SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF MECHANICAL ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

COURSE MATERIAL

Program: B.E - MTRON Semester: VI

Course: PLC and Automation

Course code: SMR1303

UNIT 1 - PROGRAMMABLE LOGIC CONTROLLERS - SMR1303

Introduction - Parts of PLC - Principles of operation - PLC sizes - PLC hardware components - I/O section - Analog I/O modules - digital I/O modules CPU processor memory module – PLC programming Simple instructions - Output control devices - Latching relays PLC ladder diagram, Converting simple relay ladder diagram in to PLC relay ladder diagram.

UNIT 1 - PROGRAMMABLE LOGIC CONTROLLERS - SMR1303

INTRODUCTION

PLC stands for Programmable Logic Controllers.

A PLC is basically a microprocessor based device that is used for controlling any machine (electrical, mechanical, and electronic). It is also used in assembly lines controlling in the industries. It is similar to a computer. It is typically based on RISC architecture. It is programmed in specific languages based on the real time purpose. It is connected to sensors, actuators, relays, contactors, etc. it is characterized by the number and type of I/O ports they provide and by their I/O scan rate

The National Electrical Manufacturers Association (NEMA) defines a PLC as a "digitally operating electronic apparatus which uses a programmable memory for the internal storage of instructions by implementing specific functions, such as logic, sequencing, timing, counting, and arithmetic to control through digital or analog I/O modules various types of machines or processes.

A PLC is able to receive (input) and transmit (output) various types of electrical and electronic signals and can control and monitor practically any kind of mechanical and/or electrical system. Therefore, it has enormous flexibility in interfacing with computers, machines, and many other peripheral systems or devices.

Control is the process in a system in which one or several input variables influence other variables.

Need for PLC

Hardwired panels were very time consuming to time, debug and change

The following requirements for computer controllers to replace hard wired panels:

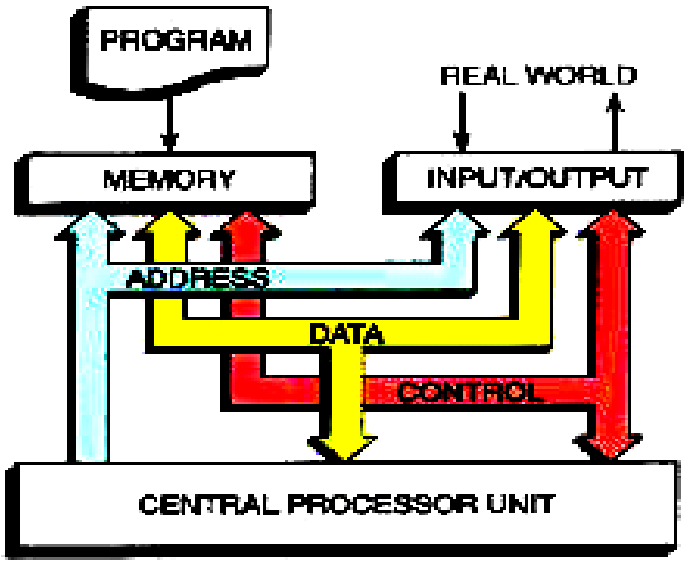
- Solid state not mechanical
- Easy to modify input and output devices
- Easily programmed and maintained by plant electricians
- Be able to function in an industrial environmentComparison

Comparison

Hard wired control systems	PLC systems
The functions are determined by physical wiring	The functions are determined by a program stored in the memory
Changing the function means changing the wiring	The control functions can be changed by simply changing the program
Can be contact making type (relays, contactors) or electronic type (logic circuits)	Consists of a control device, to which all the sensors and actuators are connected

PLC architecture

The basic architecture of the PLC can be described as below:



The basic components of a PLC are:

1. A Central Processing Unit (CPU)
2. Input module
3. Output module
4. Programming device
5. Power supply

Central Processing Unit:

It is the heart of the PLC system. The CPU is a microprocessor based control system that replaces central relays, counters, timers and sequencers. One bit processors are adequate for dealing with logic operations. The operations of a CPU are as follows:

- The CPU accepts data from various sensing devices, executes the user program from memory and sends appropriate output commands to control devices.
- A DC power source is required to produce a low -level voltage used by processor and I/O modules. This power supply can be housed in the CPU or may be a separately mounted unit, depending on the PLC system manufacturer.

Input and Output Modules:

Inputs:

Inputs are basically the physical entities that control the on and off of the machine. These devices are controlled by the either the machine operator manually or automatically sent after starting the machine. Some off the devices can be listed down as:

- Push buttons
- Limiter switches
- Pressure switches
- Flow switches
- Proxy sensor
- Emergency switch

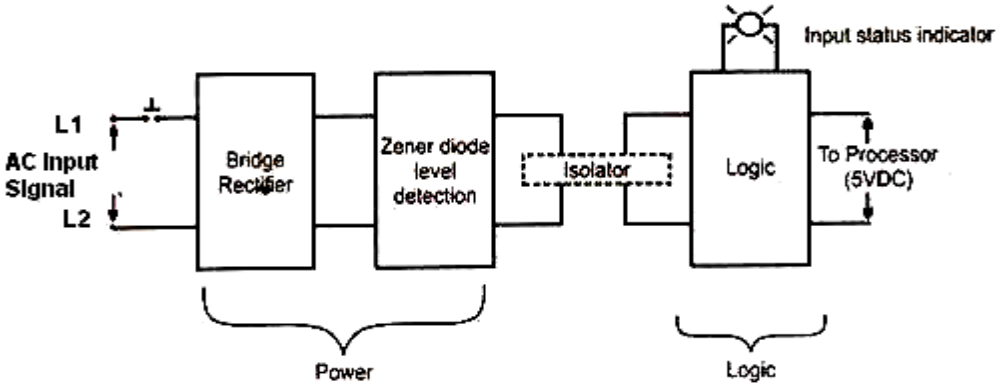
Outputs:

Outputs are the devices which receive the signals given by the PLC and perform the actions accordingly. The output devices can be listed down as:

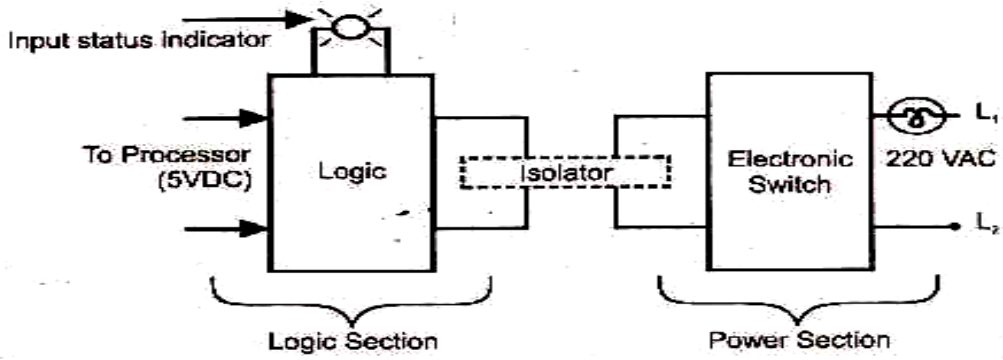
- Relays
- Contactors
- Solenoid valves
- Lamp indicators. Input-output module:

The I/O modules connect the input devices with the controller. They convert the electrical signals used in the devices into electronic signals that can be used by the control system and translate the real world values to IO table values.

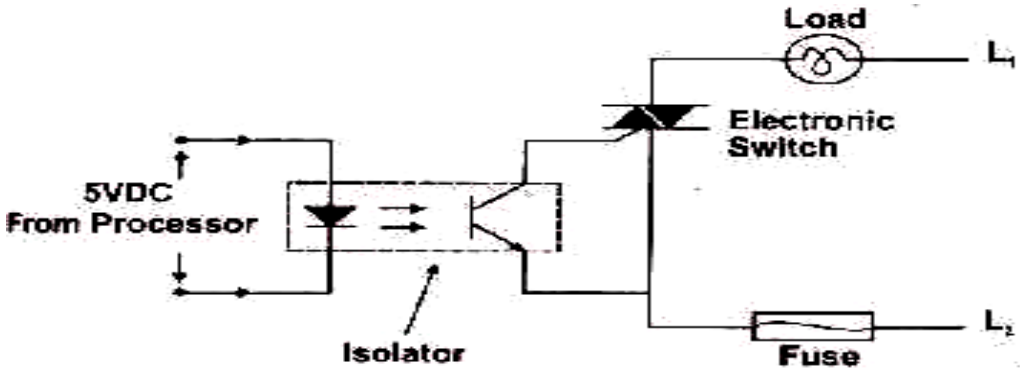
Block diagram of input module:



Block diagram of output module:



Circuit diagram of isolator circuit:



The number of I/O devices used within a control system is called its "point count".

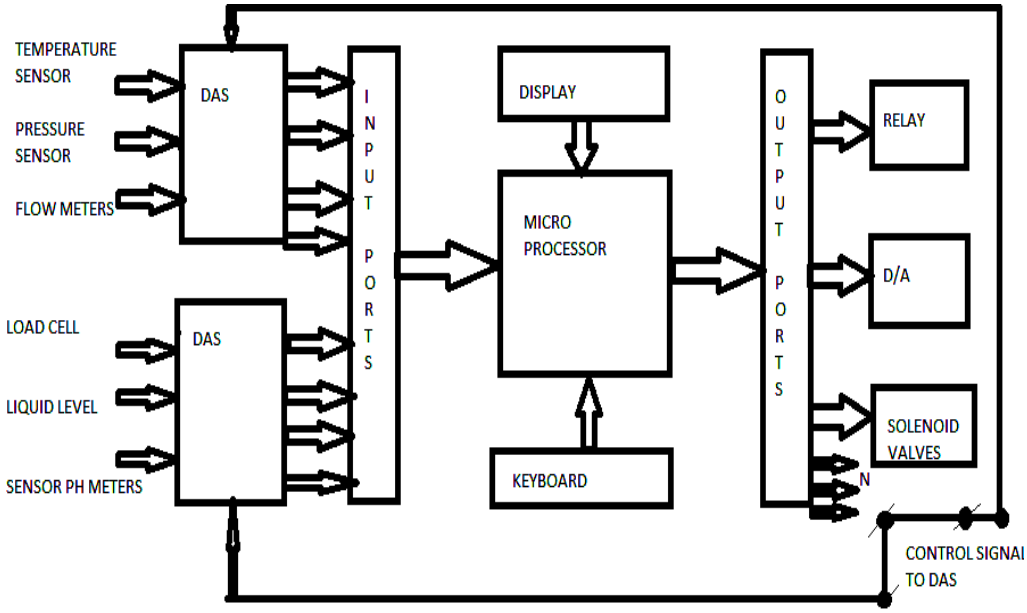
Programming device (keyboard and monitor):

Keyboard and monitor is used for programming the PLC. The data is entered in the processor with the help of the keyboard in the form of ladder diagram. This ladder diagram can be seen on the monitor screen. The programmer can communicate with the processor with the help of the programming devices. The programming unit communicates with the processor of the PLC via a serial or parallel data communication link.

PLC power supply:

The PLC is power by the AC mains supply. However some of the PLC components utilize power of about only 5V DC. PLC power supply converts the AC power supply into DC and supports these components.

The overall block diagram of the PLC:



The input signals are given to the **Data Acquisition System (DAS)** and then further sent to the input module. Further the signals are sent to the processor. The processor is connected with the programming devices. The signal from the processor is then sent to the output module and then from the output module further to the output devices.

PLC operating sequence

- Self-test: testing of its own hardware and software for faults.
- Input scan: if there are no problems, PLC will copy all the inputs and copy their values into memory.
- Logic solve/scan: using inputs, the ladder logic program is solved once and outputs are updated.
- Output scan: while solving logic, the output values are updated only in memory when ladder scan is done, the outputs will be updated using temporary values in memory.

Number systems and codes

The integral operation of programmable controllers is largely based on numbers. Numbers emulated by electronic circuitry are used to encode and store information that in turn se allow s these devices to process instructions and data required to perform each and every operation.

PLC also relies on number systems to perform even the most basic operations and store various information. The number systems commonly used in PLC are given below:

2	Binary	[0,1]
8	Octal	[0,1,2,3.4.5,6,7]
10	Decimal	[0,1,2,3,4,5,6,7,8,9]
16	Hexadecimal	[0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F]

In PLC, decimal number system is usually used for timer and counter presets, math operations and general numerical quantities.

Binary number system, bytes and words are the common units for storing digital information in memory. They are used to represent letters, numbers, punctuation marks, machine instructions and virtually any information type.

In PLC in many cases, base 8 is used for addressing memory and input/output terminal locations. Using octal for numbering is convenient in these cases since memory is comprised of groupings of 8 binary digits (bytes) and I /O modules are normally in one or more groups of eight points per module.

Hex numbers easily express coded digital data which is otherwise expressed in binary. Such codes are typically used when devices communicate. While the binary data may be easily interpreted by the receiving device, hex coded characters are really used for convenience of human operators.

Automation:

Automation is basically the delegation of human control functions to technical equipment aimed towards achieving higher productivity, superior quality of end product, efficient usage of energy and raw materials, improved safety in working conditions etc.

Advantages of PLC in Automation:

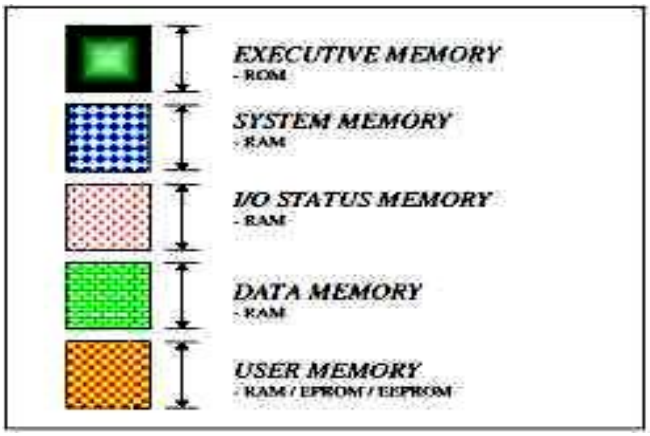
- To reduce human efforts
- To get maximum efficiency from machine and control them with human logic
- To reduce complex circuitry of entire system
- To eliminate the high costs associated with inflexible, relay controlled systems.

Areas of application of PLC in automation:

- Manufacturing/machining
- Food/beverage
- Textile industry
- Travel industry
- Aerospace
- Printing industry

Memory System:

All PLCs contain both RAM and ROM in varying amounts depending upon the design of the PLC. The use of PLC's memory is determined again by the design of the unit. However, all PLC memories can be subdivided into at least five major areas. A typical memory utilization map for a PLC is depicted in the following figure.



a. Executive Memory:

The operating system or executive memory for the PLC is always in ROM since, once programmed and developed by the manufacturer, it rarely needs changing. It is the one that actually does the scanning in a PLC. The operating system is a special machine language program that runs the PLC. It instructs the microprocessor to read each user instruction, helps the microprocessor to interpret user programmed symbols and instructions , keeps the track of all the I/O status, and is responsible for maintaining / monitoring the current status of the health of the system and all its components.

b. System memory

In order for the operating system to function, a section of the memory is allotted for system administration. As the executive program performs its duties, it often requires a place to store intermediate results and information. A section of RAM is installed for this purpose. Normally this area is allotted for use of the operating system only and is not available to the user for programming. It might be thought of as a scratch pad for the operating system to doodle on as necessary. Some PLCs use this area for storing the information which passes between programmer and operating system, e.g. the operating system generates certain error codes store in the specific address in this area during the execution of user program which can be read by user program; or the user may also give additional information to the operating system before execution of user program by writing some codes in the specific address in this area, etc.

c. I/O Status Memory - I/O Image Table

Another portion of RAM is allocated for the storage of current I/O status. Every single input/output module has been assigned to it a particular location within the input/output image table. The location within the input and output image tables are identified by addresses, each location has its own unique address.

During the execution of user program, the microprocessor scans the user program and interpret the user commands, the status of input modules used are read from the input image table (not directly from the input module itself). Various output device status generated during the execution of user program are stored in the output image table (not directly to output modules). (*Find out about input scan and output scan.*)

d. Data Memory

Whenever timers, counters, mathematics and process parameters are required, an area of memory must be set aside for data storage. The data storage portion of memory is allocated for the storage of such items as timers or counter preset/accumulated values, mathematics instruction data and results, and other miscellaneous data and information which will be used by any data manipulation functions in the user program.

Some manufacturers subdivide the data memory area into two sub-memories, one for fixed data and other for variable data. The fixed data portion can only be programmed via the programming device. The CPU is not permitted to place data values in this area. The variable portion of the data memory is available to the CPU for data storage.

e. User Program Memory

The final area of memory in a PLC is allocated to the storage of the user program. It is this memory area that the executive program instructs the microprocessor to examine or 'scan' to find the user instructions. The user program area may be subdivided if the CPU allocates a portion of this memory area for the storage of ASCII messages, subroutine programs, or other special programming functions or routines. In the majority PLCs, the internal data storage and user program areas are located in RAM.

Several systems do offer an option that places both the user program and the fixed data storage areas in EPROM type memory. The user can develop program in RAM and run the system to ensure correct operation. Once the user is satisfied that the programming is correct, a set of EPROMs is then duplicated from the RAM. Then the user can shut down the CPU and replaces the RAM with the newly programmed EPROM. Any future change would require that the EPROMs be reprogrammed.

Discrete I/O

A “discrete” data point is one with only two states on and off. Process switches, pushbutton switches, limit switches, and proximity switches are all examples of discrete sensing devices. In order for a PLC to be aware of a discrete sensor’s state, it must receive a signal from the sensor through a discrete input channel.

Inside the discrete input module is (typically) a light-emitting diode (LED) which will be energized when the corresponding sensing device turns on. Light from this LED shines on a photo-sensitive device such as a phototransistor inside the module, which in turn activates a bit (a single element of digital data) inside the PLC's memory. This opto-coupled arrangement makes each input channel of a PLC rather rugged, capable of isolating the sensitive computer circuitry of the PLC from transient voltage "spikes" and other electrical phenomena capable of causing damage.

Indicator lamps, solenoid valves, and motor contactors (starters) are all examples of discrete control devices. In a manner similar to discrete inputs, a PLC connects to any number of different discrete final control devices through a discrete output channel. Discrete output modules typically use the same form of opto-isolation to allow the PLC's computer circuitry to send electrical power to loads: the internal PLC circuitry driving an LED which then activates some form of photosensitive switching device. Alternatively, small electro mechanical relays may be used to interface the PLC's output bits to real-world electrical control devices.

Analog I/O

In the early days of programmable logic controllers, processor speed and memory were too limited to support anything but discrete (on/off) control functions. Consequently, the only I/O capability found on early PLCs were discrete in nature2. Modern PLC technology, though, is powerful enough to support the measurement, processing, and output of analog (continuously variable) signals.

All PLCs are digital devices at heart. Thus, in order to interface with an analog sensor or control device, some "translation" is necessary between the analog and digital worlds. Inside every analog input module is an ADC, or Analog-to-Digital Converter, circuit designed to convert an analog electrical signal into a multi-bit binary word. Conversely, every analog output module contains a DAC, or Digital-to-Analog Converter, circuit to convert the PLC's digital command words into analog electrical quantities.

Analog I/O is commonly available for modular PLCs for many different analog signal types, including:

1. Voltage (0 to 10 volt, 0 to 5 volt) 2. Current (0 to 20 mA, 4 to 20 mA)
3. Thermocouple (millivoltage) 4. RTD (millivoltage) 5. Strain gauge (millivoltage)

Programming of PLC:

The PLC can be programmed in different ways. There are various methods for entering and interconnecting various logic elements. Some of these can be explained as follows:

1. Entry of ladder logic diagram 2. Low level computer languages
3. High level computer languages 4. Functional blocks 5. Sequential function chart

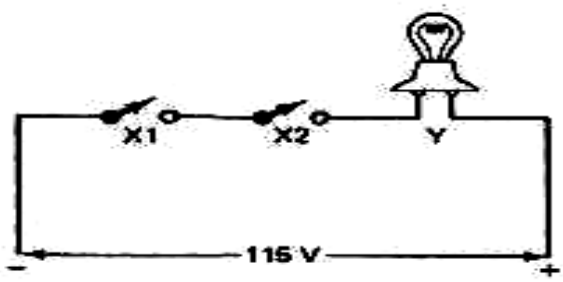
Programming Devices are:

Programming Console, Hand Programmer, PC

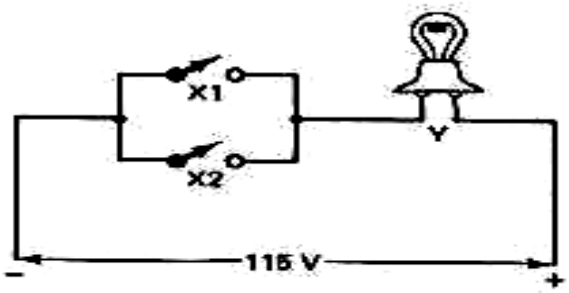
Ladder Logic Diagram

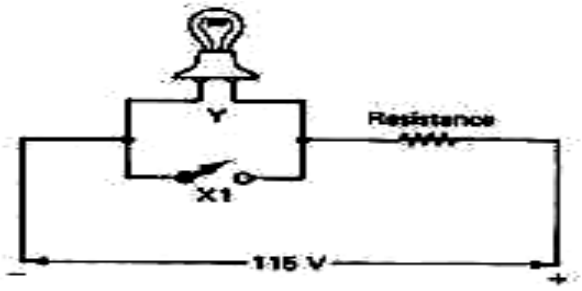
The ladder diagrams are based on three logic controls: 1. AND 2. OR 3. NOT

1. Logic AND operation



2. Logic OR operation





3. Logic NOT operation

(i)

Ladder symbol	Typical hardware components
(a) —||—	Normally open contacts (switch, relay, other ON/OFF devices)
(b) —|/|—	Normally closed contacts (switch, relay, etc.)
(c) —( )—	Output loads (motor, lamp solenoid, alarm, etc.)
(d) —[TMR 3s]—	Timer
(e) —[CTR]—	Counter

(ii)

AND —||— —||— (X1, X2)	X1 AND X2 (X1 · X2)
OR —||— (X1) / —||— (X2)	X1 OR X2 (X1 + X2)
NOT —|/|— (X1)	NOT X1

(i). Ladder logic symbols **(ii). Implementing the switching operations using ladder symbol**

Ladder Diagram:

- **First Step** : Translate all of the items we're using into symbols the PLC understands.
- **Second step** : We must tell the PLC where everything is located. In other words we have to give all the devices an address.
- **Final step** : We have to convert the schematic into a logical sequence of events.

First Step:

- The PLC doesn't understand terms like switch, relay, bell, etc.
- It prefers input, output, coil, contact, etc.
- It doesn't care what the actual input or output device actually is. It only cares that its an input or an output.
- First we replace the battery with a symbol. This symbol is common to all ladder diagrams. We draw what are called **bus bars**.
- These simply look like two vertical bars. One on each side of the diagram. Think of the left one as being + voltage and the right one as being ground. Further think of the current (logic) flow as being from left to right.
- Next we give the **inputs** a symbol. In this basic example we have one real world input. (i.e. the switch).
- We give the input that the switch will be connected to the symbol shown below. This symbol can also be used as the **contact of a relay**.

A contact symbol A coil symbol

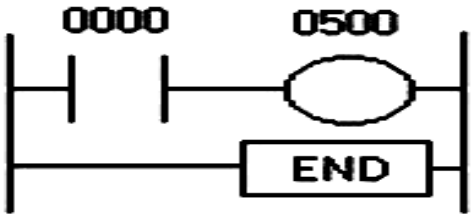
- Next we give the **outputs** a symbol. In this example we use one output (i.e. the bell).
- We give the output that the bell will be physically connected to the symbol shown below. This symbol is used as the coil of a relay.
- The AC supply is an external supply so we don't put it in our ladder. The PLC only cares about which output it turns on and not what's physically connected to it.

Second Step:

- We must tell the PLC where everything is located. In other words we have to give all the devices an address.
- Where is the switch going to be physically connected to the PLC? How about the bell? We start with a blank road map in the PLCs town and give each item an address.
- Could you find your friends if you didn't know their address? You know they live in the same town but which house? The PLC town has a lot of houses (inputs and outputs) but we have to figure out who lives where (what device is connected where).
- We'll get further into the addressing scheme later. The PLC manufacturers each do it a different way! For now let's say that our input will be called "0000". The output will be called "500".

Final Step:

- Convert the schematic into a logical sequence of events.
- The program we're going to write tells the PLC what to do when certain events take place.
- In our example we have to tell the PLC what to do when the operator turns on the switch.
- Final converted diagram.
- We eliminated the real world relay from needing a symbol as shown below.



A LoaD (contact) symbol

Basic Instructions

Load:

- The load (LD) instruction is a **normally open contact**. It is sometimes also called examine if on (**XIO**) (as in examine the input to see if its physically on). The symbol for a load instruction is shown above.
- This is used when an input signal is needed to be present for the symbol to turn on.
- When the physical input is on we can say that the instruction is True.
- We examine the input for an on signal. If the input is physically on then the symbol is on.
- An on condition is also referred to as a logic 1 state.

Load Bar:

- The Load bar instruction is a **normally closed contact**. It is sometimes also called LoaDNot or examine if closed (**XIC**) (as in examine the input to see if its physically closed) The symbol for a load bar instruction is shown below.

A LoaDNot (normally closed contact) symbol

Physical State	Instruction	Logic
OFF	TRUE	0
ON	FALSE	1

- This is used when an input signal does not need to be present for the symbol to turn on.
- When the **physical input is off** we can say that the **instruction is True**.
- We examine the input for an off signal. If the input is physically off then the symbol is on.
- With most PLCs this instruction (**Load** or **Load bar**) MUST be the first symbol on the left of the ladder.

Out:

- The Out instruction is sometimes also called an **Output Energize instruction**. The output instruction is like a **relay coil**. Its symbol looks as shown below.

An OUT (coil) symbol

An OUTBar (normally closed coil) symbol

- When there is a path of True instructions preceding this on the ladder rung, it will also be True.
- When the **instruction is True it is physically ON**.
- We can think of this instruction as a normally open output.

Out Bar:

- The Outbar instruction is sometimes also called an OutNot instruction.
- The Outbar instruction is like a **normally closed relay coil**. Its symbol looks like that shown below.

Among these, ladder logic diagram is most frequently used as whenever a personnel wants to change the PLC, he does not have to learn an entirely new programming language. Only the knowledge of the circuit diagram is enough.

This method includes the direct entry of the logic diagram into the PLC memory. This method requires the use of a keyboard and a display screen with graphics capability to display the symbols of the components and their inter relationships in the ladder logic diagram. Programming is accomplished by inserting appropriate components in the rungs of the ladder diagram.

- Ladder logic diagram uses graphic symbols similar to relay schematic circuit diagrams.
- It consists of two vertical lines representing the **power rails**
- Circuits are connected as horizontal lines between these two vertical lines. Such horizontal lines are called as **Rungs.**
- Each rung contains atleast one input and one output.
- A particular input or output can appear in more than one rung of ladder.
- Output is connected at right side and input is connected and input is connected at left side (for each rung)
- Output (which is on right side) cannot be directly connected with left side
- The ladder logic diagram consists of two types of components: Contacts & Coils

The various devices that are based on the binary logic can be stated as follows:

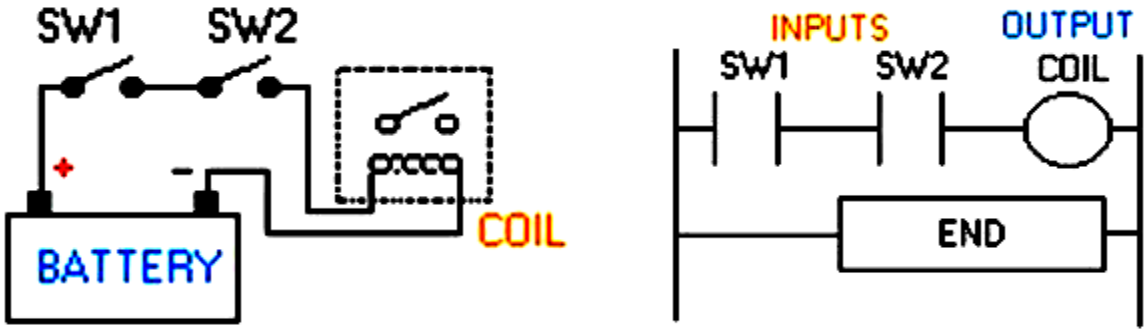
DEVICE	ONE / ZERO
INPUT	
Limit switch	Contact / no contact
Photo detector	Contact / no contact
Pushbutton switch	On /off
Timer	On / off
Control relay	Contact / no contact
Circuit breaker	Contact / no contact
OUTPUT	
Motor	On / off
Alarm buzzer	On / off
Control relay	Contact / no contact
Lights	On / off
Valves	Closed / open
Clutch	Engaged / not engaged
Solenoid	Energized / not energized

- Contacts are used to represent loads such as motors, relays, solenoids, timers, counters, etc.
- The program is entered rung by rung in the logic diagram.

- **Disadvantage:** Ladder logic diagrams are based on the ON-OFF operation. They are entirely based on the logic level 1 and logic level 0. This can be a disadvantage in using the ladder logic programming.

A Simple Example 1:

- In the above circuit, the coil will be energized when there is a closed loop between the + and - terminals of the battery.
- We can simulate this same circuit with a ladder diagram



- A ladder diagram consists of individual rungs just like on a real ladder.
- Each rung must contain one or more inputs and one or more outputs.
- The first instruction on a rung must always be an input instruction and the last instruction on a rung should always be an output (or its equivalent).
- Notice in this simple one rung ladder diagram we have recreated the external circuit above with a ladder diagram.
- Here we used the Load and Out instructions.
- Some manufacturers require that every ladder diagram include an END instruction on the last rung. Some PLCs also require an ENDH instruction on the rung after the END rung.

Example 2:

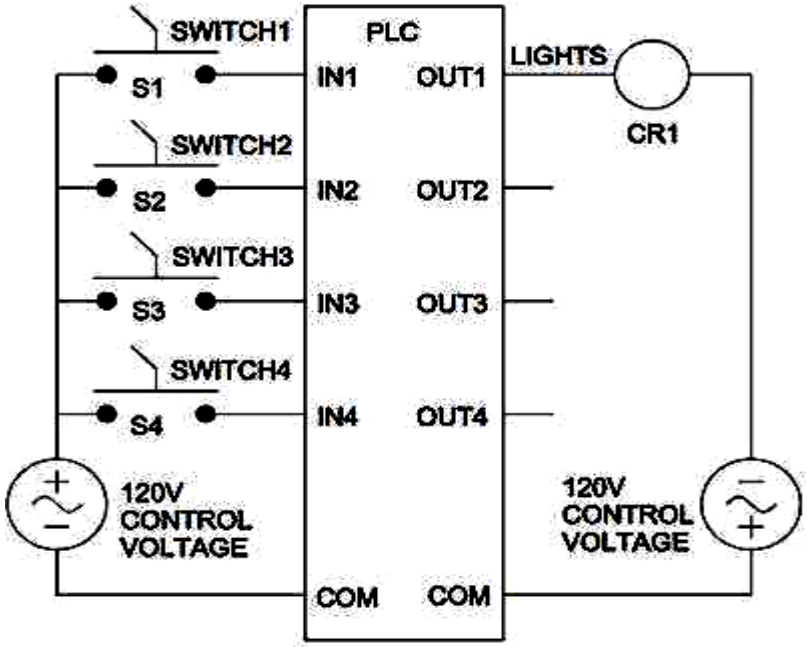
Lighting control system:

A lighting control system is to be developed. The system will be controlled by four switches, SWITCH1, SWITCH2, SWITCH3, and SWITCH4. These switches will control the lighting in a room based on the following criteria:

1. Any of three of the switches SWITCH1, SWITCH2, and SWITCH3, if turned ON can turn the lighting on, but all three switches must be OFF before the lighting will turn OFF.
2. The fourth switch SWITCH4 is a Master Control Switch. If this switch is in the ON position, the lights will be OFF and none of the other three switches have any control.

The first item we may accomplish is the drawing of the **controller wiring diagram**. All we need do is connect all switches to inputs and the lighting to an output and note the numbers of the inputs and output associated with these connections.

The remainder of the task becomes developing the **ladder diagram**. The wiring diagram is shown in Figure.



Notice that all four switches are shown as **normally open selector switches** and the output is connected to a relay coil **CR1**. We are using the relay CR1 to operate the lights because generally the current required to operate a bank of room lights is higher than the maximum current a PLC output can carry.

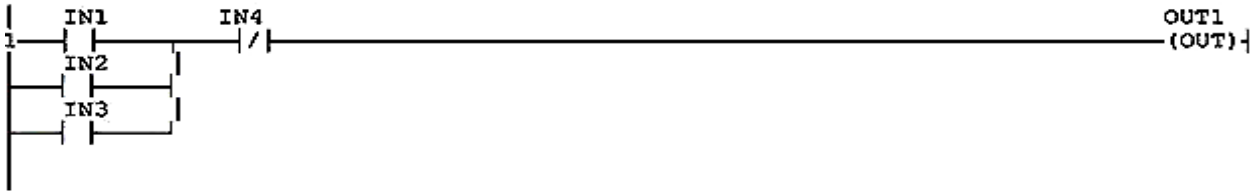
For this wiring configuration, the following definition list is apparent:

- **INPUT IN1** = SWITCH1
- **INPUT IN2** = SWITCH2
- **INPUT IN3** = SWITCH3
- **INPUT IN4** = SWITCH4 (Master Control Switch)
- **OUTPUT OUT1** = Lights control relay coil CR1

This program requires that when SWITCH4 is ON, the lights must be OFF. In order to do this, it would appear that we need a N/C SWITCH4, not a N/O as we have in our wiring diagram. However, keep in mind that once an input signal is brought into a PLC, we may use as many contacts of the input as we need in our program, **and the contacts may be either N/O or N/C**.

Therefore, we may use a N/O switch for SWITCH4 and then in the program, we will logically invert it by using N/C IN4 contacts.

The ladder diagram to implement this example problem is shown in Figure.



The ladder was printed using graphics characters (extended ASCII characters).

Notice the normally closed contact for IN4. A normally closed contact represents an inversion of the assigned element, in this case IN4, which is defined as SWITCH 4. Remember, SWITCH 4 has to be in the OFF position before any of the other switches can take control. In the OFF position, SWITCH 4 is open.

This means that IN4 will be OFF (de-energized). So, in order for an element assigned to IN4 to be closed with the switch in the OFF position, it must be shown as **a normally closed contact**. When SWITCH 4 is turned ON, the input, IN4, will become active (energized). If IN4 is ON, a normally closed IN4 contact will open.

With this contact open in the ladder diagram, none of the other switches will be able to control the output.

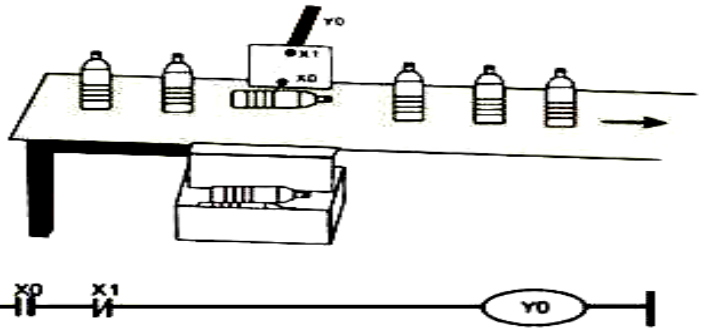
REMEMBER: A normally closed switch will open when energized and will close when de-energized.

Example 3:

Devices:

Device	Function
X0	X0=ON when the detected input signal from the bottle bottom is sheltered
X1	X1=ON when the detected input signal from the bottle neck is sheltered
Y0	Pneumatic pushing pole

Detecting the standing bottles on the conveyor and pushing the fallen bottles out



Program description

- If the bottle on the conveyor belt is upstanding, the input signal from the monitoring photocell at both bottle bottom and bottle neck will be detected. In this case, X0=ON and X1=ON. The normally open (NO) contact X0 will be activated as well as the normally closed (NC) contact X1. Y0 remains OFF and pneumatic pushing pole will not perform any action.

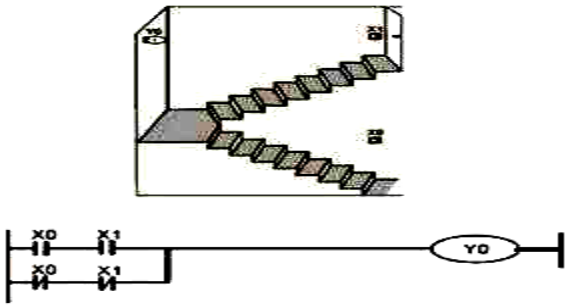
- If the bottle from the conveyor belt is down, only the input signal from monitoring photocell at the bottle bottom will be detected. In this case, X0=ON, X1=O FF. The state of the output Y 0 will be ON because the NO contact X activates and the NC contact X1 remains OFF. The pneumatic pushing pole will push the fallen bottle out of the conveyor belt

Example 4:

Devices:

Device	Function
X0	X0 turns ON when the bottom switch is turned to the right
X1	X1 turns ON when the top switch is turned to the right
Y 1	Stair light

Setting up a lightning system for users to switch on/off the lights whether they are bottom or the top of the stairs



Program Description

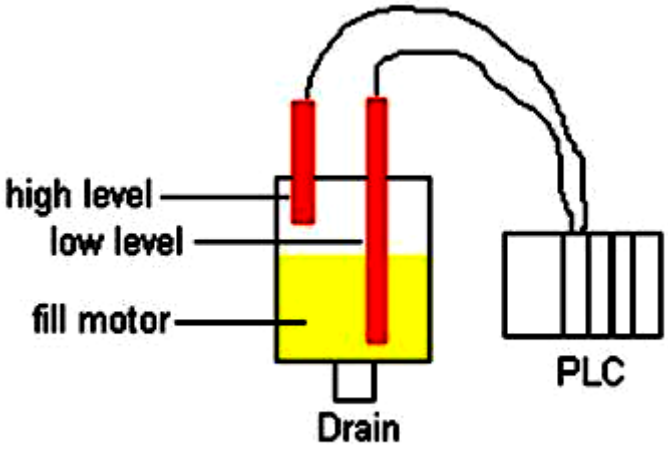
- If the states of the bottom switch and the top switch are the same, both ON or OFF, the light will be ON. If different, one is ON and the other is OFF, the light will be OFF.
- When the light is OFF, users can turn on the light by changing the state of either top switch or the bottom switch of the stairs. Likewise, when the light is ON, users can turn off the light by changing the state of one of the two switches.

Example 5:

A Level Application:

- We are controlling lubricating oil being dispensed from a tank.
- This is possible by using two sensors.
- We put one near the bottom and one near the top, as shown in the picture.

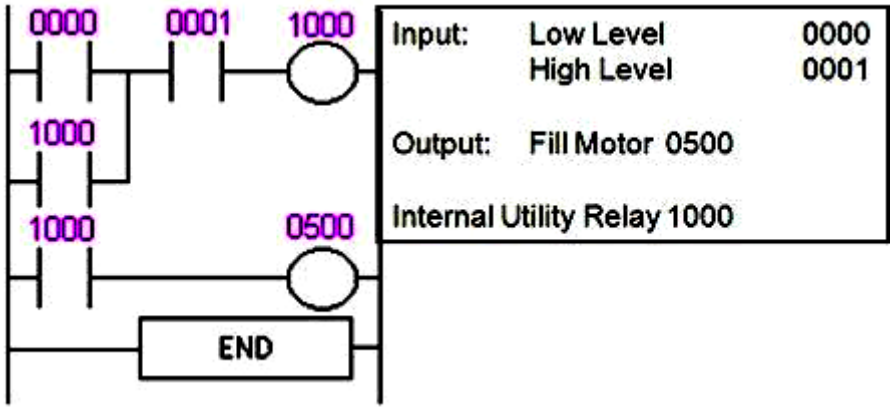
Here, we want the fill motor to pump lubricating oil into the tank until the high level sensor turns on. At that point we want to turn off the motor until the level falls below the low level sensor. Then we should turn on the fill motor and repeat the process.



Dispensing oil from a tank

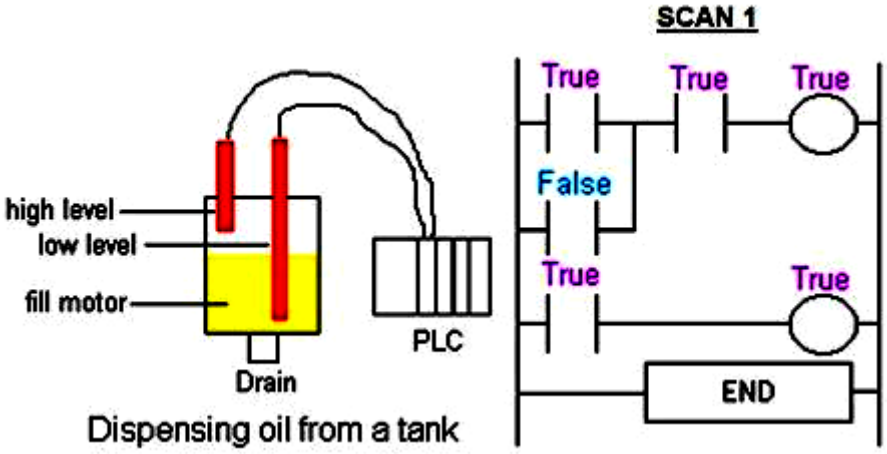
Here we have a need for 3 I/O (i.e. Inputs/Outputs):

- 2 are inputs (the sensors) and 1 is an output (the fill motor).
- Both of our inputs will be NC (normally closed) fiber-optic level sensors. When they are NOT immersed in liquid they will be ON. When they are immersed in liquid they will be OFF.



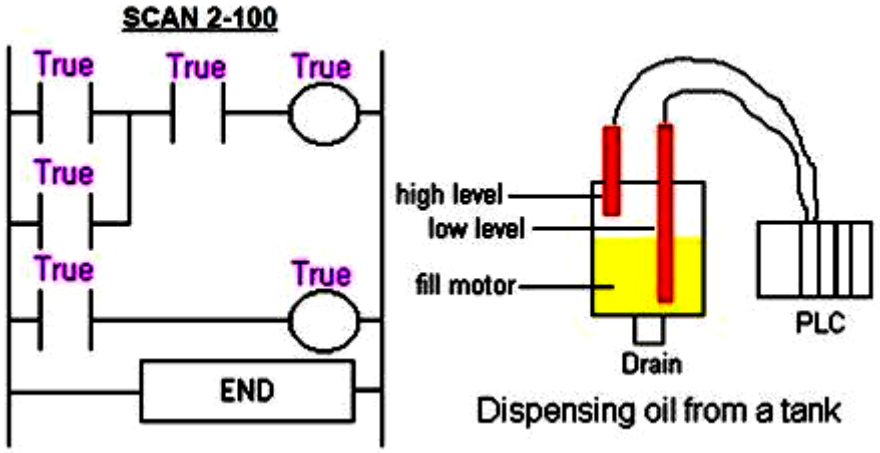
Program Scan:

- Initially the tank is empty. Therefore, input 0000 is TRUE and input 0001 is also TRUE.

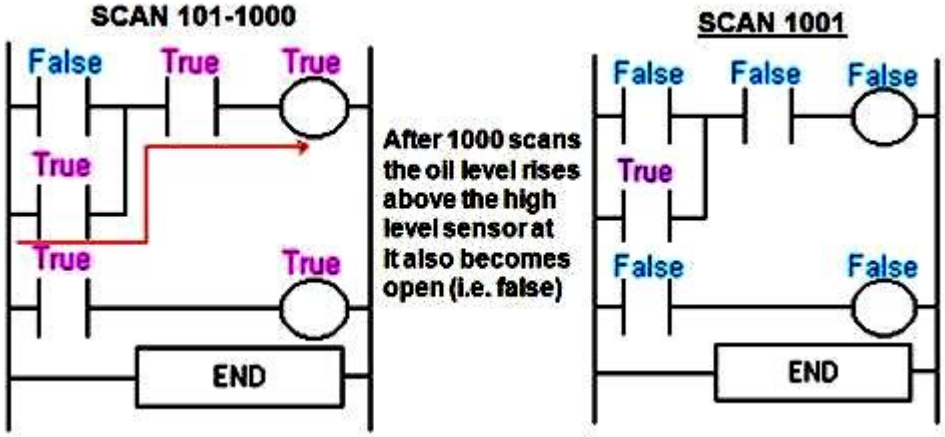


Dispensing oil from a tank

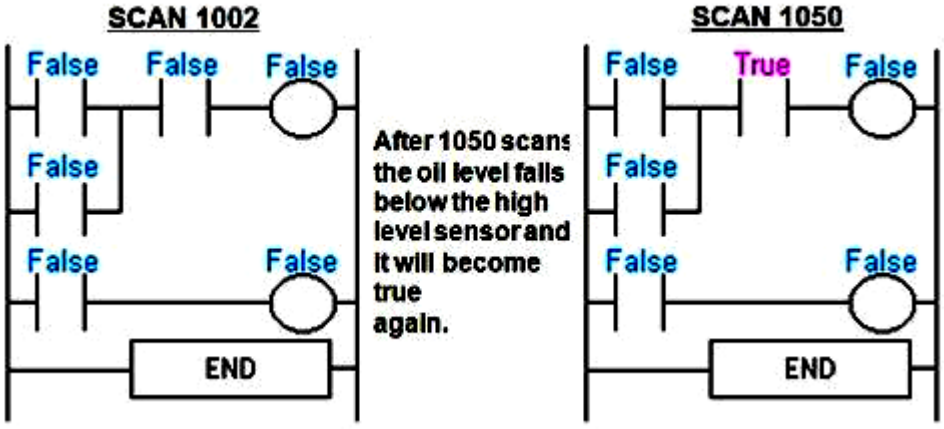
- Gradually the tank fills because 500(fill motor) is on.
- After 100 scans the oil level rises above the low level sensor and it becomes open. (i.e. FALSE).



