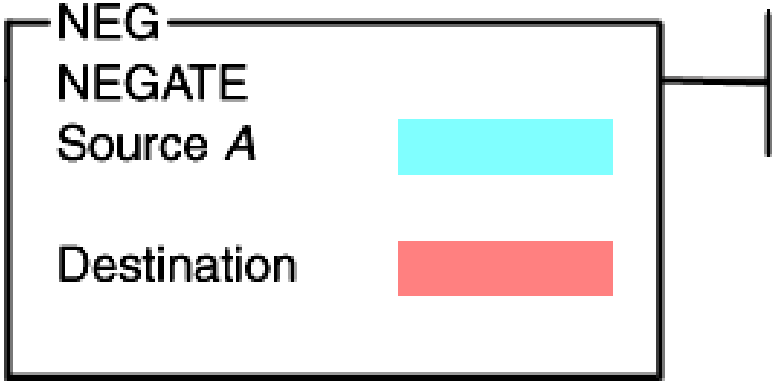
When the rung is true, the square root of the number in source A, N7:101 (144), will be calculated and the answer (12) placed in the destination, N7:105.



If the value of the source is negative, the instruction will store the square root of the absolute value of the source at the destination.

Negate Instruction (NEG)

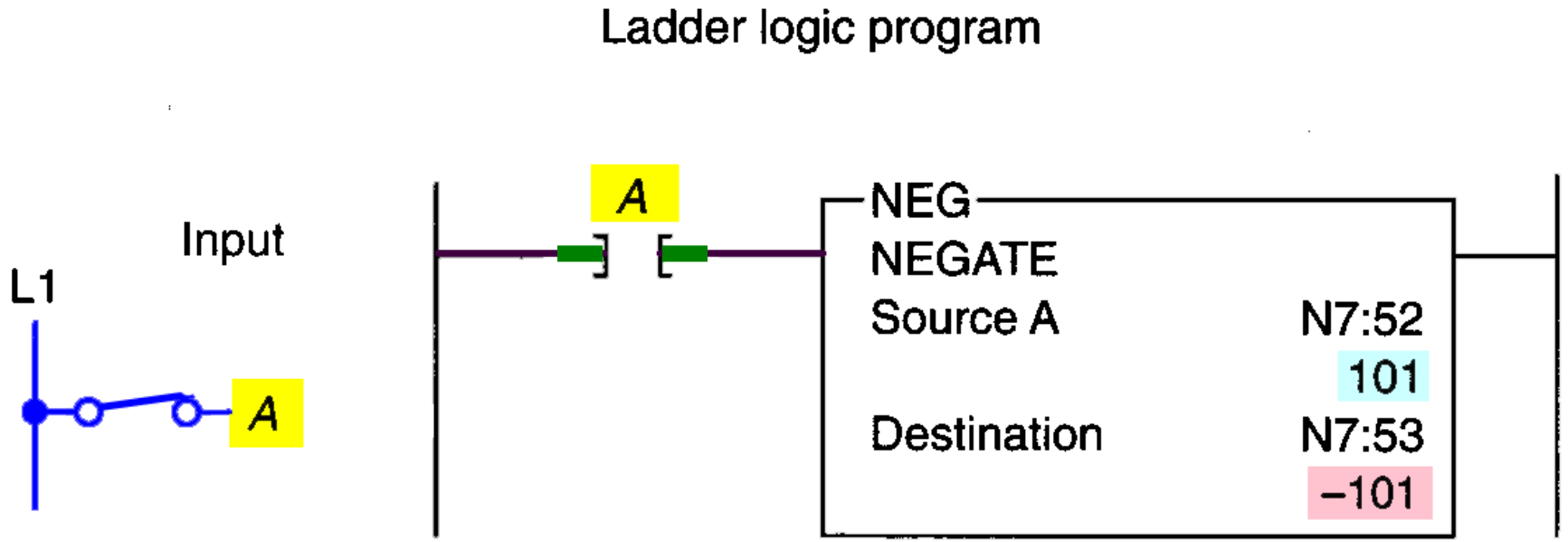
The Negate (NEG) instruction is an output instruction that negates (changes the sign of) of a value.



When rung conditions are true, the negate instruction changes the sign of source A and stores the result in the destination.

Negate Instruction (NEG)

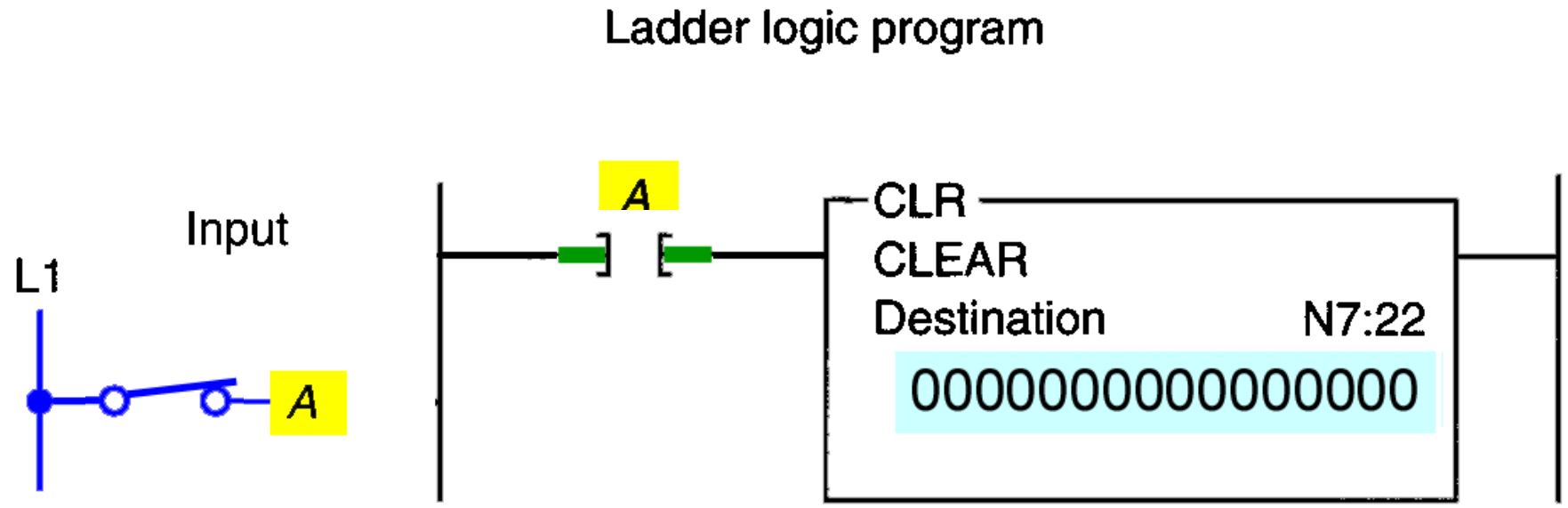
When the rung is true, the sign of the number in source A, N7:52 (101), will be changed and the result (-101) placed in the destination, N7:53.



