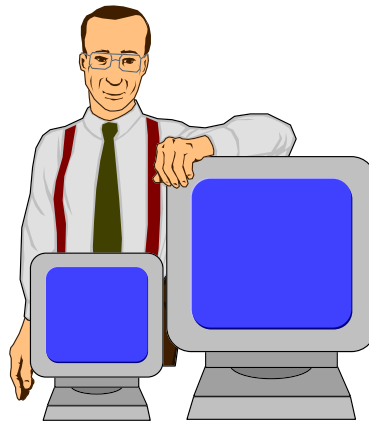
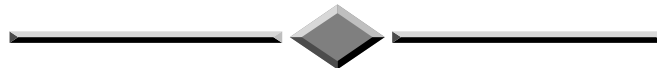


Computer Applications



for Engineers

Student Name

Note: Any working out, that may be required, may be shown in the margins next to the relevant activity.

Contents

Hardware Theory	1
The components of a computer system.....	1
What is a Processor.....	2
The control unit	3
The Arithmetic/Logical Unit (ALU).....	3
Registers	3
Input Units	5
Output units	5
Types of computer output	6
Memory	6
The Bus Subsystem	7
Some of the ASCII codes used:	8
Storage memory units	9
Fast Access (IC) Memory	10
Random Access Memory (RAM)	10
Read only Memory (ROM)	11
Medium access memory	12
Magnetic discs	12
Optical discs	12
CD-ROM.....	12
WORM.....	12
Long Access Memory.....	13
Machine Code Instructions	14
Instruction Sets	14
Instruction Format	14
Some Machine Code Examples	15
Flowcharting.....	18
Flowchart Symbols	18
Converting Hexadecimal (Hex) into decimal	20
Some more of the ASCII codes used:	21
HIGH AND LOW LEVEL LANGUAGES	22
INTRODUCTION	22
OBJECTIVES.....	22
PROGRAMMING LANGUAGES.....	23
Problems and Programming.....	23
Algorithms	24
Programming Language Levels	24
MACHINE CODE.....	24
ASSEMBLY CODE	25
Assembler Actions.....	25
DISADVANTAGES OF LOW LEVEL LANGUAGES	26
HIGH LEVEL LANGUAGES	27
FORTRAN	27
BASIC	28
HIGH LEVEL LANGUAGES	30
High Level Language Translation	30
Interpreters.....	30
Compilers	31

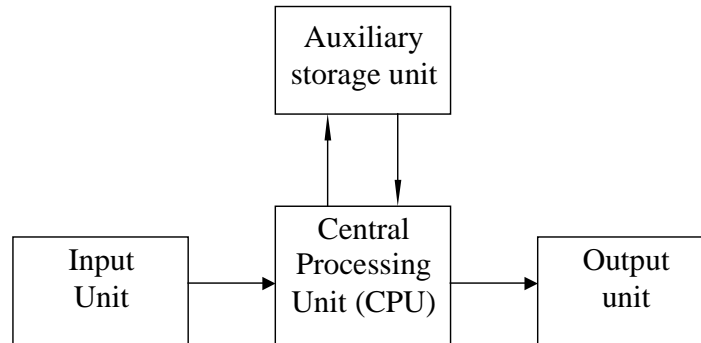
Disadvantages of High Level Languages	31
Memory Space	31
Execution Time.....	32
Isolation From the Hardware	32
Comparison of High and Low Level Languages	32
Some High Level Languages.....	33
SUMMARY.....	34
Engineering data information.....	35
Engineering drawings	36
Relational Databases.....	39
Database Management Systems.....	40
ENGINEERING DATA PROCESSING	41
Management Information	41
Engineering Application Costs	42
Methods of Access to the Computer.....	43
Batch processing.....	43
On-line systems	44
Real-time systems.....	44
SUMMARY.....	45
SOME SPECIFIC APPLICATIONS	46
INTRODUCTION	46
OBJECTIVES.....	46
PROCESS CONTROL	47
Process Controllers	47
Computer Process Controllers	48
COMPUTER-AIDED DESIGN	51
CAD Systems.....	51
Parts libraries.....	53
COMPUTER AIDS TO MANUFACTURE	54
Manufacturing Administration	54
SIMULATION AND PREDICTION	56
Simulations in Computer-aided Design	56
Circuit modelling.....	57
3D modelling	57
Other Simulations	57
SUMMARY.....	58
TOUCH SCREEN	61
LIGHT PEN.....	61
JOYSTICK	61
APPLICATIONS OF Optical Character Readers (OCR).....	61
OPTICAL MARK READERS (OMR).....	62
A typical bar code.....	63
MAGNETIC INK CHARACTER READER (MICR)	63
PRINTERS AND PLOTTERS	65
PRINTERS	65
IMPACT PRINTERS	65
DOT-MATRIX PRINTING	65
SCHEMATIC ARRANGEMENT OF A DOT MATRIX PRINTER MECHANISM.....	66

SOLID FONT PRINTING	66
DAISY-WHEEL HEAD.....	66
BARREL PRINTERS.....	66
Layout of a Barrel printer	67
GOLF-BALL HEAD.....	67
Non-Impact Printers.....	68
THERMAL PRINTERS	68
ELECTROSENSITIVE PRINTERS	68
INK-JET PRINTERS.....	68
LASER PRINTERS.....	69
PLOTTERS	69
Schematic layout of a Flat Bed plotter.....	69
Schematic layout of a Drum plotter.....	70
PRINTERS AND PLOTTERS SUMMARY.....	71
Paper feed mechanisms.....	72
REQUIRED COMPUTER FACILITIES.....	73

Hardware Theory

The components of a computer system

No matter what type of computer whether simple or complex we can reduce it down to the same simple structure, as shown in the diagram below.



The CPU is the core of the computer. All the processing is done in the CPU and from here all the elements of the system are controlled.

- The Auxiliary storage unit is the data storage area. It holds the users permanent or semi-permanent data. (Note this also includes the programs which cause the computer to do the given task.)
- Instructions to the computer and data enter the system from the outside world and have to be turned into a form that the computer can understand. The channel for that is provided by the input unit.
- The output unit does the opposite job. The responses to the instructions from the user have to be converted into a form suitable for human comprehension typically this is in the form of text or pictures on paper or screen.

The input and output units are usually lumped together and referred to as the Input/Output subsystem or just 'I/O' this is usually a separate unit from the motherboard (the board on which the CPU sits) and is built on a separate board or 'I/O' card that plugs into the motherboard. I/O cards take several forms and could be a video or a sound card for example.

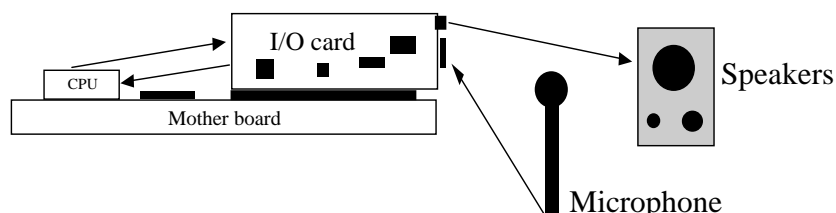


Diagram showing information flows of the system

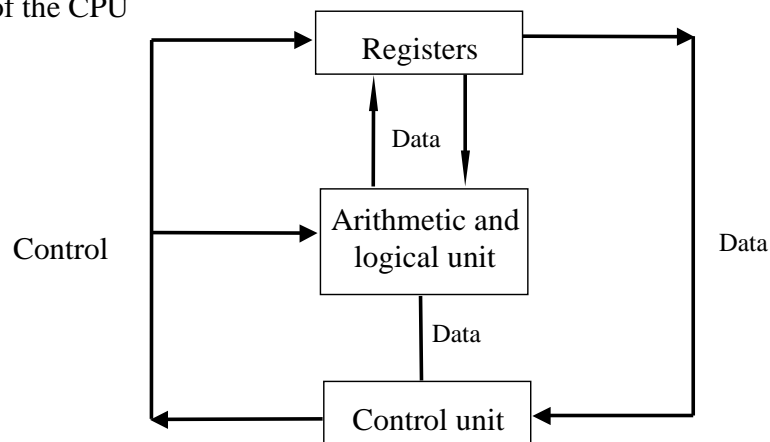
What is a Processor

The CPU, is a functional unit that interprets and carries out instructions. This is the central nervous system of the computer, and is often contrasted to the devices that surround the processor and memory, such as keyboards, display screens, and disc and tape drives, which are known as peripherals.

The CPU is essentially the heart of the computer. Anything that happens anywhere in the computer or in any device connected to it, is directed by the CPU.

The CPU works sequentially, in other words it completes one operation before starting the next. The operations to be done are specified by the instructions which make up the program being carried out by the computer. Even when the computer appears to be carrying out multiple tasks it is allocating time slices to each of these and runs the 'CODE' sequentially. The speed of the processor makes it appear as though all tasks are being carried out simultaneously. However, this is not so.

The block structure of the CPU



For simplicity, you can regard the CPU as being divided into three blocks, as shown in the diagram above.

A timer called the clock releases precisely timed electrical signals that provide a regular heartbeat for the processors work. This is measured in Megahertz (MHz), which means million cycles per second, are used to measure the computer's speed. For example a 'Pentium 200' processor's clock ticks at 200 MHz or 200 million times every second. A processor is composed

of two functional units a control unit and an arithmetic logic unit as well as a set of special workspaces called registers.

The control unit

The control unit is a functional unit that supervises the operation of the processor in some ways, it is analogous to an intelligent telephone switch board because it makes connections between various functional units of the computer system. It calls into operation each unit that is required by the program currently in operation and like a switchboard with call waiting the processor can be interrupted.

The control unit receives instructions from memory and determines their type or decodes them. It then breaks each instruction into a series of simple small steps or actions. By doing this, it controls the step-by-step operation of the entire computer system.

The Arithmetic/Logical Unit (ALU)

The ALU is the functional unit that provides the computer with logical and computational capabilities. Data is brought into the ALU by the control unit, and the ALU performs whatever arithmetic or logical operations that are required to help carry out the instructions.

Arithmetical operations include adding, subtracting, multiplying and dividing. Logical operations make a comparison and take action based on a result. For example, two numbers might be compared to determine whether they are equal. If they are equal, processing will continue; if they are not equal, processing will switch to another instruction.