- Even when the low level sensor is false there is still a path of true logic from left to right. This is why we used an internal relay. Relay 1000 is latching the output (500) on. It will stay this way until there is no true logic path from left to right.(i.e. when 0001 becomes false).

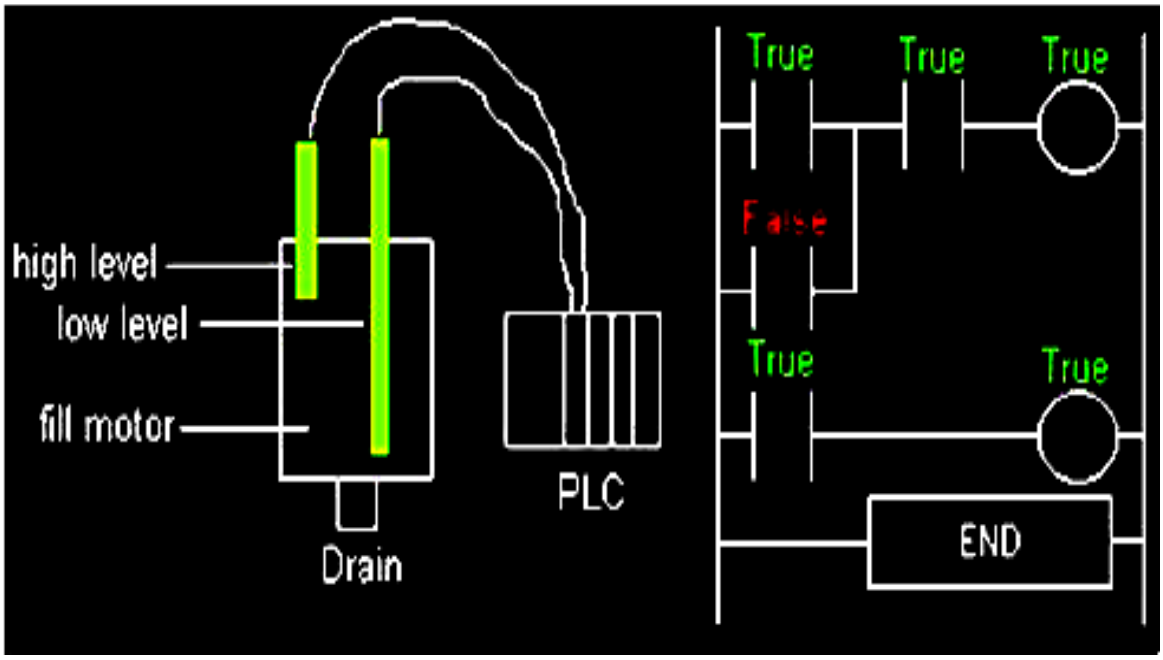


- Since there is no more true logic path, output 500 is no longer energized (true) and therefore the motor turns off.



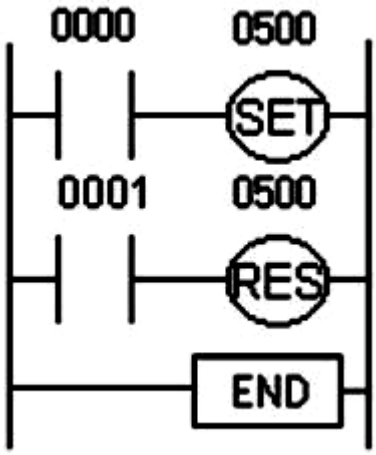
• Even though the high level sensor became true there still is NO continuous true logic path and therefore coil 1000 remains false!.

- After 2000 scans the oil level falls below the low level sensor and it will also become true again.
- At this point the logic will appear the same as SCAN 1 above and the logic will repeat as illustrated above.

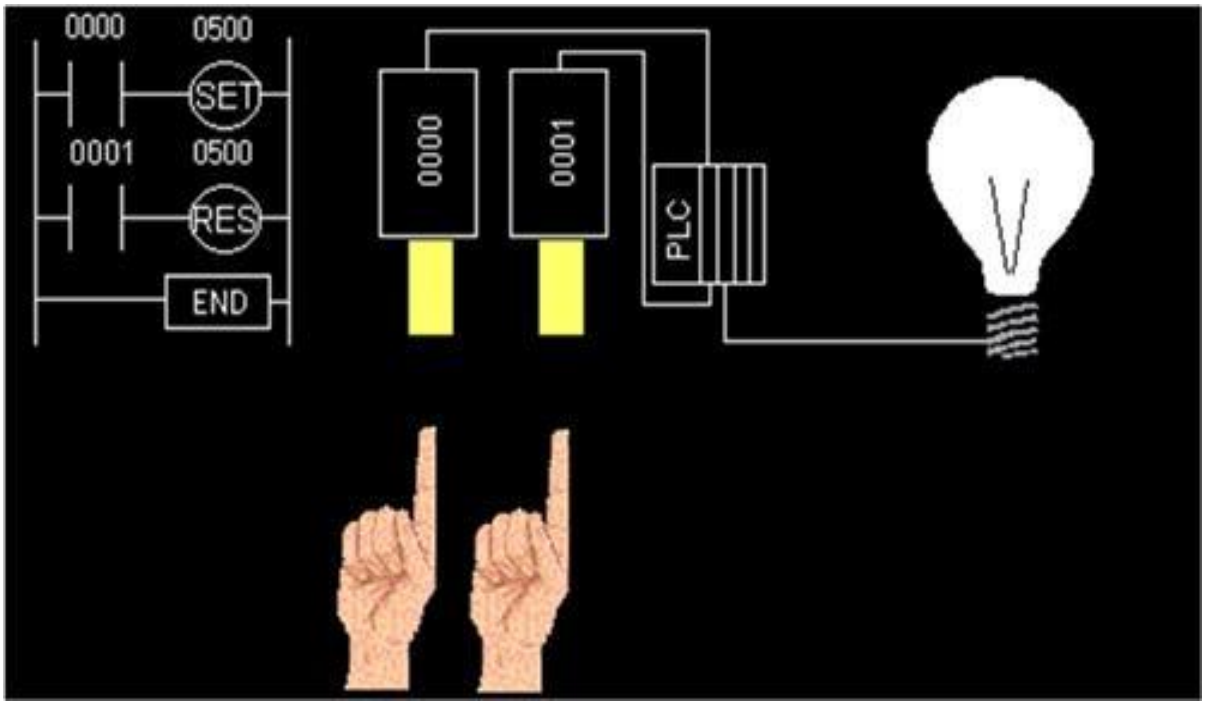


Latch Instruction:

- The latching instructions let us use momentary switches and program the plc so that when we push one the output turns on and when we push another the output turns off.
- Picture the remote control for your TV. It has a button for ON and another for OFF:
- When I push the ON button the TV turns on.
- When I push the OFF button the TV turns off.
- I don't have to keep pushing the ON button to keep the TV on. This would be the function of a latching instruction.
- The latch instruction is often called a SET or OTL (output latch).
- The unlatch instruction is often called a RES (reset), OUT (output unlatch) or RST (reset). The diagram below shows how to use them in a program.



Here we are using 2 momentary push button switches. One is physically connected to input 0000 while the other is physically connected to input 0001. When the operator pushes switch 0000 the instruction "set 0500" will become true and output 0500 physically turns on. Even after the operator stops pushing the switch, the output (0500) will remain on. It is latched on. The only way to turn off output 0500 is turn on input 0001. This will cause the instruction "res 0500" to become true thereby unlatching or resetting output 0500.



• What would happen if input 0000 and 0001 both turn on at the exact same time
• Will output 0500 be latched or unlatched?
• To answer this question we have to think about the scanning sequence. The ladder is always scanned from top to bottom, left to right.
• The first thing in the scan is to physically look at the inputs.
• 0000 and 0001 are both physically on.
• Next the PLC executes the program.
• Starting from the top left, input 0000 is true therefore it should set 0500.
• Next it goes to the next rung and since input 0001 is true it should reset 0500.
• The last thing it said was to reset 0500. Therefore on the last part of the scan when it updates the outputs it will keep 0500 off. (i.e. reset 0500).

PLC features and benefits

Solid-state Components	- Low component failure - Low space and power consumption - No mechanical wear
Microprocessor-based	- An intelligent decision-making device - Communication with other smart devices - Multifunctional capabilities
Programmable	- Simplifies logic changes - Allows flexible control system design - Allows better accuracy and repeatability
Modular Components	- Easily installed and replaced - Minimizes hardware purchases - Flexible installation configurations - Neat appearance inside control panel - Easily wired and maintained
Diagnostic Indicators	- Reduces troubleshooting/downtime - Helps isolate field wiring problems - Signal correct operation and faults
Variety of Standard I/O Interfaces Modules	- Controls variety of standard devices - Eliminates customized interfaces - Reduces engineering design time/costs - Allows standardized wiring diagrams
Quick I/O Disconnects	- Serviceable without disturbing field wiring
Local and Remote Input/Output Subsystems	- I/O interfaces can be conveniently placed - Elimination of long wire/conduit runs
Software Timers & Counters	- Eliminates hardware cost and problems - Achieves better timing accuracy - Easily changed presets
Software Control Relays	- Eliminates hardware cost and problems - Reduces space requirements - Unlimited number of relay contacts
All Process Data Stored In Memory	- Production management and maintenance information can be used to generate reports

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF MECHANICAL ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

COURSE MATERIAL

Program: B.E - MTRON Semester: VI
Course: PLC and Automation
Course code: SMR1303

UNIT 2 - PROGRAMMABLE LOGIC CONTROLLERS - SMR1303

Timer instructions ON Delay, OFF Delay and Retentive Timers-UP Counter, DOWN Counter and UP down Counters, program control instructions - Data manipulating instructions-math instructions.

Timers

In many control tasks there is a need to control time. For example, a motor or a pump might need to be controlled to operate for a particular interval of time, or perhaps be switched on after some time interval. PLCs thus have timers as built-in devices. Timers count fractions of seconds or seconds using the internal CPU clock.

PLC manufacturers differ on how timers should be programmed and hence how they can be considered. A common approach is to consider timers to behave like relays with coils which when energised result in the closure or opening of contacts after some preset time. The timer is thus treated as an output for a rung with control being exercised over pairs of contacts elsewhere (Figure 2.1 (a)). This is the predominant approach used in this book. Some treat a timer as a delay block which when inserted in a rung delays signals in that rung reaching the output (Figure 2.1 (b)).

There are a number of different forms of timers that can be found with PLCs. With small PLCs there is likely to be just one form, the on -delay timers. These are timers which come on after a particular time delay (Figure 2.2 (a)). Off-delay timers are on for a fixed period of time before turning off (Figure 2.2 (b)). Another type of timer that occurs is the pulse timer. This timer switches on or off for a fixed period of time (Figure 2.2 (c)). Figure 2.3 shows the IEC 1131-3 standard symbols for timers. TON is used to denote on-delay, TOF off-delay. TP pulse timers. On-delay is also represented by T− 0 and off-delay by 0− T. The time duration for which a timer has been set is termed the preset and is set in multiples of the time base used. Some time bases are typically 10 ms, 100 ms, 1 s, 10 s and 100 s. Thus a preset value of 5 with a time base of 100 ms is a time of 500 ms. For convenience, where timers are involved in this text, a time base of 1 s has been used.

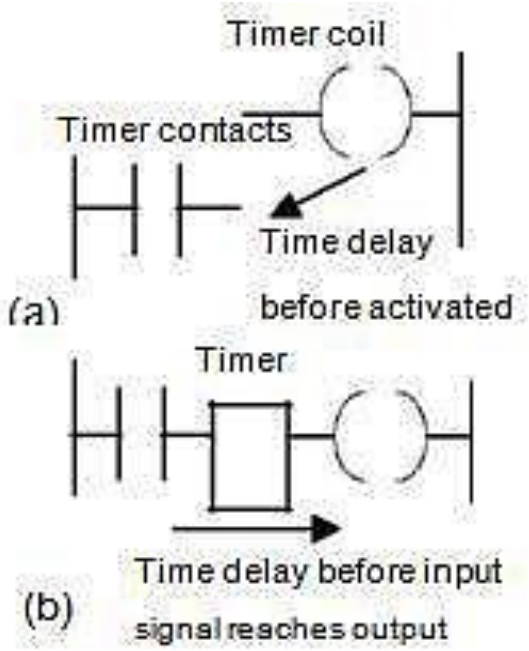


Figure 2.1 *Treatment of timers*

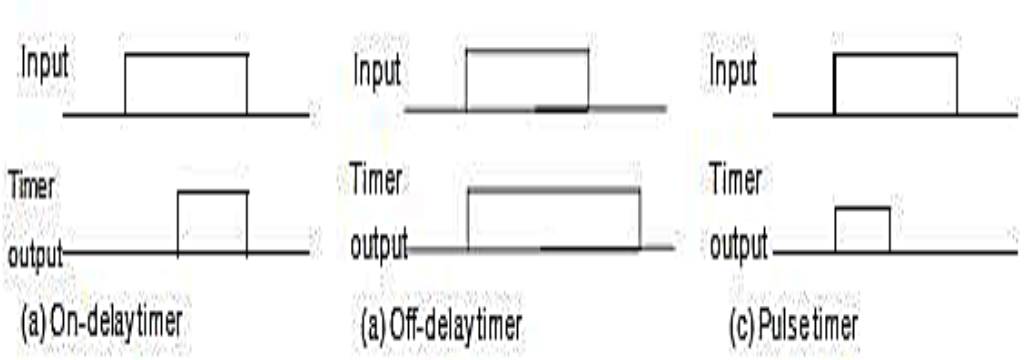


Fig 2.2 *Timers: (a) on-delay, (b) off-delay, (c) pulse*

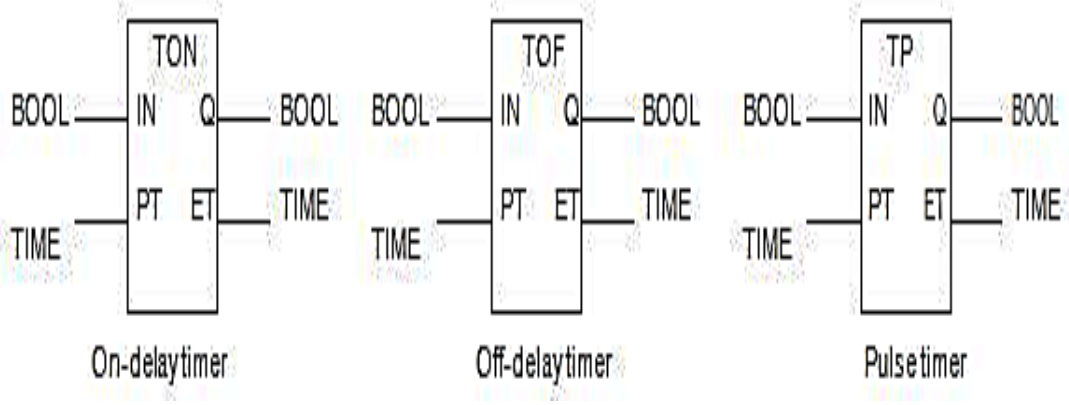


Figure 2.3 *IEC 1131-1 standards. BOOL indicates a Boolean input/output, i.e. on/off. IN is the input. Q is the output. ET is the elapsed time output. PT is the input used to specify the time.*

Programming timers

All PLCs generally have delay-on timers, small PLCs possibly having only this type of timer. Figure 2.4 (a) shows a ladder rung diagram involving a delay-on timer. Figure 2.4 (a) is typical of Mitsubishi. The timer is like a relay with a coil which is energized when the input In 1 occurs (rung 1). It then closes, after some preset time delay, its contacts on rung 2. Thus the output occurs some preset time after the input In 1 occurs. Figure 2.4 (b) shows the timer to be a delay item in a rung, rather than as a relay, the example being for Siemens. When the signal at the timer's start input changes from 0 to 1, the timer starts and runs for the programmed duration, giving its output then to the output coil. The time value (TV) output can be used to ascertain the amount of time remaining at any instant. A signal input of 1 at the reset input resets the timer whether it is running or not. Techniques for the entry of preset time values vary. Often it requires the entry of a constant K command followed by the time interval in multiples of the time base used.

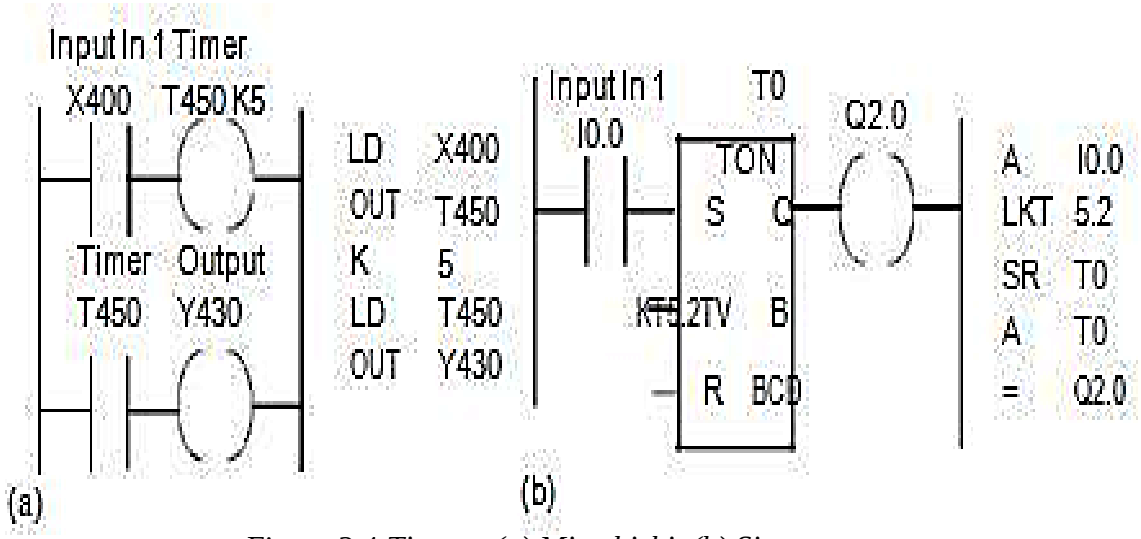


Figure 2.4 *Timers: (a) Mitsubishi, (b) Siemens*

Sequencing

As an illustration of the use of a timer, consider the ladder diagram shown in Figure 2.5 (a). When the input In 1 is on, the output Out 1 is switched on. The contacts associated with this output then start the timer. The contacts of the timer will close after the preset time delay, in this case 5.5 s. When this happens, output Out 2 is switched on. Thus, following the input In 1, Out 1 is switched on and followed 5.5 s later by Out 2. This illustrates how timed sequence of outputs can be achieved. Figure 2.5 (b) shows the same operation where the format used by the PLC manufacturer is for the timer to institute a signal delay.

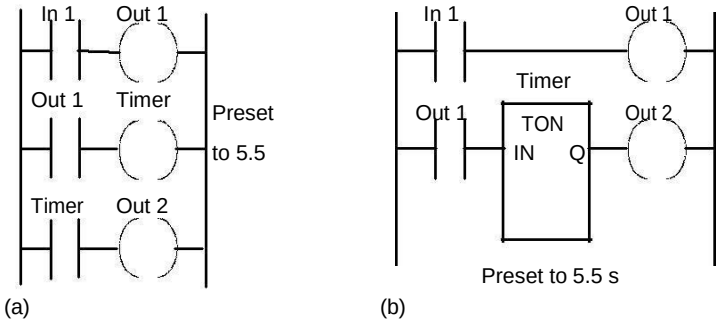


Figure 2.5 *Sequenced outputs*

Figure 2.6 shows two versions of how timers can be used to start three outputs, e.g. three motors, in sequence following a single start button being pressed. In (a) the timers are programmed as coils, whereas in (b) they are programmed as delays. When the start push button is pressed there is an output from internal relay IR1. This latches the start input. It also starts both the timers, T1 and T2, and motor 1. When the preset time for timer 1 has elapsed then its contacts close and motor 2 starts. When the preset time for timer 2 has elapsed then its contacts close and motor 3 starts. The three motors are all stopped by pressing the stop push button. Since this is seen as a complete program, the end instruction has been used.

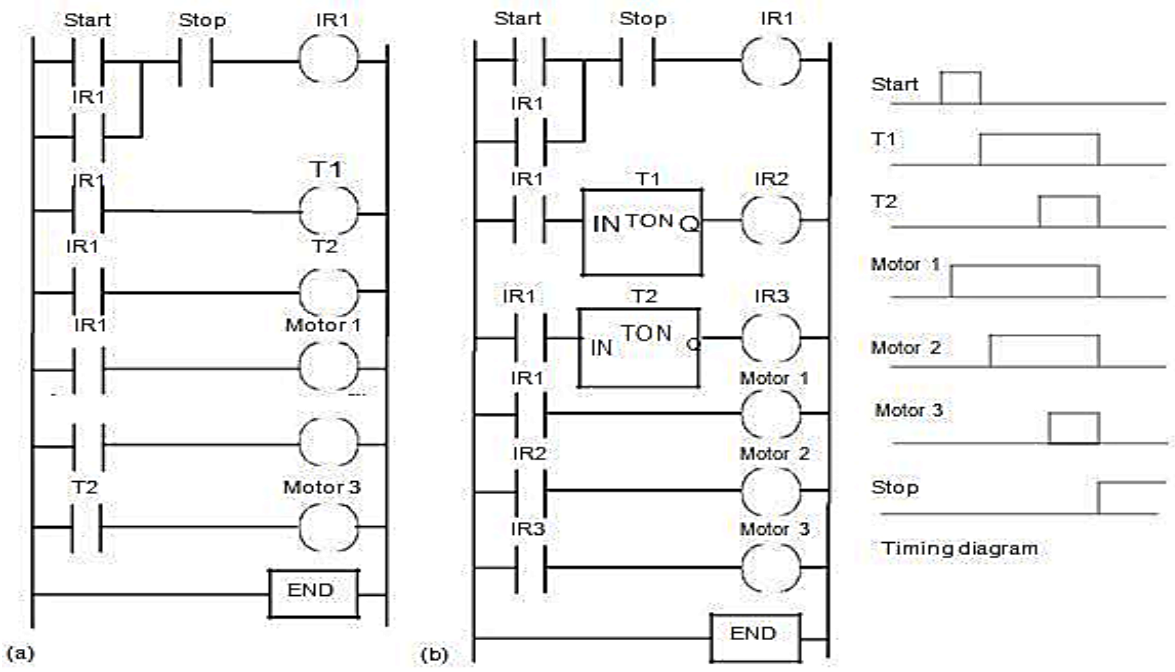


Figure 2.6 *Motor sequence*

Cascaded timers

Timers can be linked together, the term *cascaded* is used, to give longer delay times than are possible with just one timer. Figure 2.7 (a) shows the ladder diagram for such an arrangement. Thus we might have timer 1 with a delay time of 999 s. This timer is started when there is an input to In 1. When the 999 s time is up, the contacts for timer 1 close. This then starts timer 2. This has a delay of 100 s. When this time is up, the timer 2 contacts close and there is an output from Out 1. Thus the output occurs 1099 s after the input to In 1. Figure 2.7 (b) shows the Mitsubishi version of this ladder diagram and the program instructions for that ladder.

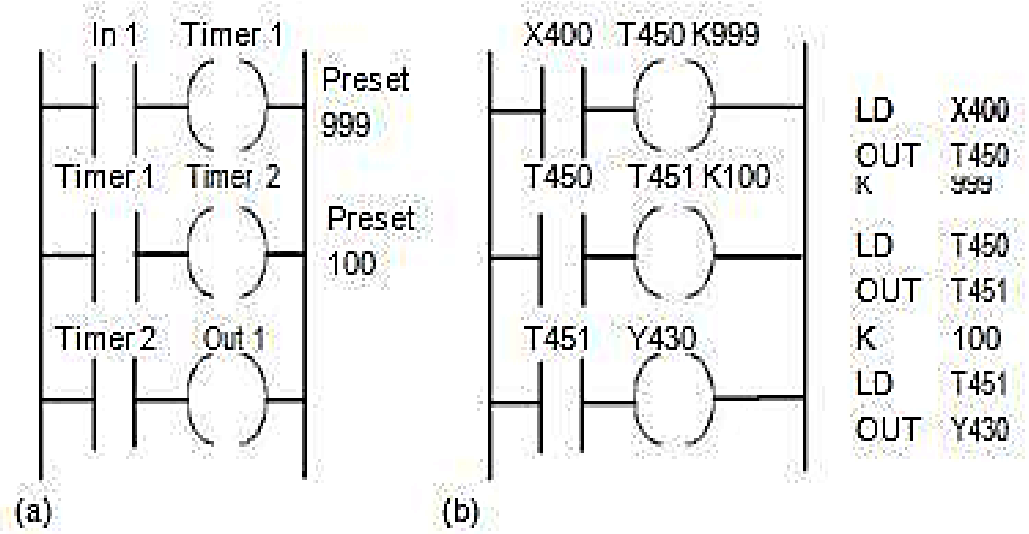


Figure 2.7 *Cascaded timers*

On-off cycle timer

Figure 2.8 shows how on-delay timers can be used to produce an *on -off cycle timer*. The timer is designed to switch on an output for 5 s, then off for 5 s, then on for 5 s, then off for 5 s, and so on. When there is an input to In 1 and its contacts close, timer 1 starts. Timer 1 is set for a delay of 5 s. After 5 s, it switches on timer 2 and the output Out 1. Timer 2 has a delay of 5 s. After 5 s, the contacts for timer 2, which are normally closed, open. This results in timer 1, in the first rung, being switched off. This then causes its contacts in the second rung to open and switch off timer 2. This results in the timer 2 contacts resuming their normally closed state and so the input to In 1 causes the cycle to start all over again.

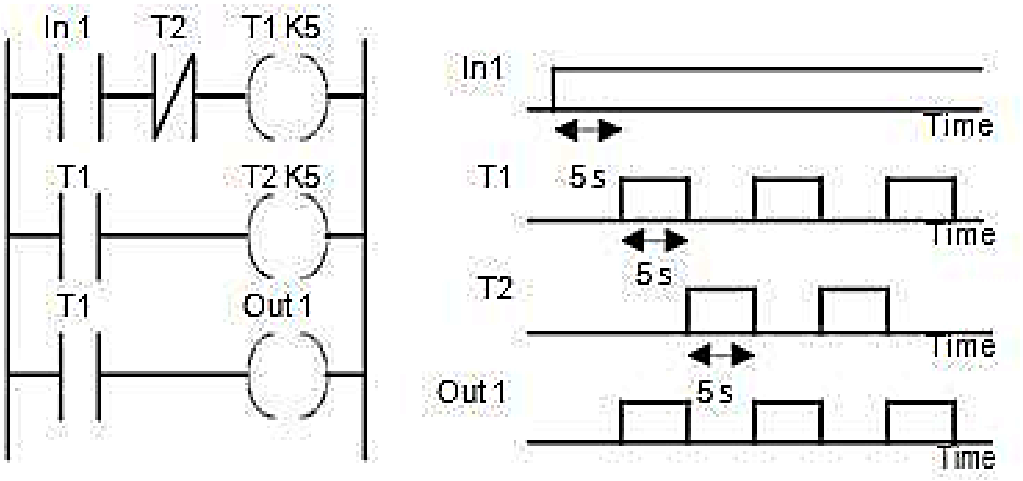


Figure 2.8 *On-off cycle timer*

Figure 2.9 shows how the above ladder program would appear in the format used with a timer considered as a delay, rather than as a coil. This might, for example, be with Siemens or Toshiba. When input In 1 closes, the timer T1 starts. After its preset time, there is an output to Out 1 and timer T2 starts. After its preset time there is an output to the internal relay IR1. This opens its contacts and stops the output from Out 1. This then switches off timer T2. The entire cycle can then repeat itself.

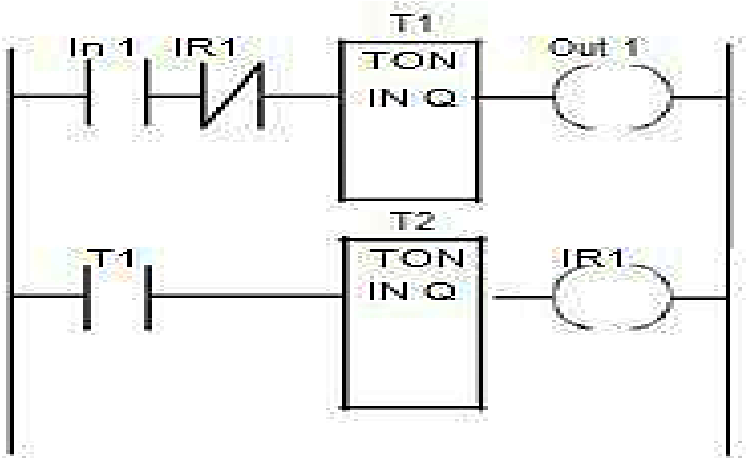


Figure 2.9 *On–off cycle timer*

Off-delay timers

Figure 2.10 shows how a on-delay timer can be used to produce an *off-delay timer*. With such an arrangement, when there is a momentary input to In 1, both the output Out 1 and the timer are switched on. Because the input is latched by the Out 1 contacts, the output remains on. After the preset timer time delay, the timer contacts, which are normally closed, open and switch off the output. Thus the output starts as on and remains on until the time delay has elapsed. Some PLCs have, as well as on-delay timers, built-in off-delay timers and thus there is no need to use an on-delay timer to produce an off-delay timer. Figure 2.11 illustrates this for a Siemens PLC, giving the ladder diagram and the instruction list. Note that with this manufacturer, the timer is considered to be a delay item in a rung, rather than as a relay. In the rectangle symbol used for the timer, the 0 precedes the T and indicates that it is an on-delay timer. As an illustration of the use of an off-delay timer, consider the Allen-Bradley program shown in Figure 2.12. TOF is used to indicate that it is an off-delay, rather than on-delay (TON) timer. The time base is set to 1:0 which is 1 s. The preset is 10 so the timer is preset to 10 s. In the first rung, the output of the timer is taken from the EN (for enable) contacts. This means that there is no time delay between an input to I:012/01 and the EN output. As a result the EN contacts in rung 2 close immediately there is an I:012/01 input. Thus there is an output from O:013/01 immediately the input I:012/01 occurs. The TT (for timer timing) contacts in rung 3 are energised just while the timer is running. Because the timer is an off-delay timer, the timer is turned on for 10 s before turning off. Thus the TT contacts will close when the set time of 10 s is running. Hence output O:012/02 is switched on for this time of 10 s. The DN (for done) contacts which are normally closed, open after the 10 s and so output O:013/03 comes on after 10 s. The DN contacts which are normally open, close after 10 s and so output O:013/04 goes off after 10 s.

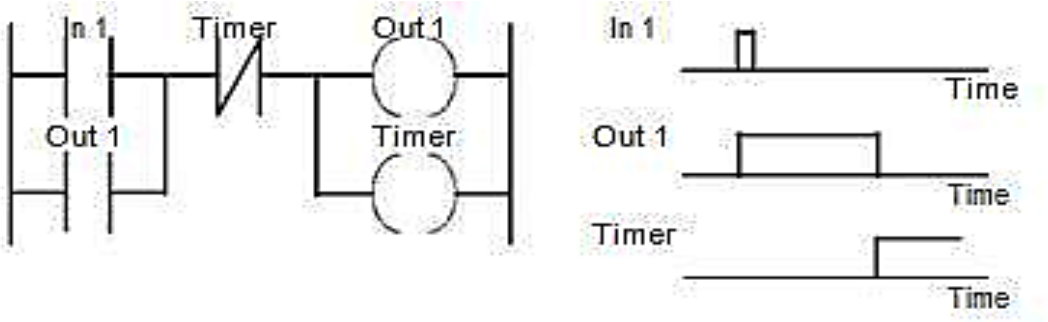


Figure 2.10 *Off-delay timer*

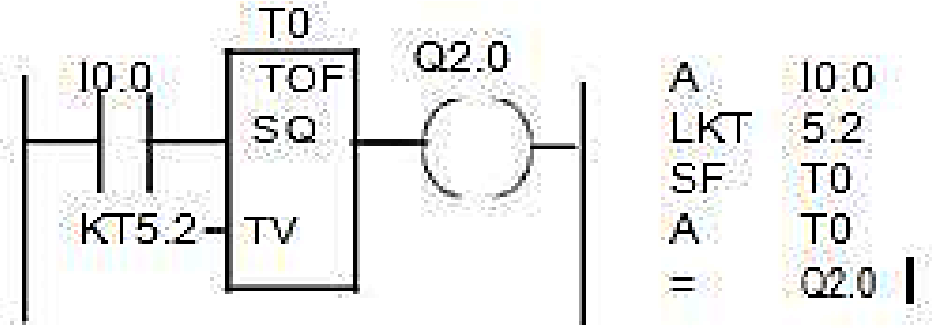


Figure 2.11 *Off-delay timer*

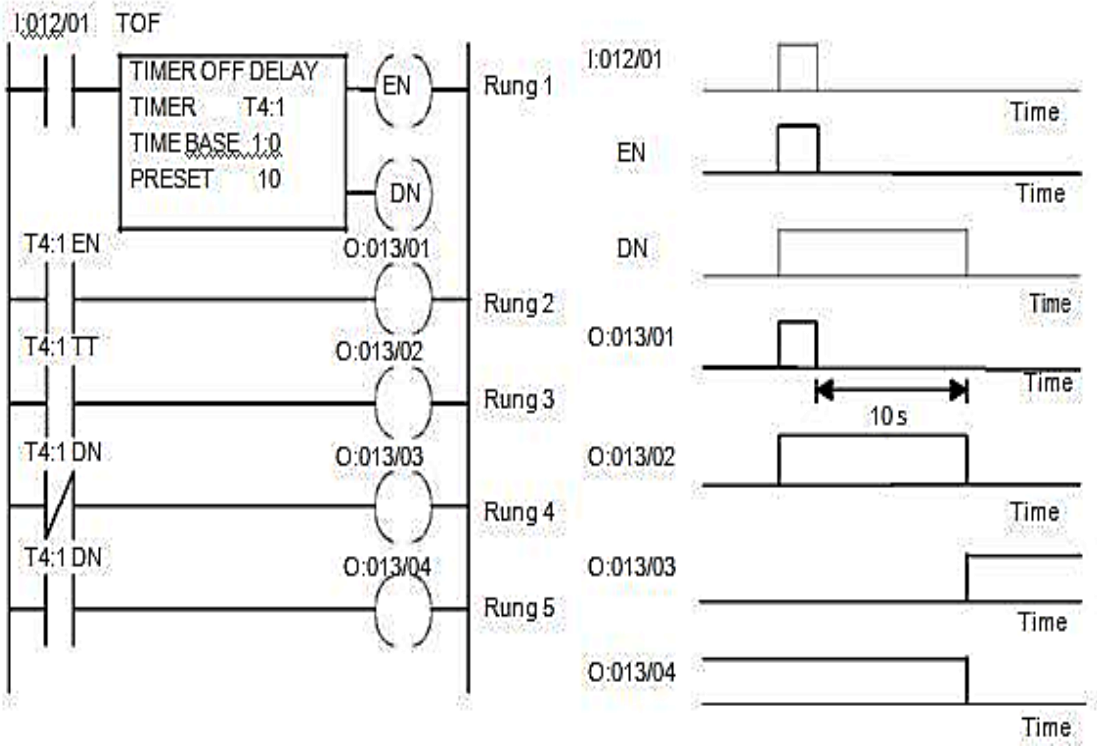


Figure 2.12 *Application of an off-delay timer*

Pulse timers

Pulse timers are used to produce a fixed duration output from some initiating input. Figure 2.13 (a) shows a ladder diagram for a system that will give an output from Out 1 for a predetermined fixed length of time when there is an input to In 1, the timer being one involving a coil. There are two outputs for the input In 1. When there is an input to In 1, there is an output from Out 1 and the timer starts. When the predetermined time has elapsed, the timer contacts open. This switches off the output. Thus the output remains on for just the time specified by the timer.

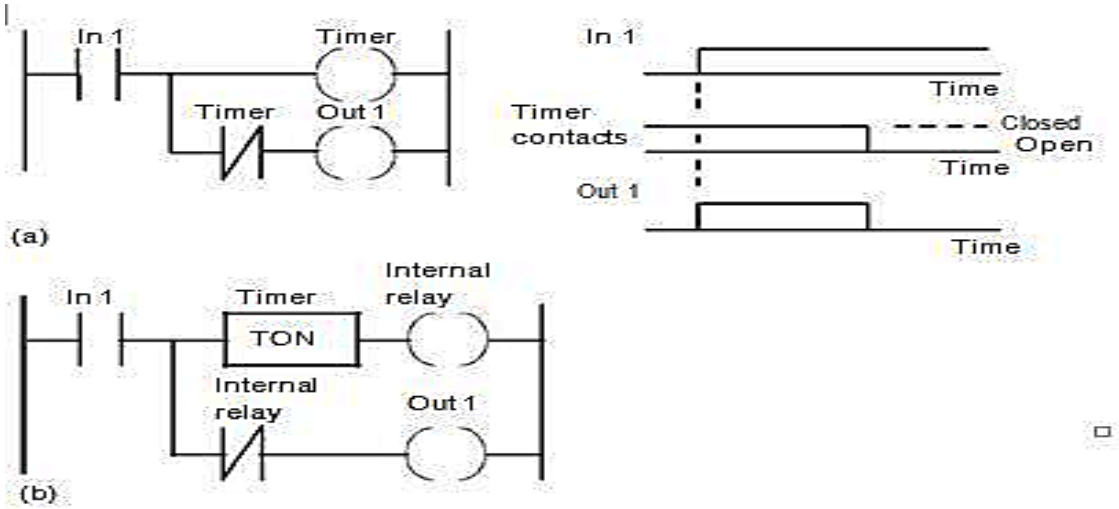


Figure 2.13 *Pulse-on timer*

Figure 2.13 (b) shows an equivalent ladder diagram to Figure 2.13 (a) but employing a timer which produces a delay in the time taken for a signal to reach the output.

In Figure 2.13, the pulse timer has an output switched on by an input for a predetermined time, then switching off. Figure 2.14 shows another pulse timer that switches an output on for a predetermined time after the input ceases. This uses a timer and two internal relays. When there is an input to In 1, the internal relay IR 1 is energised. The timer does not start at this point because the normally closed In 1 contacts are open. The closing of the IR 1 contacts means that the internal relay IR 2 is energised. There is, however, no output from Out 1 at this stage because, for the bottom rung, we have In 1 contacts open. When the input to In 1 ceases, both the internal relays remain energised and the timer is started. After the set time, the timer contacts, which are normally closed, open and switch off IR 2. This in turn switches off IR 1. It also, in the bottom rung, switches off the output Out 1. Thus the output is off for the duration of the input, then being switched on for a predetermined length of time.

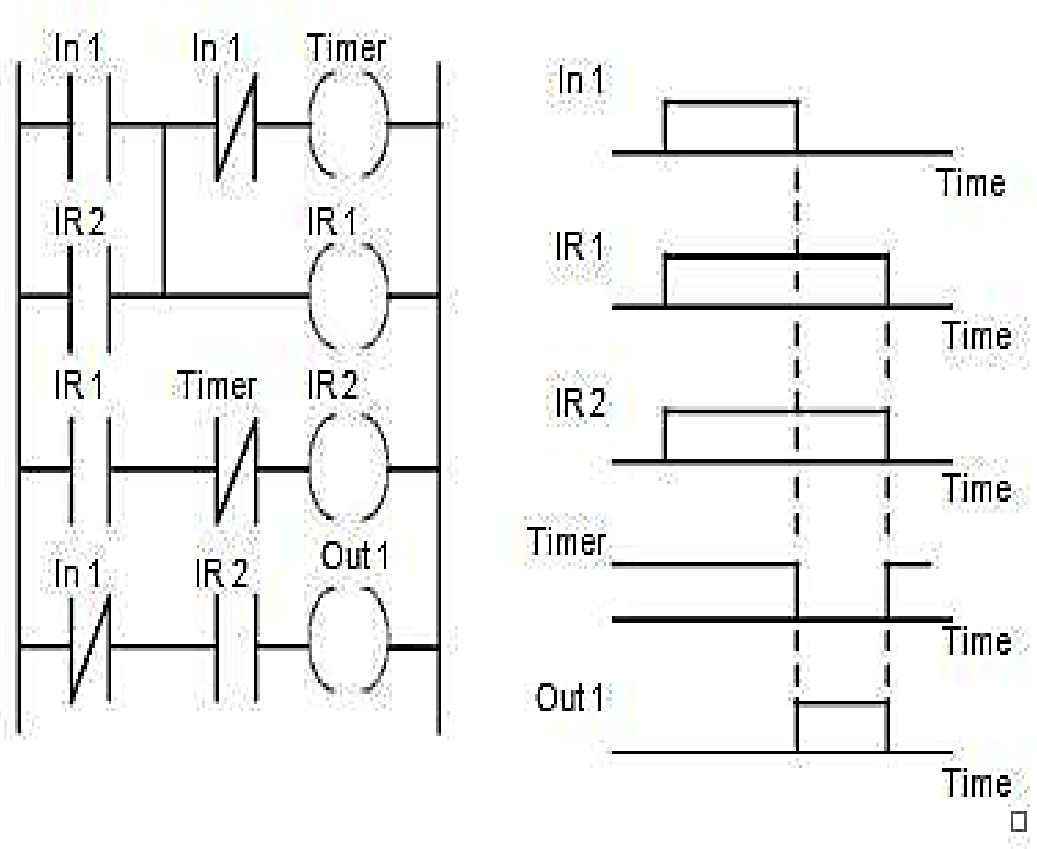


Figure 2.14 *Pulse timer on, when output ceases*

Programming examples

Consider a program (Figure 2.15) that could be used to flash a light on and off as long as there is some output occurring. Thus we might have both timer 0 and timer 1 set to 1 s. When the output occurs, then timer 0 starts and switches on after 1 s. This closes the timer 0 contacts and starts timer 1. This switches on after 1 s and, in doing so, switches off timer 0. In so doing, it switches off itself. The lamp is only on when timer 0 is on and so we have a program to flash the lamp on and off as long as there is an output.

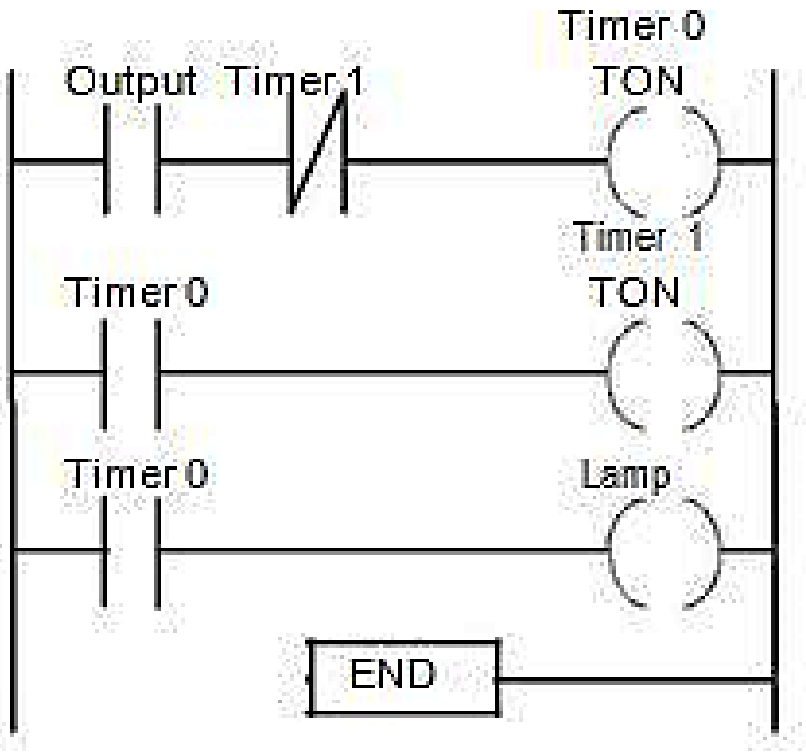


Figure 2.15 *Flashing light*

Counters

Counters are provided as built-in elements in PLCs and allow the number of occurrences of input signals to be counted. This might be where items have to be counted as they pass along a conveyor belt, or the the number of revolutions of a shaft, or perhaps the number of people passing through a door.