Positive numbers will be stored in straight binary format, and negative numbers will be stored in two's complement.

Clear (CLR) Instruction

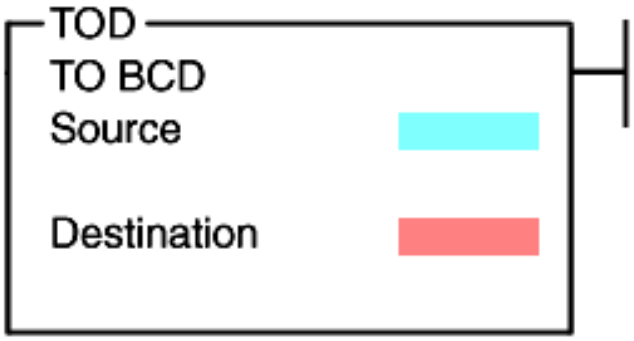
The Clear (CLR) instruction is used to set the destination value of a word to 0.



When the rung is true, the clear instruction changes the value stored in the destination address, N7:22, to 0.

Convert To BCD (TOD) Instruction

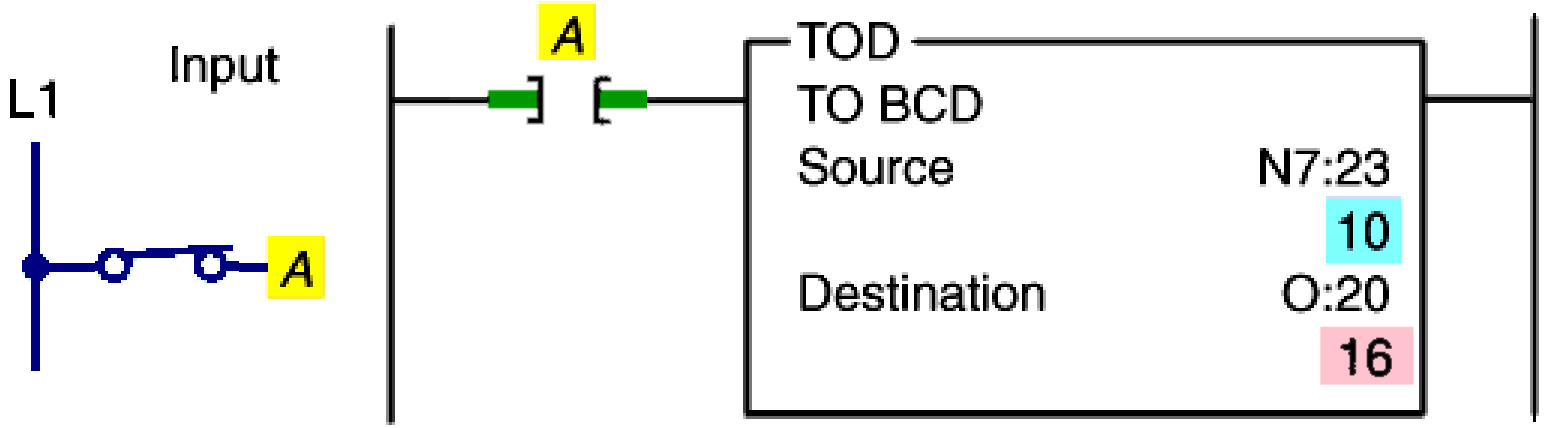
The *convert to* BCD (TOD) output instruction is used to convert 16-bit integers into binary coded decimal (BCD) values.



When rung conditions are true, the TOD instruction converts the 16-bit integer stored at source A to BCD and places the answer in the destination. 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.

Convert To BCD (TOD) Instruction

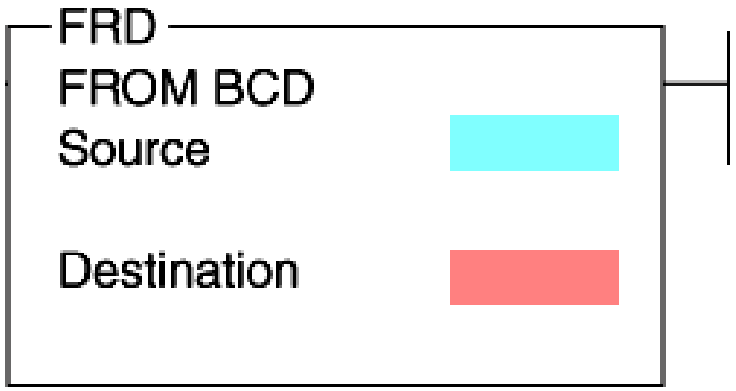
When input A is true, the TOD instruction will convert the binary bit pattern at the source address, N7:23, into a BCD bit pattern of the same decimal value at the destination address O:20



The source displays the value 10, which is the correct decimal value; however, the destination displays the value 16. Since the processor interprets all bit patterns as binary, the value 16 is the binary interpretation of the BCD bit pattern. The bit pattern for 10 BCD is the same as the bit pattern for 16 binary.

Convert From BCD (FRD) Instruction

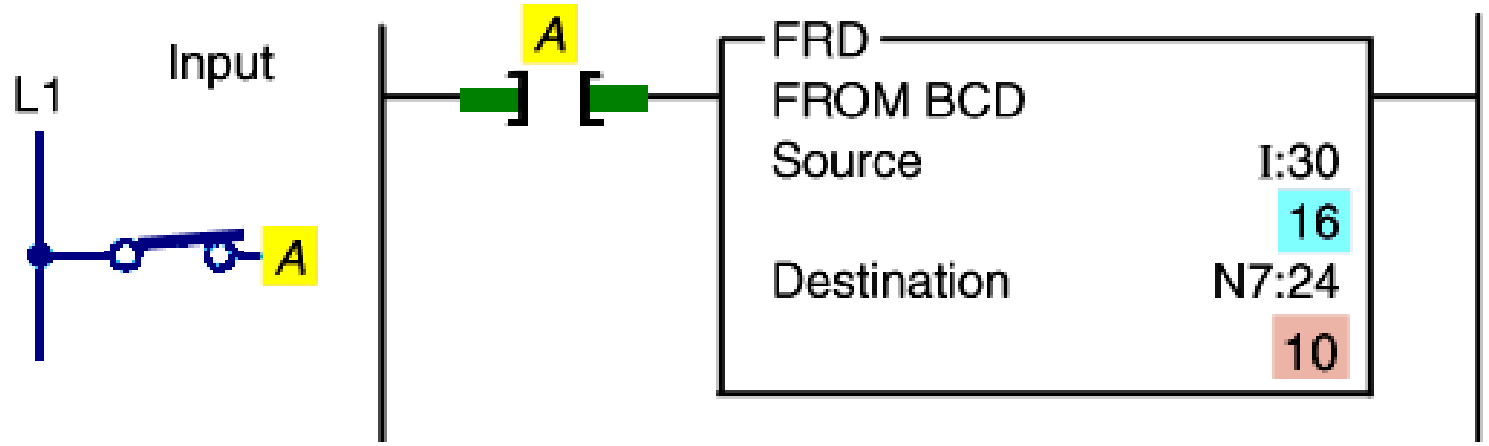
The *convert from* BCD (FRD) output instruction is used to convert binary coded decimal (BCD) values to integer values.



