Live Webinar: Introduction to PLCs

Course Number: EE-02-500W

PDH: 3

Approved for: AK, AL, AR, GA, IA, IL, IN, KS, KY, LA, MD, ME, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, OH, OK, OR, PA, SC, SD, TN, TX, UT, VA, VT, WI, WV, and WY

New Jersey Professional Competency Approval #24GP00025600
North Carolina Approved Sponsor #S-0695
Maryland Approved Provider of Continuing Professional Competency
Indiana Continuing Education Provider #CE21800088

This document is the course text. You may review this material at your leisure before or after you purchase the course. In order to obtain credit for this course, complete the following steps:

1) Log in to My Account and purchase the course. If you don't have an account, go to New User to create an account.

2) After the course has been purchased, review the technical material and then complete the quiz at your convenience.

3) A Certificate of Completion is available once you pass the exam (70% or greater). If a passing grade is not obtained, you may take the quiz as many times as necessary until a passing grade is obtained (up to one year from the purchase date).

If you have any questions or technical difficulties, please call (508) 298-4787 or email us at admin@PDH-Pro.com.

1 Programmable logic controllers

This chapter is an introduction to the programmable logic controller, its general function, hardware forms and internal architecture. This overview is followed up by more detailed discussion in the following chapters.

1.1 Controllers

What type of task might a control system have? It might be required to control a sequence of events or maintain some variable constant or follow some prescribed change. For example, the control system for an automatic drilling machine (Figure 1.1(a)) might be required to start lowering the drill when the workpiece is in position, start drilling when the drill reaches the workpiece, stop drilling when the drill has produced the required depth of hole, retract the drill and then switch off and wait for the next workpiece to be put in position before repeating the operation. Another control system (Figure 1.1(b)) might be used to control the number of items moving along a conveyor belt and direct them into a packing case. The inputs to such control systems might be from switches being closed or opened, e.g. the presence of the workpiece might be indicated by it moving against a switch and closing it, or other sensors such as those used for temperature or flow rates. The controller might be required to run a motor to move an object to some position, or to turn a valve, or perhaps a heater, on or off.

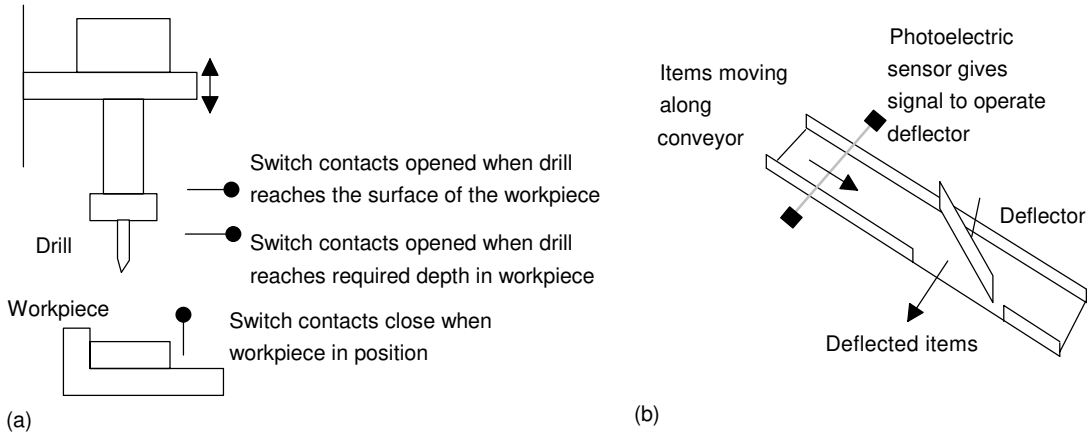


Figure 1.1 *An example of a control task and some input sensors: (a) an automatic drilling machine, (b) a packing system*

What form might a controller have? For the automatic drilling machine, we could wire up electrical circuits in which the closing or opening of switches would result in motors being switched on or valves being actuated. Thus we might have the closing of a switch activating a relay which, in turn, switches on the current to a motor and causes the drill to rotate (Figure 1.2). Another switch might be used to activate a relay and switch on the current to a pneumatic or hydraulic valve which results in pressure being switched to drive a piston in a cylinder and so results in the workpiece being pushed into the required position. Such electrical circuits would have to be specific to the automatic drilling machine. For controlling the number of items packed into a packing case we could likewise wire up electrical circuits involving sensors and motors. However, the controller circuits we devised for these two situations would be different. In the 'traditional' form of control system, the rules governing the control system and when actions are initiated are determined by the wiring. When the rules used for the control actions are changed, the wiring has to be changed.

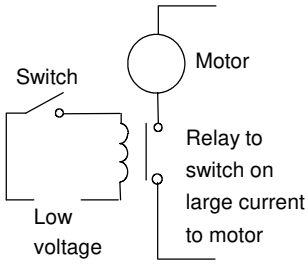


Figure 1.2 *A control circuit*

1.1.1 Microprocessor controlled system

Instead of hardwiring each control circuit for each control situation we can use the same basic system for all situations if we use a microprocessor-based system and write a program to instruct the microprocessor how to react to each input signal from, say, switches and give the required outputs to, say, motors and valves. Thus we might have a program of the form:

If switch A closes
Output to motor circuit
If switch B closes
Output to valve circuit

By changing the instructions in the program we can use the same microprocessor system to control a wide variety of situations.

As an illustration, the modern domestic washing machine uses a microprocessor system. Inputs to it arise from the dials used to select the required wash cycle, a switch to determine that the machine door is closed, a temperature sensor to determine the temperature of the water and

a switch to detect the level of the water. On the basis of these inputs the microprocessor is programmed to give outputs which switch on the drum motor and control its speed, open or close cold and hot water valves, switch on the drain pump, control the water heater and control the door lock so that the machine cannot be opened until the washing cycle is completed.

1.1.2 The programmable logic controller

A *programmable logic controller* (PLC) is a special form of microprocessor-based controller that uses a programmable memory to store instructions and to implement functions such as logic, sequencing, timing, counting and arithmetic in order to control machines and processes (Figure 1.3) and are designed to be operated by engineers with perhaps a limited knowledge of computers and computing languages. They are not designed so that only computer programmers can set up or change the programs. Thus, the designers of the PLC have pre-programmed it so that the control program can be entered using a simple, rather intuitive, form of language, see Chapter 4. The term *logic* is used because programming is primarily concerned with implementing logic and switching operations, e.g. if A or B occurs switch on C, if A and B occurs switch on D. Input devices, e.g. sensors such as switches, and output devices in the system being controlled, e.g. motors, valves, etc., are connected to the PLC. The operator then enters a sequence of instructions, i.e. a program, into the memory of the PLC. The controller then monitors the inputs and outputs according to this program and carries out the control rules for which it has been programmed.

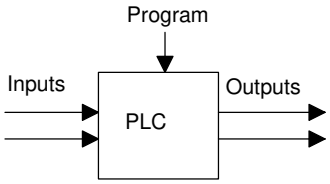


Figure 1.3 *A programmable logic controller*

PLCs have the great advantage that the same basic controller can be used with a wide range of control systems. To modify a control system and the rules that are to be used, all that is necessary is for an operator to key in a different set of instructions. There is no need to rewire. The result is a flexible, cost effective, system which can be used with control systems which vary quite widely in their nature and complexity.

PLCs are similar to computers but whereas computers are optimised for calculation and display tasks, PLCs are optimised for control tasks and the industrial environment. Thus PLCs are:

1 Rugged and designed to withstand vibrations, temperature, humidity and noise.
2 Have interfacing for inputs and outputs already inside the controller.

3 Are easily programmed and have an easily understood programming language which is primarily concerned with logic and switching operations.

The first PLC was developed in 1969. They are now widely used and extend from small self-contained units for use with perhaps 20 digital inputs/outputs to modular systems which can be used for large numbers of inputs/outputs, handle digital or analogue inputs/outputs, and also carry out proportional-integral-derivative control modes.

1.2 Hardware

Typically a PLC system has the basic functional components of processor unit, memory, power supply unit, input/output interface section, communications interface and the programming device. Figure 1.4 shows the basic arrangement.

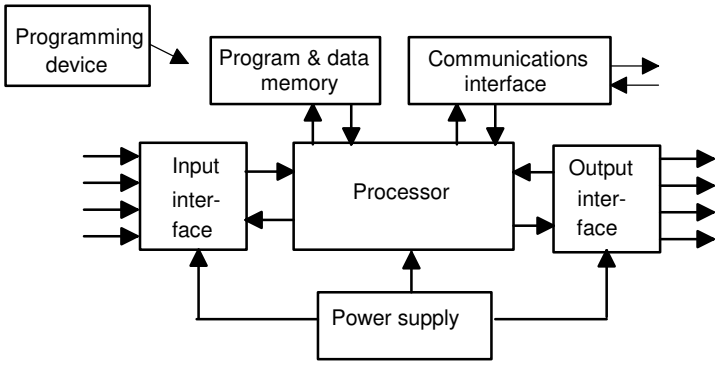


Figure 1.4 *The PLC system*

1 The *processor unit* or *central processing unit (CPU)* is the unit containing the microprocessor and this interprets the input signals and carries out the control actions, according to the program stored in its memory, communicating the decisions as action signals to the outputs.
2 The *power supply unit* is needed to convert the mains a.c. voltage to the low d.c. voltage (5 V) necessary for the processor and the circuits in the input and output interface modules.
3 The *programming device* is used to enter the required program into the memory of the processor. The program is developed in the device and then transferred to the memory unit of the PLC.
4 The *memory unit* is where the program is stored that is to be used for the control actions to be exercised by the microprocessor and data stored from the input for processing and for the output for outputting.
5 The *input and output sections* are where the processor receives information from external devices and communicates information to external devices. The inputs might thus be from switches, as illustrated in Figure 1.1(a) with the automatic drill, or other sensors such as photo-electric cells, as in the counter mechanism in Figure 1.1(b), temperature sensors, or flow sensors, etc. The outputs might be to motor starter coils, solenoid valves, etc. Input and output

interfaces are discussed in Chapter 2. Input and output devices can be classified as giving signals which are discrete, digital or analogue (Figure 1.5). Devices giving *discrete* or *digital signals* are ones where the signals are either off or on. Thus a switch is a device giving a discrete signal, either no voltage or a voltage. *Digital* devices can be considered to be essentially discrete devices which give a sequence of on–off signals. *Analogue* devices give signals whose size is proportional to the size of the variable being monitored. For example, a temperature sensor may give a voltage proportional to the temperature.

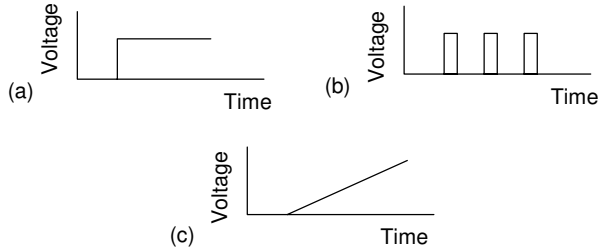


Figure 1.5 *Signals: (a) discrete, (b) digital, (c) analogue*

6 The *communications interface* is used to receive and transmit data on communication networks from or to other remote PLCs (Figure 1.6). It is concerned with such actions as device verification, data acquisition, synchronisation between user applications and connection management.

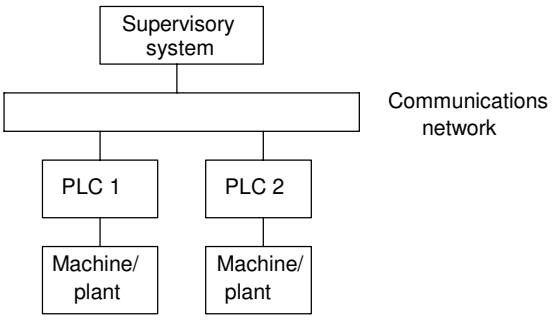


Figure 1.6 *Basic communications model*

1.3 Internal architecture

Figure 1.7 shows the basic internal architecture of a PLC. It consists of a central processing unit (CPU) containing the system microprocessor, memory, and input/output circuitry. The CPU controls and processes all the operations within the PLC. It is supplied with a clock with a frequency of typically between 1 and 8 MHz. This frequency determines the operating speed of the PLC and provides the timing and synchronisation for all elements in the system. The information within the PLC is carried by means of digital signals. The internal paths along which digital signals flow are called *buses*. In the physical sense, a bus is just a number of

conductors along which electrical signals can flow. It might be tracks on a printed circuit board or wires in a ribbon cable. The CPU uses the *data bus* for sending data between the constituent elements, the *address bus* to send the addresses of locations for accessing stored data and the *control bus* for signals relating to internal control actions. The *system bus* is used for communications between the input/output ports and the input/output unit.

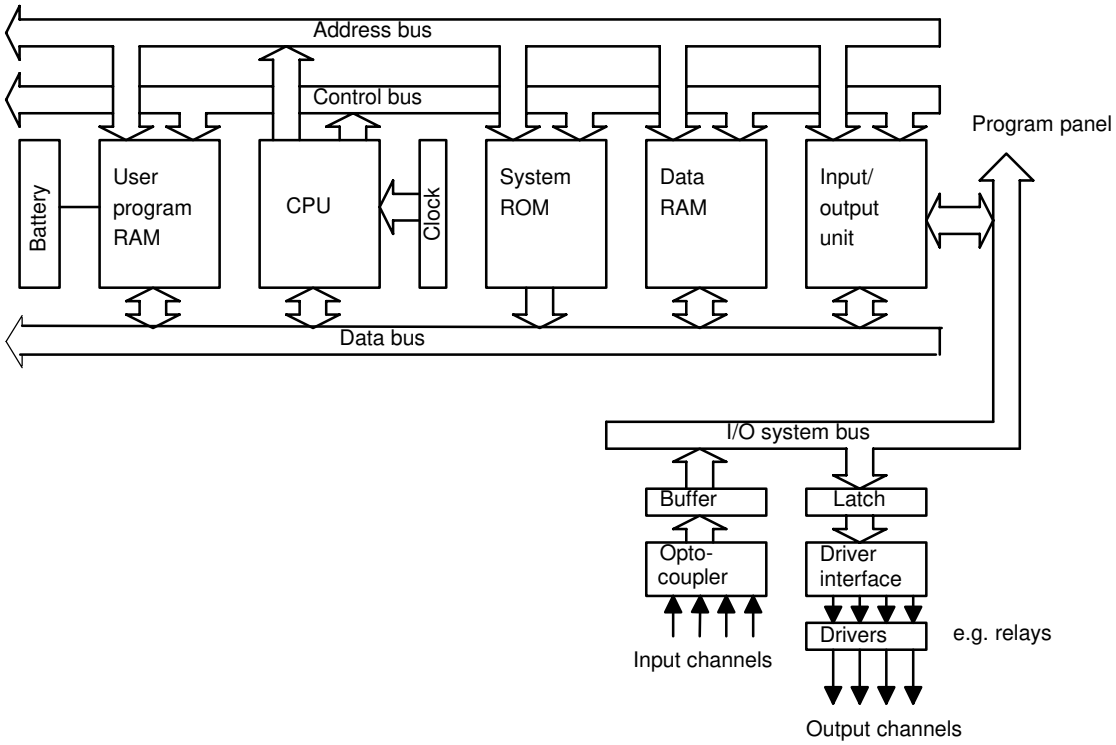


Figure 1.7 *Architecture of a PLC*

1.3.1 The CPU

The internal structure of the CPU depends on the microprocessor concerned. In general they have:

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

1.3.2 The buses

The buses are the paths used for communication within the PLC. The information is transmitted in binary form, i.e. as a group of *bits* with a bit

being a binary digit of 1 or 0, i.e. on/off states. The term *word* is used for the group of bits constituting some information. Thus an 8-bit word might be the binary number 00100110. Each of the bits is communicated simultaneously along its own parallel wire. The system has four buses:

1 The *data bus* carries the data used in the processing carried out by the CPU. A microprocessor termed as being 8-bit has an internal data bus which can handle 8-bit numbers. It can thus perform operations between 8-bit numbers and deliver results as 8-bit values.
2 The *address bus* is used to carry the addresses of memory locations. So that each word can be located in the memory, every memory location is given a unique *address*. Just like houses in a town are each given a distinct address so that they can be located, so each word location is given an address so that data stored at a particular location can be accessed by the CPU either to read data located there or put, i.e. write, data there. It is the address bus which carries the information indicating which address is to be accessed. If the address bus consists of 8 lines, the number of 8-bit words, and hence number of distinct addresses, is $2^8 = 256$. With 16 address lines, 65 536 addresses are possible.
3 The *control bus* carries the signals used by the CPU for control, e.g. to inform memory devices whether they are to receive data from an input or output data and to carry timing signals used to synchronise actions.
4 The *system bus* is used for communications between the input/output ports and the input/output unit.

1.3.3 Memory

There are several memory elements in a PLC system:

1 System *read-only-memory* (*ROM)* to give permanent storage for the operating system and fixed data used by the CPU.
2 *Random-access memory (RAM)* for the user's program.
3 *Random-access memory (RAM)* for data. This is where information is stored on the status of input and output devices and the values of timers and counters and other internal devices. The data RAM is sometimes referred to as a *data table* or *register table*. Part of this memory, i.e. a block of addresses, will be set aside for input and output addresses and the states of those inputs and outputs. Part will be set aside for preset data and part for storing counter values, timer values, etc.
4 Possibly, as a bolt-on extra module, *erasable and programmable read-only-memory (EPROM)* for ROMs that can be programmed and then the program made permanent.

The programs and data in RAM can be changed by the user. All PLCs will have some amount of RAM to store programs that have been developed by the user and program data. However, to prevent the loss of programs when the power supply is switched off, a battery is used in the PLC to maintain the RAM contents for a period of time. After a program

has been developed in RAM it may be loaded into an EPROM memory chip, often a bolt-on module to the PLC, and so made permanent. In addition there are temporary *buffer* stores for the input/output channels.

The storage capacity of a memory unit is determined by the number of binary words that it can store. Thus, if a memory size is 256 words then it can store $256 \times 8 = 2048$ bits if 8-bit words are used and $256 \times 16 = 4096$ bits if 16-bit words are used. Memory sizes are often specified in terms of the number of storage locations available with 1K representing the number 2^{10}, i.e. 1024. Manufacturers supply memory chips with the storage locations grouped in groups of 1, 4 and 8 bits. A $4K \times 1$ memory has $4 \times 1 \times 1024$ bit locations. A $4K \times 8$ memory has $4 \times 8 \times 1024$ bit locations. The term *byte* is used for a word of length 8 bits. Thus the $4K \times 8$ memory can store 4096 bytes. With a 16-bit address bus we can have 2^{16} different addresses and so, with 8-bit words stored at each address, we can have $2^{16} \times 8$ storage locations and so use a memory of size $2^{16} \times 8/2^{10} = 64K \times 8$ which we might be as four $16K \times 8$ bit memory chips.

1.3.4 Input/output unit

The input/output unit provides the interface between the system and the outside world, allowing for connections to be made through input/output channels to input devices such as sensors and output devices such as motors and solenoids. It is also through the input/output unit that programs are entered from a program panel. Every input/output point has a unique address which can be used by the CPU. It is like a row of houses along a road, number 10 might be the 'house' to be used for an input from a particular sensor while number '45' might be the 'house' to be used for the output to a particular motor.

The input/output channels provide isolation and signal conditioning functions so that sensors and actuators can often be directly connected to them without the need for other circuitry. Electrical isolation from the external world is usually by means of *optoisolators* (the term *optocoupler* is also often used). Figure 1.8 shows the principle of an optoisolator. When a digital pulse passes through the light-emitting diode, a pulse of infrared radiation is produced. This pulse is detected by the phototransistor and gives rise to a voltage in that circuit. The gap between the light-emitting diode and the phototransistor gives electrical isolation but the arrangement still allows for a digital pulse in one circuit to give rise to a digital pulse in another circuit.

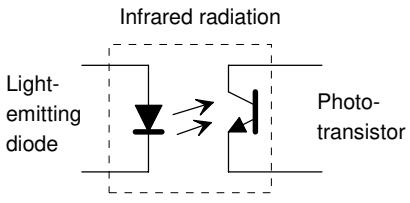


Figure 1.8 *Optoisolator*

The digital signal that is generally compatible with the microprocessor in the PLC is 5 V d.c. However, signal conditioning in the input channel,

with isolation, enables a wide range of input signals to be supplied to it (see Chapter 3 for more details). A range of inputs might be available with a larger PLC, e.g. 5 V, 24 V, 110 V and 240 V digital/discrete, i.e. on–off, signals (Figure 1.9). A small PLC is likely to have just one form of input, e.g. 24 V.

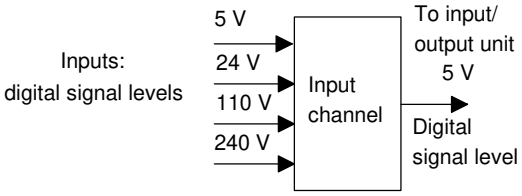


Figure 1.9 *Input levels*

The output from the input/output unit will be digital with a level of 5 V. However, after signal conditioning with relays, transistors or triacs, the output from the output channel might be a 24 V, 100 mA switching signal, a d.c. voltage of 110 V, 1 A or perhaps 240 V, 1 A a.c., or 240 V, 2 A a.c., from a triac output channel (Figure 1.10). With a small PLC, all the outputs might be of one type, e.g. 240 V a.c., 1 A. With modular PLCs, however, a range of outputs can be accommodated by selection of the modules to be used.

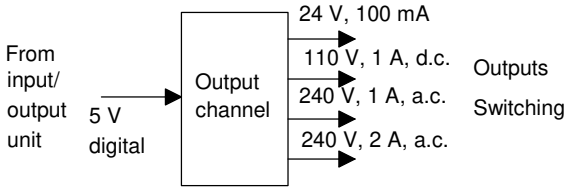


Figure 1.10 *Output levels*

Outputs are specified as being of relay type, transistor type or triac type (see Chapter 3 for more details):

1 With the *relay type*, the signal from the PLC output is used to operate a relay and is able to switch currents of the order of a few amperes in an external circuit. The relay not only allows small currents to switch much larger currents but also isolates the PLC from the external circuit. Relays are, however, relatively slow to operate. Relay outputs are suitable for a.c. and d.c. switching. They can withstand high surge currents and voltage transients.
2 The *transistor type* of output uses a transistor to switch current through the external circuit. This gives a considerably faster switching action. It is, however, strictly for d.c. switching and is destroyed by overcurrent and high reverse voltage. As a protection, either a fuse or built-in electronic protection are used. Optoisolators are used to provide isolation.

3 *Triac* outputs, with optoisolators for isolation, can be used to control external loads which are connected to the a.c. power supply. It is strictly for a.c. operation and is very easily destroyed by overcurrent. Fuses are virtually always included to protect such outputs.

