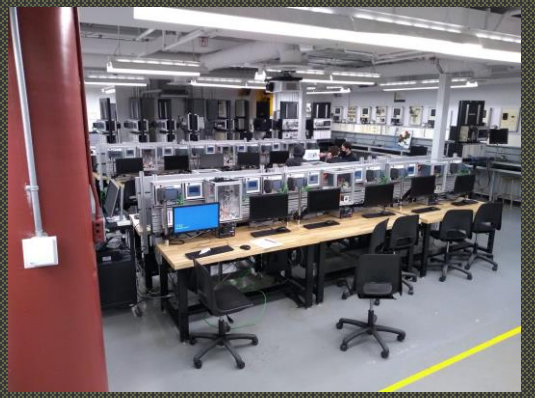


PROGRAMMABLE LOGIC CONTROLLERS
MENG 3500

HUMBER

MATH INSTRUCTIONS

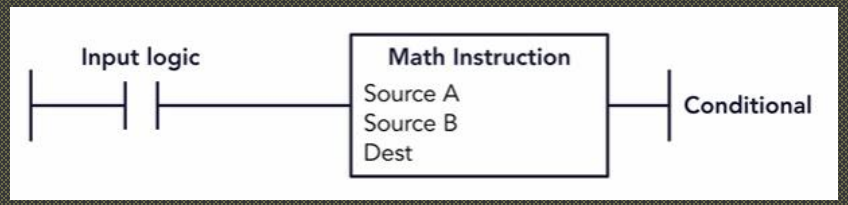
The PLC's mathematical processing capability enables it to execute arithmetic operations on values stored in memory words or registers.

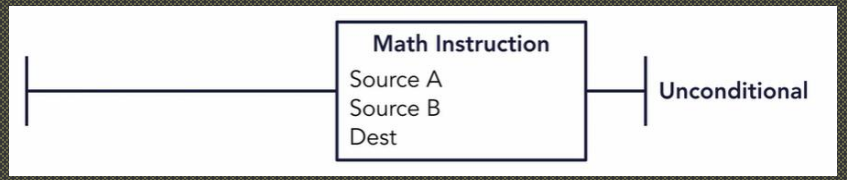
The specific math instructions available depend on the PLC model and its processor.

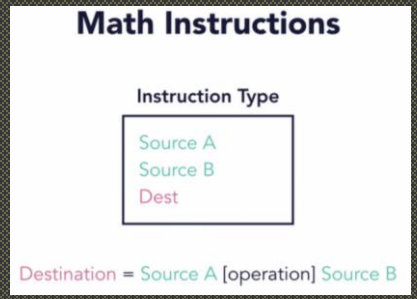
These math instructions typically require two input values, conduct the designated arithmetic operation, and then store the result in a designated memory location.

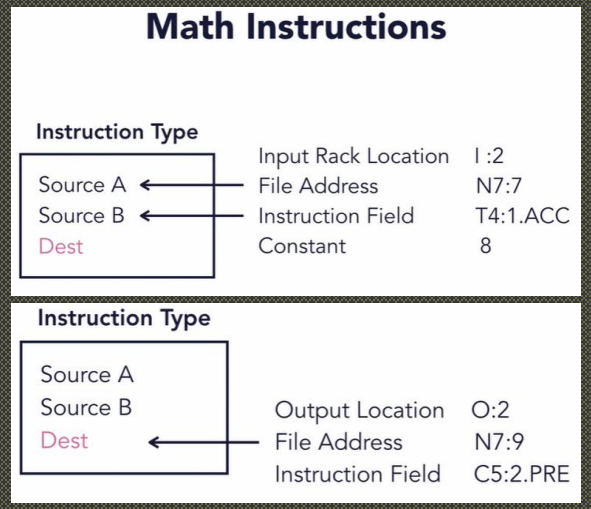
Among the fundamental mathematical functions carried out by CompactLogix PLCs are, but not limited to:

- Addition (ADD)
- Subtraction (SUB)
- Multiplication (MUL)
- Division (DIV)
- Square Root (SQR)
- Clear (CLR)







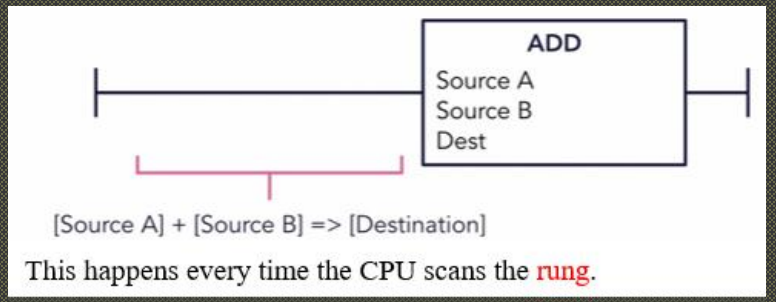


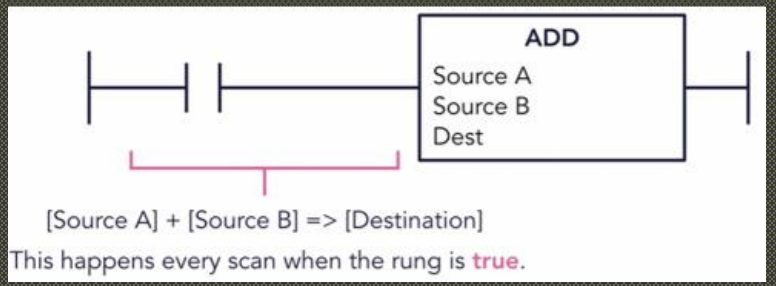
All these operations are executed as output instructions.

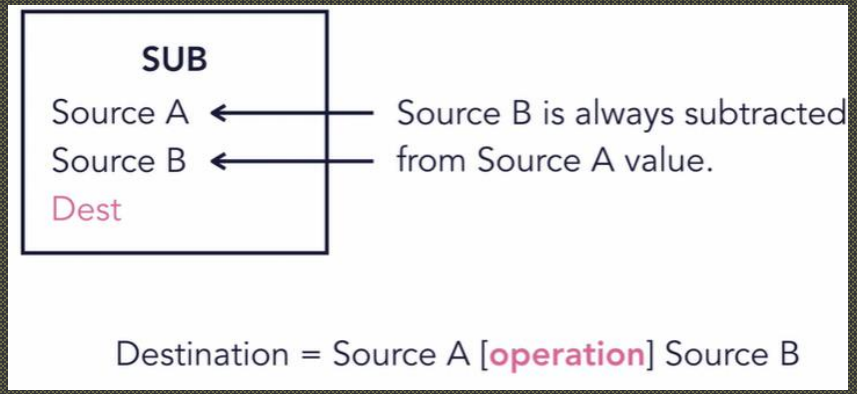
HUMBER

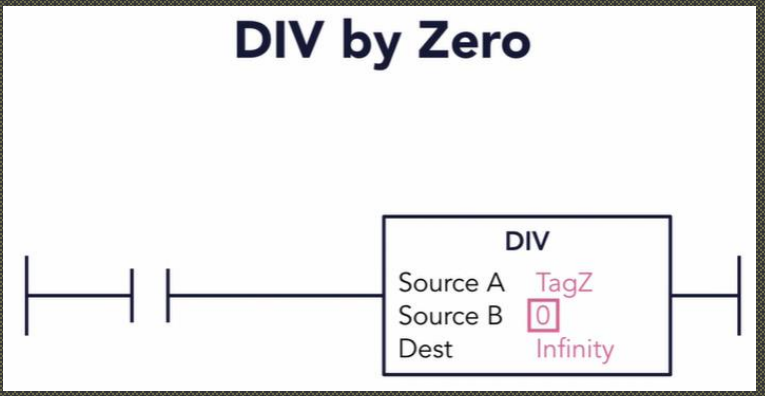
https://www.linkedin.com/learning/learning-plc-ladder-logic-2-diving-deeper/

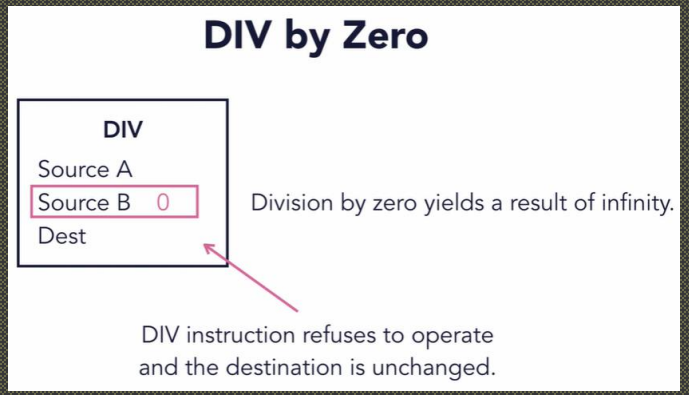
MATH INSTRUCTIONS







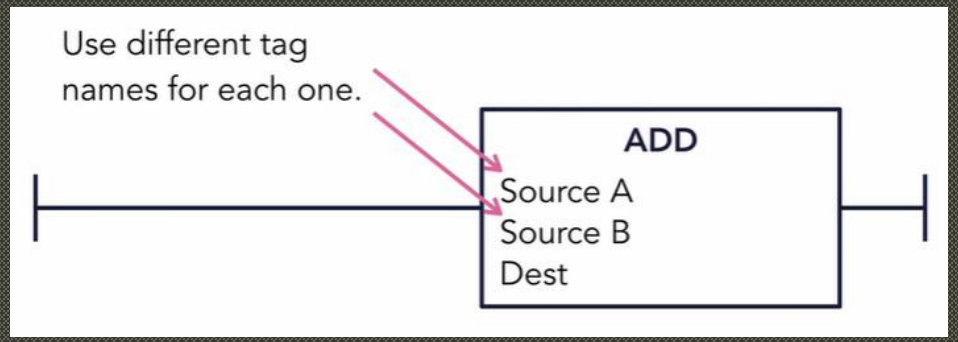


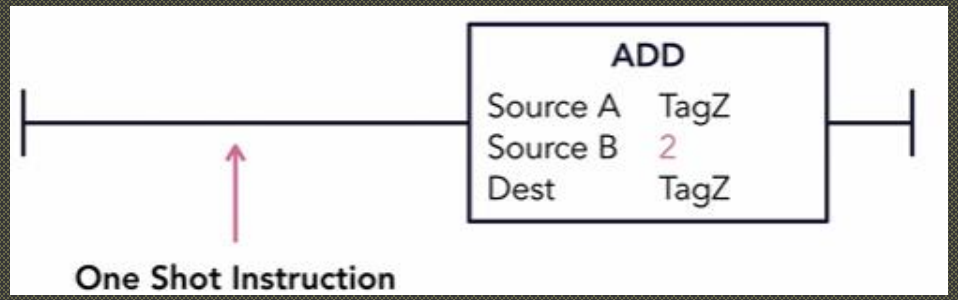


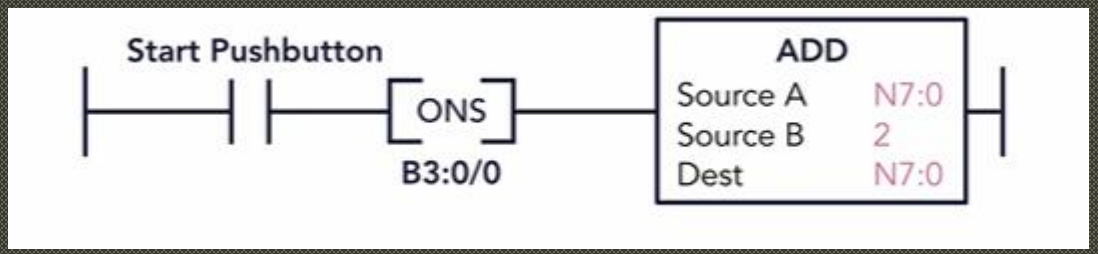
HUMBER

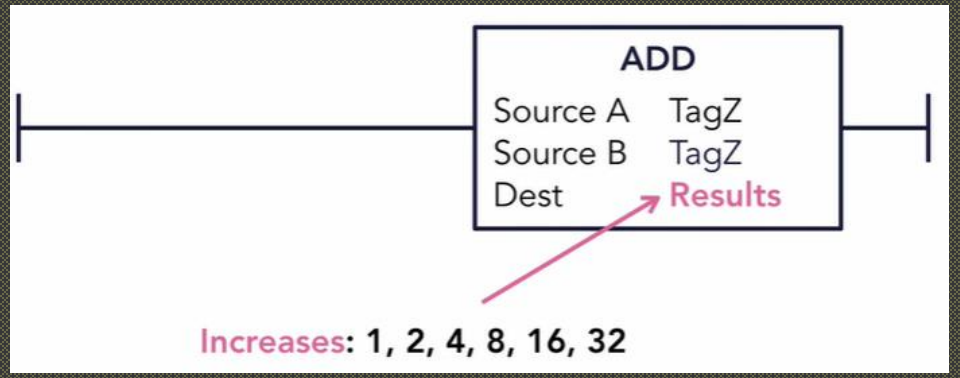
https://www.linkedin.com/learning/learning-plc-ladder-logic-2-diving-deeper/

MATH INSTRUCTIONS









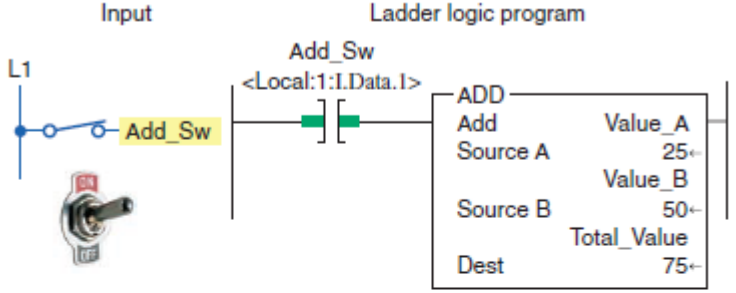
https://www.linkedin.com/learning/learning-plc-ladder-logic-2-diving-deeper/

HUMBER

MATH INSTRUCTIONS

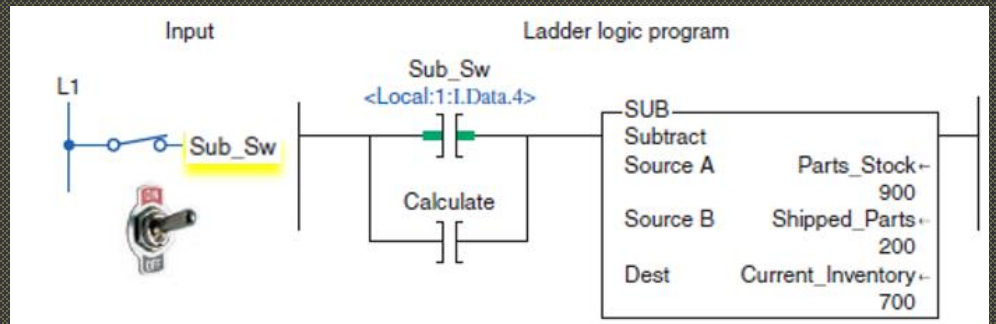
Addition and Subtraction Instructions

The Source can be a constant value or a tag. The result of the ADD or SUB instructions is put in the destination (Dest) tag.



Tag Name	Value	Style	Data Type
Total_Value	75	Decimal	DINT
Value_A	25	Decimal	DINT
Value_B	50	Decimal	DINT
ADD_Sw	1	Decimal	BOOL

Figure 15-67 ADD instruction rung and its Monitor Tags window.



Tag Name	Value	Style	Data Type
Parts_Stock	900	Decimal	DINT
Shipped_Parts	200	Decimal	DINT
Current_Inventory	700	Decimal	DINT
Sub_Sw	1	Decimal	BOOL
Calculate	0	Decimal	BOOL

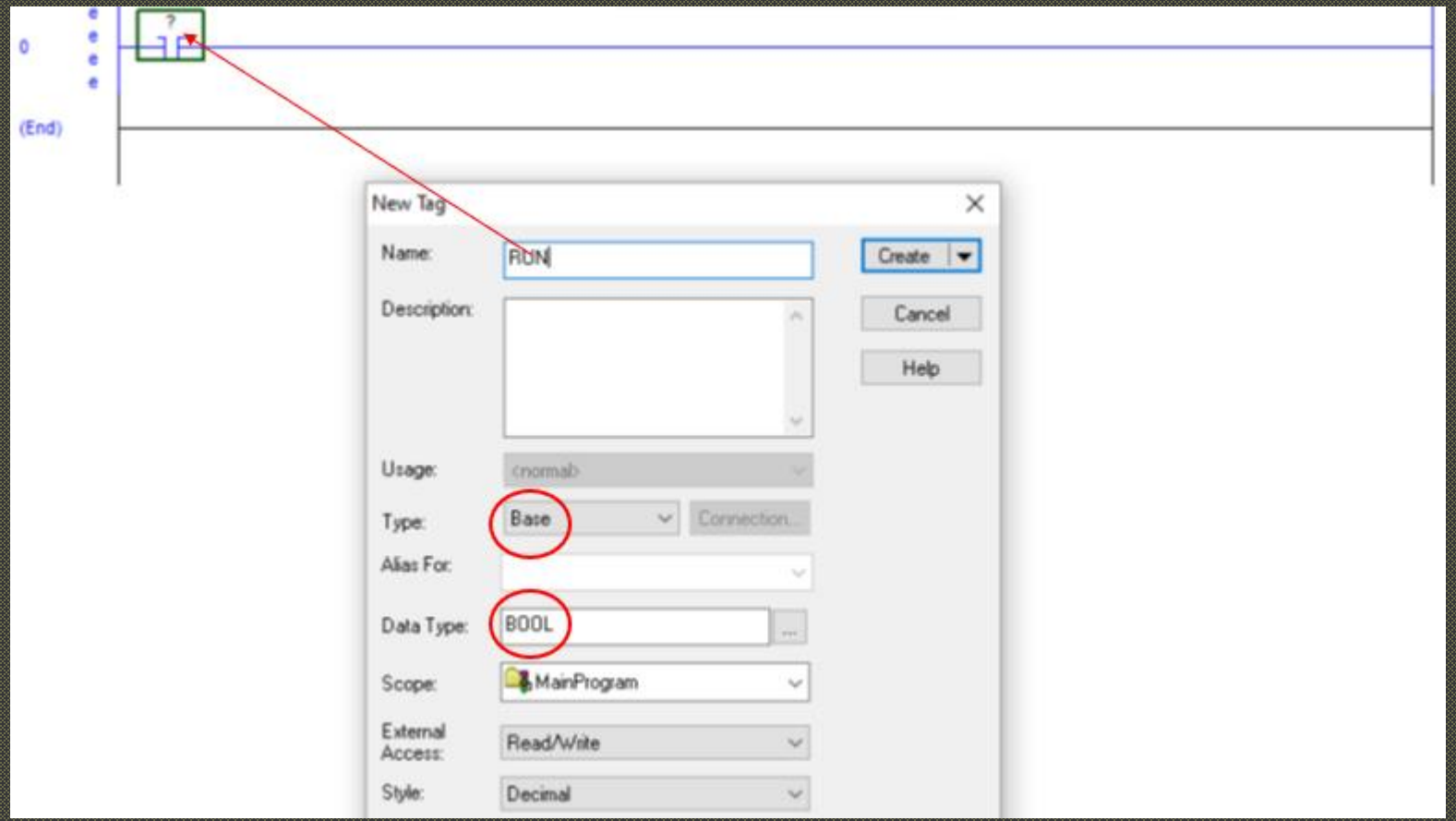
Figure 15-68 SUB instruction rung and its Monitor Tags window.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

