Unit – 2

Programming of PLC

Contents

Fundamentals of logic

- The PLC, like all digital equipment, operates on the binary principle.
- The term **binary principle** refers to the idea that many things can be thought of as existing in only one of two states.
- These states are 1 and 0. The 1 and 0 can represent ON or OFF, open or closed, true or false, high or low, or any other two conditions.
- The key to the speed and accuracy with which binary information can be processed is that there are only two states, each of which is distinctly different.
- There is no in-between state so when information is processed the outcome is either yes or no.

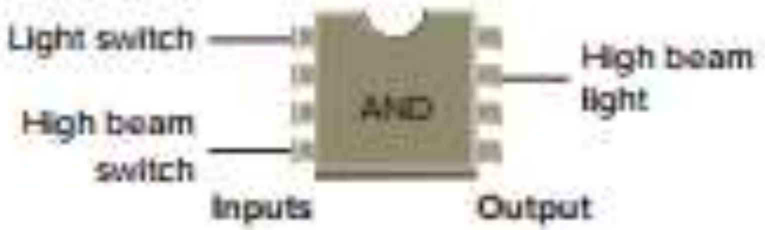


Figure 4-1 The logical AND.

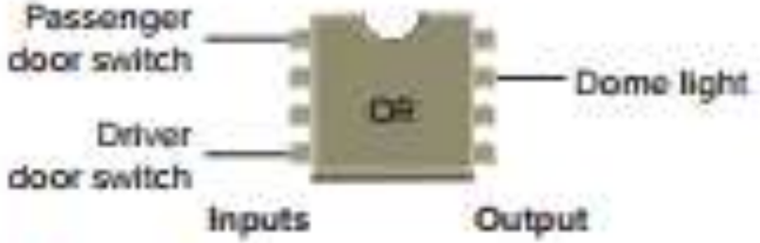
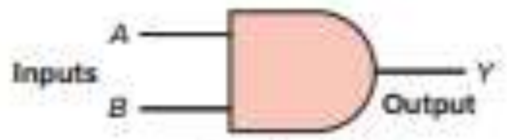


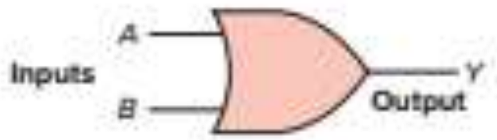
Figure 4-2 The logical OR.



Two-input AND gate symbol

AND truth table

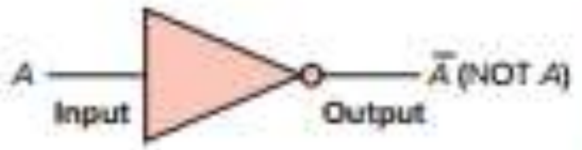
Inputs		Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



Two-input OR gate symbol

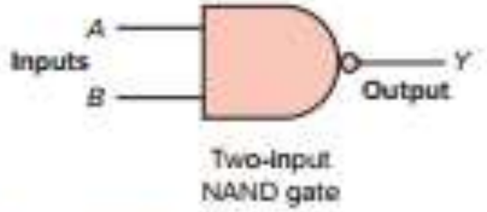
OR truth table

Inputs		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1



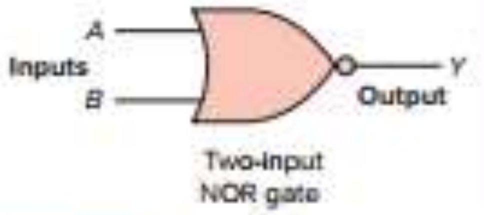
NOT truth table

A	NOT A
0	1
1	0



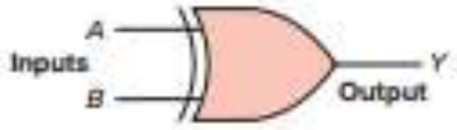
NAND truth table		
Inputs		Output
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Figure 4-12 NAND gate symbol and truth table.



NOR truth table		
Inputs		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Figure 4-13 NOR gate symbol and truth table.



Truth table

Inputs		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Figure 4.14 The XOR gate symbol and truth table.

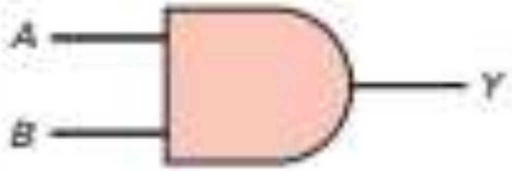
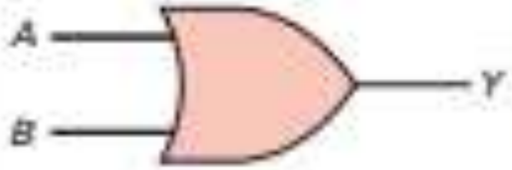
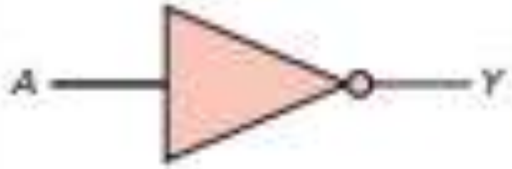
Logic symbol	Logic statement	Boolean equation	Boolean notations
	Y is 1 if A and B are 1	$Y = A \cdot B$ or $Y = AB$	Symbol / Meaning $\cdot$ and $+$ or $\overline{\ }$ not $\circ$ invert $=$ result in
	Y is 1 if A or B is 1	$Y = A + B$	
	Y is 1 if A is 0 Y is 0 if A is 1	$Y = \overline{A}$	

Figure 4-15 Boolean algebra as related to AND, OR, and NOT functions.

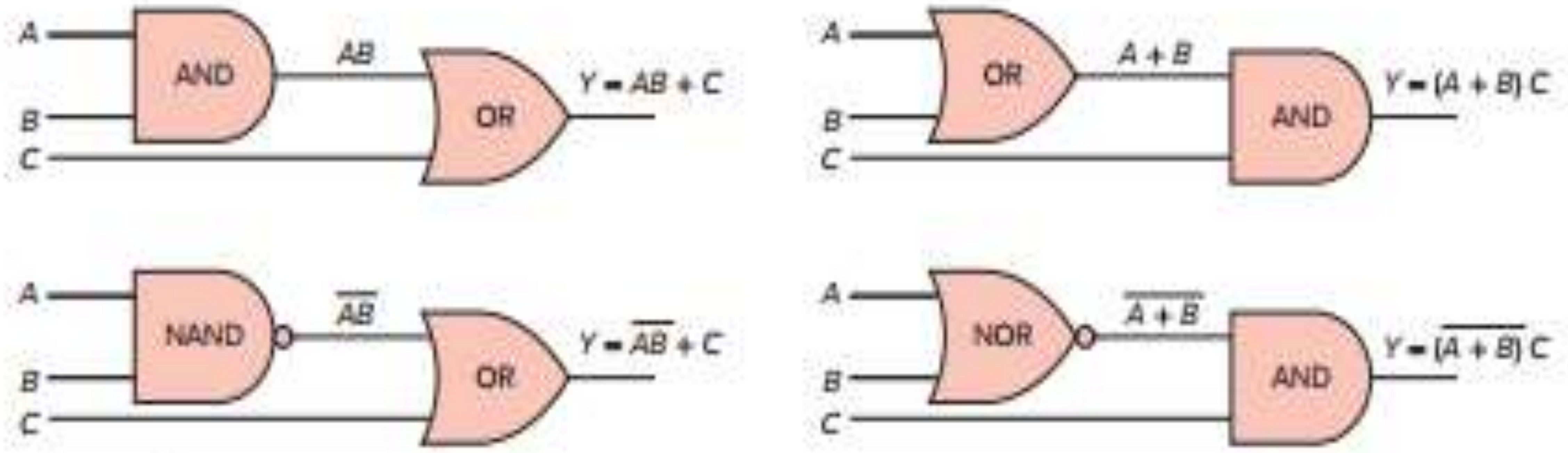


Figure 4-17 Logic operators used in combination to form Boolean equations.

Hardwired Logic versus Programmed Logic

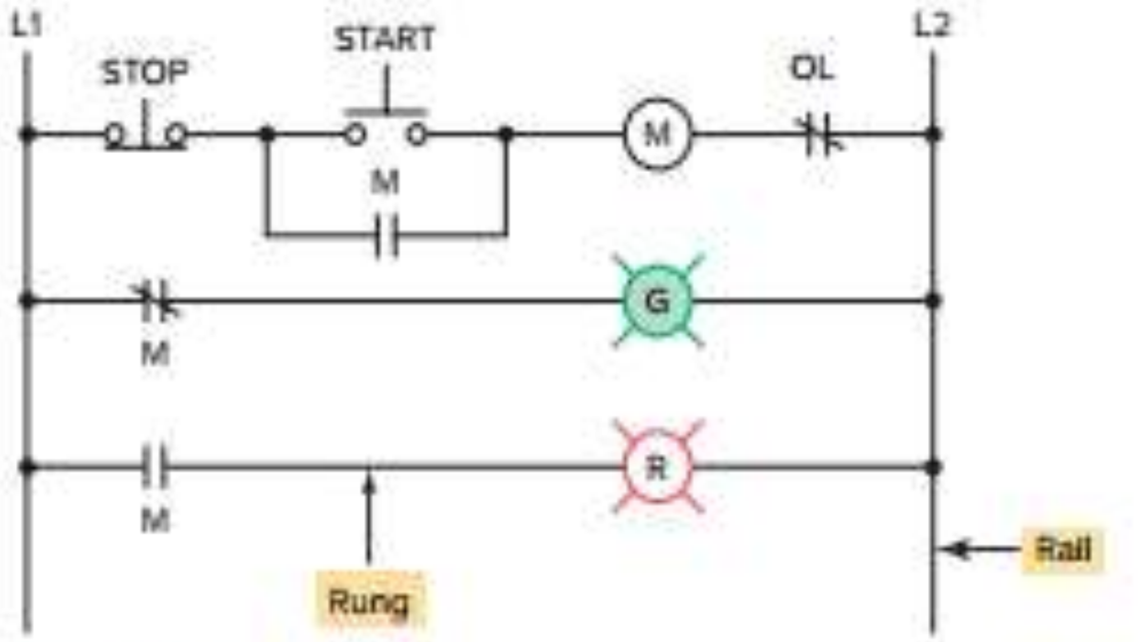


Figure 4-22 Motor stop/start relay ladder schematic.

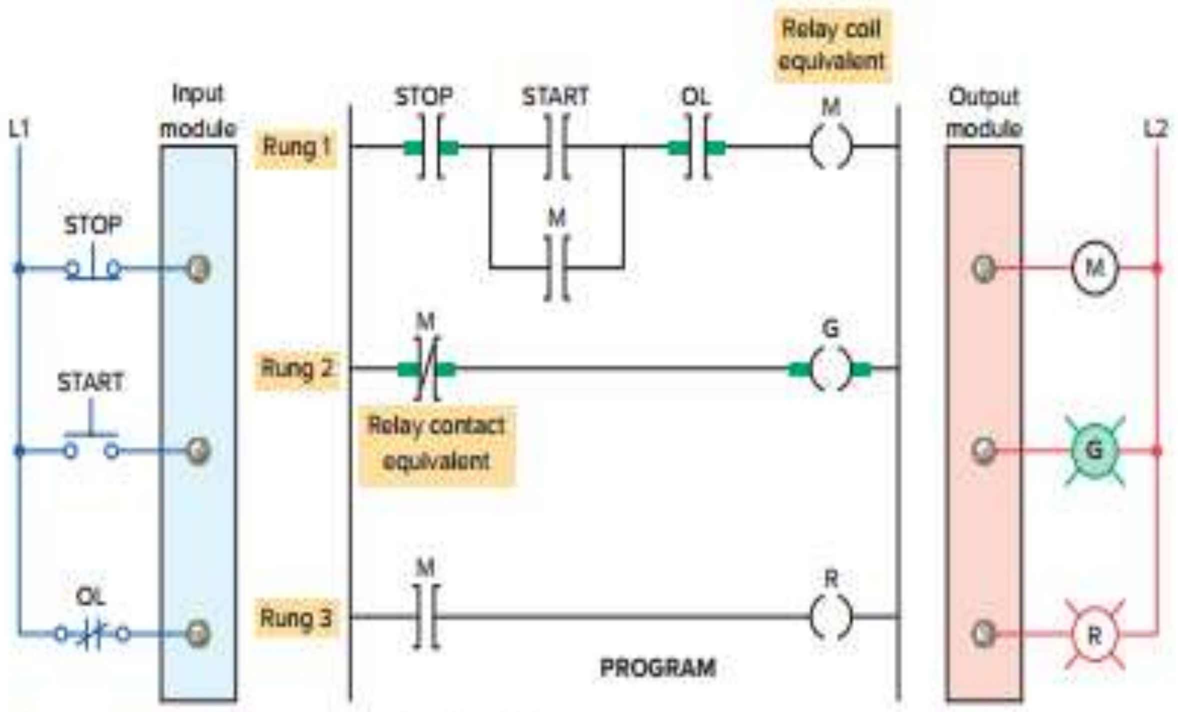
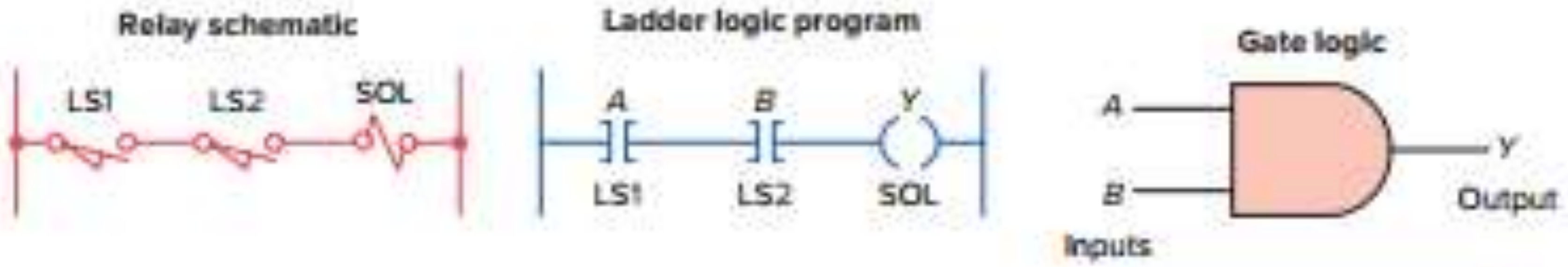
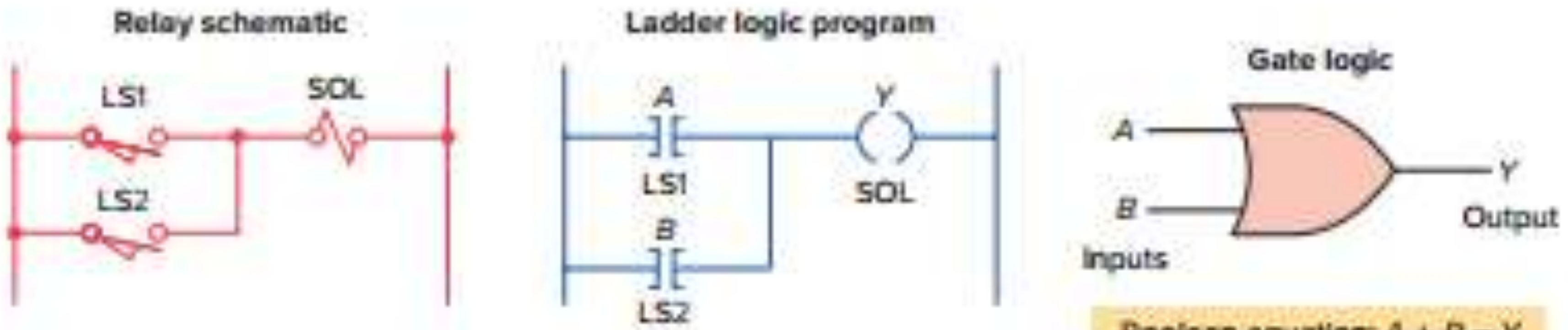


Figure 4-23 Motor stop/start ladder logic program.



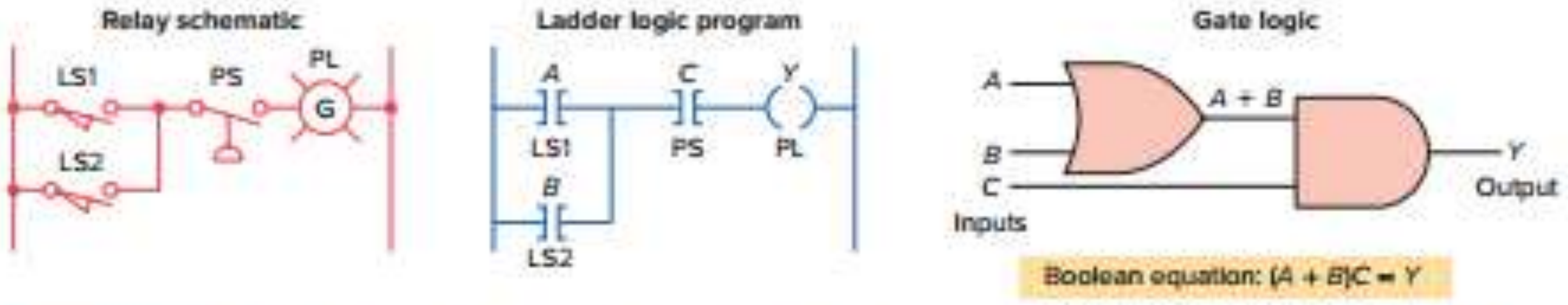
Boolean equation: $AB = Y$

Example 4-1 Two limit switches connected in series and used to control a solenoid valve.

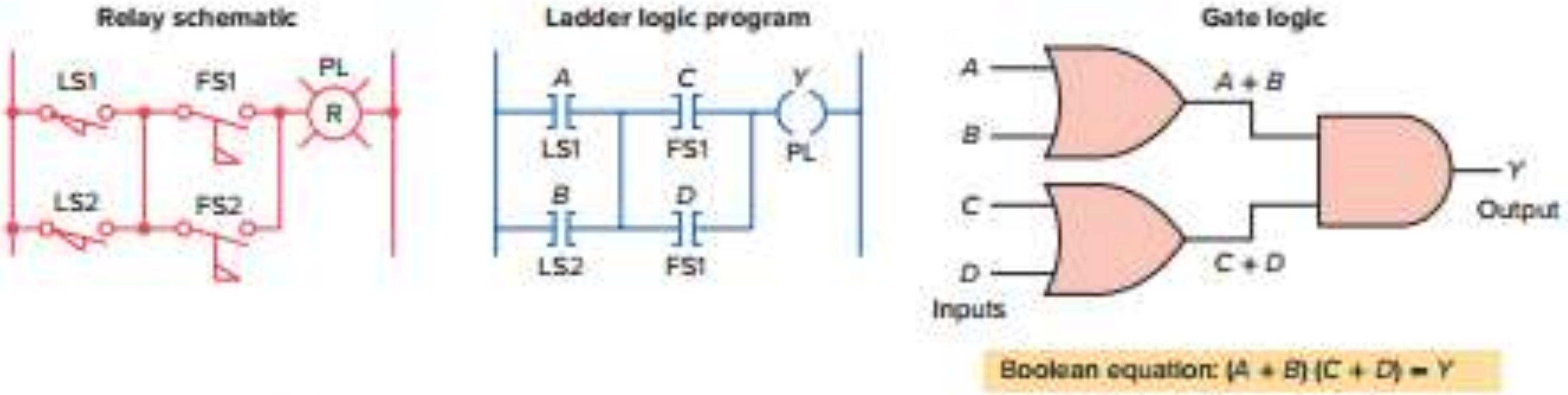


Boolean equation: $A + B = Y$

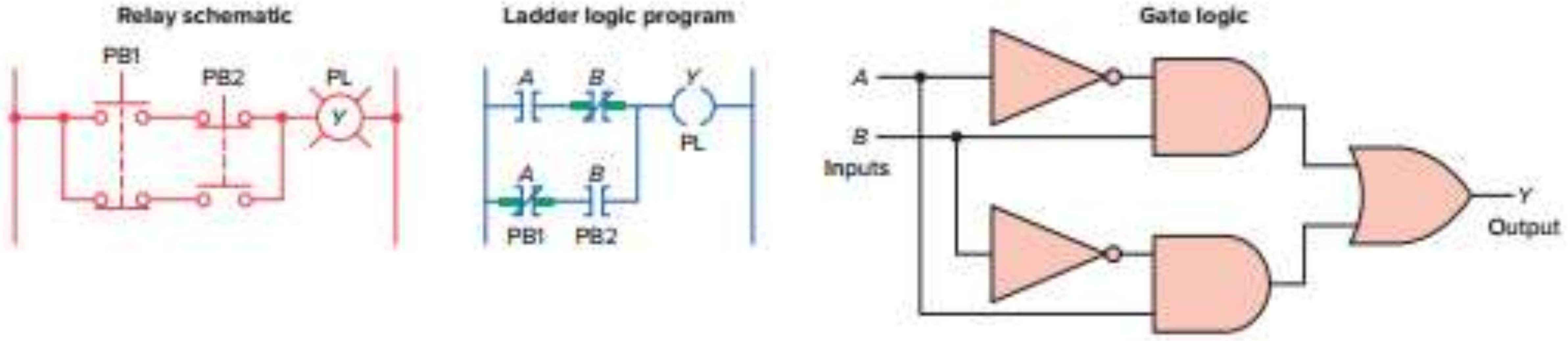
Example 4-2 Two limit switches connected in parallel and used to control a solenoid valve.



Example 4-3 Two limit switches connected in parallel with each other and in series with a pressure switch.



Example 4-4 Two limit switches connected in parallel with each other and in series with two sets of flow switches (that are connected in parallel with each other), and used to control a pilot light.



Boolean equation: $\bar{A}B + A\bar{B} = Y$
$A \oplus B = Y$

Example 4-8 Exclusive-OR circuit. The output lamp of this circuit is ON only when pushbutton A or B is pressed, but not both. This circuit has been programmed using only the normally open A and B pushbutton contacts as the inputs to the program.

