PLC
Programmable Logic Controller

Lecture 1

3rd

Systems and Control Engineering

Lecturer : Yazen Hudhaifa Shakir

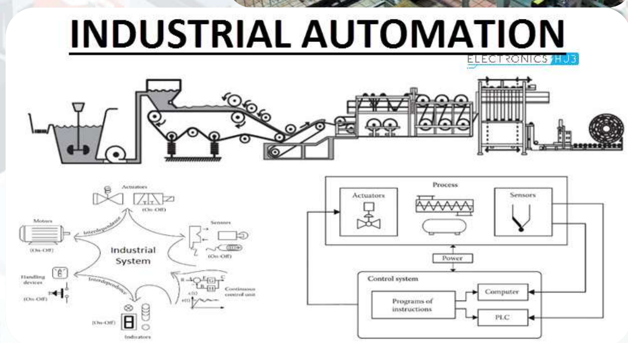
By the end of this Lecture should know:

- What is Automation
- Brief Introduction to Classic Control
- Reading schematic diagrams according (NEMA and IEC standards)
- What is PLC and Its Hardware
- I/Os for PLC
- Types of Signals
- Numbering systems
- PLC Programming Languages

Automation (increase efficiency and productivity at work.)

automation: is the creation and application of technologies to produce and deliver goods and services with minimal human intervention.

Industrial automation: is the use of control systems, such as computers or robots, and information technologies for handling different processes and machineries in an industry to replace a human being.



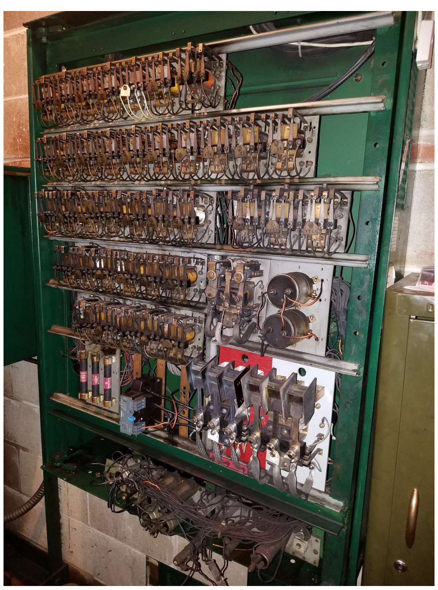
Automation Pyramid

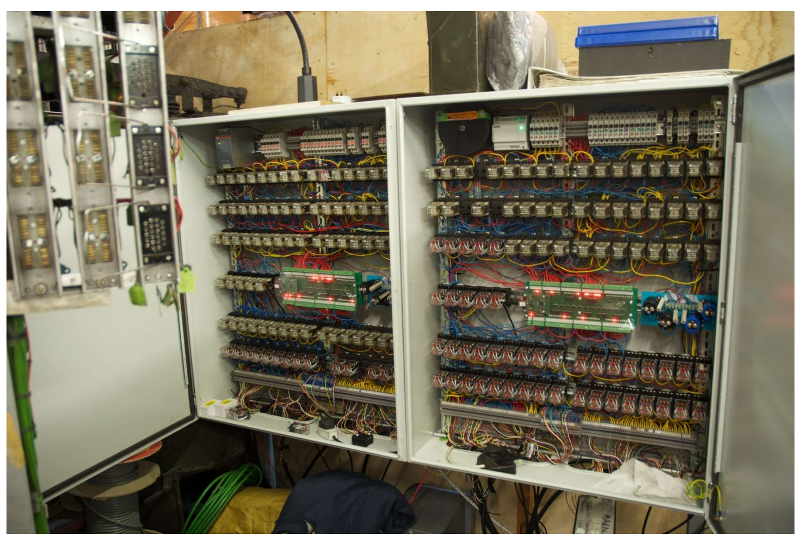
These five levels can classify all the processes through manufacturing from the field to the complete management of a factory or a plant in order to say that this system is automated should go within this criteria accordingly.

Case study (Asphalt Mixing plant)

- some calculations such displaying weight, temperature, total tonnage.
- set cycle (batch) time, gates open close timing, alarms, etc.
- Storing and displaying data as per different parameters.
- display reports

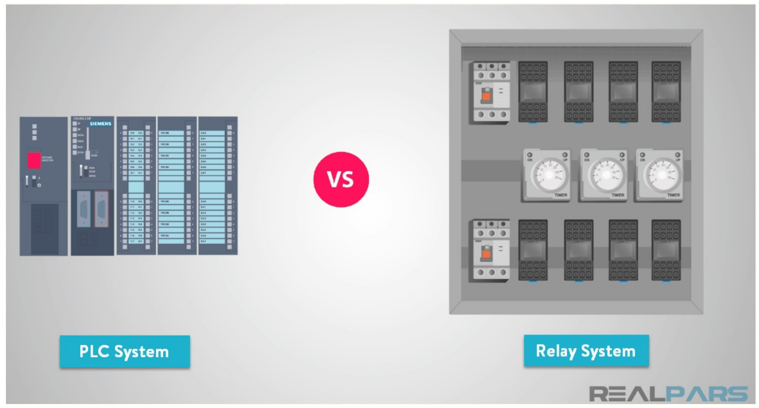
Classic Control (أنظمة التحكم التقليدي)





Classic Control (أنظمة التحكم التقليدي) VS PLC

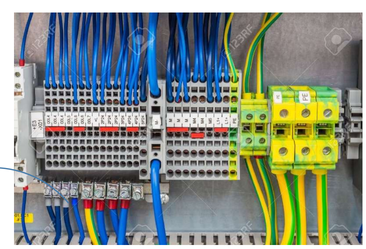
- Use relays , timers , switches and contactors
- Difficult to adaption or modification is expensive
- Large space required for the control circuits
- Consume more power
- Vibrations lead to frequent maintenance for relays
- Noisy
- Need more experience to maintain



Control Panel components

miniature circuit breakers (MCBs)

(POWER CIRCUIT) اولا دائرة القوى

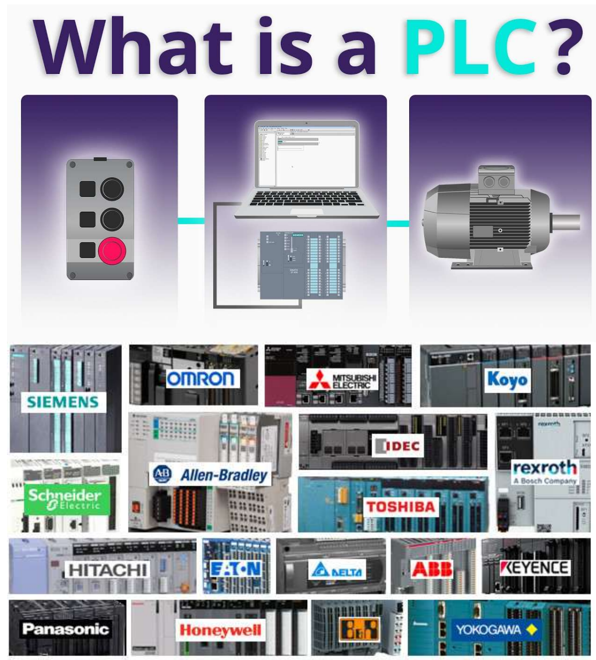


(CONTROL CIRCUIT) ثانيا دائرة التحكم

moulded case circuit breaker (MCCB)

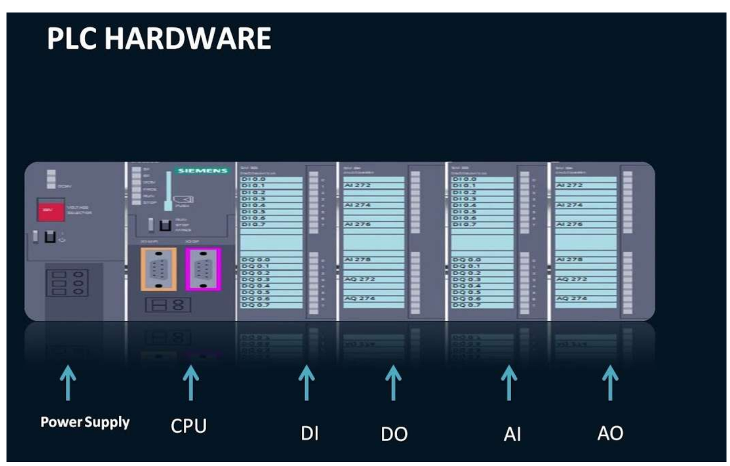
What is PLC

- PLC stands for "Programmable Logic Controller". A PLC is a computer specially designed to operate reliably under harsh industrial environments – such as extreme temperatures, wet, dry, and/or dusty conditions.
- PLCs are used to automate industrial processes such as a manufacturing plant's assembly line, an ore processing plant, or a wastewater treatment plant. (https://www.electrical4u.com/)
- a PLC can perform discrete and continuous functions that a PC cannot do, and a PLC is much better suited to rough industrial environments.



Five common components of the PLC unit

1. Power Supply
2. Processor
3. Rack/Mounting (A rack is used to assemble different components of a PLC at a one place.)
4. Input Assembly (AI,DI)
5. Output Assembly (AO, DO)

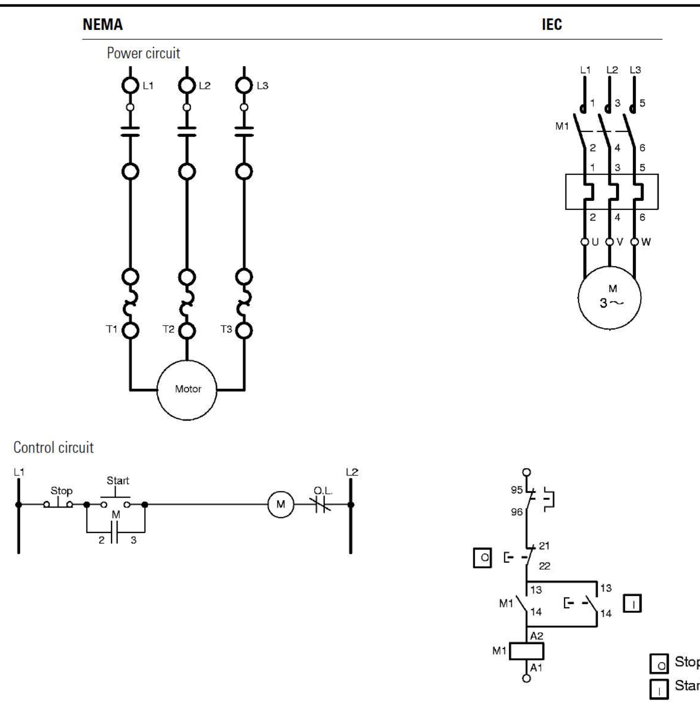


Advantages of PLC

- Advanced technology.
- Less wiring compared with classic control
- Compactness.
- Increased reliability.
- More flexibility.
- Communication capability.
- Faster response time.

NEMA Vs IEC

- There are two primary languages in the world of electric motors: *NEMA in North America, which stands for the National Electrical Manufacturers Association*, and IEC, which stands for the *International Electromechanical Commission* and is used in most of the rest of the world.
- https://www.nema.org/standards
- NEMA and IEC standards use different terms, though they are very similar in ratings, often to the point of being interchangeable.



Signal Of I/O Modules

- Discrete Type
  - I/P Devices
    - Push Buttons
    - Switches
    - Limit Switch
    - Proximity Sensor
    - Photo Sensor
    - Contacts
  - O/P Devices
    - Relays
    - Contactors
    - Lamps
    - Alarms Hooters
    - LED
    - Solenoied Valves
    - Coils
- Analog Type
  - I/P Devices
    - RTD
    - Thermocouple
    - Transducers
    - Transmitter
  - O/P Devices
    - Analog Ports
    - Drive
    - Acuters
    - Valve
    - Positioner

http://plctraining1.blogspot.in/

I/O for PLC

- Digital Inputs:

The input to the PLC normally will be 24 VDC or might be 110 V AC or 220 AC in condition that the signal voltage compatible with input module.

Digital inputs are binary inputs (0 or 1) that are applied to the PLC. Binary inputs are basically voltages, varying from 5V to 230V depending upon the type of card used. In simple words, any push-button, switches or sensors produces digital inputs to a PLC

I/O for PLC

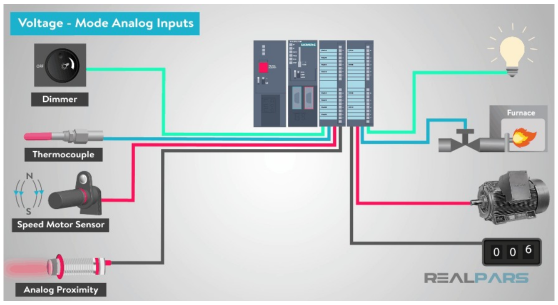
- Analog Inputs:

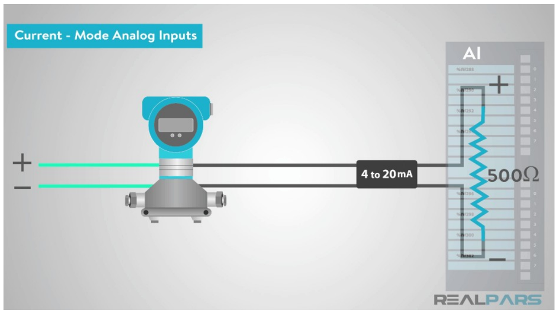
The Analog Input Module (AIN) is a key subsystem in the PLC. AINs come in many variations to condition real world physical parameters, such as, Temperature, Pressure, Force, or Strain. Typically, these AIN inputs are command signals in both voltage (e.g. ±10V , 0-10V or 0-5V) and current form (e.g. 4-20mA).

It can be divided into:

a) Voltage-Mode Analog Inputs

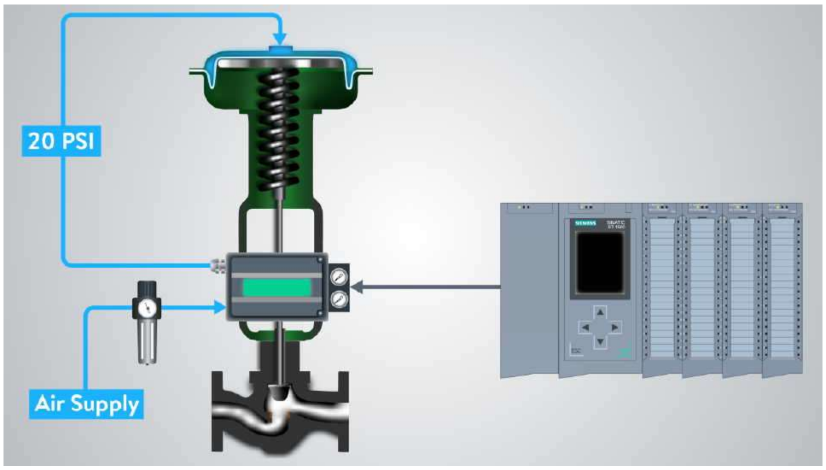
b) Current-Mode Analog Inputs





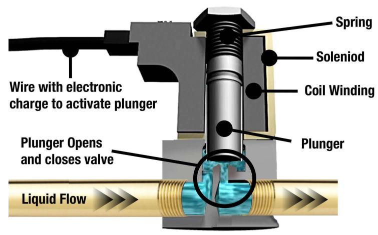
I/O for PLC

- PLC-system analog outputs—commonly used to control actuators, valves, and motors in industrial environments—employ standard analog output ranges such as **±5 V, ±10 V, 0 V to 5 V, 0 V to 10 V, 4 to 20 mA, or 0 to 20 mA**. ... The first uses a dedicated DAC in each channel to generate its analog control voltage or current.



I/O for PLC (Digital Outputs)

- Digital outputs are **binary outputs (0 or 1) from the PLC**. It is a processed control output from the PLC to the field. It is used to ON or OFF any piece of field equipment. DO is like a contact of a relay when the pre-programmed conditions are satisfied the contacts are closed.

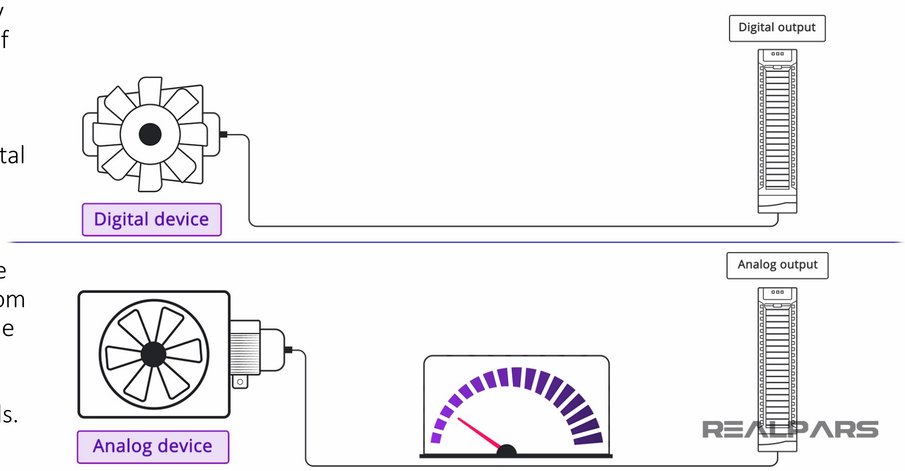


Parts of a Solenoid Valve

Analog devices are a bit more complicated because they have an infinite number of possible conditions.

To illustrate, a fan is a digital device if it can be turned either on or off.

But, what if the fan can be adjusted for any speed from Off to On? In that case, the fan is an analog device because it has an infinite number of possible speeds.

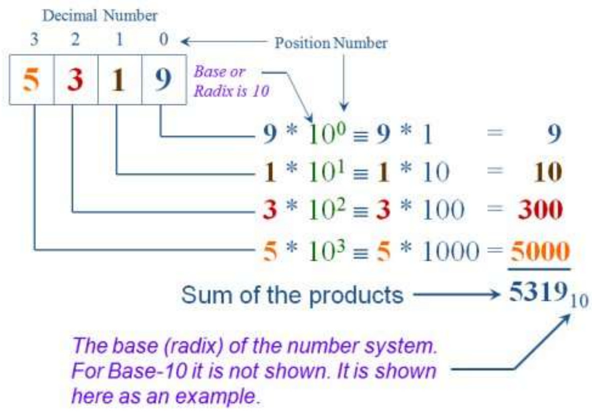


Numbers in PLC (decimal number system)

- The decimal number system is the number system we use everyday, from counting to simple math like checking store recepts.
- The word 'deci' means 10, therefore there are ten numbers (digits) in the decimal number system. The valid numbers in a base 10 number system are:

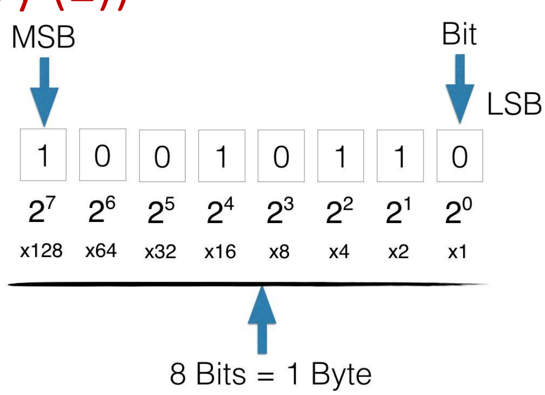
0, 1, 2, 3, 4, 5, 6, 7, 8, 9

- All number systems have a 'base'. The base is the same as the number of valid numbers in the system. Since the decimal number system has 10 valid numbers it is a 'base 10' number system. The base is also referred to as a 'radix' and in the PLC/PAC software the user can change the radix (base) of the numbers for display purposes.
- A decimal number showing its base would be written: 257_{10} or 257d
- Where the small letter 'd' designates that it it a decimal number. We normally do not write the base on decimal numbers. If the base is not displayed the number is assumed to be a decimal number.



Numbers in PLC (Binary (2))

- Computers, including PLCs, use the Base 2 numbering system called Binary or Boolean. There are only two valid digits in Base 2: 0 and 1 (OFF and ON).
- Each digit in the Base 2 system, when referenced by a computer, is called a bit. When four bits are grouped together, they form what is known as a nibble.
- Eight bits—or two nibbles—is a byte. Sixteen bits – or two bytes – is a word (Table 1). Thirty-two bits—or two words—is a double-word.



Word															
Byte								Byte							
Nibble				Nibble				Nibble				Nibble			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 1

Binary/Decimal																
Bit #	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Power	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
Decimal Bit Value	32678	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1
Max Value	65535_{10}															

Table 2

$11011_2 = \_\_\_\_\_\_\_\_ {}_{10}$

WeTeach_CS

Hexadecimal Numbers (16)

- As you have probably noticed, the Binary numbering system is not very easy to interpret.
- For a few bits, it is easy, but larger numbers tend to take up a lot of room when writing them down and it is difficult to keep track of the bit position while doing the conversion.
- That is where using an alternate numbering system can be an advantage. One of the first numbering systems used was Hexadecimal, or Hex for short.

Decimal	Hex
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7

Table 3

Decimal	Hex
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

Table 3

Octal Numbers (8)

- The Octal numbering system is similar to the Hexadecimal system in the interpretation of the bits (Table 5). The difference is the maximum value for Octal is 7, since it is a Base 8.
- For example, 63_8 is equal to 51_{10} or $(6*8^1 + 3*8^0)$ or $(48_{10} + 3_{10})$.

