PART-2

UNIT-5: PERIPHERALS

5.1 EXPLAIN WATCHDOG TIMERS, LCD CONTROLLERS,

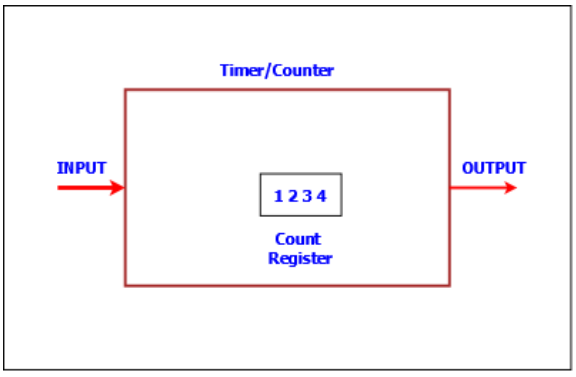
- The 8051 has two counters/timers which can be used either as timer to generate a time delay or as counter to count events happening outside the microcontroller.

TIMERS:

- Timer is a very common and useful peripheral.
- It is used to generate events at specific times or measures the duration of specific events which are external to the processor.
- It is a programmable device, i.e. the time period can be adjusted by writing specific bit patterns to some of the registers called timer-control registers.
- A timer measures time by counting pulses that occur on an input clock signal having a known period.

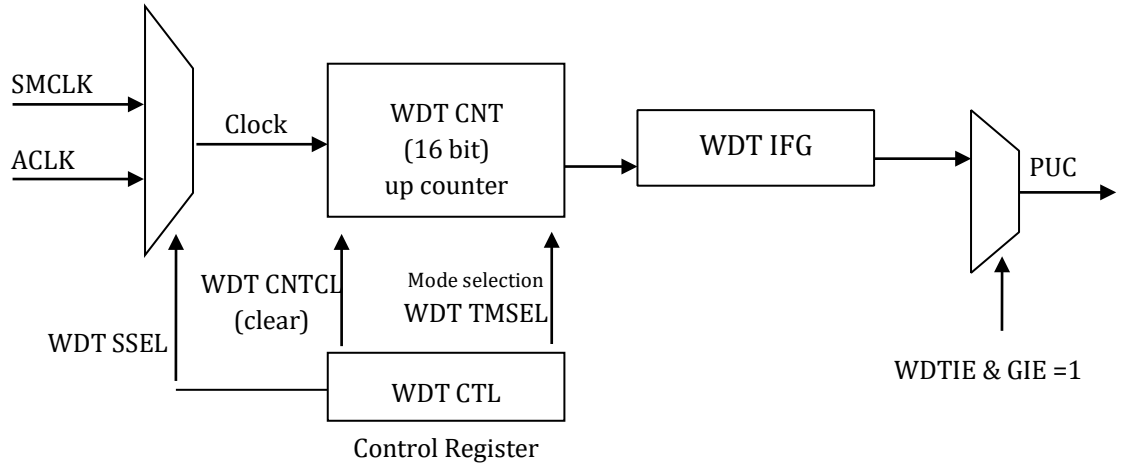
COUNTERS:

- A counter is nearly identical to a timer except that instead of counting the clock cycle, a counter counts the number of pulses of input signal.
- It can be free running device with a clock input pulse and for comparing the counts with one which is preloaded in the register.
- This device is used for the alarm and the other processors.
- It is useful for the processor interrupt at the preset time.



WATCHDOG TIMER:

- The watchdog timer is a simple electronic device that is responsible to resetting the microcontroller or microprocessor for invalid software status.
- Generally a microcontroller is programmed with software that contains several loops and number of subroutines that direct variety of activities.
- If any reason, if the loop is fail to execute, then it finds and resets the device at starting or top of the loop.
- A watchdog timer (WDT; sometimes called a computer operating properly or COP timer, or simply a watchdog) is an electronic timer that is used to detect and recover from computer malfunctions.
- During normal operation, the computer regularly restarts the watchdog timer to prevent it from elapsing, or "timing out".
- If, due to a hardware fault or program error, the computer fails to restart the watchdog, the timer will elapse and generate a timeout signal.
- The timeout signal is used to initiate corrective action or actions. The corrective actions typically include placing the computer system in a safe state and restoring normal system operation.
- Watchdog timers are commonly found in embedded systems and other computer controlled equipment where humans cannot easily access the equipment or would be unable to react to faults in a timely manner.
- In such systems, the computer cannot depend on a human to reboot it if it hangs; it must be self-reliant.

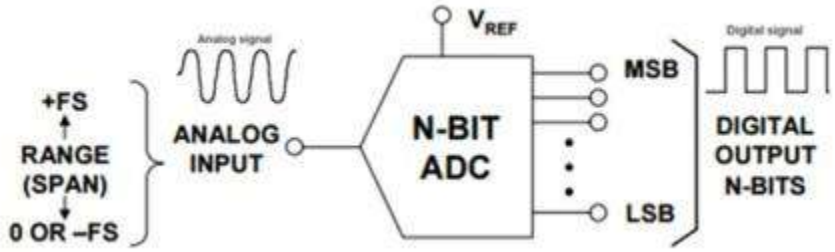


LCD CONTROLLERS:

- A liquid crystal display (LCD) is a low cost, low power device capable of displaying text and images.
- LCDs are extremely common in embedded systems.
- LCDs can be found in numerous common devices like watches, fax and copy machines and calculators.
- Each type of LCD may be able to display multiple characters.
- The LCD may permit a character to be blinking or may permit display of a cursor indicating the "current" character. Such functionality would be difficult for us to implement using software.
- Thus we use an LCD controller to provide us with a simple interface to an LCD with eight data input and one enable input.
- To send a byte to the LCD, we provide a value to the eight inputs and one enable input.
- This byte may be a control word, which instructs the LCD controller to initialize the LCD, clear the display, select the position of the cursor, and brighten the display and so on.

5.2 ANALOG TO DIGITAL CONVERTER (ADC)

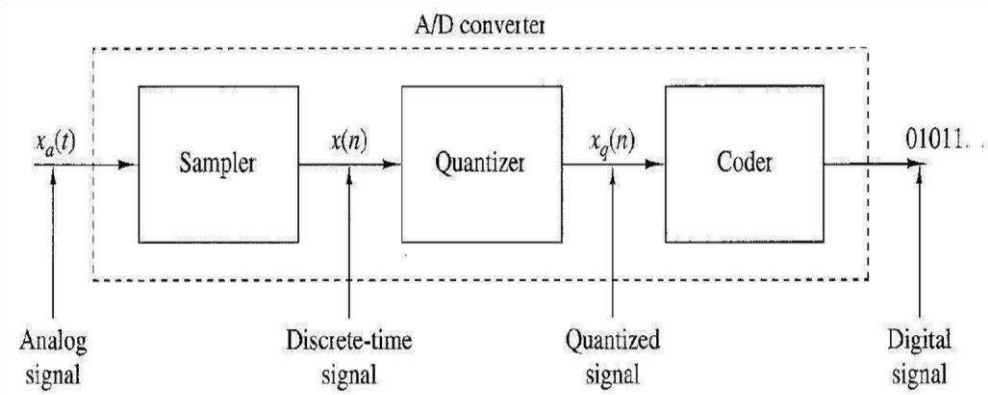
- An analog to digital converter (ADC) is an electronic device which converts varying analog signals into digital signals so that they can easily be read by the digital devices.
- It have many applications in electronics projects. ADC converts the quantities of real world phenomenon in to digital language which is used in control systems, data computing, data transmission and information processing.
- Figure below shows the input/ output relationship of an ADC.



- Usually transducers are also used to convert the input analog variables in the form of currents or voltages.
- Basically the digital numbers used here are binary i.e. '0' and '1'. The '0' indicates the 'off' state and '1' represents the 'on' state.
- Hence all the analog values are converted into digital binary values by an ADC.

- For example, if we have to install an alarm in our house or at some facility, whose function is to set off in case of fire or overheating. Our whole alarm system will be electronic but the temperature sensor will give analog values at the output after sensing the temperature. Therefore to convert the varying values of temperature in digital or discrete values, we have to use an analog to digital converter.

Block diagram of ADC:

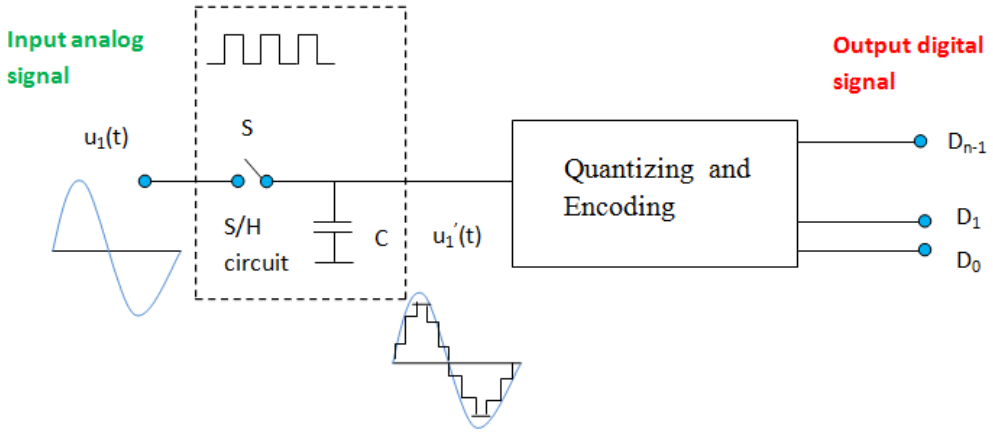


Basic parts of an analog-to-digital (A/D) converter

ADC Process

There are mainly two steps involves in the process of conversion. They are

- Sampling and Holding (S/H)
- Quantizing and Encoding (Q/E)



Sampling:

- It is the process in which a continuous time signals or analog signal is converted into discrete time signal. This is done by the device sampler.

Quantization:

- It is a process as in which a discrete time signal is converted into quantized signal.
- It can also defined as the process of conversion from continuous value to discrete value. This process is done by the device called quantizer.

Coding:

- Coding is the process in which the discrete valued signal is converted into corresponding binary form.
- Analog always refers to continuous valued signal and digital always refers to discrete valued signal.
- By converting from analog to digital we use a device called digital processor to compute the analog value to digital value. For this the formula is $\mathbf{e/V_{max} = d / (2^n - 1)}$

  Where V_{max} = Maximum voltage of analog signal

  n = no. of bits available for digital encoding

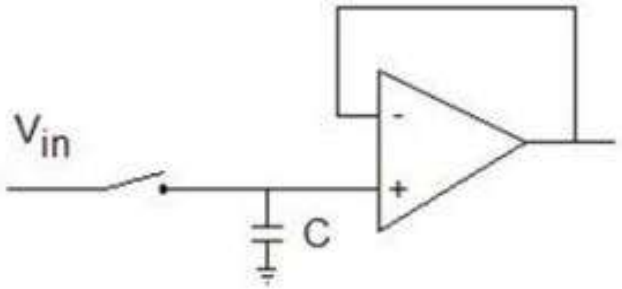
  d = the present digital encoding

  e = the present analog voltage

- The resolution of the digital signal can be found by the formula $\mathbf{V_{max} / (2^n - 1).}$

Sampling and Holding

- An analog signal continuously changes with time, in order to measure the signal we have to keep it steady for a short duration so that it can be sampled.
- We could measure the signal repeatedly and very fast, and then find out the right time scale.
- Or we could measure the signal at different timings and then average it. Or preferably we can hold the signal for a specific duration and then digitize the signal and sample the value.
- This is done by a sample and hold circuit. For, at least the time required for digitization, it keeps the value stable.
- Figure shows the circuit for sampling and holding of a signal.

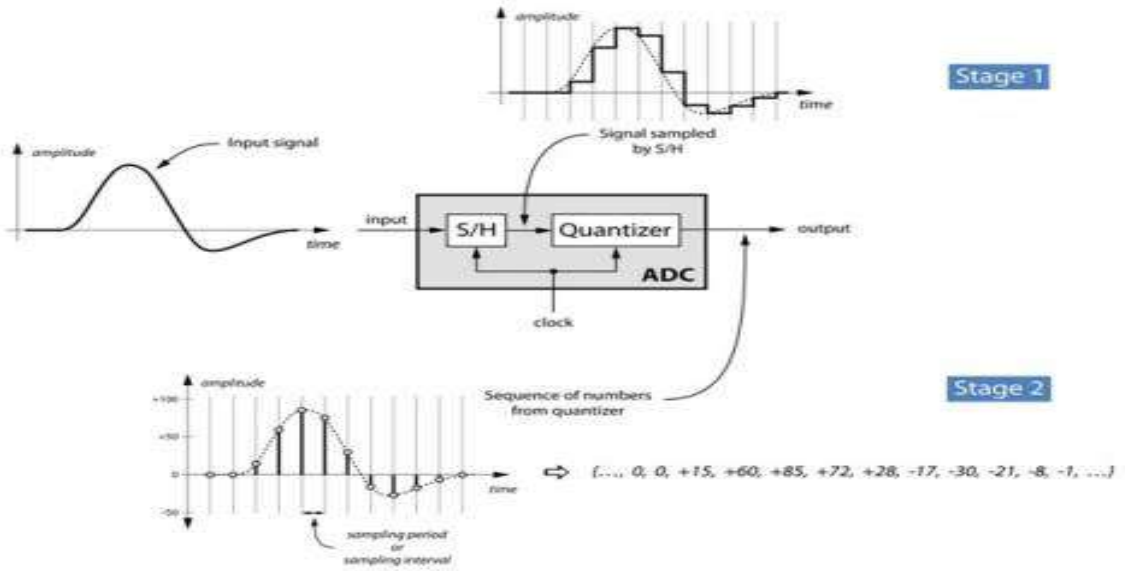


Sampling and holding circuit

- We keep the switch normally open, and when we want to find a measurement, we close the switch momentarily.

Quantizing and Encoding

- On the output of (S/H), a certain voltage level is present. We assign a numerical value to it.
- The nearest value, in correspondence with the amplitude of sampling and holding signal, is searched. And this value cannot be just any value, it should be from a limited set of possible values.
- It depends on the range of the quantizer and the range in given in a power of 2 i.e. 2^n ($2^8 = 256$, $2^{10}=1024$ etc.)
- After identifying the closest value, a numerical value is assigned to it and it is encoded in the form of a binary number.
- The binary encoded numbers generated by quantizer are represented by 'n' bits.
- The resolution of an ADC can also be denoted by 'n' bit.
- The figure shows the whole conversion process:



Sampling, Holding and Quantizing

- The values achieved after the quantization and encoding process cannot be said to be thoroughly accurate. These are only the approximations of the real world values.
- The accuracy of the quantizer highly depends on the resolution of the qunatizer, greater the resolution, more accurate the values will be.
- The ADC resolution is limited by a number of constraints, out of which, time is a major issue.
- If the set of possible values, from which the closest value is to be searched, is greater, then it will surely take more time. But to accelerate this process, more techniques have been developed.
- The process of quantizing and encoding is demonstrated in the table below.

Analog signal			Digital o/p
7.5	7	7Δ=7V	111
6.5	6	6Δ=6V	110
5.5	5	5Δ=5V	101
4.5	4	4Δ=4V	100
3.5	3	3Δ=3V	011
2.5	2	2Δ=2V	010
1.5	1	1Δ=1V	001
0.5	0	0Δ=0V	000

- From the above table we can observe that only one digital value is used to represent the whole range of voltage in an interval. Thus, an error will occur and it is called quantization error. This is the noise introduced by the process of quantization. Here the maximum quantization error is

$$\pm \frac{1}{2}\triangle V = \pm 0.5V$$

5.3 REAL - TIME CLOCKS:

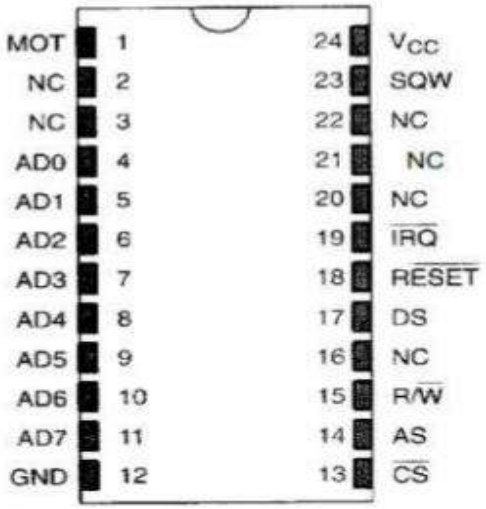
- A real time clock (RTC) keeps the time and date in an embedded system.
- Real time clocks are typically composed of a crystal – controlled oscillator, cascade counters and battery backup.
- The crystal – controlled oscillator generates a very consistent high-frequency digital pulse that feeds the cascaded counters.
- The first counter counts these pulses up to the oscillator frequency, which corresponds to exactly one second.
- At this point it generates a pulse that feeds the next counter. This counter counts up to 59, at which point it generates a pulse feeding the minute counter. The hour, date, month and year counters work in the same manner. Real- time clocks adjust for the leap years.
- The rechargeable back-up battery is used to keep the real-time clock running the system is powered off.

5.4 DS 12887 RTC CHIP & ITS INTERFACING

RTC DS12C887:

- RTC DS 12C887 is widely used to provide exact time and date in many applications such as x86 IBM PC.
- This RTC provides time components hour, minute and second in addition to the date/calendar components of year, month and day.
- This chip uses an internal lithium battery, which keeps the time and date updated even when the power is off.
- The DS12C887 works on CMOS technology to keep the power consumption low.
- It has a total of 128 bytes of non-volatile RAM.
- It uses 14 bytes of RAM for storing the values of clock/calendar and control registers.
- The rest 114 bytes of RAM are for general purpose data storage.
- The internal registers are accessible only when the RTC is powered by an external power source.
- When the external power is turned off, the RTC clock keeps running from the internal lithium battery source but the internal registers of the RTC cannot be accessed.
- RTC 12C887 becomes active when a voltage greater than 4.25V is applied and the internal registers can be accessed after 200 msec.

PIN DIAGRAM



SIGNAL DESCRIPTIONS

GND, V_{CC}:

- DC power is provided to the device on these pins. V_{CC} is the +5 volt input.
- When 5 volts are applied within normal limits, the device is fully accessible and data can be written and read.
- When V_{CC} is below 4.25 volts typical, reads and writes are inhibited. However, the timekeeping function continues unaffected by the lower input voltage.
- As V_{CC} falls below 3 volts typical, the RAM and timekeeper are switched over to an internal lithium energy source.
- The timekeeping function maintains an accuracy of ±1 minute per month at 25°C regardless of the voltage input on the V_{CC} pin.

MOT (Mode Select):

- This is an input pin that allows the choice between the Motorola and Intel microcontroller bus timings.
- The MOT pin is connected to GND for the Intel timing. That means when we connect DS12887 to the 8051, MOT=GND

SQW (Square Wave Output):

- Square wave is an output pin.
- we can program the DS12887 to provide up to 15 different square wave
- The frequency of the square wave is set by programming register A.

AD0-AD7 (Multiplexed Bidirectional Address/Data Bus):

- The multiplexed address/data pins provide both address and data to the chip.
- Addresses are latched into the DS12887 on the falling edge of the AS (ALE) signal.

AS (Address Strobe Input):

- AS (address strobe) is an input pin.
- On the falling edge it will cause the address to be latched into the DS12887.
- The AS pin is used for demultiplexing the address and data and is connected to the ALE pin of the 8051 chip.

DS (Data Strobe or Read Input):

- Data strobe or read is an input.
- When MOT=GND for Intel timing, the DS pin is called the RD (read) signal is connected to the RD PIN OF THE 8051.

R/ W (Read/Write Input):

- Read/Write is an input pin.
- When MOT=GND for the Intel timing, the R/W pin is called WR (write) signal and is connected to the WR pin of the 8051.

CS (Chip Select Input):

- Chip select is an input pin and an active low signal.
- During the read (RD) and write (WR) cycle time of Intel timing, the CS must be low in order to access the chip.
- It must be noted that the CS works only when the external Vcc is connected.
- In other words when Vcc falls below 4.25V, the chip select input is internally forced to an inactive level regardless of the value of CS at the input pin.
- This is called the write-protected state.
- When the DS12887 is in write-protected state, all inputs are ignored.

IRQ' (Interrupt Request Output):

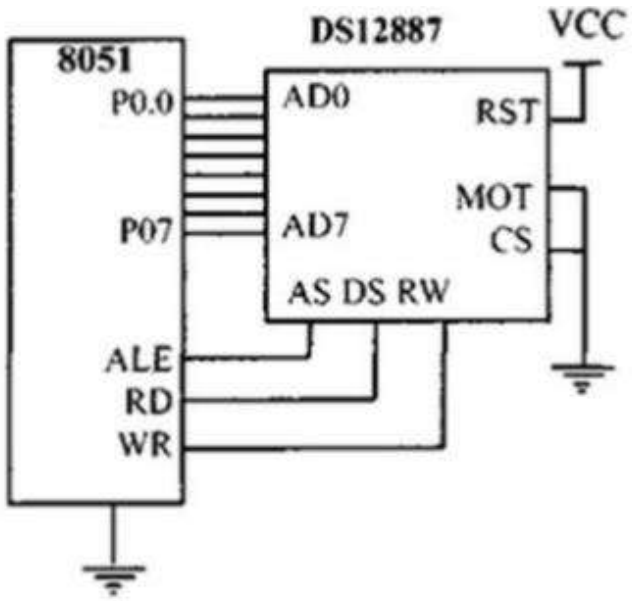
- Interrupt request is an output pin and active low signal.
- To use IRQ' the interrupt-enable bits in register B must be set high.

