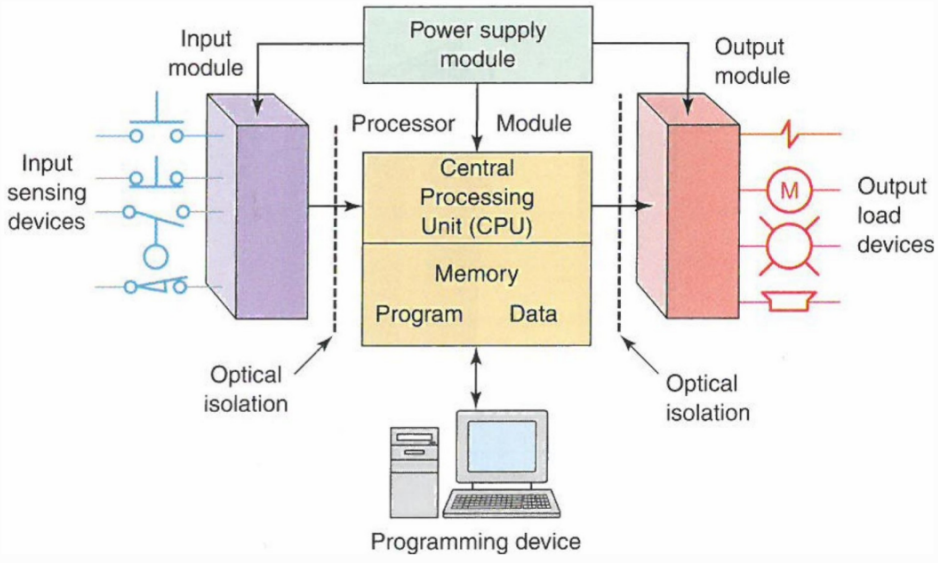
Disadvantages of relay control circuits:

- Cost
- Flexibility
- Reliability
- Communication capability
- Response time
- Troubleshooting

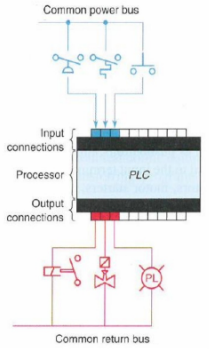
→ replaced by PLC

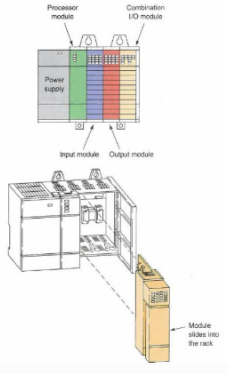
- Most widely used in industrial process control technology
- Programmable controller has eliminated much of the hardwiring
- Designed for multiple inputs and outputs
- Basically a digital computer designed for use in machine control (abbreviation PLC)
- Capable of performing switching, timing, counting, calculating, comparing, processing of analog signals

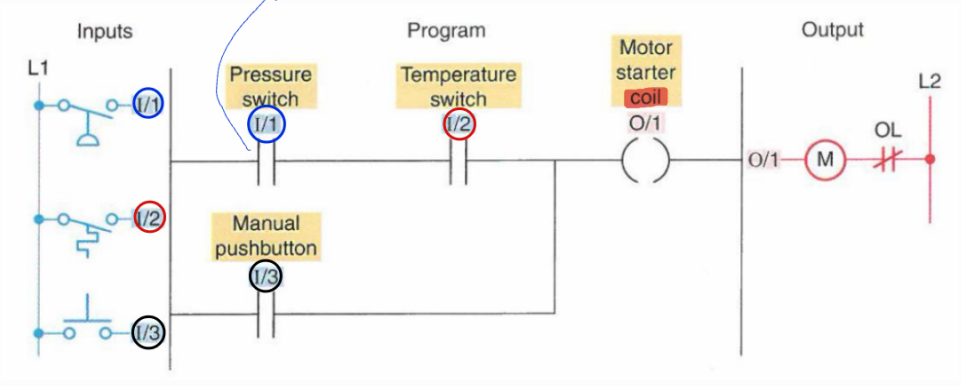
PLC Hardware: Relationship between input & output are determined by the user program.



- Fixed I/O configuration
- Modular/scalable I/O configuration

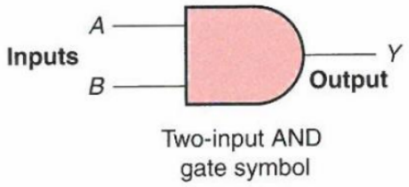






Logic gate:

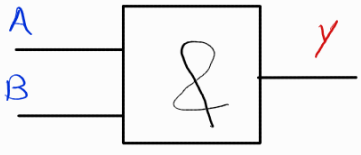
* AND Function: (connected in series)



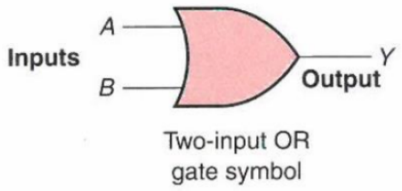
Two-input AND gate symbol

AND truth table

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



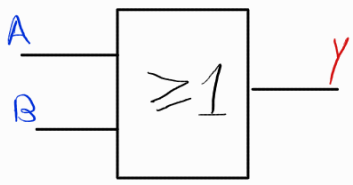
* OR Function: (connected in parallel)



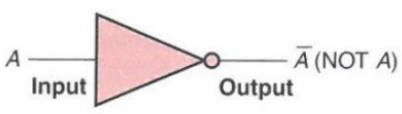
Two-input OR gate symbol

OR truth table

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

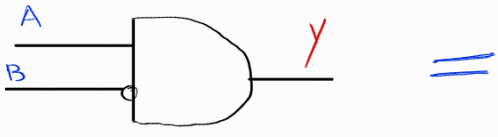


* NOT Function:

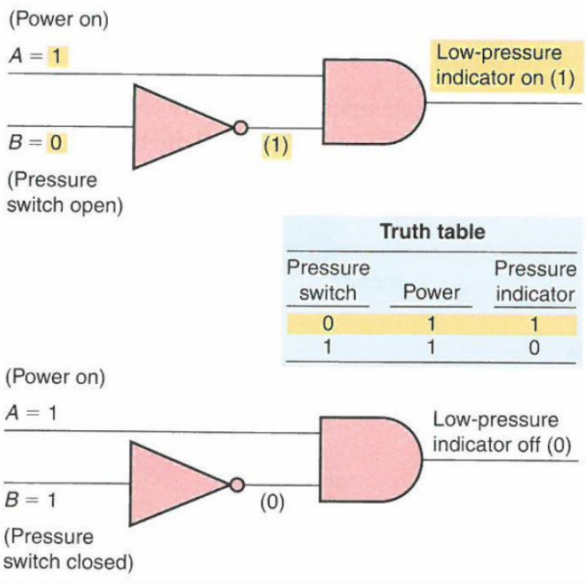


NOT truth table

A	NOT A
0	1
1	0



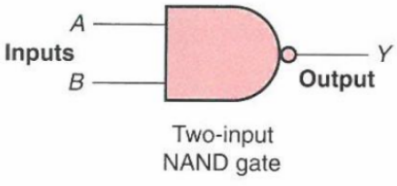
=



Truth table

Pressure switch	Power	Pressure indicator
0	1	1
1	1	0

NAND

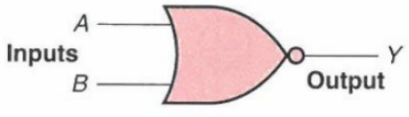


Two-input NAND gate

NAND truth table

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

NOR

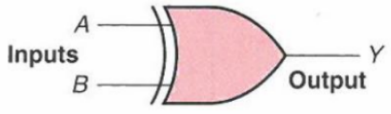


Two-input NOR gate

NOR truth table

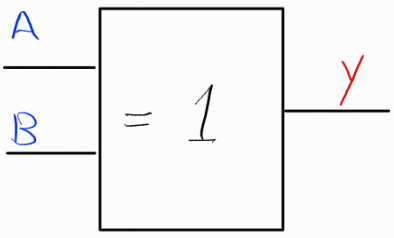
Inputs		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

* XOR Function: (Exclusive OR function)

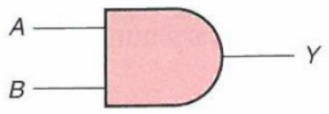
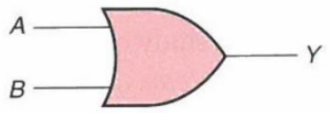
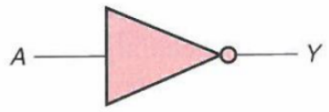


Truth table

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



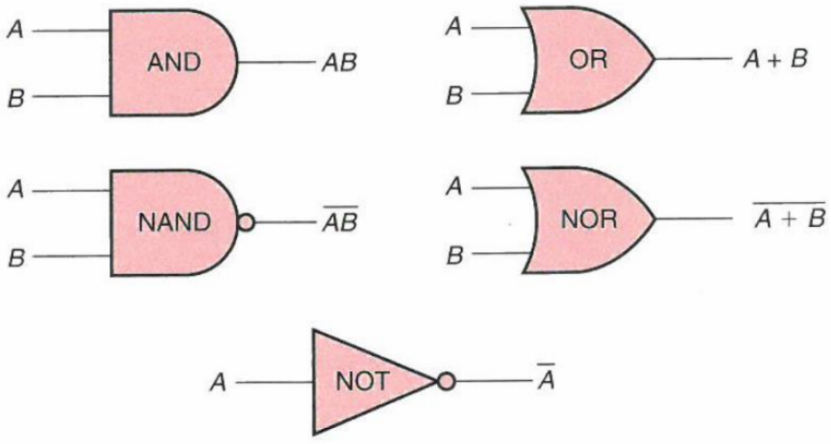
Boolean Algebra:

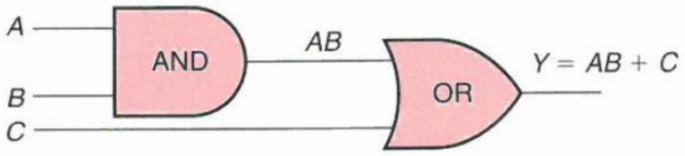
Logic symbol	Logic statement	Boolean equation	Boolean notations
	Y is 1 if A and B are 1	$Y = A \cdot B$ or $Y = AB$	Symbol · Meaning and; + or; − not; ° invert; = result in
	Y is 1 if A or B is 1	$Y = A + B$	
	Y is 1 if A is 0 Y is 0 if A is 1	$Y = \overline{A}$	

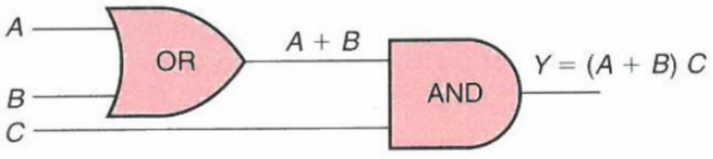
$\overline{AB} \neq \overline{A} \cdot \overline{B}$

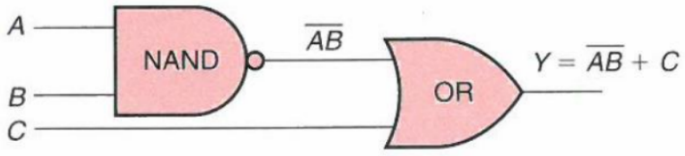
XOR → $A \oplus B$ Boolean exp.