Forms of counter

A counter is set to some preset number value and, when this value of input pulses has been received, it will operate its contacts. Thus normally open contacts would be closed, normally closed contacts opened.

There are two types of counter, though PLCs may not include both types. These are down-counters and up-counters. *Down-counters* count down from the preset value to zero, i.e. events are subtracted from the set value. When the counter reaches the zero value, its contacts change state. Most PLCs offer down counting. *Up-counters* count from zero up to the preset value, i.e. events are added until the number reaches the preset value. When the counter reaches the set value, its contacts change state.

Different PLC manufacturers deal with counters in slightly different ways. Some count down (CTD), or up (CTU), and reset and treat the counter as though it is a relay coil and so a rung output. In this way, counters can be considered to consist of two basic elements: one relay coil to count input pulses and one to reset the counter, the associated contacts of the counter being used in other rungs. Figure 2.16 (a) illustrates this. Mitsubishi is an example of this type of manufacturer. Others treat the counter as an intermediate block in a rung from which signals emanate when the count is attained. Figure 2.16 (b) illustrates this. Siemens is an example of this type of manufacturer.

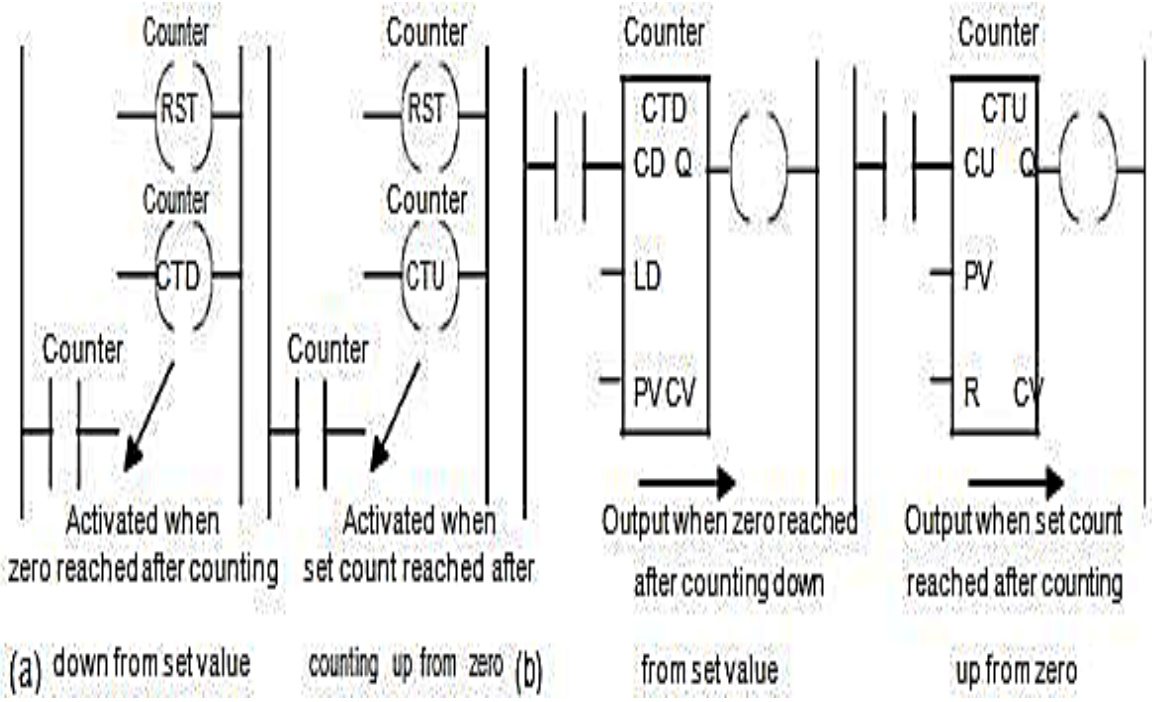


Figure 2.16 *Forms of representation of counters. In (a) RST is reset. In (b), the IEC 1131-3 representation, CD is count down input, LD is for loading the input, PV is for the preset value, CV the current count value, CU is count up input, and R is for the reset input.*

Programming

Figure 2.17 shows a basic counting circuit. When there is a pulse input to In 1, the counter is reset. When there is an input to In 2, the counter starts counting. If the counter is set for, say, 10 pulses, then when 10 pulse inputs have been received at In 2, the counter's contacts will close and there will be an output from Out 1. If at any time during the counting there is an input to In 1, the counter will be reset and start all over again and count for 10 pulses.

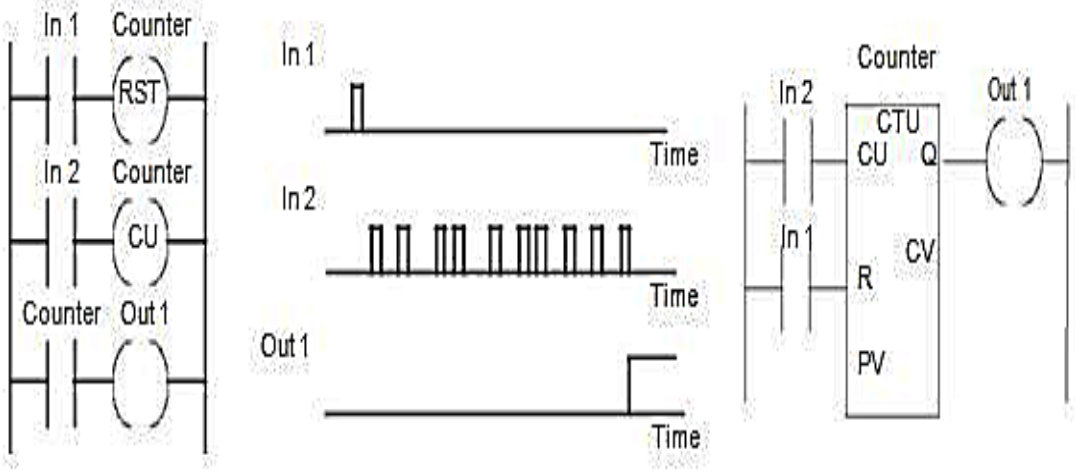


Figure 2.17 *Basic counter program*

Up and down counting

It is possible to program up - and down-counters together. Consider the task of counting products as they enter a conveyor line and as they leave it, or perhaps cars as they enter a multi-storage parking lot and as they leave it. An output is to be triggered if the number of items/cars entering is some number greater than the number leaving, i.e. the number in the parking lot has reached a 'saturation' value. The output might be to illuminate a 'No empty spaces' sign. Suppose we use the up-counter for items entering and the count down for items leaving. Figure 2.18 (a) shows the basic form a ladder program for such an application can take. When an item enters it gives a pulse on input In 1. This increases the count by one. Thus each item entering increases the accumulated count by 1. When an item leaves it gives an input to In 2. This reduces the number by 1. Thus each item leaving reduces the accumulated count by 1. When the accumulated value reaches the preset value, the output Out 1 is switched on. Figure 2.18 (b) shows the implementation of this program with an Allen-Bradley program.

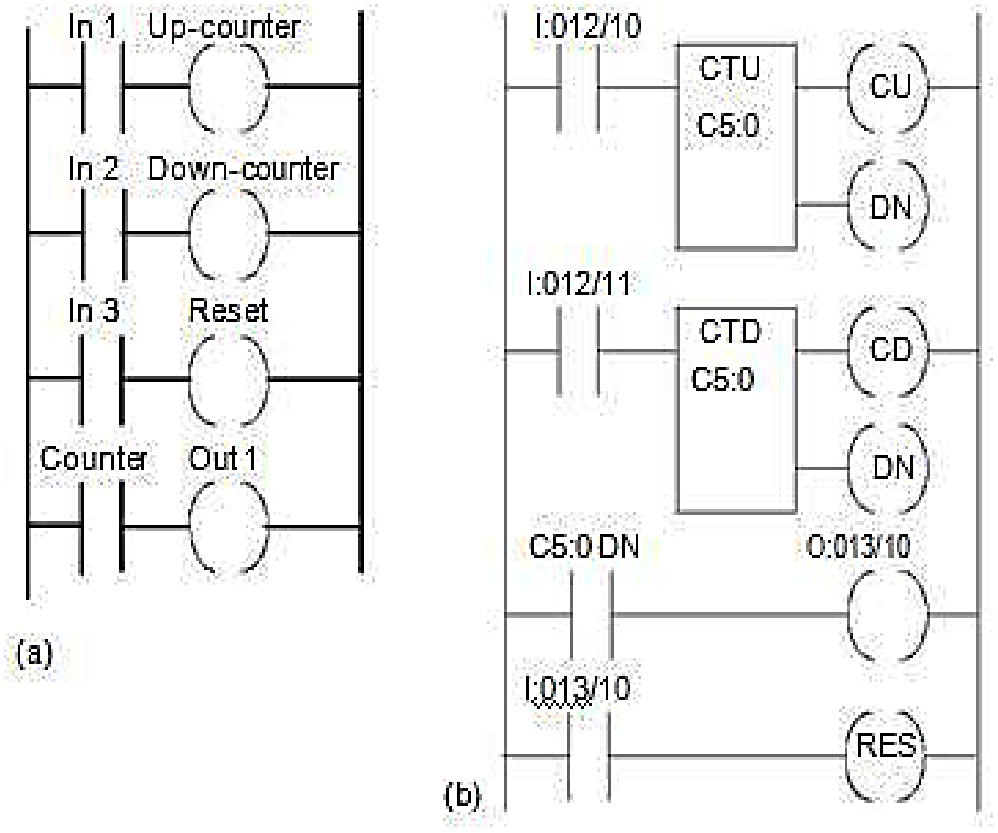


Figure 2.18 *(a) Using up- and down-counters, (b) Allen-Bradley program*

Up-down counters are available as single entities. Figure 2.19 shows the IEC 1131-3 standard symbol. The counter has two inputs CU and CD and counts up the number of pulses detected at the input CU and counts down the number of pulses detected at input CD. If the counter input reaches zero, the QD output is set on and the counting down stops. If the count reaches the maximum value PV, the QU output is set on and the counting up stops. CV is the count value. LD can be used to preset the counter output CV with the value PV. The reset R clears the counter input to zero.

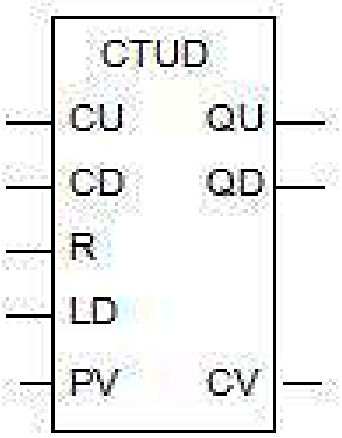


Figure 2.19 *IEC 1131-3 - standard symbol for up-down counter*

Figure 2.20 shows how the above system might appear for a Siemens PLC and the associated program instruction list. CU is the count up input and CD the count down. R is the reset. The set accumulator value is loaded via F0.0, this being an internal relay.

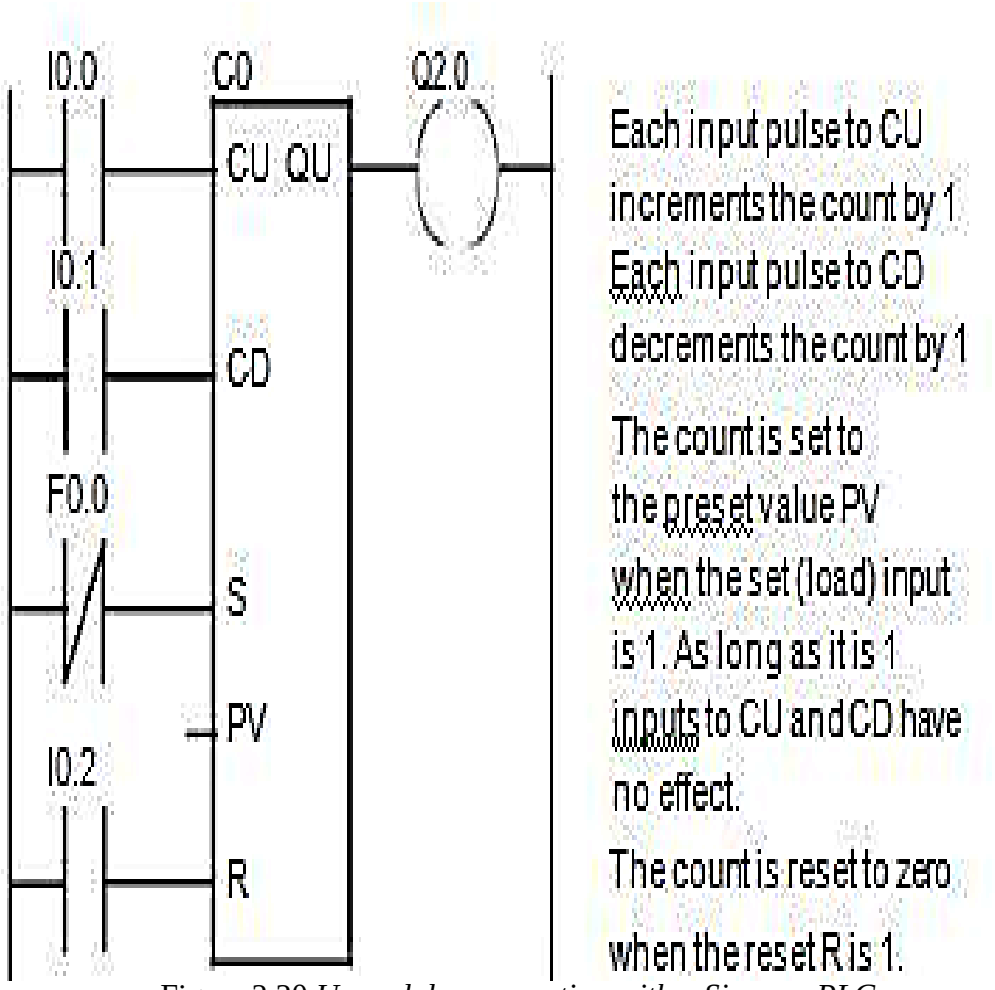


Figure 2.20 *Up and down counting with a Siemens PLC*

Timers with counters

A typical timer can count up to 16 binary bits of data, this corresponding to 32 767 base time units. Thus, if we have a time base of 1 s then the maximum time that can be dealt with by a timer is just over 546 minutes or 9.1 hours. If the time base is to be 0.1 s then the maximum time is 54.6 minutes or just short of an hour. By combining a timer with a counter, longer times can be counted. Figure 2.21 illustrates this with an Allen-Bradley program. If the timer has a time base of 1 s and a preset value of 3600, then it can count for up to 1 hour. When input I:012/01 is activated, the timer starts to time in one second increments. When the time reaches the preset value of 1 hour, the DN bit is set to 1 and the counter increments by 1. The DN bit setting to 1 also reset the timer and the timer starts to time again. When it next reaches its preset time of 1 hour, the DN bit is again set to 1 and the counter increments by 1. With the counter set to a preset value of 24, the counter DN bit is set to 1 when the count reaches 24 and the output O:013/01 is turned on. We thus have a timer which is able to count the seconds for the duration of a day and would be able to switch on some device after 24 hours.

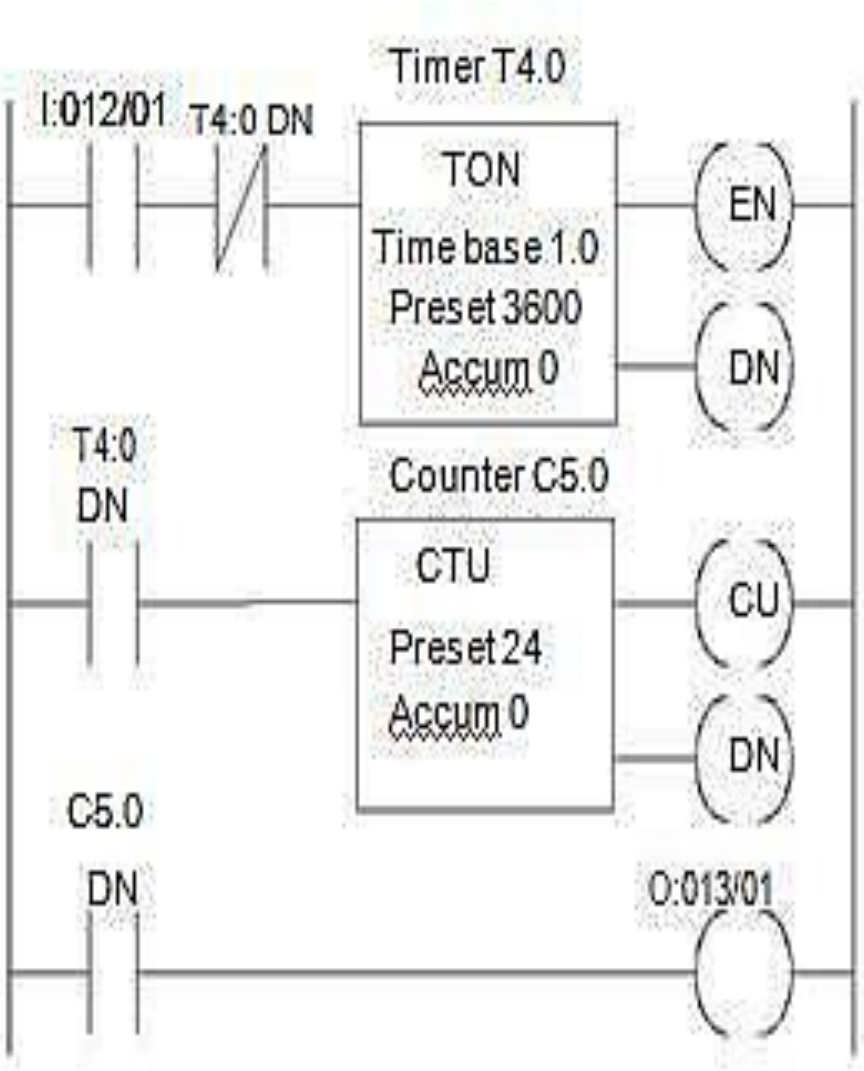


Figure 2.21 *Using a counter to extend the range of a timer*

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF MECHANICAL ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

COURSE MATERIAL

Program: B.E - MTRON Semester: VI
Course: PLC and Automation
Course code: SMR1303

UNIT 3 - PROGRAMMABLE LOGIC CONTROLLERS - SMR1303

Traffic light control, 24 hour clock design, Automatic stacking process, temperature control, Automatic control of warehouse door, Automatic lubrication of supplier Conveyor belt, motor control.

UNIT 3 - PROGRAMMABLE LOGIC CONTROLLERS - SMR1303

1. Continuous bottle filling system:-

This is one of the important application of PLC in the bottle filling industry where we want our bottles, which are moving on the conveyor belt, to be automatically detected at the appropriate position and get it filled by any desired liquid and also after getting filled the queued bottle gets chance to be filled. If this whole process is carried out manually it will really take a long time and also the quantities will be quite lesser. So PLC becomes requisite controller for these types of industry.

Here also just a small demonstration of the process was performed with the help of PLC where a ladder diagram was created to control the process and the ladder diagram was run the PLC trainer kit to see its justification.

Objective:-

We will implement a control program that detects the position of a bottle via a limit switch then waits for 0.5 secs, and then fills the bottle until a photo detector detects the filled condition of the bottle. After the bottle is filled, the buzzer sounds and the control program will again wait for

0.7 secs. before moving to the next bottle .Until the limit switch signals ,the feed motor,M1 runs while there are fixed rollers which carries the filled bottles. Motor, M2 keeps running after the process has been started.

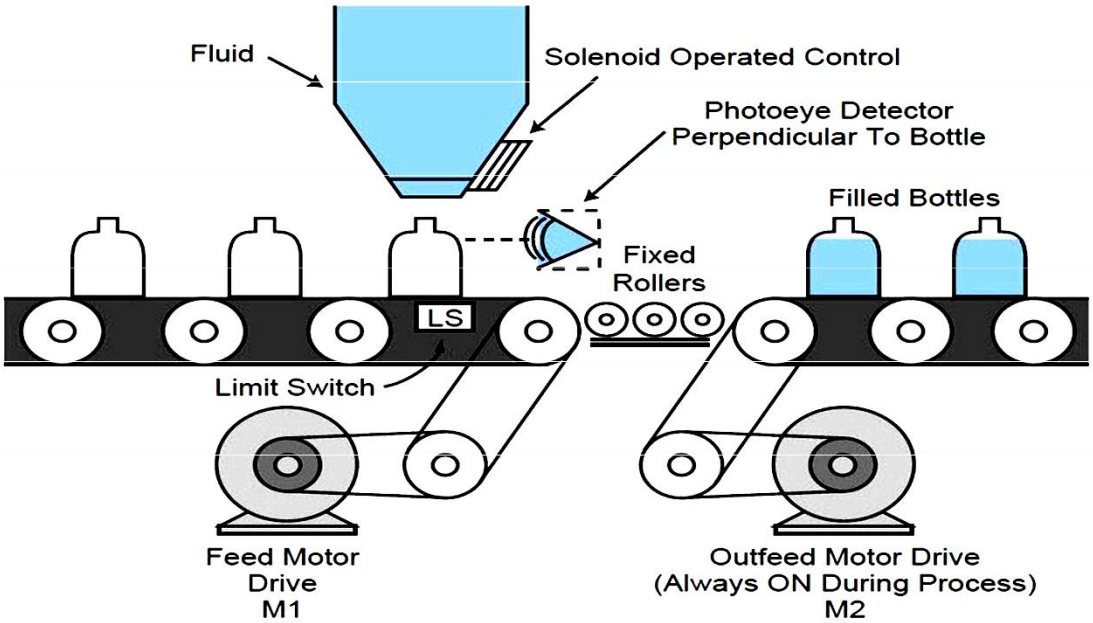


Figure 4.1 Bottle filling system

Inputs	address
Start	I0:15
Stop	I1:15
Limit switch(LS)	I2:15
Photo detector(PE)	I3:15

Outputs	address
Feed motor(M1)	O0:15
Outfeed motor(M2)	O1:15
Solenoid valve(S1)	O2:15
Light(L1)	O3:15
Buzzer(B1)	O4:15

Table 1: Inputs and outputs employed

Ladder diagram:-

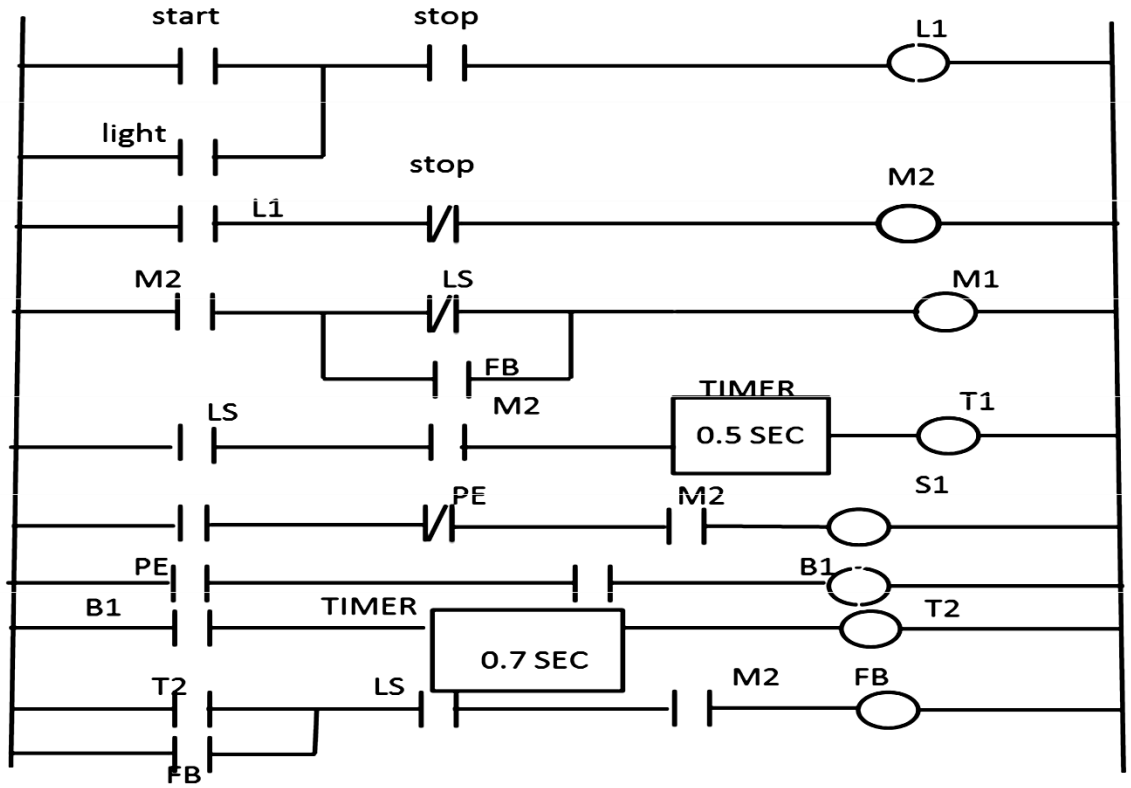


Figure 4.2 ladder diagram for bottle filling system

Observation:-

Once the start button is pressed the green light (L1) turns ON and remains ON until stop button is pressed. As light turns ON out feed motor (M2) starts running. After M2 runs and if either limit switch (LS) has not signaled or filled bottle condition is fulfilled motor (M1) starts. After limit switch has signaled timer, T1 gets activated. After T1 gives done (DN) signal and photo eye detector (PE) is disabled, solenoid valve gets in operation. A s PE signals solenoid stops and buzzer (B1) sounds after which timer, T2 gets enabled which stops the process for 0.7 seconds. Once the filled bottle condition is activated the cycle starts again.

The ladder diagram was successfully checked in the PLC simulator and all the prescribed conditions were observed completely.

2. Batch mixing system

This is another commonly applied application of PLC where two liquids are mixed in required proportion to form a batch .Rate of the flow is already fixed. We only control the time of the flow. Level of the liquids in the tank are sensed by the level sensor switches

Objective:-

We try a simple blending of water and acid in a container where we only have three level sensors(L1,L2, and L3) and two liquids flowing in through two solenoid valves, solenoid a(water control) and solenoid b(acid control)and draining out through solenoid c(blend outflow).The batch is to be controlled by timer. After required level of blend is sensed (by L1) the mixer runs for 3 mins by the motor. They are mixed in ratio of 3:2. The process initiates with the drain valve open, water and acid valves closed, mixer motor is off, and the tank is empty.

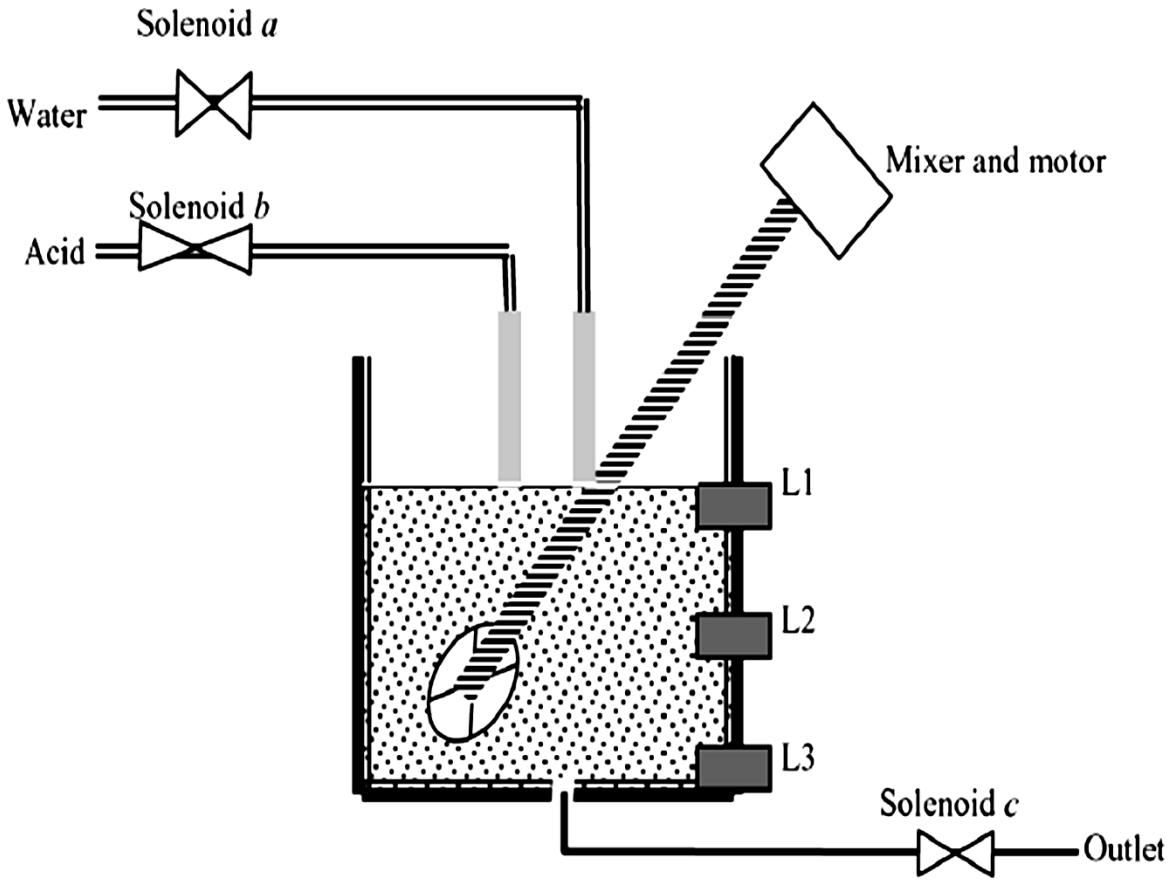


Figure 4.3 Batch mixing system

Ladder diagram:-

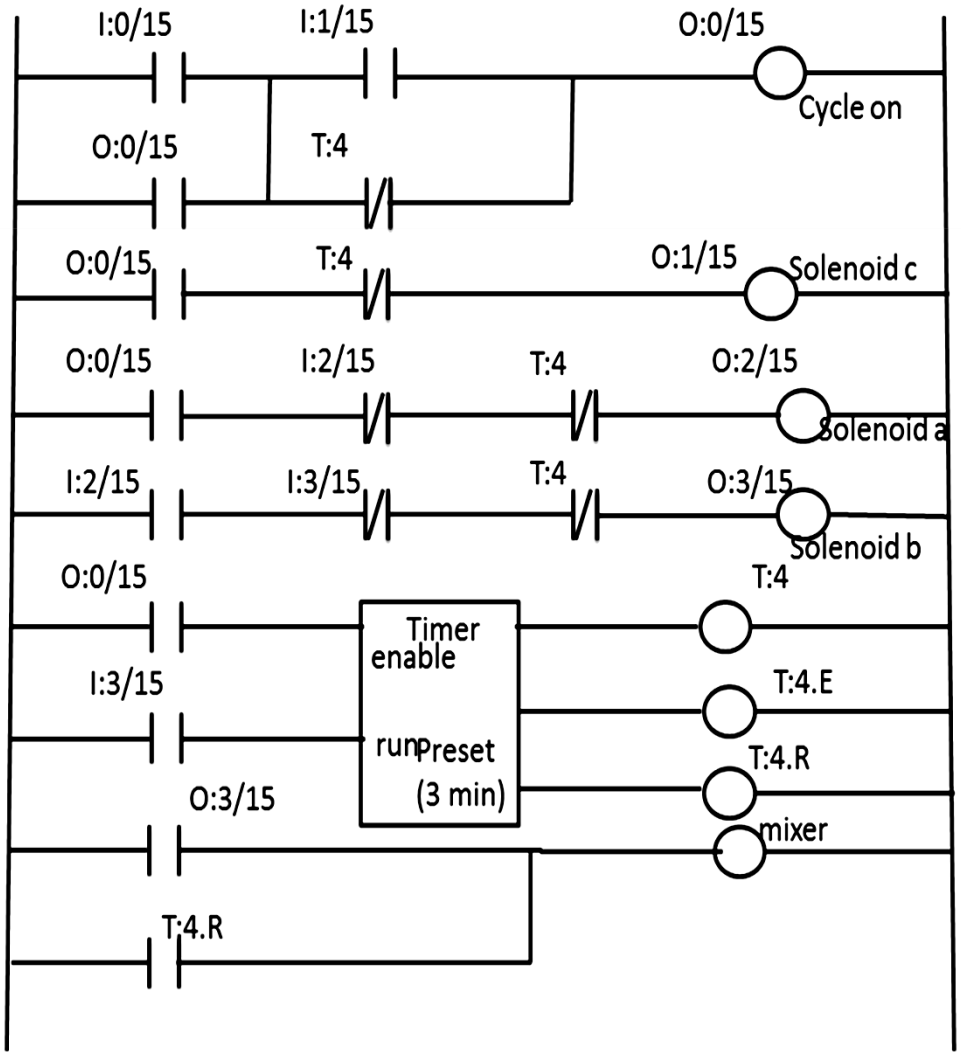


Figure 4.4 Ladder diagram for batch mixing system

Observation:-

When start button is pressed water is filled upto L2 and it ends as L2 is closed. First of all as start is pressed output O:0/15 turns ON and remains ON until tank is emptied. Rung 2 closes normally open drain valve, before timer T:4 activates. Rung 3 energises solenoid a until L2 doesn't signal, once it signals solenoid a gets de-energised. Then motor is turned ON and mix it for 3 mins. Similarly acid is filled upto L3 by solenoid b.as level gets detected by L3 solenoid b de-energises and then mixer gets started and it runs for 3 minutes. After time delay of 3 mins solenoid c opens and the blend gets drained out .Once the blend gets out completely, the process cycle restarts.

The ladder diagram was successfully checked in the PLC simulator and all the prescribed conditions were observed completely.

3. 3 -stage air conditioning system:-

A simple air conditioner consists of a single air compressor motor which gets switched off when temperature of the space being controlled falls below the setting on the thermostat. Thermostats are provided with a differential setting to avoid on and off of the compressor motor. The three stage air conditioning system helps in conservation of electrical power.

Objective:-

There are two motors compressors in the system. One is of low horsepower and other one is of high horsepower rating. These motors are designated as C1 and C2 in the case. The system is installed in a hall to maintain the temperature between 20^0C-24^0C depending on the number of viewers in the hall and the atmospheric temperature.

The motors of C1 and C2 are run on three conditions of the thermostats. The three conditions described below are also the control requirements of the air conditioning system:-

1. Compressor 1 and compressor 2 should turn on when the temp. of the hall is above 28^0C .

2. Only compressor 2 should turned on when the temp of the hall is above 24^0C and below 28^0C.

3. Only compressor 1 is turned on when the temp. is above 20^0C and below 24^0C. A pre-condition for running any compressor is that chilling water flow switch FS1 should be closed. Chilling water flow necessary to take away heat from the compressed cooling water.

Three thermostat with different settings are used for the control of compressor motor running in three different stages described above. The three thermostats T1, T2, T3, are set at temp 20^0C, 24^0C and 28^0C respectively.

The control of three stages with three the thermostats and Flow switch (FS1) of air conditioning system can be understood from the control circuit shown in fig.4.5.
The start push button, stop push button and overload contacts for compressor motor have not been shown in the circuit for sake of simplicity.

Working of the control circuit:-

1) When chilled water flow is maintained, flow switch FS1 will actuate and close its contacts. Closing of contact FS1 causes application of high logic signal to terminal 2 of all gates.

(2) When the temp. in the cinema hall will be above 28^0C the contacts of all the thermostat will be closed. Closed contact of T1 will give a high input to terminal 1 of AND1 but closed contact of T2 will give low logic to terminal 3 of AND1 as there is a not gate in series. The output of AND1, therefore becomes low.

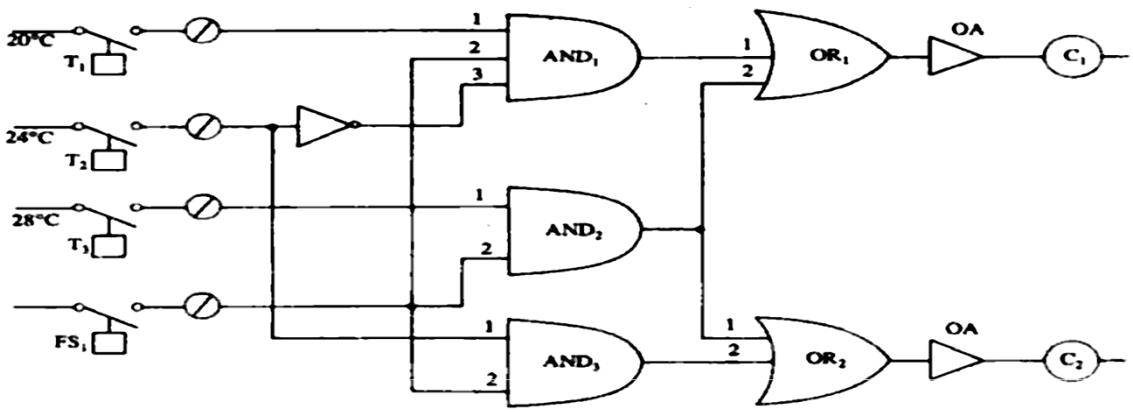


Figure 4.5 logic control circuit for 3 stage air conditioning system

(3) Closed contact of T3 will give a high input to terminal 1 of AND2 while the other terminal 2 is already high due to closure of FS1 AND2 thus gives a high output which is applied to both OR1 and OR2,as each terminal of OR1andOR2 is now high, their output is also high. Output from OR1 leads to energisation of contactor C1 and output from OR2 leads to energisation of contactor C2 through their respective amplifiers. Thus compressor 1 and 2 will run when temp in the hall will above 28^0C.

(4) When the hall temp is below 28^0C and above 24^0C, contact of thermostat T3 will open, while contacts of T 1andT 2 are closed. Due to open contact of T3 there is low signal at terminal 1 of AND2 and therefore its output is low. Output of AND1 is also low as closed Contact of T2 gives a low signal at terminal 3 due to a not gate in series, in this case AND3 Will have a high output as its input terminal1 has high signal from the closed contact of Thermostat T2, it is to be noted that supply to terminal1 of AND3 is taken prior to the NOT Gate. High output from AND3 goes to terminal 2 of OR2 which then gives a high output. This output energises contactor C2 through the amplifier. Thus compressor 2 will run only when the temperature is above 24^0C but below 28^0C.

(5) When the temp. falls below 24^0C contact of thermostat T2 opens and output of AND3 will Go low due to a low signal in its input terminal the open contact of T2 will however give a high signal to terminal 3 of AND 1 (due to not gate in series), It will get switched on as its terminal 1 and 2 are already high. The high output from and1 then goes to terminal 1 of or1 which then gives a high output to energise contactor C1.thus one compressor will run when temp is below 24^0C but above 20^0C. When temp falls below 20^0C, contact of thermostat T1 also opens and terminal 1 of AND1 goes low and it is switched off .Thus Compressor 1 also stops when temp falls below 20^0C.

(6) Compressor 1 continues to run if temp. In the hall remains 20^0C -24^0C, if due to more viewers in the hall, compressor 1 is unable to maintain, temp. Below 24^0C. Compressor 1 will be switched off. If the load is still more and compressor 2 alone cannot cope up and temp. Goes above 28^0C then compressor 1 will also start to bring down the temperature.

Ladder diagram:-

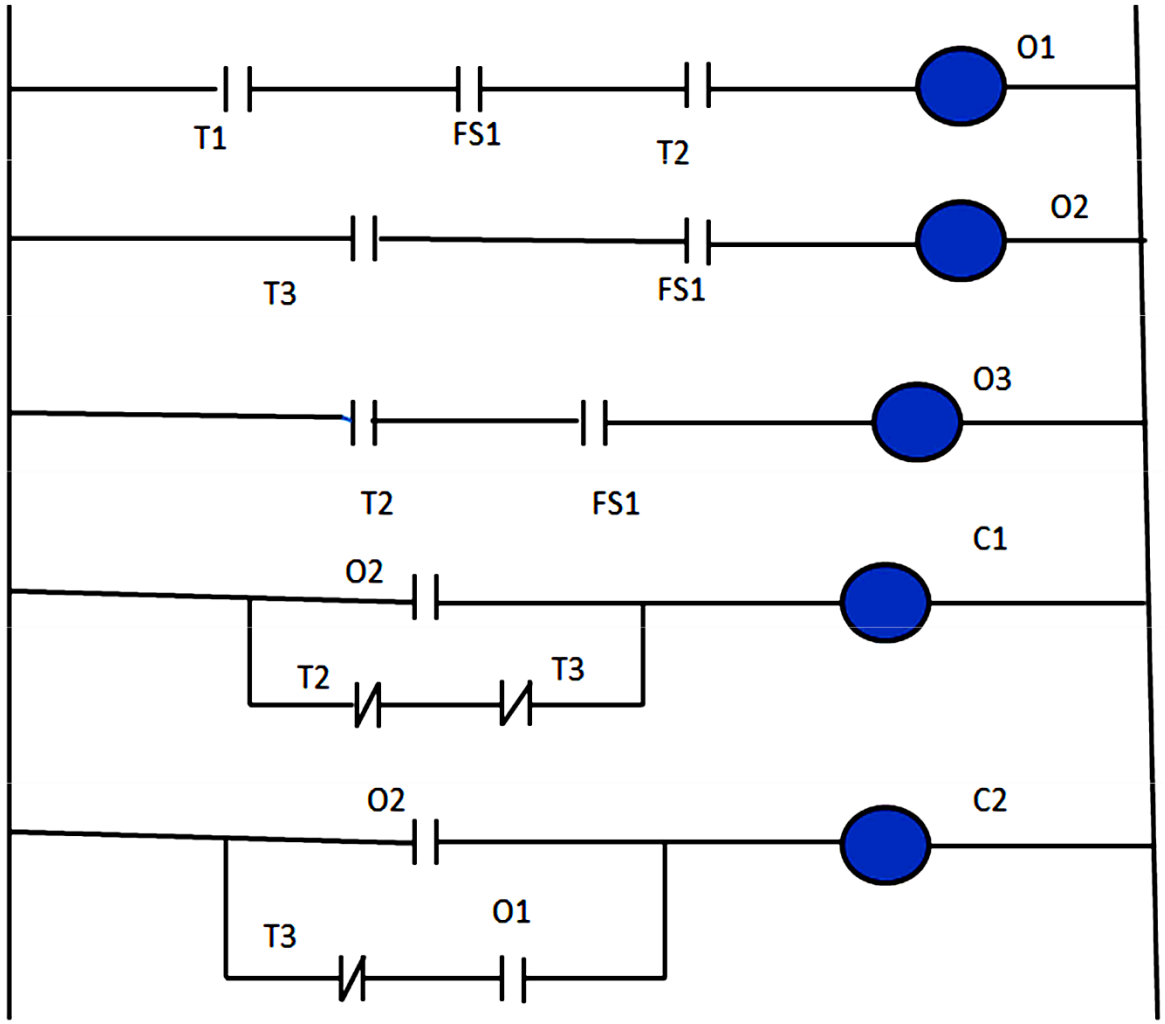


Figure 4.6 Ladder diagram for 3 stage air conditioning system

Observation:-

The ladder diagram was satisfactorily realized in the lab and all the conditions were tested .The outputs are same as expected

4. Control of Planar machine:-

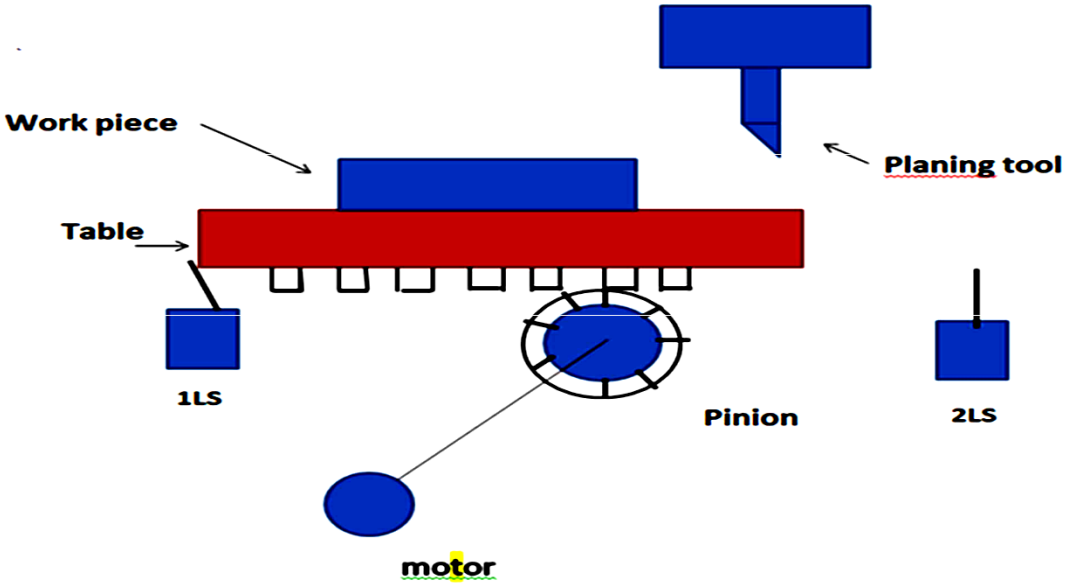


Figure 4.7 Planar machine

In this machine, the work piece or the job placed on the table moves to and fro by rack and pinion arrangement mounted on the shaft of the squirrel cage motor. Here the cutting tool is fixed while the job placed on the table is worked upon by the movement of table. Movement of the table is controlled between two limits left and right by switches 1LS and 2LS. When the table moves left to right, tool works on the job while it remains ideal during right to left motion of the table. At the end of right to left motion, tool gets feed for the next cut on the job. Various control requirements for the job are as follows:-

(1) The motor is to be start manually by pressing start push-button. Once the motor starts it reversed automatically at the end of right or left stroke by limit switches2LSand1LS.

