NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Electrical and Electronic Engineering

FRUIT PACKING OTOMATION SYSTEM (PLC)

GRADUATION PROJECT
EE – 400

Student: Aliksan Taşdemir (951112)

Supervisor: Özgür C. Özerdem

Lefkoşa - 2001

ACKNOWLEDGEMENTS

First of all, I want to thank to my teacher's the supervisor Mr.Özgür C. ÖZERDEM , Mr. Kaan UYAR, and Mr.Mherdad KHALEDİ for supporting me durring my education.

Also thanks to Prof.Haldun GÜRMEN , Prof. Dr. Fakhreddin MAMEDOV ,

Prof. Dr. Şenol BEKTAŞ and Prof. Dr. Khalil İSMAİLOV

Then I wish to thank to my friends Altınay ORMAN , Bora E. TUGLU, Orhan ŞADİ Fatoş GÜVEN and Bülent GÜVEN for their help. And finally I want to thank everybody who helped and suppurted me from my first day at the university till my graduation.

ABSTRACT

The aim of the project is to provide an introduction to the desing structure and operation of PLC's and an insight into their applications.

Chapter 1: Describes the basic of PLC's and their logic based operation

Chapter 2-3: Mainly deal with hardware aspects relating to PLC's. The basic of a PLC's internal hardware structure and program execution are covered in chapter 2-3

Chapter 4: Contains information on the PLC programming languages covered by the international standard

Chapter 5 : Covers all the major espects of ladder function block programming showing how counters , timer and shift registers can be implemented.

Chapter 6 : Covered a set of tutorial-type worked examples.

Chapter 7 : A number of application examples are given in this chapter.

ABBREVIATIONS

Absolute encoder A shaft or linear encoder that generates a unique number for resolvable position

A.C. or a.c. Alternating current

Access time The time taken for data to be read from memory

Acumulator A CPU register used to hold the results of arithmetic and logic operations

ADC Anologue to digital converter.

Address The numerical assingment of a particular memory location

ALU Arithmetic logic unit.

ANSI American national standard institute

ASCII American standard code for information interchange

Bandwidth Iformation carrying capacity of a communication channel expressed in bits/s.

Baud A unit of transmission.

BCD Binary code decimal

Bit Binary digit

Boolean data Data represented as a single bit.

Boot To load an operating system

bps Bts per second

BS **br**itish standart

Buffer A bolck of memory used for temporary storage

Bus A set of conductors used for communicating signals

Byte A group of eight bits

CAD Computer aided design

CAM Computer aided manufacture

CIM Computer integrated manufacture

Clock Aperiodic signal used for synchronization

Closed-loop control A system in which the output signal is measured and fed back to the input point.

CNC Computer numerical control

Contact bounce The problem relating to mechanical switches which produce a noisy signal when swithed

Compiler A program to translate a high-level language into machine code

Counter A function block that gives an output when a set number of pulses have been applied to the input

CPU Central processing unit

CSMA/CD Carrier sense multiple access with collusion detection

Current sinking The action of receiving current

DAC Digital to analog convertor

EEPROM Electrically erasble programmableread only memory

EPROM Electrically programmableread only memory

FET Field effect transistor

Flag A single bit variable used the indicate that some conditions has occurred

Full dublex A full dublex data link allows the transmission data simultaneously İn bth direction

Gateaway A device thas connects and allows messages to be communicated between to or more dıfferet network system

Gray code A binary code in which consecutive codes differ in only a single bit position

Hexadecimal Base 16 number system

Hz Hertz .the unit of frequency equal to one cycle per second

IC Integrated circuit

IEC İnternational electrotechnical commisions

IEE Institute of electrical enngineers

IEEE Institute of electrical and eletronic eng.

Ladder diagram A programming language in which a circuit, consisting of contacts , coils and other elements, such as functions blocks, and bounded between left and right vertical lines.

LSB Least significant bit

Machine code Binary number that represent CPU instruction

MSB Most significant bit

MTBF Mean time between failure

N/C Normaly closed

N/O Normaly open

Off-delay timer A function block that holds its output high for a specified duration when its input changes from high to low

On-delay timer A function block that delays setting its output high when its input changes from low to high

Program counter A CPU register containing the address of next instruction to be executed

Programming language IEC – defined programming languages are function block diagram (FBD)

RAM Random access memory

ROM Read only memory

Run The run mode executes the user program stored in memory

Scan PLC program execution loop to continosly read input values and set outputs according to the program requirement

State The condition of a Boolean variable

Tachogenerator A permanent magnet d:c: motor operated as a voltage generator:

TTL Transistor-transistor logic.

Two's complement A system for representing negative numbers in binary notation

CHAPTER 1
Information about PLC

1.1 Basics of programmable logic controllers

An apt definition of a programmable logic controller (PLC) is that it is a 'digitally operating system, designed for use in an industrial environment which uses a programmable memory for internal storage of user-orientated instructions for implementing specific functions such as logic, sequencing, timing, counting and arithmetic, to control, through digital or analogue inputs and outputs, various types of machines or processes. Figure 1.1 shows how the control action is achieved. Input devices (e.g. limit switches, proximity switches) and output devices (e.g. relay coils, signal lamps) from the machine or process to be controlled are connected to the PLC. A user enters a sequence of instructions (known as the program) into the PLC's program memory. The controller then continuously monitors the states of the inputs and activates outputs according to the control algorithm defined by the user's program.

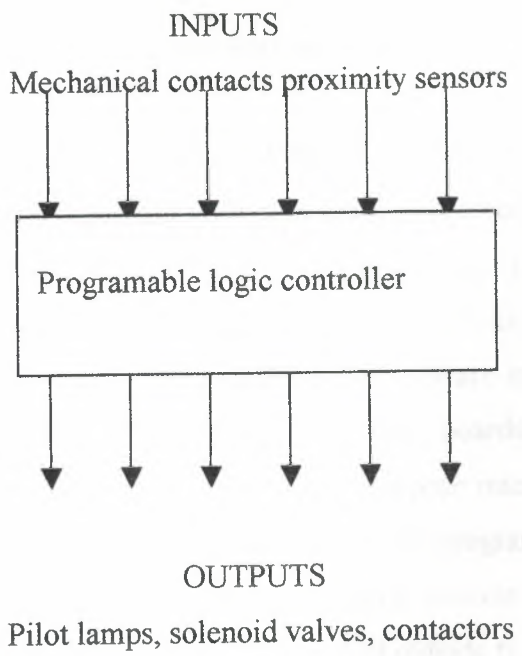

INPUTS
Mechanical contacts proximity sensors
Programable logic controller
OUTPUTS
Pilot lamps, solenoid valves, contactors

Figure 1.1 the control action of a PLC.

Because the stored program can be modified, new control features can be added or old ones changed without rewiring the input and output devices. The result is a flexible system which can be used for control tasks that can vary in nature and complexity.

The main differences between a PLC and, say, a microcomputer are that:

1. Programming is predominantly concerned with control logic and function block operations.
2. Interfacing circuits are integral to the controller.
3. PLCs are rugged, being packaged to withstand vibration, temperature, humidity and noise.

Some typical programmable controllers are shown in Fig. 1.2.2 Systems range from:

1.Stand-alone single unit PLC systems. These systems have a fixed number of input/output (I/O) points and are suited for small I/O automation tasks. Programming consoles are usually hand-held units which attach directly to thePLC.

2.Modular PLC systems. These systems are composed of self-contained hardware building blocks (modules), which plug directly into a proprietary bus back plane. A minimum configuration would utilize a CPU unit, a power supply unit and input/output (both digital and analogue) modules. The modular approach to PLC hardware means that the number and type of I/O points can be expanded. Special function modules for operations such as networking and computer linking are often provided. A host computer is usually used to develop program code, which is then targeted to the PLC hardware.

3.Computer bus based systems. Although not strictly a PLC system, it is possible to construct a custom computer controller using processor boards and peripheral interface boards which plug into a common bus back plane (rack). Various international bus standards have been developed for this purpose such as the STE (Euro card) bus and the VME bus. The modular approach to target hardware means that processing power can be increased by adding additional processor boards (assuming multiprocessing is supported by a bus scheduler) and special purpose interface boards can be designed to the bus standard. Bus based systems are usually programmed using high level language (e.g. PASCAL and C) cross compilers which execute on a host computer and which target hardware on the bus. Such systems fall outside the scope of this book.

1.2 Logic

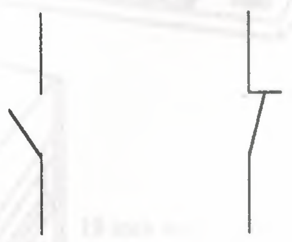
The term 'logic' features in the name 'programmable logic controllers' because programming is based on the logic demands of input devices. Programs implemented are predominantly logical rather than discrete versions of continuous algorithms. This section provides an introduction to logic.

The field of logic is concerned with systems that work on a straightforward two-state basis. A common electric light switch can be either *on* or off and these alternate possibilities can be labeled as true and false or as 1 and 0 (binary form) respectively. A Boolean variable (i.e. a logical variable) such as A can be used to represent any switch-like element, which can have one of two states. For example, it is possible to define thatA=1 when the switch on andA=O when the switch is off

Any condition in which there are two possibilities can be defined using a Boolean variable. For example, a work piece can be in or out of position. This may be represented as a simple binary statement B=1 (work piece is in position) or B=0 (work piece is not in position). Another example is a lamp, which can be either on or off.

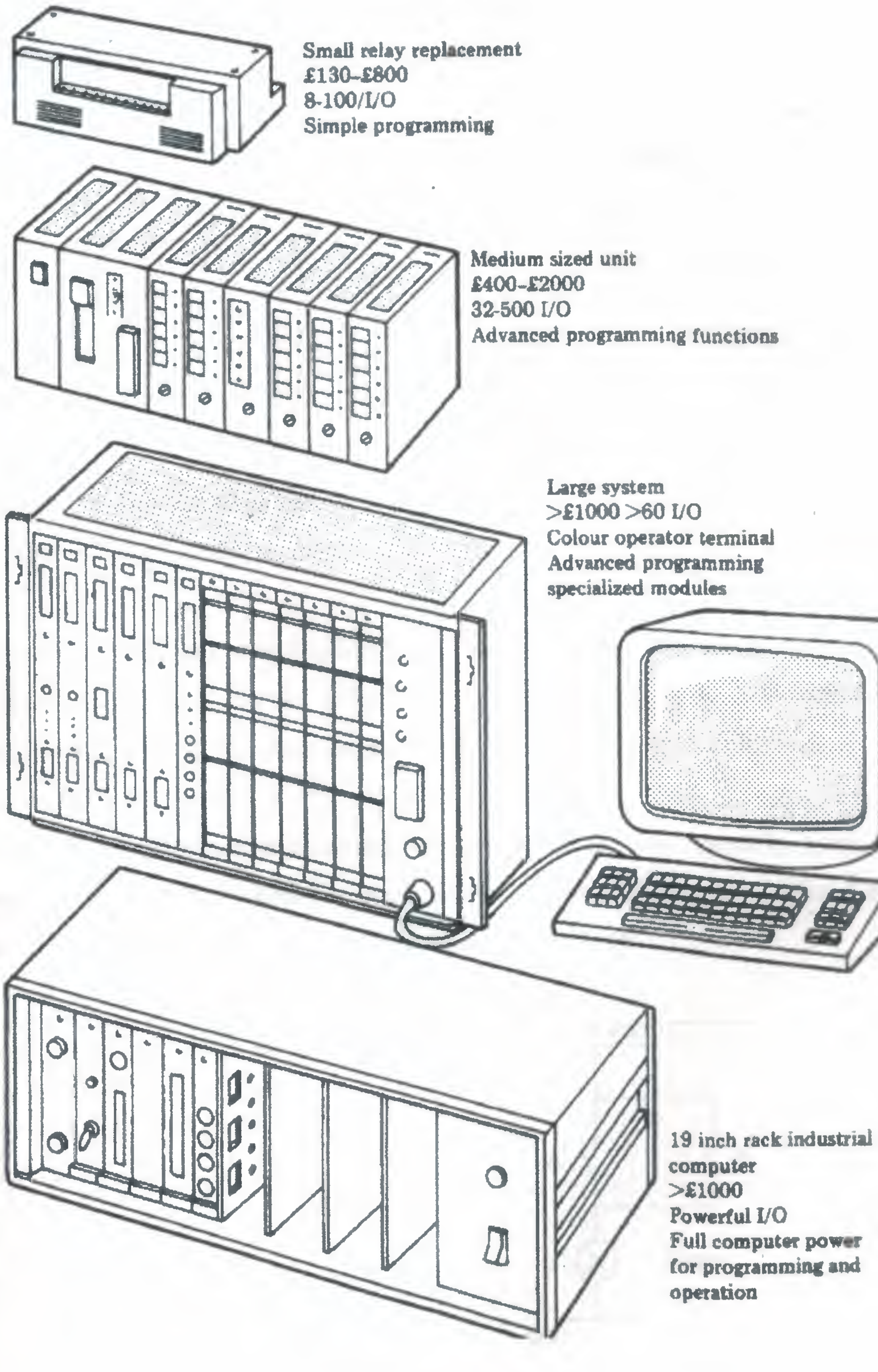
If the Boolean variable A is used to represent the on and off positions of a single throw normally open (abbreviated to N/O) switch the variable NOT A represents a normally closed (abbreviated to N/C) switch. The NOT logic function inverts the state of the input. The variable NOT A can be written as A, where the over bar denotes negation. Figure 1.3 tabulates the truth values (i.e. 0 and 1) of A against those of NOTA for the two switch positions.

Switch position	A	A
Off	0	1
On	1	0



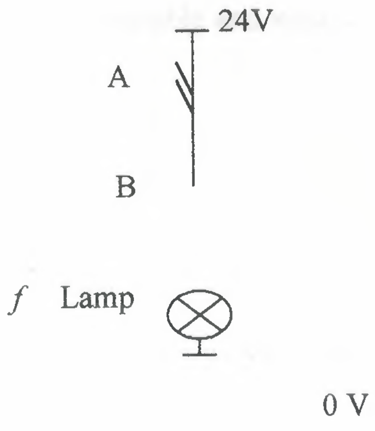
Truth table for the NOT function Normally open Normally closed

Figure 1.3 The NOT function.



The AND logic function describes the operation of two normally open single-pole, single-throw switches connected in series as shown in Fig. 1.4. Current flows only when both the two switches A and *B* are on, i.e. when A=1 *and B=1*. The output f can be written as the Boolean expression:

A	B	$f = AB$
0	0	0
0	1	0
1	0	0
1	1	1



Truth table for the AND gate

Figure 1.4 The AND function.

Note that this is not a multiplication but the logical notation used to mean that f is 1 if A is 1 and B is 1. For an AND function with three inputs A,B,C (e.g. three N/O switches in series) the Boolean expression for the output f would be written as

$f = ABC$

The OR logic function describes the operation of two normally open switches connected in parallel as shown in Fig. 1.5 Current flows when either switch *A* or switch *B* are in the on position. The Boolean notation for *A* OR *B*

A	B	$f = AB$
0	0	0
0	1	0
1	0	0
1	1	1

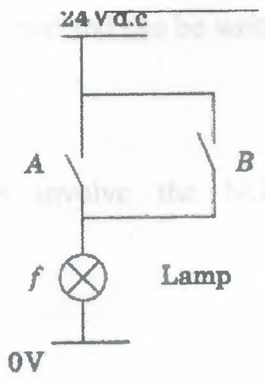


Figure 1.5 The OR function.

1.3 The laws of Boolean algebra

The above introduction has established the link between switching circuit diagrams and Boolean expressions. Rev. George Bode *(1815—64)* developed laws to analyze and construct logical statements. Boolean algebra deals with two-valued variables and is useful when analyzing switching circuits such as ladder diagrams.

The laws of Boo lean algebra consist of postulates and theorems. The four postulates, which deal with the combination of one Boolean variable and with, the constants 0 and 1 are

A+0=A (1.1)

A+1=1 (1.2)

A0=0 (1.3)

A1=A (1.4)

The two postulates that deal with one variable with itself are called the idem potent laws and are

A+A=A (1.5)

AA=A (1.6)

The commutative laws emphasize the fact that the position of the variables in an expression is not important. The commutive laws are

A + B = B + A (1.7)

AB = BA (1.8)

The associative laws deal with the use of brackets at can be expressed as

(A+B) + C = A + (B+C) (1.9)

(AB)C = A(BC) (1.10)

The distributive laws show how factors are combined and can be written as

(AB) + (AC) =A (B+C) (1.11)

(A+B)(A+C) =A + (BC) (1.12)

The complementarity's and involution laws involve the NOT function. The complementarity's laws are

$A + \bar{A} = 1$ (1.13)

AA=0 (1.14)

The involution law is

(A)=A (1.15)

Two other_postulates that involve the NOT function are called De Morgan's theorems which are

(AB)=A+B (1.16)

$$\overline{(A+B)}=\bar{A}\,\bar{B} \qquad (1.17)$$

The function A AND *B* all negated (i.e. $\overline{(AB)}$ is called the NAND function. NAND is an abbreviation for NOT AND. Similarly, the function *A* OR *B* all negated (i.e. $\overline{(A+B))}$ is called the NOR function. NOR is an abbreviation for NOT OR.

There are two branches of logic called combinational logic and sequential logic. A combinational logic system is one in which the output is a direct and unique consequence of the input conditions. A sequential logic system is one that depends on the sequence in which the inputs occur. This introduction has dealt with combinational logic. Sequential logic devices (e.g. flip-flops, shift registers) are discussed in Chapter 5.

1.4 Ladder diagrams

PLCs were developed to offer a flexible (e.g. programmable) alternative to conventional electrical circuit relay-based control systems built using discrete devices. The terminology and other concepts used to describe the operation of a PLC are based on conventional relay control terminology. The relationship is such that inputs are referred to as contacts, outputs are referred to as coils and memory elements (bits) are referred to as auxiliary relays.

The International Electro technical Commission (IEC) advocates five programming methods for PLCs, which are fully discussed in Chapter 5. Of these five, the predominant programming method used by all mainstream PLC systems is the ladder diagram method. This is a graphical programming technique, which has been evolved from conventional electrical circuit relay logic control methods. PLC systems provide a design environment (either in the form of software tools running on a host computer terminal or a hand-held LCD graphic programming console), which allow ladder diagrams to be developed and diagnosed.

In its simplest form, a ladder diagram is a network of contacts and coils bounded on the left and (optionally) on the right by vertical lines called power rails. The key features of a ladder diagram are:

1. Contacts represent the states of Boolean variables. For example, a contact called 'start_fan' might represent the ON and OFF state of a switch which is used to initiate a ladder rung which operates a fan. The ladder symbols for normally open and normally closed contacts are and respectively.

2. A typical ladder rung is a horizontal line of contacts which start from the left power rail and which provide the logic to operate a coil (or coils). A coil is a Boolean variable on the right of a ladder rung, which can be set to a true state when the contacts connecting it to the left power rail are on. The ladder symbol for a coil is a pair of parenthesis( ).

3. When contacts of a ladder rung are in the true state notional power is deemed to flow from the left power rail through the contacts to operate a coil (or coils) at the right-hand end of the rung. The drawing of the right power rail is optional as its use is implied.

4. The normal convention for evaluating ladder rungs is from top to bottom. Each ladder rung is scanned and evaluated one after the other starting from the top. The cyclic scan based operation is further discussed in Chapter 2.

5.Function block elements such as timers, counters and shift registers can be Connected into a ladder diagram provided that their inputs and outputs are Boolean variables. Function block elements are discussed further in Chapter 5.

An example ladder diagram is shown in Fig. 1.6. A ladder diagram is drawn as a set of rungs with each one representing a control action. An input contact represents the state of a Boolean variable. In the first ladder rung the inputs labeled A and *B* are connected in series and represent a two-input AND function. The output f_1 is 1 (true) if A is 1 and *B* is 1. In the notation of Boolean algebra the AND function is written as $f_1 =$ AB.

The second ladder rung represents a three-input (i.e. the inputs *C, D* and *F)* AND function in which two normally closed (N/C) contacts are used. The third ladder rung represents a two-input OR function. This time the contacts have been labeled (identified) as switch 1 and switch 2 and the output as the Boolean variable 'pump motor'.

The names that are used to reference Boolean variables are called 'identifiers'.

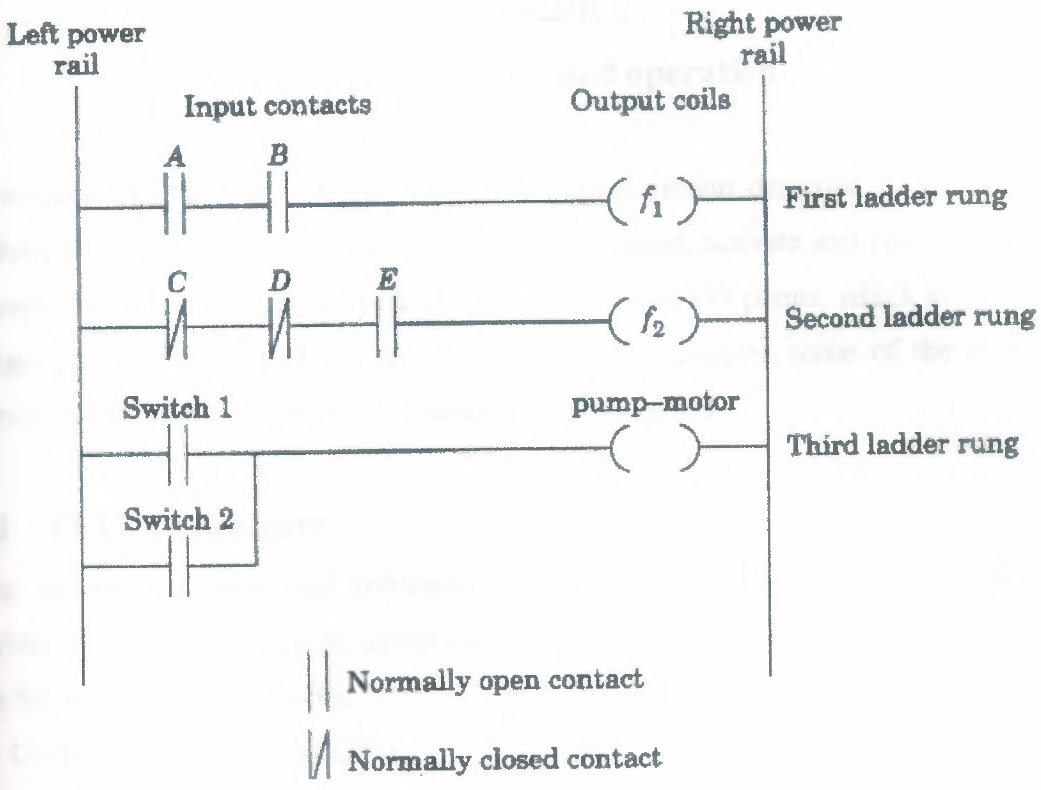


Figure **1.6** An example ladder diagram.

In Fig. 1.6 the names switch 1, switch 2 and pump motor which refer to input and utput devices are examples of identifiers. The naming of contacts and coils in this way ids the readability of a ladder diagram. However, note that each identifier has to be ssigned a unique memory address used to identify an I/O point. The binary, octal and exadecimal number systems (see Appendix 2) are widely used in the direct referencing f I/O points.

CHAPTER 2

Design, structure and operation

A modern PLC is a microprocessor-based control system designed to operate in an industrial environment. PLCs are programmed to sense, activate and control industrial equipment and therefore incorporate a large number of I/O points, which allow a wide range of electrical signals to be interfaced. In this chapter some of the important concepts behind the design of PLC hardware are presented.

2.1 PLC architecture

The internal hardware and software configuration of a PLC is referred to as its architecture. Being a microprocessor-based system the design of a PLC is based around the following building blocks:

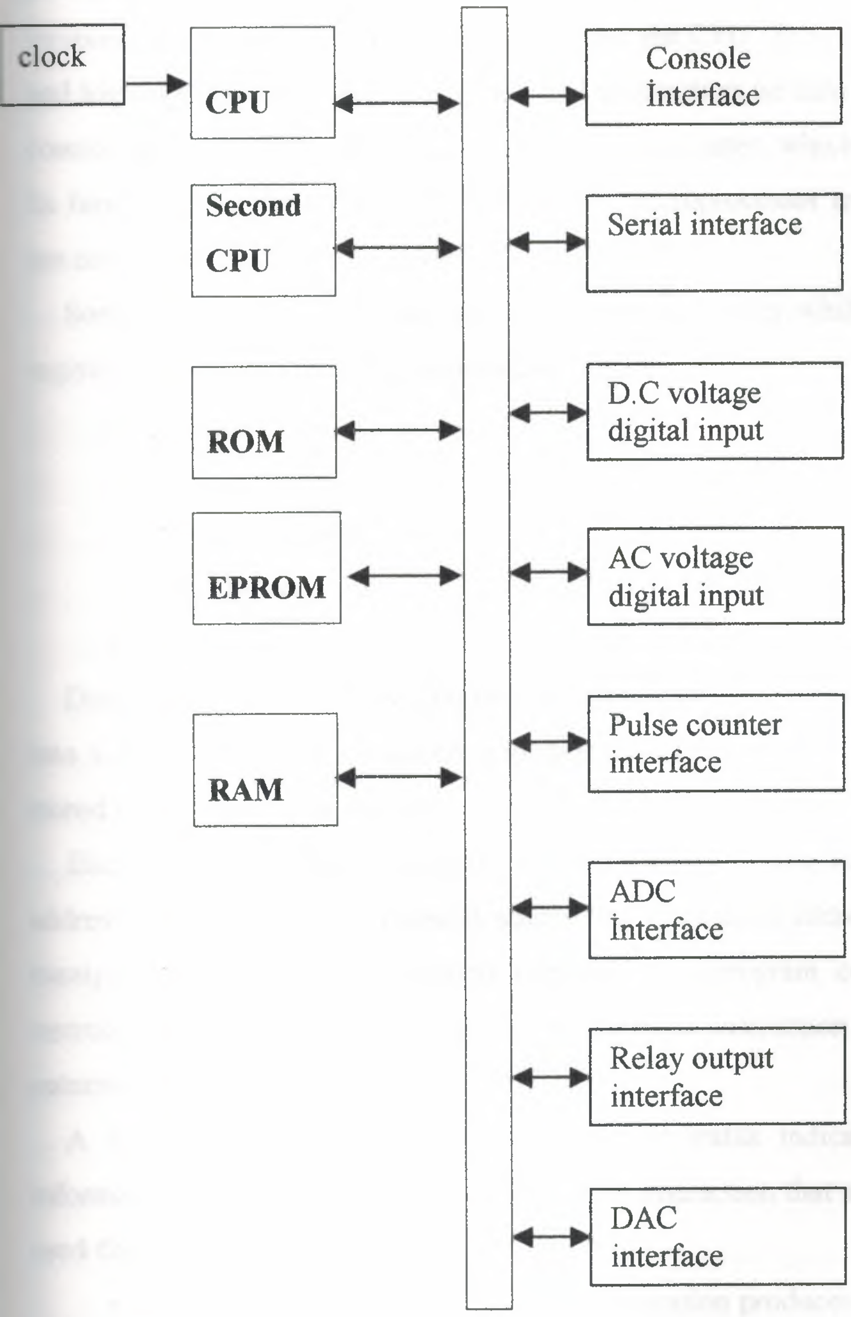
- Central processing unit (CPU)
- Memory
- Input/output interface devices

These elements are semiconductor integrated circuits (ICs). They are interconnected by means of a bus as shown in Fig. 2.1. A bus is a group of lines over which digital information (e.g. 8 bits of data) can be transferred in parallel. In most systems there are three distinct buses: the data bus, the address bus and the control bus.

A program written by the user determines the operation of a PLC system. High-level programs such as ladder diagrams are converted into binary number Instruction codes so that they can be stored using digital memory devices such as RAM (random access memory) or programmable ROM (read only memory). Each successive instruction is fetched, decoded and executed by the CPU.

2.2 CPU

A CPU generally takes the form of a single microprocessor device. The function of the CPU is to control the operation of memory and I/O devices in the system and to process data in accordance with the program. To do this it requires a clock signal to sequence its internal operations.



BUS (address data and control)

Figure 2.1 Simplified architecture of a PLC system.

The essential elements of a microprocessor are:

- •Registers
- •Arithmetic logic unit (ALU)
- •Control unit

A register is a byte (8 bits), word (16 bits) or long word (32 bits) of memory which is part of the microprocessor as opposed to general purpose memory. A register is used for temporary storage of data and addresses within the CPU. The ALU performs arithmetic and logical operations such as addition and subtraction on data stored in registers. The control unit is basically a set of counters and logic gates, which is driven by the clock. Its function is to control the units within the microprocessor to ensure that operations are carried out in the correct order.

Some CPU registers are accessible to the programmer while others are not. Some registers common to most microprocessor devices are:

- Data registers
- Address registers
- A program counter
- A flag register
- A stack pointer

Data registers hold data, which is to be operated on by the ALU. A bit pattern moved into a data register can be added, subtracted, compared, etc., with another bit pattern stored in a separate data register.

Each memory location in RAM and ROM for storing data is given an unique address. The programmer to specify source and destination addresses of data items to be manipulated can use CPU address registers. The program counter (also called the instruction pointer) holds the address of the next instruction to be executed and is automatically incremented by the CPU.

A flag register is a collection of single-bit status indicators (flags) that holds information about the result of the most recent instruction that affects them. Commonly used flags are:

- Carry bit Set to 1 if a binary addition operation produces a carry or a subtraction operation produces a borrow.
- Zero bit Set to 1 whenever the result of an operation is zero.
- Negative bit In signed binary arithmetic it indicates the sign (positive or negative) of the result.
- Overflow Set to 1 when the result of an arithmetic operation cannot be represented in the specified register size.

A stack is a variable length data structure in which the last data item inserted is the first to be removed. A stack pointer register contains the address of the top element of the stack. The CPU uses the stack to store subroutine return addresses for example.

The execution time of each instruction code takes a specific number of clock cycles. The clock cycle time is the reciprocal of the clock frequency. For example, a 10 MHz clock has a clock cycle of 0.1 $_{\mu}$s.

2.3 Memory

Memory devices store groups of binary digits (usually bytes) at individual locations identified by their own unique addresses. Memory devices are ICs having an address input (commonly 16 bits wide) and an in/out data port (commonly 8 bits wide). There are two main types of memory, namely RAM (random access memory) and ROM (read only memory). The number of binary digits it can hold determines the storage capacity of a memory device. A 1K-byte memory device is capable of storing 1024 (i.e. 210) bytes.

A large proportion of the total addressable memory space is devoted to RAM, which is capable of having data written to it and read from it by the CPU. RAM is used for program and data storage. A backup battery supply is needed to retain the memory contents of RAM, as stored data is lost when the power is removed.

There are two main types of RAM, namely SRAM (an abbreviation for static RAM) and DRAM (an abbreviation for dynamic RAM). Static RAM holds data in flip-flop type cells, which stay in one state until rewritten. SRAM has a fast data access time of typically 10—20 ns. Dynamic RAM requires special circuitry to provide a periodic refresh signal in order to maintain the stored data. This is because DRAM technology stores data in capacitor type cells, which must be periodically refreshed as capacitors discharge as time increases. DRAM is cheaper to manufacture but because a refresh signal is used access time is slower than that of SRAM, typically 50—60 ns.