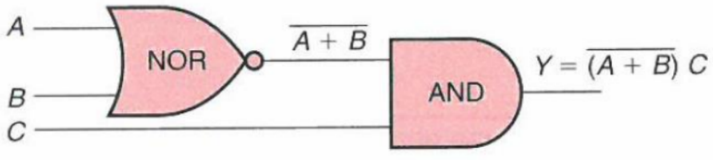
Examples:











COMMUTATIVE LAW

$A + B = B + A$

$A \cdot B = B \cdot A$

ASSOCIATIVE LAW

$(A + B) + C = A + (B + C)$

$(A \cdot B) \cdot C = A \cdot (B \cdot C)$

DISTRIBUTIVE LAW

$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$

$A + (B \cdot C) = (A + B) \cdot (A + C)$ → only in Boolean Algebra

This law holds true only in Boolean algebra.

$\overline{A}\,\overline{B} \neq \overline{A \cdot B}$

$\overline{A} \cdot \overline{B} = \overline{A + B}$

$\overline{A \cdot B} = \overline{A} + \overline{B}$

$\overline{A} + \overline{B} \neq \overline{A + B}$

XOR Function:

Truth Table:

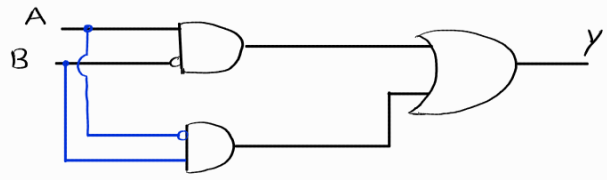
A	B	Y
0	0	0
1	0	1
0	1	1
1	1	0

Boolean expression:

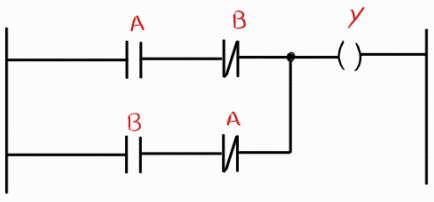
$Y = A\overline{B} + \overline{A}B$

$Y = A \oplus B$

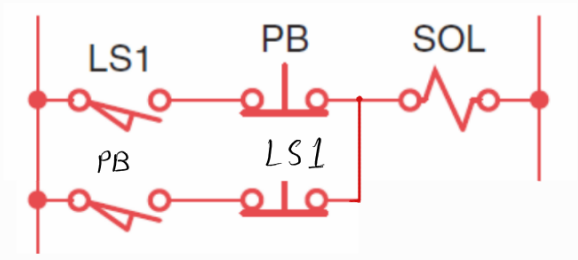
Gate logic:



Ladder logic program:



Relay schematic:



A motor control circuit contains two start/stop switches each. When either start switch (NO) (Normally open) is pressed, the motor runs. Either stop switch (NC) (Normally closed) stops the motor when it is pressed

- Relay schematic
- Ladder logic program
- Gate logic

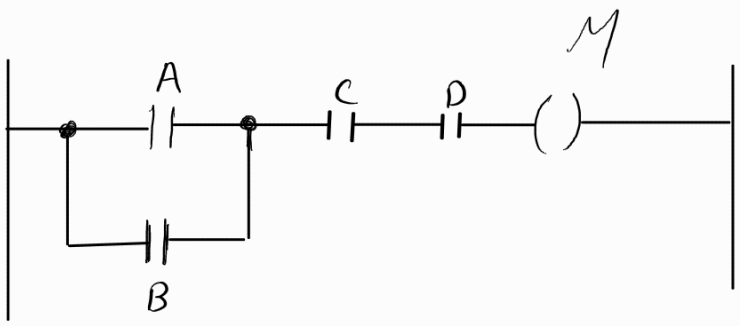
A: NO start switch 1

B: NO start switch 2

C: NC stop switch 1

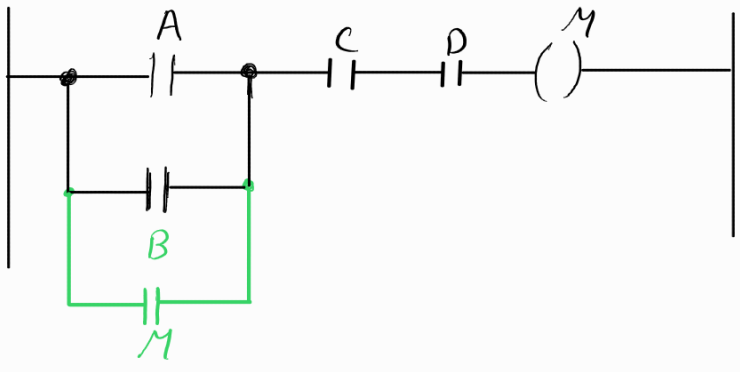
D: NC stop switch 2

M: output motor



Because it's switch, we don't need the seal in construct

When they're buttons not Switches

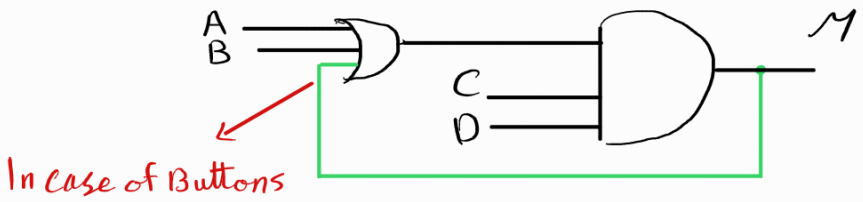


Boolean Expression:

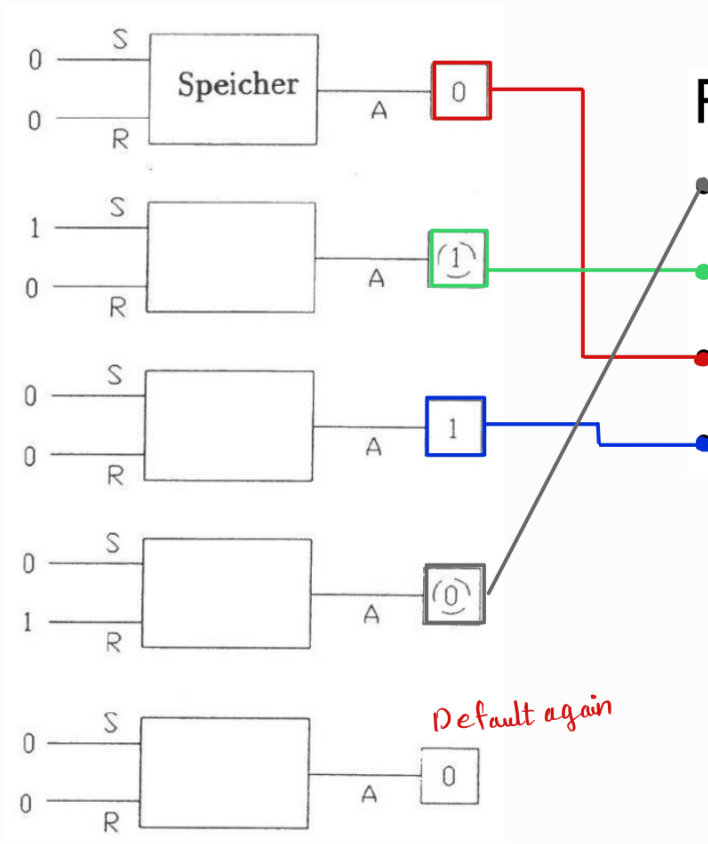
(with switches) → $M = (A+B) \cdot CD$

(with Buttons) → $M = (A+B+M) \cdot CD$

Gate Logic:



Example of a memory function



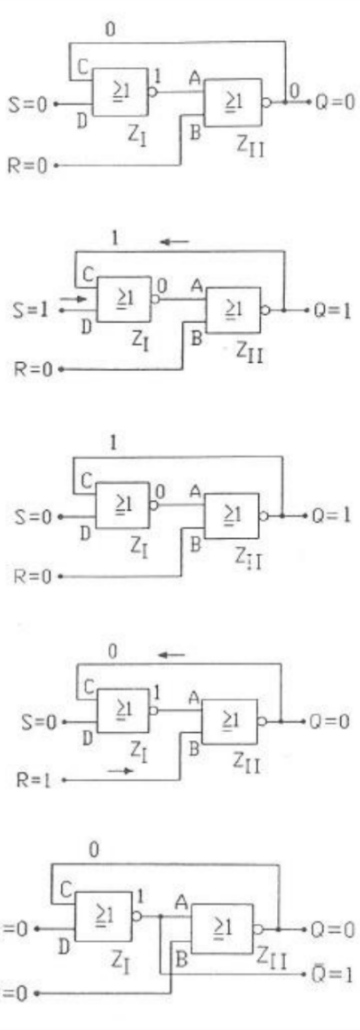
Find the correct sequence

- Reset memory
- Setting memory
- Default setting
- Save/maintain state

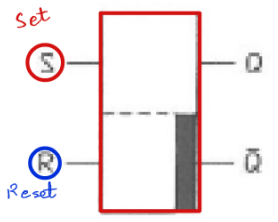
Flip-Flop elements

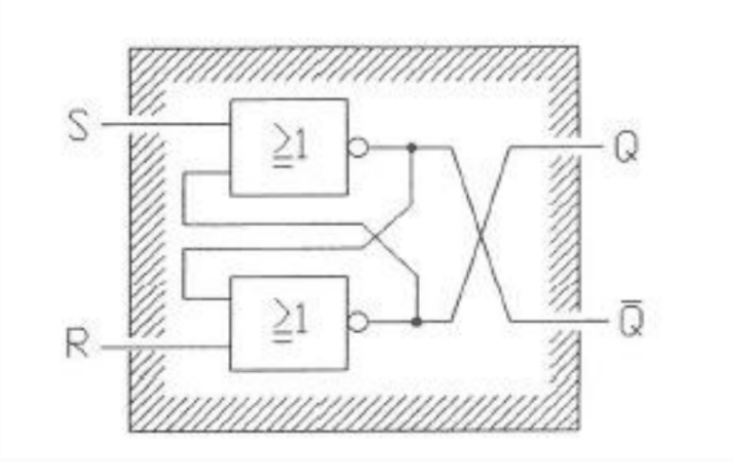
- Bistable
- means → Can, on command, flip into one stable state of flop back again to the other
- Basis RS-Flip-Flop using NOR-Gates

