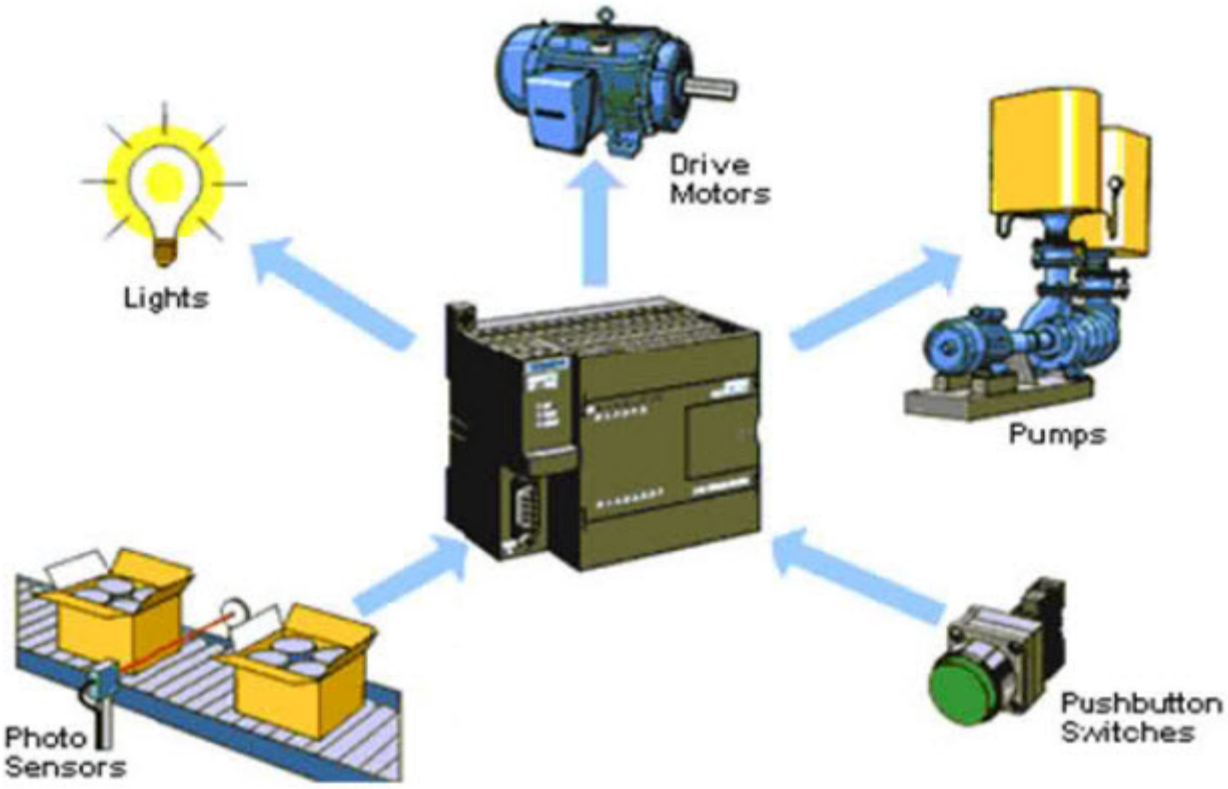
Basics of
Programmable Logic Controller



Prepared By
Eng. Ahmad Mamdouh Kabsha

Basics of Programmable Logic Controller

First Edition

Collected and prepared

By

Eng. / Ahmad Mamdouh Kabsha

Index

Basics of Programmable Logic Controller (PLC)

Introduction:

- **Definition:** PLC is a type of computer commonly used in commercial and industrial control applications, capable of storing instructions, such as sequencing, timing, counting, arithmetic, data manipulation, and communication.

- **Historical background:**

The Hydramatic Division of the General Motors Corporation specified the design criteria for the first programmable controller in 1968 by Dick Morley to eliminate the high costs associated with inflexible, relay controlled systems; some of the initial specifications included the following:

1. The new control system had to be cheaper than relay systems.
2. The system had to survive in an industrial environment.
3. The input and output interfaces had to be easily replaceable.
4. The controller had to be designed in modular form, so that subassemblies could be removed easily for replacement or repair.
5. The control system needed the capability to pass data collection to a central system.
6. The system had to be reusable.
7. The method used to program the controller had to be simple, so that it could be easily understood by plant personnel.

And by 1969, the programmable controller had its first product offspring. These early controllers met the original specifications and opened the door to the development of a new control technology.

PLC at that time was called a PC for Programmable Controller and later, after that changed the name to PLC, was produced by Dick Morley's company called Modicon, Modicon stands for **MO**dular **DI**gitial **CON**troller.

Pic.1 Dick Morley

Example for the advantage of using PLC:

Let's say that two push buttons, SW1 and SW2, are connected to a PLC, two lamps, Lmp1 and Lmp2, are also connected to the PLC.
If SW1 turns on, lamp Lmp1 turns on also and if push button SW2 turns on, lamp Lmp2.

Let's say that you want to change this around so that SW1 controls Lmp2 and SW2 controls Lmp1.

1. In a traditional system, you would have to rewire the circuit so that the wiring from the push button SW1 goes to the lamp Lmp2 and vice versa.

2. When using PLC, making this change is as simple as making a small change in the control program.

- **PRINCIPLES OF OPERATION:**

A programmable controller, as illustrated in Figure 1, consists of two basic sections:

i. The central processing unit.
ii. The input/output system.

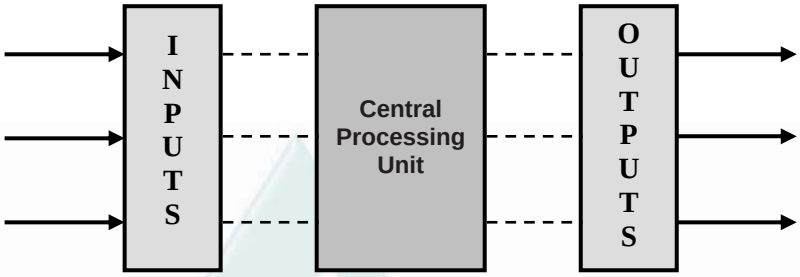


Figure 1

The central processing unit: The central processing unit (CPU) governs all PLC activities.

The input/ output system: is the **interface** by which field devices are connected to the controller and the main purpose of the interface is to condition the various signals received (Inputs like push buttons, switches, and measurement devices) or sent to (outputs Lights, horns, motors, and valves) external field devices.

The I/O interfaces provide the connection between:

i. CPU and the information providers (inputs).
ii. CPU and controllable devices (outputs).

During its operation, the CPU completes a scan cycle (Figure 2) consists of:

1. Reading the input data from the field devices via the input interfaces.
2. Execute the control program stored in the memory system based on status of the input data to calculate the updated status of outputs.
3. Write the updated outputs through output interface.

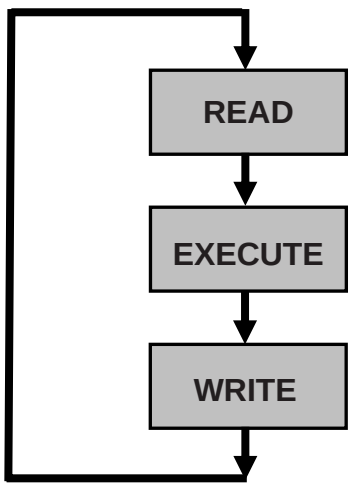


Figure 2

I/O modules could be divided to two types according to power supply:

Source I/O Devices and Modules:

On figure 4, an input source module. Note that one terminal of the module is connected to the +VDC. Now note the input device. One side of the input device is wired to VDC common. According to the definition, this input device is connected type sink, (one connection of a sink device or module is connected to VDC common.) These connections define another fact, source modules are wired with sink devices.

A source module supplies the I/O power supply +VDC to the I/O device. The I/O device supplies the return path to the VDC common.

The polarity of a module can always be determined by looking at how the field devices are wired to the module. In the figure on the right, since one side of the field device is wired the VDC common, as stated above the device is sinking. Since the field device is sink, then by definition the module must be source.

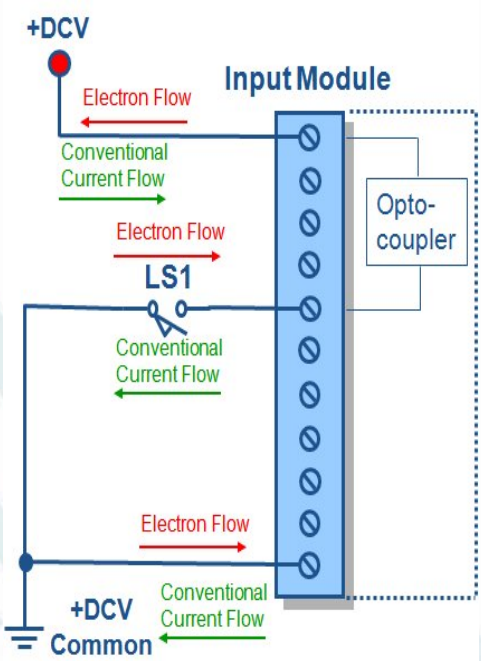


Figure 3

On figure 4, an output source module. Note that one terminal of the module is connected to the +VDC. Now note the output device. One side of the output device is wired to VDC common. According to the definition, this output device is connected type sink, (one connection of a sink device or module is connected to VDC common.) These connections define another fact, source modules are wired with sink devices.

