Chapter 8 :   Programmable Logic Controller (PLC)

8.1 The Structure and Features of Programmable Logic Controller

Programmable logic controllers (PLCs) have been used in industry in one form or another for the past twenty over years. The PLC is designed as a replacement for the hard-wired relay and timer logic to be found in traditional control panels, where PLC provides **ease and flexibility of control** based on programming and executing logic instructions. The internal functions such as timers, counters and shift registers making sophisticated control possible using even the smallest PLC.

The structure of a PLC can be divided into four parts. They are **input/output modules**, **central processing unit** (CPU), **memory** and **programming terminal.**

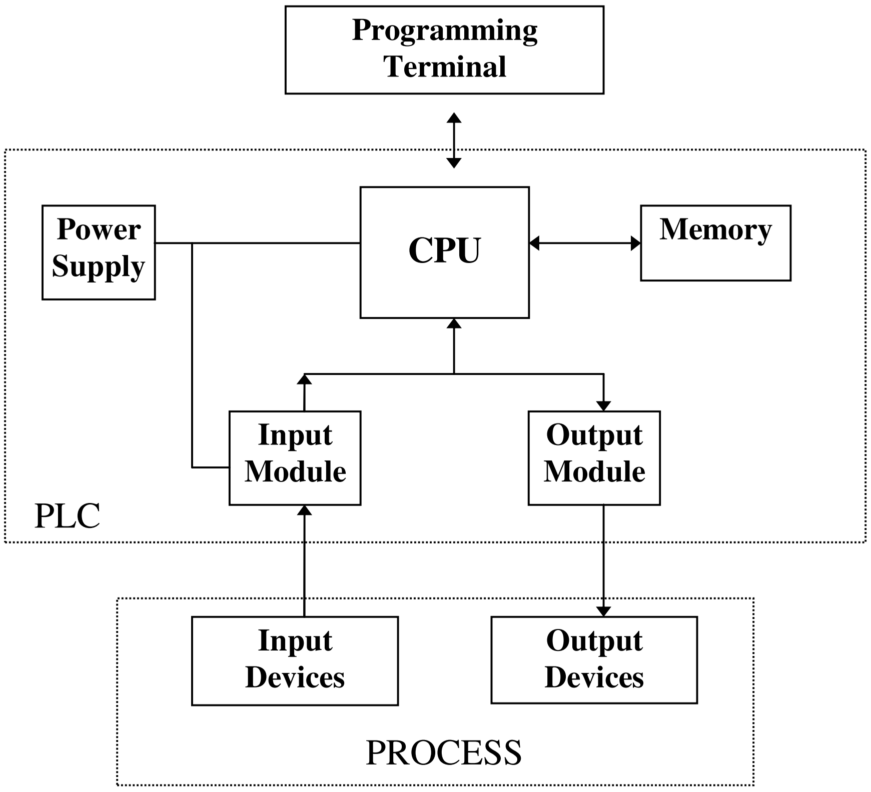


Fig. 1 :   **Programmable logic controller (PLC) structure** ❋

A programmable controller operates by examining the input signals from a process and carrying out logic instructions (which have been programmed into its memory) on these input signals, producing output signals to drive process equipment or machinery. Standard interfaces built-in to PLC allow them to be directly connected to process actuators and transducers without the need for intermediate circuitry or relays.

PLCs require shorter installation and commissioning times than do hard-wired systems. Although PLCs are similar to 'conventional' computers in term of hardware technology, they have **specific features suited for industrial control**: ✳

(a) Rugged, noise immune equipment;

(b) Modular plug-in construction, allowing easy replacement or addition of units (e.g. input/output);

(c) Standard input/output connections and signal levels;

(d) Easily understood programming language;

(e) Ease of programming and reprogramming in-plant;

(f) Capable of communicating with other PLCs, computers and intelligent devices;

(g) Competitive in both cost and space occupied with relay and solid-state logic systems;

These features make programmable controllers highly desirable in a wide variety of industrial-plant and process-control situations.

8.2 PLC in comparison with other control systems

Relay Control System

By connecting the input and output contacts in series and/or parallel, any desired logic functions may be produced. Combinations of various logic elements may be used to create fairly complex control plans. For a simple task, the number of control relays required could be so numerous that it can result in a large control panel. A typical relay system may consist of several hundred or thousand switching contacts, which presents the design engineer with a considerable task. It is also extremely difficult to change the control function of a panel once it has been wired up, and is likely to involve a complete re-wiring of the system. Together with the other disadvantages of cost, speed and reliability, the above drawbacks for relay control system have led to the replacement of relay control systems by modern alternatives based on electronics and microprocessors. Relay continues to be used extensively as output devices (actuators) on other types of control system, being ideal for the conversion of small control signals to higher-current/higher-voltage driving signals.

Digital Logic Control Systems

Digital ICs, which deal exclusively with binary signals, process this information through various logic 'gates'. Logic gates operate at much higher speeds and consume considerably less power than an equivalent relay circuit. Although digital ICs have the advantage of small size, *it cannot switch higher power signal.* Relay is used to convert small control signals to higher power driving signals.

Electronic Continuous Control System

The operational amplifier (op-amp) available for analog computing operations, which involve the performance of mathematical operations such as integration, differentiation, etc., were quickly adopted into the field of continuous control (Closed-loop feedback systems) and provided a much simplified solution to complex control functions compare with existing discrete electronic systems. Analog control is now heavily based on linear integrated circuits, and remains the fastest form of control available.

However, the 'fine tuning' of feedback systems during design and commissioning remains a difficult task. This, coupled with the fixed nature of electronic circuit construction, results in a control medium that cannot easily have its function changed - the complete electronic system may have to be replaced if this proves necessary.

Computer Control System

Today, powerful low-cost micro- and mini- computers are available, and are often used in both sequence and continuous control systems. Microprocessor-based control panels are small enough to locate at (or near) the point of final control, simplifying connection requirements.

In large processes it is now common for several microcontrollers to be used instead of a single large mainframe control computer, with resulting benefits in performance, cost and reliability. Each micro can provide optimal local control, as well as being able to send or receive control data via other microcontrollers or a **host supervisory computer** (mini or micro). This is termed **distributed control** and allows for greater sophistication of control than was with a centralized strategy using a single large computer, since the control function is divided between several dedicated processors.

A distributed control hierarchy need not consist exclusively of mini- and micro-computers, but can include other intelligent devices such as CNC machines, robots and programmable logic controllers.

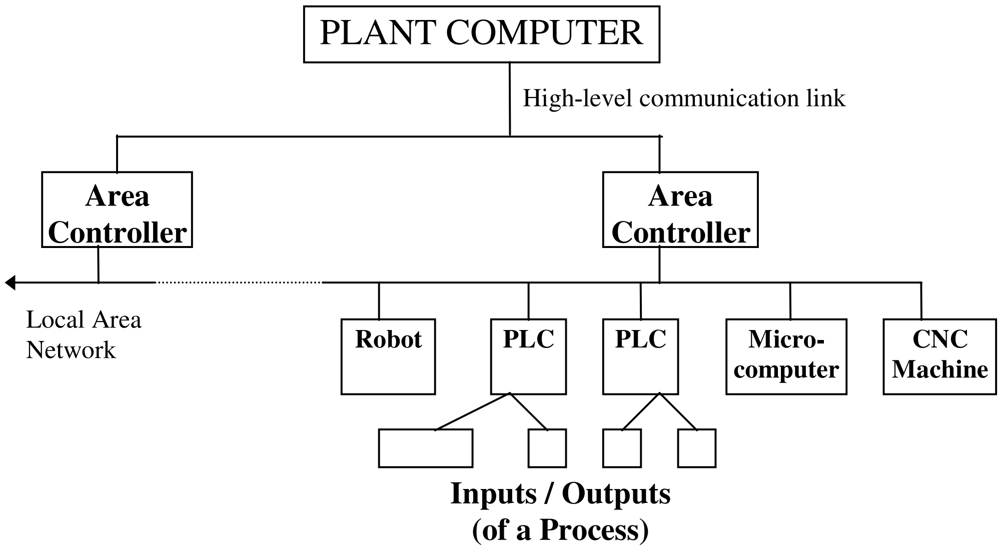


Fig. 2 : **Distributed control structure**

Current trends in computer control

The availability of powerful, low-cost personal and industrial microcomputers based on the IBM PC standard has led to the development of a wide range of add-on interface boards that equip the PC to act as an effective industrial controller. This has resulted in increased use of such systems for many control applications.