1.3.5 Sourcing and sinking

The terms *sourcing* and *sinking* are used to describe the way in which d.c. devices are connected to a PLC. With sourcing, using the conventional current flow direction as from positive to negative, an input device receives current from the input module, i.e. the input module is the source of the current (Figure 1.11(a)). If the current flows from the output module to an output load then the output module is referred to as sourcing (Figure 1.11(b)). With sinking, using the conventional current flow direction as from positive to negative, an input device supplies current to the input module, i.e. the input module is the sink for the current (Figure 1.12(a)). If the current flows to the output module from an output load then the output module is referred to as sinking (Figure 1.12(b)).

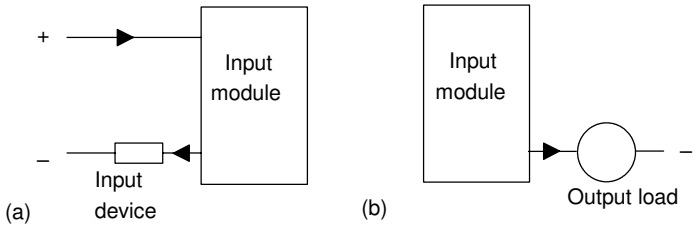


Figure 1.11 *Sourcing*

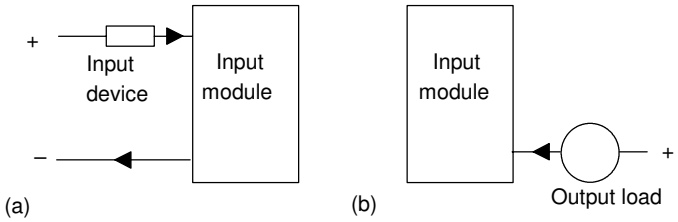


Figure 1.12 *Sinking*

1.4 PLC systems

There are two common types of mechanical design for PLC systems; a *single box*, and the *modular/rack types*. The single box type (or, as sometimes termed, brick) is commonly used for small programmable controllers and is supplied as an integral compact package complete with power supply, processor, memory, and input/output units. Typically such a PLC might have 6, 8, 12 or 24 inputs and 4, 8 or 16 outputs and a memory which can store some 300 to 1000 instructions. Figure 1.13 shows the Mitsubishi MELSEC FX3U compact, i.e. brick, PLC and Table 1.1 gives details of models in that Mitsubishi range.

Figure 1.13 *Mitsubishi Compact PLC – MELSEC FX3U (By permission of Mitsubishi Electric Europe)*

Table 1.1 *Mitsubishi Compact PLC – MELSEC FX3U Product range (By permission of Mitsubishi Electric Europe)*

Type	FX3U-16 MR	FX3U-32 MR	FX3U-48 MR	FX3U-64 MR	FX3U-80 MR
Power supply	100-240 V AC				
Inputs	8	16	24	32	40
Outputs	8	16	24	32	40
Digital outputs	Relay				
Program cycle period per logical instruction	0.065 μs				
User memory	64k steps (standard), FLROM cassettes (optional)				
Dimensions in mm (W × H × D)	130 × 90 × 86	150 × 140 × 86	182 × 90 × 86	220 × 90 × 86	285 × 90 × 86

Some brick systems have the capacity to be extended to cope with more inputs and outputs by linking input/output boxes to them. Figure 1.14 shows such an arrangement with the OMRON CPM1A PLC. The base input/output brick, depending on the model concerned, has 10, 20, 30 or 40 inputs/outputs (I/O). The 10 I/O brick has 6 d.c. input points and four outputs, the 20 I/O brick has 12 d.c. input points and 8 outputs, the 30 I/O brick has 18 d.c. input points and 12 outputs and the 40 I/O brick has 24 d.c. input points and 16 outputs. However, the 30 and 40 I/O models can be extended to a maximum of 100 inputs/outputs by linking expansion units to the original brick. For example a 20 I/O expansion module might be added, it having 12 inputs and 8 outputs, the outputs being relays, sinking transistors or sourcing transistors. Up to three expansion modules can be added. The outputs can be relay or transistor outputs.

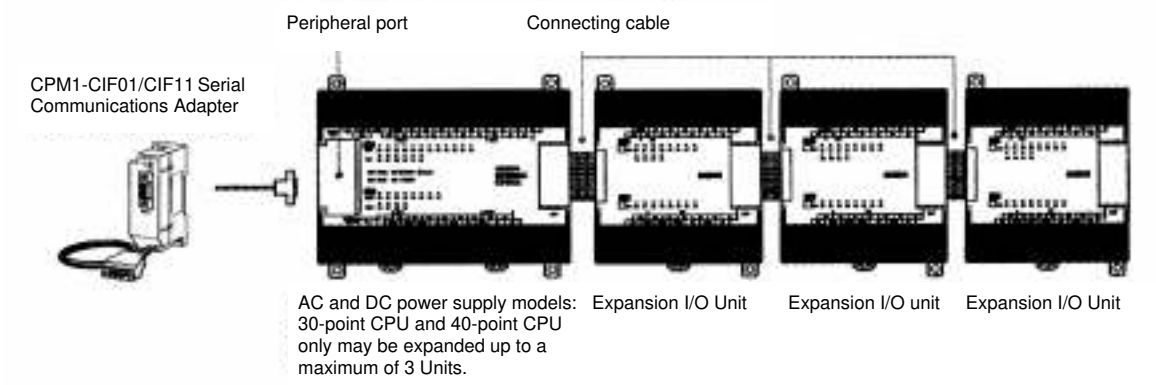


Figure 1.14 *Basic configuration of the OMRON CPM1A PLC (By permission of Omron Electronics LLC)*

Systems with larger numbers of inputs and outputs are likely to be modular and designed to fit in racks. The modular type consists of separate modules for power supply, processor, etc., which are often mounted on rails within a metal cabinet. The rack type can be used for all sizes of programmable controllers and has the various functional units packaged in individual modules which can be plugged into sockets in a base rack. The mix of modules required for a particular purpose is decided by the user and the appropriate ones then plugged into the rack. Thus it is comparatively easy to expand the number of input/output (I/O) connections by just adding more input/output modules or to expand the memory by adding more memory units.

An example of such a modular system is provided by the Allen-Bradley PLC-5 PLC of Rockwell automation (Figure 1.15). PLC-5 processors are available in a range of I/O capacity and memory size, and can be configured for a variety of communication networks. They are single-slot modules that are placed in the left-most slot of a 1771 I/O chassis. Some 1771 I/O chassis are built for back-panel mounting and some are built for rack mounting and are available in sizes of 4, 8, 12, or 16 I/O module slots. The 1771 I/O modules are available in densities of 8, 16, or 32 I/O per module. A PLC-5 processor can communicate with I/O across a DeviceNet or Universal Remote I/O link.

A large selection of 1771 input/output modules, both digital and analogue, are available for use in the local chassis, and an even larger selection available for use at locations remote from the processor. Digital I/O modules have digital I/O circuits that interface to on/off sensors such as pushbutton and limit switches; and on/off actuators such as motor starters, pilot lights, and annunciators. Analogue I/O modules perform the required A/D and D/A conversions using up to 16-bit resolution. Analogue I/O can be user-configured for the desired fault-response state in the event that I/O communication is disrupted. This feature provides a safe reaction/response in case of a fault, limits the extent of faults, and provides a predictable fault response. 1771 I/O modules include optical coupling and filter circuitry for signal noise reduction.

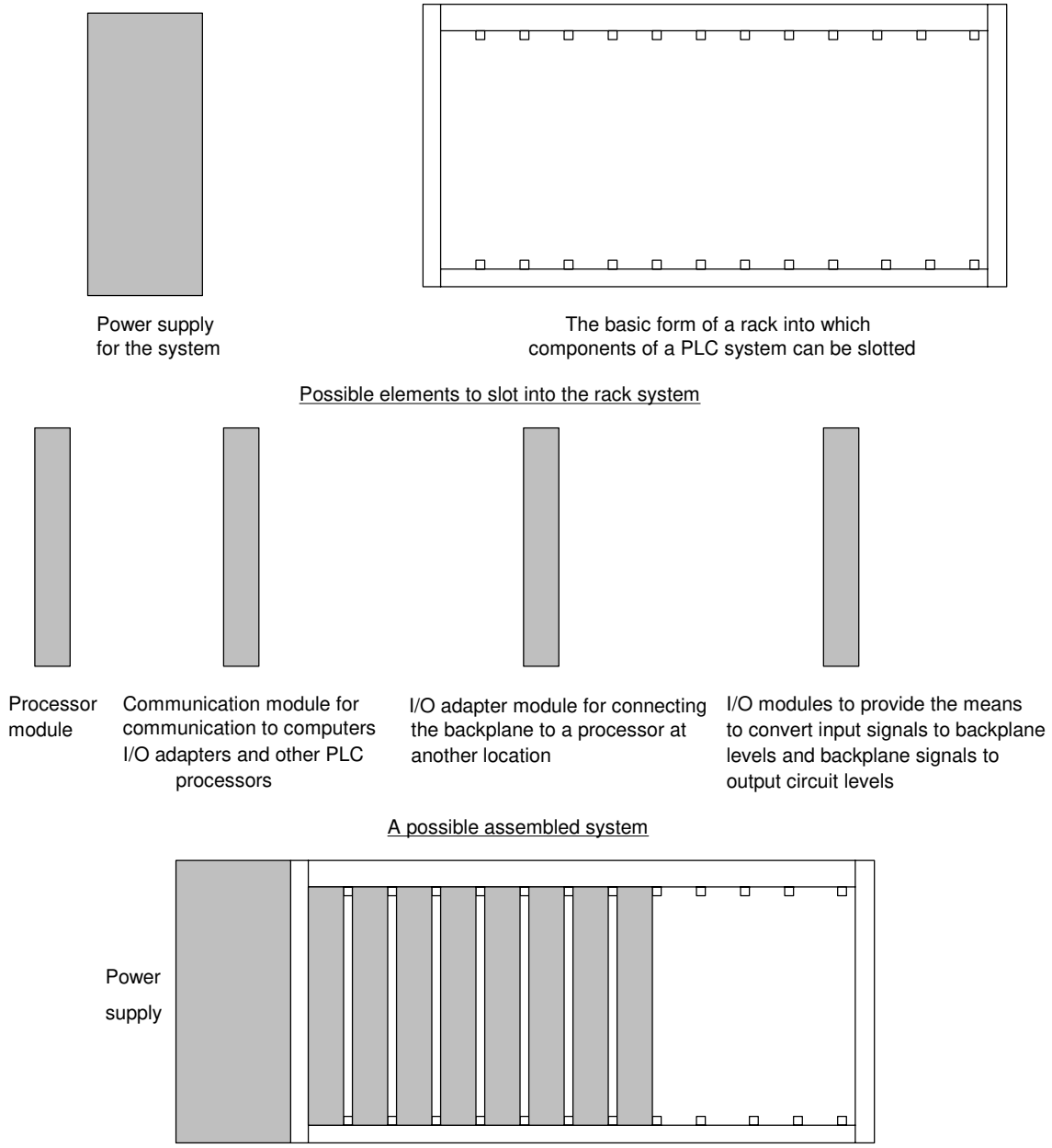


Figure 1.15 *A possible arrangement of a rack system, e.g. the Rockwell Automation , Allen-Bradley PLC-5*

Digital I/O modules cover electrical ranges from 5…276V a.c. or d.c. and relay contact output modules are available for ranges from 0…276 V ac or 0…175 V dc. A range of analogue signal levels can be accomodated, including standard analogue inputs and outputs and direct thermocouple and RTD temperature inputs.

1.4.1 Programming PLCs

Programming devices can be a hand-held device, a desktop console or a computer. Only when the program has been designed on the programming device and is ready is it transferred to the memory unit of the PLC.

1 *Hand-held programming devices* will normally contain enough memory to allow the unit to retain programs while being carried from one place to another.
2 *Desktop consoles* are likely to have a visual display unit with a full keyboard and screen display.
3 *Personal computers* are widely configured as program development work-stations. Some PLCs only require the computer to have appropriate software; others require special communication cards to interface with the PLC. A major advantage of using a computer is that the program can be stored on the hard disk or a CD and copies easily made.

PLC manufacturers have programming software for their PLCs. For example, Mitsubishi have *MELSOFT*. Their GX Developer supports all MELSEC controllers from the compact PLCs of the MELSEC FX series to the modular PLCs including MELSEC System Q and uses a Windows based environment. It supports the programming methods (see Chapter 4) of instruction list (IL), ladder diagram (LD) and sequential function chart (SFC) languages. You can switch back and forth between IL and LD at will while you are working. You can program your own function blocks and a wide range of utilities are available for configuring special function modules for the MELSEC System Q – there is no need to program special function modules, you just configure them. The package includes powerful editors and diagnostics functions for configuring MELSEC networks and hardware, and extensive testing and monitoring functions to help get applications up and running quickly and efficiently. It offers off-line simulation for all PLC types and thus enables simulation of all devices and application responses for realistic testing.

As another illustration, Siemens have *SIMATIC STEP 7*. This fully complies with the international standard IEC 61131-3 for PLC programming languages. With STEP 7, programmers can select between different programming languages. Besides ladder diagram (LAD) and function block diagram (FBD), STEP 7 Basis also includes the Instruction List (STL) programming language. Other additional options are available for IEC 61131-3 programming languages such as Structured Text (ST) called SIMATIC S7-SCL or a Sequential Function Chart (SFC) called SIMATIC S7-Graph which provides an efficient way to describe sequential control systems graphically. Features of the whole engineering system include system diagnostic capabilities, process diagnostic tools, PLC simulation, remote maintenance, and plant documentation. S7-PLCSIM is an optional package for STEP 7 that allows simulation of a SIMATIC S7 control platform and testing of a user program on a PC, enabling testing and refining prior to physical hardware installation. By testing early in a project's development, overall project quality can be improved. Installation and commissioning can thus be quicker and less

expensive as program faults can be detected and corrected early on during development.

Likewise, Rockell Automation have *RSLogix* for the Allen-Bradley PLC-5 family of PLCs, OMRON has CX-One and Telemecanique have ProWorx 32 for its Modicon range of PLCs.

2 Input–output devices

This chapter is a brief consideration of typical input and output devices used with PLCs. The input devices considered include digital and analogue devices such as mechanical switches for position detection, proximity switches, photoelectric switches, encoders, temperature and pressure switches, potentiometers, linear variable differential transformers, strain gauges, thermistors, thermotransistors and thermocouples. Output devices considered include relays, contactors, solenoid valves and motors.

2.1 Input devices

The term *sensor* is used for an input device that provides a usable output in response to a specified physical input. For example, a thermocouple is a sensor which converts a temperature difference into an electrical output. The term *transducer* is generally used for a device that converts a signal from one form to a different physical form. Thus sensors are often transducers, but also other devices can be transducers, e.g. a motor which converts an electrical input into rotation.

Sensors which give digital/discrete, i.e. on–off, outputs can be easily connected to the input ports of PLCs. Sensors which give analogue signals have to be converted to digital signals before inputting them to PLC ports. The following are some of the more common terms used to define the performance of sensors.

1 *Accuracy* is the extent to which the value indicated by a measurement system or element might be wrong. For example, a temperature sensor might have an accuracy of ±0.1°C. The *error* of a measurement is the difference between the result of the measurement and the true value of the quantity being measured errors can arise in a number of ways, e.g. the term *non-linearity error* is used for the error that occurs as a result of assuming a linear relationship between the input and output over the working range, i.e. a graph of output plotted against input is assumed to give a straight line. Few systems or elements, however, have a truly linear relationship and thus errors occur as a result of the assumption of linearity (Figure 2.1(a)). The term *hysteresis error* (Figure 2.1(b)) is used for the difference in outputs given from the same value of quantity being measured according to whether that value has been reached by a continuously increasing change or a continuously decreasing change. Thus, you might obtain a different value from a thermometer used to measure the same temperature of a liquid if it is reached by the liquid warming up to the measured temperature or it is reached by the liquid cooling down to the measured temperature.

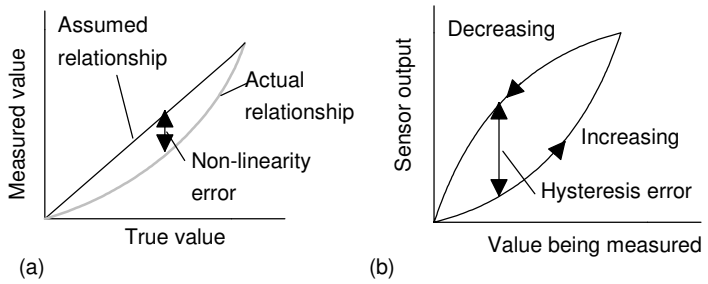


Figure 2.1 *Some sources of error: (a) non-linearity, (b) hysteresis*

2 The *range* of variable of system is the limits between which the input can vary. For example, a resistance temperature sensor might be quoted as having a range of −200 to +800ºC.

3 When the input value to a sensor changes, it will take some time to reach and settle down to the steady-state value (Figure 2.2). The *response time* is the time which elapses after the input to a system or element is abruptly increased from zero to a constant value up to the point at which the system or element gives an output corresponding to some specified percentage, e.g. 95%, of the value of the input. The *rise time* is the time taken for the output to rise to some specified percentage of the steady-state output. Often the rise time refers to the time taken for the output to rise from 10% of the steady-state value to 90 or 95% of the steady-state value. The *settling time* is the time taken for the output to settle to within some percentage, e.g. 2%, of the steady-state value.

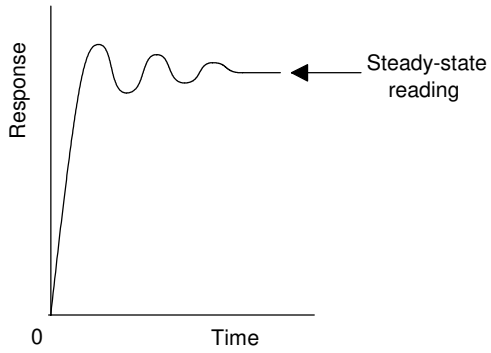


Figure 2.2 *Response of a sensor or measurement system to a sudden input. You can easily see such a response when the current in an electrical circuit is suddenly switched on and an ammeter reading observed.*

4 The *sensitivity* indicates how much the output of an instrument system or system element changes when the quantity being measured changes by a given amount, i.e. the ratio ouput/input. For example, a thermocouple might have a sensitivity of 20 μV/ºC and so give an output of 20 μV for each 1ºC change in temperature.

5 The *stability* of a system is its ability to give the same output when used to measure a constant input over a period of time. The term *drift* is often used to describe the change in output that occurs over time. The drift may be expressed as a percentage of the full range output. The term *zero drift* is used for the changes that occur in output when there is zero input.

6 The term *repeatability* is used for the ability of a measurement system to give the same value for repeated measurements of the same value of a variable. Common cause of lack of repeatability are random fluctuations in the environment, e.g. changes in temperature and humidity. The error arising from repeatability is usually expressed as a percentage of the full range output. For example, a pressure sensor might be quoted as having a repeatability of ±0.1% of full range. Thus with a range of 20 kPa this would be an error of ±20 Pa.

7 The *reliability* of a measurement system, or element in such a system, is defined as being the probability that it will operate to an agreed level of performance, for a specified period, subject to specified environmental conditions. The agreed level of performance might be that the measurement system gives a particular accuracy.

The following are examples of some of the commonly used PLC input devices and their sensors.

2.1.1 Mechanical switches

A mechanical switch generates an on–off signal or signals as a result of some mechanical input causing the switch to open or close. Such a switch might be used to indicate the presence of a workpiece on a machining table, the workpiece pressing against the switch and so closing it. The absence of the workpiece is indicated by the switch being open and its presence by it being closed. Thus, with the arrangement shown in Figure 2.3(a), the input signals to a single input channel of the PLC are thus the logic levels:

Workpiece not present 0
Workpiece present 1

The 1 level might correspond to a 24 V d.c. input, the 0 to a 0 V input.

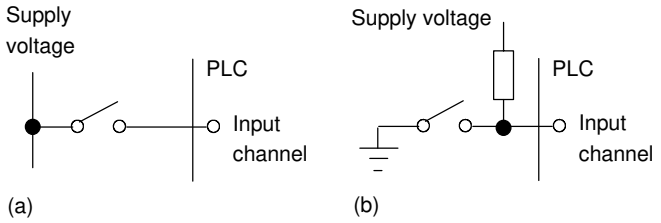


Figure 2.3 *Switch sensors*

With the arrangement shown in Figure 2.3(b), when the switch is open the supply voltage is applied to the PLC input, when the switch is closed the input voltage drops to a low value. The logic levels are thus:

Workpiece not present 1
Workpiece present 0

Switches are available with *normally open (NO)* or *normally closed (NC)* contacts or can be configured as either by choice of the relevant contacts. An NO switch has its contacts open in the absence of a mechanical input and the mechanical input is used to close the switch. An NC switch has its contacts closed in the absence of a mechanical input and the mechanical input is used to open the switch.

The term *limit switch* is used for a switch which is used to detect the presence or passage of a moving part. It can be actuated by a cam, roller or lever. Figure 2.4 shows some examples. The cam (Figure 2.4(c)) can be rotated at a constant rate and so switch the switch on and off for particular time intervals.

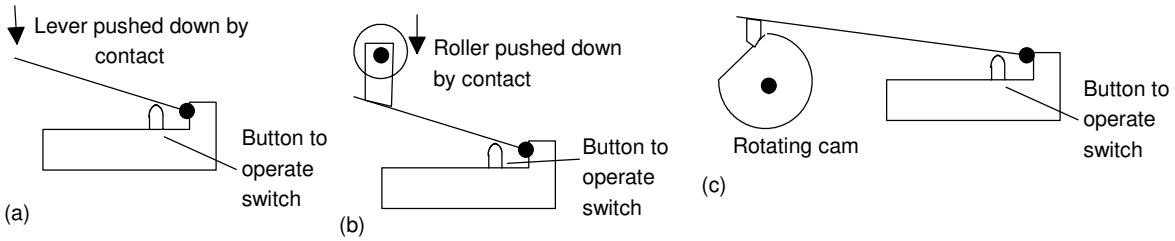


Figure 2.4 *Limit switches actuated by: (a) lever, (b) roller, (c) cam*

2.1.2 Proximity switches

Proximity switches are used to detect the presence of an item without making contact with it. There are a number of forms of such switches, some being only suitable for metallic objects.

The *eddy current* type of proximity switch has a coil which is energised by a constant alternating current and produces a constant alternating magnetic field. When a metallic object is close to it, eddy currents are induced in it (Figure 2.5(a)). The magnetic field due to these eddy currents induces an e.m.f. back in the coil with the result that the voltage amplitude needed to maintain the constant coil current changes. The voltage amplitude is thus a measure of the proximity of metallic objects. The voltage can be used to activate an electronic switch circuit, basically a transistor which has its output switched from low to high by the voltage change, and so give an on–off device. The range over which such objects can be detected is typically about 0.5 to 20 mm.