A source module supplies the I/O power supply +VDC to the I/O device. The I/O device supplies the return path to the VDC common.

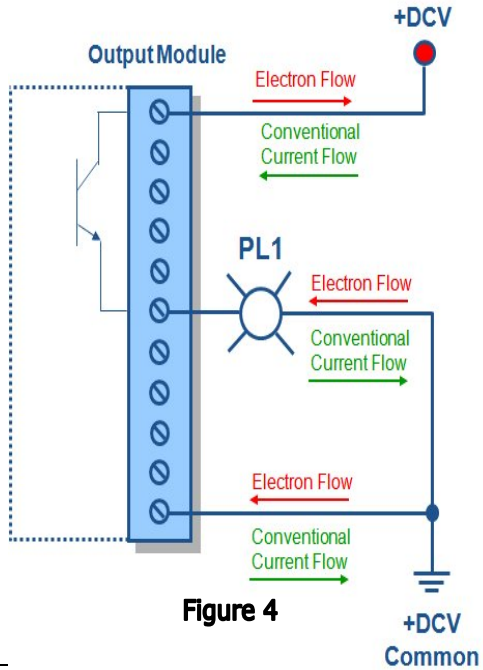


Figure 4

The polarity of a module can always be determined by looking at how the field devices are wired to the module. In the figure on the right, since one side of the field device is wired the VDC common, as stated above the device is sinking. Since the field device is sink, then by definition the module must be source.

Sink I/O Devices and Modules:

On figure 5, an input sink module. Note that one terminal of the module is connected to the DCV common. Now note the input device. One side of the input device is wired to +VDC. According to the definition, this input device is connected type source, (one connection of a souce device or module is connected to +VDC.) These connections define another fact, sink modules are wired with source devices.

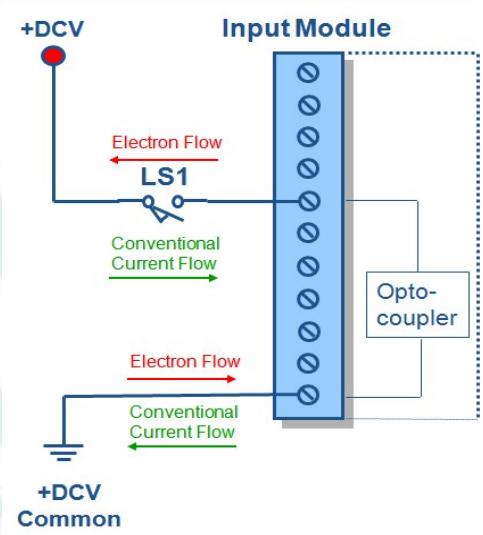


Figure 5

A sink module supplies the I/O power supply return path or the path to the DCV common.

The polarity of a module can always be determined by looking at how the field devices are wired to the module. In the figure on the right, since one side of the field device is wired the +VDC, as stated above the device is sourcing. Since the field device is source, then by definition the module must be sink.

On figure 6, an output sink module. Note that one terminal of the module is connected to the DCV common. Now note the output device. One side of the output device is wired to +VDC. According to the definition, this output device is connected type source, (one connection of a souce device or module is connected to +VDC.) These connections define another fact, sink modules are wired with source devices.

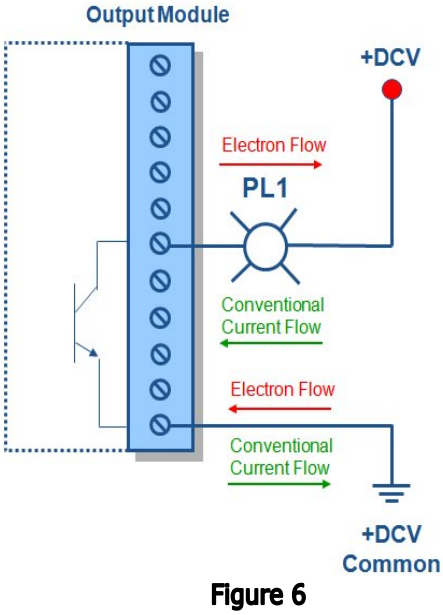


Figure 6

A sink module supplies the I/O power supply return path or the path to the DCV common.

The polarity of a module can always be determined by looking at how the field devices are wired to the module. In the figure on the right, since one side of the field device is wired the +VDC, as stated above the device is sourcing. Since the field device is source, then by definition the module must be sink.

PLC ranges:

1. Micro PLCs:
   Micro PLCs are used in applications controlling up to 32 input and output devices.

2. Small PLCs:
   Small PLCs are used in applications controlling 32 up to 128 input and output devices.

3. Medium PLCs:
   Medium PLCs are used in applications controlling 64 up to 1024 input and output devices.

4. Large PLCs:
   Large PLCs are used in applications controlling 512 up to 4096 input and output devices.

5. Very large PLCs:
   Very large PLCs are used in applications controlling 2048 up to 8192 input and output devices.

Overlapping areas in PLC ranges reflect enhancements, by adding options, of the standard features of the PLCs within a particular segment. These options allow a product to be closely matched to the application without having to purchase the next larger unit.

Numbering systems:

Why do we have to learn number systems?
Well...let's think of it this way. If you were visiting a foreign country you would want to learn a little bit of the language that is spoken in that country, like, where is the bus station? Can you direct me to the train station? Think of PLC as a foreign country and the language spoken is numbers.

We must consider the following basics of numbering system:

- Every number system has a base or radix.
- Every system can be used for counting.
- Every system can be used to represent quantities or codes.
- Every system has a set of symbols.

Decimal

The decimal number system is the number system we use everyday. 'Deci' means 10; therefore there are 10 valid numbers, 0 through 9. Decimal is a 'base 10' number system.

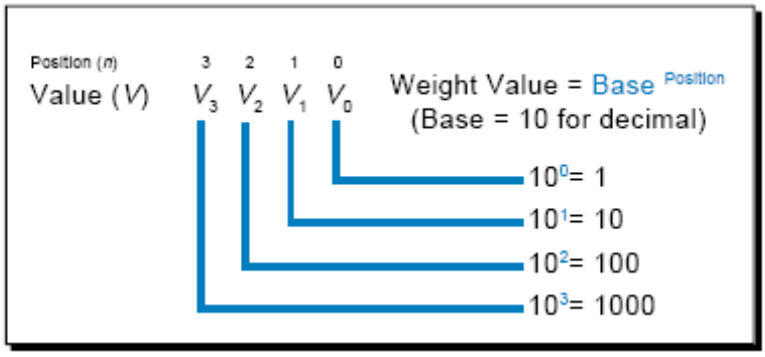


Figure 6

Binary

The binary number system is the number system used by computers, PLC and any other equipment and devices that are digital. 'Bi' means 2; therefore there are 2 valid numbers, 0 and 1. Binary is a 'base 2' number system.

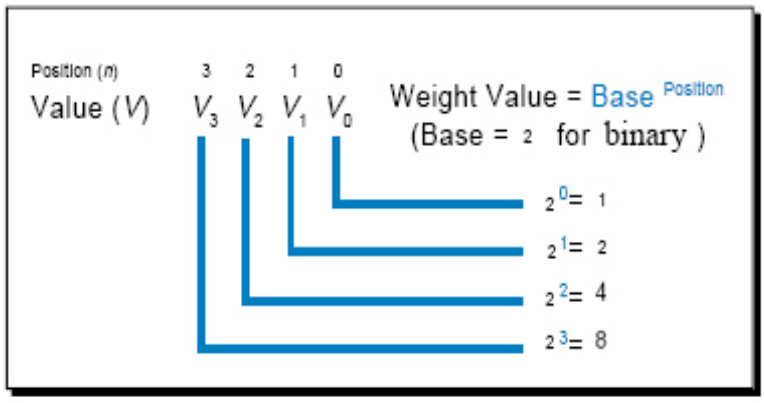


Figure 7

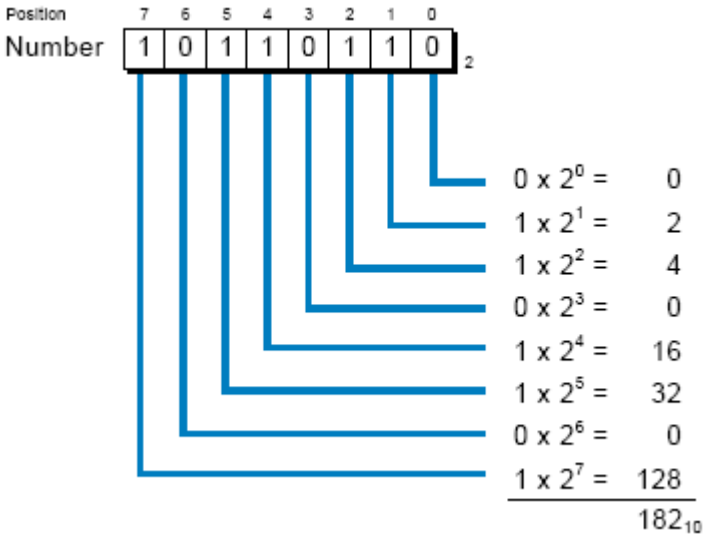


Figure 8

Each digit of a binary number is known as **a bit**, a group of 4 bits is known as a **nibble**; a group of 8 bits is a **byte**; and a group of two bytes is a **word**.