RESET' (Reset Input):

- It is an input and is active low (normally high).
- In most applications the reset pin is connected to the Vcc pin.
- In application where this pin is used, it has no effect on the clock, calendar, or RAM if it is forced low.
- The low on this pin will cause the reset of the IRQ' and clearing of the SQW pin.

DS12887 connection to 8051:

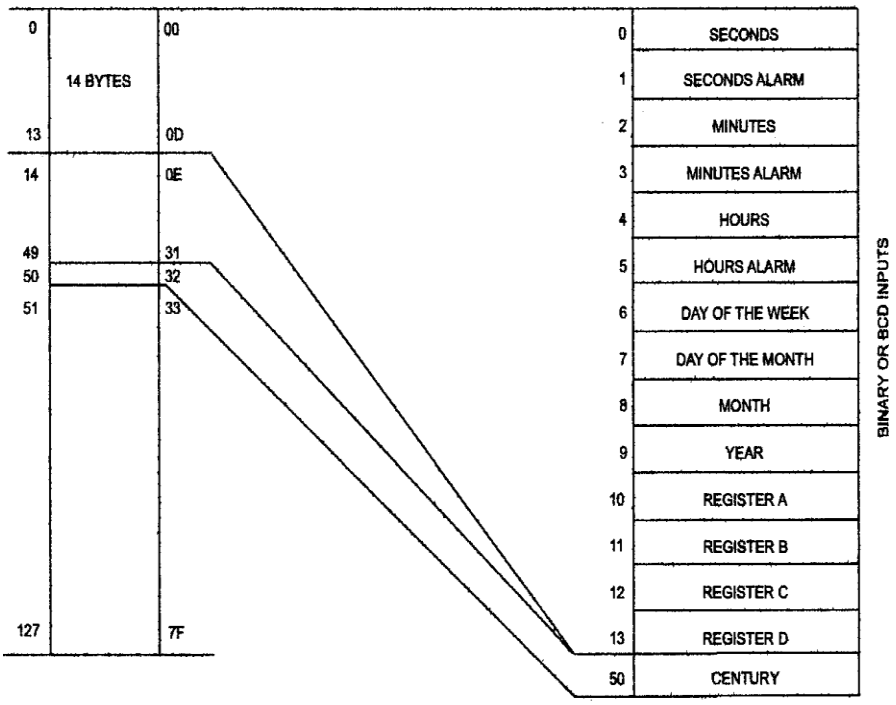
- A simple way of connecting the DS12887 to the 8051 is shown in figure.



- AD_0-AD_7 of the DS12887 are connected directly to P0 of the 8051 and there is no need for any 74xx373 latches, since Ds12887 provides the latch internally.
- To access the DS12887 in given diagram, we use the MOVX instruction since it is mapped as external memory.
- RTC DS12C887 has three interrupts, namely, Alarm interrupt, Periodic interrupt & Update interrupt. For detailed information, refer RTC interrupts.

Address Map of the DS12C887:

- The DS12C887 has a total of 128 bytes of RAM space with address 00-7FH.
- The first ten locations, 00-09, are set aside for RTC values of time, calendar, and alarm data.
- The next four locations 10-13 (0A-0D in hex) are reserved for the control and status registers A, B, C, and D.
- The next 114 bytes from address 0EH-7FH are available for general purpose data storage.
- The entire 128 bytes of RAM are accessible directly for read or write except the following.
  - Registers C and D are read only.
  - D7 bit of register A is read only.
  - The high-order bit of the second byte is read-only.



DS12887 Address Map

TIME, CALENDAR AND ALARM ADDRESS LOCATIONS AND MODES:

- The byte addresses 0-9 are set asides for the time, calendar, and alarm data. The given table shows their address locations and modes.
- It is available in both binary (hex) and BCD formats.

- The time and calendar information is obtained by reading the appropriate memory bytes.
- The time, calendar, and alarm are set or initialized by writing the appropriate RAM bytes.
- The contents of the ten time, calendar, and alarm bytes can be either Binary or Binary-Coded Decimal (BCD) format.
- Before writing the internal time, calendar, and alarm registers, the SET bit in Register B should be written to a logic one to prevent updates from occurring while access is being attempted.
- In addition to writing the ten time, calendar, and alarm registers in a selected format (binary or BCD), the data mode bit (DM) of Register B must be set to the appropriate logic level.
- All ten time, calendar, and alarm bytes must use the same data mode.
- The set bit in Register B should be cleared after the data mode bit has been written to allow the real time clock to update the time and calendar bytes.
- Once initialized, the real time clock makes all updates in the selected

ADDRESS LOCATION	FUNCTION	DECIMAL RANGE	RANGE	
			BINARY DATA MODE	BCD DATA MODE
O	Seconds	0-59	00-3B	00-59
1	Seconds Alarm	0-59	00-3B	00-59
2	Minutes	0-59	00-3B	00-59
3	Minutes Alarm	0-59	00-3B	00-59
4	Hours 12-hr, Mode	1-12	01-0C AM, 81-8C PM	01-12 AM, 81-92 PM
	Hours 24-hr, Mode	0-23	00-17	00-23
5	Hours Alarm 12-hr, Mode	1-12	01-0C AM, 81-8C PM	01-12 AM, 81-92 PM
	Hours Alarm 24-hr, Mode	0-23	00-17	00-23
6	Day of the week Sunday=1	1-7	01-07	01-07
7	Date of Month	1-31	01-1F	01-31
8	Month	1-12	01-0C	01-12
9	Year	0-99	00-63	00-99
50	Century	0-99	NA	19,20

5.5 MOTOR CONTROL: RELAY AND OPTOISOLATOR, STEPPER MOTOR INTERFACING, DC MOTOR INTERFACING:

MOTOR CONTROL-RELAYS AND OPTOISOLATORS:

- A **relay** is an electrically controllable switch widely used in industrial controls, automobiles, and appliances.
- It allows the isolation of two separate sections of a system with two different voltage sources.
- For example, a +5 V system can be isolated from a 120 V system by placing a relay between them. One such relay is called an electromechanical (or electromagnetic) relay (EMR).
- The EMRs have three components: the coil, spring, and contacts.
- In given diagram, a digital +5 V on the left side can control a 12 V motor on the right side without any physical contact between them.
- When current flows through the coil, a magnetic field is created around the coil (the coil is energized), which causes the armature to be attracted to the coil.
- The armature's contact acts like a switch and closes or opens the circuit. When the coil is not energized, a spring pulls the armature to its normal state of open or closed.
- In the block diagram for electomechanical relays (EMR) we do not show the spring, but it does exist internally. There are all types of relays for all kinds of applications. In choosing a relay the following characteristics need to be
- The contacts can be normally open (NO) or normally closed (NC). In the NC type, the contacts are closed when the coil is not energized. In the NO, the contacts are open when the coil is unenergized.
- There can one or more contacts. For example, we can have SPST (single pole, single throw), SPDT (single pole, double throw), and DPDT (double pole, double throw) relays.
- The voltage and current needed to energize the coil. The voltage can vary from a

few volts to 50 volts, while the current can be from a few mA to 20 mA. The relay has a minimum voltage, below which the coil will not be energized. This minimum voltage is called the "pull-in" voltage. In the datasheet for relays we might not see current, but rather coil resistance. The V/R will give you the pull- in current. For example, if the coil voltage is 5 V, and the coil resistance is 500 ohms, we need a minimum of 10mA (5V/500 ohms=10mA) pull-in current.

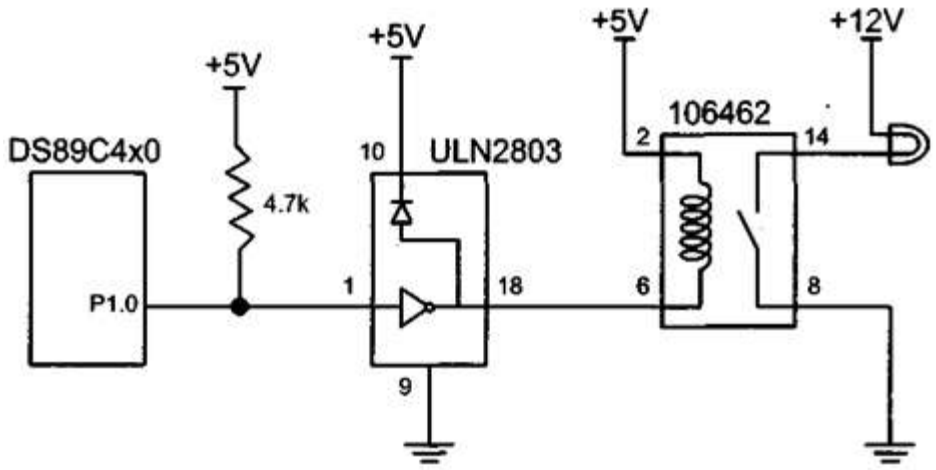
- The maximum DC/AC voltage and current that can be handled by the contacts. This is in the range of a few volts to hundreds of volts, while the current can be from a few amps to 40 A or more, depending on the relay. Notice the difference between this voltage/current specification and the voltage/current needed for energizing the coil. The fact that one can use such a small amount of voltage/current on one side to handle a large amount of voltage/current on the other side is what makes relays so widely used in industrial controls. Examine Table 17-1 for some relay characteristics.

Selected DIP Relay Characteristics

Part No.	Contact Form	Coil Volts	Coil Ohms	Contact Volts-Current
106462CP	SPST-NO	5 VDC	500	100 VDC-0.5 A
138430CP	SPST-NO	5 VDC	500	100 VDC-0.5 A
106471CP	SPST-NO	12 VDC	1000	100 VDC-0.5 A
138448CP	SPST-NO	12 VDC	1000	100 VDC-0.5 A
129875CP	DPDT	5 VDC	62.5	30 VDC-1 A

Driving a relay

- Digital systems and microcontroller pins lack sufficient current to drive the relay.
- While the relay's coil needs around 10 mA to be energized, the microcontroller's pin can provide a maximum of 1–2 mA current. For this reason, we place a driver, such as the ULN2803, or a power transistor between the microcontroller and the relay as shown in Figure given below.



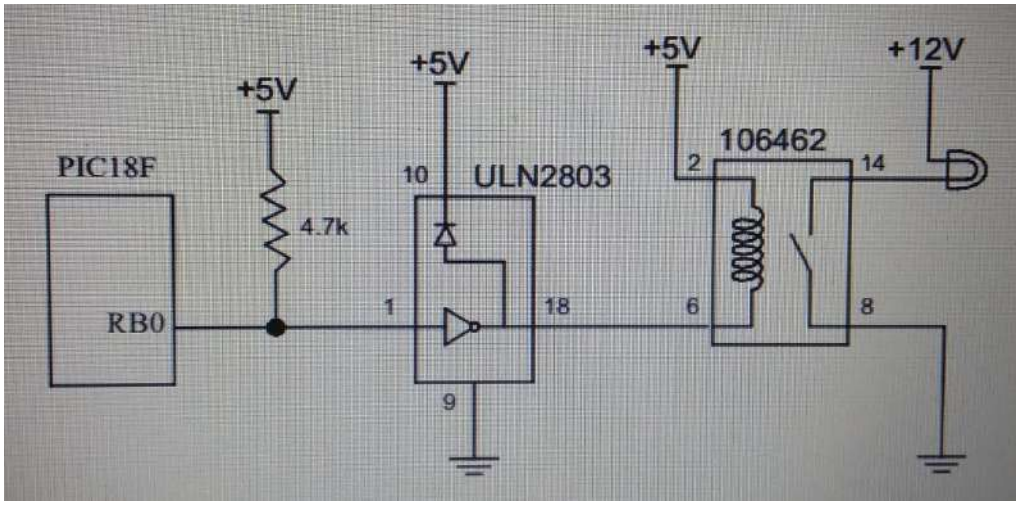
Solid-state relay

- Another widely used relay is the solid-state relay. In this relay, there is no coil, spring, or mechanical contact switch.
- The entire relay is made out of semiconductor materials. Because no mechanical parts are involved in solid-state relays, their switching response time is much faster than that of electromechanical relays.
- Another advantage of the solid-state relay is its greater life expectancy.
- The life cycle for the electromechanical relay can vary from a few hundred thousand to a few million operations.
- Wear and tear on the contact points can cause the relay to malfunction after a while.
- Solid-state relays, however, have no such limitations. Extremely low input current and small packaging make solid- state relays ideal for microprocessor and logic control switching.
- They are widely used in controlling pumps, solenoids, alarms, and other power applications.
- Some solid-state relays have a phase control option, which is ideal for motor-speed control and light-dimming applications.

Selected Solid State Relay Characteristics:

Part No.	Contact Style	Control Volts	Contact Volts	Contact Current
143058CP	SPST	4–32 VDC	240 VAC	3 A
139053CP	SPST	3–32 VDC	240 VAC	25 A
162341CP	SPST	3–32 VDC	240 VAC	10 A
172591CP	SPST	3–32 VDC	60 VDC	2 A
175222CP	SPST	3–32 VDC	60 VDC	4 A
176647CP	SPST	3–32 VDC	120 VDC	5 A

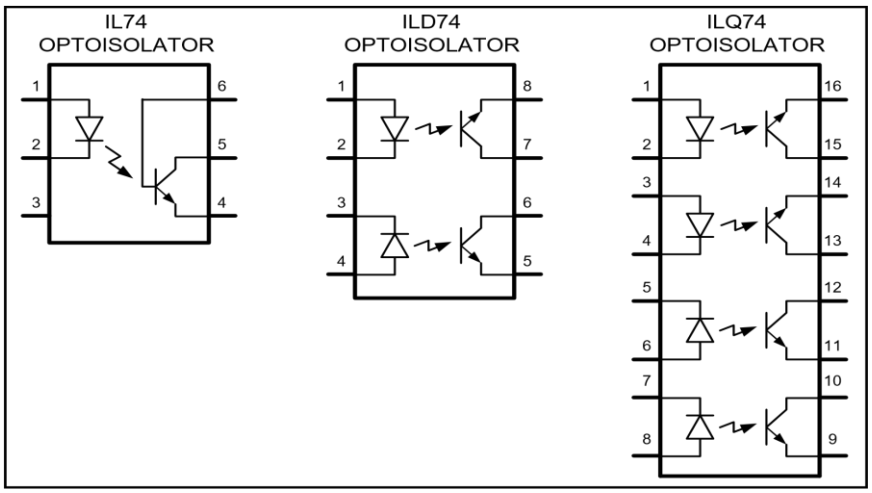
- Diagram shows control of a fan using a solid-state relay (SSR).



Optoisolators

- In some applications we use an optoisolators (also called optocoupler) to isolate two parts of a system.
- An example is driving a motor. Motors can produce what is called back EMF, a high-voltage spike produced by a sudden change of current as indicated in the V = Ldi/dt formula.
- In situations such as printed circuit board design, we can reduce the effect of this unwanted voltage spike (called ground bounce) by using decoupling capacitors
- In systems that have inductors (coil winding), such as motors, a decoupling capacitor or a diode will not do the job.
- In such cases we use optoisolators. An optoisolators has an LED (light-emitting diode) transmitter and a photo sensor receiver, separated from each other by a gap.
- When current flows through the diode, it transmits a signal light across the gap and the receiver produces the same signal with the same phase but a different current and amplitude.
- Optoisolators are also widely used in communication equipment such as modems.

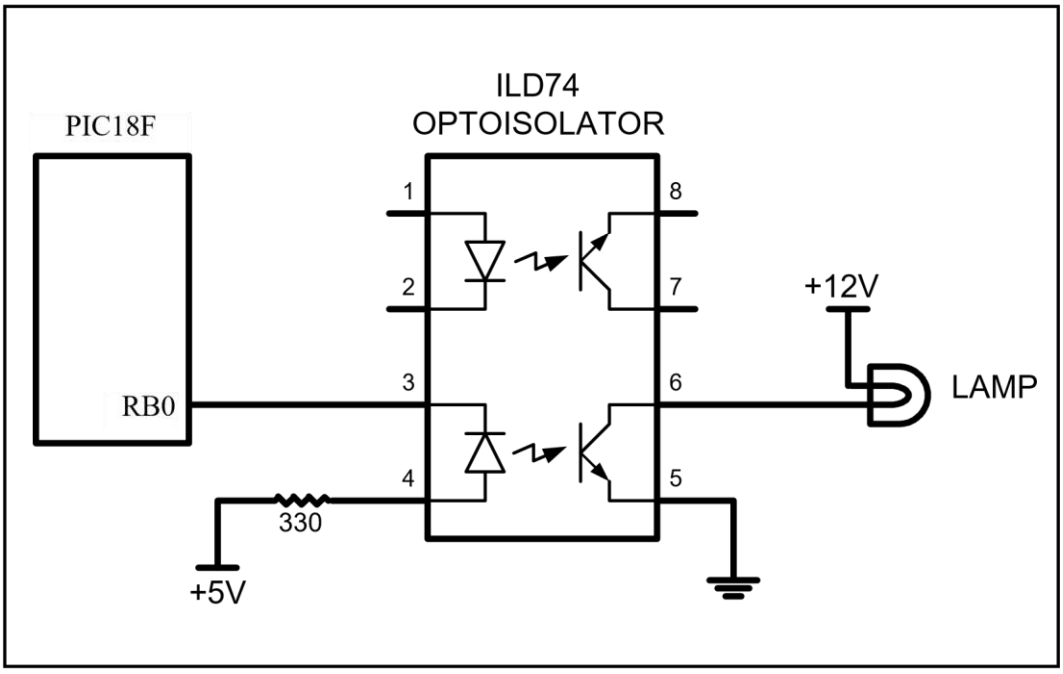
- This device allows a computer to be connected to a telephone line without risk of damage from power surges. The gap between the transmitter and receiver of optoisolators prevents the electrical current surge from reaching the system.



Optoisolator Package Examples

Interfacing an Optoisolator:

- The Optoisolator comes in a small IC package with four or more pins.
- There are also packages that contain more than one Optoisolator.
- When placing an Optoisolator between two circuits, we must use two separate voltage sources, one for each side.
- Unlike relays, no drivers need to be placed between the microcontroller/digital output and the optoisolators.



Controlling a Lamp via Optoisolator

STEPPER MOTOR INTERFACING:

- This section begins with an overview of the basic operation of stepper motors.
- Then we describe how to interface a stepper motor to the PIC18. Finally, we use Assembly language programs to demonstrate control of the angle and direction of stepper motor rotation.

Stepper motors:

- A stepper motor is a widely used device that translates electrical pulses into mechanical movement.
- In applications such as disk drives, dot matrix printers, and robotics, the stepper motor is used for position control.
- Stepper motors commonly have a permanent magnet rotor (also called **the shaft)** surrounded by a stator **(see Figure 1).**
- There are also steppers called variable **reluctance** stepper motors that do not have a permanent magnet rotor.
- The most common stepper motors have four stator windings that are paired with a center-tapped common as shown in **(figure 2).**
- This type of stepper motor is commonly referred to as a **four phase** or **unipolar** stepper motor.
- The center tap allows a change of current direction in (**Figure 1). Rotor Alignment** each of two coils when a winding is grounded, thereby resulting in a polarity change of the stator.
- Notice that while a conventional motor shaft runs freely, the stepper motor shaft moves in a fixed repeatable increment, which allows one to move it to a precise position. This repeatable fixed movement is possible as a result of basic magnetic theory where poles of the same polarity repel **Figure 2 Stator Winding** and opposite poles attract.