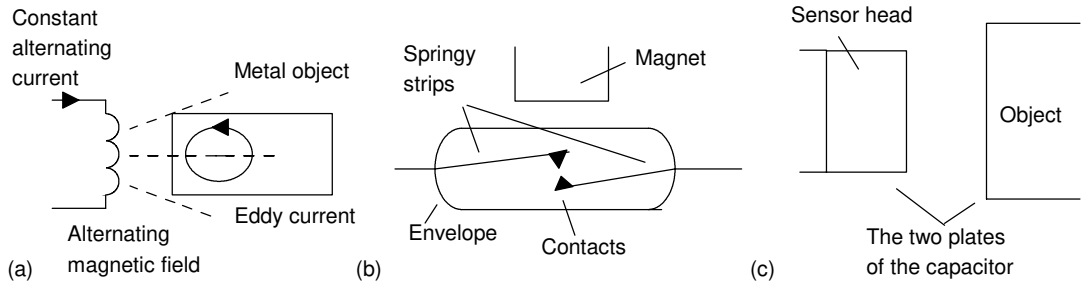


Figure 2.5 *Proximity switches: (a) eddy current, (b) reed switch*, (c) *capacitive*

Another type is the *reed switch*. This consists of two overlapping, but not touching, strips of a springy ferromagnetic material sealed in a glass or plastic envelope (Figure 2.5(b)). When a magnet or current-carrying coil is brought close to the switch, the strips become magnetised and attract each other. The contacts then close. The magnet closes the contacts when it is typically about 1 mm from the switch. Such a switch is widely used with burglar alarms to detect when a door is opened; the magnet being in the door and the reed switch in the frame of the door. When the door opens the switch opens.

A proximity switch that can be used with metallic and non-metallic objects is the *capacitive proximity switch*. The capacitance of a pair of plates separated by some distance depends on the separation, the smaller the separation the higher the capacitance. The sensor of the capacitive proximity switch is just one of the plates of the capacitor, the other plate being the metal object whose proximity is to be detected (Figure 2.5(c)). Thus the proximity of the object is detected by a change in capacitance. The sensor can also be used to detect non-metallic objects since the capacitance of a capacitor depends on the dielectric between its plates. In this case the plates are the sensor and the earth and the non-metallic object is the dielectric. The change in capacitance can be used to activate an electronic switch circuit and so give an on–off device. Capacitive proximity switches can be used to detect objects when they are typically between 4 and 60 mm from the sensor head.

Another type, the *inductive proximity switch*, consists of a coil wound round a ferrous metallic core. When one end of this core is placed near to a ferrous metal object there is effectively a change in the amount of metallic core associated with the coil and so a change in its inductance. This change in inductance can be monitored using a resonant circuit, the presence of the ferrous metal object thus changing the current in that circuit. The current can be used to activate an electronic switch circuit and so give an on–off device. The range over which such objects can be detected is typically about 2 to 15 mm.

2.1.3 Photoelectric sensors and switches

Photoelectric switch devices can either operate as *transmissive types* where the object being detected breaks a beam of light, usually infrared radiation, and stops it reaching the detector (Figure 2.6(a)) or *reflective types* where the object being detected reflects a beam of light onto the detector (Figure 2.6(b)). In both types the radiation emitter is generally a *light-emitting diode (LED)*. The radiation detector might be a *photo-transistor*, often a pair of transistors, known as a *Darlington pair*. The Darlington pair increases the sensitivity. Depending on the circuit used, the output can be made to switch to either high or low when light strikes the transistor. Such sensors are supplied as packages for sensing the presence of objects at close range, typically at less than about 5 mm. Figure 2.6(c) shows a U-shaped form where the object breaks the light beam.

Another possibility is a *photodiod*e. Depending on the circuit used, the output can be made to switch to either high or low when light strikes the diode. Yet another possibility is a *photoconductive cell*. The resistance of

the photoconductive cell, often cadmium sulphide, depends on the intensity of the light falling on it.

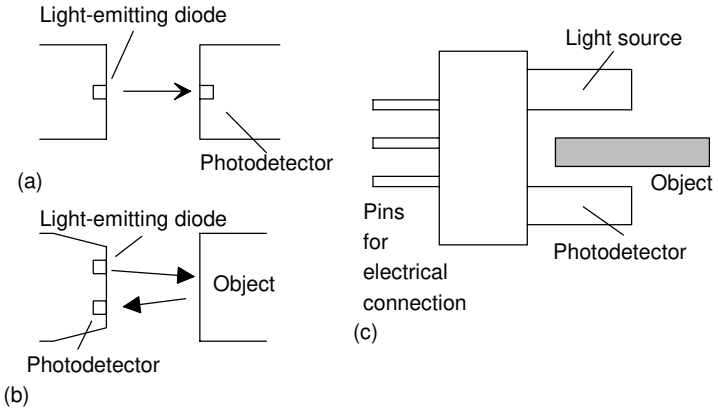


Figure 2.6 *Photoelectric sensors*

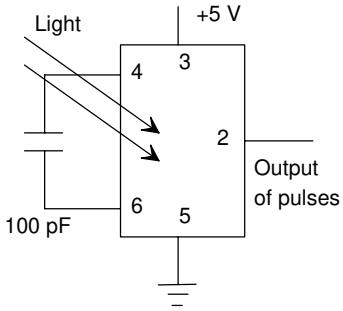


Figure 2.7 *TSL220*

With the above sensors, light is converted to a current, voltage or resistance change. If the output is to be used as a measure of the intensity of the light, rather than just the presence or absence of some object in the light path, the signal will need amplification and then conversion from analogue to digital by an analogue-to-digital converter. An alternative to this is to use a light-to-frequency converter, the light then being converted to a sequence of pulses with the frequency of the pulses being a measure of the light intensity. Integrated circuit sensors are available, e.g. the Texas Instrument TSL220, incorporating the light sensor and the voltage-to-frequency converter (Figure 2.7).

2.1.4 Encoders

The term *encoder* is used for a device that provides a digital output as a result of angular or linear displacement. An *increment encoder* detects changes in angular or linear displacement from some datum position, while an *absolute encoder* gives the actual angular or linear position.

Figure 2.8 shows the basic form of an *incremental encoder* for the measurement of angular displacement. A beam of light, from perhaps a light-emitting diode (LED), passes through slots in a disc and is detected by a light sensor, e.g. a photodiode or phototransistor. When the disc rotates, the light beam is alternately transmitted and stopped and so a pulsed output is produced from the light sensor. The number of pulses is proportional to the angle through which the disc has rotated, the resolution being proportional to the number of slots on a disc. With 60 slots then, since one revolution is a rotation of 360°, a movement from one slot to the next is a rotation of 6°. By using offset slots it is possible to have over a thousand slots for one revolution and so much higher resolution.

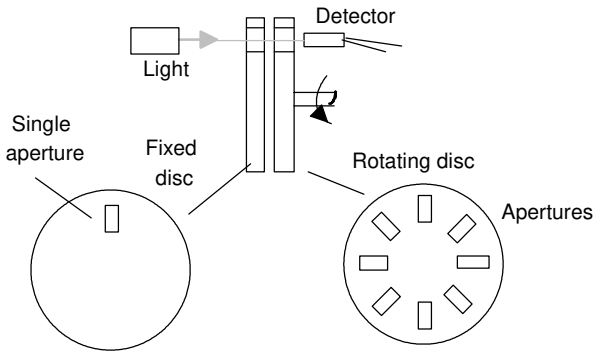


Figure 2.8 *Basic form of an incremental encoder*

The *absolute encoder* differs from the incremental encoder in having a pattern of slots which uniquely defines each angular position. With the form shown in Figure 2.9, the rotating disc has four concentric circles of slots and four sensors to detect the light pulses. The slots are arranged in such a way that the sequential output from the sensors is a number in the binary code, each such number corresponding to a particular angular position. With 4 tracks there will be 4 bits and so the number of positions that can be detected is $2^4 = 16$, i.e. a resolution of $360/16 = 22.5^o$. Typical encoders tend to have up to 10 or 12 tracks. The number of bits in the binary number will be equal to the number of tracks. Thus with 10 tracks there will be 10 bits and so the number of positions that can be detected is 2^{10}, i.e. 1024, a resolution of $360/1024 = 0.35^o$.

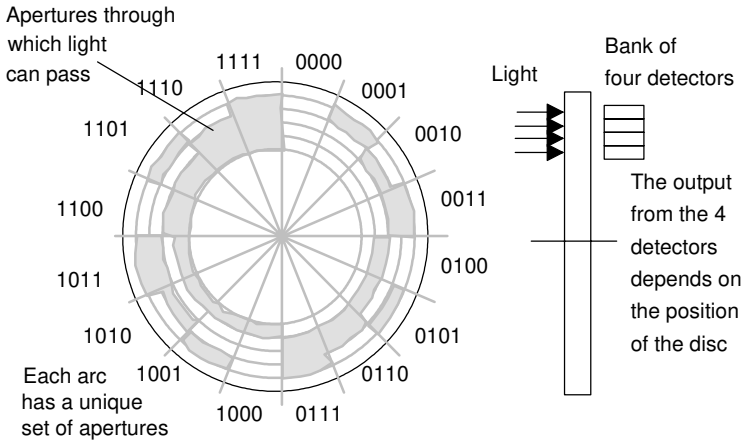


Figure 2.9 *The rotating wheel of the absolute encoder. Note that though the normal form of binary code is shown in the figure, in practice a modified form of binary code called the Gray code is generally used. This code, unlike normal binary, has only one bit changing in moving from one number to the next. Thus we have the sequence 0000, 0001, 0011, 0010, 0011, 0111, 0101, 0100, 1100, 1101, 1111.*

2.1.5 Temperature sensors

A simple form of temperature sensor which can be used to provide an on–off signal when a particular temperature is reached is the *bimetal element*. This consists of two strips of different metals, e.g. brass and iron, bonded together (Figure 2.10). The two metals have different coefficients of expansion. Thus when the temperature of the bimetal strip increases the strip curves, in order that one of the metals can expand more than the other. The higher expansion metal is on the outside of the curve. As the strip cools, the bending effect is reversed. This movement of the strip can be used to make or break electrical contacts and hence, at some particular temperature, give an on–off current in an electrical circuit. The device is not very accurate but is commonly used in domestic central heating thermostats.

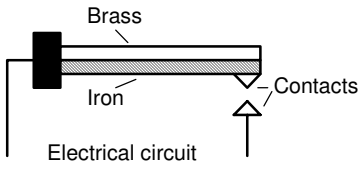


Figure 2.10 *Bimetallic strip*

Another form of temperature sensor is the *resistive temperature detector (RTD)*. The electrical resistance of metals or semiconductors changes with temperature. In the case of a metal, the ones most commonly used are platinum, nickel or nickel alloys, the resistance of which varies in a linear manner with temperature over a wide range of temperatures, though the actual change in resistance per degree is fairly small. Semiconductors, such as thermistors, show very large changes in resistance with temperature. The change, however, is non-linear. Such detectors can be used as one arm of a Wheatstone bridge and the output of the bridge taken as a measure of the temperature (Figure 2.11(a)). Another possibility is to use a potential divider circuit with the change in resistance of the thermistor changing the voltage drop across a resistor (Figure 2.11(b)). The output from either type of circuit is an analogue signal which is a measure of the temperature.

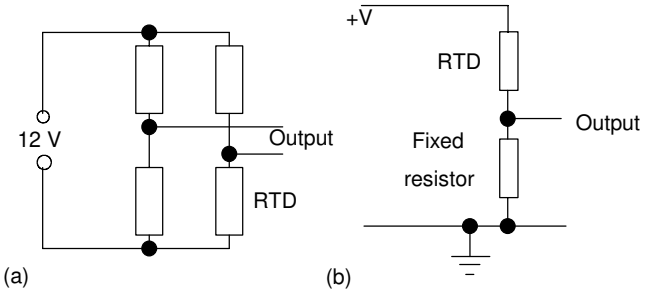


Figure 2.11 *(a) Wheatstone bridge, (b) potential divider circuits*

Thermodiodes and *thermotransistors* are used as temperature sensors since the rate at which electrons and holes diffuse across semiconductor

junctions is affected by the temperature. Integrated circuits are available which combine such a temperature-sensitive element with the relevant circuitry to give an output voltage related to temperature. A widely used integrated package is the LM35 which gives an output of 10 mV/°C when the supply voltage is +5 V (Figure 2.12(a)).

A digital temperature switch can be produced with an analogue sensor by feeding the analogue output into a comparator amplifier which compares it with some set value, producing an output giving a logic 1 signal when the temperature voltage input is equal to or greater than the set point and otherwise an output which gives a logic 0 signal. Integrated circuits, e.g. LM3911N, are available, combining a thermotransistor temperature-sensitive element with an operational amplifier. When the connections to the chip are so made that the amplifier is connected as a comparator (Figure 2.12(b)), then the output will switch as the temperature traverses the set point and so directly give an on–off temperature controller.

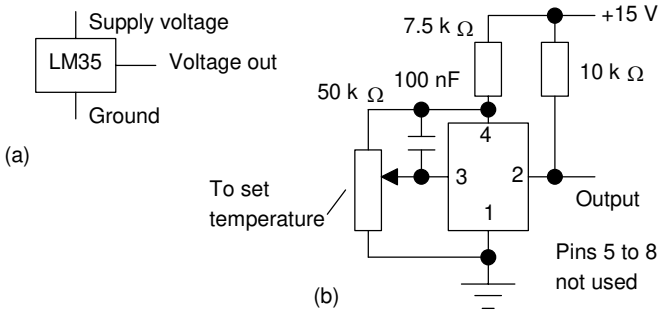


Figure 2.12 *(a) LM35, (b) LM3911N circuit for on–off control*

Another commonly used temperature sensor is the *thermocouple*. The thermocouple consists essentially of two dissimilar wires A and B forming a junction (Figure 2.13). When the junction is heated so that it is at a higher temperature than the other junctions in the circuit, which remain at a constant cold temperature, an e.m.f. is produced which is related to the hot junction temperature. The voltage produced by a thermocouple is small and needs amplification before it can be fed to the analogue channel input of a PLC. There is also circuitry required to compensate for the temperature of the cold junction since its temperature affects the value of the e.m.f. given by the hot junction. The amplification and compensation, together with filters to reduce the effect of interference from the 50 Hz mains supply, are often combined in a signal processing unit.

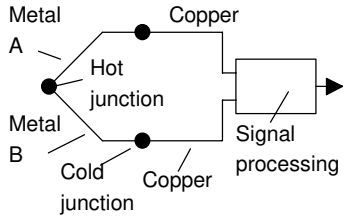


Figure 2.13 *Thermocouple*

2.1.6 Position/displacement sensors

The term *position sensor* is used for a sensor that gives a measure of the distance between a reference point and the current location of the target, a *displacement sensor* being one that gives a measure of the distance between the present position of the target and the previously recorded position.

Resistive linear and angular position sensors are widely used and relatively inexpensive. These are also called *linear and rotary potentiometers*. A d.c. voltage is provided across the full length of the track and the voltage signal between a contact which slides over the resistance track and one end of the track is related to the position of the sliding contact between the ends of the potentiometer resistance track (Figure 2.14). The potentiometer thus provides an analogue linear or angular position sensor.

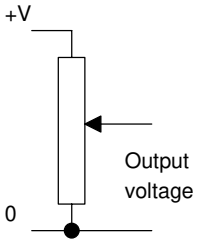


Figure 2.14 *Potentiometer*

Another form of displacement sensor is the *linear variable differential transformer (LVDT)*, this giving a voltage output related to the position of a ferrous rod. The LVDT consists of three symmetrically placed coils through which the ferrous rod moves (Figure 2.15).

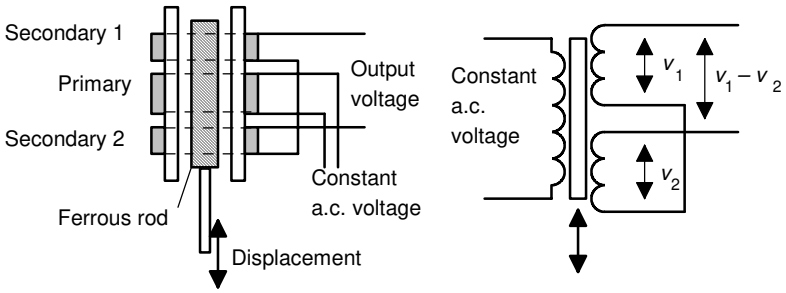


Figure 2.15 *LVDT*

When an alternating current is applied to the primary coil, alternating voltages, v_1 and v_2, are induced in the two secondary coils. When the ferrous rod core is centred between the two secondary coils, the voltages induced in them are equal. The outputs from the two secondary coils are connected so that their combined output is the difference between the two voltages, i.e. $v_1 - v_2$. With the rod central, the two alternating voltages are equal and so there is no output voltage. When the rod is displaced from its central position there is more of the rod in one secondary coil than the other. As a result the size of the alternating voltage induced in one coil is greater than that in the other. The difference between the two secondary coil voltages, i.e. the output, thus depends on the position of the ferrous rod. The output from the LVDT is an alternating voltage. This is usually converted to an analogue d.c. voltage and amplified before inputting to the analogue channel of a PLC.

Capacitive displacement sensors are essentially just parallel plate capacitors. The capacitance will change if the plate separation changes, the area of overlap of the plates changes, or a slab of dielectric is moved into or out of the plates (Figure 2.16). All these methods can be used to give linear displacement sensors. The change in capacitance has to be converted into a suitable electrical signal by signal conditioning.

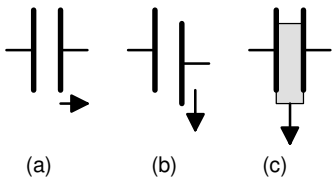


Figure 2.16 *Capacitor sensors: (a) changing the plate separation, (b) changing the area of overlap, (c) moving the dielectric*

2.1.7 Strain gauges

When a wire or strip of semiconductor is stretched, its resistance changes. The fractional change in resistance is proportional to the fractional change in length, i.e. strain.

$$\frac{\Delta R}{R} = G \times \text{strain}$$

where ΔR is the change in resistance for a wire of resistance R and G is a constant called the *gauge factor*. For metals the gauge factor is about 2 and for semiconductors about 100. Metal resistance strain gauges are in the form of a flat coil in order to get a reasonable length of metal in a small area. Often they are etched from metal foil (Figure 2.17(a)) and attached to a backing of thin plastic film so that they can be stuck on surfaces, like postage stamps on an envelope. The change in resistance of the strain gauge, when subject to strain, is usually converted into a voltage signal by the use of a Wheatstone bridge (Figure 2.17(b)). A problem that occurs is that the resistance of the strain gauge also changes with temperature and thus some means of temperature compensation has to be used so that the output of the bridge is only a function of the strain. This can be achieved by placing a dummy strain gauge in an opposite arm of the bridge, that gauge not being subject to any strain but only the temperature (Figure 2.18).

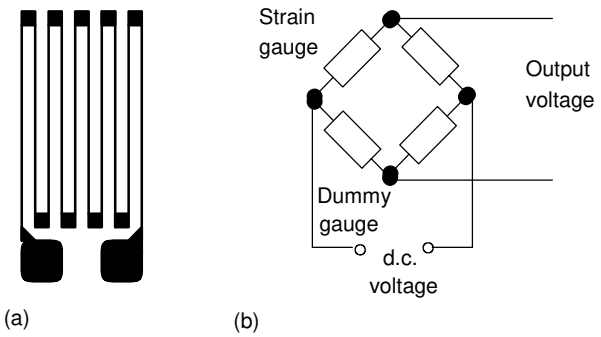


Figure 2.17 *(a) Metal foil strain gauge, (b) Wheatstone bridge circuit with compensation for temperature changes*

An alternative which is widely used is to use four active gauges as the arms of the bridge and arrange it so that one pair of opposite gauges are in tension and the other pair in compression. This not only gives temperature compensation but also gives a much larger output change when strain is applied. The following paragraph illustrates systems employing such a form of compensation.

By attaching strain gauges to other devices, changes which result in strain of those devices can be transformed, by the strain gauges, to give voltage changes. They might, for example, be attached to a cantilever to which forces are applied at its free end (Figure 2.18(a)). The voltage change, resulting from the strain gauges and the Wheatstone bridge, then becomes a measure of the force. Another possibility is to attach strain

gauges to a diaphragm which deforms as a result of pressure (Figure 2.18(b)). The output from the gauges, and associated Wheatstone bridge, then becomes a measure of the pressure.

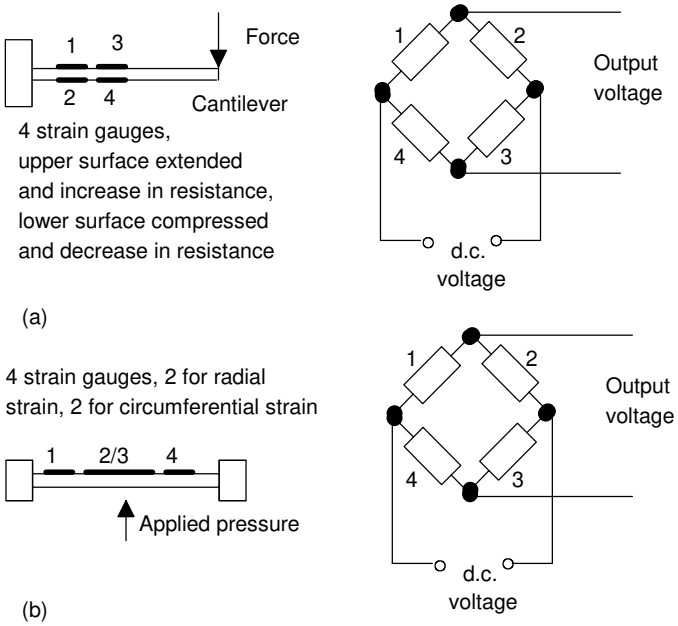


Figure 2.18 *Strain gauges used for (a) force sensor, (b) pressure sensor*

2.1.8 Pressure sensors

Commonly used pressure sensors which give responses related to the pressure are diaphragm and bellows types. The diaphragm type consists of a thin disc of metal or plastic, secured round its edges. When there is a pressure difference between the two sides of the diaphragm, the centre of it deflects. The amount of deflection is related to the pressure difference. This deflection may be detected by strain gauges attached to the diaphragm (see Figure 2.18(b)), by a change in capacitance between it and a parallel fixed plate or by using the deflection to squeeze a piezoelectric crystal (Figure 2.19(a)). When a piezoelectric crystal is squeezed, there is a relative displacement of positive and negative charges within the crystal and the outer surfaces of the crystal become charged. Hence a potential difference appears across it. An example of such a sensor is the Motorola MPX100AP sensor (Figure 2.19(b)). This has a built-in vacuum on one side of the diaphragm and so the deflection of the diaphragm gives a measure of the absolute pressure applied to the other side of the diaphragm. The output is a voltage which is proportional to the applied pressure with a sensitivity of 0.6 mV/kPa. Other versions are available which have one side of the diaphragm open to the atmosphere and so can be used to measure gauge pressure; others allow pressures to be applied to both sides of the diaphragm and so can be used to measure differential pressures.