The range of interfaces includes multi-point digital I/O cards, analog I/O and communications cards. These are fitted to the chosen personal computer or to a 'ruggedized' industrial version, both machines having several expansion slots that are use to hold up to six interface cards.

Appropriate software applications packages are then loaded and run on the system. This software handles the configuring of the I/O cards, together with data processing required to carry out the control plan. There is frequently a very 'friendly' operator interface via a dynamic graphics display. This presents process data to the operator, including system status and exception conditions. It may allow limited operator intervention in certain circumstances.

The following table provides a general comparison between various control media in terms of the capabilities. Any further technical information should be obtained from the respective manufacturer's data sheets on each specific system.

Characteristic	**Relay systems**	**Digital/Analog logic**	**Computers**	**PLC systems**
Price per function	Fairly low	Low	High	Low
Physical size	Bulky	Very compact	Fairly compact	Very compact
Operating speed	Slow	Very fast	Fairly fast	Fast
Electrical noise immunity	Excellent	Good	Quite good	Good
Installation	Time-consumin design and insta	Design and test f tuning time-consuming	Programming extremely time-consuming	Simple to progra and install
Capable of complicated operations	No	Yes	Yes	Yes
Ease of changing functions	Very difficult	Difficult	Quite simple	Very simple
Ease of maintenance	Poor- large number of contacts	Poor if ICs soldered	Poor- several custom boards	Good- few standard cards

Table 1 : **Comparison of control systems**

Programmable logic controllers emerge from the comparison as the better overall choice for a control system.

If the ultimate in operating speed or resistance to electrical noise is required, hardwired digital logic and relays are chosen respectively.

For handling complex functions a conventional computer is still marginally superior to a large PLC equipped with relevant function cards, but only in terms of creating the functions, not using them.

Here the PLC is more efficient through passing values to the special function module, which then handles the control function independently of the main processor - a multiprocessor system.

8.3 PLC - Hardware

Programmable logic controllers are purpose-built computers consisting of three functional areas: *processing*, *memory* and *input/output*. Input conditions to the PLC are sensed and then stored in memory, where the PLC performs the programmed logic instructions on these input states. Output conditions are then generated to drive associated equipment. The action taken depends totally on the control program held in memory.

Central Processing Unit (CPU)

The CPU controls and supervises all operations within the PLC, carrying out programmed instructions stored in the memory. An internal communications highway, or *bus system*, carries information to and from the CPU, memory and I/O units, under control of the CPU.

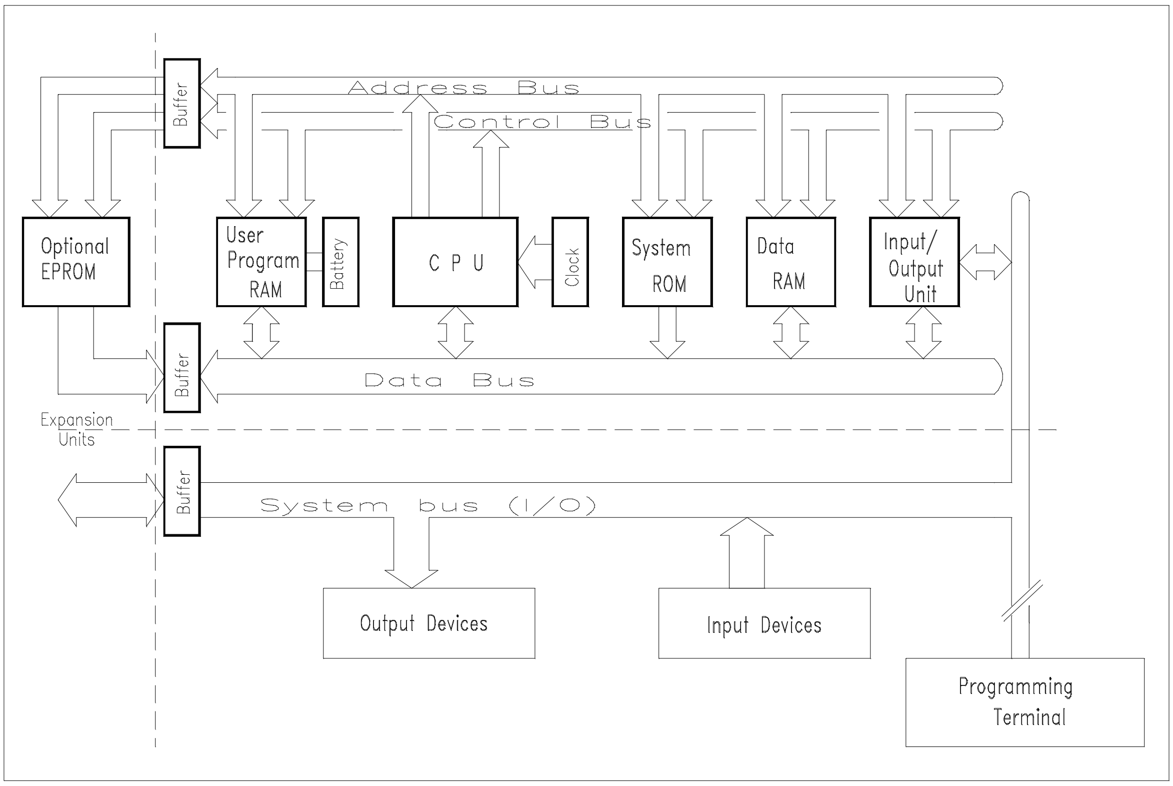


Fig. 3 Block diagram of PLC CPU architecture

Virtually all modern PLCs are *microprocessor-based*, using a 'micro' as the system CPU. Some larger PLCs also employ additional microprocessors to control complex, time-consuming functions such as mathematical processing, three-term PID control, etc.

Memory ✱

All PLCs contain both RAM and ROM in varying amounts depending upon the design of the PLC. The use of a 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.

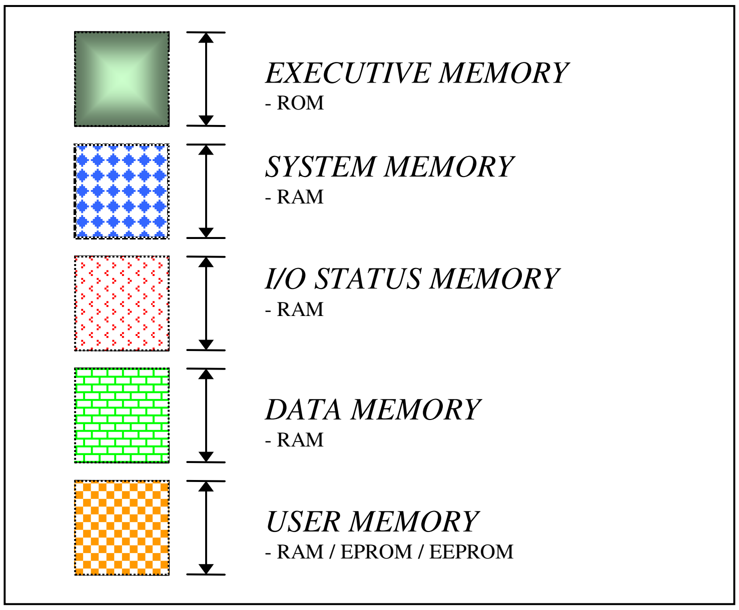


Fig. 4 : **Typical PLC memory utilization map** ✱

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 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.

Input/Output Module Units

The input/output unit of PLCs handles the job of interfacing high power industrial devices to the low-power electronic circuitry that stores and executes the control program.