In Figure 9 represent a binary number composed of 16 bits, with the **Least Significant Bit (LSB)**, the lowest valued bit in the word, and the **Most Significant Bit (MSB)**, the largest valued bit in the word.

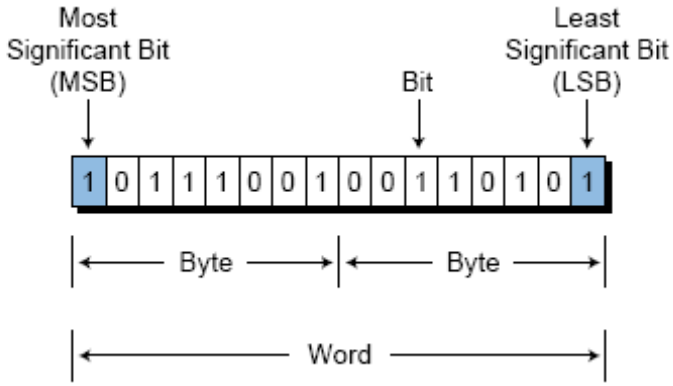


Figure 9

Octal

The octal number system is a number system used by early PLC to represent binary numbers using fewer digits.
'Octa' means 8, therefore there 8 valid numbers, 0 through 7. There are no 8's or 9's in this number system. Octal is a 'base 8' number system.

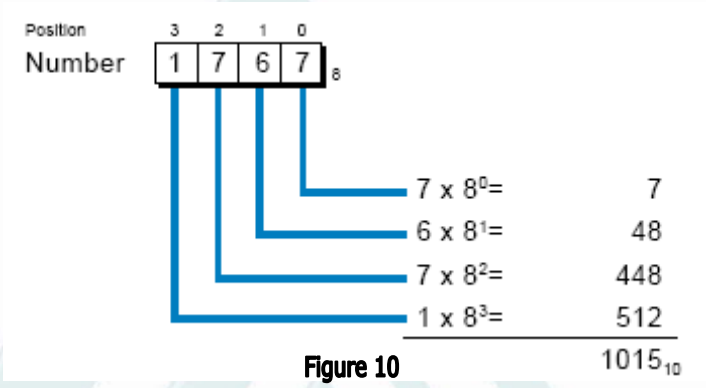


Figure 10

The octal system has a base of 8 (2^3), making it possible to represent any binary number in octal by grouping binary bits in groups of three. In this manner, a very large binary number can be easily represented by an octal number with significantly fewer digits. For example:

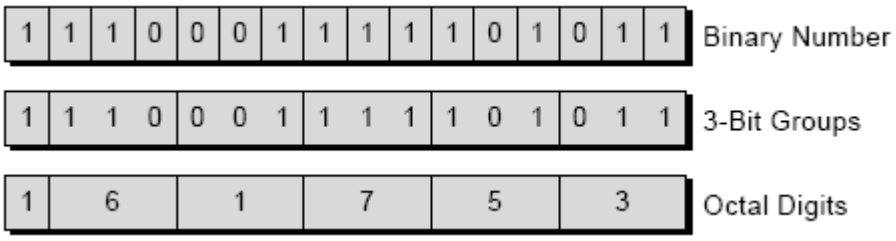


Figure 11

Hexadecimal

The Hexadecimal number system is a number system used by all PLC, computers and other equipment and devices that are digital. 'Hexa' means 16, therefore there are 16 valid numbers, 0 through 9 and A through F. Hexadecimal is a 'base 16' number system.

The hexadecimal system is used for the same reason as the octal system, to express binary numbers using fewer digits. The hexadecimal numbering system uses one digit to represent four binary digits (or bits), instead of three as in the octal system.

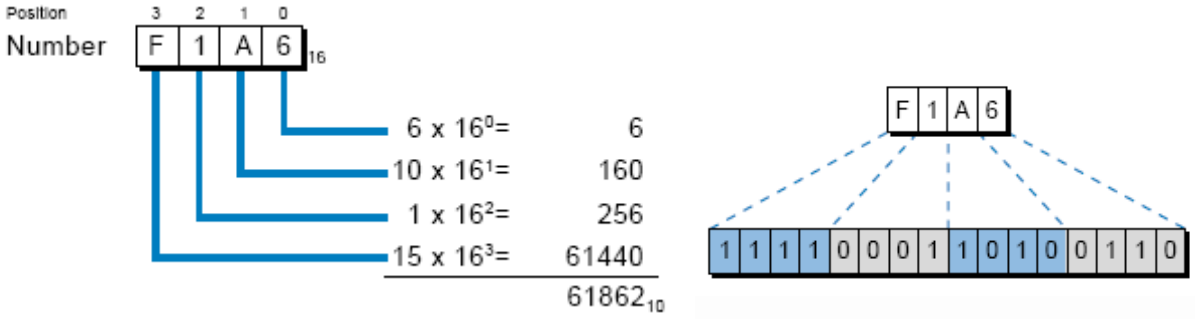


Figure 12

Table 1 shows a hexadecimal count example of the numbers 0 through 15 with their decimal, binary and octal equivalents.

Decimal	Binary	Octal	Hexadecimal
0	0	0	0
1	1	1	1
2	01	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

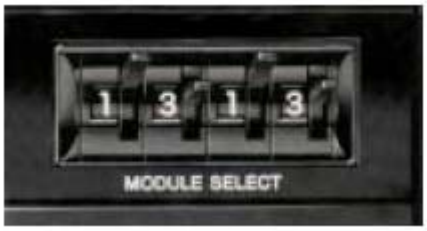
Table 1

Binary Coded Decimal (BCD)

Binary Coded Decimal (BCD) is not really a number system. It is a method to make human machine interface (HMI) easier by representing the decimal number 0 - 9 in a 4-bit binary code. The following example will make it easer to understand.

Decimal	8	9	4	3
BCD	1000	1001	0100	0011

Typical PLC applications of BCD codes include data entry (time, volume, weight, etc.) via thumbwheel switches (TWS) shown on Pic.2, data display via seven segment displays shown on Pic.3.



Pic. 2

Pic. 3

ONE'S AND TWO'S COMPLEMENT

The one's and two's complements of a binary number are operations used by PLC, as well as computers, to perform internal mathematical calculations. To *complement* a binary number means to change it to a negative number, PLC and computers performs only addition operation so it uses the one's and two's complements of a binary number to allow the basic arithmetic operations subtraction, multiplication, and division to be performed through addition operation.

ONE'S COMPLEMENT

The one's complement method used to obtain negative numbers by places an extra bit (sign bit) in the most significant (left-most) position and lets this bit determines whether the number is positive or negative, the number is positive if the sign bit is 0 and negative if the sign bit is 1.

So to negate a number using one's complement obtained by placing a 1 in the most significant bit position and inverting each bit in the number (changing 1s to 0s and 0s to 1s).

Example: see that we need to have negative no. of 23

$23_{10} = (10111)_2$
Convert 1s to 0s and 0s to 1s 01000
First put 1 on MSB to indicate negative sign 101000 = - 23

TWO'S COMPLEMENT

The two's complement is similar to the one's complement in the sense that one extra digit is used to represent the sign. The two's complement computation, however, is slightly different. In the one's complement, all bits are inverted; but in the two's complement, each bit, from right to left, is inverted only after the first 1 is detected.

Example: see that we need to have negative no. of 23

$23_{10} = (10111)_2$
Convert 1s to 0s and 0s to 1s after first 1 from right 01001
First put 1 on MSB to indicate negative sign 101001 = - 23

Logic Functions (Boolean algebra)

Logic function or Boolean algebra was developed in the 1800's by James Bool, an Irish mathematician, It was found to be extremely useful for designing digital circuits, and it is still heavily used by electrical engineers and computer scientists.

Logic functions based on fundamental logic functions AND, OR, and NOT, These functions combine binary variables to form statements. Each function has a rule that determines the statement outcome (TRUE or FALSE) and a symbol that represents it.

AND Function

The AND output is TRUE (1) only if all inputs are TRUE (1) other wise the output is FALSE (0), an AND function can have an unlimited number of inputs, but it can have only one output.

Figure13 and table 2 show the symbol of AND gate and it's truth table.

Truth table: is a table of a logic function listing all possible values the function can achieve due to several combinations of inputs.

Figure 13

AND Truth Table		
Inputs		Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Table 2

HOW AND gate Works

AND logic gate have two inputs consists of two transistors in series as shown in figure 14.