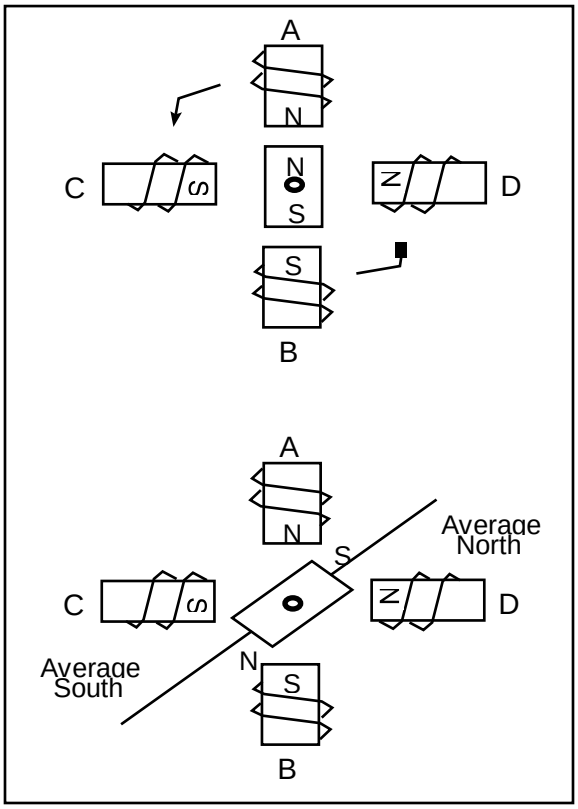


Figure-1 Rotor Alignment

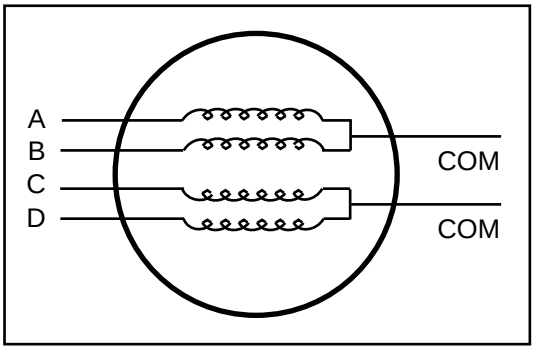
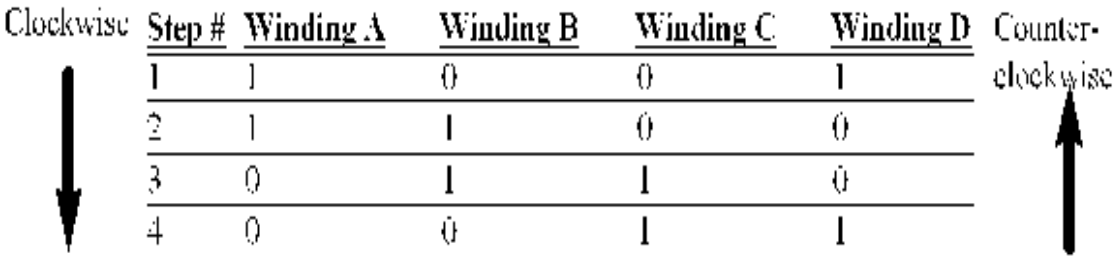


Figure-2 stator windings configuration

- The direction **Configuration** of the rotation is dictated by the stator poles. The stator poles are determined by the current sent through the wire coils.
- As the direction of the current is changed, the polarity is also changed causing the reverse motion of the rotor.
- The stepper motor discussed here has a total of six leads: four leads representing the four stator windings and two commons for the center-tapped leads.
- As the sequence of power is applied to each stator winding, the rotor will rotate. There are several widely used sequences, each of which has a different degree of precision.

- Note that although we can start with any of the sequences once we start we must continue in the proper order.
- For example, if we start with step 3 (0110), we must continue in the sequence of steps 4, 1, 2, etc.



Step #	Winding A	Winding B	Winding C	Winding D
1	1	0	0	1
2	1	1	0	0
3	0	1	1	0
4	0	0	1	1

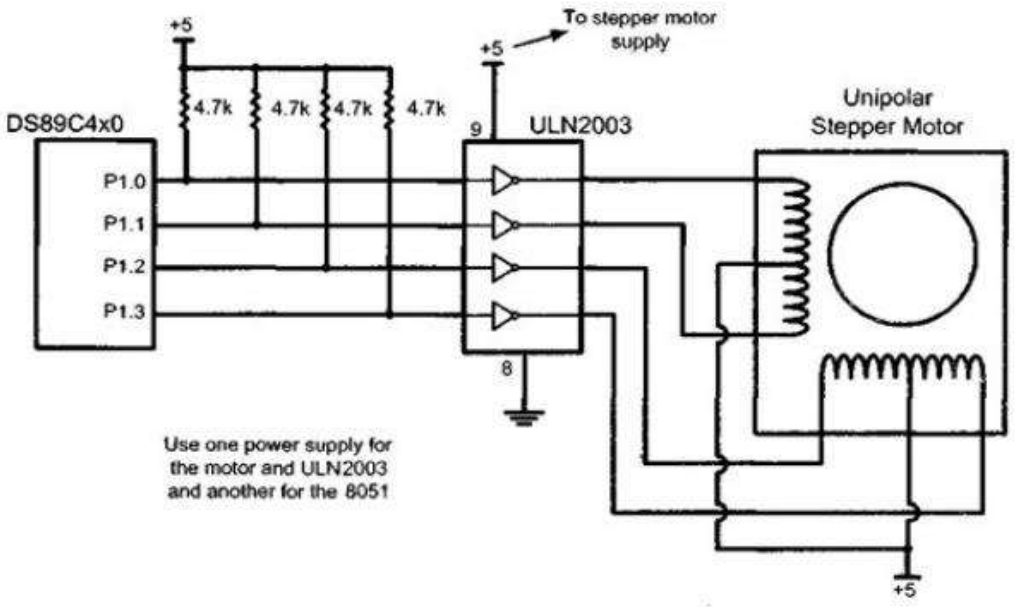
Step Angle:

- **How much movement is associated with a single step?**
- This depends on the internal construction of the motor, in particular the number of teeth on the stator and the rotor. The **step angle** is the minimum degree of rotation associated with a single step.
- Various motors have different step angles. Given Table shows some step angles for various motors.

Stepper Motor Step Angles

Step Angle	Steps per Revolution
0.72	500
1.8	200
2.0	180
2.5	144
5.0	72
7.5	48
15	24

- In a given table, notice the term **steps per revolution**. This is the total number of steps needed to rotate one complete rotation or 360 degrees (e.g.180 steps × 2 degrees = 360).
- It must be noted that perhaps contrary to one's initial impression, a stepper motor does not need more terminal leads for the stator to achieve smaller steps.
- All the stepper motors discussed in this section have four leads for the stator winding and two COM wires for the center tap.
- Although some manufacturers set aside only one lead for the common signal instead of two, they always have four leads for the stators.



8051 connection to stepper motor

DC MOTORS:

- A direct current (DC) motor is another widely used device that translates electrical pulses into mechanical movement.
- In the DC motor we have only + and – leads. Connecting them to a DC voltage source moves the motor in one direction.
- By reversing the polarity, the DC motor will move in the opposite direction.
- One can easily experiment with the DC motor. For example, small fans used in many motherboards to cool the CPU are run by DC motors.
- By connecting their leads to the + and – voltage source, the DC motor moves.
- While a stepper motor moves in steps of 1 to 15 degrees, the DC motor moves continuously.
- In a stepper motor, if we know the starting position we can easily count the number of steps the motor has moved and calculate the final position of the motor.
- This is not possible in a DC motor. The maximum speed of a DC motor is indicated in rpm and is given in the data sheet.
- The DC motor has two **rpms: no-load and loaded.** The manufacturer's data sheet gives the no-load rpm. The no-load rpm can be from a few thousand to tens of thousands.
- The rpm is reduced when moving a load and it decreases as the load is increased.
- For example, a drill turning a screw has a much lower rpm speed than when it is in the no-load situation. DC motors also have voltage and current ratings.

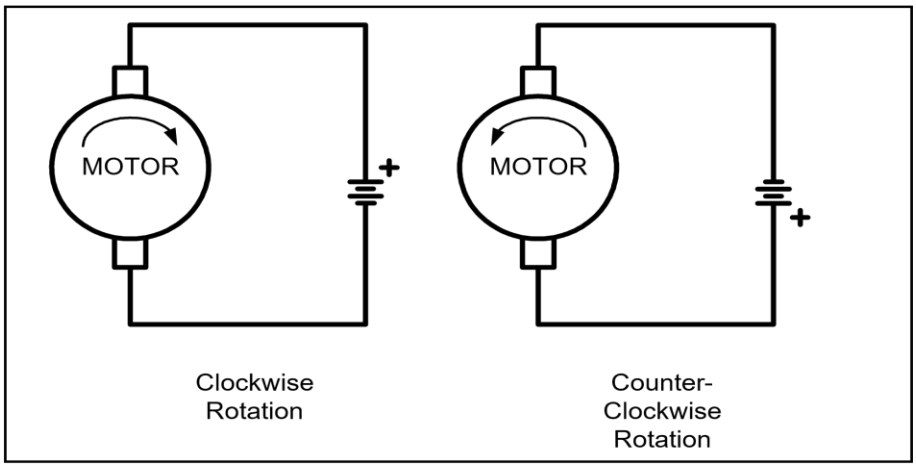
- The nominal voltage is the voltage for that motor under normal conditions, and can be from 1 to 150 V, depending on the motor.
- As we increase the voltage, the rpm goes up.
- The current rating is the current consumption when the nominal voltage is applied with no load, and can be from 25 mA to a few amps.
- As the load increases, the rpm is decreased, unless the current or voltage provided to the motor is increased, which in turn increases the torque. With a fixed voltage, as the load increases, the current (power) consumption of a DC motor is increased.
- If we overload the motor it will stall, and that can damage the motor due to the heat generated by high current consumption.

Unidirectional control

- In a given figure the DC motor rotation for clockwise (CW) and counterclockwise (CCW) rotations.

Bidirectional control

- With the help of relays or some specially designed chips we can change the direction of the DC motor rotation. Figures 17-14 through 17-17 show the basic concepts of H-Bridge control of DC motors.



DC Motor Rotation (Permanent Magnet Field)

Selected DC Motor Characteristics

Part No.	Nominal Volts	Volt Range	Current	RPM	Torque
154915CP	3 V	1.5–3 V	0.070 A	5,200	4.0 g-cm
154923CP	3 V	1.5–3 V	0.240 A	16,000	8.3 g-cm
177498CP	4.5 V	3–14 V	0.150 A	10,300	33.3 g-cm
181411CP	5 V	3–14 V	0.470 A	10,000	18.8 g-cm

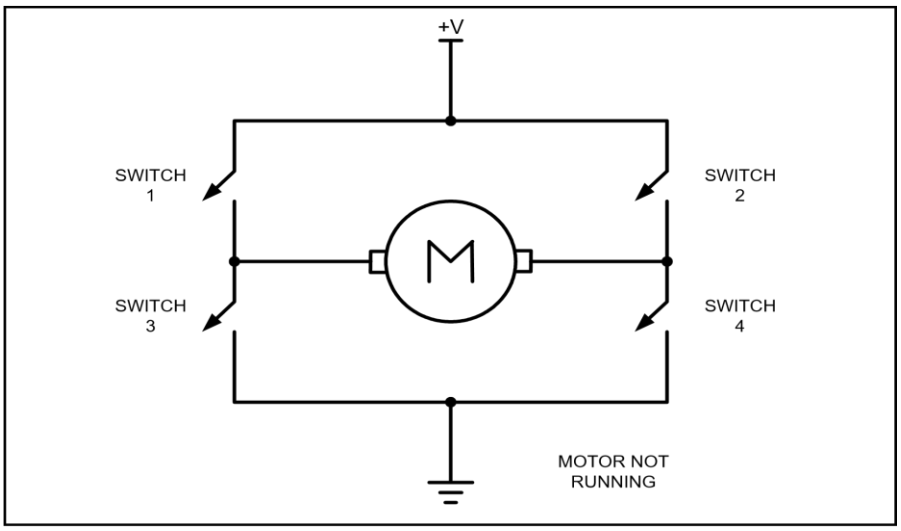


Figure-1 H-bridge Motor Configuration

- Figure-2 shows the switch configuration for turning the motor in one direction. When switches 1 and 4 are closed, current is allowed to pass through the motor

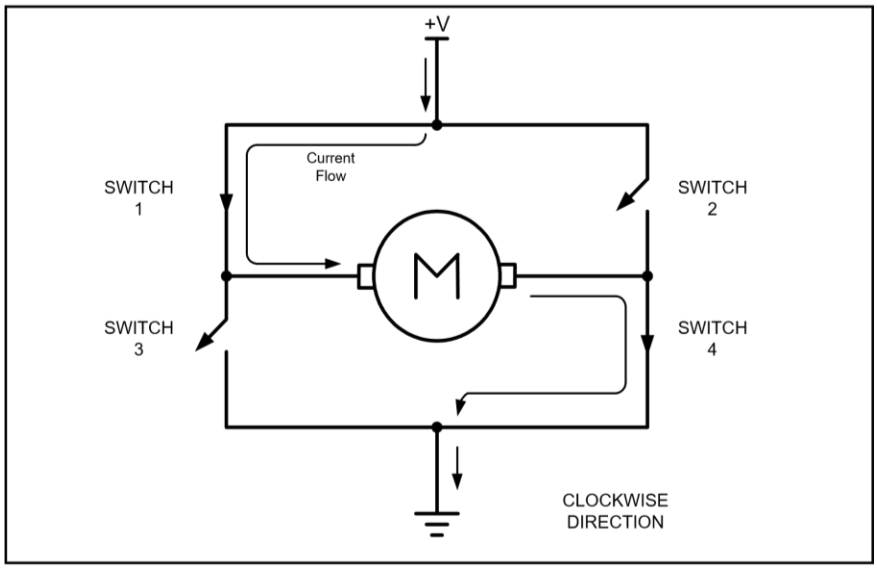


Figure-2 H-bridge Motor Clockwise Configuration

- Figure 3 shows the switch configuration for turning the motor in the opposite direction from the configuration of Figure 2. When switches 2 and 3 are closed, current is allowed to pass through the motor.

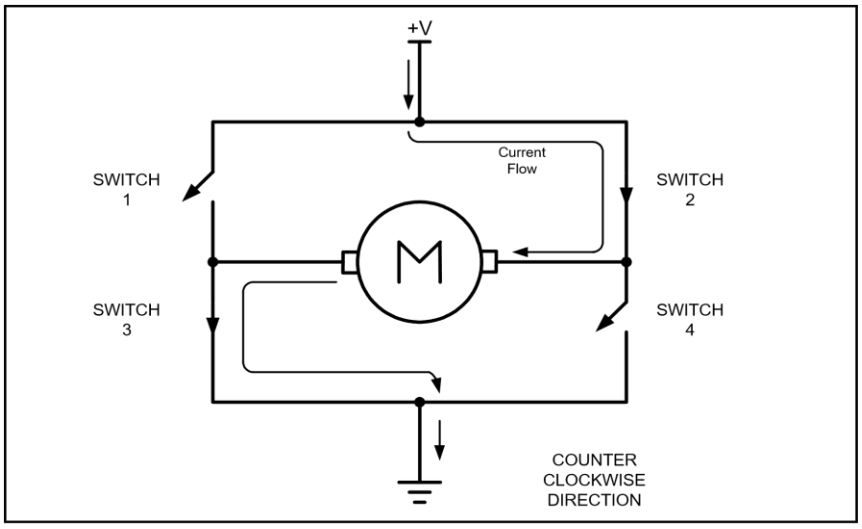


Figure-3 H-bridge Motor Counterclockwise Configuration

- Figure 4 shows an invalid configuration. Current flows directly to ground, creating a short circuit. The same effect occurs when switches 1 and 3 are closed or switches 2 and 4 are closed.
- In a given below Table shows some of the logic configurations for the H-Bridge design.
- H-Bridge control can be created using relays, transistors, or a single IC solution such as the L293. When using relays and transistors, you must ensure that invalid configurations do not occur.

Motor Operation	**SW1**	**SW2**	**SW3**	**SW4**
Off	Open	Open	Open	Open
Clockwise	Closed	Open	Open	Closed
Counterclockwise	Open	Closed	Closed	Open
Invalid	Closed	Closed	Closed	Closed

- Although we do not show the relay control of an H-Bridge. Given Example shows a simple program to operate a basic H-Bridge.

  A switch is connected to pin RD7 (PORTD.7). Using a simulator, write a program to simulate the H-Bridge in Table 17-10. We must perform the following:

  (a) If DIR = 0, the DC motor moves clockwise.
  (b) If DIR = 1, the DC motor moves counterclockwise.

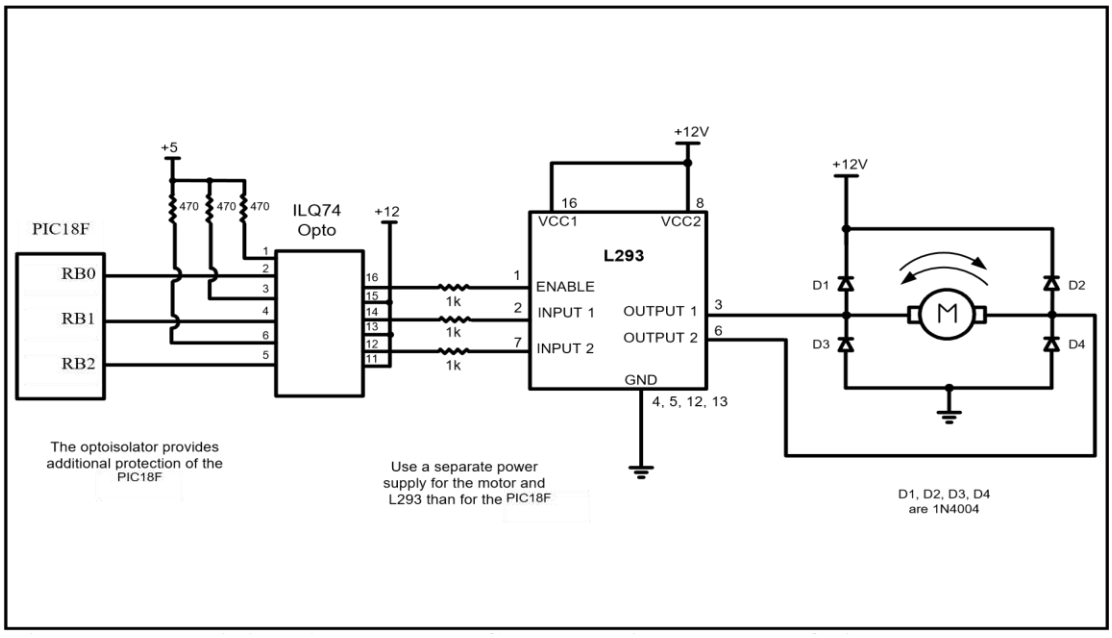
Solution:

```
BCF TRISB,0                ;PORTB.0 as output for switch 1
BCF TRISB,1                ;      .1 "              switch 2
```

```
        BCF TRISB,2                ;          .2 ”          switch 3
        BCF TRISB,3                ;          .3 ”          switch 4
        BSF TRISD,7                ;make PORTD.7 an input DIR
MONITOR:
        BTFSS PORTD,7
        BRA CLOCKWISE
        BSF PORTB,0                ;switch 1
        BCF PORTB,1                ;switch 2
        BCF PORTB,2                ;switch 3
        BSF PORTB,3                ;switch 4
        BRA MONITOR
CLOCKWISE:
        BCF PORTB,0                ;switch 1
        BSF PORTB,1                ;switch 2
        BSF PORTB,2                ;switch 3
        BCF PORTB,3                ;switch 4
        BRA MONITOR

        END
```

- In given below Figure shows the connection of the L293 to a PIC18. Be aware that the L293 will generate heat during operation. For sustained operation of the motor, use a heat sink.



Bidirectional Motor Control Using an L293 Chip

- Given example shows control of the L293.
  Above Figure shows the connection of an L293. Add a switch to pin RD7 (PORTD.7). Write a program to monitor the status of SW and perform the following:

  (a) If SW = 0, the DC motor moves clockwise.
  (b) If SW = 1, the DC motor moves counterclockwise.

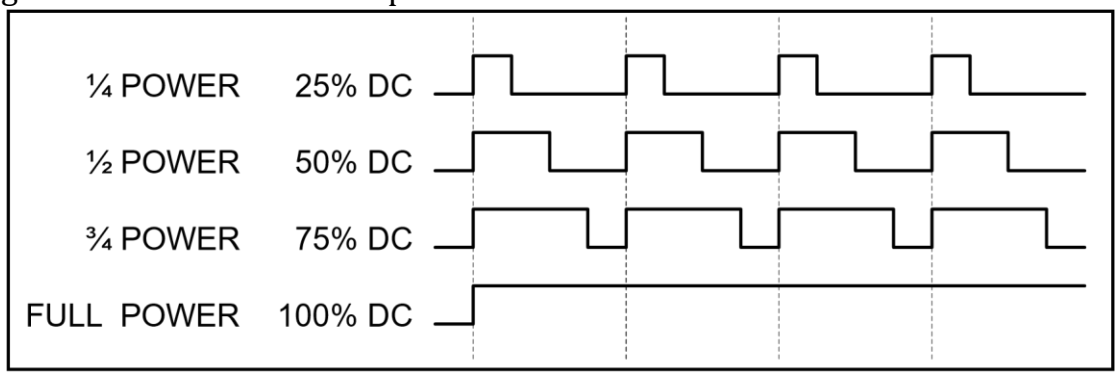
  Solution:

```
        BCF  TRISB,0
        BCF  TRISB,1
        BCF  TRISB,2
        BSF  TRISD,7
       BSF  PORTB,0              ;enable the chip
CHK BTFSS PORTD,7
        BRA CWISE
        BCF  PORTB,1              ;turn the motor counterclockwise
        BSF  PORTB,2
        BRA  CHK
CWISEBSF  PORTB,1
        BCF  PORTB,2              ;turn motor clockwise
        BRA  CHK
```

Pulse width modulation (PWM):

- The speed of the motor depends on three factors: (a) load, (b) voltage, and (c) current.
- For a given fixed load we can maintain a steady speed by using a method called **pulse width modulation** (PWM).
- By changing (modulating) the width of the pulse applied to the DC motor we can increase or decrease the amount of power provided to the motor, thereby increasing or decreasing the motor speed.
- Notice that, although the voltage has a fixed amplitude, it has a variable duty cycle. That means the wider the pulse, the higher the speed.
- PWM is so widely used in DC motor control that some microcontrollers come with the PWM circuitry embedded in the chip.