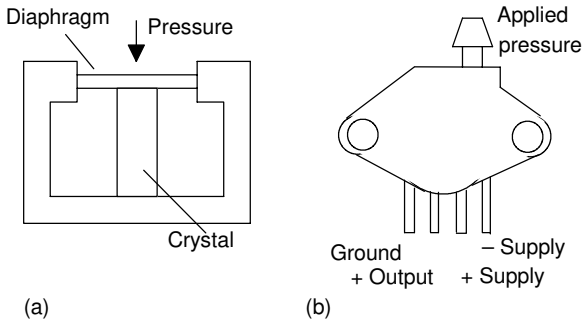


Figure 2.19 *(a) Piezoelectric pressure sensor, (b) MPX100AP*

Pressure switches are designed to switch on or off at a particular pressure. A typical form involves a diaphragm or bellows which moves under the action of the pressure and operates a mechanical switch. Figure 2.20 shows two possible forms. Diaphragms are less sensitive than bellows but can withstand greater pressures.

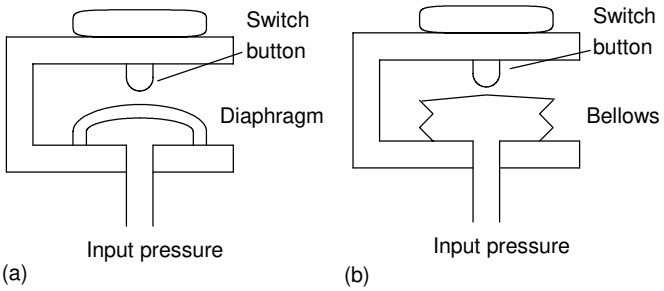


Figure 2.20 *Examples of pressure switches*

2.1.9 Liquid level detector

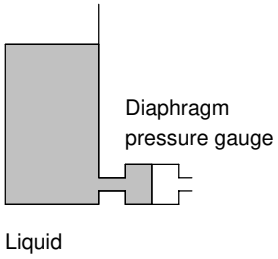


Figure 2.21 *Liquid level sensor*

Pressure sensors may be used to monitor the depth of a liquid in a tank. The pressure due to a height of liquid h above some level is $h\rho g$, where ρ is the density of the liquid and g the acceleration due to gravity. Thus a commonly used method of determining the level of liquid in a tank is to measure the pressure due to the liquid above some datum level (Figure 2.21).

Often a sensor is just required to give a signal when the level in some container reaches a particular level. A float switch that is used for this purpose consists of a float containing a magnet which moves in a housing with a reed switch. As the float rises or falls it turns the reed switch on or off, the reed switch being connected in a circuit which then switches on or off a voltage.

2.1.10 Fluid flow measurement

A common form of fluid flow meter is that based on measuring the difference in pressure resulting when a fluid flows through a constriction. Figure 2.22 shows a commonly used form, the *orifice flow meter*. As a result of the fluid flowing through the orifice, the pressure at A is higher

than that at B, the difference in pressure being a measure of the rate of flow. This pressure difference can be monitored by means of a diaphragm pressure gauge and thus becomes a measure of the rate of flow.

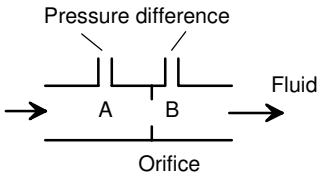


Figure 2.22 *Orifice flow meter*

2.1.11 Smart sensors

The term *smart sensor* is used for a sensor which is integrated with the required buffering and conditioning circuitry in a single element. The circuitry with the element usually consists of data converters, a processor and firmware, and some form of non-volatile EEPROM memory (electrically erasable programmable read only memory, it is similar to EPROM – see Chapter 1). The term non-volatile is used because the memory has to retain certain parameters when the power supply is removed.

Because the elements are processor-based devices, such sensors can be programmed for specific requirements. For example, it can be programmed to process the raw input data, correcting for such things as non-linearities, and then send the processed data to a base station. It can be programmed to send a warning signal when the measured parameter reaches some critical value.

The IEEE 1451.4 standard interface for smart sensors and actuators is based on an electronic data sheet (TEDS) format which is aimed at allowing installed analogue transducers to be easily connected to digital measurement systems. The standard requires the non-volatile EEPROM embedded memory to hold and communicate data which will allow a plug-and-play capability. It thus would hold data for the identification and properties for the sensor and might also contain the calibration template, so facilitating digital interrogation.

2.2 Output devices

The output ports of a PLC are of the relay type or optoisolator with transistor or triac types depending on the devices connected to them which are to be switched on or off. Generally, the digital signal from an output channel of a PLC is used to control an actuator which in turn controls some process. The term *actuator* is used for the device which transforms the electrical signal into some more powerful action which then results in the control of the process. The following are some examples.

2.2.1 Relay

Solenoids form the basis of a number of output control actuators. When a current passes through a solenoid a magnetic field is produced and this can then attract ferrous metal components in its vicinity. One example of such an actuator is the *relay*, the term *contactor* being used when large currents are involved. When the output from the PLC is switched on, the

solenoid magnetic field is produced and pulls on the contacts and so closes a switch or switches (Figure 2.23). The result is that much larger currents can be switched on. Thus the relay might be used to switch on the current to a motor.

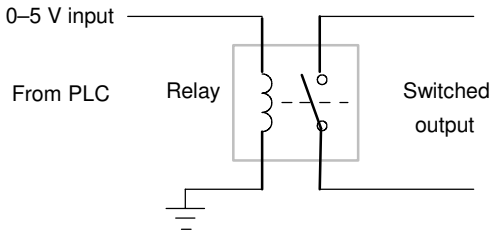


Figure 2.23 *Relay used as an output device*

2.2.2 Directional control valves

Another example of the use of a solenoid as an actuator is a *solenoid operated valve*. The valve may be used to control the directions of flow of pressurised air or oil and so used to operate other devices such as a piston moving in a cylinder. Figure 2.24 shows one such form, a *spool valve*, used to control the movement of a piston in a cylinder. Pressurised air or hydraulic fluid is inputted from port P, this being connected to the pressure supply from a pump or compressor and port T is connected to allow hydraulic fluid to return to the supply tank or, in the case of a pneumatic system, to vent the air to the atmosphere. With no current through the solenoid (Figure 2.24(a)) the hydraulic fluid of pressurised air is fed to the right of the piston and exhausted from the left, the result then being the movement of the piston to the left. When a current is passed through the solenoid, the spool valve switches the hydraulic fluid or pressurised air to the left of the piston and exhausted from the right. The piston then moves to the right. The movement of the piston might be used to push a deflector to deflect items off a conveyor belt (see Figure 1.1(b)) or implement some other form of displacement which requires power.

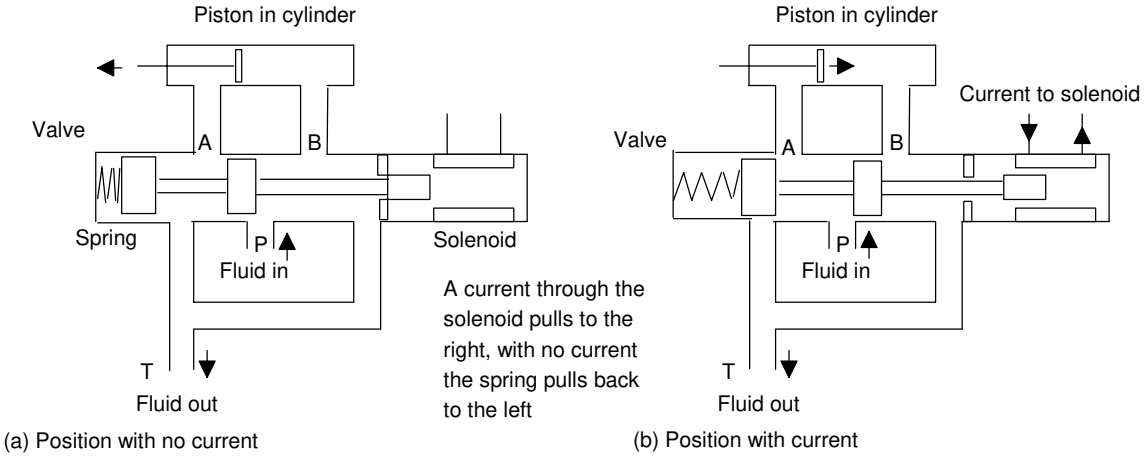


Figure 2.24 *An example of a solenoid operated valve*

With the above valve there are the two control positions shown in Figure 2.24(a) and (b). Directional control valves are described by the number of ports they have and the number of control positions. The valve shown in Figure 2.24 has four ports, i.e. A, B, P and T, and two control positions. It is thus referred to as a 4/2 valve. The basic symbol used on drawings for valves is a square, with one square being used to describe each of the control positions. Thus the symbol for the valve in Figure 2.24 consists of two squares (Figure 2.25(a)). Within each square the switching positions are then described by arrows to indicate a flow direction or a terminated line to indicate no flow path. Figure 2.25(b) shows this for the valve shown in Figure 2.24. Figure 2.26 shows some more examples of direction valves and their switching positions.

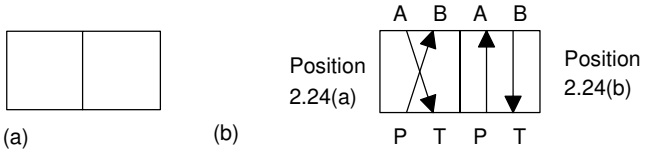


Figure 2.25 *(a) The basic symbol for a two position valve, (b) the 4/2 valve*

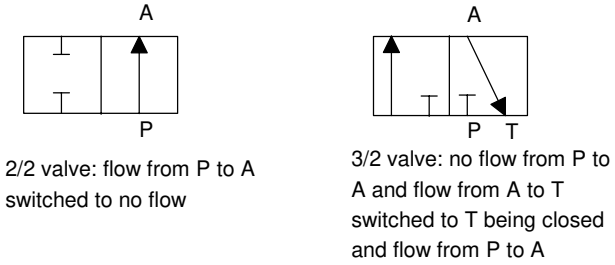


Figure 2.26 *Direction valves*

In diagrams, the actuation methods used with valves are added to the symbol; Figure 2.27 shows examples of such symbols. The valve shown in Figure 2.24 has a spring to give one position and a solenoid to give the other and so the symbol is as shown in Figure 2.27(d).

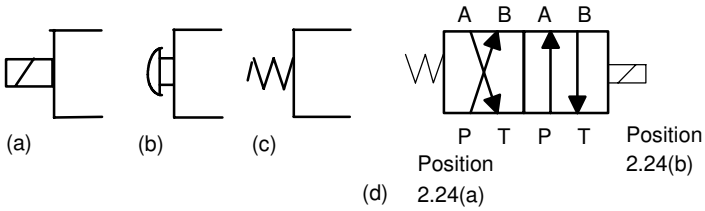


Figure 2.27 *Actuation symbols: (a) solenoid, (b) push button, (c) spring operated, (d) a 4/2 valve*

Direction valves can be used to control the direction of motion of pistons in cylinders, the displacement of the pistons being used to implement the required actions. The term *single acting cylinder* (Figure 2.28(a)) is used for one which is powered by the pressurised fluid being applied to one side of the piston to give motion in one direction, it being

returned in the other direction by possibly an internal spring. The term *double acting cylinder* (Figure 2.28(b)) is used when the cylinder is powered by fluid for its motion in both piston movement directions. Figure 2.29 shows how a valve can be used to control the direction of motion of a piston in a single-acting cylinder; Figure 2.30 shows how two valves can be used to control the action of a piston in a double acting cylinder.

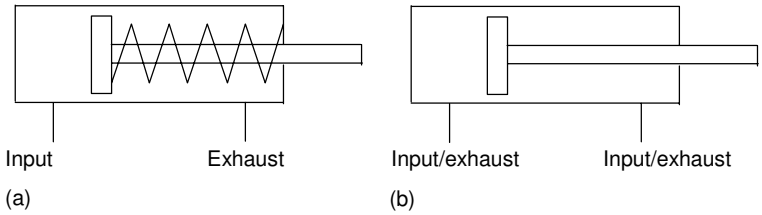


Figure 2.28 *Cylinders: (a) single acting, (b) double acting*

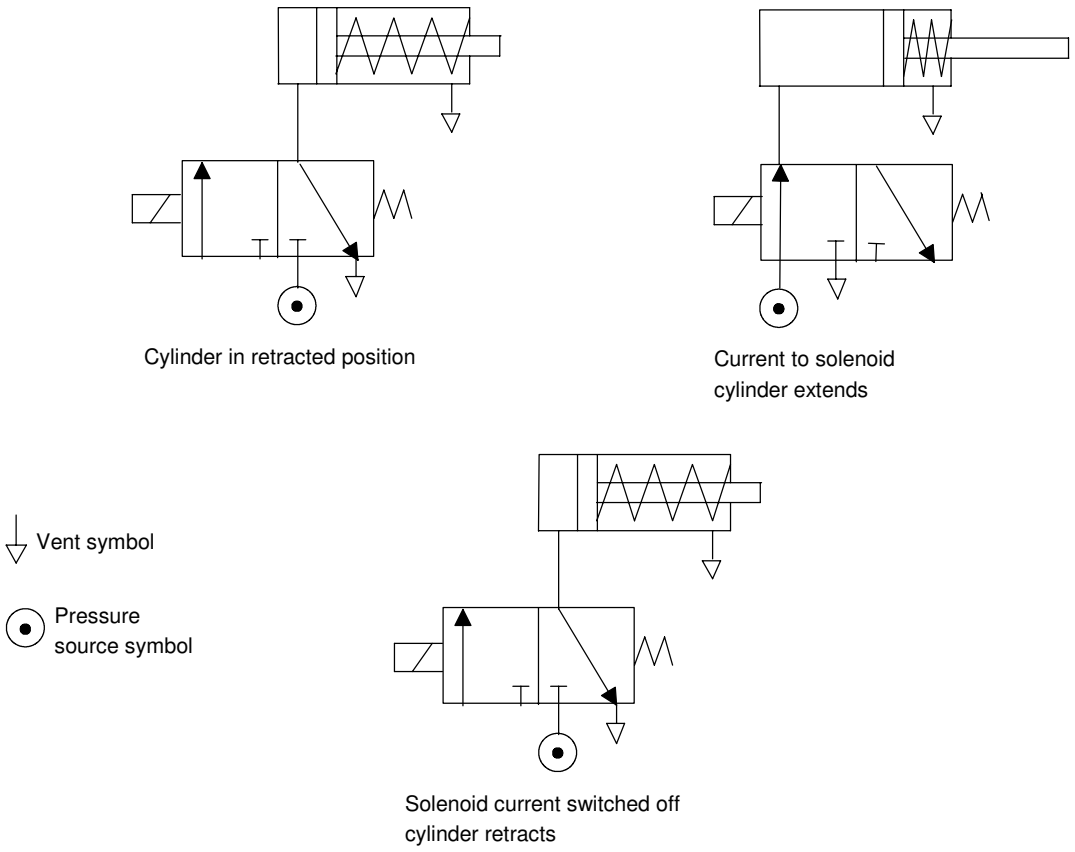


Figure 2.29 *Control of a single-acting cylinder*

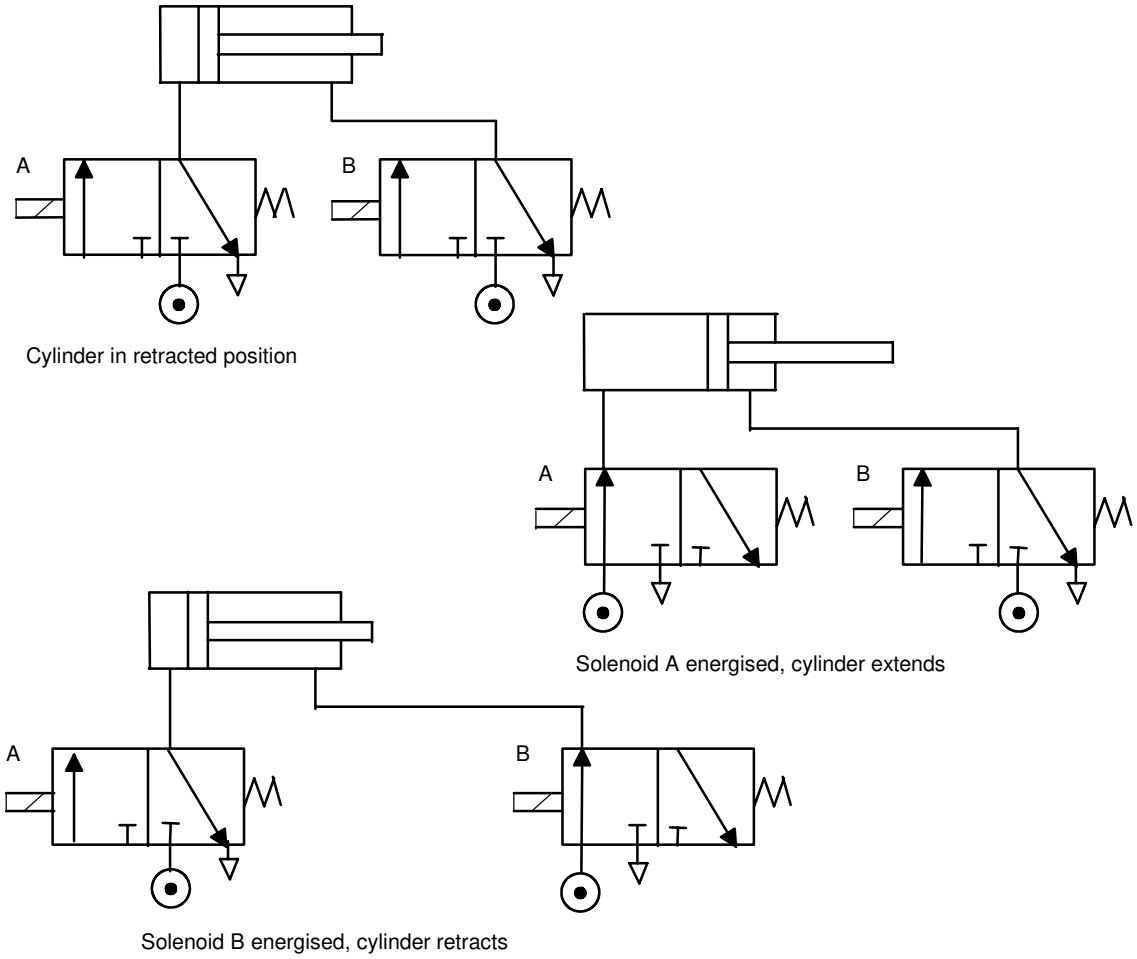


Figure 2.30 *Control of a double acting cylinder*

2.2.3 Motors

A *d.c. motor* has coils of wire mounted in slots on a cylinder of ferromagnetic material, this being termed the *armature*. The armature is mounted on bearings and is free to rotate. It is mounted in the magnetic field produced by permanent magnets or current passing through coils of wire, these being termed the *field coils*. When a current passes through the armature coil, forces act on the coil and result in rotation. Brushes and a commutator are used to reverse the current through the coil every half rotation and so keep the coil rotating. The speed of rotation can be changed by changing the size of the current to the armature coil. However, because fixed voltage supplies are generally used as the input to the coils, the required variable current is often obtained by an electronic circuit. This can control the average value of the voltage, and hence current, by varying the time for which the constant d.c. voltage is switched on (Figure 2.31). The term *pulse width modulation (PWM)* is used since the width of

the voltage pulses is used to control the average d.c. voltage applied to the armature. A PLC might thus control the speed of rotation of a motor by controlling the electronic circuit used to control the width of the voltage pulses.

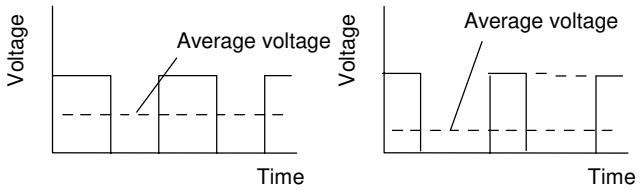


Figure 2.31 *Pulse width modulation*

Many industrial processes only require the PLC to switch a d.c. motor on or off. This might be done using a relay. Figure 2.32(a) shows the basic principle. The diode is included to dissipate the induced current resulting from the back e.m.f. Sometimes a PLC is required to reverse the direction of rotation of the motor. This can be done using relays to reverse the direction of the current applied to the armature coil. Figure 2.32(b) shows the basic principle. For rotation in one direction, switch 1 is closed and switch 2 opened. For rotation in the other direction, switch 1 is opened and switch 2 closed.

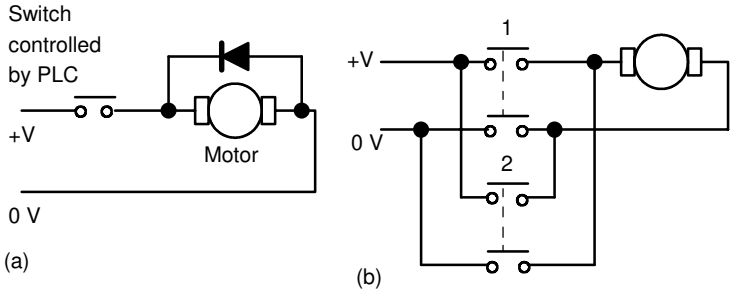


Figure 2.32 *D.c. motor: (a) on–off control, (b) directional control*

Another form of d.c. motor is the *brushless d.c. motor*. This uses a permanent magnet for the magnetic field but, instead of the armature coil rotating as a result of the magnetic field of the magnet, the permanent magnet rotates within the stationary coil. With the conventional d.c. motor, a commutator has to be used to reverse the current through the coil every half rotation in order to keep the coil rotating in the same direction. With the brushless permanent magnet motor, electronic circuitry is used to reverse the current. The motor can be started and stopped by controlling the current to the stationary coil. To reverse the motor, reversing the current is not so easy because of the electronic circuitry used for the commutator function. One method that is used is to incorporate sensors with the motor to detect the position of the north and south poles. These

sensors can then cause the current to the coils to be switched at just the right moment to reverse the forces applied to the magnet. The speed of rotation can be controlled using pulse width modulation, i.e. controlling the average value of pulses of a constant d.c. voltage.

Though a.c. motors are cheaper, more rugged and more reliable than d.c. motors, the maintaining of constant speed and controlling that speed is generally more complex than with d.c. motors. As a consequence, d.c. motors, particularly brushless permanent magnet motors, tend to be more widely used for control purposes.

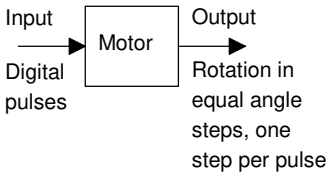


Figure 2.33 *The stepping motor*

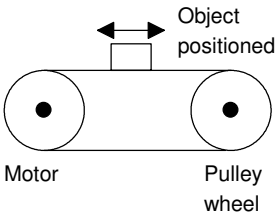


Figure 2.34 *Linear positioning*

2.2.4 Stepper motors

The *stepper* or *stepping motor* is a motor that produces rotation through equal angles, the so-termed *steps*, for each digital pulse supplied to its input (Figure 2.33). Thus, if one input pulse produces a rotation of 1.8° then 20 such pulses would give a rotation of 36.0°. To obtain one complete revolution through 360°, 200 digital pulses would be required. The motor can thus be used for accurate angular positioning.