MATH INSTRUCTIONS

Addition Instruction

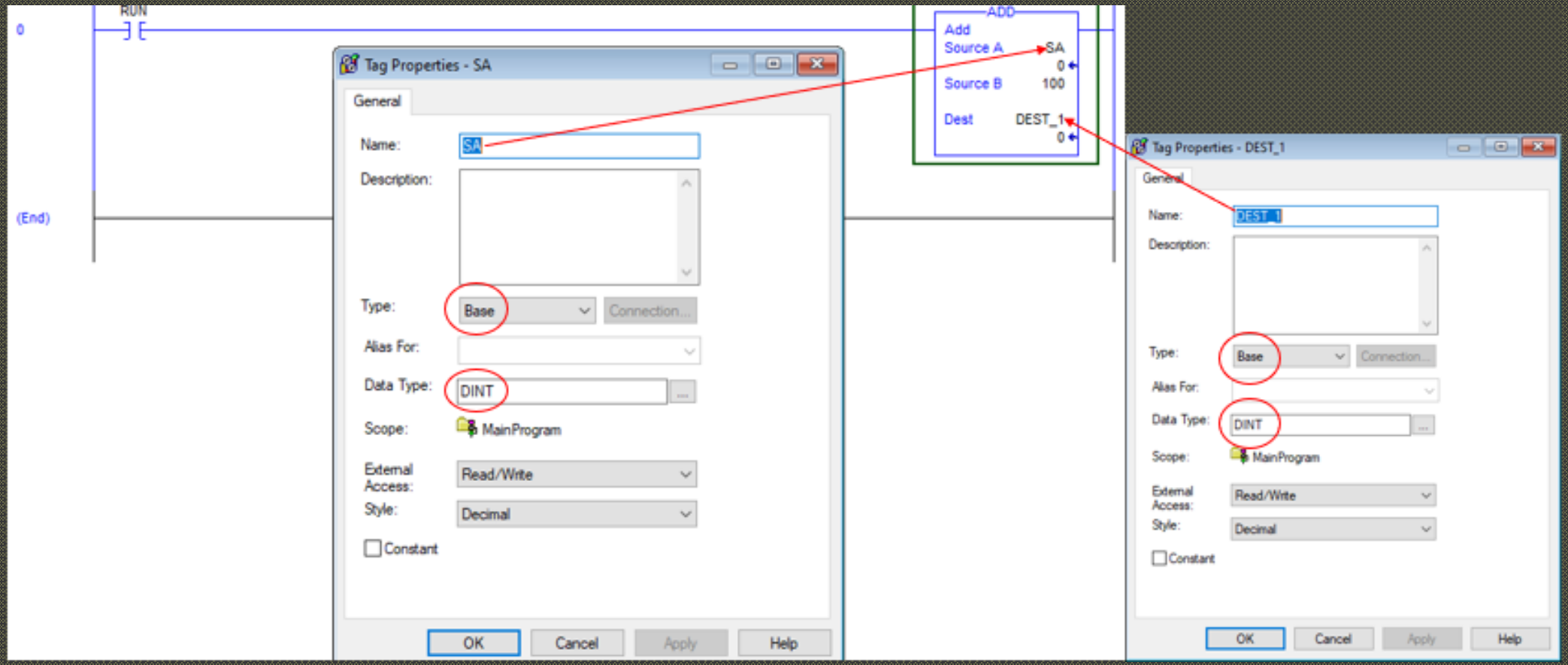
When the RUN (Type: Base; Data Type: BOOLEAN) bit becomes TRUE, the output ADD instruction will be executed **every scan.**



HUMBER

MATH INSTRUCTIONS

Addition Instruction

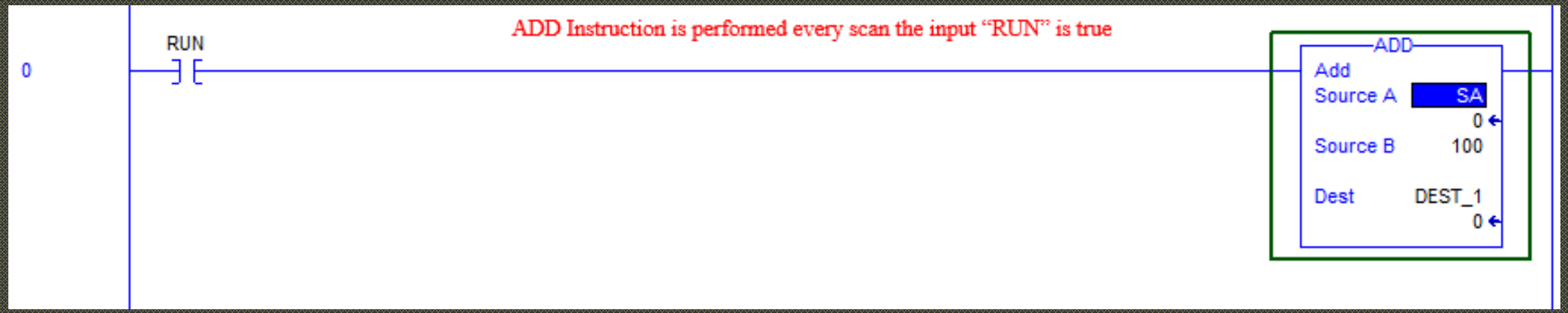


HUMBER

MATH INSTRUCTIONS

Addition Instruction

Destination tag different form the Source A and Source B tags

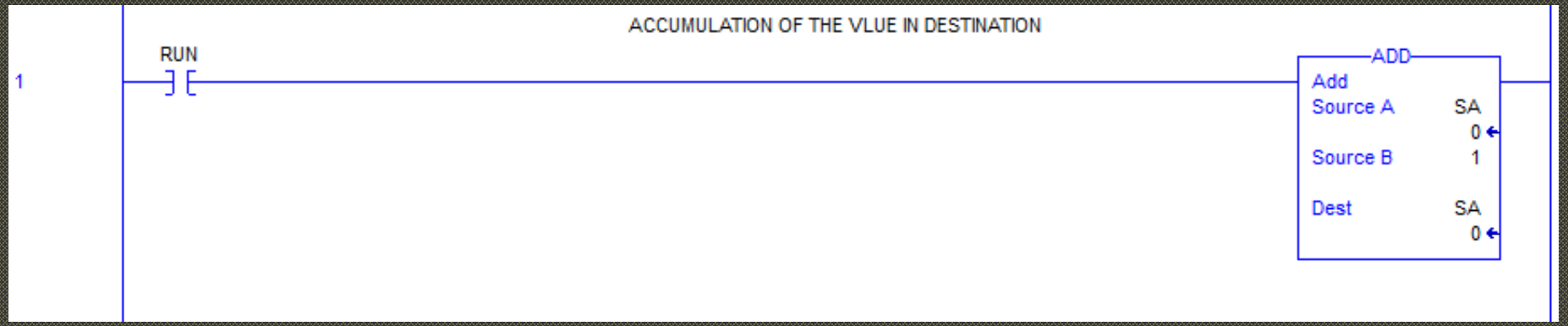


HUMBER

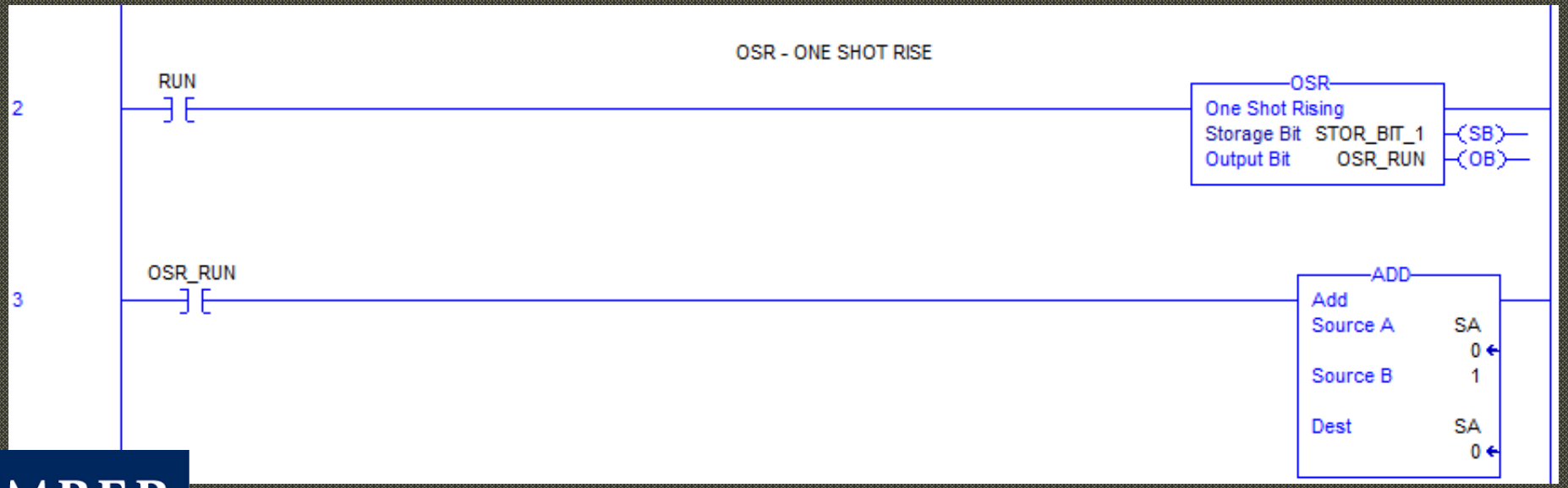
MATH INSTRUCTIONS

Addition Instruction

Destination tag, same as the Source A tag, causes continuous accumulation in the Destination.



To avoid accumulation, use output bit of the OSR instruction.

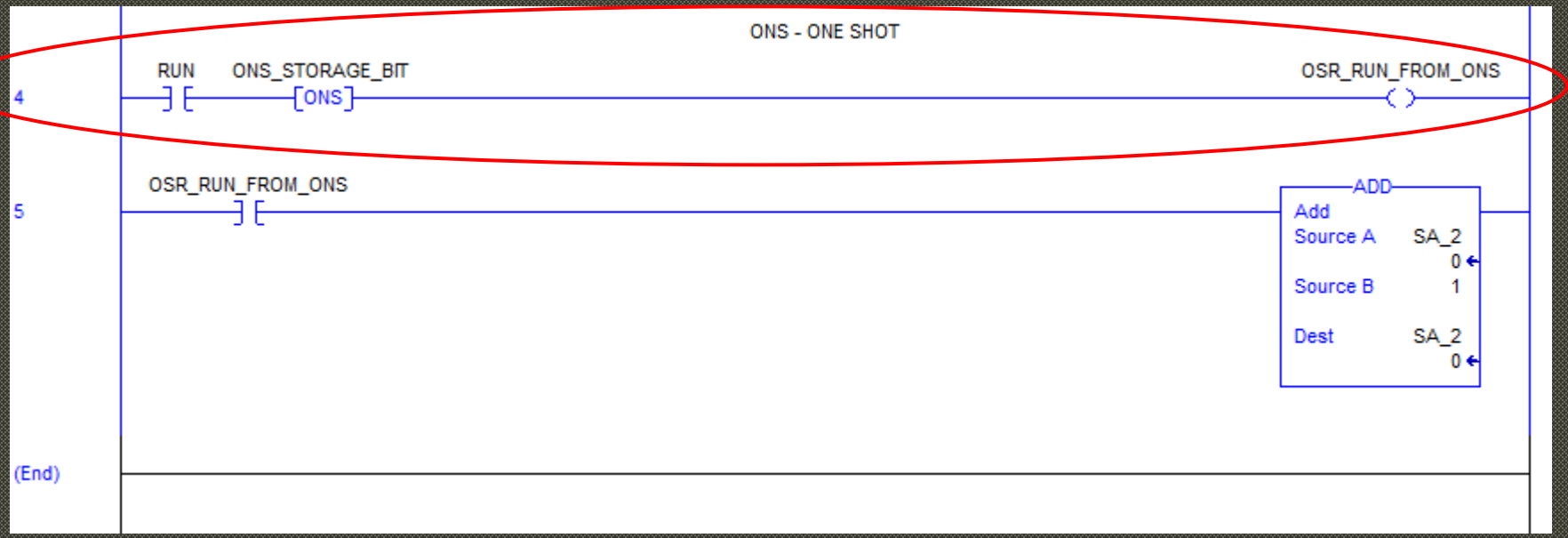


HUMBER

MATH INSTRUCTIONS

Addition Instruction

You may use ONS instruction instead of OSR or OSF



Operates akin to OSR

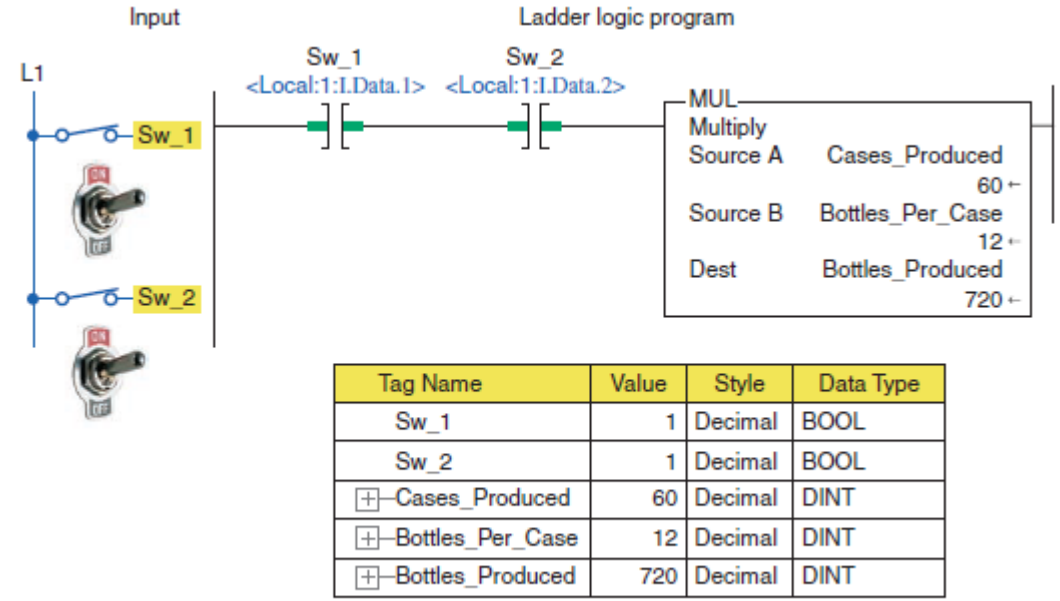
Other mathematical instructions are also encountering the same issue with the Destination as the one observed.

HUMBER

MATH INSTRUCTIONS

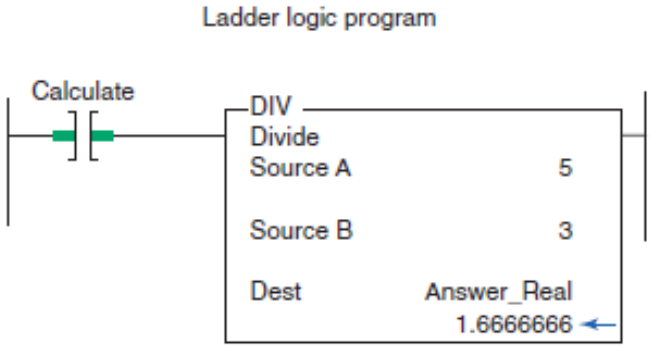
Multiplication and Division Instructions

The Source can be a constant value or a tag. The result of the MUL or DIV instructions is put in the destination (Dest) tag.



Tag Name	Value	Style	Data Type
Sw_1	1	Decimal	BOOL
Sw_2	1	Decimal	BOOL
⊞ Cases_Produced	60	Decimal	DINT
⊞ Bottles_Per_Case	12	Decimal	DINT
⊞ Bottles_Produced	720	Decimal	DINT

Figure 15-69 MUL instruction rung and its Monitor Tags window.



Tag Name	Value	Style	Data Type
Calculate	1	Decimal	BOOL
Answer_Real	1.6666666	Float	REAL

Figure 15-70 DIV instruction rung and its Monitor Tags window.