Octal																
Bit #	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Power	8^5	8^4			8^3			8^2			8^1			8^0		
Decimal Bit Value	1	4	2	1	4	2	1	4	2	1	4	2	1	4	2	1
Max Value	1	7			7			7			7			7		

Table 5

BCD Numbers

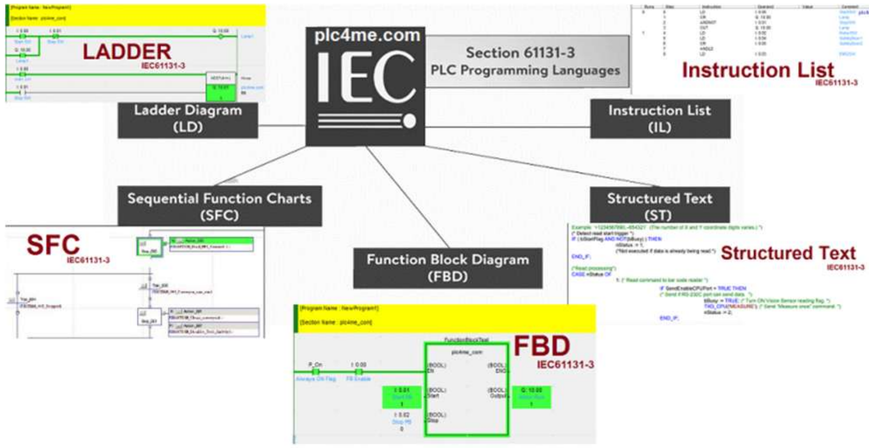
- The BCD numbering system, like Octal and Hexadecimal, relies on bit-coded data (Table 6). It is Base 10 (Decimal), but it is Binary Coded Decimal. There is a big difference between BCD and Binary, as we will see later.
- One plus of BCD coding is that it reads like a Decimal number, whereas 867 BCD would mean 867 Decimal. No conversion is needed. However, as with all things computer related, there are snags to worry about

BCD																
Bit #	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Power	10^3				10^2				10^1				10^0			
Decimal Bit Value	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1
Max Value	9				9				9				9			

Table 6

PLC Programming Languages

- Ladder Diagram (LD)
- Sequential Function Charts (SFC)
- Function Block Diagram (FBD)
- Structured Text (ST)
- Instruction List (IL)



#Basic Ladder Logic & I/O Mapping

Lecture 2

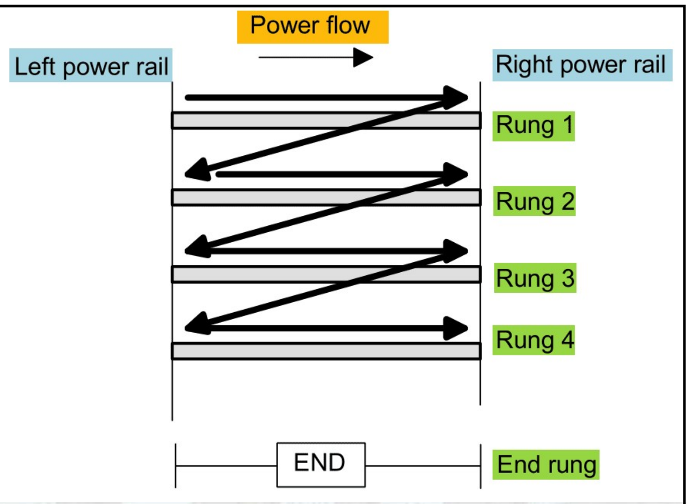
3rd

Systems and Control Engineering

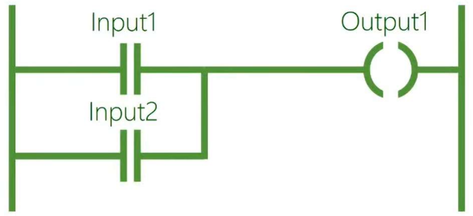
Lecturer : Yazen Hudhaifa Shakir

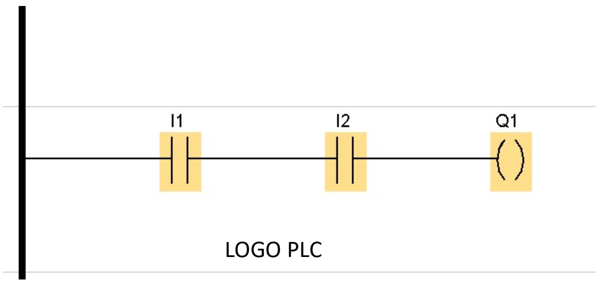
What is a Ladder Diagram

- Ladder diagrams are also called line diagrams or elementary diagrams.
- Ladder diagrams are used to represent the function of the control circuit and associated devices but they don't show the components of the control circuit and their actual physical positions.
- As control circuit becomes more complex while a ladder diagram can be less complicated to read than a wiring or connection diagram.
- Connection diagrams or wiring diagrams shows the components of the control circuit in a resemblance of their actual physical location.

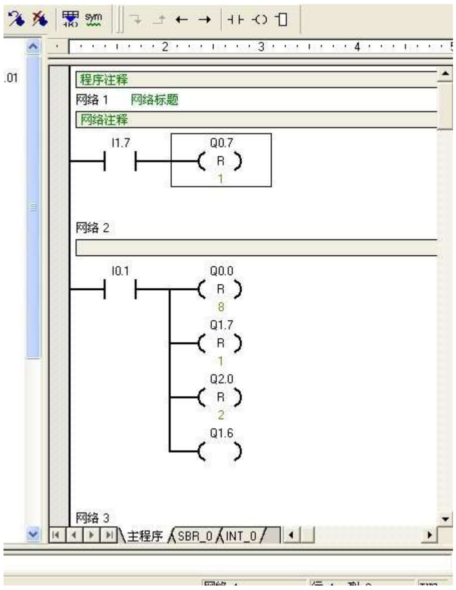


Examples





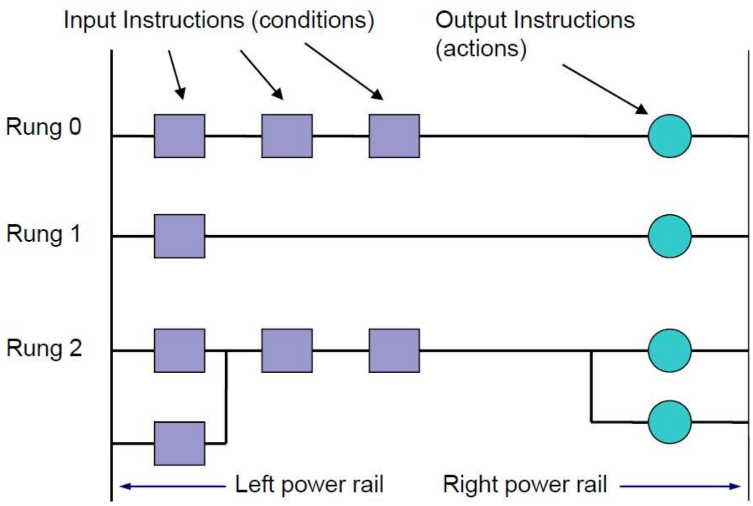
LOGO PLC



S7-200 PLC

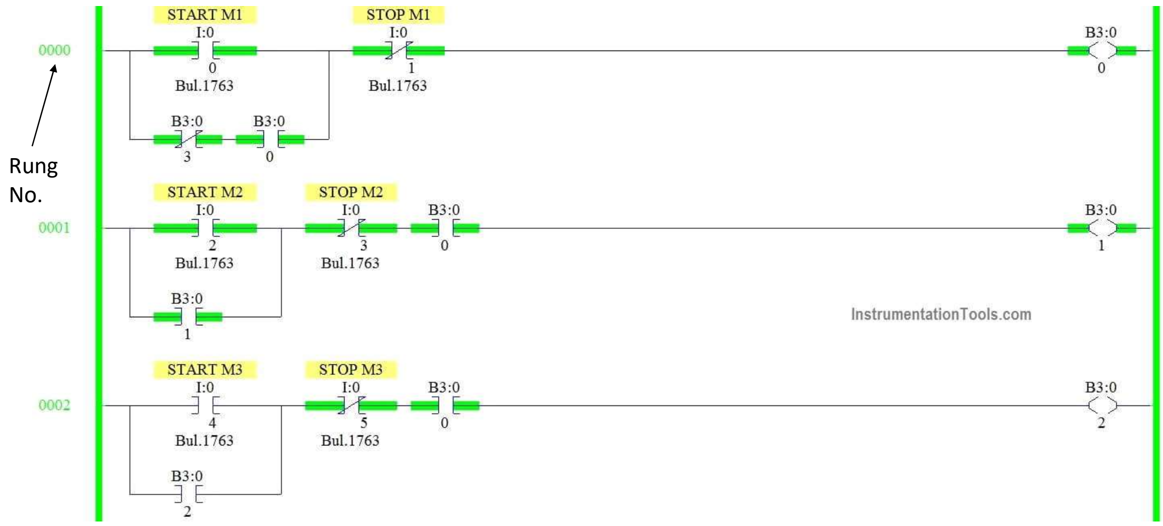
Anatomy of a Ladder Program

- Input instructions are entered on the left
- Output instructions are entered on the right
- The power rails simulate the power supply lines
- L1 and L2 for AC circuits and +24v and ground for DC circuits
- Most PLCs allow more than one output per rung



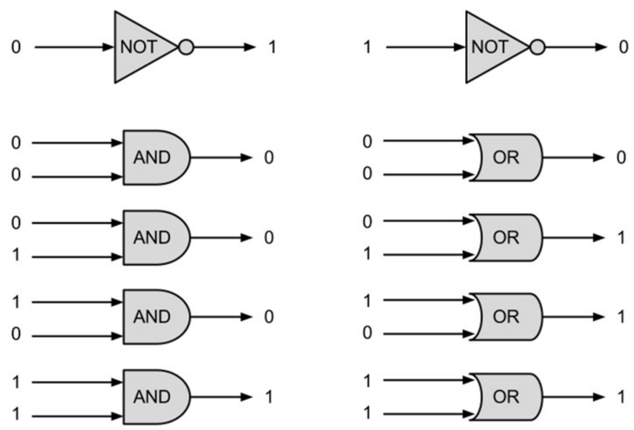
<u>Note: The basic sequence is altered whenever jump or subroutine instructions are executed.</u>

A 3-rung example ladder program



Logic Functions

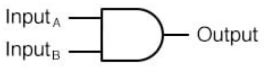
- PLC programming is a logical procedure
- In a PLC program, “things” (inputs and rungs) are either TRUE or FALSE
- If the proper input conditions are TRUE:
- The rung becomes TRUE and an output action occurs(for example, a motor turns on)
- If the proper input conditions are not TRUE: The rung becomes FALSE and an output action doesn't occur



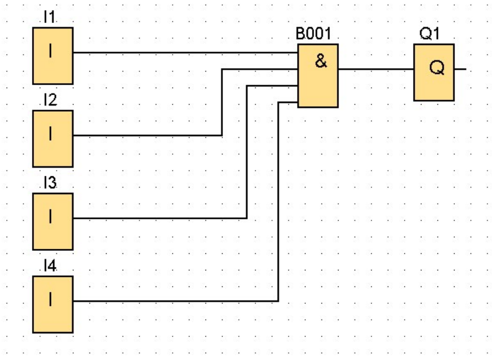
AND Gate (Logic Function)

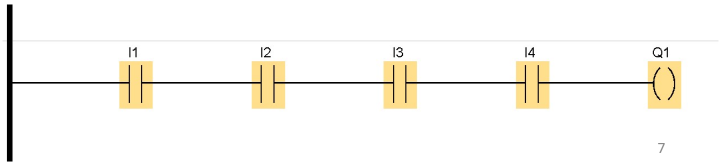
- TRUE (1)- switch ON
- FALSE (0)- switch off
- In order to get an output all the inputs related must be on state or (1)

2 - input AND gate

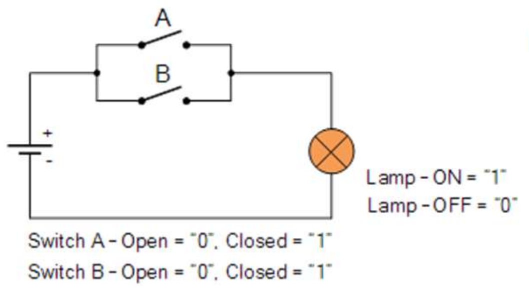


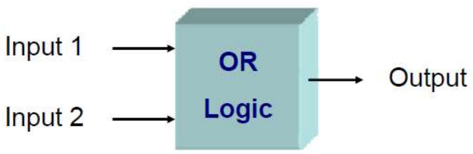
A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1





OR Gate (Logic Function)

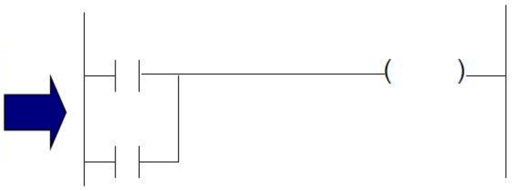




Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	1

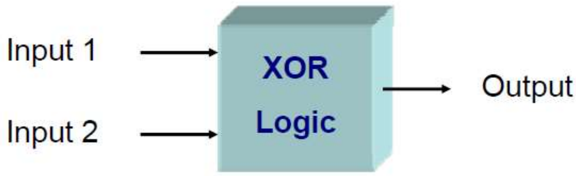
0 → False
1 → True

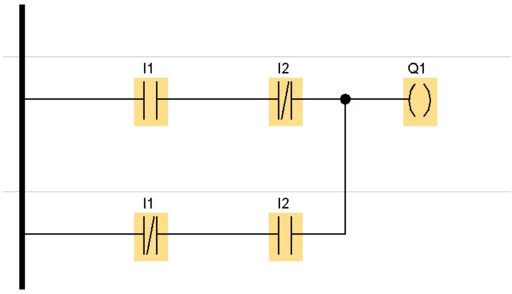
Contacts ORed together



Logic Functions - Exclusive OR

- In addition to ANDing and ORing, the Exclusive OR (XOR) is also useful.
- When the inputs are DIFFERENT, the XOR output is true



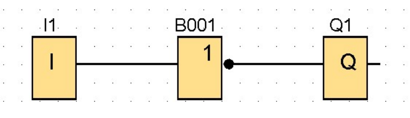


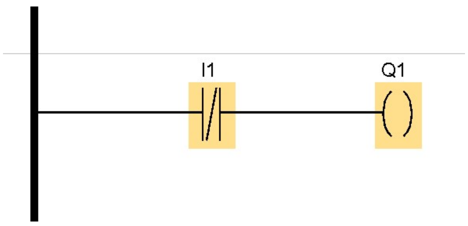
Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	0

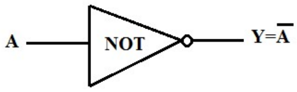
0 → False
1 → True

NOT Gate Implementation

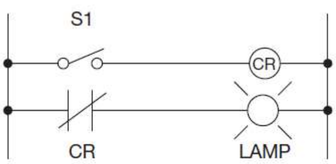
- N.C. CR Contact Controlling a Lamp with NOT logic







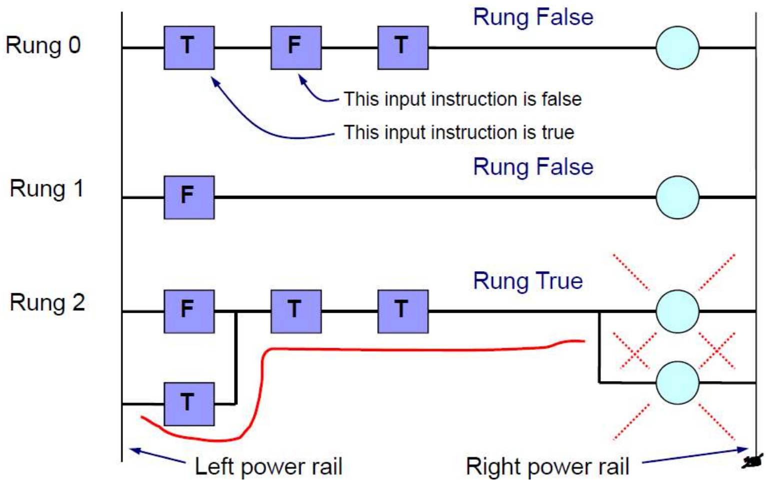
Input	Output
A	$Y=\bar{A}$
0	1
1	0



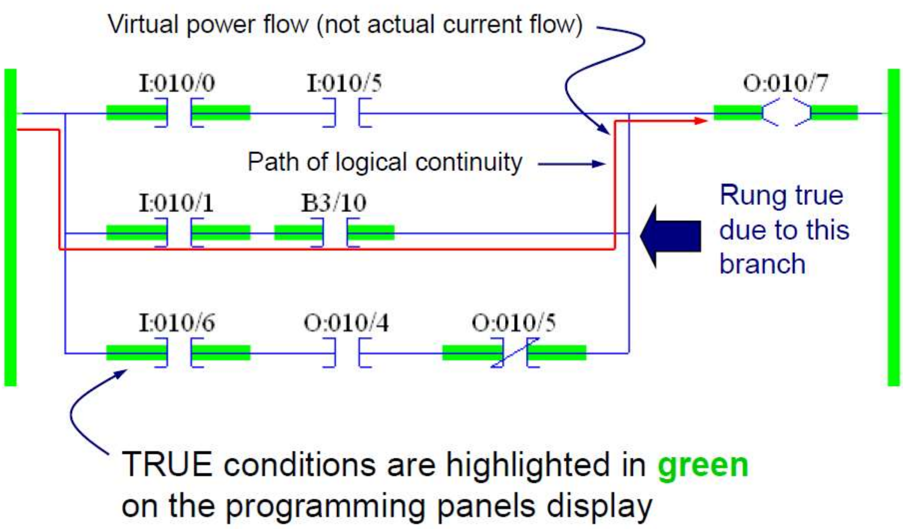
CR	LAMP
0	**1**
1	**0**

Logical Continuity – Example 1

- **Logical continuity** in a ladder rung occurs when there is a continuous path of TRUE conditions from the left power rail to the output instruction(s)
- When there is logical continuity, the rung becomes true and the output becomes energized

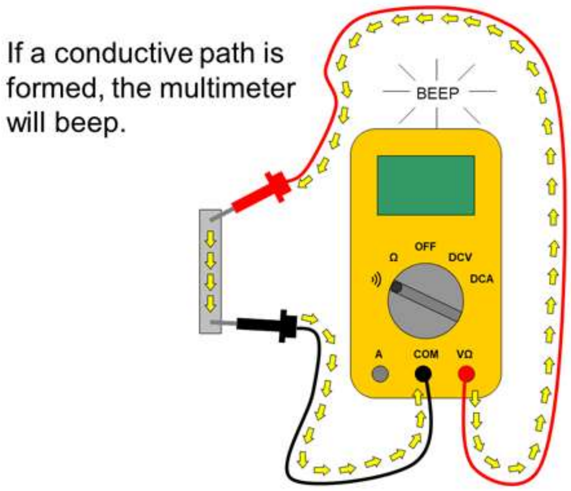


Logical Continuity – Example 2



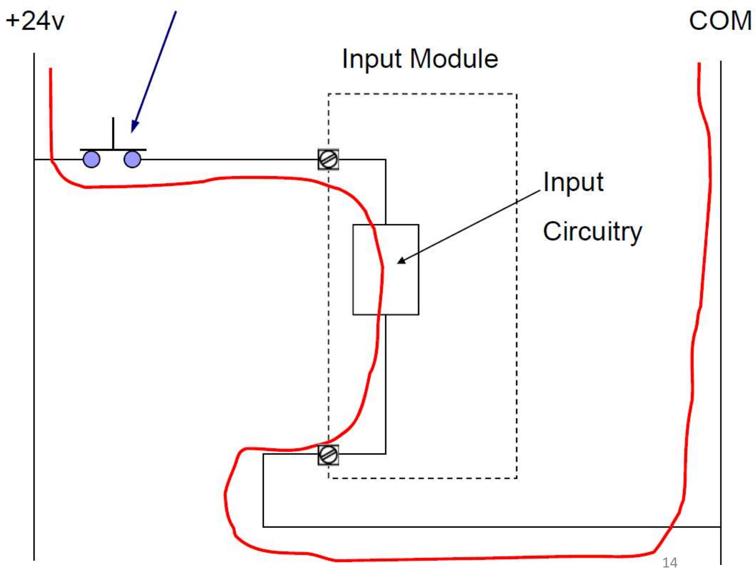
Logical Continuity vs. Electrical Continuity

- **Electrical continuity** in an input circuit, occurs when there is a complete path for current to flow.
- A PLC input circuit is a simple series circuit consisting of a:
  - Power supply,
  - Switch, and a
  - Load
- When there is electrical continuity, a bit in the PLCs memory (sometimes called the input image table) is set to a 1.