- In such microcontrollers all we have to do is load the proper registers with the values of the high and low portions of the desired pulse, and the rest is taken care of by the microcontroller.
- This allows the microcontroller to do other things. For microcontrollers without PWM circuitry, we must create the various duty cycle pulses using software, which prevents the microcontroller from doing other things.
- The ability to control the speed of the DC motor using PWM is one reason that DC motors are preferable over AC motors.
- AC motor speed is dictated by the AC frequency of the voltage applied to the motor and the frequency is generally fixed.
- As a result, we cannot control the speed of the AC motor when the load is increased. We can also change the DC motor's direction and torque. See Figure given below for PWM comparisons.

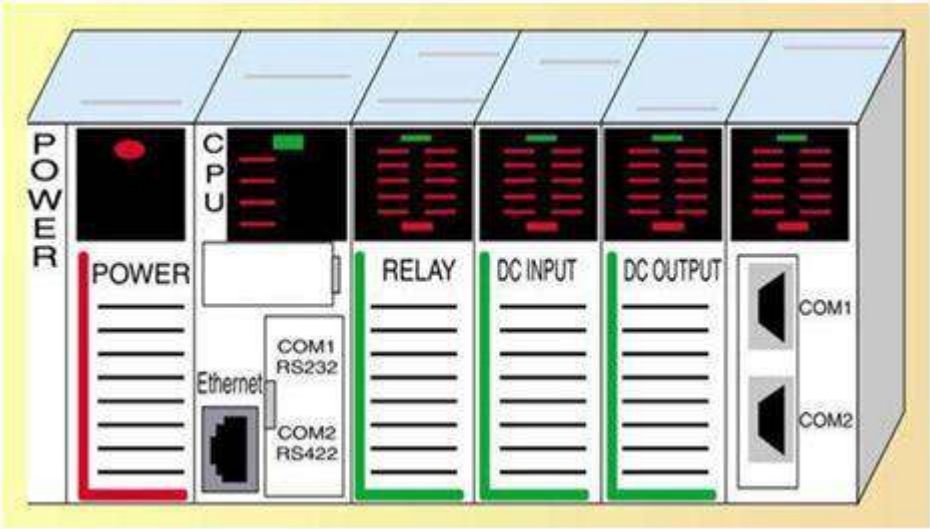


Pulse Width Modulation Comparison

UNIT-6: PROGRAMMABLE LOGIC CONTROLLERS

INTRODUCTION:

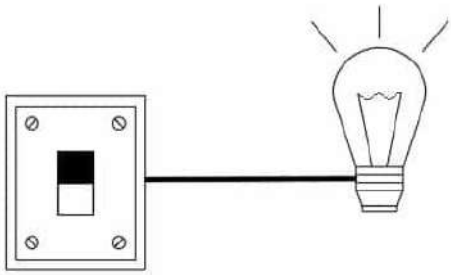
- A programmable logic controller (PLC) is a specialized computer used to control machines and processes.
- It uses a programmable memory to store instructions and execute specific functions that include on/off control, Timing, Counting, Sequencing, arithmetic and data handling.
- Programmable logic controllers are used for the control and operation of manufacturing process equipment and machinery.



Programming Logic Controller (PLC)

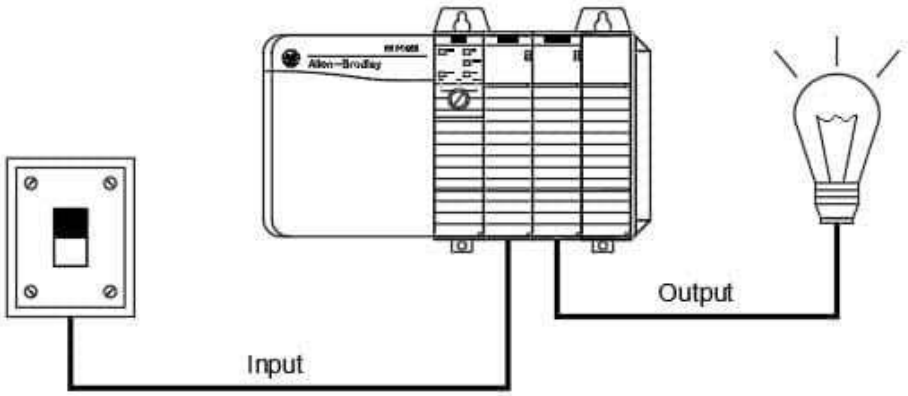
Example

- Imagine you have a switch that turns on a light. The light has two states, ON or OFF, and will respond almost instantaneously to someone either flipping the switch ON, or OFF.



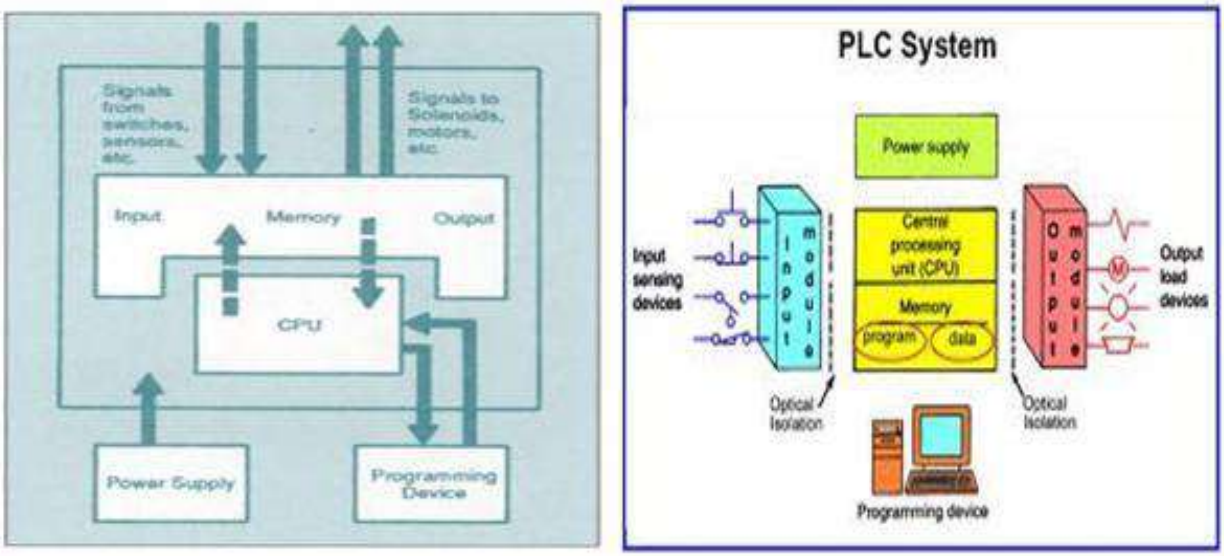
- Now imagine you wire this switch up directly to the light and your boss comes to tell you that he would like the light to come on precisely 30 seconds after the switch turns on…if there is a problem! In order to achieve this it will require additional hardware, i.e., a timing relay, and some rewiring. Now enter the PLC!

- With a PLC there will be no need to buy additional hardware every time a change is required. All that will be needed is a simple programming change that will delay the light from turning on until 30 seconds after the switch is thrown.



- The light switch becomes an "input" to the PLC, and the light itself is an "output".

6.1 PLC ARCHITECTURE:



PLC Internal Architecture

A basic PLC system consists of the following sections:

Input/ Output Section:

- The input section or input module consists of devices like sensors, switches and many other real world input sources.
- The input from the sources is connected to the PLC through the input connector rails.
- The output section or output module can be a motor or a solenoid or a lamp or a heater, whose functioning is controlled by varying the input signals.

CPU or Central Processing Unit:

- It is the brain of the PLC. It can be a hexagonal or an octal microprocessor.
- It carries out all the processing related to the input signals in order to control the output signals based on the control program.

Programming Device:

- It is the platform where the program or the control logic is written.
- It can be a handheld device or a laptop or a computer itself.

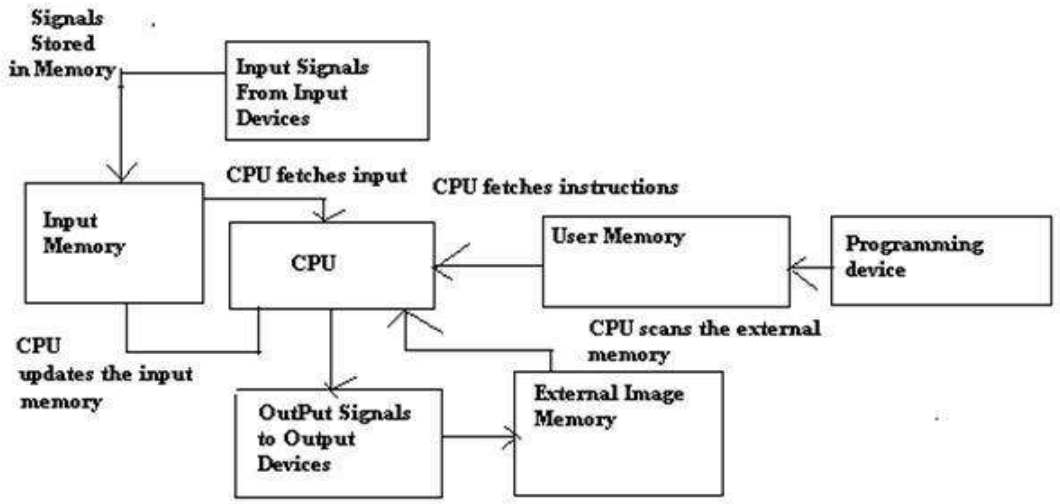
Power Supply:

It generally works on a power supply of about 24 V, used to power input and output devices.

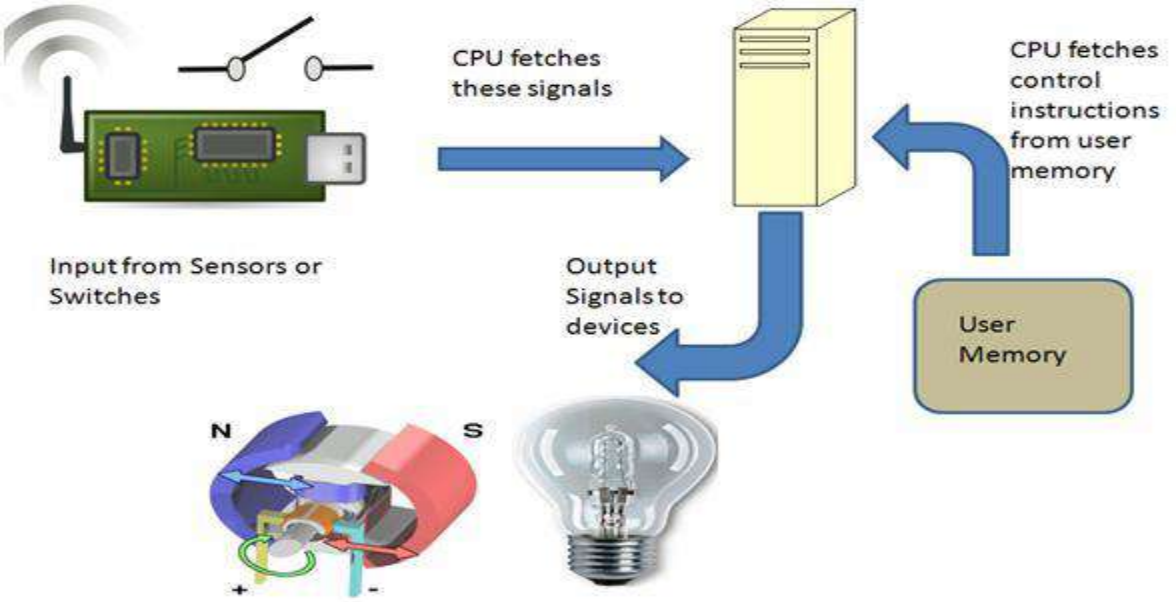
Memory:

- The memory is divided into two parts
  - The data memory and
  - The program memory.
- The program information or the control logic is stored in the user memory or the program memory from where the CPU fetches the program instructions.
- The input and output signals and the timer and counter signals are stored in the input and output external image memory respectively.

Working of a PLC:



PLC Working Schematic



Working of PLC

- The input sources convert the real time analog electric signals to suitable digital electric signals and these signals are applied to the PLC through the connector rails.
- These input signals are stored in the PLC external image memory in locations known as bits. This is done by the CPU
- The control logic or the program instructions are written onto the programming device through symbols or through mnemonics and stored in the user memory.
- The CPU fetches these instructions from the user memory and executes the input signals by manipulating, computing, processing them to control the output devices.

- The execution results are then stored in the external image memory which controls the output drives.
- The CPU also keeps a check on the output signals and keeps updating the contents of the input image memory according to the changes in the output memory.
- The CPU also performs internal programming functioning like setting and resetting of the timer, checking the user memory.

6.2 BASIC OPERATION PLC:

The operational sequence is as follows:

1. Input switch is pressed.
2. Input module places a "1″ in the input data table,
3. The ladder logic program sees the "1″ and caused a "1″ to be put into the output data table.
4. The output data table causes the output module to energize associated point.
5. The output device energizes.

The Scan Cycle

- PLCs operate by continually scanning programs and repeat this process many times per second.
- When a PLC starts, it runs checks on the hardware and software for faults, also called a self-test.
- If there are no problems, then the PLC will start the scan cycle.
- The scan cycle consists of three steps: **input scan**, **executing program(s), and output scan.**

Input Scan:

- A simple way of looking at this is the PLC takes a snapshot of the inputs and solves the logic.
- The PLC looks at each input card to determine if it is ON or OFF and saves this information in a data table for use in the next step.
- This makes the process faster and avoids cases where an input changes from the start to the end of the program.

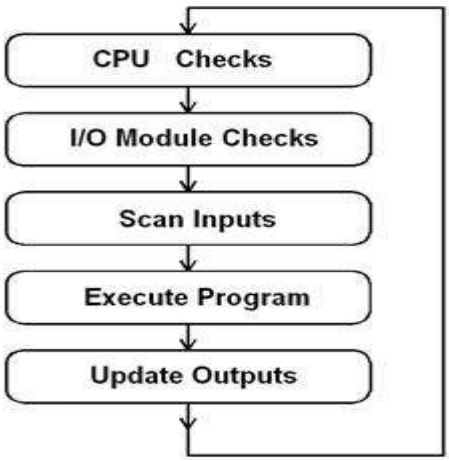
Execute Program (or Logic Execution):

- The PLC executes a program one instruction at a time using only the memory copy of the inputs the ladder logic program.

- For example, the program has the first input as ON. Since the PLC knows which inputs are ON/OFF from the previous step, it will be able to decide whether the first output should be turned ON.

Output Scan:

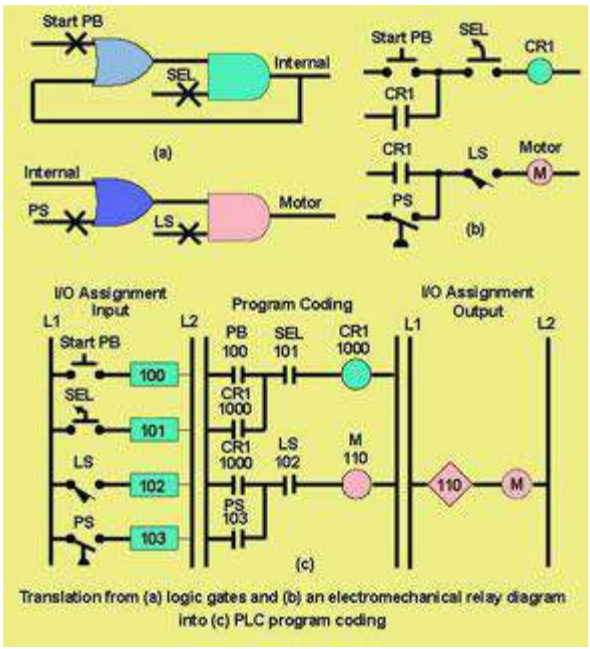
- When the ladder scan completes, the outputs are updated using the temporary values in memory.
- The PLC updates the status of the outputs based on which inputs were ON during the first step and the results of executing a program during the second step.
- The PLC now restarts the process by starting a self-check for faults.



PLC Scan Cycle

Logic Scan:

- Ladder logic programs are modeled after relay logic. In relay logic, each element in the ladder will switch as quickly as possible.
- Program elements can only be examined one at a time in a fixed sequence.
- The ladder logic scan begins at the top rung. At the end of the rung, it interprets the top output first, and then the output branched below it.
- On the second rung, it solves branches, before moving along the ladder logic rung.



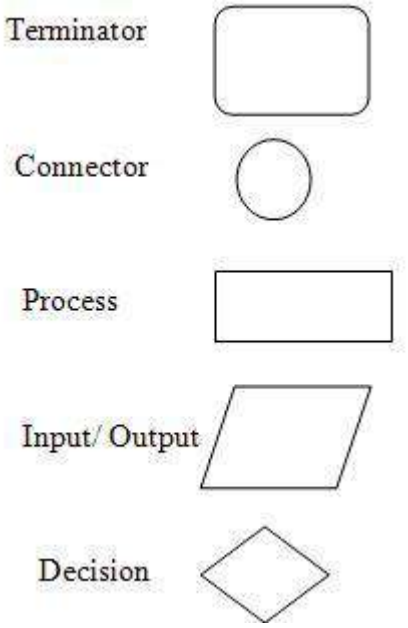
PLC Logic Scan

6.3 PLC PROGRAMMING:

- The basic functioning of the PLC relies on the control logic or the programming technique used.
- A PLC program consists of a set of instructions either in textual or graphical form, which represents the logic that governs the process the PLC is controlling
- Programming can be done using flowcharts or using ladder logic or using statement logics or mnemonics.
- Compute the flowchart. A flowchart is the symbolic representation of the instructions. It is the most basic and simplest form of control logic which involves only logic decisions.
- There are two main classifications of PLC programming languages, which is further divided into many sub-classified types.
- **Textual Language**
  - Instruction list
  - Structured text

- **Graphical Form**
  - Ladder Diagrams (LD) (i.e. Ladder Logic)
  - Function Block Diagram (FBD)
  - Sequential Function Chart (SFC)

- Different symbols are as given below:



- Write the Boolean expression for the different logic. Boolean algebra usually involves logic operations like AND, OR, NOT, NAND and NOR.
- The different symbols are:

  \+ OR operator

  . AND operator

  ! NOT operator.

- the instructions in simple statement forms like below:

  →**IF Input1 AND Input2 Then SET Output1 ELSE SET Output**

- Write the ladder logic program. It is the most important part of PLC programming.
- symbols and terminologies of ladder logic programming
- **Rung:** One step in the ladder is called a **rung.** In simpler words, the basic statement or one control logic is called a **Rung.**

  Y- Normal Output signals

  M – Motor symbol

  T – Timer

  C – Counter

- **Symbols:**

| |	Load symbol or switch in ON condition.
|/|	Load Inverse or switch in OFF condition
( )	Output Symbol
[CJP]	Conditional jump
[EJB]	End jump
[SFT]	Shift

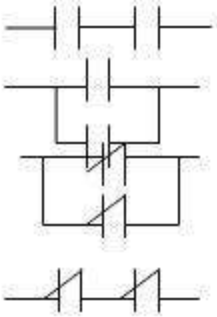
- **Basic Logic Functions using Ladder Logic**

AND logic

OR logic

NAND logic

NOR logic



- Writing Mnemonics: Mnemonics are instructions written in symbolic form. They are also known as Opcode and are used in handheld programming devices.
- Different Symbols are as given below:

  Ldi→ Load Inverse

  Ld → Load

  AN→ And logic

  OR→ Or logic

  ANI → NAND logic

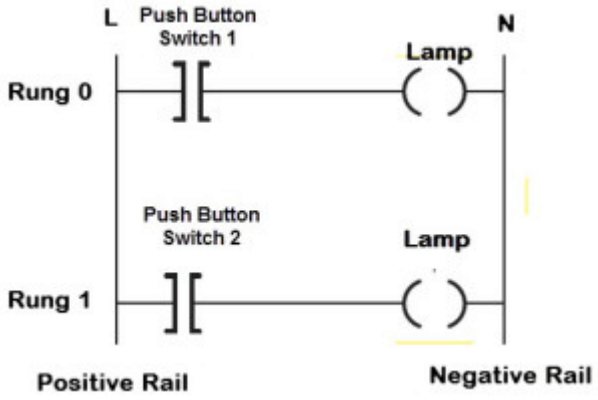
  ORI→ NOR logic

  Out →Output

- Due to the simple and convenient features, graphical representation are much preferred to textual languages.

Ladder Logic

- Ladder logic is the simplest form of PLC programming. It is also known as “relay logic”.
- The relay contacts used in relay controlled systems are represented using ladder logic.
  The below figure shows the simple example of a ladder diagram.

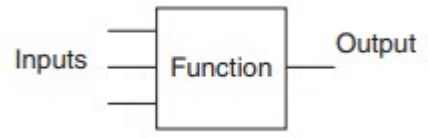


PLC Ladder Logic

- In the above-mentioned example, two push buttons are used to control the same lamp load.
- When any one of the switches is closed, the lamp will glow.
- The two horizontal lines are called rungs and two vertical lines are called rails.
- Every rung forms the electrical connectivity between Positive rail (P) and Negative rail (N). This allows the current to flow between input and output devices.