The use of transistors for the construction of logic gates depends upon their utility as fast switches. When the base-emitter diode is turned on enough to be driven into saturation for the two transistors, the collector voltage with respect to the emitter may be near zero and output point will have the same voltage of Vcc point.

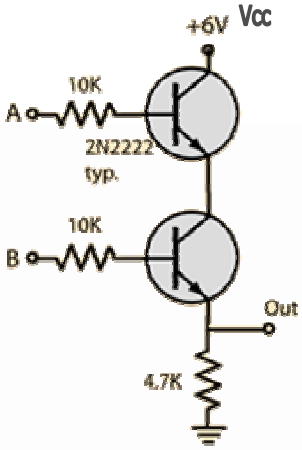


Figure 14

Example :

Show the logic gate, truth table, and circuit representations for an alarm horn that will sound if its two inputs, push buttons PB1 and PB2, are 1 (ON or depressed) at the same time.

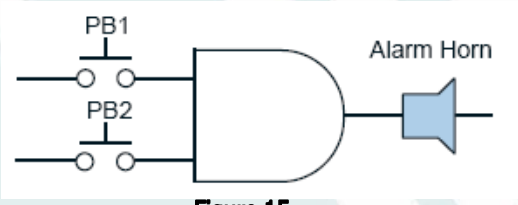


Figure 15

PB1	PB2	Alarm Horn
Not pushed (0)	Not pushed (0)	Silent (0)
Not pushed (0)	Pushed (1)	Silent (0)
Pushed (1)	Not pushed (0)	Silent (0)
Pushed (1)	Pushed (1)	Sounding (1)

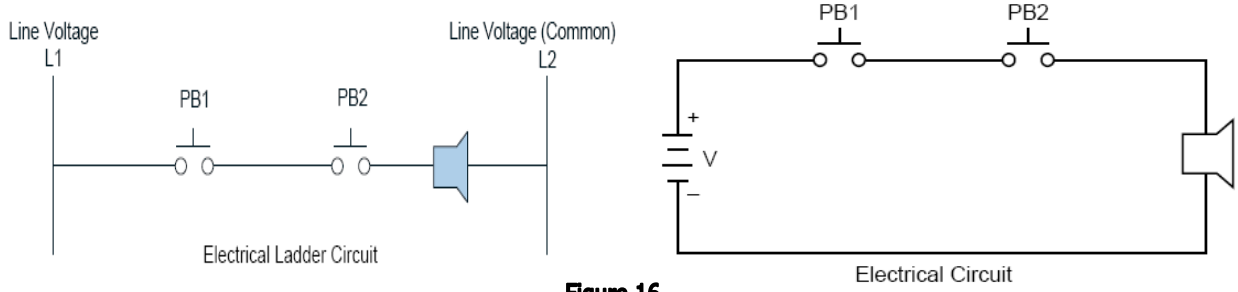


Figure 16

OR Function

The OR output is TRUE (1) if one or more inputs are TRUE (1), as with the AND function, an OR gate function can have an unlimited number of inputs but only one output.
Figure17 and table 4 show the symbol of AND gate and it's truth table.

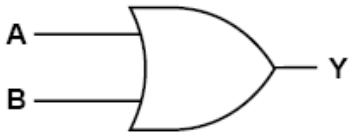


Figure 17

OR Truth Table		
Inputs		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Table 4

HOW OR gate Works

OR logic gate have two inputs consists of two transistors in parallel as shown in figure 18.

The use of transistors for the construction of logic gates depends upon their utility as fast switches. When the base-emitter diode is turned on enough to be driven into saturation for any transistors, the collector voltage with respect to the emitter may be near zero and output point will have the same voltage of Vcc point.

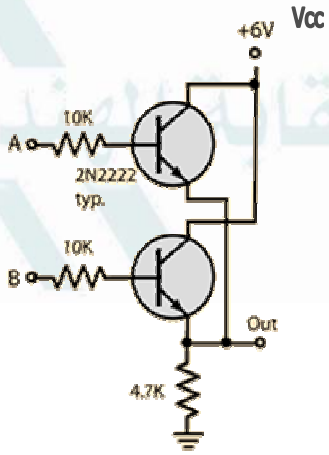


Figure 18

Example:

Show the logic gate, truth table, and circuit representations for an alarm horn that will sound if either of its inputs, push button PB1 or PB2, is 1 (ON or depressed).

Figure 19

PB1	PB2	Alarm Horn
Not pushed (0)	Not pushed (0)	Silent (0)
Not pushed (0)	Pushed (1)	Sounding (1)
Pushed (1)	Not pushed (0)	Sounding (1)
Pushed (1)	Pushed (1)	Sounding (1)

Table 5

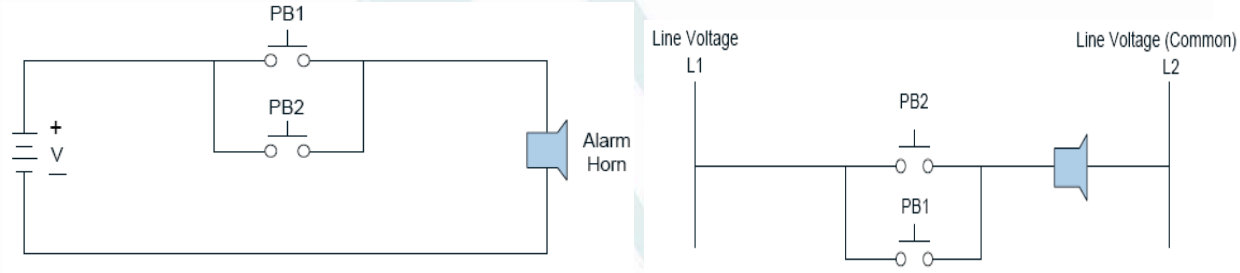


Figure 20

NOT Function

NOT operation is always the inverse of the input; therefore, it is some times called an inverter.

The NOT output is TRUE (1) if the input is FALSE and the output is FALSE (0) if the input is TRUE (1).

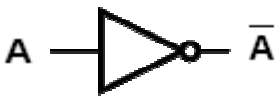


Figure 21

NOT Truth Table	
Input	Output
A	$\overline{A}$
0	1
1	0

Table 6

HOW NOT gate Works

A transistor switch with collector resistor can serve as an inverting buffer

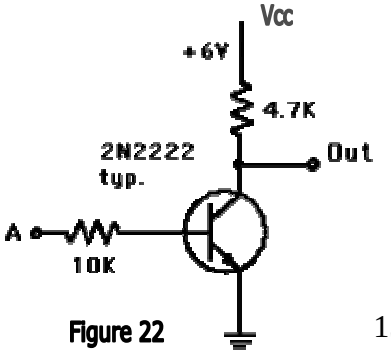


Figure 22

XOR Function

XOR gate is short for exclusive OR (figure 23), is a digital logic gate that implements exclusive disjunction means that it's output is TRUE when the inputs is different and the output is FLASE when the inputs are the same (shown by table 7)

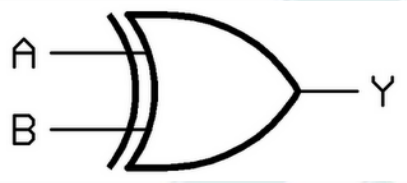


Figure 23

XOR Truth Table		
Inputs		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Table 7

Example:

Show the logic gate, truth table for a solenoid valve (V1) that will be open (ON) if selector switch S1 is ON and if level switch L1 is NOT ON (liquid has not reached level).

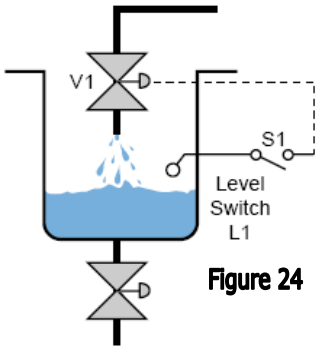


Figure 24

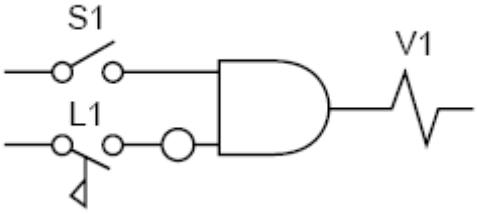


Figure 25

S1	L1 ($\overline{L1}$)	V1
0	0 1	0
0	1 0	0
1	0 1	1
1	1 0	0

Table 8

PLC architecture

As we mentioned before PLC is consists of:

The central processing unit: The central processing unit (CPU) governs all PLC activities.

The input/ output system: is the **interface** by which field devices are connected to the controller and the main purpose of the interface is to condition the various signals received (Inputs like push buttons, switches, and measurement devices) or sent to (outputs Lights, horns, motors, and valves) external field devices.

CPU consists of three components as shown in figure 26:

- Processor.
- Memory system.
- Power supply.

Some times power supply is an individual unit.