If it is used to drive a continuous belt (Figure 2.34), it can be used to give accurate linear positioning. Such a motor is used with computer printers, robots, machine tools and a wide range of instruments where accurate positioning is required.

There are two basic forms of stepper motor, the *permanent magnet* type with a permanent magnet rotor and the *variable reluctance* type with a soft steel rotor. Figure 2.35 shows the basic elements of the permanent magnet type with two pairs of stator poles.

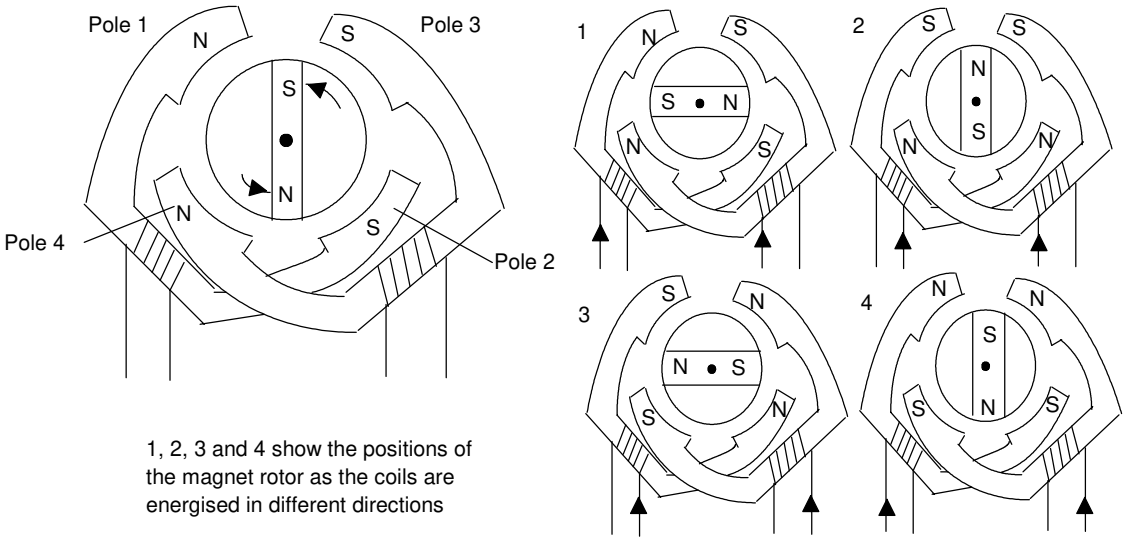


Figure 2.35 *The basic principles of the permanent magnet stepper motor (2-phase) with 90° steps*

Each pole is activated by a current being passed through the appropriate field winding, the coils being such that opposite poles are produced on opposite coils. The current is supplied from a d.c. source to the windings through switches. With the currents switched through the coils such that the poles are as shown in Figure 2.35, the rotor will move to line up with the next pair of poles and stop there. This would be, for Figure 6.35, an angle of 45°. If the current is then switched so that the polarities are reversed, the rotor will move a step to line up with the next pair of poles, at angle 135° and stop there. The polarities associated with each step are:

Step	Pole 1	Pole 2	Pole 3	Pole 4
1	North	South	South	North
2	South	North	South	North
3	South	North	North	South
4	North	South	North	South
5	Repeat of steps 1 to 4			

There are thus, in this case, four possible rotor positions: 45°, 135°, 225° and 315°.

Figure 2.36 shows the basic principle of the *variable reluctance* type. The rotor is made of soft steel and has a number of teeth, the number being less than the number of poles on the stator. The stator has pairs of poles, each pair of poles being activated and made into an electromagnet by a current being passed through the coils wrapped round them. When one pair of poles is activated, a magnetic field is produced which attracts the nearest pair of rotor teeth so that the teeth and poles line up. This is termed the position of *minimum reluctance*. By then switching the current to the next pair of poles, the rotor can be made to rotate to line up with those poles. Thus by sequentially switching the current from one pair of poles to the next, the rotor can be made to rotate in steps.

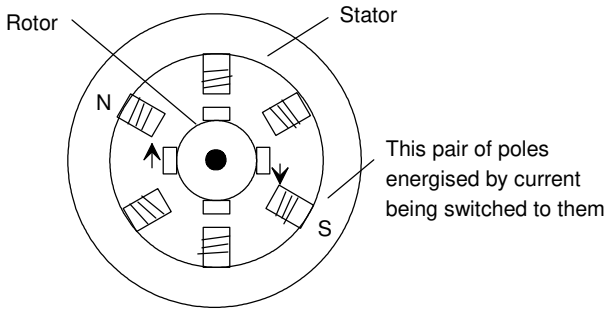


Figure 2.36 *The principle of the variable reluctance stepper motor*

There is another version of the stepper motor and that is a *hybrid stepper*. This combines features of both the permanent magnet and variable reluctance motors. They have a permanent magnet rotor encased

in iron caps which are cut to have teeth. The rotor sets itself in the minimum reluctance position in response to a pair of stator coils being energised.

To drive a stepper motor, so that it proceeds step-by-step to provide rotation, requires each pair of stator coils to be switched on and off in the required sequence when the input is a sequence of pulses (Figure 2.37). Driver circuits are available to give the correct sequencing and Figure 2.38 shows an example, the SAA 1027 for a four-phase unipolar stepper. Motors are termed *unipolar* if they are wired so that the current can only flow in one direction through any particular motor terminal, *bipolar* if the current can flow in either direction through any particular motor terminal. The stepper motor will rotate through one step each time the trigger input goes from low to high. The motor runs clockwise when the rotation input is low and anticlockwise when high. When the set pin is made low the output resets. In a control system, these input pulses might be supplied by a microprocessor.

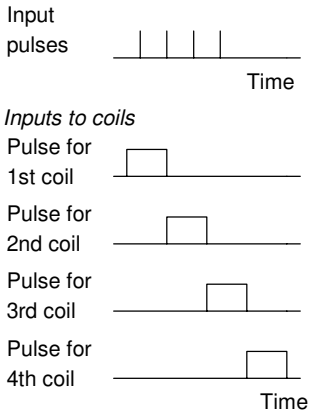


Figure 2.37 *Input and outputs of the drive system*

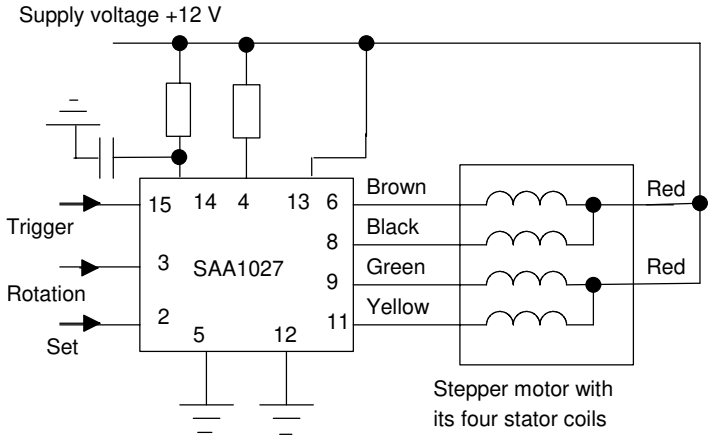


Figure 2.38 *Driver circuit connections with the integrated circuit SAA1027*

2.3 Examples of applications

The following are some examples of control systems designed to illustrate the use of a range of input and output devices.

2.3.1 A conveyor belt

Consider a conveyor belt that is to be used to transport goods from a loading machine to a packaging area (Figure 2.39). When an item is loaded onto the conveyor belt, a contact switch might be used to indicate that the item is on the belt and start the conveyor motor. The motor then has to keep running until the item reaches the far end of the conveyor and falls off into the packaging area. When it does this, a switch might be activated which has the effect of switching off the conveyor motor. The motor is then to remain off until the next item is loaded onto the belt. Thus the inputs to a PLC controlling the conveyor are from two switches and the output is to a motor.

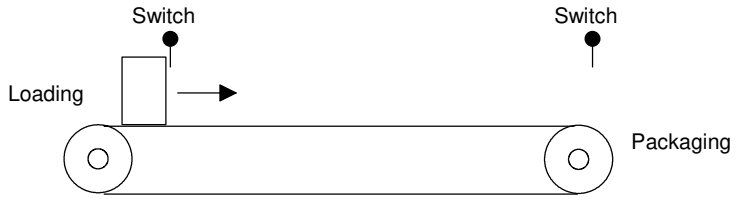


Figure 2.39 *Conveyor*

2.3.2 A lift

Consider a simple goods lift to move items from one level to another. It might be bricks from the ground level to the height where the bricklayers are working. The lift is to move upwards when a push button is pressed at the ground level to send the lift upwards or a push button is pressed at the upper level to request the lift to move upwards, but in both cases there is a condition that has to be met that a limit switch indicates that the access gate to the lift platform is closed. The lift is to move downwards when a push button is pressed at the upper level to send the lift downwards or a push button is pressed at the lower level to request the lift to move downwards, but in both cases there is a condition that has to be met that a limit switch indicates that the access gate to the lift platform is closed. Thus the inputs to the control system are electrical on–off signals from push button switches and limit switches. The output from the control system is the signal to control the motor.

2.3.3 A robot control system

Figure 6.40 shows how directional control valves can be used for a control system of a robot. When there is an input to solenoid A of valve 1, the piston moves to the right and causes the gripper to close. If solenoid B is energised, with A de-energised, the piston moves to the left and the gripper opens. When both solenoids are de-energised, no air passes to either side of the piston in the cylinder and the piston keeps its position without change. Likewise, inputs to the solenoids of valve 2 are used to extend or retract the arm. Inputs to the solenoids of valve 3 are used to

move the arm up or down. Inputs to the solenoids of valve 4 are used to rotate the base in either a clockwise or anticlockwise direction.

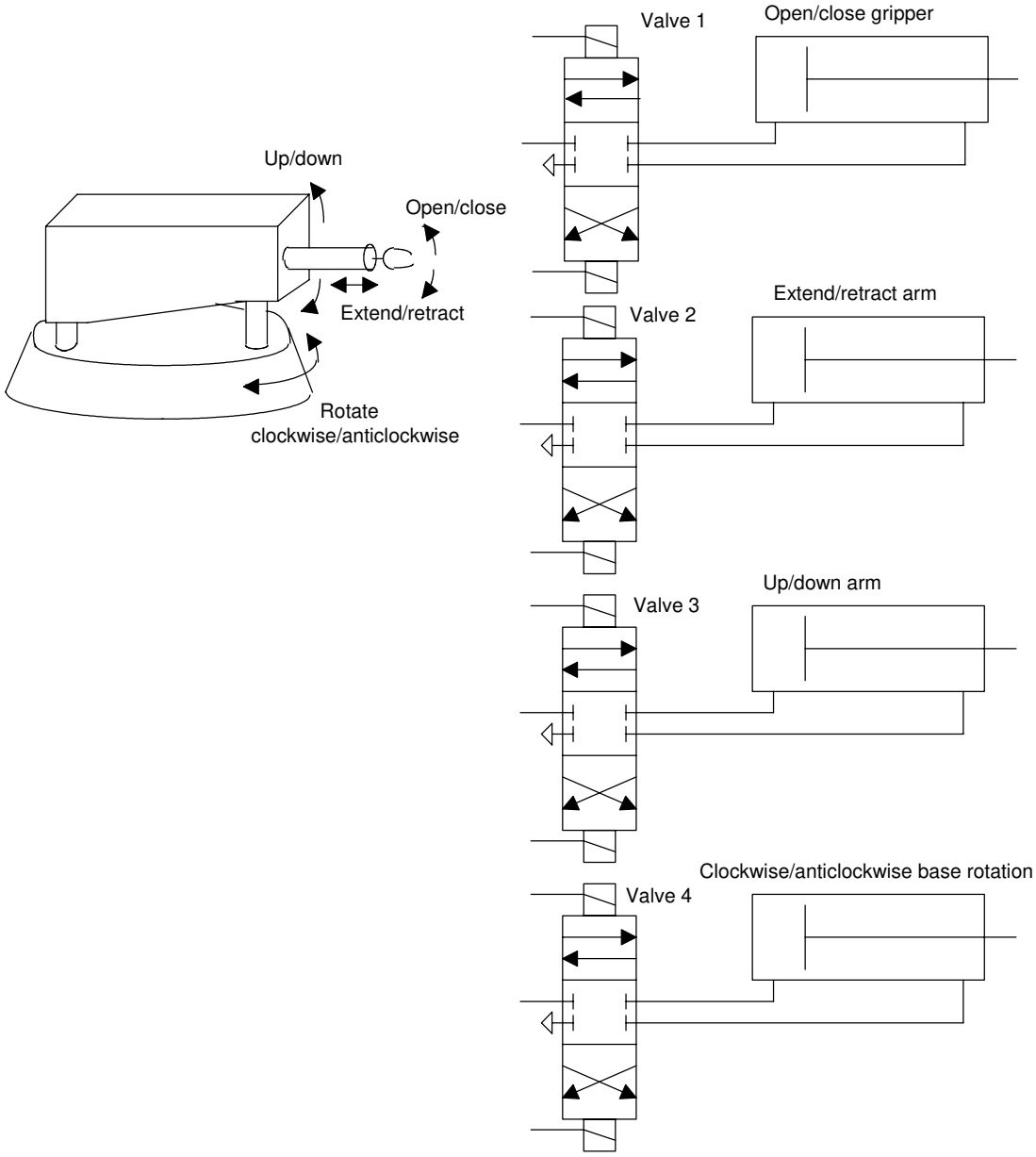


Figure 2.40 *Robot controls*

2.3.4 Liquid level monitoring

Figure 6.41 shows a method that could be used to give an on–off signal when the liquid in a container reaches a critical level. A magnetic float, a ring circling the sensor probe, falls as the liquid level falls and opens a

reed switch when the critical level is reached. The reed switch is in series with a 39 Ω resistor so that this is switched in parallel with a 1 kΩ resistor by the action of the reed switch. Opening the reed switch thus increases the resistance from about 37 Ω to 1 kΩ. Such a resistance change can be transformed by signal conditioning to give suitable on–off signals.

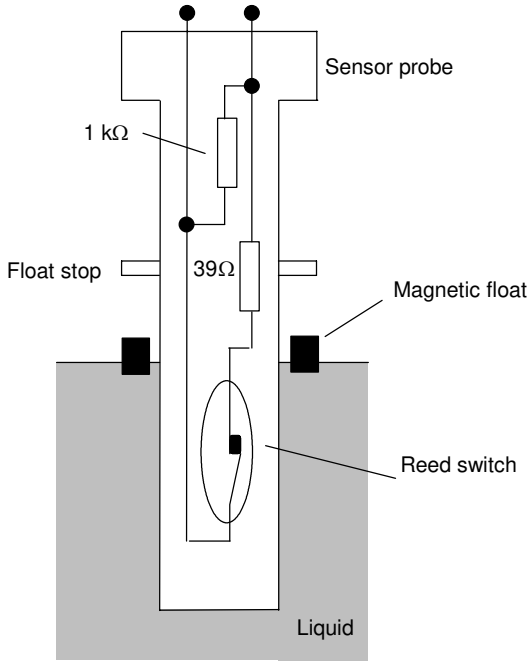


Figure 2.41 *Liquid level monitoring*

3 Number systems

The number system used for everyday calculations is the *denary* or *decimal system*. This is based on the use of the 10 digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. With a number represented by this system, the digit position in the number indicates the weight attached to each digit, the weight increasing by a factor of 10 as we proceed from right to left. Hence we have:

...	10^3	10^2	10^1	10^0
	thousands	hundreds	tens	units
Denary	1000	100	10	1

Counting can, however, be done to any base. The denary system is just convenient because we have ten fingers. If we had only two then our system for everyday counting would probably have been different. Computers, and hence PLC systems, are based on counting in twos because it is convenient for their system, their two digits being effectively just the off and on signals. When working with PLCs, other base number systems are also used, e.g. input and output addresses are often specified using the octal system, i.e. base 8.

3.1 The binary system

The *binary system* is based on just two digits: 0 and 1. These are termed *bi*nary digi*ts* or *bits*. When a number is represented by this system, the digit position in the number indicates the weight attached to each digit, the weight increasing by a factor of 2 as we proceed from right to left.

...	2^3	2^2	2^1	2^0
	bit 3	bit 2	bit 1	bit 0
Binary	1000	100	10	1

The bit 0 is termed the *least significant bit* (LSB) and the highest bit the *most significant bit* (MSB). For example, with the binary number 1010, the least significant bit is the bit at the right-hand end of the number and so is 0. The most significant bit is the bit at the left-hand end of the number and so is 1.

	2^3	2^2	2^1	2^0
	bit 3	bit 2	bit 1	bit 0
	MSB			LSB
Binary	1	0	1	0

When converted to a denary number we have, for the 1010:

	2^3	2^2	2^1	2^0
Binary	1	0	1	0
Denary	$2^3 = 8$	0	$2^1 = 2$	0

Thus the denary equivalent is 10. The conversion of a binary number to a denary number thus involves the addition of the powers of 2 indicated by the number.

The conversion of a denary number to a binary number involves looking for the appropriate powers of 2. We can do this by successive divisions by 2, noting the remainders at each division. Thus with the denary number 31:

31 ÷ 2 = 15 remainder 1 This gives the LSB
15 ÷ 2 = 7 remainder 1
7 ÷ 2 = 3 remainder 1
3 ÷ 2 = 1 remainder 1 This gives the MSB

The binary number is 11111. The first division gives the least significant bit because we have just divided the 31 by 2, i.e. 2^1 and found 1 left over for the 2^0 digit. The last division gives the most significant bit because the 31 has then been divided by 2 four times, i.e. 2^4 and the remainder is 1.

3.2 Octal and hexadecimal

Binary numbers are used in computers because the two states represented by 0 and 1 are easy to deal with switching circuits where they can represent off and on. A problem with binary numbers is that a comparatively small number requires a large number of digits. For example, the denary number 9 which involves just a single digit requires four when written as the binary number 1001. The denary number 181, involving three digits, in binary form is 10110101 and requires eight digits. Because of this, octal or hexadecimal numbers are sometimes used to make numbers easier to handle and act as a 'half-way house' between denary numbers and the binary numbers which computers work with.

3.2.1 Octal system

The *octal system* is based on eight digits: 0, 1, 2, 3, 4, 5, 6, 7. When a number is represented by this system, the digit position in the number indicates the weight attached to each digit, the weighting increasing by a factor of 8 as we proceed from right to left. Thus we have:

...	8^3	8^2	8^1	8^0
Octal	1000	100	10	1

To convert denary numbers to octal we successively divide by 8 and note the remainders. Thus the denary number 15 divided by 8 gives 1 with remainder 7 and thus the denary number 15 is 17 in the octal system. To convert from octal to denary we multiply the digits by the power of 8 appropriate to its position in the number. For example, the octal number 365 is $3 \times 8^2 + 6 \times 8^1 + 5 \times 8^0 = 245$. To convert from binary into octal, the binary number is written in groups of three bits starting with the least

significant bit. For example, the binary number 11010110 would be written as:

11 010 110

Each group is then replaced by the corresponding digit 0 to 7. The 110 binary number is 6, the 010 is 2 and the 11 is 3. Thus the octal number is 326. As another example, the binary number 100111010 is:

100 111 010 Binary
4 7 2 Octal

Octal to binary conversion involves converting each octal digit into its 3-bit equivalent. Thus, for the octal number 21 we have 1 as 001 and 2 as 010:

2 1 Octal number
010 001 Binary number

and so the binary number is 010001.

3.2.2 Hexadecimal system

The *hexadecimal system (hex)* is based on 16 digits/symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. When a number is represented by this system, the digit position in the number indicates that the weight attached to each digit increases by a factor of 16 as we proceed from right to left. Thus we have:

...	16^3	16^2	16^1	16^0
Hex	1000	100	10	1

For example, the decimal number 15 is F in the hexadecimal system. To convert from denary numbers into hex we successively divide by 16 and note the remainders. Thus the denary number 156 when divided by 16 gives 9 with remainder 12 and so in hex is 9C. To convert from hex to denary we multiply the digits by the power of 16 appropriate to its position in the number. Thus hex 12 is $1 \times 16^1 + 2 \times 16^0 = 18$. To convert binary numbers into hexadecimal numbers, we group the binary numbers into fours starting from the least significant number. Thus, for the binary number 1110100110 we have:

11 1010 0110 Binary number
3 A 6 Hex number

For conversion from hex to binary, each hex number is converted to its 4-bit equivalent. Thus, for the hex number 1D we have 0001 for the 1 and 1101 for the D:

1 D Hex number
0001 1101 Binary number

Thus the binary number is 0001 1101.

Because the external world tends to deal mainly with numbers in the denary system and computers with numbers in the binary system, there is always the problem of conversion. There is, however, no simple link between the position of digits in a denary number and the position of digits in a binary number. An alternative method that is often used is the *binary coded decimal system (BCD)*. With this system, each denary digit is coded separately in binary. For example, the denary number 15 has the 5 converted into the binary number 0101 and the 1 into 0001:

```
   1      5   Denary number
0001 0101   Binary number
```

to give in BCD the number 0001 0101.

3.2.3 Numbers in the binary, octal, hex and BCD systems

Table 3.1 gives examples of numbers in the denary, binary, octal, hex and BCD systems.

Table 3.1 *Examples of numbers in different systems*

Denary	Binary	Octal	Hex	BCD
0	00000	0	0	0000 0000
1	00001	1	1	0000 0001
2	00010	2	2	0000 0010
3	00011	3	3	0000 0011
4	00100	4	4	0000 0100
5	00101	5	5	0000 0101
6	00110	6	6	0000 0110
7	00111	7	7	0000 0111
8	01000	10	8	0000 1000
9	01001	11	9	0000 1001
10	01010	12	A	0001 0000
11	01011	13	B	0001 0001
12	01100	14	C	0001 0010
13	01101	15	D	0001 0011
14	01110	16	E	0001 0100
15	01111	17	F	0001 0101
16	10000	20	10	0001 0110
17	10001	21	11	0001 0111

3.3 Binary arithmetic

Addition of binary numbers uses the following rules:

$$0 + 0 = 0$$

$$0 + 1 = 1 + 0 = 1$$

1 + 1 = 10

1 + 1 + 1 = 11

Consider the addition of the binary numbers 01110 and 10011.

	01110
	10011
Sum	100001

For bit 0 in the sum, 0 + 1 = 1. For bit 1 in the sum, 1 + 1 = 10 and so we have 0 with 1 carried to the next column. For bit 2 in the sum, 1 + 0 + the carried 1 = 10. For bit 3 in the sum, 1 + 0 + the carried 1 = 10. We continue this through the various bits and end up with the 100001.