Functional Block Diagrams:

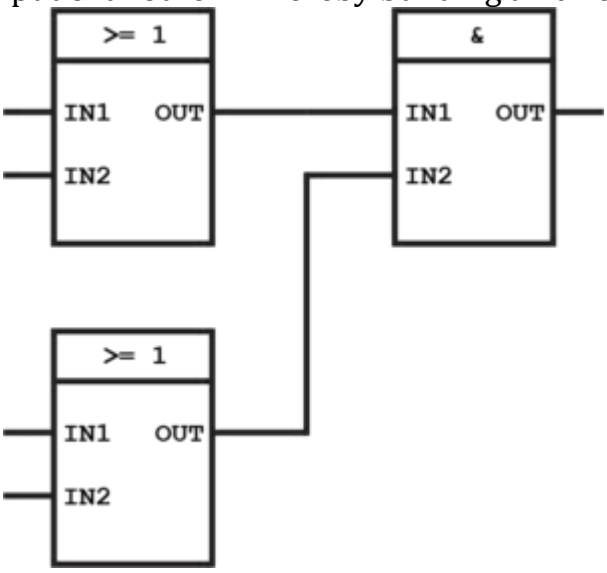
- Functional Block Diagram (FBD) is a simple and graphical method to program multiple functions in PLC.
- PLC Open has described using FBD in the standard IEC 61131-3.
- A function block is a program instruction unit which, when executed, yields one or more output values.
- It is represented by a block as shown below. It is represented as a rectangular block with inputs entering on left and output lines leaving at the right. It gives a relation between the state of input and output



Function Block

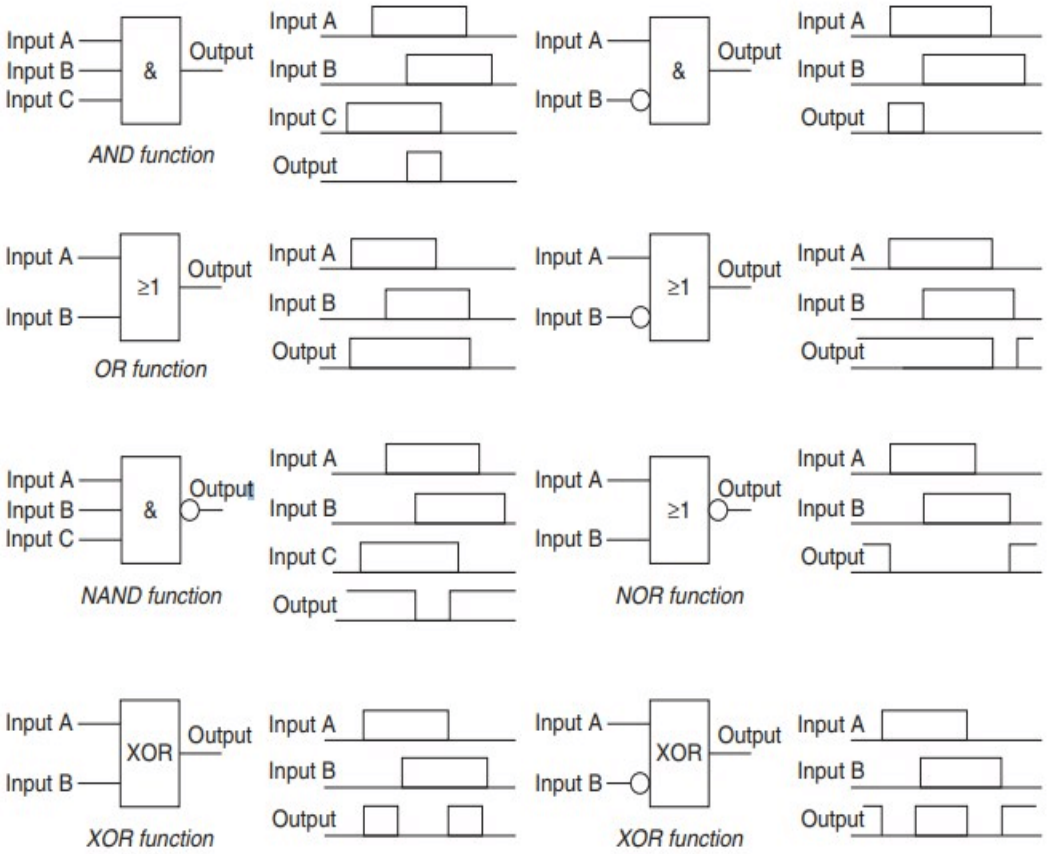
- The advantage of using FBD is that any number of inputs and outputs can be used on the functional block.

- When using multiple input and output, you can connect the output of one function block to the input of another. Whereby building a **Function Block Diagram**.



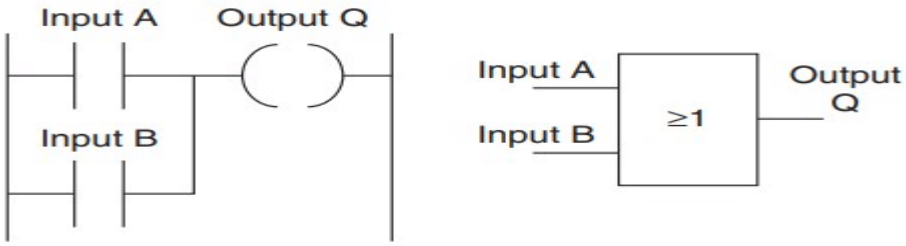
Example Functional Block Diagram

- The figure below shows various function blocks used in FBD programming.

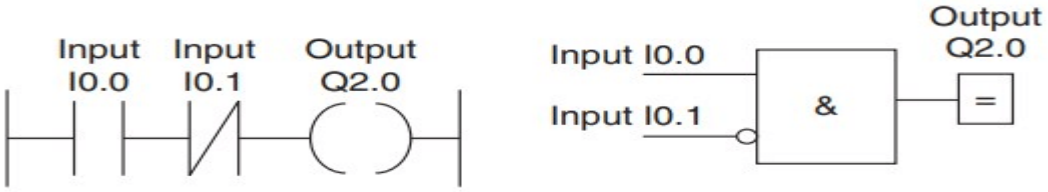


Functional Block Programming

- The figure below shows a ladder diagram and its function block equivalent in Siemens notation.



Ladder to functional block (Source)



Ladder to functional block diagram (Source)

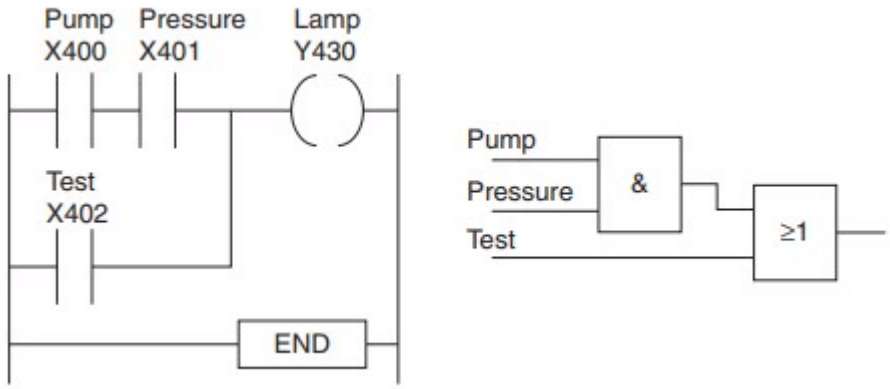
Structured Text Programming:

- Structured text is a textual programming language that utilizes statements to determine what to execute.
- It follows more conventional programming protocols but it is not case sensitive.
- A series of statements (logic) is constituted of expressing assignments and relationships using several operators.
- The structures text operators are listed below in the image.

Order	Operation
1.	( )
2.	function (…)
3.	**
4.	- (negate)
5.	NOT
6.	*, /, MOD
7.	+, - (subtract)
8.	<, <=, >, >=
9.	=, <>
10	&, AND
11.	XOR
12.	OR

Structured Text Programming
PLC Programming Example 1:

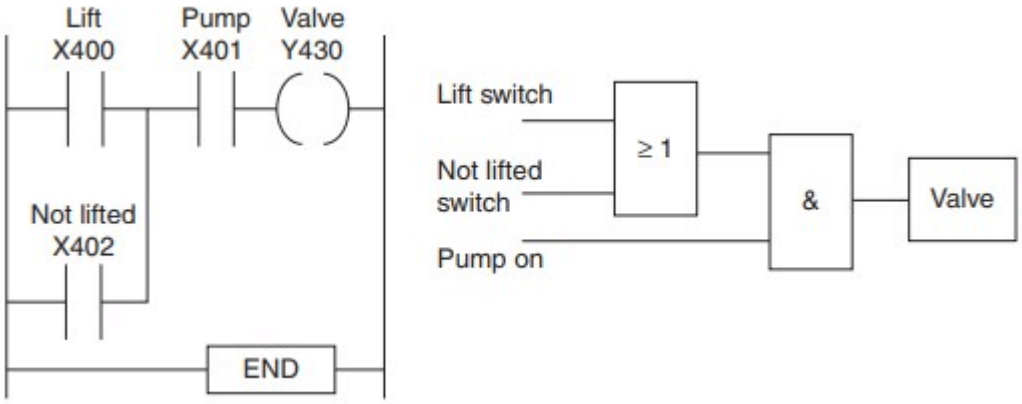
- A signal lamp is required to be switched on if a pump is running and the pressure is satisfactory, or if the lamp test switch is closed.
- In this application, if there should be an output from the lamp inputs from both pump and pressure sensors are required. Hence, AND logic is used.
- OR logic is used for the test input condition, it is required to give an output of lamp on regardless of whether there is a signal from the AND system.
- By using END or RET instruction in the ladder diagram, that means PLC has reached the end of the program.
- The function block diagram and the ladder diagram are shown below in the figure.



PLC Program to Test Lamp Glowing

Example 2:

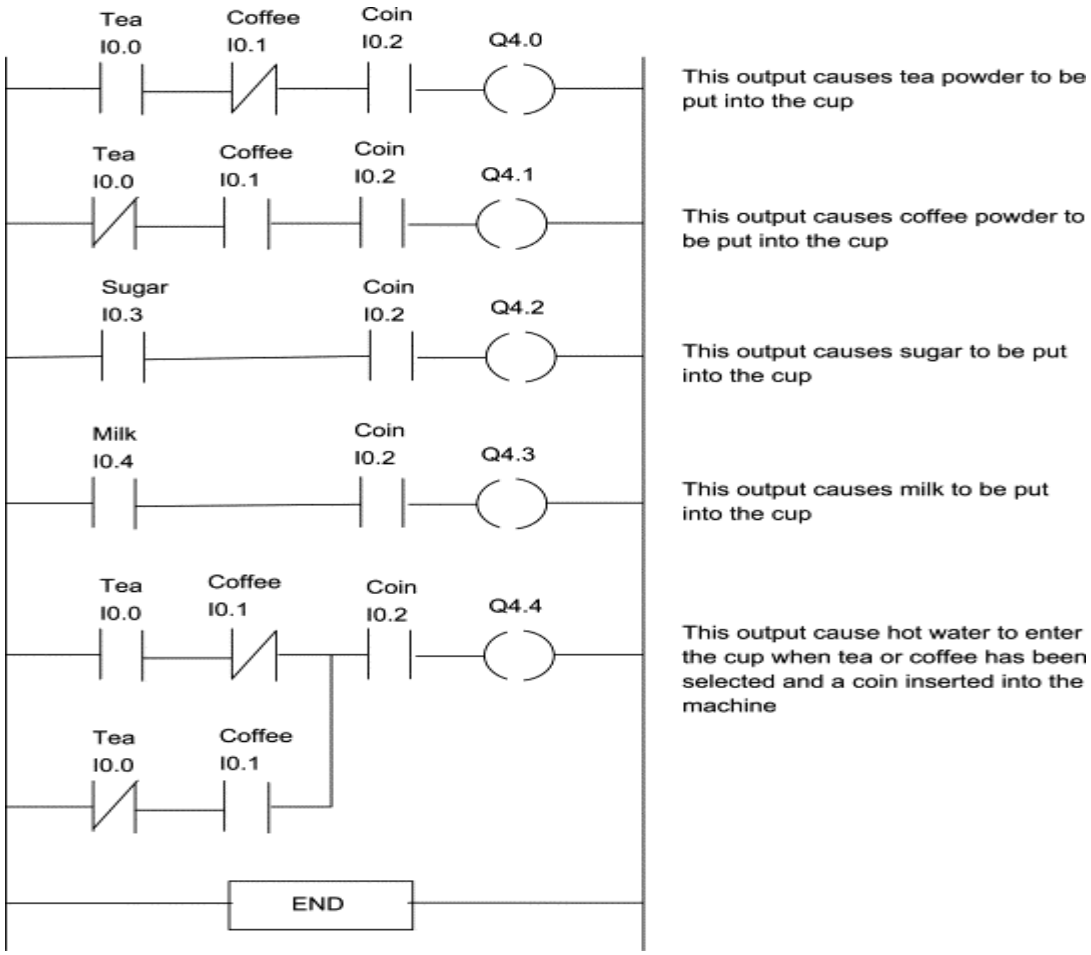
- consider a valve which is to be operated to lift a load when a pump is running and either the lift switch is operated or a switch operated indicating that the load has not already been lifted and is at the bottom of its lift channel.
- OR logic is used for two switches and an AND logic is used with two switches and the pump.
- Valve will be operated only if the pump is ON and two switches are operated.



PLC Program to Operate Valve

Example 3:

- Consider a drinks machine that allows the selection of tea or coffee, milk or no milk, sugar or no sugar, and will supply the required hot drink on the insertion of a coin.
- In given below figure, it is seen that either tea or coffee is selected using the first OR logic gate.
- The first AND gate give an output when either Tea or coffee is selected and a coin is inserted into the machine.
- The output from this AND gate is given to second AND gate.
- The second AND gate operate only when hot water combines with tea. Milk and sugar are optional additions which can occur after a coin has been inserted.

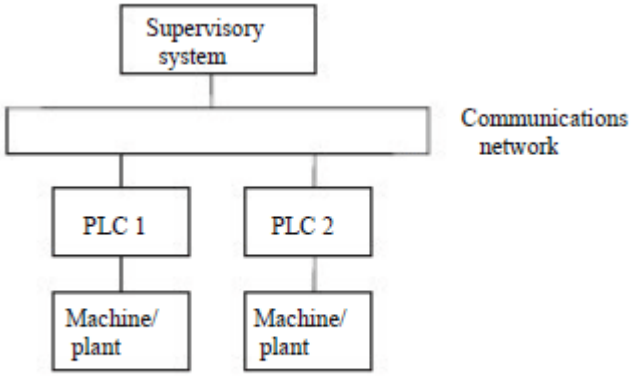


Ladder Logic for Drinking Machine Application

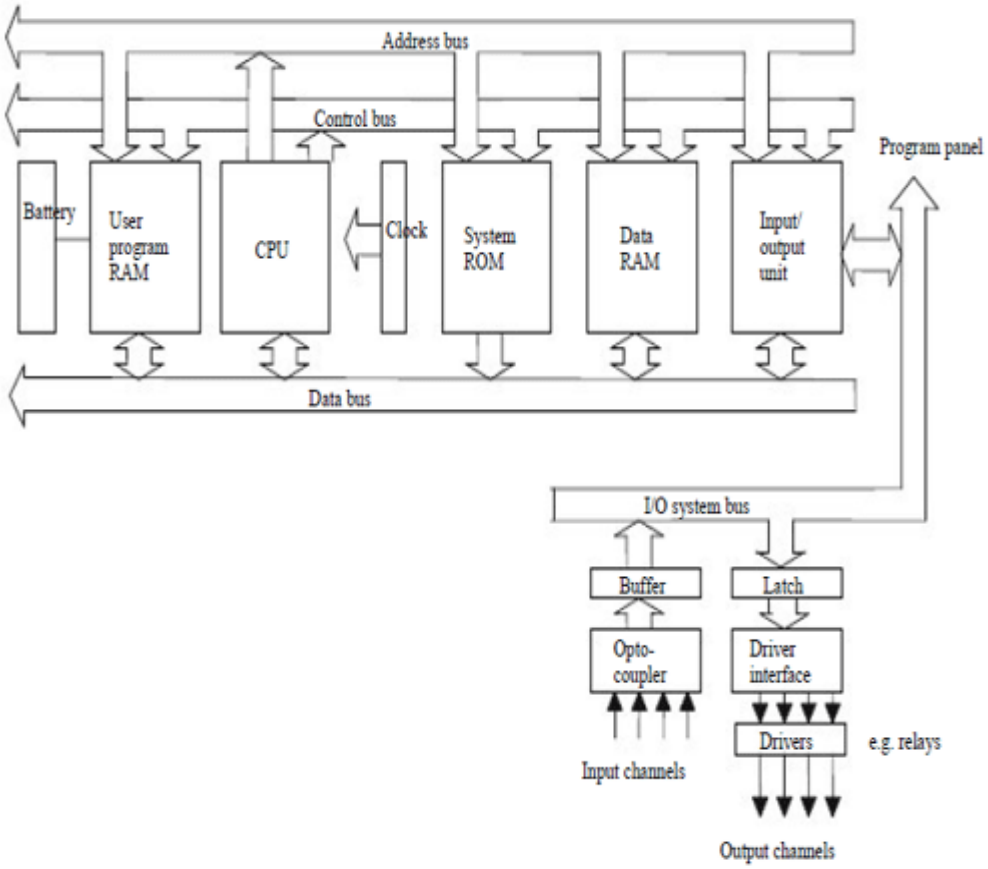
6.4 INTERNAL OF A PLC:

- It consists of a central processing unit (CPU) containing the system microprocessor, memory, and input/output circuitry.
- The CPU controls and processes all the operations within the PLC.
- It is supplied with a clock that has a frequency of typically between 1 and 8 MHz
- This frequency determines the operating speed of the PLC and provides the timing and synchronization for all elements in the system.
- The information within the PLC is carried by means of digital signals.
- The internal paths along which digital signals flow are called **buses**.
- In the physical sense, a bus is just a number of conductors along which electrical signals can flow. It might be tracks on a printed circuit board or wires in a ribbon cable.
- The CPU uses the data bus for sending data between the constituent elements, the **address bus** to send the addresses of locations for accessing stored data, and the **control bus** for signals relating to internal control actions.

- The system bus is used for communications between the input/output ports and the input/output unit.



Basic communications model.



Architecture of a PLC.

The CPU:

The internal structure of the CPU depends on the microprocessor concerned. In general, CPUs have the following:

- An arithmetic and logic unit (ALU) that is responsible for data manipulation and carrying out arithmetic operations of addition and subtraction and logic operations of AND, OR, NOT, and EXCLUSIVE-OR.
- Memory, termed registers, located within the microprocessor and used to store information involved in program execution.
- A control unit that is used to control the timing • of operations.

The Buses:

The buses are the paths used for communication within the PLC. The information is transmitted in binary form, that is, as a group of bits, with a bit being a binary digit of 1 or 0, indicating on/off states. The term word is used for the group of bits constituting some information. Thus an 8-bit word might be the binary number 00100110. Each of the bits is communicated simultaneously along its own parallel wire. **The system has four buses:**

- The data bus carries the data used in the processing done by the CPU. A microprocessor termed as being 8-bit has an internal data bus that can handle 8-bit numbers. It can thus perform operations between 8-bit numbers and deliver results as 8-bit values.

- The address bus is used to carry the addresses of memory locations. So that each word can be located in memory, every memory location is given a unique address. Just like houses in a town are each given a distinct address so that they can be located, so each word location is given an address so that data stored at a particular location can be accessed by the CPU, either to read data located there or put, that is, write, data there. It is the address bus that carries the information indicating which address is to be accessed. If the address bus consists of eight lines, the number of 8-bit words, and hence number of distinct addresses, is 28 ¼ 256. With 16 address lines, 65,536 addresses are possible.

- The control bus carries the signals used by the CPU for control, such as to inform memory devices whether they are to receive data from an input or output data and to carry timing signals used to synchronize actions.

The system bus is used for communications between the input/output ports and the input/ output unit.

Memory:

To operate the PLC system there is a need for it to access the data to be processed and instructions, that is, the program, which informs it how the data is to be processed. Both are stored in the PLC memory for access during processing. There are several memory elements in a PLC system:

- System read-only-memory (ROM) gives permanent storage for the operating system and fixed data used by the CPU.
- Random-access memory (RAM) is used for the user's program.
- Random-access memory (RAM) is used for data. This is where information is stored on the status of input and output devices and the values of timers and counters and other internal devices. The data RAM is sometimes referred to as a data table or register table. Part of this memory, that is, a block of addresses, will be set aside for input and output addresses and the states of those inputs and outputs. Part will be set aside for preset data and part for storing counter values, timer values, and the like.

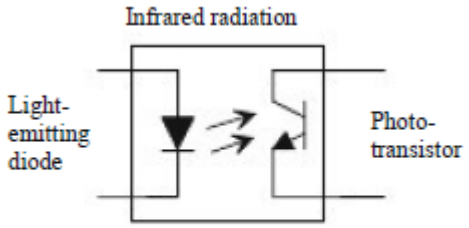
- Possibly, as a bolt-on extra module, erasable and programmable read-only-memory (EPROM) is used to store programs permanently.