The PLC operating system (i.e. the program that allows the user to develop and run applications software) is stored in a type of memory referred to as ROM. Once programmed, this type of memory can only be read from and not written to but does not lose its contents when the power is removed.

Common types of read only memory that are user programmable are EPROM (erasable programmable read only memory) and EEPROM (electrically erasable programmable read only memory). EPROM's are programmed using a dedicated programmer and erased by exposing a transparent quartz window found in the top of ach device to ultraviolet light. The erasing process clears all memory locations and akes between 10 and 20 minutes. EEPROM is erased using electrical pulses rather than ıltraviolet light. Most PLC systems provide facilities that allow application software to e stored in either EPROM or EEPROM.

A memory map is used to indicate how address locations are allocated to ROM, RAM and input/output devices. If I/O devices are placed in the memory address space they are said to be memory mapped and access to I/O points is via load and store memory instructions. This has the advantage that common sets of instructions are used for memory and I/O operations. A memory map of a typical PLC system is shown in Fig. 2.2.

2.4 Bus

A bus can be considered as a set of lines over which digital information (e.g. 16-bit address or 8-bit data) can be transferred in parallel. In most systems there are three distinct buses:

- Data bus
- Address bus
- Control bus

The data bus is a bi-directional path on which data can flow between the micro-processor, memory and I/O. An 8-bit microprocessor has a data bus, which is 8 lines wide. A 16-bit microprocessor has a data bus, which is 16 lines wide.

The address bus is a unidirectional set of lines, which carry binary number addresses. The CPU generates addresses during the execution of a program to specify the source and destination points of the various data items to be moved along the data bus. An address identifies a particular memory location or I/O point.

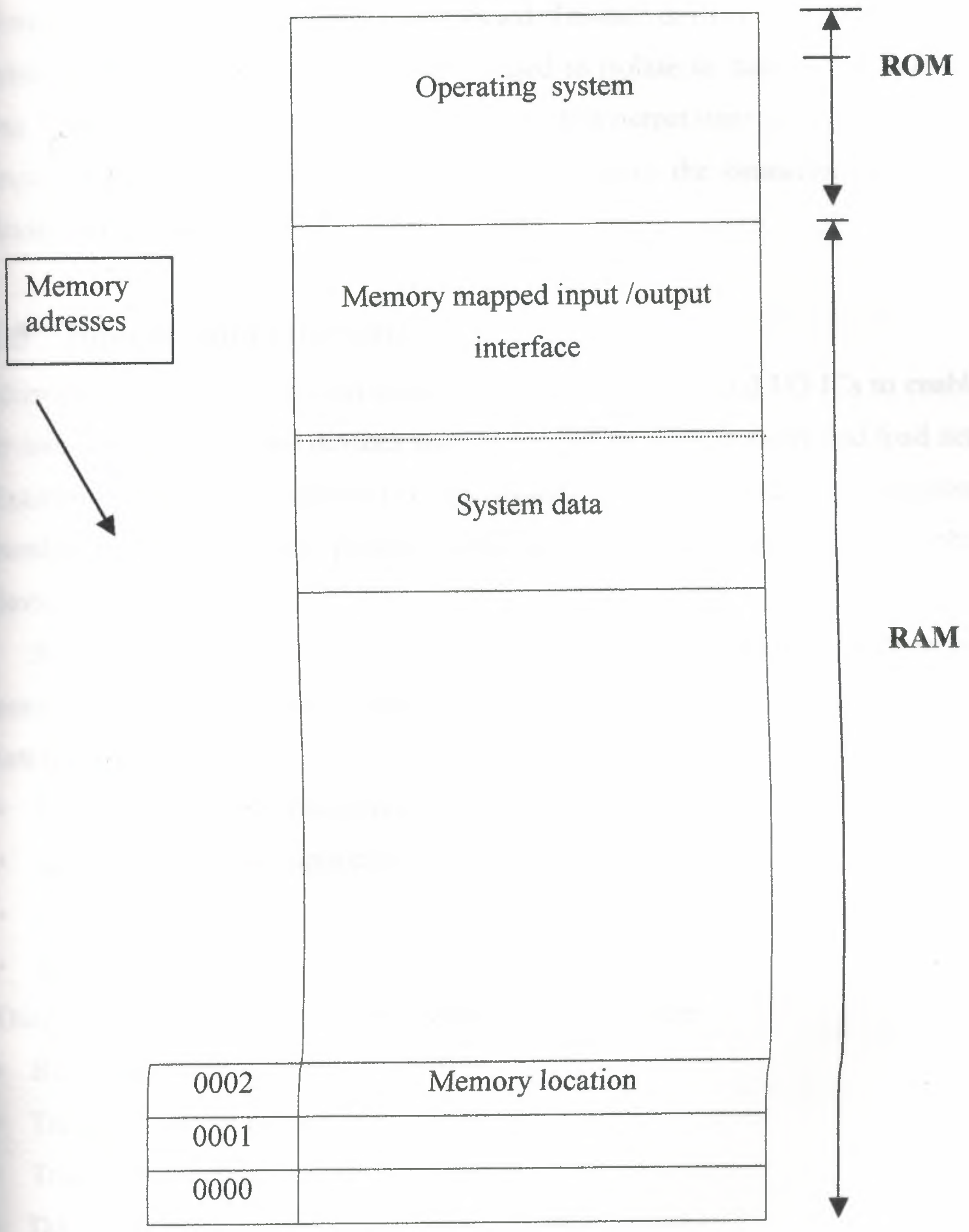


Figure 2.2 Memory map.

The control bus consists of a set of signals generated by the CPU to control the devices in the system. An example is the read/write control line, which selects one of two operations, either a write operation where the CPU is outputting data on the data bus or a read operation where the CPU is inputting data from the data bus.

Digital devices sharing a bus must be tri-state. This means that when the output lines are not in use they are put into a high impedance state so that they will not load the bus. Therefore an output line can have one of three possible conditions, which are logic 0 low), logic 1 (high) and output disconnected. Tri-state devices incorporate a chip select also called chip enable) input, which is used to isolate its data output lines from the ous. When the chip select line is not active the data output lines are placed in to the high mpedance or tri-state. The control bus co-ordinates the connection of the various devices to the data bus.

2.5 Input/output interfaces

Microprocessors are supported with special purpose peripheral I/O ICs to enable them to interface with external devices such as keypad displays, sensors and load actuators. Examples of special purpose I/O ICs include keyboard controllers, programmable parallel interface devices, programmable serial interface devices and counter/timer devices.

As far as the user is concerned, it is the front end circuits to which sensors and actuators are connected that is important. Input points can include the following types of interface:

- D.C. voltage digital input circuit
- A.C. voltage digital input circuit
- Pulse counter circuit
- ADC interface

Output points can include the following types of interface:

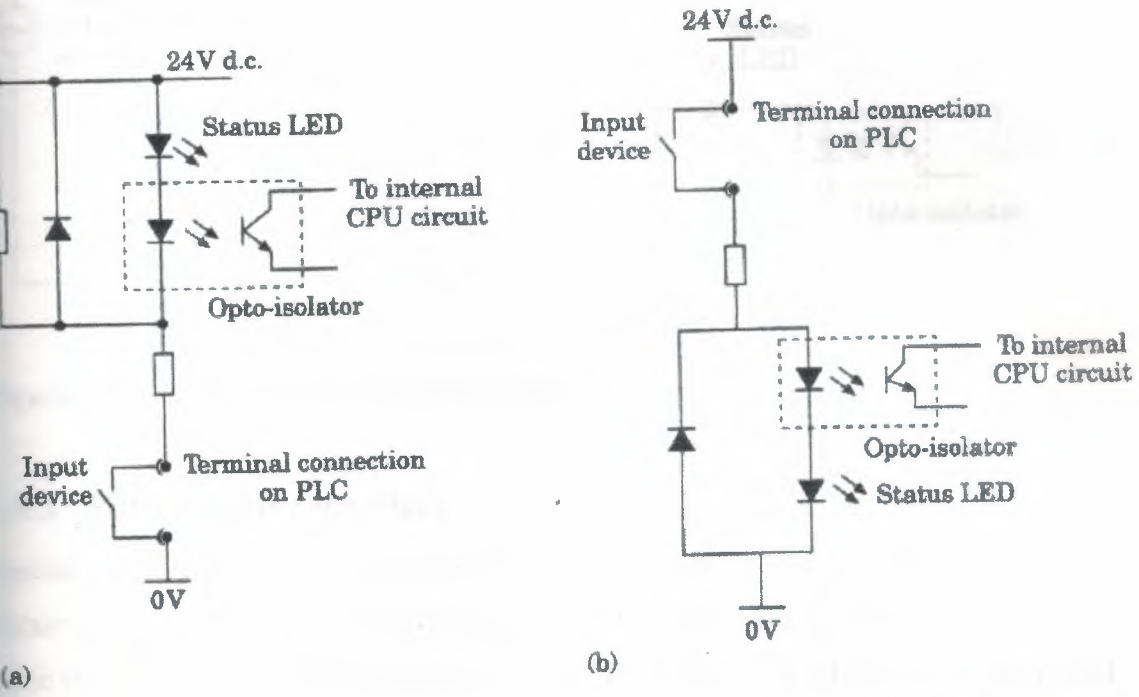
- Relay output circuit
- Transistor output circuit
- Triac output circuit
- DAC interface

2.5.1 D.C. Voltage digital input circuit

Figure 2.3 illustrates typical 24 V D.C. input circuits for connecting current sinking and sourcing input devices. With the sink input interface the input device when turned on connects the circuit to the 0 V line of the D.C. supply. Current then flows through the status LED (light-emitting diode) used to indicate the current logic state of the input point and the opto-isolator.

An opto-isolator combines an LED and photoelectric transistor. When current is passed through the LED it emits light, causing the photoelectric transistor to turn on.

ovided a separate supply is used for the LED and photoelectric transistor circuits a ery large degree of isolation is maintained between the two components.

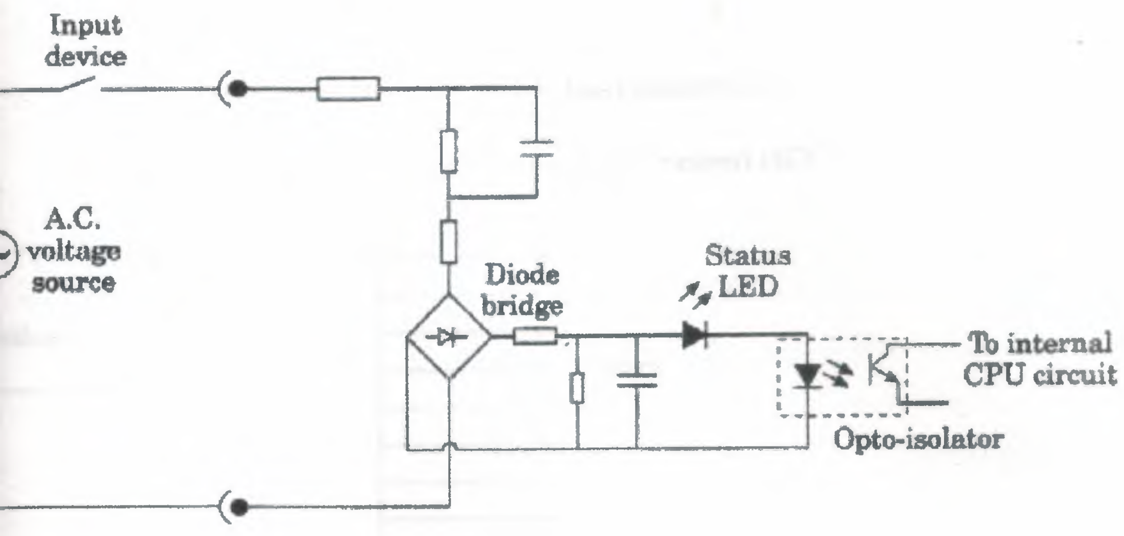


igure 2.3 D.C. voltage digital input circuits: (a) sink input circuit and (b) source nput circuit.

With the current sourcing interface the input device, when turned on, connects the ircuit to the positive polarity of the supply (i.e. 24 V). Current flows from the supply hrough the status LED and opto-isolator circuit.

.5.2 A.C. Voltage digital input circuit

igure 2.4 illustrates an a.c. voltage digital input circuit. A full-wave diode bridge ircuit is used to convert the a.c. input signal into a rectified D.C. signal. The status ED and opto-isolator are turned on by the rectified D.C. signal.



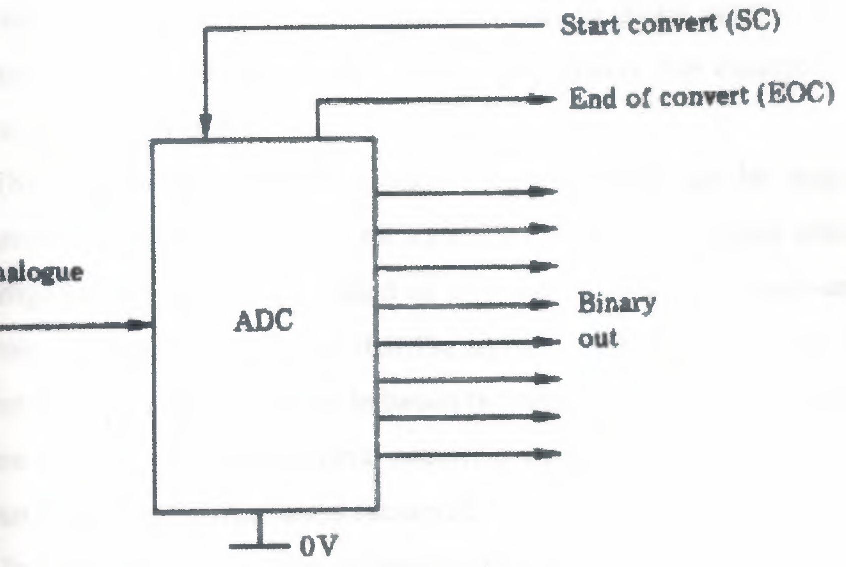
gure 2.4 A.C. voltage digital input circuit.

.3 Pulse counter interface

ecial purposes ICs are available which integrate a high-speed digital pulse counter cuit and buffer memory. Input modules based around these ICs can be used to read lse streams from an input device such as a shaft encoder. The advantage of using such dules is that high-speed pulse counting is not slowed down by the cyclic scan eration of a PLC because the counter circuit operates independently. Provided the emory buffer is accessible to the ladder program it can be read at the point in time en the pulse count value is required. It is also possible to use a technique whereby the gh-speed counter circuit interrupts the CPU when the count value reaches some pre-fined limit.

.4 Analogue to digital converter (ADC) interface

n analogue input circuit will incorporate an analogue to digital converter (ADC) vice, as shown in Fig. 2.5. An ADC accepts an analogue input signal and converts it o an output binary value that corresponds to the level of the analogue input. Most OC devices incorporate a start convert (SC) pin, an end of convert (EOC) pin and a mple and hold circuit. When the start convert signal is pulsed the ADC samples and lds (stores) the analogue value input to the device at that time. The ADC converts the alogue signal into a digital value and then produce an end of convert signal when the nversion is complete.



ure 2.5 Analogue to digital converter (ADC).

The time taken by the ADC to convert an analogue signal into a digital number is led the conversion time. There are a number of different types, of AD. Devices nmercially available, which work on different principles. One common type uses a hnique called successive approximation to obtain short conversion times. This olves comparing the output of a digital to analogue converter (DAC) with the voltage be converted by making a series of successive guesses (i.e. approximations) at the ue of the binary number required. The fastest type of ADC is the flash converter, ich determines simultaneously all the bits for the digital number representing the alogue input level.

According to sampling theory, an ADC should sample an input signal at lead twice fast as the input's highest frequency component. If the input signal i sampled too wly, aliasing occurs. In this case, a high frequency signal is represented by an oneous lower frequency value. A technique to prevent an aliasing. Error condition curring is to precede the ADC with an anti-aliasing filter which band limits the input nal to half the sample frequency.

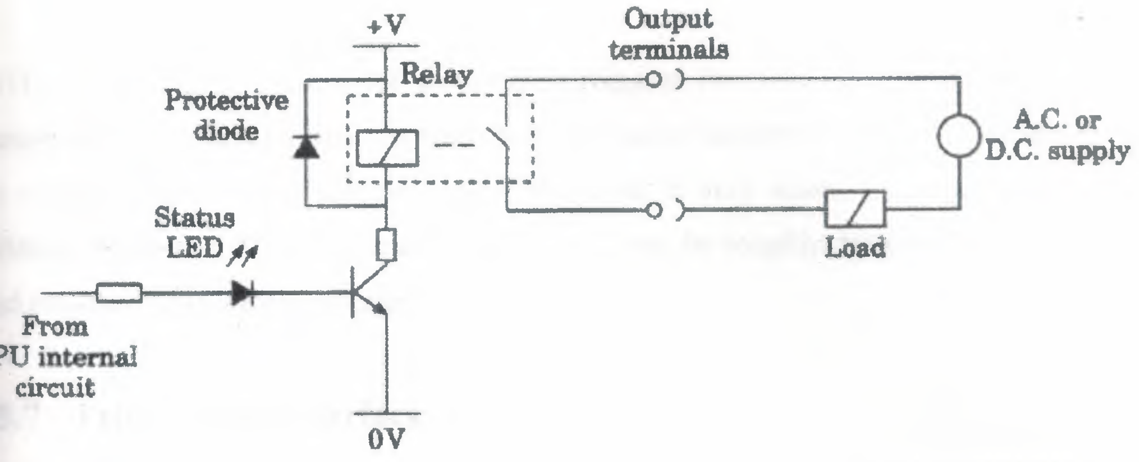
Speed of operation is not the only consideration when selecting an ADC. An 8-bit OC approximates the analogue input voltage into one of 28 or 256 discreet levels lled quantization levels. If there are 256 quantization levels including, zero, there will 255 steps between them. This means that, for an 8-bit ADC designed so that the aximum of the analogue input is limited to 5 V, the quantization interval (i.e. solution) is 5/255 or 19.6 mV. The only way to obtain more quantization levels is to crease the number of bits used in the conversion. For example, a 12-bit ADC will ve 212 or 4096 levels.

The input range of an analogue voltage interface can be unipolar (0—10 V for ample) or bipolar (± 10 V for example). In addition, input channels can often be nfigured as either single-ended or differential inputs. A single-ended input has one rminal connected to 0 V so that the signal varies with respect to 0 V. A differential put measures the difference between two signal leads. Differential inputs can provide ise immunity as a noisy signal occurring equally on both signal leads is cancelled out hen the voltage difference is measured.

In industrial applications analogue currents rather than voltages are often used. This because the resistance of the cable reduces a voltage applied to one end of a cable hereas the current remains fixed. The resistance of a cable is proportional to its length nd so the reduction of voltage increases with length. A commonly used current input nge is 4—20 mA. By using this range a broken cable gives a result of 0 mA, which is lentified as an error. Measuring the voltage across a known resistance through which e current is passing and applying Ohm's law performs the input of current to an ADC.

.5.5 Relay output circuit

Figure 2.6 illustrates the circuit for a relay output interface. The NPN transistor is sed to switch current through the relay coil to close its contact. The internal circuit of he PLC controls the transistor. The diode is connected across the relay coil to protect he transistor from the effects of back e.m.f. This is the reverse voltage developed in the elay coil, which causes an inductive current, which opposes the normal flow of current. Both A.C. and D.C. supplied loads can be connected through the relay output terminals.



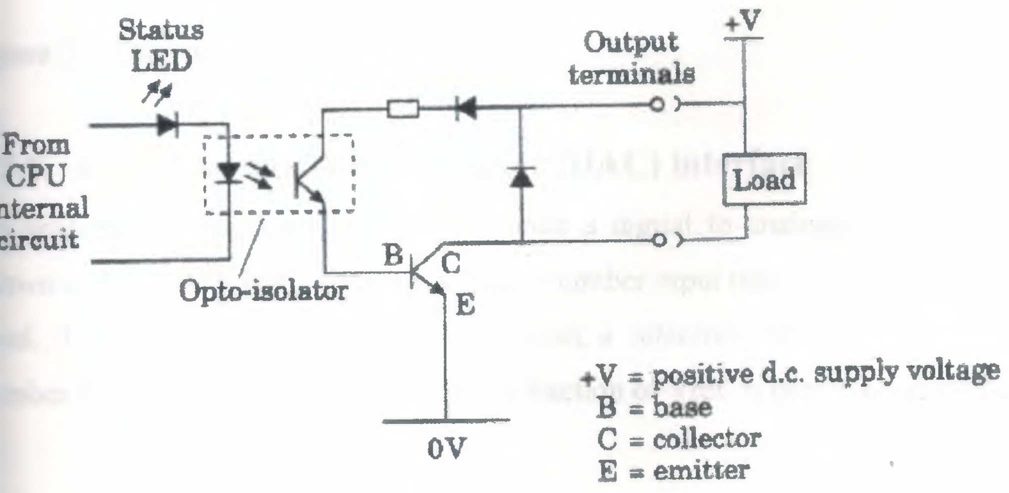
gure 2.6 Relay output circuit.

5.6 Transistor output circuit

A transistor output circuit is used for switching d.c. Voltages. Figure 2.7 shows a pical opto-isolated NPN transistor switch output circuit. The principle of operation of ransistor switch ideally relies on there being:

•An open circuit (i.e. infinite resistance) between the collector and emitter when the se—emitter circuit is not forward biased

•A short circuit (i.e. zero resistance) between the collector and emitter when the se—emitter circuit is forward biased

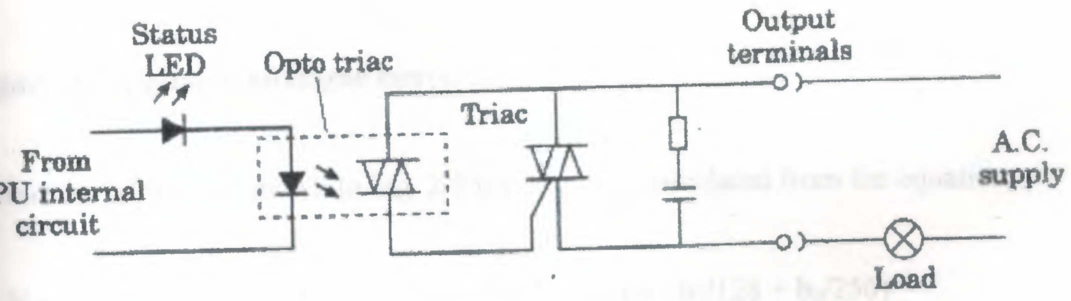


gure 2.7 NPN transistor switch circuit.

he ideal switching action of a transistor requires that the base current be large ugh for the collector current to reach its maximum or saturation value. At saturation, voltage drop between the collector and emitter is very small. Consequently, the ector is close to 0 V and the collector current may be roughly determined from the d resistance and supply voltage.

.7 Triac output interface

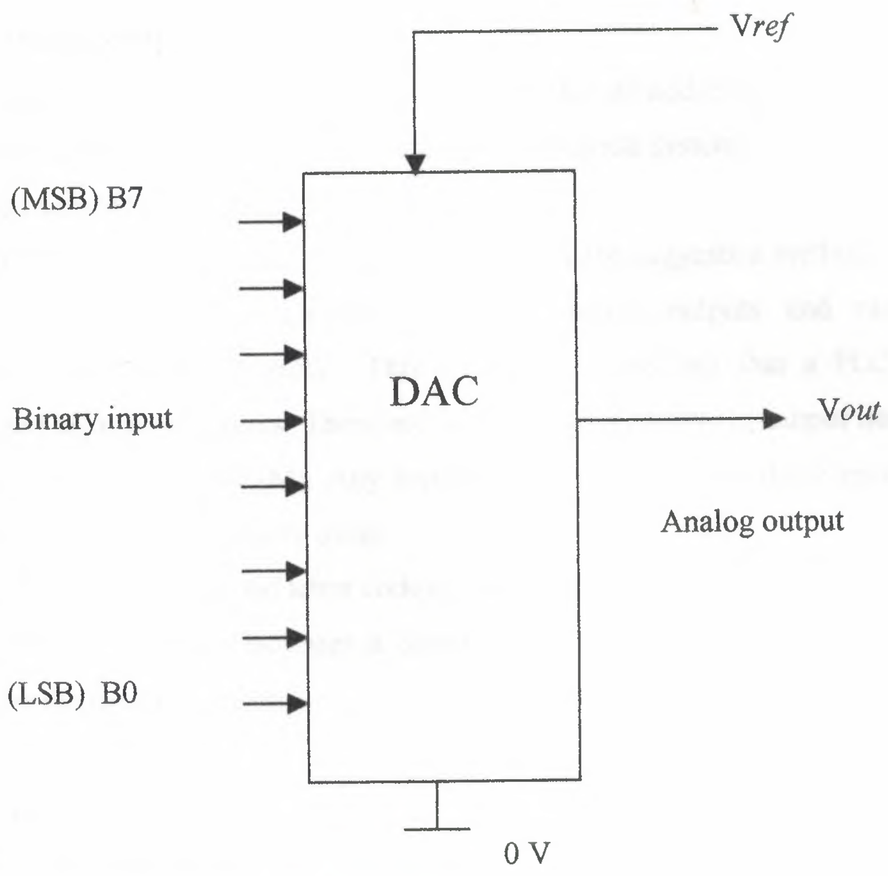
riac output device is used for switching a.c. voltages. Triac devices that incorporate a o crossing circuit to monitor the a.c. cycle have the advantage that they turn off at o current. Inductive loads should be turned off at a zero current crossing point to vent interference. A typical opto-isolator-based triac output circuit is shown in Fig.



gure 2.8 Triac output circuit.

5.8 Digital to analogue converter (DAC) interface

n analogue output module will incorporate a digital to analogue (DAC) device as own in Fig. 2.9. A DAC converts a binary number input into a proportional Analogue vel. The analogue output is produced from a reference voltage Vref. The binary umber input to the DAC determines what fraction of Vref. is presented at the output.



ire 2.9 Digital to analogue converter (DAC).

or the 8-bit DAC shown in Fig. 2.9 the output is calculated from the equation

$$V_{out} = V\,(b_7/2 + b_6/4 + b_5/8 + b_4/16 + b_3/32 + b_2/64 + b_1/128 + b_0/256)$$

Vhere the bits b_0 to b_7 can take the values 0 or 1 and are the binary inputs. Here b_7 is nost significant bit (MSB) and b_0 the least significant bit (LSB).

n 8-bit DAC has 28 or 256 quantization levels and 255 steps between them. This ns that for an 8-bit DAC with a reference voltage of 10 V the quantization interval resolution) is 10/255 or 39.2 mV. A 12-bit DAC with a reference voltage of 10 V a quantization interval of 10/4095 or 2.4 mV.

The output of a DAC begins to change when it receives a new data value at the input. period required before the output is valid is called the settling time. Generally, ling times are small, being specified in microseconds.

6 Input/output assignment

ch input and output connection point on a PLC has an address used to identify the I/O . Each manufacturer uses a proprietary identification system, which is related to the e and number of input/output options possible.

The IEC 113 1-3 programming languages standard suggests a method for the direct presentation of data associated with the inputs, outputs and memory of a ogrammable logic controller.' This is based on the fact that a PLC memory is ganized into three regions. These are input image memory (I), output image memory) and internal memory (M). Any memory location including those representing I/O s can be referenced directly using

(first letter code) (second letter code)(numeric field)

ere the % character indicates a directly referenced variable. The first letter code ecifies the memory region:

I = input memory

0 = output memory

M = internal memory

e second letter code specifies how memory is organized:

X = bit

B = byte

W = word

D = double word

L = long word

If a second letter code is not given it is assumed to be a bit.

The numeric field component is used to identify the memory location. It supports the oncept of I/O channels since numeric fields can be separated using a period.

Some examples of direct referencing of I/O memory as advocated in the IEC 1131-3 andard are:

%X1 (* input memory bit 1 *)

%I1 (* also input memory bit 1 *)

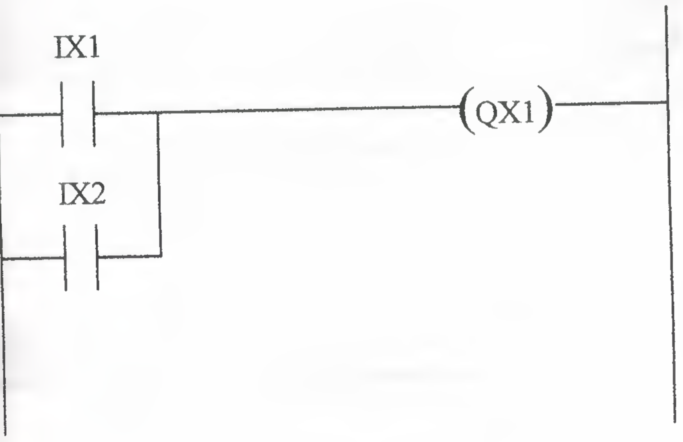
%1B2 (* input memory byte 2 *)

%IX1O.4 (* input byte address 10 bit 4 *)

%QX1 (* output memory bit 1 *)

Jsing names that identify the purpose of each contact and coil in a ladder diagram aids

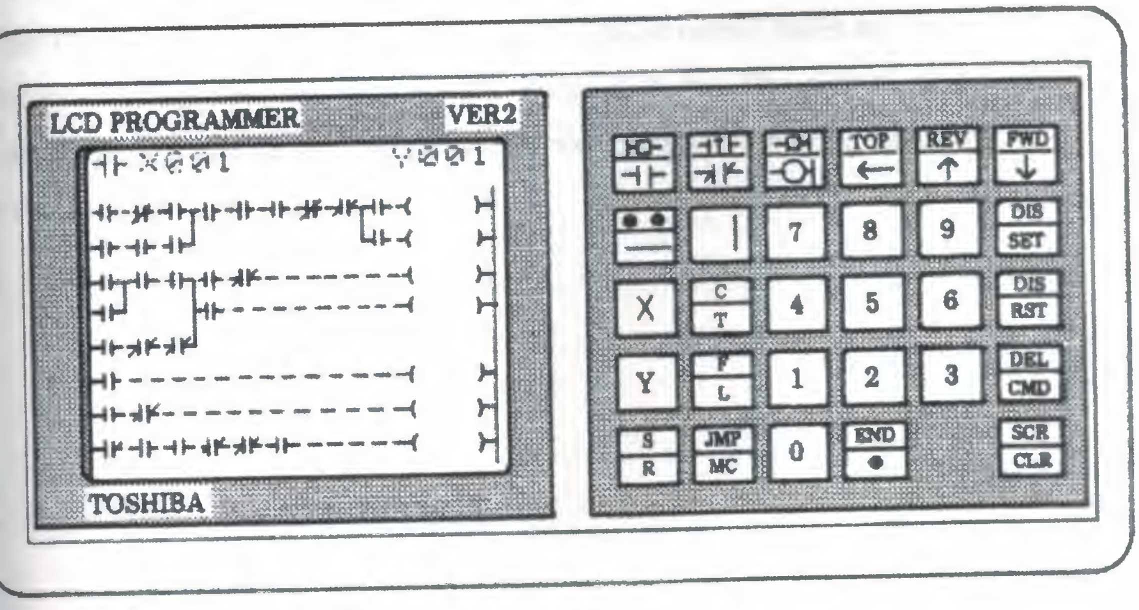
ability. For example, the contact given the name 'start_fan' clearly identifies the
ose of the input signal. This variable may be the input point directly referenced as
X1 (e.g. input memory bit 1). Input and output memory locations can be assigned
ctly on a ladder diagram as shown in Fig. 2.10.