Registers

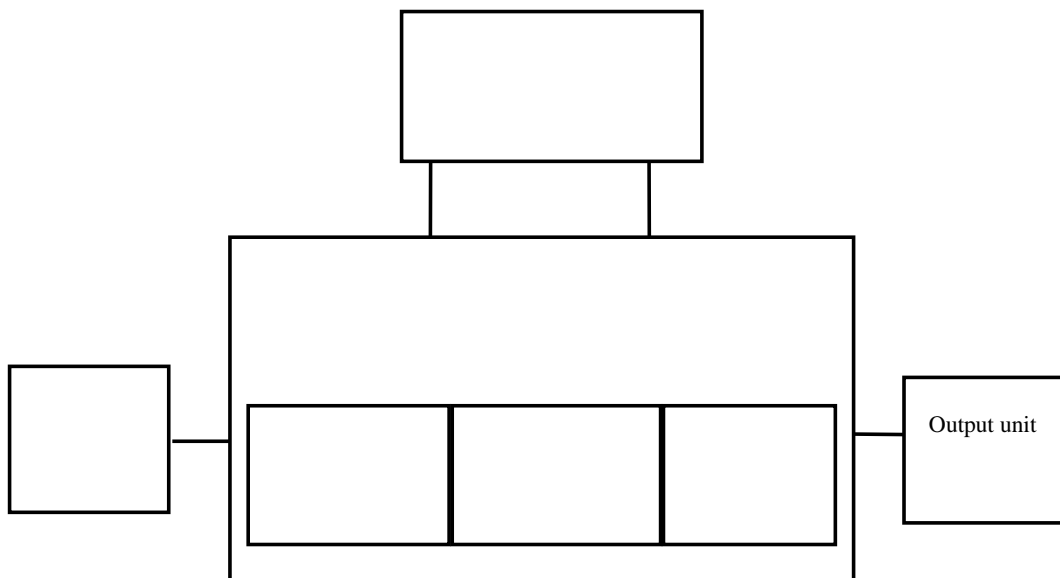
A register is a storage location inside the processor. Registers in the control unit are used to keep track of the overall status of the program that is running. Control unit registers store information such as the current instructions and location of the next instruction to be executed.

In the ALU, registers store data items that are added, subtracted, multiplied, divided or compared. Other registers store the results of arithmetic and logic operations.

Activity 1

1. Label each of the boxes shown in the diagram shown below.
2. Draw arrows on the lines connecting the boxes to show the different direction of data flow.

To start you off, the output box has been done for you



Input Units

An information system, generally, is activated by the input of data. Inputs can not only activate processing they can also deactivate it, in the same way as a light switch can both turn on and off a light. The primary role of an input unit is to **enable data** entry.

The input unit is a peripheral device (not part of the main computer) that is connected to and controlled by the computer.

Input is a process that involves the use of a device to encode or transform data into digital codes that the computer can process. For example, if you press the letter A on the keyboard of the terminal or a personal computer, you activate an information processing cycle. The key is simply a switch that senses a finger touch and triggers a cycle to accomplish the following.

1. The key stroke is encoded and converted into machine readable code,
2. The encoded piece of data is stored in a memory location for latter processing,
3. Output is provided by displaying the letter A on the computer's monitor.

Typical input units

Bar code reader	Magnetic ink character recognition (MICR)
Magnetic card reader	Mouse
Digitiser/ Digitising tablet	Optical mark reader (OMR)
Document reader (or OCR)	Point of sale terminal (POS)
Electric communication (network)	Terminal
Image processor	Touch sensitive screen
Joystick	Voice activated
Keyboard	Light pen

Output units

The information that a computer produces is no more than a stream of coded symbols. In the majority of cases it is the job of the output device to decode these symbols into a form of information that it easy for humans to comprehend, such as text, pictures or sound.

However, there is an exception to this rule in the case when the output is not intended for human consumption. A good example of this is saving a file to disc for latter input to the computer.

Another example would be the PLC sending its output to control machines instead of providing machine operators with the information.

Types of computer output

The four major types are:

- Text, consisting of words, numbers, and other symbols in the form of written language.
- Images, in the form of video, graphics and animation.
- Sound, that consists of music or voice.
- Machine readable data, which are symbols in a form that can be interpreted by other computer systems or machines can use.

Of these four output device it is text that is still the most commonly used as a means of communication, and is the easiest of the output technologies to implement. However, as technology progresses, both input and output devices are becoming more varied.

Buggy	Light Emitting Diodes (LED)
Computer Output on Microfilm (COM)	Machine tools
Liquid crystal Display (LCD)	Plotters
Printers	Robots
Voice	Visual display unit (VDU)
Terminal	

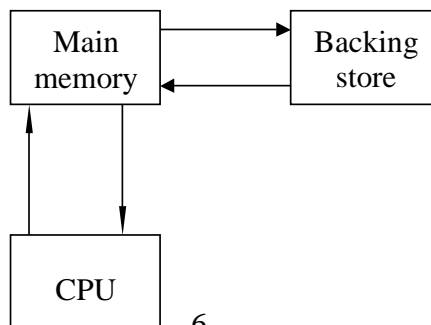
An output unit is a Peripheral device that is connected to and controlled by a computer to enable data to be interpreted by human's or machine readable to control other machines.

Memory

The computer memory is organised so that items which are being processed at any one time are held in relatively small, fast memory. These items are drawn from slower, but very large memory which holds all the data. The arrangement of main memory and backing store is shown in block form below.

Diagram showing the relationship between the

CPU Main memory and Backing store



The Bus Subsystem

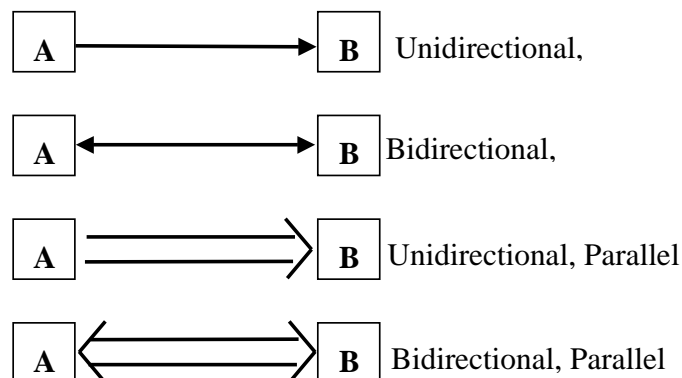
We have seen diagrams that have shown lines indicating information flows, in a primitive way they indicate the bus subsystem of the computer.

A Bus is simply an electrical path connecting the different components of a computer system; typically, they are wires. They are classified as being either unidirectional or bidirectional depending on whether signals pass in only one direction or both. The signals are usually in binary form corresponding to the bits which are being transferred from one component to another.

The 'bus' is also classed as having serial or parallel transmission capability. A serial bus can carry only one signal at a given time. A parallel bus can carry several signals at the same time.

When information is coded as groups of coded signals, the byte, the parallel bus offers greater processing speed, since all signals are transferred at the same time. In the case of the byte the serial but would take eight times longer to transmit the information than a parallel bus.

Bus Symbols and
Types



Some of the ASCII codes used:

Keyboard character	ASCII Code	Keyboard character	ASCII Code
Space	32	M	77
,	44	N	78
.	46	O	79
A	65	P	80
B	66	Q	81
C	67	R	82
D	68	S	83
E	69	T	84
F	70	U	85
G	71	V	86
H	72	W	87
I	73	X	88
J	74	Y	89
K	75	Z	90
L	76		

Binary code value's

128	64	32	16	8	4	2	1
1	1	1	1	1	1	1	1

How messages are stored in memory		
Binary code	ASCII code	Decimal value
01010100	T	84
01001000	H	72
01000101	E	69
00100000		32
01000010	B	66
01001001	I	73
01000111	G	71
00100000		32
01000100	D	68
01001111	O	79
01000111	G	71

The Letter A Corresponds to:

65 (in ASCII)

1000001 (in binary)

Convert the following code into a ASCII

01010100 01001000 01000001

01010100 00100000 01010111

01000001 01010011 00100000

01000101 01000001 01010011

01011001

0110 0101(binary coded decimal (BCD))

Answer

Storage memory units

Memory storage is usually required in large volumes. It would therefore be cumbersome to refer to 'thousands' or 'millions' of byte's. Hence two large units of measurement are employed.

Using standard prefixes i.e.

1 Kilobyte equals 1024 bytes

1 Megabyte equals 1048,576 bytes

The Kilobyte and Megabyte are shortened further to Mbyte and Kbyte respectively.

There are many devices for the storage or memory (the two terms being interchangeable).

However, here we shall only consider those in principle use. Firstly there are two types of access that must be defined sequential access memory (SAM) and random access memory (RAM)

SAM scheme places items in succession, each new item being placed further down the line from the start. the time taken to retrieve data is determined by its position in the store. A typical SAM scheme would be a tape drive.

Within RAM any item of data takes the same time as any other item. The position of the data causes no delay.

Fast Access (IC) Memory

The IC chip has a matrix of cells, each cell holding one bit, which may be 0 or 1. It has a random access structure. The address from the CPU is decoded, giving the location for reading or writing the item of data. However, to you as a user, the addresses are defined simply as a sequence of numbers between zero and the maximum storage value. Typically, a memory chip inputs or outputs is eight bits in one access. However, this may be 16 or 32 bits in one access.

Random Access Memory (RAM)

The internal memory is used for storing programs and data currently being processed. This memory is referred to by a variety of names e.g.

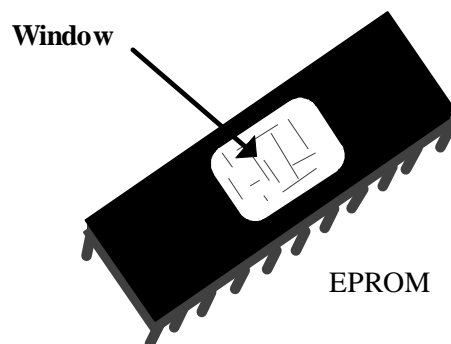
- main memory
- primary storage
- immediate access storage
- main store

The part of memory used to store programs and data currently being used is called RAM. This term is used for memory that can be both read from and written to. The time required to access any part of RAM is the same for any location in memory. RAM must be seen as being *volatile* memory; it only retains memory whilst the computer is switched on. This is the reason that all programs and data must be saved and stored on magnetic disc or tape.