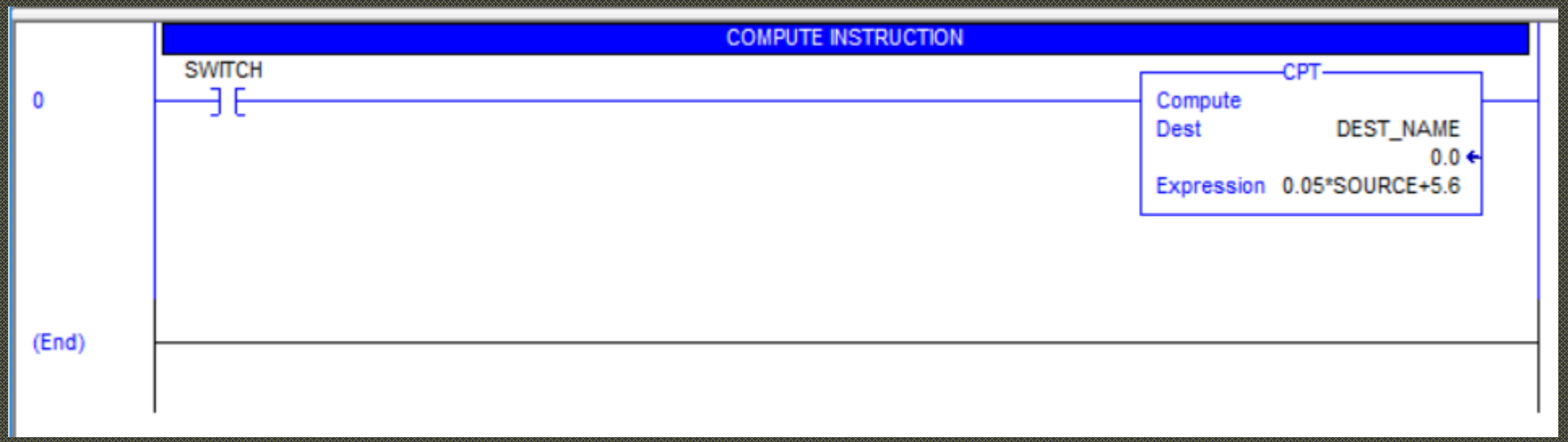
(Frank D. Petruzella Programmable Logic Controllers 4th edition)

HUMBER

MATH INSTRUCTIONS

Compute Instruction

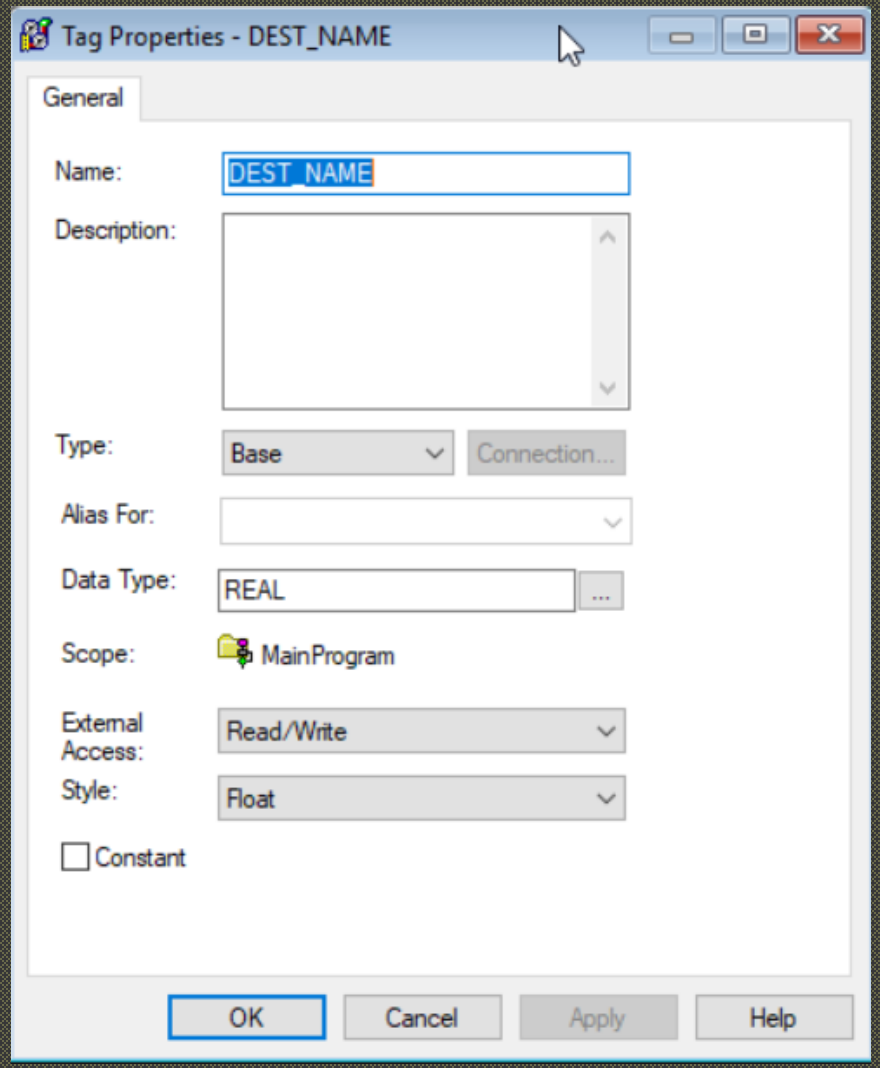
- CPT instruction is used to calculate the formula in the Expression and store the result in the Destination
- CPT is an output instruction

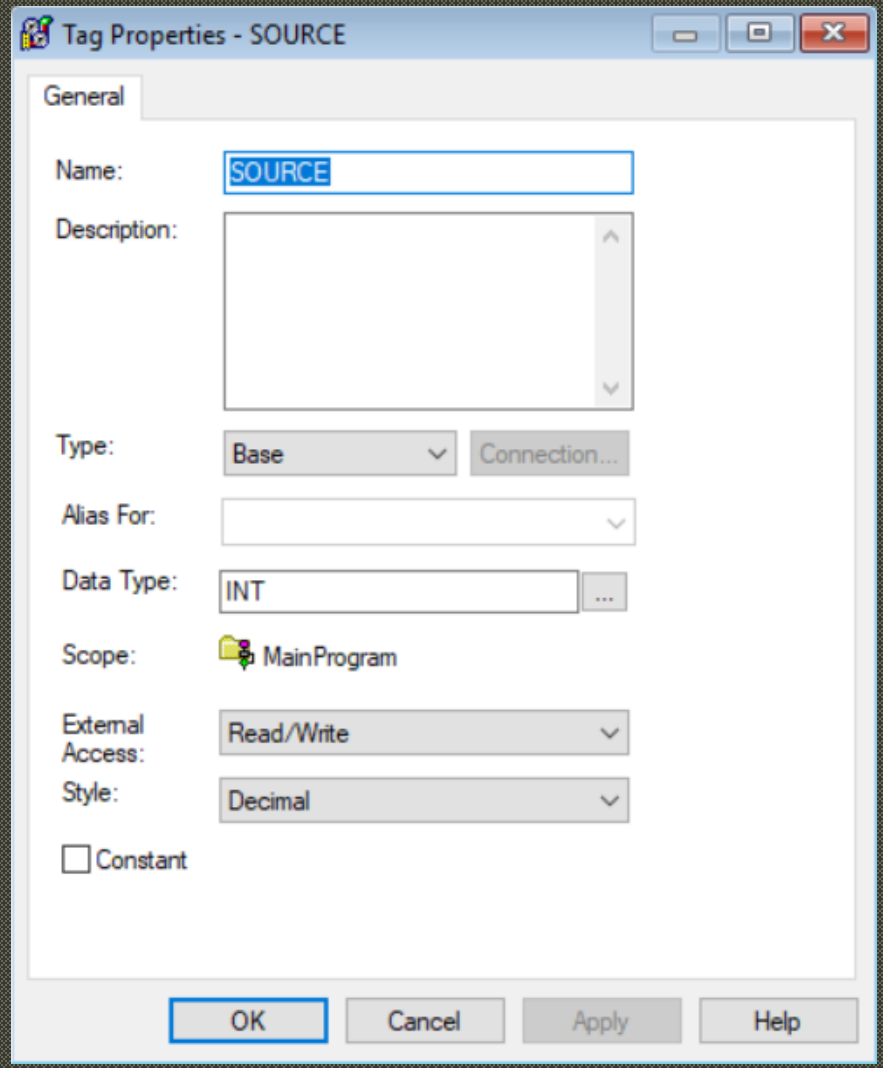


HUMBER

MATH INSTRUCTIONS

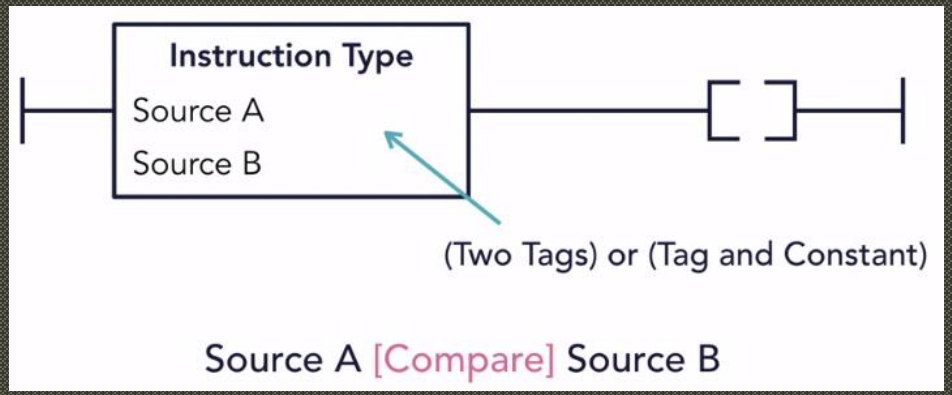
Compute Instruction

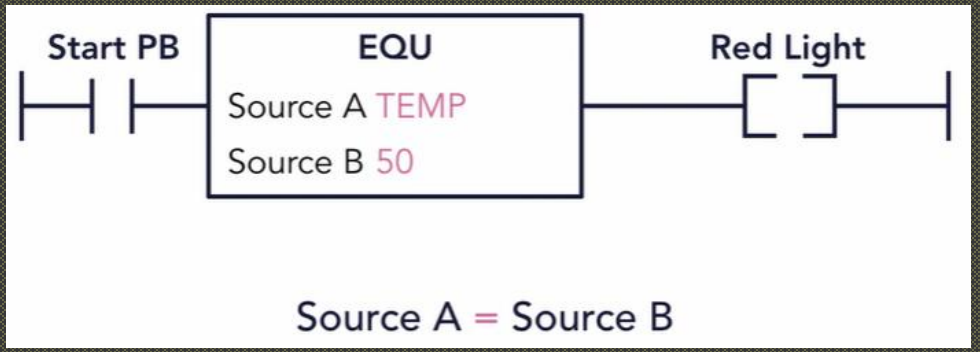


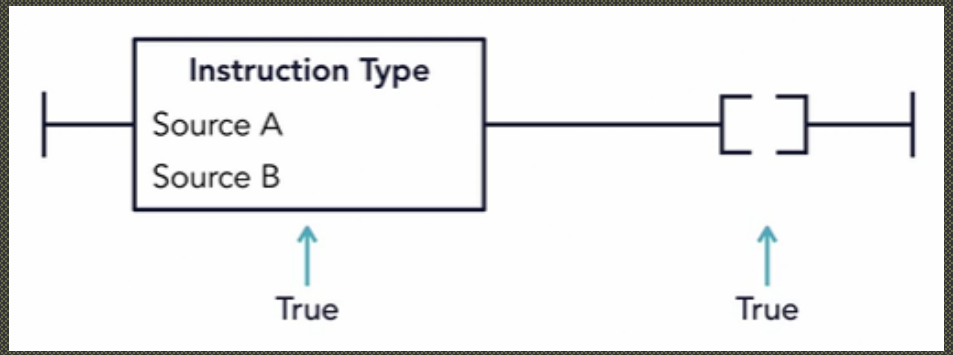


HUMBER

COMPARISON INSTRUCTIONS



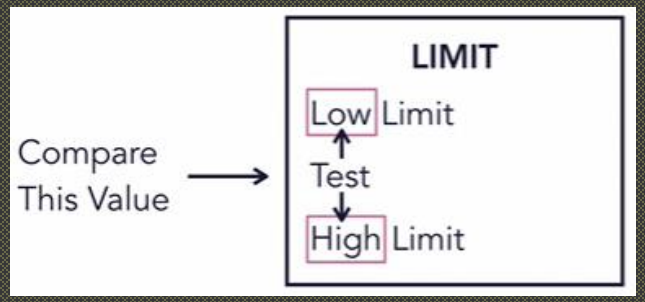


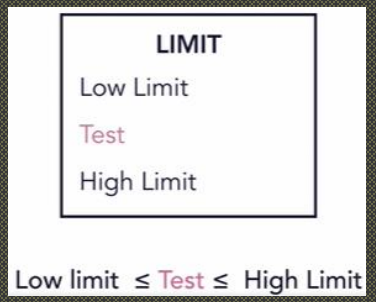


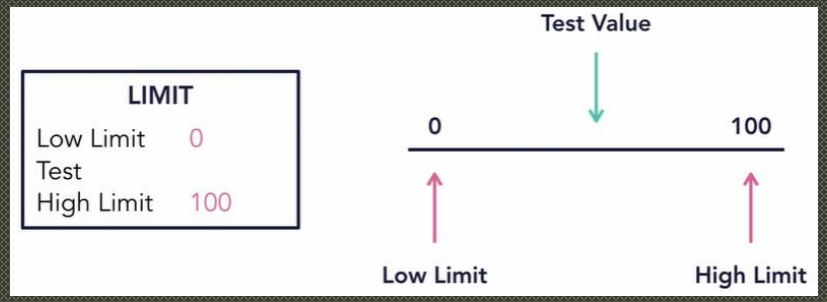
Instruction	
EQU	Equal
NEQ	Not Equal
GRT	Greater than
GEQ	Greater than or Equal
LES	Less than
LEQ	Less than or Equal

https://www.linkedin.com/learning/learning-plc-ladder-logic-2-diving-deeper/

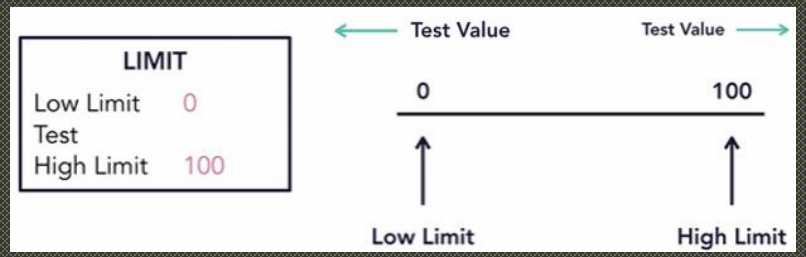
COMPARISON INSTRUCTIONS

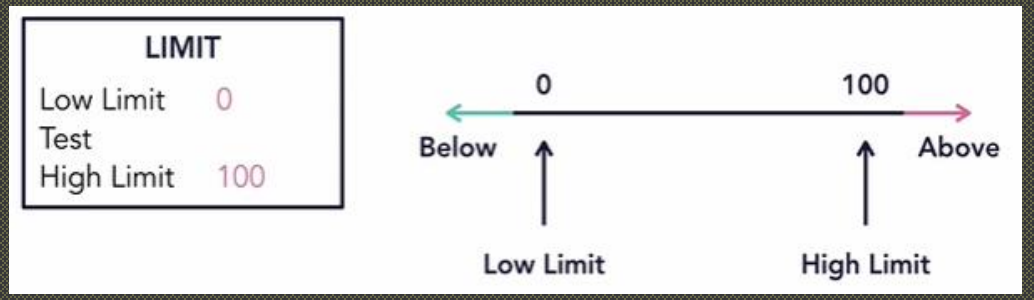


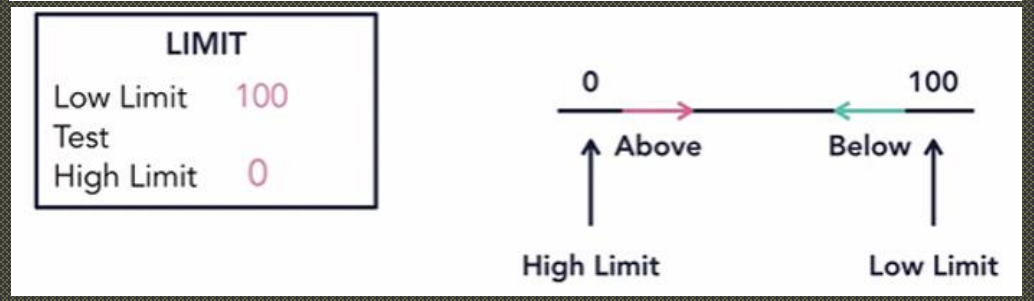




What if we want to test outside the limits?







https://www.linkedin.com/learning/learning-plc-ladder-logic-2-diving-deeper/

COMPARISON INSTRUCTIONS

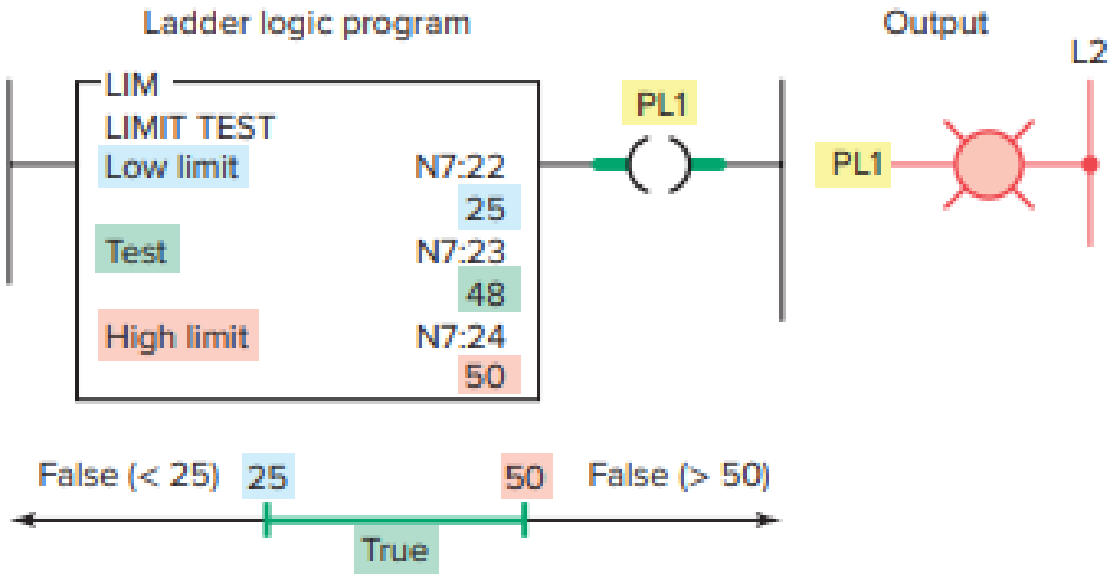


Figure 10-28 LIM instruction where the low limit value is less than the high limit value.

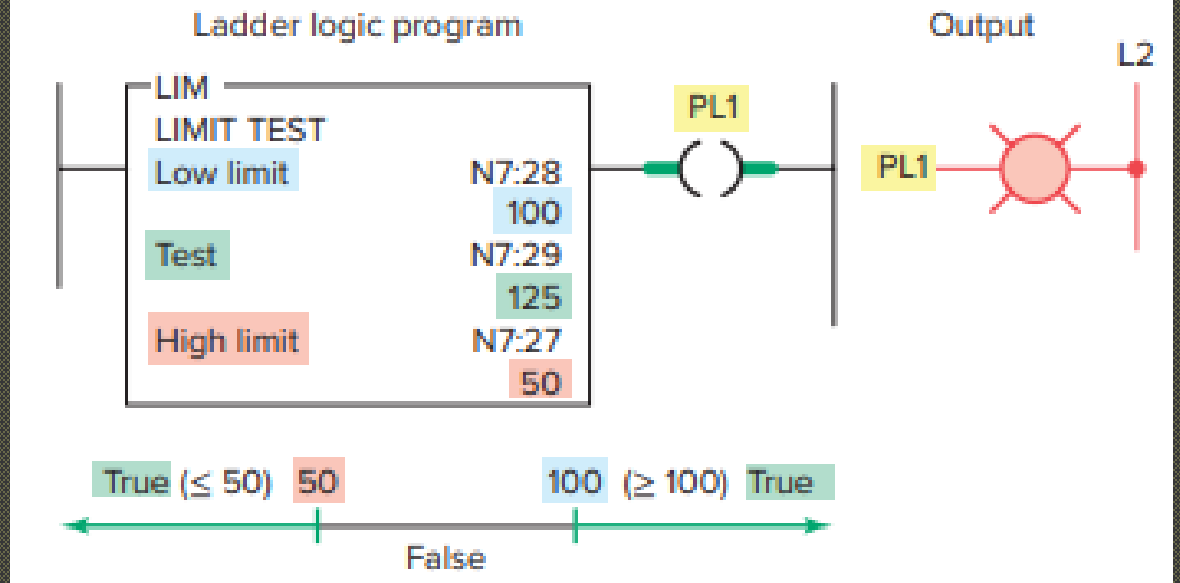
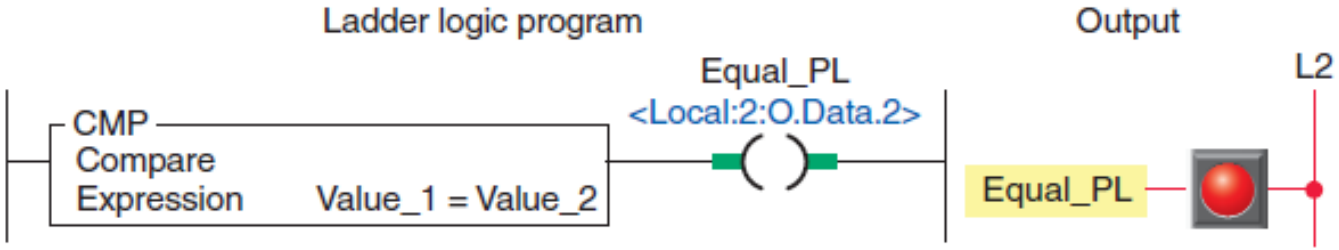


Figure 10-29 LIM instruction where the low limit value is greater than the high limit value.