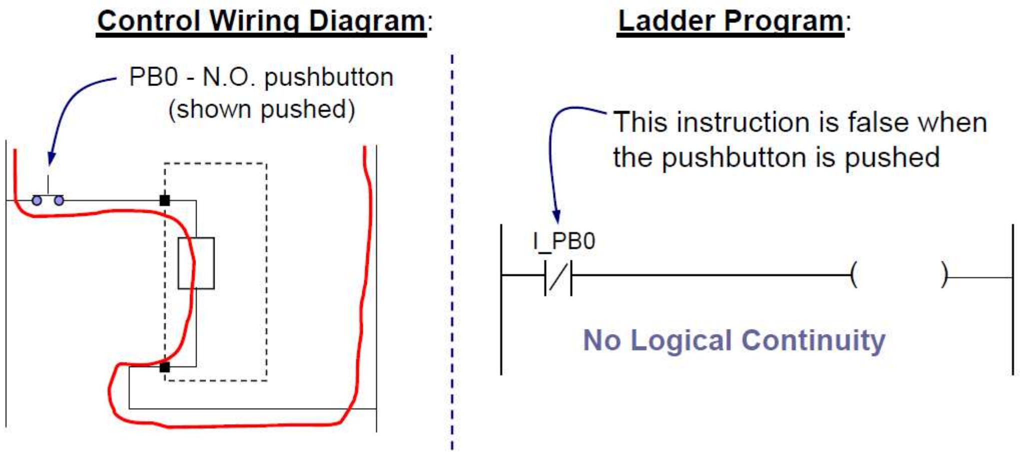


Electrical Continuity

- When pressing push button electricity flows from +24 V side to COM as indicated in red colour.
- *(Normally open (N.O.) pushbutton [shown pushed]*

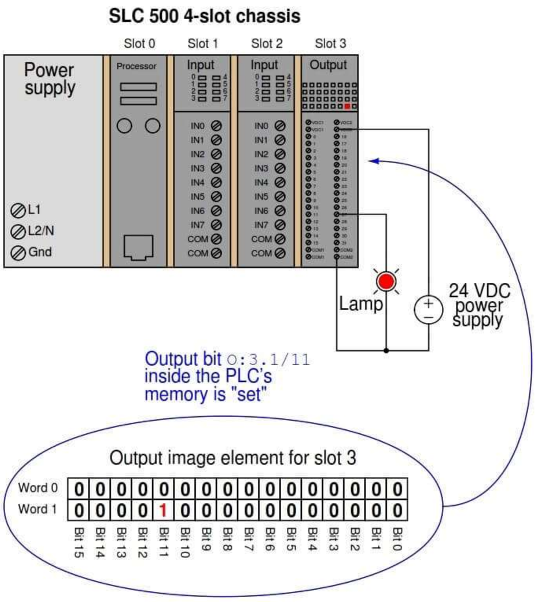


Logical vs. Electrical Continuity



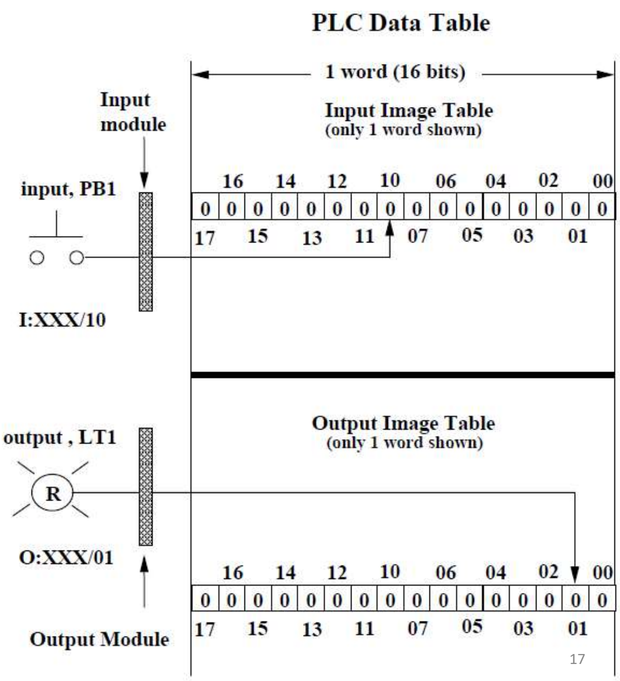
I/O Mapping

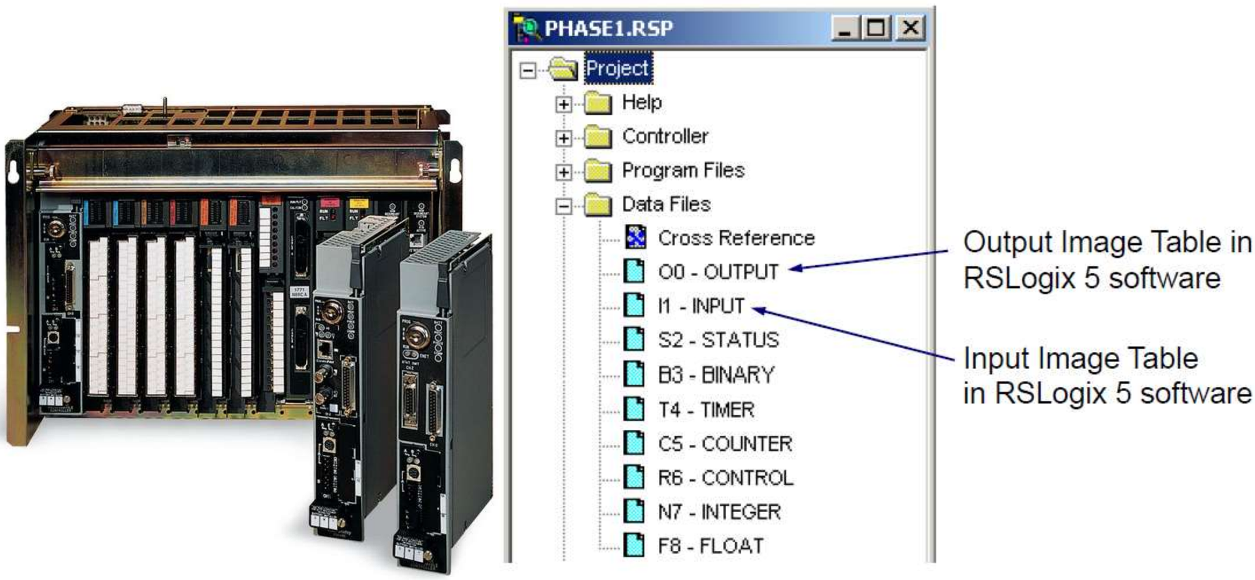
- Every discrete input is assigned to a specific bit in the PLC's memory (**input image table**)
  - If there is electrical continuity, the bit is set to a 1
  - If there is no electrical continuity, the bit is reset to a 0
- Every discrete output is assigned to a specific bit in the PLC's memory (**output image table**)
  - In order for an output to turn on, its associated bit must first be set to a 1



I/O Mapping

- **Note**: Bit addresses are given in octal for an Allen-Bradley PLC-5 system



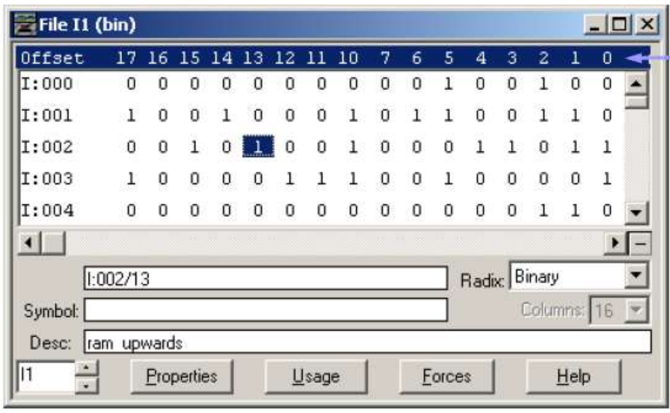


→ *PLC-5 Systems*

I/O Mapping

Input Image Table in RSLogix 5:

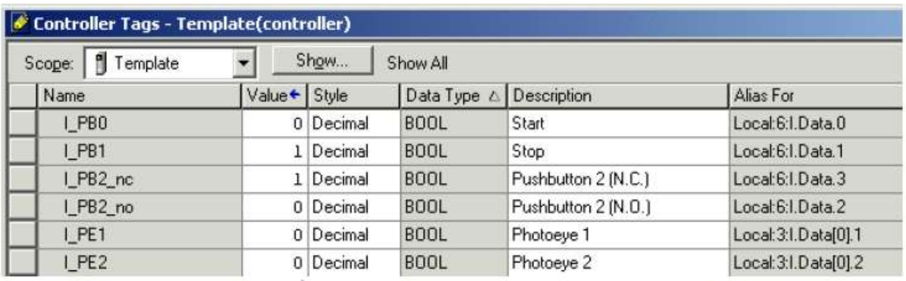
1 word = 16 bits (bits are numbered in octal for a PLC-5)



Offset	17	16	15	14	13	12	11	10	7	6	5	4	3	2	1	0
I:000	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0
I:001	1	0	0	1	0	0	0	1	0	1	1	0	0	1	1	0
I:002	0	0	1	0	1	0	0	1	0	0	0	1	1	0	1	1
I:003	1	0	0	0	0	1	1	1	0	0	1	0	0	0	0	1
I:004	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

I/O Mapping

ControlLogix tag database:



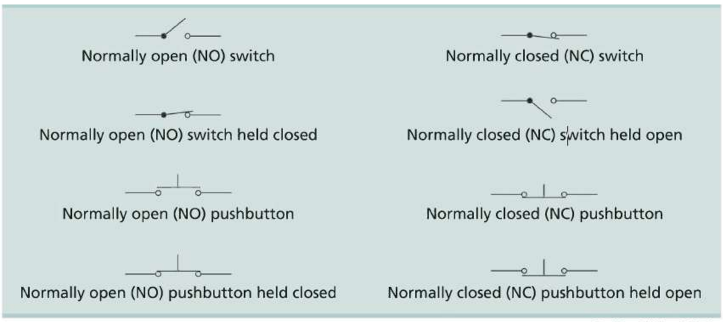
Name	Value	Style	Data Type	Description	Alias For
I_PB0	0	Decimal	BOOL	Start	Local:6:I.Data.0
I_PB1	1	Decimal	BOOL	Stop	Local:6:I.Data.1
I_PB2_nc	1	Decimal	BOOL	Pushbutton 2 (N.C.)	Local:6:I.Data.3
I_PB2_no	0	Decimal	BOOL	Pushbutton 2 (N.O.)	Local:6:I.Data.2
I_PE1	0	Decimal	BOOL	Photoeye 1	Local:3:I.Data[0].1
I_PE2	0	Decimal	BOOL	Photoeye 2	Local:3:I.Data[0].2

Alias tag (a pointer to a base address)

Current tag value

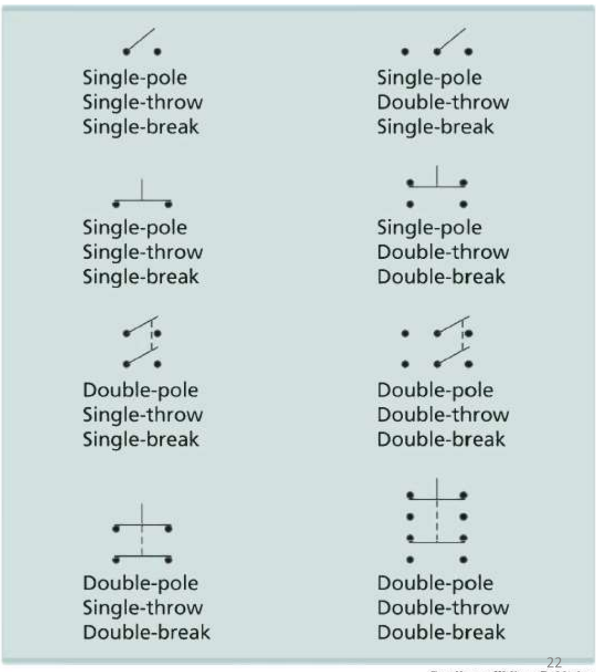
Base address (real address)

Pushbuttons and Switches (discrete input devices,)



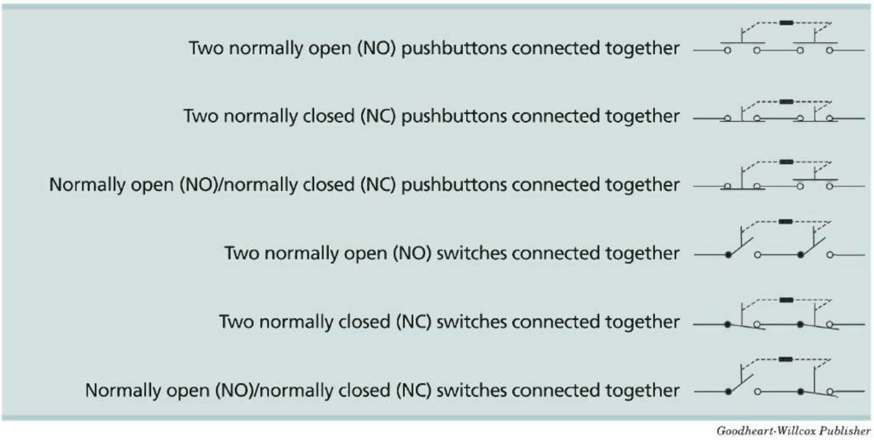
‘’The two general types of pushbuttons and switches available in industry are normally open (NO) and normally closed (NC). In their natural states, ***normally open (NO) devices*** are open—they do not allow current to flow through them. However, they can be forced closed so that current can flow through them. On the contrary, ***normally closed (NC) devices*** are closed in their natural state—they allow current to flow from one terminal to the other. However, they can be forced open to stop the flow of current.’’

Electrical symbols and descriptions of various switches with one or more poles and throws.

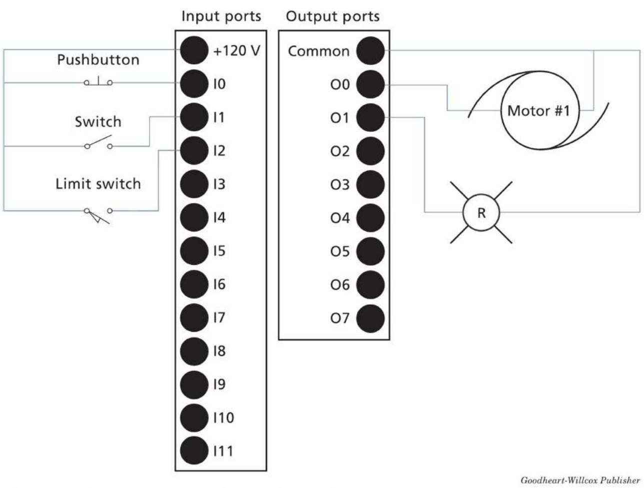


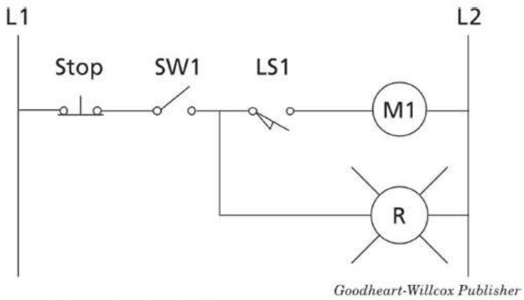
Goodheart-Willcox Publisher

The dashed lines (-------) in these electrical symbols indicate that the input devices change state at the same time.



Example 1 : Programming Ladder Logic Diagrams with Normally Open and Normally Closed Switches



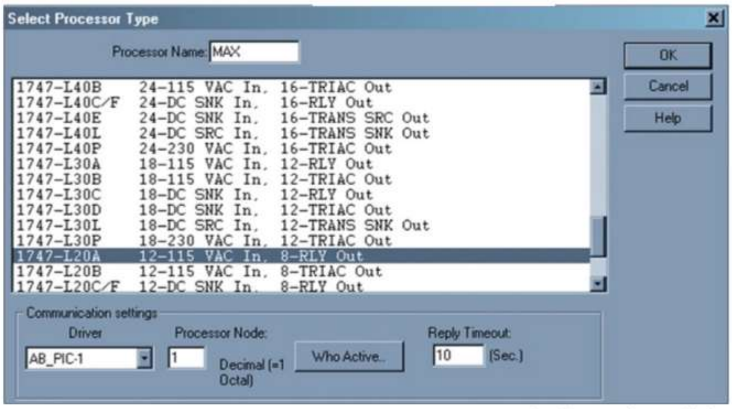


Relay Logic Diagram

Input Ports	Output Ports
NC stop pushbutton=I:0/0	Motor#1=O:0/0
NO start switch =I:0/1	Red light=O:0/1
Limit switch=I:0/2	

I/O mapping

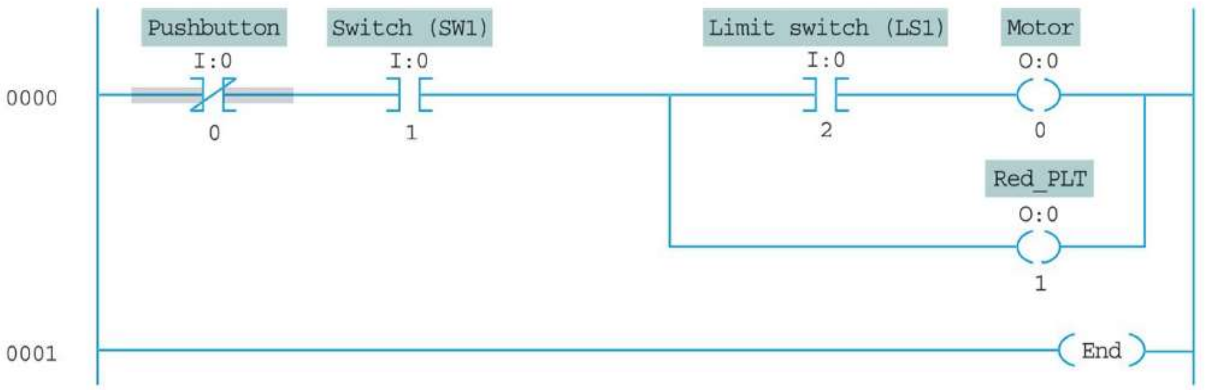
Select PLC Type with its configuration



Goodheart-Willcox Publisher

PLC type 1747-L20A for the Allen-Bradley fixed PLC with twelve 120 VAC inputs and *eight 120 VAC outputs*.

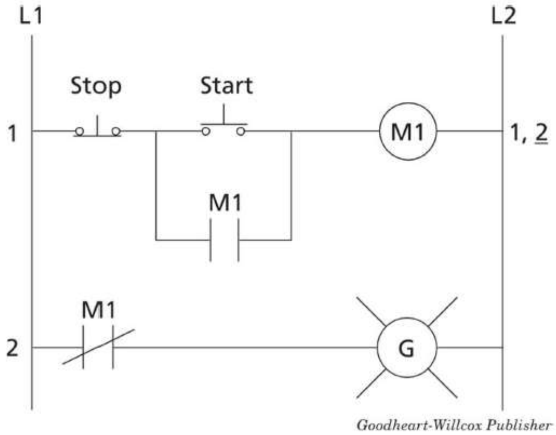
Ladder Diagram PLC type 1747-L20A for the previous example



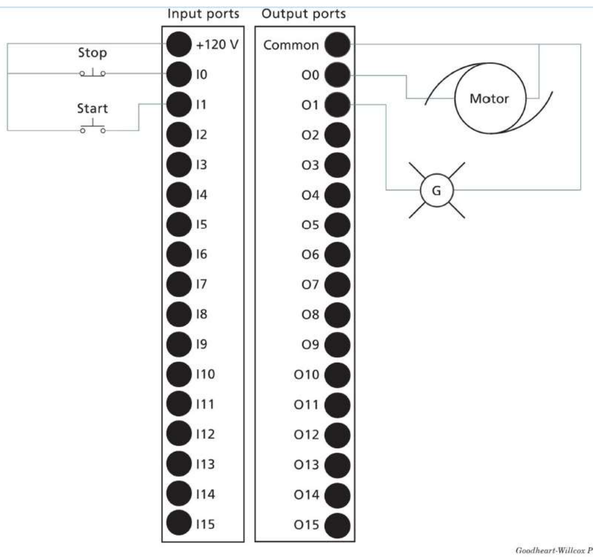
Goodheart-Willcox Pu

Example 2: Programming Ladder Logic Diagrams with Normally Open and Normally Closed Contacts

The modular Allen-Bradley SLC 5/03 PLC will be used to create the PLC ladder logic diagram for the relay logic diagram



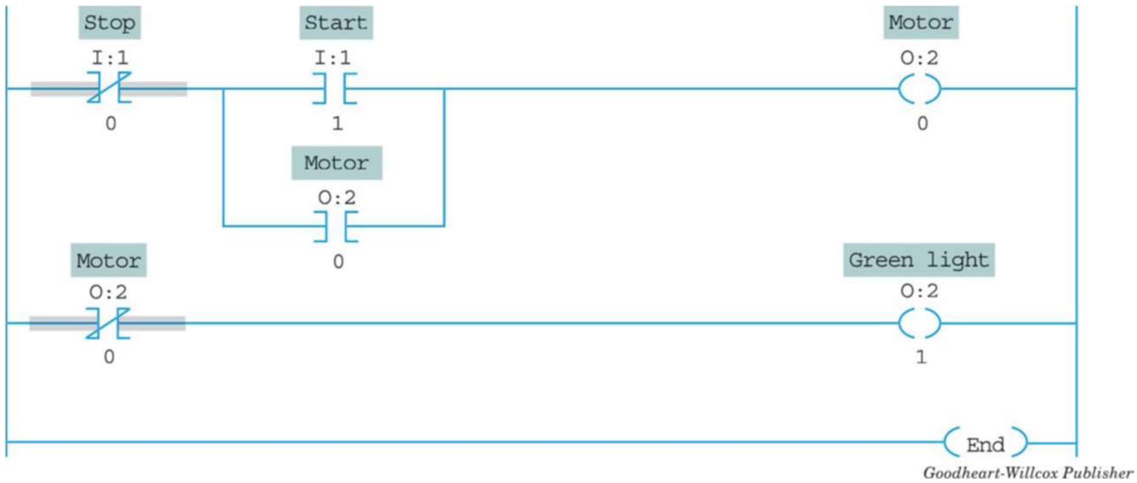
Ex2: Wiring configuration



Input Ports	Output Ports
Stop pushbutton=I:1/0	Motor#1=O:2/0
Start pushbutton =I:1/1	Green light=O:2/1

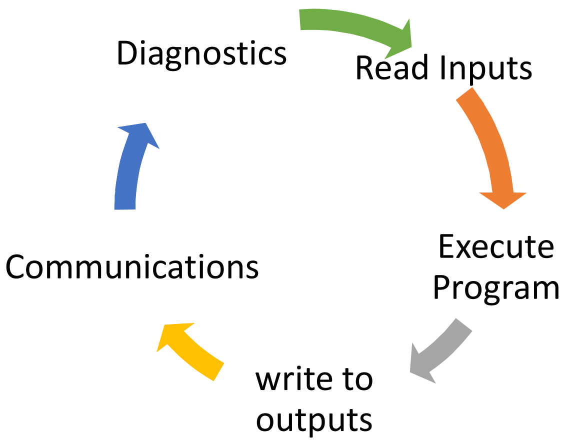
I/O addressing

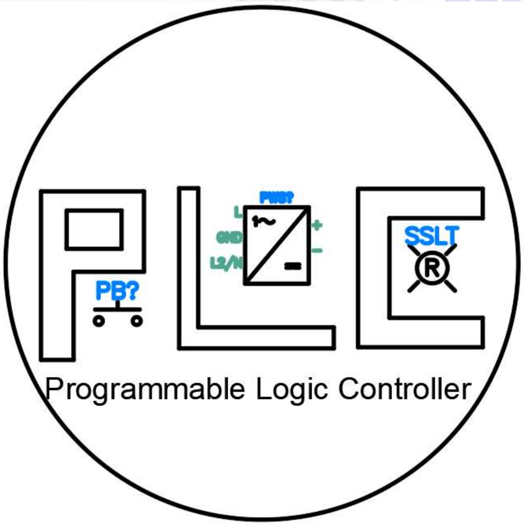
Ladder Diagram Programming for previous example



PLC Program Scan Time

- **Input scan:** The processor reads the input ports and updates the input status table.
- **Program scan:** The processor executes the PLC program and updates the output status table.
- **Output scan:** Output status table values are transferred to the output terminals.