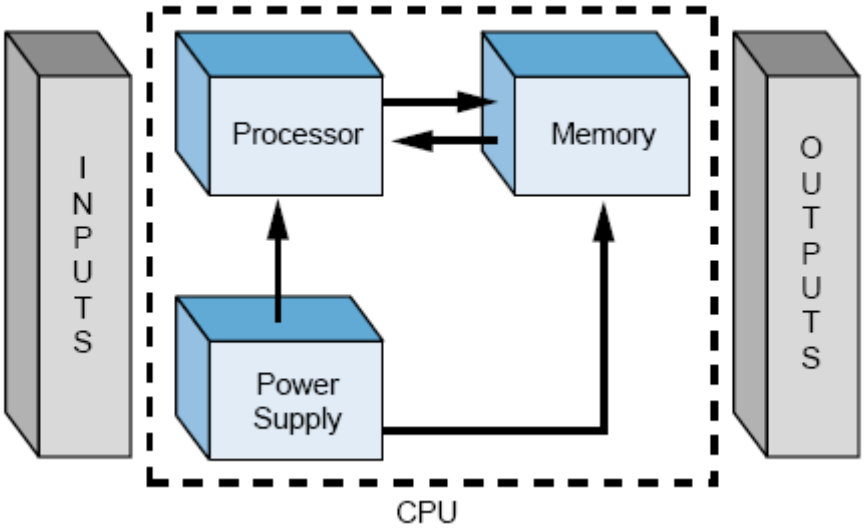


Figure 26

Processor

The principal function of the processor is to command and govern the activities of the entire system. It performs this function by interpreting and executing a collection of system programs known as the executive. The executive, a group of supervisory programs, is permanently stored in the processor and is considered a part of the controller itself. By executing the executive, the processor can perform all of its control, processing, communication, and other functions.

The CPU of a PLC system may contain more than one processor to execute the system's duties and/or communications, because extra processors increase the speed of these operations. This approach of using several microprocessors to divide control and communication tasks is known as multiprocessing.

Memory

The total memory system in a PLC is actually composed of two different memories (see Figure 27):

- Executive memory
- Application memory

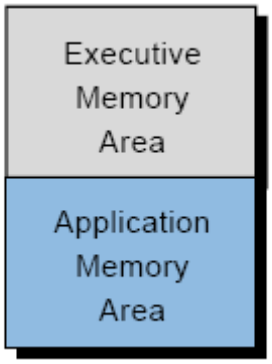


Figure 27

Executive memory: is a collection of permanently stored programs such as execution of the control program and communication with peripheral devices.

Application memory: provides a storage area for the user-programmed instructions that form the application program.

The executive and application memory sections are not the same; they are not always stored in the same type of memory.

For example, the executive requires a memory that permanently stores its contents and cannot be erased or altered either by loss of electrical power or by the user.

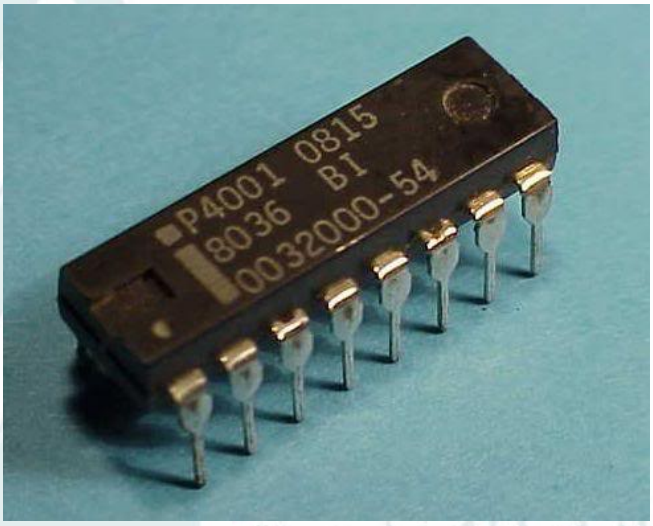
Now we will describe six type of memory:

Read-only memory (ROM)

Is designed to permanently store a fixed program, once the manufacturer programs the original set of instructions on ROM, the user can never change it.

Programmable read-only memory (PROM)

Is a special type of ROM because it can be programmed, although a PROM is programmable and, like any other ROM, has the advantage of nonvolatility, it has the disadvantage of requiring special programming equipment.



Pic. 4

Erasable programmable read-only memory (EPROM)

Is a specially designed PROM that can be reprogrammed after being entirely erased by using an ultraviolet (UV) light source.

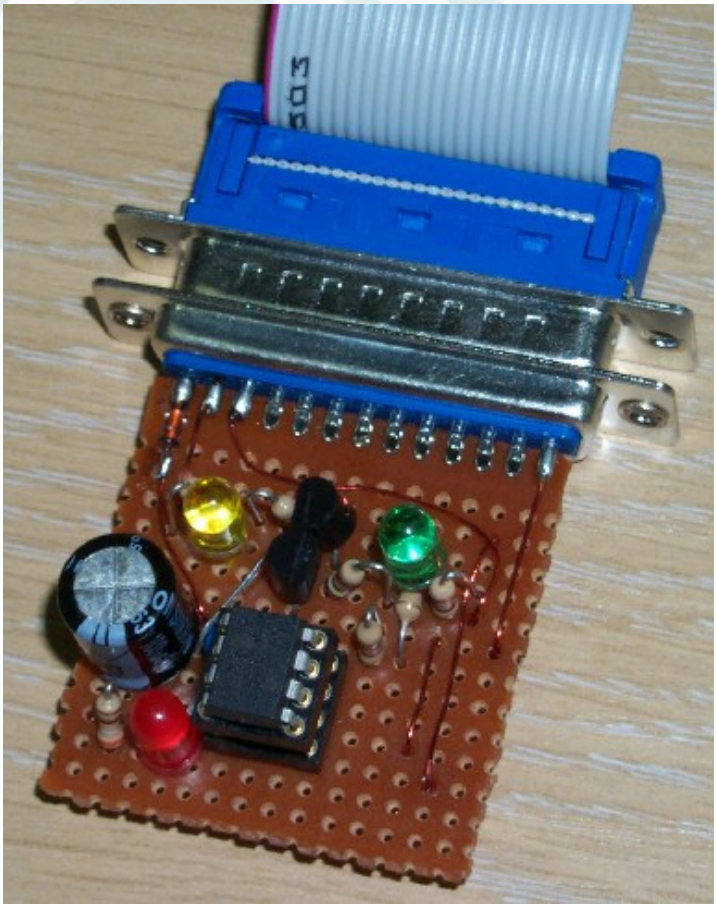
Pic. 5

. **Electrically alterable read-only memory (EAROM)**

Is similar to EPROM, but instead of requiring an ultraviolet light source to erase it, an erasing voltage on the proper pin of an EAROM chip can wipe the chip clean.

Electrically erasable programmable read-only memory (EEPROM)

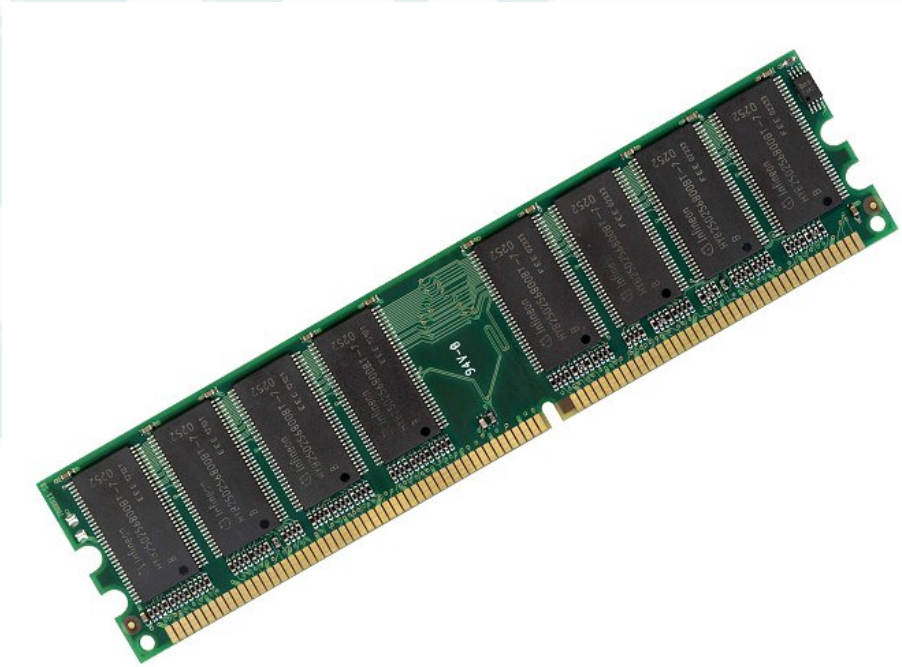
Is an integrated circuit memory storage device that was developed in the mid- 1970s. Like ROMs and EPROMs, it is a nonvolatile memory, yet it offers the same programming flexibility as RAM does, It provides permanent storage for the program and can be easily changed with the use of a programming device (e.g., a PC) or a manual programming unit. These two features help to eliminate downtime and delays associated with programming changes.