1. Default setting
2. Setting memory
3. Save/maintain memory
4. Reset memory
5. Default setting

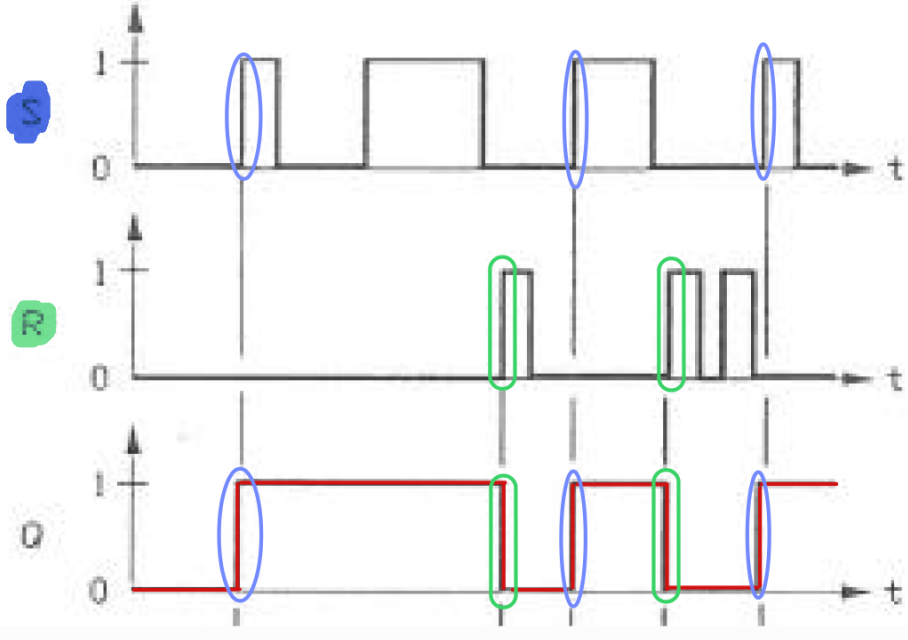


Symbol:

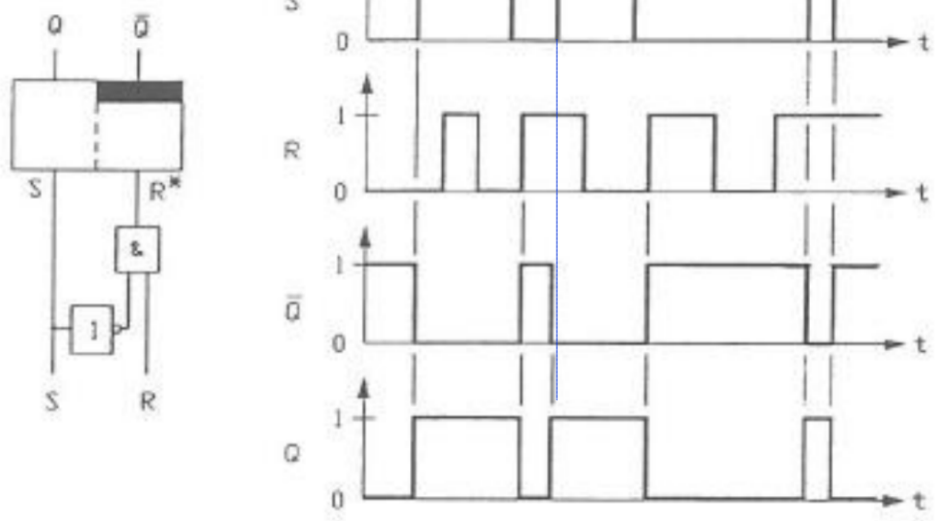


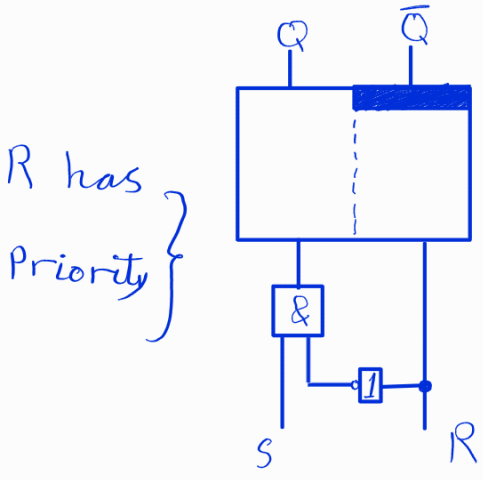


- Combination R = S = 1 not allowed
- Pre-logic defines priority of either R or S

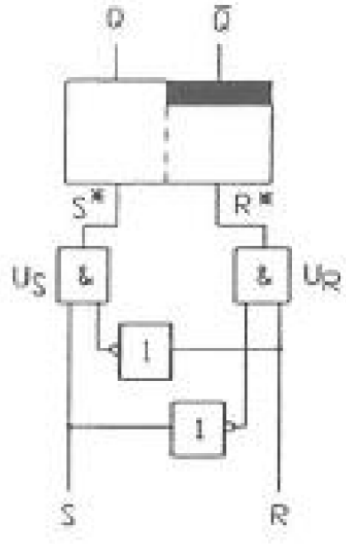


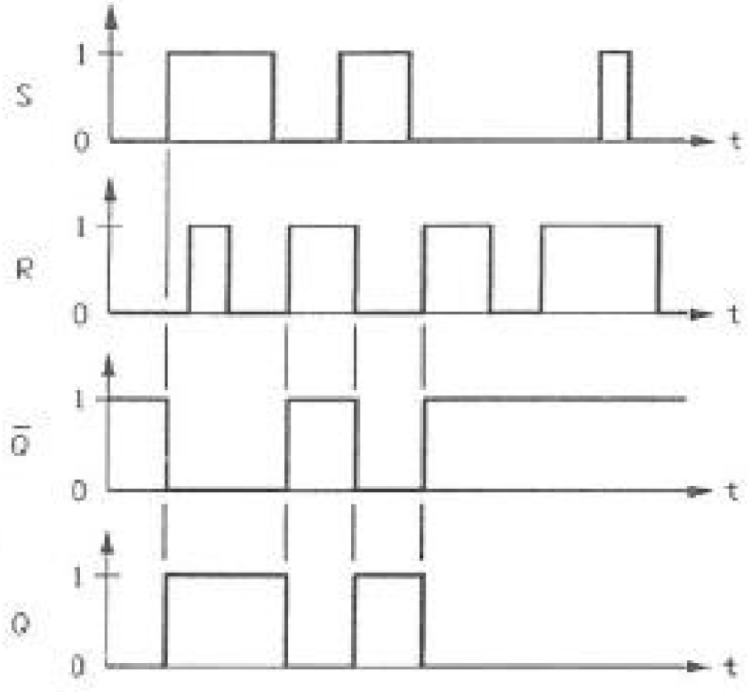
The pre-logic in this case priorities S over R

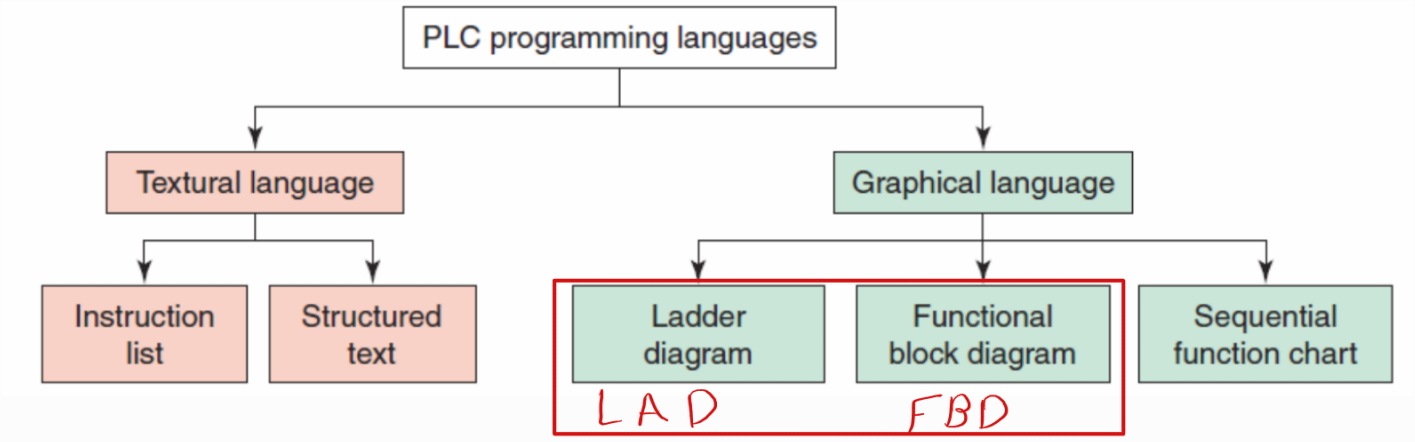




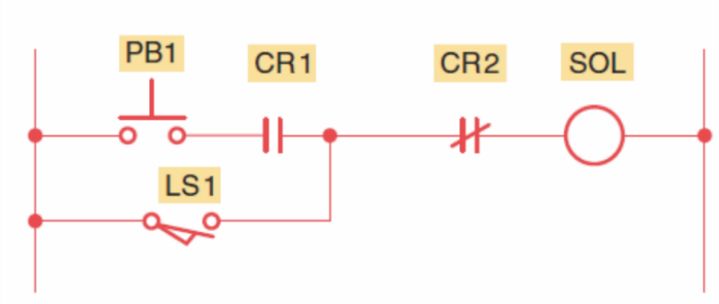
Locking:-



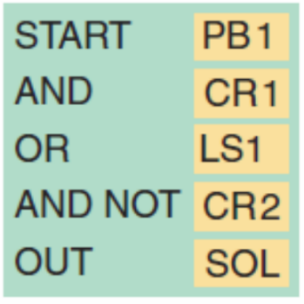




Instruction list: programming language consists of basic AND, OR, and NOT logic gate functions.