Read only Memory (ROM)

Read only memory as the name implies can only be read but not written to. This means it can be accessed but not changed. It is therefore seen as a permanent storage medium. However, it should be noted that like RAM the ROM can be accessed randomly. ROM is termed **non-volatile** memory; **not being lost when the machine is turned off**. The ROM area is used to hold the operating system of the computer, which is all the data and instructions which the computer requires to be able to function.

Programs stored in ROM are permanently stored in memory, ready for when the computer is switched on. For example the instructions that tell the computer how to display text and how to read the keyboard or disc units. The information held in ROM chips are fixed into it during manufacture and can only be changed by physically changing the chips involved. You may come across other forms of ROM, such as the EPROM (erasable programmable ROM) and the PROM (programmable ROM). These are designed to allow the user rather than the manufacturer, to be able to program the chip, or even change its contents after they have been entered. Both the EPROM and PROM are classed as having non-volatile memory. They do not lose their contents when the machine is switched off. However, the contents may in the EPROM's case be changed or erased using ultraviolet light aimed at the chips window.



Medium access memory

Magnetic disc's provide medium access memory but are quickly being supplemented by optical disc's.

Magnetic discs

Contrasting with RAM magnetic disc provides a permanent but changeable medium for data storage. The data does not vanish when the machine is switched off. It permanently remains until it is purposely over written. The capacity is much larger at a lower cost than RAM. However, data access times are much longer than 'fast access memory'.

Information is magnetically recorded from its binary form on the surface of a rotating disc contained in a disc drive. There are two varieties the 'hard' disc which is a metal disc usually sealed in a chamber and the floppy disc; which used a disc of plastic coated in a recordable medium; which can be removed and re-inserted.

Capacities of hard drives vary from approximately 1 Gbytes to 10 Gbytes; in comparison most floppy disc's vary from about 360 Kbytes to 1.4 Mbytes

Optical discs

The compact disc (CD) acts as an ideal storage medium in that it can store vast amounts of data. The optical disc can hold from around 200 Mbytes to 4,000 Mbytes, depending on the physical size of the disc. There are two types of optical disc : the CD-ROM and the WORM

CD-ROM

As the name suggests is a read only device. Discs, like ROM are filled with data at their manufacturing point. The drive is designed to allow the disc to be removed and replaced with another. Thus, the CD storage capacity is in principle unlimited.

WORM

WORM is an acronym of Write Once - Read Many (times) which also describes the disc's purpose.

It is an optical disc on which the user can write data once . The data cannot be changed once written, so you must insure the information is correct. This kind of data medium could be used for situations where there is a lot of unchanging data to be recorded, perhaps with a much smaller volume of data which changes relatively slowly.

Long Access Memory

Magnetic tape has unlimited storage capacity - you just use another reel when the present one runs out of space. Records are made sequentially on data tape; just as in audio tape. If the data you want is at the very end of the tape, it can't be retrieved until the tape has unwound to that point. Should the data be on a tape that is not on the drive unit the wait would be even longer.

ACTIVITY 3

You are using a magnetic tape which is 3,600 feet long and contains records numbered 1 to 40,000. Each record is 800 bytes long and is separated from the next record by a gap of equal size to the record. Together with its gap the record takes upto one inch on the tape.

The tape moves at five feet per second. If your program calls for the CPU to read in record number 31,421 approximately how long does the CPU have to wait for the record to become available.

Answer

Machine Code Instructions

For the computer to be able use the information we supply it needs to be translated into machine code. We will now take a brief look at the nature of that form.

Instruction Sets

Computers work in a step by step fashion, finishing one instruction before beginning another. The instructions therefore are kept simple; each instruction directing the CPU to perform one operation. A typical operation would be adding one number to another, writing a piece of data into main memory or reading one character typed on the keyboard.

All the individual operations a computer is capable of executing are defined in its instruction set. The details of these instructions vary from one make of computer to another. However, the instructions do much the same job.

Instruction Format

For an instruction to do its job, it has to specify two things

- What to do with the data
- Where to find the data

Thus the general format of an instruction is basically very simple. It has a two part structure:

OPERATION\OPERAND

The **'operation'** is the simple operation to be carried out: for example, add, subtract, write, or read.

The **'operand'** part specifies the data on which the operation is to be performed. Usually it gives the address (in memory) of the data, but sometimes it does hold the actual data.

The computer works in binary, so the instructions in their machine code form are coded in binary. The size of an instruction ranges from two to six bytes, so a two part structure is not always easy to detect.

Some Machine Code Examples

Microcomputers are often described as 8 bit or 16 bit machines, or even higher bit. The bit refers to the largest bit that the computer (CPU) can process as any one unit. For example an 8 bit computer can do an arithmetic operation on an 8 bit binary number. The equivalent instruction on an 16 bit machine will do the same operation on a number which is twice as big.

The operation/operand structure of the 8-bit machine code can be seen from the following examples taken from the instruction set of the widely used microprocessor, the 6502. It can address a 64 Kbyte memory

Read data - a three byte instruction

Load byte from the main memory address 1,638 decimal into accumulator (a CPU register):

1st byte	2nd byte	3rd byte
OPERATION	OPERAND	
10101101	01100110	00000110

If you carry out the conversion of 1,638 into binary you may be wondering why the operand is apparently wrong. You would expect to see 00000110 01100110. The reason for this is that the microprocessor requires the address contained in the instructions to be presented in the reverse order -low byte followed by high byte.

The loading instruction destroys whatever was in the accumulator and replaces it with the new data. the data in the memory address remains unchanged.

Write data- three byte instruction

Store the number in the accumulator in main memory address 7,951 decimal:

1st byte	2nd byte	3rd byte
OPERATION	OPERAND	
10001101	00001111	00011111

You should not that the writing operation destroys the previous contents of memory in the address. The data in the accumulator is not effected.

Add a number -a three byte instruction

Add the contents of the number held in main memory address 2,456 decimal to the number already contained in the accumulator.

1st byte	2nd byte	3rd byte
OPERATION	OPERAND	
11101101	10011000	00001001

The contents of the memory address are not changed. The information in the accumulator is changed; a write operation or other operation would then be performed. before the accumulator is loaded again otherwise the addition would be lost.

ACTIVITY 4

Write the operation and operand to write the data stored in the accumulator to address 3,106

Answer

ACTIVITY 5


Consider the following specification

Memory address 1723 holds a number x ; 3679 contains the number y . Write a program to add them together and put the result in address 2466. Finally replace the number in 1723 with $2(x + y)$

Use the three operational codes you know with the appropriate addresses to write a program.

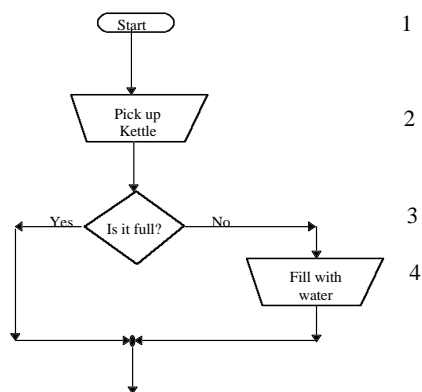
Start by loading one of the numbers into the accumulator.

Write each instruction on a fresh line and write at the side of the instruction a brief comment on what is being done. Keep the comments short, for example “x is added”, “result is stored in memory”.

A large, empty rectangular box with a thin black border, intended for students to write their instructions and comments. The box is currently blank.

Flowcharting

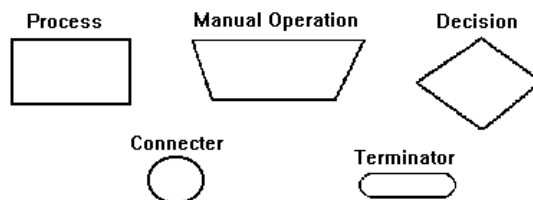
Specifying clearly and completely the steps in an operation is not simple, even if the operation itself is. However, there are tools to help. The earliest technique to assist in the job of specification that was applied to computer work was flowcharting. Its method is describing the process using block diagrams. You use flowcharts to show the flow of activity and control through any process or procedure. An example of a simple flow chart is given below.



Flowchart Symbols

Each action is briefly described inside the symbol whose shape indicates the nature of the action contained. The boxes are joined using lines bearing arrows that indicate the direction in which the sequence flows. If necessary symbols can be identified by numbers or letters or both, written at the side.

Flowcharting has been standardised for many years. There is a British Standard, BSI 4058, which defines the symbols to be used for programming and related activities. A small selection can be seen below



- Process** - any process carried out at computer speed
- Manual operation** - any process carried out at human speed
- Decision** - on which of two or more alternative paths to take
- Connection** - used to avoid crossing lines on a diagram. A letter or figure inside the terminating connector is matched to the same one on the continuing connector.
- Terminator** - used to show 'start' or 'end' of flow chart or some part of it.

Arrows connect the symbols to show the direction of flow.

ACTIVITY 6

If you look at the diagram on page 18 you may realise that the box labelled 'Fill with water' hides a lot of operations. For example, the kettle must be taken over to the tap.

Expand the box labelled 'Fill with water' to include the chief actions. Draw your flowchart going down the page, not across it. Keep the original flow chart to roughly its original size. Number your new boxes 4.1, 4.2 and so on.

Converting Hexadecimal (Hex) into decimal

The sixteen symbols 0 to F, form the digits of a number system based on base 16, hence the name of hexadecimal (6+10)

One of the uses of hex is to get rid of the seemingly endless strings of 1's and 0's you encounter in binary.