Pic. 6

Random-access memory (RAM)

Often referred to as *read/write memory (R/W)* is designed so that information can be written into or read from the memory storage area. Random-access memory does not retain its contents if power is lost.
Random-access memory normally uses a battery backup to sustain its contents in the event of a power outage.



Pic. 7

Power supply

The system power supply plays a major role in the total system operation. In fact, it can be considered the “first-line manager” of system reliability and integrity. Its responsibility is not only to provide internal DC voltages to the system components (i.e., processor, memory, and input/output interfaces), but also to monitor and regulate the supplied voltages and warn the CPU if something is wrong. The power supply, then, has the function of supplying well-regulated power and protection

Pic. 7

PLC Programming Languages

Programming languages used to program PLC categorized on three groups:

Graphical languages
- Ladder Diagrams (LD)
- Function Block Diagram (FBD)

Text-based languages
- Instruction List (IL)
- Structured Text (ST)

Flow chart
- Sequential Flow Chart (SFC)

Ladder Diagram Language (LD)

This language is called with this name because they resemble a ladder, it composed by two vertical lines called Rail and horizontal lines each line called Rung as shown in figure 28.

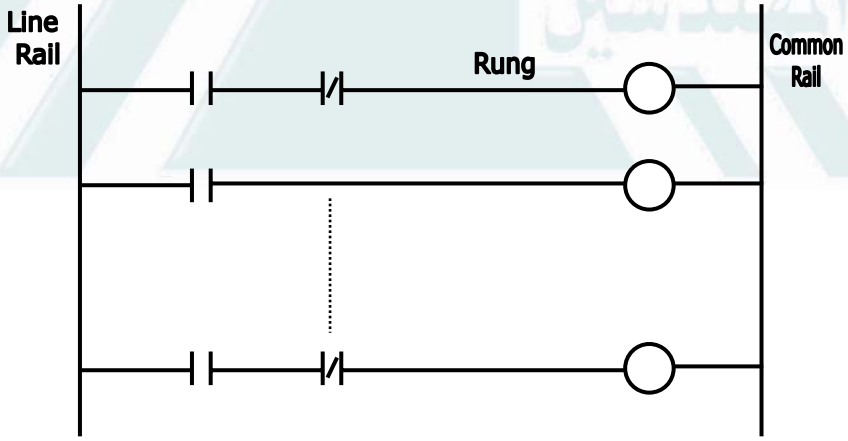


Figure 28

Rung consists of contact symbols and coil symbols, contact represent input signal to the rung and coil represent outputs of the rung.

The ladder logic will be interpreted left-to-right, top-to-bottom, the time of execute the rung is very small so that it seems that all rung execute at the same time.

Ladder diagram could be considered as electrical circuits because it has line rail and common rail and coil is energized when all contacts of the rung met the condition of being closed so power flow from Line rail to common rail as shown in figure 29.

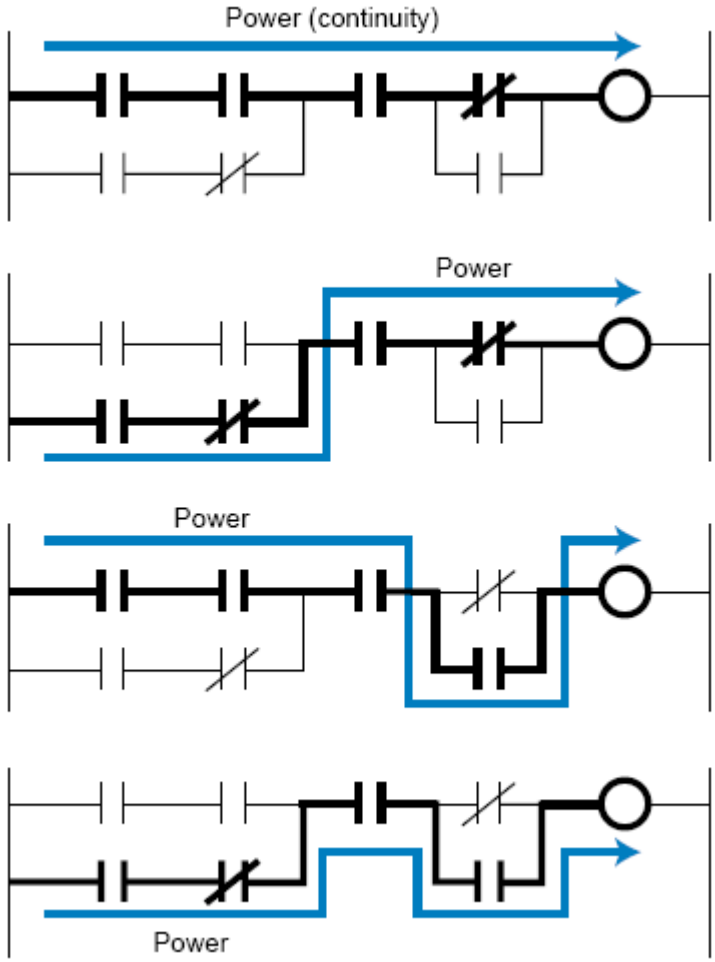


Figure 29

Symbols of Ladder Diagram

Normally open contact:

When de energized performs as open circuit and doesn't pass power, when energized performs as closed circuit and pass power.

Normally closed contact:

When de energized performs as open circuit and doesn't pass power, when energized performs as closed circuit and pass power.

Coil 

If any left-to-right path of contacts passes power to coil it will be energized (TRUE) and if no continues path of contacts passes power to coil it will be de energized (FALSE).

NOT coil 

If any left-to-right path of contacts passes power to coil it will be de energized (FALSE) and if no continues path of contacts passes power to coil it will be energized (TRUE).

Latched coil 

If any left-to-right path of contacts passes power to coil it will be energized and remain energized.

Unlatched coil 

If any left-to-right path of contacts passes power to coil will unlatch a latched coil with the same address.

Reset Coil

If any left-to-right path of contacts passes power to coil will reset coil with the same address.

Examples

Write Ladder program for car ignition safety system which make ignition for car motor if the Door is closed (indicated by door NO Switch) and seat belt (indicated by belt NO Switch) is inserted and car key is turned on.

DSW: Door switch ON (door closed) OFF (door open)
BSW: Belt switch ON (belt closed) OFF (belt open)
KSW: Key switch ON (key turned ON) OFF (key turned OFF)
Ignition: The output

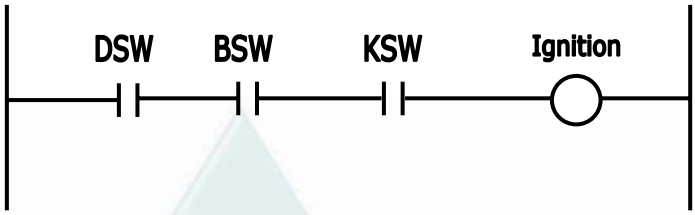


Figure 29

Examples

Write Ladder program for motor with two push buttons one push button when pressed motor start rotate and the other push button when pressed the motor start rotate in reverse direction.

PBF: Door switch ON (door closed) OFF (door open)
PBR: Belt switch ON (belt closed) OFF (belt open)
Motor Forward: The output
Motor Reverse: The output

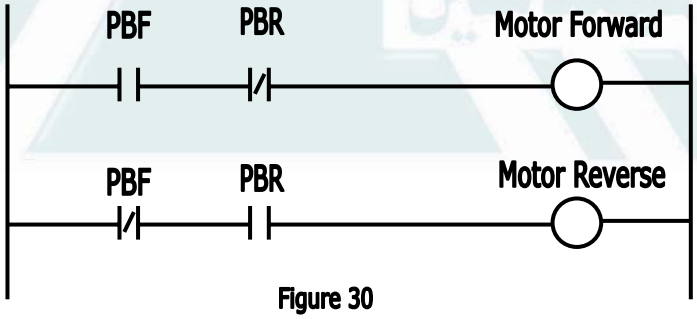


Figure 30

Examples

Write Ladder program for motor with three push buttons one push button for start signal, one push button for speed 1 and one push button for speed 2, the motor works only when start push button pushed and one of the two speed push buttons pushed.

Start: push button for start
L1: push button for speed 1
L2: push button for speed 2
Motor: output

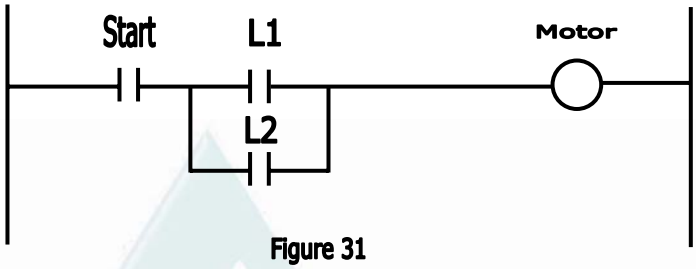


Figure 31

Timer Block

Timers activate or deactivate an output after a time interval has expired.

Timers may have one or more *time bases* (TB) which they use to time an event. The time base is the resolution, or accuracy, of the timer. For instance, if a timer must time a 10 second event, the user must choose the number of times the time base must be counted to get to 10 seconds.