When rung conditions are true, the FRD instruction converts the BCD value to the equivalent integer value and stores the converted value in the destination. 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.

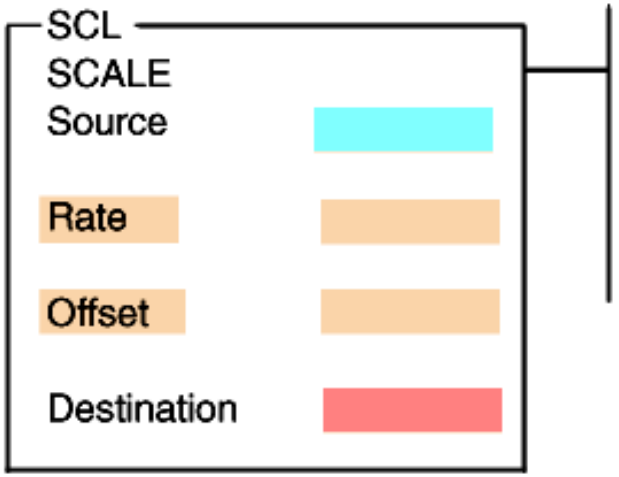
Convert From BCD (FRD) Instruction

When input A is true, the FRD instruction will convert the BCD bit pattern stored at the source address, I:30, into a binary bit pattern of the same decimal value at the destination address, N7:24.



Scale Data (SCL) Instruction

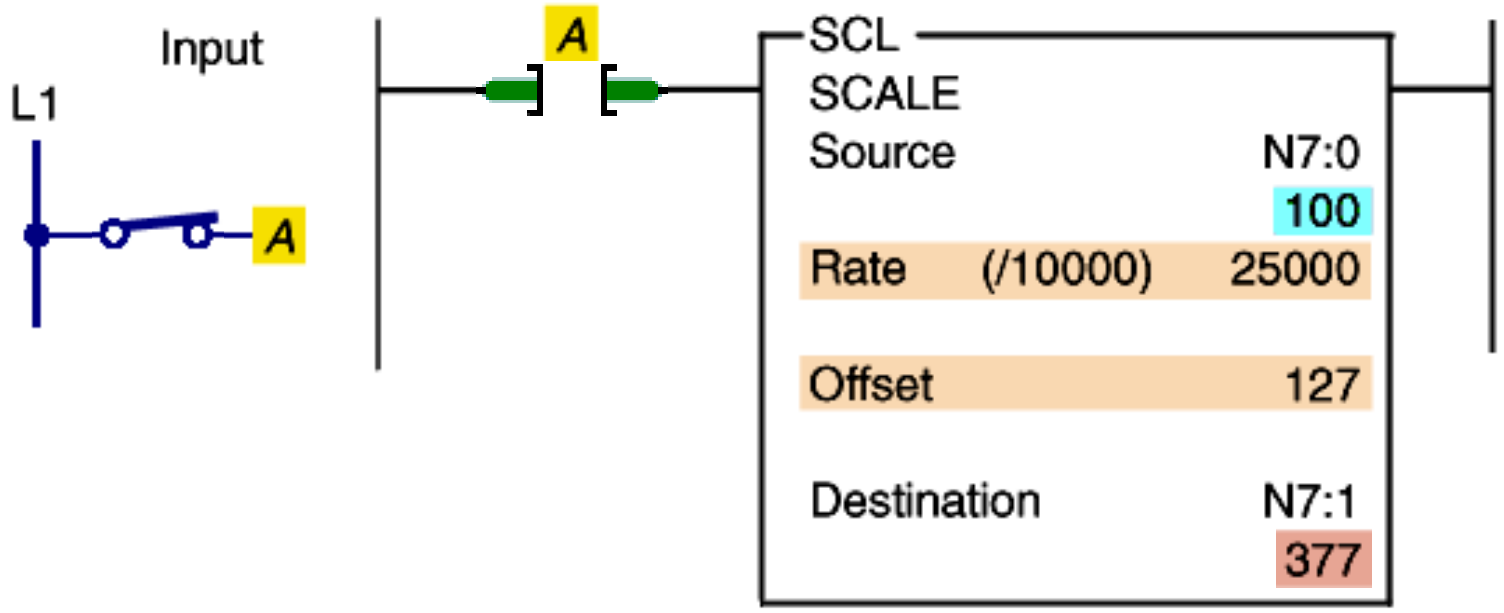
The *scale data* (SCL) output instruction is used to allow very large or very small numbers to be enlarged or reduced by the rate value.



When rung conditions are true, this instruction multiplies the source by a specified rate. The rounded result is added to an offset value and placed in the destination. You can use this instruction to scale data from your analog module and bring it into the limits prescribed by the process variable or another analog module. For instance, you can use the SCL instruction to convert a 4-20 mA input signal to a PID process variable, or, to scale an analog input to control an analog output.

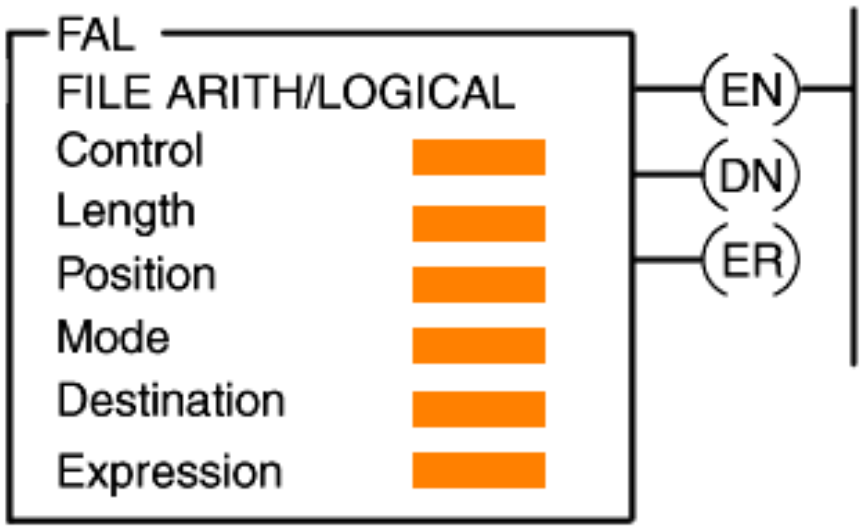
Scale Data (SCL) Instruction

When input A is true, the number 100 stored at the source address, N7:0, is multiplied by 25,000, divide by 10,000, and added to 127. The result is placed in the destination address, N7:1.