(2) There should be provision of jogging the motor by jog push button.

(3) If the machine table is lying in between extreme position, machine should fail to start. Selection of initial direction of travel should be possible through right and left push button, PBR and PBL.

(4) There should be delay in starting the motor in left to right stroke so as to allow the tool To get the feed for the fresh cut on the job.

(5) The machine should stop on pressing the stop push –button or on over load tripping of motor.

(6) Interlocking of coolant pump motor (running) should be provided as a precondition for the starting of machine.

Control circuit for the machine adhering to the above mentioned control requirement shown in fig:-

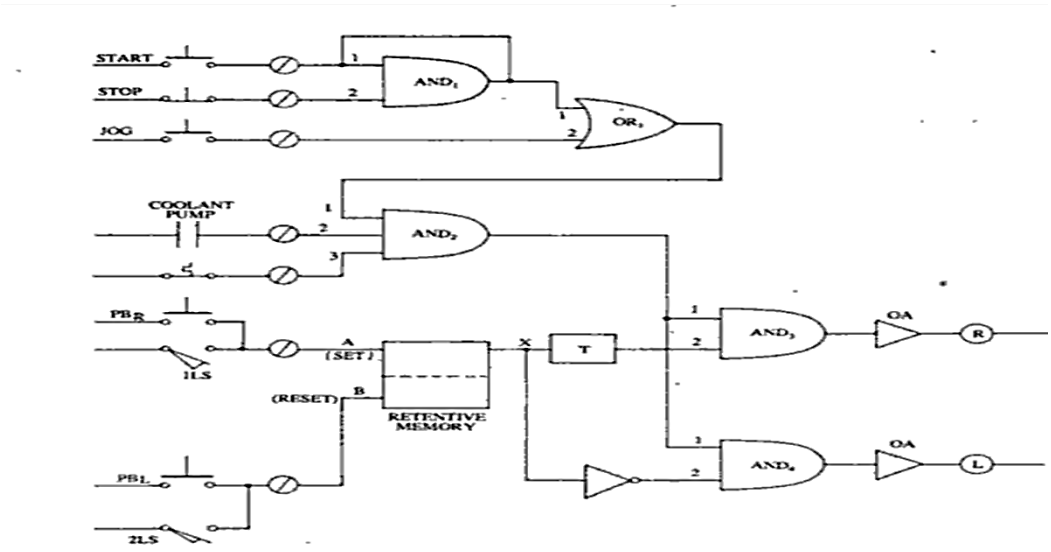


Figure 4.8 logic control circuit for planar machine

There are two major sectors of this control circuit. One is the control of starting and stopping of the motor, and the other is the automatic reversing control:-

Working of the circuit:-

1. When start-push button is pressed, input terminal 1 of AND1 becomes high, its output also goes high as its input terminal2 is also high due to close contact of stop-push button. Output is fed back to terminal1 for holding the output high.
2. Output of AND_1 appears at terminal 1 of OR1 making the output of OR1high which appears at terminal I of AND2. If motor overload relay contact is CLOSED and coolant pump motor is running, then terminals 2and 3 of AND2 also have a high signal. Thus all the three input terminals of AND2, are high and so its output goes high.
3. Output of AND2 appears at terminal 1 of AND3 and AND4 Depending upon the condition of memory element whether it is in set mode or in reset, either AND3 or AND_4 output will go high and will energies their respective contactor R or L. Desired direction of travel may however be selected by pressing the right or left traverse push buttonsPB3 or PB4before pressing the START-push button.
4. To understand the reversing action of circuit, it is assumed that initially the machine table is in extreme left position so that the limit switch ILS is in actuated condition and its normally open (NO) contact is closed. Thus, a high signal appears at terminal A of the retentive memory through closed contact of ILS. Memory element gets set and its output terminal X goes high.
5. High output from terminal X appears at terminal 2 of AND3 after some delay set by the timer T while at terminal 2 of AND4 a low signal appears because of the NO T gate being in series with the output from X.

6. As both the terminals I and 2 of AND3, arc now high, its output becomes high which energize motor contactor R through the amplifier. Machine table thus moves in right direction.

7. When extreme right position is reached, limit switch 2LS gets actuated. Is normally Open (NO) contact closes and resets the memory element. Output at terminal X goes low and thus terminal 2 and AND3 also becomes low. Output of AND3 therefore goes low and contactor R is de-energized. At the same time when memory element gets reset, terminal 2 of AND4 goes high due to a NOT gate inverting the low output from X

8. Thus, due to actuation of limit switch 2LS terminal 2 and AND4 goes high while its terminal 1 is already high. Hence, a high output appears which energize motor contactor L through the amplifier. Motor now runs in reverse direction to move the table from right to left.

9. When the table reaches extreme left position limit switches ILS gets actuated and memory element is set thus terminal 2 of AND4 goes low and its output becomes low. Contactor l is therefore is de-energized. After a delay set on timer Again Contactor R is energized as output of AND3 becomes high. The to and fro motion, due to setting and resetting of memory elements by actuation of limit switches. Continues till the stop push Button is pressed or overloads trips to make the terminal1 of AND3 and AND4 low.

10. If the machine table is required to be moved slowly in steps for the adjustment of the tool with respect to the job position then jog- push button.

Ladder diagram:-

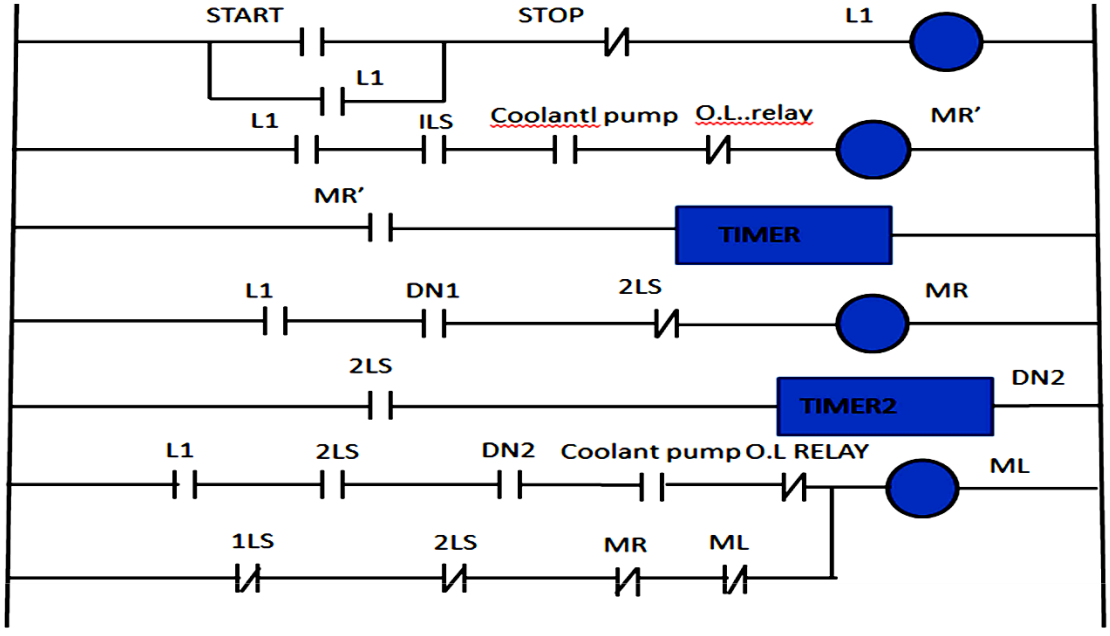


Figure 4.9 Ladder diagram for planar machine control

Observation:-

The ladder diagram was satisfactorily implemented in the lab satisfying all the desired conditions. So the process can be easily controlled by the PLC ladder diagram.

5. Speed control of dc motor

Speed control of a motor means the intentional variation of speed according to the requirement of the work-load connected with the motor. This can be done by mechanical means, such as by using stepped pulleys, a set of change gears, a friction clutch mechanism, etc. However control of speed by electrical means has greater advantages over mechanical speed controls. The dc motors offer easy speed control and that's why dc motors are preferred over other types of motors in many applications. Various speed control method can be obtained from its expression which is: $N=V - I_aR_a/K\phi$, Whereas, N=speed of motor, I_a=armature current, R_a=armature resistance, Φ=field flux, So it can be concluded that speed of dc motor depends upon
a. The applied voltage, b. The field flux, c. Drop in armature circuit resistance I_aR_a
And accordingly speed can be controlled by varying the above factors.

A. Speed control by varying the field flux:-

In shunt field winding motor, variable resistor called field regulator is connected in series while in series motor a resistor called diverter is connected in parallel with series field winding. When field circuit resistance is varied, the field current and so the flux varies. But by introducing field regulator, the field circuit resistance can only be increased i.e. the field flux can only be decreased, and thereby the speed of the motor can be increased. It is not possible to decrease motor's speed by this method. Similarly by increasing the diverter circuit resistance, the field current can be reduced and thereby the speed of series motor cab be increased. Reduction of speed is not possible by using diverter.

B. Speed control by connecting a resistance in series with armature:-

A resistance called the controller is connected in series with the armature .Here the speed of the motor can be reduced as desired. Using desired value of the controller resistance, the speed can be reduced or increased to a great extent. The field winding should be connected across the supply terminals, otherwise the flux produced will be badly affected and sufficient torque may not be produced to rotate the motor.
The disadvantages of this method are as follows:-

a. The overall efficiency of the system is low as much of the input energy is dissipated in the controller as heat.
b. The controller has relatively high cost.
c. The speed may vary largely with variation of load.

C. Speed control by controlling the voltage applied across the armature terminals:-

In this method of speed control the armature is supplied with a variable voltage with the help of a motor-generator set since the supply voltage available from the electricity authority cannot be varied at will. This system of speed control is also known as the Ward-Leonard system.

If a reversing switch is incorporated, by changing the polarity of the armature supply terminals, speed can be varied in the opposite direction also. This system is advantageous over other system in following ways:-

a. It provides smooth control of speed over a wide range in both directions
b. The system is more efficient at low speeds as there are no resistors connected in Series with the armature circuit.

We apply this strategy only to control the motor speed as other methods are loss and inefficient. Here instead of adding extra motor- generator set we control speed by PLC. The PLC supplies control signals to the MOSFET switches .These control signals have variable duty cycle that depends on the speed required. Depending on the duty cycle the motor gets the average voltage and accordingly speed varies.

Consider the circuit shown below .Here we have a dc voltage source V, a resistor R, inductor L, diode D, and a semiconductor switch Q (shown here as an N-channel insulated gate MOSFET). The signal applied to the gate of the switch Q is a pulse train with constant frequency f (and constant period T), but with varying pulse width t. The amplitude of the signal applied to the gate will cause the switch to transition between cutoff and saturation with very short rise and fall times. The relative values of R and L are selected such that the time constant = L/R is at least 10 times the period T of the pulse train applied to the gate of Q. The long L/R time constant will have a low-pass filtering effect on the chopped output of the switch Q, and will effectively smooth the current into dc with very little ac component.

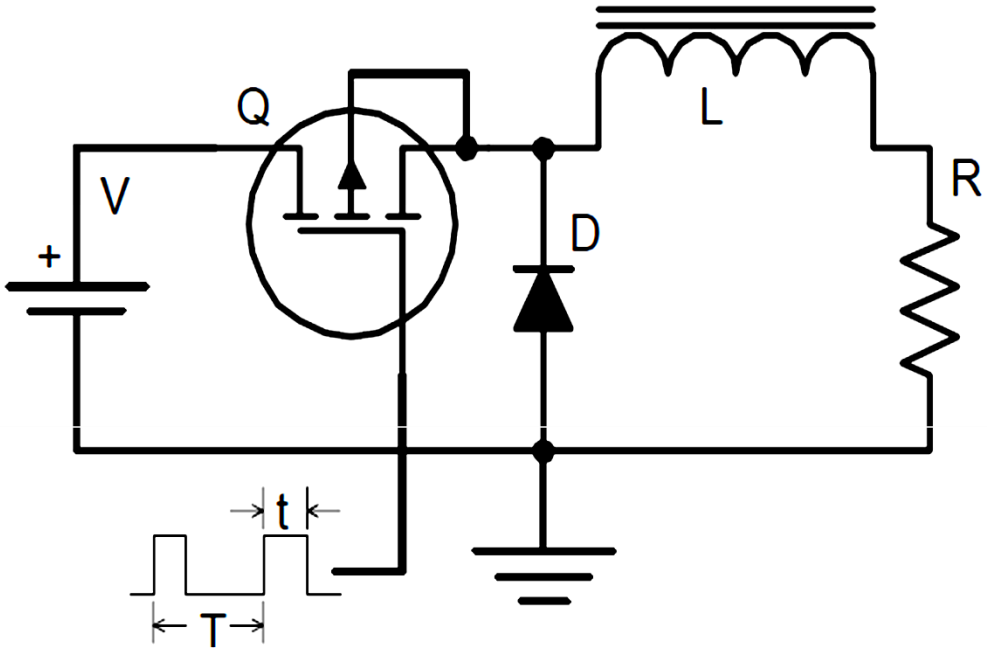


Figure 4.10 simple dc switch voltage controller

For the switch Q, the ratio of the on-time t to the period T is defined as the **duty cycle**, and is represented as a percentage between zero and 100%.For any duty cycle between 0% and 100%, the average resistor voltage will be a corresponding percentage of the voltage V and also the power transferred to the motor. For example, if we adjust the applied gate pulses so that the duty cycle is 35% (i.e., ON for 35% of the time, OFF for 65% of the time), then the voltage on the resistor R will be 35% of the input voltage V. This is because, during the time that the switch is ON, the inductor L will store energy; during the time the switch is OFF, the inductor will give up some of its stored energy keeping current flowing in the circuit through inductor L, resistor R, and the forward biased diode D (in this application, the diode is called a **freewheeling diode** or **commutation diode**).

As in the above figure we have resistance and inductor connected in series we can model dc armature as lumped resistance and lumped inductance in series.

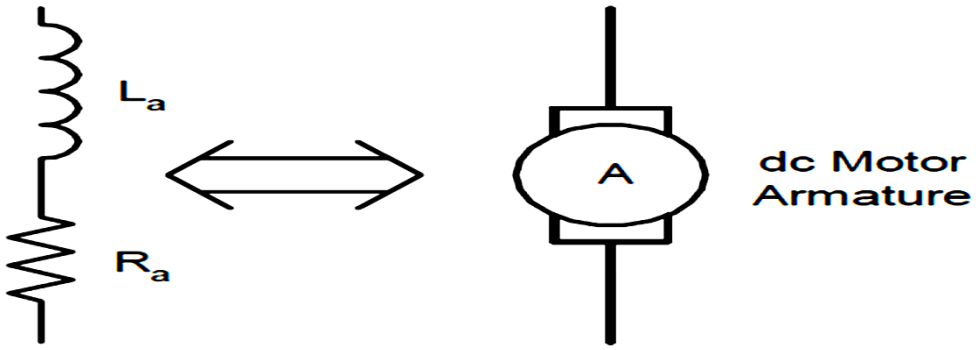


Figure 4.11 dc motor armature model

Since we can model the dc motor armature as a series resistance and inductance, we can substitute the armature in place of the resistor and inductor in our dc switch circuit which looks as

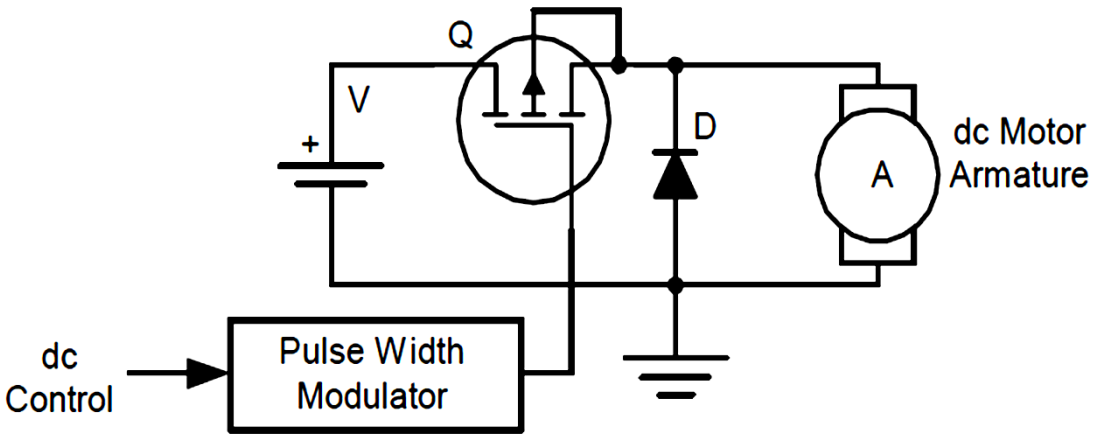


Figure 4.12 dc motor speed control

PLC implementation:-

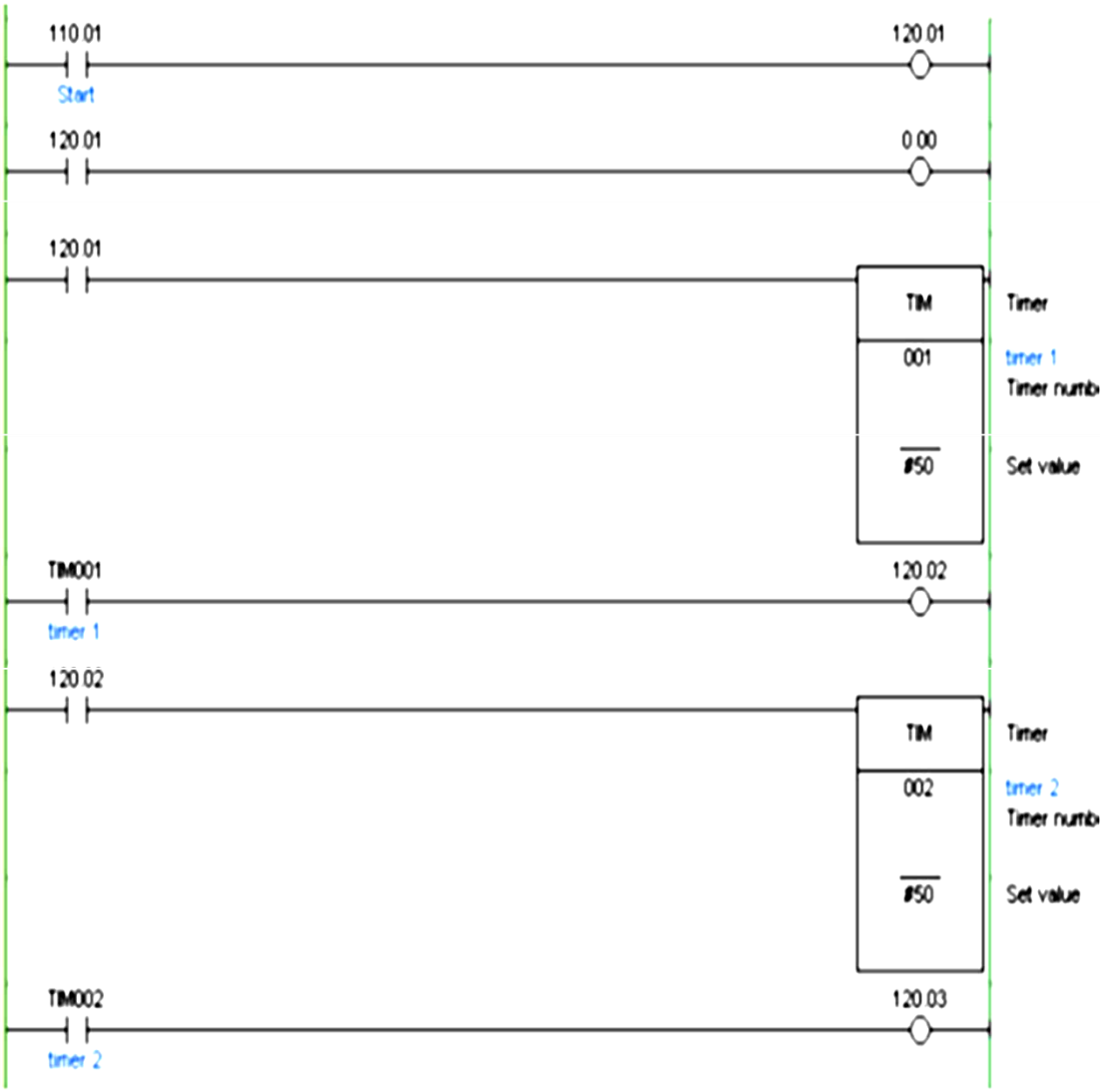


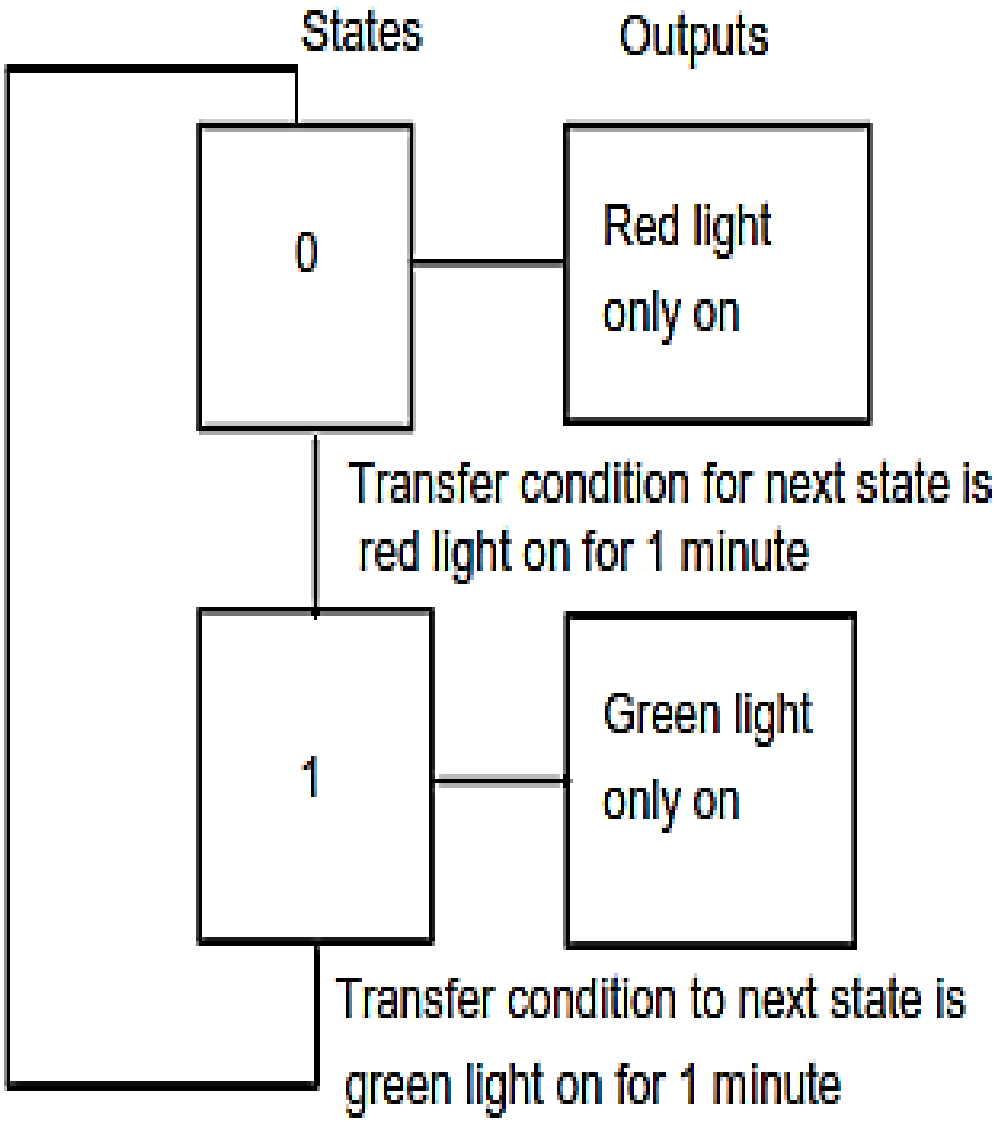
Figure 4.13 ladder diagram for generation of pulses

For generation of pulses, first whole time period is set in counter by any n-bit according to resolution required. Now according to the percentage of rpm required with respected to maximum rpm of the motor, the counter is also set at the same percentage (value with respect to n-digit maximum value) as the voltage. We have comparator to check that value. Till fixed value signals passes while it remains off for remaining period of the pulse. In this way we generate signals of varying duty cycle pulses.

The ladder diagram when simulated gave the same result .Signal remained ON until the counter finished counting and became off for rest period. The process is repeated in every time period.

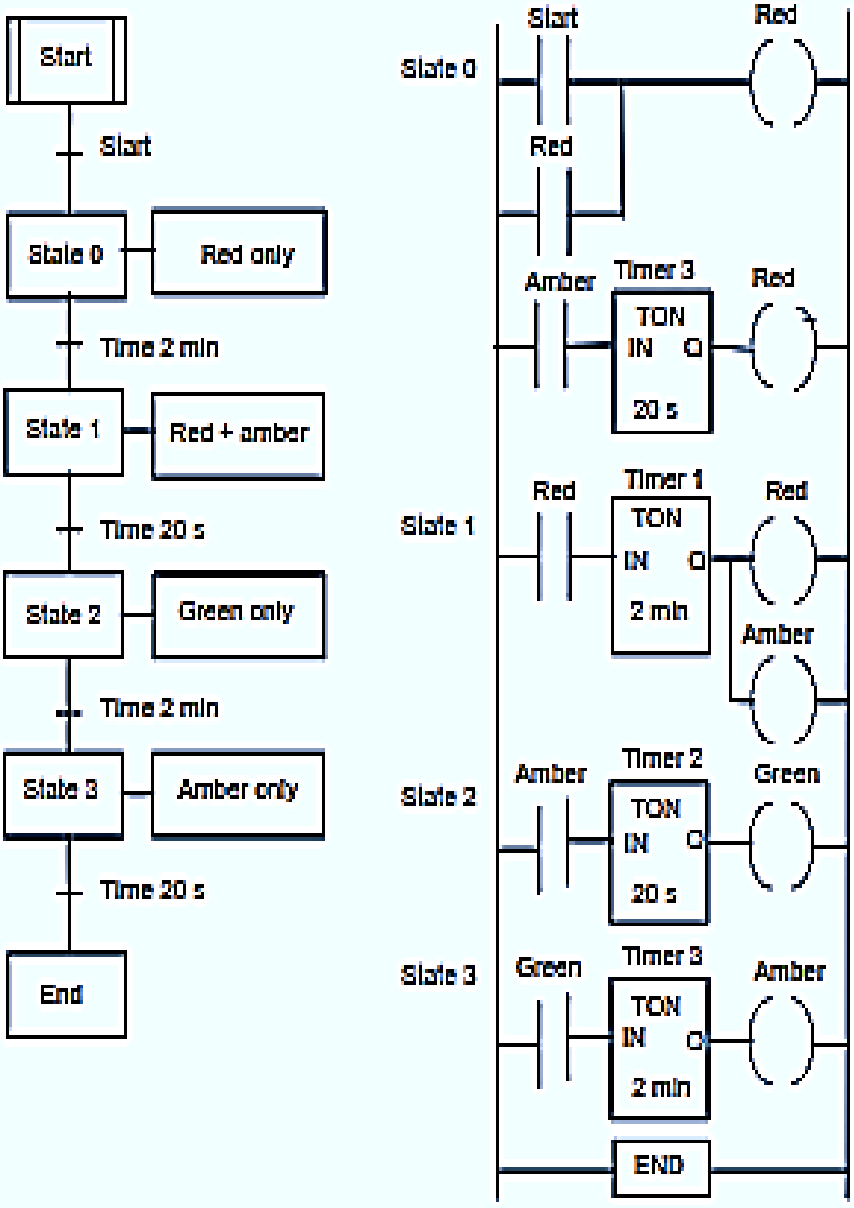
6. Traffic Light controller

If we wanted to describe a traffic lamp sequence, one way we could do this would be to represent it as a sequence of functions or states such as red light state and green light state and the inputs and outputs to each state. Figure illustrates this. State 0 has an input which is triggered after the green light has been on for 1 minute and an output of red light on. State 1 has an input which is triggered after the red light has been on for 1 minute and an output of green light on.



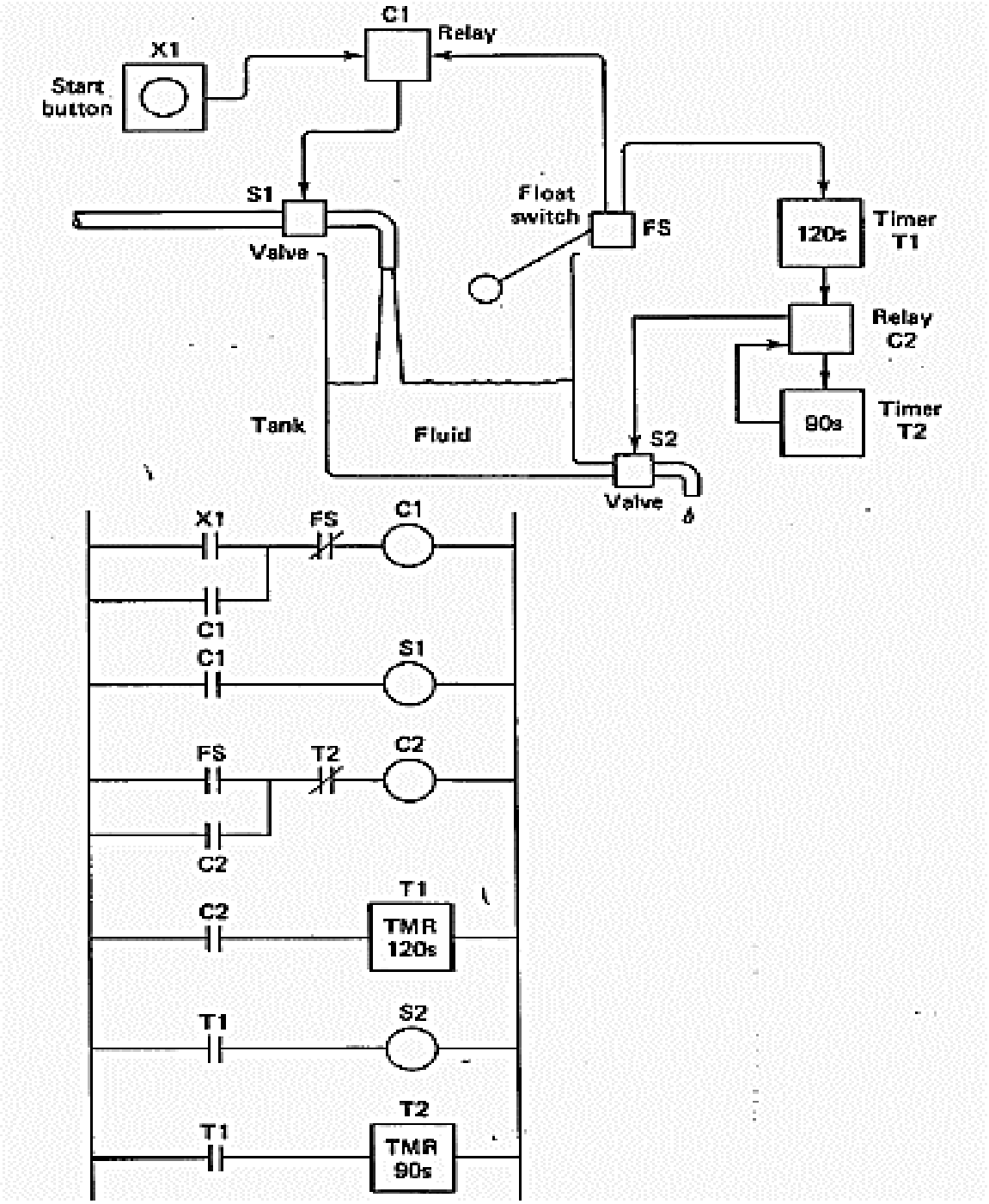
Sequence for traffic lights

As an illustration of programming involving timers consider the sequencing of traffic lights to give the sequence red only, red plus amber, green, amber, then repeat itself. A simple system might just have the sequence triggered by time, with each of the possible states occurring in sequence for a fixed amount of time. Figure 9.16 shows the sequential function chart and a possible ladder program to give the sequence.



Traffic light sequence

7. Automatic water filling in tank



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF MECHANICAL ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

COURSE MATERIAL

Program: B.E - MTRON Semester: VI
Course: PLC and Automation
Course code: SMR1303

UNIT 4 - PROGRAMMABLE LOGIC CONTROLLERS - SMR1303

Networking of PLCs-Data communication-Fieldbus, PROFI bus, and Mod bus-OSI Model types-OPC function. Supervisory Control and Data Acquisition-Architecture-Remote terminal unit-Master terminal unit-Data Storage.

UNIT 4 - PROGRAMMABLE LOGIC CONTROLLERS - SMR1303

Introduction to Networks & Data Communications

In Data Communications, data generally are defined as information that is stored in digital form. Data communications is the process of transferring digital information between two or more points. Information is defined as the knowledge or intelligence. Data communications can be summarized as the transmission, reception, and processing of digital information. For data communications to occur, the communicating devices must be part of a communication system made up of a combination of hardware (physical equipment) and software (programs). The effectiveness of a data communications system depends on four fundamental characteristics: delivery, accuracy, timelines, and jitter.

A data communications system has five components:

1. **Message:** The message is the information (data) to be communicated. Popular forms of information include text, numbers, pictures, audio, and video.
2. **Sender:** The sender is the device that sends the data message. It can be a computer, workstation, telephone handset, video camera, and so on.
3. **Receiver:** The receiver is the device that receives the message. It can be a computer, workstation, telephone handset, television, and so on.
4. **Transmission medium:** The transmission medium is the physical path by which a message travels from sender to receiver. Some examples of transmission media include twisted-pair wire, coaxial cable, fiber-optic cable, and radio waves.
5. **Protocol**: A protocol is a set of rules that govern data communications. It represents an agreement between the communicating devices.

The Internet:

The **Advanced Research Projects Agency Network** (**ARPANET**) was an early packet switching network and the first network to implement the protocol suite TCP/IP. Both technologies became the technical foundation of the Internet. The ARPANET was initially funded by the Advanced Research Projects Agency (ARPA) of the United States Department of Defense. The packet switching methodology employed in the ARPANET was based on concepts and designs by Americans Leonard Kleinrock and Paul Baran, British scientist Donald Davies, and Lawrence Roberts. The TCP/IP communications protocols were developed for the ARPANET by computer scientists Robert Kahn and Vint Cerf, and incorporated concepts from the French CYCLADES project directed by Louis Pouzin.

As the project progressed, protocols for internetworking were developed by which multiple separate networks could be joined into a network of networks. Access to the ARPANET was expanded in 1981 when the National Science Foundation (NSF) funded the Computer Science Network (CSNET). In 1982, the Internet protocol suite (TCP/IP) was introduced as the standard networking protocol on the ARPANET. In the early 1980s the NSF funded the establishment for national supercomputing centers at several universities, and provided interconnectivity in 1986 with the NSFNET project, which also created network access to the supercomputer sites in the United States from research and education organizations. The ARPANET was decommissioned in 1990.

PROTOCOLS & Standards

Protocol: A protocol is a set of rules that govern data communications. It represents an agreement between the communicating devices.

- Syntax
- Semantics
- Timing

An association of organizations, governments, manufacturers and users form the standards organizations and are responsible for developing, coordinating and maintaining the standards. The intent is that all data communications equipment manufacturers and users comply with these standards. **The primary standards organizations for data communication are:**

1. International Standard Organization (ISO)

ISO is the international organization for standardization on a wide range of subjects. It is comprised mainly of members from the standards committee of various governments throughout the world. It is even responsible for developing models which provides high level of system compatibility, quality enhancement, and improved productivity and reduced costs. The ISO is also responsible for endorsing and coordinating the work of the other standards organizations.

2. International Telecommunications Union-Telecommunication Sector (ITU-T)

ITU-T is one of the four permanent parts of the International Telecommunications Union based in Geneva, Switzerland. It has developed three sets of specifications: the V series for modem interfacing and data transmission over telephone lines, the X series for data transmission over public digital networks, email and directory services; the I and Q series

for Integrated Services Digital Network (ISDN) and its extension Broadband ISDN. ITU-T membership consists of government authorities and representatives from many countries and it is the present standards organization for the United Nations.

3. Institute of Electrical and Electronics Engineers (IEEE)

IEEE is an international professional organization founded in United States and is comprised of electronics, computer and communications engineers. It is currently the world's largest professional society with over 200,000 members. It develops communication and information processing standards with the underlying goal of advancing theory, creativity, and product quality in any field related to electrical engineering.

4. American National Standards Institute (ANSI)

ANSI is the official standards agency for the United States and is the U.S voting representative for the ISO. ANSI is a completely private, non-profit organization comprised of equipment manufacturers and users of data processing equipment and services. ANSI membership is comprised of people form professional societies, industry associations, governmental and regulatory bodies, and consumer goods.

5. Electronics Industry Association (EIA)

EIA is a non-profit U.S. trade association that establishes and recommends industrial standards. EIA activities include standards development, increasing public awareness, and lobbying and it is responsible for developing the RS (recommended standard) series of standards for data and communications.

Layered Tasks

To reduce the design complexity, most of the networks are organized as a series of **layers** or **levels**, each one build upon one below it. The basic idea of a layered architecture is to divide the design into small pieces. Each layer adds to the services provided by the lower layers in such a manner that the highest layer is provided a full set of services to manage communications and run the applications. The benefits of the layered models are modularity and clear interfaces, i.e. Open architecture and comparability between the different providers' components. A basic principle is to ensure independence of layers by defining services provided by each layer to the next higher layer without defining how the services are to be performed. This permits changes in a layer without affecting other layers. The basic elements of a layered model are services, protocols and interfaces. A **service** is a set of actions that a layer offers to another (higher) layer.

Protocol is a set of rules that a layer uses to exchange information with a peer entity. These rules concern both the contents and the order of the messages used. Between the layers service interfaces are defined. The messages from one layer to another are sent through those interfaces.

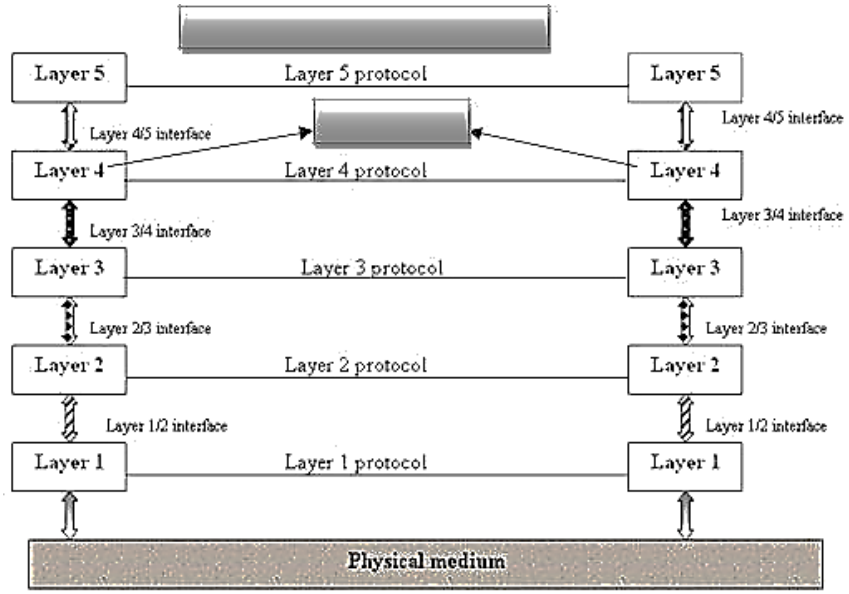


Figure 4.1: Layered Architecture

In a n-layer architecture, layer n on one machine carries on conversation with the layer n on other machine. The rules and conventions used in this conversation are collectively known as the layer-n protocol. Basically, a protocol is an agreement between the communicating parties on how communication is to proceed. Five-layer architecture is shown below**;** the entities comprising the corresponding layers on different machines are called **peers**. In other words, it is the peers that communicate using protocols. In reality, no data is transferred from layer n on one machine to layer n of another machine. Instead, each layer passes data and control information to the layer immediately below it, until the lowest layer is reached. Below layer-1 is the physical layer through which actual communication occurs.

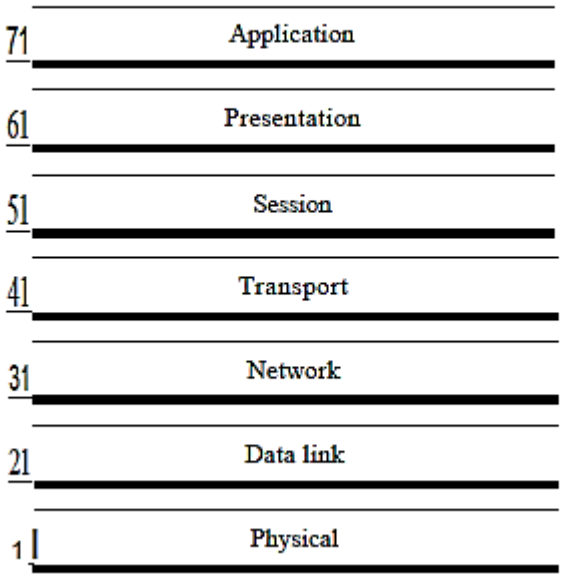
With layered architectures, communications between two corresponding layers requires a unit of data called a **protocol data unit (PDU**). A PDU can be a header added at the beginning of a message or a trailer appended to the end of a message. Data flows downward through the layers in the source system and upwards at the destination address. As data passes from one layer into another, headers and trailers are added and removed from the PDU. This process of adding or removing PDU information is called **encapsulation/decapsulation.** Between each pair of adjacent layers there is an **interface**.

The interface defines which primitives operations and services the lower layer offers to the upper layer adjacent to it. A set of layers and protocols is known as **network architecture**. A list of protocols used by a certain system, one protocol per layer, is called **protocol stack**.