- The programs and data in RAM can be changed by the user. All PLCs will have some amount of RAM to store programs that have been developed by the user and program data. However, to prevent the loss of programs when the power supply is switched off, a battery is used in the PLC to maintain the RAM contents for a period of time. After a program has been developed in RAM it may be loaded into an EPROM memory chip, often a bolt-on module to the PLC, and so made permanent. In addition, there are temporary buffer stores for the input/output channels.
- The storage capacity of a memory unit is determined by the number of binary words that it can store. Thus, if a memory size is 256 words, it can store 256 Â 8 ¼ 2048 bits if 8-bit words are used and 256 Â 16 ¼ 4096 bits if 16-bit words are used. Memory sizes are often specified in terms of the number of storage locations available, with 1K representing the number 210, that is, 1024. Manufacturers supply memory chips with the storage locations grouped in groups of 1, 4, and 8 bits. A 4K Â 1 memory has 4 Â 1 Â 1024 bit locations.

- A 4K Â 8 memory has 4 Â 8 Â 1024 bit locations. The term byte is used for a word of length 8 bits. Thus the 4K Â 8 memory can store 4096 bytes. With a 16-bit address bus we can have 216 different addresses, and so, with 8-bit words stored at each address, we can have 216 Â 8 storage locations and so use a memory of size 216 Â 8/210 ¼ 64K Â 8, which might be in the form of four 16K Â 8-bit memory chips.

Input/output Unit:

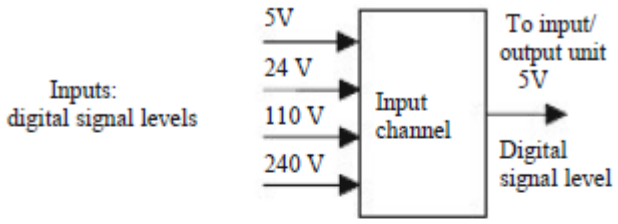
- The input/output unit provides the interface between the system and the outside world, allowing for connections to be made through input/output channels to input devices such as sensors and output devices such as motors and solenoids. It is also through the input/output unit that programs are entered from a program panel. Every input/output point has a unique address that can be used by the CPU. It is like a row of houses along a road; number 10 might be the "house" used for an input from a particular sensor, whereas number 45 might be the "house" used for the output to a particular motor.
- The input/output channels provide isolation and signal conditioning functions so that sensors and actuators can often be directly connected to them without the need for other circuitry.



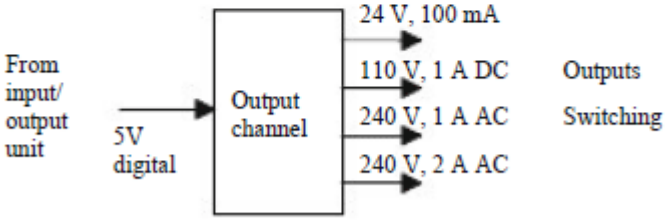
An optoisolator.

- Electrical isolation from the external world is usually by means of optoisolators (the term optocoupler is also often used) shows the principle of an Optoisolator. When a digital pulse passes through the light-emitting diode, a pulse of infrared radiation is produced.
- This pulse is detected by the photo transistor and gives rise to a voltage in that circuit. The gap between the light-emitting diode and the photo transistor gives electrical isolation, but the arrangement still allows for a digital pulse in one circuit to give rise to a digital pulse in another circuit.
- The digital signal that is generally compatible with the microprocessor in the PLC is 5 V DC. However, signal conditioning in the input channel, with isolation, enables a wide range of input signals to be supplied to it (see Chapter 3 for more details). A range of inputs might be available with a larger PLC, such as 5 V, 24 V, 110 V, and 240 V digital/discrete, that is, on/ off, signals. A small PLC is likely to have just one form of input, such as 24 V.
- The output from the input/output unit will be digital with a level of 5 V. However, after signal conditioning with relays, transistors, or triacs, the output from the output channel might be a 24 V, 100 mA switching signal; a DC voltage of 110 V, 1 A; or perhaps 240 V, 1 A AC or 240 V, 2 A AC, from a triacs output channel. With a small PLC, all the outputs might be of one type, such as 240 V, 1 A AC. With modular

PLCs, however, a range of outputs can be accommodated by selection of the modules to be used.



Input levels.



Output levels.

Outputs are specified as being of relay type, transistor type, or triacs type:

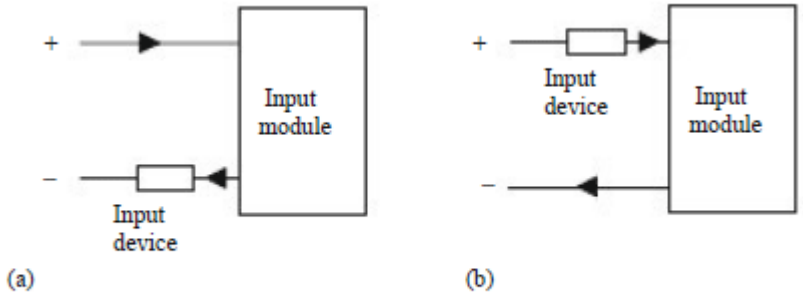
- With the relay type, the signal from the PLC output is used to operate a relay and is able to switch currents of the order of a few amperes in an external circuit. The relay not only allows small currents to switch much larger currents but also isolates the PLC from the external circuit. Relays are, however, relatively slow to operate. Relay outputs are suitable for AC and DC switching. They can withstand high surge currents and voltage transients.

- The transistor type of output uses a transistor to switch current through the external circuit. This gives a considerably faster switching action. It is, however, strictly for DC switching and is destroyed by over current and high reverse voltage. For protection, either a fuse or built-in electronic protection is used. Optoisolators are used to provide isolation.

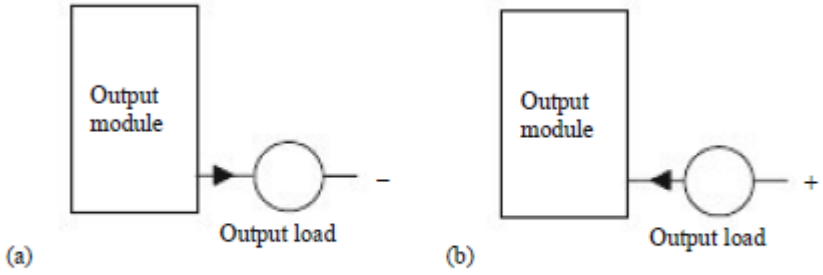
- Triacs outputs, with optoisolators for isolation, can be used to control external loads that are connected to the AC power supply. It is strictly for AC operation and is very easily destroyed by over current. Fuses are virtually always included to protect such outputs.

Sourcing and Sinking

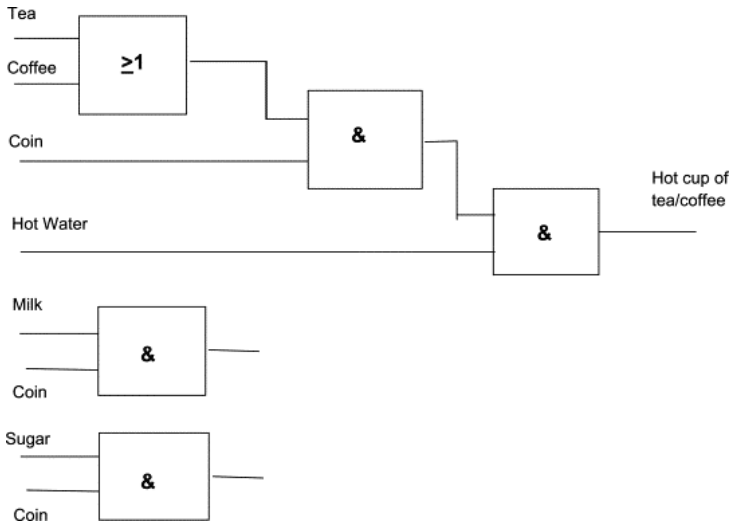
- The terms sourcing and sinking are used to describe the way in which DC devices are connected to a PLC.
- With sourcing, using the conventional current flow direction as from positive to negative, an input device receives current from the input module, that is, the input module is the source of the current.
- With sinking, using the conventional current flow direction, an input device supplies current to the input module, that is, the input module is the sink for the current.
- If the current flows from the output module to an output load, the output module is referred to as sourcing.
- If the current flows to the output module from an output load, the output module is referred to as sinking



Inputs: (a) sourcing; (b) sinking.



Outputs: (a) sourcing; (b) sinking.



FBD for Drinking Machine

6.5 STATE THE DIFFERENCE BETWEEN A PROGRAMMABLE CONTROLLER AND A COMPUTER

PLC:

- The input- output capabilities are large and can be used in various industrial processes.
- The input-output cards are mounted on racks and are visible.
- The noise is more.
- The possibility of mishap exists as input-output cards are on the rack.
- The connections are rugged, accessible and organized.
- The processor architecture is simple.
- It is used for controlling industrial processes and control logic programming.
- The interface is by using simple devices such as indicators push buttons etc.
- It uses ladder programming.
- It has limited expandability.

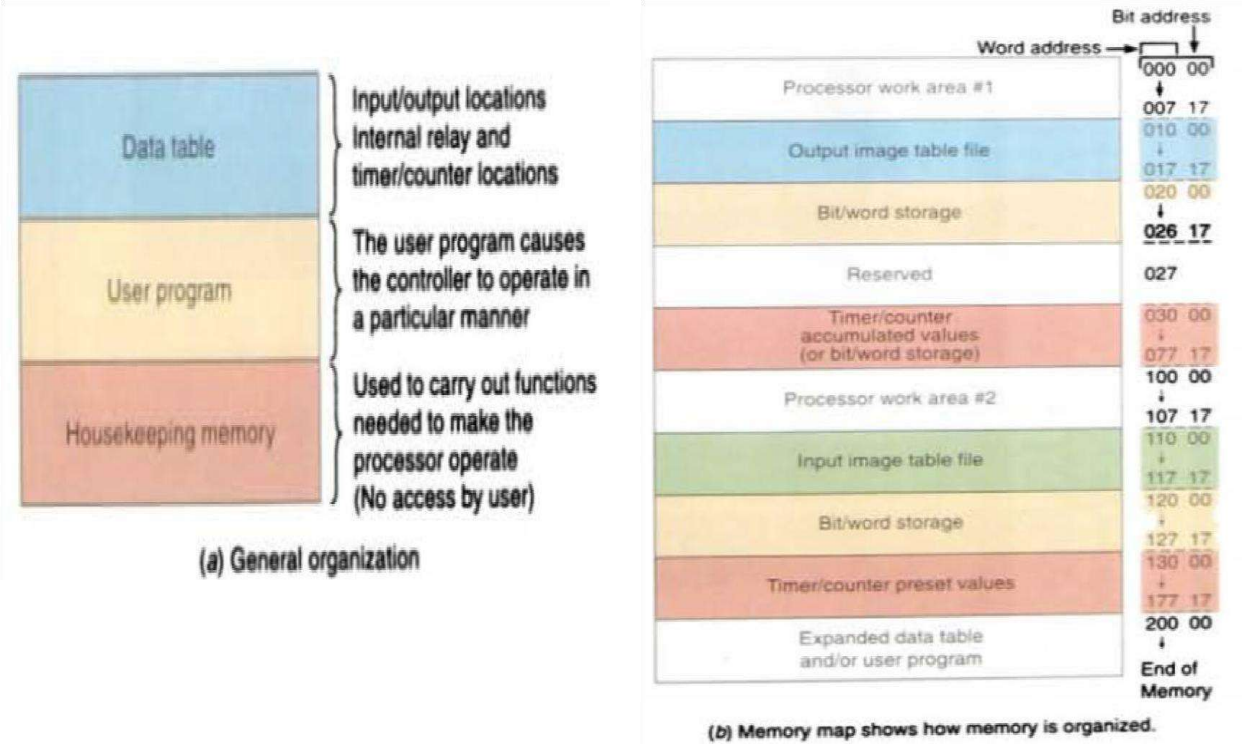
PC:

- Limited input-output capabilities in terms of analog and digital cards which are suited for laboratory purposes.
- The input-output cards are mounted on board and are not visible.
- Noise is less
- No possibility of mishap
- There exists a bunch of cables and very small room available for cable connections.
- The processor architecture is complicated.
- It is used for many advanced computation and high level programming language is used.

- The easy and good interface is not possible. It uses microprocessor programming
- It has great flexibility and high reliability.

6.6 PLC MEMORY ORGANIZATION:

- The term processor memory organization refers to how certain areas of memory in a given PLC are used.
- Figure given below shows an illustration of the Allen-Bradley PLC-2 memory organization, known as a **memory map**.
- Every PLC has a memory map. The memory space can be divided into two broad **categories:** the **user program** and the **data table**.
- The individual sections, their order, and the sections' length will vary and may be fixed or variable, depending on the manufacturer and model.



Memory map for Allen-Bradley PLC-2

- The user program is where the logic ladder program is entered and stored. The user program will account for most of the total memory of a given PLC system. It contains the logic that controls the machine operation. This logic consists of instructions that are programmed in a ladder logic format. Most instructions require one word of memory.
- The data table stores the information needed to carry out the user program. This includes information such as the status of input and output devices, timer and counter values, data storage, and so on. Contents of the data table can be divided into two categories: status data and numbers or codes. Status is ON/OFF type of information represented by 1s and 0s, stored in unique bit locations. Number or code information is represented by groups of bits that are stored in unique byte or word locations.
- A processor file is the collection of program files and data files created under a particular processor file name. It contains all the instructions, data, and configuration information pertaining to a user program.

Figure shown below shows typical program and data file memory organization for an Allen Bradley SLC-500 controller. The contents of each file are outlined in the sections that follow.

Program files:

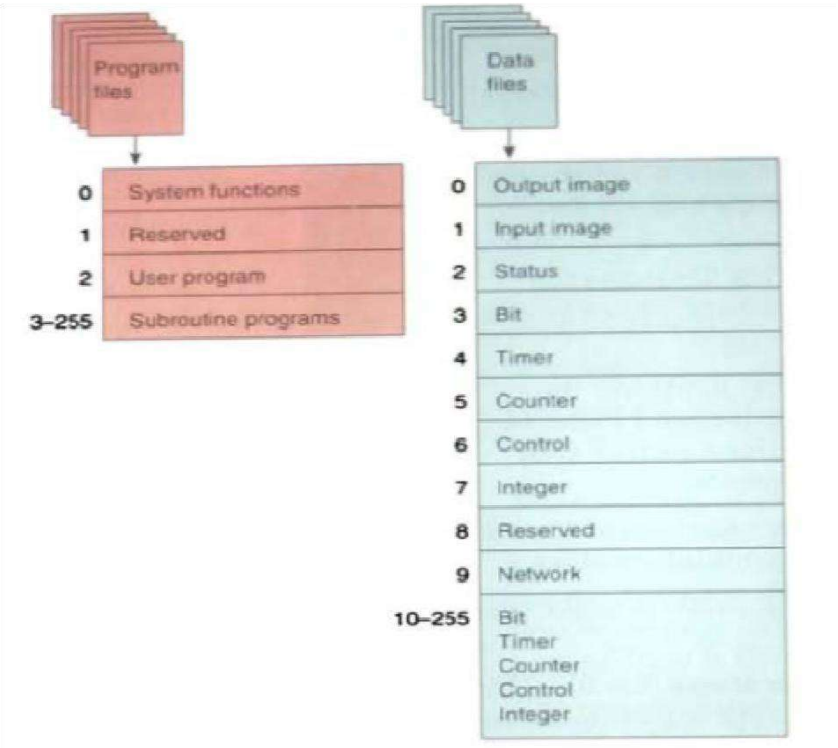
Program files are the areas of processor memory where ladder logic programming is stored. They may include:

System functions (file 0): This file is always included and contains various system related information and user programmed information such as processor type. I/O configuration processor files name, and password.

Reserved (file 1): This file is reserved by the processor and is not accessible to the user.

Main ladder program (file 2): This file is always included and contains user programmed instructions that define how the controller is to operate.

Subroutine ladder program (files3- 255): These files are user-created and are activated according to subroutine instructions residing in the main ladder program file.



Program and data file memory organization for an Allen-Bradley

SLC-500 controller Data Files:

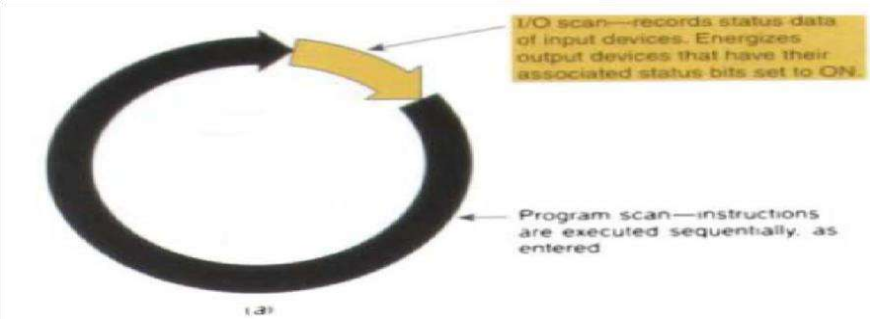
The data file portion of the processor's memory stores input and output status, processor status, the status of various bits, and numerical data. All this information is accessed via the ladder logic program. These files are organized by the type of data they contain and may include:

- **Output (file 0):** This file stores the state of the output terminals for the controller.
- **Input (file 1):** This file stores the status of the input terminals for the controller.
- **Status (file 2):** This file stores controller operation information. This file is useful for troubleshooting controller and program operation.
- **Bit (file 3):** This file is used for internal relay logic storage.
- **Timer (file 4):** This file stores the timer accumulated and preset values and status bits.
- **Counter (file 5):** This file stores the counter accumulated and preset values and status bits.
- **Control (file 6)**: This file stores the length, pointer position, and status bit for specific instructions such as shift registers and sequencers.

- **Integer (file 7):** This file is used to store numerical values or bit information.
- **Reserved (file 8):** This file is not accessible to the user.
- **Network communications (file 9):** This file is used for network communications if installed or used like files 10-255.
- **User-defined (files 10-255):** These files are user-defined as bit, timer, counter, control, and/or integer data storage.

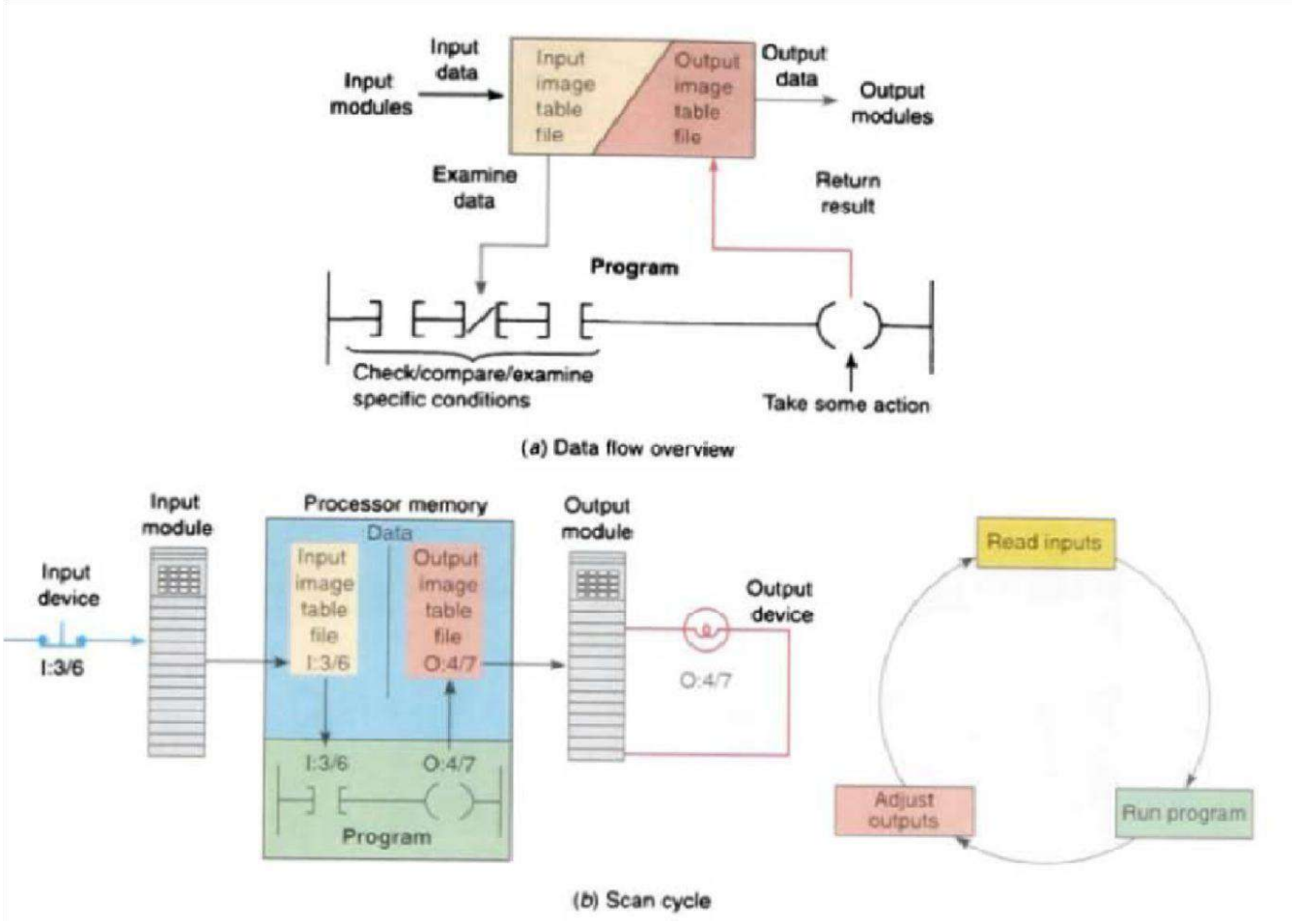
6.7 PROGRAM SCAN OF A PLC:

- During each operating cycle, the processor reads all the inputs, takes these values, and energizes or de-energizes the outputs according to the user program. This process is known as a scan. Figure shown below shows a single PLC scan, which consists of the I/O scan and the program scan. Because the inputs can change at any time, the PLC must carry on this process continuously.