COMPARISON INSTRUCTIONS

- The compare **(CMP)** instruction performs a comparison on the arithmetic operations specified by the expression. The expression may contain arithmetic operators, comparison operators, and tags.

- The execution of a CMP instruction is slightly slower and uses more memory than the execution of the other comparison instructions.
- The advantage of the CMP instruction is that it allows you to enter complex expressions in one instruction.

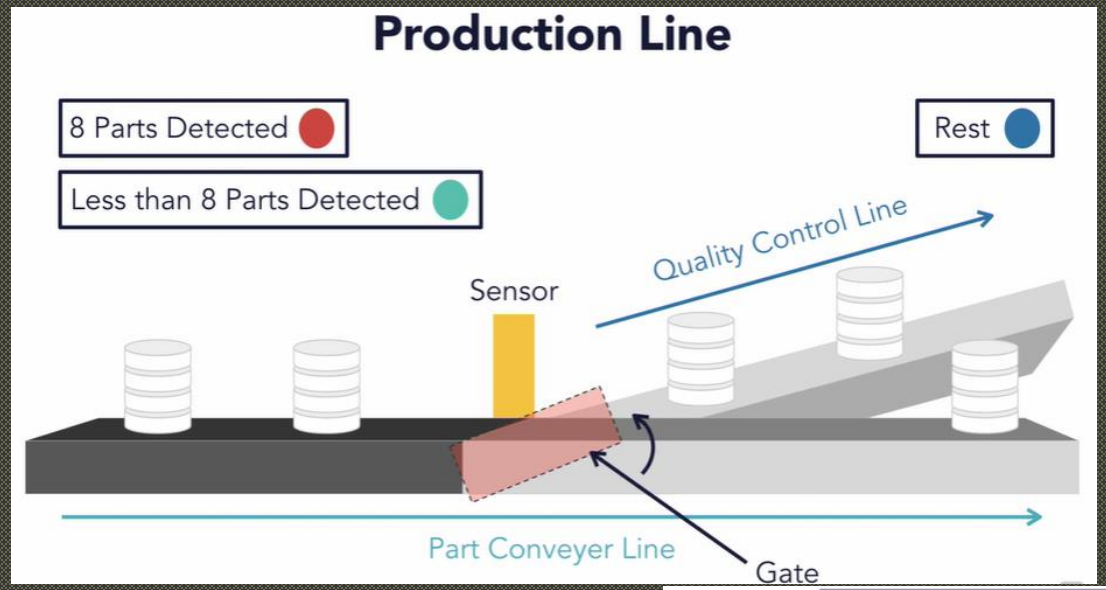


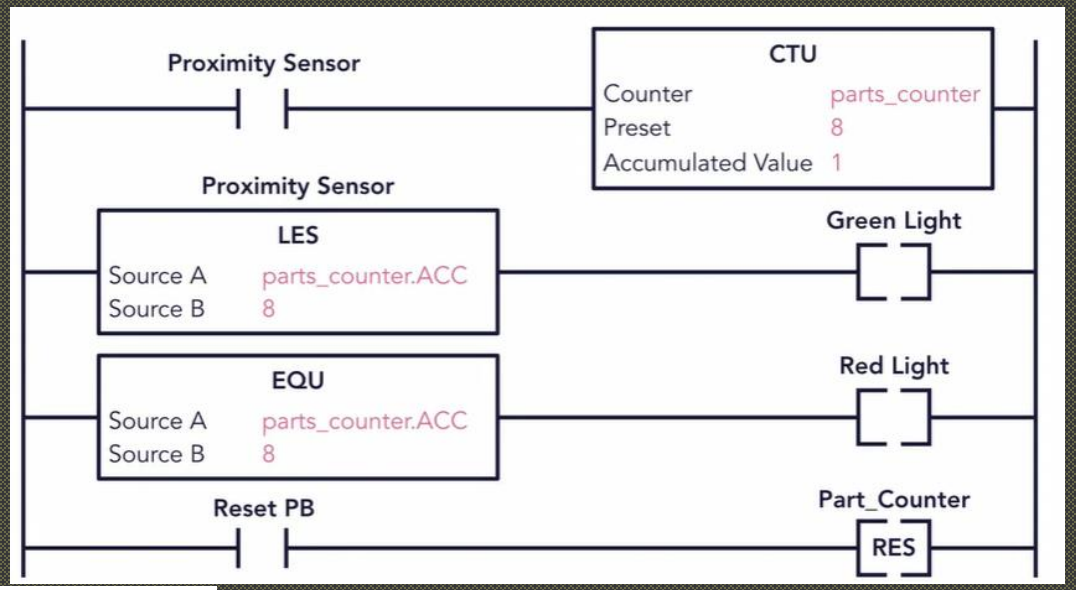
Tag Name	Value	Style	Data Type
Equal_PL	1	Decimal	BOOL
+-Value_1	300	Decimal	DINT
+-Value_2	300	Decimal	DINT

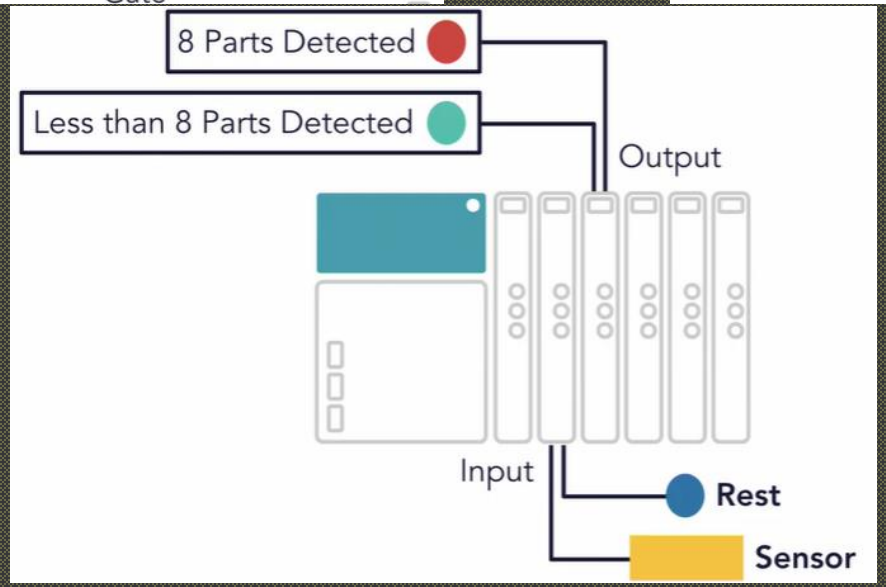
Figure 15-77 CMP instruction rung.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

COMPARISON INSTRUCTIONS - Example





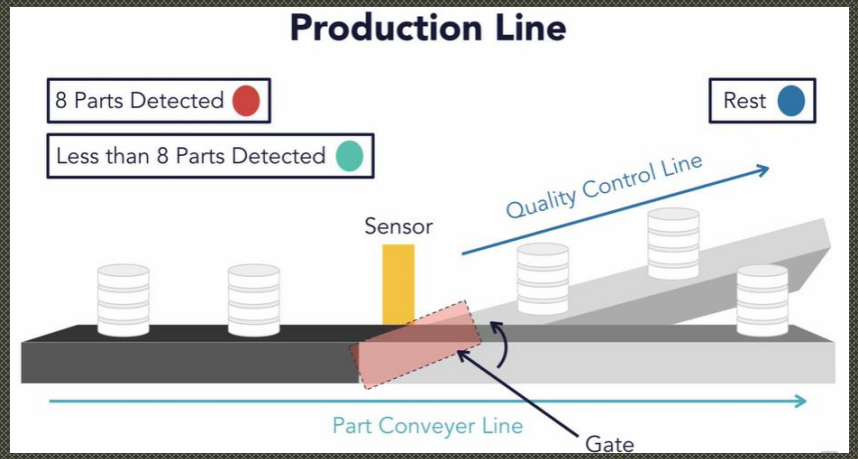


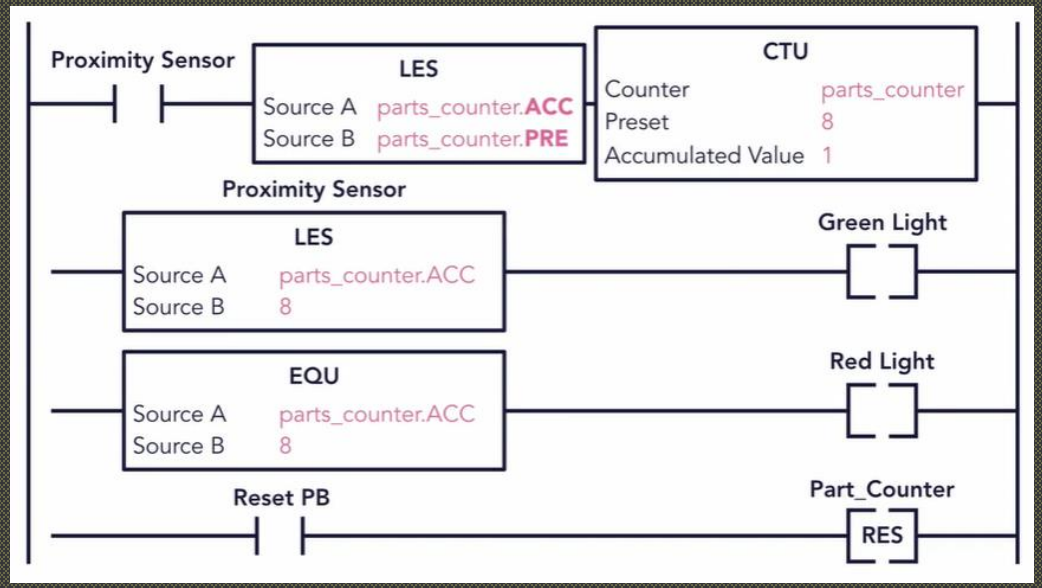
- Proximity sensor is used to count the number of damaged parts
- The Gate, used to separate the damaged parts is not part of the program

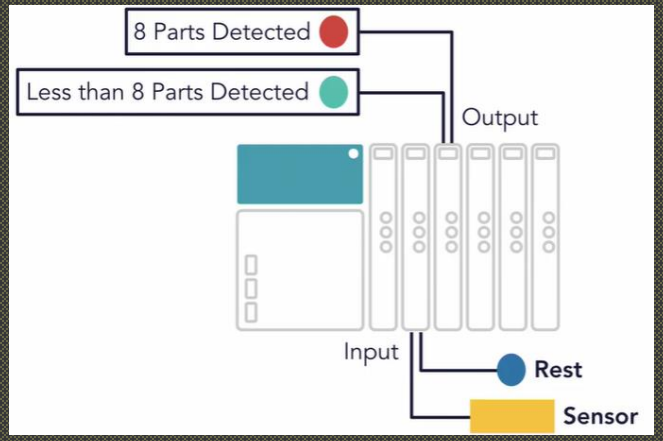
https://www.linkedin.com/learning/learning-plc-ladder-logic-2-diving-deeper/

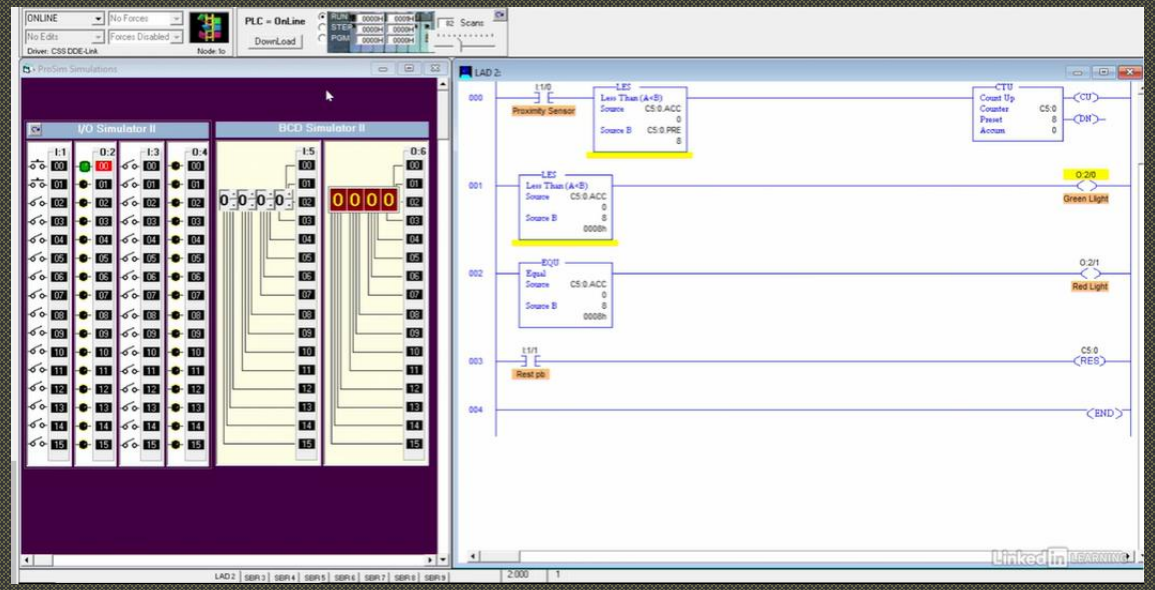
COMPARISON INSTRUCTIONS - Example

To prevent the Counter from exceeding the Preset Value





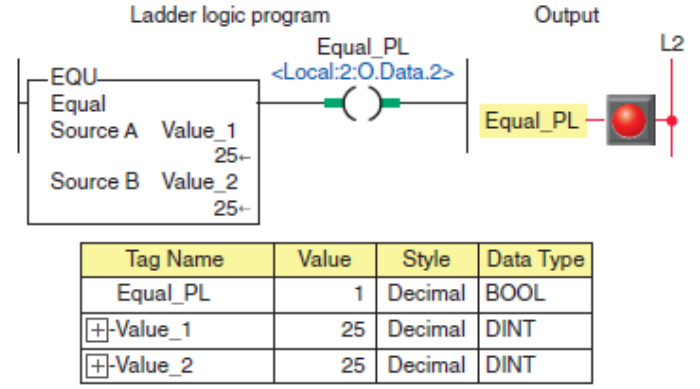




https://www.linkedin.com/learning/learning-plc-ladder-logic-2-diving-deeper/

COMPARISON INSTRUCTIONS

- Comparison instructions are used to compare two values. They can be used to see if two values are equal, if one value is greater or less than the other, and so on;
- **All of them are input instructions;**
- The value stored at Source A is compared to the value stored at Source B;
- Source A and Source B may be **SINT, INT, DINT, or REAL** data types;



Tag Name	Value	Style	Data Type
Equal_PL	1	Decimal	BOOL
⊞-Value_1	25	Decimal	DINT
⊞-Value_2	25	Decimal	DINT

Figure 15-73 EQU instruction rung and its Monitor Tags window.

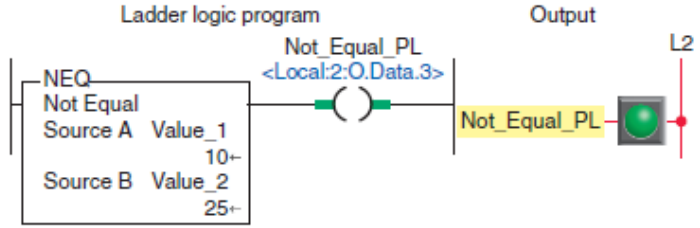


Figure 15-74 NEQ instruction rung.

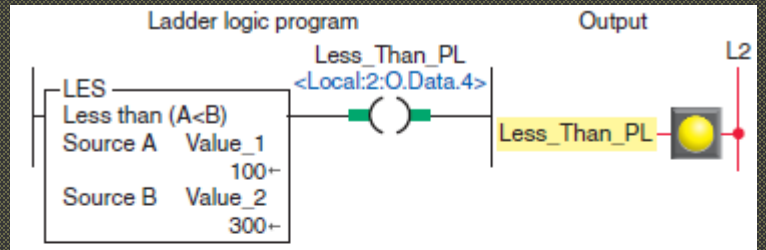


Figure 15-75 LES instruction rung.

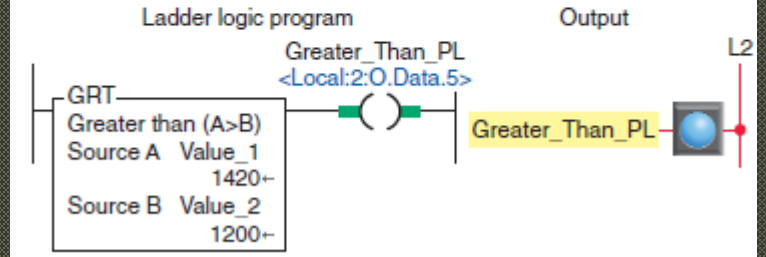


Figure 15-76 GRT instruction rung.