OSI MODEL

The OSI model is based on a proposal developed by the International Standards Organization (ISO) as a first step toward international standardization of the protocols used in the various layers (Day and Zimmermann, 1983). It was revised in 1995(Day, 1995). The model is called the ISO-OSI (Open Systems Interconnection) Reference Model because it deals with connecting open systems—that is, systems that are open for communication with other systems. The OSI model is a layered framework for the design of network systems that allows communication between all types of computer systems. It consists of seven separate but related layers, each of which defines a part of the process of moving information across a network. Figure 4.2:

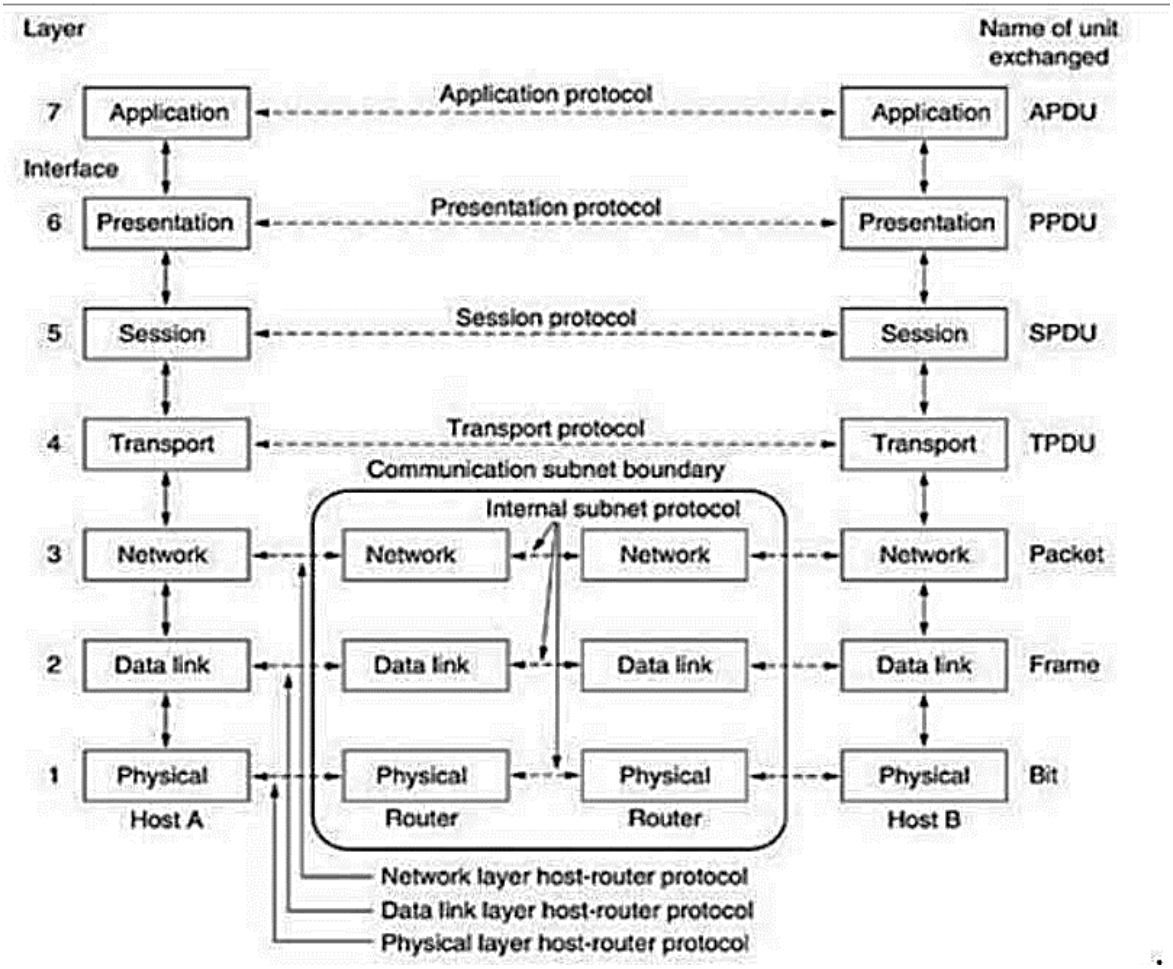
Seven layers of the OSI model



The OSI model is composed of seven ordered layers: physical (layer 1), data link (layer 2), network (layer 3), transport (layer 4), session (layer 5), presentation (layer 6), and application (layer 7). Figure below shows the layers involved when a message is sent from device A to device B. As the message travels from A to B, it may pass through many intermediate nodes. These intermediate nodes usually involve only the first three layers of the OSI model.

1. Physical Layer

The physical layer coordinates the functions required to carry a bit stream over a physical medium. It deals with the mechanical and electrical specifications of the interface and transmission medium. It also defines the procedures and functions that physical devices and interfaces have to perform for transmission to occur. Figure 4.3:



The physical layer is also concerned with the following:

o **Physical characteristics of interfaces and medium**. The physical layer defines the characteristics of the interface between the devices and the transmission medium. It also defines the type of transmission medium.

o **Representation of bits**. The physical layer data consists of a stream of bits (sequence of Os or 1s) with no interpretation. To be transmitted, bits must be encoded into signals--electrical or optical. The physical layer defines the type of encoding.

o **Data rate.** The transmission rate-the number of bits sent each second-is also defined by the physical layer. In other words, the physical layer defines the duration of a bit, which is how long it lasts.

o **Synchronization of bits**. The sender and receiver not only must use the same bit rate but

also must be synchronized at the bit level. In other words, the sender and the receiver clocks must be synchronized.

o **Line configuration**. The physical layer is concerned with the connection of devices to the media. In a point-to-point configuration, two devices are connected through a dedicated link. In a multipoint configuration, a link is shared among several devices.

o **Physical topology**. The physical topology defines how devices are connected to make a network. Devices can be connected by using a mesh topology (every device is connected to every other device), a star topology (devices are connected through a central device), a ring topology (each device is connected to the next, forming a ring), a bus topology (every device is on a common link), or a hybrid topology (this is a combination of two or more topologies).

o **Transmission mode**. The physical layer also defines the direction of transmission between two devices: simplex, half-duplex, or full-duplex. In simplex mode, only one device can send; the other can only receive. The simplex mode is a one-way communication. In the half-duplex mode, two devices can send and receive, but not at the same time. In a full-duplex (or simply duplex) mode, two devices can send and receive at the same time.

2. Data Link Layer

The data link layer transforms the physical layer, a raw transmission facility, to a reliable link. It makes the physical layer appear error-free to the upper layer (network layer). Other responsibilities of the data link layer include the following:

o **Framing**. The data link layer divides the stream of bits received from the network layer into manageable data units called frames.

o **Physical addressing**. If frames are to be distributed to different systems on the network, the data link layer adds a header to the frame to define the sender and/or receiver of the frame. If the frame is intended for a system outside the sender's network, the receiver address is the address of the device that connects the network to the next one.

o **Flow control**. If the rate at which the data are absorbed by the receiver is less than the rate at which data are produced in the sender, the data link layer imposes a flow control mechanism to avoid overwhelming the receiver.

o **Error control**. The data link layer adds reliability to the physical layer by adding mechanisms to detect and retransmit damaged or lost frames. It also uses a mechanism to recognize duplicate frames. Error control is normally achieved through a trailer added to the end of the frame.

o **Access control**. When two or more devices are connected to the same link, data link layer protocols are necessary to determine which device has control over the link at any given time.

3. Network Layer

The network layer is responsible for the source-to-destination delivery of a packet, possibly across multiple networks (links). Whereas the data link layer oversees the delivery of the packet between two systems on the same network (links), the network layer ensures that each packet gets from its point of origin to its final destination. If two systems are connected to the same link, there is usually no need for a network layer. However, if the two systems are attached to different networks (links) with connecting devices between the networks (links), there is often a need for the network layer to accomplish source-to-destination delivery.

Other responsibilities of the network layer include the following:

o **Logical addressing**. The physical addressing implemented by the data link layer handles the addressing problem locally. If a packet passes the network boundary, we need another addressing system to help distinguish the source and destination systems. The network layer adds a header to the packet coming from the upper layer that, among other things, includes the logical addresses of the sender and receiver.

o **Routing**. When independent networks or links are connected to create internet works (network of networks) or a large network, the connecting devices (called routers or switches) route or switch the packets to their final destination. One of the functions of the network layer is to provide this mechanism.

4. Transport Layer

The transport layer is responsible for process-to-process delivery of the entire message. A process is an application program running on a host. Whereas the network layer oversees source- to-destination delivery of individual packets, it does not recognize any relationship between those packets. It treats each one independently, as though each piece belonged to a separate message, whether or not it does. The transport layer, on the other hand, ensures that the whole message arrives intact and in order, overseeing both error control and flow control at the source- to-destination level.

Other responsibilities of the transport layer include the following:

o **Service-point addressing**. Computers often run several programs at the same time. For this reason, source-to-destination delivery means delivery not only from one computer to the next but also from a specific process (running program) on one computer to a specific process (running program) on the other. The transport layer header must therefore include a type of address called a service-point address (or port address). The network layer gets each packet to the correct computer; the transport layer gets the entire message to the correct process on that computer.

o **Segmentation and reassembly**. A message is divided into transmittable segments, with each segment containing a sequence number. These numbers enable the transport layer to reassemble the message correctly upon arriving at the destination and to identify and replace packets that were lost in transmission.

o **Connection control**. The transport layer can be either connectionless or connection oriented. A connectionless transport layer treats each segment as an independent packet and delivers it to the transport layer at the destination machine. A connection oriented transport layer makes a connection with the transport layer at the destination machine first before delivering the packets. After all the data are transferred, the connection is terminated.

o **Flow control**. Like the data link layer, the transport layer is responsible for flow control. However, flow control at this layer is performed end to end rather than across a single link.

o **Error control.** Like the data link layer, the transport layer is responsible for error control. However, error control at this layer is performed process-to-process rather than across a single link. The sending transport layer makes sure that the entire message arrives at the receiving transport layer without error (damage, loss, or duplication). Error correction is usually achieved through retransmission.

5. Session Layer

The services provided by the first three layers (physical, data link, and network) are not sufficient for some processes. The session layer is the network dialog controller. It establishes, maintains, and synchronizes the interaction among communicating systems.

Specific responsibilities of the session layer include the following:

o **Dialog control.** The session layer allows two systems to enter into a dialog. It allows the communication between two processes to take place in either half duplex (one way at a time) or full-duplex (two ways at a time) mode.

o **Synchronization**. The session layer allows a process to add checkpoints, or synchronization points, to a stream of data.

6. Presentation Layer

The presentation layer is concerned with the syntax and semantics of the information exchanged between two systems. **Specific responsibilities of the presentation layer include the following:**

o **Translation**. The processes (running programs) in two systems are usually exchanging information in the form of character strings, numbers, and so on. The information must be changed to bit streams before being transmitted. Because different computers use different encoding systems, the presentation layer is responsible for interoperability between these different encoding methods. The presentation layer at the sender changes the information from its sender-dependent format into a common format. The presentation layer at the receiving machine changes the common format into its receiver-dependent format.

o **Encryption**. To carry sensitive information, a system must be able to ensure privacy. Encryption means that the sender transforms the original information to another form and sends the resulting message out over the network. Decryption reverses the original process to transform the message back to its original form.

o **Compression**. Data compression reduces the number of bits contained in the information. Data compression becomes particularly important in the transmission of multimedia such as text, audio, and video.

7. Application Layer

The application layer enables the user, whether human or software, to access the network. It provides user interfaces and support for services such as electronic mail, remote file access and transfer, shared database management, and other types of distributed information services. Specific services provided by the application layer include the following: o **Network virtual terminal**. A network virtual terminal is a software version of a physical terminal, and it allows a user to log on to a remote host.

o **File transfer, access, and management**. This application allows a user to access files in a remote host (to make changes or read data), to retrieve files from a remote computer for use in the local computer, and to manage or control files in a remote computer locally.

o **Mail services**. This application provides the basis for e-mail forwarding and storage.

o **Directory services**. This application provides distributed database sources and access for global information about various objects and services.

Field Bus

Introduction to Fieldbus Systems

Fieldbus is consisting of two terms, Field and Bus [Fieldbus Introduction]. To start, the meaning of Field, as defined in industrial world, is a geographical or contextual limited area. From the industry point of view the Field is an abstraction of the plant levels. As for the term Bus is a well-known word in computer science as a set of common line that electrically (or even optically) connects various units (circuits) in order to transfer the data among them.

The origin of the fieldbus was to replace any point-to-point links between the field devices (Field Devices are simply the Sensors and Actuators of the plant) and their controllers (PLC's) by a digital single link on which all the information is transmitted serially and multiplexed in time. We will see that the fieldbus transfers, in most cases, this information in small-sized packets in serial manner. Choosing the serial transmission has many merits in comparison with other kinds of transmission like parallel transmission. For instance, the sequential or serial transmission reduces the total required number of the connecting lines over greater distances than that of the point-to-point or even parallel transmissions.

A set of rules must be defined in order to accomplish data transfer between the units along the bus. This set of rules is called Communication Protocol or just the Protocol. This is unlike the case of the ordinary point-to-point transmission where any two connected entities send and receive data from each other whenever the data is available. The protocol is responsible for two important rules on the bus, the mechanism that any unit can acquire or size the bus, and the synchronization between those multi-units on the bus.

Concept of Field bus technology and layers: The seven layers of the OSI model are

1- **Application layer** which provides the services that are required by specific applications.
2- **Presentation layer** is responsible for the data interpretation, which allows interoperability among different equipments.
3- **Session layer** is concerned with execution of any remote actions.
4- **Transport layer** is responsible for the end-to-end communication control.

5- **Network layer** is concerned with logical addressing process of nodes and routing schemes.

6- **Data link layer** is responsible for the access to the communication medium, and for the logical transfer of the data.

7- **Physical layer** is concerned with the way that the communication is done.

Any communication system that is based on the OSI seven layers will provide higher flexibility and compatibility with products from different vendors. Modification to the **MAP** project was necessary as the node implementation become more complex in order to support all the services of the OSI reference model. The modification allowed the short length control data packets, which occurs at high rates, to be directly transmitted through the application layer to the data link layer. The resulting field bus is referred to as a 3-layered Architecture. These layers are: the Application layer, the Data link layer, the Physical layer.

One may assume that the other four layers of the OSI model that are not available in the fieldbus hierarchy have disappeared along with their own functions and services. The main function of the presentation layer is to support the interoperability between different equipments which is currently done by the application layer in the field bus. The assembling and disassembling of data packets which was the function of the transport layer is done now by the data link layer in the fieldbus network. If routers to be used in some fieldbus networks, then the routing service, which was assigned to the network layer, is done by the application layer in most cases in the fieldbus.

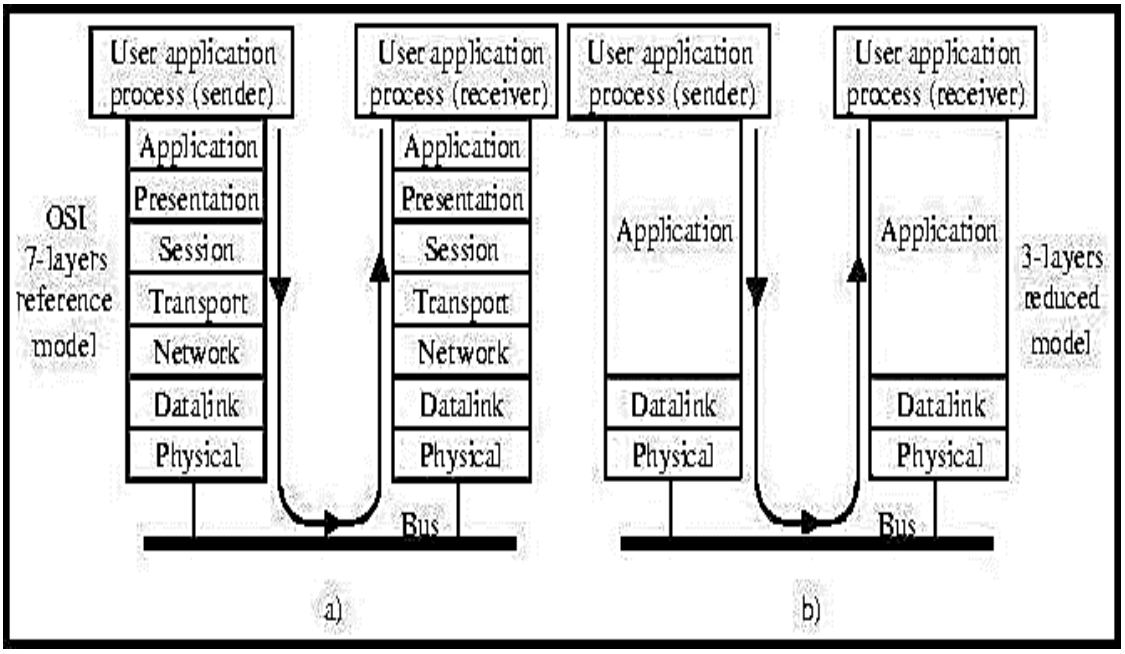


Figure 4.4: (a) The OSI 7-layers reference model, and (b) The reduced fieldbus 3-layer structure

There are several protocols and services that are laid in the 3-layerd hierarchy of the fieldbus network. This will lead to a great difficulty in evaluating one and unique international fieldbus standard. There are several fieldbus protocols in the world. The designer of the DCCS communications system has multi -option solutions to fulfill his system requirements. These requirements are varied from one situation to another. In many situations the quality of services and the system throughput are the common requirements in all the DCCS systems. Also a fast response time is usually required by the real-time computer controlled networks designers.

Different Types of Fieldbuses

The fieldbus technology started at the early beginnings of the 80's in many countries. For example, the Factory Instrumentation Protocol (FIP), the Controller Area Network (CAN), and the Process Field Bus (PROFIBUS), the Process Network (P-NET) protocol, are all appeared in the same time period at the beginning of the 1980's.

CERN Recommendations on Fieldbuses

CERN is the European Organization that was established in 1954 to mainly serve the research of physics. But with the time passed the CERN became more interested in other aspects that may have some relationship with the physics. For example, the CERN involved in computer networking as the World Wide Web (**WWW**) was first developed at CERN in the beginnings of the 1990's.Now-a-days CERN is engaged in a large scale project of LHC experiments where fieldbuses is used to control and monitor the equipments of these LHC experiments. The working group finally found out that recommending one fieldbus may not satisfy all the users, so instead they had recommended three Fieldbuses. These are:

1- Controller Area Network (**CAN**).

2- Process Field Bus (**PROFIBUS**).

3- World Factory Instrumentation Protocol (**WorldFIP**).

Controller Area Network (CAN)

The **CAN** protocol is a priority-based bus network using a Carrier Sense Multiple Access with Collision Avoidance (**CSMA/CA**) medium access scheme. In this protocol any station can access the bus whenever it becomes idle. The collision resolution mechanism is very simple and is supported by the frame structure, or more specifically by its twelve leading bits, denoted as start bit and identifier fields. This identifier field serves for two different

purposes. On one hand it is used to identify any message stream in a CAN network. Take for example a sequence of messages concerning the remote reading of a specific process variable. On the other hand, it is a priority field, which enables the collision resolution mechanism to schedule the contending messages.

This collision resolution mechanism in CAN works as follows: when the bus becomes idle, every station with pending messages will start to transmit. Due to its open-collector nature, the CAN bus acts as a big AND-gate, where each station is able to read the bus status. During the transmission of the identifier field, if a station is transmitting a "**1**" and reads a "**0**", it means that there was a collision with at least one higher-priority message, and this station aborts the message transmission at once. Thus the highest-priority message being transmitted will proceed without encountering any collision, and thus will be successfully transmitted to its destination. Obviously, each message stream must be uniquely identified. The collision resolution mechanism that is used by the CAN protocol imposes certain limitations to the bus length and its transmission data rate. For example, considering a bus length of 40m, the maximum data rate is 1Mbps.

Process Field Bus (PROFIBUS)

The ProfiBus communication model used to combine the distributed application processes into a common process, using communications relationships. All objects of a real device that can be communicated, such as variables, programs, data ranges are called communication objects. The process in a field device that is reachable for communication is called a virtual field device (VFD). The VFD contains the communication objects that may be manipulated by the services of the application layer.

The PROFIBUS protocol is based on a token passing procedure used by master stations to grant the bus access to each other, and a master-slave procedure used by master stations to communicate with slave stations. The PROFIBUS token passing procedure uses a simplified version of the Timed-token protocol. The medium access functions which are implemented at the layer-2 of the OSI reference model, is called Fieldbus Data Link (FDL). In addition to controlling the bus access and the token cycle time, the FDL is also responsible for the provision of data transmission services for the application layer. PROFIBUS supports four data transmission services which are: Send Data with No-Acknowledge (SDN); Send Data with Acknowledge (SDA); Request Data with Reply (RDR) and Send and Request Data (SRD). The SDN is an unacknowledged service used for broadcasts from a master station to

all other stations on the bus. An important characteristic of other services is that they are immediately answered, with a response or an acknowledgement. This feature, also called "immediate-response", is particularly important for the real-time bus operation.

In addition to these services, industrial applications sometimes require the use of periodical transmission methods. A FDL-controlled polling method may be used to scan field devices, such as sensors or actuators.

World Factory Instrumentation Protocol (WorldFIP)

A WorldFIP network interconnects stations with two types; either a bus arbitration station or production/consumption stations. At any given time instant, only one station can be the bus arbitration station. Hence, in WorldFIP, the medium access control is centralized, and performed by the active bus arbitrator which is known as the BA. WorldFIP supports two basic types of transmission services: exchange of identified variables and exchange of messages. In WorldFIP, the exchange of identified variables is based on a Producer/Consumer model, which relates producers and consumers within a distributed system. In order to manage any transactions associated with a single variable, a unique identifier is associated with each variable. The WorldFIP Data Link Layer (DLL) is made up of a set of produced and consumed buffers, which can be locally accessed or remotely accessed through network services. In WorldFIP networks, the bus arbitrator table (BAT) regulates the scheduling of all buffer transfers. There are two types of buffer transfers that can be considered in WorldFIP. These are: periodic and aperiodic. The BAT imposes the schedule of the periodic buffer transfers, and also regulates the aperiodic buffer transfers in the times that there are no periodic buffer transactions.

Industrial application of field bus

In compliance with hazardous area requirements, such as low power and intrinsic safety, the use of the RS485 interface with its high transmission rates is also possible in and Ex zone. This intrinsically safe bus is obtained by using an RS485 IS Coupler. The SIEMENS Fieldbus isolating transformer provides this functionality, requires no programming, and has a repeater to amplify the signal data on the bus line. Figure 1 shows the various topologies possible with RS485 IS network.

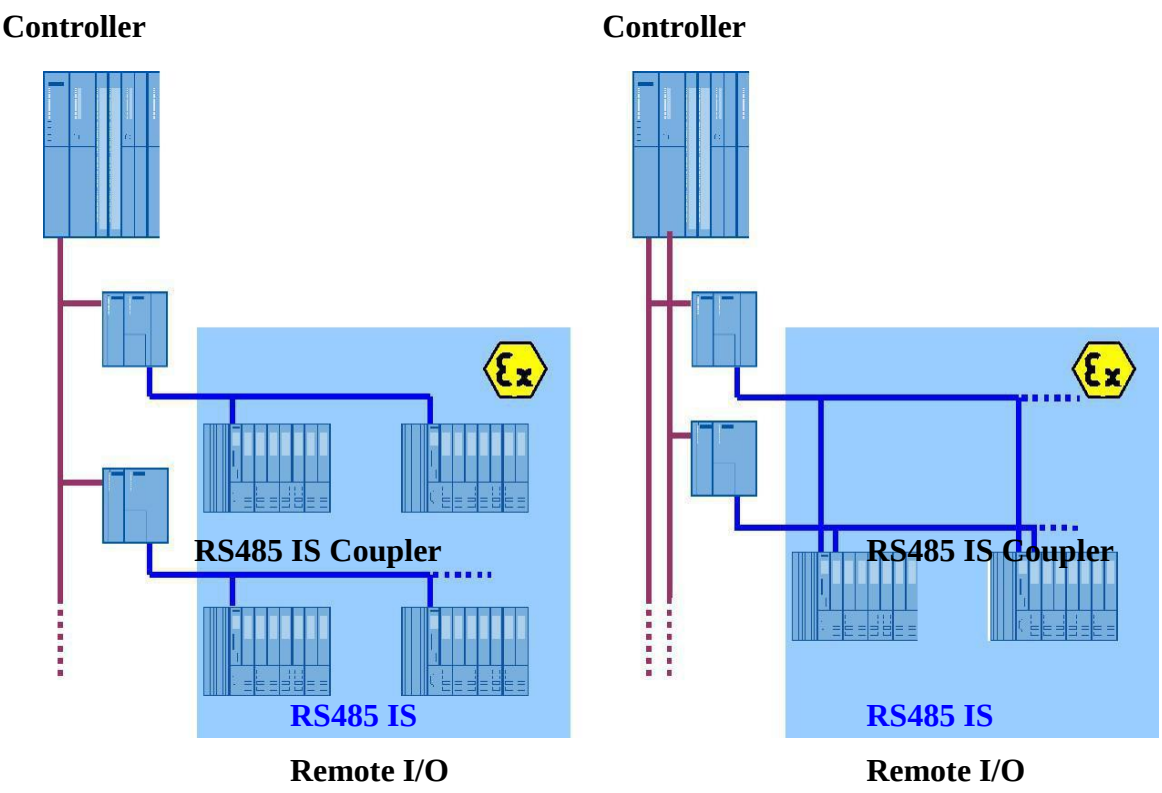


Figure 4.5: RS485 IS Topologies

In the MBP-IS version, this transmission technology is especially suitable for use in hazardous areas in Process Automation, and is therefore widely used in applications of the chemical, oil and gas industries. Explosion protection is implemented via limiting power in the incoming bus supply or more frequently in the installation components in the field. Working on field devices during active operation is made possible, for example, by means of intrinsically safe ignition protection. Linear and tree structures and combinations of both are thus possible. In practice, the "trunk & spur topology" (Fig) has established itself as the de-facto standard, as it is especially clear and well-laid-out.

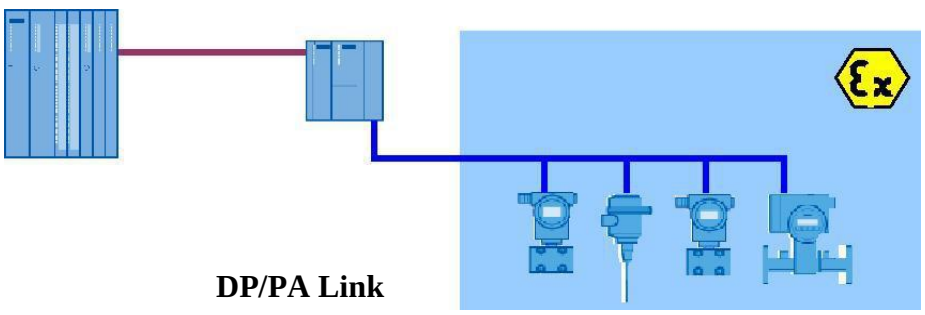


Figure 4.6: Single Line Topology

REDUNDANT MASTER and DP TOPOLOGY

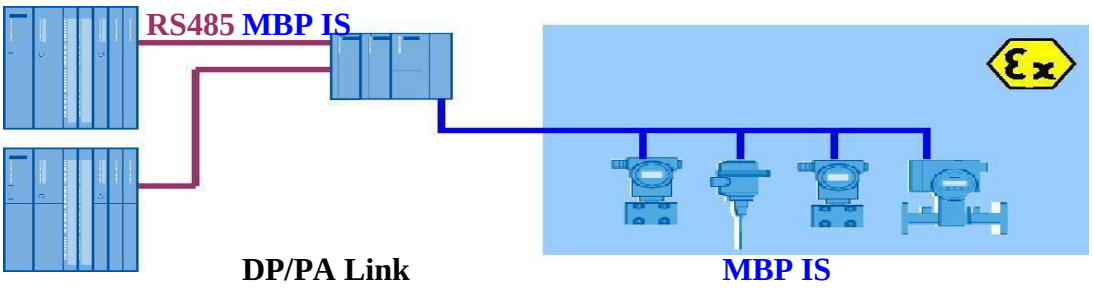


Figure 4.7: Integration of MBP-IS with PROFIBUS

PROFIBUS PA also offers ring topology to provide redundancy mechanism on the MBP bus. The combination of redundant couplers and field devices with active field distributors implements ring redundancy and creates expanded media redundancy. Sub segments which have become effective due to a short circuit or wire break are automatically and seamlessly operated further via a coupler each in a line structure.

RING TOPOLOGY on PA

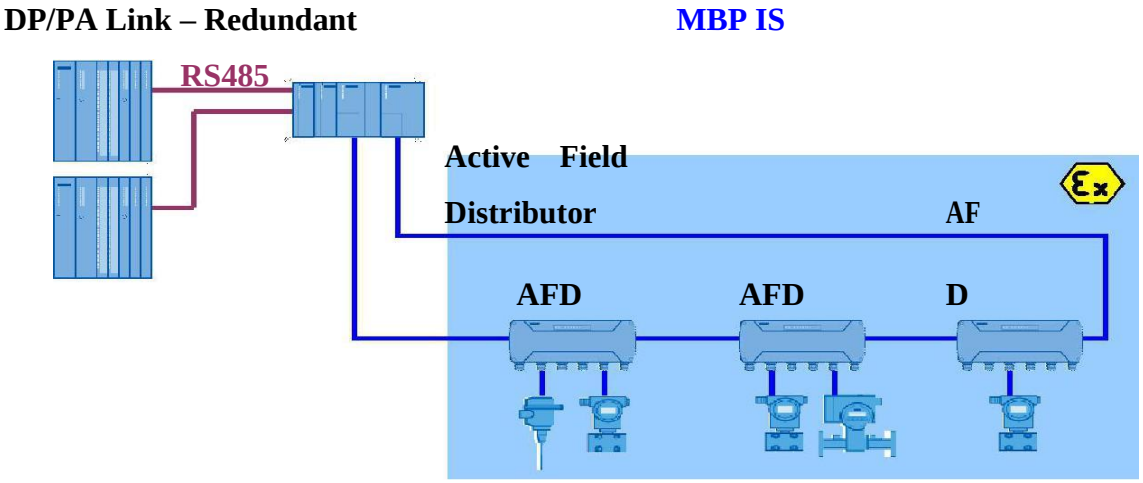


Figure 4.8: PA Ring Topology

Supervisory control

SCADA is an acronym for Supervisory Control and Data Acquisition. SCADA systems are used to monitor and control a plant or equipment in industries such as telecommunications, water and waste control, energy, oil and gas refining and transportation. These systems encompass the transfer of data between a SCADA central host computer and a number of Remote Terminal Units (RTUs) and/or Programmable Logic Controllers (PLCs), and the central host and the operator terminals.

Components of SCADA

1. Human Machine Interface (HMI): It is an interface which presents process data to a human operator, and through this, the human operator monitors and controls the process.

2. Supervisory (computer) system: It gathers data on the process and sending commands *(*or control*)* to the process.

3. Remote Terminal Units (RTUs): It connects to sensors in the process, converting sensor signals to digital data and sending digital data to the supervisory system.

4. Programmable Logic Controller (PLCs): It is used as field devices because they are more economical, versatile, flexible, and configurable than special-purpose RTUs.

5. Communication infrastructure: It provides connectivity to the supervisory system to the Remote Terminal Units.

Architecture of SCADA

The block diagram of SCADA system shown in the figure represents the basic SCADA architecture. The SCADA (supervisory control and data acquisition) systems are different from distributed control systems that are commonly found in plant sites. When distributed control systems cover the plant site, SCADA system cover much larger geographic areas.

The below figure depicts an integrated SCADA architecture which supports TCP/IP, UDP and other IP based communication protocols as well as industrial protocols like Modbus TCP, Modbus over TCP or Modbus over UDP. These all work over cellular, private radio or satellite networks.

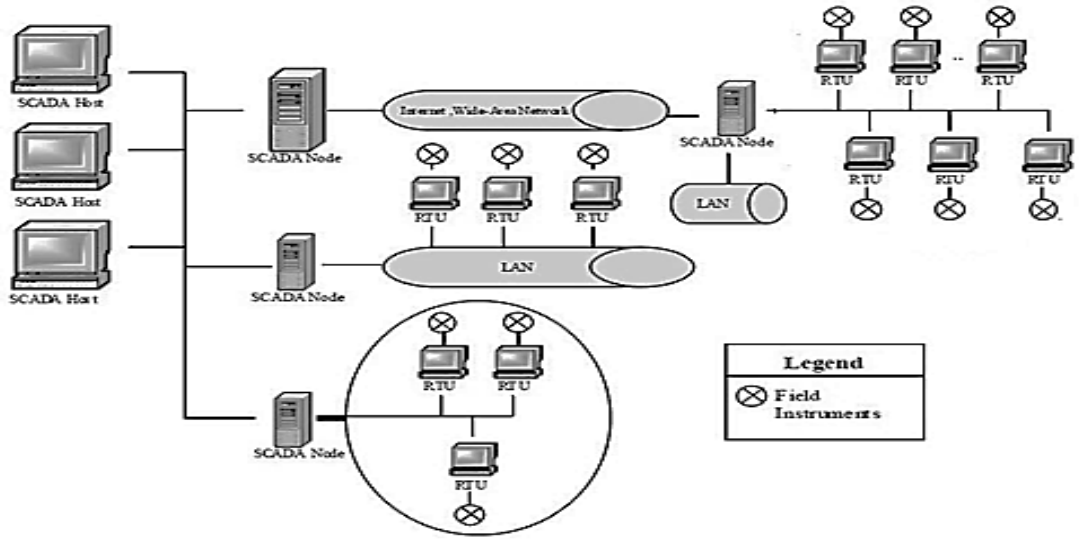


Figure 4.9: Architecture of SCADA

In complex SCADA architectures, there are a variety of wired and wireless media & protocols involved in getting data back to the monitoring site. This allows implementation of powerful IP based SCADA networks over landline, mixed cellular and satellite systems. SCADA communications can utilize a diverse range of wired and wireless media.

The choice of the existing communication depends on the characterization of a number of factors. The factors are remoteness, available communications at the remote sites, existing communications infrastructure, polling frequency and data rates. These factors impact the final decision for SCADA architecture. Therefore, a review of SCADA systems evolution allows us to better understand many security concerns.

Types of SCADA systems

1. First Generation: Monolithic or Early SCADA systems
2. Second Generation: Distributed SCADA systems
3. Third Generation: Networked SCADA systems
4. Fourth Generation: Internet of things technology, SCADA systems

Monolithic or Early SCADA Systems

Minicomputers are used earlier for computing the SCADA systems. In earlier times, during the time of first generation, monolithic SCADA systems were developed wherein the common network services were not available. Hence, these are independent systems without having any connectivity to other systems.

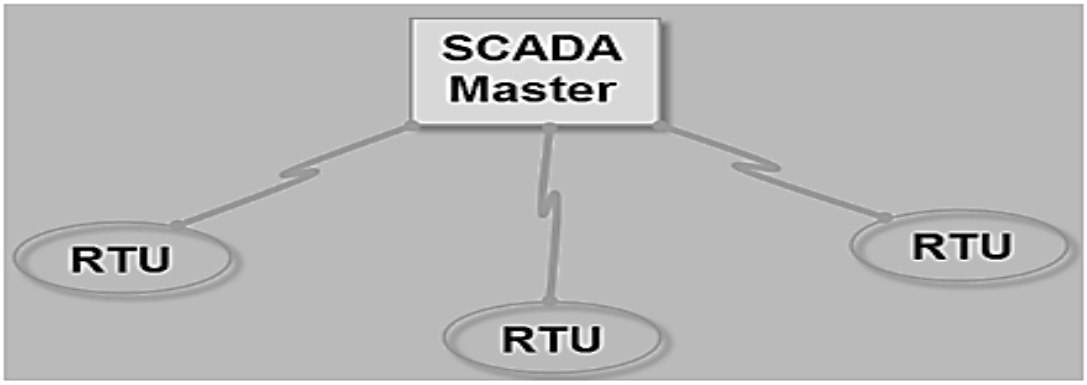


Figure 4.10: Monolithic or Early SCADA Systems

Distributed SCADA Systems

In the second generation, the sharing of control functions is distributed across the multiple systems connected to each other using Local Area Network (LAN). Hence, these were termed as distributed SCADA systems. These individual stations were used to share real-time information and command processing for performing control tasks to trip the alarm levels of possible problems.

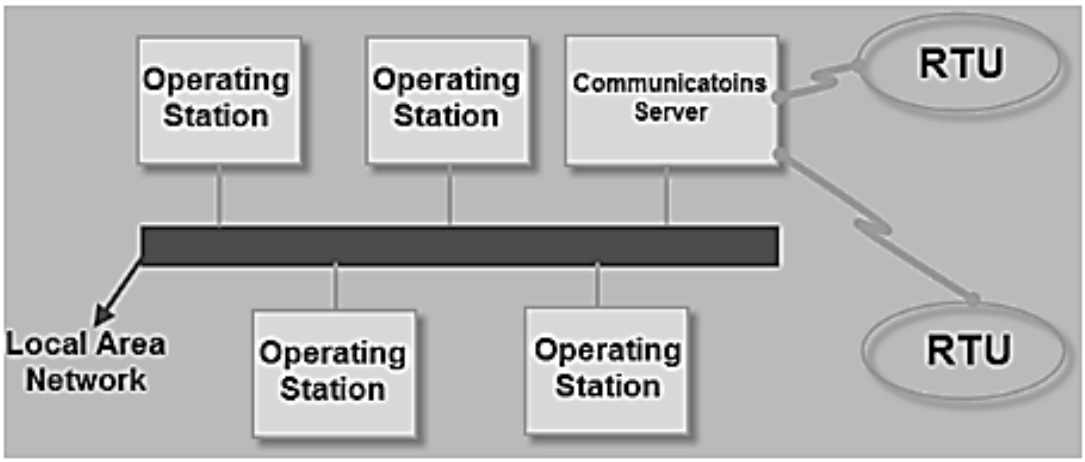


Figure 4.11: Distributed SCADA Systems

The cost and size of the station were reduced compared to the first generation system, as each system of the second generation was responsible for performing a particular task with reduced size and cost.

Networked SCADA Systems

The current SCADA systems are generally networked and communicate using Wide Area Network (WAN) Systems over data lines or phone. These systems use Ethernet or Fiber Optic Connections for transmitting data between the nodes frequently. These third generation SCADA systems use Programmable Logic Controllers (PLC) for monitoring and adjusting the routine flagging operators only in case of major decisions requirement.

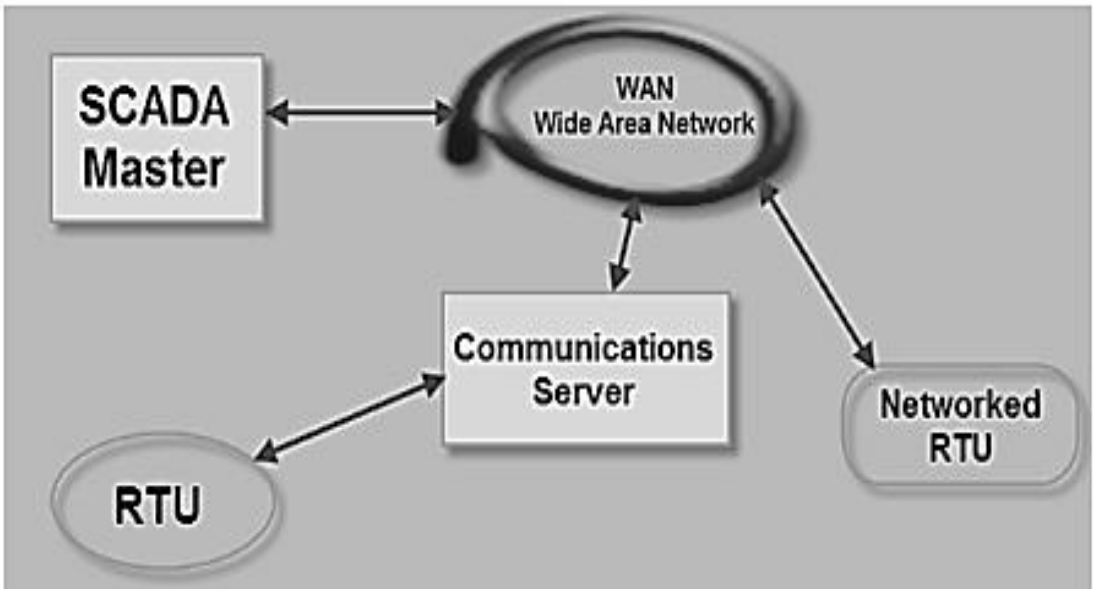


Figure 4.12: Networked SCADA Systems

Internet of Things

In fourth generation, the infrastructure cost of the SCADA systems is reduced by adopting the internet of things technology with the commercially available cloud computing. The maintenance and integration is also very easy for the fourth generation compared to the earlier SCADA systems. These SCADA systems are able to report state in real time by using the horizontal scale from the cloud computing facility; thus, more complex control algorithms can be implemented which are practically sufficient to implement on traditional PLC.

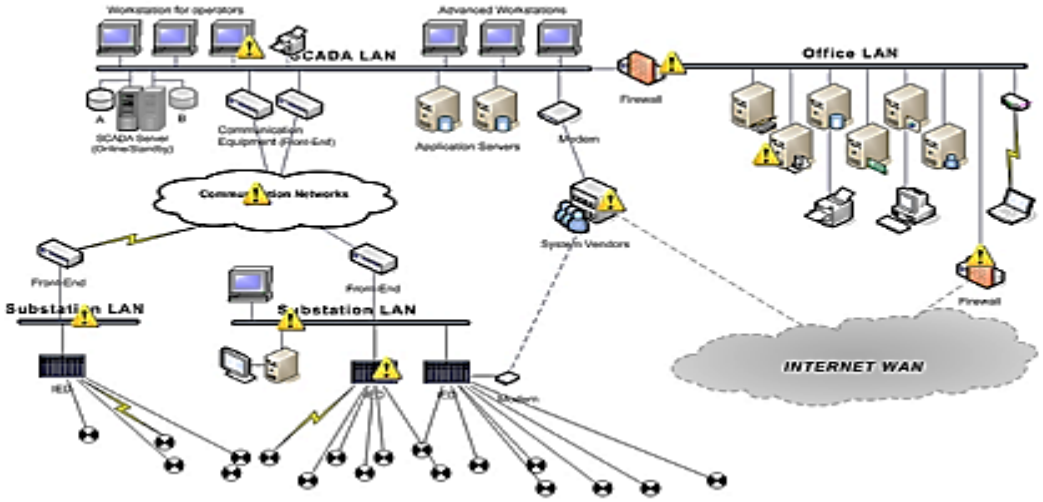


Figure 4.13: Internet of Things

Applications of SCADA

- SCADA systems are used for monitoring a variety of data like flows, currents, voltages, pressures, temperatures, water levels, and etc.,
- Manufacturing Industries
- Waste Water Treatment and Distribution Plants
- SCADA in Power System

Scheduling control

- Scheduling refers to a set of policies and mechanisms to control the order of work to be performed by a computer system
- Of all the resources in a computer system that are scheduled before use, the CPU is by far the most important

- Multiprogramming is the (efficient) scheduling of the CPU
- The basic idea is to keep the CPU busy as much as possible by executing a (user) process until it must wait for an event, and then switch to another process

Scheduler

In multiprogramming systems, when there is more than one run able process (i.e., ready), the operating system must decide which one to activate. The decision is made by the part of the operating system called the scheduler, using a scheduling algorithm.

In the beginning—there was no need for scheduling, since the users of computers lined up in front of the computer room or gave their job to an operator.

Batch processing—the jobs were executed in first come first served manner.

Multiprogramming—life became complicated! The scheduler is concerned with deciding policy, not providing a mechanism.

Types of scheduling

- Short-term scheduling
- Medium-term scheduling
- Long-term scheduling

Short-term scheduling

Short-term scheduler, also known as the process or CPU scheduler, controls the CPU sharing among the ‘‘ready’’ processes. The selection of a process to execute next is done by the short-term scheduler. Usually, a new process is selected under the following circumstances:

• When a process must wait for an event.

• When an event occurs (e.g., I/O completed, quantum expired).

• When a process terminates.

Medium-term scheduling

Medium-term scheduling involves suspending or resuming processes by swapping (rolling) them out of or into memory

Long-term scheduling

Long term scheduling is done when a new process is created. It initiates processes and so controls the degree of multi-programming (number of processes in memory).

Simple policies for long term scheduling are:

- Simple First Come First Served (FCFS): it's essentially a FIFO scheme. All job requests (e.g. a submission of a batch program, or an user trying to log in in a time shared system) are honoured up to a fixed system load limit, further requests being refused tout court, or enquired for later processing.
- Priority schemes. Note that in the context of long term scheduling ``priority" has a different meaning than in dispatching: here it affects the choice of a program to be entered the system as a process, there the choice of which ready process should be executed.

Digital control interfacing - process inputs, outputs interface

An interface is the common point at which two or more systems communicate with each other. Digital I/O interfaces are commonly used in PC based DAQ systems to provide monitoring and control for industrial processes, generate patterns for testing in the laboratory and communicate with peripheral equipment such as data loggers and printers which have parallel digital I/O capabilities. It is clear that these types of digital, or discrete, inputs and outputs (I/O) are much easier for microprocessor-based data acquisition systems to deal with than analog signals as the computer has a fully binary environment. Similar to analog-to-digital converters used for analog I/O, digital I/O is designed to deal directly with Transistor-to-Transistor Logic (TTL) level voltage changes. TTL typically sets the low-voltage level between 0 and 0.8 V and the high-voltage level between 2.0 and 5.0 V. Voltage levels between 0.8 and 2.0 V are not allowed. A voltage change, then, from the high range to the low range (or vice versa) represents a digital change of state from high to low, on to off, etc. and because acquiring an analog signal is more complex than acquiring a digital one, analog I/O channels also are more expensive.