#Discrete SENSORs WIRING, Actuators & Latches

Lecture 3

3rd

Systems and Control Engineering

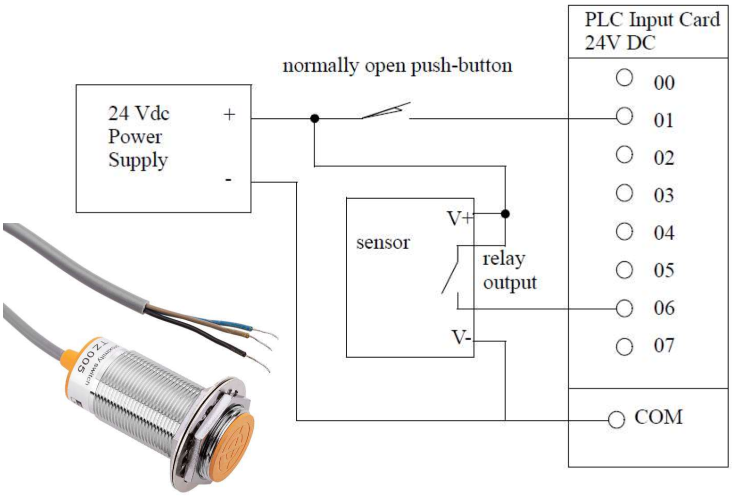
Lecturer : Yazen Hudhaifa Shakir

Typical outputs from sensors

- Sinking/Sourcing - Switches current on or off.
- Plain Switches - Switches voltage on or off.
- Solid State Relays - These switch AC outputs.
- TTL (Transistor Transistor Logic) - Uses 0V and 5V to indicate logic levels.

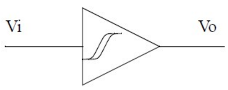
1- Simple Switch and Relay to PLC

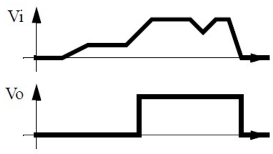
- In the figure a NO contact switch is connected to input *01.*
- A sensor with a relay output is also shown.
- The internal switch (probably a relay) will be closed allowing current to flow and the positive voltage will be applied to input *06*.
- For example, Proximity sensor has three wires (two for power and one is an input signal to PLC)

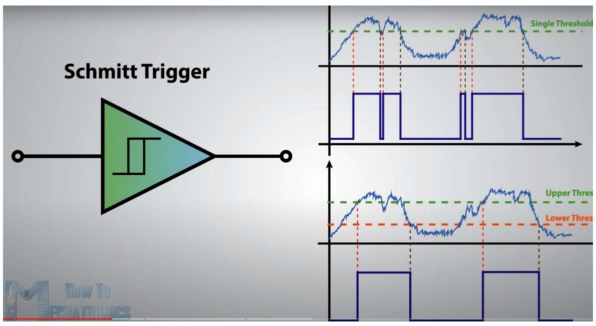


2- Transistor Transistor Logic (TTL)

- The range are 0V for false and 5V for true. The voltages can actually be slightly larger than 0V, or lower than 5V and still be detected correctly.
- TTL outputs are common on electronic devices and computers, and will be necessary sometimes. When connecting to other devices simple circuits can be used to improve the signal, such as the Schmitt trigger.
- If a sensor has a TTL output the PLC must use a TTL input card to read the values. If the TTL sensor is being used for other applications it should be noted that the maximum current output is normally about 20mA.

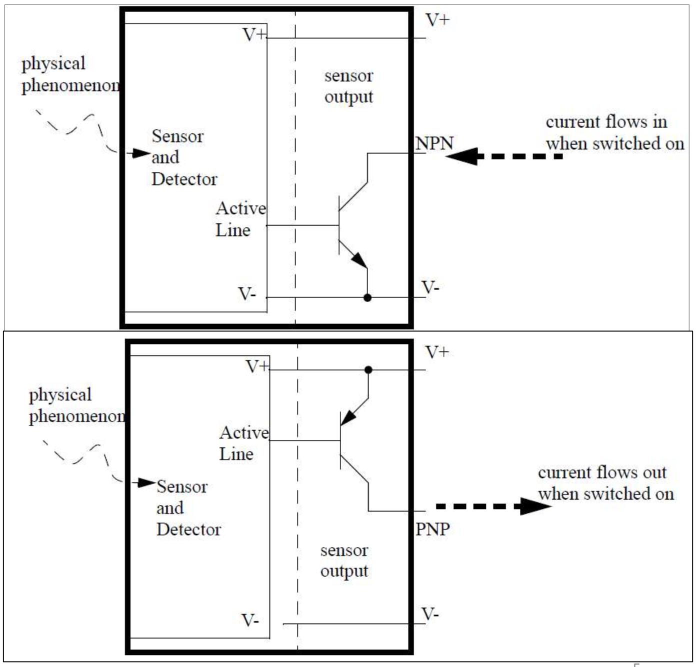






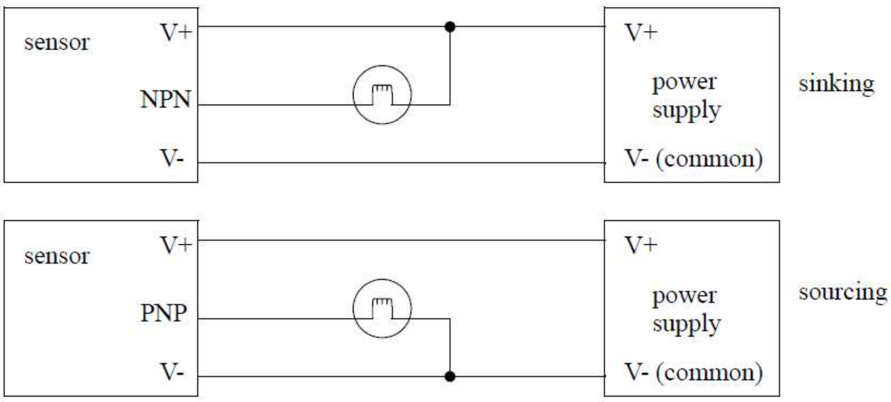
3- Sinking/Sourcing

- **Sinking sensors** allow current to flow into the sensor to the voltage common, while
- **Sourcing sensors** allow current to flow out of the sensor from a positive source.
- *Note: Therefore, sinking and sourcing terminology applies **only to DC input and output circuits.***
- ***https://www.youtube.com/watch?v=Rout56r1n0I***

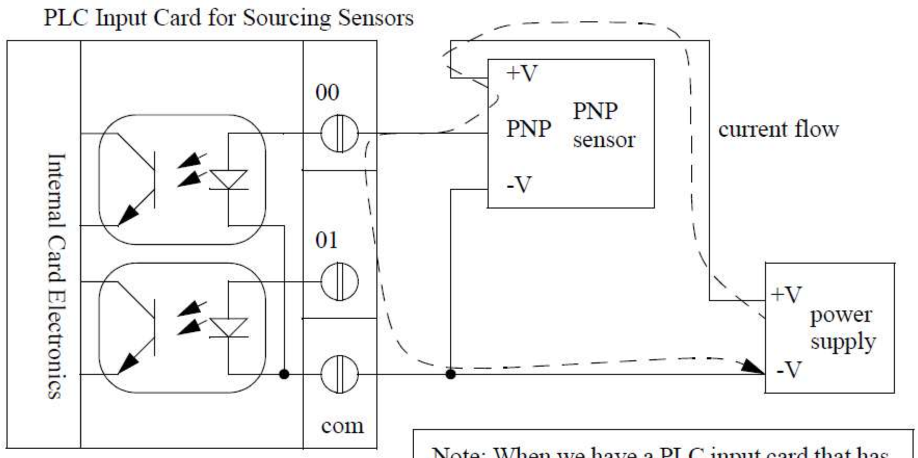


Direct Control Using NPN/PNP Sensors

check the current and voltage ratings for the sensors.

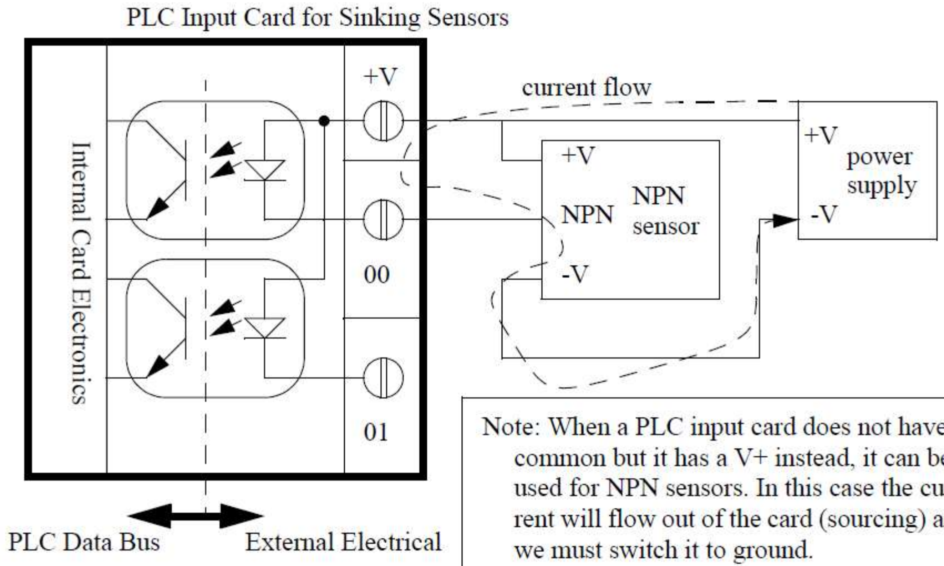


Three Wires Sourcing Sensors



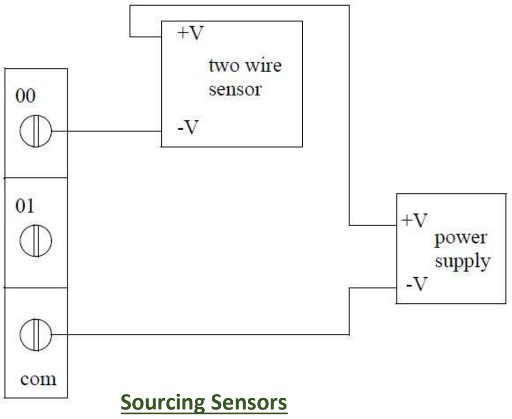
Note: When we have a PLC input card that has a common then we can use PNP sensors. In this case the current will flow into the card and then out the common to the power supply.

Three Wires Sinking Sensors

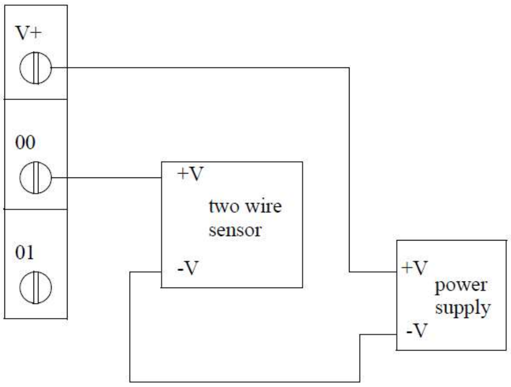


Note: When a PLC input card does not have a common but it has a V+ instead, it can be used for NPN sensors. In this case the current will flow out of the card (sourcing) and we must switch it to ground.

Two Wires Sensors (Sinking and Sourcing)



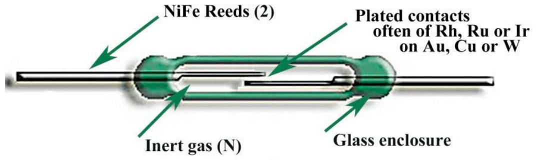
Sourcing Sensors

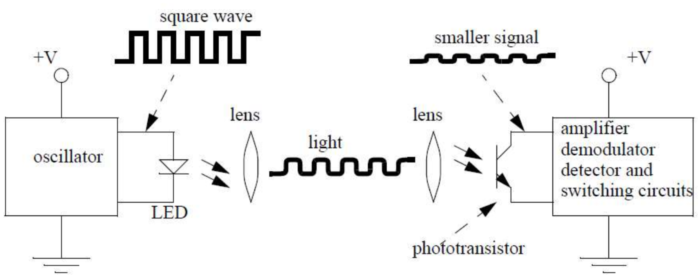


Sinking Sensors

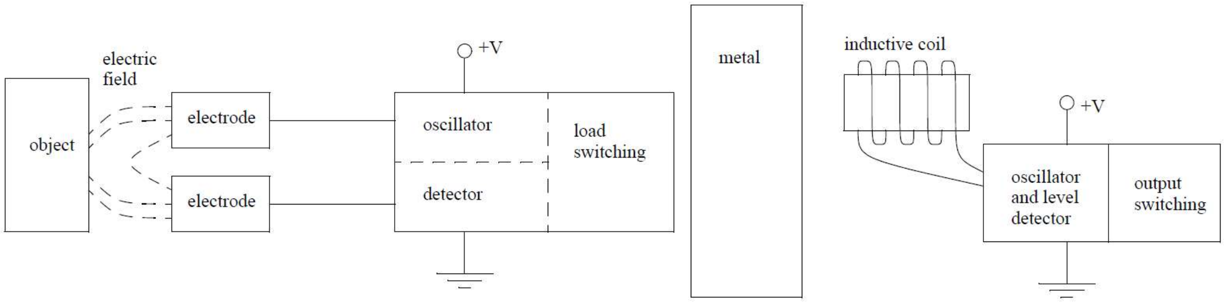
3- Solid state relays

1. Contact Switches
2. Reed Switches
3. Optical (Photoelectric) Sensors
4. Capacitive Sensors (Capacitive sensors are able to detect most materials at distances up to a few centimetres.)
5. Inductive Sensors
6. **Ultrasonic**
7. **Hall effect**





Capacitive Vs Inductive sensors



Capacitive sensors   Inductive sensors

Proximity sensors detect the presence or absence of objects using electromagnetic fields, light, and sound. There are many types, each suited to specific applications and environments.

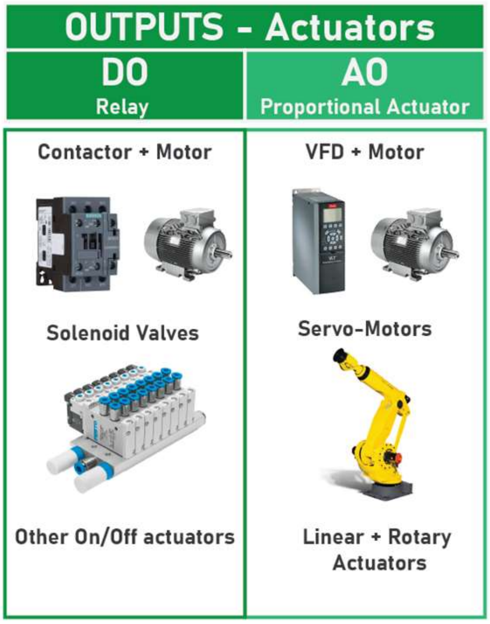
Proximity sensor comparison

Technology	Sensing range	Applications	Target materials
Inductive	<4-40 mm	Any close-range detection of ferrous material	Iron Steel Aluminum Copper etc.
Capacitative	<3-60 mm	Close-range detection of non-ferrous material	Liquids Wood Granulates Plastic Glass etc.
Photoelectric	<1mm- 60 m	Long-range, smalll or large target detection	Silicon Plastic Paper Metal etc.
Ultrasonic	<30 mm- 3 m	Long-range detection of targets with difficult surface properites. Color/reflectivity insensitive.	Cellophane Foam Glass Liquid Powder etc.

LOGICAL ACTUATORS

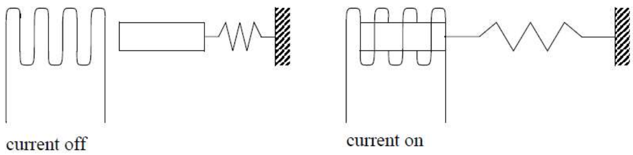
- Solenoids, valves and cylinders
- Hydraulics and pneumatics
- Other actuators

- Actuators Drive motions in mechanical systems. Most often this is by converting electrical energy into some form of mechanical motion.



SOLENOIDS

- Solenoids are the most common actuator components.
- The basic principle of operation is there is a moving ferrous core (a piston) that will move inside wire coil as shown



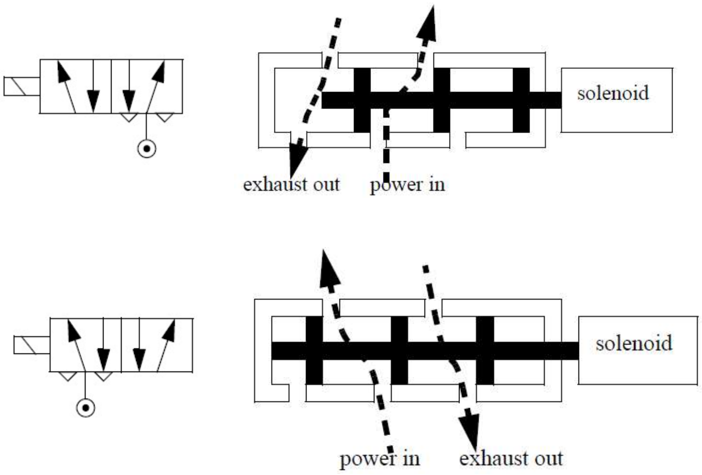
Normally Close type Solenoid Valve

When Solenoid valve OFF

When Solenoid valve ON

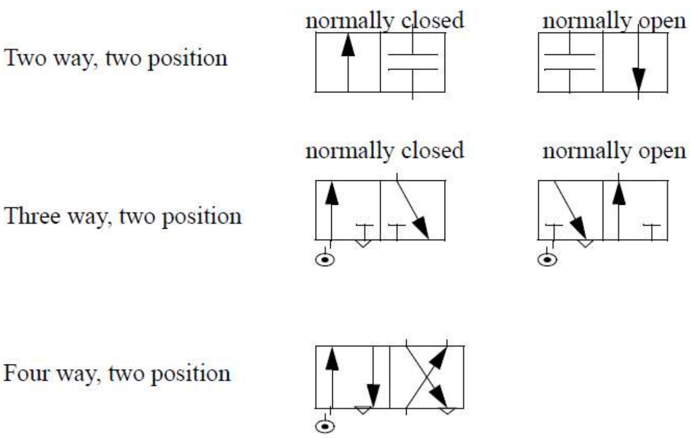
VALVES

- The flow of fluids and air can be controlled with solenoid controlled valves. An example of a solenoid controlled valve is shown.
- *A Solenoid Controlled 5 Ported, 4 Way 2 Position Valve*



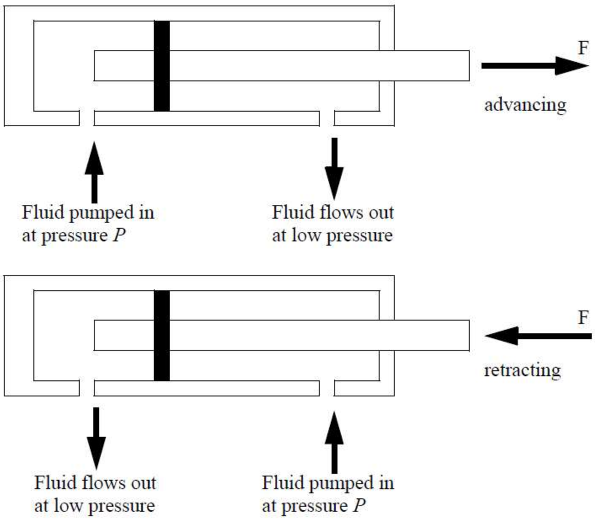
Some of ISO Valves Symbols

- 2-way normally open - these have one inlet, and one outlet. When un-energized, the valve is open, allowing flow. When energized, the valve will close. These are used to stop flows. When system power is off, flow will be allowed.
- 3-way normally closed - these have inlet, outlet, and exhaust ports. When un-energized, the outlet port is connected to the exhaust port. When energized, the inlet is connected to the outlet port. These are used for single acting cylinders.



CYLINDERS

- P = the pressure of the hydraulic fluid
- A = the area of the piston
- F = the force available from the piston rod



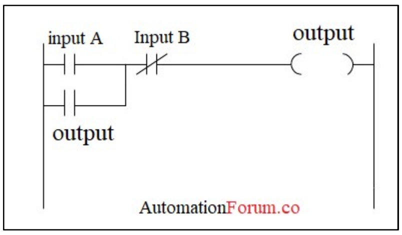
- Single acting cylinders apply force when extending and typically use a spring to retract the cylinder.
- Double acting cylinders apply force in both direction.