File Arithmetic And Logic (FAL) Instruction

Basic file arithmetic functions include file add, file subtract, file multiply, file divide, file square root, file convert from BCD, and file convert to BCD.



The *file arithmetic and logic* (FAL)is an output instruction that combines an arithmetic operation with file transfer. The Expression specifies the arithmetic operation.

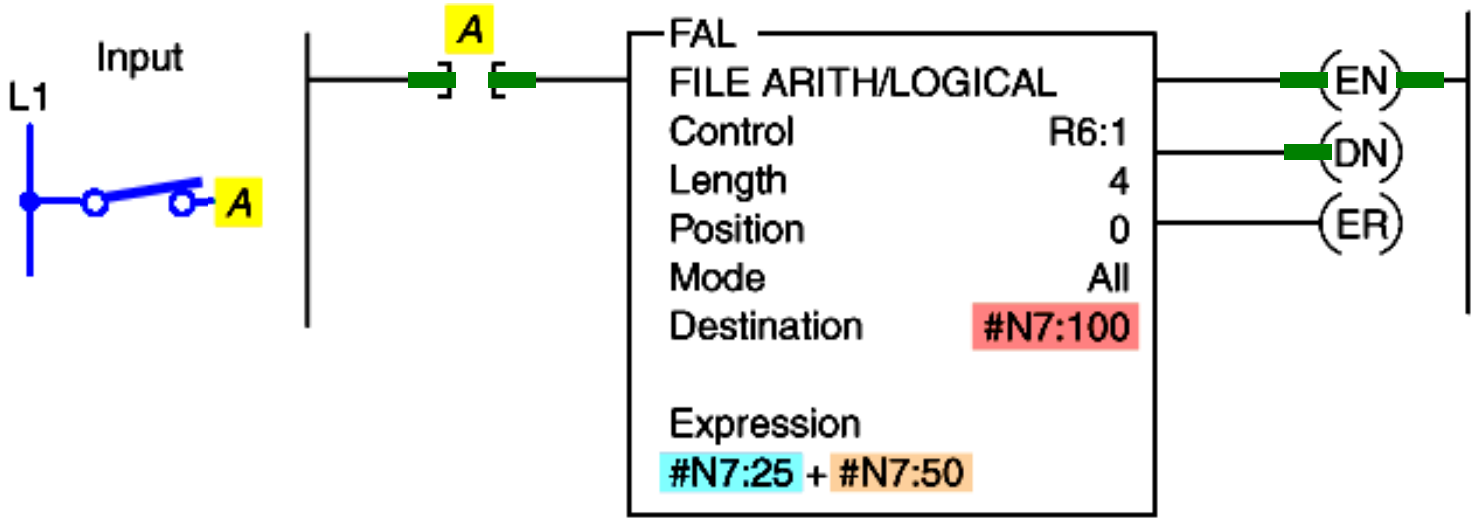
EN – Enable bit, set when the FAL is enabled

DN – Done bit, set when the instruction has operated on the last element

ER – Error bit, set when there is an overflow created by the expression.

File ADD Function Of The FAL Instruction

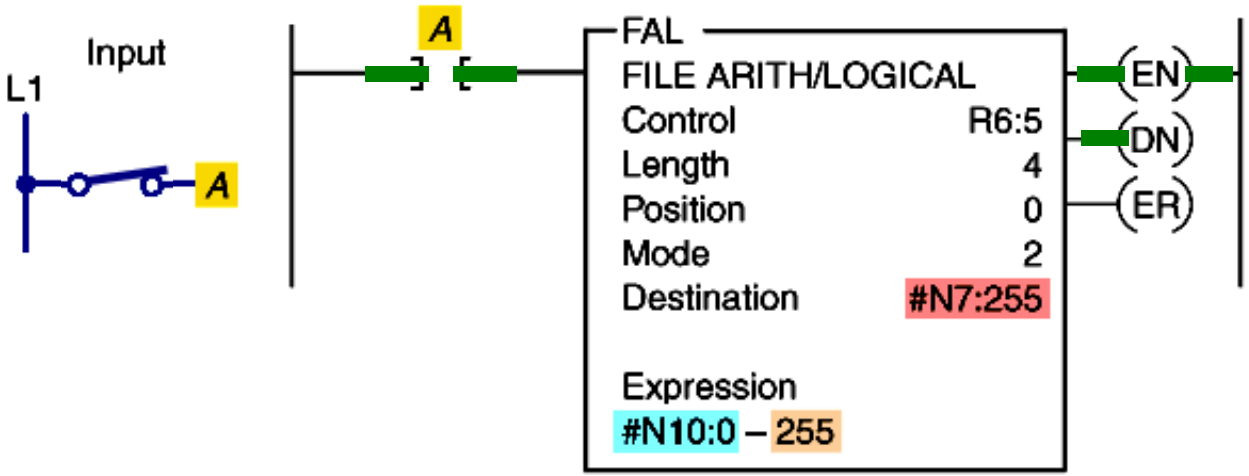
When input A is true, the expression tells the processor to add the data in file address N7:25 to the data stored in file address N7:50 and store the result in file address N7:100. The rate per scan is set at "all", so the instruction goes to completion in one scan.

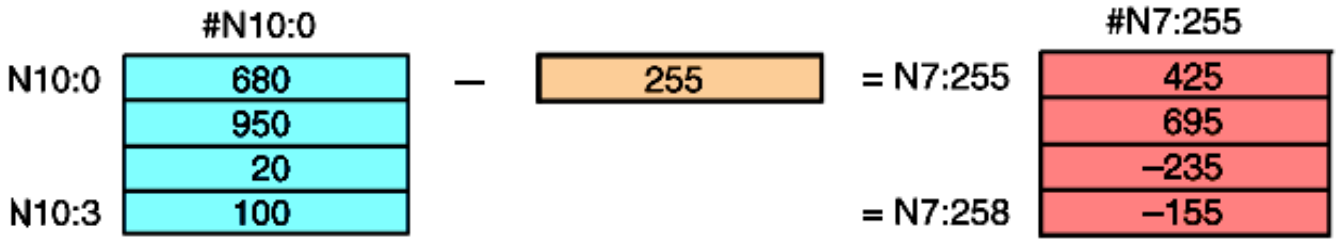


	#N7:25		#N7:50		#N7:100
N7:25	25	+ N7:50	50	= N7:100	75
	234	+	22		256
	1256	+	456		1712
N7:28	77	+ N7:53	100	= N7:103	177