COMPARISON INSTRUCTIONS

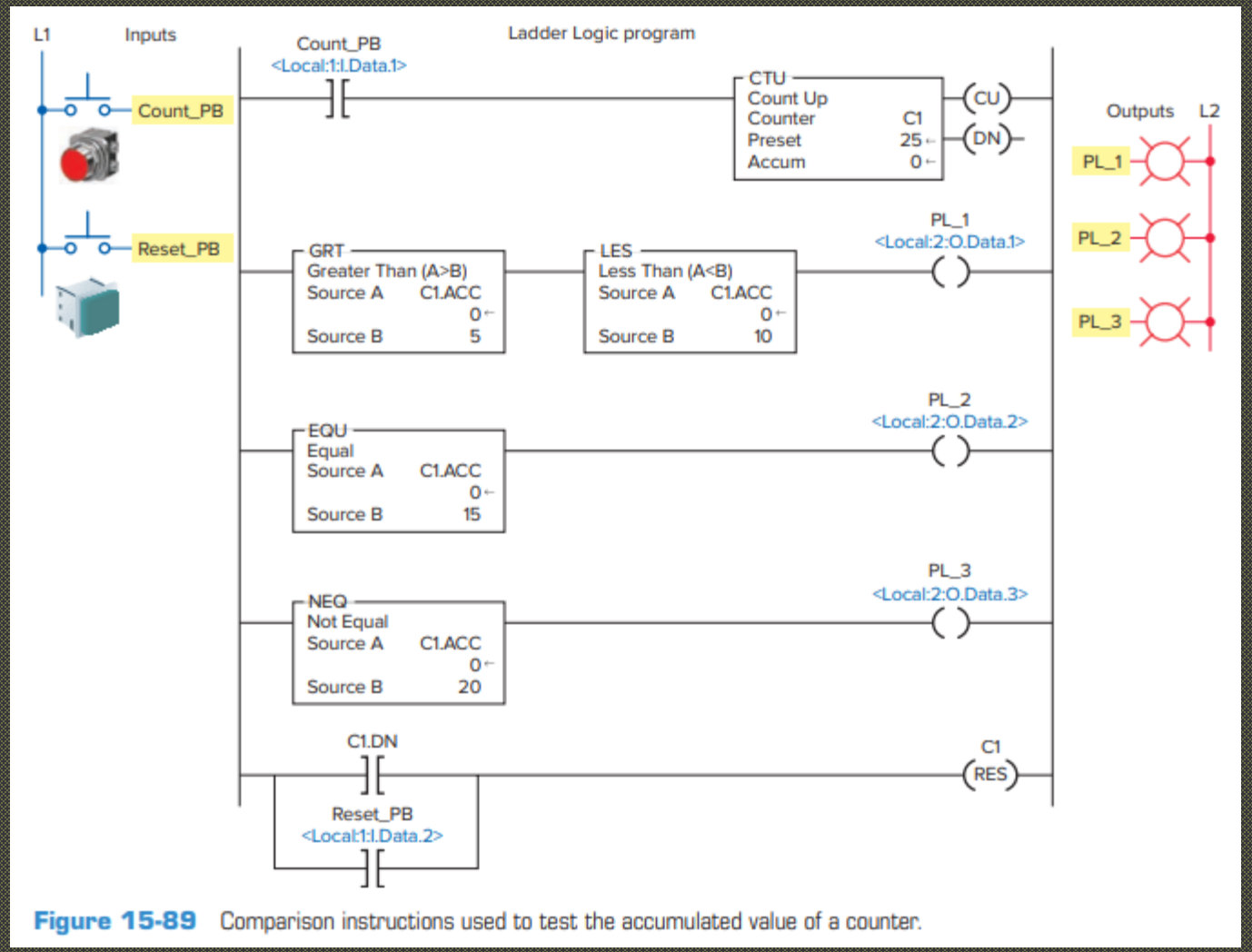
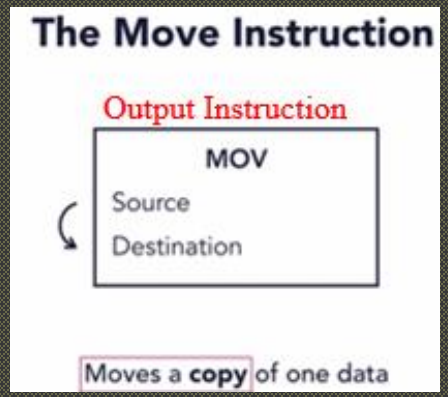


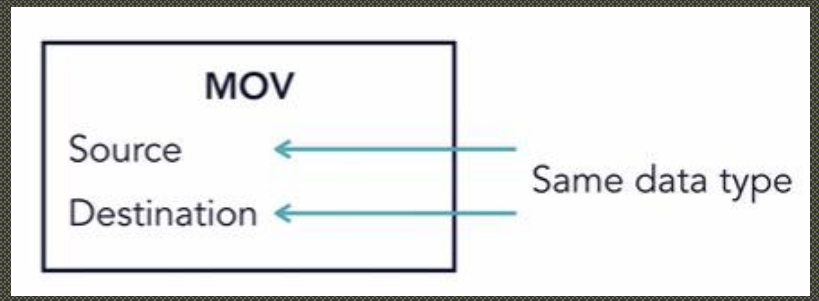
Figure 15-89 Comparison instructions used to test the accumulated value of a counter.

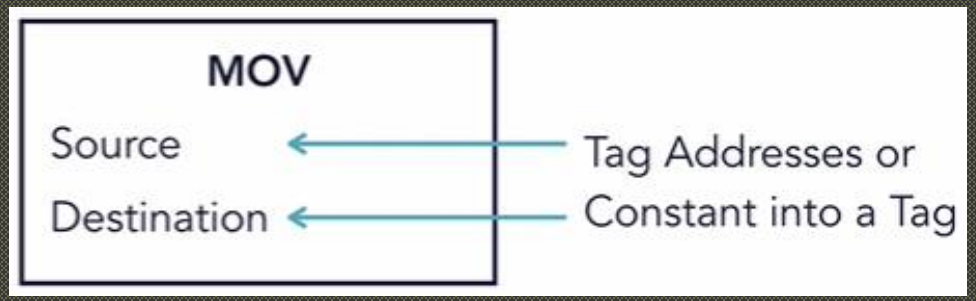
(Frank D. Petruzella Programmable Logic Controllers 5th edition)

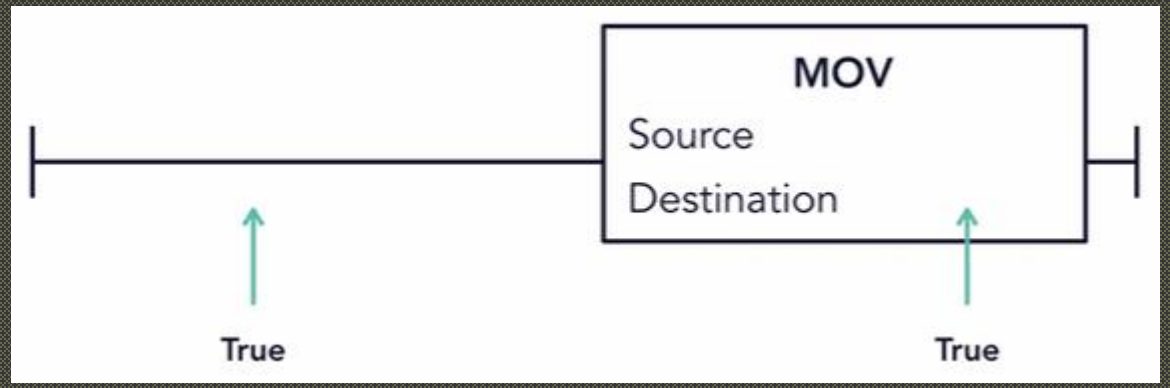
DATA TRANSFER INSTRUCTIONS

Move Instruction









DATA TRANSFER INSTRUCTIONS

Move Instruction Example

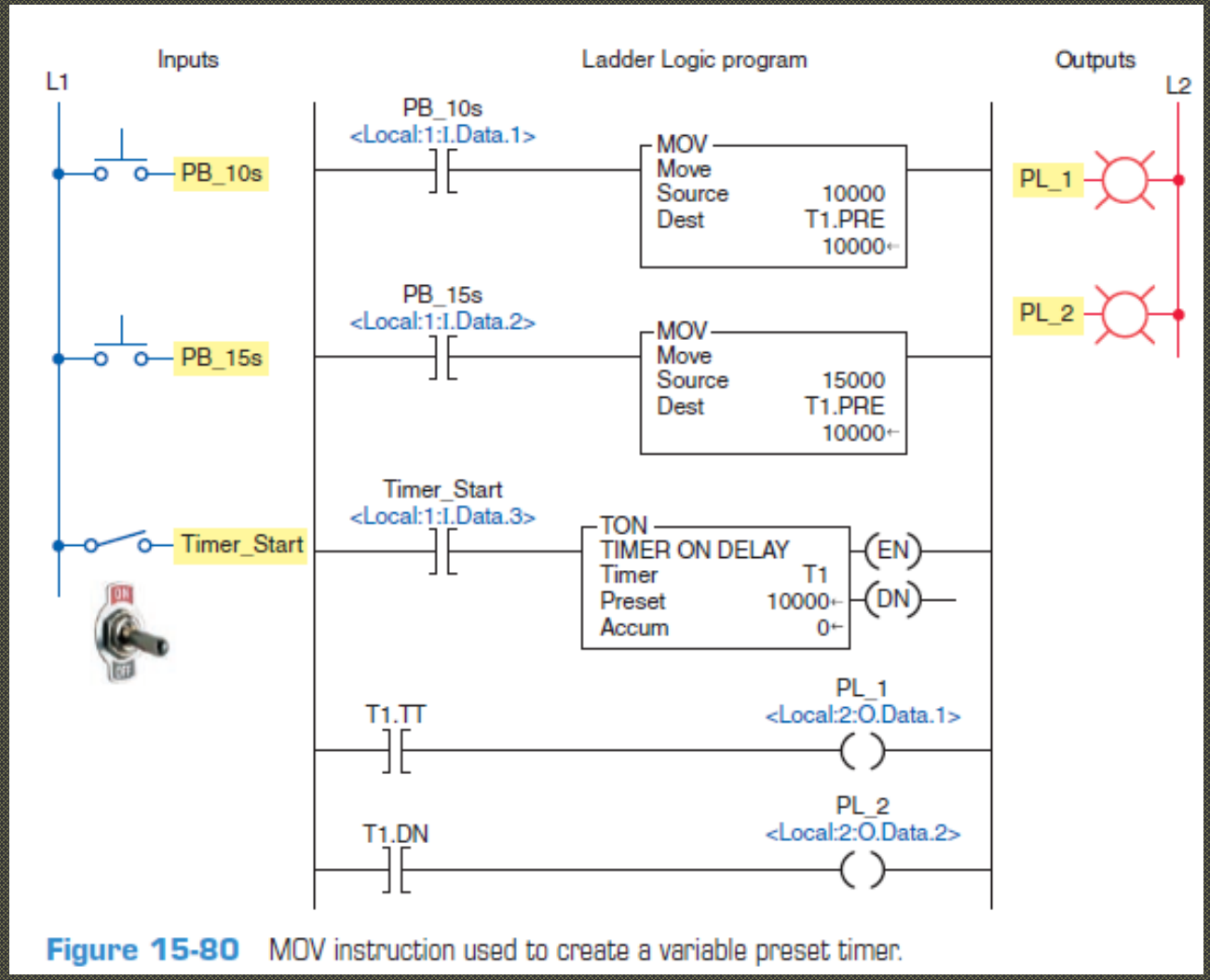
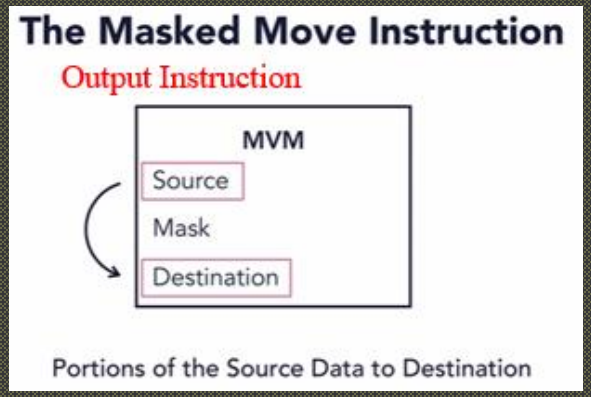


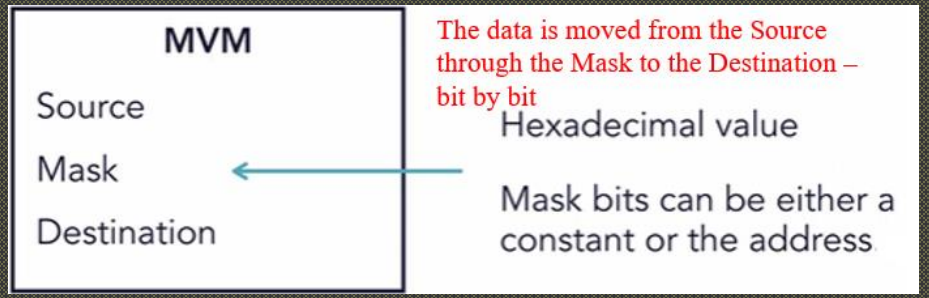
Figure 15-80 MOV instruction used to create a variable preset timer.

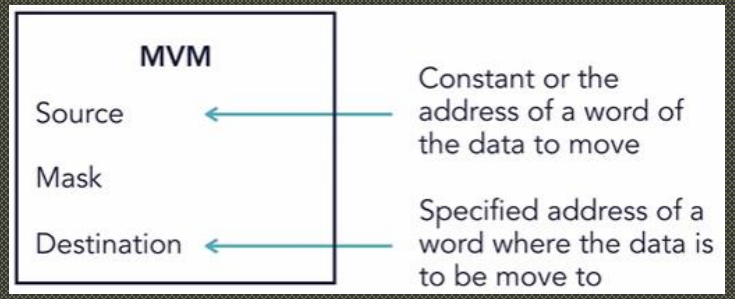
(Frank D. Petruzella Programmable Logic Controllers 4th edition)

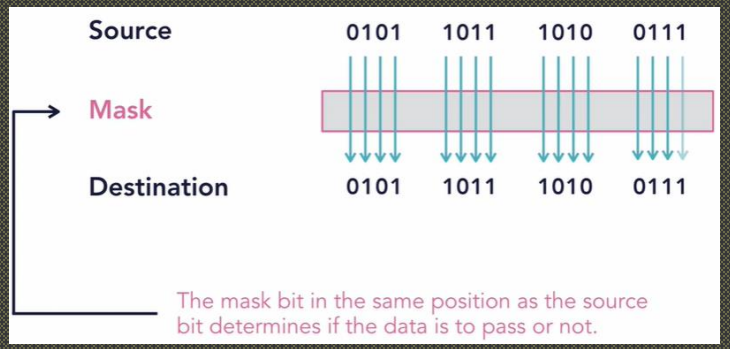
DATA TRANSFER INSTRUCTIONS

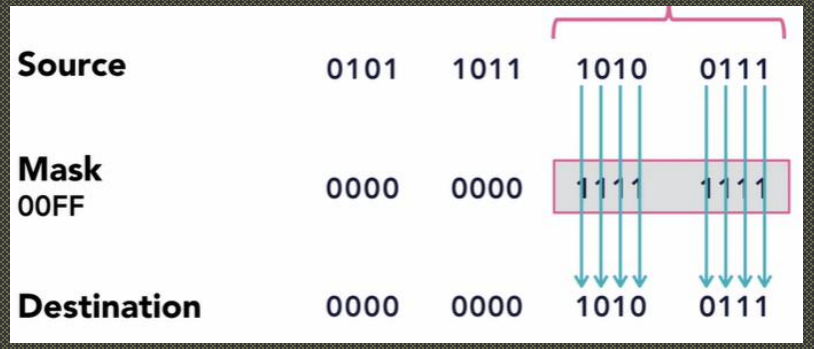
Move with Mask Instruction











https://www.linkedin.com/learning/learning-plc-ladder-logic-2-diving-deeper/

DATA TRANSFER INSTRUCTIONS

Move with Mask Instruction

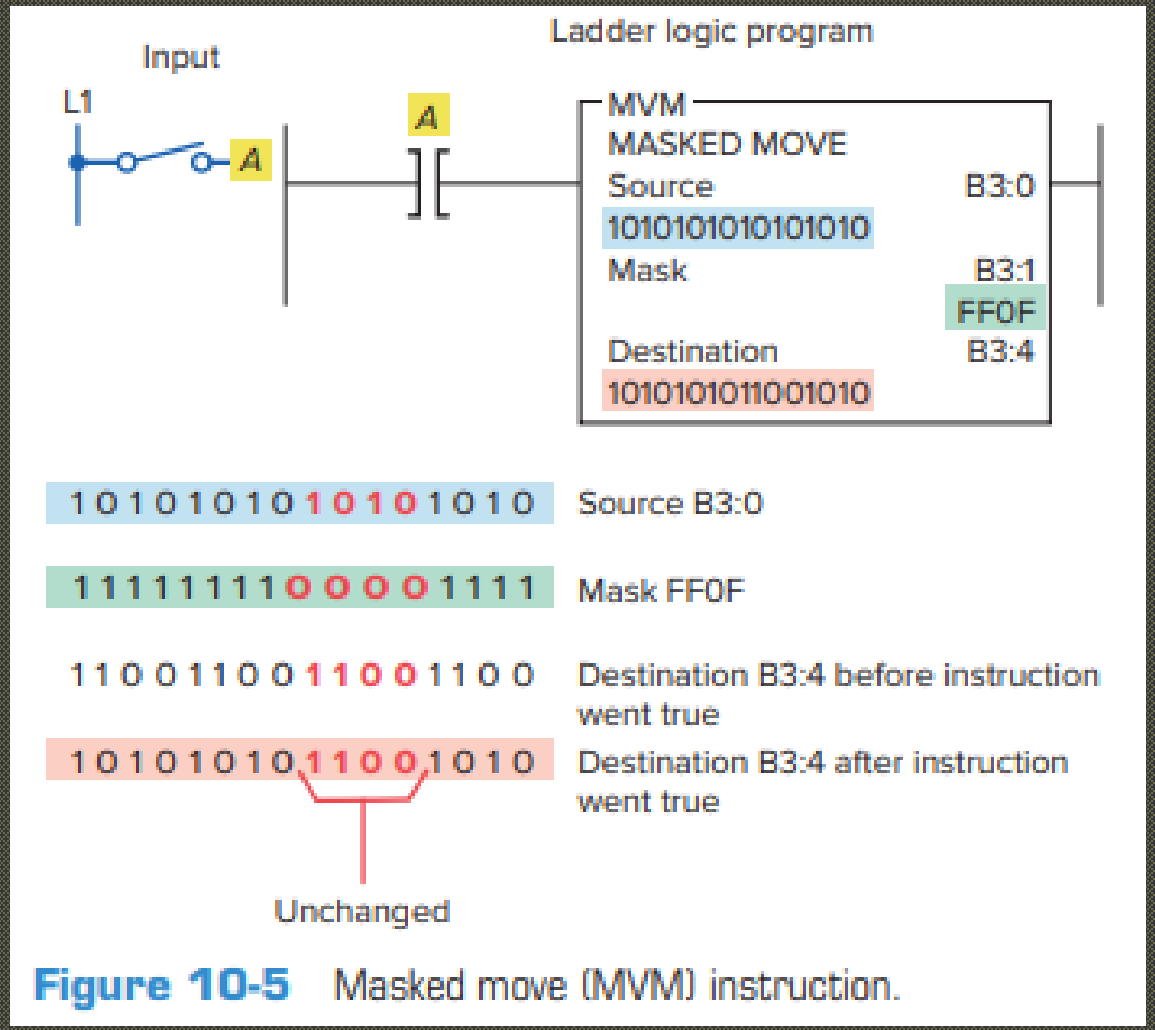


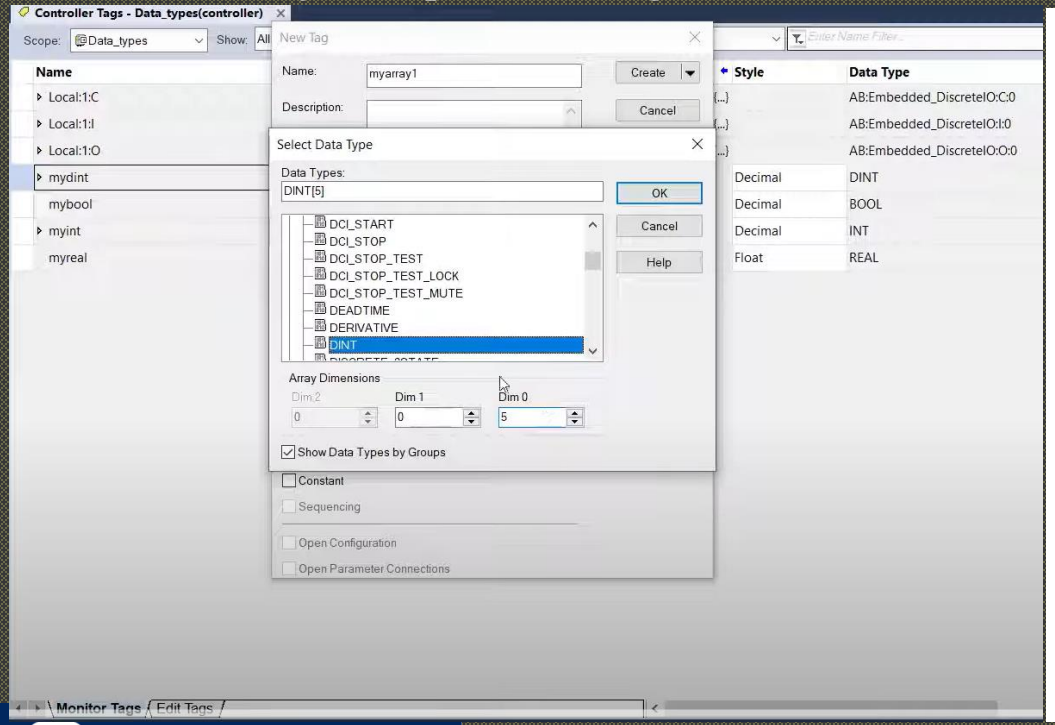
Figure 10-5 Masked move (MVM) instruction.