Subtraction of binary numbers follows the following rules:

0 – 0 = 0

1 – 0 = 1

1 – 1 = 0

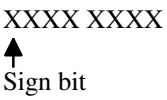
When evaluating 0 – 1, a 1 is borrowed from the next column on the left containing a 1. The following example illustrates this with the subtraction of 01110 from 11011:

	11011
	01110
Difference	01101

For bit 0 we have 1 – 0 = 1. For bit 1 we have 1 – 1 = 0. For bit 2 we have 0 – 1. We thus borrow 1 from the next column and so have 10 – 1 = 1. For bit 3 we have 0 – 1, remember we borrowed the 1. Again borrowing 1 from the next column, we then have 10 – 1 = 1. For bit 4 we have 0 – 0 = 0, remember we borrowed the 1.

3.3.1 Signed numbers

The binary numbers considered so far contain no indication whether they are negative or positive and are said to be *unsigned*. Since there is generally a need to handle both positive and negative numbers there needs to be some way of distinguishing between them. This can be done by adding a sign bit. When a number is said to be *signed* then the most significant bit is used to indicate the sign of the number, a 0 being used if the number is positive and a 1 if it is negative. Thus for an 8-bit number we have:



When we have a positive number then we write it in the normal way with a 0 preceding it. Thus a positive binary number of 10110 would be written as 010110. A negative number of 10110 would be written as 110110. However, this is not the most useful way of writing negative numbers for ease of manipulation by computers.

3.3.2 One's and two's complements

A more useful way of writing signed negative numbers is to use the two's complement method. A binary number has two complements, known as the *one's complement* and the *two's complement*. The one's complement of a binary number is obtained by changing all the 1s in the unsigned number into 0s and the 0s into 1s. Thus if we have the binary number 101101 then the one's complement of it is 010010. The two's complement is obtained from the one's complement by adding 1 to the least significant bit of the one's complement. Thus the one's complement of 010010 becomes 010011.

When we have a negative number then, to obtain the signed two's complement, we obtain the two's complement and then sign it with a 1. Consider the representation of the decimal number −6 as a signed two's complement number when the total number of bits is to be eight. We first write the binary number for +6, i.e. 0000110, then obtain the one's complement of 1111001, add 1 to give 1111010, and finally sign it with a 1 to indicate it is negative. The result is thus 11111010.

Unsigned binary number when sign ignored	000 0110
One's complement	111 1001
Add 1	1
Unsigned two's complement	111 1010
Signed two's complement	1111 1010

Table 3.2 lists some signed two's complements, given to 4 bits, for denary numbers.

When we have a positive number then we sign the normal binary number with a 0, i.e. we only write negative numbers in the two's complement form. A consequence of adopting this method of writing negative and positive numbers is that when we add the signed binary equivalent of +4 and –4, i.e. 0000 0100 and 111 1100 we obtain (1)0000 0000 and so zero within the constraints of the number of bits used, the (1) being neglected.

Table 3.2 *Signed two's complements*

Denary number	Signed 2's complement
–5	1011
–4	1100
–3	1101
–2	1110
–1	1111

Subtraction of a positive number from a positive number can be considered to be the addition of a negative number to a positive number. Thus we obtain the signed two's complement of the negative number and then add it to the signed positive number. Hence, for the subtraction of the denary number 6 from the denary number 4 we can consider the problem as being (+4) + (−6). Hence we add the signed positive number to the signed two's complement for the negative number.

Binary form of +4	0000 0100
(−6) as signed two's complement	1111 1010
Sum	1111 1110

The most significant bit, i.e. the sign, of the outcome is 1 and so the result is negative. This is the 8-bit signed two's complement for −2.

If we wanted to add two negative numbers then we would obtain the signed two's complement for each number and then add them. Whenever a number is negative we use the signed two's complement, when positive just the signed number.

3.3.3 Floating point numbers

Before discussing floating point numbers, consider *fixed point numbers*. Fixed point numbers are ones where there is a fixed location of the point separating integers from fractional numbers. Thus, 15.3 is an example of a denary fixed point number, 1010.1100 an example of a fixed point binary number and DE.2A an example of a fixed point hexadecimal number. We have, with the eight-bit binary number, four digits before the binary point and four digits after it. When two such binary numbers are added by a computing system the procedure is to recognise that the fixed point is fixed the same in both numbers so we can ignore it for the addition, carry out the addition of the numbers and then insert in the result the binary point in its fixed position. For example, suppose we want to add 0011.1010 and 0110.1000, we drop the binary point to give:

0011 1010 + 0110 1000 = 1010 0010

Inserting the binary point then gives 1010.0010.

Using fixed points does present problems. If we are concerned with very large or very small numbers we could end up with a large number of zeros between the integers and the point, e.g. 0.000 000 000 000 023. For

this reason *scientific notation* is used for such numbers. Thus, the above number might be written as 0.23×10^{-13} or 2.3×10^{-14} or 23×10^{-15}. Likewise, the binary number 0.0000 0111 0010 might be represented as 110010×2^{-12} (the 12 would also be in binary format) or 11001.0×2^{-11} (the 11 being in binary format). Such notation has a *floating point*.

A floating point number is in the form $a \times r^e$, where a is termed the *mantissa*, r the *radix* or *base* and e the *exponent* or *power*. With binary numbers the base is understood to be 2, i.e. we have $a \times 2^e$, and when we know we are dealing with binary numbers we need not store the base with the number. Thus a computing system needs, in addition to storing the sign, i.e. whether positive or negative, to store the mantissa and the exponent.

Because with floating point numbers it is possible to store a number in several different ways, e.g. 0.1×10^2 and 0.01×10^3, with computing systems such numbers are normalised. This means that they are all put in the form $0.1 \times r^e$. Thus, with binary numbers we have 0.1×2^e, e.g. if we had 0.00001001 it would become 0.1001×2^{-4}. In order to take account of the sign of a binary number we then add a sign bit of 0 for a positive number and 1 for a negative number. Thus the number 0.1001×2^{-4} becomes 1.1001×2^{-4} if negative and 0.1001×2^{-4} if positive.

Unlike fixed point numbers, floating point numbers cannot be directly added unless the exponents are the same. Thus to carry out addition we need to make the exponents the same.

3.4 PLC data

Most PLCs operate with a 16-bit word, the term *word* being used for the group of bits constituting some information. This allows a positive number in the range 0 to +65 535, i.e. 1111 1111 1111 1111, to be represented or a signed number in the range –32 768 to +32 767 in two's complement, the most significant bit then representing the sign. Such signed numbers are referred to as integers with the symbol INT being used with inputs and outputs in programs of such 16-bit words. The term SINT is used for short integer numbers where only 8-bits are used, such numbers giving the range –128 to +127. The term DINT is used for double integer numbers where 32 bits are used, such numbers giving the range -2^{31} to $+2^{31} - 1$. LINT is used for long integer numbers where 64 bits are used, such numbers giving the range -2^{63} to $+2^{63} - 1$. Where numbers are not signed the symbols UINT, USINT, UDINT and ULINT are used with integers, short integers, double integers and long integers.

Decimal fractions are referred to as real or floating point numbers, being represented by the symbol REAL for inputs and outputs in programs. These consist of two 16-bit words and so we might have 1.234567E+03 for the number $1.234\,567 \times 10^{+3}$, the E indicating that the number that follows is the exponent. The term LREAL is used for long real numbers where 64 bits are used.

The term BOOL is used for Boolean type data, such data being on/off values, i.e. 0 or 1, and thus represented by single bits.

Time duration, e.g. for the duration of a process, is represented by the IEC (International Electrotechnical Commission) standard using the symbols d for days, h for hours, m for minutes, s for seconds and ms for milliseconds as, for example, T#12d2h5s3ms or TIME#12d2h5s for 12

4 I/O processing

This chapter continues the discussion of inputs and outputs from Chapter 2 and is a brief consideration of the processing of the signals from input and output devices. The input/output (I/O) unit provides the interface between the PLC controller and the outside world and must therefore provide the necessary signal conditioning to get the signal to the required level and also to isolate it from possible electrical hazards such as high voltages. This chapter includes the forms of typical input/output modules and, in an installation where sensors are some distance from the PLC processing, their communication links to the PLC.

4.1 Input/output units

Input signals from sensors and the outputs required for actuating devices can be:

1 *Analogue*, i.e. a signal whose size is related to the size of the quantity being sensed.
2 *Discrete*, i.e. essentially just an on–off signal.
3 *Digital*, i.e. a sequence of pulses.

The CPU, however, must have an input of digital signals of a particular size, normally 0 to 5 V. The output from the CPU is digital, normally 0 to 5 V. Thus there is generally a need to manipulate input and output signals so that they are in the required form.

The input/output (I/O) units of PLCs are designed so that a range of input signals can be changed into 5 V digital signals and so that a range of outputs are available to drive external devices. It is this in-built facility to enable a range of inputs and outputs to be handled which makes PLCs so easy to use. The following is a brief indication of the basic circuits used for input and output units. In the case of rack instruments they are mounted on cards which can be plugged into the racks and so the input/output characteristics of the PLC can thus be changed by changing the cards. A single box form of PLC has input/output units incorporated by the manufacturer.

4.1.1 Input units

The terms *sourcing* and *sinking* refer to the manner in which d.c. devices are interfaced with the PLC (see Section 1.3.5). For a PLC input unit, with sourcing it is the source of the current supply for the input device connected to it (Figure 4.1(a)). With sinking, the input device provides the current to the input unit (Figure 4.1(b)).

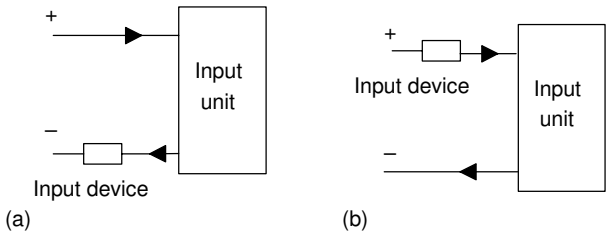


Figure 4.1 *Input unit: (a) sourcing, (b) sinking*

Figures 4.2 and 4.3 show the basic input unit circuits for discrete and digital d.c. and discrete a.c. inputs. Optoisolators (see Section 1.3.4) are used to provide protection. With the a.c. input unit, a rectifier bridge network is used to rectify the a.c. so that the resulting d.c. signal can provide the signal for use by the optoisolator to give the input signals to the CPU of the PLC. Individual status lights are provided for each input to indicate when the input device is providing a signal.

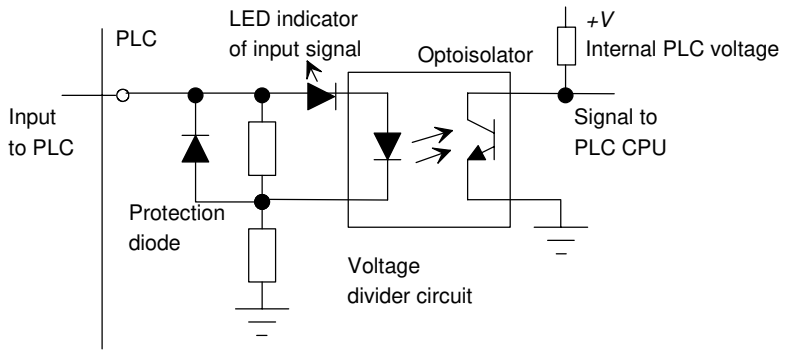


Figure 4.2 *D.C. input unit*

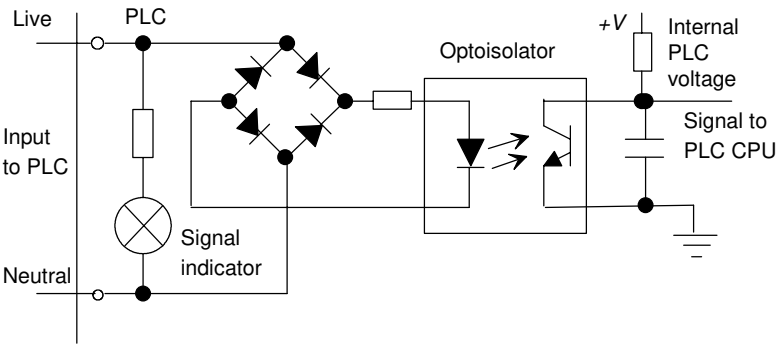


Figure 4.3 *A.C. input unit*

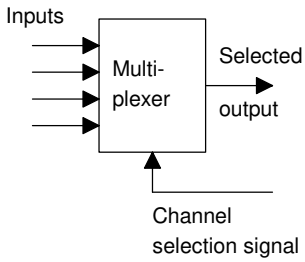


Figure 4.4 *Multiplexer*

Analogue signals can be inputted to a PLC if the input channel is able to convert the signal to a digital signal using an analogue-to-digital converter. With a rack mounted system this may be achieved by mounting a suitable analogue input card in the rack. So that one analogue card is not required for each analogue input, multiplexing is generally used (Figure 4.4). This involves more than one analogue input being connected to the card and then electronic switches used to select each input in turn. Cards are typically available giving 4, 8 or 16 analogue inputs.

Figure 4.5(a) illustrates the function of an analogue-to-digital converter (ADC). A single analogue input signal gives rise to on–off output signals along perhaps eight separate wires. The eight signals then constitute the so-termed digital *word* corresponding to the analogue input signal level. With such an 8-bit converter there are $2^8 = 256$ different digital values possible; these are 0000 0000 to 1111 1111, i.e. 0 to 255. The digital output goes up in steps (Figure 4.5(b)) and the analogue voltages required to produce each digital output are termed *quantisation levels*.

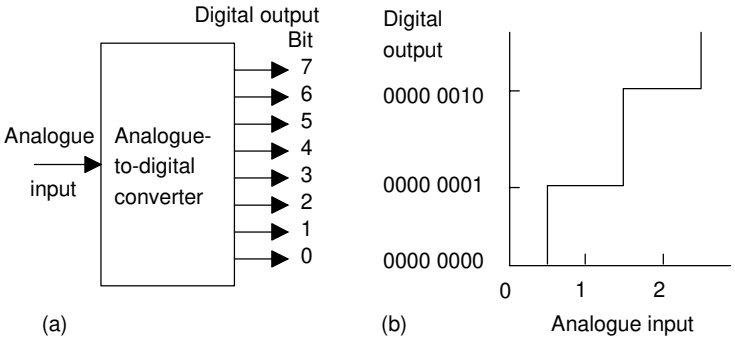


Figure 4.5 *(a) Function of an analogue-to-digital converter, (b) an illustration of the relationship between the analogue input and the digital output*

The analogue voltage has to change by the difference in analogue voltage between successive levels if the binary output is to change. The term *resolution* is used for the smallest change in analogue voltage which will give rise to a change in one bit in the digital output. With an 8-bit ADC, if, say, the full-scale analogue input signal varies between 0 and 10 V then a step of one digital bit involves an analogue input change of 10/255 V or about 0.04 V. This means that a change of 0.03 V in the input will produce no change in the digital output. The number of bits in the output from an analogue-to-digital converter thus determines the *resolution*, and hence *accuracy*, possible. If a 10-bit ADC is used then $2^{10} = 1024$ different digital values are possible and, for the full-scale analogue input of 0 to 10 V, a step of one digital bit involves an analogue input change of 10/1023 V or about 0.01 V. If a 12-bit ADC is used then $2^{12} = 4096$ different digital values are possible and, for the full-scale analogue input of 0 to 10 V, a step of one digital bit involves an analogue input change of 10/4095 V or about 2.4 mV. In general, the resolution of an n-bit ADC is $1/(2^n - 1)$.

The following illustrates the analogue-to-digital conversion for an 8-bit converter when the analogue input is in the range 0 to 10 V:

Analogue input (V)	Digital output (V)
0.00	0000 0000
0.04	0000 0001
0.08	0000 0010
0.12	0000 0011
0.16	0000 0100
0.20	0000 0101
0.24	0000 0110
0.28	0000 0111
0.32	0000 1000
etc.	

To illustrate the above, consider a thermocouple used as a sensor with a PLC and giving an output of 0.5 mV per °C. What will be the accuracy with which the PLC will activate the output device if the thermocouple is connected to an analogue input with a range of 0 to 10 V d.c and using a 10-bit analogue-to-digital converter? With a 10-bit converter there is 2^{10} = 1024 bits covering the 0 to 10 V range. Thus a change of 1 bit corresponds to 10/1023 V or about 0.01 V, i.e. 10 mV. Hence the accuracy with which the PLC recognises the input from the thermocouple is ±5 mV or ±10°C.

4.1.2 Output units

With a PLC output unit, when it provides the current for the output device (Figure 4.6(a)) it is said to be sourcing and when the output device provides the current to the output unit it is said to be sinking (Figure 4.6(b)). Quite often, sinking input units are used for interfacing with electronic equipment and sourcing output units for interfacing with solenoids.

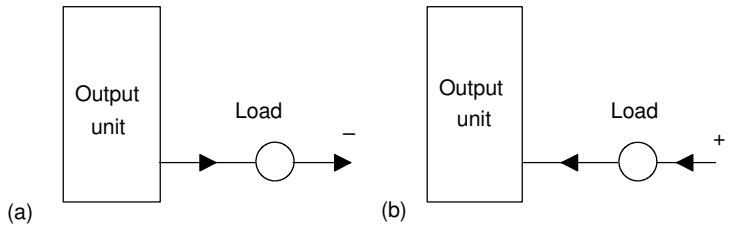


Figure 4.6 *Output unit: (a) sourcing, (b) sinking*

Output units can be relay, transistor or triac. Figure 4.7 shows the basic form of a relay output unit, Figure 4.8 that of a transistor output unit and Figure 4.9 that of a triac output unit.

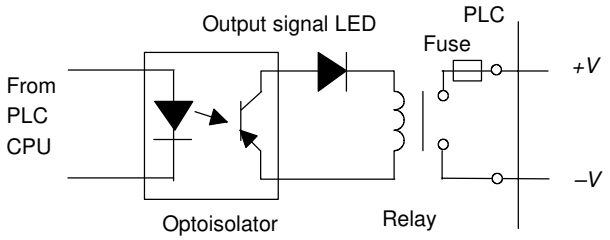


Figure 4.7 *Relay output unit*

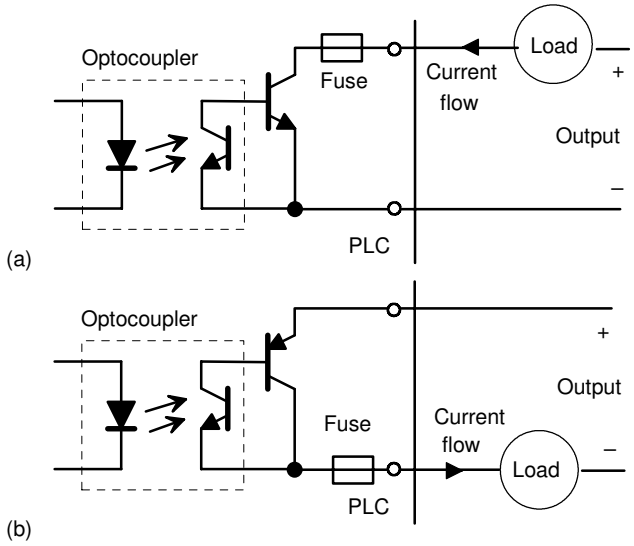


Figure 4.8 *Basic forms of transistor output: (a) current sinking, (b) current sourcing*

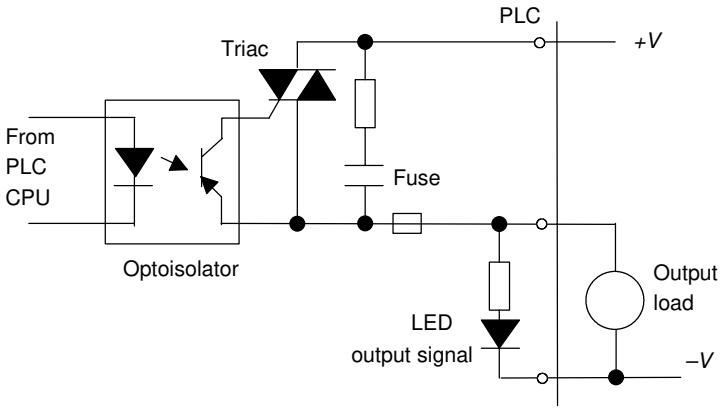


Figure 4.9 *Triac output unit*

Analogue outputs are frequently required and can be provided by digital-to-analogue converters (DACs) at the output channel. The input to the converter is a sequence of bits with each bit along a parallel line. Figure 4.10 shows the basic function of the converter.

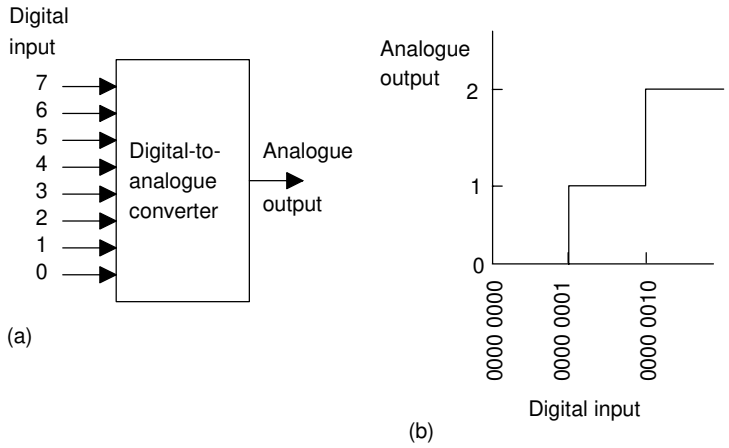


Figure 4.10 *(a) DAC function, (b) digital-to-analogue conversion*

A bit in the 0 line gives rise to a certain size output pulse. A bit in the 1 line gives rise to an output pulse of twice the size of the 0 line pulse. A bit in the 2 line gives rise to an output pulse of twice the size of the 1 line pulse. A bit in the 3 line gives rise to an output pulse of twice the size of the 2 line pulse, and so on. All the outputs add together to give the analogue version of the digital input. When the digital input changes, the analogue output changes in a stepped manner, the voltage changing by the voltage changes associated with each bit. For example, if we have an 8-bit converter then the output is made up of voltage values of 2^8 = 256 analogue steps. Suppose the output range is set to 10 V d.c. One bit then gives a change of 10/255 V or about 0.04 V. Thus we have:

Digital input (V)	Analogue output (V)
00000000	0.00
00000001	0.04
00000010	0.08 + 0.00 = 0.08
00000011	0.08 + 0.04 = 0.12
00000100	0.16
00000101	0.016 + 0.00 + 0.04 = 0.20
00000110	0.016 + 0.08 = 0.24
00000111	0.016 + 0.08 + 0.04 = 0.28
00001000	0.32
etc.	