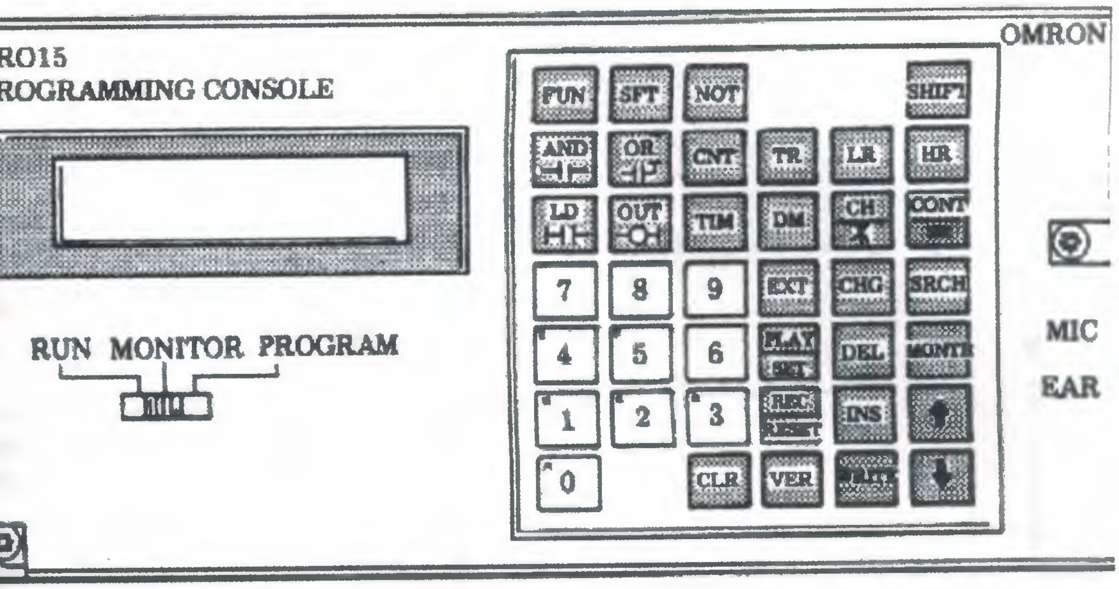


ure 2.10 Assignment of I/O points.

Keyboard and display

er programs can be entered into RAM using a program console unit consisting of a
board and display. Some typical hand-held PLC programming consoles~
orporating liquid crystal displays for viewing ladder code and diagrams are shown in
. 2.11.

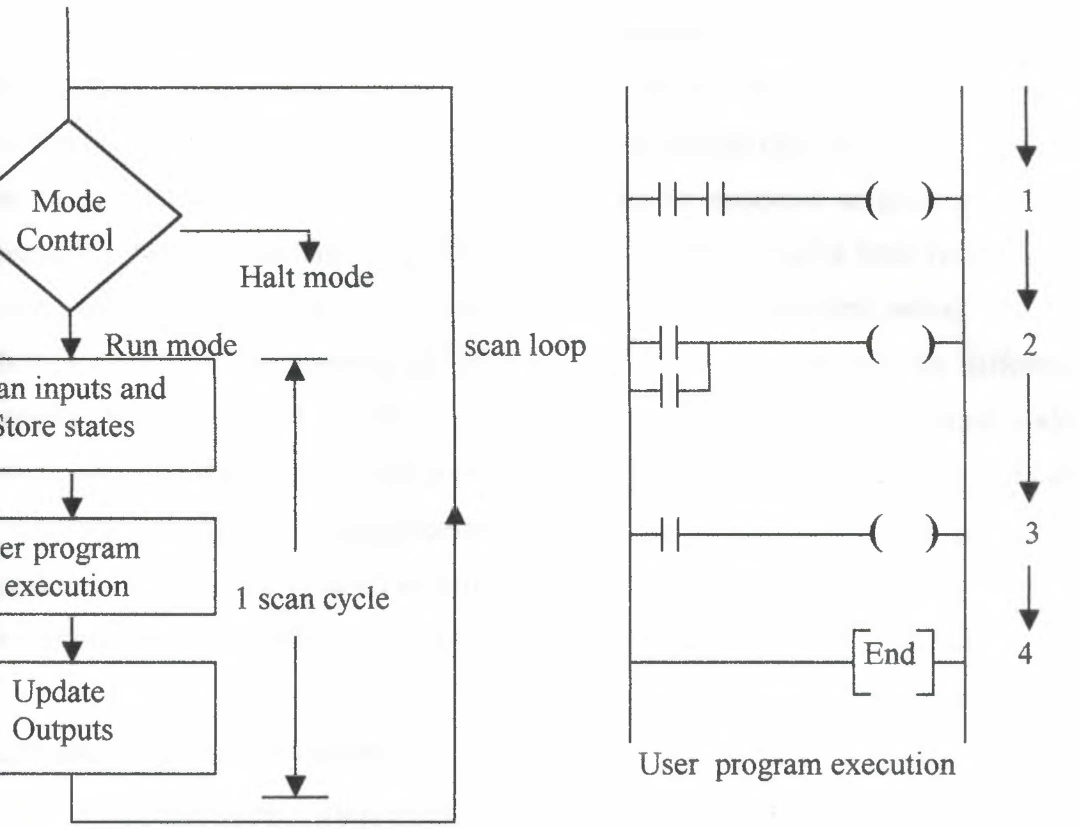




ıre 2.11 Typical PLC program consoles.

Program execution

most common approach taken to executing a PLC program is to use a cyclic scan or n program loop such that periodic checks are made on the input values. Figure 2.12 strates the way a ladder program can be executed. The program loop starts by nning the inputs to the system and storing their states in fixed memory locations rred to as input image memory. The ladder program is then executed rung-by-rung, ting at the first rung. Scanning the program and solving the logic of the various der rungs determine the output states. The updated output states are stored in fixed nory locations referred to as output image memory. The output values held in nory are then used to set and reset the physical outputs of the PLC simultaneously at end of the program scan.



re 2.12 executing a ladder program.

he time taken to complete one cycle is called the cycle or scan time. It depends on ength and complexity (e.g. the number and types of functions used) of the program. sequently, manufacturers specify an average value. For example, the average ution speed for Mitsubishi Fl series PLCs is specified as 12 us/step, with a mum program capacity of 1000 steps. This yields an average cycle time of 12 ms 000 steps. The average execution speed for Mitsubishi F2 series PLCs is specified us/step, with a maximum program capacity of 2000 steps (F2-40,60M) and 1000 (F2-20M).[5] This yields an average cycle time of 7 ms for 1000 steps.

can-based execution has a number of limitations when the PLC system has to ond to events within a specified time period. PLC systems, which accommodate rupts, can be used to spontaneously respond to a specific event such as an alarm. nterrupt is a special control signal to the CPU which tells it to stop executing the ram in hand and start executing another program stored elsewhere in memory (i.e. rupts the sequential execution of the ladder program).

Multitasking and multiprocessing

nced PLC systems incorporate a processor scheduler to enable multitasking and processing. Multitasking is the running of two or more tasks (also called sses) on a single processor such that they share processor time so that they appear n in parallel. Different tasks of a program can be executed at different rates. equently, time critical tasks (e.g. the monitoring of the state of a limit switch) can ven a high priority and scheduled to be executed within a fixed time period.

ultiprocessing is the running of tasks or processes simultaneously on different ssors. In this context a task or process is considered to be a separate code nent, which performs a discrete activity (a program organizational unit, or POU in terminology). With a multiprocessing system, tasks such as 'closed loop PID ortional, integral, derivative) control' and 'ladder circuit control' can be mapped separate processors and run concurrently, communicating with one another.

10 Development systems

C development system would normally comprise a host computer connected via a l communications RS232C port to a target PLC. The host computer provides a vare environment to perform editing, file storage, printing and program operation toring. Typically, the process of writing a program to run on the PLC consists of:

sing an editor to write/draw a source program

onverting the source program to binary object code which will run on the PLC's icroprocessor

own loading the object code from the host PC to the PLC system via the serial ommunications port

he editor enables programs to be created and modified on the host computer in r graphical form, such as a ladder diagram, or text form, such as mnemonic code. ures such as cut and paste, copy program block and address search are standard.

CHAPTER 3

Input devices

This chapter provides a brief overview of sensing devices used in manufacturing rocesses. Sensing devices with a digital output can be connected directly to the digital nput port of a PLC. Sensors, which produce an analogue voltage signal, are connected ısing an analogue to digital converter.

3.1 Digital devices

The operation of a PLC may be based on signals received from digital switching levices, which detect an event occurring. For example, the presence of an object on a onveyor belt can be detected using a proximity switch or a photoelectric detector. Some ommonly used digital switching devices are described below.

3.1.1 Pressure and temperature switches

Pressure switches monitor pressure and have a contact which changes state when a pre-et pressure threshold value is reached. When the pressure level falls below the threshold alue the contact resumes its initial position. Pressure switches are used in pneumatic and ıydraulic applications.

Thermostats monitor temperature[1]and have a contact which changes state when a pre-set emperature threshold value is reached. When the temperature value falls below the hreshold value the contact resumes its initial position. Thermostats are used in a wide ange of applications including the temperature monitoring and control of machines and eating installations. Many thermostats incorporate some hysteresis, which means that they switch on and off at different temperatures. This is referred to as differential regulation etween two threshold values.

3.1.2 Proximity switches

Proximity switches are used for non-contact object detection. They integrate a Sensing element and a transistor switch output circuit. The output changes state. When the object is in the vicinity of the sensor element. Proximity switches are classified as either inductive or capacitive, which relates to the type of sensing technology used. Inductive types are suitable for detecting ferrous and non-ferrous metals.

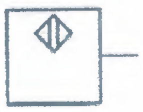
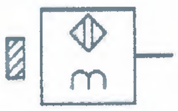
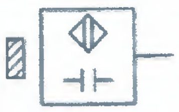
Sensor sensitive to proximity	
Inductive proximity switch	
Capacitive proximity switch	

Figure 3.1 The IEC graphical symbols for proximity switches.

Capacitive types will sense the presence of almost any material. The IEC symbols used to represent proximity switches are shown in Fig. 3.1.

The principle of operation of an inductive proximity switch is based on the generation of an alternating magnetic field, which is affected by a metal object passing within its range. A typical device would have a sensing range of 2 mm and a switching speed of 800 Hz. The principle of operation of a capacitive device is based on generating an electric field, which is modified by any object passing within its range. A typical device would have a sensing range of 1—10 mm (adjustable) and a switching speed of 400 Hz.

The d.c. switching proximity devices are available with NPN open collector and PNP open collector transistor output stages. Figure 3.2 illustrates how to connect an NPN transistor output to a PLC sink input circuit and a PNP transistor output to a PLC source input circuit.

3.1.3 Photoelectric switches

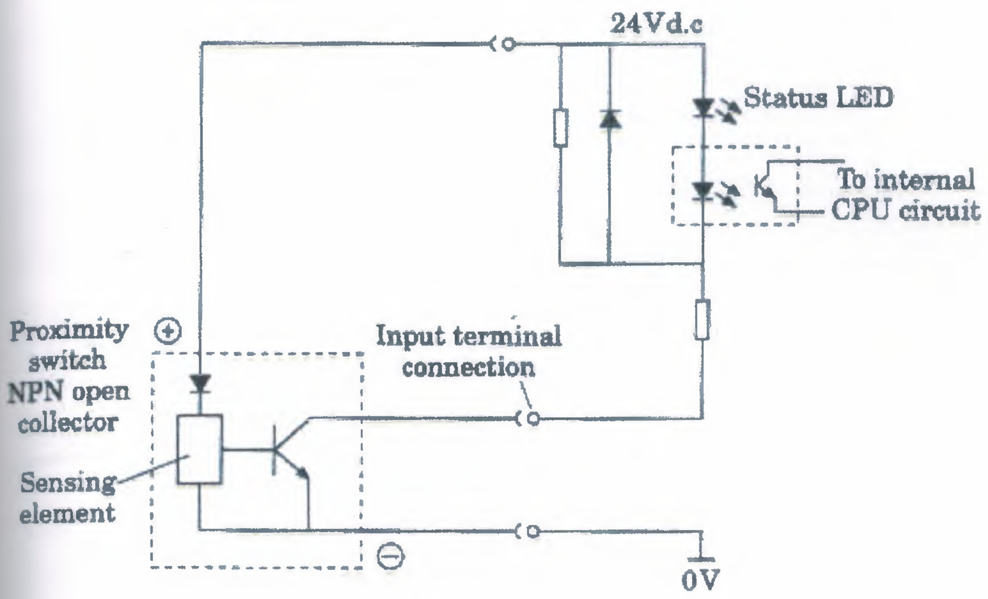
Photoelectric devices consist of a light source and a photo-receiver incorporating a transistor switch circuit. PNP and NPN transistor output devices are available. The ways in which the light source and photo-receiver can be set up to detect objects are illustrated in Fig. 3.3. Photoelectric switches are categorized as being either through beam, mirror reflection, retro-reflective or diffuse reflective.

The through-beam type of photoelectric system has a separate transmitter and receiver. An object is detected when it breaks the light beam. The through-beam photoelectric detector can be used for long-range sensing. A typical device would have a sensing range of 8000 mm.

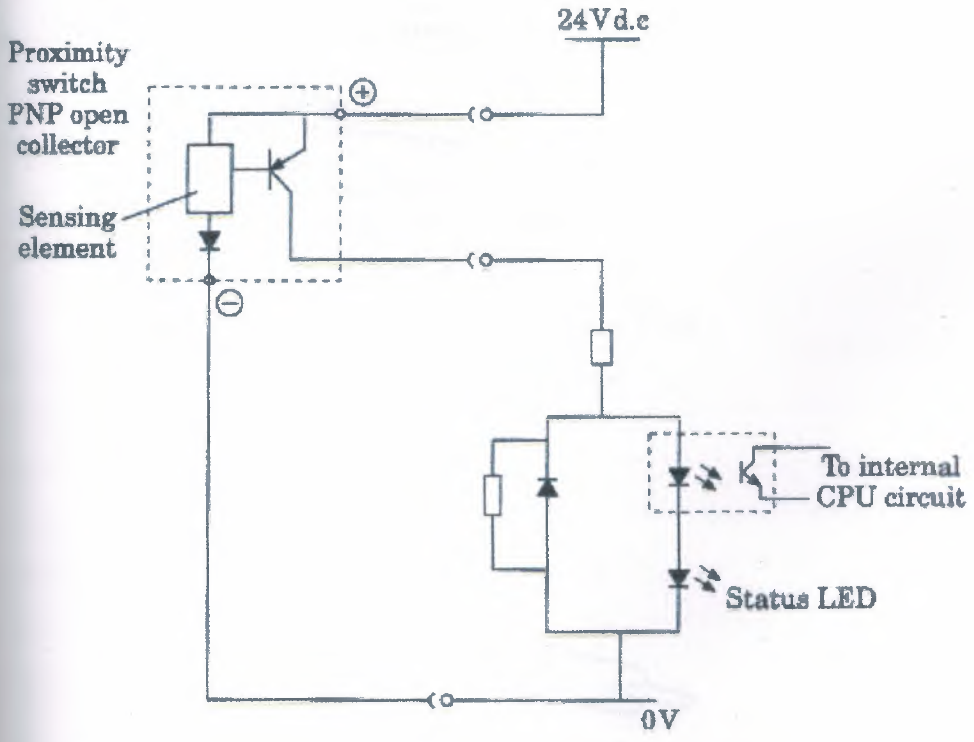
A plane surface mirror can be used to reflect back the transmitted light beam to the receiver. In this case, the object is detected when the reflected beam is broken. With a plane mirror reflector, the transmitter and receiver must be mounted such that the angle of incidence equals the angle of reflection.

The retro-reflective type of photoelectric switch uses a special type of reflector which returns transmitted incident light back in the same direction from which it was sent. This allows the transmitter and receiver to be incorporated in the same housing. With proper alignment of the photoelectric device to the reflector a typical device would have a sensing range of 2000 mm.

If a parallel beam of light is incident on a sheet of paper the light is reflected in all directions because the surface is not perfectly smooth like that of a mirror. This is an example of diffuse reflection. A diffuse reflective type of photoelectric switch contains a transmitter and receiver in the same housing and switches when the diffuse reflection level exceeds a threshold value. A typical device would have a sensing range of 100 mm.


24Vd.c
Status LED
To internal
CPU circuit
Proximity switch NPN open collector
Input terminal connection
Sensing element
0V

a)



b)

Figure 3.2 Connecting NPN and PNP transistor output proximity switches: (a) Sink input circuit connection and (b) source input circuit connection.

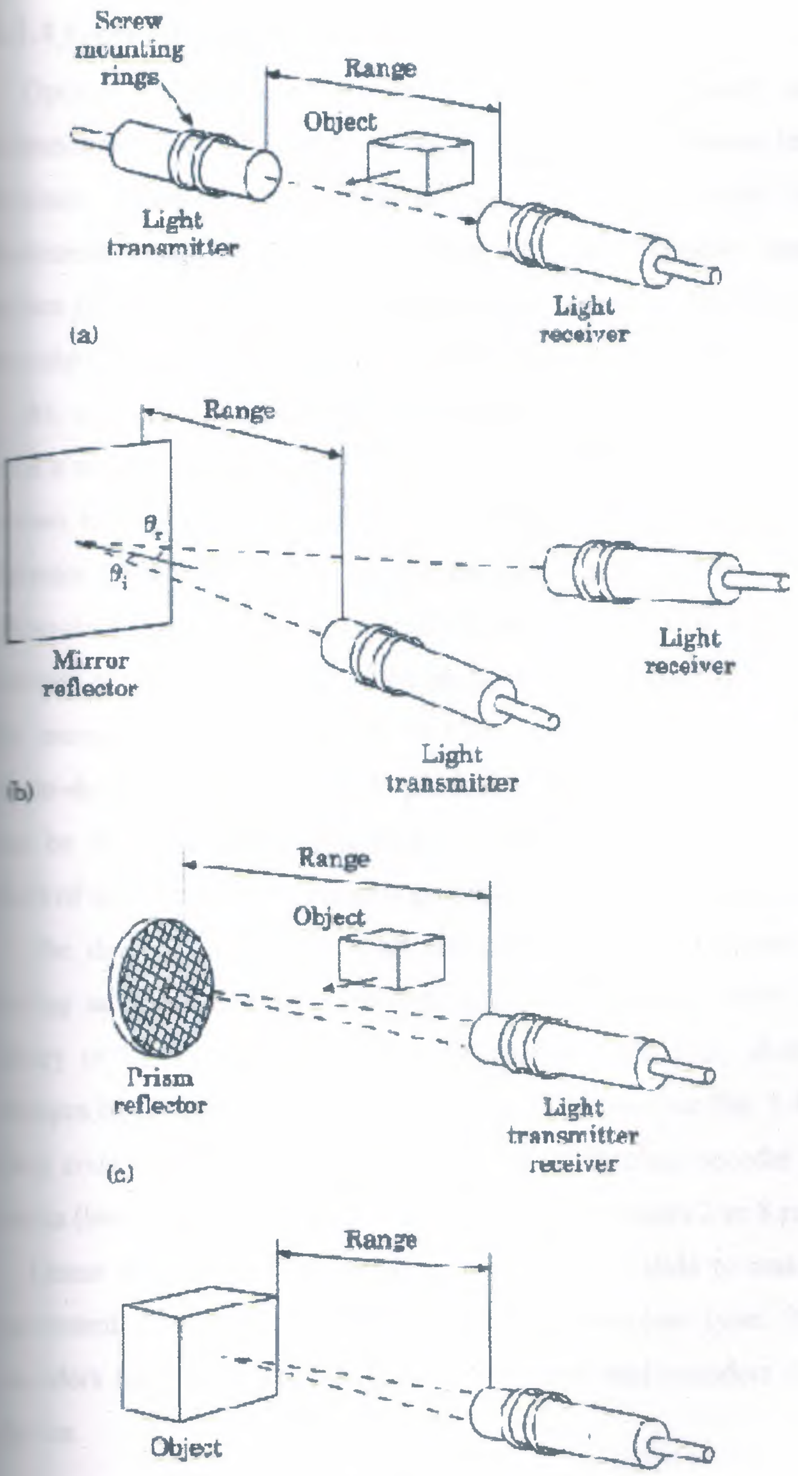


Figure 3.3 (a) through beam (b) mirror reflection (c) retro-reflection (d) diffuse reflection

3.1.4 Optical switches

Optical encoders convert either translation or rotary displacement into digital information. They operate by using a grating, which moves between a light source and detector. There are two forms of optical encoder, namely incremental and absolute. Incremental encoders produce a pulse for each resolvable change in position. Counting pulses from some reference point can make a relative measurement of position. Absolute encoders produce a unique coded number for each resolvable position.

An example of an incremental shaft encoder is shown in Fig. 3.4. The grating is a disc with a single concentric track of evenly spaced opaque and transparent regions. When light passes through a transparent region of the grating an output is obtained from the photo-detector. When the shaft is turned, the disc rotates and chops the light beam so that the photo-detector produces a series of electrical pulses. Position is measured by counting the number of pulses generated by the photo-detector as the disc rotates from a reference point. By incorporating two tracks shifted by a quarter-cycle relative to one another and two photo-detectors it is possible to produce two signals from which the direction of rotation can be determined by noting which signal rises first. If a motor the pulse rate drives the shaft of an incremental encoder is proportional to the motor's speed.

The disc of an absolute shaft encoder has several concentric tracks, with each track having an independent light source and photo-detector. With this arrangement a unique binary or Gray coded number can be produced for every shaft position. The Gray code changes by a single bit between successive positions (see Fig. 3.4). Table 3.1 compares the Gray code with binary code. Resolution of an absolute encoder depends on the number of tracks (bits). For example, a 3-bit binary output produces 2 or 8 resolvable positions.

Linear encoders use a grating in the form of a slide to make a measurement of "near movement. They may be of incremental or absolute type. Rotary and linear absolute encoders read actual position and unlike incremental encoders they do hot need a counting device.

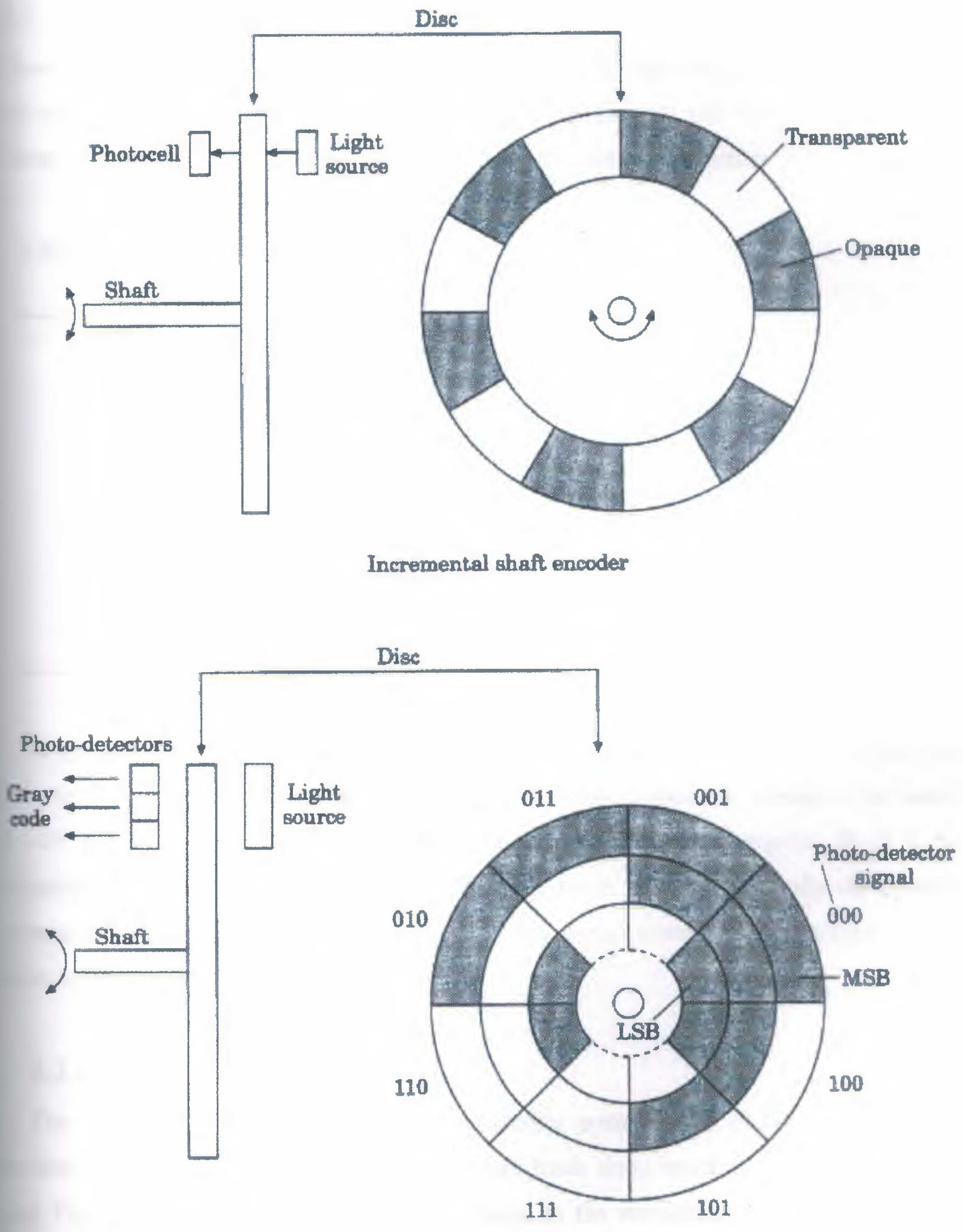


Figure 3.4 incremental and absolute snail encoders.

3.2 Analogue devices

Many applications involve monitoring analogue signals representing the variation of a physical quantity. For example, the variation of displacement can be measured using a linear variable differential transformer (LVDT), which converts a position

Table 3.1 Binary and Gray codes

Binary	Gray
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100

A typical PLC analogue input unit will accept either voltage or current signals (see Chapter 2). Current input ranges for data acquisition are commonly configured to accept 4—20 mA or 0—20 mA signals. Voltage input ranges can be unipolar *(0—5* V for example) or bipolar (± *5* V for example). The input range is selected (usually via a jumper connection) to match the sensor signal variation. Some examples of analogue voltage sensors are described below.

3.2.1. linear potentiometer

The linear potentiometer is used for measuring position and displacement. Modem devices consist of a printed circuit linear resistive track along which a slider makes contact (see Fig. 3.5*)*. The slider is mechanically linked to the movement being measured. The track of resistive material is connected across a d.c. Supply V. Assuming that the resistance is distributed linearly along the length L of the track the output voltage V_0 from the slider can be written as

$$V_O = \frac{x_i}{L} V = Kx_i$$

The output voltage is directly proportional to the position of the slider along the track.

3.2.2. Linear variable differential transformer

The linear variable differential transformer, or LVDT, is a displacement transducer. It consists of a nickel-iron rod which is free to move through primary and secondary coils. The basic arrangement is illustrated in Fig. 3.6.

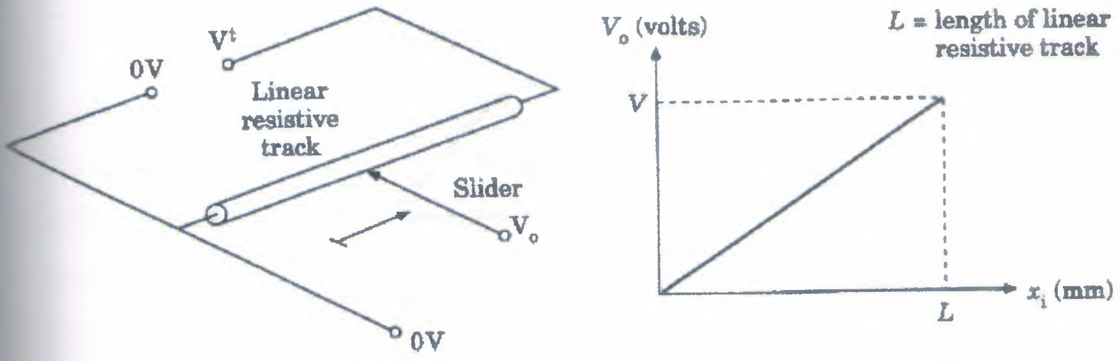


Figure 3.5 Linear potentiometer.

The primary coil is fed with alternating current so that voltages are induced in the two halves of the secondary coil. Moving the rod up and down changes the phase and voltage in the secondary windings. The output voltage versus core displacement characteristic in Fig. 3.6 shows that the phase of the output (secondary winding) relative to the input (primary winding) changes by 1800 as the core is moved through the central position. Consequently, a phase detector is required to obtain an output for each core position.

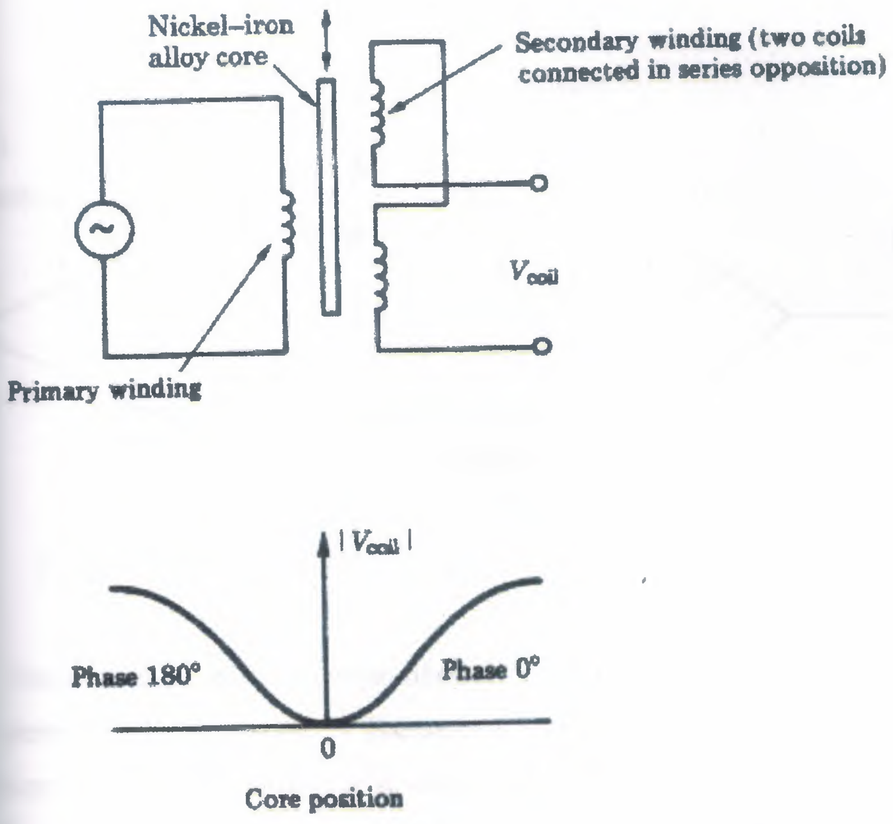


Figure 3.6 Linear variable differential transformer (LVDT).