File SUBTRACT Function Of The FAL Instruction

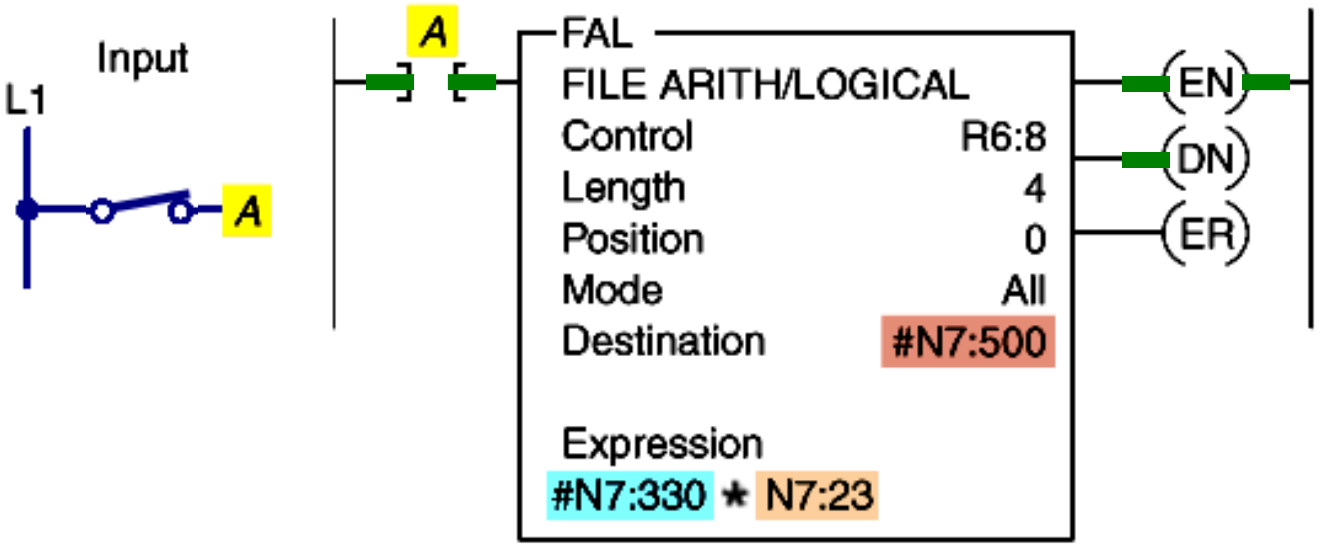
When input A is true, the processor subtracts a program constant (255) from each word of file address N10:0 and stores the result at the destination file address, N7:255. The rate per scan is set at 2, so it will take 2 scans from the moment the instruction goes true to complete its operation.

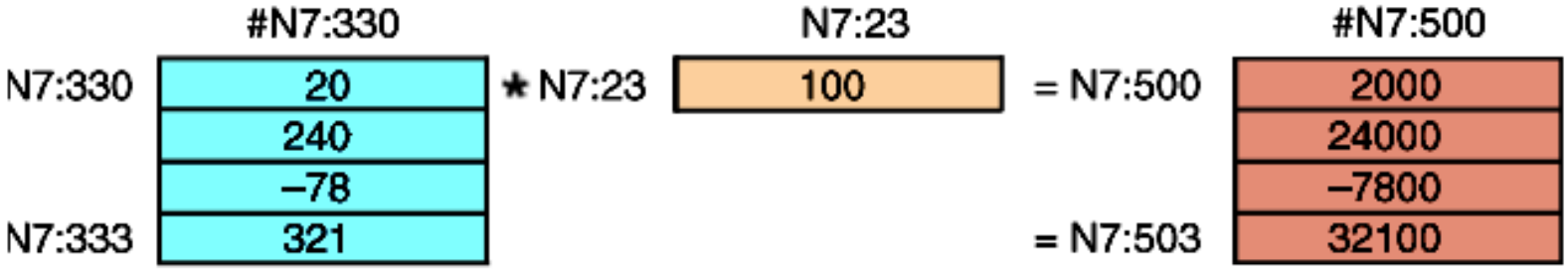




File MULTIPLY Function Of The FAL Instruction

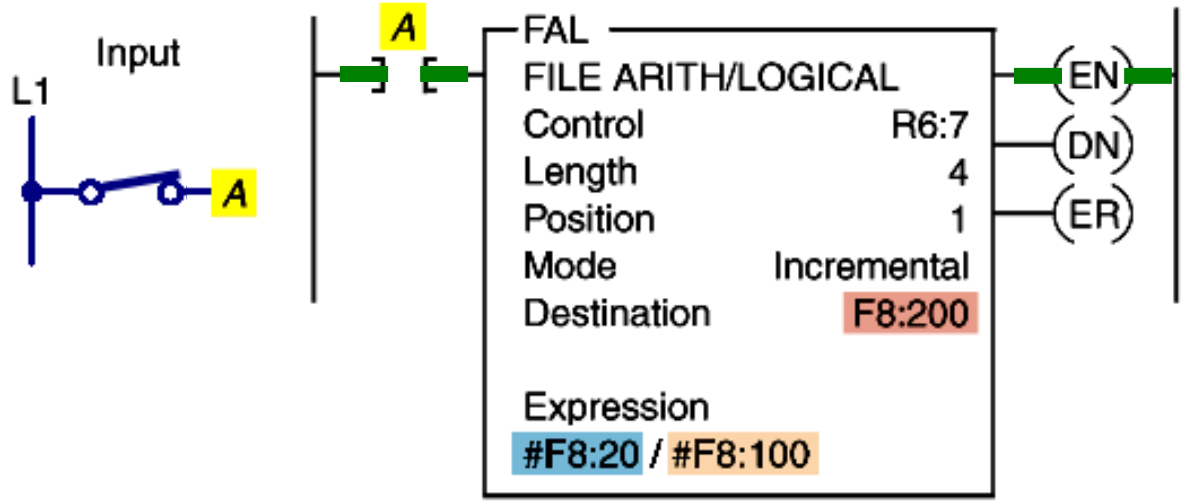
When input A is true, the data in file address N7:330 is multiplied by the data in element address N7:23 (100), with the result stored in file address N7:500.