To convert hex numbers like 74 into decimal, note that the digits represent powers of 16.

$$\mathbf{74 \text{ hex} = 7 \times 16 + 4 \quad \text{decimal} = 112 + 4 = 116}$$

The “7” and “4” above were hex numbers, but they correspond to the same digits in decimal. With a hex number like 6D you would work in the same way, but substituting the decimal equivalent for the “D”.

$$\mathbf{6D \text{ hex} = 6 \times 16 + 13 \quad \text{decimal} = 96 + 13 = 109}$$

Some more of the ASCII codes used:

Keyboard character	ASCII Code	Keyboard character	ASCII Code
^	94	m	109
_	95	n	110
`	96	o	111
a	97	p	112
b	98	q	113
c	99	r	114
d	100	s	115
e	101	t	116
f	102	u	117
g	103	v	118
h	104	w	119
i	105	x	120
j	106	y	121
k	107	z	122
l	108		

ACTIVITY 7

Work through the message you would get converting the following hex into decimal and then into ASCII

Hex	74	68	69	73	20	69	73	20	6D
Decimal									
ASCII									
Hex	6F	72	65	20	64	61	74	61	
Decimal									
ASCII									

HIGH AND LOW LEVEL LANGUAGES

INTRODUCTION

In this section, we will look in more detail at programming. When you write a program, you are solving a problem. We communicate our instructions to the computer using programming languages. These are grouped into two distinct levels - low and high. The lowest level is what the computer understands, that is, the machine code you met on page 14. Above that is assembly code, which makes writing machine code a little easier.

High level languages (HLL) are designed to meet human need - they can be easily used or understood by people. A complex translation process is required to convert them into a machine code language. The conversion is carried out by utility programs called compilers and interpreters.

You will see small samples of HLL programs and will have the chance to write a short program yourself in the language called BASIC. We will look at the advantages and disadvantages of each of the two levels of languages.

We will conclude by listing some major high level languages and the fields in which they are principally used.

OBJECTIVES

When you have completed this section you will be able to:

- describe programming as a problem-solving activity
- define what is meant by a 'language' in programming
- describe principal features of assembler languages and the relationship to machine code
- describe the principal features of and the reason for using high level languages (HLLS)
- give examples of short HLL program using BASIC
- describe translation functions of interpreters and compilers
- describe the generations of computer languages
- list the principal HLLs and their primary applications.

PROGRAMMING LANGUAGES

A natural language is a means of communication between people. It can be used to convey from one person to others anything from abstruse philosophical concepts to abstract emotions to orders for things to be done.

There are two forms of any natural language: the spoken, which vanishes when the sound dies away, and the written, using symbols to represent the sounds, which is relatively permanent.

For communication to be effective, both the speaker and the listener must know the language being used. A dialogue between an English person and a Chinese won't carry much information or meaning if each speaks only his or her own tongue.

Also, so that all speakers can attach the same meanings to words and groups of words, each language must have a set of rules, its grammar and its syntax, to govern how words, phrases and sentences may be used.

To instruct our computers in what we want them to do, we have to employ some formal method of communication with them which both computer and human can understand. In other words, we must have a common language of communication.

Problems and Programming

Programming is a problem-solving activity. We delegate more and more of our problems to our computers to handle, especially those things we find difficult, time-consuming or boring.

Problems which we hand over range from the solution of simple equations to control of factory production lines to enhancement of images from far distant planets.

There are two parts to the solution of any problem by a computer.

The first part being working out how to solve the problem and the second preparing the instructions for the computer to work out the actual solution.

Algorithms

In most cases, when you describe how to solve a problem, you are really specifying how to do a task of some kind. You set down a method which consists of a sequence of simple steps. If the sequence is followed faithfully, it will always reach the desired result. You have written what is termed an algorithm

Algorithms are not exclusive to computer programs. You will find that algorithms also abound in the everyday world.

PROGRESS QUESTION

Bearing in mind that an algorithm describes a sequence of steps to carry out a task, list some everyday examples of algorithms.

As your examples of algorithms listed in answer to the Progress Question might have suggested to you, the flowchart may be a useful tool in the design and construction of algorithms.

Programming Language Levels

By convention, programming languages are divided into two classes: *low level and high level*

The low level languages (LLS) are those closest to the computer's own internal code. The high level languages (HLLS) are very remote from the computer's understanding and have the aim of simplifying the programmer's task. HLLs are modelled as far as practicable on a natural language. In most cases, the language is English and is independent of the machine on which it is used.

MACHINE CODE

The computers internal code is the lowest of the LLS. You have already written a very small program in a machine code and have probably decided that you would not like to write a large

one. Even using hex rather than pure binary notation, writing a program in machine code is still a chore.

Quite early in the history of computing, the use of machine code was assigned strictly to the machines themselves. People would use something more friendly.

ASSEMBLY CODE

The 'something more friendly' is *assembly* code. Since (in 6502 code), AD means 'load', why not replace it with a mnemonic form we can understand, like 'load', say, and let the computer translate it back into AD? This is the principle of assembly code- mnemonic operators and mnemonic operands.

As in the case of the files put away by the operating system, we don't care where the data is stored just so long as the program can always find it. All you need do is give the data a name when it is first stored and ever after refer to it by that name. The computer can work out the addresses - you don't need to know them.

Each instruction in assembly code will translate to exactly one machine code instruction. The program which does the translation into machine code is termed an assembler. An assembly code is said to have a one-to-one correspondence with machine code. An assembly language gives a much better way of communicating your directions to a computer than machine code.

The 6502 mnemonics for the instructions you know are:

AD LDA (LoaD Accumulator)

6D ADC (AdD with Carry to accumulator contents)

8D STA (STore Accumulator contents).

Assembler Actions

An assembly language has a syntax, just as a natural language has. That is to say, it has rules about the way in which instructions may be assembled. For example, certain kinds of operation may take only specific types of operand.

The assembler reads your program and for each instruction:

- checks that it is syntactically correct
- replaces the mnemonic with the equivalent operation code
- for data being read by the program, for example, x, y, it translates the operand name into the memory address the data actually occupies
- for data computed by the program, for example, x plus y, it replaces the operand name with the address the data will occupy when it exists.

It provides a list of your program instructions with your original coding and the equivalent machine code. This list is called a *program listing*. The listing also shows any syntax errors you may have made. It will also indicate some other kinds of error, for example, giving the same name to two (or more) different items of data.

The assembler also produces the machine code for the computer to execute, *but only when you have eliminated all the mistakes it has advised to you.*

DISADVANTAGES OF LOW LEVEL LANGUAGES

You have already encountered one disadvantage of machine code - it is very unfriendly in form. Binary and hex are not natural to human thinking. Moreover, you have no indication of any errors until you execute the program. Locating an error then is quite a tough job.

Assembly code is an improvement, even allowing some errors to be removed before the program gets its first trial run. However, assembly code is still not satisfactory. In common with machine code, it has several drawbacks.

The languages are far removed from the terms in which the problem is expressed. Look at Figure 1. On the left the task to be done is described in a flowchart box. On the right is an assembly program to implement it. They are not obviously related.

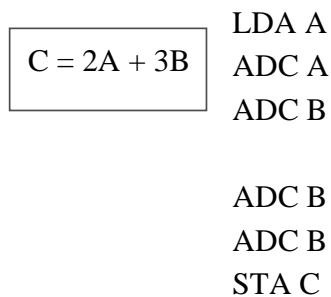


Figure 1

Even for this very simple case, you have to think out the algorithm with some care. The language forces attention away from the problem to be solved and focuses it on the details of the program.

The languages are difficult to read. You cannot grasp quickly what even the little program in Figure 1 does. You have to spend time and care to work out the actions.

Low level languages offer plenty of scope for error, since you have to deal constantly with minute detail.

Every make and type of computer has its machine code which is different from any other computer's code. The machine code of an IBM PC will not run on an Apple Macintosh computer. Since the assembler is tied to the machine code, it follows that assembly programs can be executed only on the computer for which they were originally written. This is an expensive restriction. If you change to a new brand of computer all your assembly programs for the old machine will have to be rewritten for the new.

HIGH LEVEL LANGUAGES

You will see that there is a need for a language which will

- simplify our programming by matching the terminology of our problems
- be executable on any computer, regardless of make, with little or no change.

The high level languages (HLLS) provide an answer to the requirement. One of the earliest HLLs is FORTRAN. It dates from the mid-fifties but is still in major use today. It was designed to simplify the programming of the computational problems encountered in science and engineering.

FORTRAN

FORTRAN (FORmula TRANslation) uses what it calls *statements*, not instructions, for its smallest program building blocks. A FORTRAN statement does the work of several assembly instructions.

For example, the polynomial:

$$y = ax^4 + bx^3 + ex^2 + dx + e$$

is evaluated by the single FORTRAN statement:

$$Y = A*X**4 + B*X**3 + C*X**2 + D*X + E$$

Once you know that '*' means 'multiply' and '**' stands for 'raise to the specified power', you will agree that the FORTRAN statement is a very good match to the terms of the problem.

BASIC

The HLL that is probably in the widest use today came into being about a decade after FORTRAN. BASIC (Beginners All-purpose Symbol Instruction Code) was designed to make a high level language easy to use by anyone, not just professional programmers. Today, primary-school children are taught its rudiments.

BASIC also has statements as building blocks - in fact, most HLLs do. The polynomial we used for the FORTRAN illustration could appear in BASIC as:

$$100 Y = A*X^4 + B*X^3 + C*X^2 + D*X + E$$

Note the two differences from the FORTRAN statement: the '**' is replaced by '^' and there is a number on the left. This number is the statement number. Each statement, in early forms of BASIC must have a unique number to identify it.

The following is a short BASIC program to work out the expression in the box in Figure 1 and print the square root of the result.

```
10 C = 2*A + 3*B
20 ROOT = SQR(C)
30 PRINT ROOT
40 END
```

You should note particularly the right-hand side of statement 20. HLLs have a number of utilities built into them. In this case the job of taking the square root is done for you. You call the function named SQR with the name of the item whose square root you want and tell BASIC

the name of the place for -the result. The square root is always returned as a positive number.

PRINT is another labour-saving device - a command that can be used inside your program.

The END statement (Number 40) is required to tell the computer that your program ends here.

You could shorten the program.

```
10 C = 2*A + 3*B
20 PRINT SQR(C)
30 END
```

You could make it shorter still.

```
10 PRINT SQR(2*A + 3*B)
20 END
```

A high level language is a flexible tool.

Like assembly code, each HLL, and there are many more than just FORTRAN and BASIC, has a set of syntax rules.