Program Scan

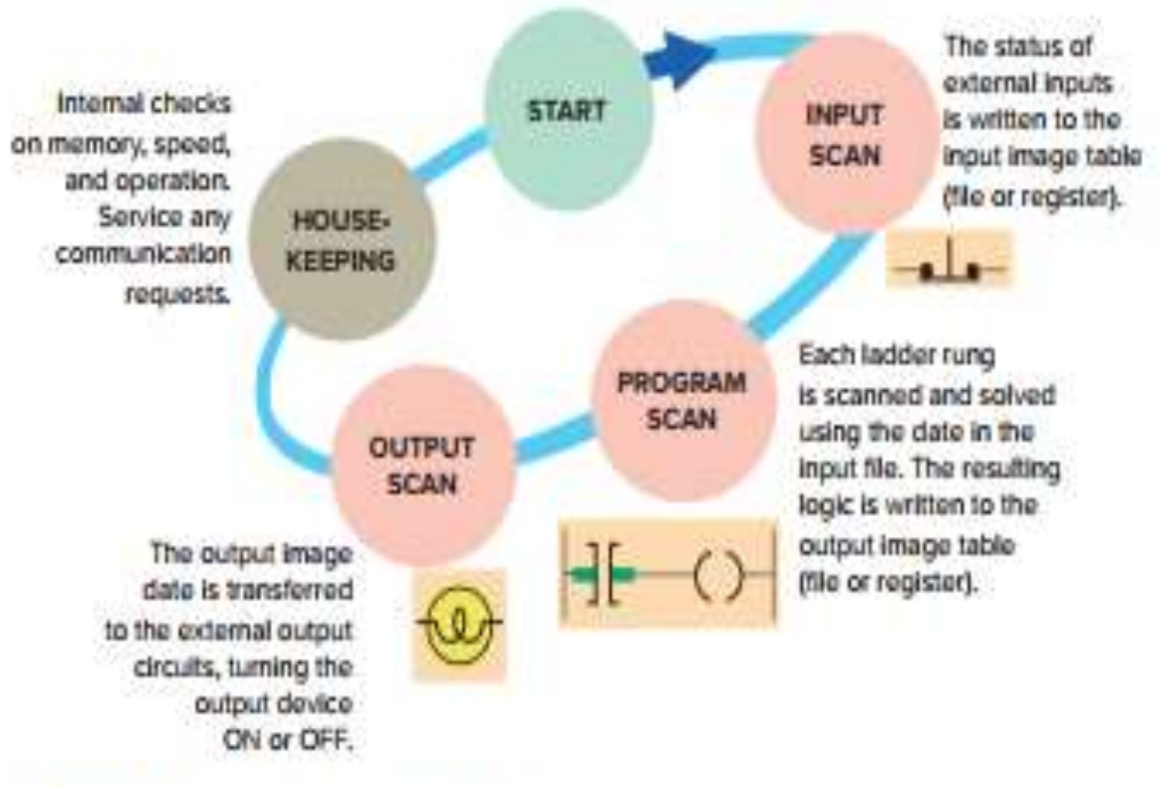
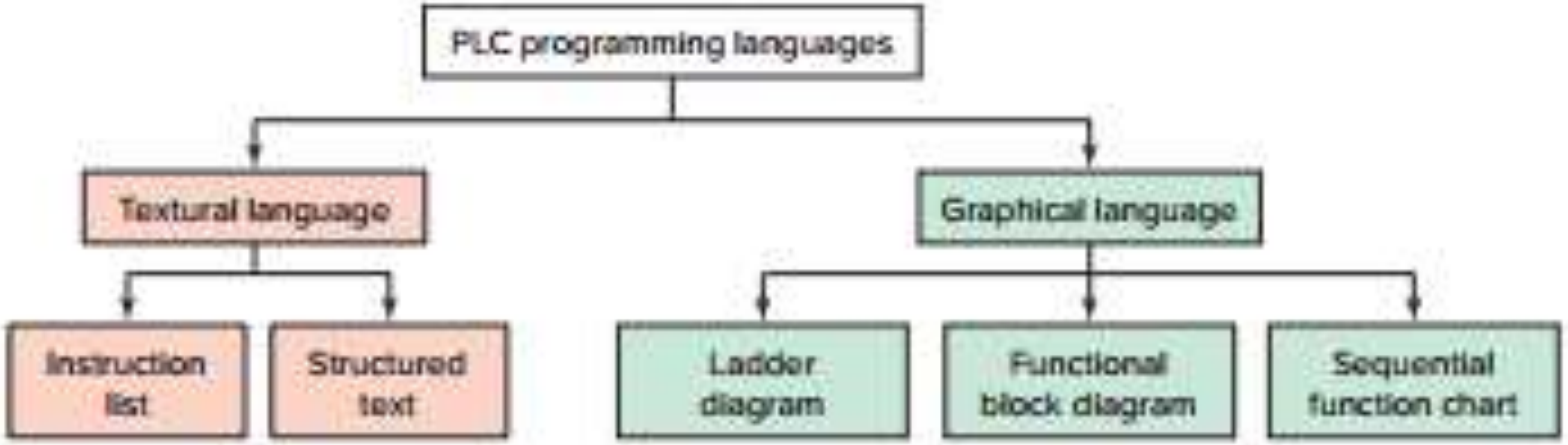

START
INPUT SCAN
The status of external inputs is written to the input image table (file or register).
HOUSE-KEEPING
Internal checks on memory, speed, and operation. Service any communication requests.
PROGRAM SCAN
Each ladder rung is scanned and solved using the date in the input file. The resulting logic is written to the output image table (file or register).
OUTPUT SCAN
The output image date is transferred to the external output circuits, turning the output device ON or OFF.

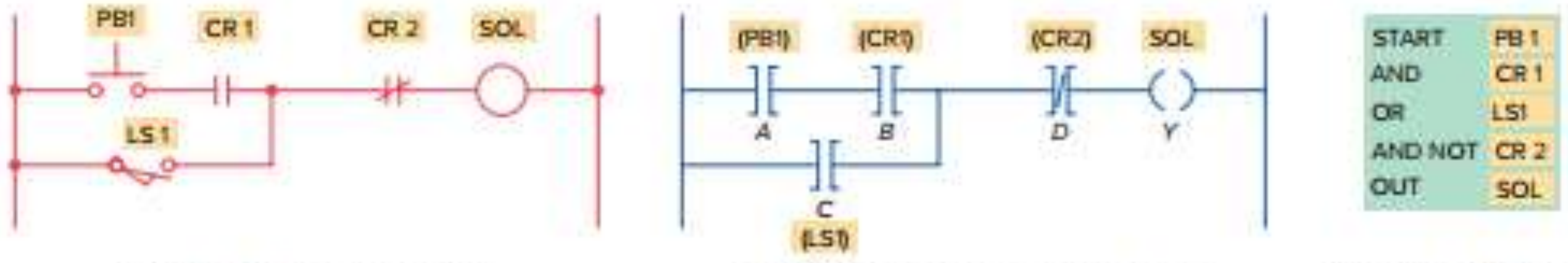
Figure 5-8 PLC program scan cycle.

- When a PLC executes a program, it must know—in real time—when external devices controlling a process are changing.
- During each operating cycle, the processor reads all the inputs, takes these values, and energizes or deenergizes the outputs according to the user program.
- This process is known as a ***program scan cycle.***

PLC Programming Languages



- **Ladder Diagram (LD)**—a symbolic depiction of instructions arranged in rungs similar to ladderformatted schematic diagrams.
- **Function Block Diagram (FBD)**—a graphical depiction of process flow using simple and complex interconnecting blocks. (also called as Graph Chart)
- **Sequential Function Chart (SFC)**—a graphical depiction of interconnecting steps, actions, and transitions.
- **Instruction List (IL)**—a low-level, text-based language that uses mnemonic instructions.
- **Structured Text (ST)**—a high-level, text-based language such as BASIC, C, or PASCAL specifically developed for industrial control applications.



```
START    PB 1
AND      CR 1
OR       LS1
AND NOT  CR 2
OUT      SOL
```

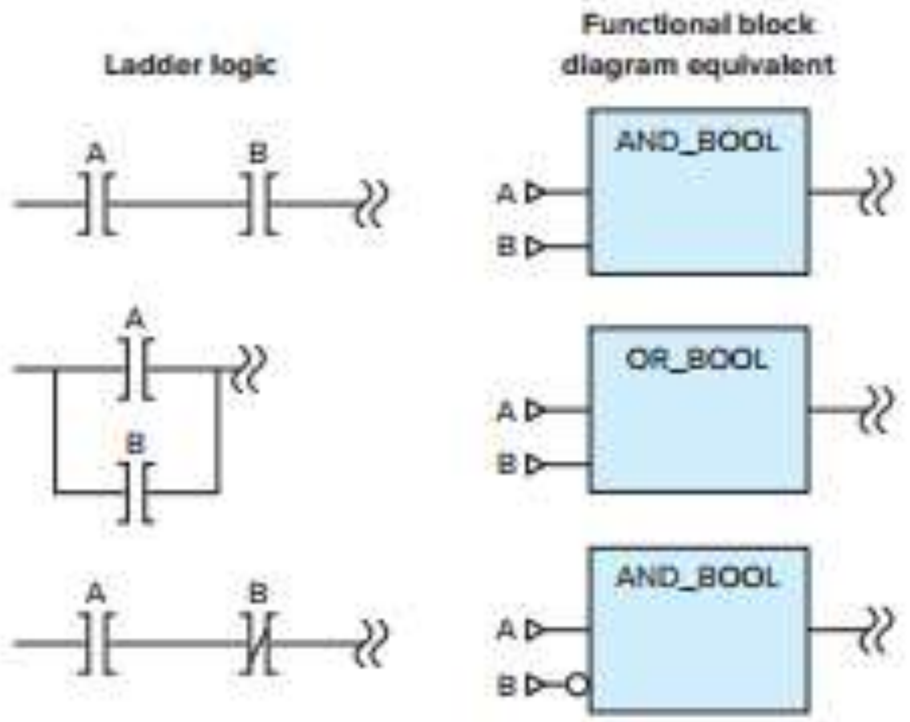
(a) Hardwired relay control circuit

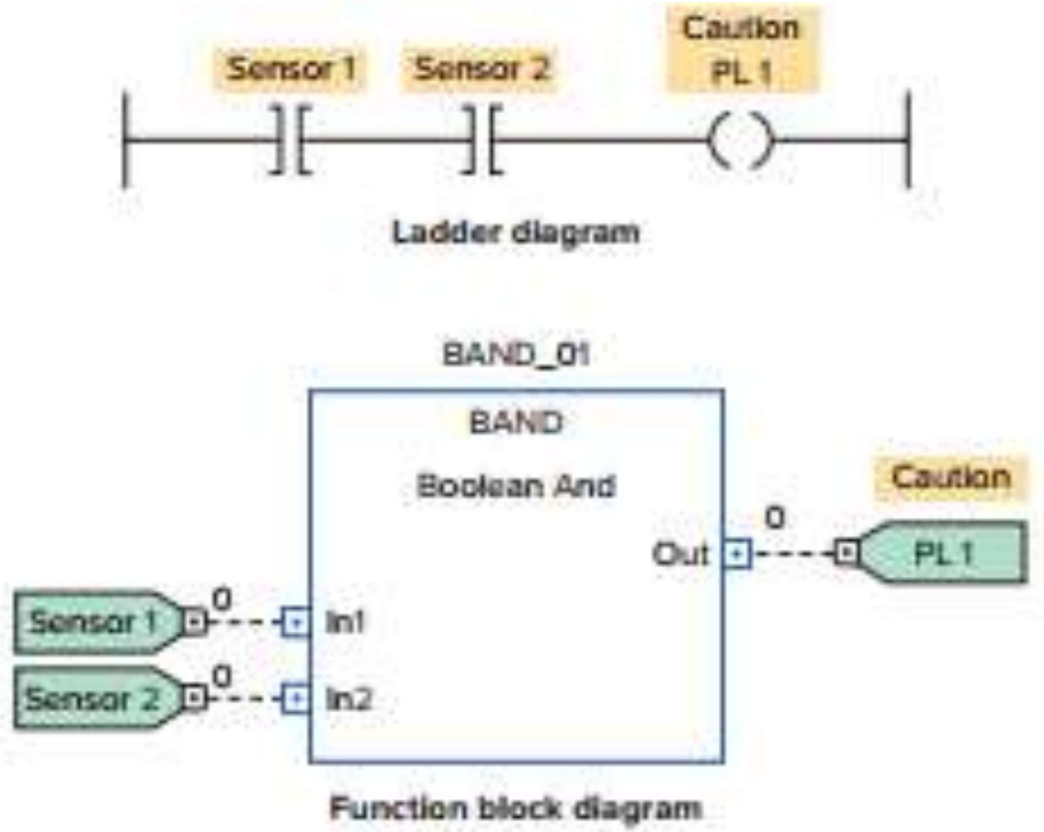
(b) Equivalent ladder diagram (LD) program

(c) Equivalent instruction list (IL) program

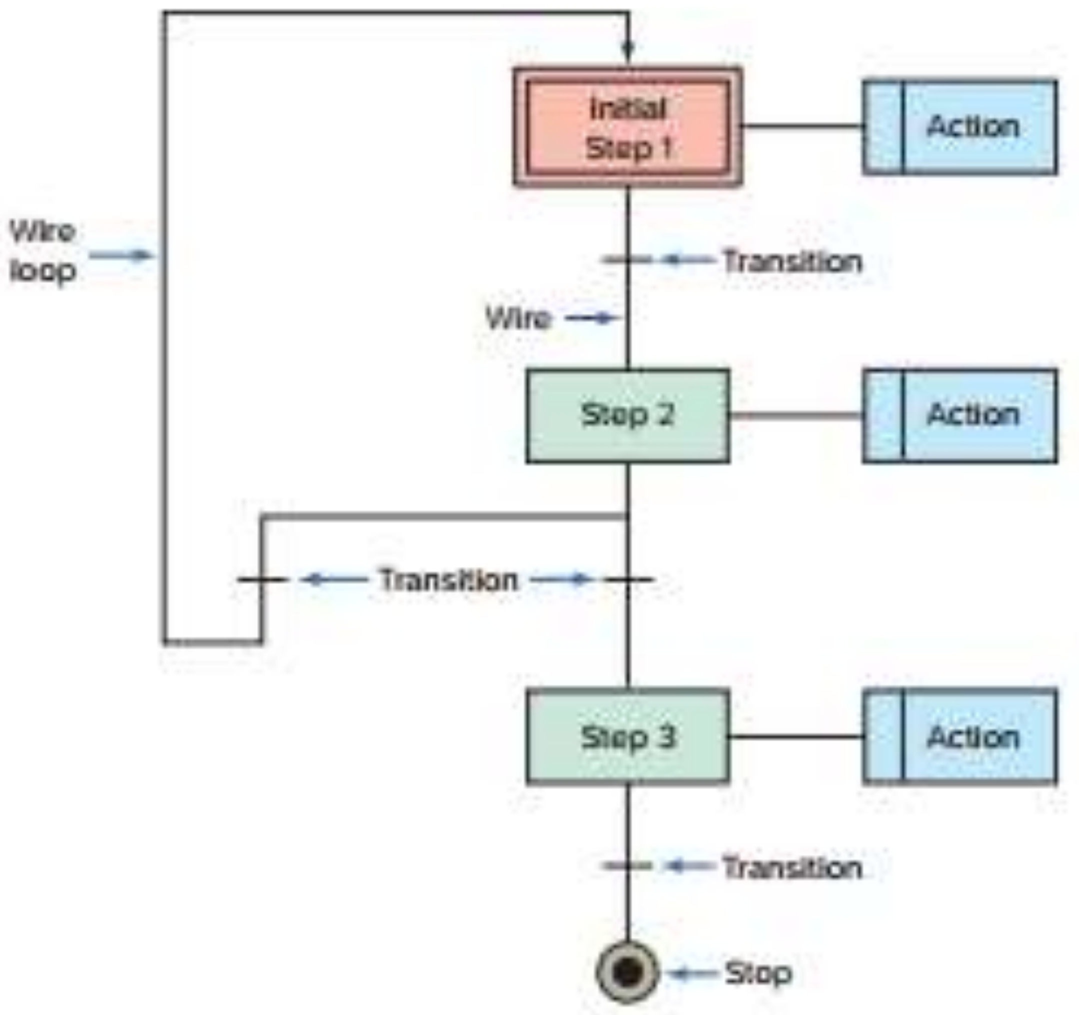
Figure 5-15 Comparison of ladder diagram and instruction list programming.

Functional block Diagram

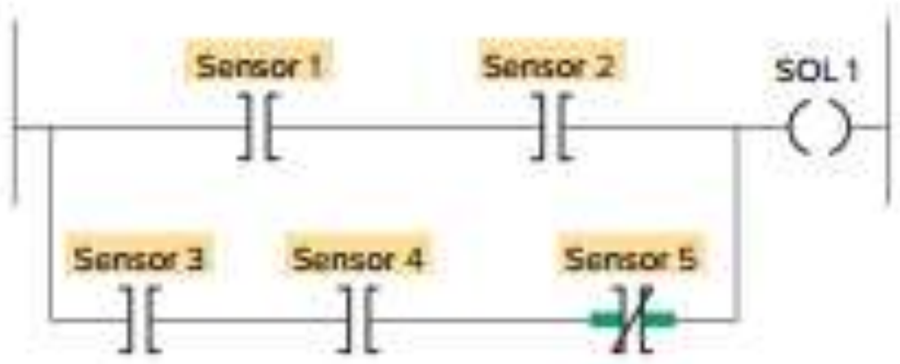




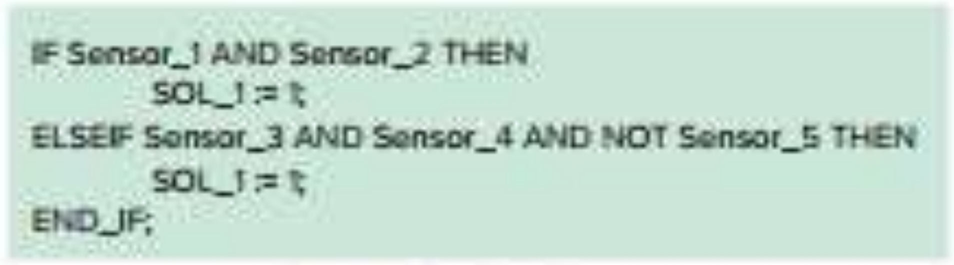
Sequential Functional Chart



Structured Text



Ladder diagram (LD) program



```
IF Sensor_1 AND Sensor_2 THEN
          SOL_1 := 1;
ELSEIF Sensor_3 AND Sensor_4 AND NOT Sensor_5 THEN
          SOL_1 := 1;
END_IF;
```

Structured text (ST) program

Figure 5-19 PLC ladder and equivalent structured text program.

Example

A temperature control system consists of four thermostats controlling three heating units.

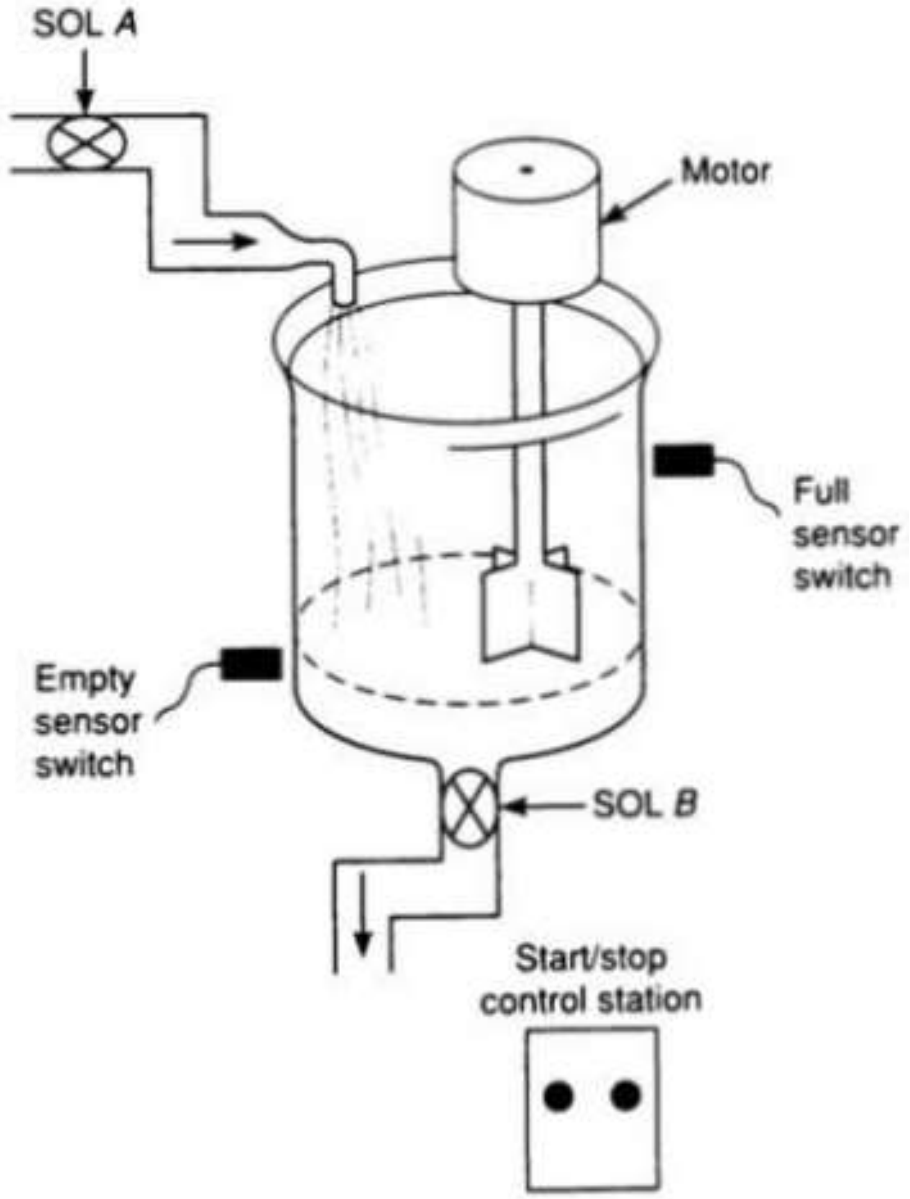
The thermostat contacts are set to close at 50°, 60°, 70°, and 80°F, respectively.

The PLC ladder logic program is to be designed so that at a temperature below 50°F, three heaters are to be ON.

From 50° to 60°F, two heaters are to be ON. For 60° to 70°F, one heater is to be ON.

Above 80°F, there is a safety shutoff for all three heaters in case one stays on because of a malfunction.

A master switch is to be used to turn the system ON and OFF. Prepare a typical PLC program for this control process.


SOL A
Motor
Full sensor switch
Empty sensor switch
SOL B
Start/stop control station

Timers

In general, there are three different PLC timer types: the on-delay timer (**TON**), off-delay timer (**TOF**), and retentive timer on (**RTO**).