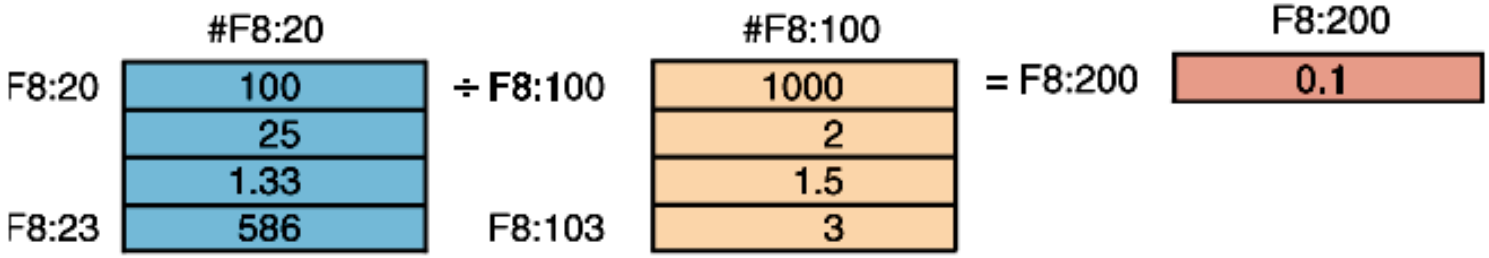




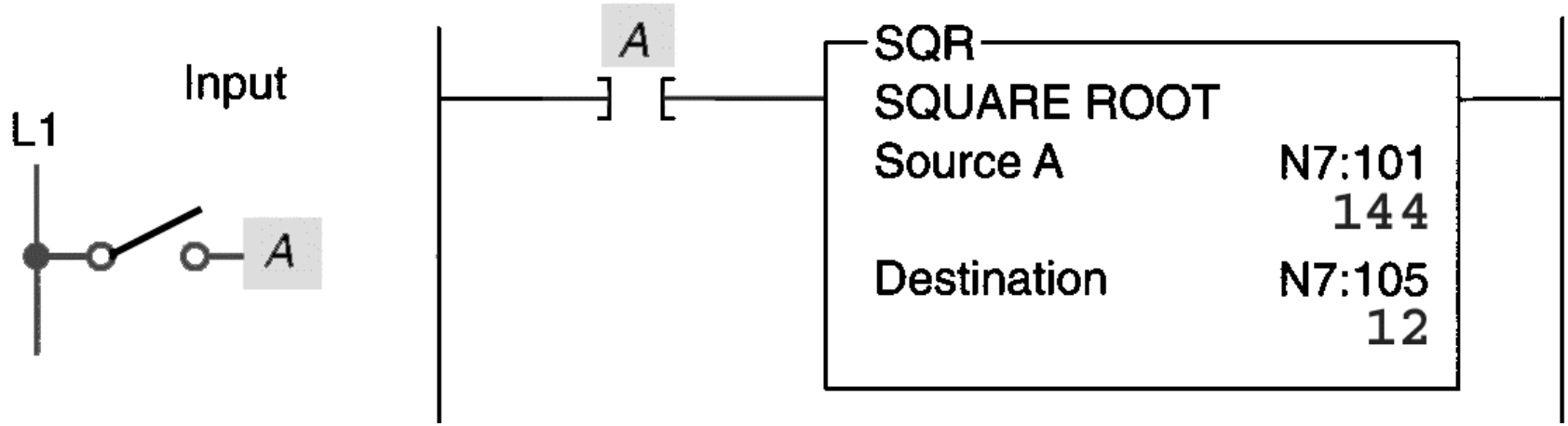
File DIVIDE Function Of The FAL Instruction

When input A is true, the data in file address F80:20 is divided by the data in element address F8:100, with the result stored in element address F8:200. The mode is incremental, so the instruction operates on one set of elements for each false-to-true transition of the instruction.





11. **For the math operation shown, under which of the following conditions would the number stored at the destination address be 6?**



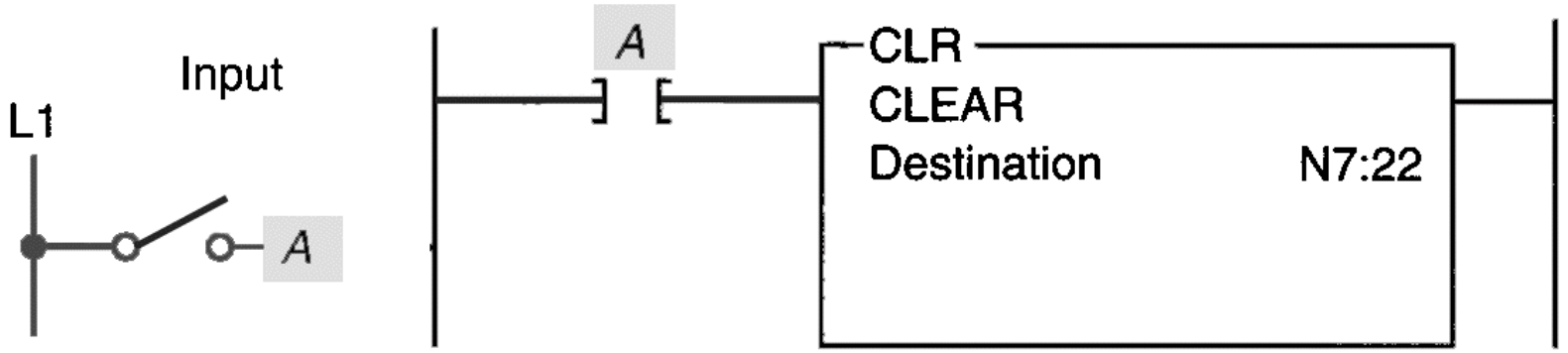
a. Input A true and source A equal to 6

b. Input A false and source A equal to 6

c. Input A true and source A equal to 36

d. Input A false and source A equal to 36

12. For the math operation shown, the number stored at N7:22:



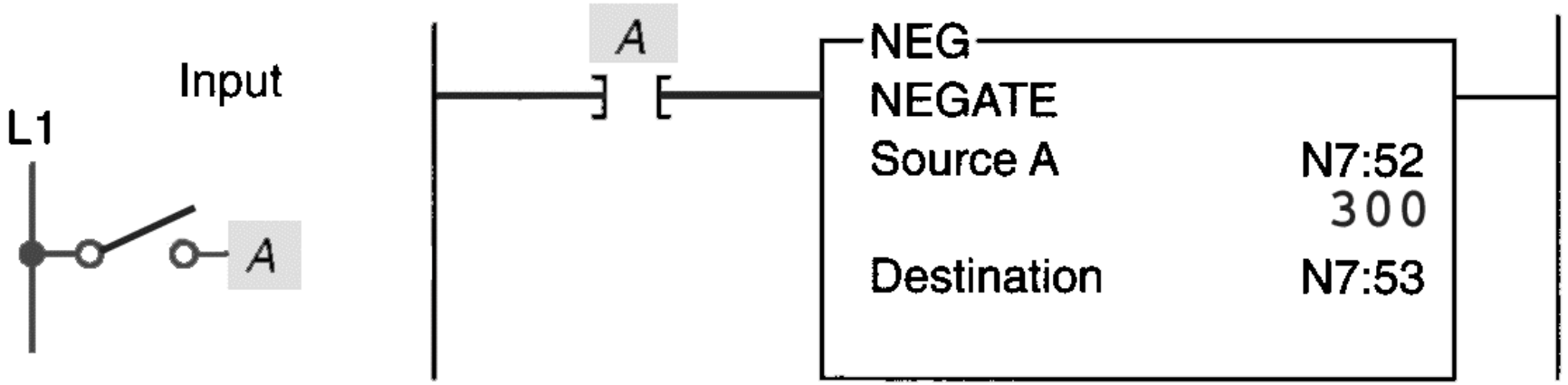
a. changes to 0 when input A is true

b. changes to 1 when input A is true

c. is always 0

d. is always 1

13. **For the rung shown, when input A is _____ the number stored at N7:53 will be _____.**



a. true, -300

b. false, -300

c. true, +300

d. false, +300

14. Which of the following parameters of the file arithmetic and logic (FAL) instruction specifies the arithmetic operation?