Therefore, if the timer has a time base of 1 second, then the timer must count ten times before it activates its output. This number of counts is referred to as *ticks*.

Timers are used in applications to add a specific amount of delay to an output.

There three types of timers:

1. On-Delay Timer 2. Off-Delay Timer 3. Retentive Timer

1. On-Delay Timer:

In this type when input is on it waits a predefined interval of time and turn its output on.

The timer could have one or two inputs depend on the PLC type.

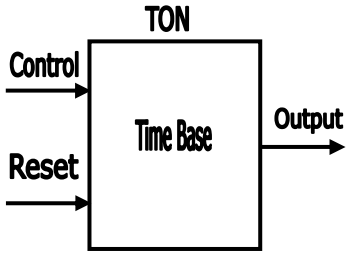


Figure 32

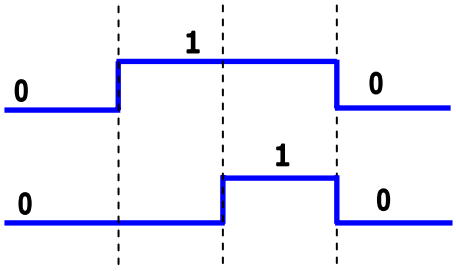


Figure 34

Figure 33

Example: write a ladder program to make a motor rotate after start push button pressed for 10 seconds.

Start: motor start push button.
TON: On delay timer with time base 10 S.
Motor: the output.

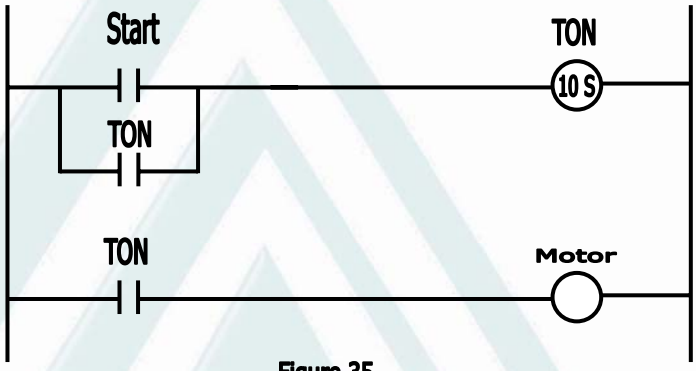


Figure 35

Here we use the output of the Timer as internal flag and as a latch, also make interlock between Rung 1 and Rung 2.

Internal Flag: means that the value of TON output stored on memory with another address and used as input

Interlock: we use output of TON in Rung 1 as input in Rung 2, so rung 2 will not execute until the output of TON (on Rung1)is true.

Latch: a way to keep output activated after inputs became off.

2. Off-Daly Timer:

In this type when input is on it waits a predefined interval of time and turn its output off.

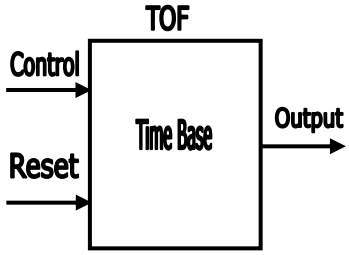


Figure 36

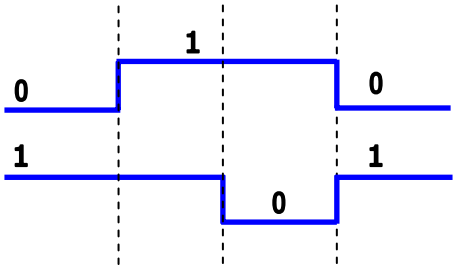


Figure 37

Example: write the part of ladder program to make a motor stop rotating after 5 seconds when stop push button pressed for 5 seconds.

Start: motor start push button.
TOF: Off delay timer with time base 10 S.
Motor: the output

Figure 38

Example: write a ladder program to make a motor rotate after start push button pressed for 5 seconds and make a motor stop rotating after 5 sec since stop push button pressed.

Start: motor start push button.
TON: On delay timer with time base 5 S.
TOF: Off delay timer with time base 5 S.
Motor: the output.

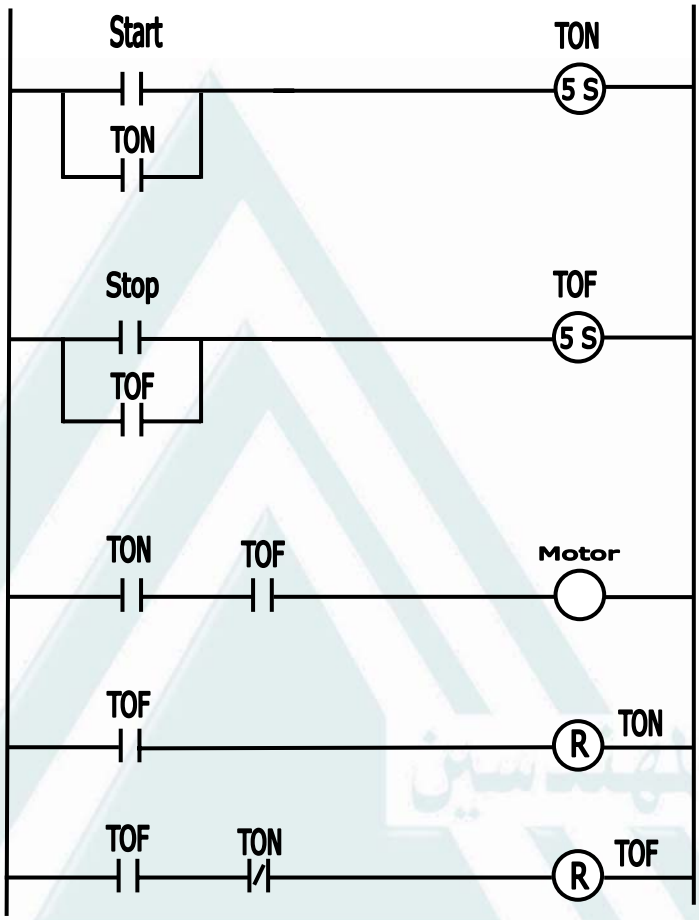


Figure 39

3. Retentive Timer:

Retentive or Accumulating timer- holds or retains the current elapsed time when the sensor turns off in mid-stream. It is called RTO or TMRA.

Typical applications for retentive timers include tracking the time before maintenance is needed.

This type of timer needs 2 inputs one for timing and one for reset.

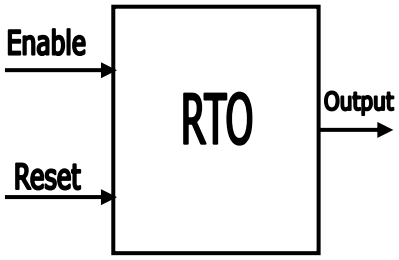


Figure 40

Counter Block

Counters compare an accumulated value to a preset value to control circuit functions.

There are two types of counters:

Count Up counter (CTU) Count Down counter (CTD)

There are types of PLC have a type of counter that have both function of CTU and CTD.

Count Up counter (CTU)

The Count Up Counter (CTU) counts up from the current value each time the count up input goes from off to on, when the current value is greater than or equal to the preset value , the counter output goes on, The counter resets when the reset input turns on.

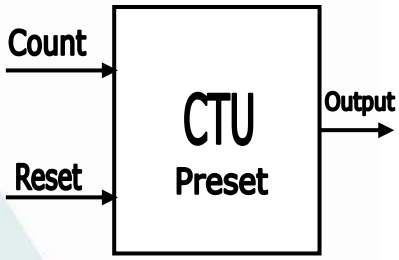


Figure 40

Example: write a ladder program used to count parts as detected by a photoelectric cell input. The preset value of counts is 5, Also add the instructions necessary to implement an output coil that indicates that the count has reached 5 and make the system when reach 5 resets and start count again.

PC: photoelectric cell input
CTU: Count Up counter with preset of 5
Output coil: have the same output of CTU
Use the value of output as internal flag to reset CTU

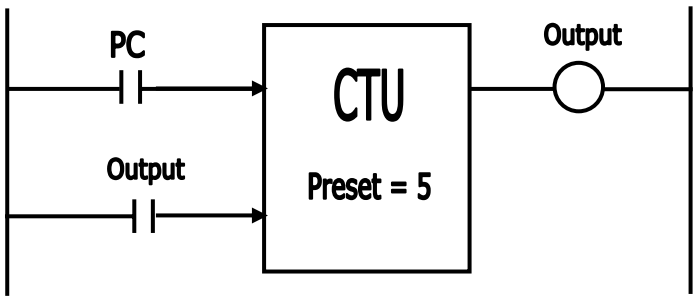


Figure 41

Count Down counter (CTD)

Counts down from the current value each time the count down input goes from off to on, when the current value is equal to zero, the counter output goes on, the counter stops counting at zero The counter resets and loads the current value with the preset value when the reset input turns on.