PROGRESS QUESTION

In addition to '*' and '+', BASIC provides the other arithmetic operators: '-' for subtraction and '/' for division. Write a BASIC program to take the square root of $(B^2 - 4AC)$. Store the result as SR. Print the value of $(SR - B)$ divided by $2A$.

HIGH LEVEL LANGUAGES

You will see that there is a need for a language which will

- simplify our programming by matching the terminology of our problems
- be executable on any computer, regardless of make, with little or no change.

The high level languages (HLLS) provide an answer to the requirement. One of the earliest HLLs is FORTRAN. It dates from the mid-fifties but is still in major use today. It was designed to simplify the programming of the computational problems encountered in science and engineering.

High Level Language Translation

As you have seen from these simple examples, an HLL program has nothing to do with any machine code. It could therefore be executed on any computer provided there is a means of translating the HLL statements into the necessary machine code. An HLL is described as being *portable or machine-independent*.

Two methods are used for this translation, employing special utility programs called *interpreters and compilers*.

Interpreters

An interpreter takes each HLL statement and converts it automatically into the machine code for the target computer and then executes that segment of machine code. The interpreter then moves on to the next statement and repeats the operation. It continues like this until the whole program is executed.

BASIC is an interpreted language. Having written your BASIC program you could execute it on an IBM or compatible PC, if the operating system has a BASIC interpreter in its library. You could take that same program to an Apple Macintosh and run it, provided that an interpreter is present.

Interpreting means slow execution because the statement translation process adds its own time to the code execution. However, errors are notified when they occur, and can be corrected immediately.

Compilers

A compiler translates the whole program into the appropriate machine code, putting what is called the *object code* program into store at the end of the process. It is this program which is executed by the computer. As you would expect, the execution time is much faster than that of an interpreted program. FORTRAN is a compiled language.

The compiler reports any errors it detects in your program at the end of the compilation. After you have corrected them, the whole compilation must be repeated. This could happen several times before the program is ready to run, so the compilation can sometimes be rather protracted and tedious in comparison with interpretation. However, if the program is to be executed on a regular basis, then compilation is the best option. The execution time is much shorter.

As in the case of the interpreter, you need a compiler for each computer on which you wish to run your HLL program.

Disadvantages of High Level Languages

There are three principal disadvantages which can arise with the use of HLLS. They are concerned with:

- memory space
- execution time
- isolation from hardware.

Memory Space

The object code is likely to take up more memory space than a human-written version would. The same code will be produced for any element each time it appears, even though there may be better ways of coding the element on different occasions.

In a major application, the memory space available may be limited. Some programs may then become what is termed *space-critical* and will have to be written in the low level language

instead of the HLL in order to fit in the available memory. The non-critical parts will still be programmed in the HLL.

Execution Time

Automatic translation can bring about time problems too. More instructions mean more time to execute. Moreover, the translator choice may have been more concerned with memory space than time. A completely different choice of instructions to do the same job may speed it up considerably.

Where a signal has to be accepted from a peripheral in a very short time, the program to do it is likely to be *time-critical*. As in space-critical situations, the time-critical program may have to be written in the LLL instead of the HLL, to respond in time.

Isolation From the Hardware

Most programmers and users do not need access to the hardware other than reading and writing. However, the systems programmers, who write the operating systems and the utilities, do need to be able to drive the peripherals directly. High level languages do not offer such capabilities. Hardware does not really exist for an HLL.

The solution to the problem is again to write the bulk of the program in the chosen HLL and link it with assembly code written for the parts which operate or control the hardware.

Comparison of High and Low Level Languages

The chief features of the different levels of language are compared in Table 1.

Ease of programming	HLL Good	Assembly Poor	Machine V. poor
Degree of liability to programmer error	Less than LLLs	Severe	V. severe
Memory required	Can be higher than LLLs	Can be compact	
Execution speed	Can be lower than LLLs	Best possible	
Ease of finding non-syntax faults	Difficult	More difficult	
Cost of program development	Can be less than LLLs	High	
Portability	Good	Non-existent	

Table 1

You should note that the comparisons in Table 1 are not absolute. For example, a HLL program could execute much faster than a sloppily written assembly program.

- Finding a non-syntax fault in an HLL program may sometimes require studying the object code or an assembler version of it. The fault-finding task then becomes similar to the task for an LLL.
- You should note that coding is only a part of programming. Much analysis, specification and planning has to be done before coding begins. The actual coding may not contribute a large amount to the total cost of any particular program.

Some High Level Languages

There are now many HLLs in regular use. Some are purely general purpose languages, some are designed to be most effective if they are employed in particular fields.

The following are probably those in widest use.

BASIC: A general purpose language, BASIC can be put to almost any programming use. It has been used to implement many systems from business accounting to zoology. It is the language used for home computer and arcade games programs.

FORTRAN: A language designed for computation, FORTRAN has not been much applied to uses outside science and engineering.

PASCAL: (named after the French mathematician): Originally designed as a language for teaching programming, PASCAL is now being used for business and technical programming.

COBOL: (Common Business-Oriented Language): COBOL is of the same vintage as FORTRAN and was designed to do for the business community what FORTRAN was doing for the scientific. It is still heavily used in the business world.

C: (so named because it was derived from a language called 'B'): C is something of a hybrid, since it combines some of the properties of HLLs with those of LLLS. It gives some control over the translation process making access to the hardware easier. It is popular with systems programmers.

Finally, you will encounter in your general reading frequent references to '4GLs' or 'fourth generation languages'. Languages evolve in capability. PASCAL, for example, has more facilities than its ancestor, a language called ALGOL. PASCAL is described as a third generation language, with FORTRAN and COBOL being of the second.

In all the first three generations, the user must specify in the program how to solve the problem. The fourth generation languages take a radically different approach. The user describes the *problem* and leaves the solution to be worked out by the language software. Certain 4GLs provide the basis for *Artificial Intelligence (AI)* in computers.

SUMMARY

In this section of the module we examined the programming languages used to communicate our instructions to the computer. There are many such languages but they can all be classed as low or high level.

You can draw the distinction between the two levels:

- low level languages (LLL) are close to the computer's machine code and are difficult for humans to use in programming
- high level languages (HLL) are designed to make programming easier but programs must go through a complex translation process before the computer can execute them.

In general, no LLL program can be run on any computer but the make or model for which it was written. HLLs are portable between many makes and models. You know that this is possible, however, only if a compiler or interpreter to do the language translation is available on each computer.

You have written small programs in both assembly code and BASIC, a popular HLL. Even in these tiny samples, you will have noticed how much easier the high level coding is to write and read.

Finally, you were introduced to the concept of generations in programming languages which is used to describe the evolution capabilities as new languages are designed.

Engineering data information

Good engineering thrives on good records. Think of the various documents used in any engineering activity you know. You will find they span the whole operation from the concept of a product to its dispatch to the customer. They are also concerned with the customer's evaluation of the process.

Activity 8

List some of the documents that are used in engineering.

As you can see the engineering information storage required is vast. The computer is an ideal tool therefore to handle this amount of information that in many can be seen to be duplicated within a paper/physical system.

Take one small nut, which comes in many threads, shapes, materials and sizes. The kinds of data relating to nuts you would expect to find in a parts list and their relative memory usage could be:

part number	4	bytes	(allows upto 9,999 parts)
name	20	bytes	
diameter	3	bytes	
Material	10	bytes	
thread pitch	2	bytes	
type	12	bytes	(locking, non-locking)
supplier name and address	100	bytes	
alternative supplier name and address	100	bytes	
'used on' list	100	bytes	

The total storage space needed for this nut is 351 bytes

The storage requirements for a parts list of 9,999 items could be at least 9,999 x 351, which is just over 3.5 Mbytes. Many parts need much more data to define them than the nut does, so the value in practice could be far greater than 3.5 Mbytes.

Engineering drawings

Drawings can be stored in computer memories. An image of the drawing can be created by using a pattern of bits, as in the case of the electricity diode in figure 2. the image is contained in a matrix of eight bits by eight bits.

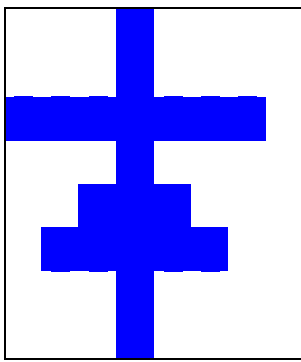


figure 2

This is stored in binary form :

```
0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0
1 1 1 1 1 1 1 0
0 0 0 1 0 0 0 0
0 0 1 1 1 0 0 0
0 1 1 1 1 1 0 0
0 0 0 1 0 0 0 0
0 0 0 1 0 0 0 0
```

Activity 9

Express in hex the bit pattern in figure 2

In practical systems, the images are built up of very tiny dots, but the principle is the same. a detailed engineering drawing of A4 size may need an image space of a megabyte or 1,025,000 Kbytes.

An engineering department may have hundreds of thousands of drawings to file- storage requirements may run into hundreds of megabytes.

We have really, only begun to consider the amount of information required for engineering but it is clear already that this amount is extremely large.

How is this information to be filed so that it can be easily retrieved when required?

The solution to this problem is the use of databases. The term data base describes a structured collection of data, held in computer storage, which supports the operation of the organisation which owns the data. Ideally there is one corporate database which serves all the departments within the company as shown in figure 3.