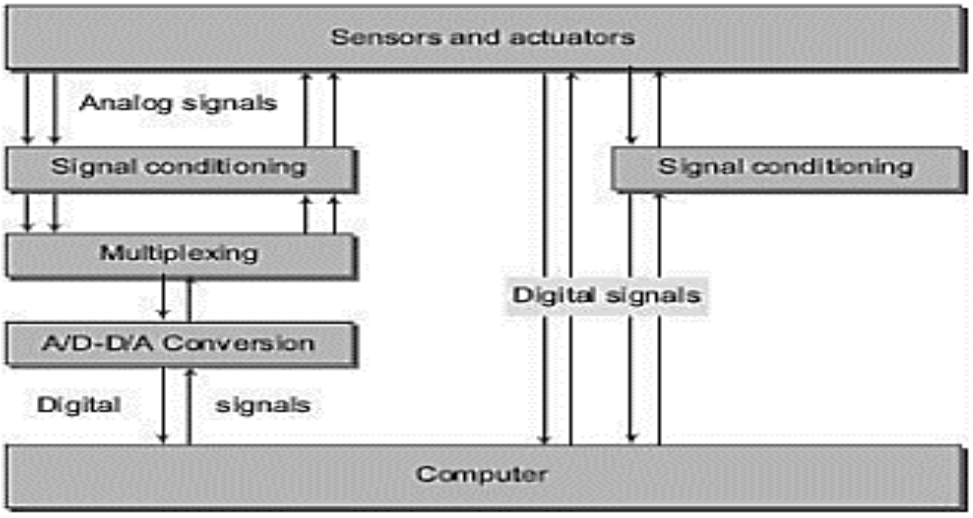


Figure 4.14: Digital I/O cards

Digital Inputs

Many types of digital input signals from switch closures relay contacts, or TTL compatible interfaces can be read directly by digital I/O cards (Figure 1.5). Other types of inputs may require some signal-conditioning, most likely to reduce higher level voltage changes to TTL levels. A variety of signal-conditioning modules are available to provide isolation and other digital-conditioning functions. The most common type of digital input is the contact closure (Figure 1.5.1). Essentially, a sensor or switch of some type closes or opens a set of contacts in accordance with some process change. An applied electrical signal then determines whether the circuit is open or closed. Current flows if the circuit is closed, registering a "1" in a transistor at the computer interface. Conversely, an open circuit retains a high voltage (and no current), registering a "0" at the transistor.

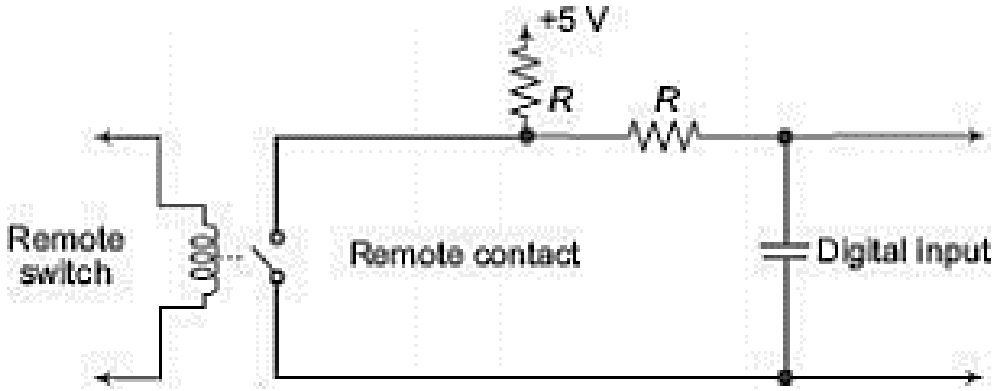


Figure 4.15: Contact type digital input

Digital Outputs

At its simplest, a digital output provides a means of turning something on or off. Applications range from driving a relay to turning on an indicator lamp to transmitting data to another computer. For latching outputs, a "1" typically causes the associated switch or relay to latch, while a "0" causes the switch to unlatch. Devices can be turned on or off, depending on whether the external contacts are normally open or normally closed. Standard TTL level signals can be used to drive 5 V relay coils; a protective diode is used to protect the digital output circuitry (Figure 1.4.3). Because data acquisition boards can typically supply only 24 mA of driving current, they are intended primarily to drive other logic circuits, not final control elements. Scaling may be needed so that logical voltage levels are sufficient to cause switching in larger relays. Outputs intended to drive larger solenoids, contactors, motors, or alarms also may require a boost.

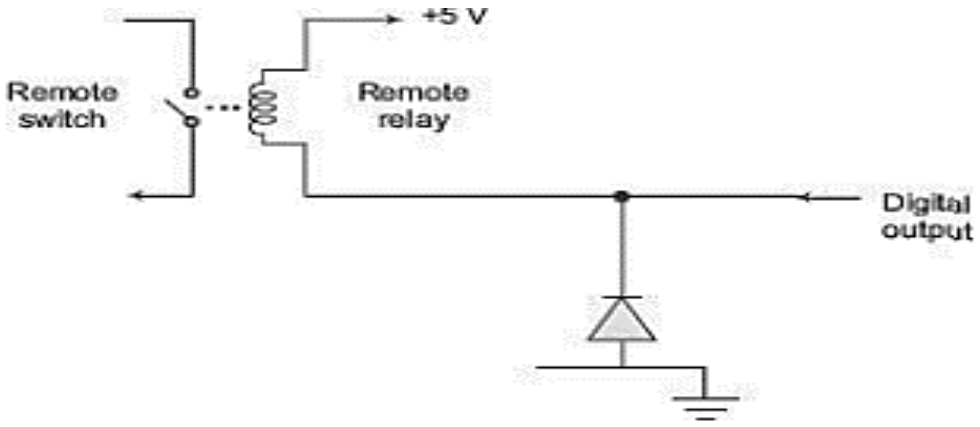


Figure 4.16: Contact type digital input

Counter/Timer and Pulse I/O

A somewhat separate class of digital I/O is pulse inputs and outputs, which typically is associated with frequency, counting or totalisation applications. Pulse inputs might be used to count the rotations of a turbine flow meter; pulse outputs might be used to drive a stepping motor. Pulse inputs are handled in much the same way as digital logic inputs, but the output of the sensing circuit is normally connected to a counter rather than a specific bit position in the input register. Successive pulses increment or decrement the counter. Add an elapsed time measure and a frequency or pulse rate can readily be determined. Similar to an analog-to-digital converter, a counter is characterized by its number of bits—an N-bit counter can accumulate up to 2N discrete events. Thus, a 16-bit counter can count to 2 16 = 65,536.

Data Logger

Data loggers are is stand-alone devices that can record information electronically from internal or external sensors or other equipment that provide digital or serial outputs.

Features

(a**) Stand-alone Operation**: Most data loggers are normally configured with a PC, some models can be configured from the front panel provided by the manufacturer. Once the data loggers are configured, they don't need the PC to operate.

(b) **Support for Multiple Sensor Types**: Data loggers often have universal input type which can accept input from common sensors like thermocouple, RTD, humidity, voltage, etc.

(c) **Local Data Storage**: All data loggers have local data storage or internal memory unit, so all the measured data is stored within the logger for later transfer to a PC.

(d) **Automatic Data Collection**: Data loggers are designed to collect data at regular intervals, 24 hours a day and 365 days a year if necessary, and the collection mode is often configurable.

Data logging and recording are both analog terms in the field of measurement. Data logging is basically measuring and recording of any physical phenomena or electrical parameter over a period of time. The physical phenomena can be temperature, strain, displacement, flow, pressure, voltage, current, resistance, power, and many other parameters

The data logger collects information about the state of any physical system from the sensors. Then the data logger converts this signal into a digital form with the help of an A/D converter. This digital signal is then stored in some electronic storage unit, which can be easily transferred to the computer for further the analysis.

Components of data logger

1. Hardware components like sensors signal conditioning, and analog-to-digital converter.
2. Long-term data storage, typically onboard memory or a PC
3. Software for collecting data, analyzing and viewing

Industrial Data logging and display

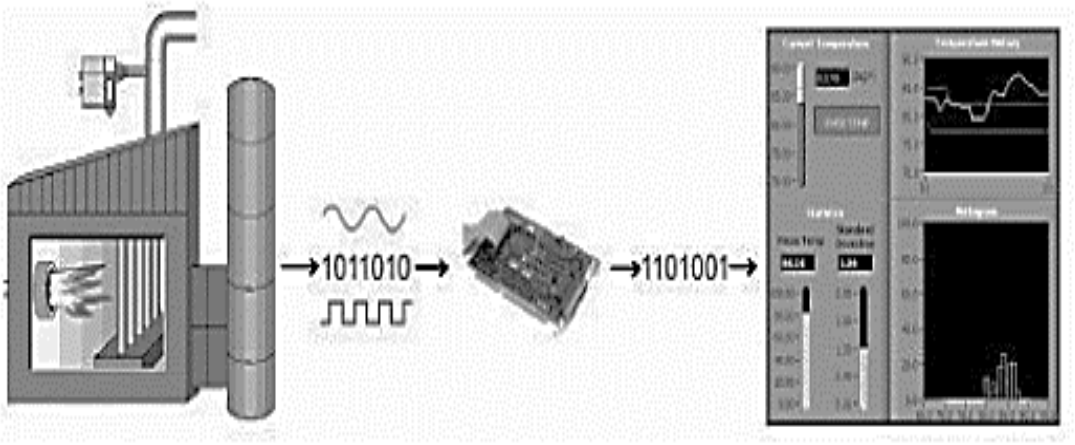


Figure 4.17: Industrial Data logging and display

Data acquisition system

A data acquisition system is a device or an integrated system used to collect information about the state or condition of various parameters of any process. For example, collecting day-to-day temperature of a particular location can be termed data acquisition. In simple words DAQ is defined as “Data acquisition is the process by which physical phenomena from the real world are transformed into electrical signals that are measured and converted into a digital format for processing, analyzing, and storage by a computer”.

Basic components of data acquisition systems

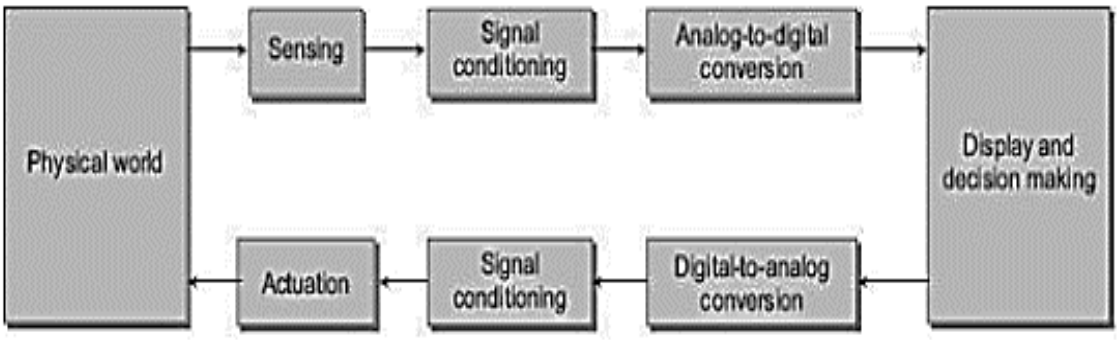


Figure 4.18: Basic components of data acquisition systems

- Sensors and transducers
- Field wiring
- Signal conditioning
- Data acquisition hardware
- PC (operating system)
- Data acquisition software

Sensors and transducers

Sensors and Transducers Transducer/Sensor converts input energy from one form to another form. According to the type, output sensors are classified in two types: digital sensors and analog sensors. The sensors which can produce a digital output signal, that is a digital representation of the input signal, having discrete values of magnitude measured at discrete times, are called digital sensors. A digital sensor must output logic levels that are compatible with the digital receiver. Examples of digital sensors include switches and position encoders. Analog sensors produce an output signal that is directly proportional to the input signal, and is continuous in both magnitude and in time. Most physical variables such as temperature, pressure and acceleration are continuous in nature and are readily measured with an analog sensor.

Signal Conditioning

Most sensors and transducers generate signals that must be conditioned before a measurement or DAQ device can reliably and accurately acquire the signal. This front-end processing is referred to as signal conditioning.

A signal conditioner may create excitation for certain transducers such as strain gauges and resistance temperature detectors, which require external excitation voltages or currents.

The main tasks performed by signal conditioning are as follows: Filtering, Amplification, Linearisation, Isolation, Excitation.

PC

The PC used in a data acquisition system can greatly affect the speeds at which data can be continuously and accurately acquired, processed, and stored for a particular application. Where high-speed data acquisition is performed with a plug-in expansion board, the throughput provided by bus architectures, such as the PCI expansion bus, is higher than that delivered by the standard ISA or EISA expansion bus of the PC.

The particular application, the microprocessor speed, hard-disk access time, disk capacity and the types of data transfer available, can all have an impact on the speed at which the computer is able to continuously acquire data. All PCs, for example, are capable of programmed I/O and interrupt-driven data transfers. The use of Direct Memory Access (DMA), in which dedicated hardware is used to transfer data directly into the computer's memory, greatly increases the system throughput and leaves the computer's microprocessor free for other tasks. In normal operation, the data acquired, from a plug-in data acquisition board or other DAQ hardware (e.g. data logger), is stored directly to system memory. Where the available system memory exceeds the amount of data to be acquired, data can be transferred to permanent storage, such as a hard disk, at any time. The speed at which the data is transferred to permanent storage does not affect the overall throughput of the data acquisition system.

A/D Conversion

A process of converting an analog signal into a digital signal comprises measuring the amplitude of the analog signal at consistent time intervals and producing a set of signals representing the measured digital value. The information in the digital signals and the known time interval enables one to convert the digital signal back to the analog signal. Analog to digital conversion of a continuous input signal normally occurs in two steps: sampling and quantisation. The sampler takes a time-varying analog input signal and converts it to a fixed voltage, current, electrical charge, or other output level. The quantiser takes the constant sampled level and compares it to the closest level from a discrete range of values called quantisation levels.

D/A Conversion

A digital-to-analog converter, or simply DAC, is a semiconductor device that is used to convert a digital code into an analog signal. Digital-to-analog conversion is the primary means by which digital equipment such as computer-based systems are able to translate digital data into real-world signals that are more understandable to or useable by humans, such as music, speech, pictures, video, and the like.

Display devices

After collecting information about the state of some process, the next consideration is how to present it in a form where it can be readily used and analysed. Standards of good practice for presenting data in either graphical or tabular form are covered, using either paper or a computer monitor screen as the display medium.

Example: Recorders, CRT

Advantages

- Reduced data redundancy
- Reduced updating errors and increased consistency
- Greater data integrity and independence from applications programs
- Improved data access to users through use of host and query languages
- Improved data security
- Reduced data entry, storage, and retrieval costs
- Facilitated development of new applications program

Disadvantages

- Database systems are complex, difficult, and time-consuming to design
- Substantial hardware and software start-up costs
- Damage to database affects virtually all applications programs
- Extensive conversion costs in moving form a file-based system to a database system
- Initial training required for all programmers and users

Computer control action- treatment of inputs, outputs, and control strategies

One of the primary requirements for a computer is that it is able to take data in, and produce output that contains data to be interpreted. I/O (Input/output) refers to the processes by which the CPU communicates with devices that handle the input and output.

Interaction of Computer and I/O Devices

The basic problem in the management of I/O devices is most easily seen by comparison to the memory system. With the memory system all timings are known. All memory transactions are controlled by a clock signal on bus. One can design a control system based on assumptions that a memory operation will complete within a fixed time.

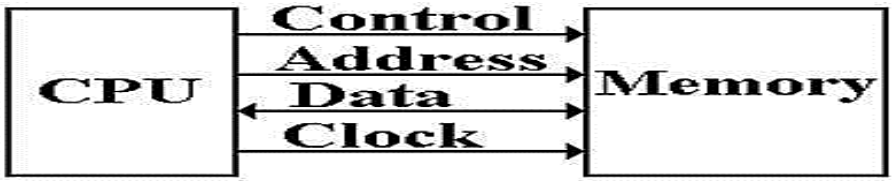


Figure 4.19: Interaction of Computer and I/O Devices

Timing of I/O operations presents some challenges to the system designer. For many I/O devices, the time to complete the data transfer either cannot be estimated or varies quite a bit. We need protocols for interfacing these I/O devices to the CPU.

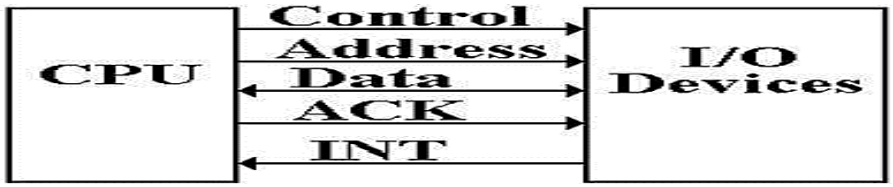


Figure 4.20: Interaction of Computer and I/O Devices with acknowledgement

The Four Strategies

Here are the simple definitions of the four I/O strategies.

Program Controlled I/O

This is the simplest to implement. The executing program manages every aspect of I/O processing. I/O occurs only when the program calls for it. If the I/O device is not ready to perform its function, the CPU waits for it to be ready; this is **"busy waiting"**. The next two strategies are built upon program controlled I/O.

Interrupt Driven I/O

In this variant, the I/O device can raise a signal called an **"interrupt"** when it is ready to perform input or output. The CPU performs the I/O only when the device is ready for it. In some cases, this interrupt can be viewed as an alarm, indicating an undesirable event.

Direct Memory Access

This variant elaborates on the two above. The I/O device interrupts and is sent a "word count" and starting address by the CPU. The transfer takes place as a block.

I/O Channel

This assigns I/O to a separate processor, which uses one of the above three strategies.

Question Bank for Unit - 3 & 4

Part- A

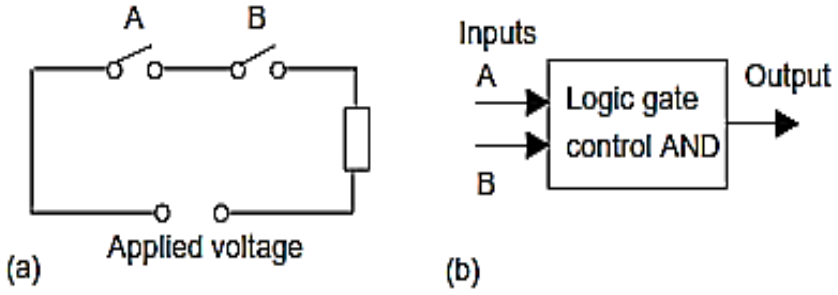
1. Explain DC motor?

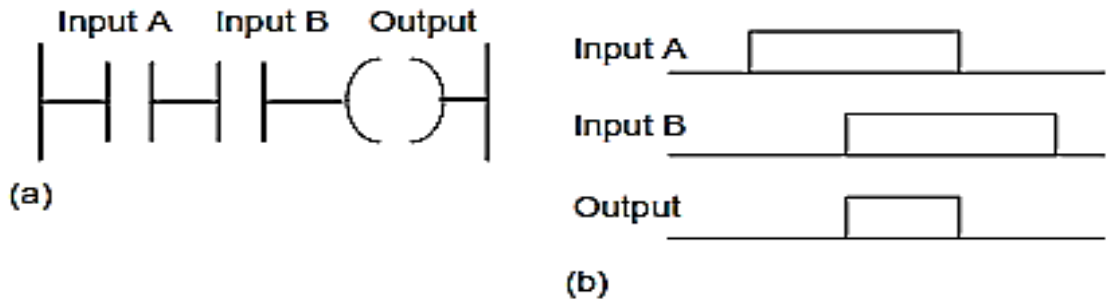
A DC motor has coils of wire mounted in slots on a cylinder of ferromagnetic material, this being termed the armature. The armature is mounted on bearings and is free to rotate. It is mounted in the magnetic field produced by permanent magnets or current passing through coils of wire, these being termed the field coils. When a current passes through the armature coil, forces act on the coil and result in rotation. Brushes and a commutator are used to reverse the current through the coil every half rotation and so keep the coil rotating. The speed of rotation can be changed by changing the size of the current to the armature coil. However, because fixed voltage supplies are generally used as the input to the coils, the required variable current is often obtained by an electronic circuit. This can control the average value of the voltage, and hence current, by varying the time for which the constant DC voltage is switched on. The term pulse width modulation (PWM) is used since the width of the voltage pulses is used to control the average DC voltage applied to the armature. A PLC might thus control the speed of rotation of a motor by controlling the electronic circuit used to control the width of the voltage pulses.

2. Explain which form of logic gate system is given by a ladder diagram with a rung having two normally open sets of contacts in parallel? (Explain AND gate?)

Figure (a) shows a situation where an output is not energized unless two, normally open, switches are both closed. Switch A and switch B have both to be closed, which thus gives an AND logic situation. We can think of this as representing a control system with two inputs A and B (Figure (b)). Only when A and B are both on is there an output. Thus if we use 1 to indicate an on signal and 0 to represent an off signal, then for there to be a 1 output we must have A and B both 1. Such an operation is said to be controlled by a logic gate and the relationship between the inputs to a logic gate and the outputs is tabulated in a form known as a truth table.

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





3. Explain which form of logic gate system is given by a ladder diagram with a rung having two normally closed gates in parallel? (Explain OR gate?)

Figure (a) shows an electrical circuit where an output is energized when switch A or B, both normally open, are closed. This describes an OR logic gate (Figure (b)) in that input A or input B must be on for there to be an output. The truth table is:

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

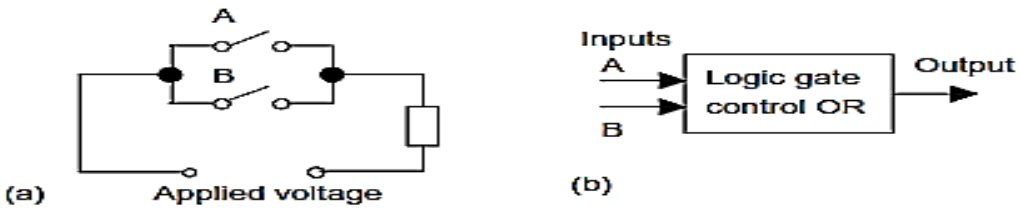
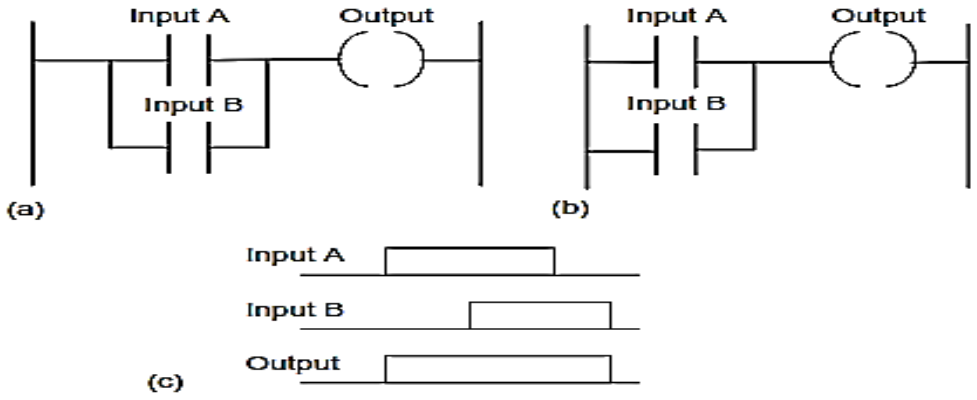


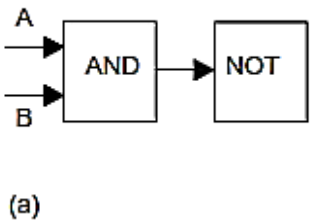
Figure (a) shows an OR logic gate system on a ladder diagram, Figure (b) showing an equivalent alternative way of drawing the same diagram. The ladder diagram starts with | |, normally open contacts labelled input A, to represent switch A and in parallel with it | |, normally open contacts labelled input B, to represent switch B. Either input A or input B have to be closed for the output to be energized (Figure (c)). The line then terminates with O to represent the output. In general: Alternative paths provided by vertical paths from the main rung of a ladder diagram, i.e. paths in parallel, represent logical OR operations.



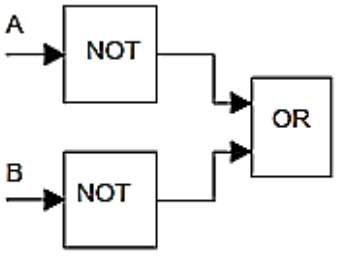
4. Explain which form of logic gate system is given by a ladder diagram with a rung having two normally open gates in series? (Explain NAND gate?)

Suppose we follow an AND gate with a NOT gate (Figure (a)). The consequence of having the NOT gate is to invert all the outputs from the AND gate. An alternative, which gives exactly the same results, is to put a NOT gate on each input and then follow that with OR (Figure (b)). The same truth table occurs, namely:

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

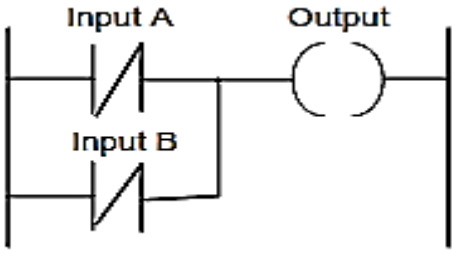


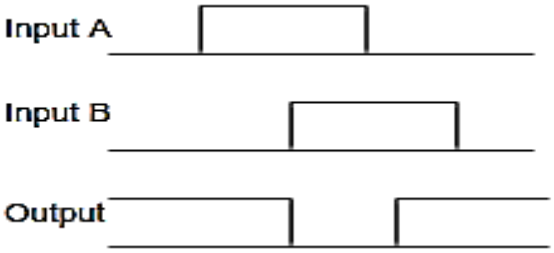
(a)



(b)

Both the inputs A and B have to be 0 for there to be a 1 output. There is an output when input A and input B are not 1. The combination of these gates is termed a NAND gate.

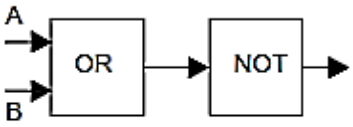




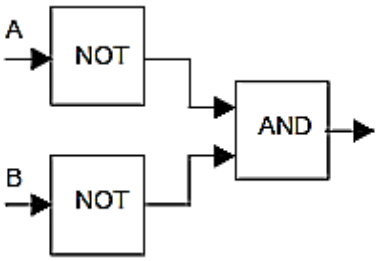
5. Explain which form of logic gate system is given by a ladder diagram with a rung having two normally closed gates in series? (Explain NOR gate?)

Suppose we follow an OR gate by a NOT gate (Figure (a)). The consequence of having the NOT gate is to invert the outputs of the OR gate. An alternative, which gives exactly the same results, is to put a NOT gate on each input and then an AND gate for the resulting inverted inputs (Figure (b)). The following is the resulting truth table:

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

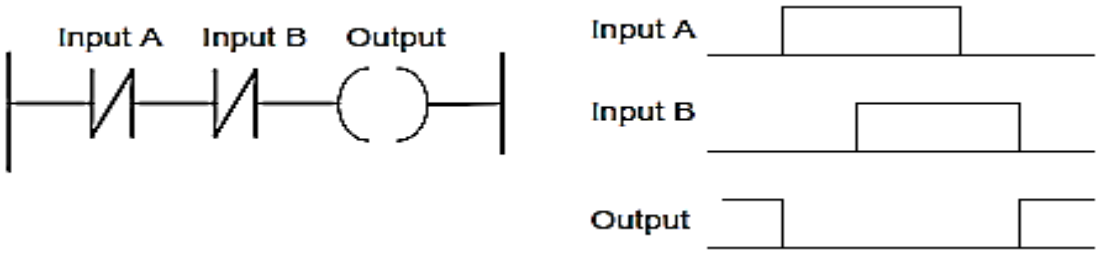


(a)



(b)

The combination of OR and NOT gates is termed a NOR gate. There is an output when neither input A or input B is 1. Figure shows a ladder diagram of a NOR system. When input A and input B are both not activated, there is a 1 output.



6. **Explain SCADA?**

7. **Give some advantages and disadvantages of SCADA?**

8. **Explain PROFI bus.**

9. **Explain Field bus.**

10. **Compare OSI and field bus.**

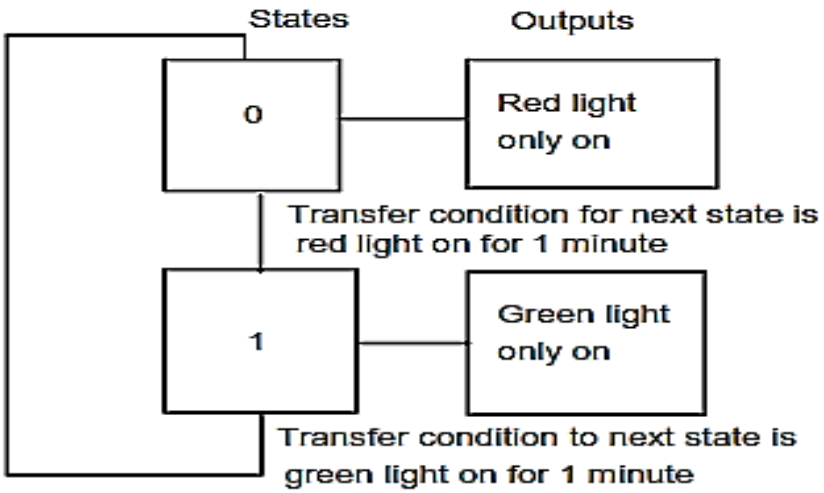
11. **What are the various standards of field bus?**

12. **What are the basic components of Data Acquisition System?**

Part-B

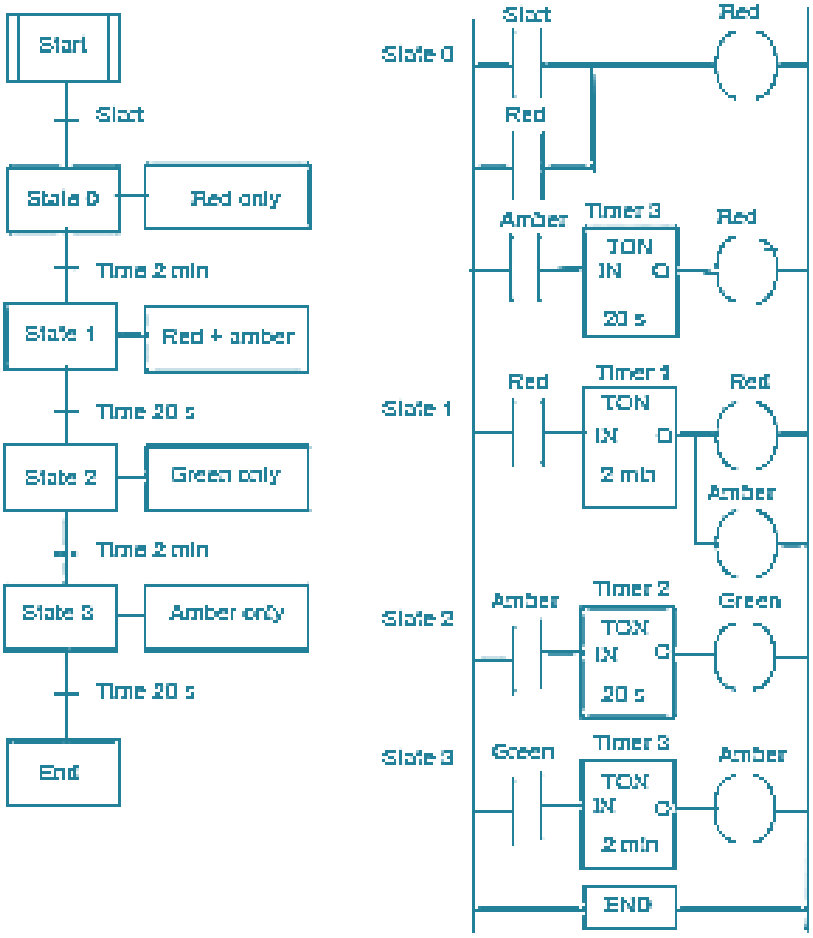
1. **Design a PLC Ladder for Traffic Light controller.**

If we wanted to describe a traffic lamp sequence, one way we could do this would be to represent it as a sequence of functions or states such as red light state and green light state and the inputs and outputs to each state. Figure illustrates this. State 0 has an input which is triggered after the green light has been on for 1 minute and an output of red light on. State 1 has an input which is triggered after the red light has been on for 1 minute and an output of green light on.



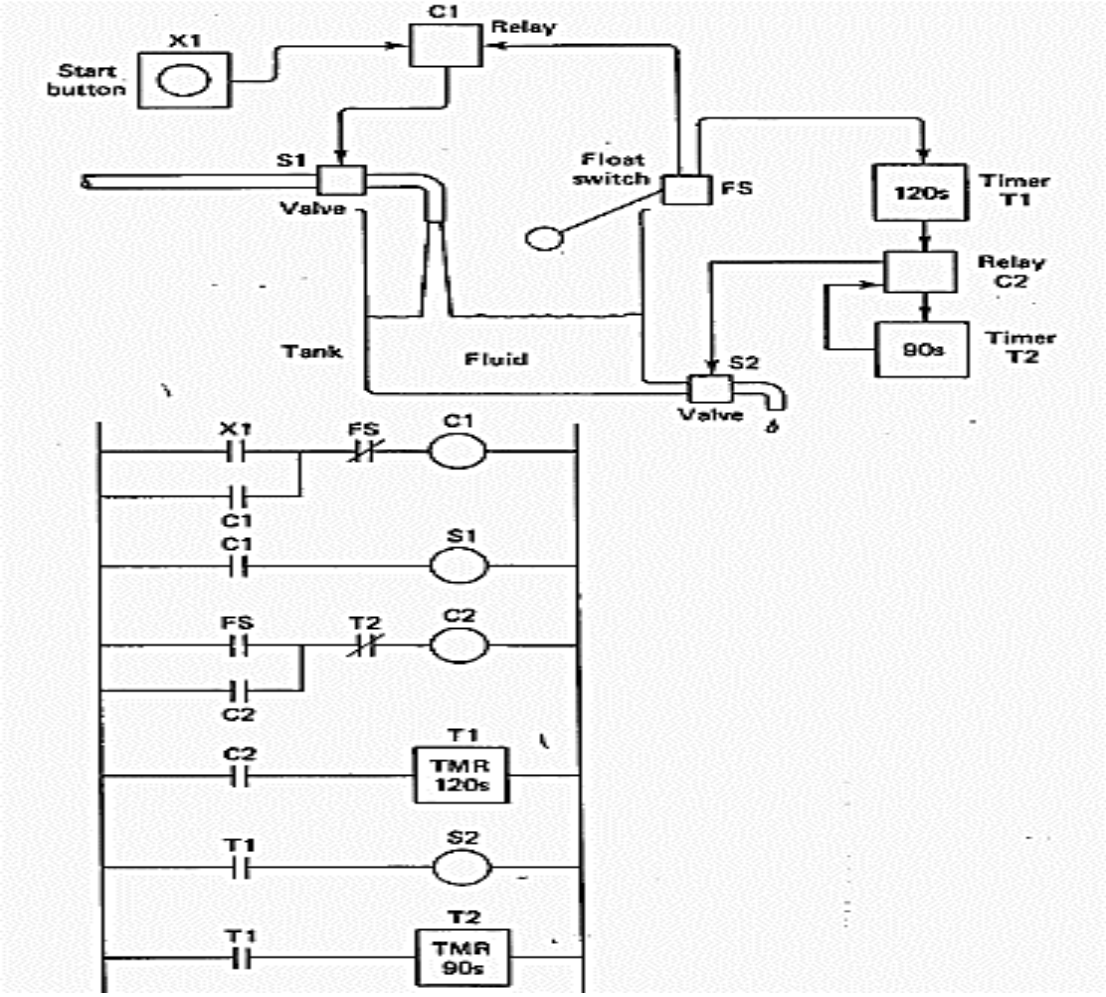
Sequence for traffic lights

As an illustration of programming involving timers consider the sequencing of traffic lights to give the sequence red only, red plus amber, green, amber, then repeat itself. A simple system might just have the sequence triggered by time, with each of the possible states occurring in sequence for a fixed amount of time. Figure 9.16 shows the sequential function chart and a possible ladder program to give the sequence.



Traffic light sequence

2. Design a PLC Ladder for Automatic water filling in tank.

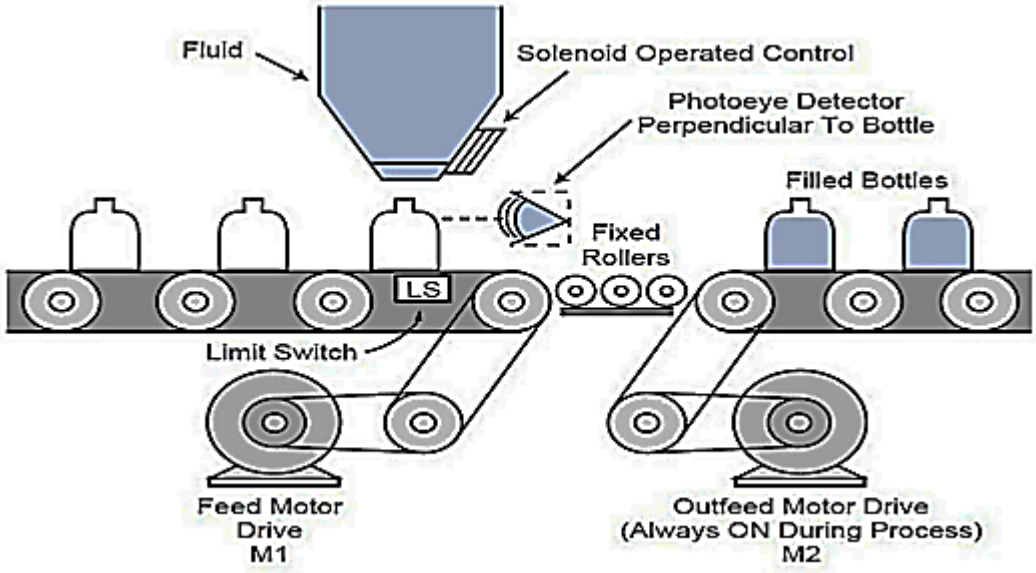


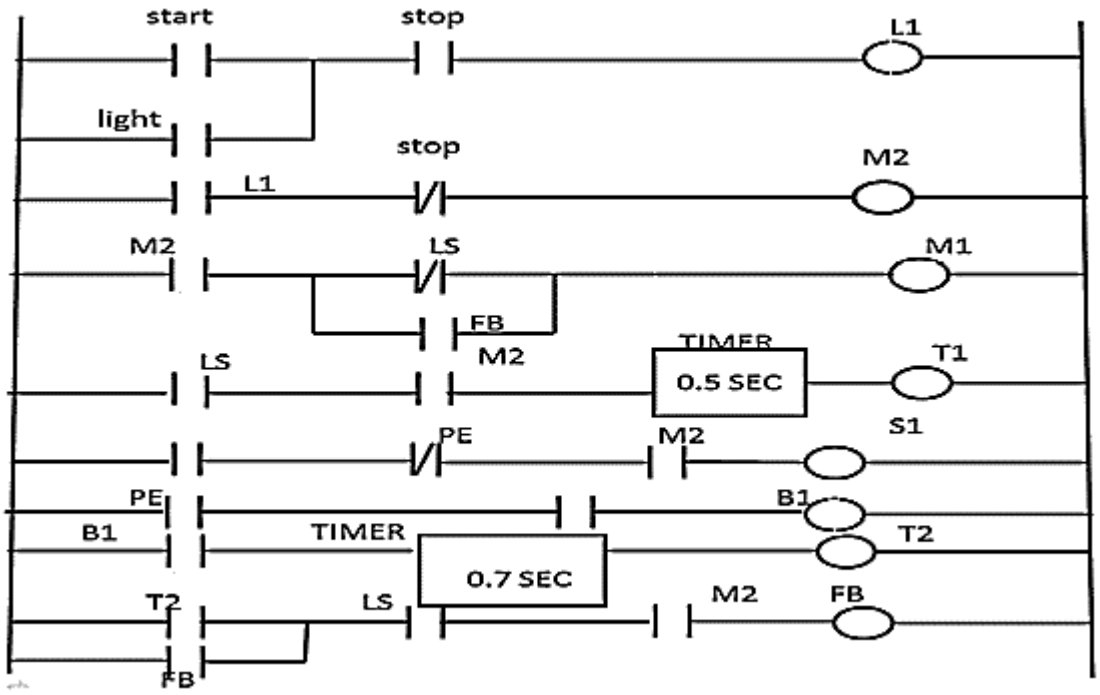
3. Design a PLC Ladder for Continuous bottle filling system.

This is one of the important application of PLC in the bottle filling industry where we want our bottles, which are moving on the conveyor belt, to be automatically detected at the appropriate position and get it filled by any desired liquid and also after getting filled the queued bottle gets chance to be filled. If this whole process is carried out manually it will really take a long time and also the quantities will be quite lesser. So PLC becomes requisite controller for these types of industry.

Here also just a small demonstration of the process was performed with the help of PLC where a ladder diagram was created to control the process and the ladder diagram was run the PLC trainer kit to see its justification.

Objective:- We will implement a control program that detects the position of a bottle via a limit switch then waits for 0.5 secs, and then fills the bottle until a photo detector detects the filled condition of the bottle. After the bottle is filled, the buzzer sounds and the control program will again wait for 0.7 secs. Before moving to the next bottle .Until the limit switch signals, the feed motor, M1 runs while there are fixed rollers which carries the filled bottles. Motor, M2 keeps running after the process has been started.





4. Design a PLC Ladder for automatic controlling the Conveyor belt.

Consider a conveyor belt that is to be used to transport goods from a loading machine to a packaging area (Figure). When an item is loaded onto the conveyor belt, a contact switch might be used to indicate that the item is on the belt and start the conveyor motor. The motor then has to keep running until the item reaches the far end of the conveyor and falls off into the packaging area. When it does this, a switch might be activated which has the effect of switching off the conveyor motor. The motor is then to remain off until the next item is loaded onto the belt. Thus the inputs to a PLC controlling the conveyor are from two switches and the output is to a motor.

Conveyor

5. **Explain in detail about SCADA with neat block diagram.**

6. **Explain the basic components of data acquisition system with neat diagram.**

7. **Discuss in detail about the Field Bus.**

8. **Discuss in detail about the Process Field Bus (PROFIBUS)**

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF MECHANICAL ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

COURSE MATERIAL

Program: B.E - MTRON **Semester: VI**

Course: PLC and Automation

Course code: SMR1303

UNIT 5 - PROGRAMMABLE LOGIC CONTROLLERS - SMR1303

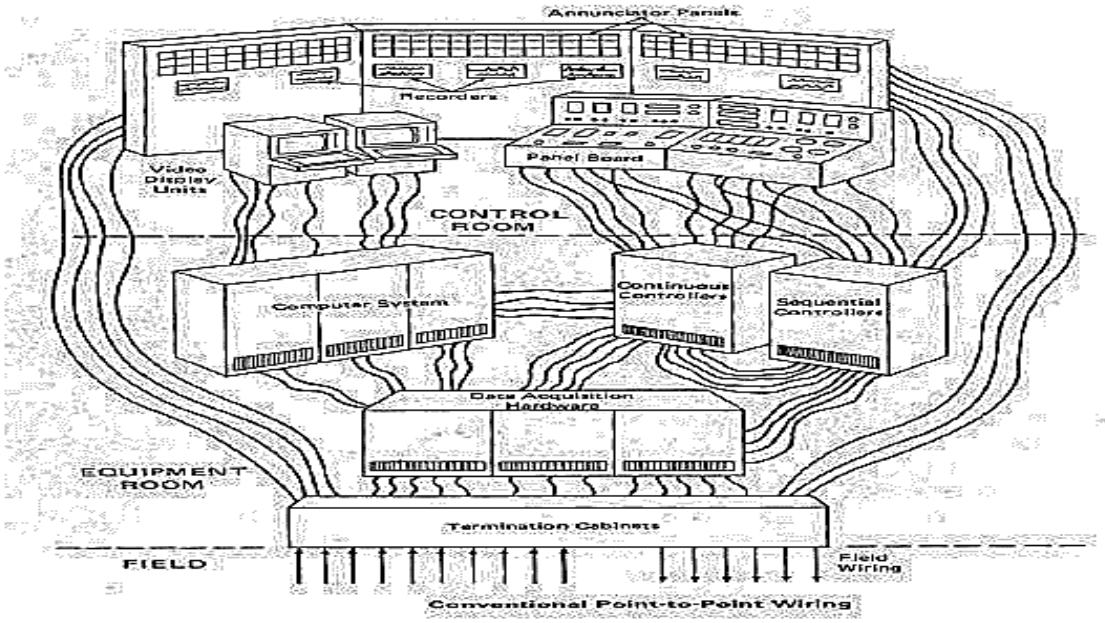
Evolution - Architectures - Comparison - Local control unit - Process interfacing issues - Communication facilities. Operator interfaces - Low level and high level operator interfaces -Operator displays - Engineering interfaces - Low level and high level engineering interfaces Applications of DCS in - Pulp and paper environment -Power plant - Petroleum – Refining environment Introduction to Soft PLC.