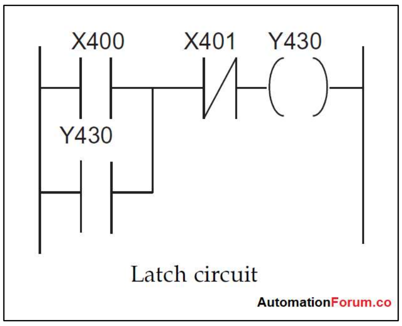
LATCHES

More complex systems means more control sequences. Thus, the combinational logic cannot control the process or we must increase the sensors to detect all conditions

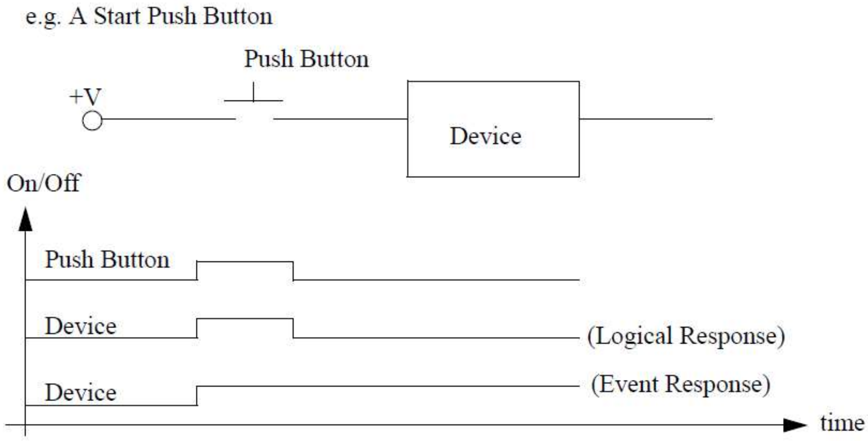
- In these cases we can use events to estimate the condition of the system. Typical events used by a PLC include:

- *first scan of the PLC - indicating the PLC has just been turned on*
- *time since an input turned on/off - a delay*
- *count of events - to wait until set number of events have occurred*
- *latch on or unlatch - to lock something on or turn it off*



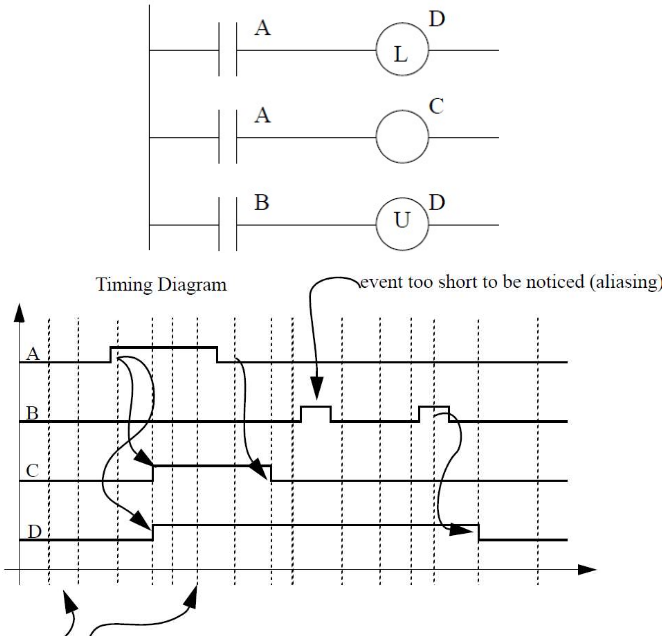


The device is event based if it can respond to one or more things that have happened before. If the device responds only one way to the immediate set of inputs, it is logical.

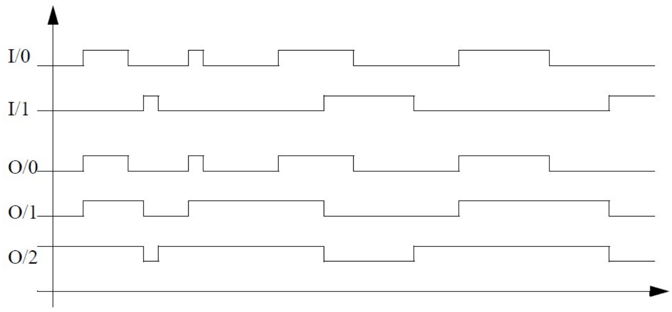


LATCHES

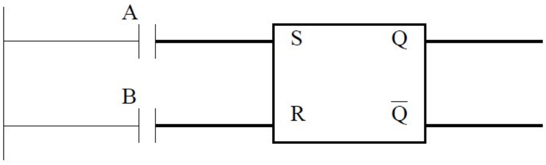
- These lines indicate PLC input/output refresh times.
- The space between the lines is the scan time for the ladder logic.
- The spaces may vary if different parts of the ladder diagram are executed each time through the ladder (as with state space code)

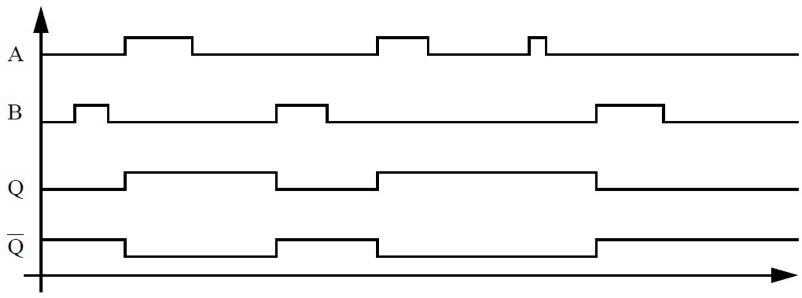


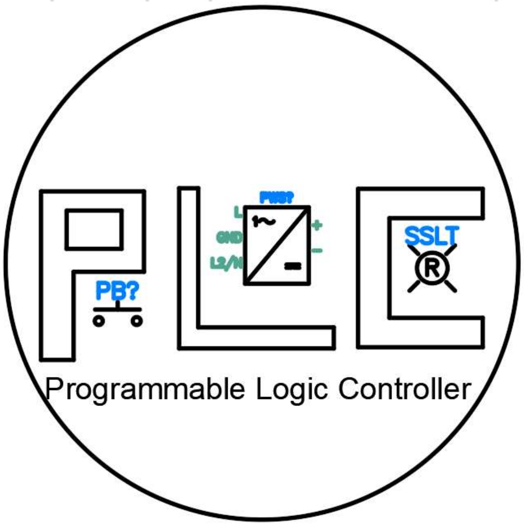
A Latch Example



Flip Flops : (Latches are not used universally by all PLC vendors, others such as Siemens use flip-flops. These have a similar behaviour to latches, but a different notation)







#Timers and Counters –Part1

Lecture 4

3rd

Systems and Control Engineering

Lecturer : Yazen Hudhaifa Shakir

Timers

- There are four fundamental types of timers inside PLC :

1- ON Delay Timer

2- Off Delay Timer

3- Retentive On Delay

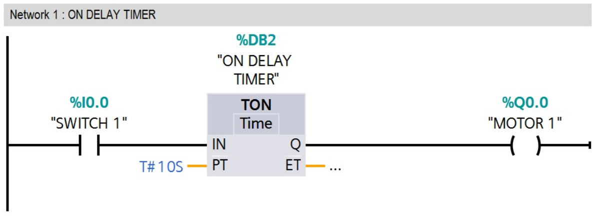
4- Retentive Off Delay

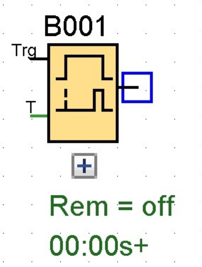
	on-delay	off-delay
retentive	RTO	RTF
nonretentive	TON	TOF

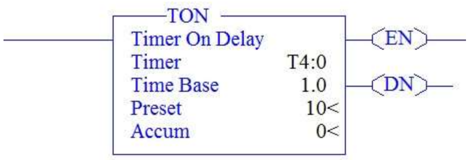
TON - Timer ON
TOF - Timer OFf
RTO - Retentive Timer On
RTF - Retentive Timer oFf

ON Delay Timer (TON)

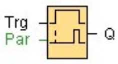
- All these blocks has the same function which is after some time will turn the output on







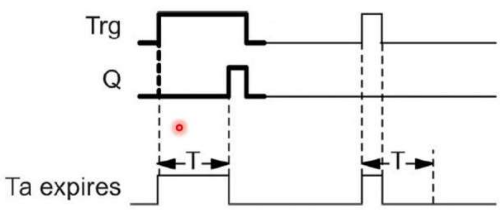
On delay timer in Logo Soft



Short description

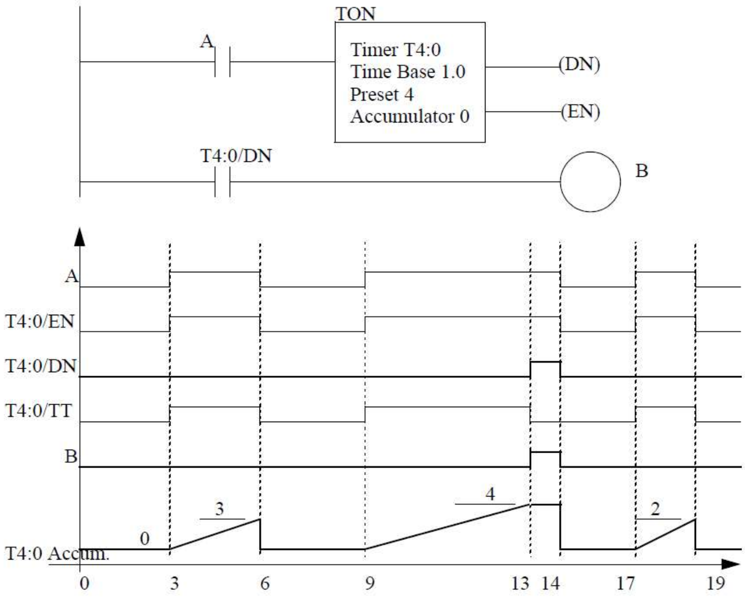
The output does not switch on until a configured delay time has expired.

Connection	Description
Trg input	The Trg (Trigger) input triggers the on-delay time.
Parameter	**T:** represents the on-delay time after which the output is switched on (output signal transition 0 to 1). **Retentivity** on = the status is retentive in memory.
Output **Q**	**Q** switches on after a specified time T has expired, provided Trg is still set.



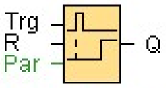
Allen-Bradley TON timer

- The first item is the timer number *T4:0*. This is a location in the PLC memory that will store the timer information
- The *T4:* indicates that it is timer memory, and the *0* indicates that it is in the first location.
- The time base is *1.0* indicating that the timer will work in 1.0 second intervals.
- The Preset is the delay for the timer, in this case it is 4. To find the delay time multiply the time base by the Preset value 4*1.0s = 4.0s.
- The accumulator value gives the current value of the timer as *0*.



Retentive on Delay (RTO)

(A *one-shot at the input triggers* a configurable time. LOGO! sets the output upon expiration of this time)



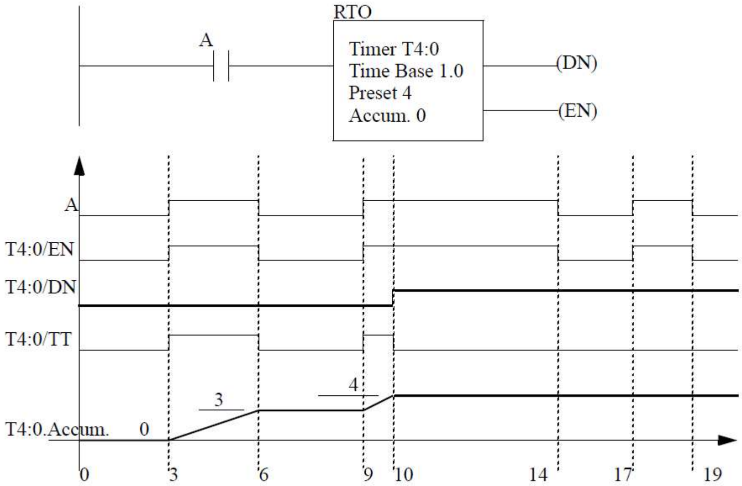
Short description

A one-shot at the input triggers a configurable time. LOGO! sets the output upon expiration of this time.

Connection	Description
Input **Trg**	Trigger the on-delay time via the **Trg** (Trigger) input.
Input **R**	Reset the on-delay time and reset the output to 0 via input **R** (Reset). Reset takes priority over **Trg**.
Parameter	T is the on-delay time for the output (output signal transition 0 to 1).
Output **Q**	**Q** switches on upon expiration of the time T.

Allen-Bradley RTO timer

- It is the same as TON
- **except** that it is retentive.
- The most significant difference is that when the input *A* is turned off the accumulator value does not reset to zero.
- As a result the timer turns on much sooner, and the timer does not turn off after it turns on.



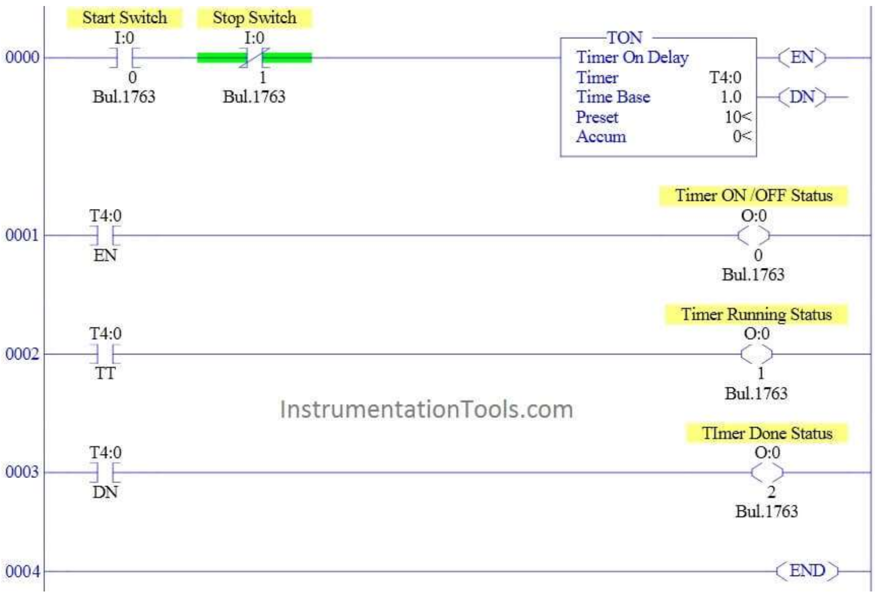
Example for TON

RUNG 0000
Start and Stop Switch is to turn ON/OFF the timer.

RUNG 0001
When start switch is pressed, Timer Enable bit (T4:0/EN) is turned ON to show the status of TIMER ON/OFF condition

RUNG 0002
When start switch is pressed, along with the timer enable bit timer timing bit also turned ON to show the status of timer's accumulator (T4:0/ACC)value is currently in running state.

RUNG 0003
When start switch is pressed, timer starts running from zero to preset valve, timer's done bit is turned ON when accumulator value reached preset value.



#Timers and Counters plus Examples –Part2

Programmable Logic Controller

Lecture 5

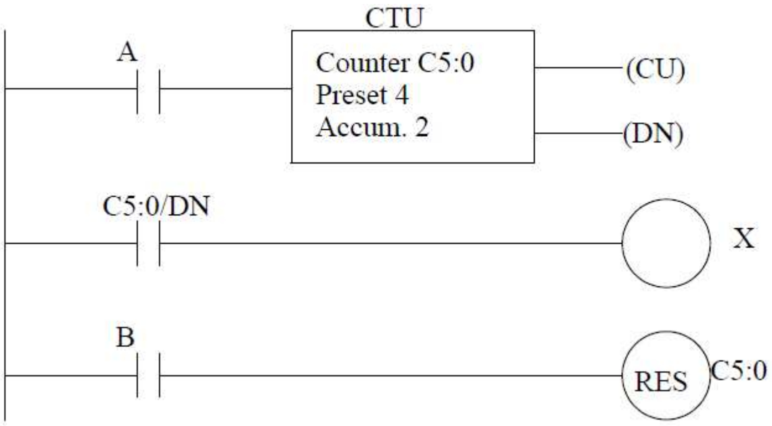
3rd

Systems and Control Engineering

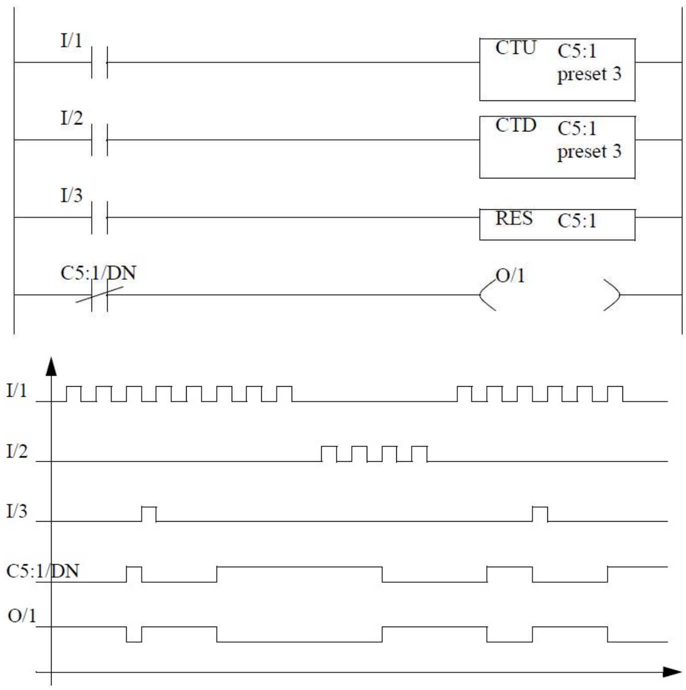
Lecturer : Yazen Hudhaifa Shakir

Counters

- There are two basic counter types: count-up and count-down.
- When the input to a count-up counter goes true the accumulator value will increase by 1 ***(no matter how long the input is true.)***

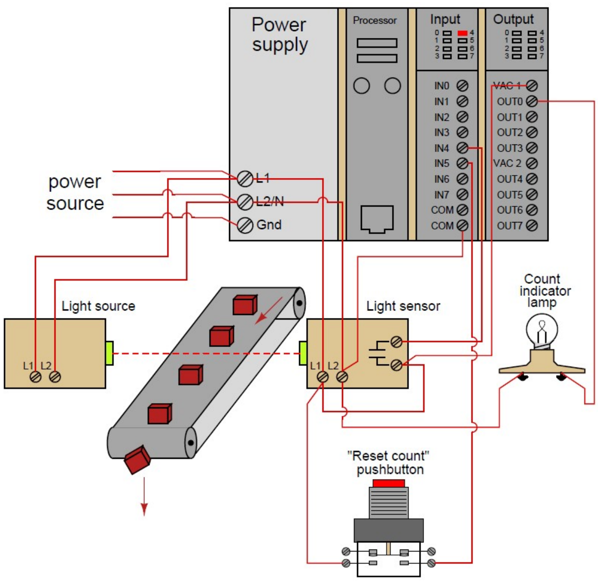


the example input *I/1* drives the count-up instruction for counter *C5:1*. Input *I/2* drives the count-down instruction for the same counter location. The preset value for a counter is stored in memory location *C5:1* so both the count-up and count-down instruction must have the same preset. Input *I/3* will reset the counter.

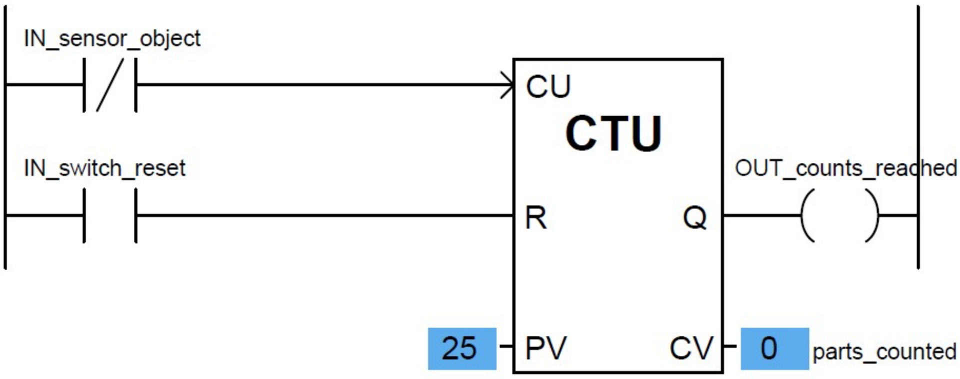


Ex: A PLC-based system designed to count objects as they pass down a conveyor belt:

1. In this system, a continuous (unbroken) light beam causes the light sensor to close its output contact, energizing discrete channel IN4.
2. A push-button switch connected to activate discrete input IN5 when pressed will serve as a manual "reset" of the count value.
3. An indicator lamp (exceeding Limit)

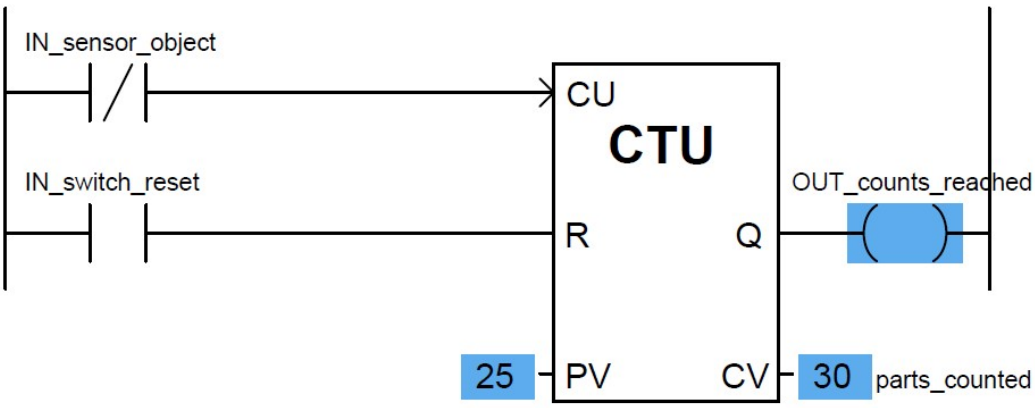


These are two inputs with one output



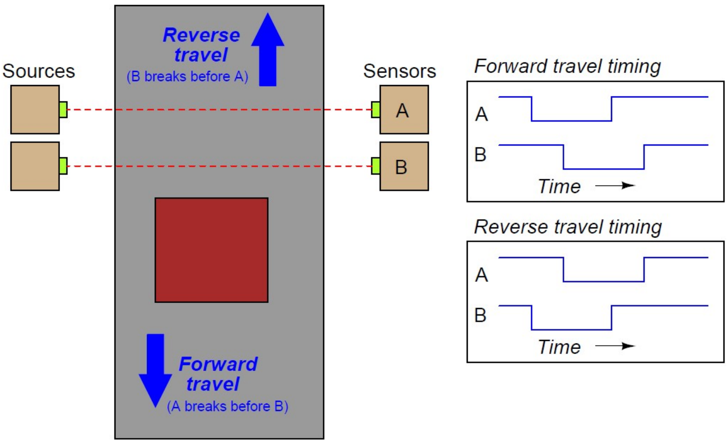
CV (Current value) , PV (Pre-set Value) , According to the IEC 61131-3 programming standard, this counter output should activate whenever the current value is equal to or greater than the pre-set value (Q is active if CV ≥ PV).