Most PLCs operate internally at between 5 and 15V d.c. (common TTL and CMOS voltages), whilst signal from input devices can be much greater, typically 24V d.c. to 240V a.c. at several amperes.

The I/O module units form the *interface* between the microelectronics of the programmable controller and the real world outside, and must therefore provide all necessary signal conditioning and isolation functions. This often allows a PLC to be *directly connected* to process actuators and input devices without the need for intermediate circuitry or relays.

To provide this signal conversion, programmable controllers are available with a choice of input/output units to suit different requirements. For example:

Input	**Output**
5 V (TTL level)	24 V 100 mA dc
24 V dc/ac	110 V 1 A ac
110 V ac	240 V 1 A ac (triac)
240 V ac	240 V 1 A ac (relay)

It is standard practice for all I/O channels to be electrically isolated from the controlled process, using opto-isolator circuits on the I/O modules. An opto-isolator allows small signal to pass through, but will clamp any high-voltage spikes or surges down to the same small level. This provides protection against switching transients and power-supply surges, normally up to 1500 V.

In small self-contained PLCs in which all I/O points are physically located on one casing, all inputs will be of one type (e.g. 24V) and the same for outputs (e.g. 240V triac). This is because manufacturers supply only standard function boards for economic reasons. On the other hand, modular PLCs have greater flexibility of I/O, since the user can select from several different types and combinations of input and output modules.

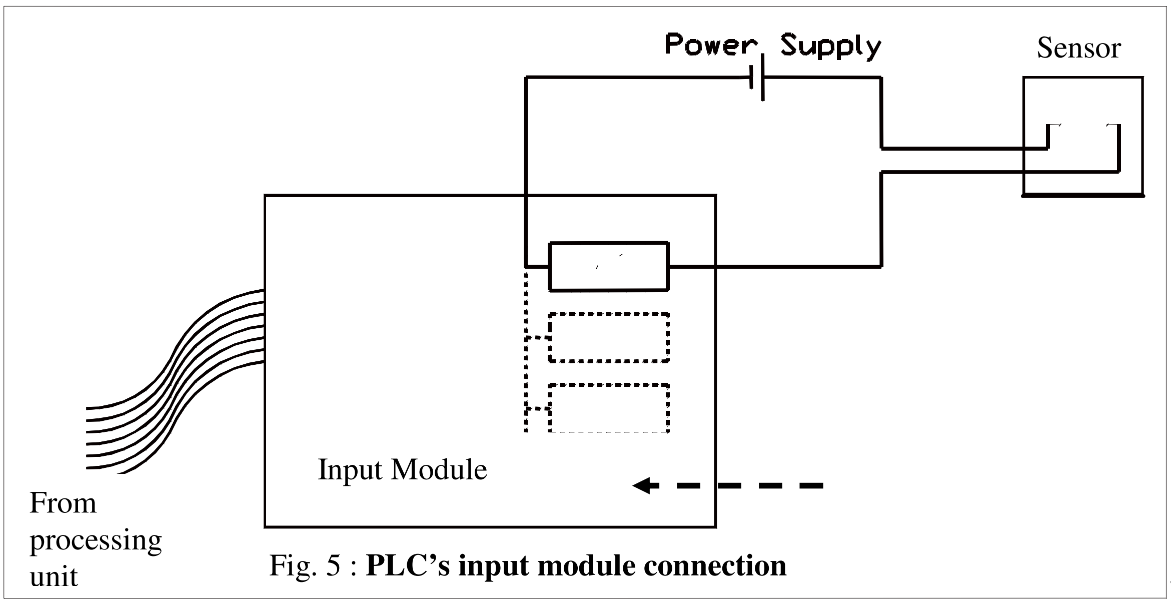


Fig. 5 : **PLC's input module connection**

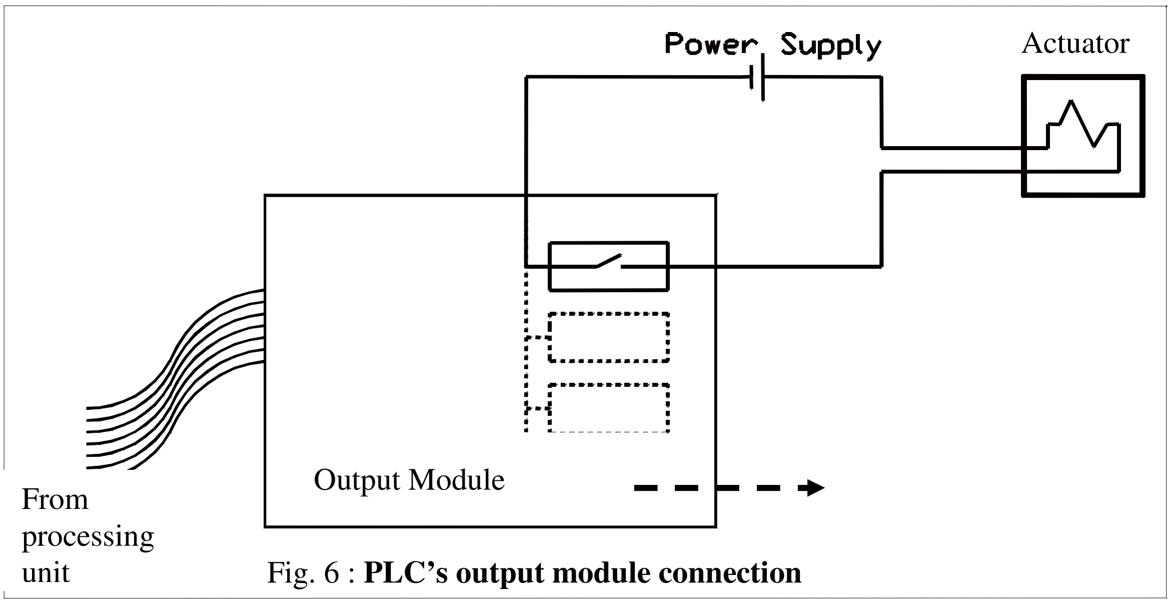
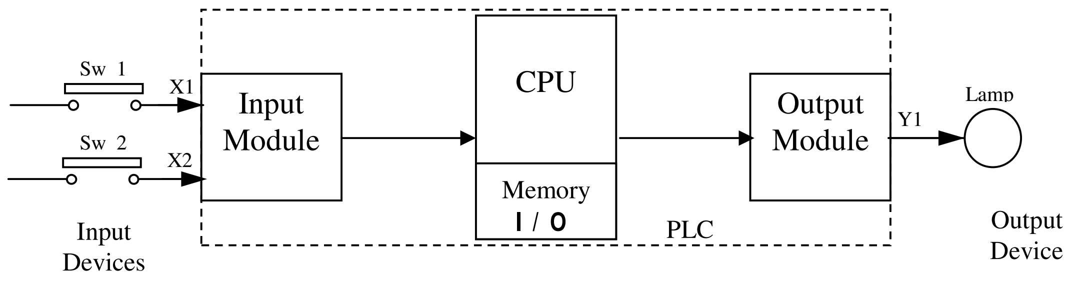