Single PLC scan

- The PLC scan time specification indicates how fast the controller can react to changes in inputs. Scan time varies with program content and length. The time required to make a single scan can vary from about 1ms to 20ms. If a controller has to react to an input signal that changes states twice during the scan time, it is possible that the PLC will never be able to detect this change.
- For example, if it takes 8ms for the CPU to scan a program, and an input contact is opening and closing every 4ms, the program may not respond to the contact changing state. The CPU will detect a change if it occurs during the update of the input image table file, but the CPU will not respond to every change. The scan is normally a continuous and sequential process of reading the status of inputs, evaluating the control logic, and updating the outputs.
- Figure shown below illustrates this process.



Scan Process

- When the input device connected to address I:3/6 is closed, the input module circuitry module circuitry senses a voltage and a 1 (ON) condition is entered into the input image table bit I:3/6. During the program scan, the processor examines bit I: 3/6 for a 1 (ON) condition.
- In this case, because input I : 3/6 is 1, the rung is said to be TRUE. The processor then sets the output image table bit O: 4/7 to 1. The processor turns on output O: 4/7 during the next I/O scan, and the output device (light) wired to this terminal becomes energized. This process is repeated as long as the processor is in the RUN mode. If the input device were to open, a 0 would be placed in the input image table. As a result, the rung would be called FALSE. The processor would then set the output image table bit O: 4/7 to 0, causing the output device to turn off.

6.8 INTERNAL PLC INSTRUCTIONS:

Relay-type (Basic) instructions: I, O, OSR, SET, RES, T, C

Data Handling Instructions:

- **Data move Instructions:** MOV, COP, FLL, TOD, FRD, DEG, RAD (degrees to radian).
- **Comparison instructions**: EQU (equal), NEQ (not equal), GEQ (greater than or equal), GRT (greater than).
- Mathematical instructions.
- Continuous Control Instructions (PID instructions).

- **Program flow control instructions**: MCR (master control reset), JMP, LBL, JSR, SBR, RET, SUS, REF
- **Specific instructions:** BSL, BSR (bit shift left/right), SQO (sequencer output), SQC (sequencer compare), SQL (sequencer load).
- **High speed counter instructions:** HSC, HSL, RES, HSE
- **Communication instructions:** MSQ, SVC
- **ASCII instructions:** ABL, ACB, ACI, ACL, CAN

Internal Relays:

- Auxiliary relays, markers, flags, coils, bit storage.
- Used to hold data, and behave like relays, being able to be switched on or off and switch other devices on or off.
- They do not exist as real-world switching devices but are merely bits in the storage memory.

Internal Relays Use:

- In programs with multiple input conditions or arrangements.
- For latching a circuit and for resetting a latch circuit. Giving special built-in functions with PLCs.

Retentive relays (battery-backed relays):

Such relays retain their state of activation, even when the power supply is off. They can be used in circuits to ensure a safe shutdown of plant in the event of a power failure and so enable it to restart in an appropriate manner.

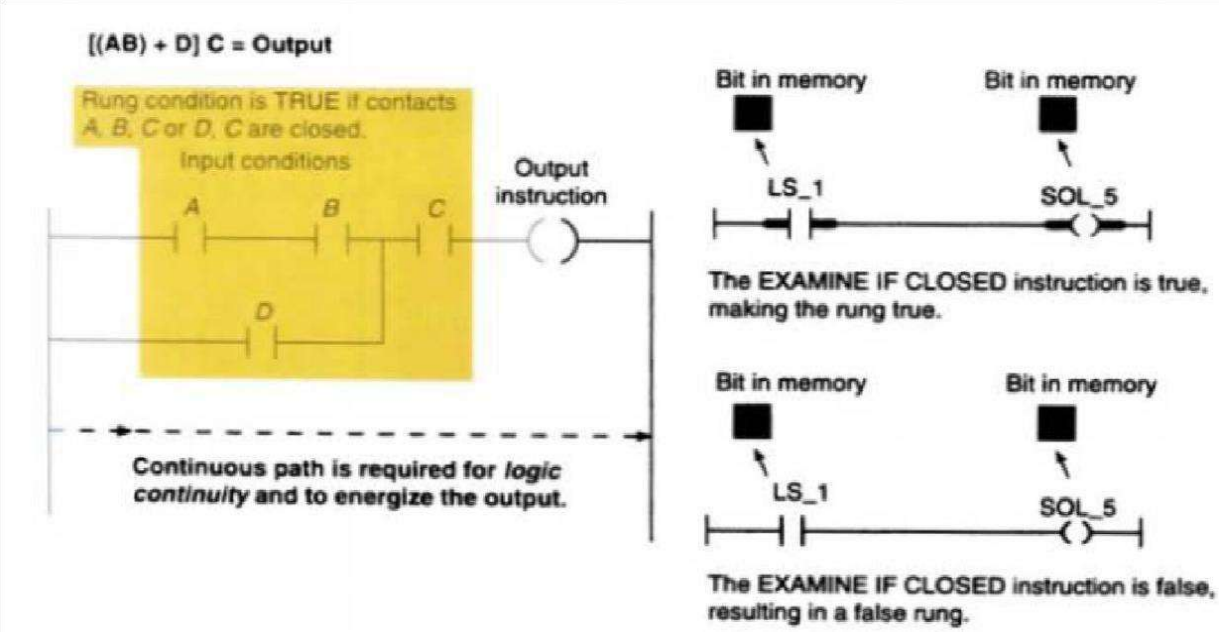
Latch Instructions (Set and Reset)

- The set instruction causes the relay to self-hold, i.e. latch. It then remains in that condition until the reset instruction is received.

- The latch instruction is often called a SET or OTL (output latch).
- The unlatch instruction is often called a RES (reset), OTU (output unlatch) or RST (reset).

6.9 LADDER RUNG DIAGRAM:

- The main function of the ladder logic diagram program is to control outputs based on input conditions. This control is accomplished through the use of what is referred to as a ladder rung. In general, a rung consists of a set of input conditions, represented by contact instructions, and an output instruction at the end of the rung, represented by the coil symbol (Fig. given below).



Ladder rung

- Each contact or coil symbol is referenced with an address number that identifies what is being evaluated and what is being controlled. The same contact instruction can be used throughout the program whenever that condition needs to be evaluated.
- For an output to be activated or energized, at least one left-to-right path of contacts must be closed. A complete closed path is referred to as having logic continuity.
- When logic continuity exists in at least one path, the rung condition is said to be TRUE. The rung condition is FALSE if no path has continuity.
- During controller operation, the processor determines the ON/OFF state of the bits in the data files, evaluates the rung logic, and changes the state of the outputs

according to the logical continuity of rungs. More specifically, input instructions set up the conditions under which the processor will make an output instruction true or false.

- **These conditions are as follows:**

1. When the processor finds a continuous path of true input instructions in a rung, the OUTPUT ENERGIZE [OTE] output instruction will become (or remain) true. We then say that rung conditions are TRUE.
2. When the processor does not find a continuous path of true input instructions in a rung, the OTE input instruction will become (or remain) false. We then say that rung conditions are FALSE.

6.10 PLC TIMER:

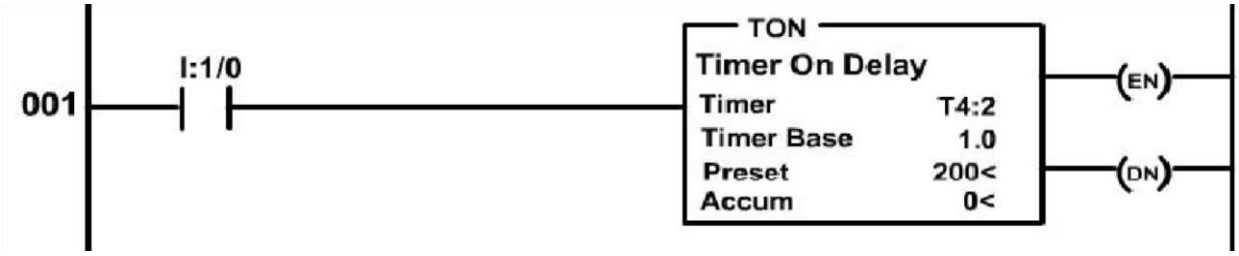
- All PLC's have timer instructions.
- Timers are output instructions that are internal to the programmable logic controller.
- Timers provide timed control of the devices that they activate or de-activate.

Basic functions of timer:

- Timers are used to delay an action.
- Timers are used to run an operation for a predetermined period of time.
- Timers are also used to record the total accumulated time of continuous or intermediate events.

Timer's Instructions:

- Timers consists of following parts: timer address, preset value, timer base, and accumulated value, as shown in figure below.



- In the above figure, term instruction name is timer on delay (TON), timer base is 1.0 seconds, and timer address is T4:0, accumulated value of zero (0) and a preset value of 200.
- Each timer instruction has three very useful status bits. These bits are timer enable (E N), timer timing (TT), and timer done (DN).

- There are 3 types of timers: **On- delay timer**, **Off-delay timer, and retentive timer.**

On delay timer:

- Use this instruction to program a time delay after instructions become true.
- On – delay timers are used when an action is to begin a specified time after the input becomes true.
- For example, a certain step in the manufacturing is to begin 45 seconds after a signal is received from a limit switch. The 45- seconds delay is the on-delay timers preset value.

Off- delay timer:

- Off- delay timer instructions is used to program a time delay to begin after rung input goes false.
- As an example, when an external cooling fan on a motor is provided, the fan has to run all the time the motor is running and also for certain time (say 10min) after the motor is turned off. This is a ten minute off- delay timer. The ten-minute timing period begins as soon as the motor is turned off.

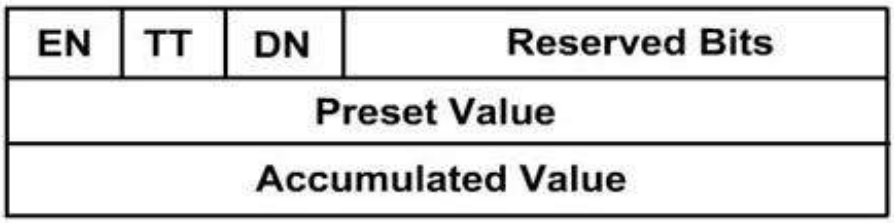
Retentive timer:

- Retentive timer is a timer which retains the accumulated value in case of power loss, change of processor mode or rung state going from true to false (rung state transition).
- Retentive timer can be used to track the running time of a motor for its maintenance purpose.
- Each time the motor is turned off, the timer will remember the motor's elapsed running time.
- The next time the motor is turned on, the time will increase from there. This timer can be reset by using a **reset instruction.**

Reset:

- This instruction is used to reset the accumulated value of counter or timer.
- It is used to reset a retentive timer's accumulated value to zero. A typical timer element:-
- A timer element is made up of three 16 bit words:
  - Word 0: 3 status bits (EN, TT, DN).
  - Word 1: Preset values.

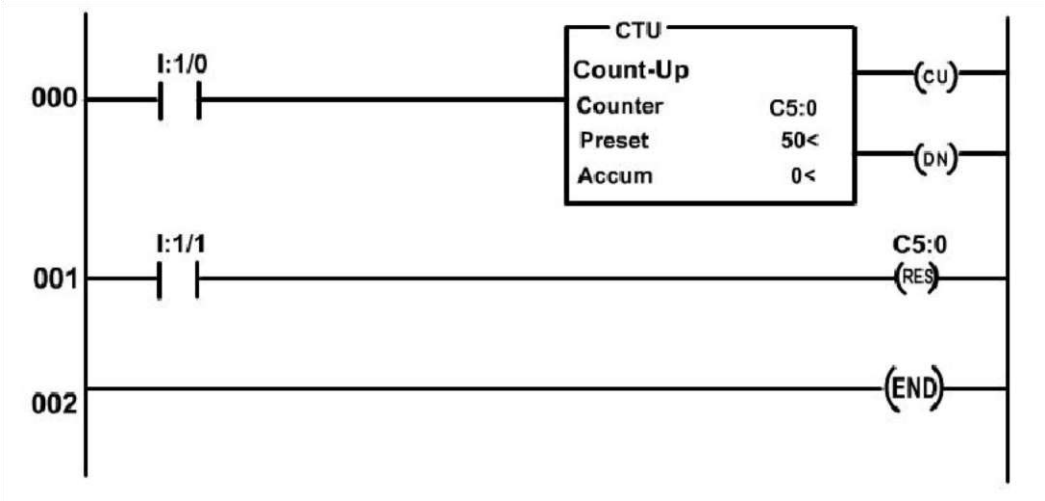
- Word 2: Accumulated value.



6.11 PLC as a COUNTER:

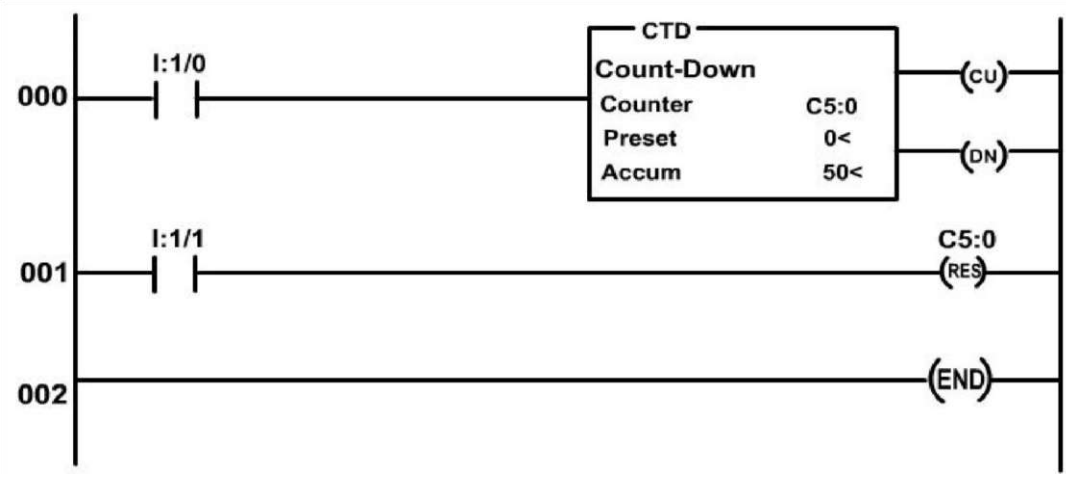
- A counter is a simple device intended to do one simple thing-count.
- Every PLC has counter instructions. Using counters sometimes be little challenging because many manufacturers seem to use them different way.
- In other words, the instruction symbol used and method of programming will change for different manufacturers.
- A typical counter counts from 0 up to a predetermined value, called the "preset" value.
- **For example,** if you wanted to count from certain value, say from 0 to 50, it could be counting up using a count-up or up-counter.
- Here, the number "50" is the predetermined value, which is nothing but preset value.
- The current count or accumulated count is called as the "accumulated value". If the counter had counted 25 pieces that had passed on the conveyor, the accumulated count would be 25.
- When all 50 pieces had passed on the conveyor, the preset value and counter accumulated value would be equal.
- At this point the counter would signal other logic within the PLC program that the batch of 50 was completed and it should now take some action.
- The next action the PLC has to take is to move the box containing 50 parts on to the next station for carton sealing.
- To start counting the next batch, a reset instruction would be used to reset the counter's accumulated value back to zero.

- The fig below shows an **up-counter**, counting from 0 to 50.



- The fig below shows a **down-counter**, counting from 50 to 0.

-.

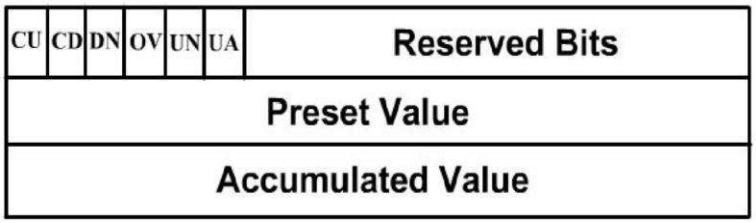


PLC Counter Instructions

Instruction	This instruction is used to	Functional description
Count Down	Count down from a desired value to zero	An operator interface display shows the operator the number of parts remaining to be made for a lot of,say 50 parts ordered.
Count Up	Count from zero up to a desired value	Counting the number of parts produced during a specific work shift or batch.Also counting the number of rejects from a batch.
High-speed Counter	Count input pulses that are too fast,from normal input points and modules	Most fixed programmable logic controllers have a high-speed set of input points that allow interface to high-speed inputs. Signals from an incremental encoder would be a typical high-speed input.
Counter Reset	To reset a counter or timer	Used to reset a counter to zero so that another counting sequence can begin.

As for explanation, let us take ALLEN-BRADELY counters:

- In Allen-Bradley counter, the default counter file is file 5.
- The counter data is stored in counter file. Each counter consists of three 16-bit words and is known as “counter element”.
- In a single processor file, there can be many counter files. Any data file, which is greater than file 8 can be assigned as an additional counter file.
- Each counter file can have up to 256 counter elements. A counter instruction is one element.
- A counter element is made up of three 16-bit words. Thus, the counter instruction contains the three parts i.e. **word0, word1 and word2.**
    - **Word zero** is for status bits. Status bits include CU, CD, DN, OV, UN, and UA. Along with their associated instructions.
    - **Word one** is for the preset value.
    - **Word two** is for accumulated value.



Addressing a counter:

- To address the counter as a unit the addressing format used is C5:4.
  - Where, C= C identifies this as a counter file.
  - 5= this is counter file 4 which is default. Any unused file from 10 to 255 can be assigned to counters.
  - : 4= the colon used here is called the file separator. It separates the file, file 5, from the specific counter, in this case, counter 4 in counter file 5.
- To address the counter 14's accumulated value, the address used is C5:14.ACC.
  - Where, C= C identifies this as a counter file.
  - 5= this represents the counter file 5.
  - : 14= the colon is called the file separator. The colon separates the file, file 5, from the specific counter; in this case, counter 14 in counter file 5.
  - . = The point is called the word delimiter. The word delimiter is used to separate the counter number, called the structure, from the sub element. The sub element is ACC for the accumulated value. Similarly, preset value can be accessed as C5:14.PRE.
- The addressing format of counter status bits is as follows:
- C5:14/DN is the address for counter file 5, counter 14's done bit.
- C5:14/CU is the address for counter file 5, counter 14's count-up-enable bit.
- C5:14/EN is the address for counter file 5, counter 14's enable bit.

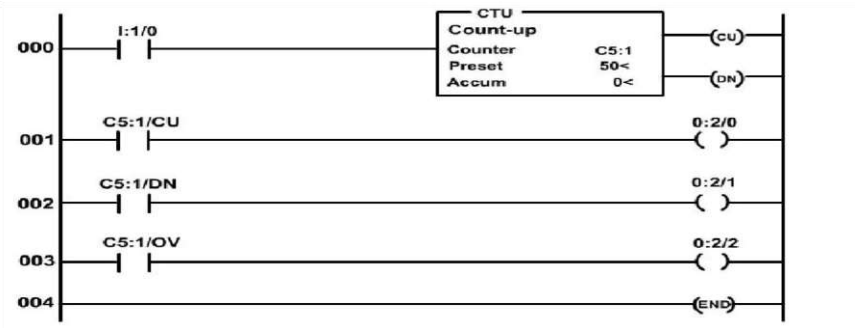
Working of a counter:

- A counter instruction is always an output instruction. The counter instruction counts each time the input logic changes the rung state from false to true. This input logic can be signal from an external device, for e.g. limit switch or sensor, or a signal from internal logic. Every time the counter instruction sees a false-to-true rung transition, a count-up counters accumulated value is incremented by one.
- The working of down-counter is little different. Each time when count-down counter sees a false-to-true rung transition, its accumulated value is decremented

by one. Since the accumulated value gets decremented by 1 when each time the input logic changes the rung state from false to true, the accumulated value must be chosen as the starting point of the count.

- Counters are retentive in nature. The counter will retain its accumulated value or the on or off status of the done, overflow and underflow bits through a power loss.