After 30 piece passed

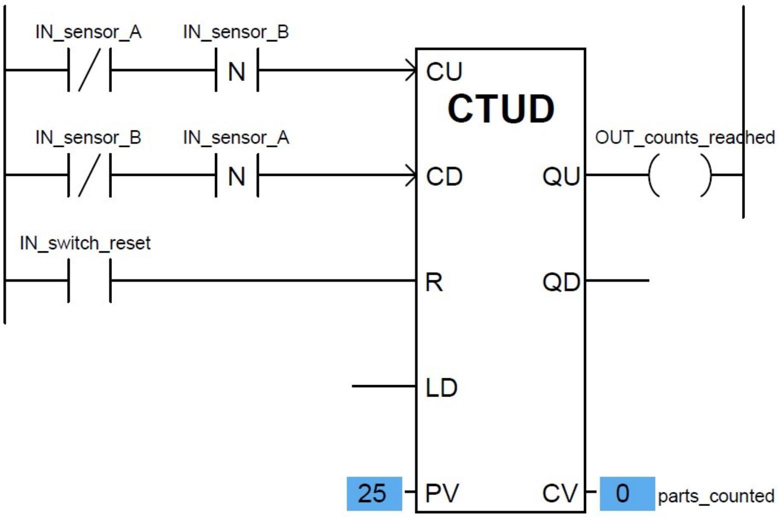


Conveyor belt forward and reverse

- If, for instance, the conveyor belt were ever reversed in direction, the sensor would continue to count objects that had already passed by before (in the forward direction) as those objects retreated on the belt.
- This would be a problem because the system would "think" more objects had passed along the belt (indicating greater production) than actually did.

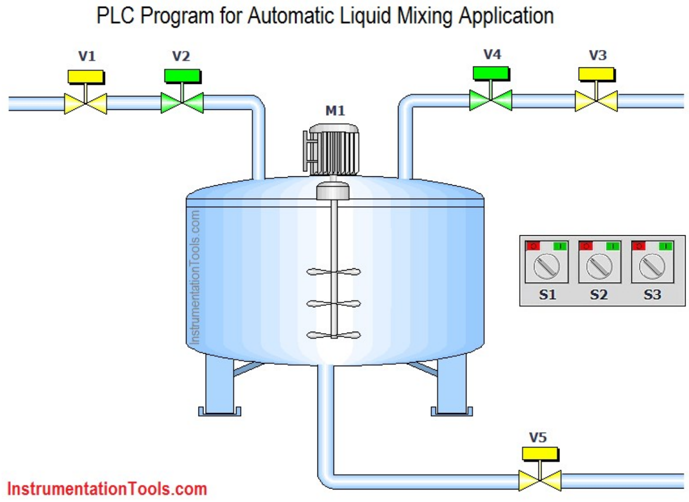


One solution to this problem is to use an up/down counter, capable of both incrementing (counting up) and decrementing (counting down), and equip this counter with two light-beam sensors capable of determining direction of travel.



Ex: Automatic Liquid mixing Application

- In many industries, there are lots of mixing system are used for solutions mixing. Some plants use complete automation or semi-automation.
- In manual system, there are so many disadvantages such as lack of Accuracy, Time delay problem, loss of liquids, Time consuming etc.
- Here we are discussing semi-automatic application of mixing system.



Problem Solution

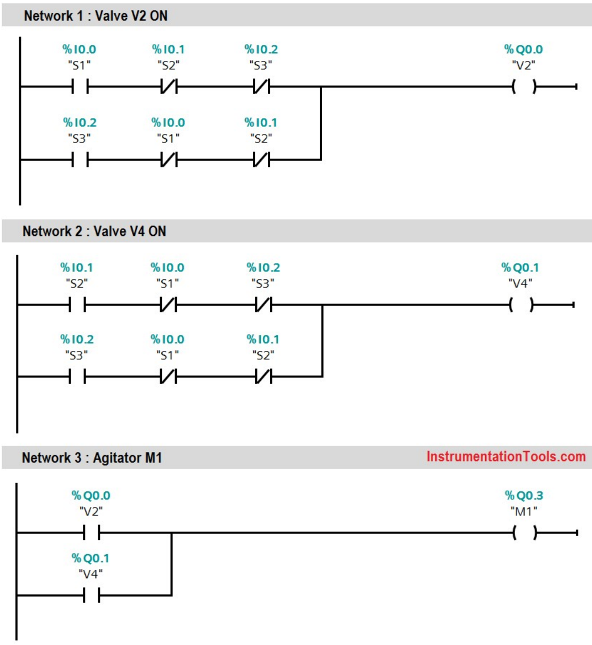
- For this example, we use PLC programming and for that we use Siemens S7-1200 PLC.
- For easy explanation, we can consider simple example of mixing system as shown above.
- In this application pure unmixed solution can be prepared by the operator using switches S1 and S2. And mixed solution or material can be prepared by the operator using switch S3.
- Operator observes the level of the tank and he can discharge the liquid inside tank by operating valve V5.
- Also the agitator motor M will be in running while tank is being filled. We will provide interlock system so operator cannot operate both switches at same time.
- V1, V3 and V5 are manual valves which is not connected to the PLC.
- V2 and V4 are electronically operated valves which can be controlled by PLC.

I/O Assignment

There are three switches S1, S2 & S3
- •S1 : I0.0
- •S2 : I0.1
- •S3 : I0.3

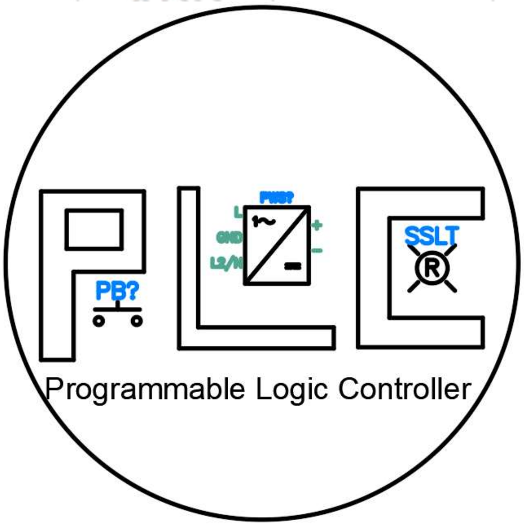
Digital Outputs

We have two valves V2 & V4. one Agitator Motor M1
- •V2 : Q0.0
- •V4 : Q0.1
- •M1 : Q0.2



Program Description

•For this application, we used S7-1200 PLC and TIA portal software for programming.
•In Network 1, we have taken NO contact of S1 (I0.0) and NC contact of S2 (I0.1) and S3 (I0.2) in series. By activating switch S1 operator can START the valve V2 for solution 1 (Liquid 1).
•In Network 2, we have taken NO contact of S2 (I0.1) and NC contact of S1 (I0.0) and S3 (I0.2) in series. By activating switch S2 (I0.1) operator can START the valve V4 (Q0.1) for solution 2 (Liquid 2).
•For both Networks 1 & 2, A parallel connection we have taken, NO contact of S3 (I0.2) and in series with NC contact of S1 (I0.0) and S2 (I0.1).
Because of the above parallel connection, operator can operate both valves by activating switch S3 (I0.2) for mixed solution (Liquid 1 & Liquid 2)
As per our condition, agitator M1 (Q0.2) should be activated automatically while tank is being filled. So we have taken NO contact of V2 (Q0.1) and in parallel NO contact of V4 (Q0.1) so agitator will be activated automatically by operating any switch.



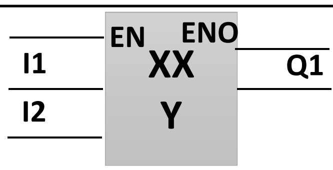
#Comparison and Math Operations

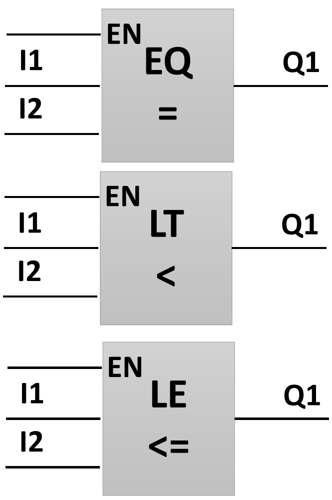
Lecture 6

3rd
Systems and Control Engineering

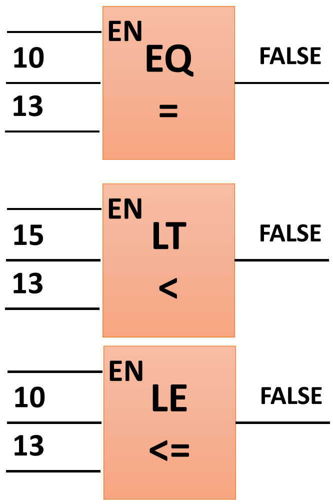
Lecturer : Yazen Hudhaifa Shakir

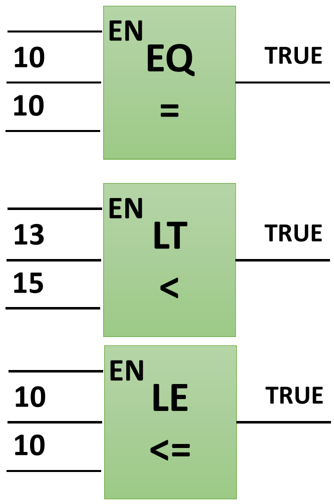
Logical operator (=, <, <= ,> ,>= ,<>)



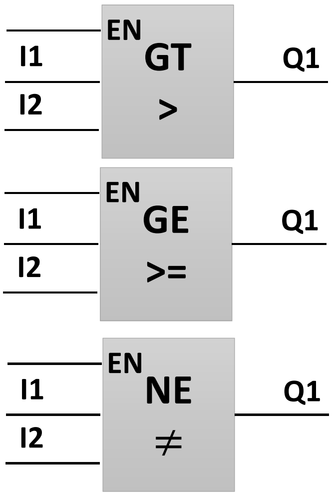


- **Equal Function block**
- **Less Than Function block**
- **Less Than or Equal Function block**

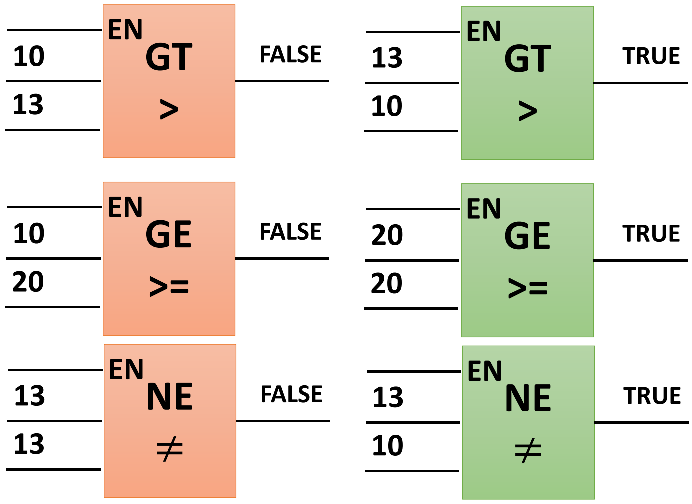




Logical operator Continued .....

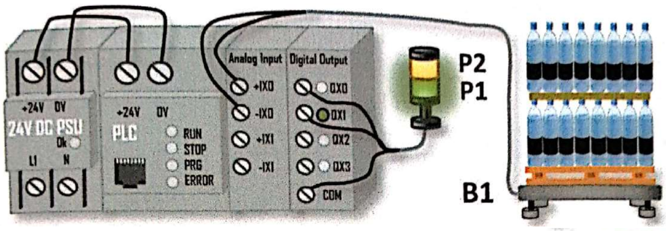


- Greater Than Function block
- Greater Than or equal Function block
- Not Equal

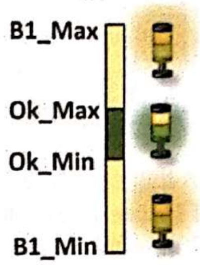


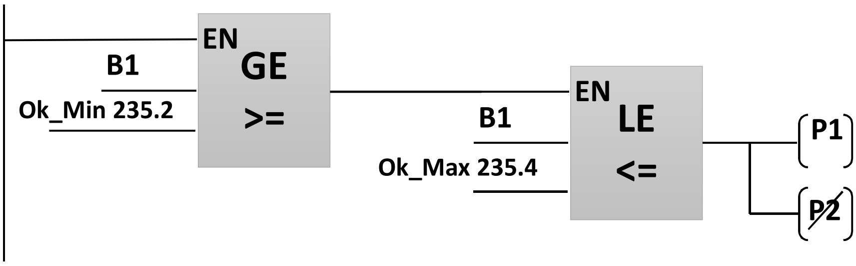
Example on Logical operators (Greater and Less than or equal)

- P1 (Green Light)
- P2 (Yellow Indicator)
- B1_Max=2000 Kg
- B1_Min=0 Kg

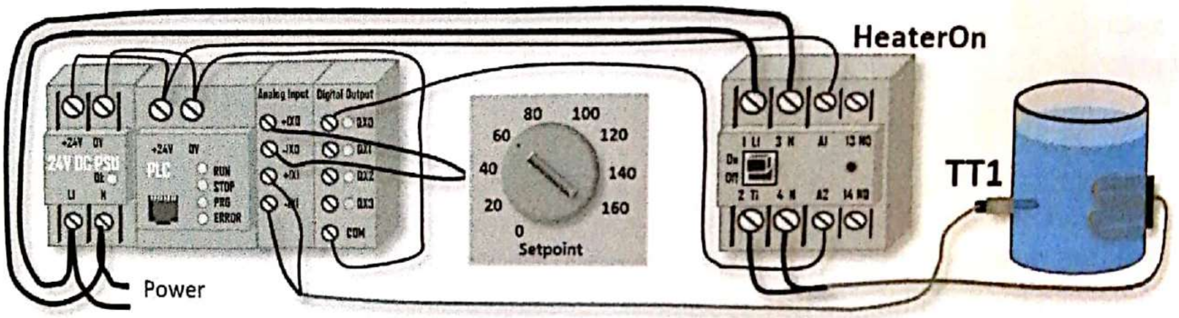


@Licenced: Tom Mejer Automation

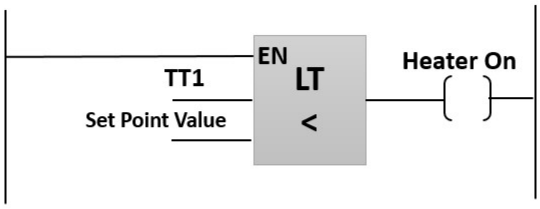




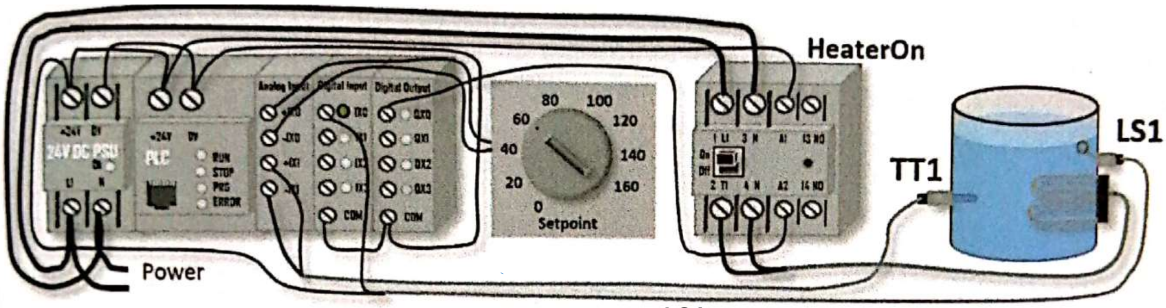
Measuring the temperature inside the tank



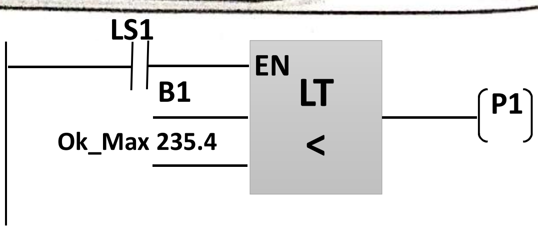
- The data types for **TT1** and **Setpoint** are both **REAL**
- **Heater On is BOOL**



Same Example with Protection of Heater

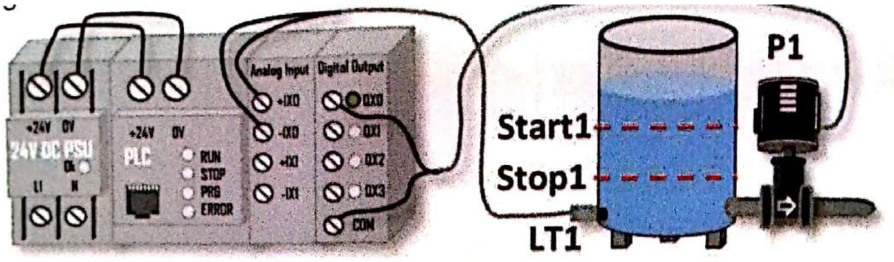


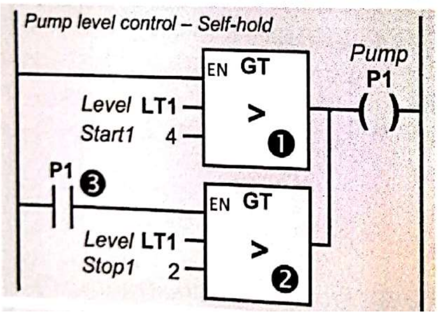
- LS1 is a Level switch (NO contact in Ladder)

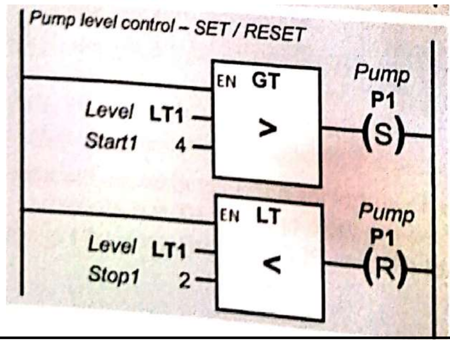


Pump Controlled by Water Level

- The Pump will stop when the Water level is below **Stop1**
- **Level Sensor LT1** measure level from 0 – 5 meters

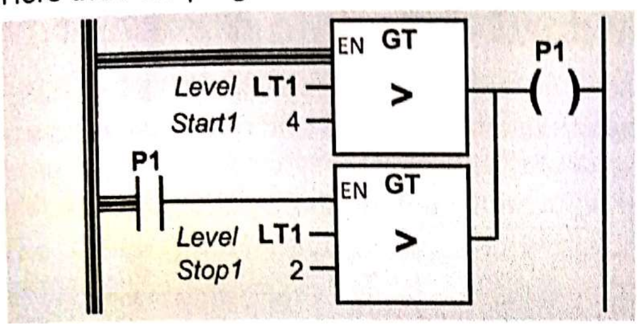


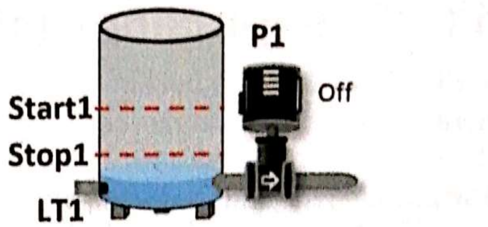




Self-hold to run (self starting)

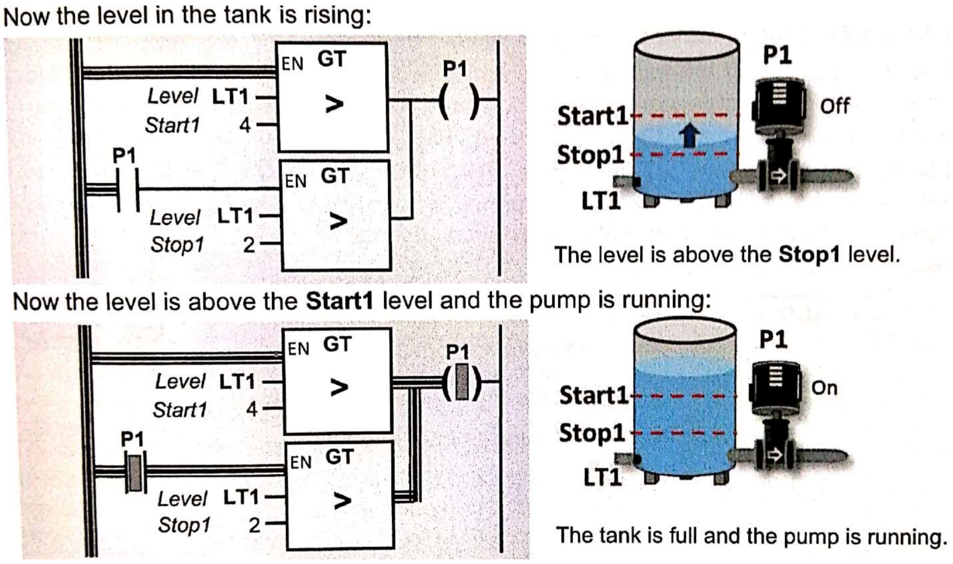
Here the PLC program with self-hold is set to run:

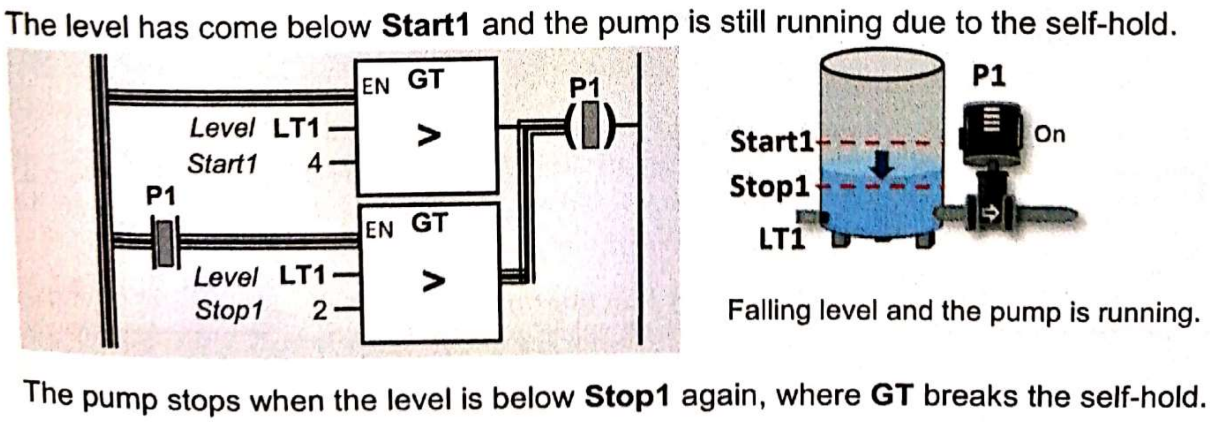




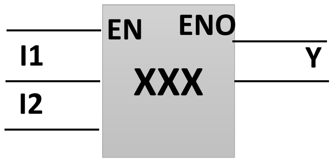
Empty tank and the pump has stopped.