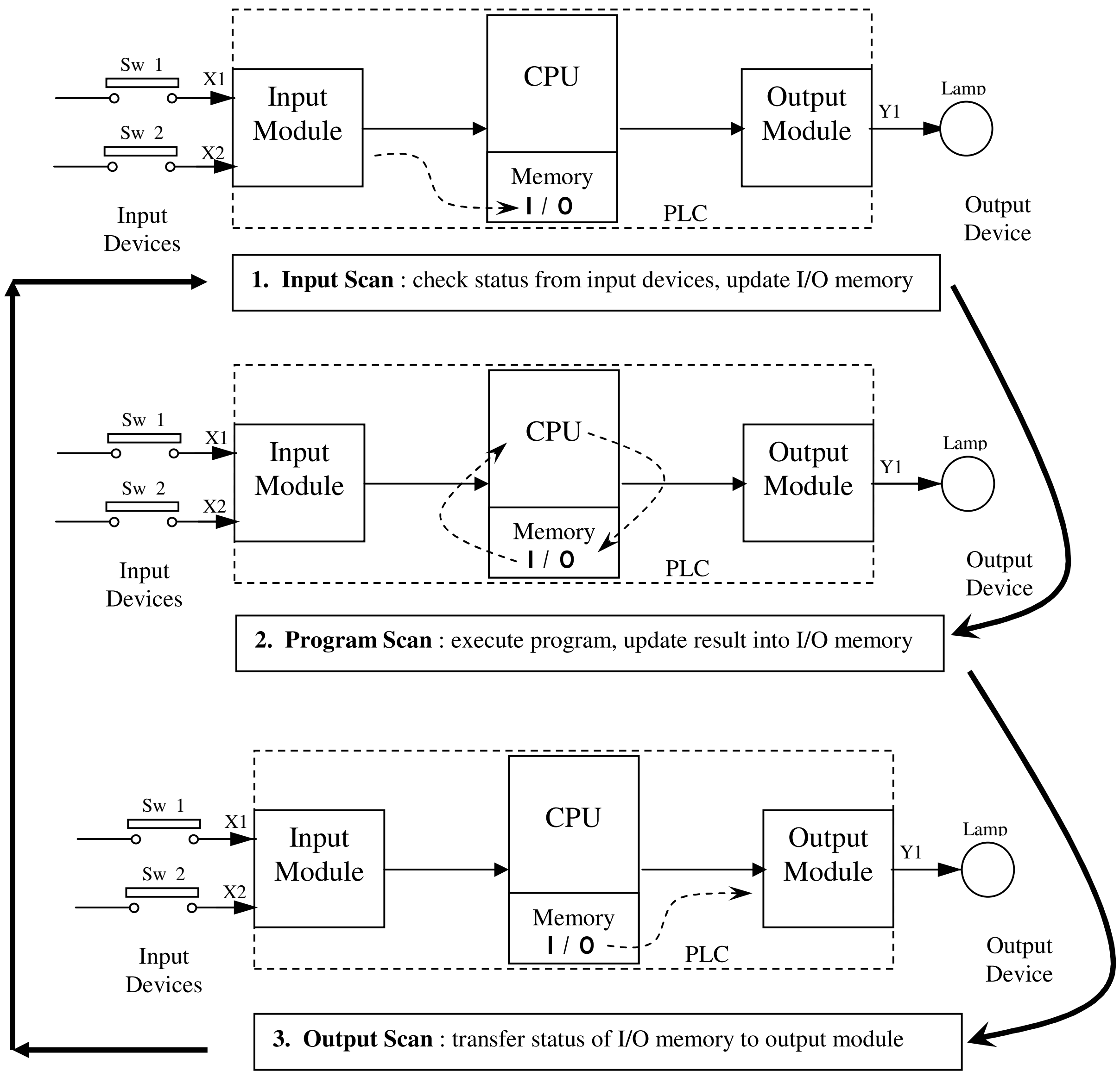
Fig. 6 : **PLC's output module connection**

In all cases the input/output module units are designed with the aim of simplifying the connection of input devices and actuators to the PLC. For this purpose, all PLCs are equipped with standard screw terminals or plugs on every I/O point, allowing the rapid and simple removal and replacement of a faulty I/O card. Every input/output module point has a unique address or channel number which is used during program development to specify the monitoring of an input or the activating of a particular output within the program. Indication of the status of input/output channels is provided by light-emitting diodes (LEDs) on the PLC or I/O unit, making it simple to check the operation of processed inputs and outputs from the PLC itself.

How does a PLC work ?



Eg. **PLC Program** : Sw_1 AND Sw_2 = Lamp



1. **Input Scan** : check status from input devices, update I/O memory

2. **Program Scan** : execute program, update result into I/O memory

3. **Output Scan** : transfer status of I/O memory to output module

8.4 Internal Operation and Signal Processing

The CPU of the PLC executes the user-program **over and over again** when it is in the **RUN** mode. The following figure shows the entire repetitive series of events.

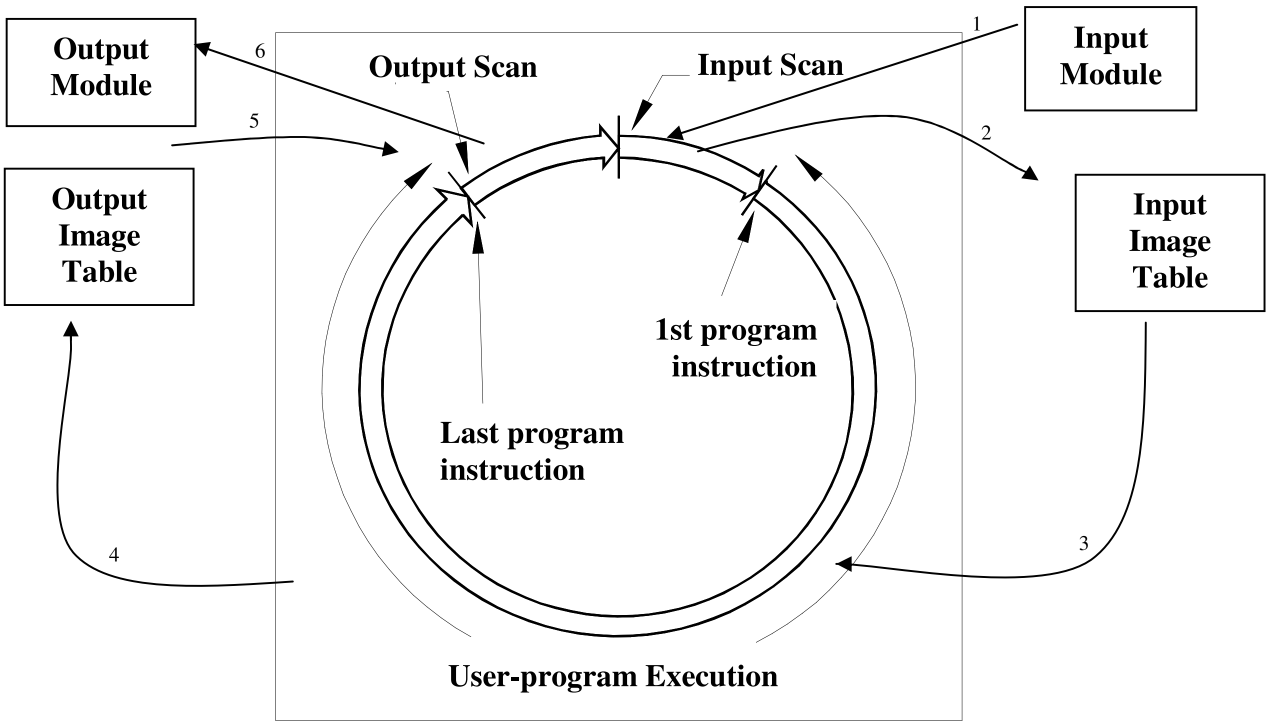


Fig. 7 : **PLC scan cycle** ❋

(a) **Input scan**

During the input scan, the **current status of every input module is stored in the input image (memory) table**, bringing it up-to-date. Thus all the status of the input devices (which in turn is connected to the input module) are updated in the input memory table.

(b) **Program scan**

Following the input scan, the CPU enters its user **program execution**, or **program scan**. The execution involves starting at the program's **first instruction**, then moving on to the second instruction and carrying out its execution sequence. This continues to the **last program instruction**. Throughout the user-program execution, the CPU continually keeps **its output image (memory) table up-to-date**.

(c) **Output scan**

During **program scan**, the output modules themselves are **not kept continually up to date**. Instead, **the entire output image table** is transferred to the output modules **during the output scan** which comes after the program execution. Thus the output devices are activated accordingly during the output scan.

Note that by virtue of the cyclic nature of the program I/O scan, the status of the inputs and outputs cannot be changed within the same scan cycle. If an input signal changes state after the input scan, it will not be recognized until the next input scan occurs.

The time to update all inputs and outputs depends on the total number to be copied, but is typically a few milliseconds in length. The total program execution time (or cycle time) depends on the length of the control program. Each instruction takes 1-10 μs to execute depending on the particular programmable controller employed. So a 1K (1024) instruction program typically has a cycle time of 1-10 ms. However, programmable controller programs are often much shorter than 1000 instructions, namely 500 steps or less.