These timer commands can be summarized as follows:

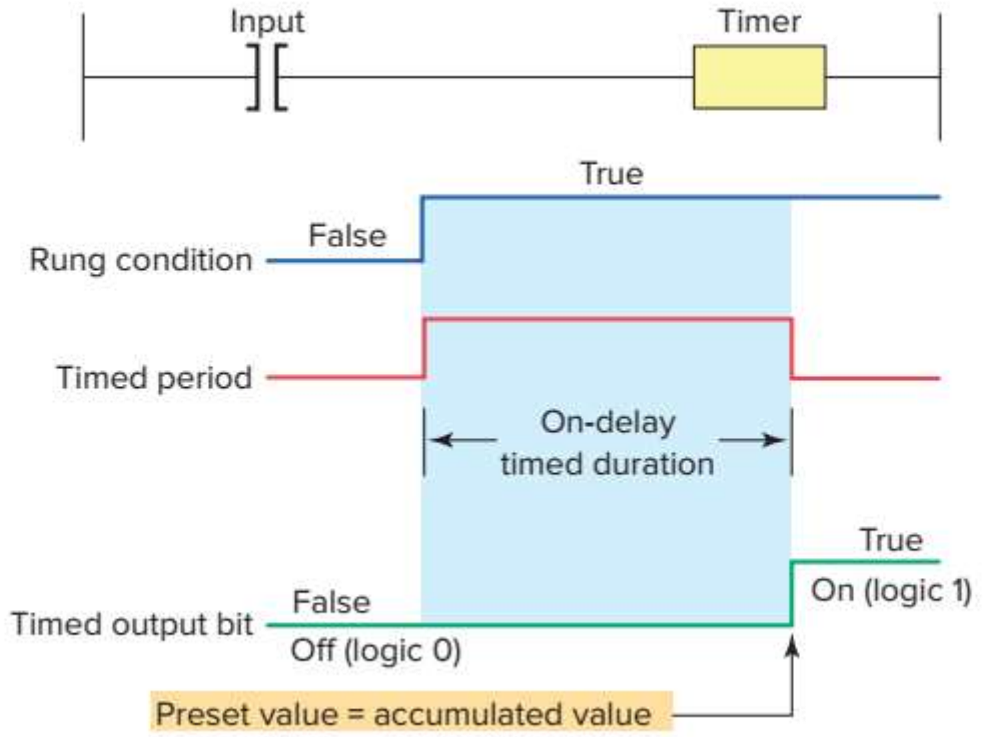
TON (Timer On Delay)—Counts time-based intervals when the instruction is true.

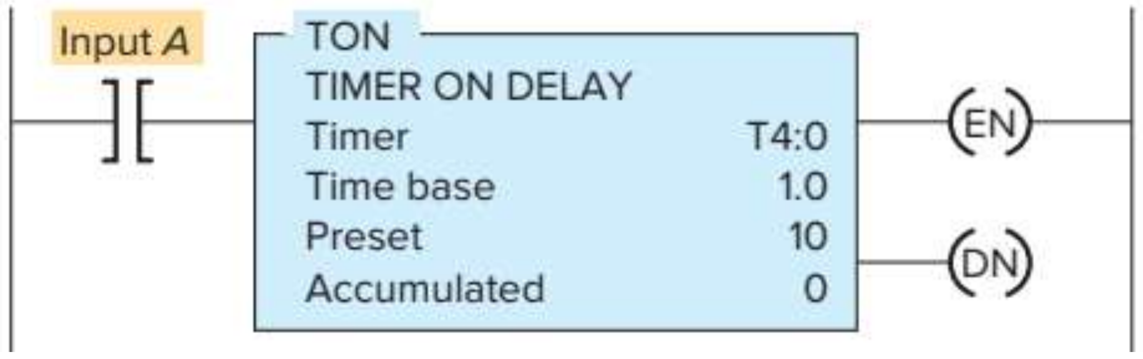
TOF (Timer Off Delay)—Counts time-based intervals when the instruction transitions from a true to false condition.

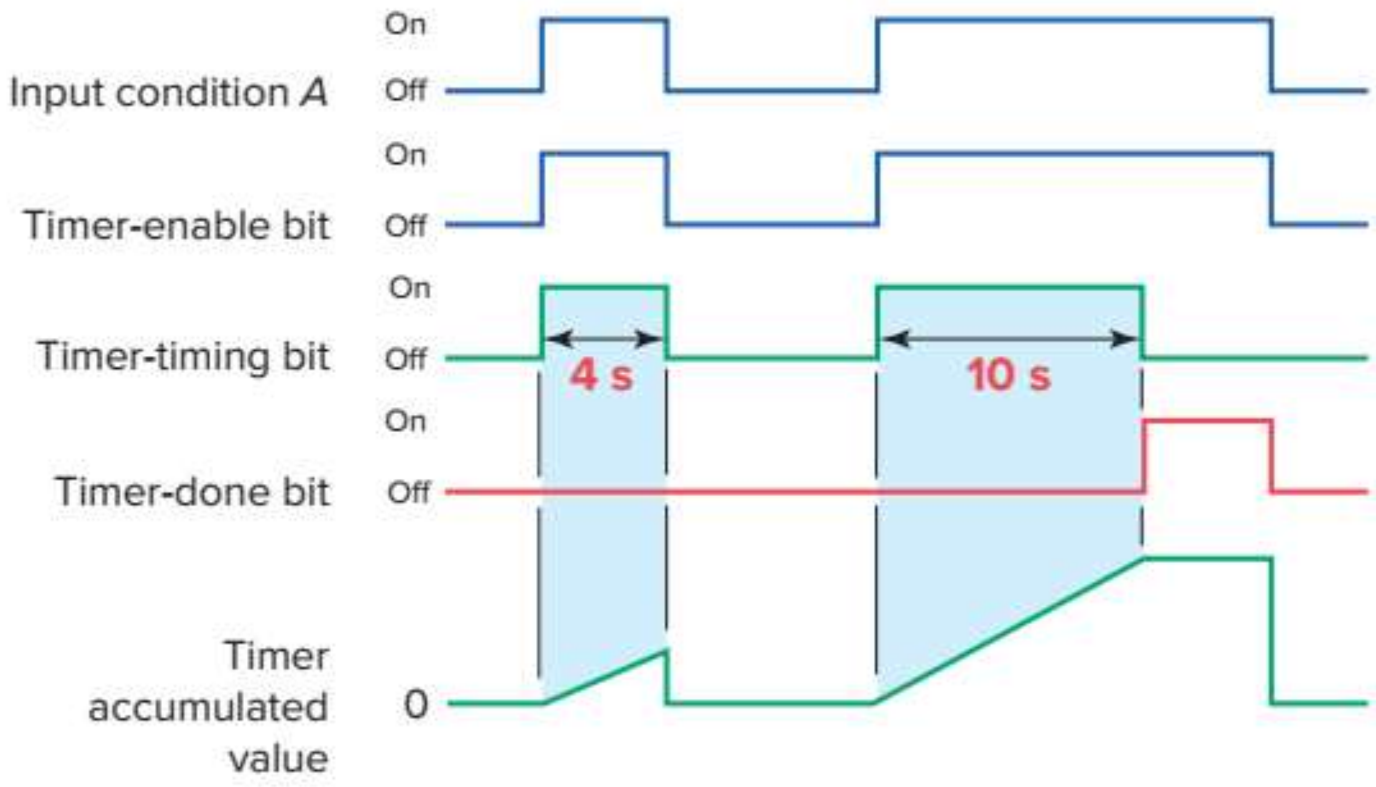
RTO (Retentive Timer On)—Counts time-based intervals when the instruction is true and retains the accumulated value when the instruction goes false or when power cycle occurs.

RES (Reset)—Resets a retentive timer’s accumulated value to zero

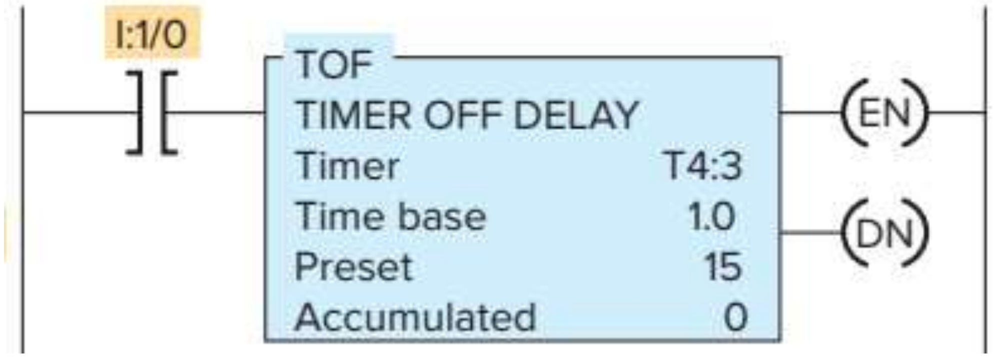
On Delay Timer

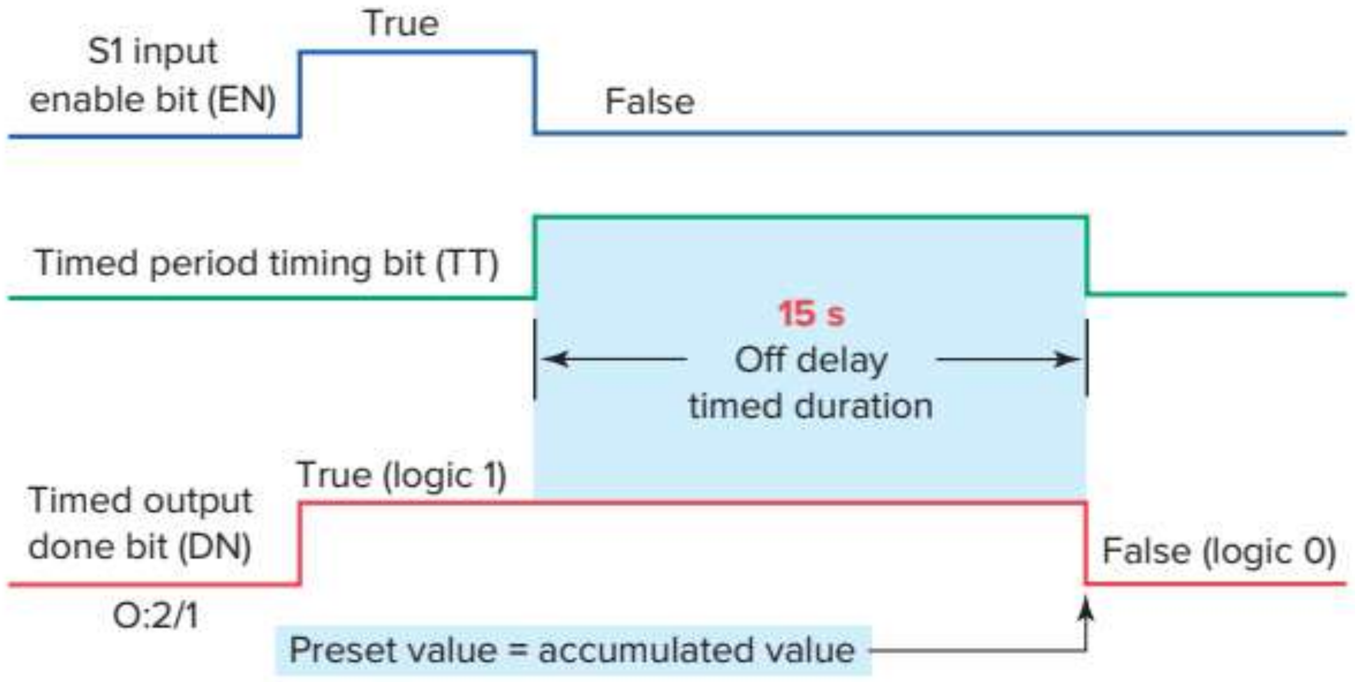



Input A
TON
TIMER ON DELAY
Timer T4:0
Time base 1.0
Preset 10
Accumulated 0
EN
DN

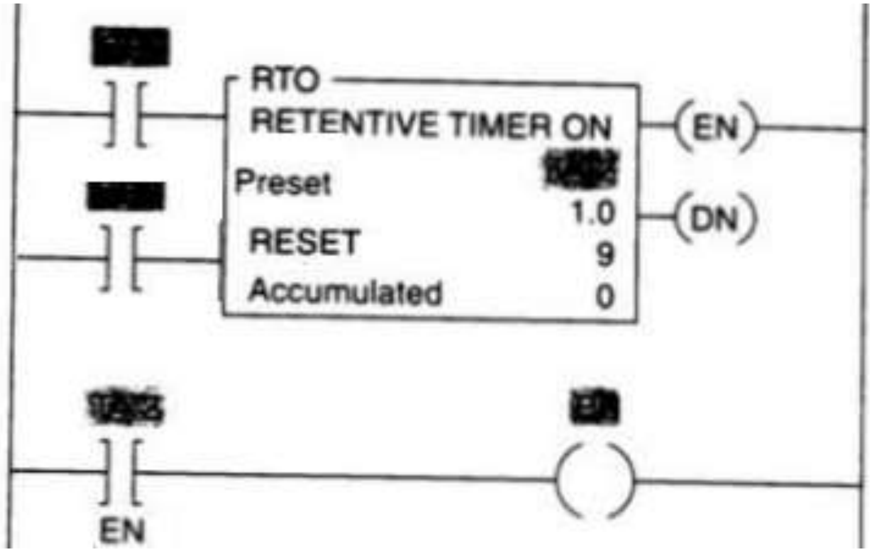

Input condition A
On
Off
Timer-enable bit
On
Off
Timer-timing bit
On
Off
4 s
10 s
Timer-done bit
On
Off
Timer accumulated value
0

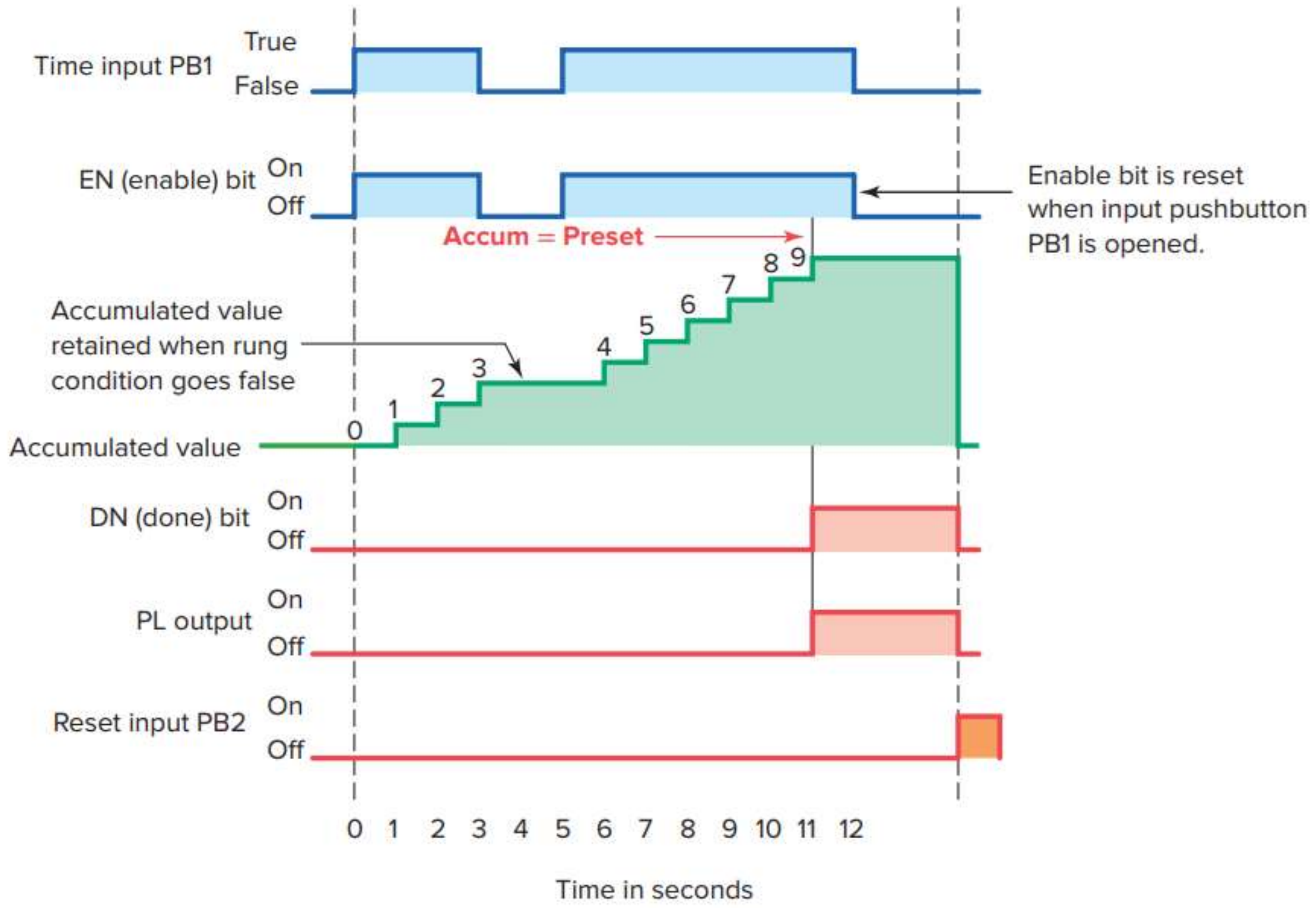
Off- Delay Timer





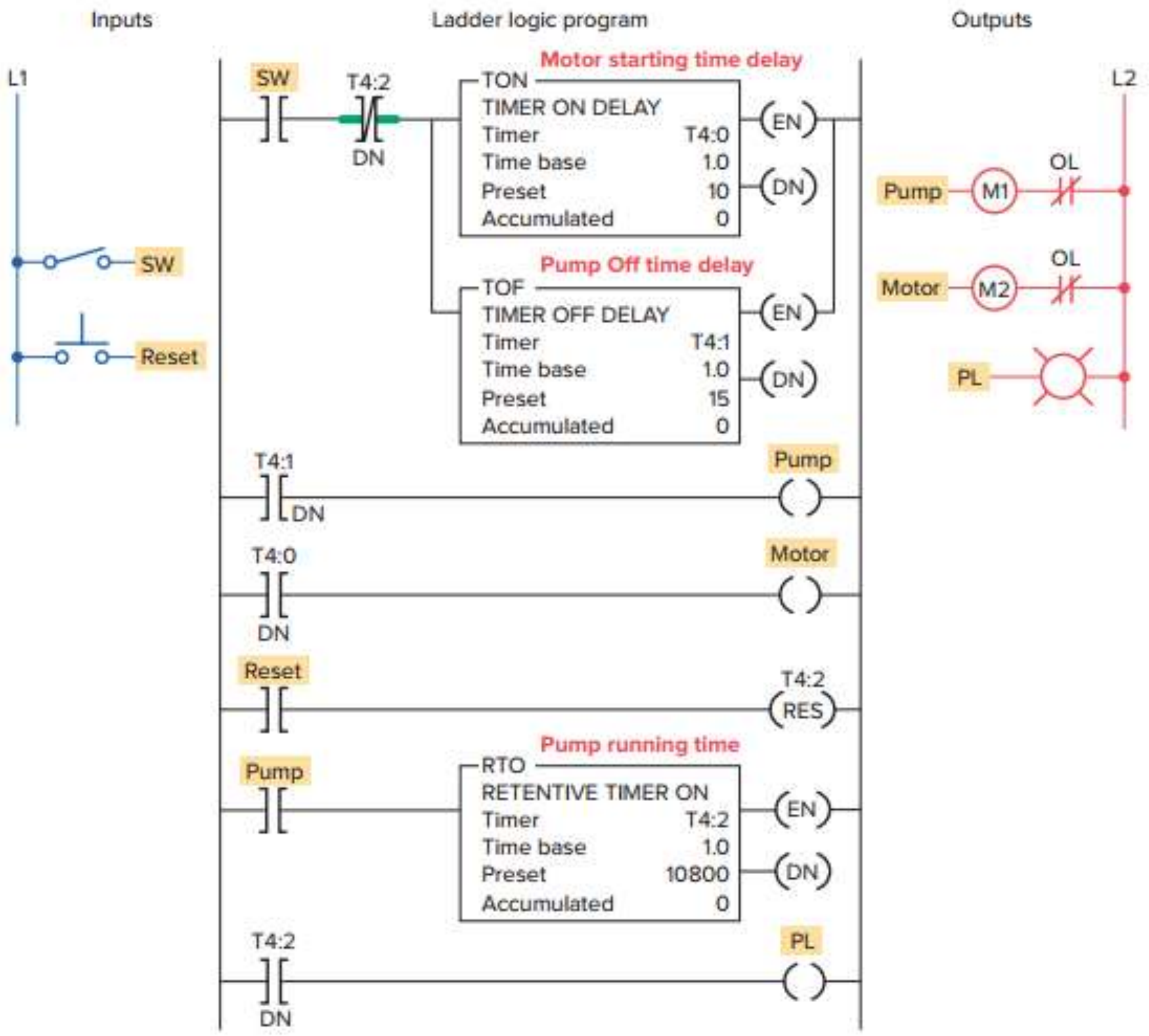
Retentive Timer





Example

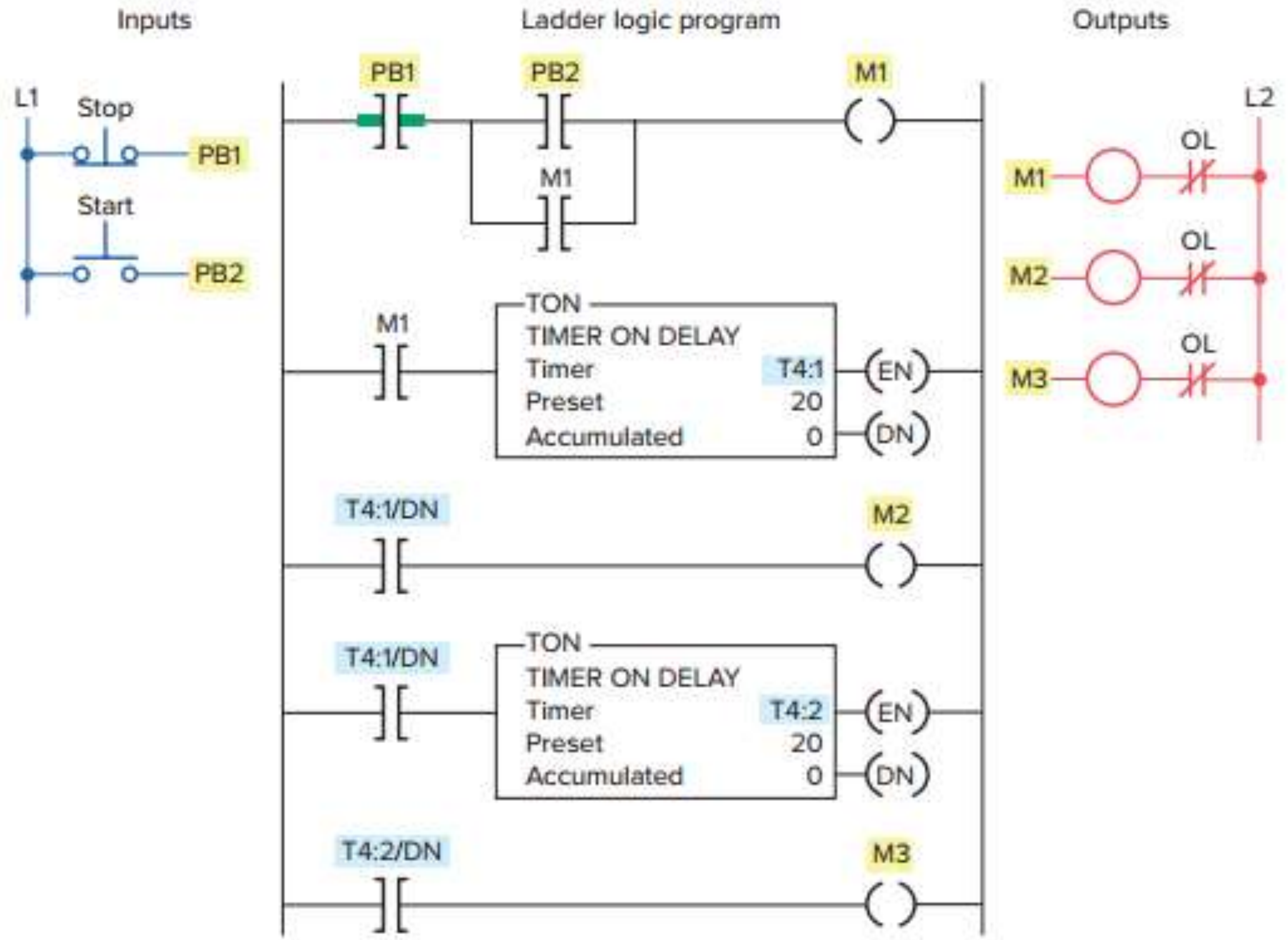
- The operation of the program can be summarized as follows:
- To start the machine, the operator turns SW on.
- Before the *motor* shaft starts to turn, the bearings are supplied with oil by the *pump* for 10 seconds. The bearings also receive oil when the machine is running.
- When the operator turns SW off to stop the machine, the oil pump continues to supply oil for 15 s.
- A retentive timer is used to track the total running time of the pump. When the total running time is 3 hours, the motor is shut down and a pilot light is turned on to indicate that the filter and oil need to be changed.
- A reset button is provided to reset the process after the filter and oil have been changed.