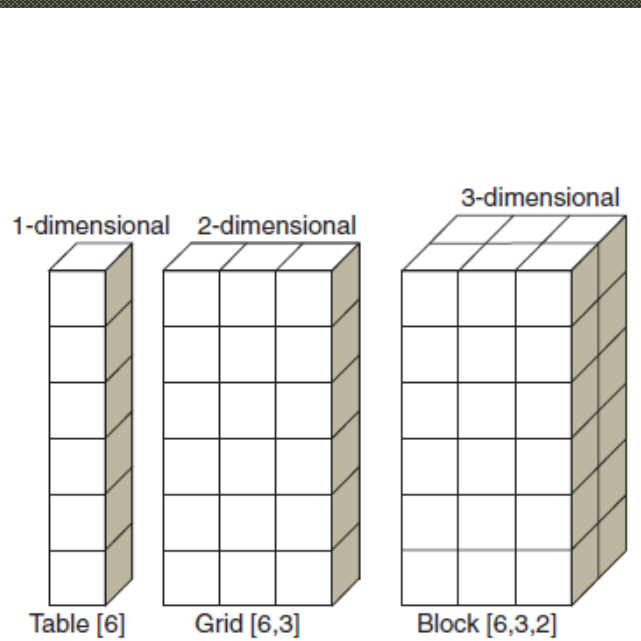
(Frank D. Petruzella Programmable Logic Controllers 5th edition)

DATA TRANSFER INSTRUCTIONS

Arrays

- Many control programs require the ability to store blocks of information in memory in the form of tables that can be accessed at runtime.
- An **Array** is a tag type that contains a block of multiple pieces of data. Each element of an array must be of the same data type (e.g., BOOL, SINT, or INT).
- An array occupies a contiguous block of controller memory.





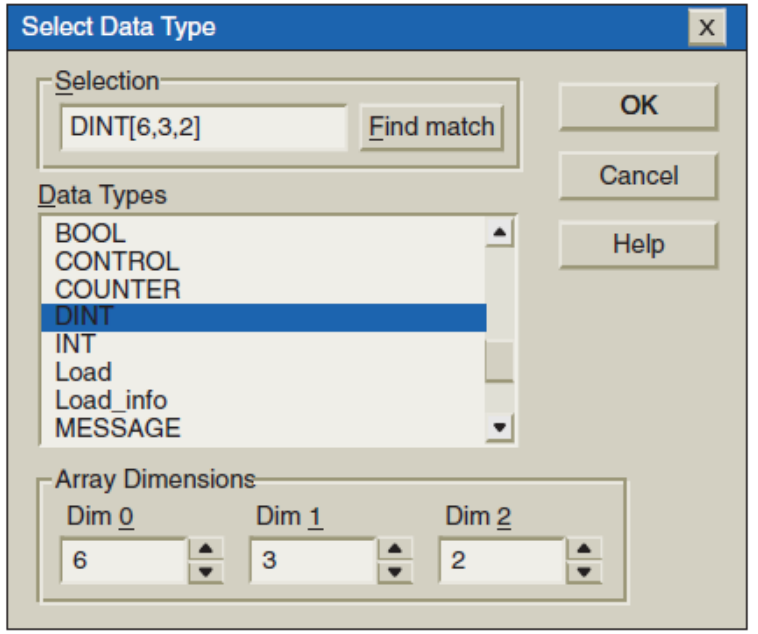


Figure 15-24 Types of arrays.

(Frank D. Petruzella Programmable Logic Controllers 4th edition)

DATA TRANSFER INSTRUCTIONS

Arrays

- Arrays are like tables of values
- A single tag within the array is one element
- The elements start with 0 and extend to the number of elements minus 1
- The use of arrayed data types offers the fastest data throughput (output) from a ControlLogix/CompactLogix processor

Array - Temp
Data Type - INT[5]

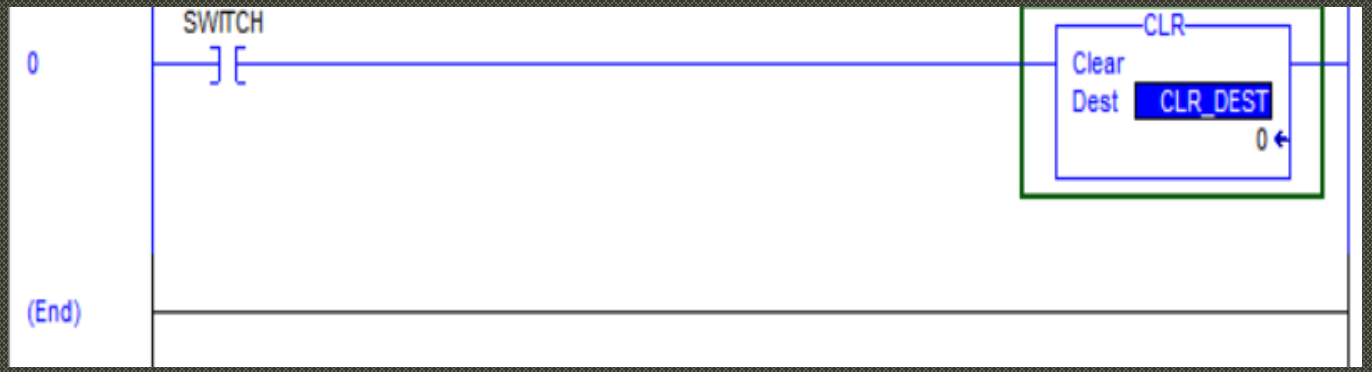
Temp[0]	297
Temp[1]	200
Temp[2]	180
Temp[3]	120
Temp[4]	100

Figure 15-25 Memory layout for a one-dimensional array.

DATA TRANSFER INSTRUCTIONS

Clear Instruction

- The **Clear (CLR)** instruction is an output instruction that can move 0s to the Destination memory tag.
- **This is an output instruction.**



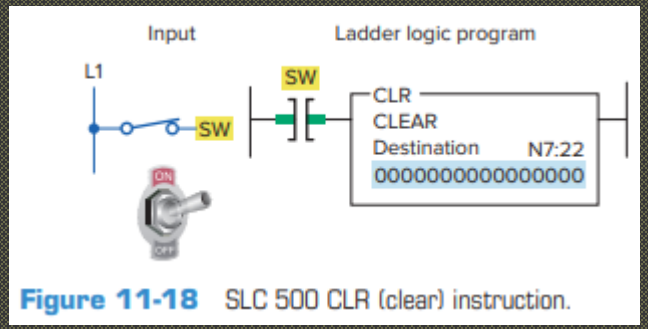


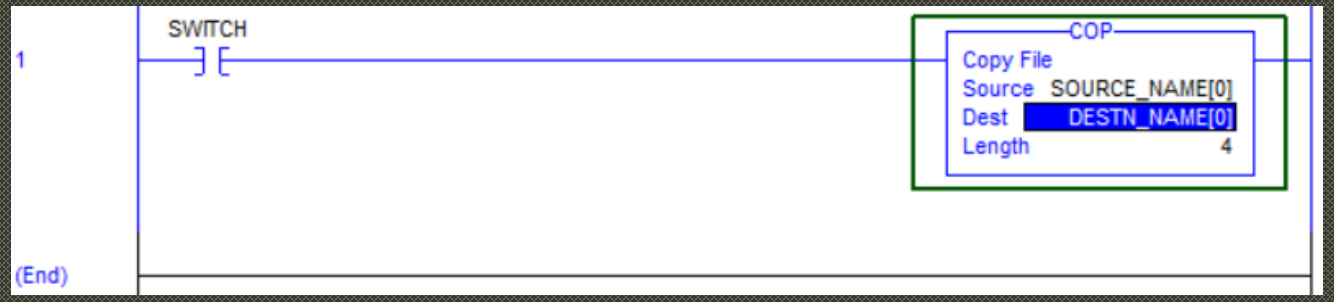
Figure 11-18 SLC 500 CLR (clear) instruction.

(Frank D. Petruzella Programmable Logic Controllers 5th edition)

DATA TRANSFER INSTRUCTIONS

Copy Instruction

The **Copy (COP)** instruction is an output instruction that copies the group of Source elements to another group of Destination elements.



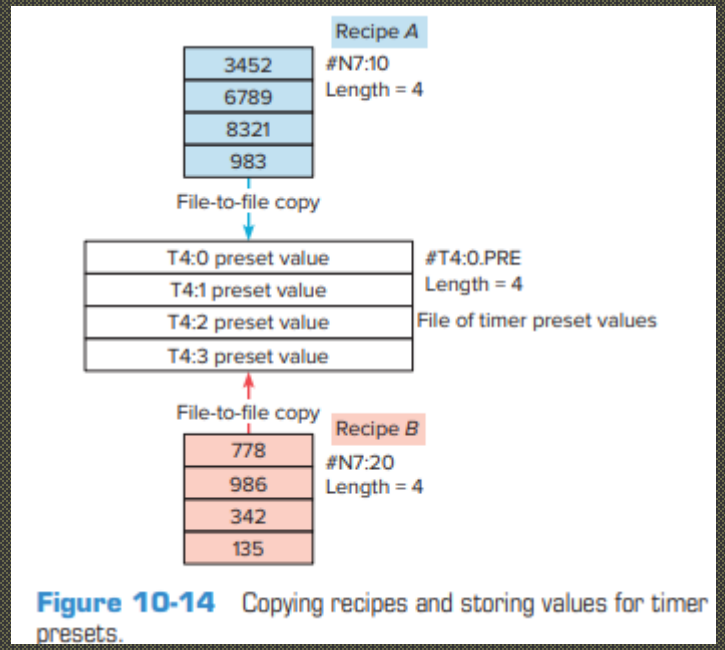


Figure 10-14 Copying recipes and storing values for timer presets.

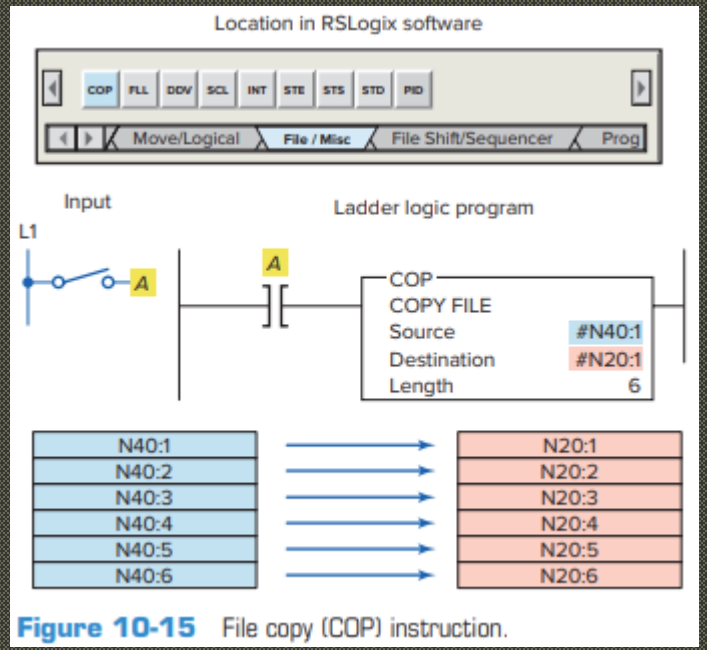
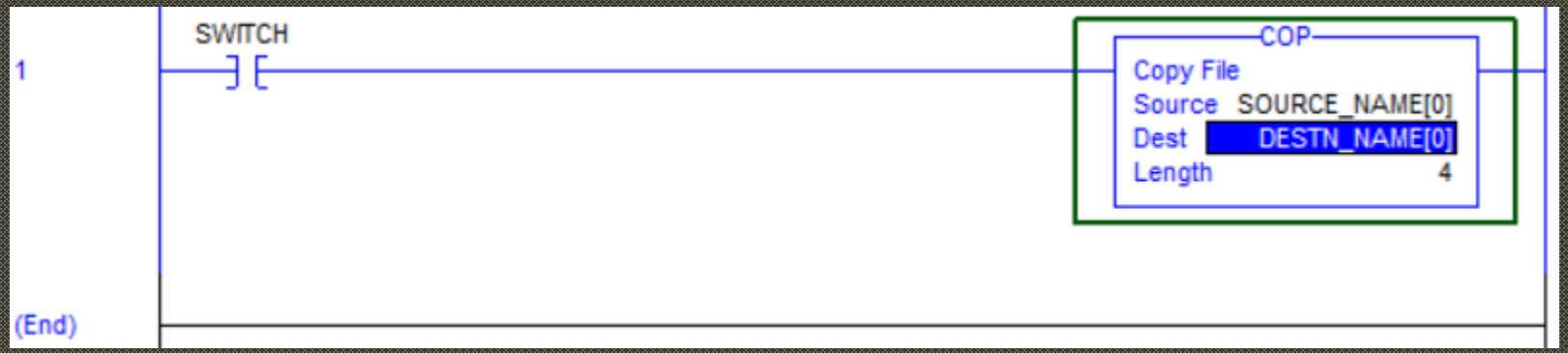


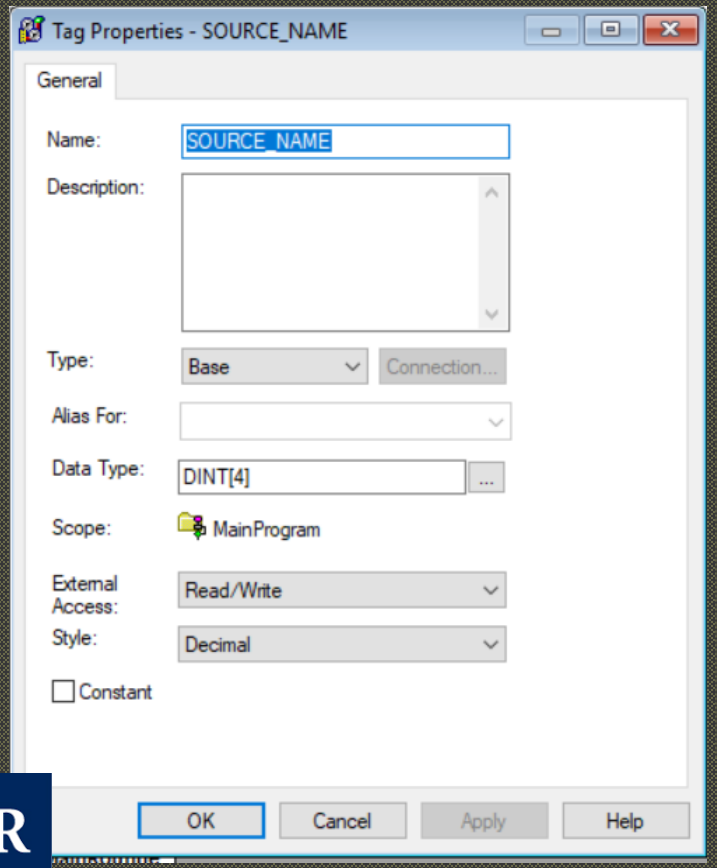
Figure 10-15 File copy (COP) instruction.

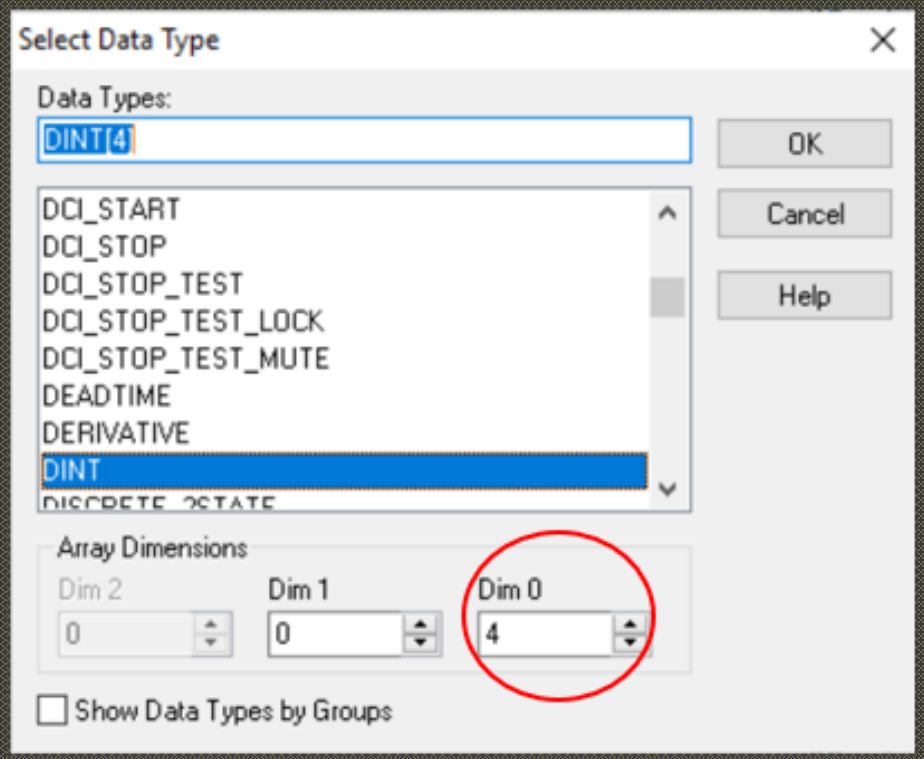
(Frank D. Petruzella Programmable Logic Controllers 5th edition)

DATA TRANSFER INSTRUCTIONS

Copy Instruction

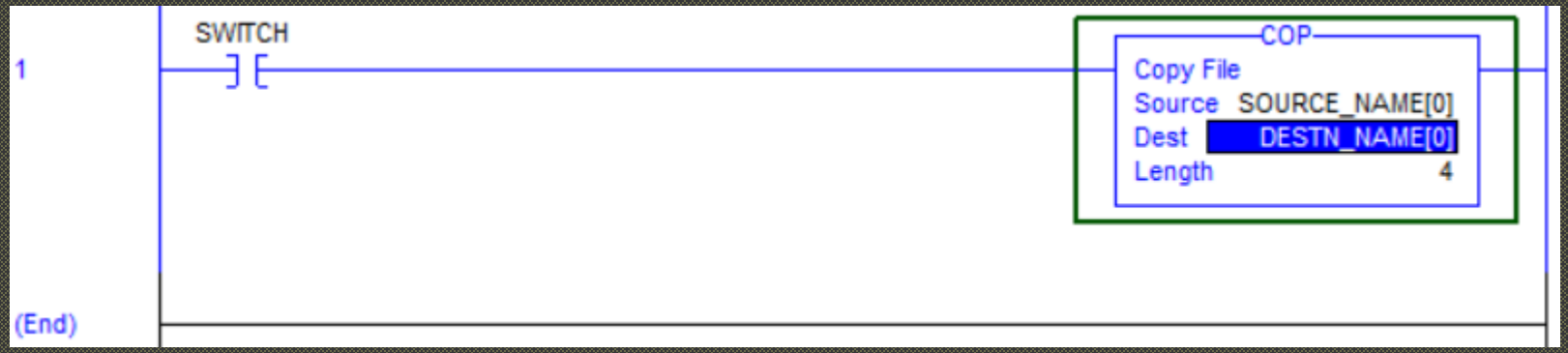






DATA TRANSFER INSTRUCTIONS

Copy Instruction



Scope: MainProgram Show: All Tags

Name	Value	F	Style	Data Type
CLR_DEST	0		Decimal	DINT
SWITCH	0		Decimal	BOOL
SOURCE_NAME	{...}	{.	Decimal	DINT[4]
SOURCE_NAME[0]	0		Decimal	DINT
SOURCE_NAME[1]	0		Decimal	DINT
SOURCE_NAME[2]	0		Decimal	DINT
SOURCE_NAME[3]	0		Decimal	DINT
DESTN_NAME	{...}	{.	Decimal	DINT[4]
DESTN_NAME[0]	0		Decimal	DINT
DESTN_NAME[1]	0		Decimal	DINT
DESTN_NAME[2]	0		Decimal	DINT
DESTN_NAME[3]	0		Decimal	DINT

DATA TRANSFER INSTRUCTIONS

Fill File Instruction

The **Fill File (FLL)** instruction is an output instruction that performs the word-to-file copy in the first scan the rung is true.