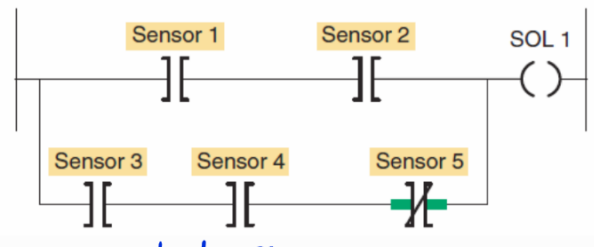


Hardwired relayed control circuit



Equivelant instruction list program

Structured text: Is a high level language primarily used to implement more complex procedures.

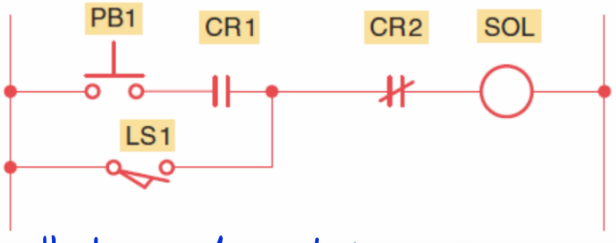


Lader Diagram

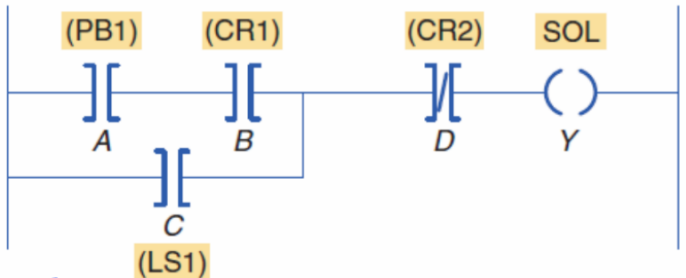
```
IF Sensor_1 AND Sensor_2 THEN
        SOL_1 := 1;
ELSEIF Sensor_3 AND Sensor_4 AND NOT Sensor_5 THEN
        SOL_1 := 1;
END_IF;
```

Equivalent structured text program

Ladder diagram: Is a common used PLC Language and designed to mimic hardware relay logic.



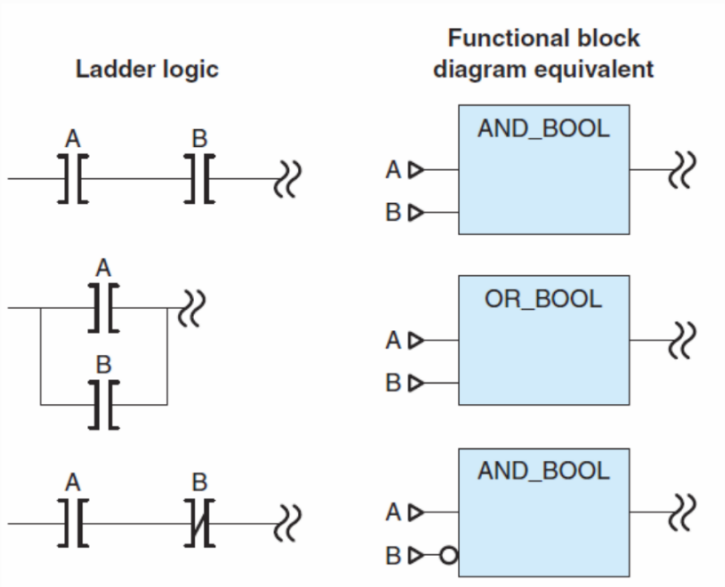
Hardware relay control circuit

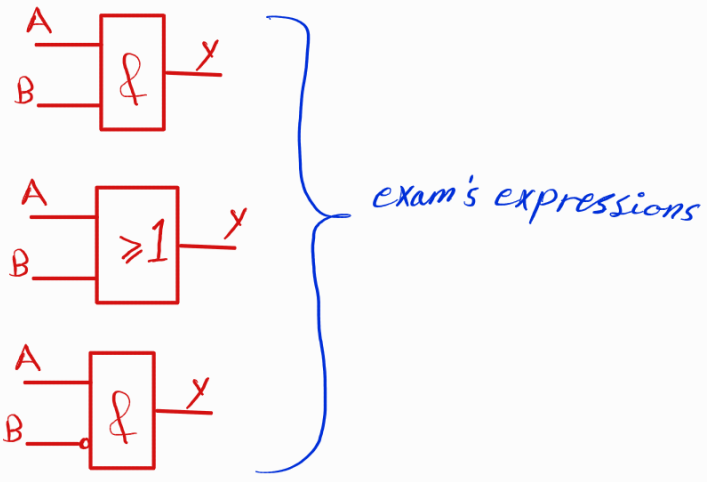


Equivalent Ladder diagram program

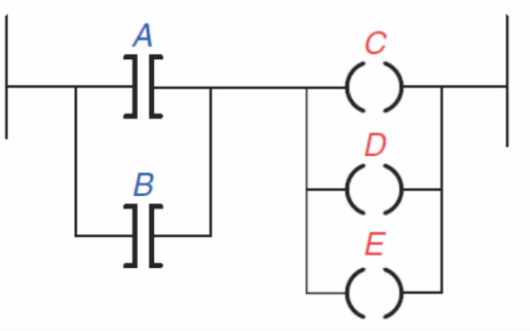
Functional block diagram:

programming uses instructions that are programmed as blocks wired together to accomplish certain functions.

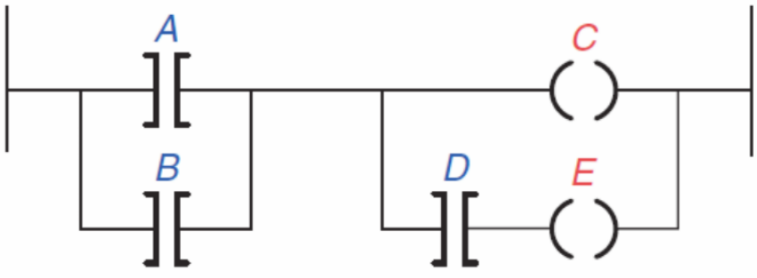




Output branching allows a true logic path to control multiple outputs. However, a branch can only have one output. Thus, multiple outputs must be arranged in parallel.

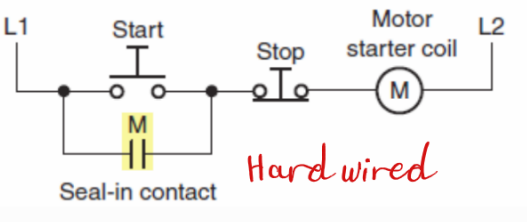


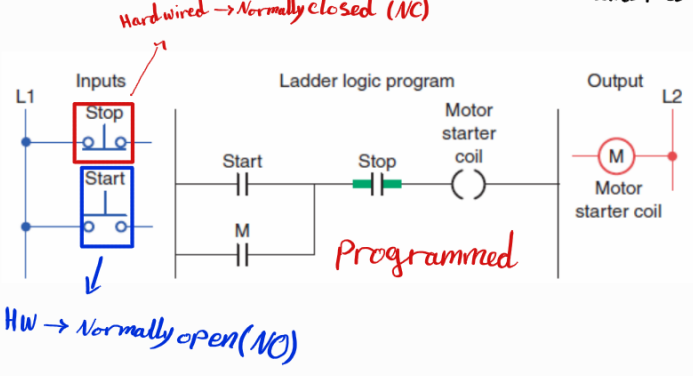
Either A or B provides a true logical path to all three output instructions: C, D, and E.

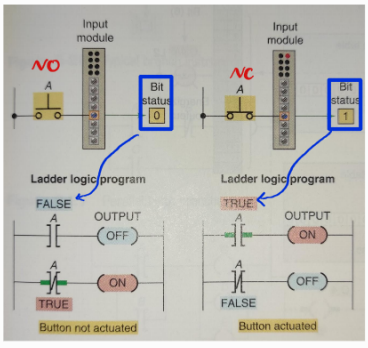


Additional input instructions can be programmed in the output branches.

A Seal-in Circuit is a method of maintaining current flow after a momentary switch has been pressed and released.