Inputs
Ladder logic program
Outputs
L1
L2
SW
Reset
SW
T4:2
DN
Motor starting time delay
TON
TIMER ON DELAY
Timer T4:0
Time base 1.0
Preset 10
Accumulated 0
EN
DN
Pump Off time delay
TOF
TIMER OFF DELAY
Timer T4:1
Time base 1.0
Preset 15
Accumulated 0
EN
DN
Pump
M1
OL
Motor
M2
OL
PL
T4:1
DN
Pump
T4:0
DN
Motor
Reset
T4:2
RES
Pump running time
Pump
RTO
RETENTIVE TIMER ON
Timer T4:2
Time base 1.0
Preset 10800
Accumulated 0
EN
DN
T4:2
DN
PL

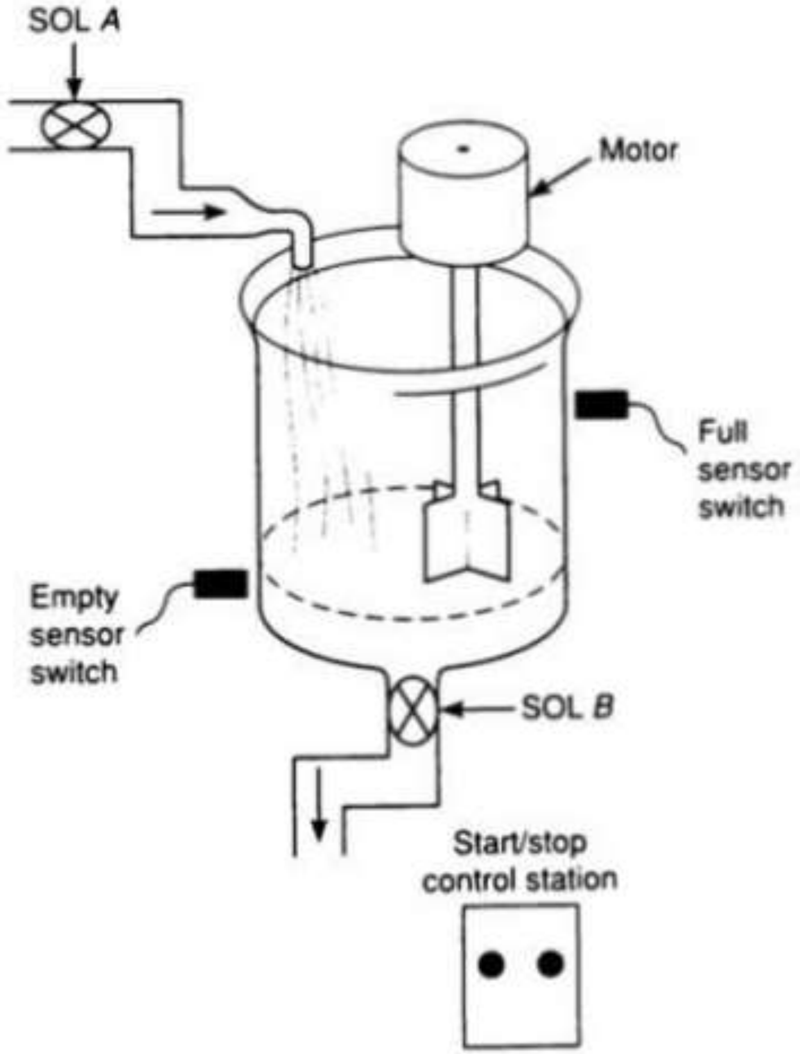
Cascading Timers

The operation of the circuit can be summarized as follows:

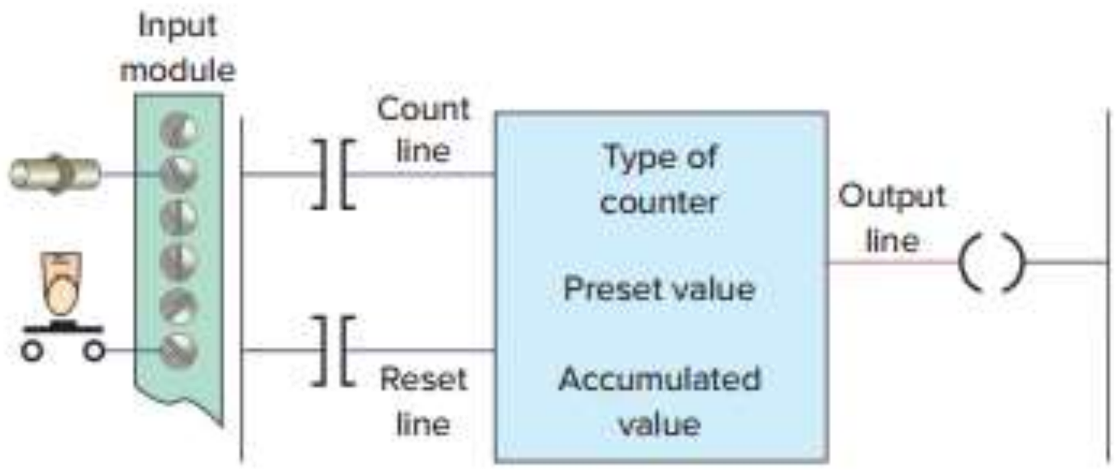
- Motor starter coil M1 is energized when the momentary start pushbutton PB2 is actuated.
- As a result, motor 1 starts, contact M1-1 closes to seal in M1, and timer coil TD1 is energized to begin the first time-delay period.
- After the preset time period of 20 s, TD1-1 contact closes to energize motor starter coil M2.
- As a result, motor 2 starts and timer coil TD2 is energized to begin the second time-delay period.
- After the preset time period of 20 s, TD2-1 contact closes to energize motor starter coil M3, and so motor 3 starts.

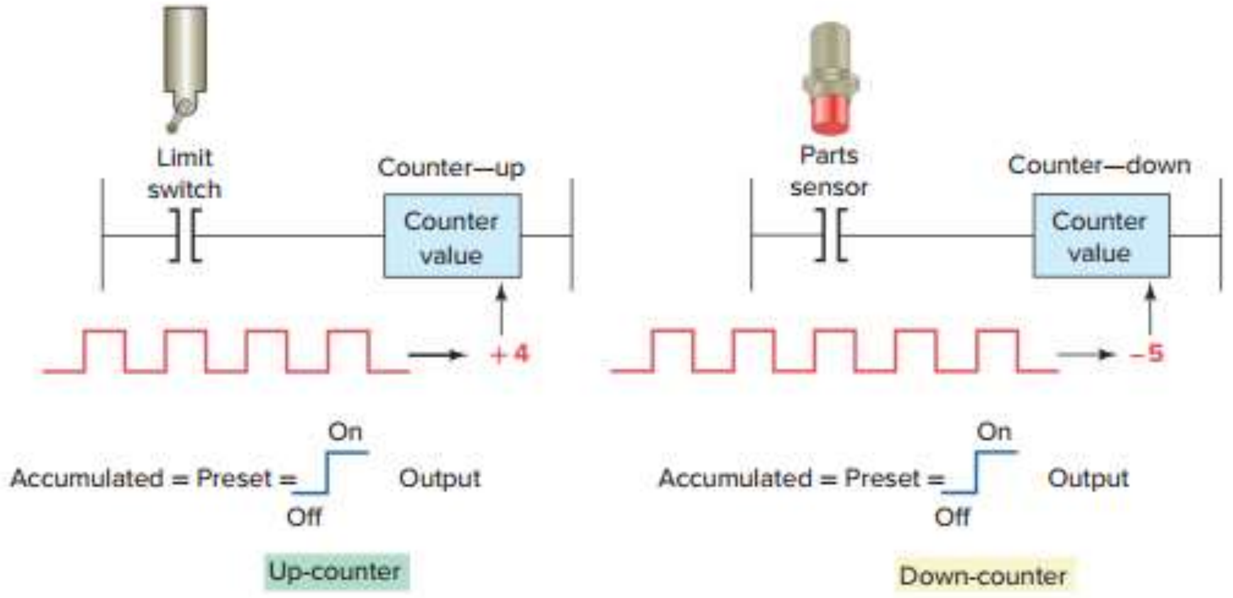


Inputs
Ladder logic program
Outputs
L1
Stop
PB1
Start
PB2
PB1
PB2
M1
M1
L2
OL
M1
OL
M2
OL
M3
M1
TON
TIMER ON DELAY
Timer T4:1
Preset 20
Accumulated 0
EN
DN
T4:1/DN
M2
T4:1/DN
TON
TIMER ON DELAY
Timer T4:2
Preset 20
Accumulated 0
EN
DN
T4:2/DN
M3


SOL A
Motor
Full sensor switch
Empty sensor switch
SOL B
Start/stop control station

Counters


Input module
Count line
Type of counter
Preset value
Accumulated value
Reset line
Output line


Limit switch
Counter—up
Counter value
+4
Accumulated = Preset =
On
Off
Output
Up-counter
Parts sensor
Counter—down
Counter value
−5
Accumulated = Preset =
On
Off
Output
Down-counter

Up-Counter

The up-counter is an output instruction whose function is to increment its accumulated value on false-to-true transitions of its instruction.

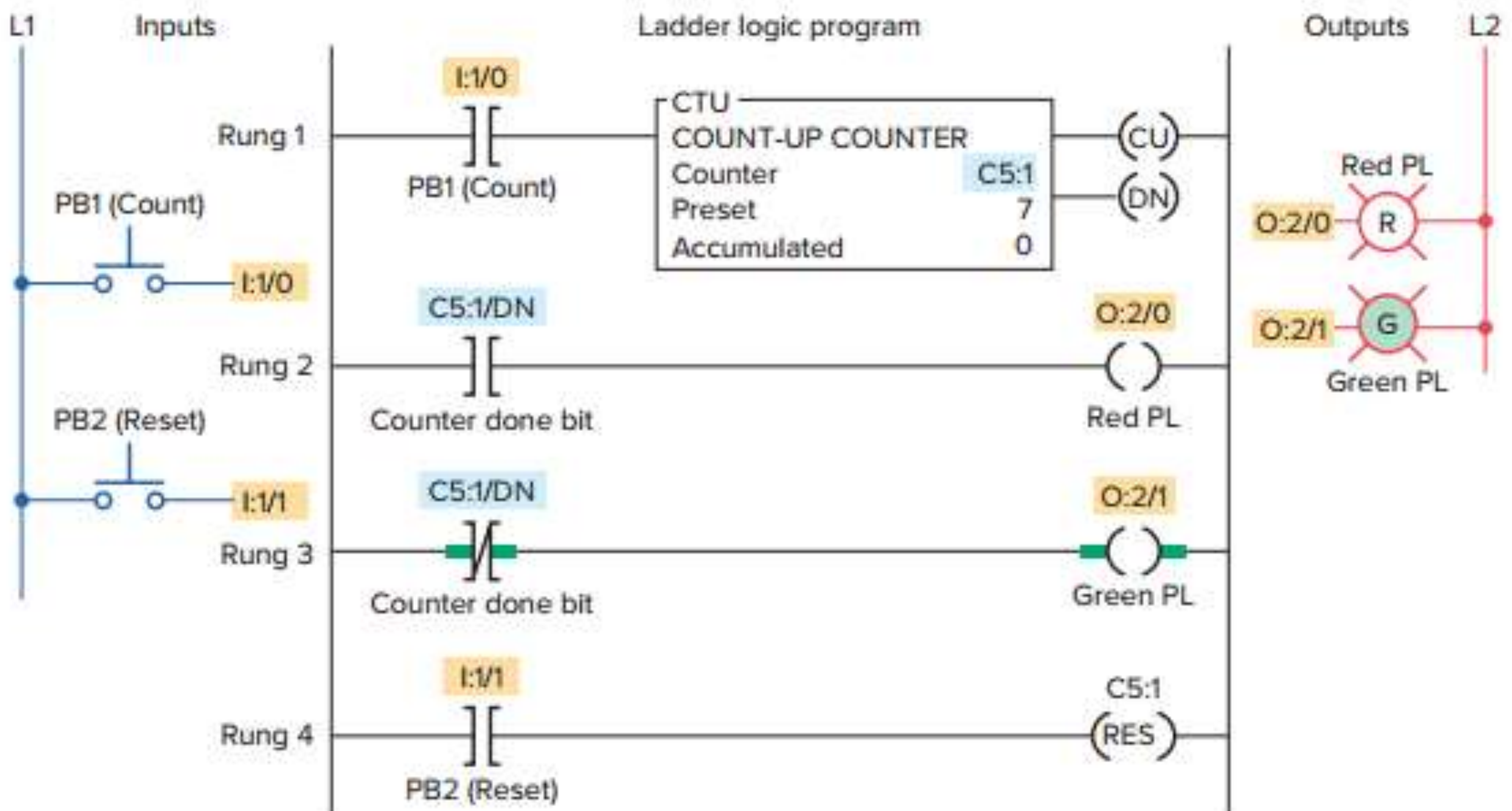
It thus can be used to count false-to-true transitions of an input instruction and then trigger an event after a required number of counts or transitions.

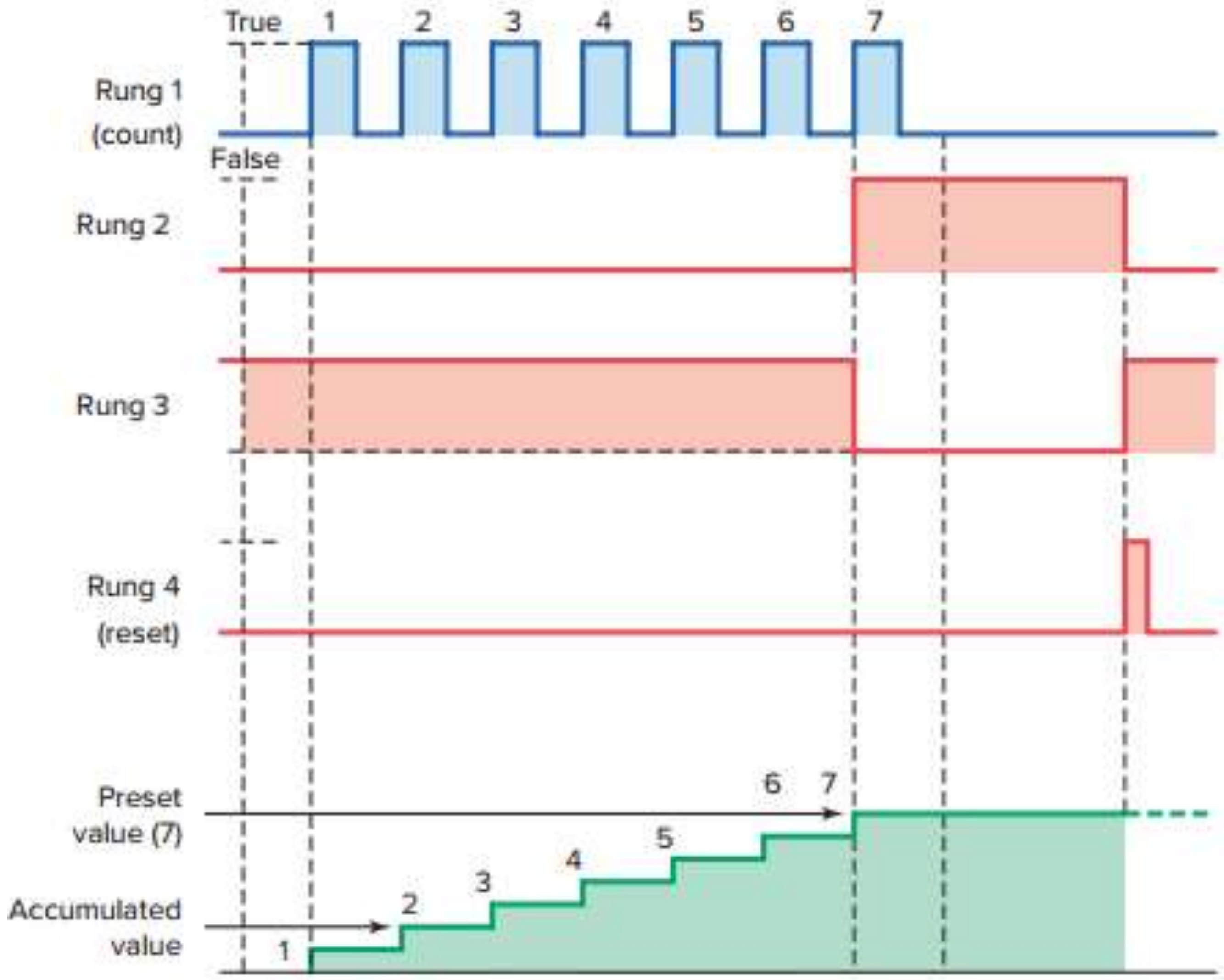
The up-counter output instruction will increment by 1 each time the counted event occurs.

Example

The operation of the program can be summarized as follows:

- Operating pushbutton PB1 provides the off-to-on transition pulses that are counted by the counter.
- The preset value of the counter is set for 7 Each false-to-true transition of rung 1 increases the
counter's accumulated value by 1.
- Output O:2/1 is energized as long as the accumulated value is less than 7.
- After 7 pulses, or counts, when the preset counter value equals the accumulated counter value, output DN is energized.
- As a result, rung 2 becomes true and energizes output O:2/0 to switch the red pilot light on.
- At the same time, rung 3 becomes false and deenergizes output O:2/1 to switch the green pilot light off.
- The counter is reset by closing pushbutton PB2, which makes rung 4 true and resets the accumulated count to zero.
- Counting can resume when rung 4 goes false again.