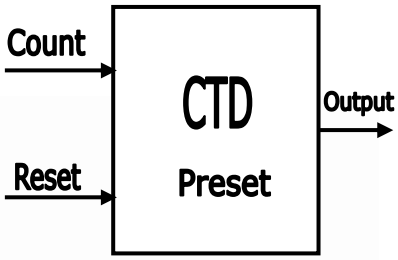


Figure 42

Example: write a ladder program used to lock a gate barrier when five persons pass the gate detected by a photoelectric cell input and make the system when reach 5 resets and start count again.

PC: photoelectric cell input
CTD: Count Up counter with preset of 5
Output: order to barrier to close.
Use the value of output as internal flag to reset CTU

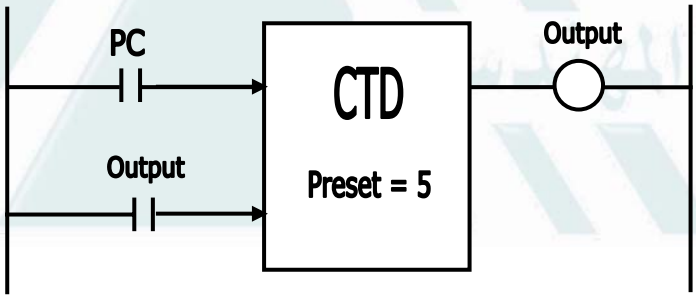


Figure 42

FUNCTION BLOCK DIAGRAM (FBD)

Is a graphical language that allows the user to program elements (e.g., PLC function blocks) in such a way that they appear to be wired together like electrical circuits.

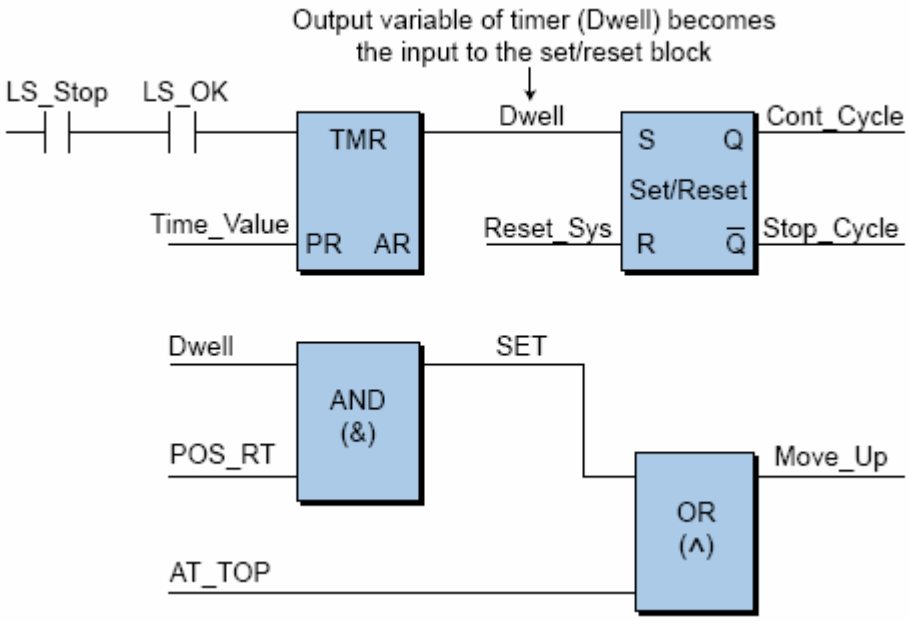


Figure 43

Instruction list (IL)

Is a low-level language similar to the machine or assembly language used with microprocessors (see Figure 10-10). This type of language is useful for small applications, as well as applications that require speed optimization of the program or a specific routine in the program, IL can be used to create custom function blocks.

Instructions		Comments
LD	b1	(*current result:=TRUE*)
AND	b2	(*current result:=b1 AND b2*)
ANDN	b3	(*current result:=b1 AND b2 AND NOT b3*)
ST	b0	(*b0:=current result*)

Structured text (ST)

Is a high-level language that allows structured programming, meaning that many complex tasks can be broken down into smaller ones. ST resembles a BASIC- or PASCAL-type computer language (see Figure 44), which uses subroutines to perform different parts of the control function and passes parameters and values between the different sections of the program.

```
IF Manual AND NOT Alarm THEN
                Level:=Manual_Level;
                Mixer:=Start AND NOT Reset
ELSE_IF         Other_Mode THEN
                Level:=Max_Level;
ELSE            Level:=(Level_Indic × 100)/Scale;
END_IF;
```

Figure 44

Sequential Function Chart (SFC)

Is a flowchart-like framework that can organize the subprograms or subroutines (programmed in LD, FBD, IL, and/or ST) that form the control program, SFC is particularly useful for sequential control operations, where a program flows from one step to another once a condition has been satisfied (TRUE or FALSE).

The SFC programming framework contains three main elements that organize the control program:

- Steps
- Transitions
- Actions

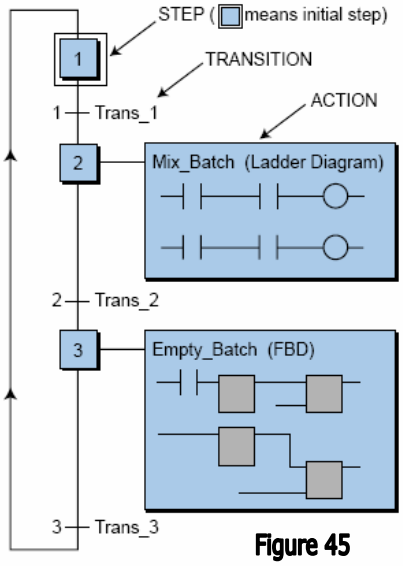


Figure 45

PLC utilization in industrial environment and practical live

Industrial environment

Suppose that there is a distribution switch board have two feeders A and B , each feeder connected through breakers and there is a bus tie breaker between A and B just in case one feeder is lost and other is healthy, we use PLC to make automatic transfer between feeders in such case and the conditions for transfer is:

1. The feeders that loads will be transferred is Healthy (Not trip input from feeder's breaker).
2. The breaker of bus tie is open and inserted (inputs from feeder's breaker).

We will write a PLC program for this case.

نقابة المهندسين

Commercial environment

PLC Control for Rotary Bottle Washing Machine

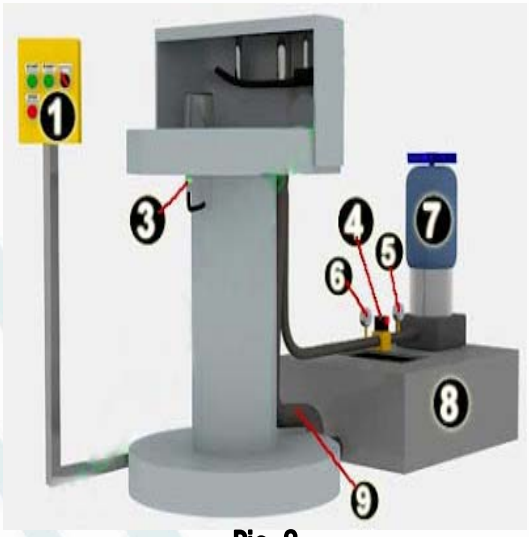
Cycle Process:

1. Process 1:
   RUN the turntable to the next point.

2. Process 2:
   The turntable is Running, Proximity Sensor ON.

3. Process 3:
   If Proximity Sensor OFF Then turntable STOP.

4. Process 4:
   Delay for stability of the turntable.



Pic. 8

5. Process 5:
   IF Jet Pump Motor = ON THEN go to next process ELSE 1 cycle process STOP.

6. Process 6:
   Solenoid Valve = ON.

7. Process 7:
   Delay for the water spray ON.

8. Process 8:
   Solenoid Valve = OFF, and Delay for the water spray OFF.

1. Control Panel box

a. Push Button (Normally Open Contact) for START
b. Push Button (Normally Open Contact) for START 1 Cycle
c. Selector Switch (2 position) for Water Spray OFF/ON
d. Push Button (Normally Open Contact) for STOP

2. Turntable consists of:

10 rack holes in Turntable
Electric Motor with Reduction Gearbox

If necessary use electric motor with brake
Coupling for Motor Shaft and Turntable Shaft
Contactor with Contactor Thermal Overload Relay (TOR)

3. Proximity Sensor.

4. Solenoid Valve Normally Closed.

5. Inlet Water Gauge Pressure.

6. Outlet Water Gauge Pressure.

7. Water Jet Pump Motor with Safety CUT-OUT, and consists of : Contactor with Contactor Thermal Overload Relay (TOR).

8. Water Reservoir.

9. Drainage Pipe

PLC Input and Output Devices :

1. PLC Input :

1 Input for START
1 Input for Water Spray OFF/ON
1 Input for STOP
1 Input for Proximity Sensor

Total PLC Input, minimum of 4 Input.

2. PLC Output:

1 Output for Contactor to Electric Motor Turntable

1 Output for Contactor to Water Jet Pump Motor

1 Output for Solenoid Valve

Total PLC output, minimum of 3 Output.

We will write a PLC program for this case.

نقابة المهندسين