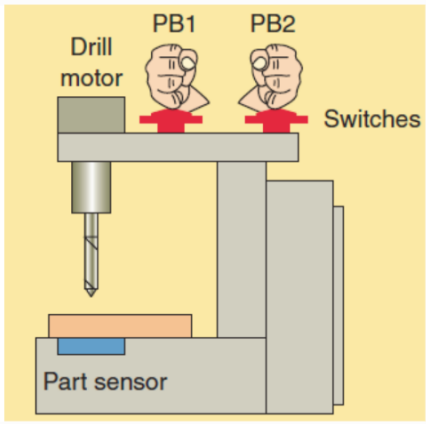


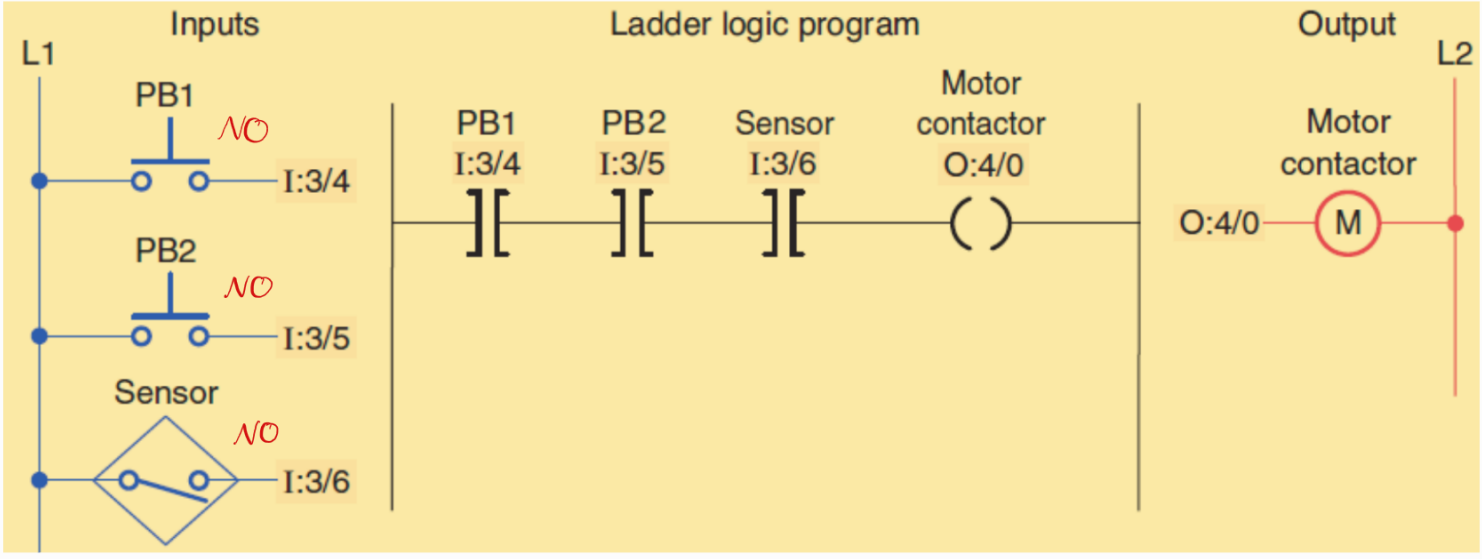




Writing a Ladder Logic Diagram from a narrative description

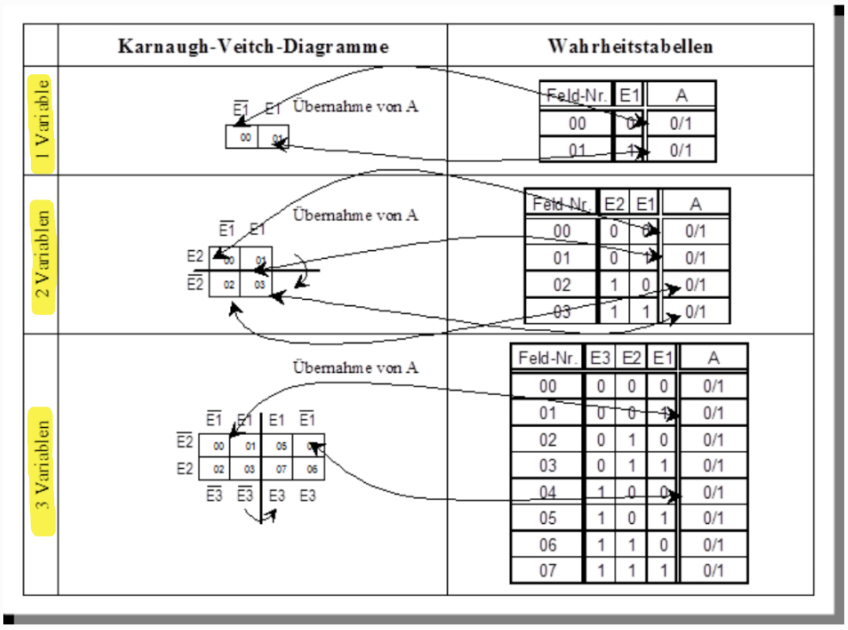
Drilling process that requires the drill press to turn on only if there is a part present and the operator has one hand on each of the start button.





Karnaugh-Veitch (KV) - Diagram:

- Tool to find optimized Logical function.
- Truth table in a 2D-array
- Using symmetries and fields can be summarized. ($S1+\overline{S1}=1$)



4 Variables

	E3		$\overline{E3}$		
E4	1	0	0	1	$\overline{E2}$
	1	1	0	0	E2
$\overline{E4}$	1	1	1	1	
	0	1	1	1	$\overline{E2}$
	$\overline{E1}$	E1		$\overline{E1}$	

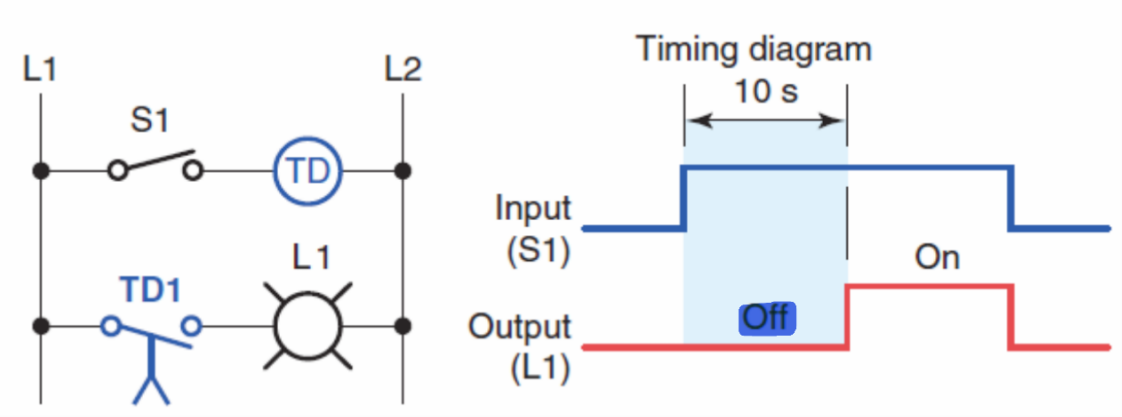
$A = E2 \cdot E3 + E1 \cdot \overline{E4} + \overline{E3} \cdot \overline{E4} + \overline{E1} \cdot \overline{E2} \cdot \overline{E3}$

Industrial Timing Tasks:

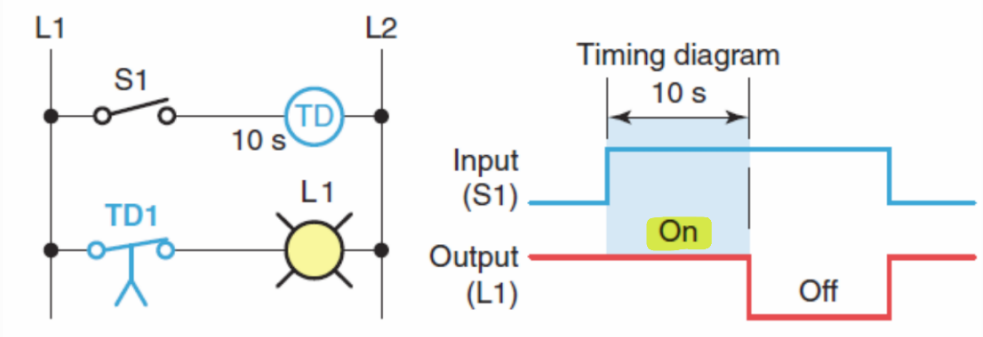
- Control of traffic lights
- Time shift motor activation.
- Warning sequence

* On-delay timer (DOE):

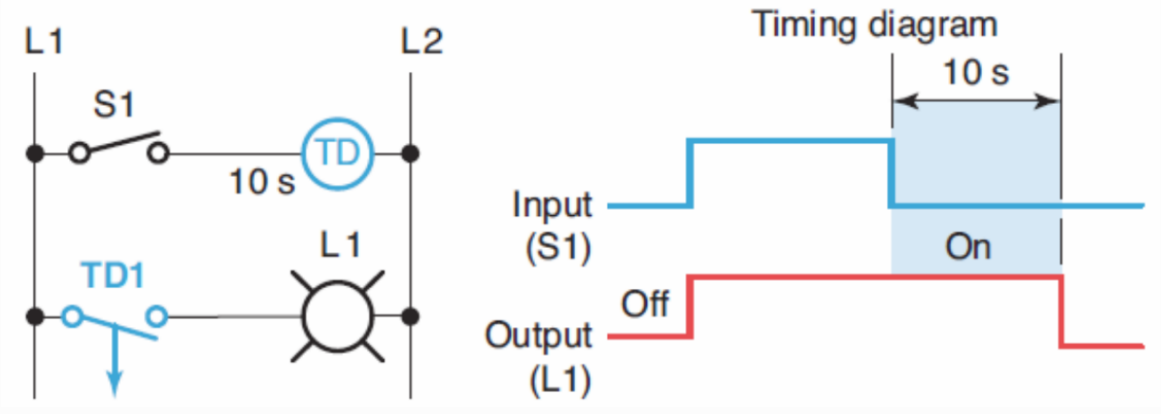
- It delays the ON time a given num. of seconds.



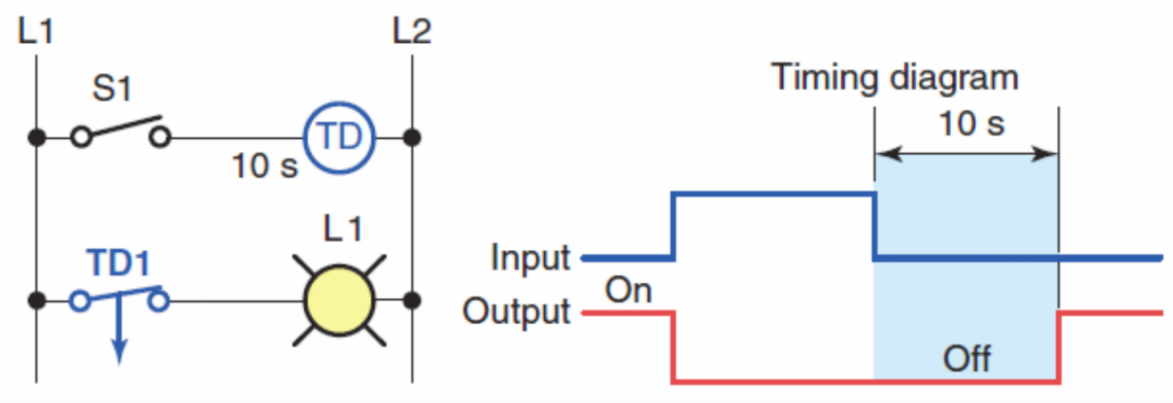
On-delay timer circuit that uses a **normally closed**, timed open (**NCTO**) contact.