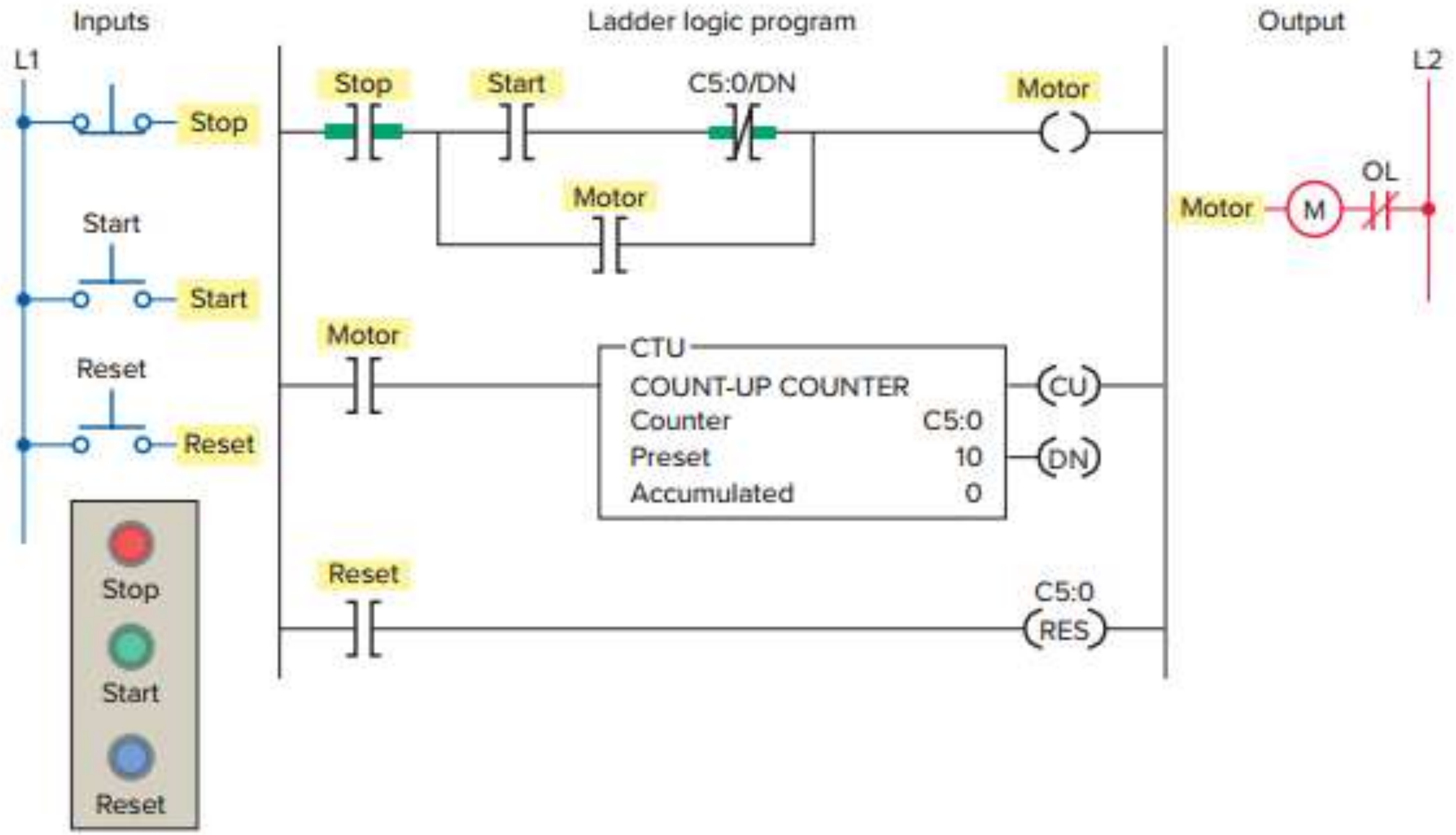

L1
Inputs
Ladder logic program
Outputs
L2
Rung 1
I:1/0
PB1 (Count)
CTU
COUNT-UP COUNTER
Counter C5:1
Preset 7
Accumulated 0
CU
DN
PB1 (Count)
I:1/0
Red PL
O:2/0
R
O:2/1
G
Green PL
C5:1/DN
Rung 2
Counter done bit
O:2/0
Red PL
PB2 (Reset)
I:1/1
C5:1/DN
Rung 3
Counter done bit
O:2/1
Green PL
I:1/1
Rung 4
PB2 (Reset)
C5:1
RES


True
1
2
3
4
5
6
7
Rung 1
(count)
False
Rung 2
Rung 3
Rung 4
(reset)
Preset
value (7)
Accumulated
value
1
2
3
4
5
6
7

Example

A PLC counter program used to stop a motor from running after 10 operations. The operation of the program can be summarized as follows:

- Up-counter C5:0 counts the number of on/off times the motor starts.
- The preset value of the counter is set to 10
- A counter done bit examine-off instruction is programmed in series with the motor output instruction.
- A motor output examine-on instruction is used to increment the accumulated value of the counter for each off/on operation.
- After the count of 10 is reached the counter done bit examine-off instruction goes false preventing the motor from being started.
- Closure of the reset pushbutton resets the accumulated count to zero.

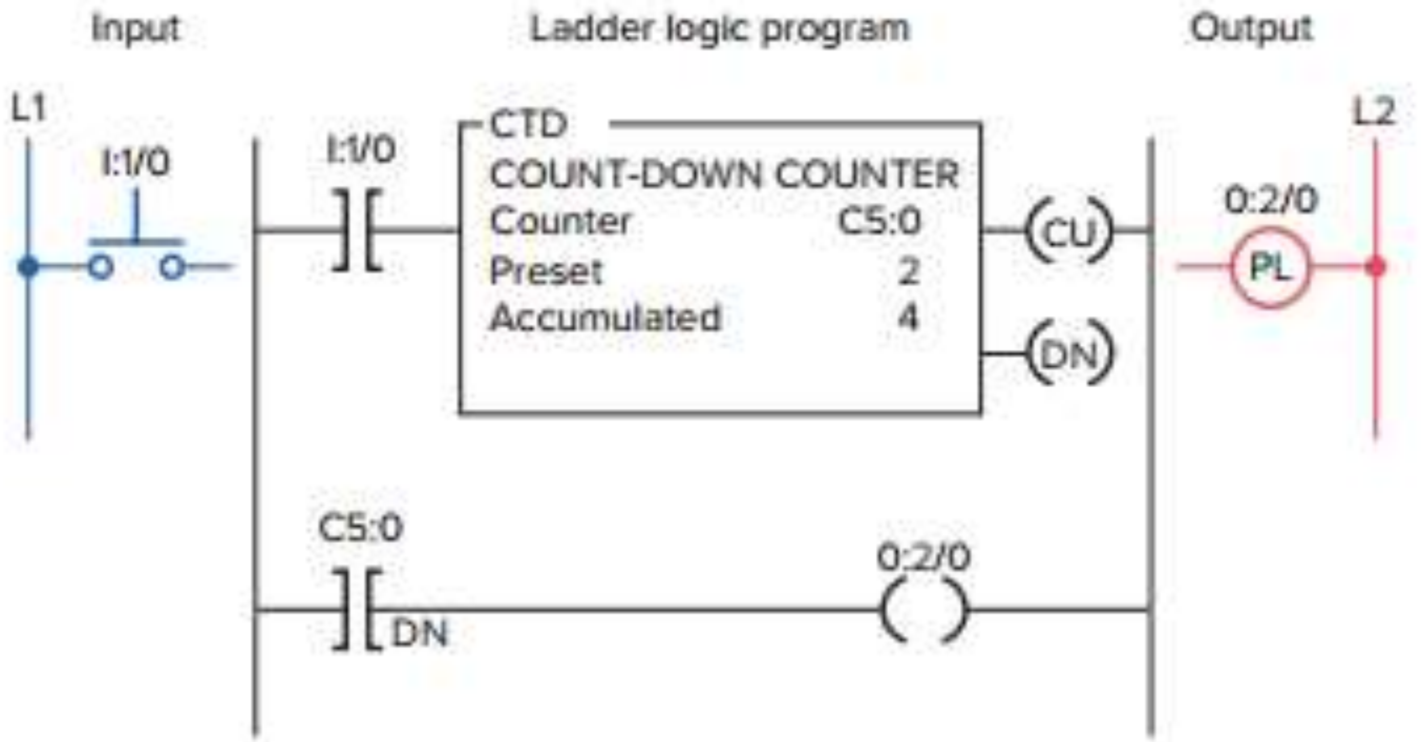

Inputs
Ladder logic program
Output
L1
L2
Stop
Start
Reset
Stop
Start
C5:0/DN
Motor
Motor
Motor
OL
M
Motor
CTU
COUNT-UP COUNTER
Counter C5:0
Preset 10
Accumulated 0
CU
DN
Reset
C5:0
RES
Stop
Start
Reset

Down-Counter

The down-counter instruction will count down or decrement by 1 each time the counted event occurs. Each time the down-count event occurs, the accumulated value is decremented. Normally the down-counter is used in conjunction with the up-counter to form an up/ down-counter.

The program of contains a count-down counter instruction, the operation of which can be summarized as follows:

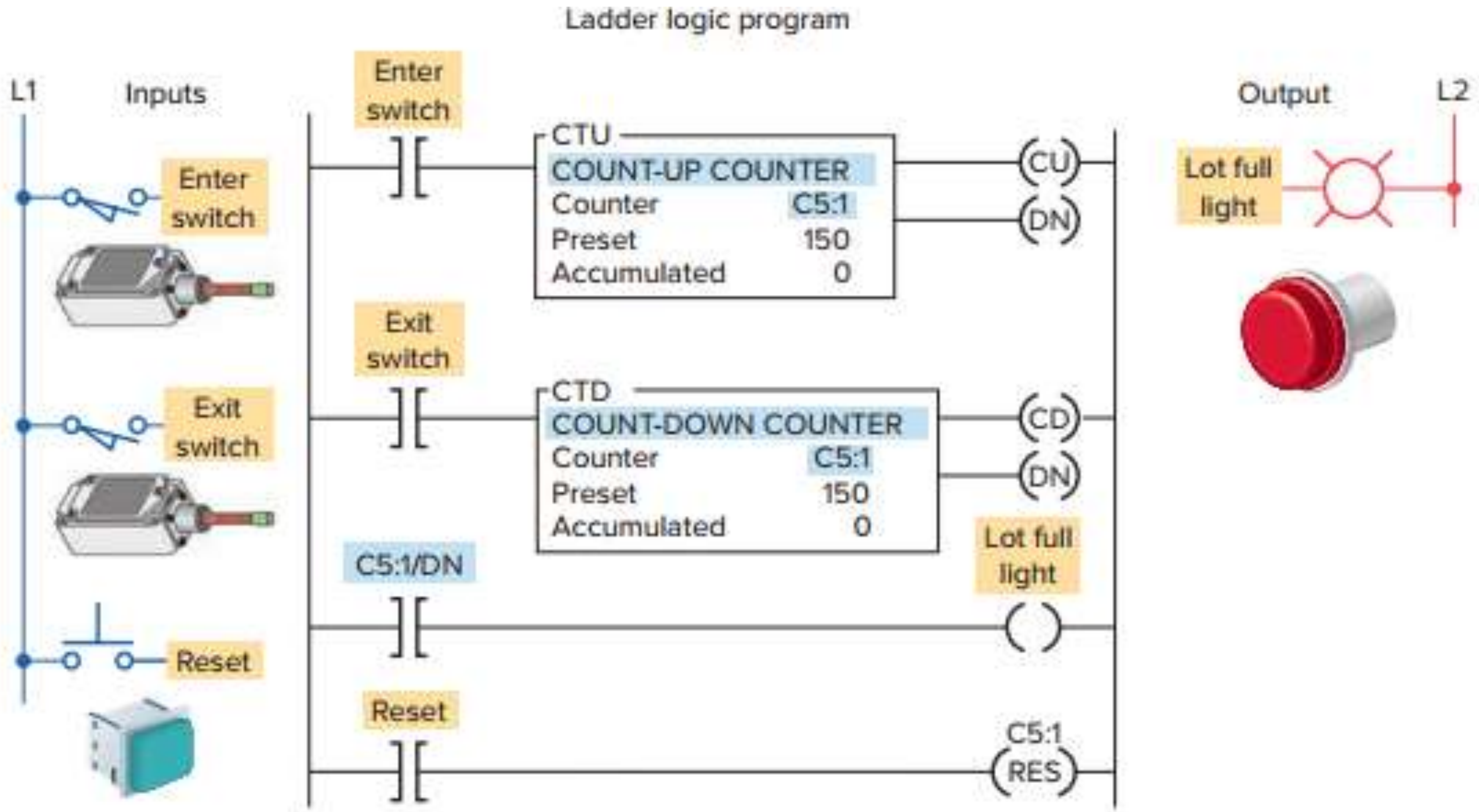
- With the program in the state shown, the CTD done bit will be set (1) and output 0:2/0 will be energized because the accumulated value of 4 is greater than the preset value of 2.
- When the CTD instruction rung makes a false-to-true transition, the accumulated value decreases by one count to 3.
- When the input rung condition makes another false-to-true transition, the accumulated value will decrease to 2.
- When the input makes one more false-to true transition, the accumulated value will drop to 1.
- At this point the accumulated value of 1 is less than the preset value of 2 so the done bit will be reset (0) de-energizing output O:2/0


Input
Ladder logic program
Output
L1
I:1/0
I:1/0
CTD
COUNT-DOWN COUNTER
Counter C5:0
Preset 2
Accumulated 4
CU
DN
L2
0:2/0
PL
C5:0
DN
0:2/0

Example

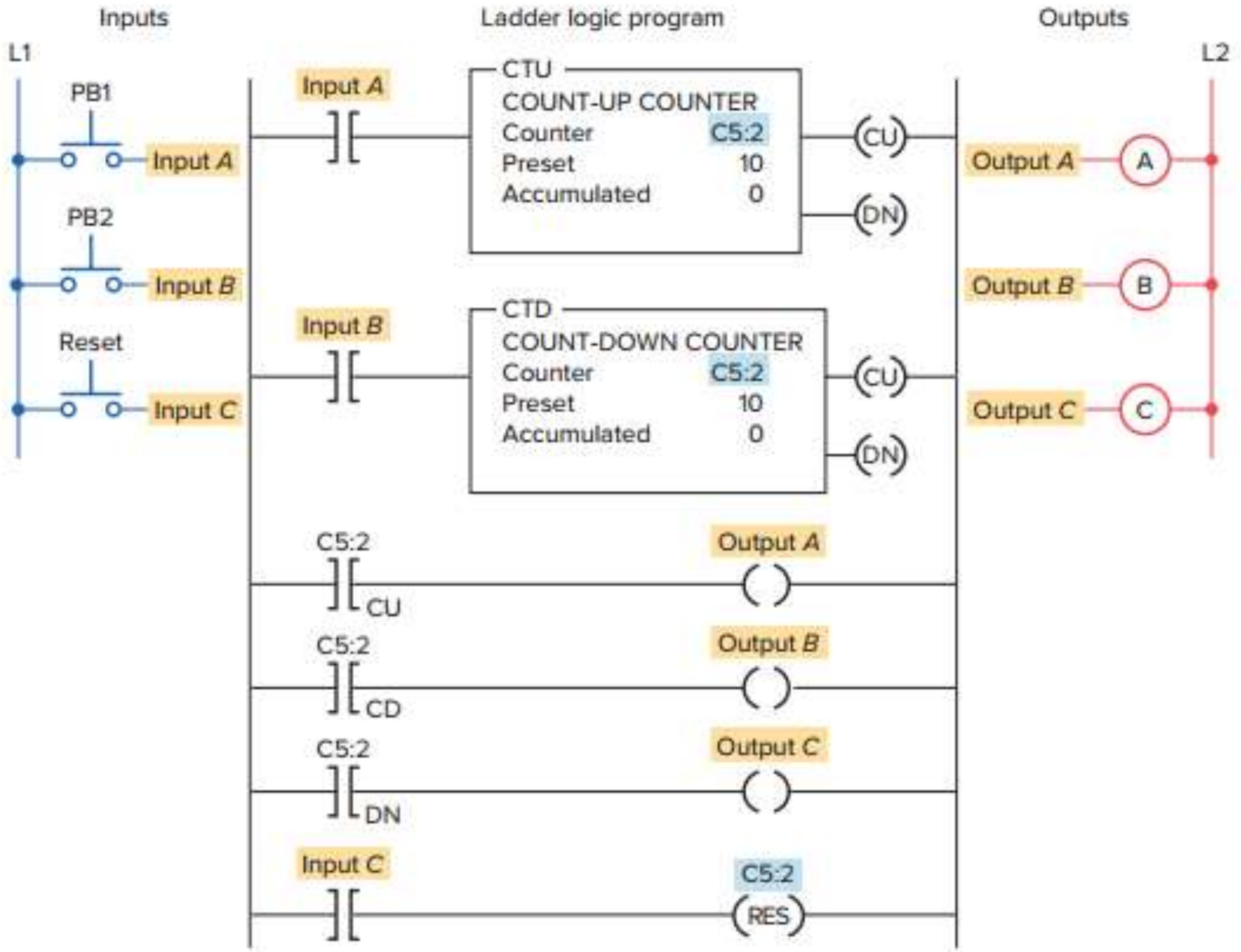
The operation of the program can be summarized as follows:

- As a car enters, the enter switch triggers the upcounter output instruction and increments the accumulated count by 1.
- As a car leaves, the exit switch triggers the downcounter output instruction and decrements the accumulated count by 1.
- Because both the up- and down-counters have the same address, C5:1, the accumulated value will be the same in both instructions as well as the preset.
- Whenever the accumulated value of 150 equals the preset value of 150, the counter output is energized by the done bit to light up the Lot Full sign.
- A reset button has been provided to reset the accumulated count.


Ladder logic program
L1
Inputs
Enter switch
Exit switch
Reset
Enter switch
CTU
COUNT-UP COUNTER
Counter C5:1
Preset 150
Accumulated 0
CU
DN
Exit switch
CTD
COUNT-DOWN COUNTER
Counter C5:1
Preset 150
Accumulated 0
CD
DN
C5:1/DN
Lot full light
Reset
C5:1
RES
Output
L2
Lot full light

Up – Down Counter

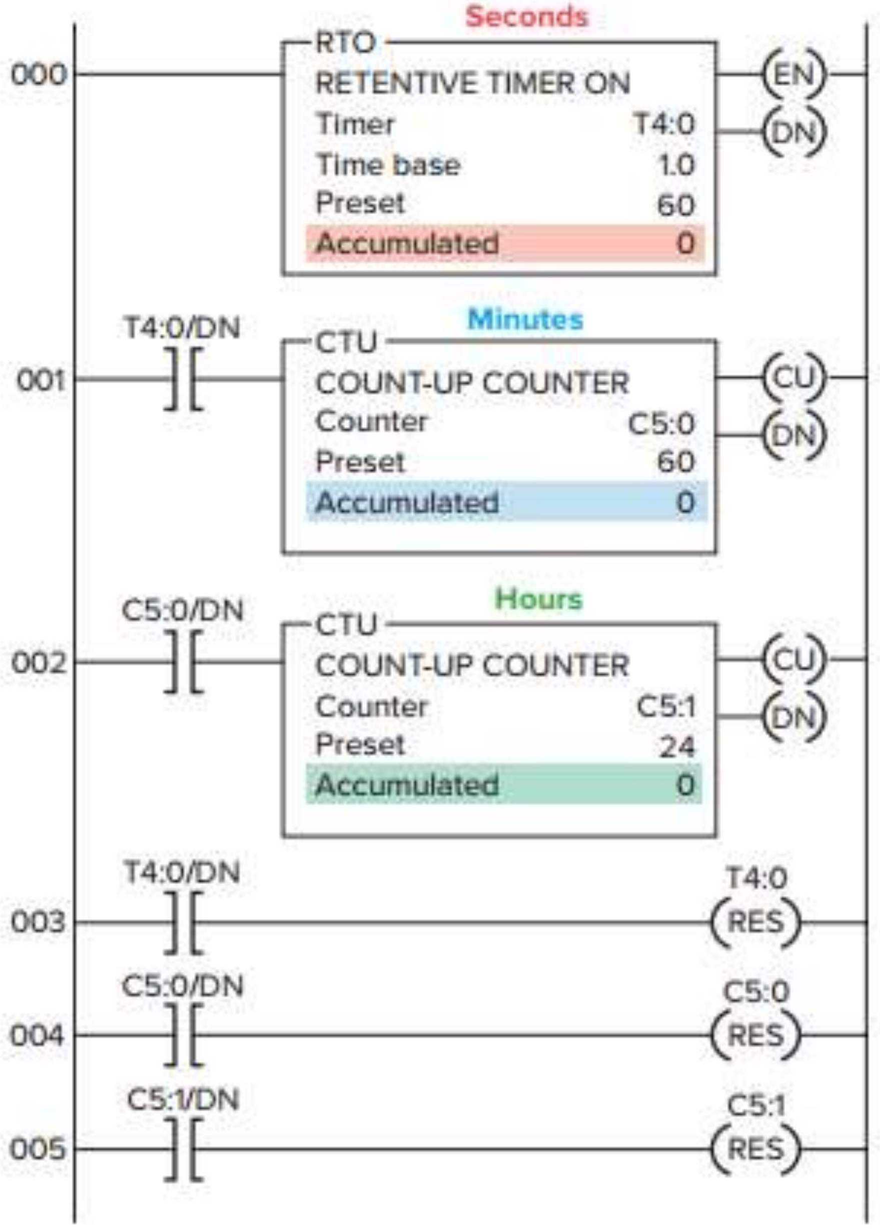
- The operation of the program can be summarized as follows:
- When the CTU instruction is true, C5:2/CU will be true, causing output *A* to be true.
- When the CTD instruction is true, C5:2/CD will be true, causing output *B* to be true.
- When the accumulated value is greater than or equal to the preset value, C5:2/DN will be true, causing output *C* to be true.
- Input *C* going true will cause both counter instructions to reset. When *reset* by the **RES instruction,** the accumulated value will be reset to 0 and the done bit will be reset.


Inputs
Ladder logic program
Outputs
L1
L2
PB1
Input A
PB2
Input B
Reset
Input C
Input A
CTU
COUNT-UP COUNTER
Counter C5:2
Preset 10
Accumulated 0
CU
DN
Output A
A
Output B
B
Output C
C
Input B
CTD
COUNT-DOWN COUNTER
Counter C5:2
Preset 10
Accumulated 0
CU
DN
C5:2
CU
Output A
C5:2
CD
Output B
C5:2
DN
Output C
Input C
C5:2
RES

Assignment:

24-hour clock program.

Ladder logic program



High-Speed Counters

The maximum counting frequency of a traditional PLC’s counter is limited by the scan time of the processor. When the frequency of the input signal is higher than that of the scan time, it is necessary to utilize **a high-speed counter (HSC),** to avoid errors.

Data manipulation instructions allow numerical data stored in the controller's memory to be operated on within the control program. It includes operations involving moving or transferring numeric information stored in one memory word location to another word in a different location, and carrying out simple operations such as converting from one data format to another.

The commands can be summarized as follows:

MOV (Move)—Moves the source value to the destination.

MVM (Masked Move)—Moves data from a source location to a selected portion of the destination.

AND (And)—Performs a bitwise AND operation.

OR (Or)—Performs a bitwise OR operation.

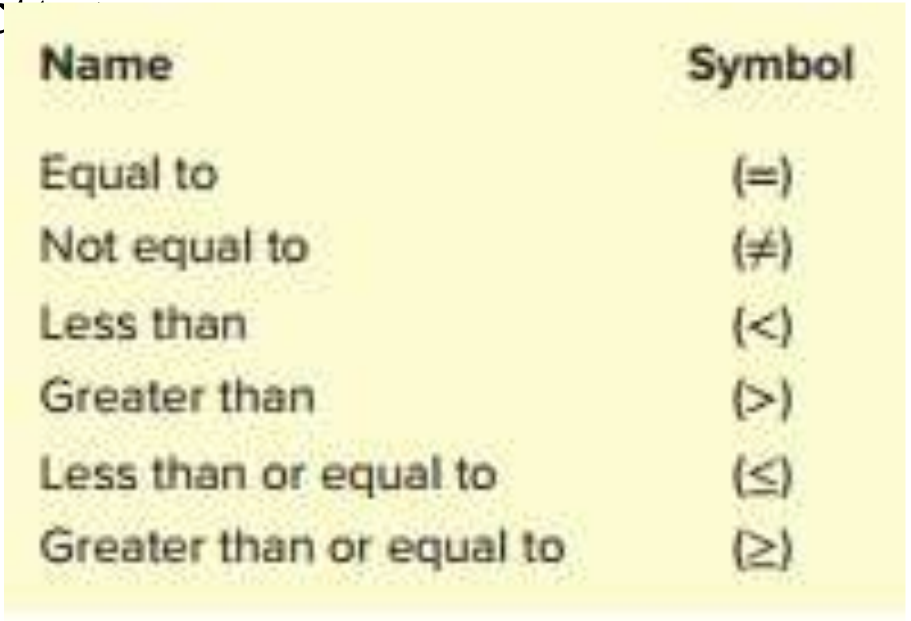
XOR (Exclusive Or)—Performs a bitwise XOR operation.