- When input A goes true, the value in N15:5 is copied into N20:1 through N20:6
- Because the instruction transfers to the end of the file, the file will be filled with the same data value in each word
- The FLL instruction is frequently used to zero all the data in a file

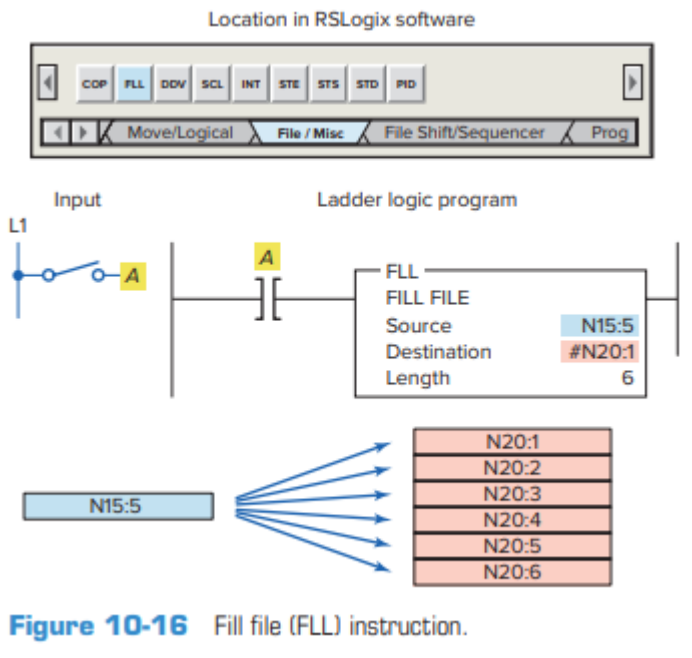


Figure 10-16 Fill file (FLL) instruction.

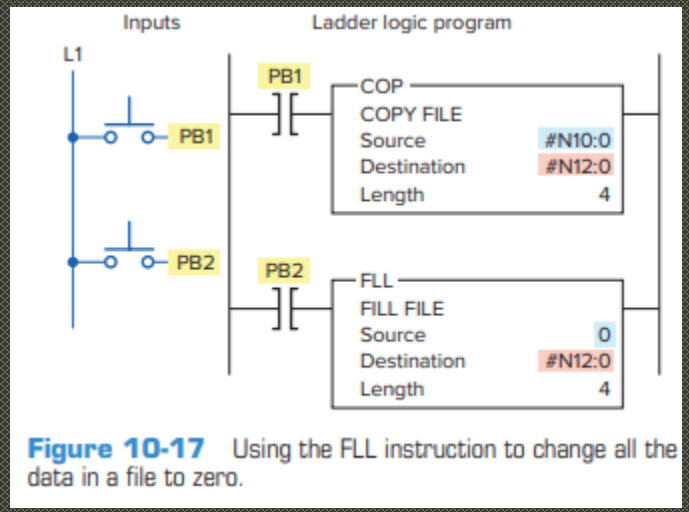
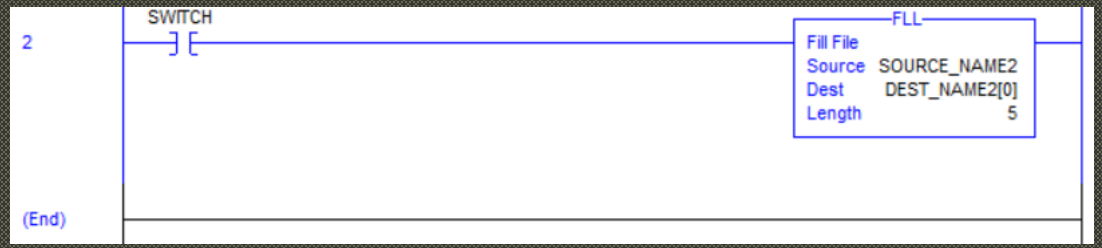
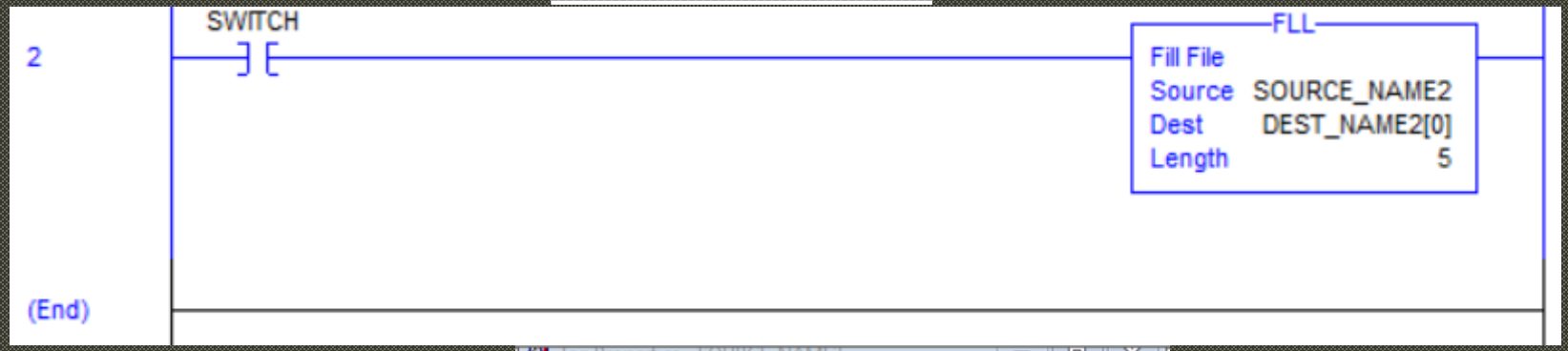


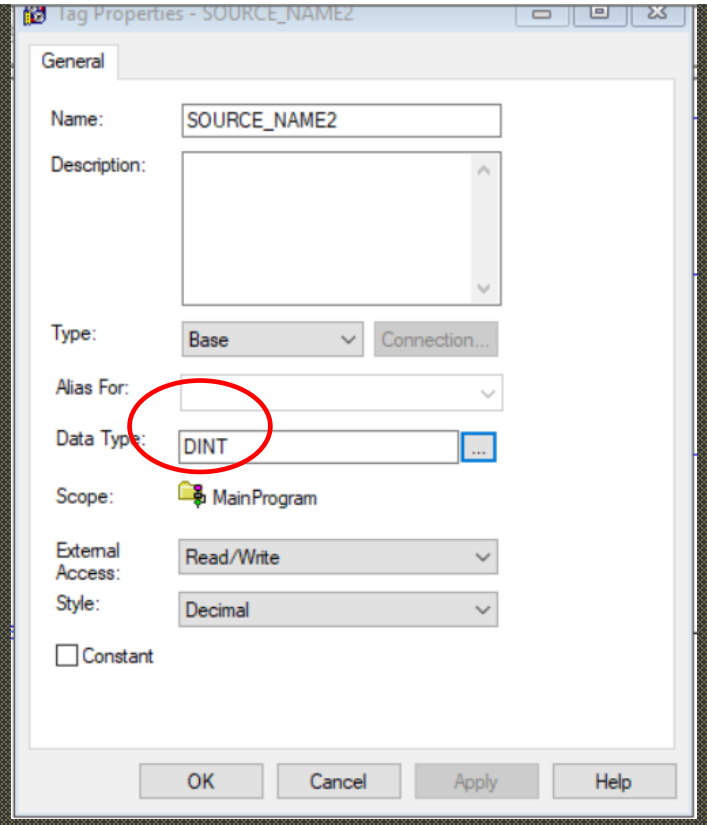
Figure 10-17 Using the FLL instruction to change all the data in a file to zero.



DATA TRANSFER INSTRUCTIONS

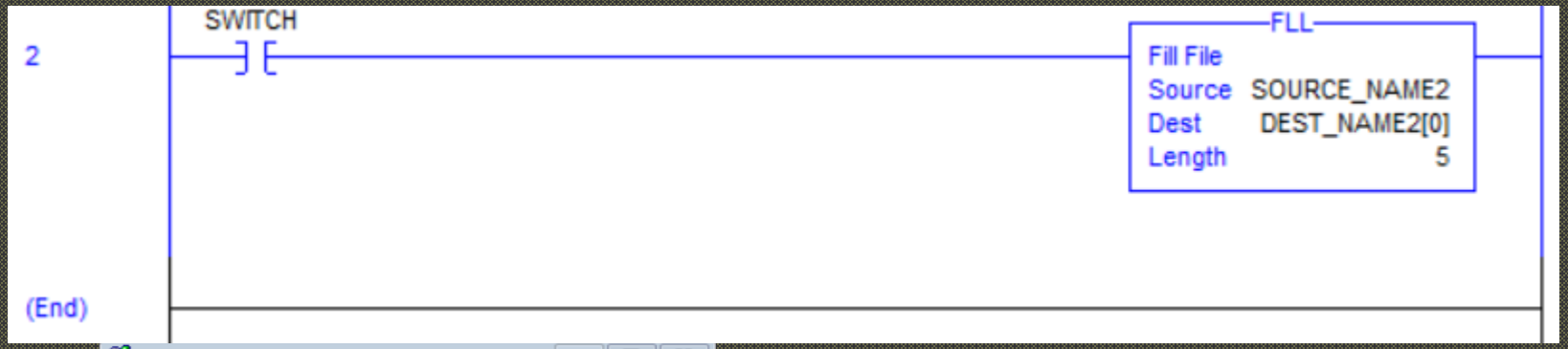
Fill File Instruction

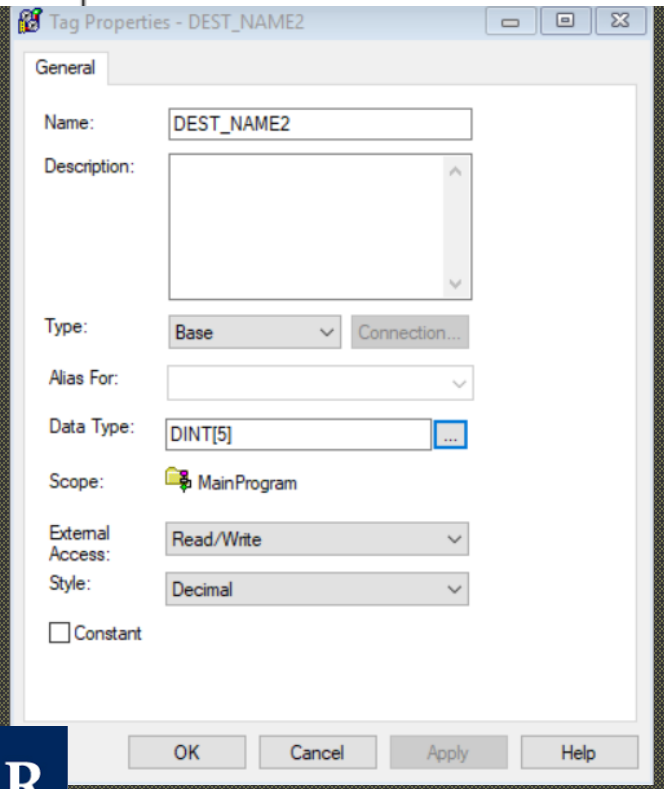


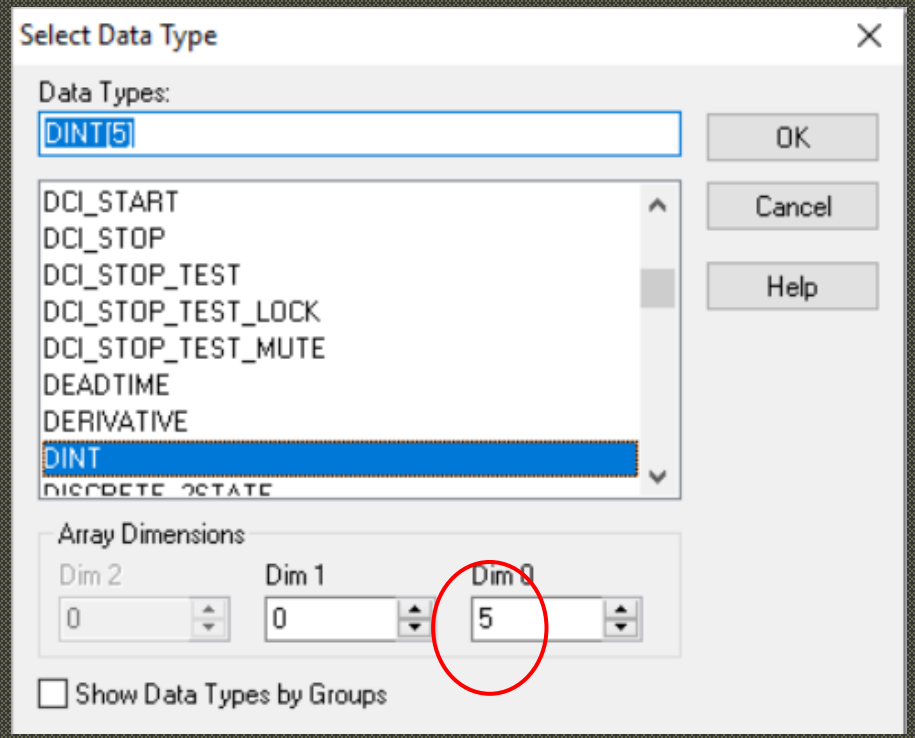


DATA TRANSFER INSTRUCTIONS

Fill File Instruction

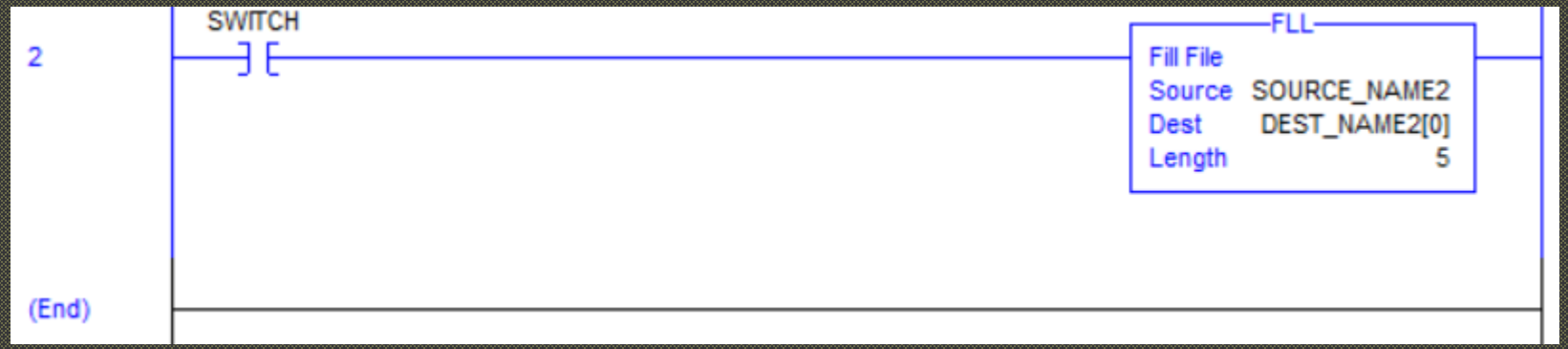






DATA TRANSFER INSTRUCTIONS

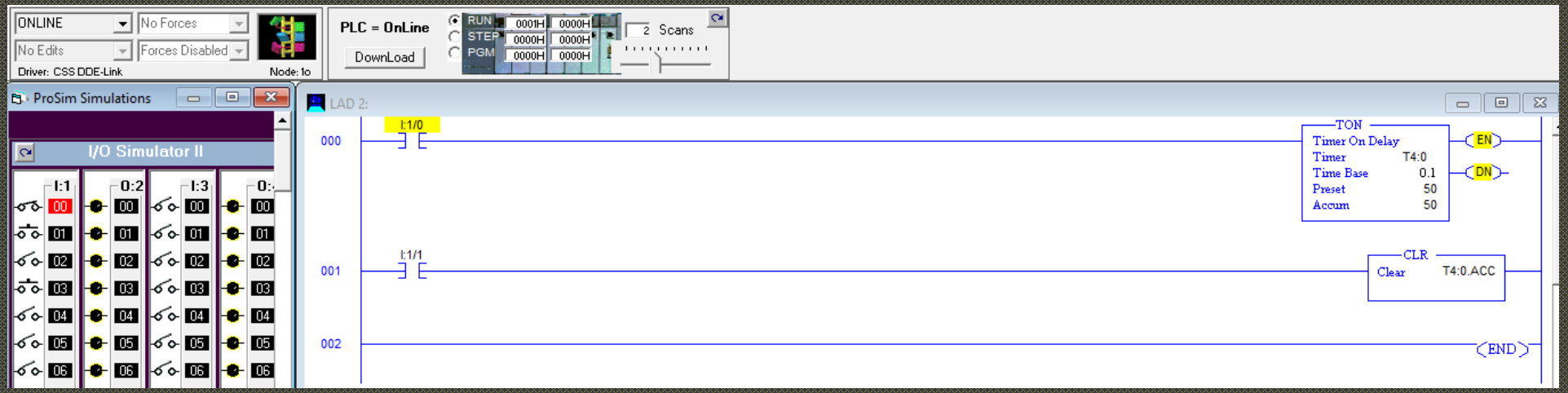
Fill File Instruction



+ SOURCE_NAME2	0		Decimal	DINT
− DEST_NAME2	{...}	{.	Decimal	DINT[5]
+ DEST_NAME2[0]	0		Decimal	DINT
+ DEST_NAME2[1]	0		Decimal	DINT
+ DEST_NAME2[2]	0		Decimal	DINT
+ DEST_NAME2[3]	0		Decimal	DINT
+ DEST_NAME2[4]	0		Decimal	DINT