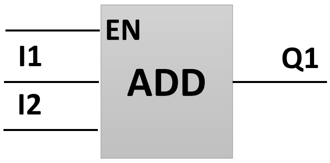
The pump P1 will start after Start1 Level reached



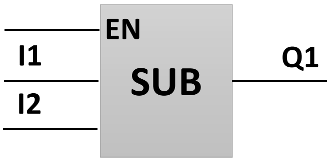


Math Functions (+, - , x , e , /)

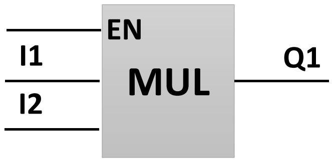




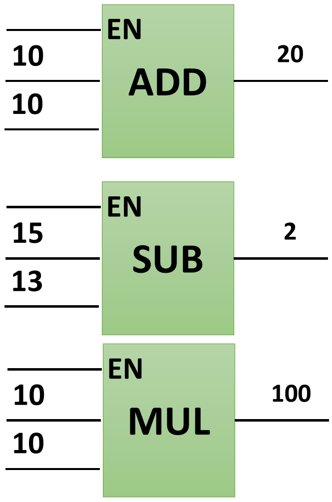
- **Addition,**
- **Adds two numbers together**
- **Q1=I1+I2**



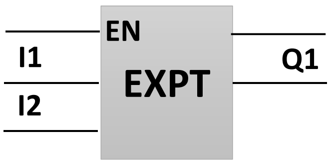
- **Subtract ,**
  Q1=I1-I2



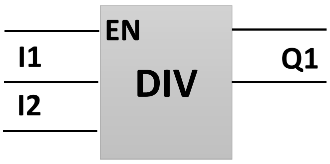
- **Multiply,**
  Q1=I1*I2



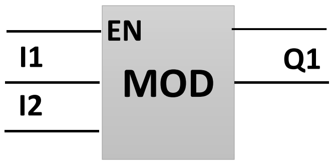
Math Functions (+, - , x , e , /)



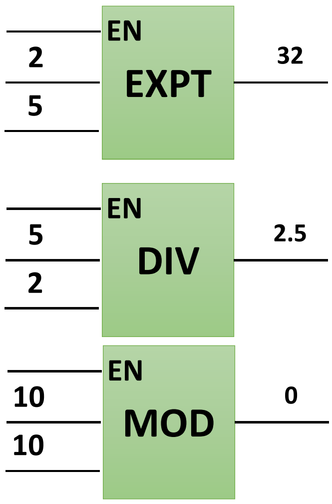
- **Exponential ,**
- $Q1 = I1^{I2}$



- **Division ,**
  Q1=I1/I2

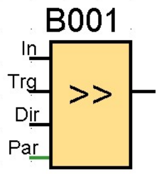


- **Modulo (Find the rest**
  after division)



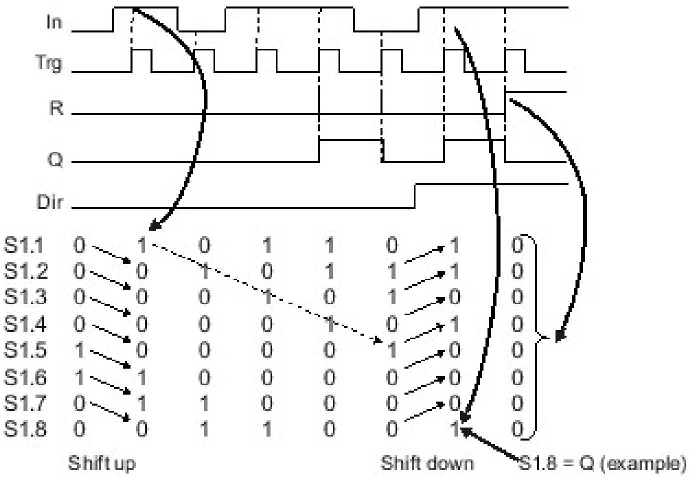
Shift registers

The shift register function reads an input value and shifts the bits. The output value corresponds with the configured shift register bit. The shift direction can be changed at a special input. For device series 0BA4 to 0BA6, you can use only one shift register in one circuit program.



Q=S1.1

Rem = off



#Conversion , Math and Selecting using in Ladder Diagram

Lecture 7

3rd

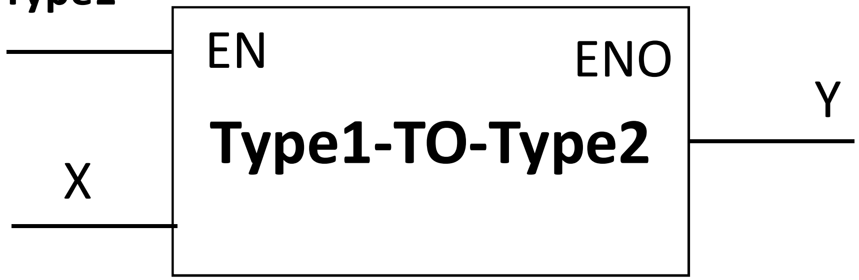
Systems and Control Engineering

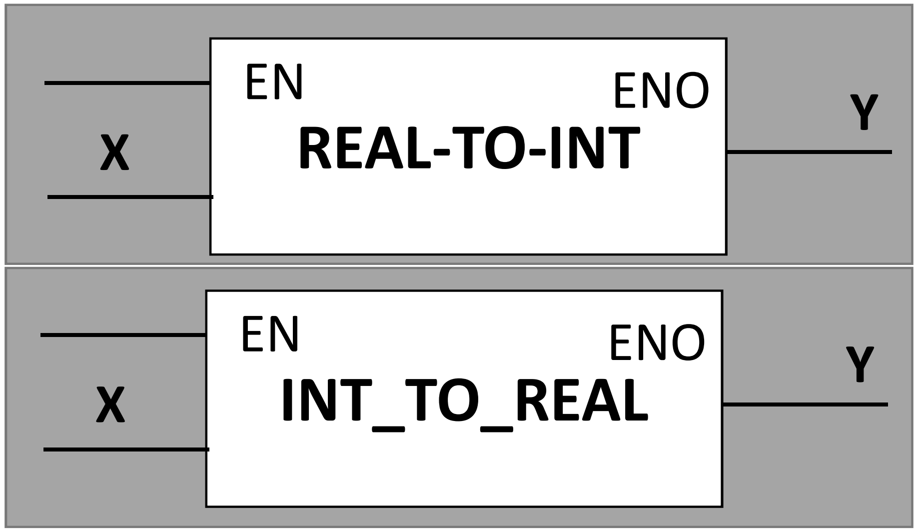
Lecturer : Yazen Hudhaifa Shakir

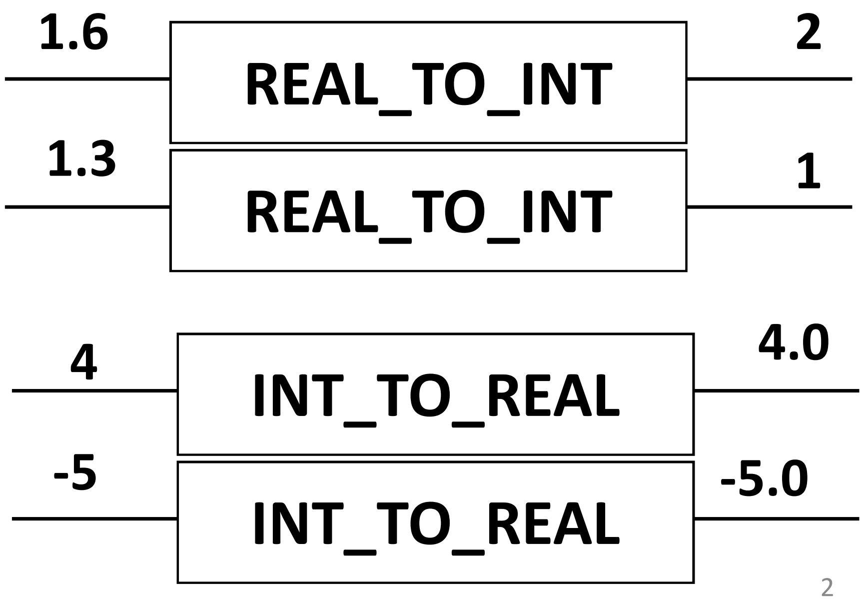
Conversion Functions between data Types

- In order to ensure that the data types of the inputs and outputs are the same for each block inside the PLC program , different data conversation functions have been built in , some PLC vendors provides more than 100 different conversion functions ready to use

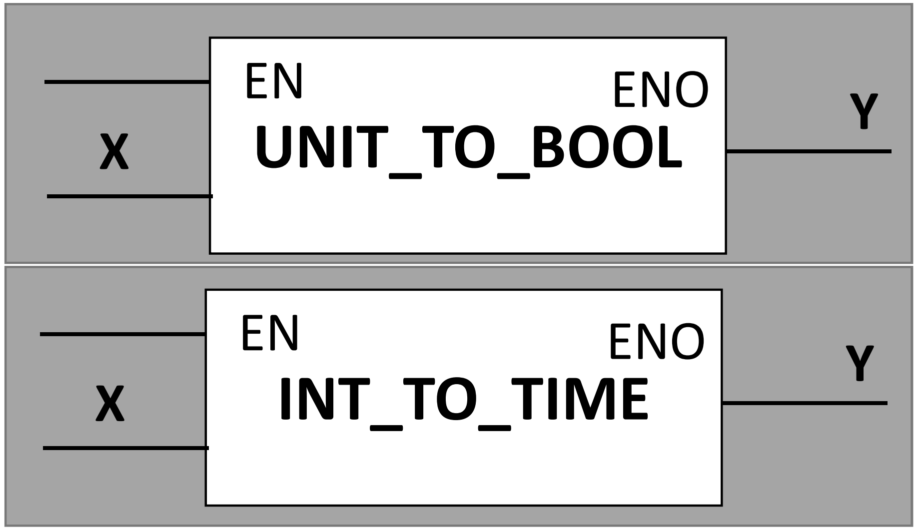
- **Type 2 Data Type will replace Type1**



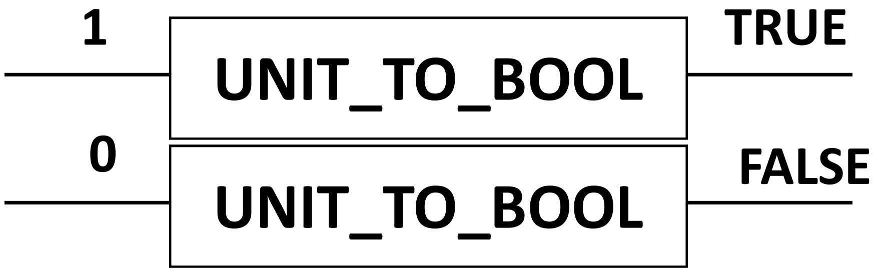




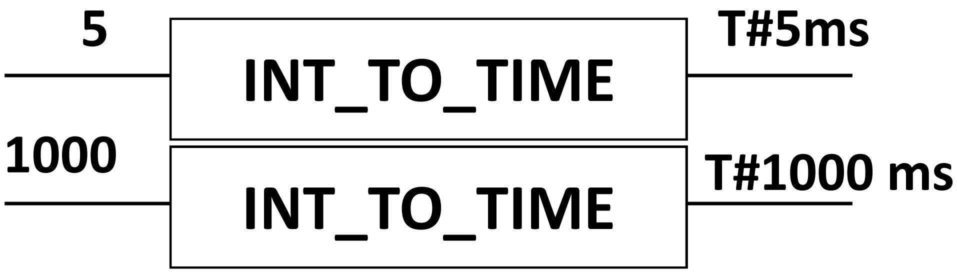
Conversion Functions between data Types



- Convert from an unsigned integer (UNIT) to a Boolean (BOOL)



- Convert from integer to time and normally time in milliseconds

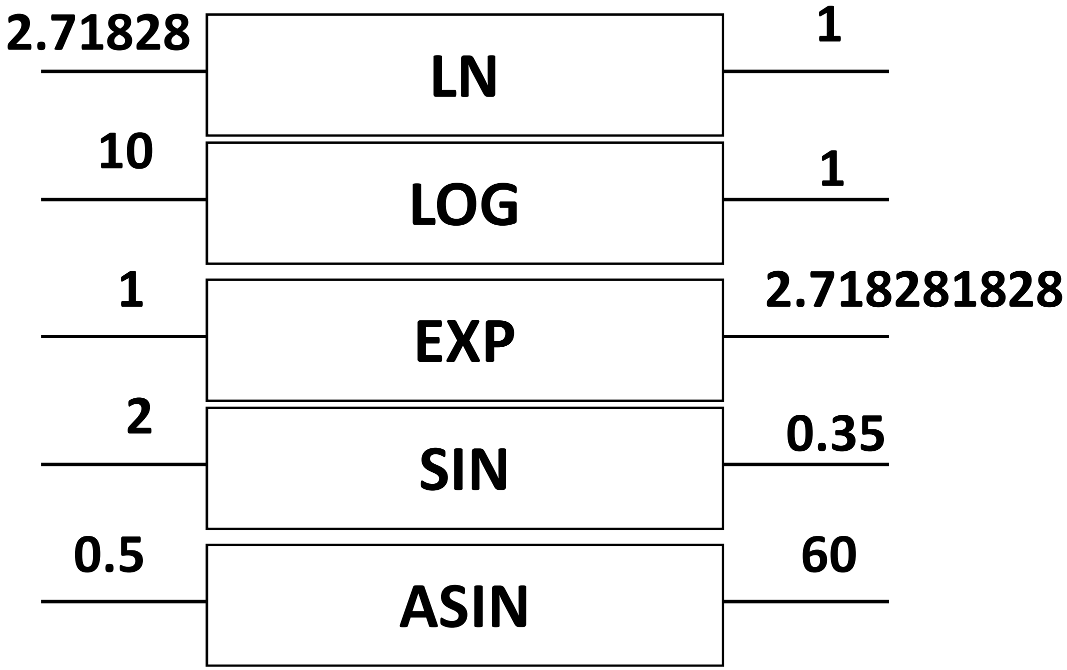


Math Functions (scientific calculator)

Input	Function	Output
3.9	TRUNC	3
7.8	FRAC	0.8
-1.3	ABS	1.2
7.8	FLOOR	7
2	SQR	4
16	SQRT	4

Input	Function	Output
-2.5	TRUNC	-2
-4.49	FRAC	0.49
4.4	ABS	4.4
-3.8	FLOOR	-4
4	SQR	16
25	SQRT	5

Math Functions (scientific calculator) (continued)



1- Natural Log

2- Log with base 10

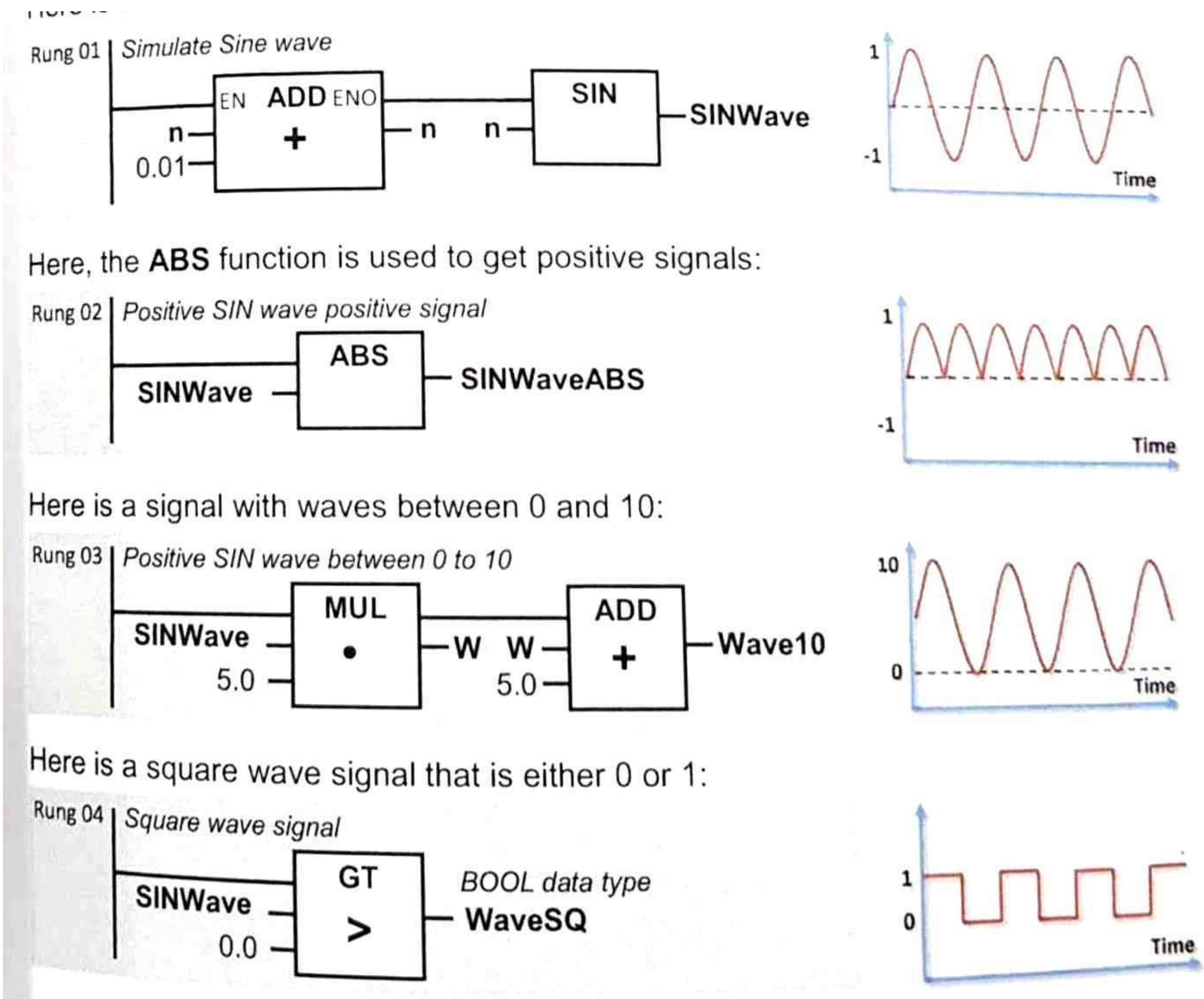
3-Exponential function

4-sin and cos and Tan in degrees

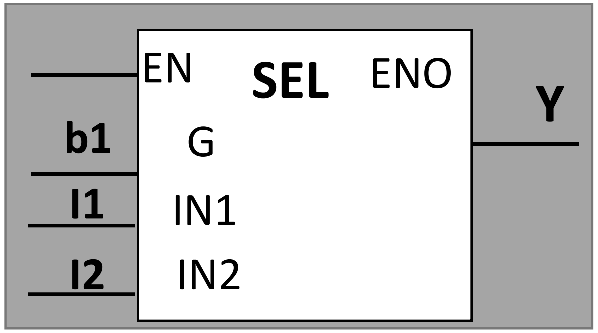
5- Sine inverse

Simulation signals : can be used when testing the program without real hardware

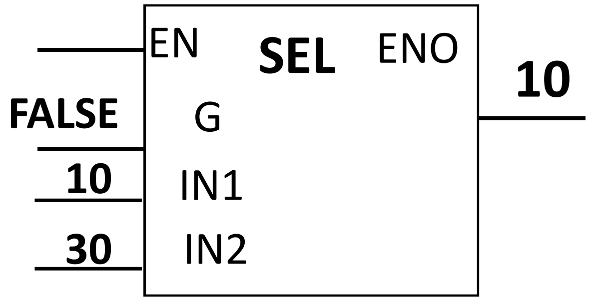
n: (REAL) variable
0.01 is the frequency