NOT (Not)—Performs a bitwise NOT operation.

CLR (Clear)—Sets all bits of a word to zero.

Data Compare Instructions

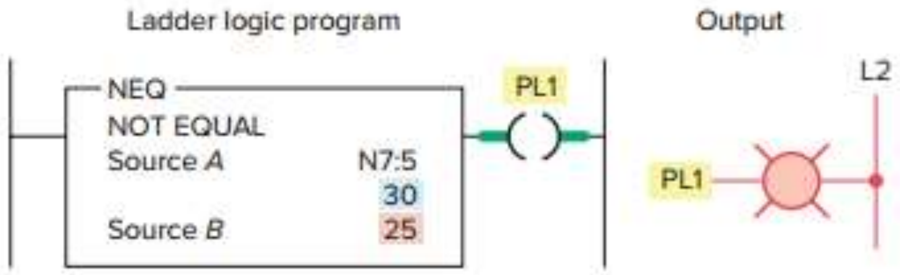
Data transfer operations are all **output** instructions, whereas ***data compare*** instructions are ***input*** instructions. Data compare instructions are used to compare numerical values. These instructions compare the data stored in two or more words (or registers) and make decisions based on the program instruc

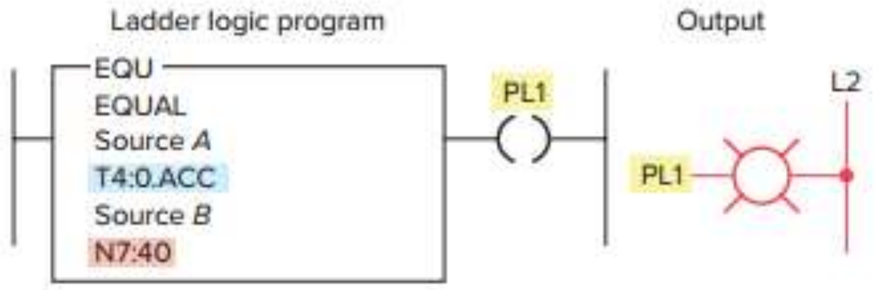


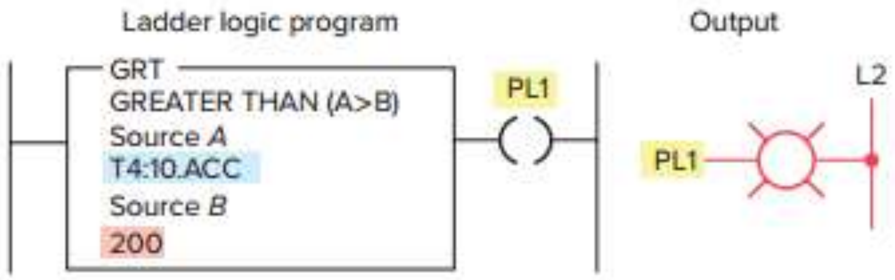
Name	Symbol
Equal to	(=)
Not equal to	(≠)
Less than	(<)
Greater than	(>)
Less than or equal to	(≤)
Greater than or equal to	(≥)

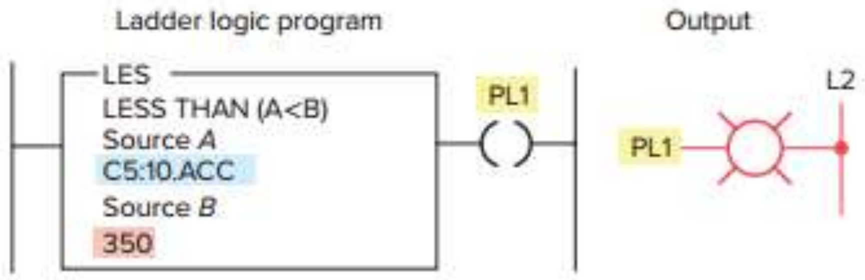
The compare instructions can be summarized as follows:

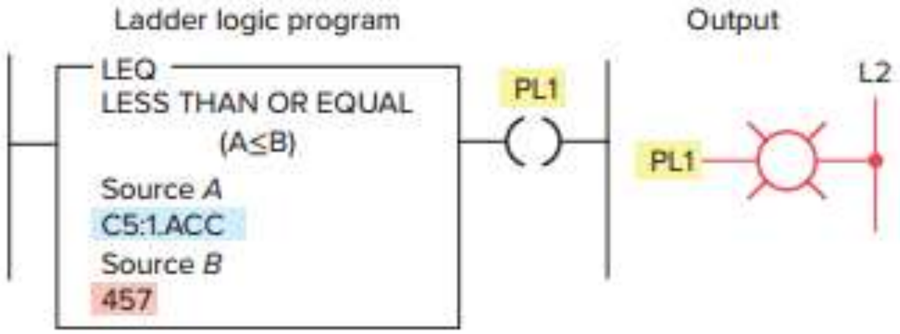
- **LIM (Limit test)**—Tests whether one value is within the limit range of two other values.
- **MEQ (Masked Comparison for Equal)**—Tests portions of two values to see whether they are equal. Compares 16-bit data of a source address to 16-bit data at a reference address through a mask.
- **EQU (Equal)**—Tests whether the value of Source A is equal to the value of Source B.
- **NEQ (Not Equal)**—Tests whether the value of Source A is not equal to the value of Source B.
- **LES (Less Than)**—Tests whether the value of Source A is less than the value of Source B.
- **GRT (Greater Than)**—Tests whether the value of Source A is greater than the value of Source B.
- **LEQ (Less Than or Equal)**—Tests whether the value of Source A is less than or equal to the value of Source B.
- **GEQ (Greater Than or Equal)**—Tests whether the value of Source A is greater than or equal to the value of Source B

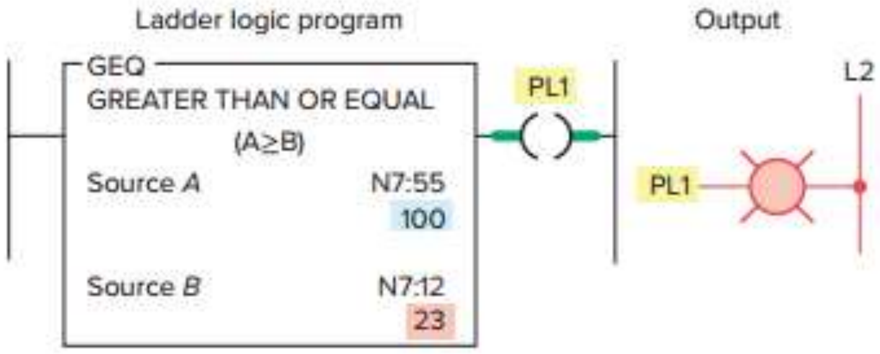

Ladder logic program
Output
NEQ
NOT EQUAL
Source A
N7:5
30
Source B
25
PL1
PL1
L2

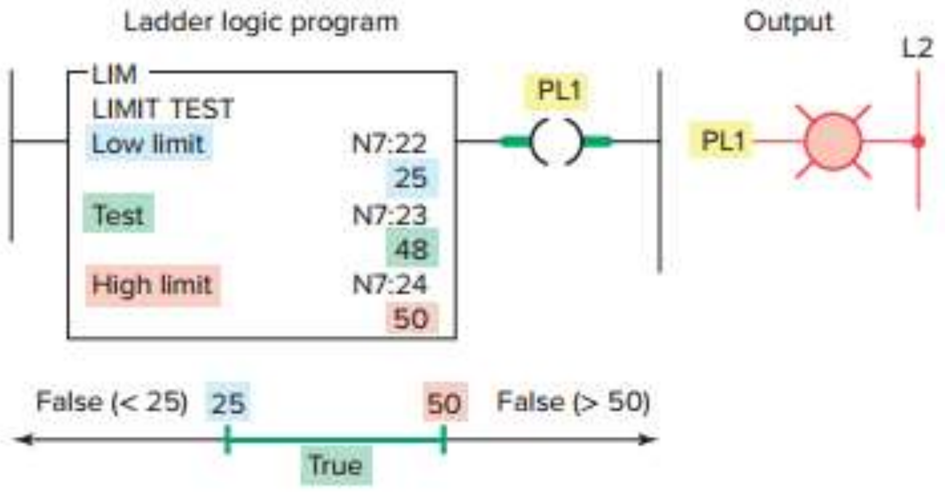

Ladder logic program
Output
EQU
EQUAL
Source A
T4:0.ACC
Source B
N7:40
PL1
PL1
L2

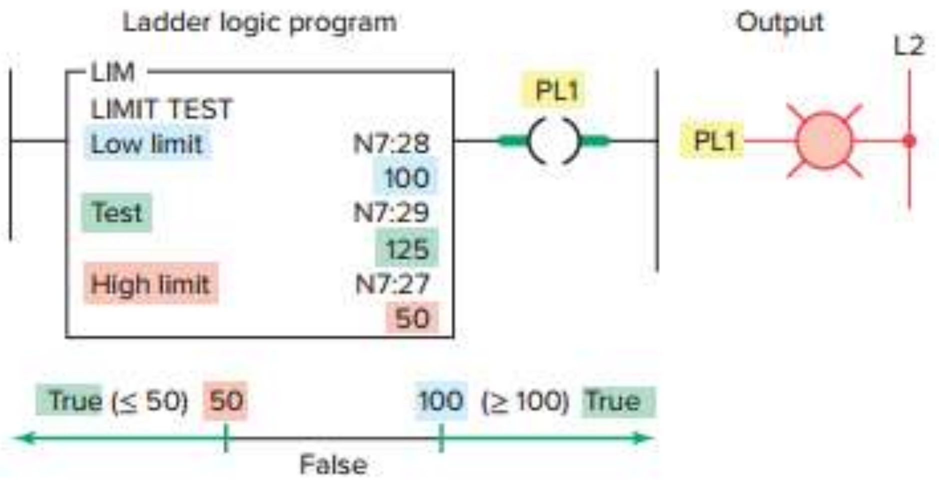

Ladder logic program
Output
GRT
GREATER THAN (A>B)
Source A
T4:10.ACC
Source B
200
PL1
PL1
L2


Ladder logic program
Output
LES
LESS THAN (A<B)
Source A
C5:10.ACC
Source B
350
PL1
PL1
L2


Ladder logic program
Output
LEQ
LESS THAN OR EQUAL
(A≤B)
Source A
C5:1.ACC
Source B
457
PL1
PL1
L2


Ladder logic program
Output
GEQ
GREATER THAN OR EQUAL
(A≥B)
Source A
N7:55
100
Source B
N7:12
23
PL1
PL1
L2


Ladder logic program
Output
L2
LIM
LIMIT TEST
Low limit N7:22 25
Test N7:23 48
High limit N7:24 50
PL1
PL1
False (< 25) 25
50 False (> 50)
True


Ladder logic program
Output
L2
LIM
LIMIT TEST
Low limit N7:28 100
Test N7:29 125
High limit N7:27 50
PL1
PL1
True (≤ 50) 50
100 (≥ 100) True
False

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.

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.

- The basic math instructions are ADD, SUB, MUL, and DIV. Each of these instructions has three parameter fields. Namely, Source A, Source B and Destination fields.
- The **Source A** and **Source B** fields can be an input rack location, file address, instruction field, or a fixed value.

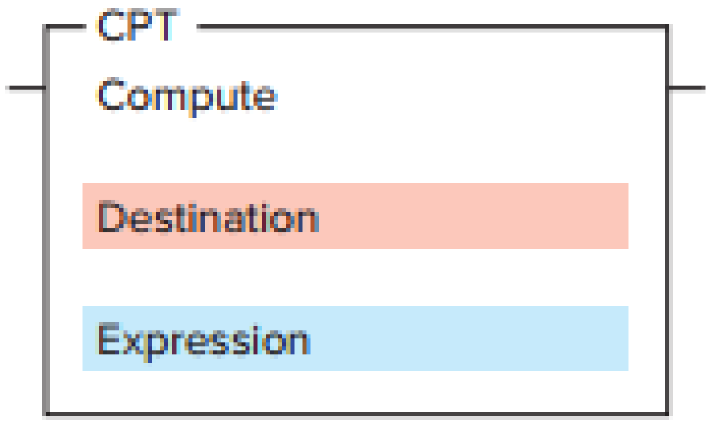
For example:
Input Location I:1
File Address N7:5
Instruction Field C5:2.ACC
Fixed Value 30

- The **Destination** fields can be an output location, fle address, or an instruction field.

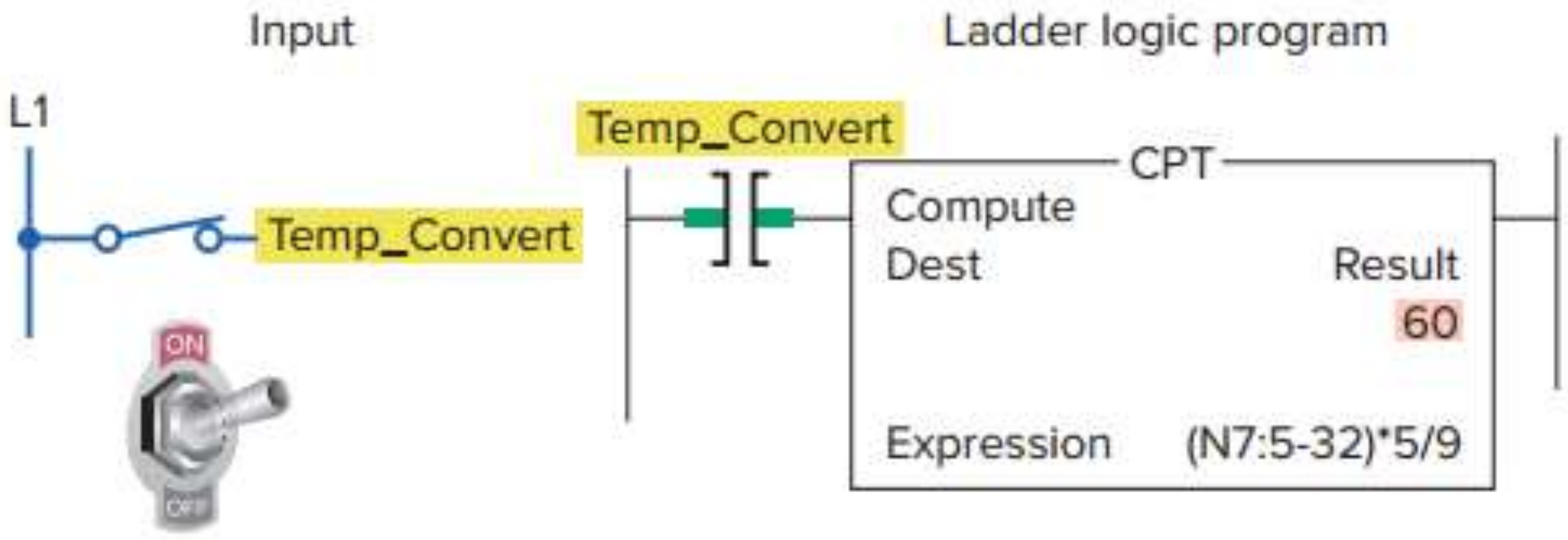
For example:
Output location O:2
File Address N7:8
Instruction Field T4:1.PRE

CPT (compute) instruction

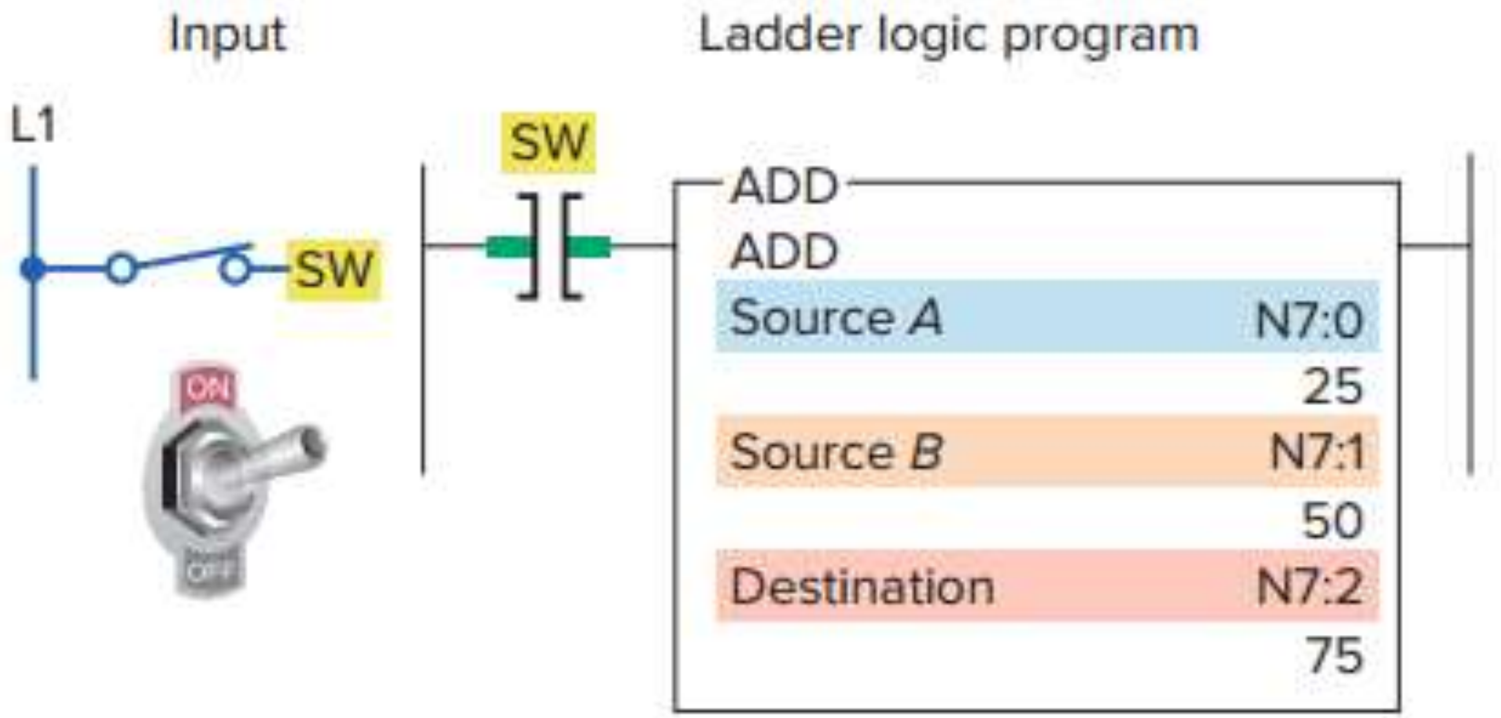
- When CPT instruction is executed, then copy, arithmetic, logical, or conversion operation residing in the expression field of this instruction is performed and the result is sent to the destination.
- The execution time of a CPT instruction is longer than that of a single arithmetic operation and uses more instruction words.
- The main advantage of the compute instruction is that it allows you to enter quite complex expressions in one instruction



Compute instruction used to convert from Fahrenheit to Celsius



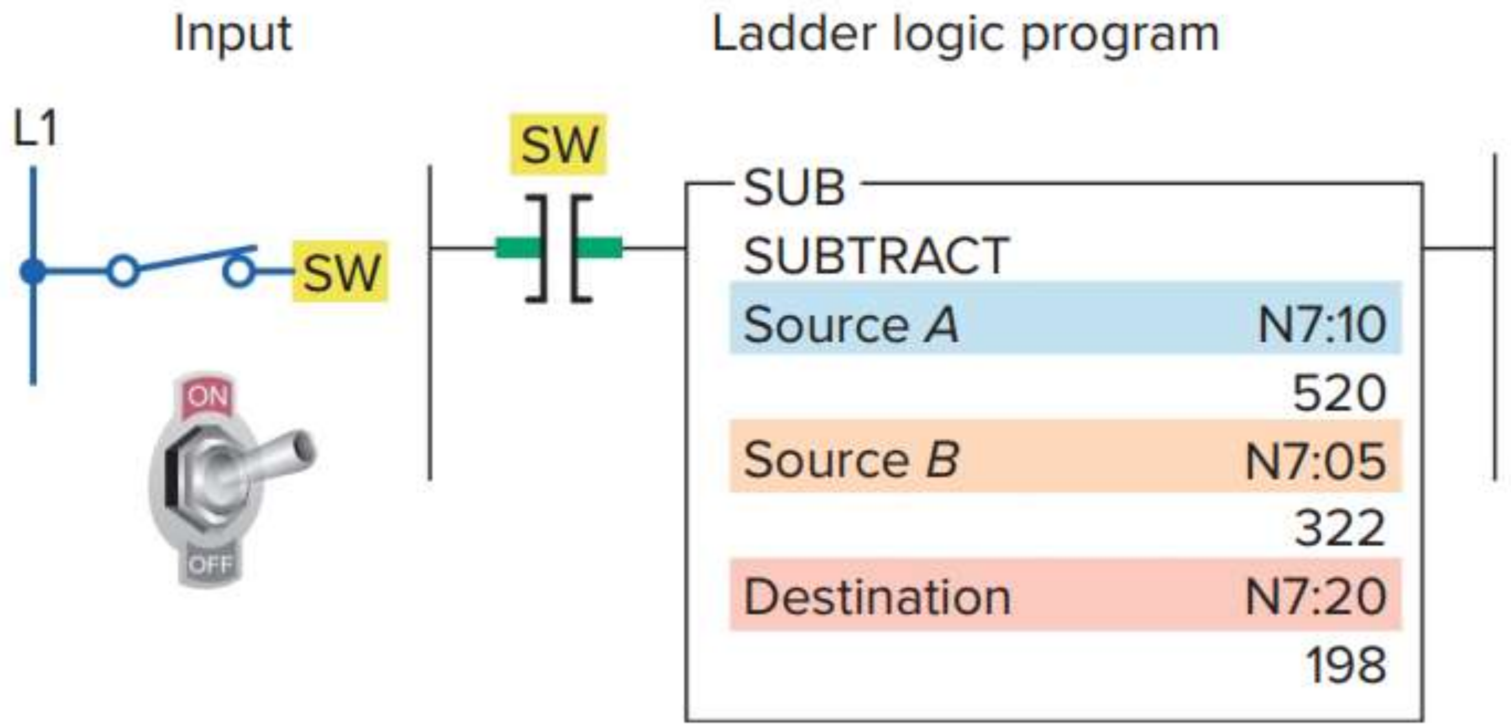
Addition Instruction



Example :

To ***ADD*** the accumulated counts of two up-counters. This application requires a pilot light to come on when the sum of the counts from the two counters is equal to or greater than 350.