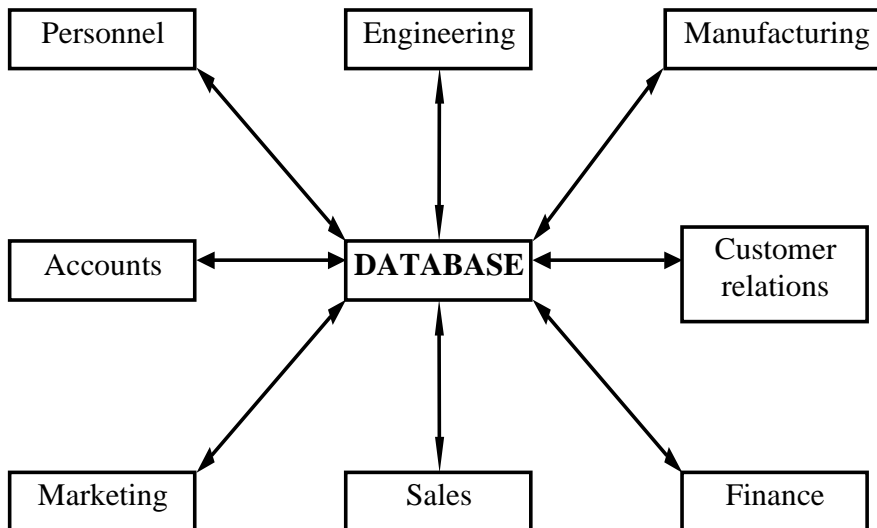
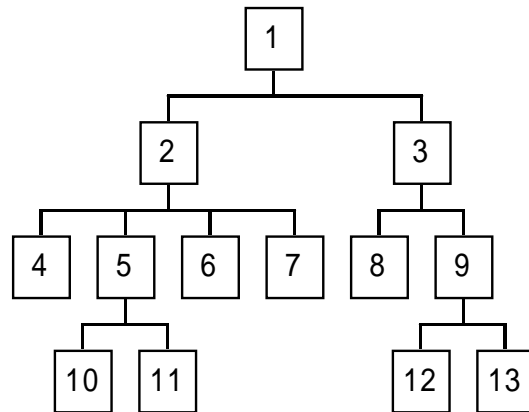


Figure 3

Hierarchical Databases

Each number in the box is an element of the tree and is termed a *node*. The node at the top of the tree - in this case is called the root. Nodes which lie at the end of the of branches for example 10,11,12 and 13, are called *leaves*. The diagram has four levels, the first being the root.



Consider the use of the hierarchical structure applied to engineering:

- the root is the product
- nodes 2 and 3 are two major units or assemblies comprising it
- nodes 4,5,6 and 7 are subassemblies making up the node 2 assembly
- nodes 8 and 9 are subassemblies making up the node 3 assembly
- nodes 10 and 11 are the two components of the node 5 subassembly
- nodes 12 and 13 are two components of the node 9 subassembly.

Activity 10

Describe any disadvantages you see in the hierarchical structure. (hint: think about selecting items at the lower level of the tree.)

Relational Databases

The relational database has a structure which is based on the mathematical theory of sets. However, you don't need any more knowledge of mathematics to use that kind of database than you do for the hierarchical. You can think of a relational database as being comprised of a collection of tables or two-dimensional arrays of items of data. Such tables are also known as flat files and an example is shown in Table 2.

Table 2

COMPONENT	COST	LEAD TIME	SUPPLIER
k3	14	20	C & Co
w4	6	15	A B & D
XI	13	21	X Corp
y4	25	45	F R T

In relational database terminology a flat file is called a relation. You could describe the table in Table 2 as a COMPONENT relation. A possible SUPPLIER relation appears in Table 3.

Table 3

SUPPLIER	ADDRESS	PAYMENT TERMS
C & CO	Atown	30 days
A B & D	Btown	C 0 D
X Corp	Atown	20 days
F R T	Kcity	30 days 2% cash

Since it does not have the linked records of the hierarchical database, the relational structure is flexible and permits easy connection between sets of data. Relations can be readily manipulated: they may be joined together, in whole or in part, or split into subrelations.

As a simple illustration of the manipulation possibilities, suppose a design engineer wants to know the locations of the suppliers of components. A new file could be built by extracting the COMPONENT data from the Table 2 relation and joining it to the SUPPLIER and ADDRESS data taken from Table 3. The resulting relation, shown in Table 4, contains only the data the engineer wants to know. It can be regarded as a modified version of the COMPONENT relation.

Table 4

COMPONENT	ADDRESS	SUPPLIER
k3	C & Co	Atown
w4	A B & D	Btown
XI	X Corp	Atown
y4	F R T	Kcity

Database Management Systems

The database has to be managed, partly to protect the data it contains from damage, partly to ensure that the users get the best possible service. Therefore, a database is buffered from the applications using it by a software system called a database management system, usually shortened to DBMS. The arrangement is shown in Figure 4 which you will recognise as a modification of Figure 3.

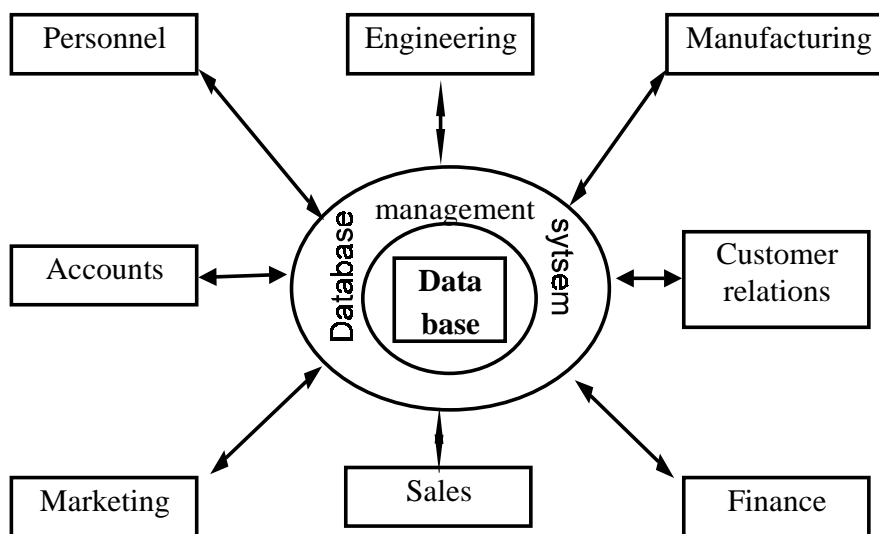


Figure 4

You will have the opportunity to use a database with its DBMS in one of the practical assignments in this module.

Activity 11

List some of the facilities you would expect a database management system (DBMS).to provide.

ENGINEERING DATA PROCESSING

There is a popular tendency to think of 'data processing' (DP) as being something which belongs to the business side of an organisation. Payroll, accounting, sales analysis and so on are quoted as examples of data processing.

Think about it for a moment, though. The solution of a equation that you programmed in BASIC was just as much the processing of data. The description of the process, or the algorithm, may be more complicated but the computer operations used are just the same as in a payroll job.

Engineering applications process data; they often have to process it much faster than business applications need do.

Management Information

A major function of DP is to provide management with the information it needs to run the organisation efficiently and profitably. Engineering contributes at least as much to management information as any of the other applications, sometimes much more.

Information for the management has to be selected and summarised. The broad picture has to be presented - details are a distraction. The managing director is not interested. in circuit diagrams - unless for some reason they are costing the organisation thousand of pounds in lost income, perhaps because they contain errors.

Management will extract from the database and other sources, such varied reports as:

- cost-effectiveness of design aids, such as computer-aided design and manufacturing systems.
- cost reduction of product designs
- process control system performance
- engineering personnel productivity
- manufacturing line productivity
- the status of research and new product design
- outstanding engineering problems
- new engineering applications.

Engineering Application Costs

Any data processing application will cost money. The cost has three parts:

- the labour, materials, and computer time charges for the development of the application (remember, an application in the end is a program)
- the cost of any additional or specialised hardware and software
- the running costs of the application when it is in service.

The second item is more likely to apply to engineering than business DP projects, although many engineering applications use no more than the hardware required by any other application.

Activity 12

The cost to develop and run a new engineering application can be estimated. That is part of the application estimate. State the likely basis on which management will permit the development to proceed.

Methods of Access to the Computer

You can classify data processing operations into three types:

- batch processing
- on-line systems
- real-time systems.

These three ways of getting your job done by the computer are not necessarily mutually exclusive.

Batch processing

Batch processing is usually applied to mainframe and large minicomputer installations. For reasons of security the system users are not allowed past the door into the computer centre. You take the relevant data for the application you require to be run and hand it over to the computer operating staff. They will put your request into a job queue along with those from other users. When the results of your job come out of the computer, the operating staff will have them delivered to you. Suppose you wanted your BASIC quadratic equation roots program to be run

for a series of values of A, B and C. Your request to the operators would include the BASIC program (if it is not already in storage) and the full set of A, B and C values. Some time later depending on the length of the queue and the time for each job in it, you would get your set of values of the corresponding roots. The batch processing mode is sketched in Figure 5

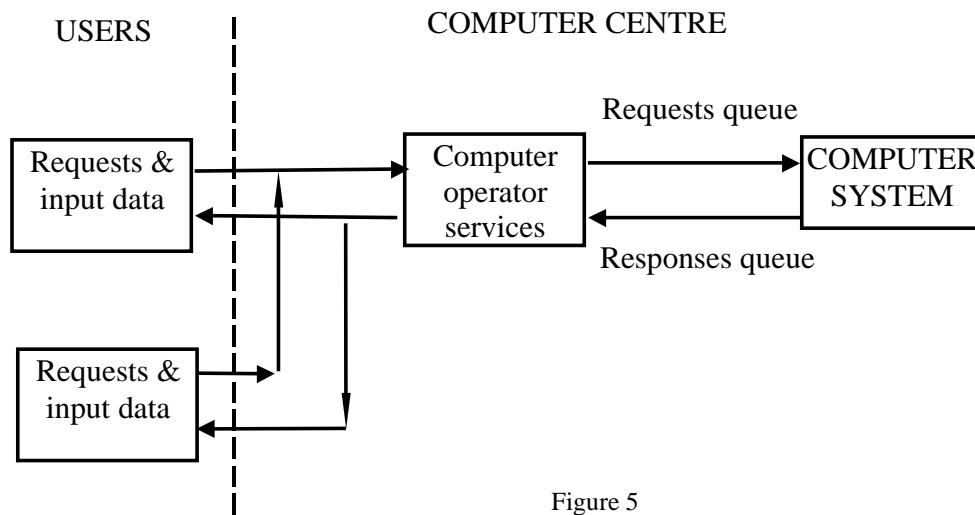


Figure 5

On-line systems

In an on-line system, you get much closer to the computer than the door of the computer room. You have a terminal comprising a visual display unit, a keyboard and perhaps a printer which give you direct access to the computer. The computer may be sometimes situated miles away in a different location. An on-line system is a multi-user system; a number of users will have access to such terminals.