Analogue output modules are usually provided in a number of outputs, e.g. 4 to 20 mA, 0 to +5 V d.c., 0 to +10 V d.c., and the appropriate output is selected by switches on the module. Modules generally have outputs in two forms, one for which all the outputs from that module have a common voltage supply and one which drives outputs having their own individual voltage supplies. Figure 4.11 shows the basic principles of these two forms of output.

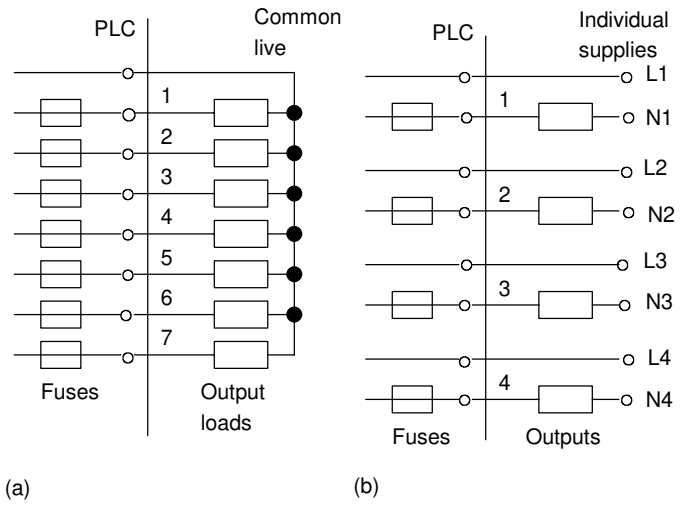


Figure 4.11 *Forms of output: (a) common supply, (b) individual supplies*

4.2 Signal conditioning

When connecting sensors which generate digital or discrete signals to an input unit, care has to be taken to ensure that voltage levels match. However, many sensors generate analogue signals. In order to avoid having a multiplicity of analogue input channels to cope with the wide diversity of analogue signals that can be generated by sensors, external signal conditioning is often used to bring analogue signals to a common range and so allow a standard form of analogue input channel to be used.

A common standard that is used (Figure 4.12) is to convert analogue signals to a current in the range 4 to 20 mA and thus to a voltage by passing it through a 250 Ω resistance to give a 1 to 5 V input signal. Thus, for example, a sensor used to monitor liquid level in the height range 0 to 1 m would have the 0 level represented by 4 mA and the 1 m represented by 20 mA. The use of 4 mA to represent the low end of the analogue range serves the purpose of distinguishing between when the sensor is indicating zero and when the sensor is not working and giving zero response for that reason. When this happens the current would be 0 mA. The 4 mA also is often a suitable current to operate a sensor and so eliminate the need for a separate power supply.

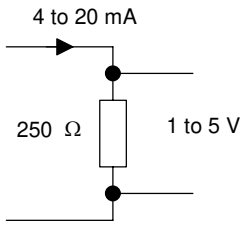


Figure 4.12 *Standard analogue signal*

A potential divider (Figure 4.13) can be used to reduce a voltage from a sensor to the required level; the output voltage level V_{out} is:

$$V_{out} = \frac{R_2}{R_1 + R_2} V_{in}$$

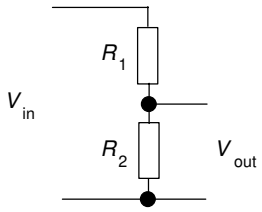


Figure 4.13 *Potential divider*

Amplifiers can be used to increase the voltage level; Figure 4.14 shows the basic form of the circuits that might be used with a 741 operational amplifier with (a) being an inverting amplifier and (b) a non-inverting amplifier. With the inverting amplifier the output V_{out} is:

$$V_{out} = -\frac{R_2}{R_1} V_{in}$$

and with the non-inverting amplifier:

$$V_{out} = \frac{R_1 + R_2}{R_1} V_{in}$$

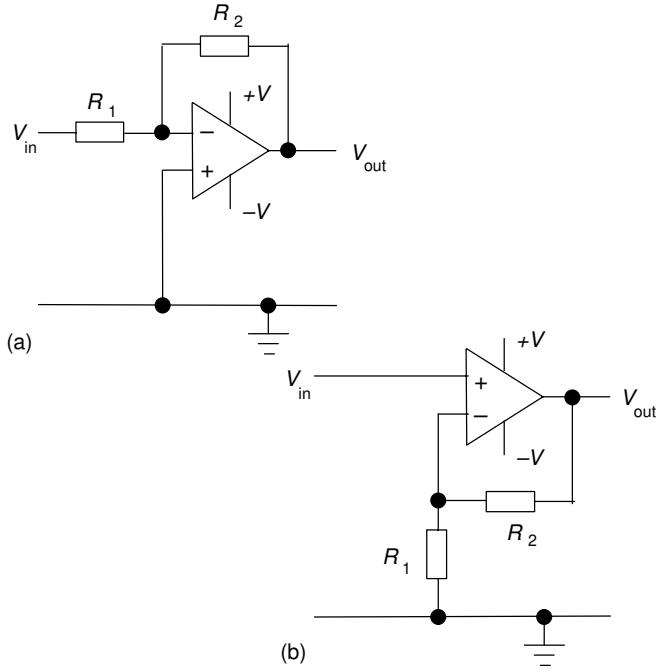


Figure 4.14 *Operational amplifier circuits*

Often a differential amplifier is needed to amplify the difference between two input voltages. Such is the case when a sensor, e.g. a strain gauge, is connected in a Wheatstone bridge and the output is the difference between two voltages or a thermocouple where the voltage difference between the hot and cold junctions is required. Figure 4.14 shows the basic form of an operational amplifier circuit for this purpose. The output voltage V_{out} is:

$$V_{out} = \frac{R_2}{R_1}(V_2 - V_1)$$

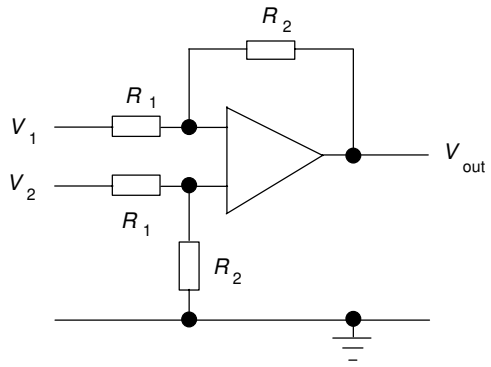


Figure 4.15 *Differential amplifier*

As an illustration of the use of signal conditioning, Figure 4.16 shows the arrangement that might be used for a strain gauge sensor. The sensor is connected in a Wheatstone bridge and the out-of-balance potential difference amplified by a differential amplifier before being fed via an analogue-to-digital converter unit which is part of the analogue input port of the PLC.

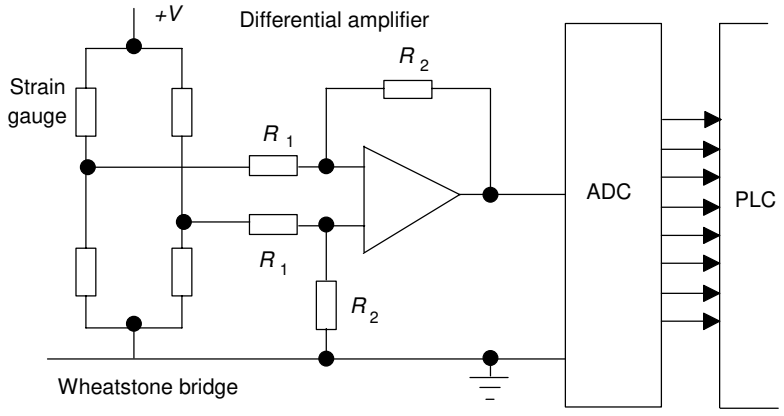


Figure 4.16 *Signal conditioning with a strain gauge sensor*

4.3 Remote connections

When there are many inputs or outputs located considerable distances away from the PLC, while it would be possible to run cables from each such device to the PLC a more economic solution is to use input/output modules in the vicinity of the inputs and outputs and use just a single core cable to connect each, over the long distances, to the PLC instead of the multicore cable that would be needed without such distant I/O modules (Figure 4.17).

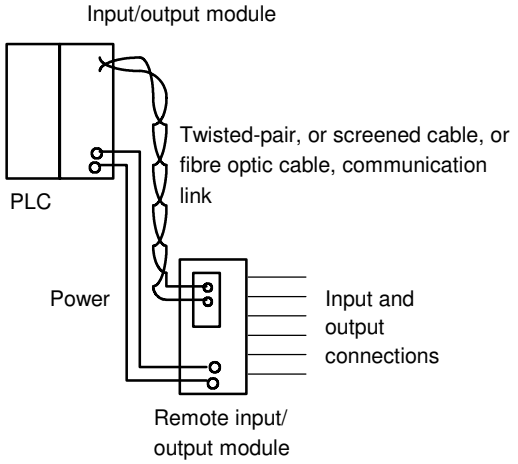


Figure 4.17 *Use of remote input/output module*

In some situations a number of PLCs may be linked together with a master PLC unit sending and receiving input/output data from the other units (Figure 4.18). The distant PLCs do not contain the control program since all the control processing is carried out by the master PLC.

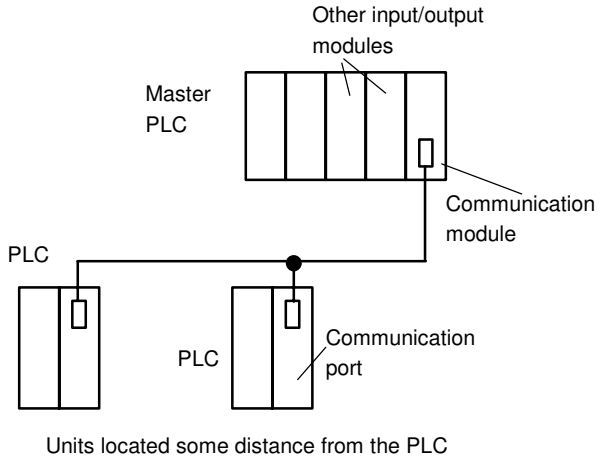


Figure 4.18 *Use of remote input/output PLC systems*

The cables used for communicating data between remote input/output modules and a central PLC, remote PLCs and the master PLC are

typically *twisted-pair cabling*, often routed through grounded steel conduit in order to reduce electrical 'noise'. *Coaxial cable* enables higher data rates to be transmitted and does not require the shielding of steel conduit. *Fibre-optic cabling* has the advantage of resistance to noise, small size and flexibility and is now becoming more widely used.

4.3.1 Serial and parallel communications

Serial communication is when data is transmitted one bit at a time (Figure 4.19(a)). Thus if an 8-bit word is to be transmitted, the eight bits are transmitted one at a time in sequence along a cable. This means that a data word has to be separated into its constituent bits for transmission and then reassembled into the word when received. *Parallel communication* is when all the constituent bits of a word are simultaneously transmitted along parallel cables (Figure 4.19(b)). This allows data to be transmitted over short distances at high speeds.

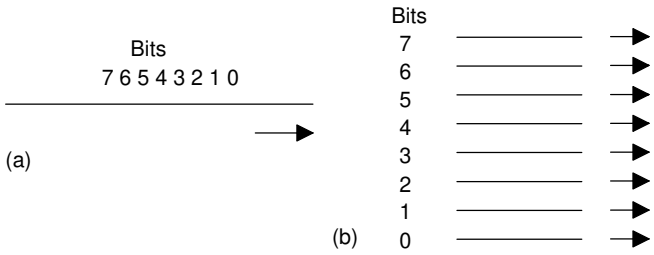


Figure 4.19 *(a) Serial communication, (b) parallel communication*

Serial communication is used for transmitting data over long distances. It is much cheaper to run, for serial communication, a single core cable over a long distance than the multicore cables that would be needed for parallel communication. With a PLC system, serial communication might be used for the connection between a computer, when used as a programming terminal, and a PLC. Parallel communication might be used when connecting laboratory instruments to the system. However, internally, PLCs work, for speed, with parallel communications. Thus, circuits called UARTS (universal asynchronous receivers–transmitters) have to be used at input/output ports to converts serial communications signals to parallel.

4.3.2 Serial standards

For successful serial communications to occur, it is necessary to specify:

1 The voltage levels to be used for signals, i.e. what signal represents a 0 and what represents a 1.
2 What the bit patterns being transmitted mean and how the message is built up. Bear in mind that a sequence of words are being sent along the same cable and it is necessary to be able to be able to determine when one word starts and finishes and the next word starts.
3 The speed at which the bit pattern is to be sent, i.e. the number of bits per second.

4 Synchronisation of the clocks at each end. This is necessary if, for example, a particular duration transmitted pulse it to be recognised by the receiver as just a single bit rather than two bits.
5 Protocols, or flow controls, to enable such information as 'able to receive data' or 'not ready to receive data' to be received. This is commonly done by using two extra signal wires (termed handshake wires), one to tell the receiver that the transmitter is ready to send the data and the other to tell the transmitter that the receiver is ready to receive data.
6 Error-checking to enable a bit pattern to be checked to determine if corruption of the data has occurred during transmission.

The most common standard serial communications interface used is the *RS232*. Connections are made via 25-pin D-type connectors (Figure 4.20) with usually, though not always, a male plug on cables and a female socket on the equipment. Not all the pins are used in every application.

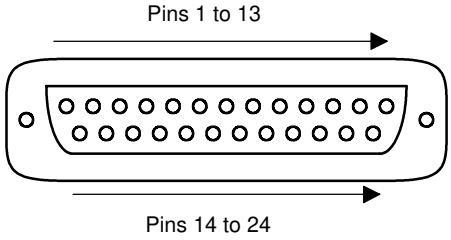


Figure 4.20 *D connector*

The minimum requirements are:

Pin 1: Ground connection to the frame of chassis
Pin 2: Serial transmitted data (output data pin)
Pin 3: Serial received data (input data pin)
Pin 7: Signal ground which acts as a common signal return path

A configuration that is widely used with interfaces involving computers is:

Pin 1: Ground connection to the frame of chassis
Pin 2: Serial transmitted data (output data pin)
Pin 3: Serial received data (input data pin)
Pin 4: Request to send
Pin 5: Clear to send
Pin 6: Data set ready
Pin 7: Signal ground which acts as a common signal return path
Pin 20: Data terminal ready

The signals sent through pins 4, 5, 6 and 20 are used to check that the receiving end is ready to receive a signal, the transmitting end is ready to send and the data is ready to be sent. With RS232, a 1 bit is represented by a voltage between −5 and −25 V, normally −12 V, and a 0 by a voltage between +5 and +25 V, normally +12 V.

The term *baud rate* is used to describe the transmission rate, it being approximately the number of bits transmitted or received per second. However, not all the bits transmitted can be used for data, some have to be used to indicate the start and stop of a serial piece of data, these often being termed *flags*, and as a check as to whether the data has been corrupted during transmission. Figure 4.21 shows the type of signal that might be sent with RS232. The parity bit is added to check whether corruption has occurred, with even parity a 1 being added to make the number of 1s an even number. To send seven bits of data, eleven bits may be required.

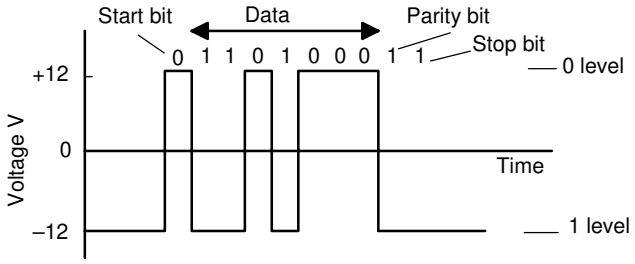


Figure 4.21 *RS232 signal levels*

Other standards such as the *RS422* and *RS423* are similar to RS232. RS232 is limited over the distances it can be used, noise limiting the transmission of high numbers of bits per second when the length of cable is more than about 15 m. RS422 can be used for longer distances. This uses a balanced method of transmission. Such circuits require two lines for the transmission, the transmitted signal being the voltage difference between the two lines. Noise affecting both lines equally will have no effect on the transmitted signal. Figure 4.22 shows how, for RS232 and RS422, the data rates that can be transmitted without noise becoming too significant depend on the distance. RS422 lines can be used for much greater distances than RS232.

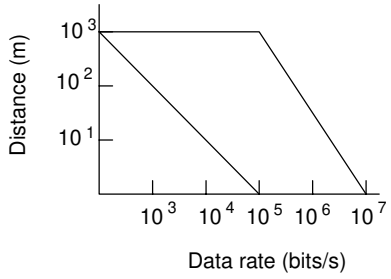


Figure 4.22 *Transmission with RS232 and RS422*

An alternative to RS422 is the *20 mA loop* which was an earlier standard and is still widely used for long distance serial communication, particularly in industrial systems where the communication path is likely to suffer from electrical noise (Figure 4.23). This system consists of a

circuit, a loop of wire, containing a current source. The serial data is transmitted by the current being switched on and off, a 0 being transmitted as zero current and a 1 as 20 mA.

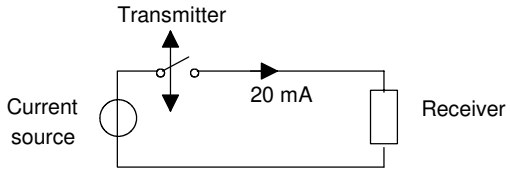


Figure 4.23 *20 mA loop*

4.3.3 Parallel standards

The standard interface most commonly used for parallel communications is *IEEE-488*. This was originally developed by Hewlett Packard to link their computers and instruments and was known as the *Hewlett Packard Instrumentation Bus*. It is now often termed the *General Purpose Instrument Bus*. This bus provides a means of making interconnections so that parallel data communications can take place between listeners, talkers and controllers. Listeners are devices that accept data from the bus, talkers place data, on request, on the bus and controllers manage the flow of data on the bus and provide processing facilities. There is a total of 24 lines, of which eight bi-directional lines are used to carry data and commands between the various devices connected to the bus, five lines are used for control and status signals, three are used for handshaking between devices and eight are ground return lines (Figure 4.24).

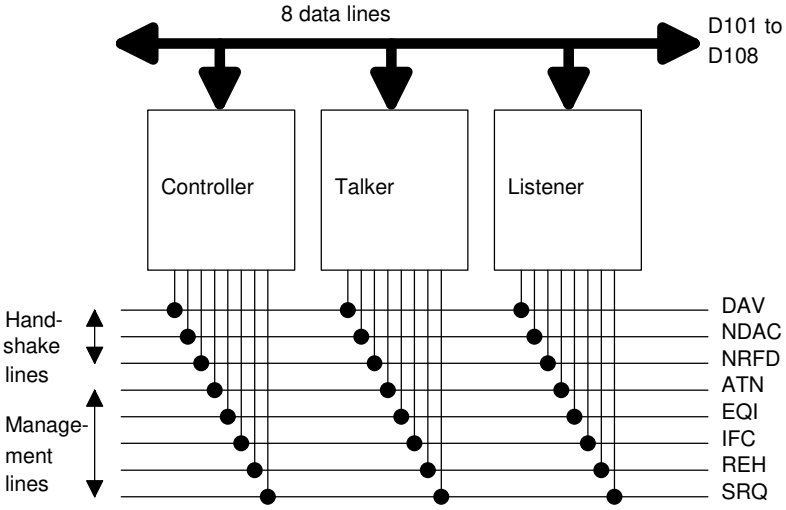


Figure 4.24 *The IEEE-488 bus structure*

Commands from the controller are signalled by taking the Attention Line (ATN) low, otherwise it is high, and thus indicating that the data lines contain data. The commands can be directed to individual devices by placing addresses on the data lines. Each device on the bus has its own

address. Device addresses are sent via the data lines as a parallel 7-bit word, the lowest 5-bits providing the device address and the other two bits control information. If both these bits are 0 then the commands are sent to all addresses, if bit 6 is 1 and bit 7 a 0 the addressed device is switched to be a listener, if bit 6 is 0 and bit 7 is 1 then the device is switched to be a talker.

As illustrated above by the function of the ATN line, the management lines each have an individual task in the control of information. The handshake lines are used for controlling the transfer of data. The three lines ensure that the talker will only talk when it is being listened to by listeners. Table 4.1 lists the functions of all the lines and their pin numbers in a 25-way D-type connector.

Table 4.1 *IEEE-488 bus system*

Pin	Signal group	Abbreviation	Signal/function
1	Data	D101	Data line 1.
2	Data	D102	Data line 2.
3	Data	D103	Data line 3.
4	Data	D104	Data line 4.
5	Management	EOI	End Or Identify. This is used to either signify the end of a message sequence from a talker device or is used by the controller to ask a device to identify itself.
6	Handshake	DAV	Data valid. When the level is low on this line then the information on the data bus is valid and acceptable.
7	Handshake	NRFD	Not Ready For Data. This line is used by listener devices taking it high to indicate that they are ready to accept data.
8	Handshake	NDAC	Not Data Accepted. This line is used by listeners taking it high to indicate that data is being accepted.
9	Management	IFC	Interface Clear. This is used by the controller to reset all the devices of the system to the start state.
10	Management	SRQ	Service Request. This is used by devices to signal to the controller that they need attention.
11	Management	ATN	Attention. This is used by the controller to signal that it is placing a command on the data lines.
12		SHIELD	Shield.
13	Data	D105	Data line 5.
14	Data	D106	Data line 6.
15	Data	D107	Data line 7.
16	Data	D108	Data line 8.
17	Management	REN	Remote Enable. This enables a device on the bus to indicate that it is to be selected for remote control rather than by its own control panel.
18		GND	Ground/common.
19		GND	Ground/common.
20		GND	Ground/common.
21		GND	Ground/common.
22		GND	Ground/common.
23		GND	Ground/common.
24		GND	Ground/common.

Figure 4.25 shows the handshaking sequence that occurs when data is put on the data lines. Initially DAV is high, indicating that there is no valid data on the data bus, NRFD and NDAC also being low. When a data word is put on the data lines, NRFD is made high to indicate that all listeners are ready to accept data and DAV is made low to indicate that new data is on the data lines. When a device accepts a data word it sets NDAC high to indicate that is has accepted the data and NRFD low to indicate that it is now not ready to accept data. When all the listeners have set NDAC high, then the talker cancels the data valid signal, DAV going high. This then results in NDAC being set low. The entire process can then be repeated for another word being put on the data bus.

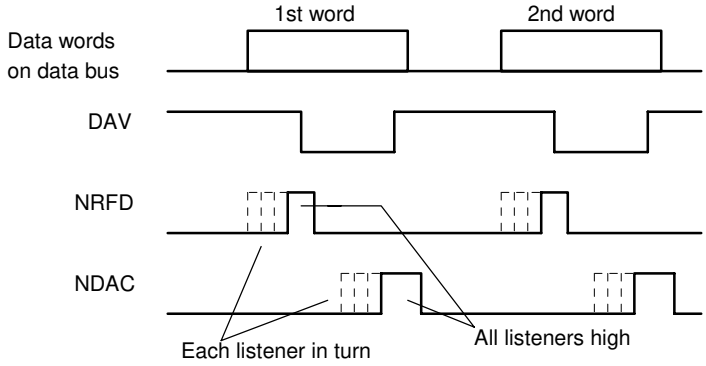


Figure 4.25 *Handshaking sequence*

4.3.4 Protocols

It is necessary to exercise control of the flow of data between two devices so what constitutes the message, and how the communication is to be initiated and terminated, is defined. This is termed the *protocol*.