Subtraction Instruction

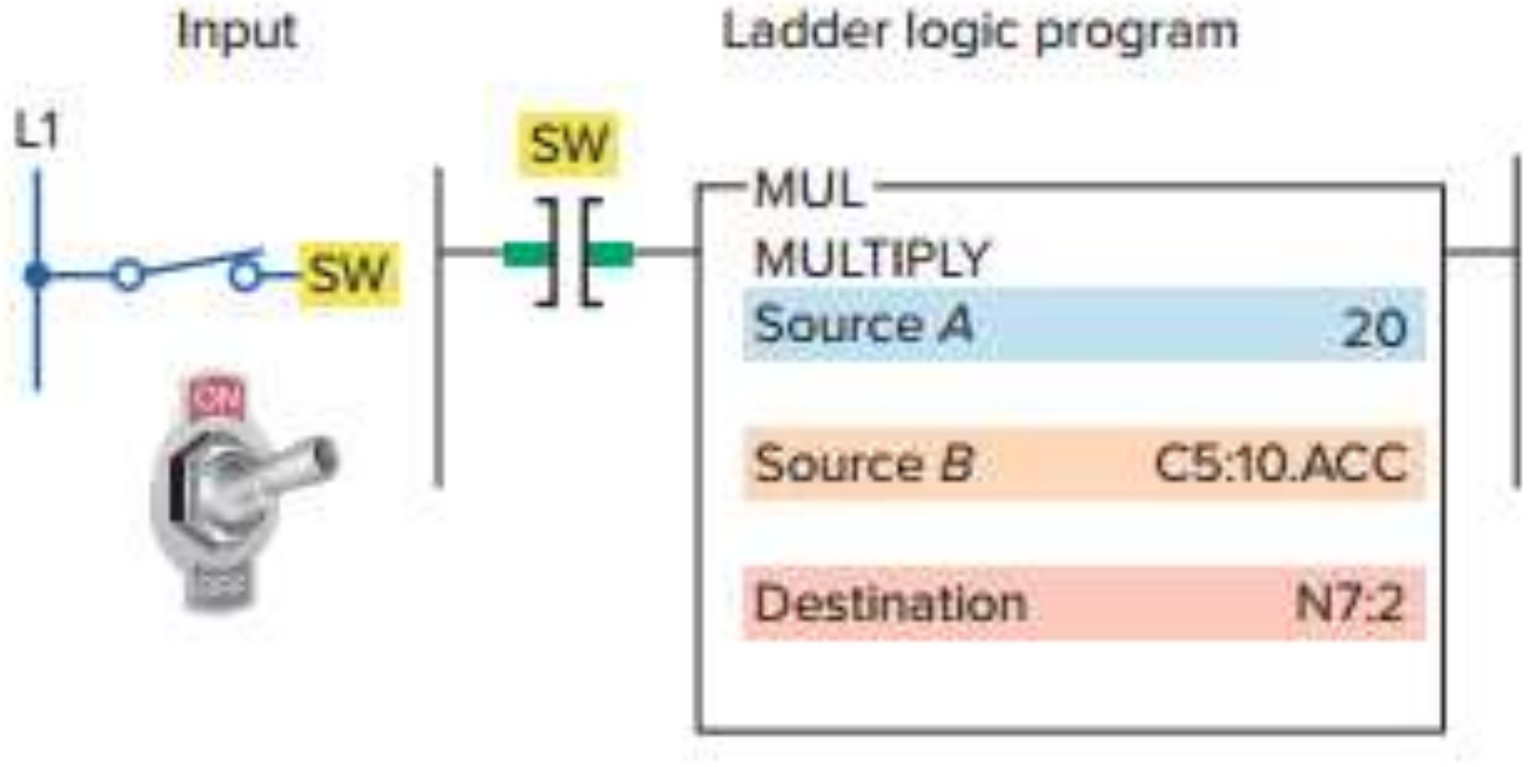


Example:

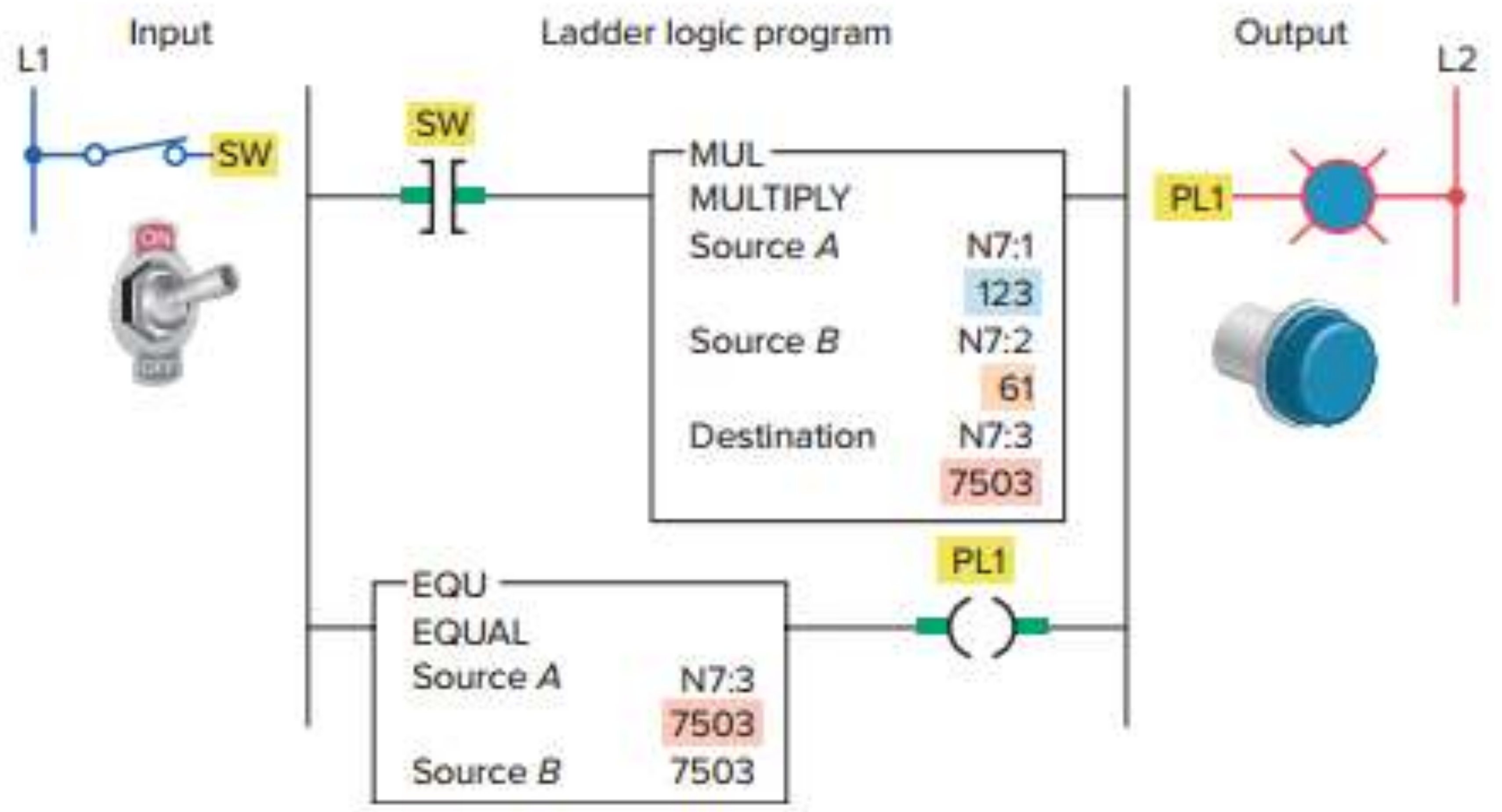
To indicate a vessel overfill condition.

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.

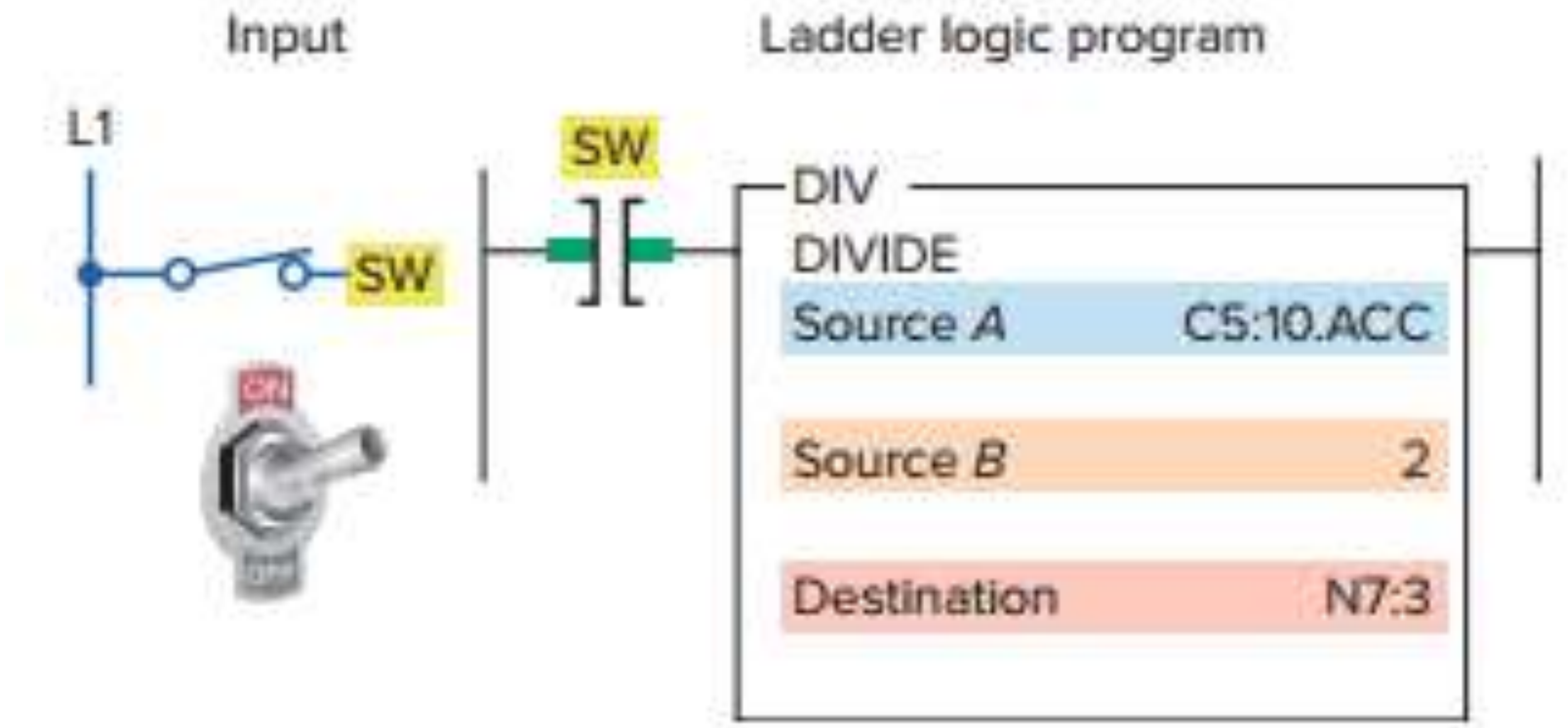
Multiplication Instruction



Example:
MUL instruction used to calculate the product of two sources.

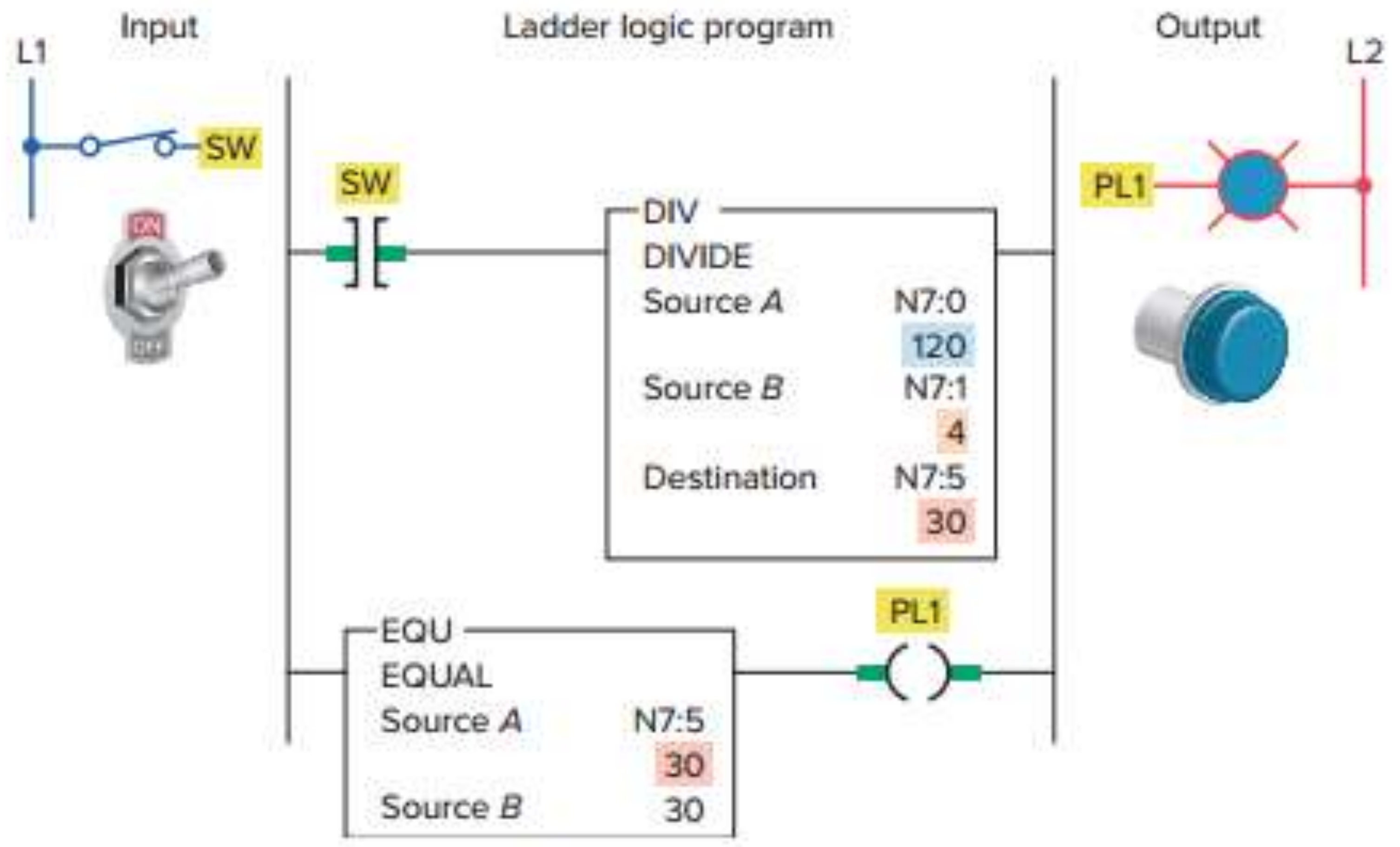


Division Instruction



Example:

DIV instruction used to calculate the value that results from dividing source *A* by source *B.*



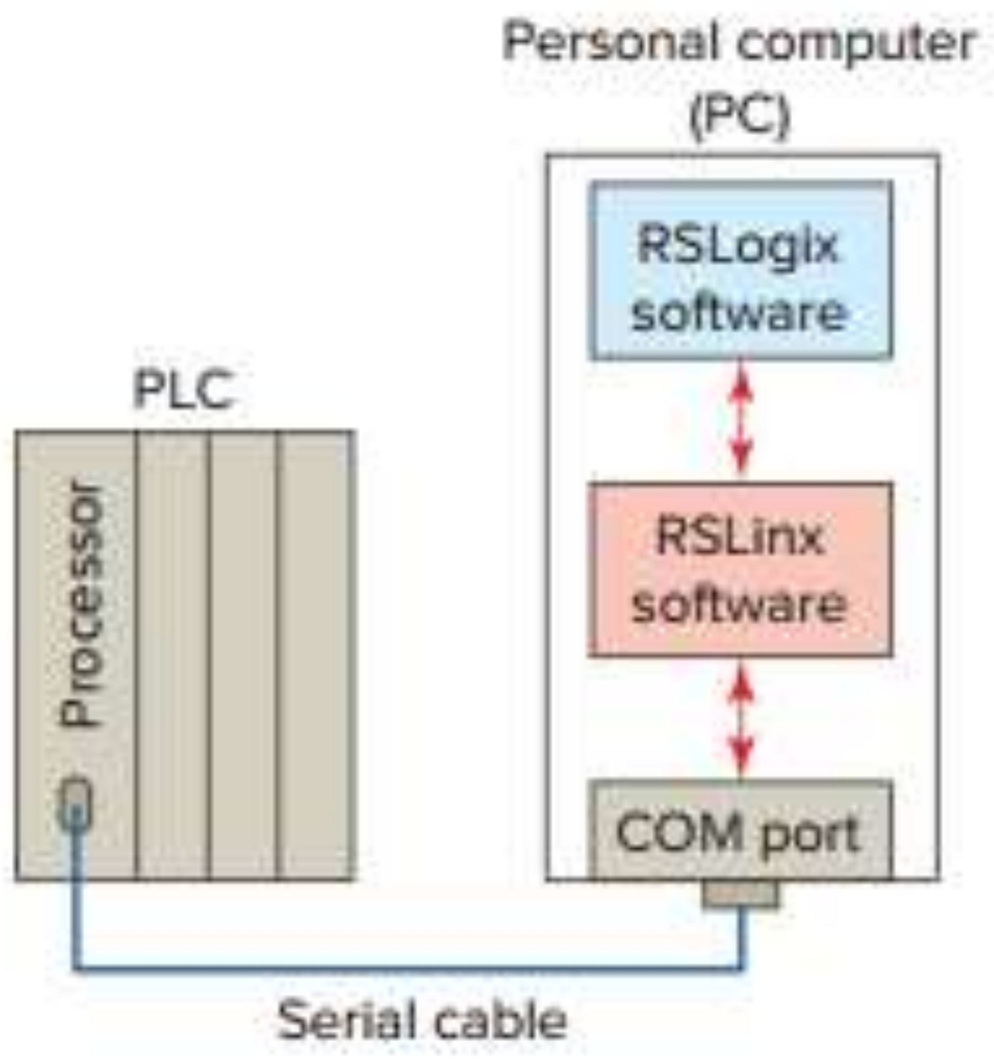
Assignment:

A program to convert Celsius temperature to Fahrenheit and vice versa using math operations.