3.2.3 Tachogenerator

When the shaft of a permanent magnet d.c. motor is driven mechanically an output voltage is produced whose magnitude is proportional to the speed of rotation and polarity depends on the direction of rotation. A permanent magnet d.c. motor used for speed sensing rather than a machine for producing power is called a tachogenerator. Low-pass filtering is required to reduce the ripple voltage superimposed on the output.

3.2.4 Temperature sensor

A thermocouple is a device that converts temperature into a voltage. It consists of two dissimilar wires, which are arranged as shown in Fig. 3.7. Thermoelectric effects produce voltage as the hot junction is heated. Thermocouple types are designated a letter which indicates the types of metals used in the thermocouple injunction (seeFig.3.7).

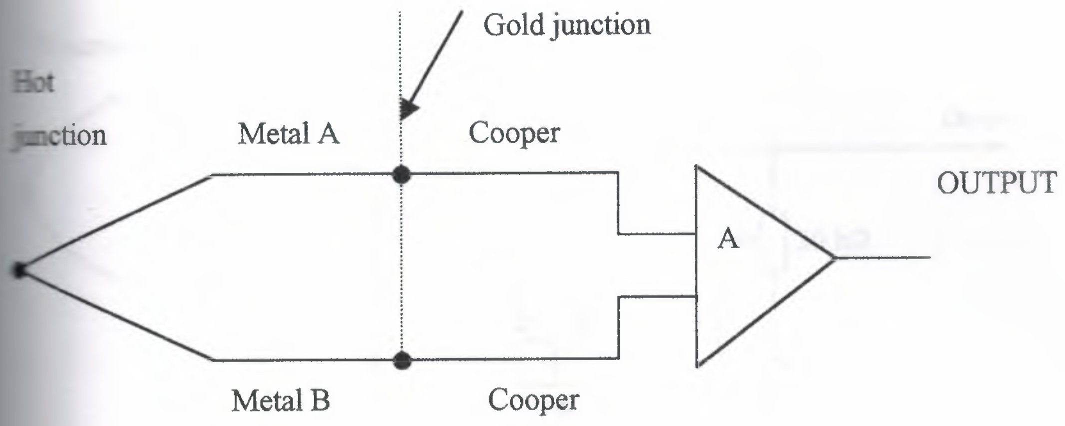


Figure 3.7 Thermocouple.

Thermocouples are non-linear devices, which means that their output voltage is not proportional to temperature. A Thermocouple is supplied with a calibration table of output voltages versus temperature. The output voltage produced by a thermocouple is at the millivolt level and needs to be amplified before it can be fed into an ADC unit. A simple thermocouple amplifier circuit based on the 741 Operational amplifiers is shown in Fig. 3.8.

Semiconductor temperature sensors (e.g. RS590) are commercially available which can generate current as temperature is increased. A typical device would have a temperature coefficient of 1 $_{\mu}$AIK and operate in the temperature range of 218-403 K.

Figure 3.9 shows a temperature switch circuit based on a semiconductor temperature sensor. The circuit converts sensor current into voltage using the resistor R1. This voltage is amplified by the operational amplifier circuit and fed to a comparator. The comparator produces an output signal when the temperature Voltage input is equal to or greater than the threshold level set by Vset. In this the comparator output level turns on the transistor switch based on T1. The transistor switch Ti allows 24 V to be applied to the PLC digital input point. Note that the output to the PLC port is held on until a base current flows in T_1

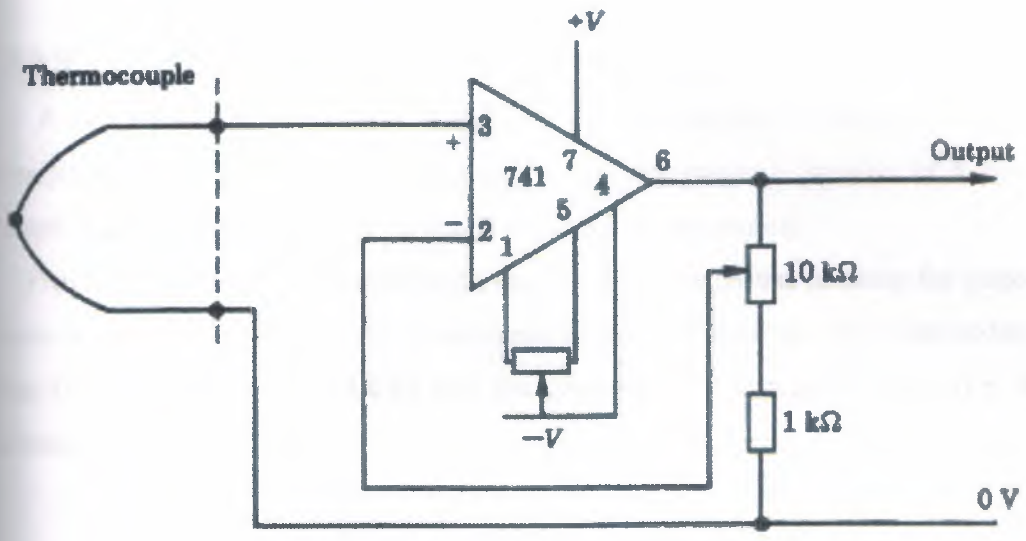


Figure 3.8 Simple thermocouple amplifier circuit

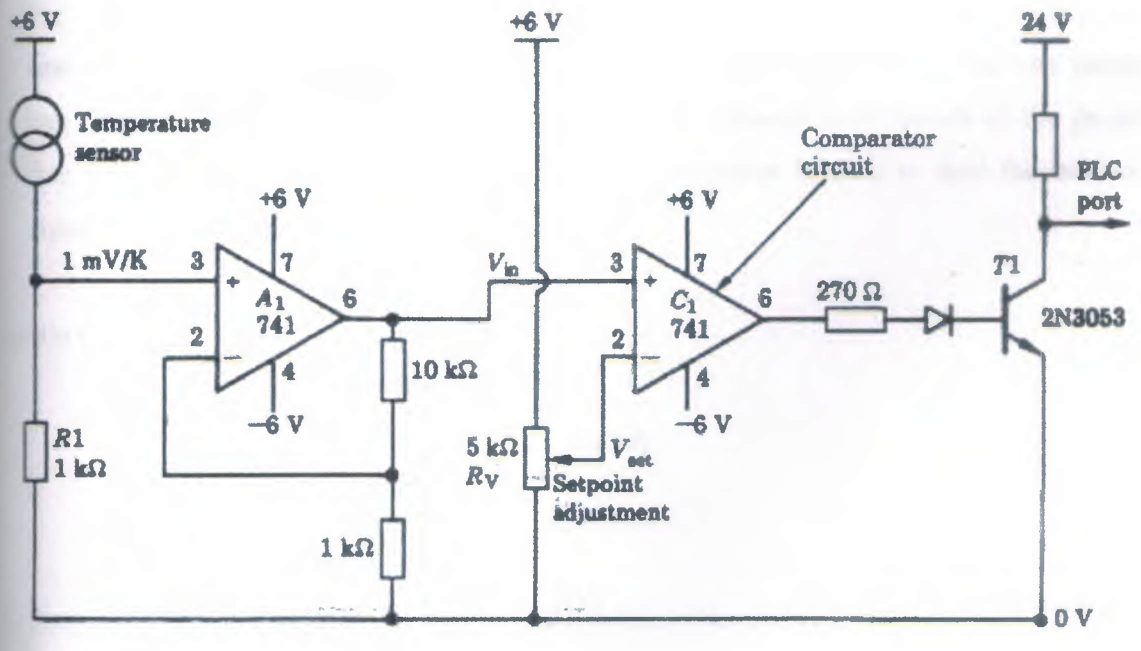


Figure 3.9 Temperature switch circuit.

3.2.5 Strain gauge

A strain gauge is a transducer whose principle of operation is based on the variation of resistance with dimensional displacement. A strain gauge is bonded to a surface of a mechanical element (e.g. a bar) in which strain is to be measured.

Provided that the variation in length under loaded conditions is along the piige-sensitive axis an increase in load causes an increase in gauge resistance. The relationship between the change in resistance ***(ΔR/R)*** and the corresponding change in strain (i.e. the length change *(ΔL/L)* is

$$G = \frac{\Delta R/R}{\Delta L/L}$$

where G is called the gauge factor. ~The gauge factor is about 2 for wire element metal alloy strain gauges and about 100 for semiconductor strain gauges.

A strain gauge is normally connected in a Whetstone bridge arrangement as shown in Fig. 3.10. The bridge is balanced under no load conditions so that any change in resistance due to loading unbalances the bridge and a signal is detected. A dummy gauge can be connected in the bridge to compensate for the change in resistance of the gauge due to temperature variations. An instrumentation amplifier is used to feed the balance signal to an ADC input point.

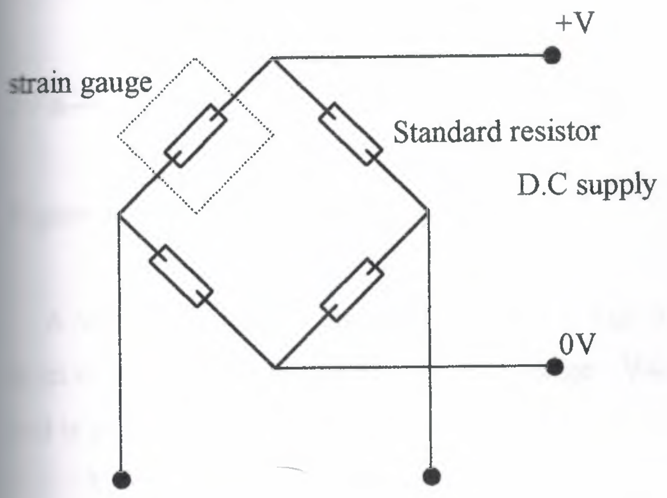


Figure 3.10 Strain gauge bridge.

3.3 Basic interfacing techniques

The voltage signal from a sensor will need to be matched to the specified voltage range and internal resistance of an ADC input port. A typical input port specification is an internal resistance of 200 kΩ for an input range of *0—5* V.

Matching the signal source voltage level to that of the input specification may require reducing or amplifying the voltage of the signal source. If the signal Source voltage ranges between *—2.5* and *+2.5* V (i.e. a bipolar signal) it will need to be converted into the range *0* *5* V (i.e. a unipolar signal) before connect leg to an input specified at *0* *5V*. It is also necessary to ensure that the
Resistance (i.e. impedance) of the signal source is less than or equal to that of the input specification. An impedance changing circuit (e.g. emitter follower) may be required.

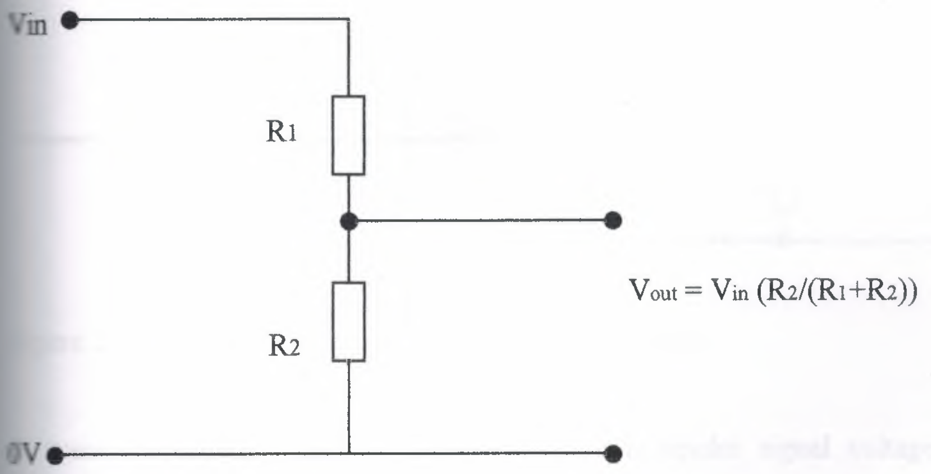


Figure 3.11 Voltage divider.

A voltage (potential) divider, as shown in Fig. 3.11, can be used to reduce tF voltage level of a transducer signal. The output voltage V_{out} is always less than ii input voltage V_{in} and is given by

$$V_{out} = V_{in} (R_2/(R_1+R_2))$$

Small signal amplifiers are used to amplify voltages at the micro volt to millivolt level.

Figure 3.12 shows circuits for inverting and non-inverting amplifiers base around a 741 operational amplifier. The ratio of the output voltage V_{out} to the input voltage V_{in} is called the gain of the amplifier. The inverting amplifier has a gain R_2/R_1. The non-inverting amplifier has a gain of $(R_2+R_1)/R_1$. The resistance R_i in these circuits ensures that both inputs to the operational amplifier see the saw resistance to 0 V and is calculated using

$R_i = (R_1R_2) / (R_2+R_1)$

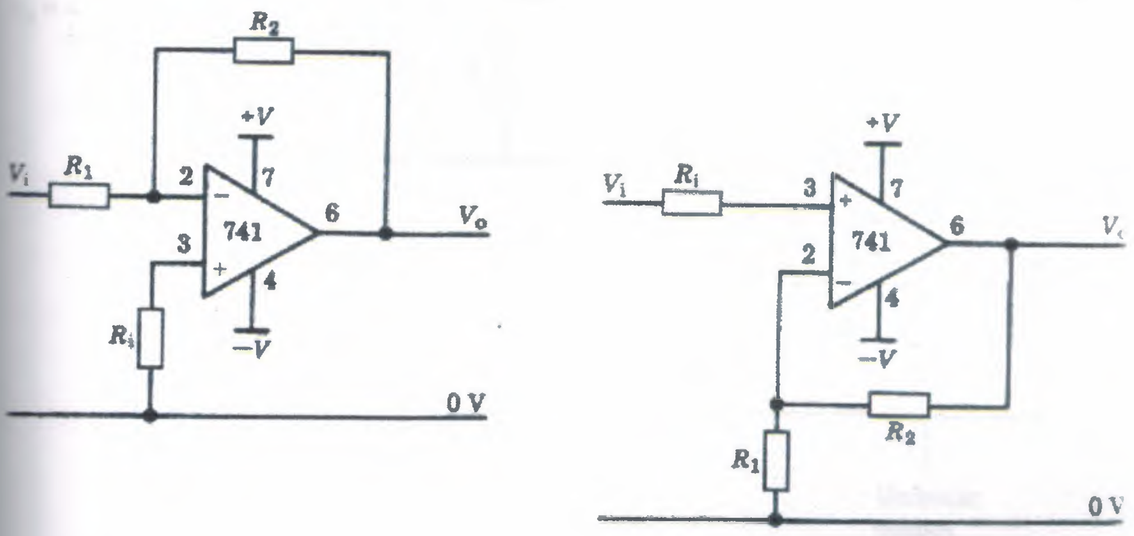
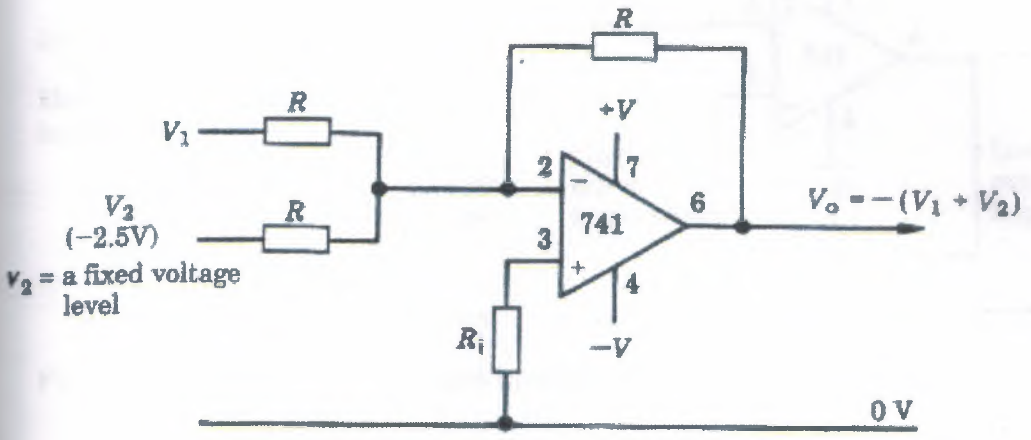


Figure **3.12** (a) Inverting and (b) non-inverting amplifiers.

A summing amplifier can be used to convert a bipolar signal voltage into a umpolar voltage. Figure 3.13 shows a circuit for a summing amplifier which makes use of a 741 operational amplifier. The output voltage Vout is given by

If the summing amplifier is to convert a bipolar signal voltage into a unipolar Voltage, one of the inputs must be held at an appropriate negative voltage. For example, if V_2 is held at *2.5* V, then summing action ensures that the voltage range *2.5* to *+2.5* V is converted into the unipolar voltage 5 – 0 V.



V_1	V_2	$V_o = -(V_1 + V_2)$
−2.5	−2.5	+5V
0	−2.5	+2.5V
+2.5	−2.5	0V

Bipolar input — Unipolar output

Figure 3.13 Summing amplifier.

The emitter follower and voltage follower (a unity gain non-inverting amplifier) Used as impedance matching circuits. Figure 3.14 shows circuits for the emitter and voltage follower. In these circuits the output voltage follows the Voltage (e.g. from the signal source). The main characteristic of both circuits the input impedance is high and the output impedance is low. Consequently, they can be used as a buffer between a sensor and ADC port to ensure that the impedance matching criteria is met. Buffer amplifiers are also used for interfacing low current Sensors.

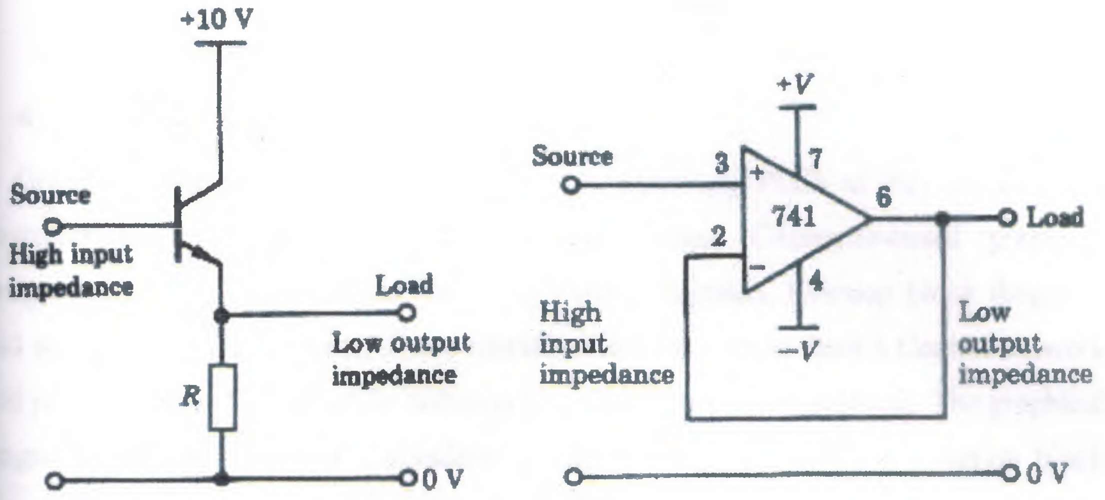


Figure 3.14 a)emitter follower and b)voltage follower

CHAPTER 4
Programming methods

4.1 Graphical languages

Graphical languages are widely used for programming PLCs as they are easy yet powerful tools for developing control applications. Computer-based graphical Programming packages exist for designing ladder diagrams, function block diagrams and sequential function charts. These literally allow the user to draw a Control network and provide a visual image of the solution to a particular control problem. The graphical languages defined in the IEC standard are ladder diagram (LD) and function block diagram (FBD). Sequential function chart (SFC elements are also defined and can be used in conjunction with either of these languages.

A circuit network can be considered as a set of interconnected graphical elements representing a control plan. Graphical languages are used to represents the flow of a conceptual quantity through a network. With a ladder diagram network the conceptual quantity is power flow and the direction of power flow defined to be from left to right. With a function block diagram network the concern of signal flow is applied and the direction of signal flow is defined to be from the output side to the input side of connected function blocks or functions. With sequential function chart the concept of activity flow is applied. Activity flow between SFC elements is from the bottom of a step through the appropriate transition to the top of the following step.

4.1.1 Ladder diagram (LD)

The ladder diagram method has been used throughout this book, as it is the predominant programming method for developing small- and medium-scale PD applications. The IEC standard is based on terminology and symbols as used b the majority of current PLC systems. The IEC standard ladder diagram graphic~ symbols are shown in Table4.4. The standard includes symbols for SET RESET coils, which allow a variable to be latched on, and then cleared at a late stage. Retentive coils (e.g. coils whose Boolean states need to be held during PL power interruption) are also defined.

Function blocks having Boolean inputs and outputs can be connected within ladder diagram. Functions used in a ladder diagram may have an execution enable (EN) input and an execution enable output (ENO). When EN is true the function is evaluated. When EN is false it remains inactive. A ENO is taken high on if successful completion of a function. It is possible to connect an ENO output to a EN of another function to provide execution control.

4.1.2 Function block diagram (FBD)

The function block diagram (FBD) is a graphical language whereby programs are expressed as a set of interconnected function blocks. An analogy cans h drawn with a system circuit diagram where connections represent signal (FBD) paths. Each function block is represented as a rectangular block with input Drawn so that they enter on the left side and outputs drawn so that they exit o the right side. Signal flow is from an output of one function block to the input of another function block. An example of a function block diagram is shown in Fig. 4.1.

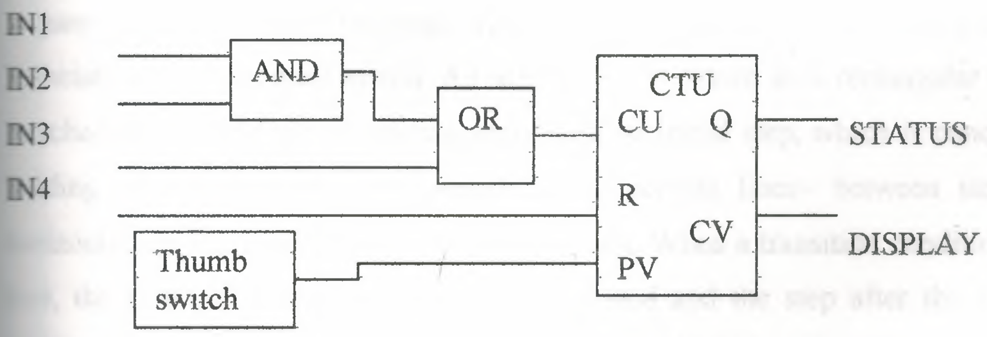


Figure 4.1 Example of a function block diagram (FBD).

The IEC standard defines a small number of standard function blocks which are the SR bitable, RS bitable, rising edge detector, falling edge detector, edge detecting inputs, up-counter, down-counter, up-down counter, pulse timer, on delay timer, off-delay timer and real-time clock. These are shown in Table *4.5*. Other special purpose blocks such as PID and ramp blocks can be designed using the ST language.

The FBD language is suited to Boolean logic and continuous (e.g. closed-loop) control applications. Although it is possible to construct a 'jump' within an FBD using a label identifier, the IEC does not recommend its use. Constructs such as IF/THEN/ELSE used with the structured text language are difficult to represent graphically within an FBD. Consequently, direct translation between ST and FBD is not always possible.

4.2 Sequential function chart (SFC)

The IEC 1131-3 standard describes the use of a graphical sequencing language referred to as the sequential function chart. This is based on Grafcet, a graphical language for developing sequential control programs, and defined as a French national standard. Telemecanique PLCs use the Grafcet language but most major PLC systems provide an option for sequential programming. In fact, the IEC 1131-3 standard uses an existing IEC standard to describe a graphical language for control sequences.

A sequence can be thought of as a series of steps that occur in a defined order. The sequential function chart language represents each step as a rectangular box, which is associated with a control action. An action can be drawn as a rectangular box that is attached to a step. Every sequence starts with an initial step, which is concerned with holding the system ready for operation. Connecting lines~ between steps have a horizontal bar representing a transition condition. When a transition condition becomes true, the step before the transition is deactivated and the step after the transition is activated. The main features of an SFC are illustrated in Fig.4.*2*.

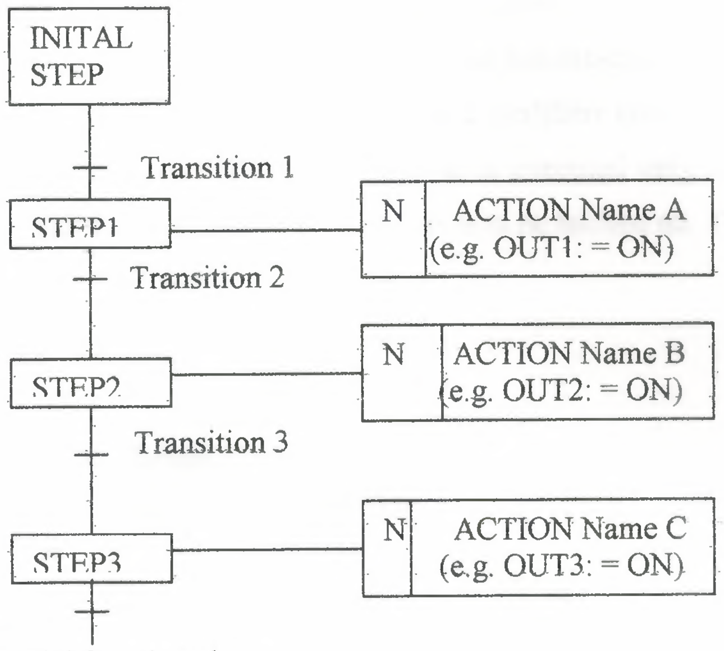

INITAL STEP
Transition 1
STEP1
N
ACTION Name A (e.g. OUT1: = ON)
Transition 2
STEP2
N
ACTION Name B (e.g. OUT2: = ON)
Transition 3
STEP3
N
ACTION Name C (e.g. OUT3: = ON)

Figure *4.2* Sequential function charts.

Steps are always separated by a transition and every transition must have a Boolean condition, which can be true. Every action and transition should be given a unique name within a program.

The software to implement transition conditions and actions can be written using any of the IEC languages. An example of a transition condition (Trans 1) written in ST is

```
TRANSITION Transl:
    : = Startbutton AND Ready;
END_TRANSITION
```

The same transition is shown as a Ladder Diagram in Fig.4.*3*. A transition can involve a timer (e.g. wait for an elapsed time to occur), a counter or any other type of function block.

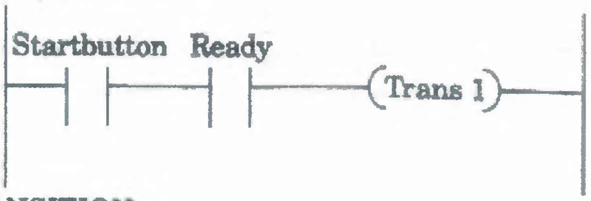
An example of a simple action written in ST is

```
ACTION Name A
    OUT1: =ON;
END_ACTION
```

The action called 'Name A' sets OUT1 to an ON state. The action will occur as a result of an associated step (e.g. STEP 1 in Fig. *4.2)* being activated.

An action can have a qualifier, which determines the way in which the action will be executed. An action qualifier is drawn as a rectangular box attached to the left hand side of an action as shown in Figure *4.2.* Commonly used qualifiers are N (none), S (set) and R (reset). An N qualifier specifies that the action is executed only while the step is active. An S (set) qualifier specifies that the action is to be latched on. The step acts as a trigger for the latch and the action is said to be

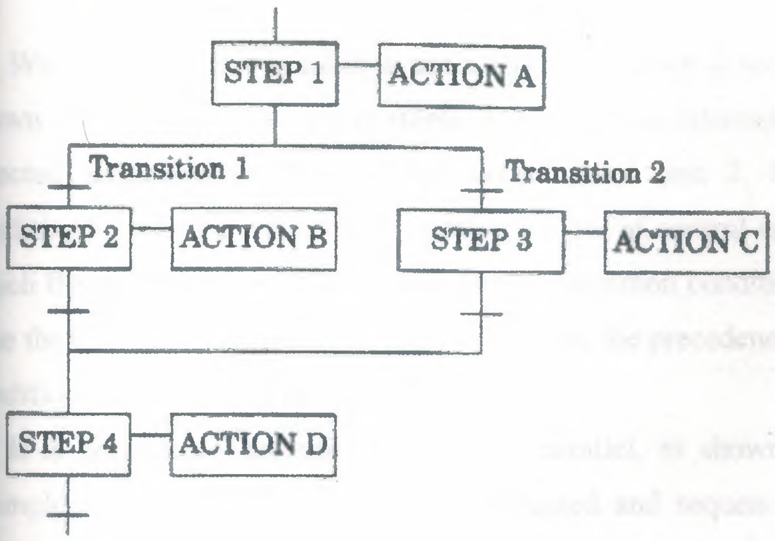
TRANSITION Trans 1



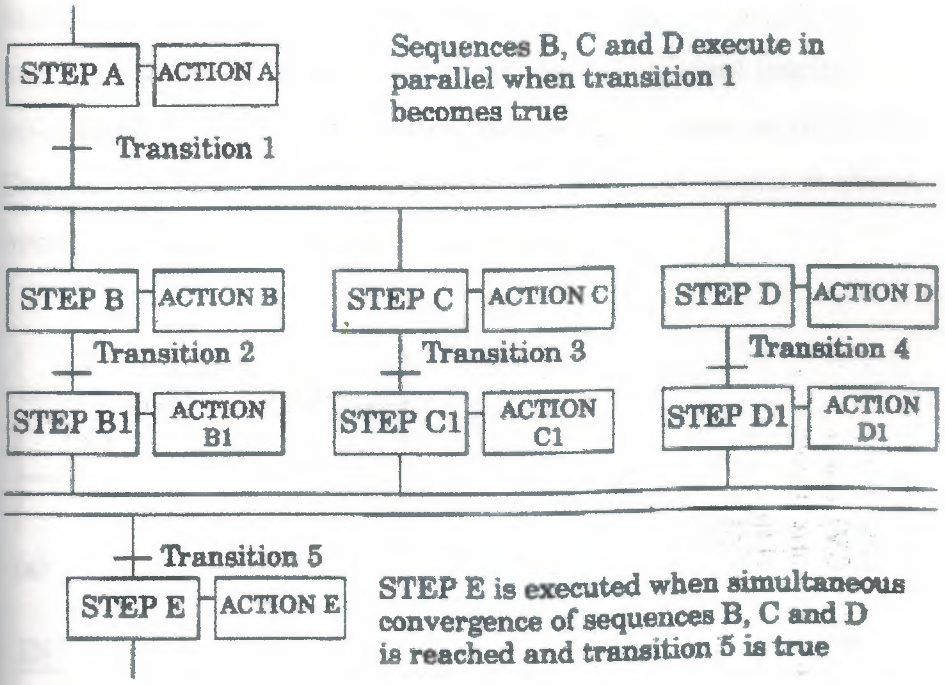
END_TRANSITION

Figure 4.3 Ladder diagram transition expression.