UNIT 5 - PROGRAMMABLE LOGIC CONTROLLERS - SMR1303

UNIT 5 - DISTRIBUTED CONTROL SYSTEMS

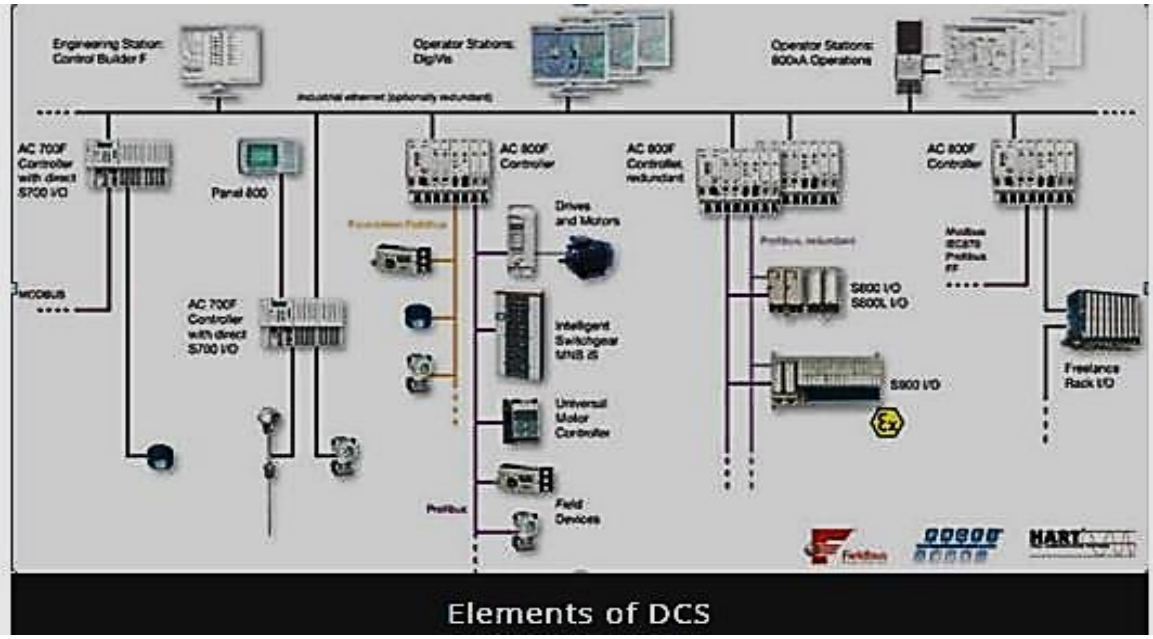
DCS Communication Facilities

In conventional non distributed control systems, the connections that allow communication between the various system elements are configured on a per-job basis. This system consists of a combination of continuous controllers, sequential controllers, data acquisition hardware, panel board instrumentation, and a computer system. The controllers communicate with each other by means of point-to-point wiring, usually within the control cabinets. This custom wiring reflects the particular control system configuration selected. The controllers are connected to the corresponding panel board instrumentation and to the computer system by means of pre fabricated cables. The computer obtains information from the data acquisition modules using similar hard wiring or cabling that is specific to the particular module configuration implemented. This approach to interconnecting system elements has proven to be expensive to design and check out, difficult to change, burdensome to document, and subject to errors. It becomes even more cumbersome if the system elements are distributed geographically around the plant. The first step taken to improve this situation was to introduce the concept of distributed multiplexing in the early 1970s. This concept was first used in the process control industry to implement large-scale data acquisition systems, which at that time had grown to several thousand inputs in size. To reduce the cost of wiring, remote multiplexers located near the sensors in the field were used to convert the inputs to digital form and transmit them back to the data acquisition computer over a shared communication system.



When distributed control systems were introduced in the late 1970s, the use of digital communications was extended to control-oriented systems as well. The communication system began to be viewed as a facility that the various elements and devices in the distributed network share, as the “black box” representation in Figure shows. Replacing dedicated point-to-point wiring and cabling with this communications facility provides a considerable number of benefits to the user: 1. The cost of plant wiring is reduced significantly (see References, since thousands of wires are replaced by the few cables or buses used to implement the shared communication system. 2. The flexibility of making changes increases, since it is the software or firmware configurations in the system elements that define the data interconnection paths and not hard wiring. 3. It takes less time to implement a large system, since the wiring labor is nearly eliminated, configuration errors are reduced, and less time is required to check out the interconnections 4 The control s stem is more reliable due to the significant reduction In physical connections in the system (a major source of failures) However, replacing hard wiring with the shared communications network of a distributed control system also raises a number of questions for the user In a conventional system, communications between system elements travel at the speed of light, essentially with zero delay. Also, since the hard-wired communication channels between elements are dedicated, there is no danger of overloading a channel. In the case of a shared communication system, the user must be able to judge whether the response time and capacity of the shared system (in addition to many other performance factors) are adequate for the application. This can be a difficult problem for the user, since there are significant differences and few standards among the various communication systems available on the market today

DCS ARCHITECTURE ISSUES



Elements of DCS

Distributed Control System continuously interacts with the processes in process control applications ones it gets instruction from the operator. It also facilitates to variable set points and opening and closing of valves for manual control by the operator. Its human machine interface (HMI), face plates and trend display gives the effective monitoring of industrial processes.

Engineering PC or controller

This controller is the supervisory controller over all the distributed processing controllers. Control algorithms and configuration of various devices are executed in this controller. Network communication between processing and engineering PC can be implemented by simplex or redundant configurations.

Distributed controller or Local control unit

It can be placed near to field devices (sensors and actuators) or certain location where these field devices are connected via communication link. It receives the instructions from the engineering station like set point and other parameters and directly controls field devices.

It can sense and control both analog and digital inputs / outputs by analog and digital I/O modules. These modules are extendable according to the number of inputs and outputs. It collects the information from discrete field devices and sends this information to operating and engineering stations.

In above figure AC 700F and AC 800Fcontrollers acts as communication interface between field devices and engineering station. Most of the cases these act as local control for field instruments.

Operating station or HMI

It is used to monitor entire plant parameters graphically and to log the data in plant database systems. Trend display of various process parameters provides the effective display and easy monitoring.

These operating stations are of different types such as some operating stations (PC's) used to monitor only parameters, some for only trend display, some for data logging and alarming requirements. These can also be configured to have control capabilities.

Communication media and protocol

Communication media consists of transmission cables to transmit the data such as coaxial cables, copper wires, fiber optic cables and sometimes it might be wireless. Communication protocols selected depends on the number of devices to be connected to this network.

For example, RS232 supports only for 2 devices and Profibus for 126 devices or nodes. Some of these protocols include Ethernet, DeviceNet, foundation filed bus, modbus, CAN, etc.

In DCS, two or more communication protocols are used in between two or more areas such as between field control devices and distributed controllers and other one between distributed controllers and supervisory control stations such as operating and engineering stations.

Important features of DCS

• To handle complex processes:

In factory automation structure, PLC-Programming Logic Controller is used to control and monitor the process parameters at high speed requirements. However due to limitation of number of I/O devices, PLC's cannot handle complex structure.

Hence DCS is preferred for complex control applications with more number of I/O's with dedicated controllers. These are used in manufacturing processes where designing of multiple products are in multiple procedures such as batch process control.

System redundancy:

DCS facilitates system availability when needed by redundant feature at every level. Resuming of the steady state operation after any outages, whether planned or unplanned is somewhat better compared to other automation control devices. Redundancy raises the system reliability by maintaining system operation continuously even in some abnormalities while system is in operation.

Lot of Predefined function blocks:

DCS offers many algorithms, more standard application libraries, pre-tested and pre-defined functions to deal with large complex systems. This makes programming to control various applications being easy and consuming less time to program and control.

Powerful programming languages:

It provides more number of programming languages like ladder, function block, sequential, etc for creating the custom programming based on user interest.

More sophisticated HMI:

Similar to the SCADA system, DCS can also monitor and control through HMI's (Human Machine Interface) which provides sufficient data to the operator to charge over various processes and it acts as heart of the system. But this type of industrial control system covers large geographical areas whereas DCS covers confined area.

DCS completely takes the entire process plant to control room as a PC window. Trending, logging and graphical representation of the HMI's give effective user interface. Powerful alarming system of DCS helps operators to respond more quickly to the plant conditions

Scalable platform:

Structure of DCS can be scalable based on the number I/O's from small to large server system by adding more number of clients and servers in communication system and also by adding more I/O modules in distributed controllers.

System security:

Access to control various processes leads to plant safety. DCS design offers perfect secured system to handle system functions for better factory automation control. Security is also provided at different levels such as engineer level, entrepreneur level, operator level, etc.

OPERATOR INTERFACES

The control and communication equipment performs the bulk of the automated functions that are required for operating an industrial process For this automated equipment to be used in a safe and effective manner, however, it is absolutely necessary to have a well-engineered human interface system to permit error-free interactions between the humans and the automated system Two distinct groups of plant personnel interact with the control system on a regular basis Instrumentation and control system engineers—These people are responsible for setting up the control system initially and adjusting and maintaining it from time to time afterwards Plant operators—These people are responsible for monitoring, supervising, and running the process through the control system during startup, operation, and shutdown conditions As the generalized distributed control system architecture, a human interface capability can be provided at one or both of two levels Through a low-level human interface (LLHI) connected directly to the local control unit or data input/output unit (DI/OU) via dedicated cabling, Through a high-level human interface (HLHI) connected to an LCU .or DI/OU only through the shared communications facility. The low-level human interface equipment used in distributed control systems usually resembles the panelboard instrumentation (stations, indicators, and recorders) and tuning devices used in conventional electric analog control systems. The HLHI equipment makes maximum use of the latest display technology (e.g., CRTs or flat-panel displays) and peripheral devices (e.g., printers and magnetic storage) that are available on the market; it is configured in a console arrangement that allows operator and engineer to be seated during use. When it is included in the system configuration, the LLHI generally is located geographically close to (within 100—200 feet of) the LCU or DI/OU to which it is connected On the other hand, the HLHI can be located anywhere in the plant, including the central control room. The needs of the application will determine whether the particular installation has one or both levels of interface Figures shows some examples of typical installations and their corresponding equipment configurations. Figure 5.1 illustrates a relatively small and simple installation A single LCU located in the plant equipment room (sometimes called the relay room) performs all of the required control functions Low-level human interface units located in the equipment room and the plant control room provide the complete operator and instrument engineer interface for the control system This type of equipment configuration is typical of d stand- alone control system for a small process or of a small digital control system installed in a plant controlled primarily with conventional electrical analog or pneumatic equipment

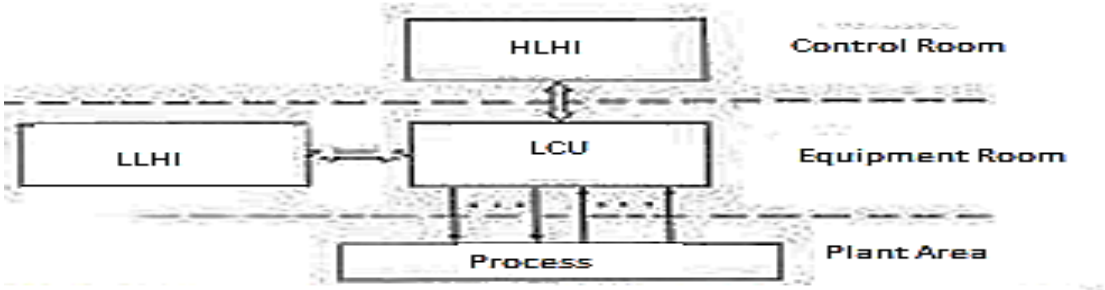


Fig Stand Alone Control Configuration

Figure shows a typical structure of a complete plantwide control system Several LCUs are used to implement the functions required in controlling the process, therefore, the control is functionally distributed However, the LCUs are all located in a central equipment room area, and so it is not a geographically distributed control system Both high-level and low-level human interface devices are located in the control room area for operational purposes Most of the operator control functions are performed using the high-level interface, the low-level interface is included in the configuration primarily to serve as a backup in case the high-level interface fails A high evel human interface is located in the instrument engineer's area so that control system monitoring and analysis can be done without disturbing plant operations This type of installation is typical of early distributed control system configurations in which equipment location and operator interface design followed conventional practices

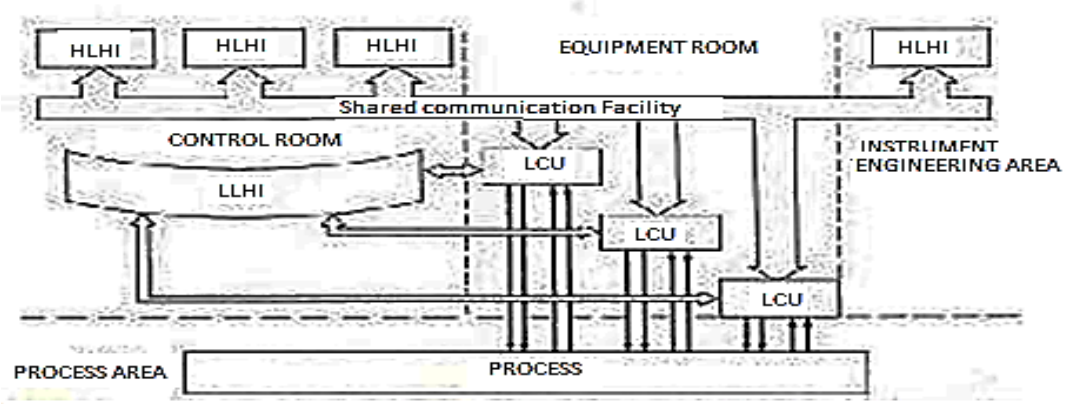


Fig. Geographically Centralized Control Configuration

Figure shows a fully distributed control system configuration In this case, each LCU is located in the plant area closest to the portion of the process that it controls Associated low-level human interface equipment (if provided) is also located in this area The control room and instrument engineering areas contain high-level human interface units, which are used to perform all of the primary operational and engineering functions The low-level units are used only as manual backup controls in case the high-level equipment fails or needs maintenance This configuration takes advantage of two areas of equipment savings that result from a totally distributed system

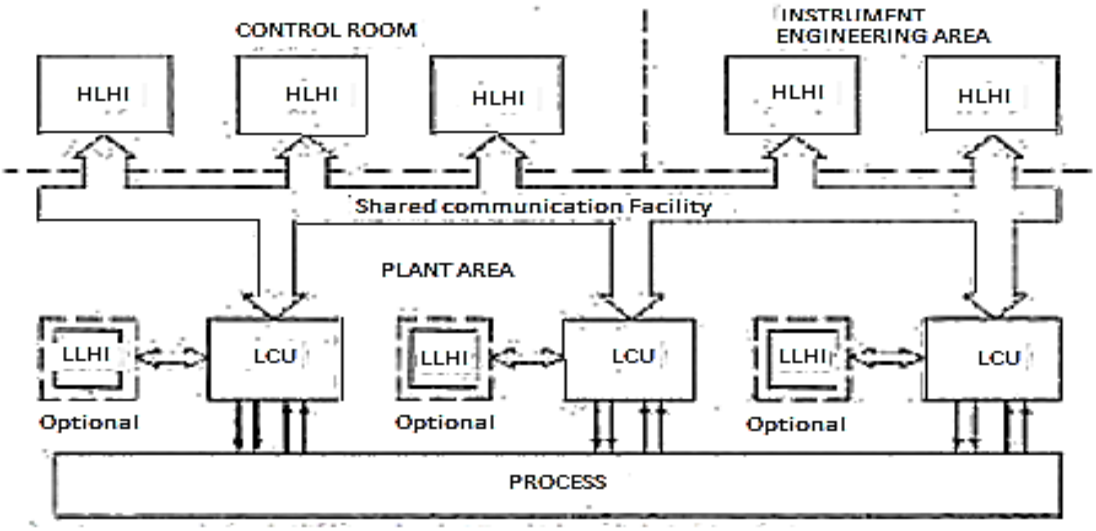


Fig. Geographically Distributed Control Configuration

Architecture, reduction in control room size (by eliminating panelboard equipment), and reduction in field wiring costs (by placing LCUs near the process).

OPERATOR INTERFACE REQUIREMENTS

Despite the continuing trend toward increased automation in process control and less reliance on the operator, the basic responsibilities of the operator have remained largely the same in the last fifty years. Most of the changes have come in the relative emphasis on the various operator functions and the means provided to accomplish them. As a result, the operator interface in a distributed control system must allow the operator to perform tasks in the following traditional areas of responsibility process monitoring, process control, process diagnostics, and process record keeping. In addition, it is important to design the operator interface system using human factors design principles (also called ergonomics) to ensure that the operator can perform these tasks in an effective manner with minimum risk of confusion or error. The following paragraphs provide a discussion of the key functional requirements in each of these areas

Process Monitoring

A basic function of the operator interface system is to allow operator (whether one or more) to observe and monitor the current state of the process this function includes the following specific requirements.

 The current values of all process variables of interest in the system must be available for the operator to view at any time This includes both continuous process variables (e g , flows, temperatures, and pressures) and logical process variables (e g , pump on/off status and switch positions) The operator must have rapid access to any variable, and the values displayed must be accurate and current If the information provided is not valid for some reason (e g , a sensor has failed or has been taken out of service for maintenance), this condition should be readily visible to the operator.

 Each process variable, rather than being identified by a hardware address only, must be identifiable by a "tag" or name assigned by the instrument engineer, a descriptor that expands on and describes the tagged variable must be associated with the tag The tag and descriptor give the variable a meaning relative to the process, an example might be to label a certain temperature with a tag of TT075B and a corresponding descriptor "COLUMN TEMPERATURE 75 IN AREA B."

 The value of the process variable must be in engineering units that are meaningful to the operator, and those units must be displayed along with the variable values. In the temperature example just given, the engineering units might be in degrees Fahrenheit or Celsius.

 In many cases, the operator is interested in variables that are functions of or combinations of the basic process variables being measured (e.g. an average of several temperatures, a maximum of several flows, or a computed enthalpy). The operator must have these computed variables available at all times in the same formats as the basic variables (i e, tags, descriptors, and engineering units).

Another monitoring function of the operator interface is to detect abnormalities in the state of the process and to report them to the operator. In its simplest form, this is the familiar function of alarming. Some of the specific requirements of this function are the following:

 The control and computing hardware in the distributed system identifies the alarm statuses of individual variables in the process. The operator interface system must report these statuses to the operator in a clear manner Types of alarms for each variable—such as high, low, and deviation (from a nominal value)—must be differentiated clearly True alarms must be differentiated from indications of process equipment status that do not denote an abnormal condition requiring operator action.

 The operator interface must also report similar alarm statuses for computed variables 3 the operator interface must either display the alarm limits along with the process variable or make them easily accessible to the operator.

 When the system has detected an alarm condition, the interface must alert the operator to this condition in unambiguous terms and require the operator to acknowledge the existence of the alarm.

 If the system detects multiple alarm conditions within a short time period, the operator interface must inform the operator that multiple alarms have occurred, preferably with some indication of the priority of the various alarm conditions.

 In some processes, "abnormal operation" can be detected only by looking at a combination of several process variables and noting if this combination is within an allowable region of operation In this case, the operator interface system must provide an appropriate mechanism to allow the operator to view this multivariable alarm status condition and interpret it properly.

When monitoring the process, an operator is interested in not only the current value of a process variable but also its trend in time. This gives the operator an idea of the direction in which the process is moving and whether or not there is trouble ahead. For this reason, the operator interface system must provide the operator with fast access to the recent history of selected process variables in the plant; these variables are called trended variables. Some specific requirements in trending are that:

- It must be possible to group the trended variables by related process function as well as by similarities in time scale of interest. For example, it might make sense to group all temperatures that are associated with a particular portion of the process.
- The trend graph must clearly label the engineering units, time increments, and absolute time of day of the trended variables.
- The operator must be able to obtain a precise reading (in engineering units) of both the current value as well as past values of the trended variable.
- If at all possible, the same graph displaying the trend should also show auxiliary information that would help the operator evaluate the status of the trended variable. This information might include the nominal value of the variable, the set point of the associated control loop, the allowed range of the variable, or the allowed rate of change.

Process Control

The process monitoring capabilities just described provide the necessary information for the operator's primary function— process control. The following specific operator interface requirements come under the category of process control

- The operator interface must allow the operator to have rapid access to all of the continuous control loops and logic sequences in the process control system.
- For each continuous control loop, the interface must allow the operator to perform all of the normal control functions changing control modes (e g , automatic, manual, or cascade), changing control outputs in manual mode, changing set points in automatic mode, and monitoring the results of these actions.
- The interface must allow the operator to perform such logic control operations as starting and stopping pumps or opening and closing valves. If interlocking logic is included in these operations, the interface must allow the operator to observe the status of the most recently requested command, the current logic state of the process, and the status of any permissives (interlocking signals) that may be preventing execution of the requested command.
- In the case of a batch control sequence, the operator interface must allow the operator to observe the current status of the sequence and to interact with it to initiate new steps or halt the sequence, as required.

 In both the continuous and sequential control cases, the interface system must allow the operator to have access to and be able to manipulate the control outputs despite any singlepoint failure in the equipment between the operator interface and the control outputs.

Process Diagnostics

Monitoring and controlling the process under normal operating conditions are relatively simple functions compared to operation under abnormal or hazardous conditions caused by failures in plant equipment or in the instrumentation and control system. The operator interface system must provide enough information during these unusual conditions to allow the operator to identify the equipment causing the problem, take measures to correct it, and move the process back to its normal operating state. The first step in this sequence is to determine whether it is the instrumentation and control equipment that is causing the problem To this end, the distributed control system should provide the following diagnostic features and make the results of the diagnostic tests available to the operator.

 Ongoing tests and reasonableness checks on the sensors and analyzers that measure the process variables of interest.
 Ongoing self-tests on the components and modules within the distributed control system itself controllers, communication elements, computing devices, and the human interface equipment itself. Historically, diagnosing problems within the process itself has been a manual function left to the operator. Operator interface systems have been designed to display all of the available process information (both relevant and irrelevant), and the operator has had to sort it all out and come up with the right diagnosis. This was not a bad approach when the operator had to contend with small processes, those characterized by only a few hundred process variables. More recently, however, processes have grown to such a size that describing them takes 5,000—10,000 process variables (many of which may be strongly interacting). It has become extremely difficult for an operator to identify the source and nature of a fault in an item of process equipment in this environment. A conventional alarming system, for example, may indicate the most immediate failure symptom but provide few clues as to the original source of the alarm condition. As a result, diagnostic functions that automatically detect process faults are now often required in distributed control systems. These functions may include:
 First-out alarming functions, which tell the operator which alarm in a sequence occurred first,
 Priority alarming functions, which rank the current alarms by their importance to process operation, allowing the operator to safely ignore the less important ones, at least temporarily,
 More advanced diagnostic functions that use a combination of alarming information and data on process variables to identify the item of failed process equipment and (in some cases) the mode of failure.

Many of these advanced alarming and diagnostic functions are application- oriented ones that the designer must configure for the specific process of interest, however, the distributed control system must support the implementation of these functions.

Process Record Keeping

One of the more tedious duties that operating people in a process plant must perform has been to walk the board, that is, to take a pencil and clipboard and periodically note and record the current values of all process variables in the plant Depending on the pro- cess, the frequency for doing this has ranged from once an hour to once every several hours This logged information, along with the trend recordings obtained automatically, serves as a useful record of plant operating status during each shift The record-keeping burden has increased significantly in recent years due (in part, at least) to governmental reporting requirements related to pollution monitoring, product liability, and worker safety regulations. Record-keeping was one of the first functions to be automated using conventional computer systems In state-of-the-art distributed control systems, this function often can be implemented in the operator interface system without the use of a separate computer Specific record-keeping requirements include the following

- Recording of short-term trending information—to record in real time mode.
- Manual input of process data—the operator must be able to enter manually collected process information into the system for record- keeping purposes. This information includes both numeric data and operator notes and journal entries.
- Recording of alarms—these are logged on a printer, a data storage device, or both, as they occur. Often, the return-to-normal status and operator acknowledgments must also be logged. The information recorded includes the tag name of the process variable, the time of alarm, and the type of alarm (high, low, or deviation). There must also be a mechanism that allows convenient review of the alarm in- formation.
- Periodic records of process variable information—The values of selected variables are logged on a printer, data storage device, or both, on a periodic basis every few minutes or every hour, depending on the dynamics of the variable The operator or instrument engineer may decide to store an averaged value Over the sampling period in- stead of the instantaneous value.
- Long-term storage and retrieval of information—the alarms and periodic logs as described above are accessible for short periods of time, commonly, for a single eight-hour shift or a single day. In addition, the same information, or a smoothed or filtered version of it, must be stored on a long-term basis (months or years). The system must include a mechanism for easy retrieval or ‘‘instant replay’‘ of such information
- Recording of operator control actions—Some process plants require the actions of the operator affecting control of the process to be recorded automatically These include changes in control mode, set point, manual output, or logic command Clearly this recording function must be implemented in such a way that the operator cannot deactivate it.

Guidelines for Human Factors Design

In the past, equipment used for operator interfacing has often been designed more for the convenience of the equipment vendor or architect-engineer than for ease of use by the operator In recent years, it has become clear that a small investment made in the proper design of human interfacing equipment pays handsome dividends fever operator errors (which can cause plant downtime or damage to equipment), less operator fatigue (which can cause a loss in productivity), and more efficient use of operating personnel. Some general design guidelines for designing operator interface systems for industrial control include the following:

- Consider the full range of expected operator population (e.g. , male and female, large and small, right-handers and left-handers).
- Take into account common minor disabilities in operators (e.g., color blindness and nearsightedness).
- Design the system for operators, not for computer programmers or engineers.
- Allow rapid access to all necessary controls and displays
- Arrange equipment and displays to make sense from an operational point of view, cluster with respect to process unit, functional operation, or both
- Make consistent use of colors, symbols, labels, and positions to minimize operator confusion
- Do not flood the operator with a lot of parallel information that is not structured in any way, the information should be prioritized, organized in a meaningful manner, and reported only when it changes significantly
- Ensure that the operator's short-term memory is not overtaxed when performing a complex sequence of operations provide aids such as operator guides, menus, prompts, or interactive sequences for assistance in these operations These aids are particularly important in stressful situations, during which short-term memory is not a reliable source of operating information
- As much as possible, design the system to detect and filter out erroneous operator inputs, when an error occurs, the system must tell the operator what the input error was and what to do next
- Make sure the control room environment (e g, light, sound levels, and layout) is consistent with the selection and design of the control room equipment In some respects, these guidelines may only seem to state obvious, common-sense design principles, however, a glance at existing operator interface designs shows that these principles are very often violated, either for design expediency or through ignorance of these ergonomic issues.

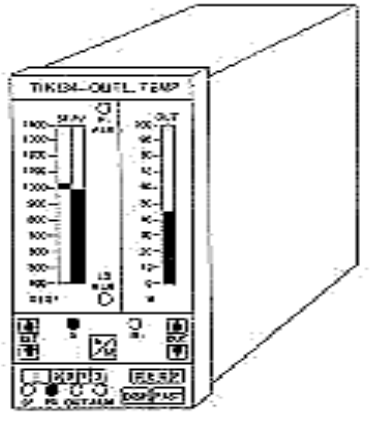
LOW LEVEL OPERATOR INTERFACES

The low level operator interface (LLOI) in a distributed control system is connected directly to the LCU and is dedicated to controlling and monitoring that LCU. This contrasts with the high Ievel operator interface (HLOI), which can be associated with multiple LCUs and is

connected to them through the shared communication facility. LLOIs are used in a variety of applications, in some cases in conjunction with high-level operator interfaces (HLOIs) and in others in place of them. In some applications, all operator functions are performed through the HLOI and no low-level interface is required except during emergency or failure conditions. There are a number of motivations for using an LLOI: It provides an interface that is familiar to operators trained to use panelboard instrumentation, since it is usually designed to resemble that type of instrumentation. It is usually less expensive than an HLOI in small applications (say, less than 50 control loops). It can provide manual backup in case the automatic control equipment or the HLOI fails. LLOI instrumentati6usually includes the following devices: control stations, indicator stations, alarm annunciators, and trend recorder. In some distributed control systems, the vendor offers exactly the same type of instrumentation as used in his conventional analog and logic control systems. More often, however, the vendor supplies smart (microprocessor- based) instrumentation, which offers the user functionality beyond that available in conventional panelboard instrumentation.

Continuous Control Station

One type of panel board instrumentation used in process control systems is the manual/automatic station associated with a continuous control loop. The stations discussed here are split stations; that is, they are separate from the LCU. Figure illustrates a typical version of a smart continuous control station in a distributed control system. As in the case of most conventional control stations, this one has bar graph indicators that display the process variable ("PV"), associated set point ("SP"), and the control output as a percent of scale ("OUT"). In addition, however, the smart station includes a shared digital display to provide a precise reading of each of these variables in engineering units. The units used are indicated in an accompanying digital display of are printed on a removable label (in this example, "DEGF" or "%"). The shared digital display also can be used to indicate the high and low alarm limits ("HI ALM" and "LO ALM") on the process variable when selected by the operator.



Pushbuttons allow the operator to change the mode of control (e.g. , manual, automatic, or cascade) and to ramp the set point ("SET") or control output ("OUT"), depending on the mode Usually, both fast and slow ramping speeds are provided for the convenience of the operator. The indications the control station often provides include any alarms associated with the process variable being controlled and an indication of the operational status (whether "healthy" or not) of the associated. To minimize requirements for spare parts, one basic control station should be used for all types of associated control loops: standard PID, cascade, ratio, or bias. The station can be customized through the configuration of options in the electronics (using jumpers or switches) and on the front plate indicators and switches (using different overlays or faceplates as appropriate). To be effective as a manual backup station in addition to its role as a single-loop operator interface, the control station must be connected directly to the control output section of the LCU or to the associated field termination panel. In this arrangement, a ‘‘hard" control output (e.g., a 4—20 ma signal) generated by the station can pass as a backup control signal in case the LCU fails or is under maintenance. The direct connection a so allows the process variable. Input to come into the station for indication to the operator during manual control. To keep the control output from going through a step change in value when the backup mode is initiated or concluded (automatic bumpless transfer), the station and the LCU must be aware of each other's nominal control output signal These signal values and other useful information (such as alarm and diagnostic status signals) are often sent over a direct serial communication link between the LCU and the station.

Manual Loader Station

Some applications use the HLOI as the primary L6Mi:l station and don't require a full-blown continuous control station for each loop. However, a device is still needed to hold the 4—20 ma control output signal. if the LCU fails or is taken off-line for maintenance or other reasons In this situation, a simple manual loader station is a low-cost alternative to the continuous control station for the purposes of backup The manual loader station is plugged in at the same point as the continuous control station but only allows the operator to run the loop in manual mode Sometimes the process variable is displayed, more often it is not Any balancing of the control output to accomplish bumpless transfer to or from backup is accomplished manually Both the continuous control station (if used for backup) and the manual loader station should be powered from a different supply than that used for the LCU, to ensure continuous backup in case of an LCU power failure.

Indicator Station

If the operator must be able to monitor process variables not associated with control loops, a panel board indicator station can be provided as a part of the LLOI family of products. The indicator station is similar to the control station in that it provides both bar graph and digital numeric readouts of the process variables in engineering units. Of course, an indicator station requires no control push buttons However, it often provides alarming and LCU diagnostic indications, as does the continuous control station.

Logic Station

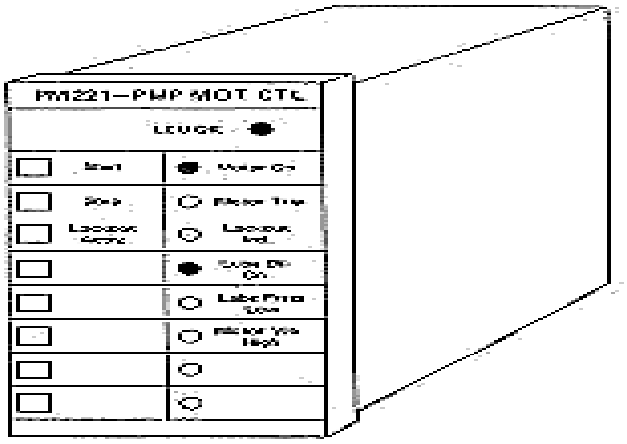


Figure illustrates a control station for a logic control or sequential control system. It consists simply of a set of pushbuttons and indicating lights that are assigned different meanings (through labels) depending on the logic functions being implemented. This type of station is used to turn pumps on and off, start automatic sequences, or provide permissives or other operator inputs to the logic system. In some systems, the logic control station performs a manual backup function similar to that performed by the continuous control station in case of a failure of an LCU. More often, however, the logic station acts simply as a low-cost operator interface; if the LCU fails, the logic outputs revert to their default or safe states. Smart Annunciators Alarm annunciators in distributed control systems are often microprocessor-based, providing a level of functionality beyond the capability of conventional hard-wired annunciator systems. These smart annunciators can provide such functions as: Alarm prioritization—The annunciator differentiates between status annunciation and true alarms (and how critical the alarms are); Annunciation and acknowledgment mode options—The operator receives a variety of audible and visible alarm annunciation signals (e g, horns, buzzers, flashing lights, and voice messages) A range of alarm acknowledgment and silencing modes also can be provided. First-out annunciation—the annunciator displays the first alarm that appears within a selected group. Alarm “cutout”—the annunciator suppresses an alarm condition if other specified status conditions are fulfilled. The last function is valuable in minimizing meaningless ‘‘nuisance ’‘alarms. For example, if a pump fails and triggers an alarm, the pump-failed status signal can be used to lock out other related alarms such as ‘‘low flow‘‘ or ‘‘pump speed low,” since they are not meaningful given the failed operating status of the pump of course, the four alarm logic functions previously listed also can be accomplished within the LCUs in the distributed system itself; however, in some applications it may be convenient to incorporate the functions externally in the annunciators.

Chart Recorders

Although conventional round chart or strip chart recorders are often used to record process variables in a distributed control system, digital recorders which use microprocessors are becoming more cost-effective and popular The digital recorder gathers trend data in its memory and displays the data to the operator using a liquid crystal panel or other flat display device For hard-copy output, the recorder uses an impact- or heat-type of printing mechanism instead of pen-and-ink to record the information. In some models, the recorder draws the chart scales as it is recording, so that plain paper instead of chart paper can be used. The recorder often provides such functions as automatically labeling time and range of variable directly on the chart, using alphabetic or numeric characters. Also, each process variable can he recorded using a different symbol or color to allow the operator to distinguish between the variables easily. Because of the flexibility of the printing mechanism and the memory capabilities of the recorder, intermittent printing of the process variables can supplement the display output without losing any of the stored information.

HIGH LEVEL OPERATOR INTERFACES

The high-level operator interface in a distributed control system is a shared interface that is not dedicated to any particular LCU Rather, the HLOI is used to monitor and control the operation of the process through any or all of the LCUs in the distributed system information passes between the HLOI and the LCUs by means of the shared communications facility. While the LLOI hardware resembles conventional panelboard instrumentation, HLOI hardware uses CRT or similar advanced display technology in console configurations often called video display units (VDUs). The HLOI accepts operator inputs through keyboards instead of the switches, push-buttons, and potentiometers characteristic of conventional operator interface panels Other digital hardware prints, stores, and manipulates information required in the operator interface system. In general, the use of microprocessor-based digital technology in the design of the HLOI system allows the development of a hardware configuration that departs radically from the design of panelboardbased operator interfaces This configuration provides the user with several significant advantages over previous approaches: Control room space is reduced significantly; one or a few VDUs can replace panelboards several feet to 200 feet in length, saving floor space and equipment expense. One can design the operator interface for a specific process plant in a much more flexible manner The hardware operations of panelboard design and implementation (e g , selecting control stations and cutting holes in panels) are eliminated. Instead, CRT display formats define the key operator interface mechanism. One can change these formats during startup and duplicate selected information displays wherever necessary. Using microprocessors permits cost-effective implementation of functions that previously could be accomplished only with expensive computers. These include color graphic displays that mimic the organization of the process, information presented in engineering units, and advanced computing and data storage and retrieval functions.

However, one must take great care in designing the HLOI to achieve these benefits while minimizing any negative effects or concerns on the part of the operating personnel. It became evident during the early introduction of this technology to the marketplace that operators accept properly designed HLOIs very quickly. This is especially true of younger operators who have been preconditioned and pretrained by video games and personal computers.

Architectural Alternatives

All high-level operator interface units in distributed control systems are composed of similar elements operator display, keyboard or other input device, main processor and memory, disk memory storage, interface to the shar4d communication facility; and hard-copy devices and other peripherals. However, the architectures of the various HLOIs on the market vary significantly depending on the way in which these common elements are structured.

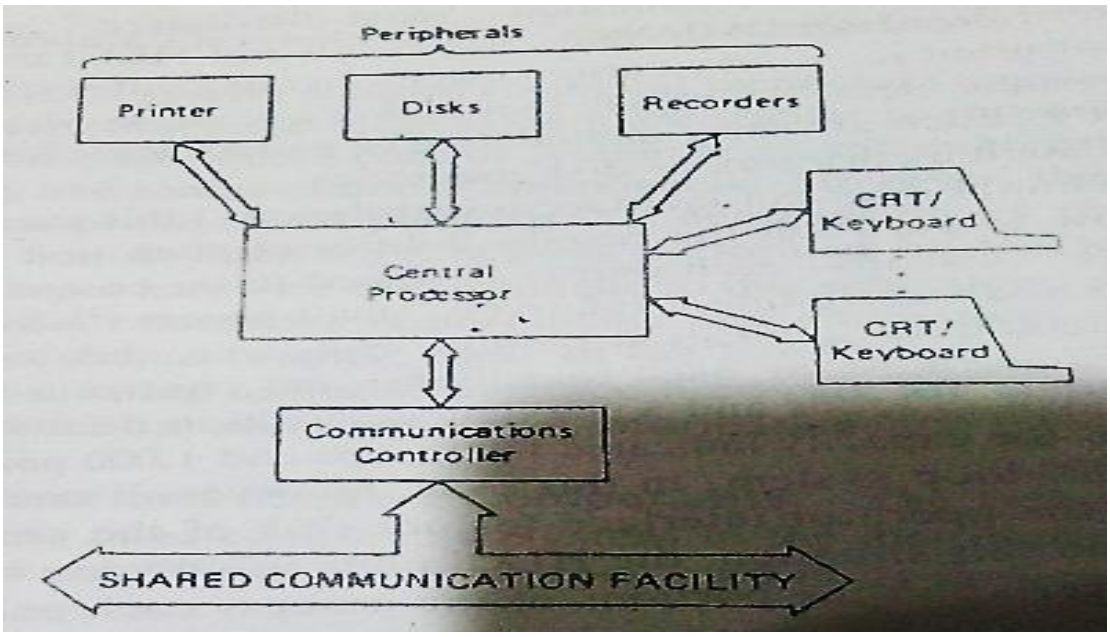


Fig. Centralized HLOI configuration

Figure shows architecture used in computer-based control systems and early distributed systems. In this architecture, there is a single central processing unit (and associated random-access memory) that performs all of the calculations, database management and transfer operations, and CRT-and-keyboard interfacing functions for the entire HLOI system. A separate communications controller interfaces the central processor with the shared communications facility. There are several advantages to this configuration. First, there is a single database of plant information that is updated from the communication system As a result, each of the CRTs has access to any of the control loops or data points in the system This is desirable, since it means that the CRTs are all redundant and can be used to back each other up in case of a failure. Other advantage is that the peripherals can be shared and need not be dedicated to any particular CRT/keyboard combination.

This can reduce the number of peripherals required in some situations. The disadvantages of this configuration are similar to those of all centralized computer system. It is an “all eggs in one basket” configuration, and so is vulnerable to single-point failures. In some cases, redundant elements can be provided; but this approach can lead to complex peripheral-switching and memory-sharing implementations. Any single-processor, single-memory configuration has limitations on the number of loops and data points it can handle before its throughput or memory capacity runs out. In many operator interface systems of this type, the display response times are long and the size of system that can be handled is severely limited. The centralized architecture is not easily scalable for cost-effectiveness: if it is designed properly to handle large systems, it may be too expensive for small ones.

Because of these limitations, most distributed control systems use a decentralized HLOI design. That is, several HLOI units provide the operator interface for the entire system. In this context, an HLOI unit refers to a single element or node that makes use of the shared communication facility. Each unit may include one or more CRTs, keyboards, or penpherals such as printers or disk memories. When the operator interface is distributed in this manner, the issue arises of how to partition the responsibilities of each unit to cover the entire process Usually, each unit is designed to be cost-effective when monitoring and controlling a relatively small process (say, 400 control loops and 1 ,000 data acquisition points) However, this means that several of these units must be used to monitor and control a larger process (say, 2,000 control loops and 5,000 data acquisition points) For example, five units of the capacity indicated (400 loops and I ,000 points) could just cover the 2,000—loop system In this case, however, if one of the control units failed, the operator interface for one—fifth of the process would be lost To avoid this situation, the HLOI units usually are configured for significant overlap in the portions of the process each unit covers.

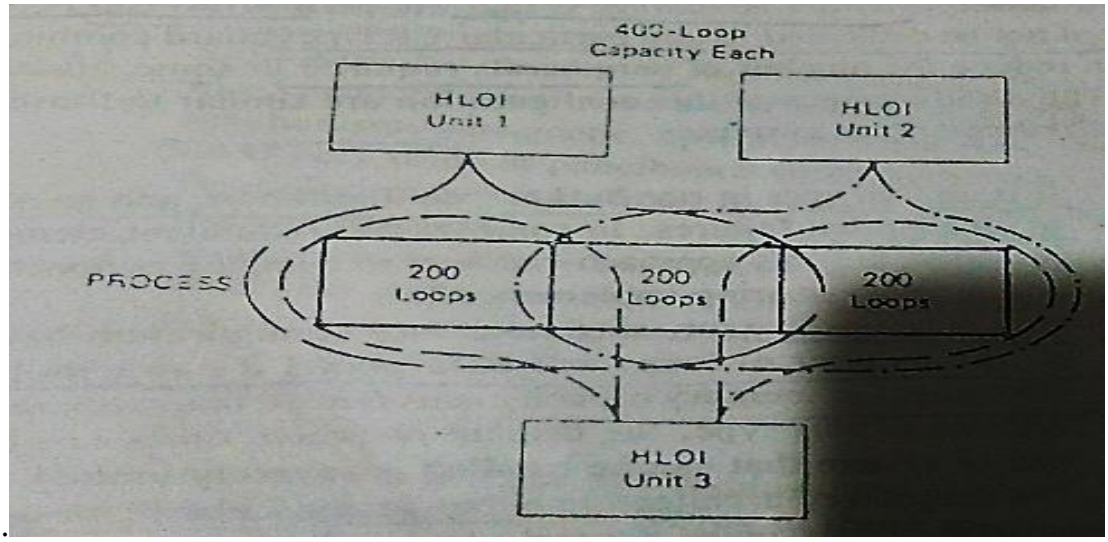


Fig. Overlap in HLOI Scope of control

Figure illustrates a two—to—one overlap configuration, in which three HLOI units control and monitor a 600—loop process With this approach, the loss of any single HLOI unit does not affect the capability of the operator interface system to control the process Overlap obviously is not an issue if each HLOI is designed to be large enough to accommodate all of the points in an installation (say, 5,000 to 10,000 points) This design approach results in a more expensive version of an HLOI than one designed to handle a smaller number of points. How ever, in this case each HLOI unit is capable of backing up any other unit in the system. The first versions of distributed HLOI systems introduced to the marketplace had a relatively fixed configuration of elements, such as that shown below.