Thus one device needs to indicate to the other to start or stop sending data. This can be done by using handshaking wires connecting transmitting and receiving devices so that a signal along one such wire can tell the receiver that the transmitter is ready to send (RTS) and along another wire that the transmitter is ready to receive, a clear to send signal (CTS). RTS and CTS lines are provided for in RS232 serial communication links.

An alternative is to use additional characters on the transmitting wires. With the ENQ/ACK protocol, data packets are sent to a receiver with a query character ENQ. When this character is received the end of the data packet has been reached. Once the receiver has processed that data, it can indicate it is ready for another block of data by sending back an acknowledge (ACK) signal. Another form, the XON/XOFF, has the receiving device sending a XOFF signal to the sending device when it wishes the data flow to cease. The transmitter then waits for an XON signal before resuming transmission.

One form of checking for errors in the message that might occur as a result of transmission is the *parity check*. This is an extra bit added to a message to ensure that the number of bits in a piece of data is always odd

or always even. For example, 0100100 is even since there is an even number of 1s and 0110100 is odd since there is an odd number of 1s. To make both these odd parity then the extra bit added at the end in the first case is 1 and in the second case 0, i.e. we have 01001001 and 01101000. Thus when a message is sent out with odd bit parity, if the receiver finds that the bits give an even sum, then the message has been corrupted during transmission and the receiver can request that the message be repeated.

The parity bit method can detect if there is an error resulting from a single 0 changing to a 1 or a 1 changing to a 0 but cannot detect two such errors occurring since there is then no change in parity. To check on such occurrences more elaborate checking methods have to be used. One method involves storing data words in an array of rows and columns. Parity can then be checked for each row and each column. The following illustrates this for seven words using even parity.

		Row parity bits
Column parity bits	00101010	1
↑	10010101	0
	10100000	0
Block	01100011	0
of data	11010101	1
	10010101	1
↓	00111100	0

Another method, termed *cyclic redundancy check codes*, involves splitting the message into blocks. Each block is then treated as a binary number and is divided by a predetermined number. The remainder from this division is then sent as the error checking number on the conclusion of the message and enables a check on the accuracy of the message to be undertaken.

4.3.5 ASCII codes

The most widely used code for the transmission of characters is the ASCII code (American Standard Code for Information Interchange). This is a seven-bit code giving 128 different combinations of bits covering lower case and upper case alphanumeric characters, punctuation and 32 control codes. As an illustration, Table 4.2 shows the codes used for capital letters. Examples of control codes are SOH, for start of heading, i.e. the first character of a heading of an information message, as 000 0001; STX, for start of text, as 000 0010; ETX, for end of text, as 000 0011; EOT, for end of transmission, as 000 0011.

4.4 Networks

The increasing use of automation in industry has led to the need for communications and control on a plant-wide basis with programmable controllers, computers, robots, and CNC machines interconnected. The term *local area network (LAN)* is used to describe a communications network designed to link computers and their peripherals within the same building or site.

Table 4.2 *Examples of ASCII codes*

	ASCII		ASCII		ASCII
A	100 0001	N	100 1110	0	011 0000
B	100 0010	O	100 1111	1	011 0001
C	100 0011	P	101 0000	2	011 0010
D	100 0100	Q	101 0001	3	011 0011
E	100 0101	R	101 0010	4	011 0100
F	100 0110	S	101 0011	5	011 0101
G	100 0111	T	101 0100	6	011 0110
H	100 1000	U	101 0101	7	011 0111
I	100 1001	V	101 0110	8	011 1000
J	100 1010	W	101 0111	9	011 1001
K	100 1011	X	101 1000		
L	100 1100	Y	101 1001		
M	100 1101	Z	101 1010		

Networks can take three basic forms. With the *star* form (Figure 4.26(a)) the terminals are each directly linked to a central computer, termed the host, or master with the terminals being termed slaves. The host contains the memory, processing and switching equipment to enable the terminals to communicate. Access to the terminals is by the host asking each terminal in turn whether it wants to talk or listen. With the *bus* or *single highway* type of network (Figure 4.26(b)), each of the terminals is linked into a single cable and so each transmitter/receiver has a direct path to each other transmitter/receiver in the network. Methods, i.e. protocols, have to be adopted to ensure that no more than one terminal talks at once, otherwise confusion can occur. A terminal has to be able to detect whether another terminal is talking before it starts to talk. With the *ring* network (Figure 4.26(c)), a continuous cable, in the form of a ring, links all the terminals. Again methods have to be employed to enable communications from different terminals without messages becoming mixed up. The single highway and the ring methods are often termed *peer to peer* in that each terminal has equal status. Such a system allows many stations to use the same network.

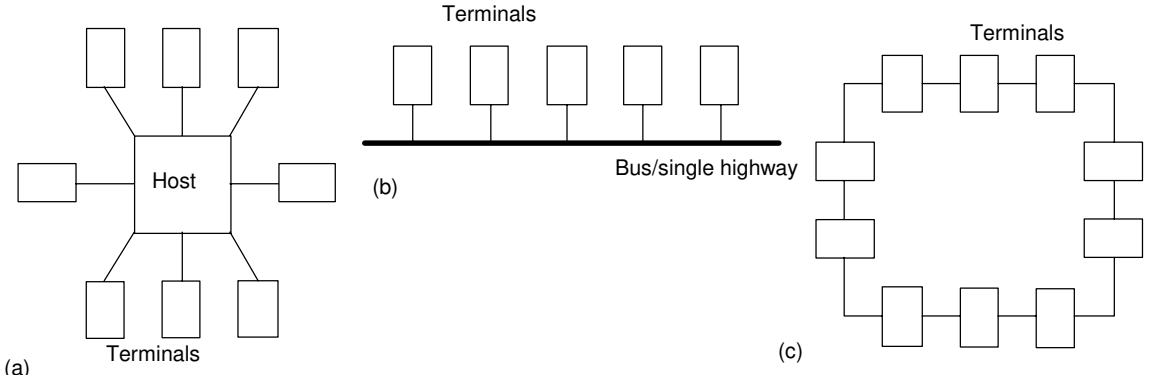


Figure 4.26 *Networks: (a) star, (b) bus/single highway, (c) ring*

With ring-based networks, two commonly used methods that are employed to avoid two stations talking at once and so giving rise to confusion are token passing and slot passing. With token passing, a special bit pattern called a token is circulated round the network. When a station wishes to transmit into the network it waits until it receives the token, then transmits the data with the token attached. Another station wishing to transmit cannot do so until the token has been freed by removal from the data by a receiver. With slot passing, empty slots are circulated into which stations can deposit data for transmission.

Bus systems generally employ the method in which a system wishing to transmit listens to see if any messages are being transmitted. If no message is being transmitted, a station can take control of the network and transmit its message. This method is known as *carrier sense multiple access* (CSMA). However, we could end up with two stations simultaneously perceiving the network to be clear for transmission and both simultaneously taking control and sending messages. The result would be a 'collision' of their transmitted data and corruption. If such a situation is detected, both stations cease transmitting and wait a random time before attempting to again transmit. This is known as *carrier sense multiple access with collision detection* (CSMA/CD).

Different PLC manufacturers adopt different forms of network systems and methods of communication for use with their PLCs. For example, Mitsubishi uses a network termed MelsecNET, Allen Bradley uses Data Highway Plus, General Electric uses GENET, Texas Instruments uses TIWAY and Siemens has four forms under the general name SINEC. Most employ peer to peer forms, e.g. Allen-Bradley. Siemens has two low level forms, SINECL1 which is a star, i.e. master-slave form, and SINECL2 which is peer to peer.

4.4.1 Distributed systems

Often PLCs figure in an entire hierarchy of communications (Figure 4.27). Thus at the lowest level we have input and output devices such as sensor and motors interfaced through I/O interfaces with the next level. The next level involves controllers such as small PLCs or small computers, linked through a network with the next level of larger PLCs and computers exercising local area control. These in turn may be part of a network involved with a large mainframe company computer controlling all.

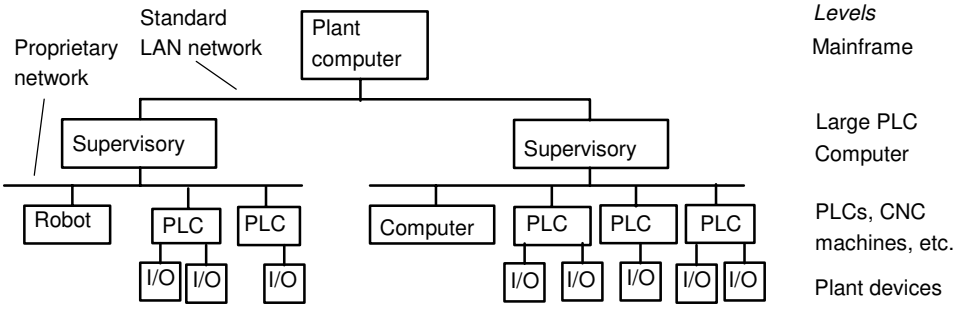


Figure 4.27 *Control hierarchy*

There is increasing use made of systems that can both control and monitor industrial processes. This involves control and the gathering of data. The term SCADA, which stands for *supervisory control and data acquisition system*, is widely used for such a system.

4.4.2 Network standards

Interconnecting several devices can present problems because of compatibility problems, e.g. they may operate at different baud rates or use different protocols. In order to facilitate communications between different devices the International Standards Organisation (ISO) in 1979 devised a model to be used for standardisation for open systems interconnection (OSI); the model is termed the *ISO/OSI model*. A communication link between items of digital equipment is defined in terms of physical, electrical, protocol and user standards, the ISO/OSI model breaking this down into seven layers (Figure 4.28).

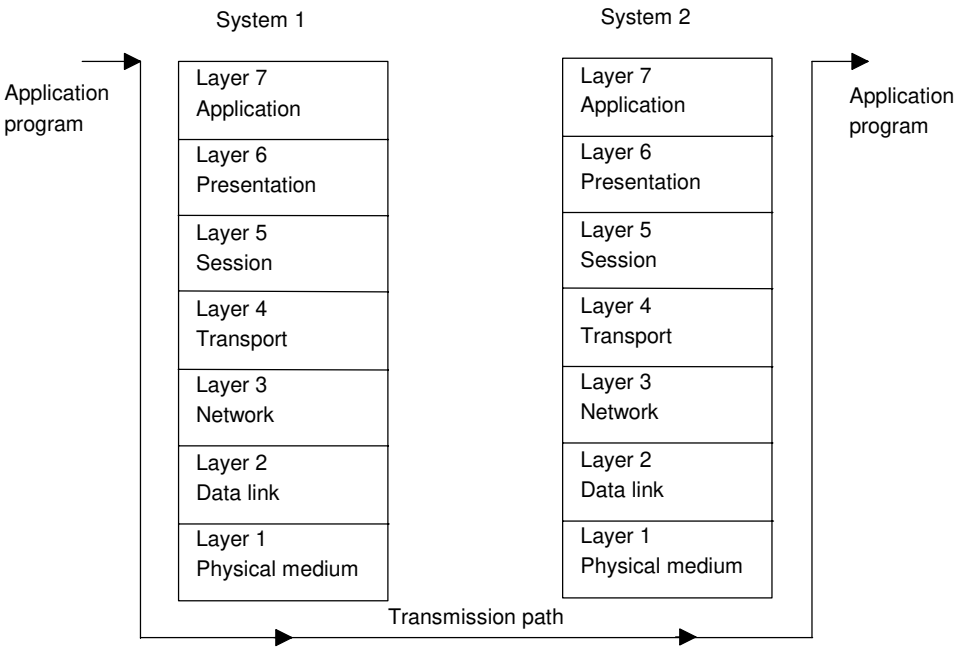


Figure 4.28 *ISO/OSI model*

The function of each layer in the model is:

Layer 1: Physical medium
: This layer is concerned with the coding and physical transmission of information. Its functions include synchronising data transfer and transferring bits of data between systems.

Layer 2: Data link
: This layer defines the protocols for sending and receiving information between systems that are directly connected to each other. Its functions include assembling bits from the physical layer into blocks

and transferring them, controlling the sequence of data blocks and detecting and correcting errors.

Layer 3: Network
This layer defines the switching that routes data between systems in the network.

Layer 4: Transport
This layer defines the protocols responsible for sending messages from one end of the network to the other. It controls message flow.

Layer 5: Session
This layer provides the function to set up communications between users at separate locations.

Layer 6: Presentation
This layer assures that information is delivered in an understandable form.

Layer 7: Application
This layer has the function of linking the user program into the communication process and is concerned with the meaning of the transmitted information.

Each layer is self-contained and only deals with the interfaces of the layer immediately above and below it; it performs its tasks and transfers its results to the layer above or the layer below. It thus enables manufacturers of products to design products operable in a particular layer that will interface with the hardware of other manufacturers.

In 1980, the IEEE (Institute of Electronic and Electrical Engineers) began Project 802. This is a model which adheres to the OSI Physical layer but subdivided the Data link layer into two separate layers: the Media Access Control (MAC) layer and the Logical Link Control (LLC) layer. The MAC layer defines the access method to the transmission medium and consists of a number of standards to control access to the network and ensure that only one user is able to transmit at any one time. One standard is IEEE 802.3 Carrier Sense Multiple Access and Collision Detection (CSMA/CD); stations have to listen for other transmissions before being able to gain control of the network and transmit. Another standard is IEEE 802.4 Token Passing Bus; with this method a special bit pattern, the token, is circulated and when a station wishes to transmit it waits until it receives the token and then attaches it to the end of the data. The LLC layer is responsible for the reliable transmission of data packets across the Physical layer.

4.4.3 Examples of commercial systems

General Motors in the United States had a problem in automating their manufacturing activities by 1990 of requiring all their systems to be able to talk to each other. They thus developed a standard communications system for factory automation applications, this being termed the *manufacturing automation protocol* (MAP). The system was for all systems on the shop floor, e.g. robot systems, PLCs, welding systems. Table 4.2 shows the MAP model and its relationship to the ISO model. In order for non-OSI equipment to operate on the MAP system, gateways may be used. These are self-contained units or interface boards that fit in

the device so that messages from a non-OSI network/device may be transmitted through the MAP broad band token bus to other systems.

Table 4.3 *MAP*

ISO layer		MAP protocol
7	Application	ISO file transfer, MMFS, FTAM, CASE
6	Presentation	
5	Session	ISO session kernel
4	Transport	ISO transport class 4
3	Network	ISO Internet
2	Data link	IEEE 802.2 class 1; IEEE 802.4 token bus
1	Physical	IEEE 802.4 broad band
	Transmission	10 mbps coaxial cable with RF modulators

Note: MMFS = manufacturing message format standard, FTAM = file transfer, CASE = common applications service; each of these provides a set of commands that will be understood by devices and the software used.

For the data link, methods are needed to ensure that only the user of the network is able to transmit at any one time and for MAP the method used is token passing. The term *broad band* is used for a network in which information is modulated onto a radio frequency carrier which is then transmitted through the coaxial cable.

MAP is not widely used, a more commonly used system being the *Ethernet*. This is a single bus system with CSMA/CD used to control access. It uses coaxial cable with a maximum length of 500 m and up to 1024 stations can be accommodated, repeaters which restore signal amplitude, waveform and timing can be used to extend this capability (Figure 4.29). Each station is connected to the bus via a transceiver, the transceiver clamping on to the bus cable. The term 'vampire tap' is used for the clamp on to the cable since stations can be connected or removed without disrupting system operation.

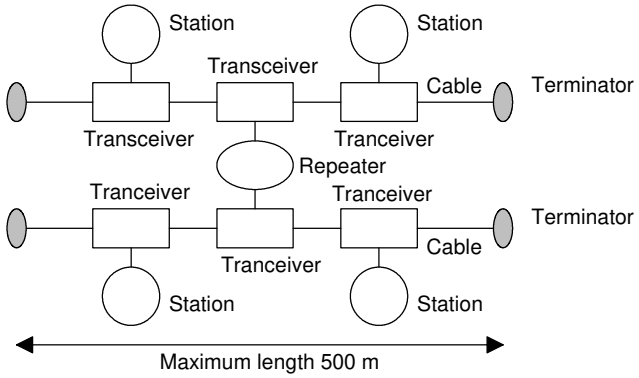


Figure 4.29 *Baseband Ethernet with repeaters. The term base band is used when the signal is transmitted as just a series of voltage levels directly representing the bits being transmitted.*

Ethernet does not have a master station, each connected station being of equal status and so we have peer-to-peer communication. A station wishing to send a message on the bus will determine whether the bus is clear and, when it is, put its message frame on the bus. There is the slight probability that more than one station will sense an idle bus and attempt to transmit. Thus each sender monitors the bus during transmission and detects when the signal on the bus does not match its own output. When such a 'collision' is detected, the transmission continues for a short while in order to give time to other stations to detect the collision and then attempts to retransmit at a later time. Each message includes a bit sequence to indicate the destination address, source address, the data to be transmitted and a message check sequence. The message check sequence contains the cycle redundancy check (see Section 4.3.4). At each receiving station the frame's destination address is checked to determine whether the frame is destined for it. If so, it accepts the message. Ethernet is widely used where systems involve PLCs having to communicate with computers. The modular Allen-Bradley PLC-5 can be configured for use with a range of communication networks by the addition of suitable modules (see Figure 1.15), a module enabling use with Ethernet.

PLC manufacturers often have their own networks, in addition to generally offering the possibility of Ethernet. The *Allen-Bradley data highway* is a peer-to-peer system developed for Allen-Bradley PLCs and uses token passing to control message transmission. The station addresses of each PLC are set by switches on each PLC. Communication is established by a single message on the data highway, this specifying the sending and receiving addresses and the length of block to be transferred.

4.5 Processing inputs

A PLC is continuously running through its program and updating it as a result of the input signals. Each such loop is termed a *cycle*. PLCs could be operated by each input being examined as it occurred in the program and its effect on the program determined and the output correspondingly changed. This mode of operation is termed *continuous updating*.

Because, with continuous updating, there is time spent interrogating each input in turn, the time taken to examine several hundred input/output points can become comparatively long. To allow a more rapid execution of a program, a specific area of RAM is used as a buffer store between the control logic and the input/output unit. Each input/output has an address in this memory. At the start of each program cycle the CPU scans all the inputs and copies their status into the input/output addresses in RAM. As the program is executed the stored input data is read, as required, from RAM and the logic operations carried out. The resulting output signals are stored in the reserved input/output section of RAM. At the end of each program cycle all the outputs are transferred from RAM to the appropriate output channels. The outputs then retain their status until the next updating. This method of operation is termed *mass I/O copying*. The sequence can be summarised as (Figure 4.30):

1 Scan all the inputs and copy into RAM
2 Fetch and decode and execute all program instructions in sequence, copying output instructions to RAM

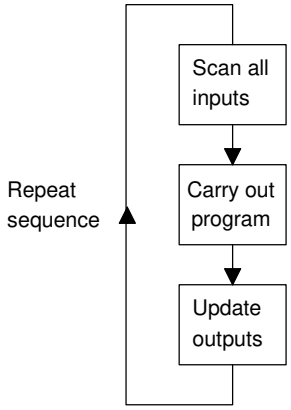


Figure 4.30 *PLC operation*

3 Update all outputs
4 Repeat the sequence

The time taken to complete a cycle of scanning inputs and updating outputs according to the program instructions, i.e. the *cycle time*, though relatively quick, is not instantaneous and means that the inputs are not watched all the time but samples of their states taken periodically. A typical cycle time is of the order of 10 to 50 ms. This means that the inputs and outputs are updated every 10 to 50 ms and thus there can be a delay of this order in the system reacting. It also means that if a very brief input cycle appears at the wrong moment in the cycle, it could be missed. In general, any input must be present for longer than the cycle time. Special modules are available for use in such circumstances.

Consider a PLC with a cycle time of 40 ms. What is the maximum frequency of digital impulses that can be detected? The maximum frequency will be if one pulse occurs every 40 ms, i.e. a frequency of 1/0.04 = 25 Hz.

The cycle or scanning time for a PLC, i.e. its response speed, is determined by:

1 The CPU used.
2 The size of the program to be scanned.
3 The number of input/outputs to be read.
4 The system functions that are in use, the greater the number the slower the scanning time.

As an illustration, the Mitsubishi compact PLC, MELSEC FX3U (see Section 1.4), has a quoted program cycle time of 0.065 μs per logical instruction. Thus the more complex the program the longer the cycle time will be.

4.6 I/O addresses

The PLC has to be able to identify each particular input and output. It does this by allocating addresses to each input and output. With a small

PLC this is likely to be just a number, prefixed by a letter to indicate whether it is an input or an output. Thus for the Mitsubishi PLC we might have inputs with addresses X400, X401, X402, etc., and outputs with addresses Y430, Y431, Y432, etc., the X indicating an input and the Y an output. Toshiba use a similar system.

With larger PLCs having several racks of input and output channels, the racks are numbered. With the Allen-Bradley PLC-5, the rack containing the processor is given the number 0 and the addresses of the other racks are numbered 1, 2, 3, etc. according to how set-up switches are set. Each rack can have a number of modules and each one deals with a number of inputs and/or outputs. Thus addresses can be of the form shown in Figure 4.31. For example, we might have an input with address I:012/03. This would indicate an input, rack 01, module 2 and terminal 03.

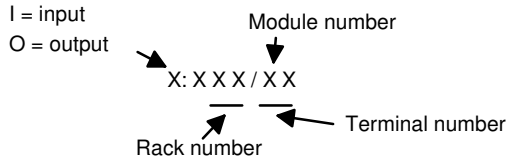


Figure 4.31 *Allen-Bradley PLC-5 addressing*

With the Siemens SIMATIC S5, the inputs and outputs are arranged in groups of 8. Each 8 group is termed a byte and each input or output within an 8 is termed a bit. The inputs and outputs thus have their addresses in terms of the byte and bit numbers, effectively giving a module number followed by a terminal number, a full stop (.) separating the two numbers. Figure 4.32 shows the system. Thus I0.1 is an input at bit 1 in byte 0, Q2.0 is an output at bit 0 in byte 2.

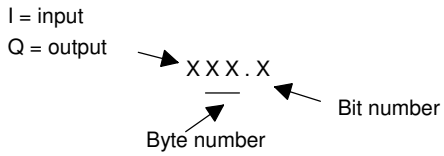


Figure 4.32 *Siemens SIMATIC S5 addressing*

The GEM-80 PLC assigns inputs and output addresses in terms of the module number and terminal number within that module. The letter A is used to designate inputs and B outputs. Thus A3.02 is an input at terminal 02 in module 3, B5.12 is an output at terminal 12 in module 5.

In addition to using addresses to identify inputs and outputs, PLCs also use their addressing systems to identify internal, software-created devices, such as relays, timers and counters.