Only one path is selected. Normally transition conditions are tested from left to right but precedence can be user defined 'stored' since it will continue as the control sequence progresses. A set or stored action is cleared using an R (reset) qualifier.



a)


STEP A
ACTION A
Sequences B, C and D execute in parallel when transition 1 becomes true
Transition 1
STEP B
ACTION B
STEP C
ACTION C
STEP D
ACTION D
Transition 2
Transition 3
Transition 4
STEP B1
ACTION B1
STEP C1
ACTION C1
STEP D1
ACTION D1
Transition 5
STEP E
ACTION E
STEP E is executed when simultaneous convergence of sequences B, C and D is reached and transition 5 is true

b)

Figure 4.4 Sequential function chart structures: (a) use of divergent branches to enter alternate sequences and (b) parallel sequences.

Within an SFC it is possible to use a divergent branch to select a path option. This is shown in Fig. 4.4(a). In this example, there are two alternate paths which could be selected according to the transition conditions 1 and 2. Normally the transition conditions are tested from left to right. The flow of control is through the branch for which the transition condition is true. If both transition conditions are true at the same time then the left-hand branch is taken. However, the precedence in which the transition conditions are tested can be user defined.

It is possible to activate sequences in parallel, as shown in Fig. 4.4(b). In this example, two parallel branches are constructed and sequence flow is simultaneous through the two branches. A pair of parallel horizontal lines are used to denote the start and end of the simultaneous sequencing of branches. The branch sequences continue independently. Convergence of the branch sequences must be obtained before they can be combined into one sequence.

4.3 Translating between languages

Simple ladder rungs involving combinational logic can usually be translated into FBD or ST, as illustrated in Fig. 4.5. The IEC 1131-3 standard specifies the operation of the up-counter, down-counter and up-down counter in terms of an ST language algorithm. Consider the function block representation of the up-counter as shown in Fig. 4.6. The operation of the up-counter function block can be .

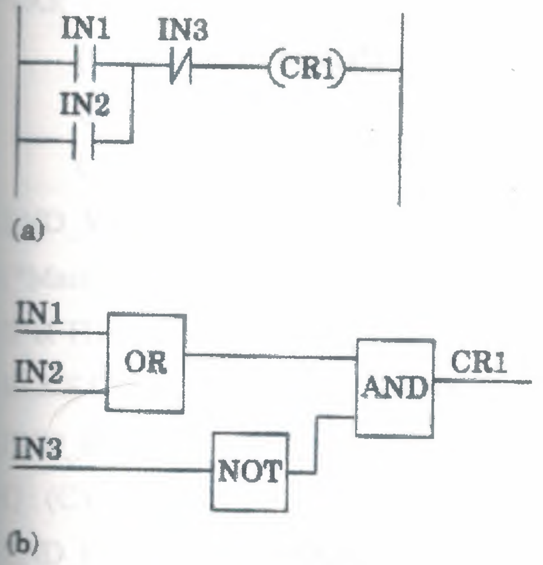


Figure 4.5 Translating a ladder diagram into other IEC languages: (a) ladder diagram, (b) function block diagram and (c) structured text.

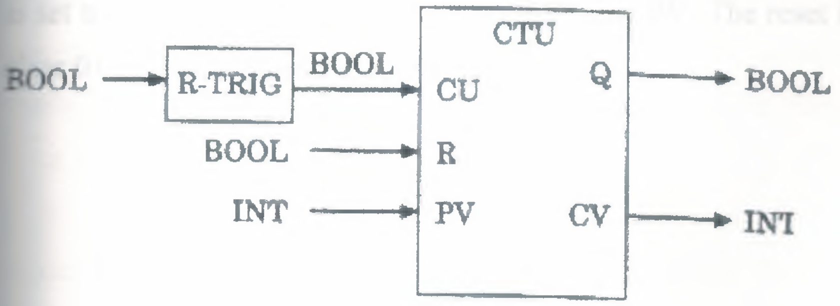


Figure 4.6 Up-counter function block.

```
  FUNCTION_BLOCK CTU          (*CTU is an abbreviation for Count Up*)
VAR_INPUT
     (*data type of inputs*)
CU : BOOL R_TRIG;              (*Count input on rising edge transition*)
     R:BOOL ;                        (*Reset*)
     PV : INT;                        (*Preset value*)
END_VAR
VAR_OUTPUT
     (*data type of outputs*)
     Q : BOOL;                      (*Current output*)
     CV: INT;                       (*Counter count value*)
END_VAR
(*Main body of function block*)
IF R THEN CV :              0;
ELSIF CU AND (CV <PV) THEN CV:=CV+1;
END_IF;
Q : (CV>=PV);
END_FUNCTION_BLOCK
```

The up counter counts the number of rising edges observed at the count-up input CU. The pre-set value (**PV**) is the maximum count value to be reached. When a new rising edge input is detected the count value (**CV**) is incremented by one. The output **Q** is set true when CV is equal to the pre-set value PV. The reset input R can be used to clear 0 (e.g. set false) and CV (e.g. set to zero).

CHAPTER 5
Ladder programming examples

This chapter is basically a programmer's guide to the ladder diagram programming method. As discussed in Chapters 1 and 5, the ladder diagram programming method has been evolved from conventional electrical circuit relay-based control methods and many of the concepts are the same.

In essence, a ladder diagram has a left-hand vertical power rail that supplies notional power through contacts arranged along horizontal rungs. Each contact represents the state of a Boolean variable. When all contacts in a horizontal rung are in the true state, power is deemed to flow along the rail and operate a coil on the right of the rung. In ladder programming terminology contacts can be referred to as inputs and coils as outputs.

Ladder diagram programming is equivalent to drawing a network containing switch elements. Modem hand-held programming consoles having liquid crystal displays allow ladder diagrams to be entered directly. Function blocks such as timers and counters can be connected into ladder diagram rungs provided that their inputs and outputs are Boolean variables. Specific details on using function blocks are given in this chapter.

5.1 Combinational logic

A combinational logic circuit is one in which the function of the output is a direct and unique consequence of the combination of the input conditions. The basic logic functions AND, OR and NOT are used in combination logic circuit designs.

A three-input OR gate has an output *f*, given by

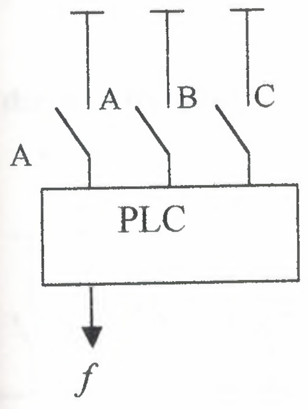
$f=A+B+C$

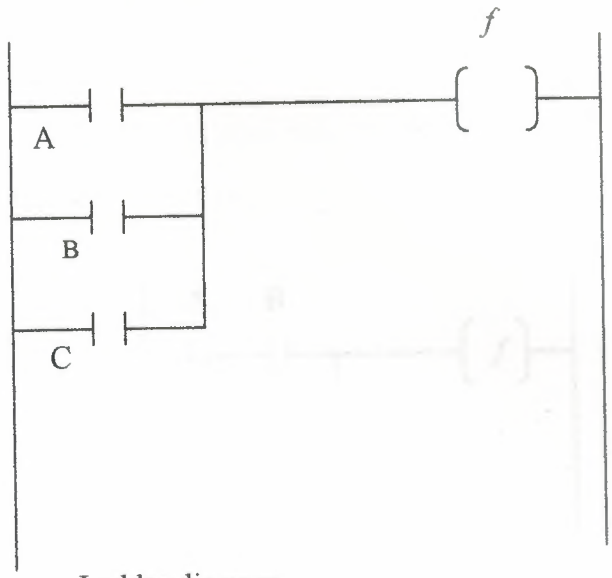
where *A*, *B*, *C* are the input Boolean variables. The ladder diagram is drawn as shown in Fig. 5.1. In this case, the output coil *f* is true (1) if *A* is true (1) or *B* is true (1) or *C* is true (1).

A three-input AND gate has an output *f*, given by

$f=ABC$

This has a ladder diagram as shown in Fig. 5.2.

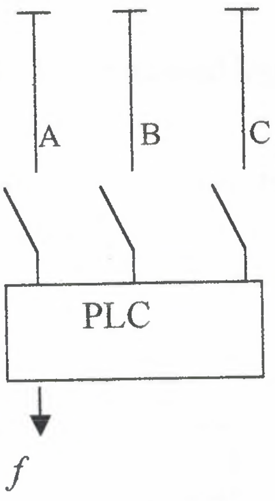


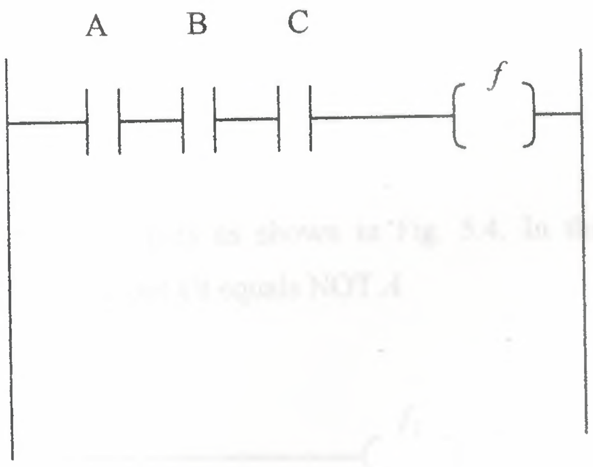


Schematic layout diagram

Ladder diagram

Figure 5.1 Three-input OR function.





Schematic layout diagram

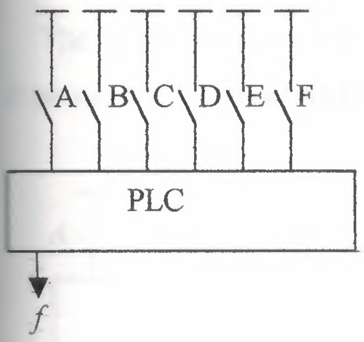
Ladder diagram

Figure 5.2 Three-input AND function.

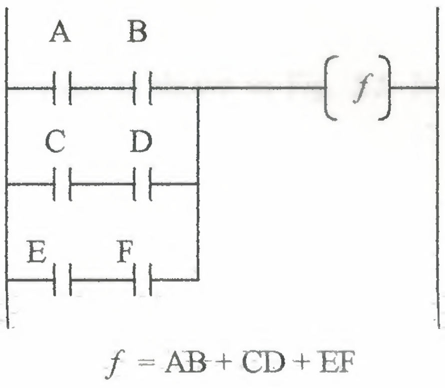
In general, any combinational logic output requirement can be drawn as a ladder diagram. For example, the output *f* represented by the function

$f = AB+CD+EF$

is drawn as the ladder diagram shown in Fig. 5.3.



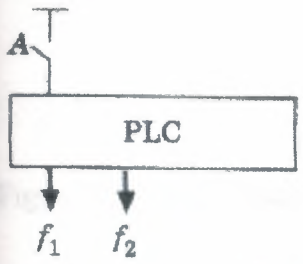
Schematic layout diagram



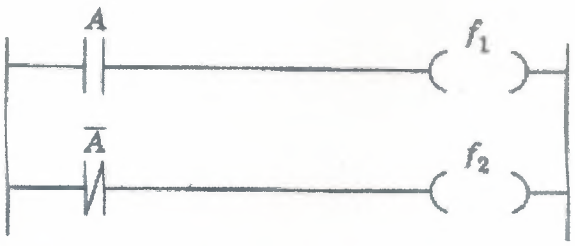
Ladder diagram

Figure 5.3 Combinational logic example

The NOT function can be used to toggle outputs as shown in Fig. 5.4. In this example, the output f_1 equals *A* (i.e. $f_1=A$) and the out **Ut** equals NOT *A*



Schematic layout diagram



Ladder diagram

Figure **5.4** Toggling outputs using the NOT function.

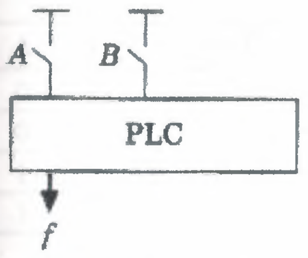
The output f of a two-input NAND function is given by

$$f=AB$$

Making use of De Morgan's theorem, which states that, can draw the ladder diagram

$$f=AB=A+B$$

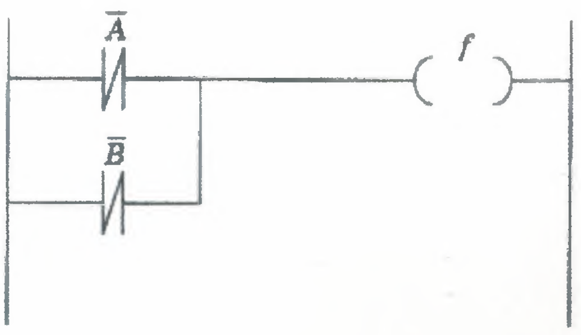
The ladder diagram for the two-input NAND function is shown in Fig. 5.5. It is drawn as NOTA in parallel (i.e. OR) with NOT B.



Schematic layout diagram

A	B	$f=\overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

Truth table



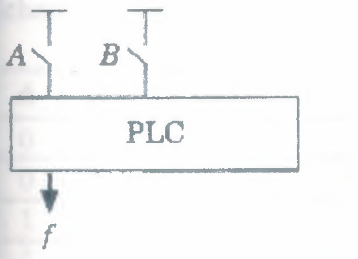
Ladder diagram

Figure 5.5 NAND ladder diagram.

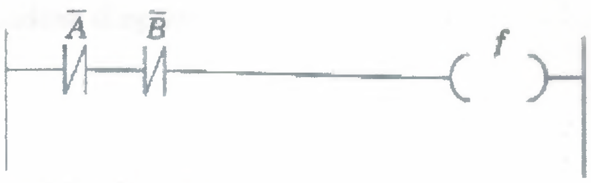
Similarly, by using De Morgan's theorem, the output f of the two-input NOR function can be written as

$f=A+B=AB$

The ladder diagram is drawn as NOT *A* in series with NOT ***B.*** This is shown in Fig.5.6.



Schematic layout diagram



Ladder diagram

A	B	$f=\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

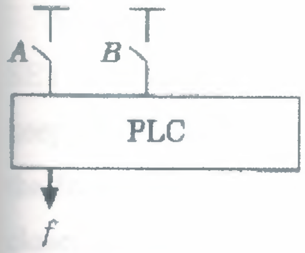
Truth table

Figure **5.6** NOR ladder diagram.

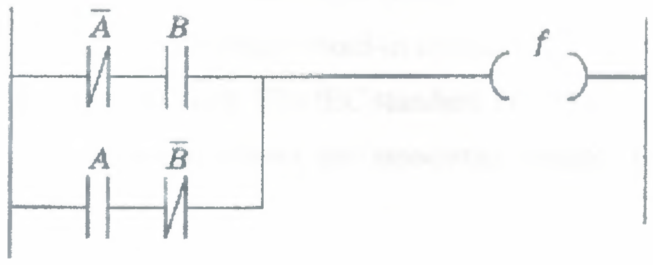
The exclusive OR function (written as XOR) is a special form of the OR function whose Boolean expression is

$f=AB+AB$

Its ladder diagram can be drawn as shown in Fig. 5.7.



Schematic layout diagram



Ladder diagram

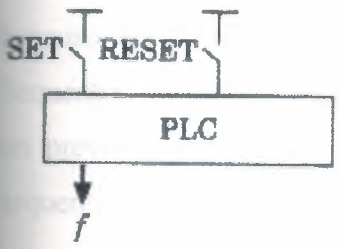
A	B	$f = \bar{A}B + A\bar{B}$
0	0	0
0	1	1
1	0	1
1	1	0

Truth table

Figure 5.7 XOR ladder diagram.

5.2 Latching

A ladder latch circuit allows an output coil to be held (set) and maintained on until a different condition occurs which is used to reset the coil off. An example of a latch circuit is shown in Fig. 5.8.



Schematic layout diagram

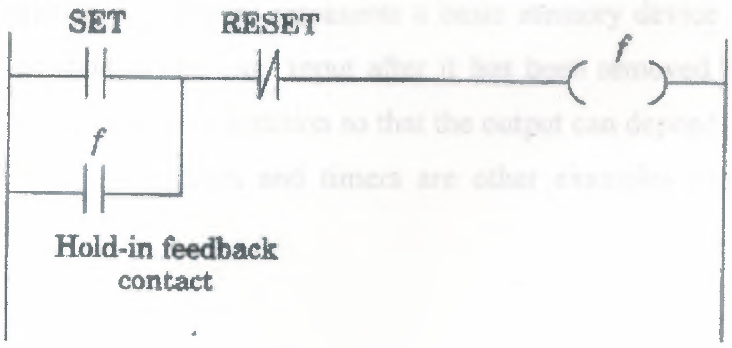


Figure 5.8 Latch circuit.

Assuming all contacts are in their initial states, the output *f* is set true when the input SET becomes true. Once set the output f is no longer determined by the input SET as it keeps itself maintained on through its feedback contact *f*. In relay circuit terminology the feedback contact is referred to as the maintaining or hold-in contact. The normally closed contact RESET can be used to clear the latch. The IEC standard 1131-3 defines a set coil symbol and a reset coil symbol which allows the associated variable to be latched on and off (see Fig. 5.9).

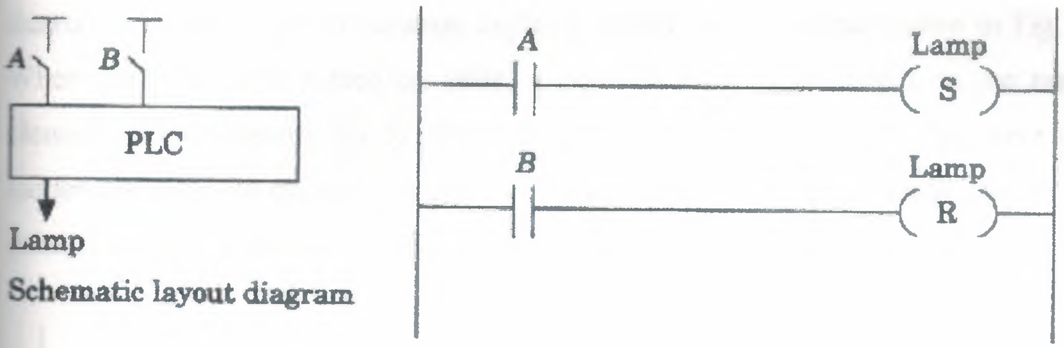


Figure 5.9 Latching using set and reset coils.

The latch is a sequential circuit and can be considered as the ‘flip-flop’ of ladder logic. The term flip-flop reflects a device that can flip into one state or flop back again to the other on command. The latch (i.e. flip-flop) represents a basic memory device because the output remembers the state of the ‘set’ input after it has been removed. Sequential devices always incorporate a memory function so that the output can depend on previous as well as present inputs. Counters and timers are other examples of sequential devices.

5.3 Generating a pulse signal

Figure 5.10 shows the characteristics of a positive-going logic pulse. The edge transition from 0 to us defined as the rising or leading edge. The edge transition from 1 to 0 is defined as the falling or trailing edge. The width of the pulse is referred to as the pulse width.

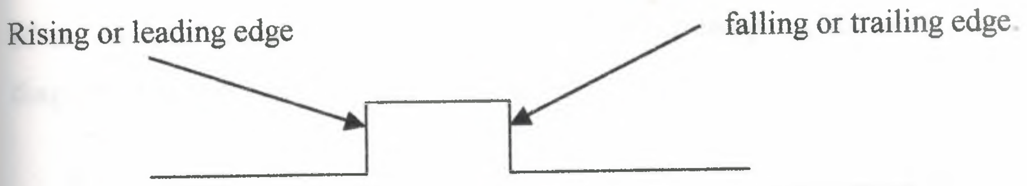


Figure **5.10** Pulse characteristics.

Many PLC systems allow a memory bit to be pulsed on command for a fixed duration of one program execution cycle. A typical circuit is that shown in Fig. 5.11. When the input *A* is turned on (true) a positive pulse is generated on the memory element MX5 (memory bit *5)*. This memory element can be used elsewhere in the ladder diagram, for example to reset a counter. The IEC function block symbol for a general purpose pulse timer is shown in Table *4.5* (Chapter *4)*.

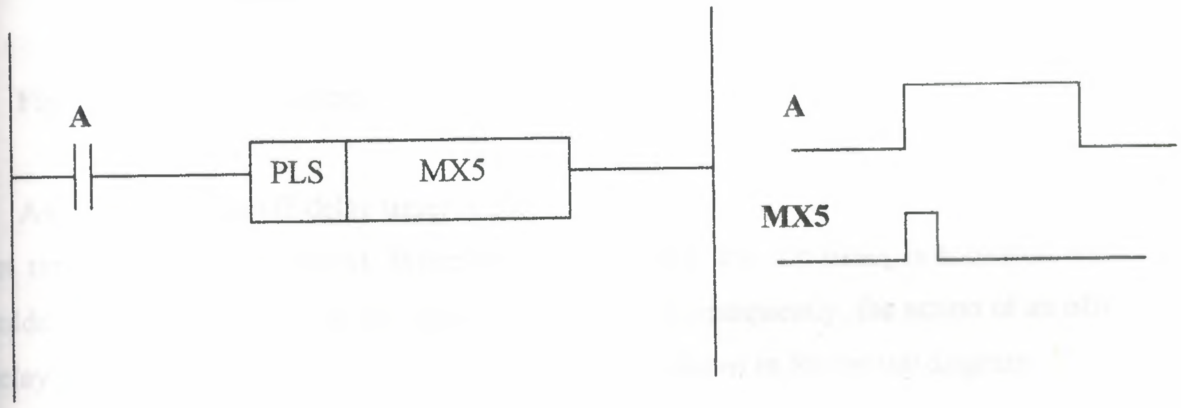


Figure **5.11** Pulsing a memory bit.

5.4 Timers

An example of an on-delay timer is shown in Fig. 5.12. The timer set value (i.e. pre-set time) is *5* seconds *(T#5s)*. When the input *A* becomes true the timer is activated and when the specified set value has elapsed the output is set true. Consequently, the action of an on-delay timer is to delay setting the output lamp on, as shown in the timing diagram. The timer is reset when the input *A* goes low.

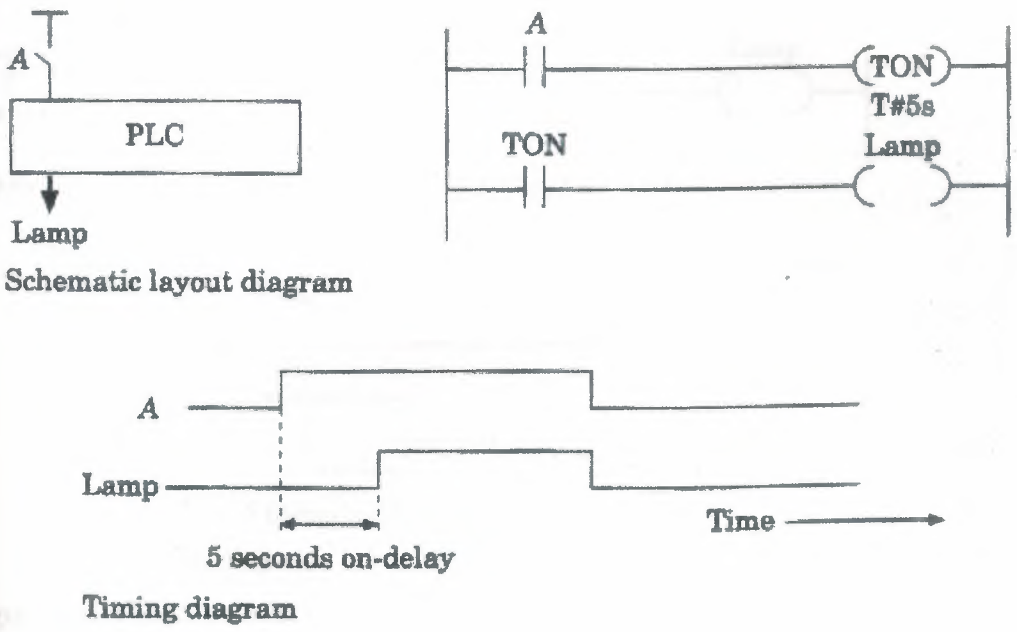
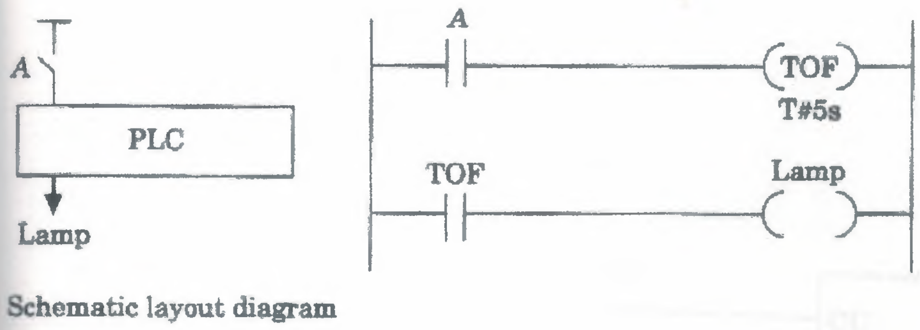


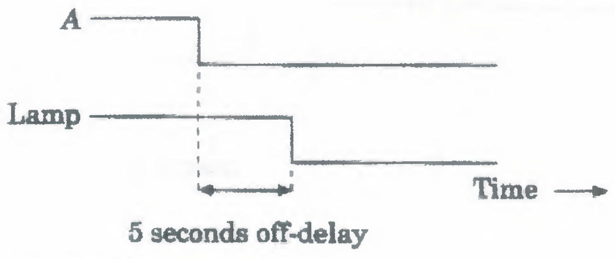
Figure 5.12 on delay timer

An example of an off-delay timer is shown in Fig. 5.13. The timer set value (i.e. pre-set time) is *5* seconds (T#5s). When the input A goes low the timer is activated and holds the output lamp on for the specified set value. Consequently, the action of an off-delay timer is to delay setting the output lamp off, as shown in the timing diagram.

If the period for which the input is turned on is shorter than the set value (preset time) of a timer the output does not change state. The IEC symbols for function block on-delay and off-delay timers are shown in examples. Generally, proprietary PLC systems use a coil representation for timers in ladder diagrams rather than a function .



Schematic layout diagram



Timing diagram

Figure **5.13** Off-delay timer.

5.5 Counters

An example of an up-counter block incorporated in a ladder diagram is shown in Fig. 5.14. A positive pulse applied using the reset contact clears the counter so that Q is false and the pre-set value (PV) is 3. If three positive-going pulses are applied using the count contact the output is set true on the rising edge of the third pulse. Any further pulses applied to the input using the count contact are ignored (i.e. counting stops) until the counter is reset.

A ladder diagram incorporating an up-down counter is shown in Fig. 5.15. This type of counter allows increment (count-up) and decrement (count-down) input signals to be connected. A reset input and pre-set value (PV) must also be provided. A rising edge applied to the count-up input increases the count value (CV) by one. A rising edge applied to the count-down input decreases the count value (CV) by one. When CV=O the output QD is set true and further changes on

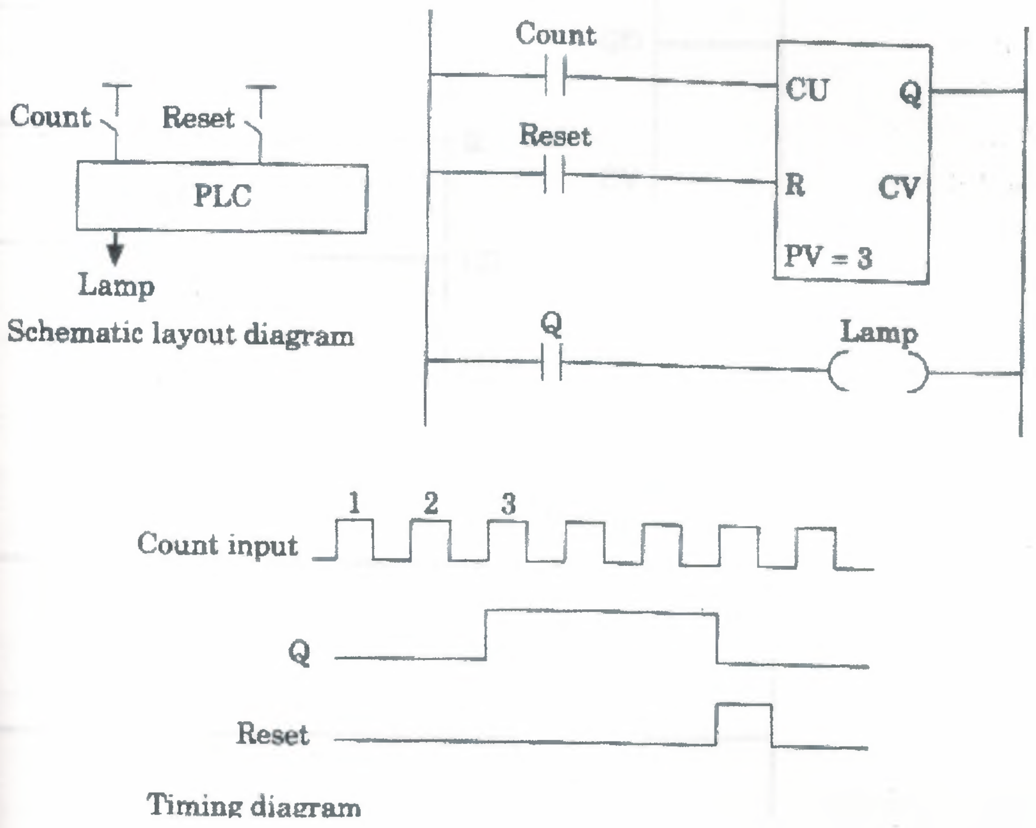


Figure 5.14 Up-counter.

the count-down input are ignored until a reset signal is applied. When **CV=PV** the output QU is set true and further changes on the count-up input are ignored until a reset signal is applied.