(a) Control

(b) Expression

(c) Mode

(d) Position

15. **Which of the following instruction is used to convert 16-bit integers into binary coded decimal values.**

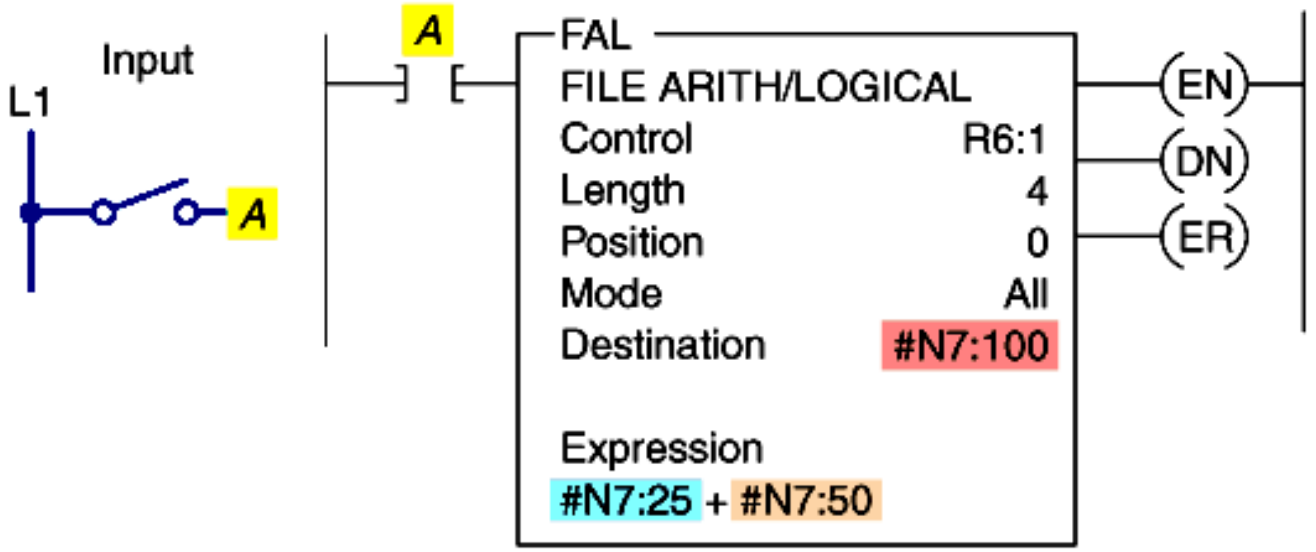
(a) FRD **(b) TOD**

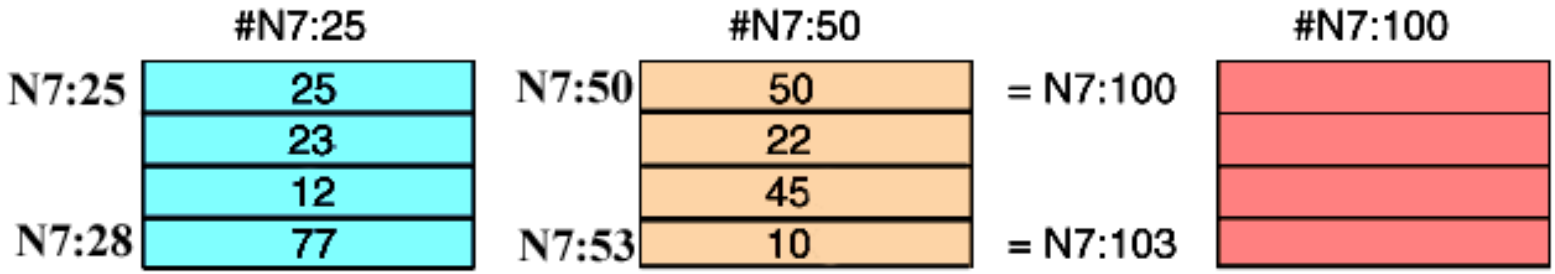
(c) SQR **(d) CLR**

16. **The scale data instruction is used to allow very large or very small numbers to be enlarged or reduced. (True/False)**

17. **For the rung shown, when input A is true the number stored at N7:101 will be _____.**