You give your job requests directly to the computer and have the results displayed on the screen, printed out or both. Any delay in receiving your response depends on the processing queues and the job lengths, as in the batch processing case.

Real-time systems

Many on-line systems are also real-time systems - the 'travel agencies' system is one. Operation in real time means that the system's response to an input is delivered within the time span that

the demanding unit can tolerate. For example, an input signal which has to be accepted by the computer within ten microseconds or be lost is a typical real-time situation.

You may be the demanding device, but your tolerated response time is much longer. Even for an urgent enquiry, you will probably wait patiently for quite a few seconds. You should note that some terminals could be replaced by interfaces to manufacturing process equipment's to provide computer control over the processes. As you will see in the next section, process control is very much a real-time operation.

Since they are comprised of many terminals linked to a common computer system, on-line and real-time systems are examples of computer networks.

SUMMARY

In this section you have looked at some of the features expected of a computer system which serves engineering applications. The ability to store large volumes of data with easy retrieval is a prime requirement. You met a system which will provide that facility, the database. You are now able to describe the principal characteristics of two types of database the hierarchical and the relational. The hierarchical type uses linked records in a top-down format. The relational database may be thought of as a collection of flat files or tables which can be joined together or split as required. You saw examples of each type. You studied the nature of the information which management asks of engineering and looked briefly at the main cost elements which have to be considered in setting up a new engineering application.

Finally, we described the methods by which the user may have access to the computer:

- 1 batch processing
- 2 on-line systems
- 3 real-time systems.

SOME SPECIFIC APPLICATIONS

INTRODUCTION

In this section we will examine several applications of the computer in the field of engineering. You will look first at process control. A process in this context is a manufacturing operation in which certain factors, for example, temperature, have to be maintained at given values. You will find that a computer can be used to control several processes at the same time.

You will be introduced to computer-aided design, in which the computer works in direct support of the design engineer. A graphics screen and a light-pen or mouse act as a pad and pencil for drawing your diagrams and schematics. Much of the purely mechanical part of producing artwork, such as joining two points by a line or drawing a circle, is done for you by the system at your command.

There are a number of aids to manufacturing offered by a computer. You will briefly study one system called computer-aided manufacture, a system which supports the shop-floor side of production. You will see that it has a relationship with computer-aided design.

Computer-modelling is a technique which allows the simulation of systems and products. In the final application in this section, we will look at some simulations which are done on different subjects.

OBJECTIVES

When you have completed this section you will be able to:

- 1 describe some features required of a controller
- 2 describe nature and requirements of process control
- 3 give examples of process control operations
- 4 describe principles and use of computer-aided design (CAD)
- 5 describe what is required in computer-aided manufacture (CAM)
- 6 explain the role of simulation and prediction in engineering.

PROCESS CONTROL

Computers, especially mini and microcomputers, have had a big impact in the field of process control. Many industrial processes require that equipment or other agents used in the process be kept operating as closely as possible to some set value. For example, a motor must be kept running at a constant speed, an enclosure held at a certain temperature or the liquid in a tank maintained at a given level.

Process Controllers

A process controller which has to do functions like this must:

- 1 constantly measure the controlling factor or variable - for example, temperature or speed - and compare it with its set point
- 2 compensate for any change in the variable by adjusting the system until the set point value is restored. If, for example, the speed drops due to an increase in the load on the motor, the controller raises the power to the motor until the correct speed is reached.

Activity 13

Name two examples of process control equipment in your home. and briefly describe the processes concerned and the controlling variables. (Hint: think about food.)

Industrial processes have traditionally used specially-designed controllers for each process. However, this can be expensive and also inflexible. If the process is modified, say, to improve it, a completely new controller may have to be designed. Since its control capabilities are essentially software-generated, you can use a computer as a controller for any kind of process.

Activity 14

Describe the two items you require to make the computer act as a process controller..

Computer Process Controllers

The typical form of a computer-based process control system is shown in Figure 6.

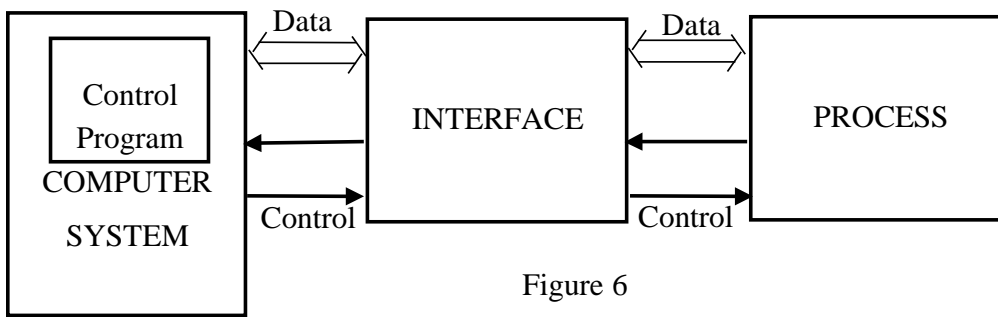


Figure 6

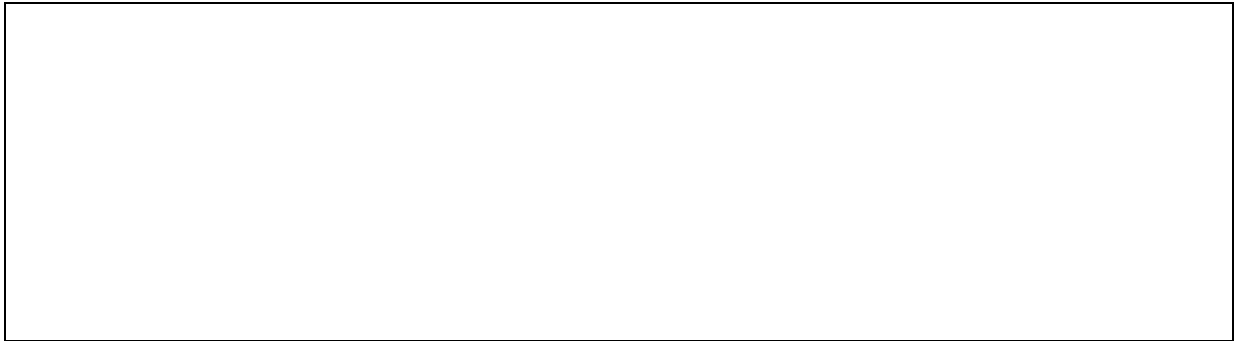
Activity 15 gives you the opportunity to draw the block diagram of a computer-controlled constant temperature enclosure.

Activity 15

The temperature in a cabinet has to be kept up to a given level. The cabinet contains:

- a temperature sensor, which gives out a dc voltage proportional to the temperature it senses
- a heating plate
- a two-position switch which turns the hot plate on when it is set to its 'on' state and off when set back to its 'off' state. The switch is operated by electrical pulses.

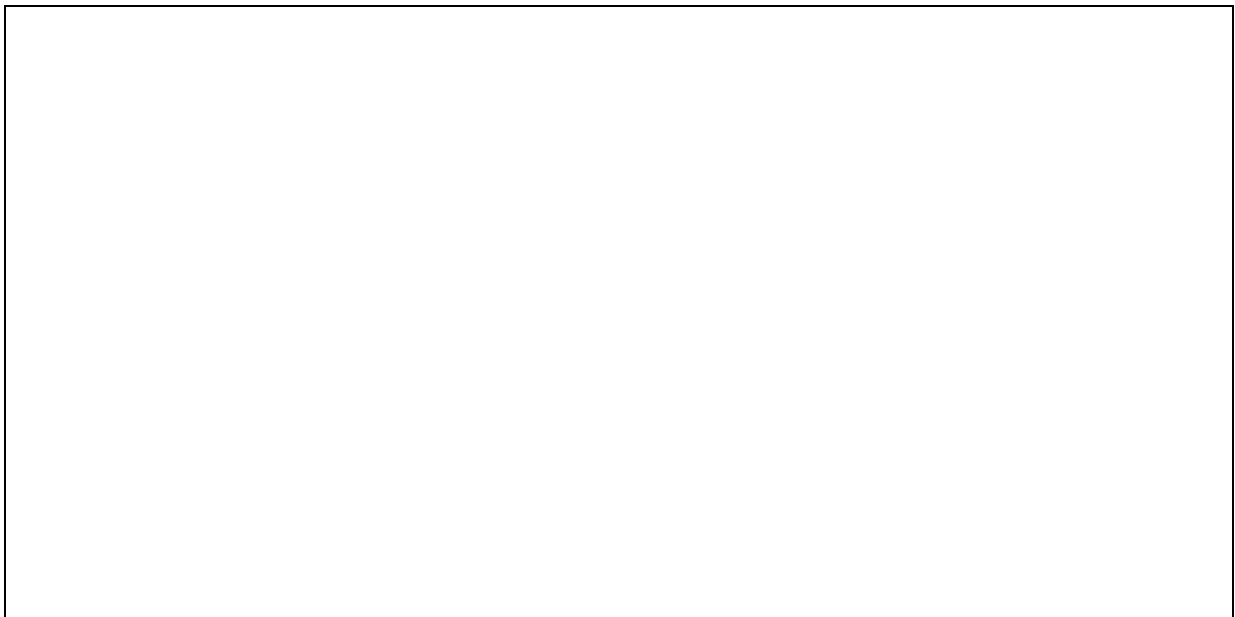
Sketch the system, showing the essential connections between the computer, interface and cabinet. Show any data conversion devices which may be needed in the interface.



You have the hardware now to control the process of Activity 8 What about the control program?

Activity 16

Draw a flowchart describing the main control cycle applying to the temperature cabinet process. Keep it simple and start by reading the temperature. Call the set point value SP.



You will have noticed that the process you have just worked on in Activity 9 is similar to the one in a domestic oven. Have you noticed the time which elapses between the heating periods in an oven? They are often very long. A microcomputer used for oven control would spend virtually the whole of its time doing nothing.

This shows another reason for the superiority of the computer as a process controller - it can control several different processes concurrently. This can provide a significant overall cost saving. In addition, it allows different but related processes to use the same source data and to report data to a common file. It is also possible for data derived from one process to be passed to another if required.