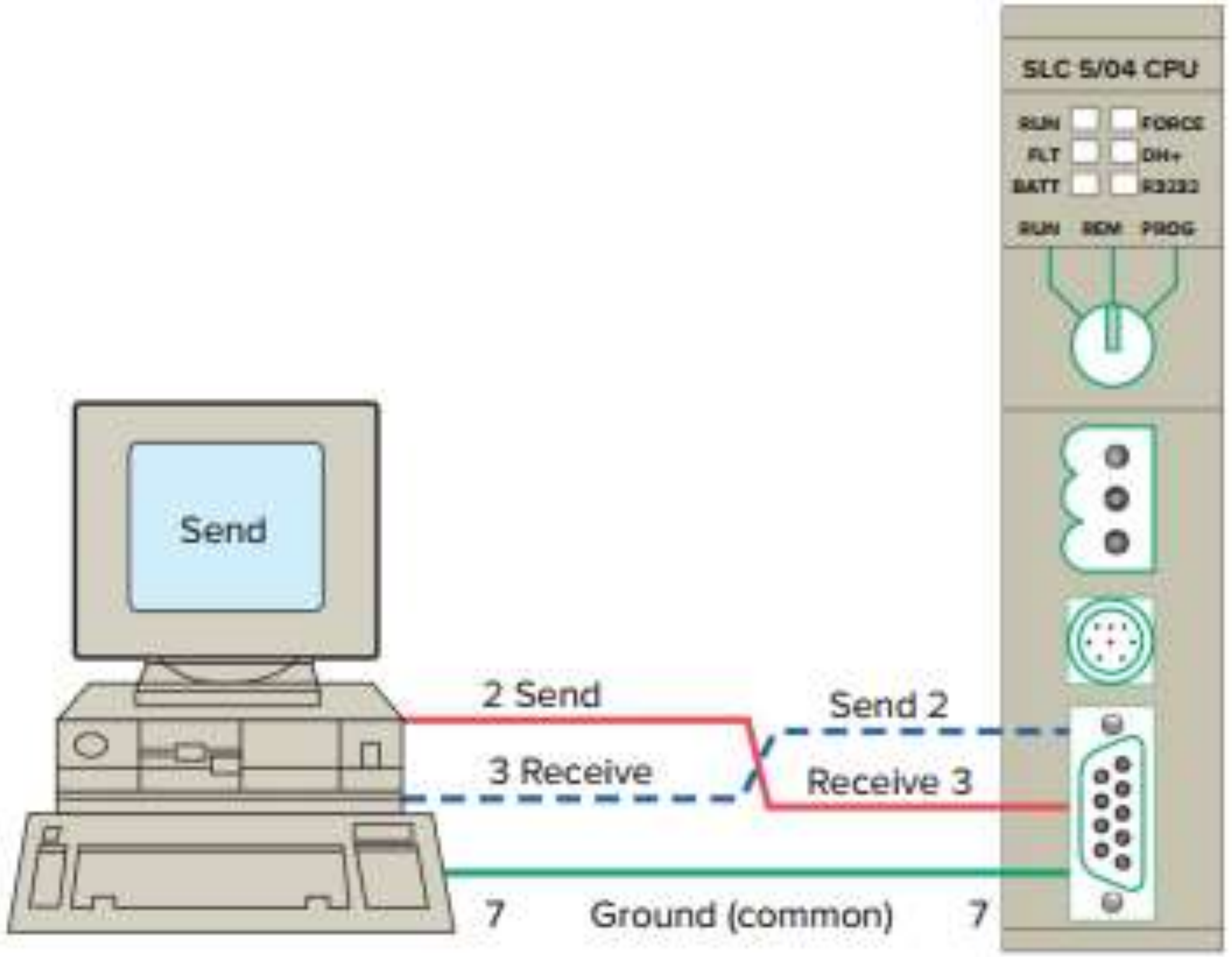
$F = (9/5 \times C) + 32$

$C = (F – 32) x 5/9$

Direct PC-to-PLC software connection

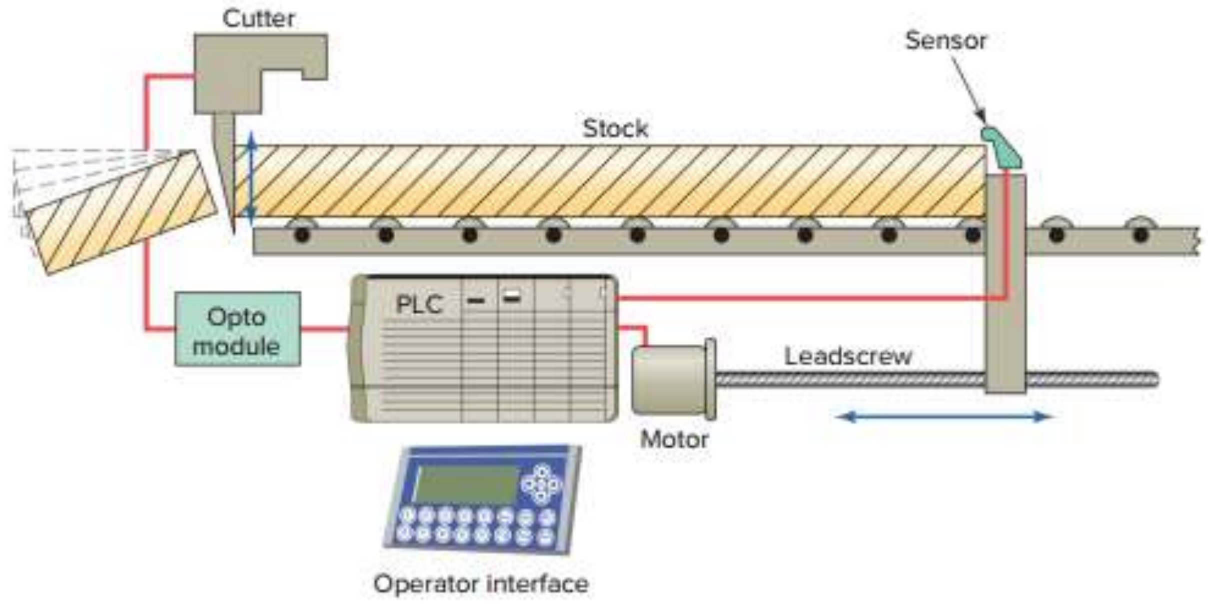


Serial wiring connection

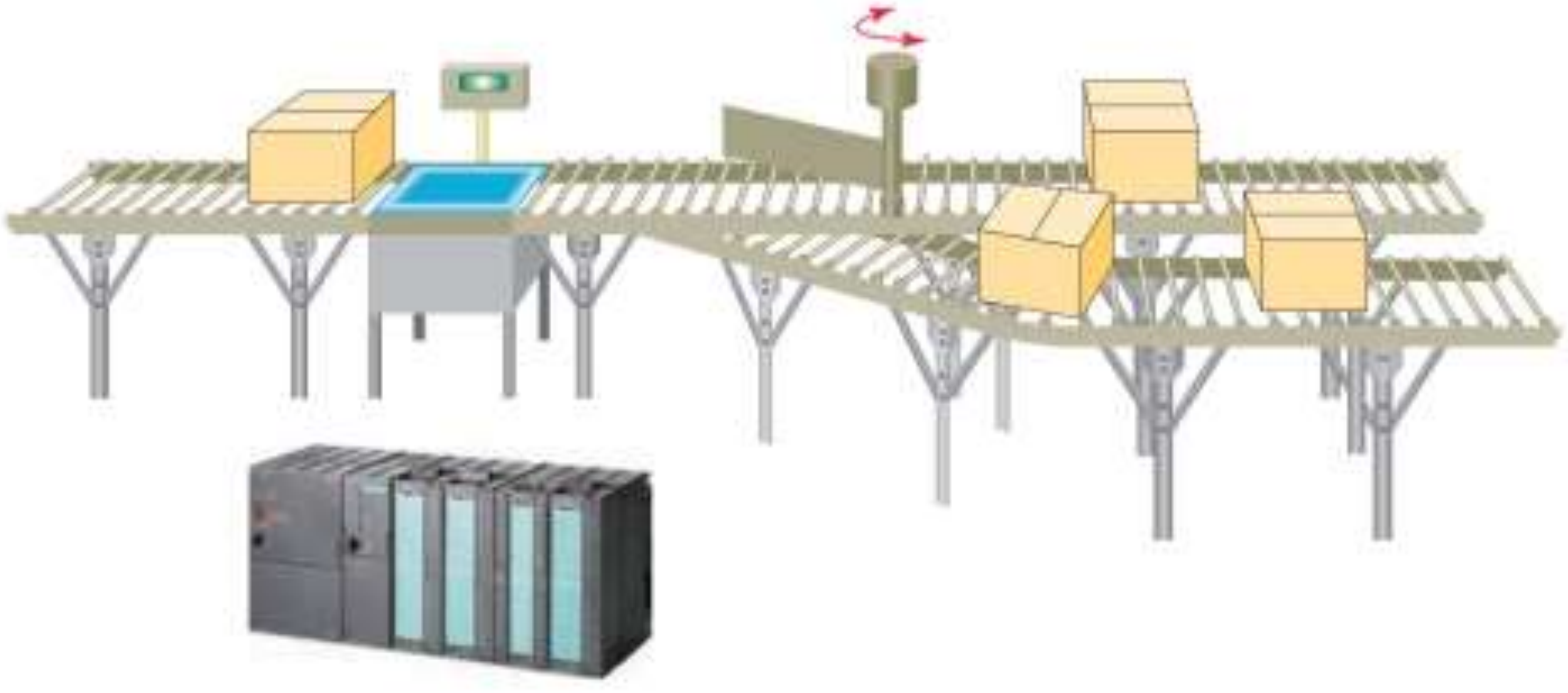


CONTROL CONFGURATIONS

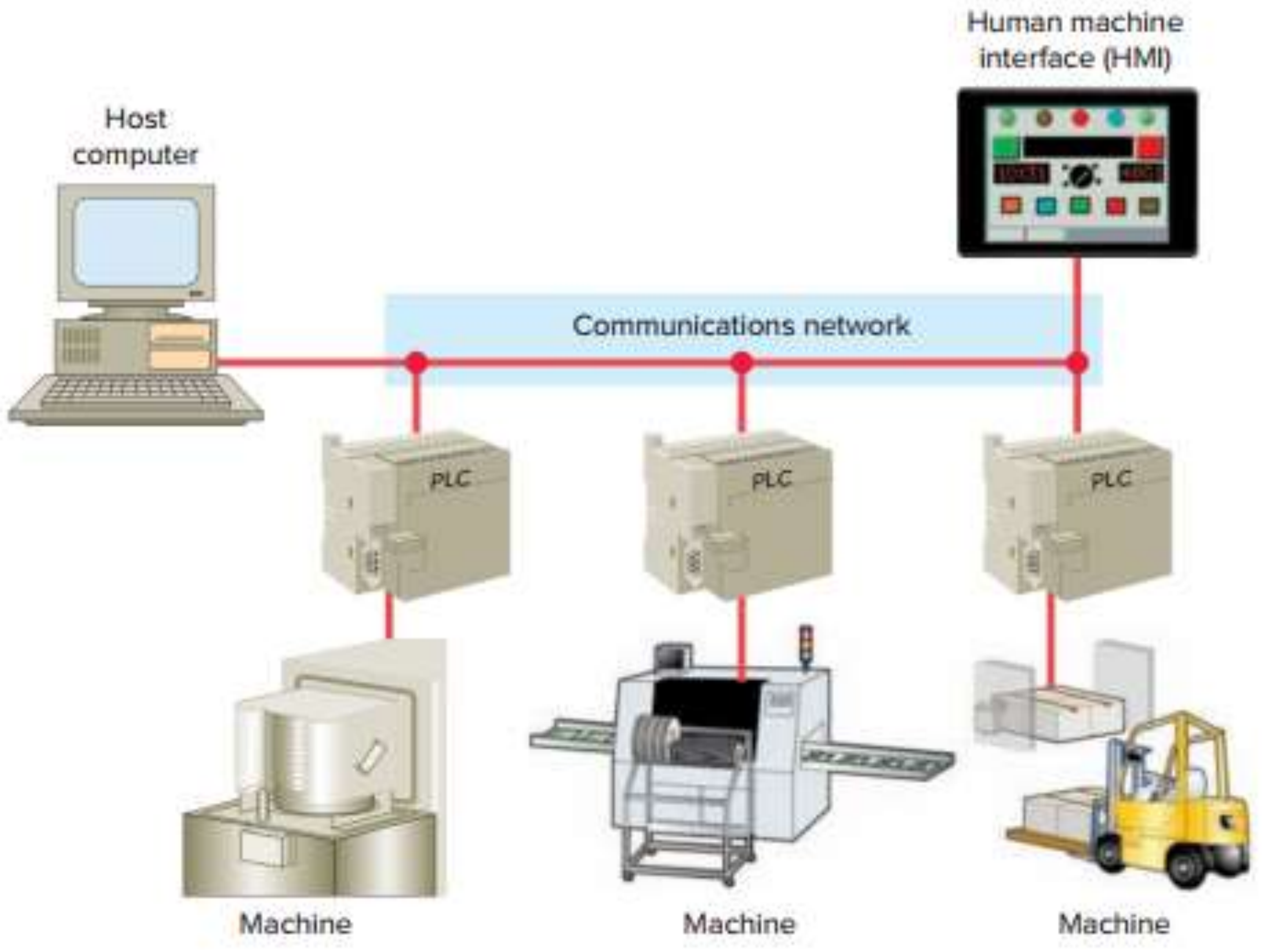
1. Individual control is used to control a single machine. This type of control does not normally require communication with other controllers .



2. Centralized control is used when several machines or processes are controlled by one central controller. The control layout uses a single, large control system to control many diverse manufacturing processes and operation



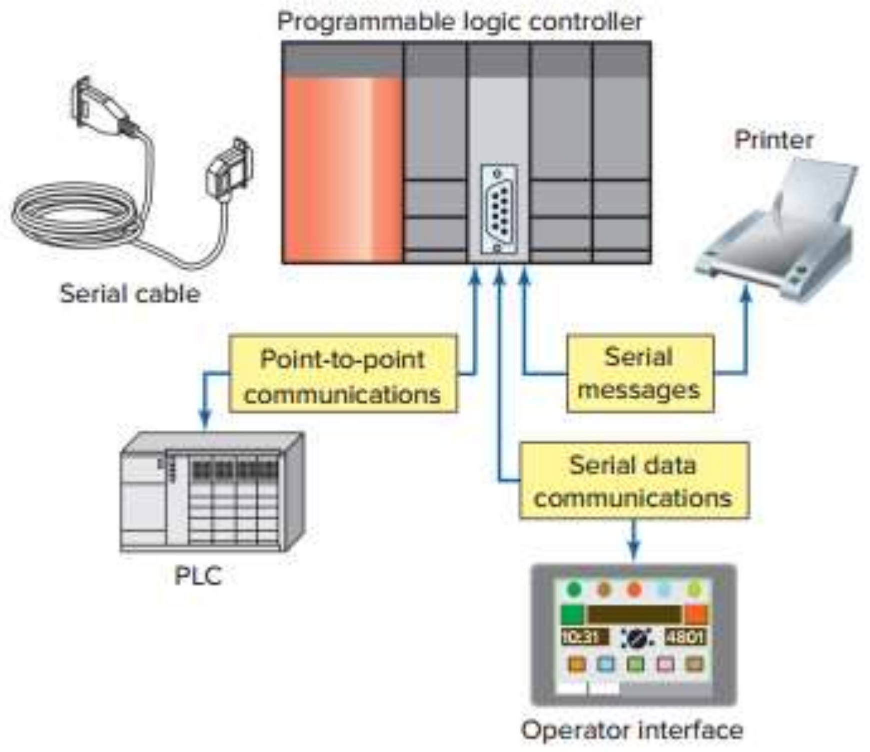
3. distributive control system (DCS) is a network based system. Distributive control involves two or more PLCs communicating with each other to accomplish the complete control task .



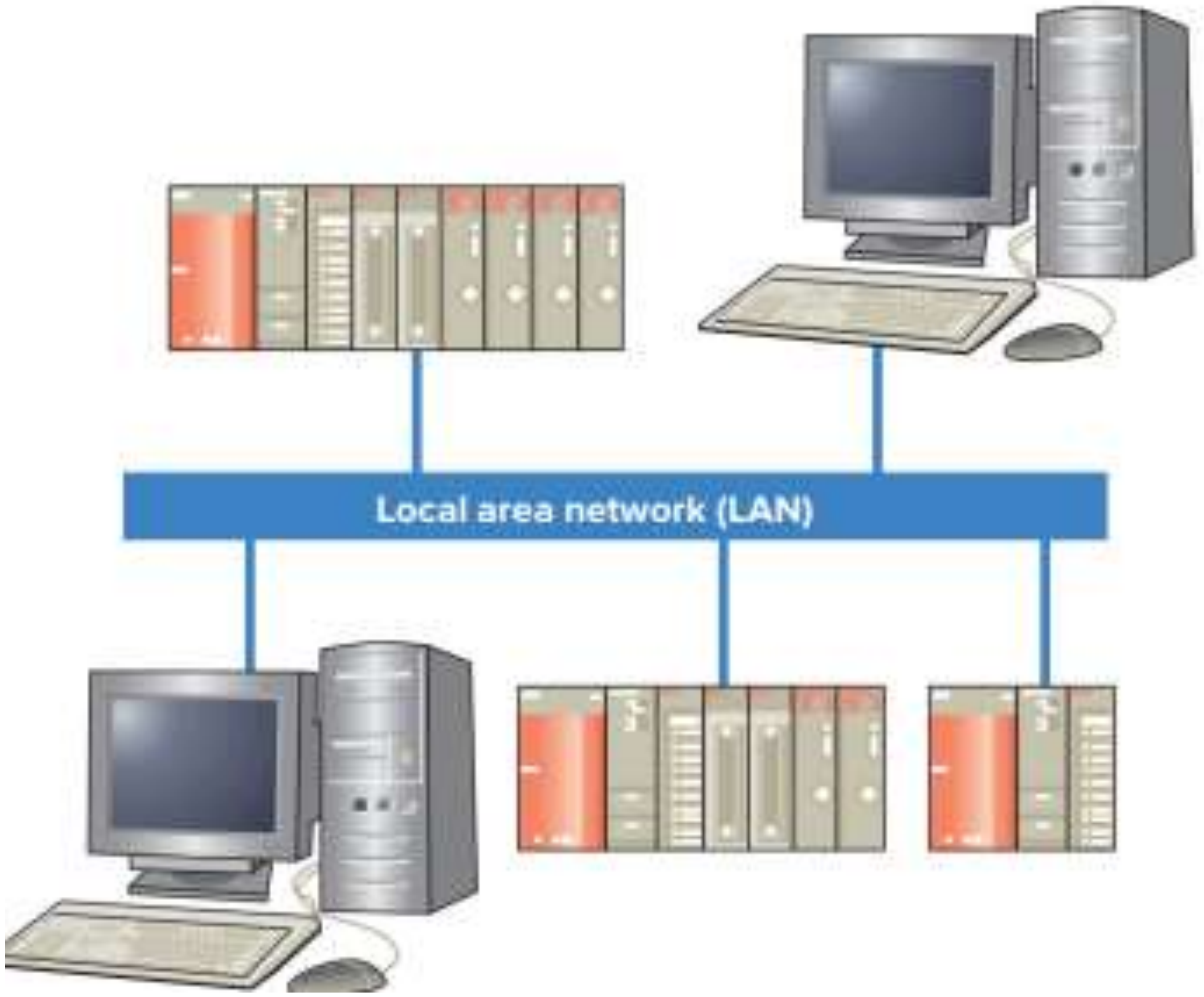
Data Communications

Data communications refers to the different ways that PLC microprocessor-based systems talk to each other and to other devices. The two general types of communications links that can be established between the PLC and other devices are point-to-point links and network links

Point-to-point serial communications link

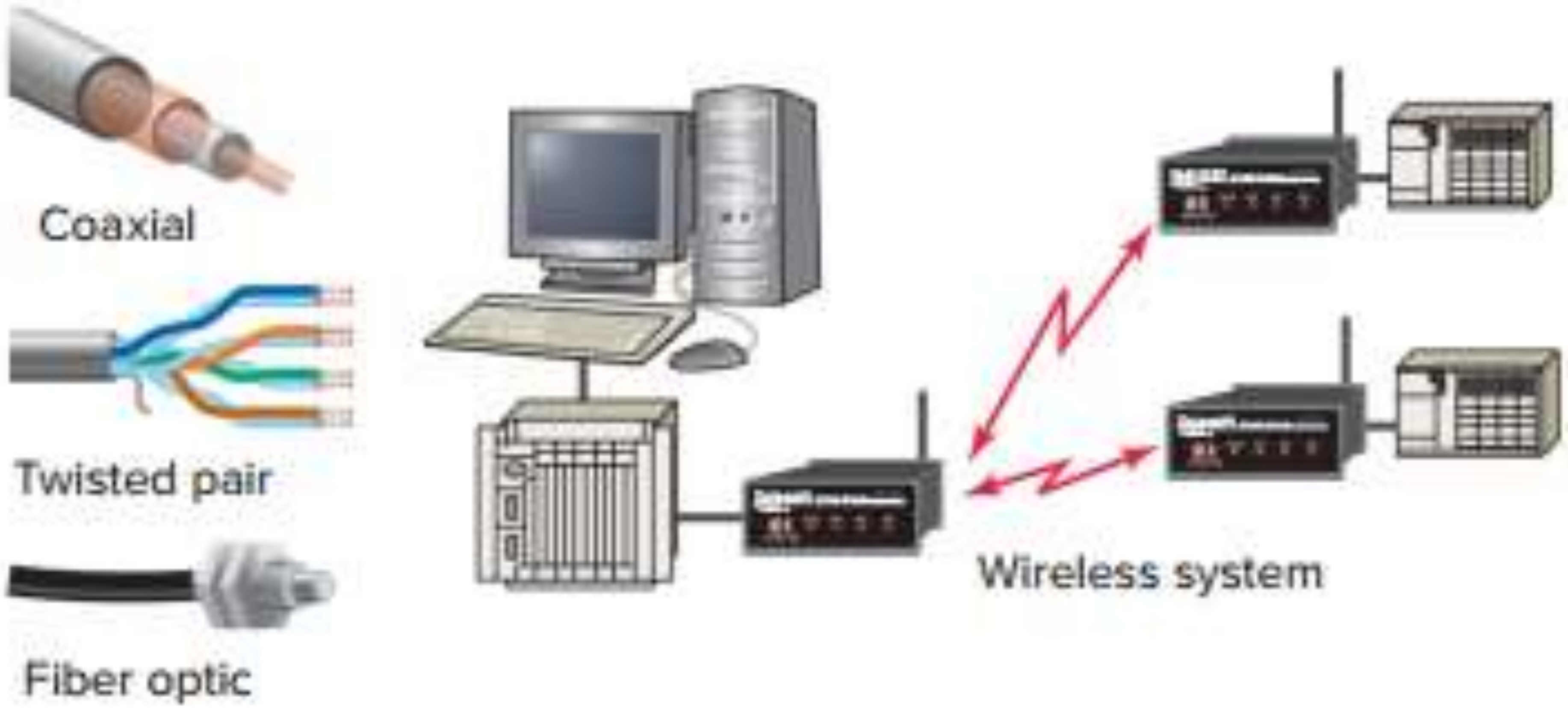


Programmable logic controller
Serial cable
Printer
Point-to-point communications
Serial messages
Serial data communications
PLC
10:31
4801
Operator interface

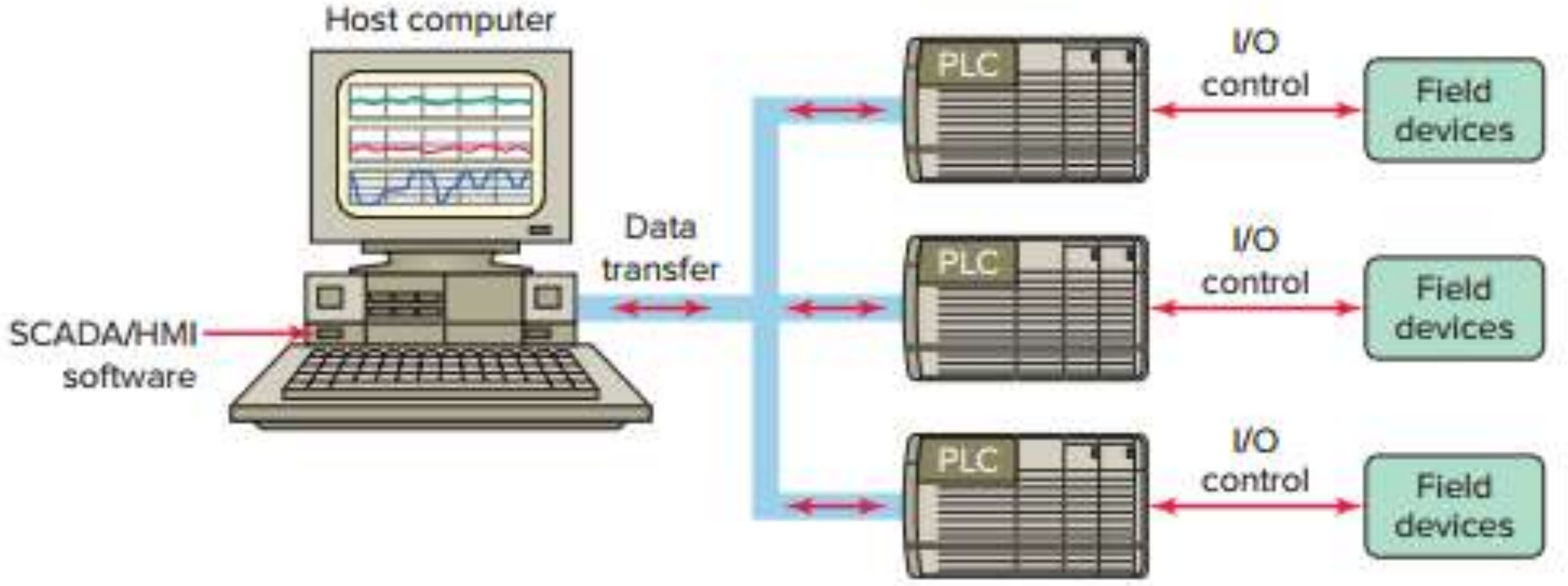


Local area network (LAN) communication link.

Transmission media.



Supervisory Control and Data Acquisition (SCADA)


Host computer
SCADA/HMI software
Data transfer
PLC
PLC
PLC
I/O control
I/O control
I/O control
Field devices
Field devices
Field devices