An example of **IO Address Assignment Table** :

Input Listing	Address	Output Listing	Address
Inductive Sensor	X0	Pilot Light	Y0
Reed Sensor	X1	Small DC Motor	Y1
Capacitive Sensor	X2	Solenoid Valve	Y3
Pushbutton	X3		

8.5 Types of PLC System

General definitions of PLC size are given in terms of program memory size and the maximum number of input/output points the system can support

PLC Size Defined	**Max I/O points**	**User memory size (No. of instructions)**
Small	40/40	1K
Medium	128/128	4K
Large	>128/>128	>4K

Table 2 Categories of PLC

However, to evaluate properly any programmable logic controller we must consider many additional features such as its processor, cycle time, language facilities, functions, expansion capability, etc.

Small PLC

In general, small and mini PLCs are designed as robust, compact units which can be mounted on or beside the equipment to be controlled. They are mainly used to replace hard-wired logic relays, timers, counters, etc. that control individual items of plant or machinery, but can also be used to coordinate several machines working in conjunction with each other.

Small PLCs can normally have their total I/O expanded by adding one or two I/O modules. However, if any further development is required, it will often mean replacement of the complete unit.

A single processor is normally used, and programming facilities are kept at a fairly basic level, including conventional sequencing controls and simple standard functions: e.g. timers and counters. Programming of small PLCs is by way of logic instruction lists (mnemonics) or relay ladder diagrams.

Program storage is by EPROM or battery-backed RAM. There is now a trend towards EEPROM memory with on-board programming facilities on several controllers.

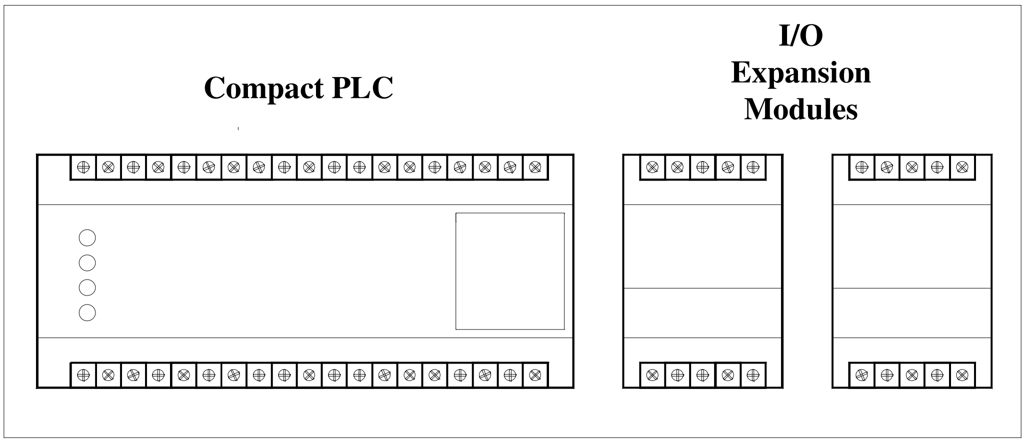


Figure 8: Typical small PLC system

Medium-sized PLC

In this range modular construction predominates with plug-in modules on rack mounting system or Back Plane system. This construction allows the simple upgrading or expansion of the system by fitting additional I/O cards into the racks, since most rack systems have space for several extra function cards. Boards are usually *ruggedized* to allow reliable operation over a range of environments.

In general this type of PLC is applied to logic control tasks that cannot be met by small controllers due to insufficient I/O provision, or because the control task is likely to be extended in the future. This might require the replacement of a small PLC, whereas a modular system can be expanded to a much greater extent, allowing for growth. A medium-sized PLC may therefore be financially more attractive in the long term.

Communications facilities are likely to be provided, enabling the PLC to be included in a *distributed control* system.

Combinations of single and multi-bit processors are likely within the CPU. For programming, standard instructions or ladder and logic diagrams are available. Programming is normally carried out via a small keypad or a VDU terminal.

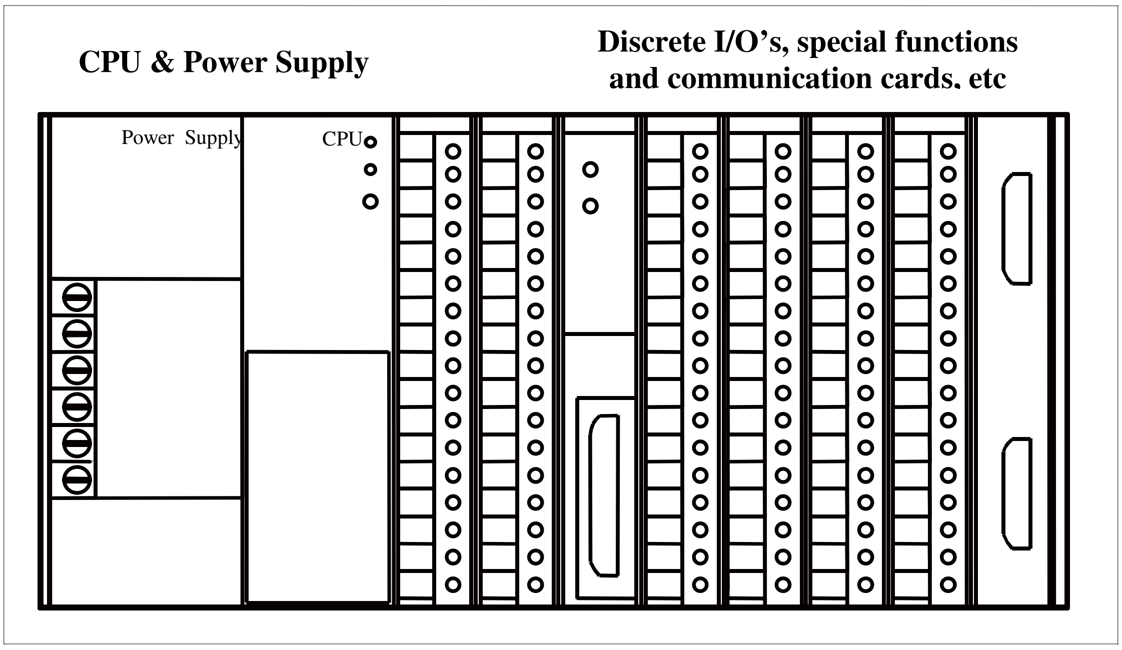

CPU & Power Supply
Discrete I/O's, special functions and communication cards, etc
Power Supply
CPU

Figure 9: Typical medium PLC system

Large PLC

Where control of very large numbers of input and output points is necessary or complex control functions are required, a large programmable controller is the obvious choice. Large PLC are designed for use in large plants or large machines requiring continuous control. They are <u>also employed as supervisory controllers</u> to monitor and control several other PLCs or intelligent machines, e.g. CNC tools.

Modular construction in Eurocard format is standard, with a wide range of function cards available including analog input/output modules. There is a move towards newer and faster processors, and also multi-processor usage in order to efficiently handle a large range of differing control tasks, e.g.:

- 16-bit processor as main processor for digital arithmetic and text handling.
- Single-bit processors as co- or parallel processors for fast counting, storage, etc.
- Peripheral processors for handling additional tasks which are time-dependent or time-critical, such as: Closed-loop (PID) control
  Position controls
  Floating-point numerical calculations
  Diagnostics and monitoring
  Communications for decentralized (distributed) I/O
  Process animation (screen graphics)
  Remote input/output racks.

This multi-processor solution optimizes the performance of the overall system as regards versatility and processing speed, allowing the PLC to handle very large programs of 100K instructions or more. Memory cards can now provide several megabytes of CMOS RAM or EPROM storage.

8.6 Selection of a Programmable Logic Controller

When an engineer is embarking on any program of automation, it must be remembered that the controller is only a tool used to perform the necessary tasks. The actual task of implementing automation is therefore paramount, often involves many methodologies in its system design phase. Once the specifications are established, the job of selecting a suitable controller will then become most important as this would determine how at ease the automation program might continue.