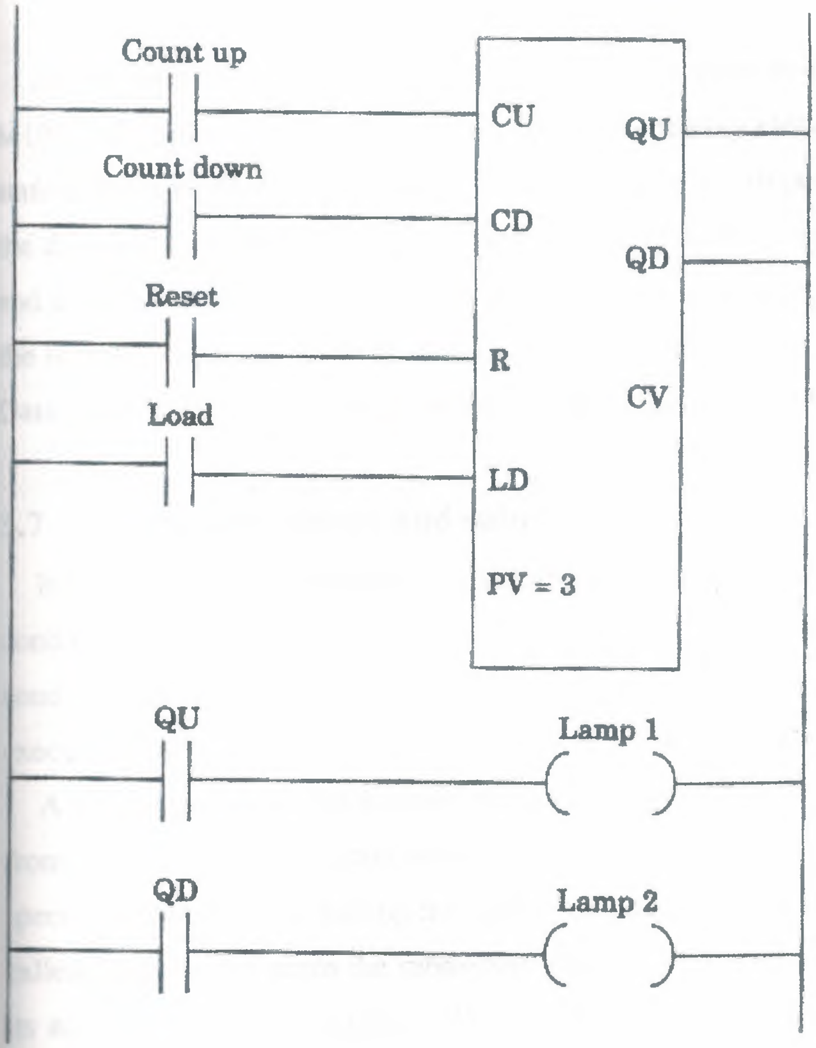


Figure 5.15 Up – down counter

5.6 Shift register

A shift register is a set of memory elements connected in series in which binary data is stored. Usually the number of memory elements is 4, 8, 16 or 32. There are three inputs, a data input, a shift input and a reset input. The data input is used to transfer a binary value into the first memory location. A pulse applied to the shift input moves all the binary values stored in the register along by one location. The reset input is used to clear all the memory elements so that they store zero.

An example of a 4-bit shift register consisting of memory elements M100, M1O1, M102 and M103 is shown in Fig. 5.16. The first memory element M100 has the same state as the data input contact on Ii. The rising edge of a shift pulse input signal shifts all the data in the register one bit to the left. Assuming the shift register is initially cleared and a logic 1 is entered into M100 a shift pulse moves the logic 1 into M1O1. Note that the memory elements M100 to M103 are used to drive the output coils Q400 to Q403. Data in the last memory location of the register is lost when a shift is applied.

5.7 Conditional jumps and subroutines

It can be required that part of a ladder program has to be inactive when an input condition occurs. A 'jump' function block can be used to skip ladder rungs on the condition that a contact is made. When a contact is turned on the cyclic scan does not execute ladder rungs between *jump* and *end-jump* labels marked on a ladder diagram.

A subroutine represents a complete ladder program which can be called repeatedly from a main ladder program when required. A subroutine is implemented using a special function coil for calling the routine in the ladder circuit. When the subroutine is called the processor scans the subroutine and returns control to the main program when its actions have been completed. Details of implementation of jumps and subroutine methods are manufacturer specific.

5.8 Arithmetic functions

Most PLCs are able to do simple binary arithmetic operations with data stored in registers using function blocks to add, subtract, multiply and divide binary and BCD numbers. Instructions to set and clear a carry flag are used as binary arithmetic operations make use of a carry in their results

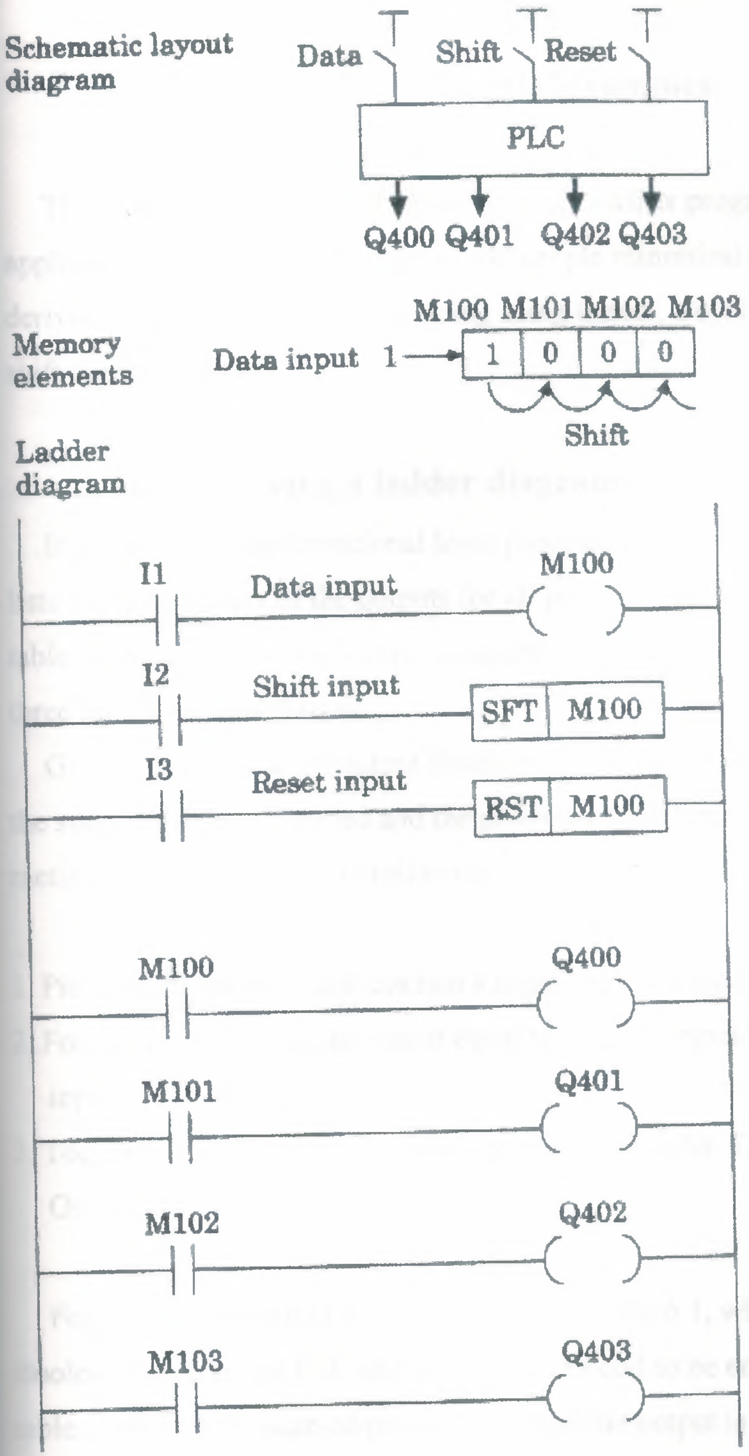


Figure 5.16 A- 4 bit shift register

CHAPTER 6
Tutorial examples

The aim of this chapter is to provide some further programming techniques useful for applications. The areas covered include simple minimization of a Boolean equation derived from a truth table, sequencing using timers, a cyclic timer circuit and a cyclic shift register circuit.

6.1 Deriving a ladder diagram from a truth table

In general, any combinational logic process can be described by a truth table, which lists the logic values of the outputs for all possible combinations of the input. A truth table with one output and n inputs consists of 2 - 8 rows. For example, a truth table with three inputs has 2 or 8 rows.

Given a truth table, an output function can be derived by one of two methods called the sum-of-products method and the product-of-sums method. The sum-of-products method is applied using the following rules:

1. Pick out all the rows that contain a logic 1 in the output column.
2 .For each row that has an output equal to logic 1, logically AND the corresponding inputs
3. Together using a negation whenever an input is zero. Take all the row products and OR them together.

For example, consider the truth shown in Table 6.1, where *A, B* and *C* are the Boolean inputs to the PLC and f is the output coil to be energized according to the truth table. Applying the sum-of-products method the output is given by

$$f=ABC+ABC+ABC$$

This can be drawn as a ladder diagram as shown in Fig. 6.1.

This expression can be reduced to a simpler form by applying the laws of Boolean algebra to produce a more economical logic circuit. This process of reducing the circuit to a simpler form is called minimization. The steps involved in minimizing this function are described below.

The output can be rewritten as

$$f=AC+AC+ABC+ABC$$

Table 6.1 Example truth table

A	*B*	*C*	*f*
o	0	0	0
o	0	1	1
o	1	0	0
o	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

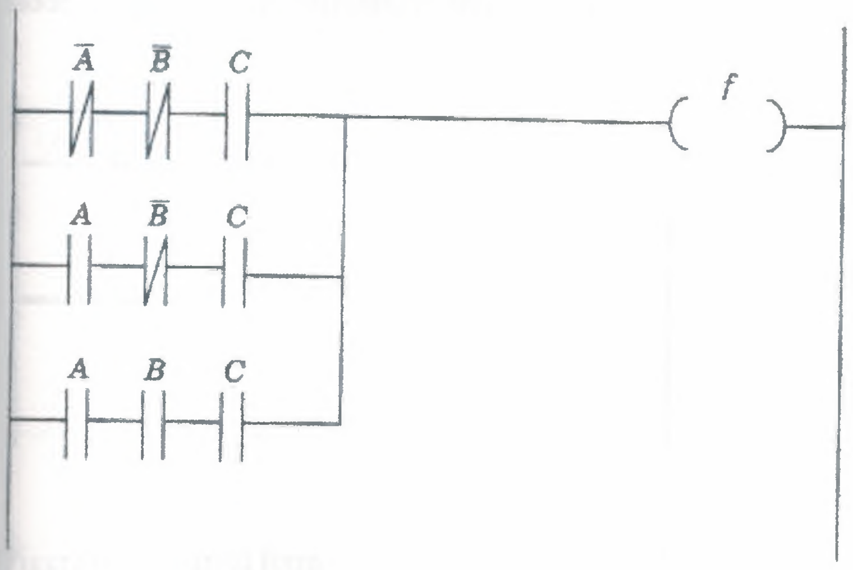


Figure 6.1 Sum of products ladder.

because the idempotent law (1.5) states that

$$AC + ABC = ABC$$

The reason for expanding the function in this way is because it allows each of the terms to be combined in such a way that they can be reduced. The output can be written as

$$f=BC(A+A) +AC(B+B)$$

Applying the complementarity law (1.13) yields

$$f=BC+AC$$

which can be written as

$$f=C(A +B)=(A +B)C$$

This expression for *f* is called the minimal because it cannot be further reduced. The ladder diagram for the minimal is drawn in Fig. 6.2.

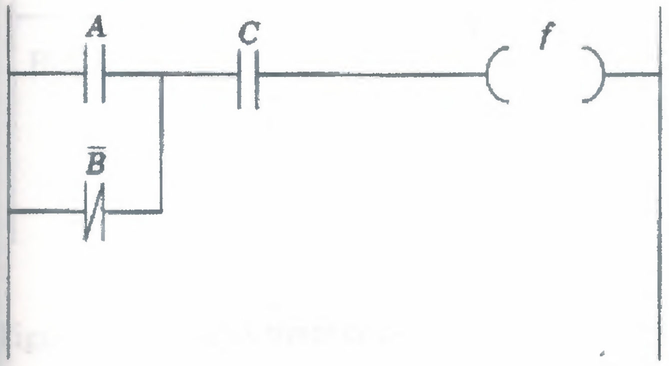


Figure 6.2 Minimal form.

An alternative approach to solving this problem would be to use the product-of-sums method. This is applied using the following rules:

1. Pick out the rows that contain a logic 0 in the output column.
2. For each row that has an output equal to logic 0 logically OR the corresponding inputs together using a negation whenever an input is a logic 1.
3. Take all the row products and AND them together.

6.2 An off-delay timer circuit

Many PLCs provide the programmer with on-delay timers only. An off-delay timer circuit can be created using an on-delay timer as shown in Fig. 6.3. The output Q1 goes high when the input Ii is set high and maintains itself high through the hold-in latch contact. When input Ii is set low the timer resets the latch after the specified pre-set value is reached. Consequently, the output is held for the specified pre-set time after the input is taken low.

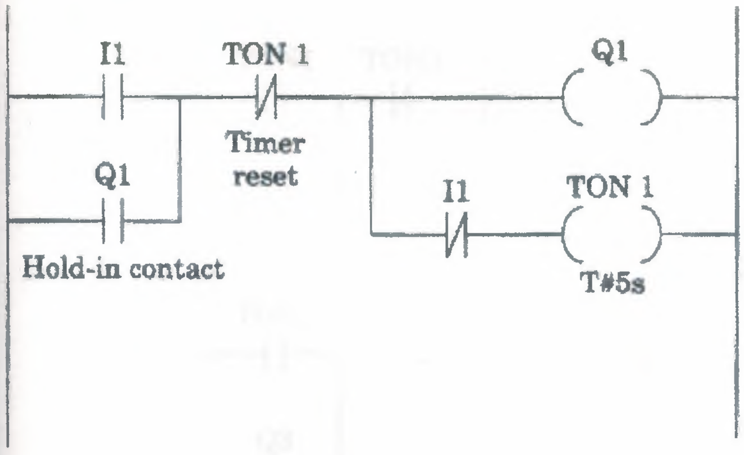
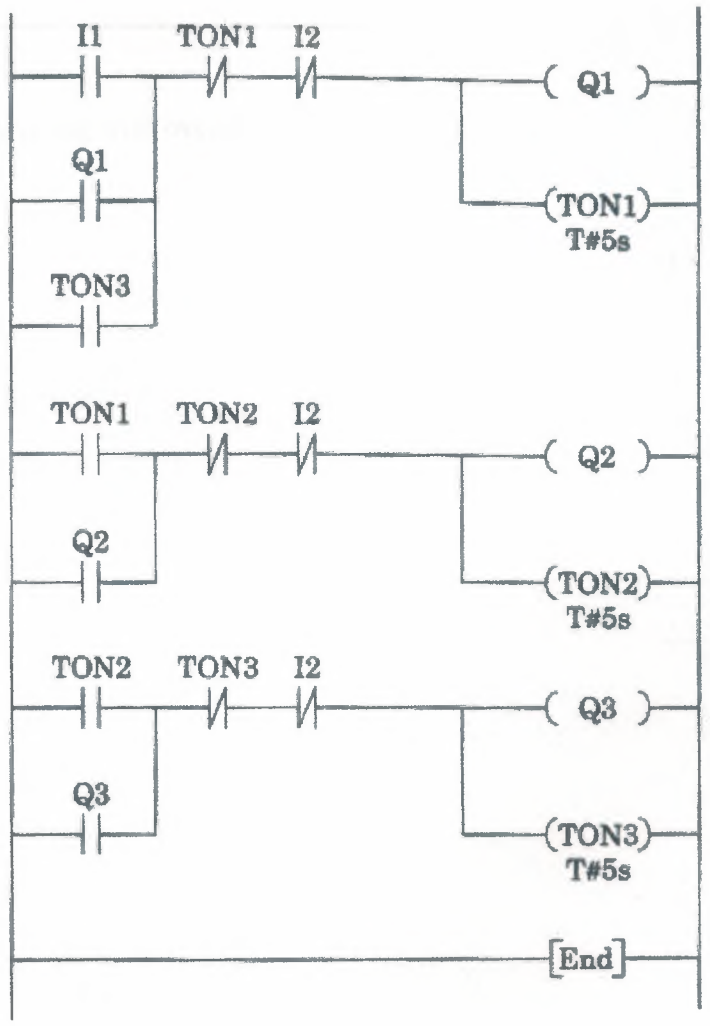


Figure 6.3 Off-delay timer circuit.

6.3 Controlling outputs using timers

Timers can be used to sequence a set of output states. An example circuit is shown in Fig. 6.4. When an input pulse is applied to Ii the outputs Q1, Q2 and Q3 are turned on and off one after the other, as illustrated in the timing diagram. Each output is held on for the specified pre-set time. The PLC scan ensures that the step-to-step sequencing of outputs turning on and off is repeated

Overlap of the output on/off transitions can be achieved by using additional timers to set the next output high before the previous output is taken low. An example ladder circuit and timing diagram are shown in Fig. *6.5.*



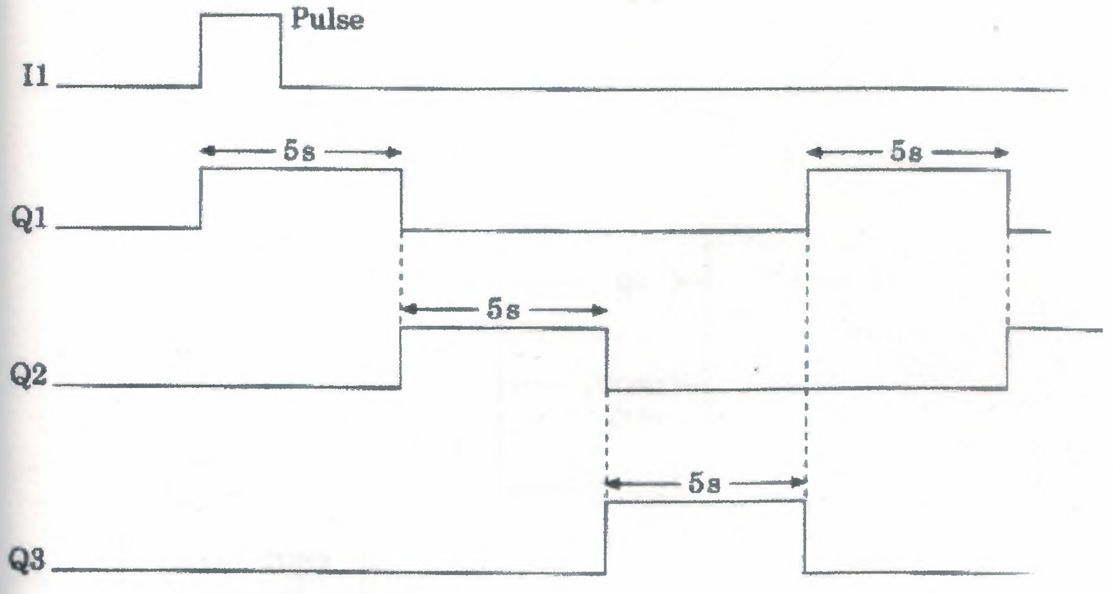
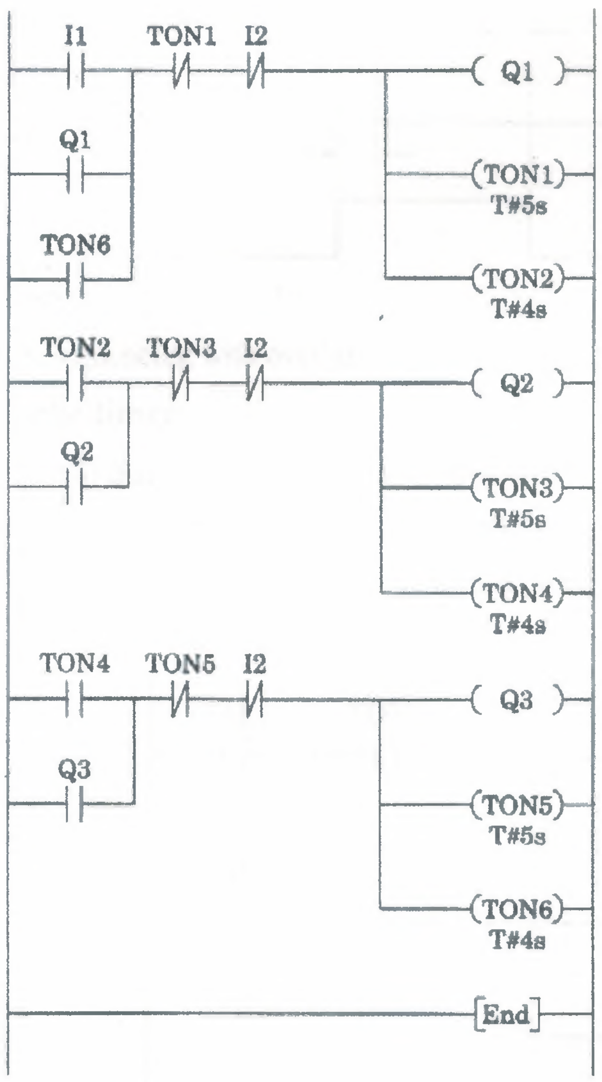


Figure 6.4 Sequencing with overlap



Ladder diagram

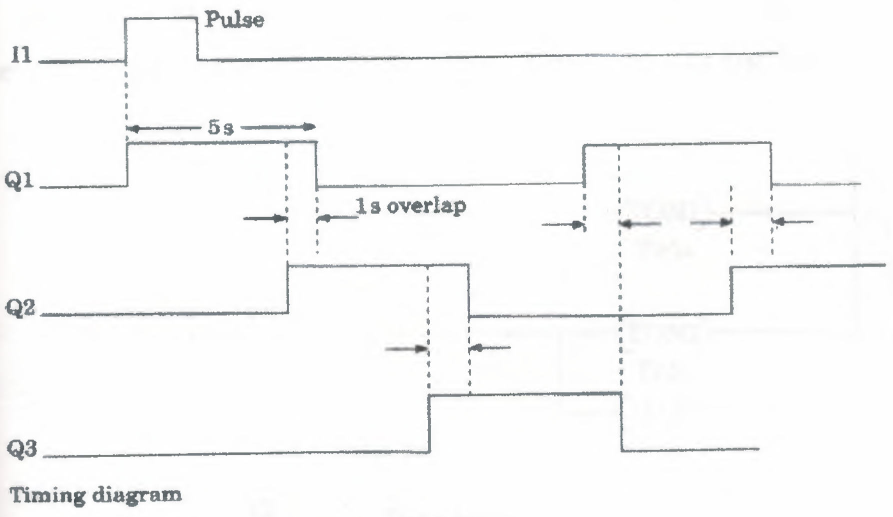


Figure 6.5 Sequencing with overlap

6.4 Cyclic timer

A ladder circuit that repeatedly turns an output on and off at regular intervals is called a cyclic timer. A cyclic timer circuit is shown in Fig. 6.6. When Ii is set high, the on-delay timers TONi and TON2 set and reset each other, with the result that Q1 is clocked on and off at *5* second intervals.

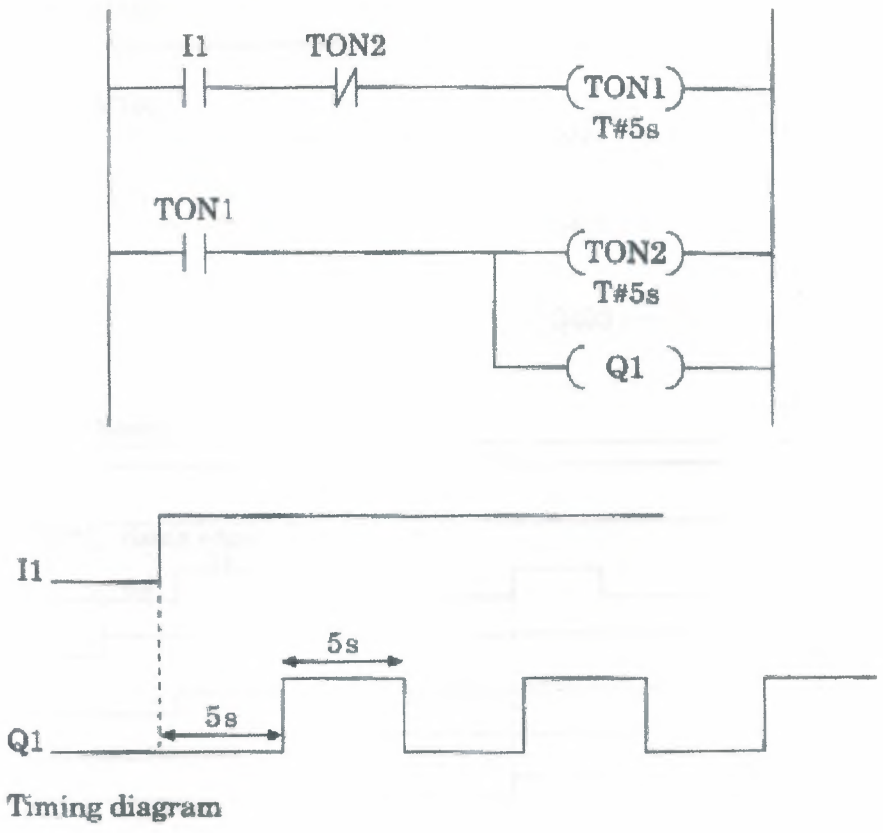


Figure 6.6 Cyclic timer circuit.

The cyclic timer circuit can be combined with the shift register to time-sequence a set of outputs turning on. An example circuit is shown in Fig. 6.7.

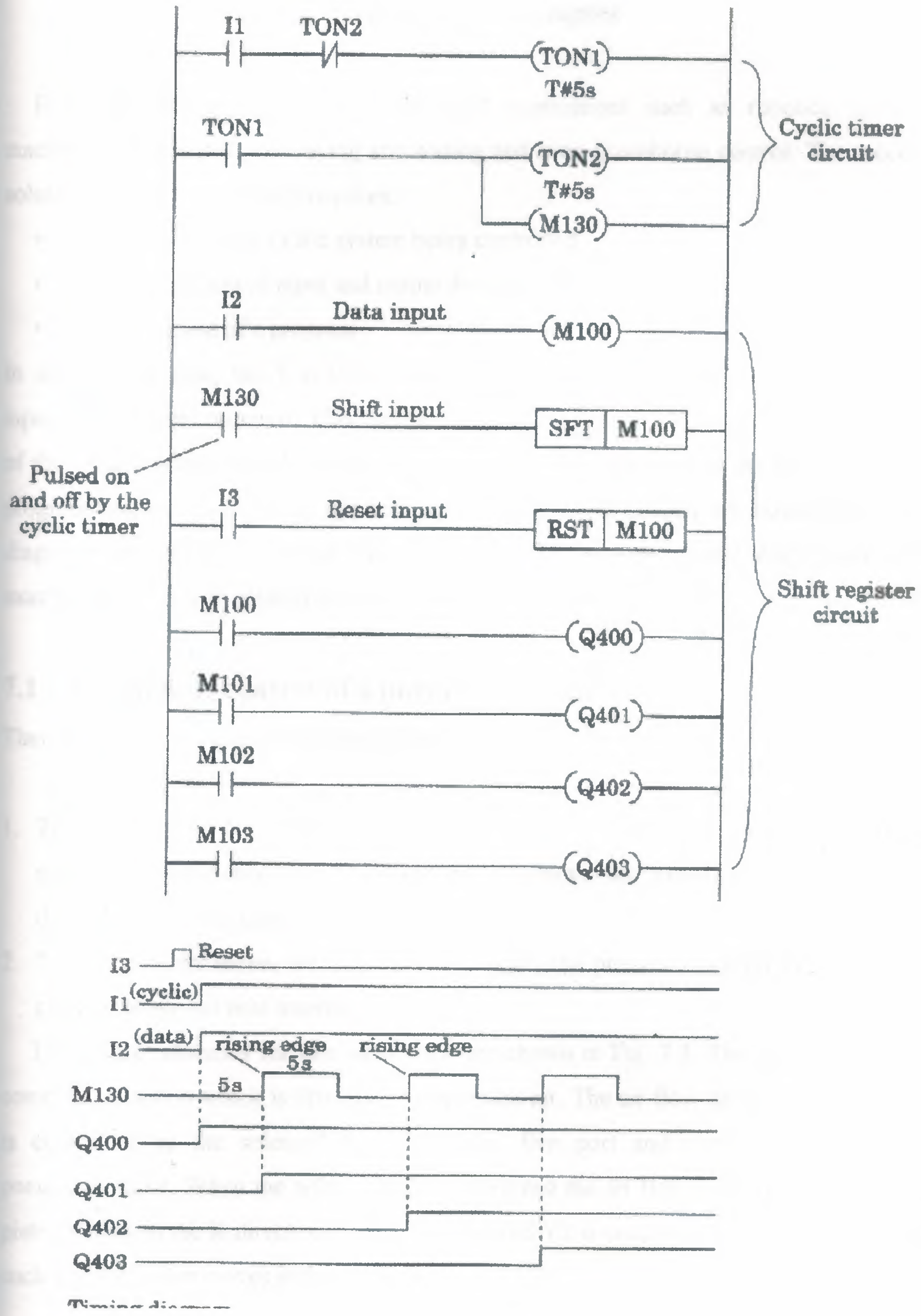


Figure 6.7 Cycle shift register

CHAPTER 7
Application examples

PLCs are used in a range of industrial applications such as robotics, packaging machinery, pneumatics, conveying and sorting and even greenhouse control. The successful solution of a control problem requires:

- An understanding of the system being controlled
- The assignment of input and output devices to PLC I/O points
- Development of a program

In any control task, the first thing that must be considered is the number and type of input/output points required. This can be assessed by drawing a schematic layout diagram of the system which identifies each input/output device connected to the PLC system. The programming solutions for the application examples of this chapter are expressed as ladder diagrams incorporating function block elements. The exception is the temperature control example in which a structured text code fragment is developed.

7.1 Example 1: control of a pneumatic piston

There are two objectives of this application:

1. To use a PLC to move the piston A shown in Fig. 7.1 out of its cylinder (A- direction) when the push-button Si is pressed and to move the piston into its cylinder (A- direction) when the push-button S2 is pressed.
2. To continuously move, with the use of a timer, the pneumatic piston in and out of its cylinder at pre-set time intervals.

The control elements for this application are shown in Fig. 7.1. The pneumatic cylinder comprises a piston which is driven by compressed air. The air flow direction to the cylinder is controlled by the solenoid-driven *5/2* (i.e. five port and two air flow directions) pneumatic valve. When the solenoid Y1 is energized the air flow direction is such that the piston moves in the K direction. When the solenoid Y2 is energized the air flow direction is such that the piston moves in the A direction.