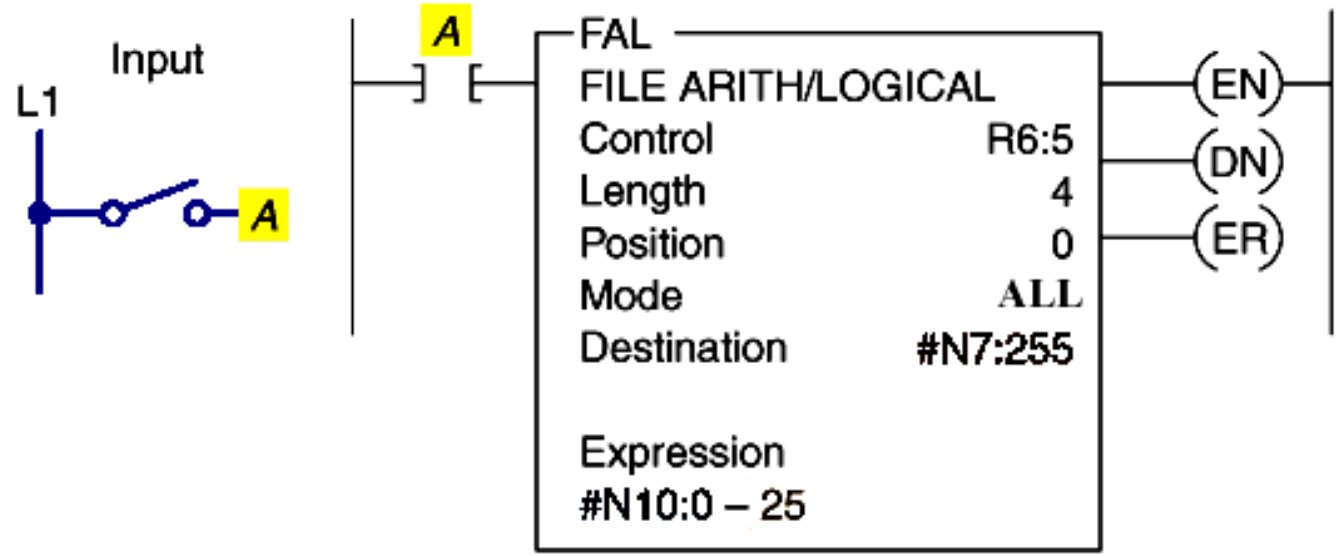
a. 88 **b. 57** **c. 45** **d. 75**

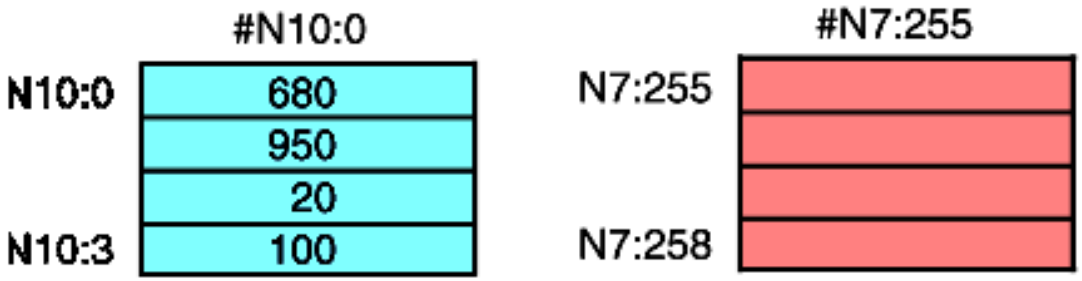




18. **For the rung shown, when input A is true the number stored at N7:258 will be _____.**

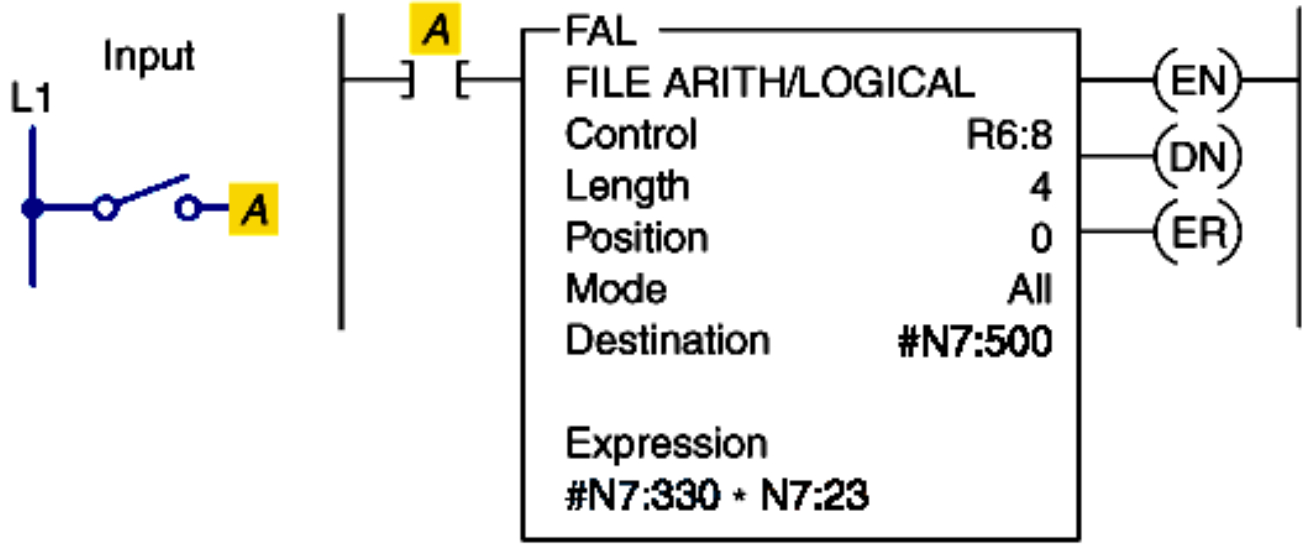
a. 75 **b. –5** **c. 925** **d. 402**





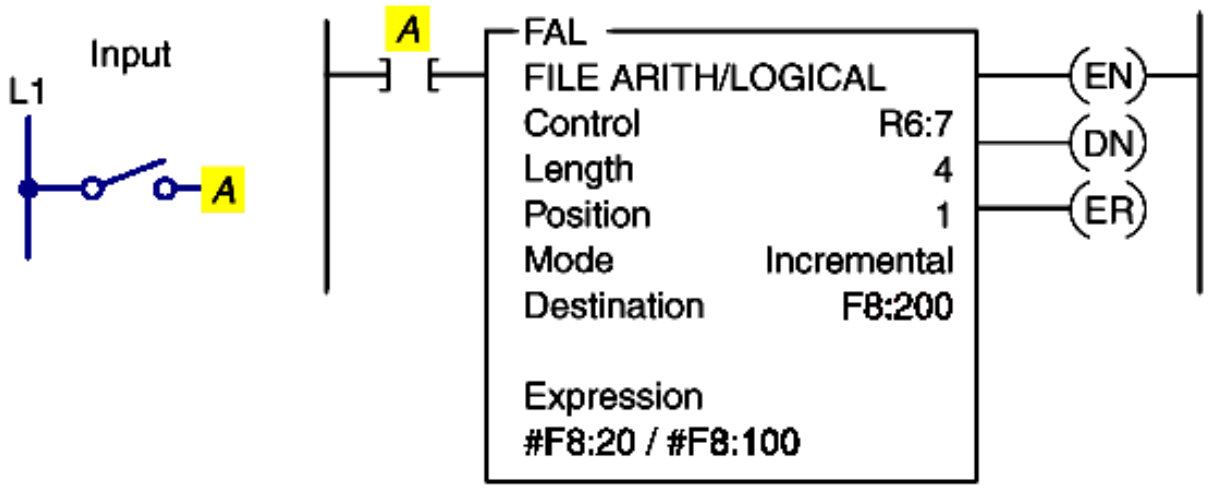
19. **For the rung shown, when input A is true the number stored at N7:501 will be _____.**

a. 100 **b. 120** **c. 350** **d. 160**



20. **For the rung shown, on the second false-to-true transition of input A (Position 2), the number stored at F8:200 will be _____.**

a. 40 **b. 30** **c. 60** **d. 12**



	#F8:20
F8:20	100
	200
	40
F8:23	800

	#F8:100
F8:100	2
	5
	10
F8:103	20

F8:200