There is a massive range of PLC systems available today, with new additions or replacements continually being produced with enhanced features of one types or another. Advances in technology are quickly adopted by manufacturers in order to improve the performance and market status of their products. However, irrespective of make, the majority of PLC in each size range are very similar in term of their control facilities. Where significant differences are to be found is in the programming methods and languages, together with differing standards of manufacturer support and backup.

Considerations in choosing a suitable PLC

With the vast choice of equipment now available, the engineer can usually obtain similar systems from several original equipment manufacturers (OEM). When the specification requires certain types of function or input/output, it is possible that one system from a single manufacturer standing out as more superior or cost effective than the other; but normally this is rarely the case. **To determine the the most suitable PLC** to be used in the automation task, there are **several basic considerations** to be made: ✳

- ✓ Necessary input/output capacity;
- ✓ Types of I/O required;
- ✓ Size of memory required;
- ✓ Speed and power required of the CPU and instruction set
- ✓ Manufacturer's support and backup.

All these topics are to a large extent interdependent, with the memory size being directly tied to the amount of I/O as well as program size. As the I/O and memory size rises, this takes longer to process and requires a more powerful, faster central processor <u>if scan times are to remain acceptable.</u>

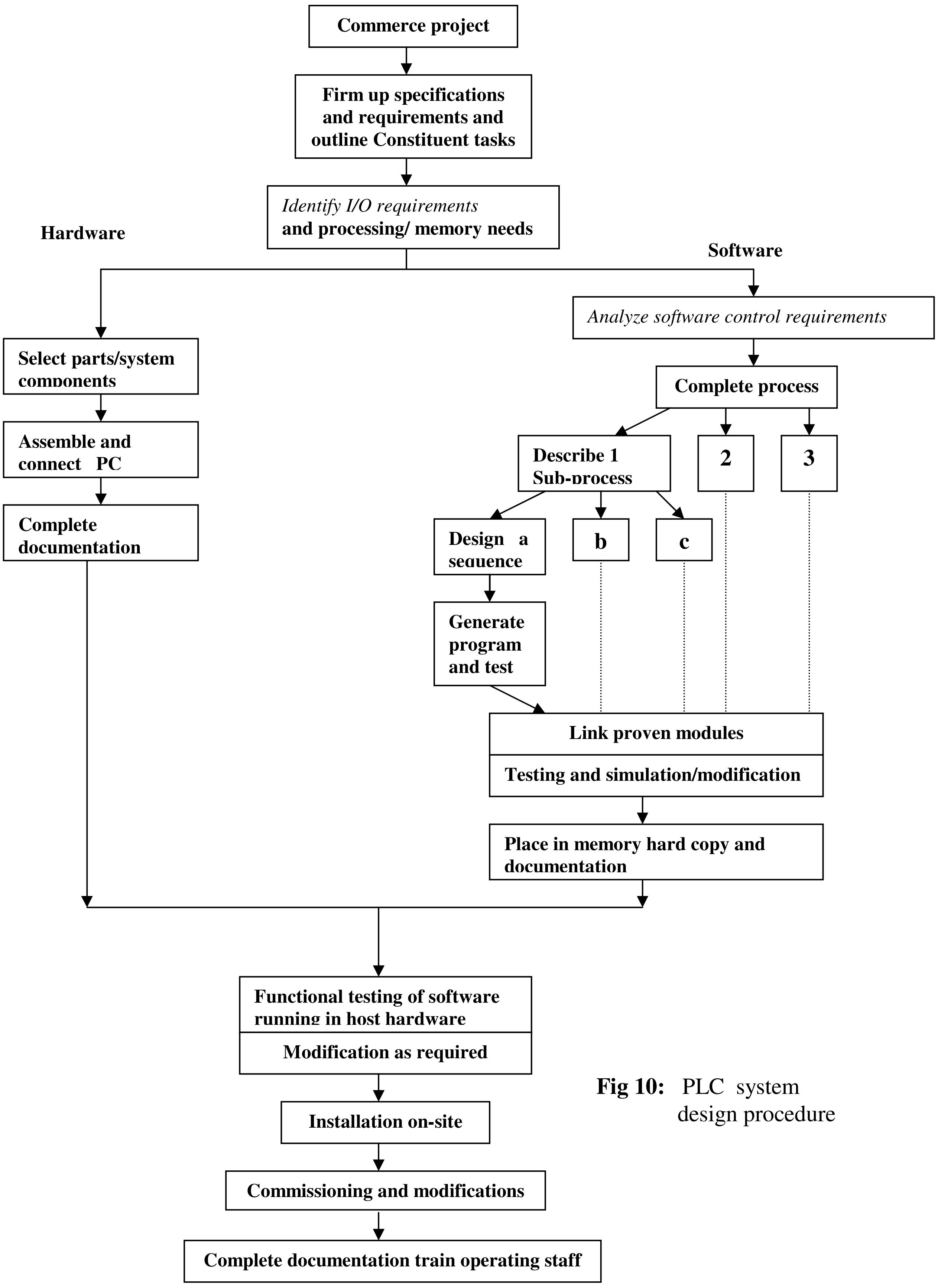

Commerce project
Firm up specifications and requirements and outline Constituent tasks
Identify I/O requirements and processing/ memory needs
Hardware
Software
Select parts/system components
Assemble and connect PC
Complete documentation
Analyze software control requirements
Complete process
Describe 1 Sub-process
2
3
Design a sequence
b
c
Generate program and test
Link proven modules
Testing and simulation/modification
Place in memory hard copy and documentation
Functional testing of software running in host hardware
Modification as required
Installation on-site
Commissioning and modifications
Complete documentation train operating staff

Fig 10: PLC system design procedure

PLC - Programmable Logic Controller

Type of I/O and Capacity Needed

Normally this I/O requirement is the engineer's very first consideration in selecting the PLC to be used. Commonsense tells us that the I/O sections of a PLC system must have enough termination points to connect all signal and control lines for the process. These termination points on the I/O sections must therefore conform to the basic system specifications with regard to:

- Voltage levels and current loadings;
- The number and type of I/O points required per module (or unit in the case of self-contained PLC);
- Isolation required between the controller and the target process;
- The need for high-speed I/O, or remote I/O, or any other special facility;
- Future needs of the plant in terms of both expansion potential and installed *spare* I/O points;
- Power supply requirements of I/O points: any need for an on-board power supply unit to drive input transducers or output actuators.

In certain cases there may be a need for signal conditioning modules to be included in the system, with obvious space demands on the main or remote racks. When the system is to be installed over a wide area, the use of a remote or decentralized form of I/O working can give significant economies in cabling the sensors and actuators to the PLC.

Memory and Programming Requirements

Depending on the type of PLC being considered, the system memory may be implemented on the same card as the CPU, or alternatively on dedicated cards. This latter method is the more adaptable, allowing memory size to be increased as necessary up to the system maximum, without a reciprocal change in CPU card.

As stated in the previous section, memory size is normally related to the *amount of I/O points* required in the system. The other factor that affects the amount of memory required is of course the control program that is to be installed. The exact size of any program cannot be defined until all the software has been designed, encoded, installed and tested. However, it is possible to accurately estimate this size based on average program complexity. A control program with complex, lengthy interlocking or sequencing routines obviously requires more memory than one for a simple process. Program size is also related to the number of I/O points, since it must include instructions for reading from or writing to each point. Special functions required for the control task may also require memory space in the main PLC memory map to allow data transfer between cards.

Finally, additional space should be provided to allow for *changes in the control program*, and for future expansion of the system.

There is often a choice of available memory type - RAM or EPROM. The RAM is the most common, allowing straightforward and rapid program alterations both before and after the system is installed. RAM contents are made semi-permanent by the provision of battery-backing on their power supply. RAM must always be used for I/O and data functions, as these involve dynamic data. EPROM memory can be employed for program storage only, and requires the use of a special EPROM eraser/programmer to alter the stored code. The use of EPROMs is ideal where the several machines are controlled by identical programmable controllers running the same program. However, until a program has been fully developed and tested, RAM storage should be used.