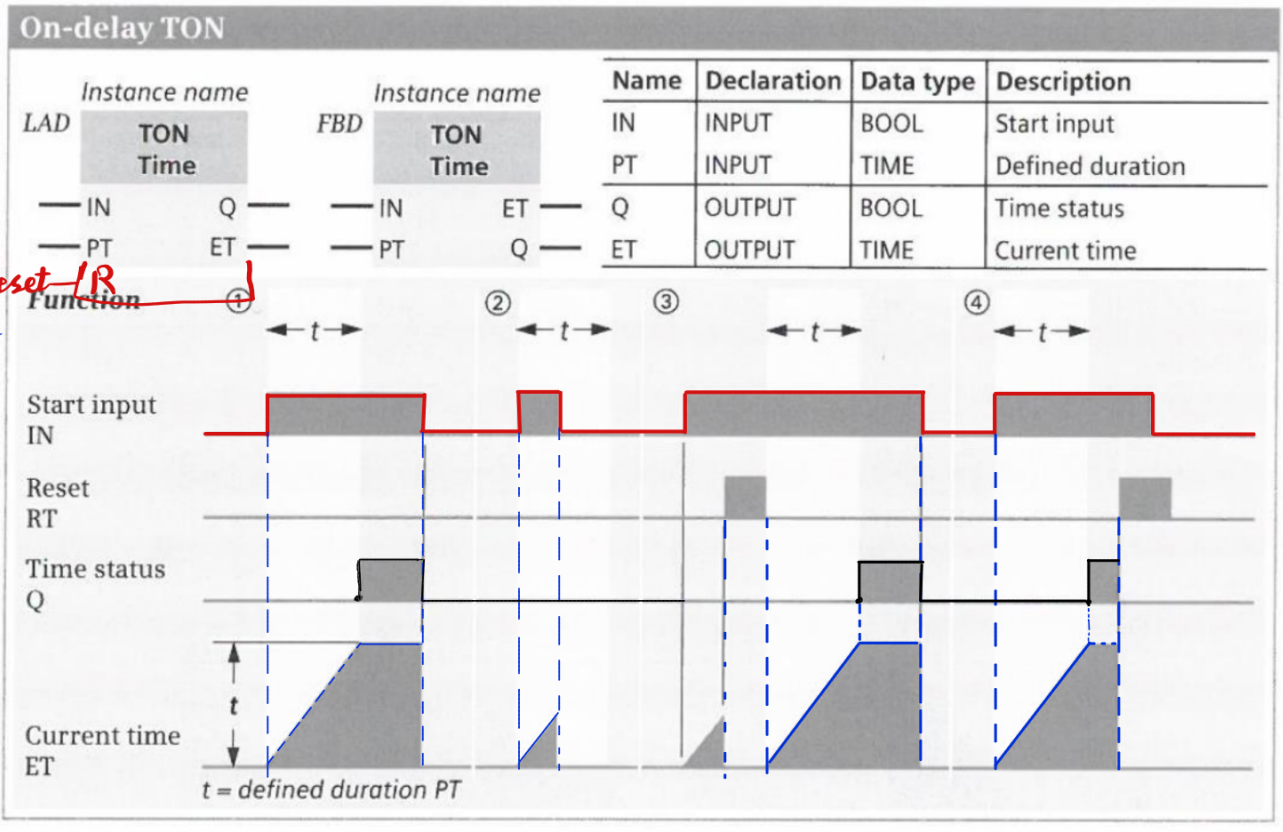


* Off-delay timer (DOD)



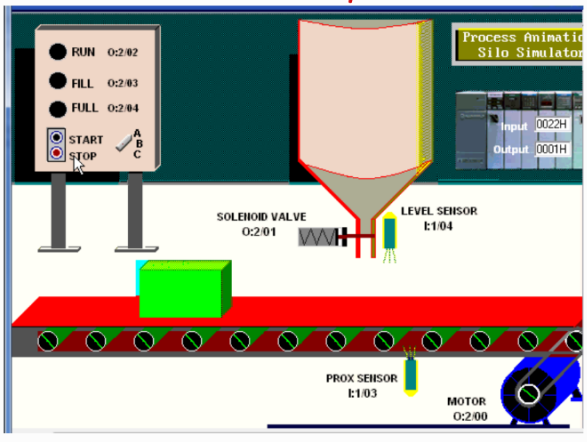
Off-delay timer circuit that uses a **normally closed**, timed closed (**NCTC**) contact.



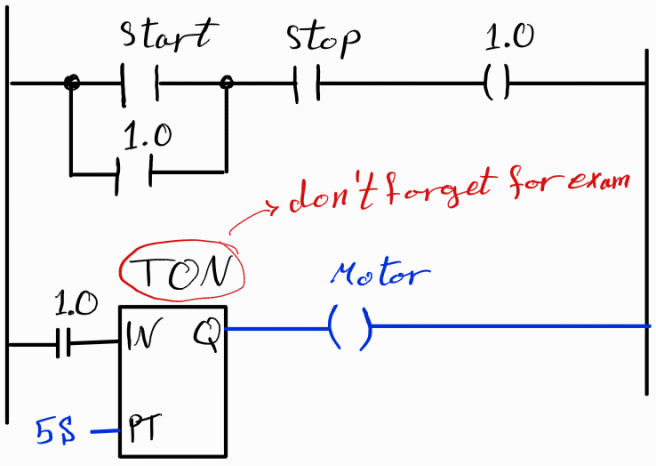


Name	Declaration	Data type	Description
IN	INPUT	BOOL	Start input
PT	INPUT	TIME	Defined duration
Q	OUTPUT	BOOL	Time status
ET	OUTPUT	TIME	Current time

Reset: either by using RT to reset, or give no input (0)



A conveyer belt starts 5sec after start button is activated.



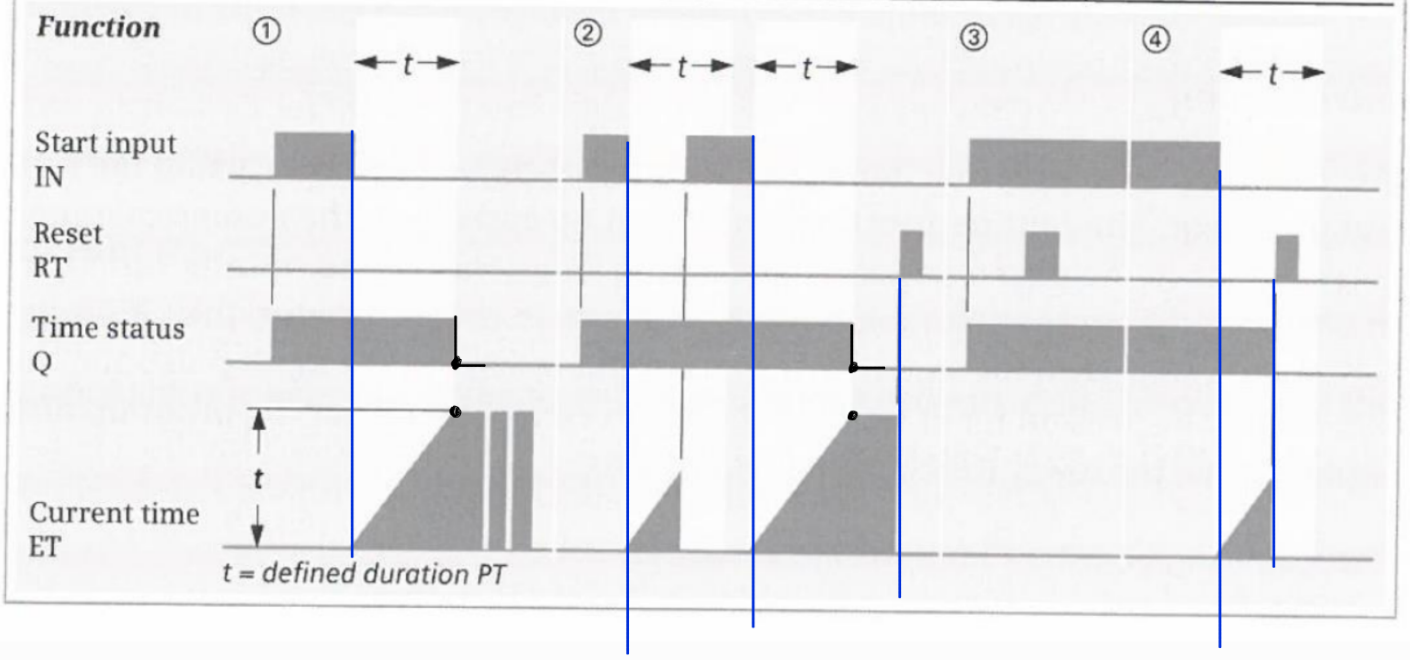
Off-delay TOF

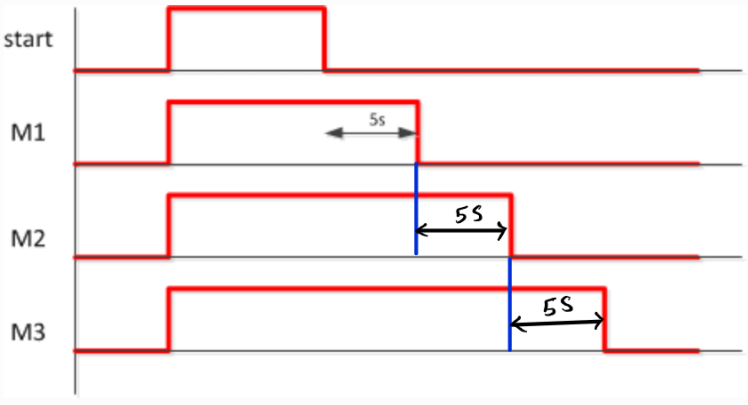
Q = 0 طالما مفيش input داخل لو من الأول ، يبقى