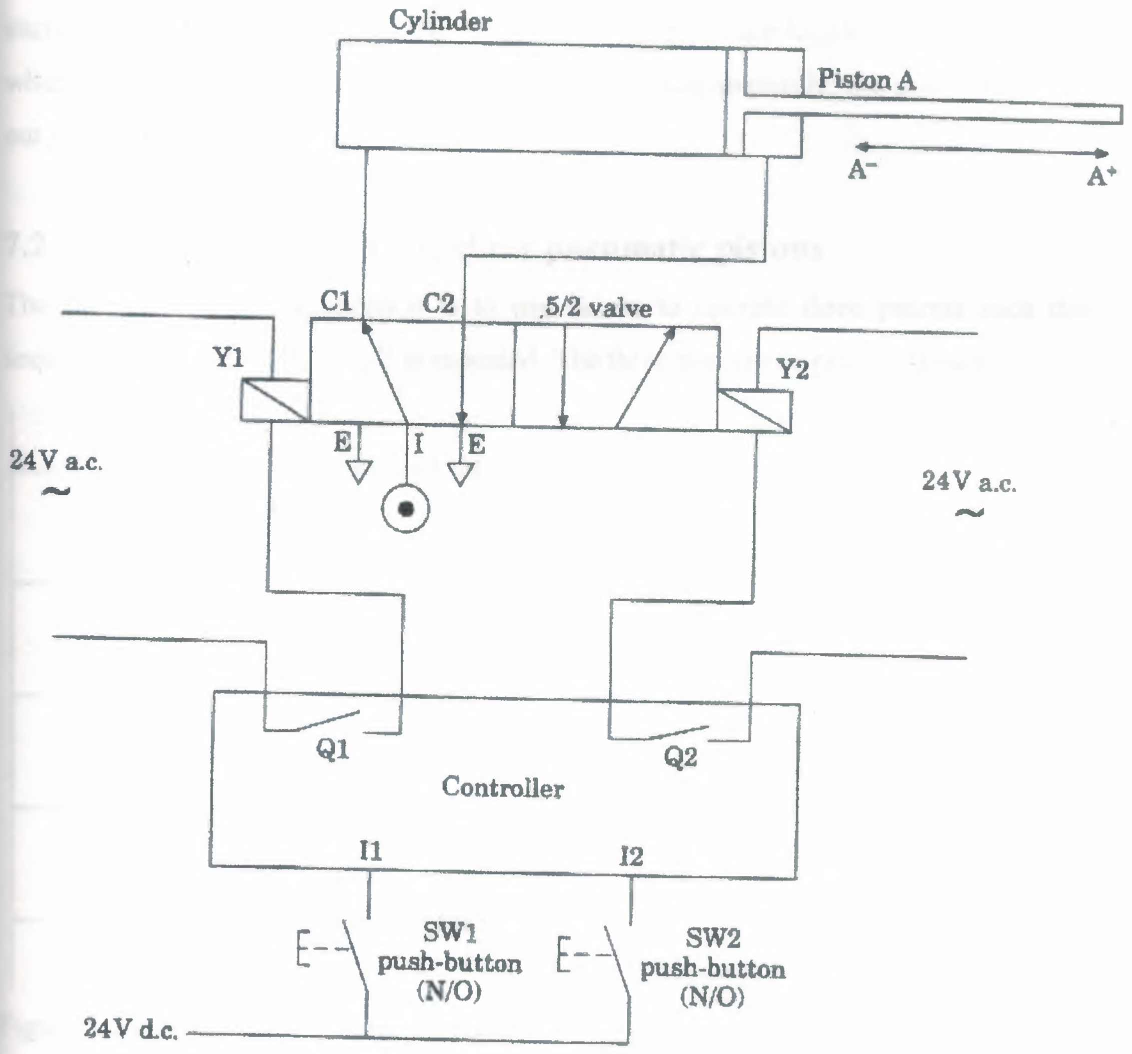


Figure 7.1 Control of a pneumatic piston.

A ladder program solution for the first objective is shown in Fig. 7.2. Two latch circuits are used. When Ii is momentarily taken high the output Q1 is energized and held on via its latch contact. When 12 is momentarily taken high the output Q1 is de-energized (i.e. 12 resets the latch) and Q2 is energized and held on via its latch contact. The Q2 latch circuit is reset when Ii is taken high. Consequently, only one of the solenoids Y1 and Y2 are on at any one time.

A ladder program solution for the second objective is shown in Fig. 7.3. It is based on a cyclic timer circuit discussed in Chapter 6. The two timers TONi and TON2 set and reset each other at *5* s intervals, with the result that Q1 and Q2 are toggled (i.e. switched so that when Q1 is on Q2 is off and vice versa) every *5* s. Consequently, the piston moves in and out of its cylinder every *5* s.

7.2 Example 2: sequencing three pneumatic pistons

The objective of this application is to use timers to operate three pistons such that the sequence A~, A-, B~, W, C~, C is repeated. The three pneumatic pistons shown

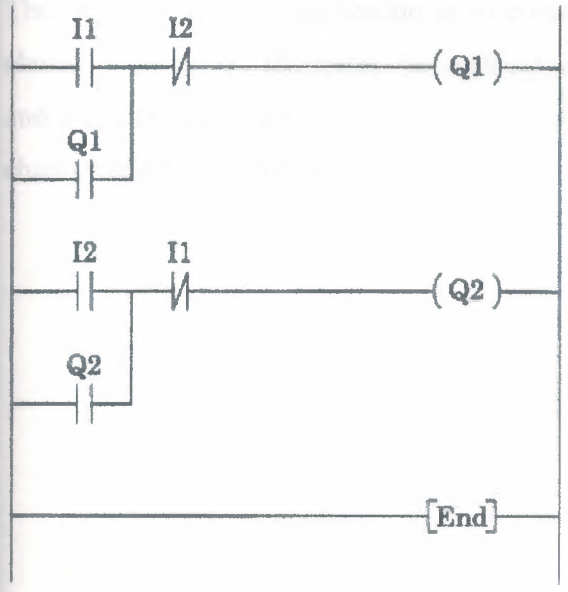


Figure 7.2 Circuit for controlling the pneumatic piston.

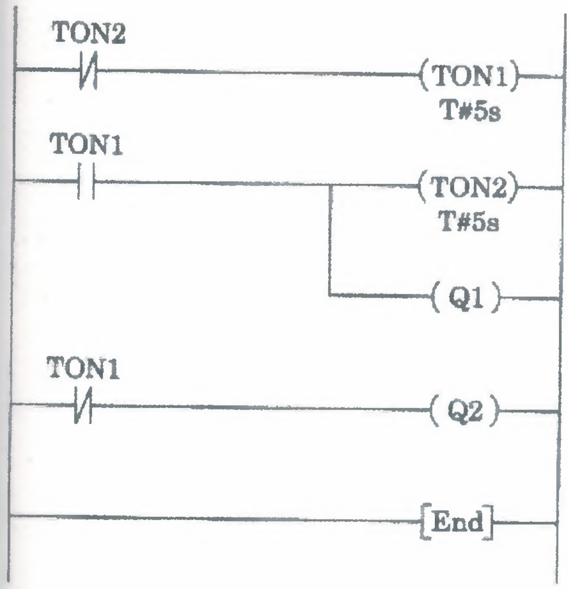


Figure 7.3 cyclic timer circuit for controlling the piston .

in Fig. 7.4 are controlled using the three 512 directional control valves. Each solenoid has to be held energized for 5 s to allow time for the piston to move. A ladder solution to this problem is shown in Fig. 7.5.

7.3 Example 3: counting and packaging

The objective of this application is to continuously divert components down one of two chutes. Figure 7.6 illustrates the application, the assignment of PLC input/output points and a ladder program solution. The requirement is for ten components to be directed down chute A and twenty components down chute B for packaging purposes.

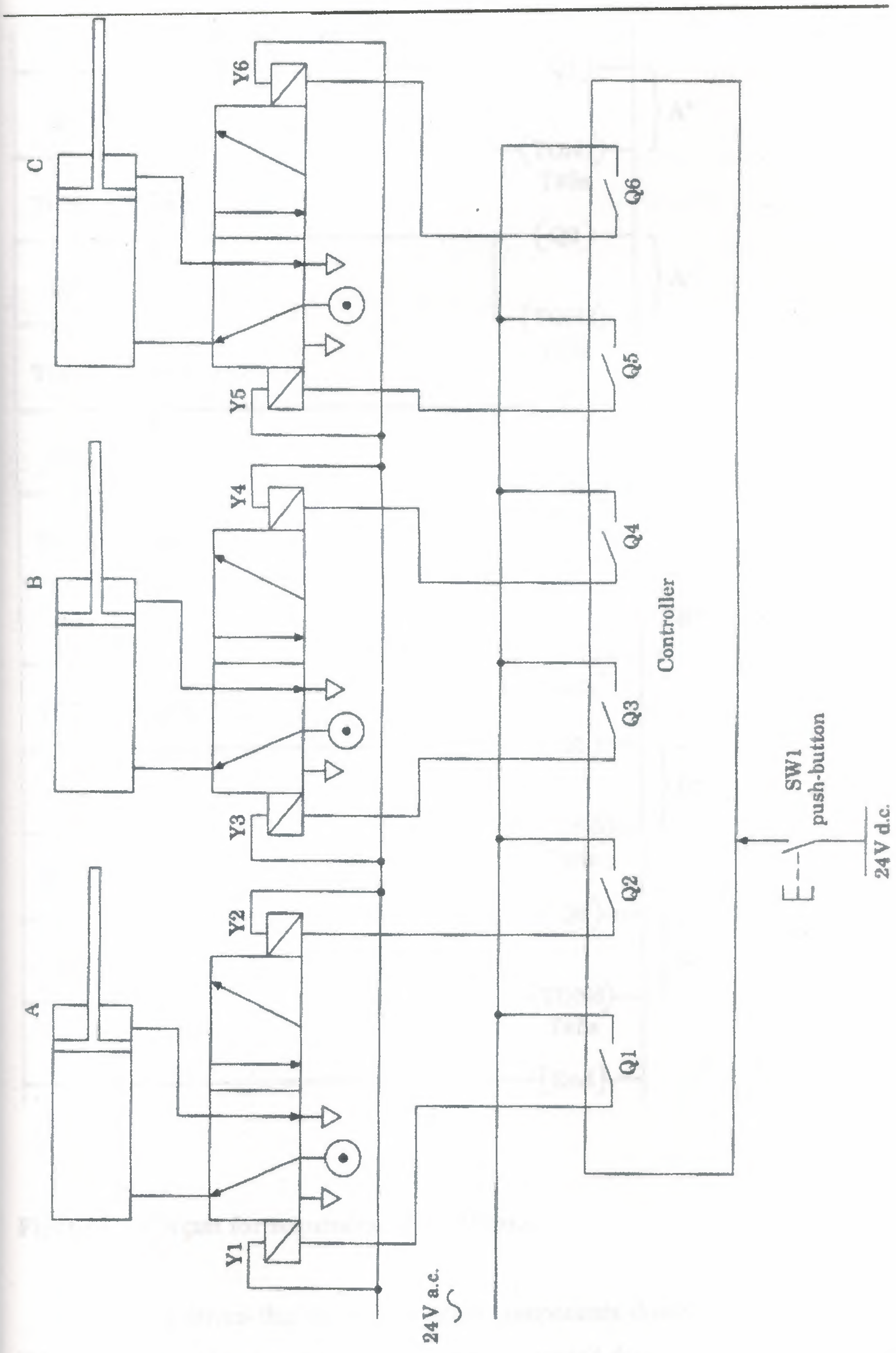


Figure 7.4 Control of the pneumatic pistons .

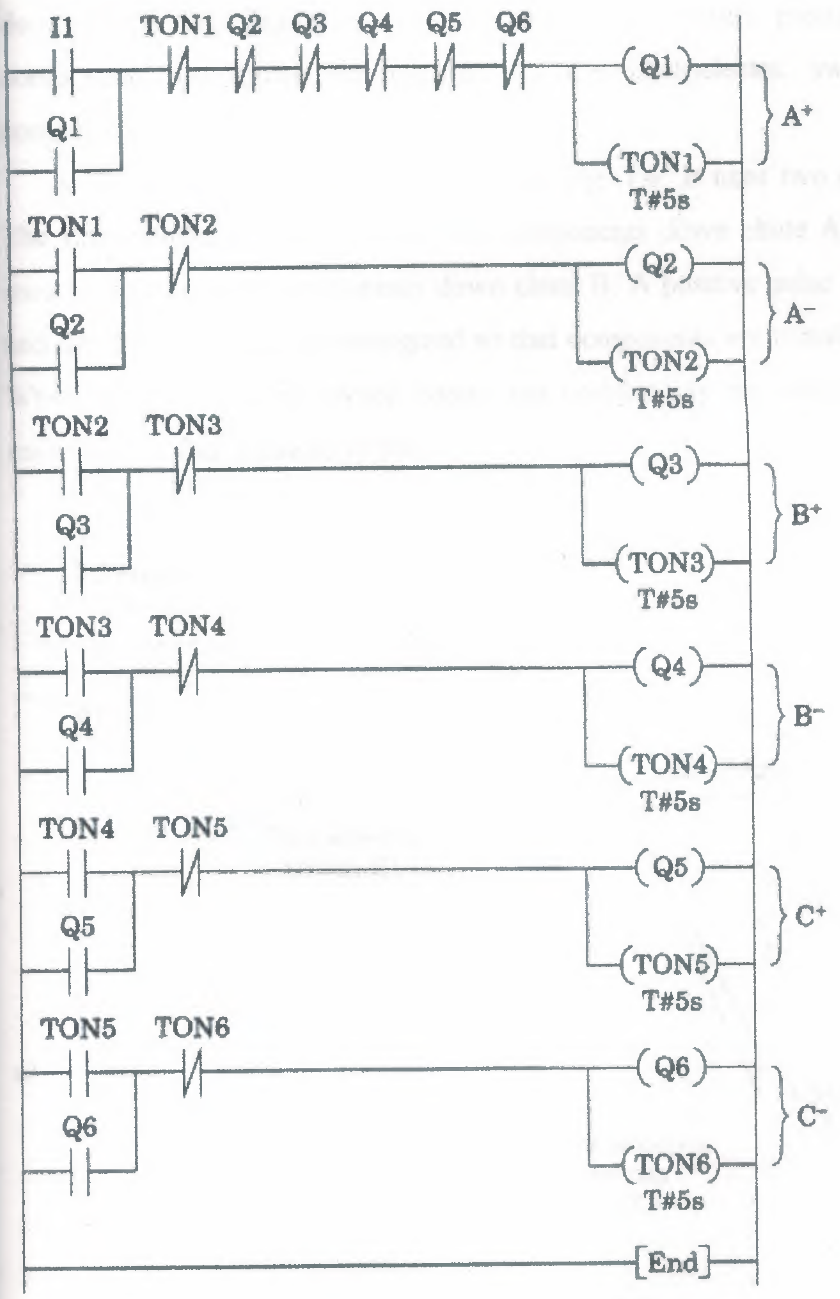
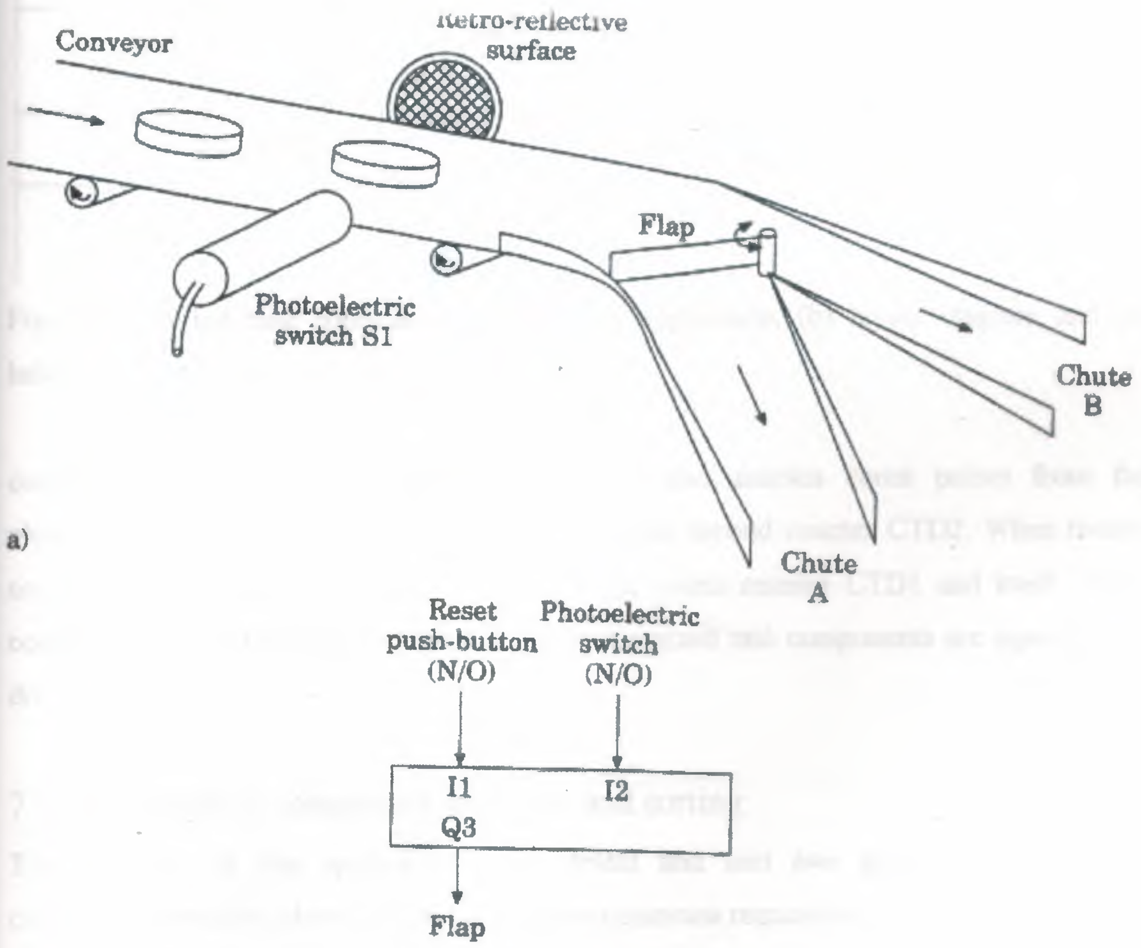


Figure 7.5 Circuit for sequencing the pistons.

A solenoid-driven flap is used to direct components down either chute A or chute B. When the solenoid is energized components are routed down path B. When the solenoid is de-energized the flap (via a spring return mechanism) positions itself so that the components are routed down chute A. The photoelectric switch is used to count components travelling along the conveyor.

A ladder program solution is shown in Fig. 7.6. It uses two counter function blocks. The first counter is used to direct ten components down chute A. The second counter is used to direct twenty components down chute B. A positive pulse on Ii resets the counters and the flap solenoid is de-energized so that components are initially routed down chute A. When the photoelectric switch counts ten components the output Q1 of counter CTD1 energizes the flap solenoid so that



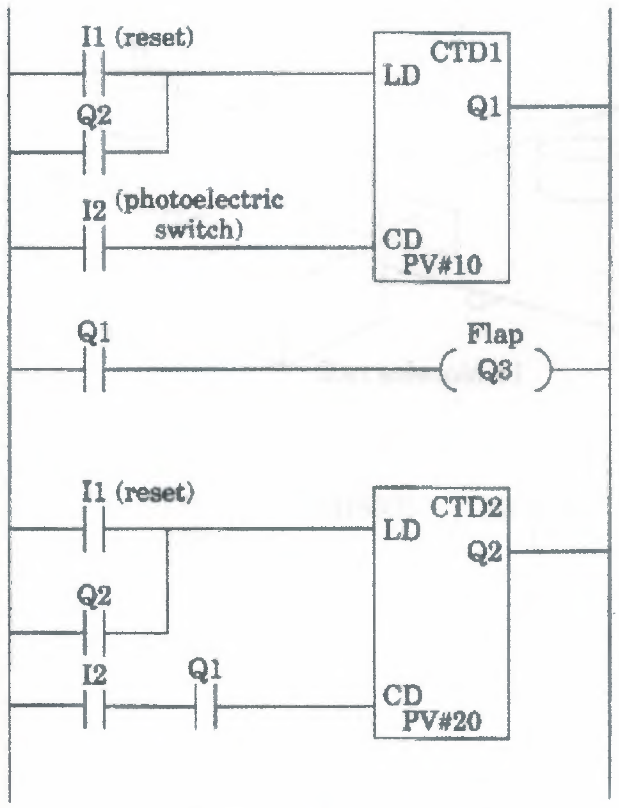


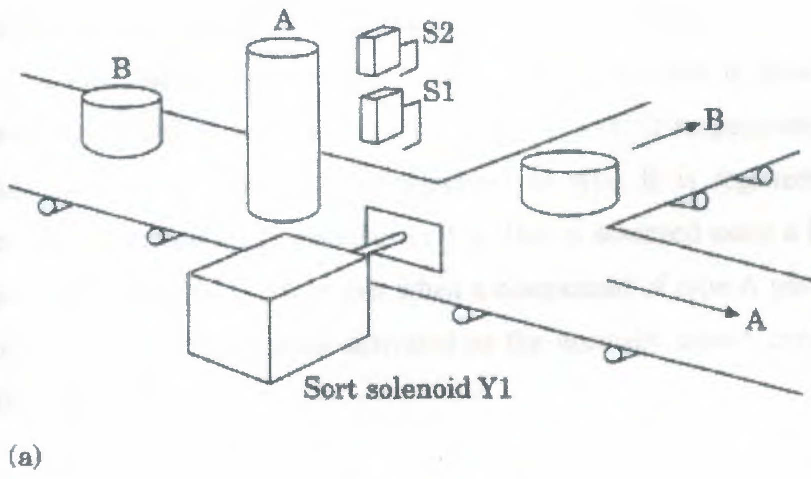
Figure 7.6 A batching application example: (a) application, (b) layout diagram and (c) ladder diagram.

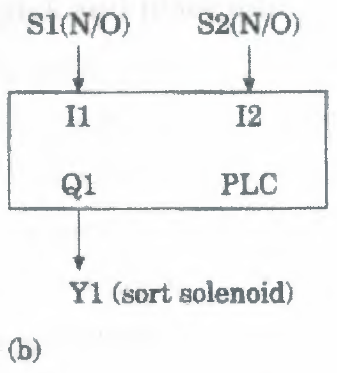
components will be routed down chute B. It also enables count pulses from the photoelectric switch to pass to the count input of the second counter CTD2. When twenty components are counted by ~TD2 its output Q2 resets counter CTD1 and itself. When counter CTD1 is reset the flap solenoid is de-energized and components are again routed down chute A.

7.4 Example 4: component detection and sorting

The objective of this application is to detect and sort two geometrically different components traveling along a conveyor. This is a common requirement in packing lines.

The example application is illustrated in Fig. 7.7. Two components labeled A and B travel along a conveyor. Component B is to be deflected by a sort solenoid





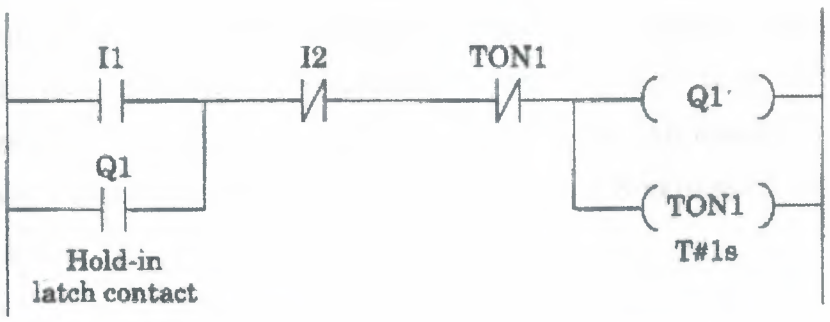


Figure 7.7 Component detection and sorting: (a) application, (b) layout diagram and (c) ladder diagram.

while component A is to be allowed to pass the sort area. Two proximity switches, S_1 and S_2, are located at different heights to produce a sorting signal. When component A is in the sort area both proximity switches S_1 and S_2 turn on. When component B is in the sort area only switch S_1 is turned on. The sort solenoid is to be energized for 1 s to deflect all components of type B down a different conveyor feed path.

A PLC layout diagram and ladder program solution is shown in Fig. 7.7. Proximity switches $S_1$and S_2 are connected to inputs Ii and 12 respectively. If input Ii (i.e. S_1)goes high but not 12 (i.e. S2) a component of type B is registered and the sort solenoid connected to coil $Q_1$is activated for 1 s. This is achieved using a latch circuit which is reset using the timer TON_1 Note that when a component of type A passes the sort area the solen id is prevented from being activated as the normally closed contact 12 breaks the power flow path to Q_1 .

7.5 Example 5: pick and place unit

The objective of this application is to show how position switches can be used to control the movement of an axis of a pick and place unit. The pick and place unit shown in Fig. 7.8 has one rotary (0) and two linear (X and Z) degrees of freedom. The motions are on/off in nature, being actuated by pneumatic pistons. For example, the X motion is in either the X~ position or the X position. The gripper can also be pneumatically operated, being either open or shut. Actuating appropriate solenoids controls all motions.

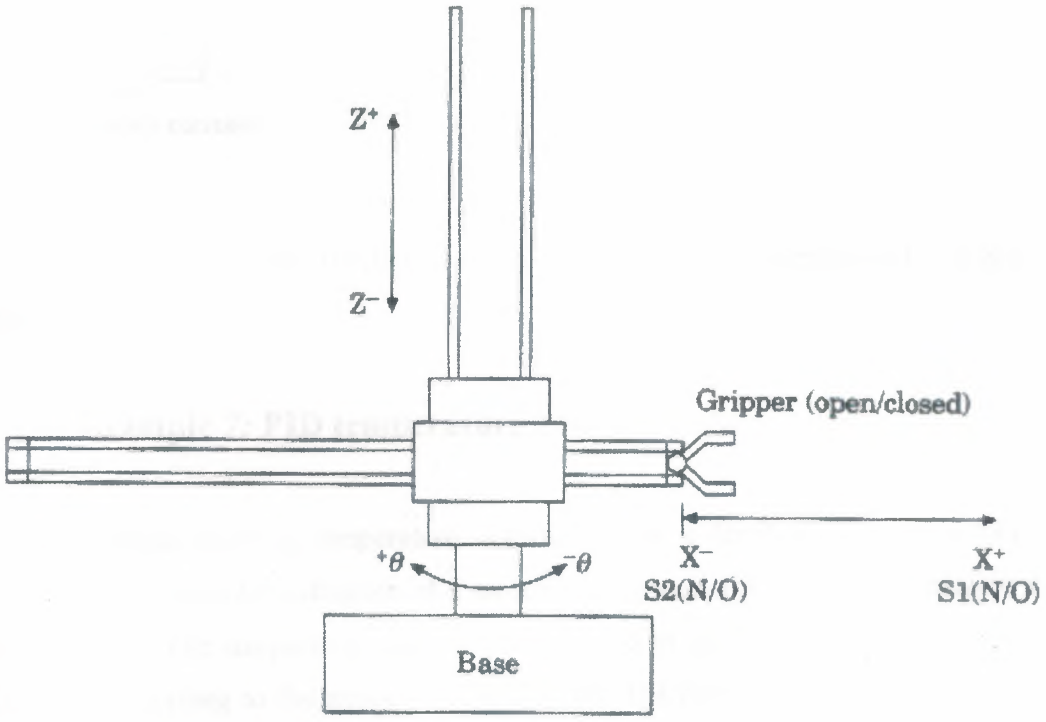
Position switches can be mounted on the unit so that they turn on when a particular motion has occurred. For example, consider two position switches S_1 and 52 mounted so that they register the fully extendedX~ position (i.e. S_1 is on and S2 is off) and the fully retracted X position (i.e. S_1 is off and S2 is on). An example ladder program fragment which uses these signals to sequence the X~ and X movements back and forth is shown in Fig. 7.8.

7.6 Example 6: checking for a missing bottle cap

The objective of this application example is to detect and reject bottles emerging from a filling and capping machine that have not been capped. An application sketch, the assignment of input and output points and a ladder program are shown in Fig. 7.9.

Bottle caps can be detected by using two retro-reflective photoelectric switches. One detects the presence of a bottle (i.e. S1) and the other detects the presence of a cap (i.e. S2).

The photoelectric switch S1 is mechanically mounted so that it is triggered just before the cap photoelectric switch S2. S1 is connected to input point I1 and S2 is connected to input point I2. Therefore S1 starts timer TON1 while S2 resets TON1. If a cap signal is not detected within 2 s once the timer TON1 has been triggered with a bottle signal the solenoid-driven reject piston is energized by TON1. The timer TON2 holds the reject solenoid energized for 1 s. The timer TON1 is used to overcome any problems relating to operation timing due to the mounting of the sensors.



Pick and place unit

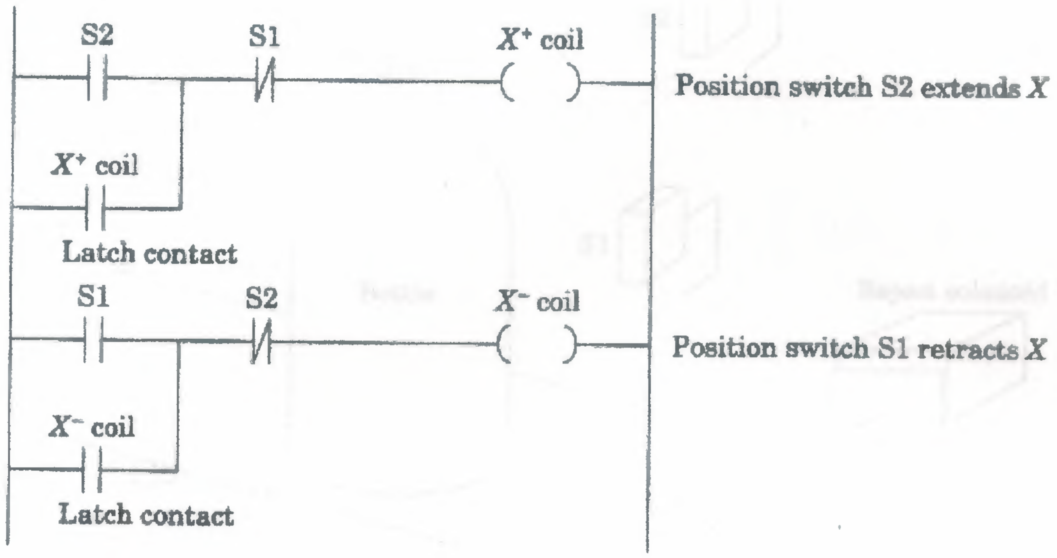
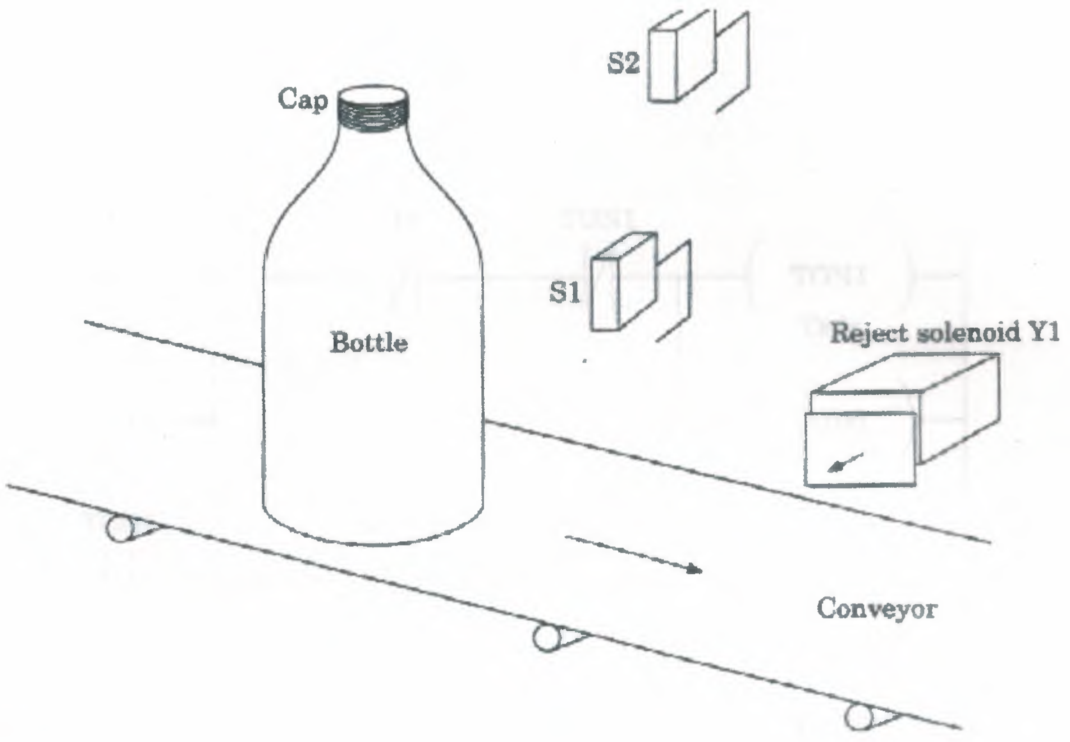


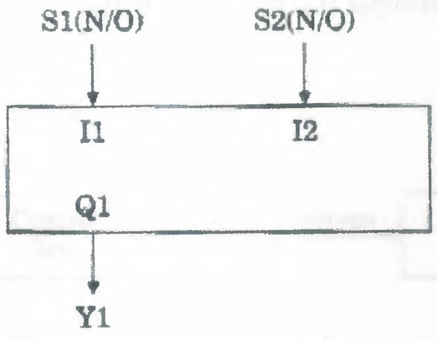
Figure 7.8 Using limit switches to control the *X* and X~ movements of a pick and place unit.