Microcomputers are often used as program terminals or development stations. The large amounts of RAM and disk storage space provided in these machines allow the development and storage of many PLC programs, including related text and documentation. Programs can be transferred between the microcomputer and the target PLC for testing and alteration. EPROM programming can also often be carried out via the microcomputer.

Instruction Set/CPU

Regardless of size, all PLCs can handle logic control, sequencing, etc. Differences start to emerge are in the area of data handling, special functions and communications. Larger programmable controllers tend to have more powerful instructions than smaller ones in these areas, but careful scrutiny of small/medium machines can often reveal the capability to perform specific functions at surprisingly good levels of performance.

In modular PLCs, there may be a choice of CPU card, offering different levels of performance in terms of speed and functionality. As the number of I/O and function cards increases, the demands on the CPU also increase, since there are greater numbers of signals to process in each cycle. This may require the use of a faster CPU card if scan time is not to suffer.

Following the selection of the precise units that will make up the PLC for a particular application, the software and hardware design functions can then be carried out independently.

Manufacturer's Support and Backup

An often neglected aspect of consideration is the manufacturer's support and backup. The information and assistance required from a manufacturer may vary considerably depending on the engineer's previous experience of PLC and control systems in general. Therefore, when considering manufacturer's support, it will be worthwhile to ask the following questions:

- Can the engineer obtain assistance with the design work?
- What proportion of the market does the potential supplier/manufacturer hold, and do they have a track record in the necessary area of application?
- Does the manufacturer offer training course on the type of PLC system likely to meet the user's needs?
- Are all relevant handbooks and manuals available in the required language, to a good standard?
- What compatibility exists between any likely system and other types of PLC from the same of different manufacturers?
- Does the programming method used suit the outline control plan for the application?

For the relatively inexperienced engineer, it is a great advantage if the supplier/manufacturer can offer assistance with the system design work, together with further help and training in actual program design and coding. Even though PLCs are relatively straightforward to program, it is usually necessary to assist the engineer through the *transition* period from one form of control or controller to another.

What this has tried to highlight is the need to consider many other factors besides cost and technical performance when attempting to choose a make or type of programmable logic controller.

8.7 PLC Programming / Combinational Logic

Ladder Programming Development

In the design of automated machine and control system for process, programmable logic controllers are often used. For the controller to carry out its intended task, a control program is necessary. The design and development of the control program is of vital importance as it constitutes the means for controlling the target process. It can be seen that the strategy allows the development and installation of both the hardware and software concurrently. This will provide the program designer with the maximum possible time to consider the control requirements and generate the software solution.

For simple control or equipment, the amount of planning and actual design work for these short programs is minimal. In practice the control program is far more complex and involved mixed logic function together with many other programmable functions provided by modern programmable logic controllers.

Therefore a formal and structured approach to software design must be adopted in order that the program can be easily understood, debugged and documented. In terms of design methodology, ladder programming is no different from the conventional computer programming. Thus considerable attention must be given to:

- Task definition
- Software design techniques
- Documentation
- Program testing

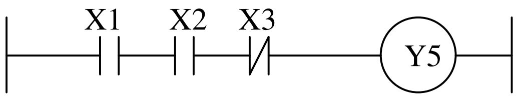
The division of programming tasks into functional blocks is an important part of software design. In logic programming, there are two different techniques that may be used to implement the function of a given block.

- **Combinational logic**, where the output is purely dependent on the combination of the inputs at any instant in time.

- **Sequential networks**, where the output is dependent not only on the actual inputs but on the sequence of the previous inputs and outputs. (memorising events)

Combinational logic design

All programmable controllers have standard logic instructions – AND, OR, NOT etc. and these may be combined to create logic networks. Boolean algebra may be used as a tool to assist in the design of logic networks. E.g. $Y5 = X1.X2.\overline{X3}$



Ladder diagram

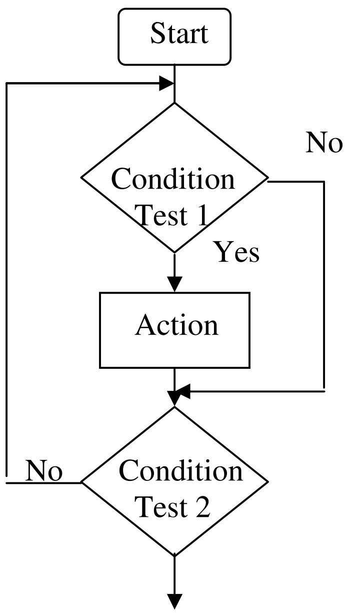
Logic symbols are often used in conjunction with Boolean algebra, as they group inputs into logical functions that are then easily transferred into Boolean terms.

Sequential control

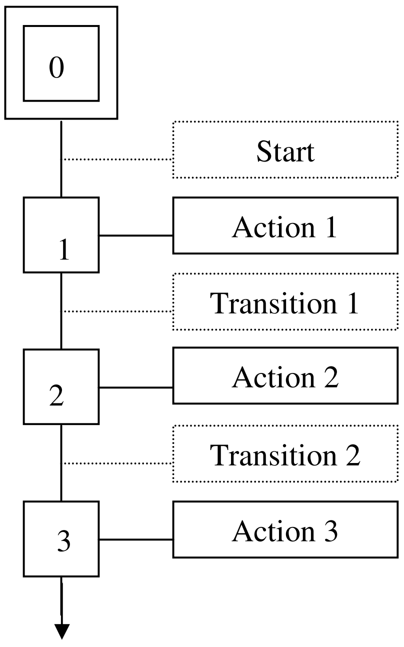
Sequential problems have long been solved using conventional logic gates as building blocks, but using certain techniques to express and identify the sequence logic equations that control the system outputs. Advanced PLC instructions such as shift registers, sequencers, master control relays, drum timers etc are provided to simplify the design and implementation of sequence systems.

The software design procedure is as follows:

1 The process is verbally described
2 This description is translated into a <u>function diagram</u> or GRAFCET.
3 The conditions are identified and converted into <u>Boolean equations</u>
4 The Boolean equations are converted into <u>ladder logic</u> for the PLC



Flowchart



GRAFCET

Flowcharts

These are associated with computer programming, but are common method of displaying the sequential operation of a control system. Flowcharts have a direct relationship with the verbal description of a control sequence, showing each test and result action within a series on interlinked symbols. Unfortunately, when used to describe larger control systems, flowcharts can become difficult to lay out, resulting in large, cumbersome diagrams.

Program Structure

It is good practice to have a structured program with definite sections dealing with specific areas of operation. By adopting this approach, the programs developed are reliable and can be easily understood. It also eases the process of debugging hence leading to a shorter down time of the machine or process involved. These sections can be broadly categorised as follow:

- ⋆ Operating Modes
- ⋆ Process operation / Sequence Logic
- ⋆ Signal Outputs
- ⋆ Status / Indicator output

A. Operating Modes

Initial position: All automated equipment is likely to have an initial or home position. This is the position that all of its actuators will adopt prior to the operation of the equipment. Therefore to signify and initialise a basic position for the equipment, the home position of each actuator can be combined logically and programmed as a step in a sequential process.