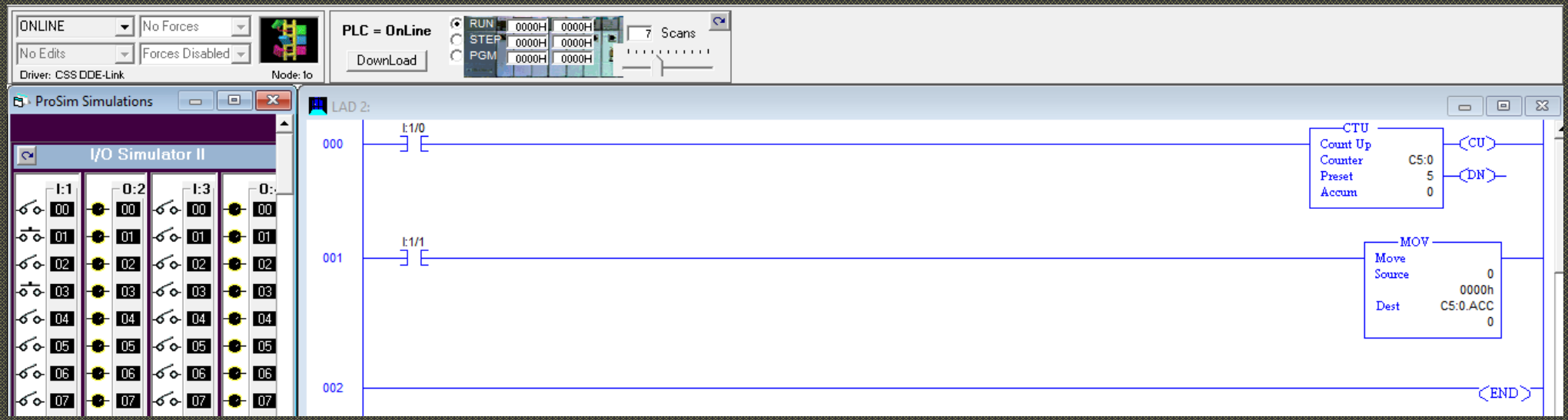
PROBLEMS USING
MATH, COMPARISON AND DATA TRANSFER INSTRUCTIONS

Problem 1. Reset a timer using the CLR instruction.



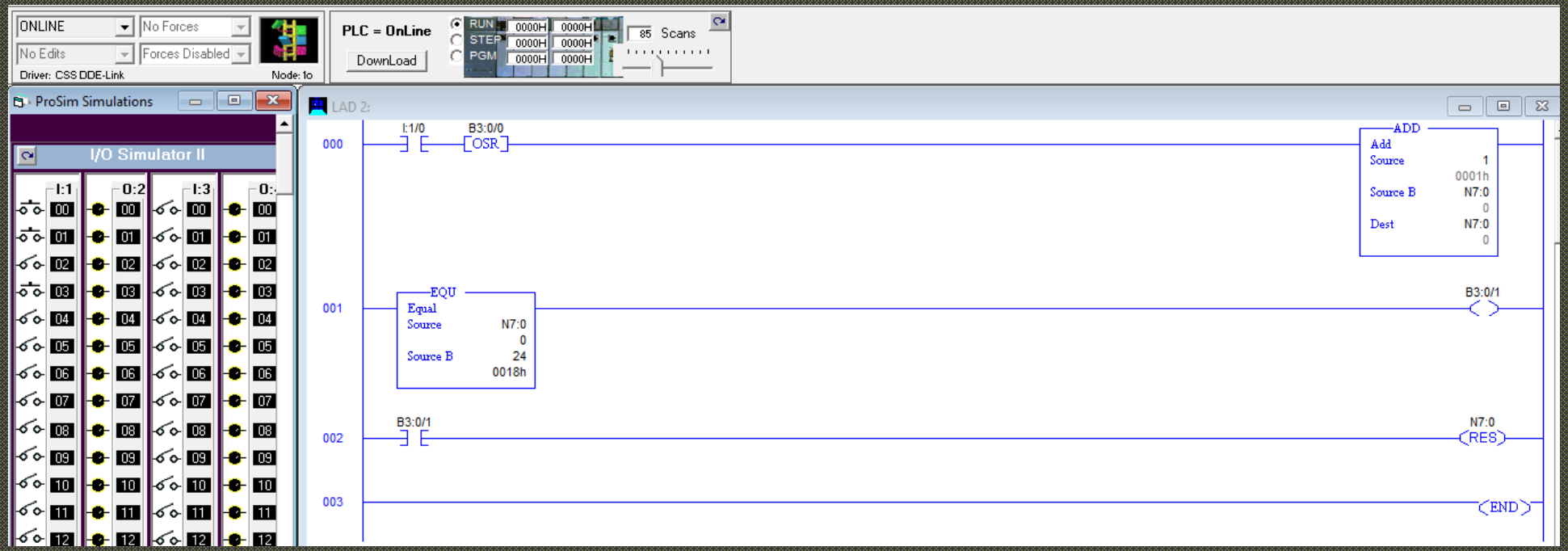
PROBLEMS USING
MATH, COMPARISON AND DATA TRANSFER INSTRUCTIONS

Problem 2. Reset a counter using the MOV instruction.



PROBLEMS USING
MATH, COMPARISON AND DATA TRANSFER INSTRUCTIONS

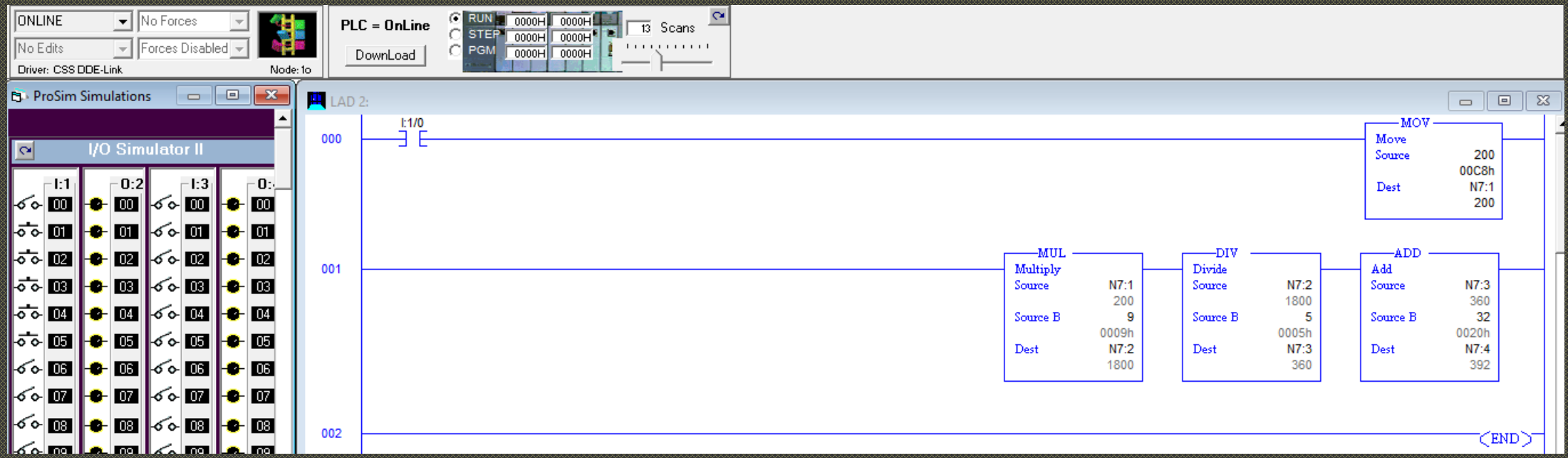
Problem 3. Using ADD instruction, write a counter program to count the number of bottles passing the sensor at the end of the conveyor. Reset the "counter" when the number of boxes reaches 24.



PROBLEMS USING
MATH, COMPARISON AND DATA TRANSFER INSTRUCTIONS

Problem 4. Using math and comparison instructions, write a program to convert the temperature from Celsius to Fahrenheit.

$$F^0 = 1.8\,C^0 + 32$$



PROBLEMS USING
MATH, COMPARISON AND DATA TRANSFER INSTRUCTIONS

Problem 5. Write the program in Problem 4, using the **CPT** instruction instead of math instructions.

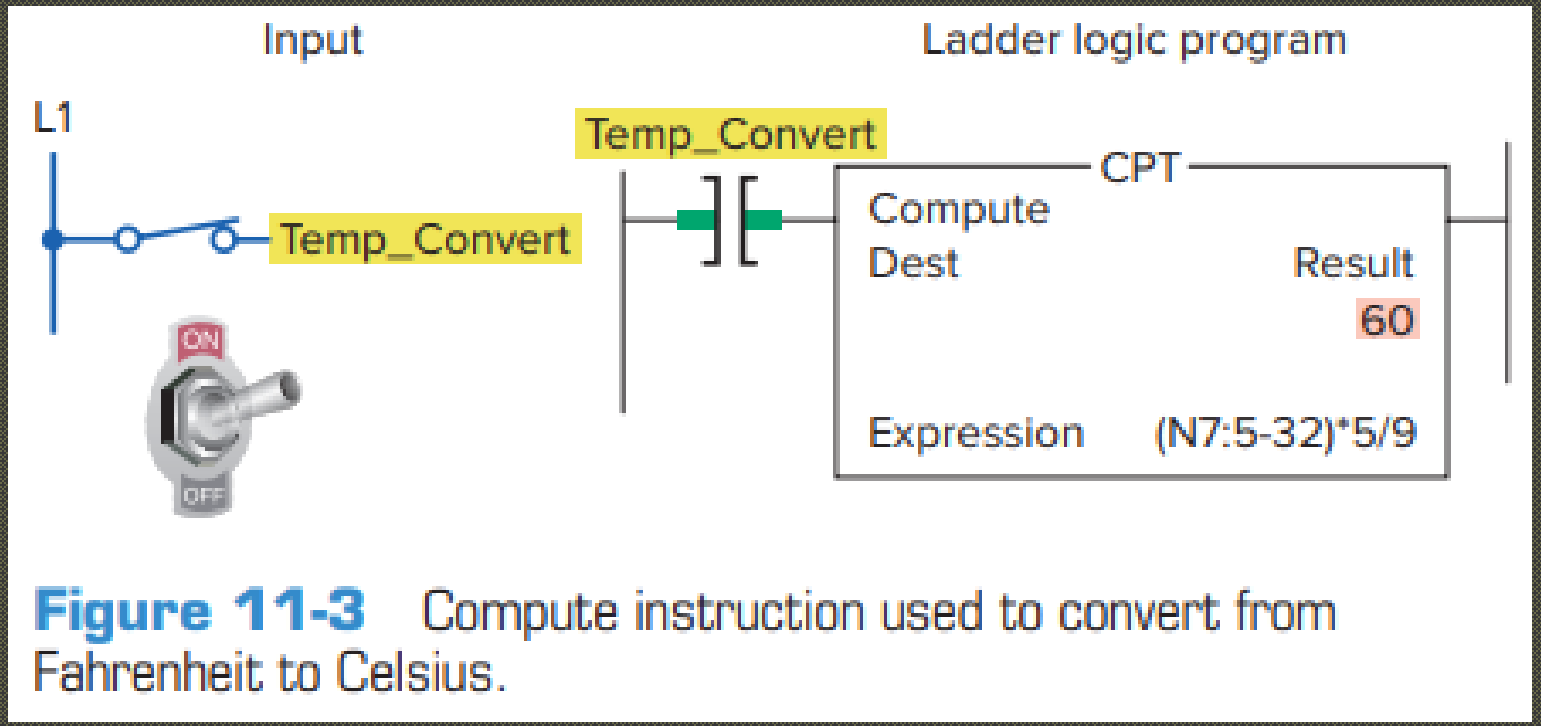
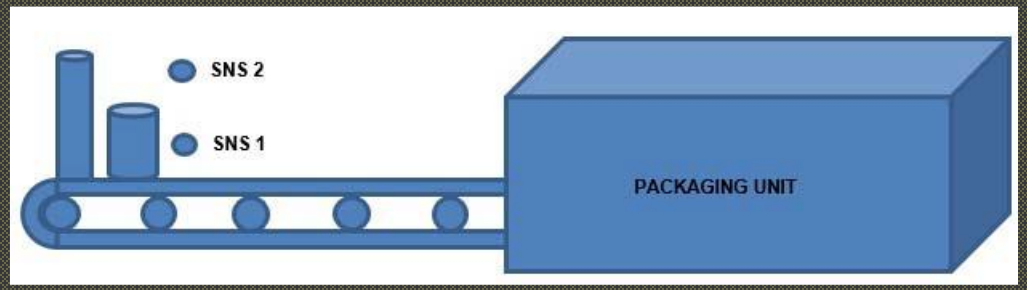


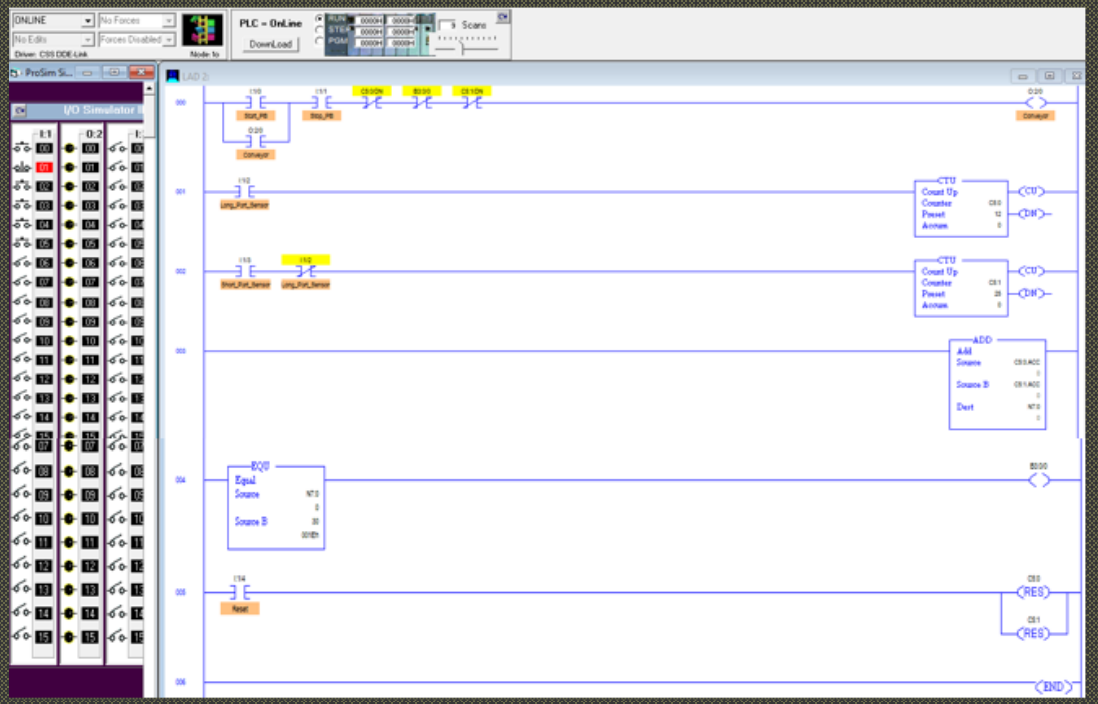
Figure 11-3 Compute instruction used to convert from Fahrenheit to Celsius.

PROBLEMS USING
MATH, COMPARISON AND DATA TRANSFER INSTRUCTIONS

Problem 6. Two optical sensors are installed in the beginning of the conveyor to determine short and long parts.

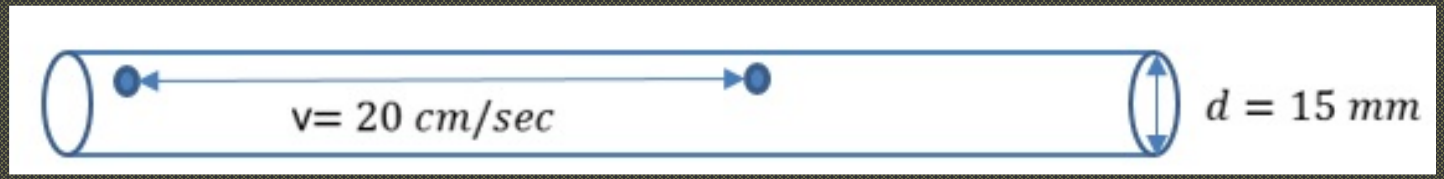
- The short part is recognized by the sensor 1 only, while the long one, by both sensors 1 and sensor 2.
- Momentary N.O. Start and N.C. Stop pushbuttons control the motion of the conveyor.
- A packaging unit at the end of the conveyor packs the parts into two boxes for both sizes.
- The conveyor must stop automatically if the number of long parts reaches 12, or the number of short parts reaches 25, or number of parts in both boxes is 30.
- All accumulated numbers must be reset by the N.O. Reset PB, and the system must be ready for the new batch.





PROBLEMS USING
MATH, COMPARISON AND DATA TRANSFER INSTRUCTIONS

Problem 7. The liquid flows through a pipe with a diameter of 15 mm with the velocity of 20 cm/sec. Write a program to calculate the amount of the liquid flowing out of the pipe in 15 minutes in liters (1L = 1000 $cm3$).



$$V=\frac{\pi * D^2}{4} * v * t$$

PROBLEMS USING
MATH, COMPARISON AND DATA TRANSFER INSTRUCTIONS

Problem 8. Write a traffic lights control program with:

- RED – 8 sec, Green – 8 sec and AMBER – 4 seconds
- WALK light will be solid ON during the first 4 seconds of GREEN
- DO NOT WALK will be solid ON at RED light, and last 2 seconds of AMBER
- The DO NOT WALK will flash at the last 4 seconds of GREEN and first 2 seconds of AMBER
- **Use one timer to control the traffic lights**
- The flashing effect still can be written using 2 cascade timers sequence

PROGRAMMABLE LOGIC CONTROLLERS

Computation, Comparison and Data Transfer Instructions, Arrays

Erickson, K. (2016) Programmable logic controllers: An emphasis on design and application (3rd edition). Rolla MO: Dogwood Valley Press.

Chapter 7

Petruzella,F. (2017) Programmable Logic Controllers (5th Edition). McGraw-Hill Education, NY.

Chapter 10
Chapter 11
Chapter 15 Array
Chapter 15 Part 5

Thank you!

Discussions?