7.7 Example *7:* PID temperature control

Applications such as temperature control require a feedback or closed-loop control system. The basic block diagram of a closed-loop system for temperature control is shown in Fig. 7.10. The temperature sensor is connected to an ADC which generates a binary numl~er according to the temperature measured. The heater is controlled by a DAC. The controller attempts to vary the input to the heater to make the measured temperature equal to the desired set-point value.

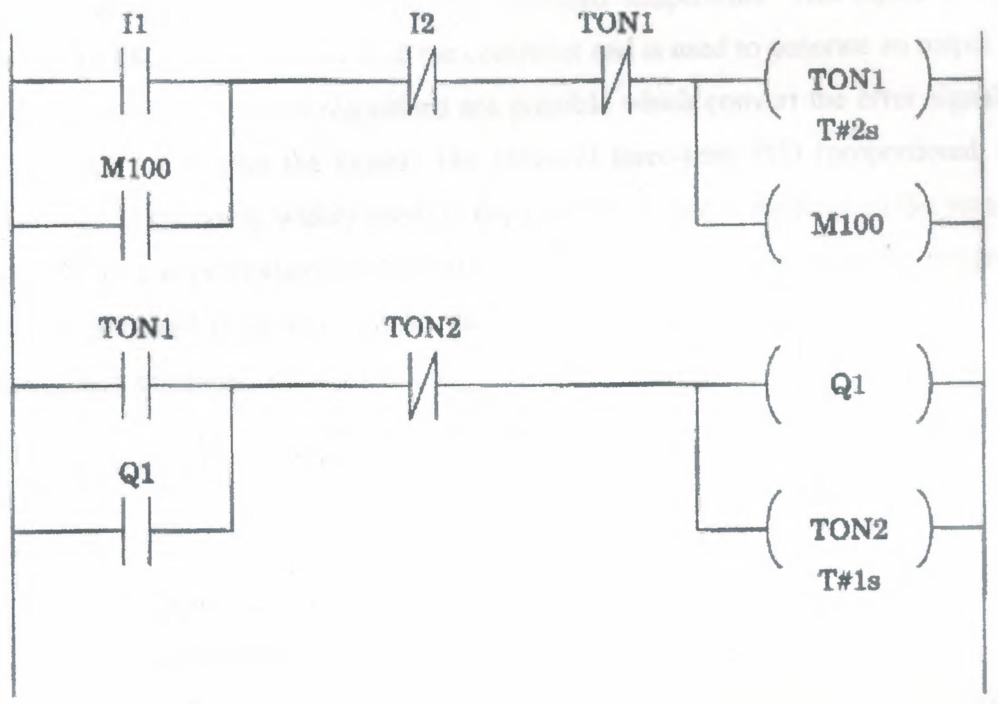

S2
Cap
S1
Bottle
Reject solenoid Y1
Conveyor

(a)


S1(N/O)
S2(N/O)
I1
I2
Q1
Y1

(b)

(b)



(c)

Figure 7.9 Checking for a missing bottle cup: (a) application, (b) layout diagram and (c) ladder diagram.

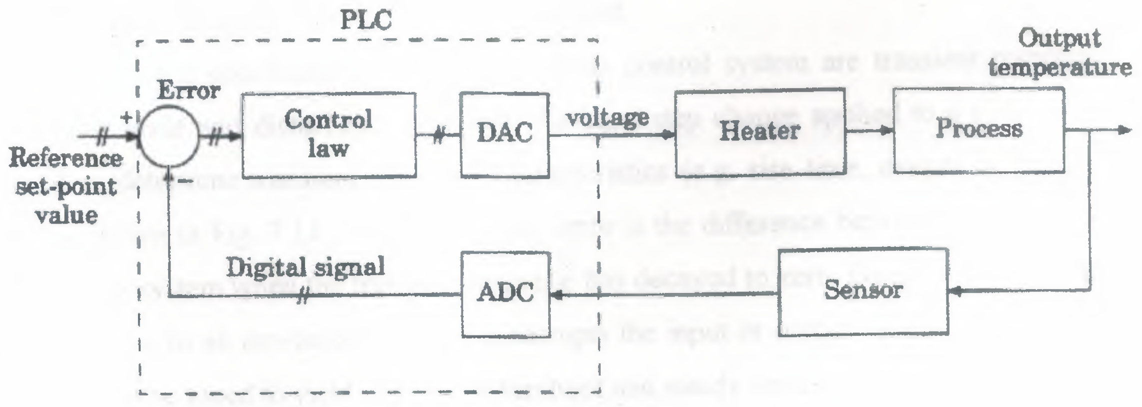


Figure 7.10 Closed-loop temperature control.

The comparator detects the difference between a reference or set-point value and the output of the ADC which represents the measured temperature. This signal is called the error signal. The error signal is fed to the controller and is used to generate an output signal.

Various types of control algorithms are possible which convert the error signal into an output signal for driving the heater. The classical three-term PID (proportional, integral, derivative) control law is widely used. In this case the output is made up of the sum of three terms. The first is proportional to the error, the second is proportional to the integral of the error and the third is proportional to the derivative or rate of change of error. The PID control action equation can be written as

$$x = K_p\left(e + \frac{1}{T_i}\int edt + T_d\frac{de}{dt}\right)$$

where

x	=	controller output KP
K_P	=	proportional gain
e	=	error
T_i	=	integral time
T_d	=	derivative time

Of course, it is possible to simplify the algorithm so that only the proportional (P) or proportional plus integral (P1) terms are implemented.

The important specifications for a closed-loop control system are transient response, steady state error and disturbance rejection. An input step change applied to a system can be used to determine transient response characteristics (e.g. rise time, overshoot, settling time), as shown in Fig. 7.11. The steady state error is the difference between the input and output of the system when the transient response has decayed to zero. Disturbance rejection is the response to an unwanted signal that corrupts the input or output of a process. A PID controller can be tuned to yield minimum overshoot and steady state error.

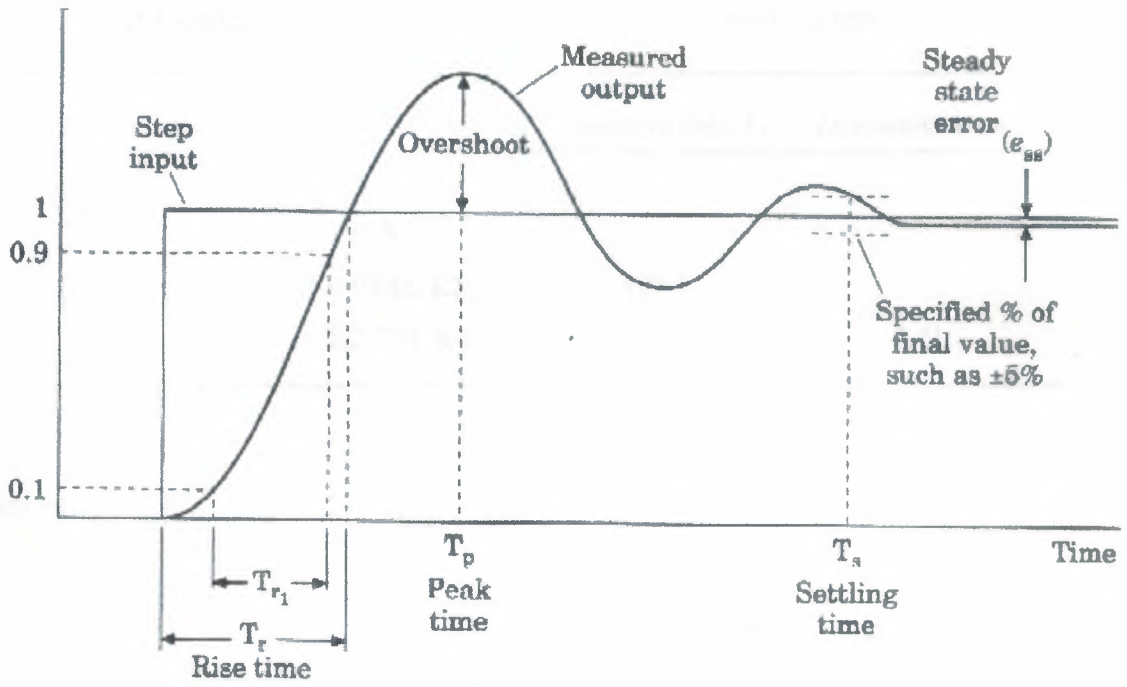


Figure 7.11 Dynamic response of an under-damped system to a step input

The Ziegler—Nichols empirical methods for estimating the controller settings of gain, integral time and derivative time can be applied when little is known about the dynamics of the system. There are two approaches that can be used, which are:

• Open-loop step response test

• Closed-loop response test

In the first method, an open-loop step response is performed on the process before the controller is installed. Applying a step change to the input and monitoring the output response achieve this. By drawing a tangent to the measured output response curve two parameters *L* and *T* are obtained, as shown in Fig. 7.12. The controller settings for the P, Pt and PID control actions are calculated using the two parameters *L, T* and the steady state gain *K* (i.e. the ratio of *B/A),* as shown in Table 7.1.

Table 7.1 Controller settings for the Ziegler—Nichols open-loop test

Control action	*Proportional gain* K_p	*Integral time* T_i	*Derivative time* T_d
P	T/(LK)		
PI	0.9T/(L K)	L/0.3	
PID	1.2 T/(L K)	2L	0.5L

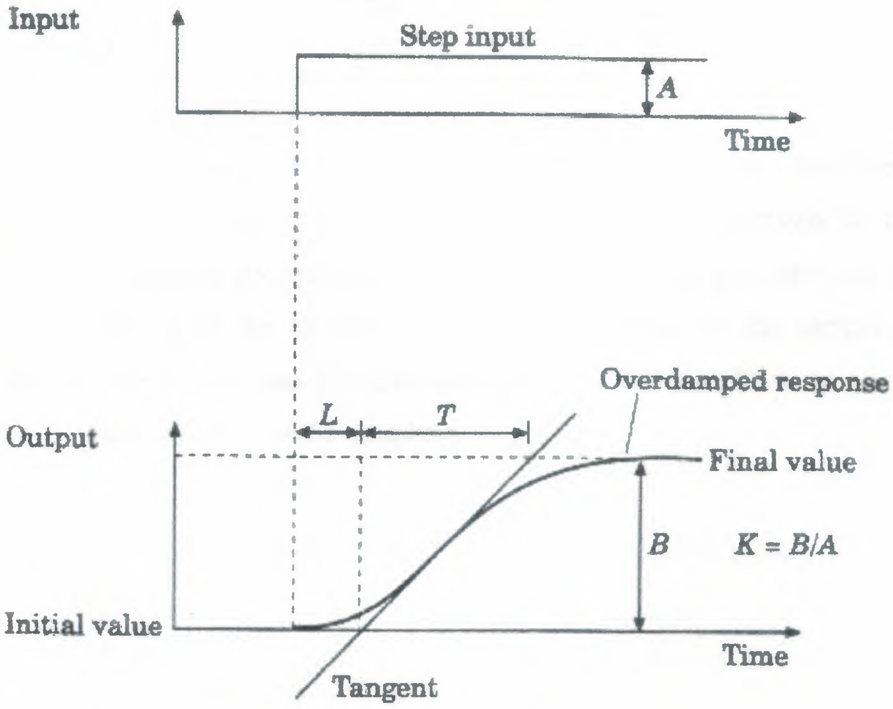


Figure 7.12 Extracting Ziegler—Nichols parameters from an open-loop step response curve of a process.

In the closed-loop method, the controller is installed and set to proportional action only with the integral and derivative terms made inoperative. Starting with a small value, the controller gain K_c is progressively increased in stages until a small step change in the set-point value causes a continuous fixed amplitude oscillation of the output temperature as measured with the sensor. The controller gain K_{osc} and the period of the oscillations T_{osc} are used to calculate the controller settings as shown in Table 7.2.

Table 7.2 Controller settings for the Ziegler—Nichols closed-loop test

Control action	*Proportional gain* Kp	*Integral time* Ti	*Derivative time* Td
P	$0.5\ K_{osc}$		
PI	$0.45\ K_{osc}$	$T_{osc}/1.2$	
PID	$0.6\ K_{osc}$	$T_{osc}/2$	$T_{osc}/8$

The PID equation is implemented digitally using numerical integration and differentiation using error values obtained at sampling intervals of τ seconds. Assuming that the sampling interval time remains constant, the integral term can be represented as the summation of all the sampled error values multiplied by the sampling interval. Likewise, the derivative term can be represented as the difference between two successive sampled error values divided by the sampling.

Table of Symbol

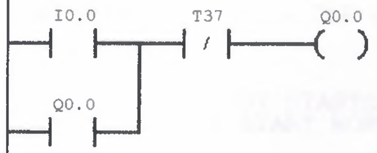
INSTRUCTION	LADDER SEMBOL	SIMATIC S7
LOAD	├┤├─	LD
AND	─┤├─	A
OR	└┤├┘	O
NOT	/	NOT
LOAD NOT	├┤/├─	LDN
AND NOT	─┤/├─	AN
OR NOT	└┤/├┘	ON
AND BLOCK		ALD
OR BLOCK		OLD
OUT	─O┤ ─( )┤	=
END	─(END)─	MEND

PROGRAM TITLE COMMENTS

Press F1 for help and example program

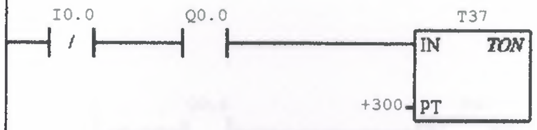
Network 1 CHOOSER BANDS

WHEN CONTACTS I0.0 NORMALY OPEN CHANGE POSITION THE OUTPUT OF Q0.0 WORK



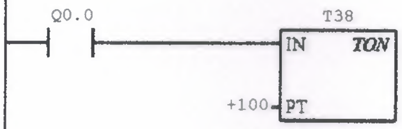
Network 2 WHEN THE Q0.0 STOP

IF I0.0 NORMALY CLOSED CONTACT AND Q0.0 NORMALY OPEN CONTACTS ARE CLOSED AND THEN T37 WILL WORK .WHEN THE T37 WORK, CONTACTS's POSITION WILL CHANGE AFTER 30 SECOND.



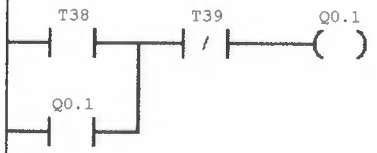
Network 3 WHEN THE Q0.1 START.

WHEN THE CONTACT POSITION OF Q 0.0 OUTPUT CHANGE T38 WILL WORK .WHEN THE T38 WORK,CONTACTS WILL CHANGE POSITION AFTER 10 SECONDS.



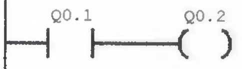
Network 4 WHEN THE WASHING START

WHEN NORMALLY OPEN CONTACTS OF T38 CLOSED Q0.1 OUTPUT WILL WORK.AND Q0.1 OUTPUT LOCK ITSELF WITH Q0.1 NORMALY OPEN CONTACTS.



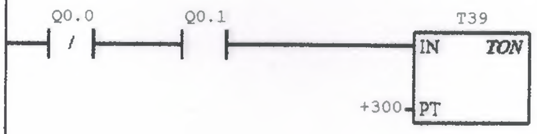
Network 5 WHEN THE JET OF WATER WORKS

WHEN Q0.1 OUTPUT STARTS TO WORK,Q0.1 OUTPUT CONTACT WILL CHANGE POSITION AND THEN Q0.2 START WORK.



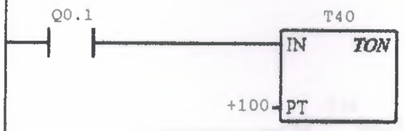
Network 6 WHEN THE WASHES STOP

IF Q0.0 NORMALY CLOSED CONTACT AND Q0.1 NORMALY OPEN CONTACTS ARE CLOSED ,THEN T39 WILL WORK .WHEN THE T37 WORK, CONTACTS's POSITION WILL CHANGE AFTER 30 SECOND.



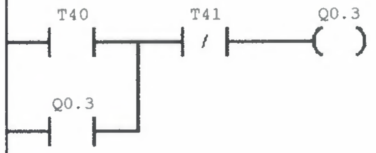
Network 7 DISINFECTOR START WAITING FOR WORK

IF Q0.1NORMALY OPEN CONTACT CLOSED, T40 START TO WORK.THEN T40 WILL CHANGE CONTACTS POSITION AFTER 30 SECONDS.



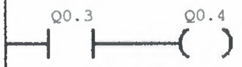
Network 8 DISINFECT IS WORK

T40 NORMALY OPEN CONTACT CLOSED Q0.3 OUTPUT WORK.LOCKED BY THE Q 0.3 NORMALY OPEN CONTACT .



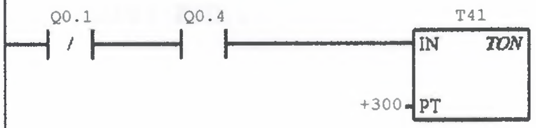
Network 9 JET OF DISINFECT LIQUID WORK

WORK THE SAME TIME Q0.3 NORMALY OPEN CONTACT WHEN THE CLOSED Q0.4 WORK.



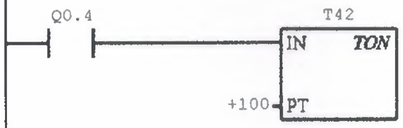
Network 10 WHEN THE DISINFECT STOP

IF Q0.1 NORMALY CLOSE CONTACTS AND Q0.4 NORMALY OPEN CONTACT CLOSED T41 WILL START WORK.WHEN THE T41 WORK, IT CHANGE THE CONTACT AFTER 30 SECONDS.



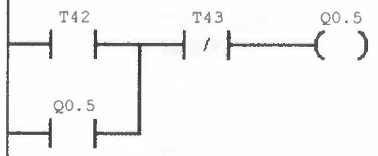
Network 11 DIMENSION START WAITING FOR WORK

WHEN THE Q0.4 NORMALY OPEN CONTACT CLOSED T42 WORK.WHEN THE T42 TIMER CHANGE THE CHANGE THE POSITIONS OFTHE CONTACTS AFTER 10 SECONDS.



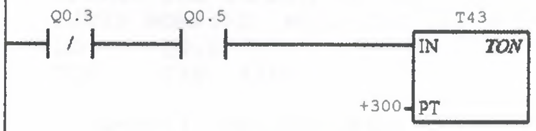
Network 12 DIMENSION WORK

T42 NORMALY OPEN CONTACT WHEN THE CLOSED Q0.5 WORK.AND THAN LOCKED BY THE Q0.5 NORMALY OPEN CLOSE CONTANCT CHANGE THE POSITION.



Network 13 WHEN THE DIMENSION STOP

IF Q0.4 NORMALY CLOSE CONTACT CLOSED AND Q0.5 NORMALY OPEN CONTACT WHEN THE CHANGE OF POSITION T43 WORK.WHEN THE T43 TIMER WORK.CAHNGE THE CONTACTS AFTER 30 SECONDS.



Network 14 END OF THE PROGRAM

THE HAPPY END..

```
//
//PROGRAM TITLE COMMENTS
//
//Press F1 for help and example program
//

NETWORK 1 //CHOISER BANDS
//WHEN THE CHANGE  POSITION OF CONTACTS I0.0 NORMALY OPEN
//Q0.0 WORK
//
//
LD      I0.0
O       Q0.0
AN      T37
=       Q0.0

NETWORK 2 //WHEN THE Q0.0 STOP
//IF I0.0 NORMALY CLOSED CONTACT İS CLOSED AND Q0.0 NORMALY OPEN
//CONTANCT IS CLOSED AND THEN T37 WORK .WHEN THE T37 WORK ,CHANGE THE
CONTACTS
//OF POSITIONS AFTER 30 SECOND.
LDN     I0.0
A       Q0.0
TON     T37, +300

NETWORK 3 //WHEN THE Q0.1 İS START
//WHEN THE CHANGE OF THE CONTACT POSITION OF Q 0.0 OUTPUT
//T38 WORKING .WHEN THE T38 WORK CHANGE THE CONTACTS AFTER 10 SECONDS.
LD      Q0.0
TON     T38, +100

NETWORK 4 //WASHES WORK
//IF T38 NORMALY OPEN CONTACTS IS CLOSED Q0.1 İS WORK.AND THEN LOCKED
//BY THE Q0.1 NORMALY OPEN CONTACTS.
LD      T38
O       Q0.1
AN      T39
=       Q0.1

NETWORK 5 //JET OF WATER WORK
//WHEN THE WORK OF Q0.1 OUTPUT CAHNGE THE CONTACT AND THAN Q0.2 WORK.
LD      Q0.1
=       Q0.2

NETWORK 6 //WHEN THE WASHES STOP
//IF Q 0.0 NORMALY CLOSED CONTACT  CLOSED AND Q0.1 NORMALY OPEN CONTACT
//CLOSED  T39 TIMER WORK .WHEN THE TIMER WORK CHANGE THE CONTACTS AFTER 30
SECOND
LDN     Q0.0
A       Q0.1
TON     T39, +300

NETWORK 7 //WHEN THE DISINFECT  WORK
//IF Q0.1NORMALY OPEN  CONTACT CLOSED T40 WORK.WHEN THE T40 TIMER CHANGE THE
//CONTACT AFTER 10 SECOND.
LD      Q0.1
TON     T40, +100

NETWORK 8 //DISINFECT IS WORK
//T40 NORMALY OPEN CONTACT CLOSED Q0.3 OUTPUT WORK.LOCKED  BY THE Q 0.3
//NORMALY OPEN CONTACT .
LD      T40
O       Q0.3
AN      T41
=       Q0.3

NETWORK 9 //JET OF DISINFECT LIQUID WORK
//WORK THE SAME TIME Q0.3 NORMALY OPEN CONTACT WHEN THE CLOSED Q0.4 WORK.
LD      Q0.3
=       Q0.4
```

```
NETWORK 10      //WHEN THE DISINFECT STOP
//IF Q0.1 NORMALY CLOSE CONTACTS CLOSED AND Q0.4 NORMALY OPEN CONTACT
//WHEN THE CLOSED T41 WORK.WHEN THE T41 WORK CHANGE THE CONTACT
//AFTER 30 SECONDS.
LDN    Q0.1
A      Q0.4
TON    T41, +300

NETWORK 11      //WHEN THE DIMENSION WORK
//WHEN THE Q0.4 NORMALY OPEN CONTACT CLOSED T42 WORK.WHEN THE
//T42 TIMER CHANGE THE CHANGE THE POSITIONS OFTHE CONTACTS AFTER 10 SECONDS.
LD     Q0.4
TON    T42, +100

NETWORK 12      //DIMENSION IS WORK
//T42 NORMALY OPEN CONTACT WHEN THE CLOSED Q0.5 WORK.AND THAN LOCKED
//BY THE Q0.5 NORMALY OPEN CLOSE CONTANCT CHANGE THE POSITION.
LD     T42
O      Q0.5
AN     T43
=      Q0.5

NETWORK 13      //WHEN THE DIMENSION STOP
//IF Q0.4 NORMALY CLOSE CONTACT CLOSED AND QO.5 NORMALY OPEN CONTACT
//WHEN THE CHANGE OF POSITION T43 WORK.WHEN THE T43 TIMER WORK.CAHNGE
//THE CONTACTS AFTER 30 SECONDS.
LDN    Q0.3
A      Q0.5
TON    T43, +300

NETWORK 14      //END OF THE PROGRAM
//THE HAPPY END..
MEND
```

CONCLUSION

In today's industry, small and big (underdeveloped, developed) factories tend to use otomation system to avoid the expense of workers and mistakes which are usually caused by the workers. The importans of speed in factories that make mass production is very big. So, these factories need machines that don't get tried and don't talk.

When we are in this position, we need PLC. These synchronous systems which are fast and dependent with each other are very important and useful inventations. They can also be used in our houses, garrages, offices and even in our cars.

Because of the fact that the programme, timing, counter and other electronic and electrocity power elements are all included in PLC , it supplies both space and economic profit (benefit) for us.

In addition to this, because of the fact that it can be scheduled , the same PLC can be used later for different propuses. In my project , there are circuit examples about various otomation systems.

If we summorize (in short , in conculision) PLC is a developed type of computer which is used in industry and it can be said to be the most important and useful invention of the last fifteen yaers of the twenty-first century .

PLC , which has a very important role in Turkiye's industry, also supplies a bazaar for electronic engineers who are interested in otomation.

REFERENCES

Reference: PLC's aid low cost automation D.H.Florence. professional Engineering, july 1988

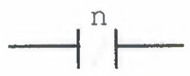
Reference: Programming Console CK-series operation manual OMRON , 1988

Reference: Hand-held Graphic programmer HP-911 operation manual, Toshiba 1995

Reference: Preparation of functıon charts for control system İnternational electrotehcnical commission, 1988

Normally Open Contact

Symbol:



Operands:		**PDS 210:**
n (bit):	I, Q, M, SM, T, C, V, S	I0.0 - I0.3 M0.0 - M7.7 SM0.0 - SM1.7

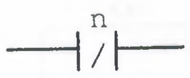
Description of operation:

The Normally Open Contact is closed when the scanned bit value stored at address n is equal to 1. Power flows through a normally open contact when closed (activated).

Used in series, a normally open contact is linked to the next LAD element by AND logic. Used in parallel, it is linked by OR logic.

Normally Closed Contact

Symbol:



Operands:		**PDS 210:**
n (bit):	I, Q, M, SM, T, C, V, S	I0.0 - I0.3 M0.0 - M7.7 SM0.0 - SM1.7

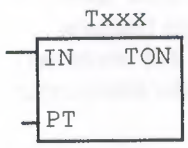
Description of operation:

The Normally Closed Contact is closed when the bit value stored at address n is equal to 0 . Power flows through the contact when closed.

Used in series, a normally closed contact is linked to the next LAD element by AND logic. Used in parallel, it is linked by OR logic.

Timer/On Delay

Symbol:



Operands:

Txxx (word):	CPU 212: 32-63 CPU 214: 32-63, 96-127 CPU 215/216: 32-63, 96-255
PT (word):	VW, T, C, IW, QW, MW, SMW, SW, AC, AIW, Constant, *VD, *AC

PDS 210:

Tx:	T0 - T3
PT:	0

Description of operation for CPUs 212, 214, 215, 216:

The On-Delay Timer (TON) box times up to the maximum value when the enabling Input (IN) comes on. When the current value (Txxx) is >= the Preset Time (PT), the timer bit turns on. It resets when the enabling input goes off. Timing stops upon reaching the maximum value.

In the status chart, you can display timer and counter values as either bits or words. If you display a timer or counter value as a bit, the output status is displayed (output on or off). If you display a timer or counter value as a word, the current value is used.

The On-Delay Timers time in one of three resolutions, depending on the timer number you use. Each increment of the current value is a multiple of the time base. For example, a preset of 20 for a 10-millisecond timer represents 200 milliseconds.

Resolution	Maximum Time	CPU 212	CPU 214	CPU 215/216
1 ms	32.767 seconds	T32	T32, T96	T32,T96
10 ms	327.67 seconds	T33-T36	T33-T36 T97-T100	T33-T36 T97-T100
100 ms	3276.7 seconds	T37-T63	T37-T63 T101-T127	T37-T63 T101-T255

Description of operation for CPU 210:

The On-Delay Timer (TON) box times up to the maximum value when the enabling input (IN) comes on. The timer resets when the enabling input turns off. Timing stops when the maximum value is reached.

Resolution	Maximum Time	CPU 210
100 ms	3276.7 seconds	T0 - T3

Understanding the S7-200 Timer Instructions

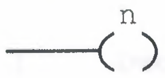
The S7-200 provides two different timer instructions: the On-Delay Timer (TON), and the Retentive On-Delay Timer (TONR). The two types of timers differ in the way that they react to the state of the enabling input.

- Both timers count up while the enabling input is on.
- Neither timer counts while the enabling input is off.
- When the enabling input is off, a TON timer automatically resets the time value, while a TONR timer does not reset, but instead retains the last count (or time) value. When a TONR timer is enabled again, new time value adds to the previous time value.

Therefore, the TON timer is best used when you are timing a single interval. The TONR timer is appropriate when you need to measure the accumulated time for a number of timed intervals.

Output

Symbol:



Operands:

n (bit): I, Q, M, SM, S, T, C, V

Operands:		**PDS 210:**
n (bit):	I, Q, M, SM, S, T, C, V	Q0.0 - Q0.3 M0.0 - M7.7

Description of operation:

An Output coil is turned on and the bit stored at address ***n*** is set to 1 when power flows to the coil.

A negated output can be created by placing a NOT (Invert Power Flow) contact before an output coil.

Access Properties of Data Areas

Area	Descrip.	Accessed as Bit	Accessed as Byte	Accessed as Word	Accessed as Dword	Can be Retentive	Can be Forced
I	Discrete inputs and image register	read/write	read/write	read/write	read/write	no	yes
Q	Discrete outputs and image register	read/write	read/write	read/write	read/write	no	yes
M	Internal memory bits	read/write	read/write	read/write	read/write	yes	yes
SM	Special memory bits (SM0 - SM29 are read-only)	read/write	read/write	read/write	read/write	no	no
V	Variable memory	read/write	read/write	read/write	read/write	yes	yes
T	Timer currents and Timer bits	T-bit read/write	no	T-current read/write	no	T0 - T31 & T64 - T65	no
C	Counter currents and Counter bits	C-bit read/write	no	C-current read/write	no	yes	no
HC	High-speed counter currents	no	no	no	read only	no	no
AI	Analog inputs	no	no	read only	no	no	yes
AQ	Analog outputs	no	no	write only	no	no	yes
AC	Accumulator registers	no	read/write	read/write	read/write	no	no