For example in a simple drill system that comprises of a drill cylinder and a clamp cylinder as shown in Fig 1, the initial position can be defined as:

- Drill cylinder retracted
- Clamp cylinder retracted

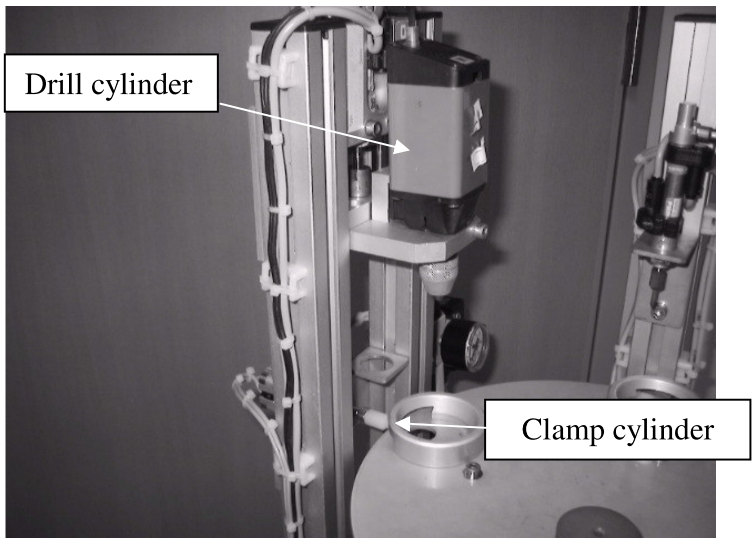


Fig. 11 : Processing Station

Enabling / Reset conditions: Most industrial processes and equipment have manual start and stop controls incorporated into the programmable logic controller program. By means of interlocking, a proper and safe start up and stopping of the process or equipment is ensured. Reset control is also included in case of system hang. It is normally used to reset the equipment to its basic position upon activation.

B. Process operation / sequence
This is the main section of the program which determines the correct functioning of the operation. This is achieved through the use of both combinational and sequential networks. The resultant outputs do not normally drive the actuators directly, but instead are used to operate intermediate marker relays.

C. Signal output
Output signals to process actuators are formed by interlocking the resulting operation sequence output with any enabling conditions that exist in the operating modes mentioned above.

D. Status / indicator outputs
Process status is often displayed using indicator lamps or alarms, etc. Such elements are programmed in this section of the software.

Standard logic instructions

The processing potential of binary signals can be described using the three basic operations:

AND / OR / NOT (negation)

These basic logic operations can be used to solve combinational control problems.

Combinational Logic Control Design

Boolean algebra can be used as a tool to assist in the design of logic networks. The original logic circuit or program is first converted into a Boolean equation. Based on the rules governing Boolean algebra the equation is simplified resulting in more economical or elegant, in terms of logic functions, solution.

When dealing with fairly complex combinational logic tasks, the requirements can also be expressed in terms of Boolean equations so that it can be simplified before translating them into ladder logic.

As example to illustrate how a ladder diagram, show in Fig. 2 is translated from the Boolean equation based on the given requirement below: -

To operate valve Y1 →	limit switches A and B and valve X are activated and both switch C and valve Z are not activated.
Valve Y1 will also operate	if switch D and valve X are activated and both level switch C and valve Z are not activated.

In **Boolean**:

$$Y1 = A.B.\overline{C}.X.\overline{Z} + D.\overline{C}.X.\overline{Z}$$

$$= (A.B + D).\overline{C}.X.\overline{Z}$$

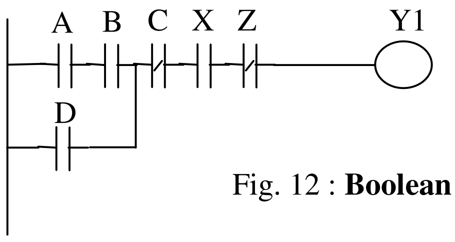


Fig. 12 : **Boolean Logic**

Good programming practice

1 Plan your program on paper first! Don't just power up your PLC and start keying in elements. 80% of your time should be spent working out the program, and only 20% keying it in.

2 Keep documentation of all elements used in the program – add comments as necessary.

3 Assume the program will find every error sequence possible – design safety into it!

4 Keep programs simple and readable. Comments would be helpful.

5 Try sectional development and testing if possible.

6 Use forcing and monitoring functions to observe program operation in situations where it is safe to do so.

Some useful Boolean logic formulae

$X + 0 = X$	$X \cdot 0 = 0$
$X + 1 = 1$	$X \cdot 1 = X$
$X + X = X$	$X \cdot X = X$
$X + X' = 1$	$X \cdot X' = 0$

De Morgan's Law:

$$(X + Y)' = X' \cdot Y'$$

$$(X \cdot Y)' = X' + Y'$$

Note : $X' = (\text{NOT } X) = \overline{X}$

Distributive laws:

$X(Y + Z) = XY + XZ$ $\qquad$ $X + YZ = (X + Y)(X + Z)$

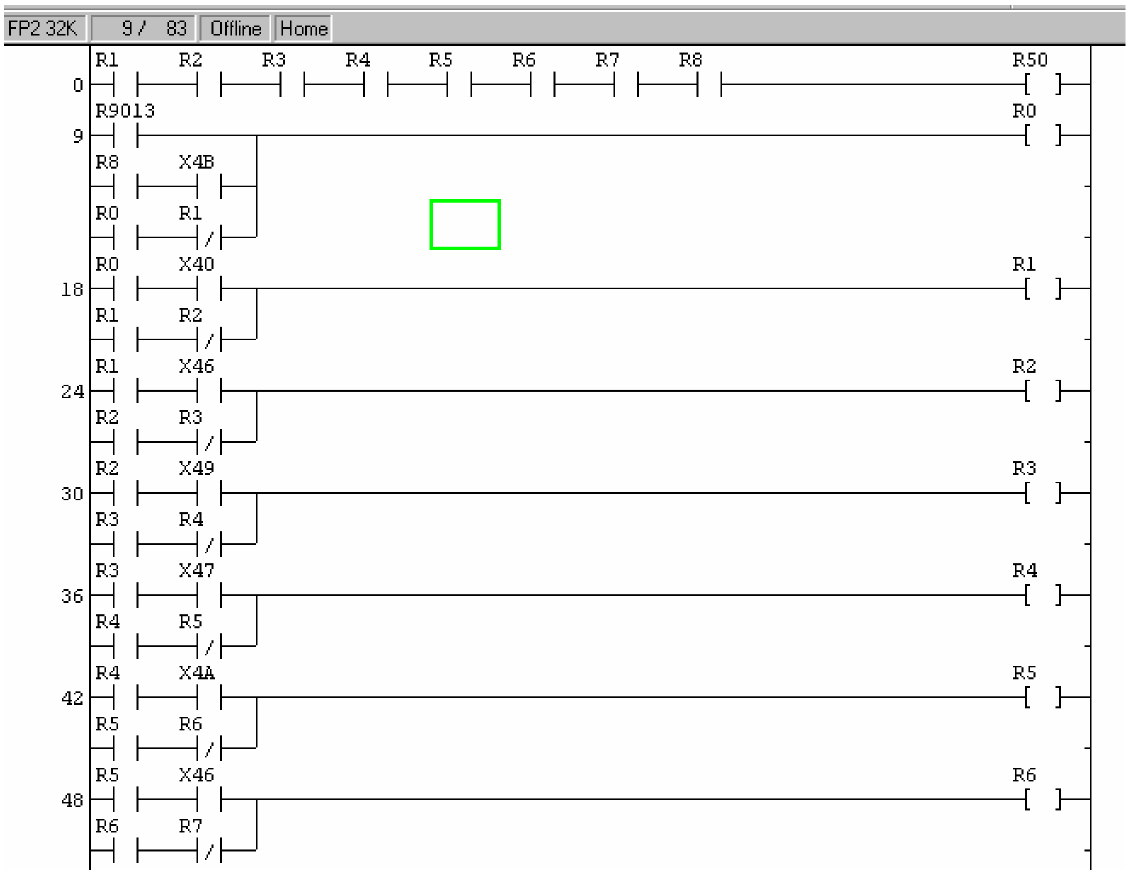
Simplification theorems:

$XY + XY' = X$ $\qquad$ $(X + Y)(X + Y') = X$

$X + XY = X$ $\qquad$ $X(X + Y) = X$

$(X + Y')Y = XY$ $\qquad$ $XY' + Y = X + Y$

Sample Ladder Diagram



eg. $\quad R0 = R9013 + R8 \cdot X48 + R0 \cdot \overline{R1}$

$R1 = R0 \cdot X40 + R1 \cdot \overline{R2}$

i.e $\quad R0 \cdot X40 \quad \Rightarrow \quad$ R0 **AND** X40

$R9013 + R8 \cdot X48 \quad \Rightarrow \quad$ R9013 **OR** (R8 **AND** X48)