	Instance name		Instance name
LAD	TOF Time	FBD	TOF Time
	IN Q		IN ET
	PT ET		PT Q

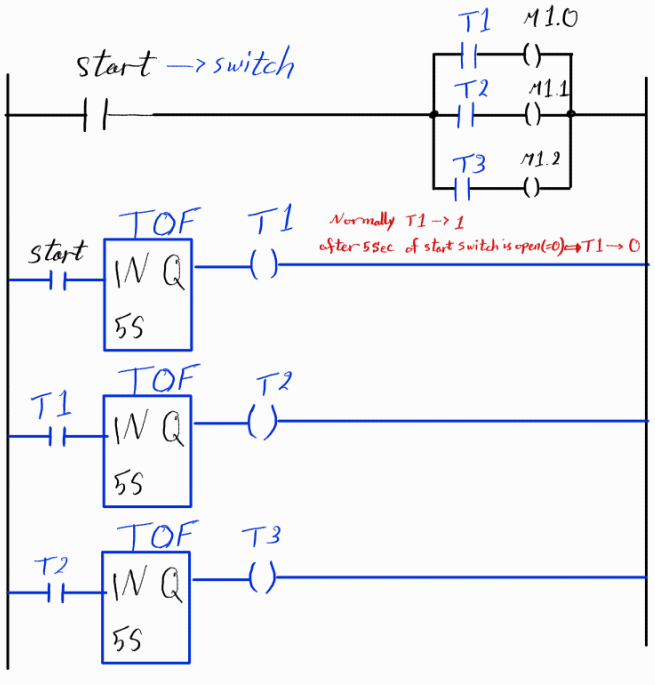
Name	Declaration	Data type	Description
IN	INPUT	BOOL	Start input
PT	INPUT	TIME	Defined duration
QO	UTPUT	BOOL	Time status
ET	OUTPUT	TIME	Current duration

Function



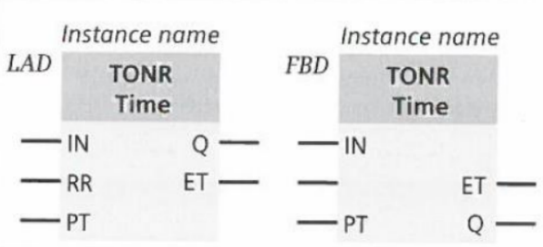


Usage of a off-delay timer instruction to switch motor off sequentially at 5s intervals.



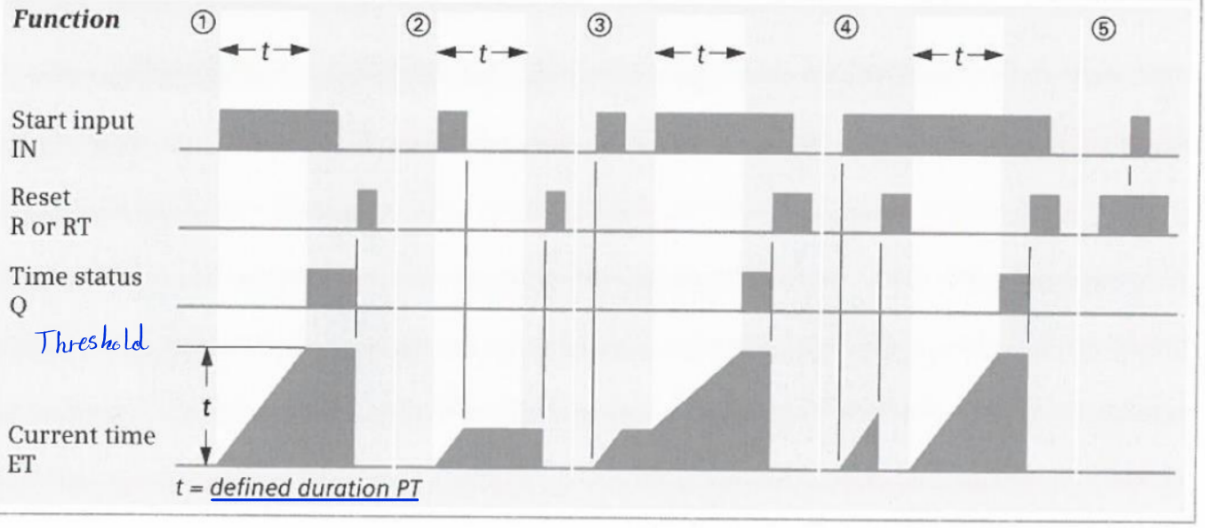
Retentive on-delay TONR / RTO, representation and function

Retentive on-delay TONR



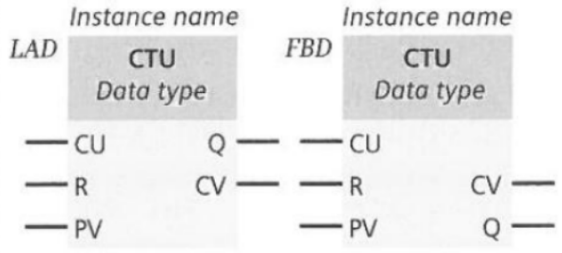
Name	Declaration	Data type	Description
IN	INPUT	BOOL	Start input
R	INPUT	BOOL	Reset input
PT	INPUT	TIME	Defined duration
Q	OUTPUT	BOOL	Time status
ET	OUTPUT	TIME	Current duration

Function



Counters:

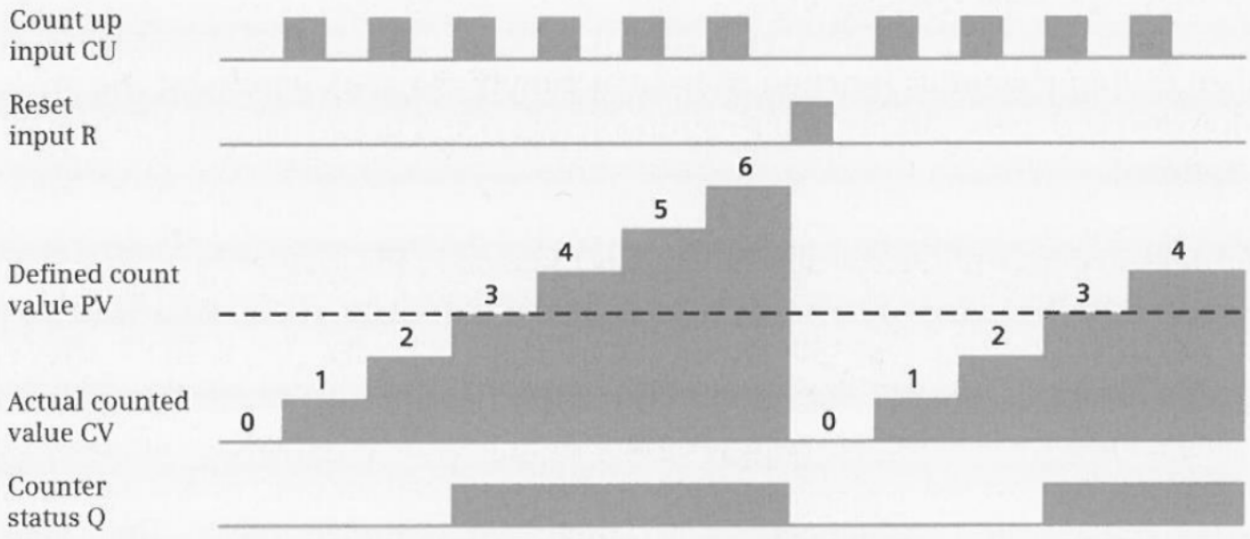
Up counter CTU



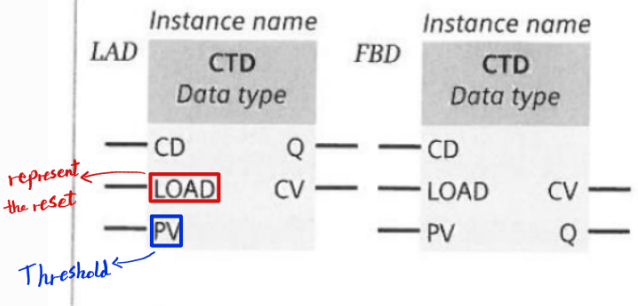
Name	Declaration	Data type	Description
CU	INPUT	BOOL	Count up input
R	INPUT	BOOL	Reset input
PV	INPUT	*Data type* *)	Defined counted value
Q	OUTPUT	BOOL	Counter status
CV	OUTPUT	*Data type* *)	Actual count value