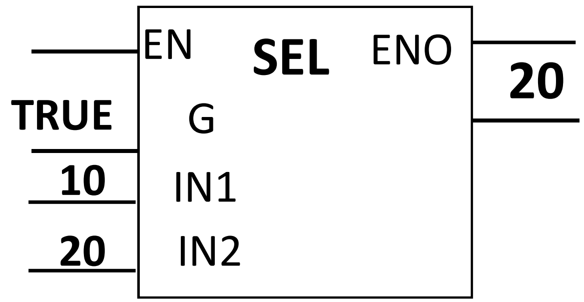


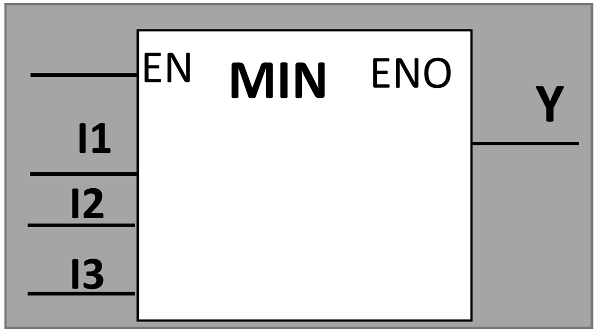
Function for selecting a value : (selecting a certain value from several values)

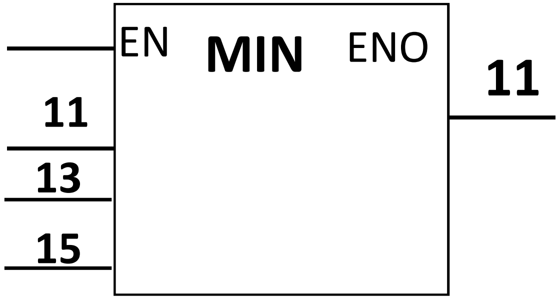


- If b1 **FALSE** the content of I1 will be copied to Y otherwise the Input 2 will be out

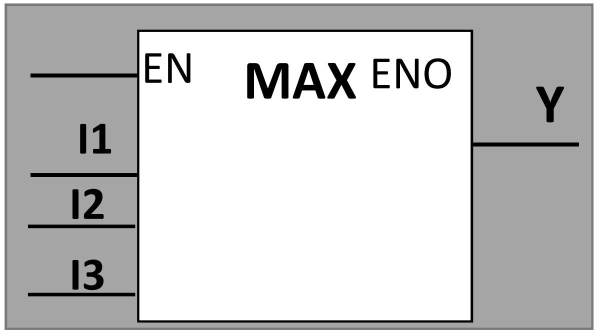


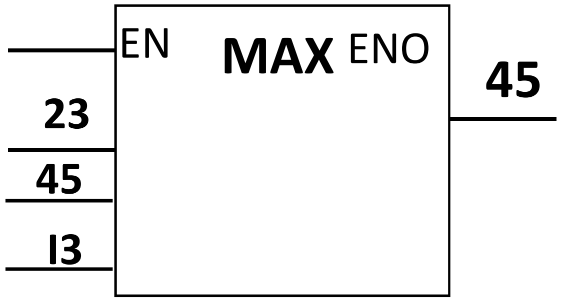






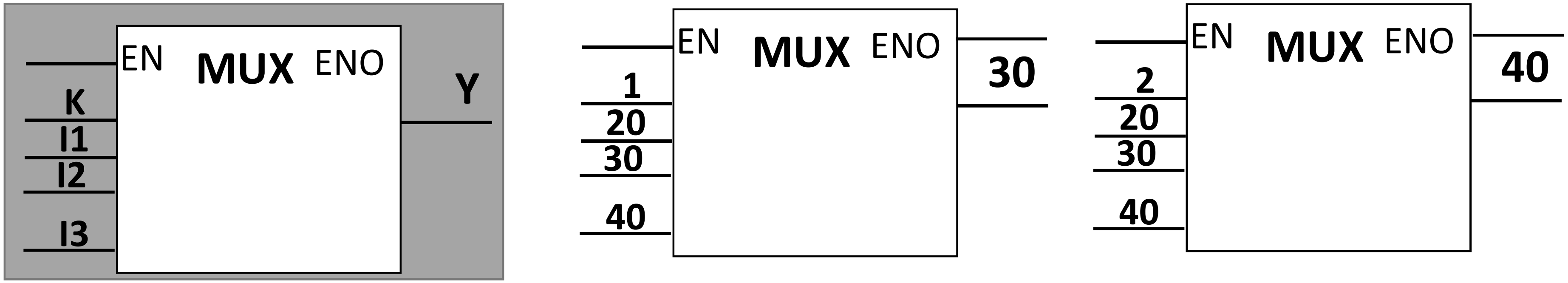
- We can use more than three inputs to choose the minimum value to be copied to Y

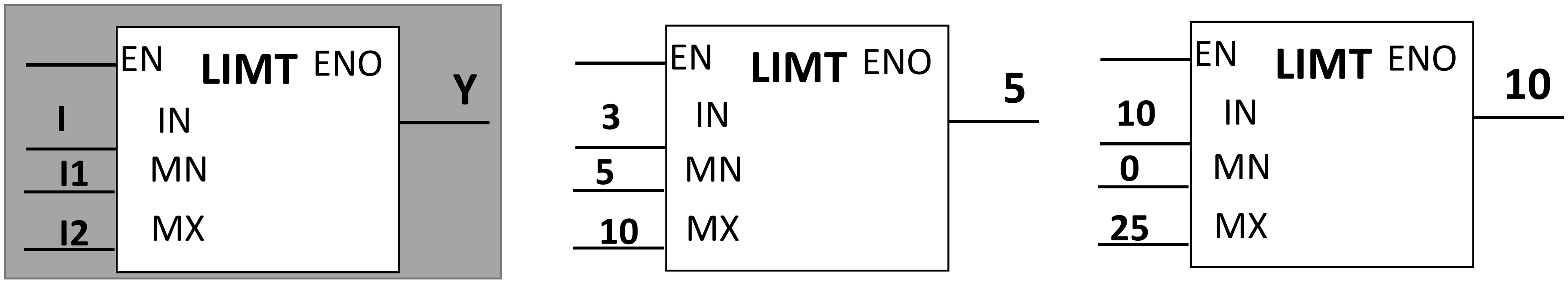




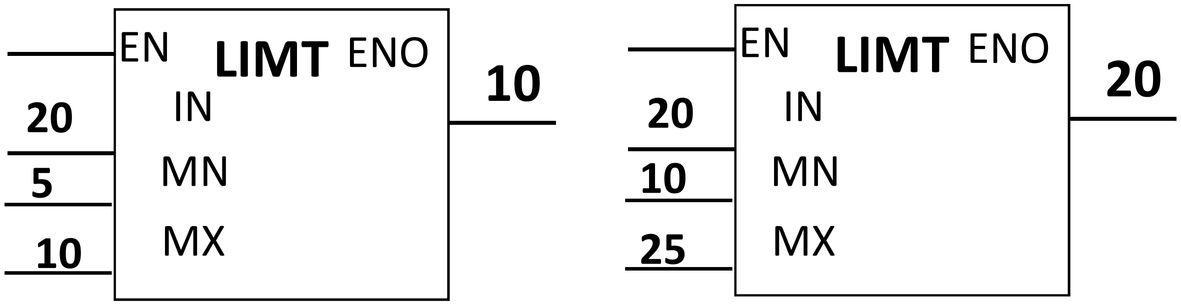
- Selecting the highest value between several inputs

- If K=0 , the contents of I1 are copied to Y
- If K=1 , the contents of I2 are copied to Y and so on , **MUX can be used if the motor has to change to several** speeds





- *LIMT Function : the contents of I are copied to Y if its value between I1 and I2*



Data types in PLC

5.1 Elementary data types (INT, REAL, BOOL...)

The following (examples) simple data types are standard in any PLC controller:

Data type	Bits	Numeral system	Note	Lowest and highest value	Example
BOOL (Bit)	1	Boolean (Binary)		FALSE/TRUE or 0 to 1	TRUE
BYTE	8	HEX (Hexadecimal)		16#0 to 16#FF	16#10
WORD	16	Binary		2#0 to 2#1111111111111111	2#0001000000000000
UINT	16	HEX (Hexadecimal)		16#0 to 16#FFFF	16#1000
		BCD (Binary-Coded Decimal)		C#0 to C#999	C#998
		Unsigned Integer (positive numbers)		0 to 65535	564
DWORD (Double word)	32	Binary		2#0 to 2#1111111111111111 1111111111111111	2#1000000100011000 1011101101111111
		HEX (Hexadecimal)		16#00000000 to 16#FFFFFFFF	16#00A21234
		Unsigned Double word (integer)		0 to 4294967295 (4.29 billion)	435
INT (Integer)	16	Signed integer		-32768 to 32767	101
DINT (Double integer)	32	Signed double integer		-2147483648 to 2147483647 (2.1 billion)	107

Data type	Bits	Numeral system	Note	Lowest and highest value	Example
REAL (Floating-point number)	32	IEEE 754 Floating-point number (Decimal number)	1	Lowest value: +/-3.402823E+38 Highest value: +/-1.175495E-38	1.234567e+13
LREAL (Long Real)	64	Dobbelt Float (Decimal number) IEEE 754		Lowest:-1.7976931348623E308 Highest: 1.79769313486232E308	3432.54
TIME (IEC time) **LTime**	32 64	IEC time Step in 1 [ms] or Step in 1 [ns]	4	T#1ns to T#24d20h31m23s	TIME#10s T#10d14h11m23s T#5s12ms23us300ns
DATE (IEC date)	16	IEC day, step 1 day		D#1990-1-1 to D#2168-12-31	D#1996-3-15 DATE#1996-3-15
TIME _OF_DAY (Time)	32	Time in a step of 1 [ms]	4	TOD#0:0:0.0 to TOD#23:59:59.999	TOD#1:10:3.3 TIME_OF_DAY#1:10:3.3
CHAR **WCHAR**	8 16	ASCII characters (letter or sign)	2	'A', 'B' etc.	'E'
STRING		Text	3	Up to 255 characters	"This is a text"

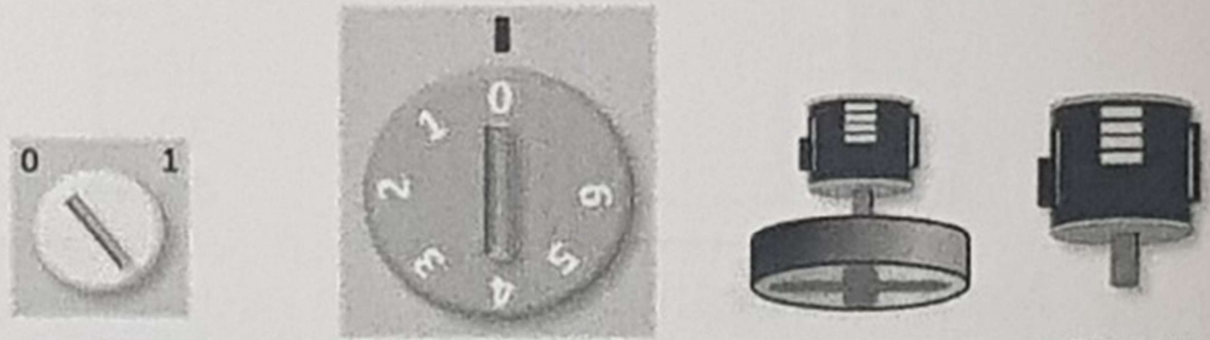
All variables must have a data type. If a variable is given a value outside the minimum and maximum value range of the data type, a run time error may occur and consequently the PLC may stop the program execution. This may again lead to strange behavior when executing the program (the program may seem unstable).

A few PLC types provide more data types than the ones listed above. In general, it is recommended to use only a few data types so that the PLC code can be copied in an easier way to other PLC types. Some special data types such as **S7TIME**, **LWORD** and **ULINT** cannot be used by all PLC types. This means that copying PLC code with special datatypes, or upgrading to a larger PLC, may take a lot of work and risk introducing errors to the code.

The three most used data types are **BOOL**, **INT** and **REAL**. The reason why **INT** is used more often than **WORD** is that **INT** provides the same amount of data as the bit size in a PLC making it a fast data type. On the other hand, if **REAL** is used, the PLC will auto-generate underlying machine code as the PLC can only work with integers.

14.1.1 Example: Setting the speed of the motor and fan

Here is a program example where the speed of a motor is set on a manual rotary switch with the variable name **MotorSwitch**. There is also a fan named **Fan**, and a **PowerSwitch** to turn on and off both the motor and the fan:

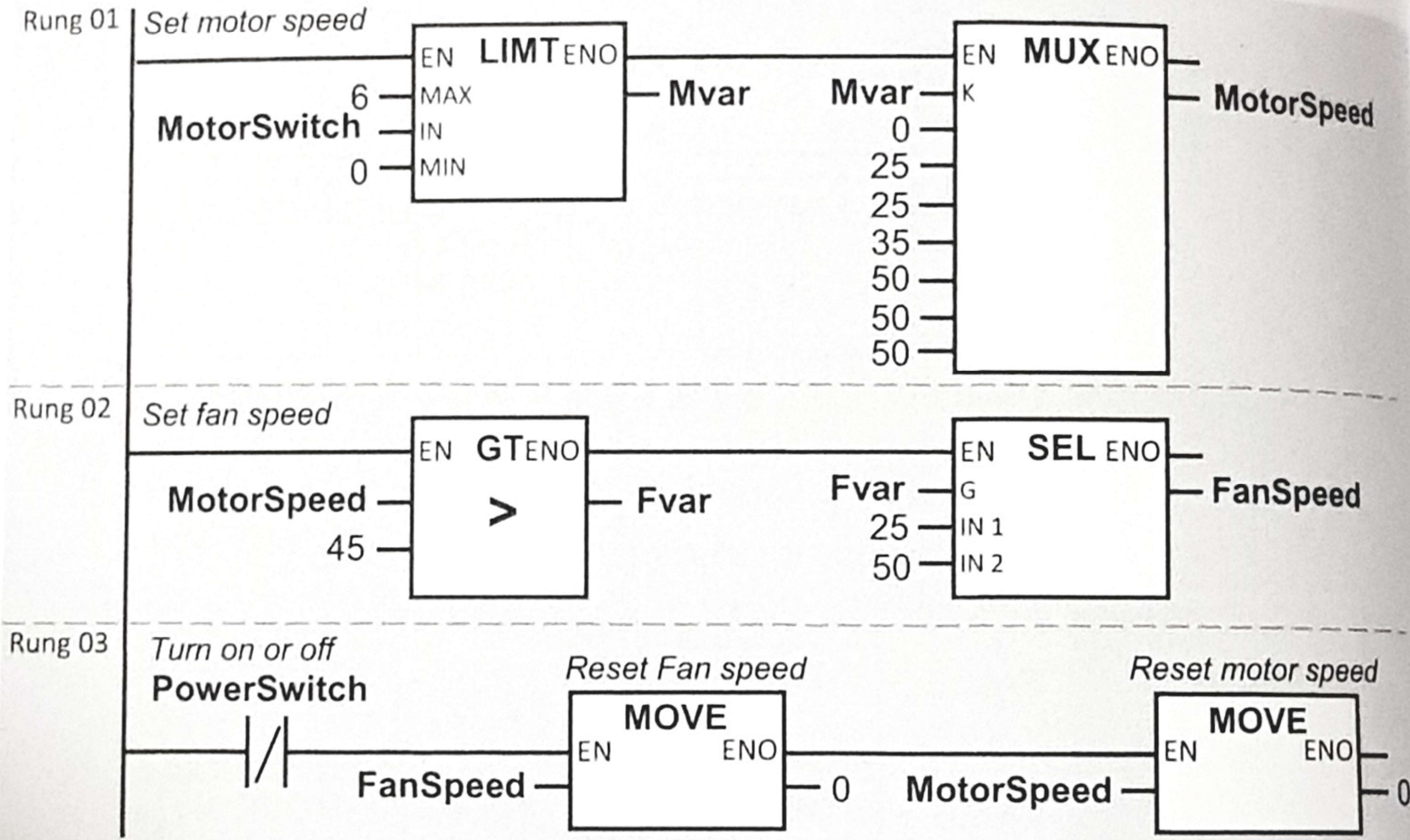


PowerSwitch **MotorSwitch** **Fan** **Motor**

The rotary switch **MotorSwitch** is set in steps from 0 to 6, which can be 7 different voltage levels received from an analog input module.

The **Fan** must run at a speed of 25 Hz. If **Motor** runs more than 45 Hz, the **Fan** must run at 50 Hz.

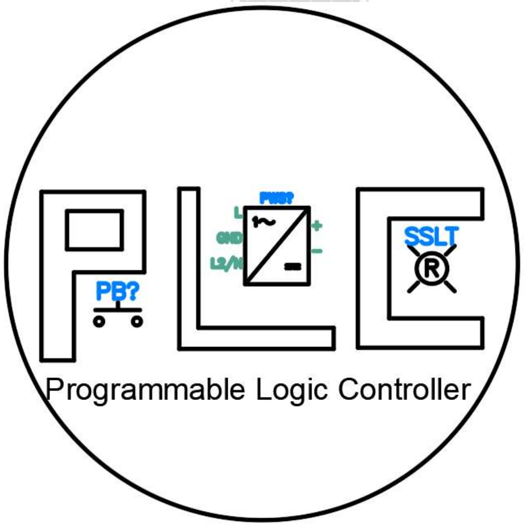
This is a program example with three networks:



I/O assignment

PROGRAM MotorControl

Scope	Name	Address	Data type	Initialization	Comment
VAR	**MotorSwitch**		**INT**		Speed 0 to 6 (analog value)
VAR	**PowerSwitch**		**BOOL**		**FALSE** in position 0
VAR	**FanSpeed**		**INT**		Analog value (Speed). 0 to 50 Hz
VAR	**MotorSpeed**		**INT**		Analog value (Speed). 0 to 50 Hz
VAR	**Mvar**		**INT**		Internal VAR for MotorSwitch
VAR	**Fvar**		**BOOL**		Internal VAR for Fan



#PID Control

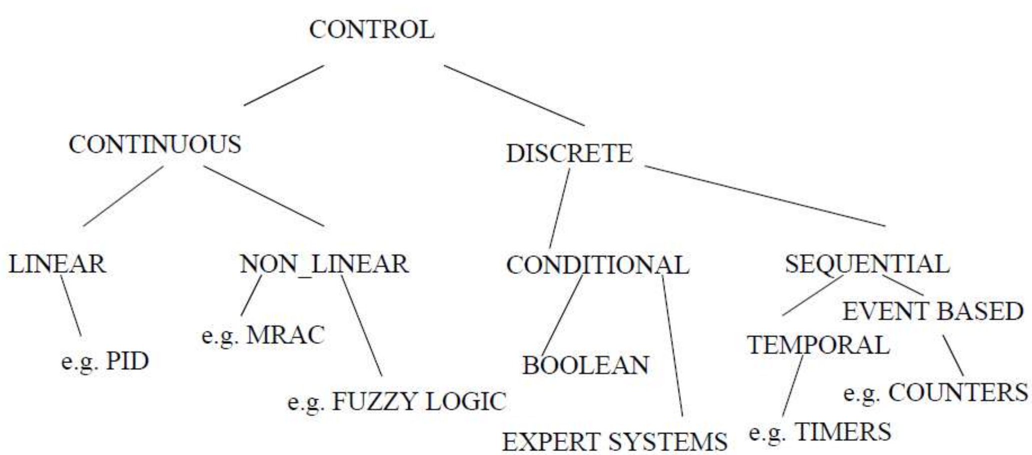
Lecture 8

3rd

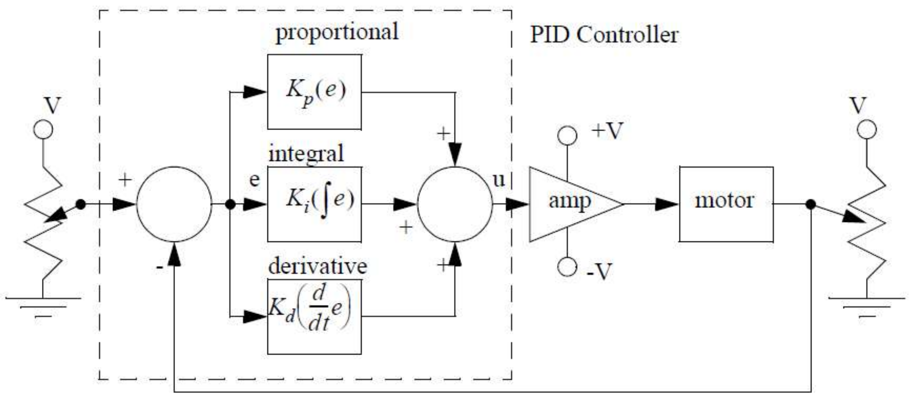
Systems and Control Engineering

Lecturer : Yazen Hudhaifa Shakir

Types of Control



PID Block diagram in the system



3

- Problem: Design an analog controller that will read an oven temperature between 1200F and 1500F. When it passes 1500 degrees the oven will be turned off, when it falls below 1200F it will be turned on again. The voltage from the thermocouple is passed through a signal conditioner that gives 1V at 500F and 3V at 1500F. The controller should have a start button and E-stop.

4