The count-up instruction:

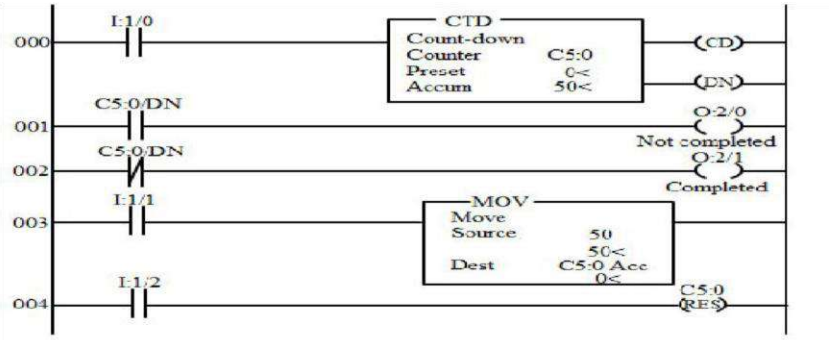


- The count-up instruction is used if we want a counter to increment one decimal value each time it register a rung transition from false to true.
- Each time input I: 1/0 has a transition from off to on, counter C5:1 will increment its accumulated value by one decimal value.
- The count-up-enabled bit, on rung 001 is set when the rung conditions are true, or enabled. In rung 002, the done bit, DN, is set when the accumulated value is equal to or greater than the preset value.
- In the event of wrap from +32,767 to -32,768, the accumulated value becomes less than the preset value and the done bit will not be reset.
- In rung 003, the count-up overflow bit, OV, is set whenever the count up counters accumulated value wraps from +32,767 to -32,768.

The count-down instruction:

- This instruction is used when we want to count down over the range of +32,767 to -32,768.
- Accumulated value will be decremented by one count, each time the instruction sees a false-to-true transition.
- Count-down instruction has many applications,
- **for example:** if we want to display the remaining number of parts to be filled for a specific order say 50 parts, then, a count-down instruction can be used. In this example, the accumulated value will be set as 50 and the preset value will be 0.

Each time a part is completed and passes the sensor, the accumulated value will be decremented by one decimal value, as shown in the figure below.



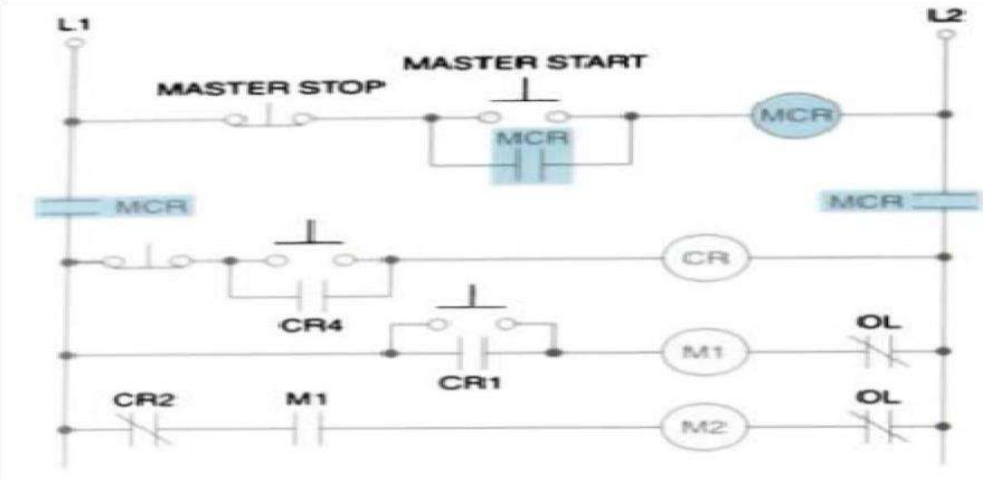
6.12 CONTROL INSTRUCTIONS OF PLC:

- Figure given below shows typical program control instructions.
- Instructions comprising the override instruction group include the master control reset (MCR) and jump (JMP) instructions.
- These operations are accomplished by using a series of conditional and unconditional branches and return instructions.

Command	Name	Description
JMP	Jump to Label	Jump forward/backward to a corresponding label instruction
LBL	Label	Specifies label location
JSR	Jump to Subroutine	Jump to a designated subroutine instruction
RET	Return from Subroutine	Exits current subroutine and returns to previous condition
SBR	Subroutine	Identifies the subroutine program
TND	Temporary End	Makes a temporary end that halts program execution
MCR	Master Control Reset	Clears all set outputs between the paried MCR instructions
SUS	Suspend	Identifies specific conditions for program debugging and system troubleshooting

- Hardwired master control relays are used in relay circuitry to provide input/output power shutdown of an entire circuit. Figure shown below is a typical hardwired master control relay circuit. In this circuit, unless the master control relay coil is energized, there is no power flow to the load side of the MCR contacts.
- The master control relay circuit shown in Figure below could not be programmed into the PLC as it appears because it contains two vertical contacts.
- For this reason, most PLC manufacturers include some form of master control relay as part of their instruction set.
- These instructions function in a similar manner to the hardwired master control relay; that is, when the instruction is true, the circuit functions normally, and when the instruction is false, outputs are switched off. Because these instructions are not hardwired but programmed, for safety reasons they should not be used as a

substitute for a hard· wired master control relay, which provides emergency I/O power shutdown



Hardwired master control relay circuit

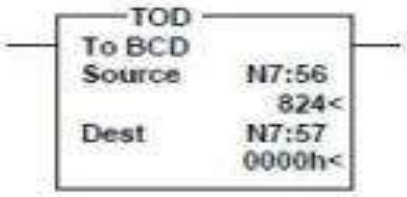
- The master control reset instruction can be programmed to control an entire circuit or to control only selected rungs of a circuit.
- The MCR is programmed to control an entire circuit. When the MCR instruction is false, or de-energized, all non-retentive (non-latched) rungs below the MCR will be de-energized even if the programmed logic for each rung is true.
- All retentive rungs will remain in their last state. The MCR instruction establishes a zone in the user program in which all non-retentive outputs can be turned off simultaneously.
- Therefore, retentive instructions should not normally be placed within an MCR zone because the MCR zone maintains retentive instructions in the last active state when the instruction goes false.

6.13 DATA HANDLING INSTRUCTION:

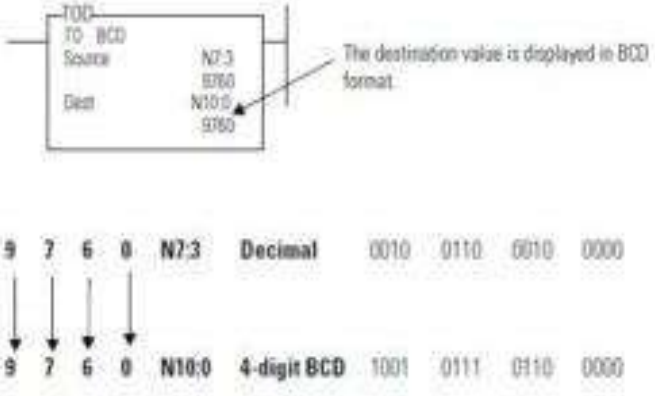
This session contains general information about the data handling instructions for plc and explains how they function in your application program.

Convert to BCD (TOD):

Use this instruction to convert 16-bit integers into BCD values. If the integer value you enter is negative, the absolute value of the number is used for conversion.

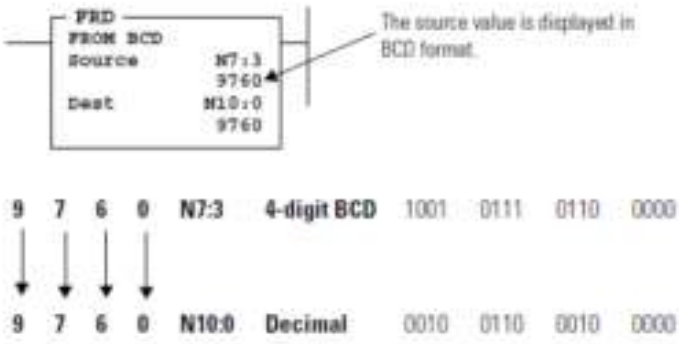


Example
The integer value 9760 stored at N7:3 is converted to BCD and the BCD equivalent is stored in N10:0. The maximum BCD value possible is 9999.



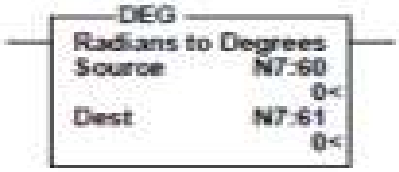
Convert from BCD (FRD)
Use this instruction to convert BCD values to integer values.

Example
The BCD value 9760 at source N7:3 is converted and stored in N10:0. The maximum source value is 9999, BCD.



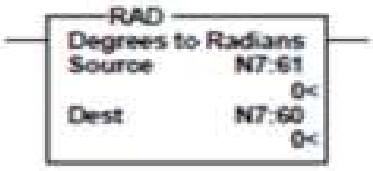
Radian to Degrees (DEG)
Use this instruction to convert radians (source) to degrees and store the result in the destination. The following formula applies:

Source * 180/P
Where π = 3.141592

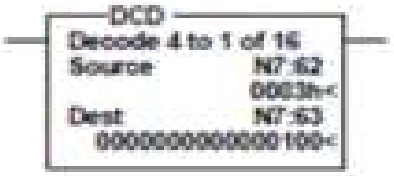


Degrees to Radians (RAD)
Use this instruction to convert degrees (source) to radians and store the result in the destination. The following formula applies:

Source * P/ 180
Where π= 3.141592

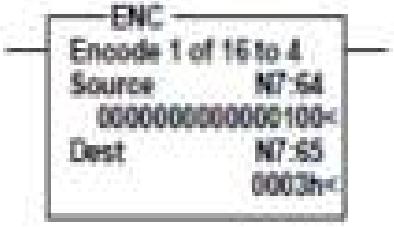


Decode 4 to 1 of 16 (DCD)
- When executed, this instruction sets one bit of the destination word.
- The particular bit that is turned on depends on the value of the first four bits of the source word. See the table below.



Encode 1 of 16 to 4 (ENC)

When the rung is true, this output instruction searches the source from the lowest to the highest bit, and looks for the first set bit. The corresponding bit position is written to the destination as an integer as shown in the table below.



6.14 I/0 INTERFACE PLC CONTROLLER INPUTS:

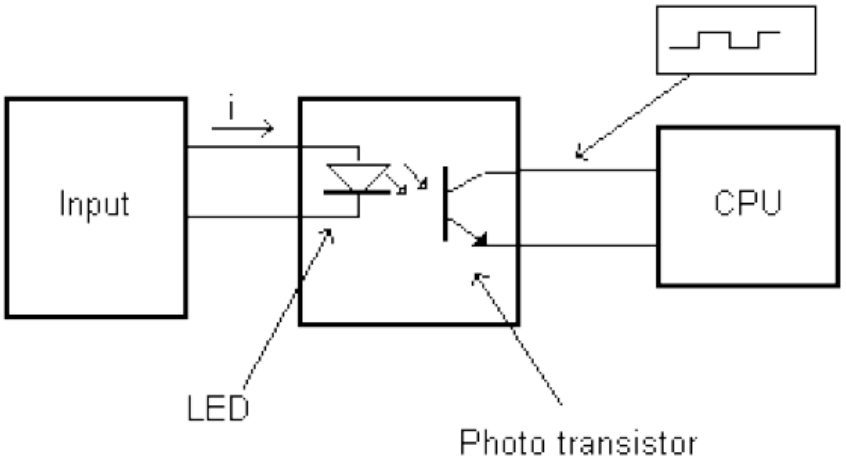
- An automated system depends to a large extent on the ability of a **PLC** controller to read signals from different types of sensors and input devices.

- To detect a work piece, see a mechanism in motion, check the pressure or the level of liquid, you need specific automatic devices such as proximity sensors, marginal switches, photoelectric sensors, level sensors, etc. Therefore, the input signals can be logical (on / off) or analog.
- One of the most frequent analog signals is a current signal of **4 to 20 mA** and a millivolt voltage signal generated by several sensors. The sensors are generally used as inputs for PLCs. You can get sensors for different purposes. They can feel the presence of some parts, measure temperature, pressure, or some other physical dimension, etc. (ex. inductive sensors can register metal objects).

Input adjustment interface:

- The input adjustment module changes the actual logic level to a level that suits the CPU unit (for example, the input of a sensor that operates with 24 VDC must be converted to a 5 VDC signal so that the CPU can process it). This is usually done through optic

Input adjustable interface



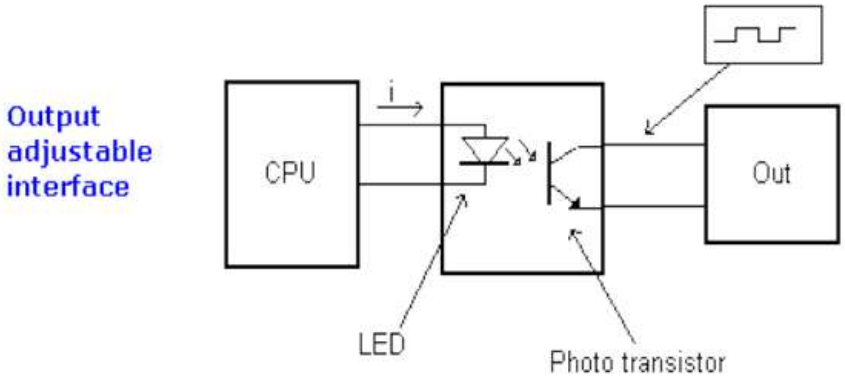
- Optical isolation, and this function can be seen in the following image
- Optical isolation means that there is no electrical connection between the external world and the CPU unit. They are separated "optically" or, in other words, the signal is transmitted through light.
- The way this works is simple. The external device brings a signal that turns on the LED, whose light in turn incites the photo transistor which in turn starts to drive, and the CPU sees it as a logical zero (the supply between the collector and the transmitter drops below 1 V).
- When the input signal stops, the LED goes off, the transistor stops conducting, the collector voltage increases and the CPU receives logic 1 as information.

PLC controller output:

- The most commonly used devices are motors, solenoids, relays, indicators, sound signaling and the like. When starting a motor or a relay, the PLC can manage or control a simple system such as the system to classify products into complex systems such as the service system to position the head of the CNC machine
- The output can be analog or digital type. The digital output signal works like a switch; the line is connected and disconnected. The analog output is used to generate the analog signal (for example, a motor whose speed is controlled by a voltage corresponding to a desired speed).

Output adjustment interface:

- The output interface is similar to the input interface. The CPU brings a signal to the LED and turns it on.



- The light incites a photo transistor that begins to conduct electricity, and therefore the voltage between the collector and emitter emits at 0.7 V, and a device connected to this output sees it as a logical zero.
- Conversely it means that there is a signal in the output and it is interpreted as logic. The photo transistor is not directly connected to an output of the PLC controller. Between a photo transistor and an output, there is usually a stronger relay or transistor capable of interrupting stronger signals.

6.15 SEQUENTIAL MOTOR CONTROL:

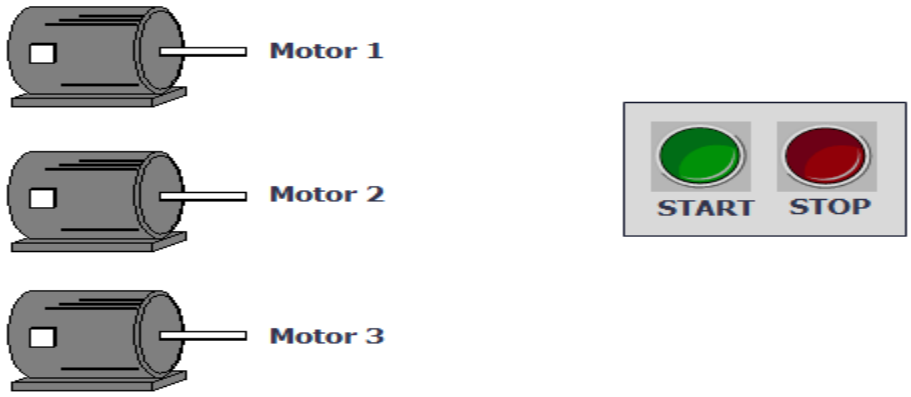
This is PLC Program for Sequential Motor Operating System.

Problem Description:

- In many industries, there are lots of motors are used. Sometimes we need to start more than one motor in an application. When we have low incoming power supply rating, then there is a chance the incoming MCB will trip when one or more motors will START in parallel because they will consume more power.
- Here we will consider one similar example where we START each motor one by one.

Problem Diagram:

PLC Program for Sequential Motor Control



Problem Solution:

- The problem can be solved by using PLC programming or relay logic.
- In this case, we have to operate motors sequentially. There are total 3 motors to be controlled in a sequence. So that each motor will start sequentially, say Motor 1 will START then after some delay then motor 2 will start and after some delay motor 3 will start.
- So that whole operation will take 10 seconds to start all motors in a sequence. By providing this delay we can avoid the problem of taking large current by motors during initial stat up.
- All motors will be operate in the sequence and 5 seconds time delay is to be provided between operations of each motor.
- Here will write logic for sequential operation for motors using PLC.

List of Inputs/Output

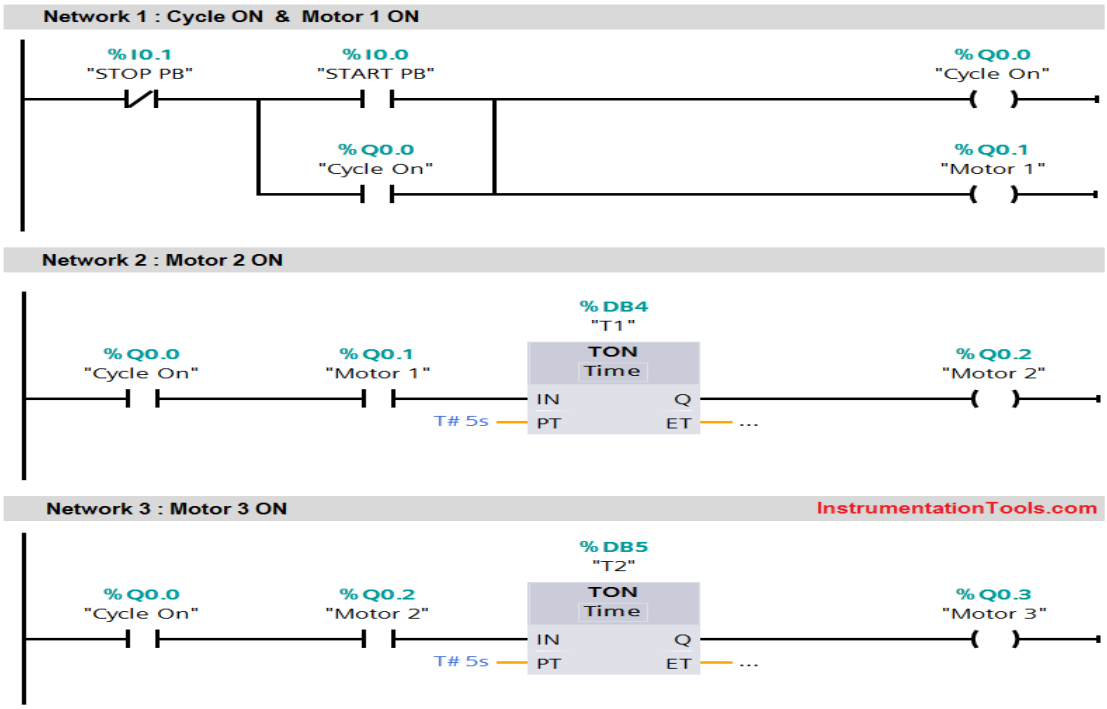
Inputs List

- Start PB : I0.0
- Stop PB: I0.1

Outputs List

- Cycle on : Q0.0
- Motor 1: Q0.1
- Motor 2 : Q0.2
- Motor 3 : Q0.3

PLC Ladder diagram for Sequential Motor Control



Program Description:

- In this application, we used Siemens S7-1200 PLC and TIA Portal Software for programming. We can also design this logic with relay circuit.

Network 1:

- In Network 1, we wrote logic for cycle ON condition. Here cycle ON (Q0.0) lamp will indicate cycle status.
- Cycle can be started by pressing START PB (I0.0) push button and can be stopped by pressing STOP PB (I0.1) push button.
- When cycle will be ON, at same time Motor 1(Q0.1) will be started. And at the same time, timer instruction will be executed.

Network 2:

- In Network 2, the NO contact of Motor 1 starts Timer T1 and when Timer for Motor 2 (Q0.1) will reach the set value 5 seconds.
- Then NO contact of the T1 will START the Motor 2 (Q0.1).

Network 3:

- In Network 3. Here logic for motor 3 is taken. There is NO contact of motor 2 is given for starting the timer of motor 3.

- When T2 will reach the set value 5s, the NO contact of the T2 will START the Motor 3(Q0.0).
- When STOP PB (I0.1) will be pressed then NC contact will be activated which makes Cycle (Q0.0) OFF. And also motor 2 and 3 will stop working.

Runtime Test Cases

PLC Program for Sequential Motor Control

Inputs	Outputs	Physical Elements
I0.0=1	Q0.0 & Q0.1=1	Cycle and Motor 1 ON
T1=1	Q0.2=1	Motor 2 ON
T2=1	Q0.3=1	Motor 3 ON
I0.1=1	Q0.0,Q0.1 & Q0.2=0	All cycle stop