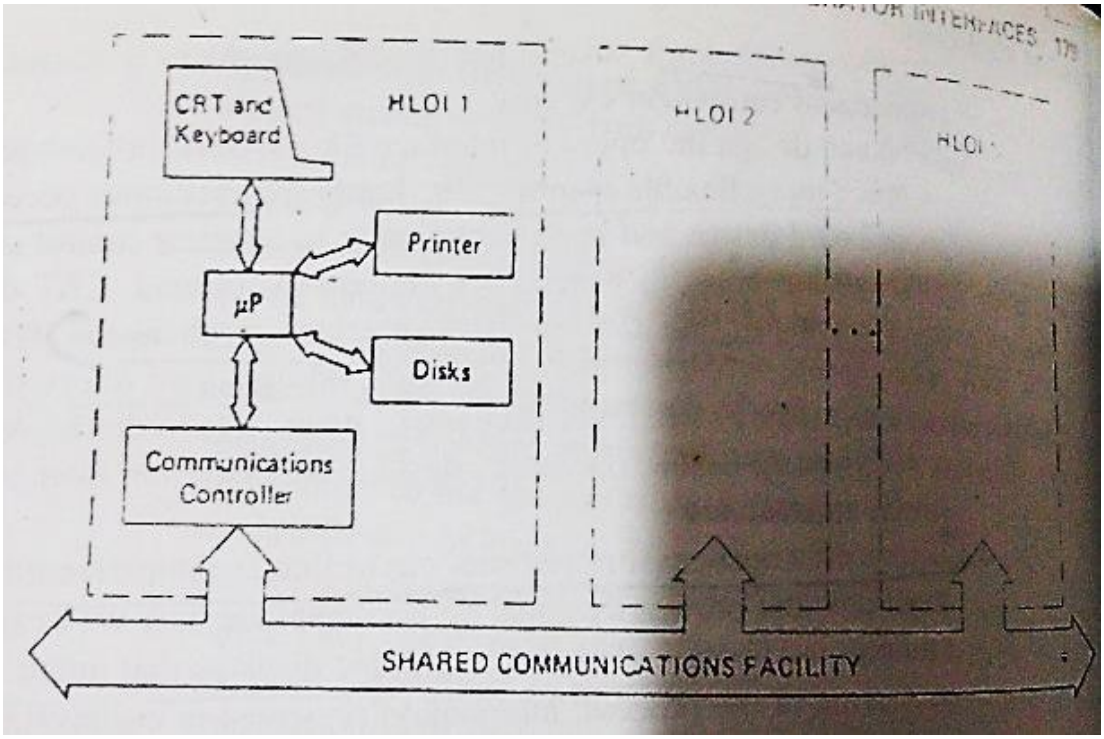


Fig. Fixed HLOI Configuration

That is, a single HLOI unit consisted of a communications controller, main processor, CRT and keyboard, and associated mass storage. The only option for the user was whether to include a printer or other hard-copy device Because of this fixed configuration of elements, the scope of control and data acquisition of the HLOI unit also was fixed. Later versions of HLOI units have been designed to be modular the user can buy the base configuration at minimum cost or expand it to handle a larger number of control loops and data points. Figure shows one example of a modular HLOI configuration.

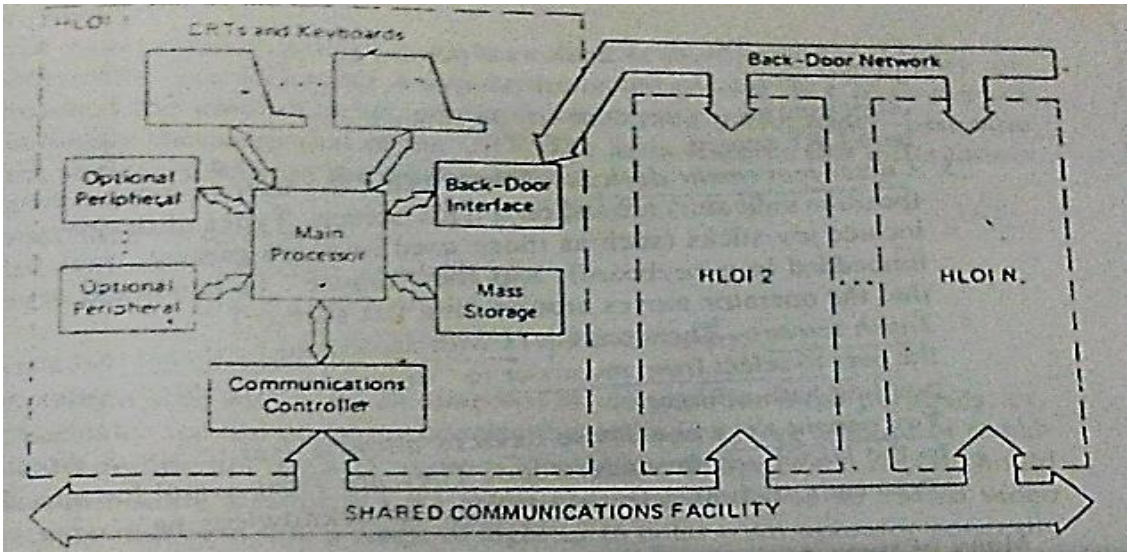


Fig. Modular HLOI configuration

The base set of hardware in this case is a communications controller, main processor, single CRT and keyboard, and mass storage unit. However, the system is designed to accommodate optional hardware such as: One or more additional CRTs to allow monitoring of a portion of the process (for example) while the primary CRT is being used for control purposes. Additional keyboards for configuration or backup purposes. Hard-copy devices such as printers or CRT screen copiers. Additional mass storage devices for long—term data storage and retrieval Interfaces to trend recorders, voice alarm systems, or other external hardware Interface ports to any special communication systems such as back- door networks to other HLOI units or diagnostic equipment Backups to critical HLOI elements such as the main processor, communications controller, or shared memory The modular approach to HLOI unit design significantly improves configuration flexibility The user can select the base configuration for small applications and add the optional hardware as the user sees fit Of course, the performance of the main processor and other hardware must be ad- equate to provide the display update and response capability required, even with the maximum size hardware configuration. 5.3.1 Hardware Elements in the Operator Interface since the HLOI system is based on digital technology, many of the hard- ware elements that go into the system are similar to those used in other portions of the distributed control system (e g, the microprocessors and the memory and communications components). However, the performance requirements of the HLOI place special demands on its elements; also, the on-line human interface functions performed by the HLOI require display and input hardware that is unique to this subsystem. Microprocessor and Memory Components.

The high-level operator interface is the most complex subsystem in the distributed control hierarchy As a result, the microprocessors used in its implementation must be faster and more powerful than microprocessors used elsewhere in the distributed system.

The microprocessor hardware selected for use in the HLOI in commercially available systems tends to have the following characteristics : It uses one or more standard 16- or 32-bit microprocessors available from multiple vendors; these are not single-sourced special components Its processors are members of a family of standard processors and related support chips It is designed to operate in a multiprocessor configuration, using a standard bus for communication between processors and an efficient real-time operating s stem as an environment for application software. The last characteristic is important, since to accomplish the desired computing and data control functions at the speeds required by the real-time operating environment, most HLOI configurations must use multiple processors. The HLOI requires the same types of semiconductor memory devices as used in the LCUs and the shared communications facility RAM for temporary storage of changing data (e g , current values of process variables), ROM for storage of predetermined, unchanging information (e g standard display formats and computational algorithms), and nonvolatile, alterable memory for storage of data that changes only infrequently (e g custom graphic display formats).

The main differences between the memory requirements for the HLOI and those for the LCU are that the HLOI requires shorter access times and greater amounts of memory to meet its high performance and large data storage requirements.

Operator Input and Output Devices

The primary function of the HLOI subsystem is to allow communications between the operator and the automatic portions of the distributed control system. Therefore, the particular data input and output devices selected are vital to the usability of the system and its acceptance by operating personnel. A great variety of operator input devices have been considered for use in industrial control systems: Keyboard—There are two kinds of keyboard in use: the conventional electric typewriter type and the flat-panel type; Light pen—This device allows a person to ‘‘draw ’‘electronically on a CRT screen. In industrial control systems, the light pen is more often used by the operator to select among various options displayed on a CRT screen Cursor-movement devices—these allow the user to move a cursor (position indicator) around on a CRT screen. Types of devices used include joy sticks (such as those used in video games), track balls (imbedded in a keyboard), and the “mouse” (a hand-held device that the operator moves around on a flat surface) Touch screens—There are CRTs with associated hardware that allows the user to select from menus or to “draw” on the screen by directly pointing with a finger. Voice-input devices—these devices allow a user to speak directly to the HLOI and be understood. They are most useful for specific commands such as selecting displays or acknowledging alarms. Many of these devices were originally developed for use in other applications, such as military or aerospace systems or in terminals for computers or computer-aided design systems In evaluating these devices, the designer of an industrial control system must remember that the needs,

background and training of an operator in a process control environment differ substantially from those of an astronaut, military aviator, or engineer. For example, the use of hand-held devices such as light pens or mice in industrial applications has been criticized on the grounds that operators prefer not to use any devices that require a separate operation— removal from or return to storage—for their use. As a result, the most popular input device technologies in industrial control systems are keyboards and touch screens. CRTs—this still is the dominant technology in the area of operator displays, CRTs come in either color or monochrome versions Flat-panel displays—these include gas plasma, vacuum fluorescent, liquid-crystal, and electroluminescent displays, most are mono chrome only. Voice-output devices—usually these are used to generate alarm messages that the operator will hear under selected plant conditions. The messages can be prerecorded on tape or computer generated by voice synthesizer.

In industrial distributed control systems, the CRT has been and continues to be the predominant visual display device used in HLOI systems. Some flat-pane1 display devices have been used in aerospace and military applications; however, in general they have not yet reached the point of development at which their performance— cost ratio seriously threatens that of the CRT The main features that differentiate the various CRTs on the market include: 1. Size of screen; 2 Number of colors available, 3 Resolution of pixels on the screen, 4 Character oriented versus bit-mapped displays.

Peripherals:

In addition to the processing, memory, input, and display devices required to perform the basic operator interface functions, the HLOI configuration must include the following additional peripheral devices to implement the full range of functions: Fixed disk drives—This type of mass memory uses non removable memory media to store large amounts of information that must be accessed rapidly (such as standard and graphical display formats) or to provide a location for temporary storage of historical data. One example of such a memory device is the Winchester disk, a magnetic memory disk with a capacity in the 5—100 Mbyte range Another is the read/write optical disk, which is used for even higher density storage (500—1 ,000 Mbytes) Removable mass memory—This type of mass memory, typified by the floppy disk, uses removable memory disks for storage of information that is to be downloaded to other elements in the distributed system (such as control configurations, tuning parameters, and custom programs) It also can be used for long-term storage of small quantities of historical data, magnetic tape can be used for large quantities Read-only optical disks, such as those in commercial video applications, are useful for storage of large amounts of unchanging information, such as text used in operator guides. Printers and plotters—.At least one printer is required to implement the logging and alarm recording functions The printer can also pro- vide a hard copy of the CRT displays, or a separate printer can be dedicated to that function A black-and-white dot matrix printer is the most prevalent type used, since it can implement either function Pen plotters, ink-jet printers, or thermal transfer printers can provide full-color hard copies of CRT displays.

Since these devices tend to be slow, it as important that the HLOI be designed so that the keyboard and CRT are active and available to the operator while the printing or plotting process is going on.

ENGINEERING INTERFACES

The functions that the plant's instrument and control system engineer performs are usually quite different and separate from those that the operator performs. This separation arises due to standard plant practices or even union contracts, which spell out the responsibilities of each group of workers. Because of this, many vendors of distributed control systems provide an engineering interface that is totally independent of the operator interface. Other vendors recognize that there are many common features and functions that both the operator and engineering interfaces require. As a result, these vendors combine the functions in a single set of hardware, but allow the hardware to take on different ‘‘personalities” depending on its current user. As in the case of the operator interface, vendors normally provide the user a choice between two levels of engineering interface hardware: Low-level, minimum-function devices that are inexpensive and justifiable for small systems High-level, full-function devices that are more powerful (and expensive), but which are needed and justifiable for medium and large sized distributed control systems.

ENGINEERING INTERFACE REQUIREMENTS

In the past, defining and setting up a monitoring and control system consisting of conventional discrete analog and sequential modules has involved a tremendous amount of engineering labor The instrument engineers had to select and procure the control and data acquisition modules, mount them in cabinets, and do a significant amount of custom wiring between modules. Then the engineers had to test and check out the entire system manually prior to field installation. Similarly, they had to select operator interface instrumentation, mount it in panel boards, wire it up, and test it. They had to prepare documentation for the entire configuration of control and operator interface hardware and the corresponding control logic diagrams, usually from manually generated drawings When errors in the control system were found, new modules had to be obtained, wiring redone, and documentation laboriously updated. A small number of general-purpose, microprocessor-based modules have replaced dozens of dedicated function hardware modules. The use of shared communication links has reduced or eliminated custom inter module wiring. Control system logic has been expressed in function block logic diagrams instead of hardware schematics CRT-based operator consoles have replaced huge panel boards filled with arrays of stations, indicators, annunciators, and recorders As a result of these simplifications in the architecture of industrial control systems, the process of engineering such a system has become simpler in a corresponding manner.

However, a significant amount of engineering work is still necessary to put a distributed control system out into the field Special hardware has been developed as a part of the distributed control system architecture to make this process as painless as possible. This engineering interface hardware must perform functions in the following categories

System configuration—Define hardware configuration and interconnections, as well as control logic and computational algorithms;

Operator interface configuration—Define equipment that the operator needs to perform his or her functions in the system, and define the relationship of this equipment with the control system itself; System documentation—Provide a mechanism for documenting the system and the operator interface configuration and for making changes quickly and easily;

System failure diagnosis—Provide a mechanism to allow the instrument engineer to determine the existence and location of failures in the system in order to make repairs quickly and efficiently.

General Requirements

The engineering interfaces in a distributed control system must be designed for the appropriate class of users of the equipment. These ergonomic requirements and several other requirements dealing with the specific functions of the engineering interface are as follows:

Access security—the engineering interface defines and modifies the control logic and tuning parameters (among other items) of the process control system. Therefore, the system must include a security mechanism to keep unauthorized users from getting access to the system configuration through the engineering interface.

Ergonomic design—the instrumentation and control engineer using the engineering interface equipment is likely to have more technical training than the process operators; however, the engineer often will not have a computer hardware or software background. Therefore, the engineering interface should be designed to accept inputs through an interactive prompt-andresponse sequence, rather than through entering of data tables or program statements. Data input formats should be in a form understandable to a process instrumentation engineer, not to a computer expert (no octal or hexadecimal. data formats, please). If the distributed control system has more than one type of engineering interface, there should be consistency of user interactions from one device to the other. Data reasonableness and consistency—As much as possible, the engineering interface system should be able to check an engineer’s inputs for reasonableness and for consistency with previously entered information.

If there is a problem, the system should advise the engineer of the situation and provide prompts that allow the engineer to correct the problem. In addition, it should not be necessary to enter the same information into the system more than once (e.g., an engineer should not have to enter an alarm limit once for alarming and again for trending). Also, the engineering interface system should keep track of the information entered to ensure consistency. If more than one engineering interface device is in use in the system at the same time, there must be a mechanism in the system to make sure that changes made by one such device are picked up by the others. Finally, if the engineering inputs for several related pieces of hard- ware and control logic must be consistent (e g, in defining parameters for a controller, an operator station, a termination unit, and a PID control algorithm), the system itself should provide a check of consistency and notify the user of any errors.

User convenience—the engineering interface devices must be designed for convenient use. For example, it should not be necessary to shut the control system down in order to connect or disconnect the engineering interface to the system. Similarly, the devices should be designed for easy storage when not in use

Cost-effectiveness—the cost to the user of the engineering interface must be small when compared to the total installed cost of the control system.

System Configuration Requirements

Setting up any process control system requires a certain amount of manual work determining control system requirements, selecting and procuring hardware, physically assembling the various modules and other elements into cabinets, and connecting external wiring between transmitters and field termination points in the control cabinets However, the introduction of distributed control systems has made it possible to automate quite a number of system configuration functions The degree of automation depends on the level of sophistication of the engineering interface equipment used. Whatever the level, the engineer must perform the following system configuration functions (as a minimum) in implementing a distributed process control system: The engineer must define the addresses of input and output points in the system with respect to their location in the shared communications facility (e.g. , by identifying the point number within the module, the module number within the cabinet, and the cabinet number within the communication hierarchy) This is equivalent to assigning telephone numbers (by area code,. exchange, and number) to people in a country so that you know where to reach them.

The engineer must define the tag names and associated descriptors that humans will use to identify certain (not necessarily all) points in the system, and relate these tags to the hardware addresses. (It is much more convenient for a human to describe a thermocouple input point as ‘‘TT 106—Reflux Temperature ‘‘than as point number 27—5—18.”). The engineer must define

any signal conditioning that needs to be done with the input points in the system: e.g., linearization, zero and span shifting, or conversion to engineering units Similar conditioning functions may need to be defined for control outputs to compensate for non linear control drive characteristics, for example. The engineer must select, configure, and tune the control and computational algorithms in the system. Data to enter and verify include PID and other control algorithms, tuning constants, alarm limits, logic sequences, batch control recipes, and high-level language statements (e g , in BASIC or FORTRAN) The engineer also must select any auxiliary functions that the system is to perform on certain points, such as trending, logging, and long-term data storage and Retrieval . When the implementation of the control and computational algorithms in item (4) requires transmitting data through the shared communications facility, the engineer must define the linkages from one element to another.

Requirements for Operator Interface Configuration

Configuring the control and computational functions of the distributed control system is only part of engineering a process control system. An equally important aspect of the job is selecting and configuring the operator interfaces to the system. The degree of support for this that the engineering interface can provide depends on how the operator interface system is implemented through panelboard instrumentation or through VDUs. Ideally, the engineering interface should support at least the following functions: It should allow the engineer to select and define the devices and mechanism the operator will use in running the process. These include hardware such as control stations, indicators, recorders, mimic panels, alarm indicators and related devices (or their equivalent in CRT displays). It should relate the operator interface devices or displays to the control and instrumentation hardware in the field. This includes labeling the devices or displays with the tags, descriptors, and engineering units appropriate to the process points being controlled or monitored.

Documentation Requirements.

Documenting the hardware configuration and functional logic diagrams of an industrial process control system is one of the most painful and costly aspects of engineering the system. In conventional control systems, almost all of this documentation is accomplished manually. Distributed control systems that include engineering interfaces can change this situation significantly In these systems the instrumentation engineer enters the bulk of the configuration information through the engineering interface and stores it in a mass data storage facility For this reason a great potential clearly exists for automating the documentation function This potential can be realized if the engineering interface meets the following requirements The engineering interface system must include a hard-copy device such as a printer to support the documentation function. The documentation system must support both tabular and graphical data formats. The formats must be adjustable so that the user can adhere to the company’s in-house documentation conventions.

One of the most difficult tasks of the instrumentation engineer is to keep track of field changes to the control system. The documentation function in the engineering interface must handle these changes as well as handle the original engineering design As much as possible, the documentation process should be automatic, requiring no special actions on the part of the instrumentation engineer. Experience has shown that documentation will not be done if it is too much trouble.

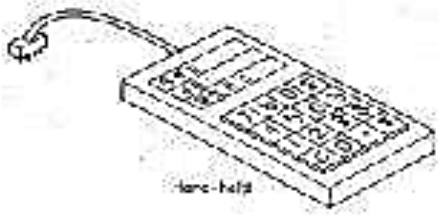
Diagnosis of System

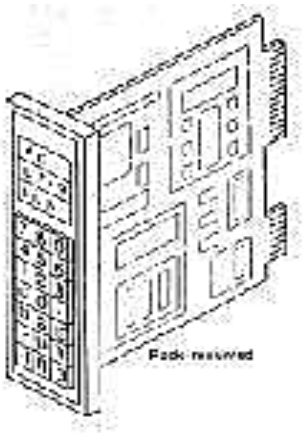
The increased intelligence of the individual devices in a distributed control system allows them to implement self-diagnostic algorithms and report any failures directly to the maintenance personnel in the plant. The engineering interface unit is one of the key mechanisms by which this failure information is reported to the instrumentation and control system engineer. The engineering interface unit can also be a very useful tool in debugging and troubleshooting control logic. To support the diagnosis of system problems, the engineering interface system must meet the following requirements:

- There must be mechanisms that allow the instrumentation engineer to identify any failed devices in the system down to the level of repairable elements These include modules, sensors, power supplies, and communication devices
- If a partial failure of an element occurs, there must be a mechanism that allows the instrumentation engineer to determine the severity and nature of the partial failure in order to identify the best course of action to solve the problem
- There must be mechanisms to expedite troubleshooting of control s stems during initial checkout as well as during on-line operation These mechanisms must be applicable to continuous, sequential, and batch control systems
- Diagnosis of process problems is not within the scope of the engineering interface system. This diagnosis is important, but it is an application-oriented function implemented within the distributed control and computing system itself

LOW LEVEL ENGINEERING INTERFACES

The low-level engineering interface is usually a microprocessor-based device designed either as an electronic module that mounts in a rack or as a hand-held portable device. To minimize cost, the device usually is designed with a minimal keyboard and alphanumeric display so that the instrument engineer can read data from and enter data into the device Some LLEIs use small CRTs or flat-panel displays to implement the human interface. Some versions of the LLEI must connect directly to and communicate with only one local control unit or data input / output unit at a time.





More sophisticated versions can connect to a local branch of the shared communications facility and are able to communicate with any one of several LCUs or DI / DOs in adjacent cabinets or areas of the plant. In either case, the LLEI can be connected or disconnected while the LCU or DI/OU is powered and in operation, it is not necessary to shut the process down Some devices include interfaces to low-cost peripherals, such as a printer (for documentation purposes), or a mass memory device, such as a tape cassette or microfloppy diskette (for storage of configuration data)

In general, the LLEI is a dedicated device that is not used for operational purposes Therefore, when it is not needed for engineering use, it can be disconnected from the distributed control system and locked up for safe- keeping. This keeps the control configuration and tuning constants secure from alteration by unauthorized personnel. Because of the minimal display and user input capabilities built into the LLEI, the interaction between user and device is often not as user-friendly as it is between user and high-level device. There is usually heavy reliance on using code words, symbols, or dedicated function keys to implement man-machine communications in the low-level device The engineer generally has to document the input data on configuration forms first, then enter the data through the interface However, the entry of data can be designed to take place in an interactive request-and-response manner.

The interface device can also make some basic reasonableness and consistency checks on data entries and input sequences and give the engineer error messages when it detects a problem. The displays and input keyboard can be made flexible enough so that instrument engineering personnel can enter and read out data in engineering units that they understand.

System Configuration

When the system provides only an LLEI, the hardware in the system is selected and configured manually. To simplify this task, most vendors of distributed systems provide a systems engineering guide or similar document that leads the user in a step-by-step manner through the hardware configuration procedure

In this situation, the primary purpose of the LLEI Is to provide a tool for configuring the algorithms in the system controllers (i e , entering control strategies, setting tuning parameters, selecting alarm limits, and so forth). Some low-level interfaces have a removable mass memory device (such as a small cassette, diskette drive, or magnetic strip) incorporated in their designs for storage of control configurations When connected to a power supply, the engineer can then use the interface to develop and edit control strategies even if the controller itself is not connected All that is required is that the configuration limitations of the target controller be stored in the engineering interface device. The availability of mass memory also makes it possible to download control configurations into the target module from the engineering interface, or conversely to "upload" the configurations from the controller. This function is very convenient when it is necessary to install a control configuration in a spare controller used to replace a failed controller. In general, the LLEI is not designed to support generation and editing of high-level language programs such as BASIC and FORTRAN. A separate terminal or HLEI is required to provide this capability.

Operator Interface Configuration

The distributed control systems that require only a low—level engineering interface generally are small ones involving a limited number of control loops In this type of system, the operator interface also is simple, usually consisting of a small number of dedicated panelboard instrumentation devices (with no shared CRT-based console). Therefore, all configuration of the operator interface is done manually.

As in the case of all panelboard instrumentation, the connections between the stations and indicators and the controllers in these small systems are established through the manual process of hard wiring or cabling. Assigning tags to the control loops and system inputs and labeling the stations and indicators with these tags and the corresponding engineering units also are manual operations. These tasks can he very time consuming and prone to error. Also, changes are difficult to make, since there is a significant lead time involved in procuring the new panelboard hardware, cabling, and labels needed to implement the changes.

Documentation

In a small system that requires only a low-level engineering interface, the vendor usually provides the user with very little or no help in automating the process of documenting the hardware and control configurations All documentation is a manual process, sometimes with the help of standard forms in the system engineering guide When changes are made to a system of this type in the field after initial start-up, documenting these changes is a manual process that can be hazard. It requires great discipline for the user organization to ensure that these field changes are tracked and identified accurately.

Diagnosis of System Problems

The LLEI, since it is not always connected to the distributed control system and often is not even located in the control room, provides little help in the way of notifying the operator of hardware failures in the control system. The operator must rely on the self-diagnostic capabilities of the distributed equipment itself to detect and make known the failure to the operator. Failures of this equipment generally are indicated at the control cabinets through alarm lights on the individual pieces of equipment. The failures are alarmed in the control room either through a conventional annunciator panel or through indications on the panelboard instrumentation. However, the LLEI can be very useful in helping the instrumentation and control system engineer identify the specific location and nature of the failure. For example, the LLEI can interrogate a controller or other element that has failed; the element, if it still can communicate, can report on the specific problem that caused the failure through the engineering interface Also, the instrument engineer can use the LLEI as a digital voltmeter to trace through the control logic and identify problems. Some LLEIs are equipped with a verify function that allows an automatic comparison of the actual configuration in the controller with the correct one stored in the interface's mass memory. This function can be very helpful in checking out a controller during initial start-up and in making sure the configuration hasn't changed after it has been in use in the field for some time.

HIGH LEVEL ENGINEERING INTERFACES

The high-level engineering interface allows a user to reap the full benefits of the flexibility and control capabilities of a distributed control system while minimizing the system engineering costs associated with such a system. The HLEI is implemented in the form of a CRT-based console, or VDU, similar to the high-level operator interface unit. The internal architecture of the VDU is modular, using multiple microprocessors. This approach provides a significant amount of flexibility in accommodating hardware options for engineering purposes (e.g., special keyboards or printers).

Dual-Console Functions:

In a few distributed systems, the engineering console is a specialized device that is dedicated to the engineering function. In most offerings, however, the engineering console can also be used as an operator's console; a key lock on the console implements the switch between the two console "personalities The first position permits only operator functions (e.g. , display selection, control operations such as mode selection and set-point modification., and trend-graph selection). The second position allows engineering functions (such as control logic configuration, modification and tuning, and system documentation) as well as operator functions.

Some systems provide a third key-lock position that allows the user to perform tuning operations but does not allow the user to modify the control logic structure Implementing dual console ‘personalities” in a single piece of hardware is very cost-effective for the user, since the engineering functions are needed only during initial system start up and occasionally thereafter when system modifications are made. During the other periods of time, the hardware is put to good use as an operator s console. However, the console is always connected to the system and is conveniently available when engineering functions need to be performed. The incorporation of a key lock ensures that the system configuration is secure while allowing authorized plant personnel access to the system.

Special Hardware Required:

The engineering functions are much more oriented towards data entry and documentation than are operator functions Because of this, the hardware requirements for an engineering console are somewhat different than those for a console used strictly for operations. One major difference is in the keyboard provided for the user The operator’s console uses a flat-panel, dedicated-function keyboard for ruggedness and simplicity of operation The engineer’s console, however, requires a general purpose keyboard to promote speed of data entry and to support wider range of human interface functions.

This keyboard usually is a push-button type whose keys are laid out either in the traditional QWERTY configuration found on most typewriters or in the more recent Dvorak configuration (a human-engineered version that repositions the letters to promote faster touch typing). The vendor may supply additional special-purpose keys to allow the user to select special characters, symbols, and colors employed in generating control configurations and displays. To further simplify the process of building custom color-graphic CRT displays, some vendors also provide a digitizer tablet and stylus, a light pen, or other device as auxiliary hardware for the engineering console. (A digitizer tablet is a touch-sensitive pad on which the engineer enters information using the stylus) This hardware supplements the cursor keys, touch screens, and other human interfacing hardware supplied with the operator’s console. When the plant operator is using the console, the special keyboard or auxiliary hardware usually is stowed away in a storage location.

Only when the console’s personality is changed to become an engineer’s console is this hardware brought out In addition to this auxiliary hardware, some vendors also supply special color-graphic printing or plotting devices, which allow more sophisticated system documentation to be generated than is possible with the standard printer that comes with an operator’s console.

Portable Engineering Interface:

Some distributed control system vendors offer a CRT-based engineering interface device that is a compromise between the full-function engineering console and the minimum-function low-level device. Usually, this is a portable unit that includes a bulk memory device such as a

floppy disk or cassette tape drive for storage of system configuration data. This unit generally is designed to plug into and interface with a single LCU or cab- met It can be very useful and cost-effective device for certain system configurations. However, its design will not be discussed here since its functions are a subset of those implemented in the HLEI console.

System Configuration

Because of its computational, display and data storage capabilities, the HLEI can play a major role in automating the process of configuring a distributed control system. The entire database that defines the configuration of the hardware, control structures, and computational algorithms of the distributed system can be stored in the HLEI.

The following information dealing with the system hardware configuration can be entered by means of an interactive dialog between the engineering interface and the instrument engineer:

- Number, type, and location (relative to the shared communications facility) of each hardware module in the LCUs and DI/OUs in the distributed system;

- Definition of any hardware options (such as expansion hardware, jumper positions, or switch settings) selected on each module;

- Definition of each input point (by type and hardware address) in each of the hardware modules;

- Number, type, and location of all operator and engineering consoles in the system;

Number, type, and location of any other devices in the system that communicate using the shared communications facility (e.g., computer interfaces, and special logging or computing devices). While the instrumentation engineer enters most of this information manually, the HLEI can acquire some of the data (such as the addresses and types of devices using the shared communications facility) automatically. It can do this by means of a sequence of broadcast messages through the communications facility asking any devices that are listening to identify themselves and transmit any configuration information that they have When implemented, this approach can save a substantial amount of the engineer' s time and effort and cut down dramatically on the number of configuration errors introduced.

In either case, once this information is entered in the system, the engineer uses the HLEI to make reasonableness and consistency checks on the information, save it on a mass memory medium, and document it in a printed format. During normal operation, this information is not used anywhere else in the distributed system. It is saved to document the hardware configuration installed in the plant for the instrument engineers' use and for downloading in case of hardware failure and replacement.

In addition to this hardware-related information, the engineering interface is also used to store information dealing with the control and computational functions of the system. At a minimum this information includes the following Point tags and associated tag no, engineering unit definitions , and related point addresses, Logic state descriptors (e g , on/off, and run/stop) that correspond to digital points in the system

Control algorithms implemented in each of the LCUs, Signal conditioning and linearization algorithms implemented in the DI/OUs, Communications linkages needed to transfer information among the control and computational algorithms in different LCU s and DI/OUs in the system using the shared communications facility; High-level language computational algorithms implemented in the LCUs.

Control and Computational Logic Configuration

One of the major benefits of using an HLEI in a distributed control system is that it simplifies the process of configuring the control and computational logic in the system. The high-level console provides one or both of the following control logic configuration capabilities:

- A fill-in-the-blanks function in which the instrument engineer interacts with the console through a sequence of prompts and responses to configure the control logic in the system
- A graphics capability in which the instrument engineer draws the control configuration on the screen with a light pen or on a digitizer tablet with a stylus This capability is similar to that provided in computer-aided-design (CAD) systems, and allows the user to define the control logic using a familiar function block format.

After laying out the basic control structure graphically, the instrument engineer can define the parameters associated with the function blocks through an interactive sequence.

Storage of Configurations As described previously, one of the basic functions of the HLEI is to provide the engineer with a tool for configuring the control logic and computational algorithms in the target hardware that will execute the algorithms. However, the HLEI can be used to perform additional useful functions if it includes a mass memory facility that allows duplicate storage of this configuration information. The mass memory facility is generally implemented using one or more types of storage media: floppy disk, hard disk, magnetic tape, bubble memory, or optical disk.

The additional functions that configuration storage allows include the following:

- The engineer can use the interface to configure the control logics and computational algorithms without the presence of the target LC1 In this situation, the configurations are stored directly into the mass memory of the interface. Of course, the interface must be aware of the limitations of the target LCU (in terms of memory size and co4utational speed, for example) and ensure that the configurations entered do not exceed these limitations. After storage, the engineer can then download the configurations at a later time to the LCUs once this hardware becomes available This feature can dramatically cut the time necessary to deliver a control system, since the job engineering can begin and the control logics can be configured and stored long before the actual target hardware is available.
- The HLEI can be used to verify that the engineer has correctly entered the control logic and algorithms into the LCU and that they have not been altered; the HLEI does this by performing a bulk comparison of the logic in the LCU with that stored in the mass memory Any mismatches are displayed for the engineer, who can correct them immediately If an LCU or other element of the system fails, it can be replaced by a new element. The appropriate control and computational algorithms can then be downloaded from the mass memory to the new or repaired element. This results in a very short mean time to repair for that element and a minimum downtime of the function performed by that element.
- The engineering interface also should support the uploading of database information from a hardware device to the interface this capability helps in documenting functions performed by modules or elements that are removed from service or moved from one control system to another.

Operator Interface Configuration

When a high-level operator interface is used in the DCS, there must be a mechanism that allows the instrument engineer to configure or change its display structures and parameters in a convenient manner. In most DCS, this mechanism is the high level engineering interface. This device allows the engineer to change the layout of the series of displays that make up the ‘‘panel board ’‘used to control plant operations.

For maximum flexibility, the engineering interface must be designed in such a way that it can configure any HLOI in the DCS. That is, the system should be able to download displays and display hierarchies configured on one engineer's console to any other operator's console in the system over the shared communications facility. This capability implies that displays configured for one operator's console can be duplicated on other consoles without the need for people physically transporting magnetic storage media (e.g., floppy disks) from one console to another.

The first step in configuring an HLOI is to structure a hierarchy of displays. This includes specifying the following information as a minimum:

- Number of areas in the plant and their identifying tag names and descriptors,
- Number of groups in each area and the tag names and descriptors for each area,
- Assignment of control loops and input points to a group or to multiple groups,
- Types of displays to be used at each level, whether preformatted displays or custom graphics;
- Linkages between displays in the hierarchy (i e , the paths that the operator must follow in moving from one display to. another; these linkages are pre specified in some systems and configurable in others,
- Assignment of points in the system to the special functions of the operator console trending, alarming, logging, and long-term data storage and retrieval, specification of the parameters involved in each special function (e g , trend periods, logging formats, and data storage frequency)
- Defining and laying out graphics displays is a more challenging task, since they are usually completely user-configurable and do not follow any standard format. Of course, the vendor's display hardware imposes limitations on the structure of these displays, such as resolution, number of colors available, pixel control method (character oriented or bit-mapped), special characters and symbols, and so forth.

Usually, the vendor provides a graphics display engineering kit that spells out these limitations and helps the user to sketch each display on a special grid form before entering it into the engineer's console In laying out each display, the engineer must enter the following information as a minimum:

Graphic symbols:

The engineer must define the new graphic symbols to be used in the operator interface displays Usually, the system vendor supplies a base set of symbols that represent standard items of process equipment These include symbols for piping, valves, pumps, and so forth The ISA standard provides a typical list of these symbols However, each user generally has a need to configure additional symbols that are appropriate to the particular application The symbol library is expanded to include these new symbols along with the standard ones, they are then also available for incorporation in the user's displays.

Static background elements:

These are elements in the display (both graphic and alphanumeric) that do not change with time These usually include the major items of process equipment, the interconnecting piping, the equipment labels, and engineering unit identifiers The information required for each element includes its shape, size, and color

Dynamic graphics elements:

These are the graphics elements in the display that change to reflect corresponding changes in the process. For example, pumps may change color when they are turned on and off, piping may change color when it contains fluid, the displayed levels of fluid in tanks may move up and down to reflect actual fluid levels, bar graphs may change to depict temperature profiles. Or the actual symbols may change For example, one symbol may represent a closed valve and another symbol an open valve In each case, the dynamic element must be linked to the corresponding tag that defines the process state represented by the graphics element For instance, the engineer can link a tag representing a digital process input to the graphics element illustrating an open or closed valve. In addition to the tag definition, the engineer must specify how the dynamic element will change (e g, in shape or color) in response to the change in tag state and enter this information into the definition of the display.

Dynamic alphanumeric elements:

The engineer must also define and link these to the corresponding tags. These elements include alphabetic labels that change color, wording, character size, or brightness as a function of process conditions. They also include numeric values or logic states that the VDU updates to reflect changes in the process variables represented by the corresponding tags. Alarm conditions can be indicated by color changes or by having the VDU place the element into a blinking condition. Usually, there are limits to the number of dynamic elements that a single graphics display can in— dude, 200 to 400 elements is a typical range

Control stations:

If the graphics display structure supports continuous or sequential control operations (or both), the engineer must define the controllable elements in the graphic display and label them with tag numbers, reference numbers, or both. This allows the operator to call up the appropriate control stations on the graphics display and connect them to these controllable elements

Poke points:

If the operator’s console provides a touch screen input capability, the display specification will include the definition of poke points or fields that will be sensitive to operator inputs. As in the case of the dynamic display elements, the engineer must link each poke point to the corresponding tag that represents a push- button input from the operator.

Documentation:

The high-level engineering interface can provide the user significant benefits in the area of documentation. After the configuration process has been completed, all information defining the hardware, control logic, computational algorithms, and displays for the system is stored in mass memory in the engineering interface. This makes it possible to completely automate the process of documenting this information in a hard-copy format. Ideally, the intelligence and printing and plotting hardware in the HLEI should be designed to generate the following documentation automatically:

- Lists of hardware modules in the distributed system, including their location by number of LCU or DI/OU.
- Documentation of the control configuration and associated tuning parameters for each LCU, in both listing and graphic formats showing the control system structure, listing includes any high-level language programs or batch control recipes executed in the LCUs.
- Listing of the tags in the distributed system, along with the associated tag descriptor information and the hardware addresses of the physical inputs or module outputs that correspond to these tags.
- Listing of the special operator interface functions that are associated with each tag, such as trending, logging, and long-term storage and retrieval functions.

Definition of the operator displays in the system, including a drawing of the display hierarchy, a listing of the displays in the hierarchy, and a printout of the formats of each display.

Diagnosis of System Problems:

Most of the hardware is microprocessor-based and has the intelligence to perform on-line selfdiagnostics to evaluate its own "health." When a failure or other problem occurs, this hardware is aware of it and reports the problem to the higher-level elements in the distributed system.

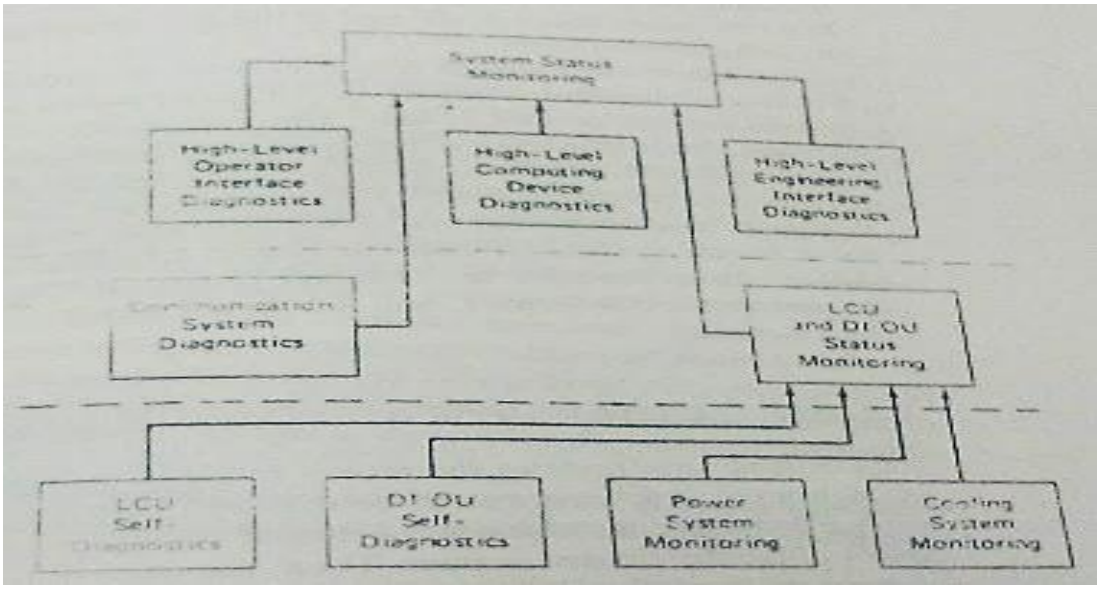


Figure: Typical Diagnostic Hierarchy

Figure shows a typical diagnostic hierarchy. At the lowest level, the LCUs and DI/OUs perform diagnostics, and report the results up through the shared communications facility The communication system elements also check their own ‘‘health’ and provide status reports to the system-level elements in the distributed system the high level operator interfaces, engineering interfaces, and computing devices. Finally, these system-level elements also execute their own online diagnostics and report the results to the other system-level elements in the system. The overall diagnostic status of the equipment in the system is indicated on the top line of each display in the HLOI system. If a problem develops, an equipment status alarm goes off and the operator or instrument engineer can call up a hierarchy of diagnostic displays to pinpoint the problem.

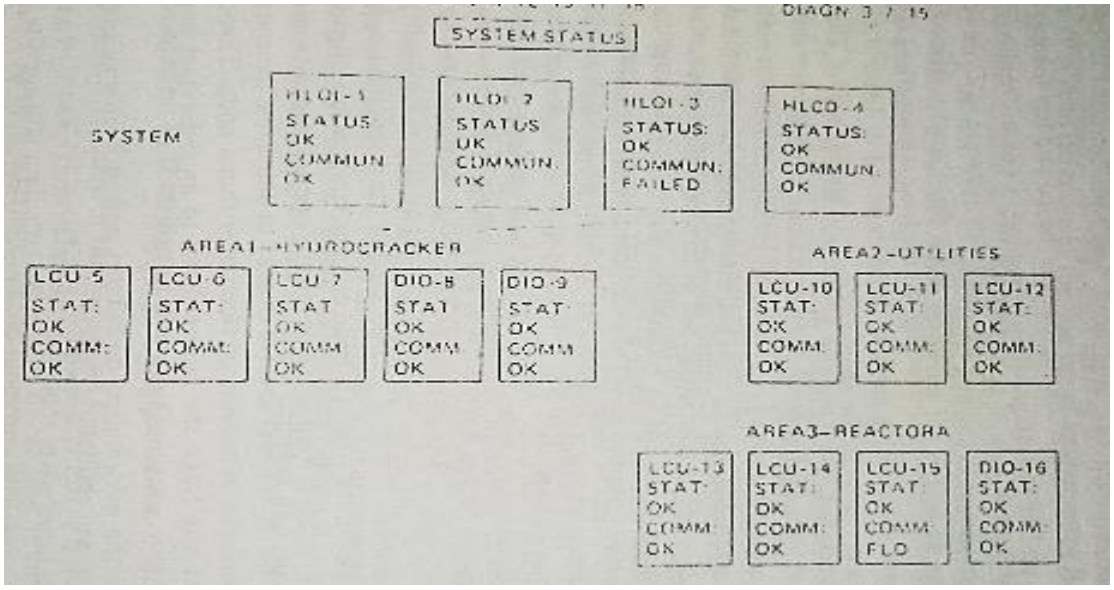


Figure: Example of a Diagnostic Display

Figure shows an example of the highest level of display in this hierarchy This display shows a map of the nodes or drops on the shared communications facility, these contain the major hardware elements that exist in the distributed system Any failure in the elements within a node are reported on this display If a failure or problem is identified at this level the operator or instrument engineer can call up the next lower level of display to determine the cause of the problem in the individual module or element within a node Through this method, the user can trace the failure down to an individual hardware element, which then can be replaced or repaired. Since the information on failures and problems is available within the distributed system through this diagnostic hierarchy, it can also be logged on a printer and saved in a long-term data storage file. This allows the user to accumulate statistics on the rates and modes of failure of the various pieces of hardware in the distributed system. This information is essential to planning a cost effective strategy for maintaining a spare-parts inventory and in providing inputs to an overall program for managing plant maintenance.

IMPORTANT QUESTIONS from UNIT - 5

Part-A

1. What is meant by geographically centralized and geographically distributed control system?
2. What is functionally distributed control system?
3. What are the responsibilities of a plant operator?
4. Mention the requirement of operator interface.
5. What are the motivations for using LLOI?
6. Mention some of the devices used for LLOI interface.
7. Explain about different types of operator display.
8. Compare LLOI and HLOI. 9. What is a Plant level display?
10. What you mean by area level display?
11. What are the requirements of engineering interface?
12. What is a Low Level Engineering Interface?
13. Mention some of applications of DCS.
14. What is the need of General purpose computer in DCS?
15. Define operator interface.
16. Mansion the significance of computers in Modern control system
17. What are the common input output devices used for interfacing with computers
18. Write the features present in operator level interface
19. Mention the protocol to be used for communication between a PC Drive/operator Pannel.
20. What are the main responsibilities of a Low level operator?

Part-B

1. Explain in detail about operator interfaces and its requirements.
2. Explain in detail about low level operator interfaces.
3. Explain in detail about low and High level operator interfaces in DCS
4. Explain in detail about High level engineering interfaces.
5. Explain in detail about low level engineering interfaces.
6. Explain General purpose computers in DCS.
7. Explain about the operator displays.
8. Mention the adv of low level and high level operator interfaces. Also explain the importance operator display used in process industry
9. Mention the role of general purpose computer in DCS; also evaluate low level and high level engineering interfaces
10. What are the different types of displays in DCS? Explain with examples.