*) USINT, UINT, UDINT, SINT, INT, DINT

Function



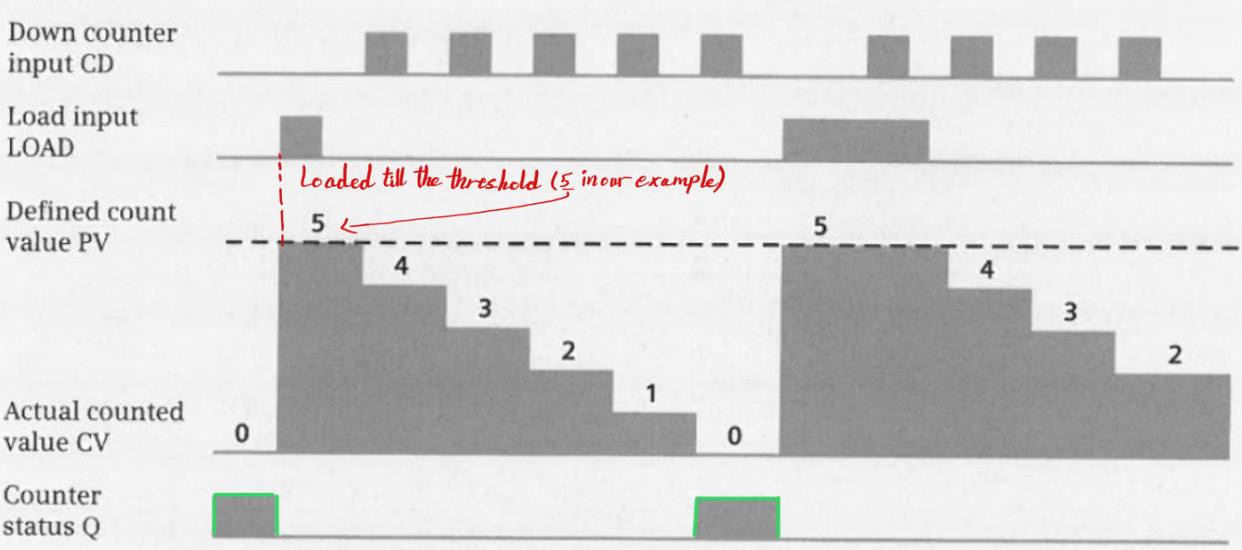
Down counter CTD



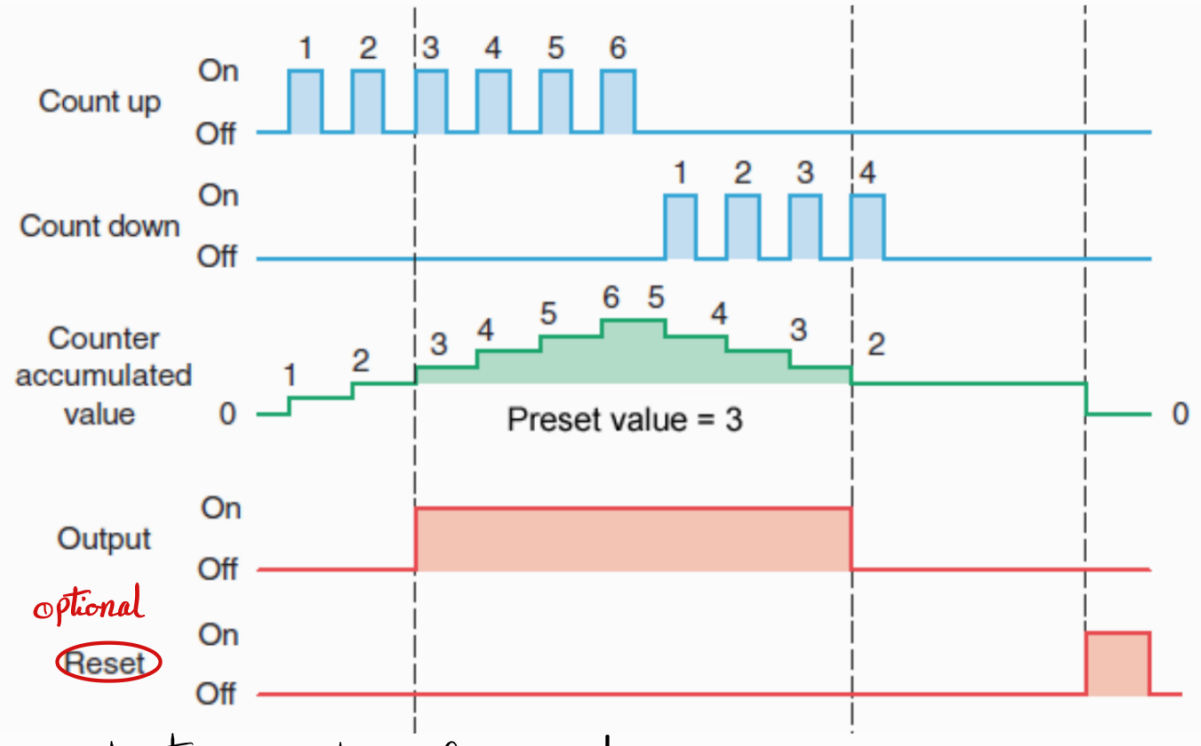
Name	Declaration	Data type	Description
CD	INPUT	BOOL	Count down input
LOAD	INPUT	BOOL	Load input
PV	INPUT	*Data type* *)	Defined count value
Q	OUTPUT	BOOL	Counter status
CV	OUTPUT	*Data type* *)	Actual count value

*) USINT, UINT, UDINT, SINT, INT, DINT

Function



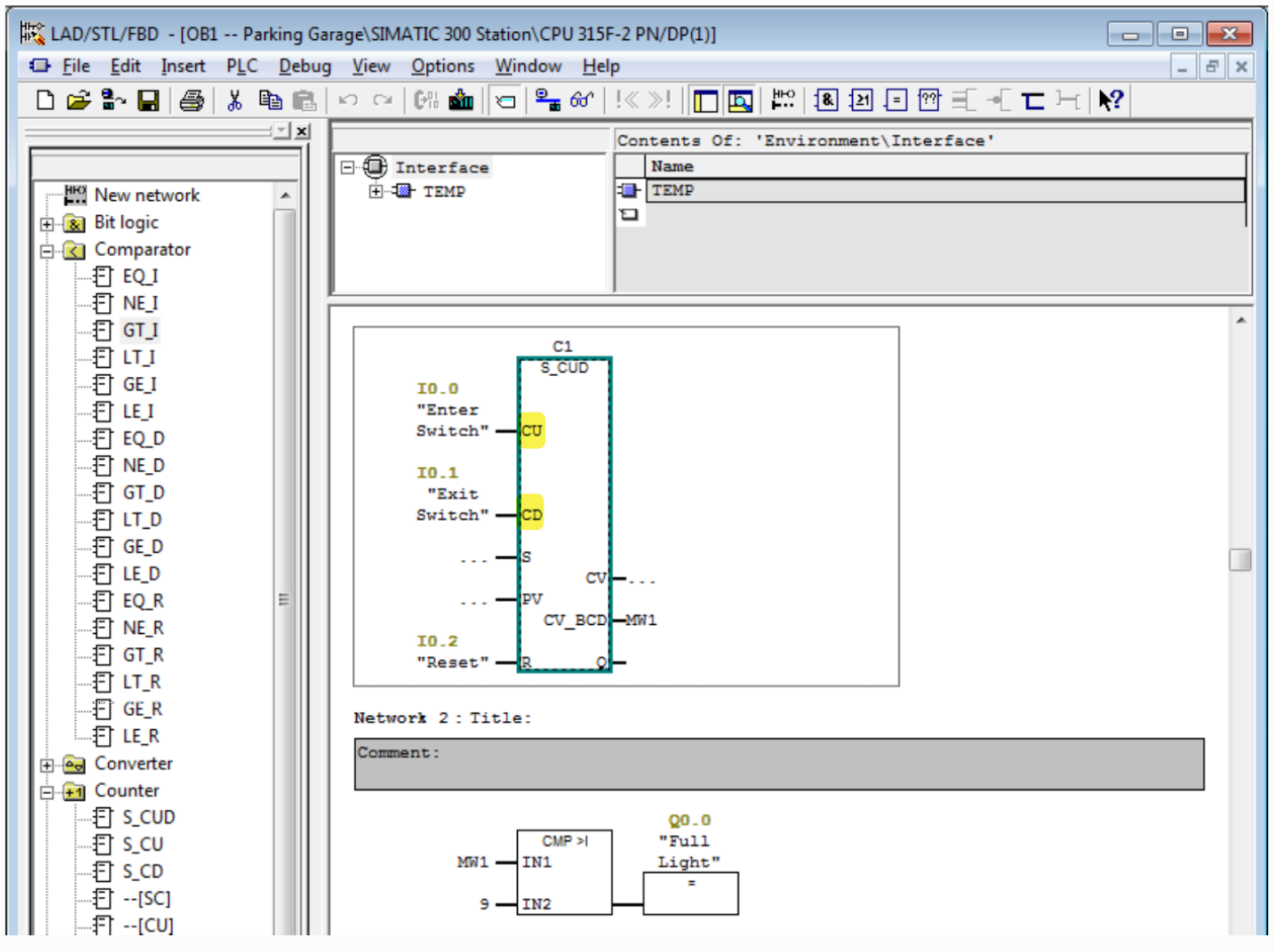
Combined Up/Down Counter : (reset is optional, unlike in single up/down it a must)



Application : parking Garage!

One application for an up/down-counter is to keep count of the cars that enter and leave a parking garage. The operation of a program can be summarized as follows:

- A car enters, the enter switch triggers the up-counter output instruction and increments the accumulated count by 1.
- As a car leaves, the exit switch triggers the down-counter output instruction and decrements the accumulated count by 1.
- Whenever the accumulated value of 150 equals the preset value of 150, a lot full light is activated.
- A reset button has been provided to reset the accumulated count.



Student No.: _____

Points:

Question 4: **16**

A packaging line is automated by use of a PLC. Figure 4.1 shows the hardware arrangement and consists of a NO start button, a NC stop button, a motor 1 to activate the main belt, and a proximity switch detecting the cans going into the packaging station. The specific sequence is as follows, when the start button is pushed.

1. The main belt is activated (by motor M1) and remains active, even when the start button is no longer pressed.
2. A proximity switch detects cans going into the packaging station.
3. Within 20 seconds (after activation according to task 1), 10 cans have to be in the packaging station, detected by the proximity switch. If this is not the case, the belt is shut off and will not be able to start again for another 30 seconds.
4. When 10 cans have passed the proximity switch, the 20 second timer for the next 10 cans is reset and the process continues.
5. The process continues and remains activated as long as the stop button is pressed. In that case the process shall stop immediately.

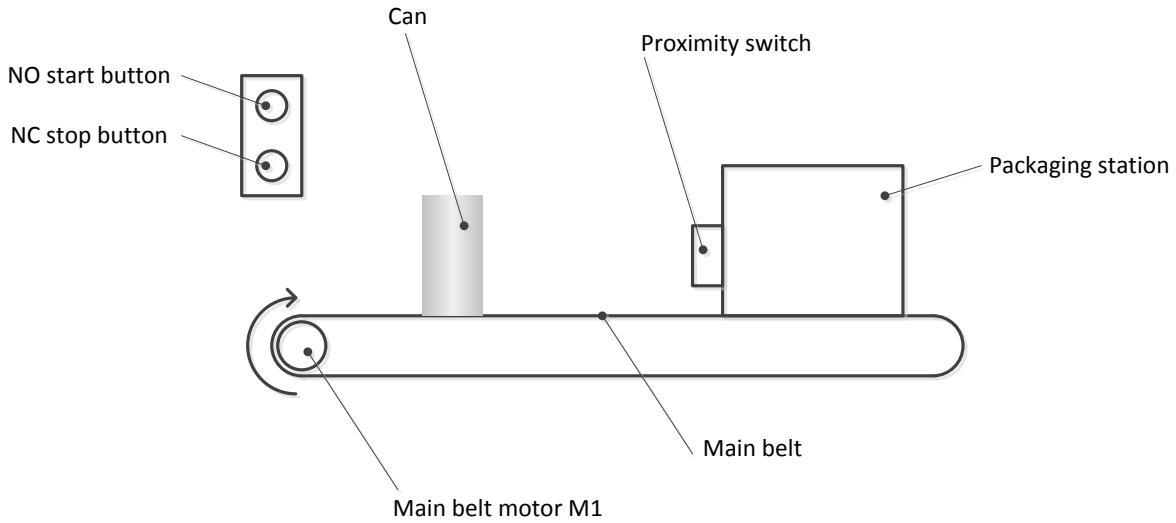


Figure 4.1: Packaging line

Develop and create the logic using a Ladder Diagram (LAD).

Student No.: _____

I1: NO start

I2: NC stop

I3: Proximity