Activity 17

Using Figure 5 as a model, sketch a block diagram of a computer-based system in a factory controlling: -

- an air-conditioning plant
- a production line
- a liquid flow line.

Keep your diagram simple.. put in only the essentials.



A computer which is to be applied to process control should:

- have many input and output channels, usually called I/O ports, to allow for connection to a variety of interfaces
- be able to respond quickly to requests from input signals
- be able to store and access quickly a variety of programs and data.

The bulk of computers, including microcomputers, easily fulfil these conditions. One well-known microcomputer CPU has 256 I/O ports available for external connections.

COMPUTER-AIDED DESIGN

In computer aided design (CAD) systems, a 'computer assists in the design process, relieving engineers of many tedious jobs and even doing things that engineers could not easily do for themselves.

CAD is used in a wide range of engineering fields including:

- aerospace, for the design of aircraft
- civil engineering, for the design of structures
- automotive engineering, for the design of car chassis, bodies, parts and so on
- electrical engineering, for the design of printed circuit boards, assemblies and integrated circuits (chips).

CAD Systems

A typical CAD system consists of a workstation with a VDU terminal, which can display both graphics and text, a keyboard, a light-pen and/or mouse, and a printer. There may also be a plotter, that is, a device which produces high quality artwork. The paper the drawing is made on usually lies on a flat bed, perhaps securely held by a vacuum or magnets. A pen is moved over the paper following x, y co-ordinates supplied by the computer. In some models several different pens can be selected by the mechanism thus furnishing a colour drawing capability. A plotter is shown in Figure 7.

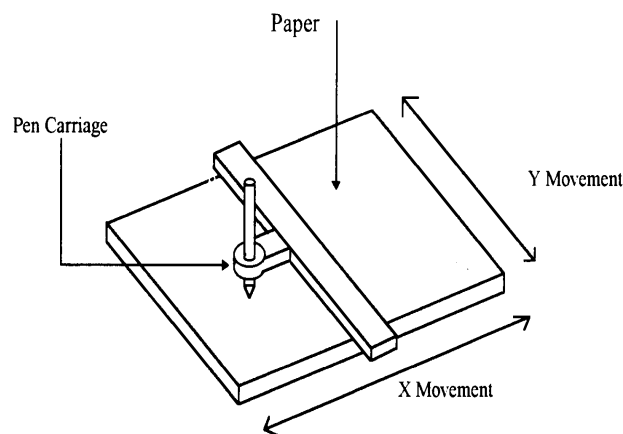


Figure 7

The workstation may be connected to a mainframe computer, but it will also have its own local computing capability.

In fact, at the lower-price end of the 'scale, the workstation may be a personal computer. All the I/O devices mentioned can be fitted to a PC.

You can use a CAD system as a large, 'intelligent' paper-and-pencil combination. The screen is the 'paper' you draw on and the light-pen or mouse acts as your pencil. The CAD software, however, does much of the actual drawing for you.

All graphics can be reduced to a set of basic symbols, such as: straight line, circle, arc, curve, point, polygon. You are presented with a menu of these graphics primitives from which you choose the one you want by pointing at it with your light-pen or mouse. You then tell the CAD system where to place it on the screen.

Activity 18

To fix the length and position of a straight line, you need only to point at the two end positions of the line. What does the system need to be told in order to place the following symbols:

- a) a circle
- b) a triangle
- c) a rectangle?



Parts libraries

The graphics Primitives can be combined to produce complex symbols, a nut, a bolt, a fan blade, a fan belt, an electrical component. The complexity of the symbol has almost no limit.

Once they are built, you do with parts symbols what you do with utilities - you file them for subsequent use by yourself or others using the CAD system. Most CAD systems are supplied with libraries of standard items by the vendors of the system.

Parts are selected from a menu like the primitives, although if the library is very large, you may have to identify them initially by a file reference number.

Symbols may be rotated, so you can have them at any orientation that suits the picture you are building up. You can also magnify them, using a zoom facility so that you can work on tiny details on any part. You can also shrink a part, in order to see how it relates to the rest of the design you are doing. Your part may appear not much bigger than a dot in such circumstances, but all the information is still there in the computer's memory and can be brought back in full on your command. CAD graphics is not limited to two dimensions (2D). 2D is primarily used for draughting work. 3D graphics is increasingly becoming the CAD tool for designers.

A CAD system computer has a limited selection of input and output devices and, therefore, not too many interfaces to attach. However, the plotter and the graphics screen have a high resolution capability, which means that they can display very finely detailed artwork. The storage requirement for this sort of document is very large.

CAD work calls for much calculation by the computer; the system has plenty of number-crunching or mathematical calculations to do. A CAD system computer therefore should possess:

- a large storage capacity, both fast and medium access
- a fast transfer rate between medium and fast access memory
- a fast computing facility, preferably hardware-based.

The numerous engineering workstations have these capabilities and so do models in the top end of the PC range.

COMPUTER AIDS TO MANUFACTURE

The computer assists manufacturing activities in a number of different areas. You have already met one of them - process control is a production operation. There are two principal categories of computer aids to manufacture:

- 1 manufacturing administration
- 2 factory operations.

Manufacturing Administration

Manufacturing resource planning (MRP) is the primary manufacturing administration computer application. MRP is concerned with three issues:

- 1 a master production schedule setting overall production targets for all products on forecast demand
- 2 detailed production plans with bills of materials to meet the master schedule
- 3 shop-floor control, releasing orders for manufacture and recording the production transactions. You should note that the first item can affect engineering design. A product at that stage may exist only in the minds of the marketing people. CAD will have to help to bring the product into reality.

As computers grow more compact and powerful, the MRP system's more detailed functions are moving to the shop-floor systems. Shop-floor Operations Computer-aided manufacture (CAM) is concerned with the use of computers to control manufacturing operations. When a product has been designed, CAM creates and uses the programs for computerised production equipment, sequences the different machines, and ensures that materials are available in the right place at the right time.

Activity 19

Specify where the data for the CAM operation comes from.

CAM may need to acquire data from a third source. Some manufacturing processes are long and complicated and are difficult to plan efficiently. Once again, the computer can be brought in to help. Computer-aided process planning systems (CAPP) are used to assist in producing more accurate and consistent process plans. Even though a factory may be highly automated, things can still go wrong.

Activity 20

Suggest two or three things which could go wrong in an automated factory.

Manufacturing schedules can be badly hit if failures aren't detected in time for remedial action to be taken. Computers can help alleviate such situations. For example, you can use real-time monitoring of process equipment performance to detect that some variables are slipping out of adjustment. The fault can be rectified before much harm is done. Again, if materials don't turn up, your CAM system could switch the affected equipment to other products very quickly. Computers have fast reaction times. Computers used in CAM have much in common with those suggested for process control. A similar range of machines would suit most manufacturing purposes. You should note, following the comments in the previous paragraph, that a networking structure would be at least beneficial, if not mandatory, to get the best results from automated manufacture.

SIMULATION AND PREDICTION

Simulation means imitation; it is used when it is expensive, difficult or impossible to derive information about some object any other way. For example, it is reasonable to ask if an electrical component will operate successfully for ten years. You could run it for ten years to find out, but by then you would have no interest in the answer - the component would be two generations out of date. You need to know if the component can be used in the equipment you are designing now.

You can find out what you want to know by simulating the effects of age. Typically, this is done by heating samples of the component in a chamber for a given period. The results of the age simulation will enable you to predict the probability of a ten-year life span.

Many simulations can be carried out without the need for any hardware other than that of a computer system. For example, the effects of heat on the value of a resistor in a circuit may be checked by varying its value and observing the changes in the simulated circuit performance. Computers have been employed in simulation work since the very early days of their existence.

Simulations in Computer-aided Design

When the design of a digital circuit is finished, you need to ensure that it works as intended. One way to do that is to build a hardware prototype and test it fully. That can be quite costly.

Activity 21

List some of the reasons why a hardware prototype may be costly.

Circuit modelling

A CAD system contains programs which, in effect, build a model of the circuit. The input/output relationships of the individual devices in the circuit are part of the library information. Your circuit diagram defines how the devices are interconnected. Other programs will operate the model. You start the simulation running by specifying what signals are to be applied, to what input lines, and the timing sequences to be used.

The system will simulate the performance of the circuit, providing a report of all data you request. The simulation can be run as soon as the design is finished - there is no delay for hardware to be made. If you have made a mistake in the circuit, it can be corrected immediately and the simulation re-run at once. Prototyping costs can be reduced; in some cases it may be possible to eliminate them altogether.

3D modelling

Another type of simulation in CAD is the 3D modelling of products. A three-dimensional picture can be produced of an object, a car, say. The picture can be rotated to show the car from any angle: viewed from above, viewed from beneath, showing front or rear. The designer is given a realistic impression of the car's appearance long before any prototype can be made.

Other Simulations

Simulation has been in use before the advent of the computer. Physical scale models have long been employed to simulate the behaviour of some product in order to predict with reasonable certainty how the real thing will behave. Computers have made the job of modelling easier and the operation of the model very much faster. There are a number of software packages, including special languages, which can be used to carry out a simulation of many kinds of system or product.

Simulation applications include:

- 1 telephone systems and subsystems
- 2 analogue electrical circuits
- 3 stock control systems
- 4 manufacturing capacity
- 5 road systems
- 6 sales forecasts
- 7 process control
- 8 business development.

Any complex product is a candidate for simulation before too many resources are put into its production.

Activity 22

How do the features of a computer for simulation work compared with those of a CAD computer.

SUMMARY

In this section we examined, some applications in different branches of engineering. We described the requirements of these applications and considered some of the capabilities of suitable computers to accomplish them.

You are now able to describe process control and to show that a computer may be a cheaper process controller than one specially designed for the process. The computer is flexible and fast and can handle more than one process at a time.

You met the computer-aided design system, in which the computer directly serves the designer.

CAD uses interactive graphics to enable the designer to draw product diagrams, schematics and other drawings on a high resolution screen. The drawing tool is a light-pen or mouse.

The drawing is held in memory and supplies, often via the corporate database, the design information required by the CAM or computer-aided manufacture system.

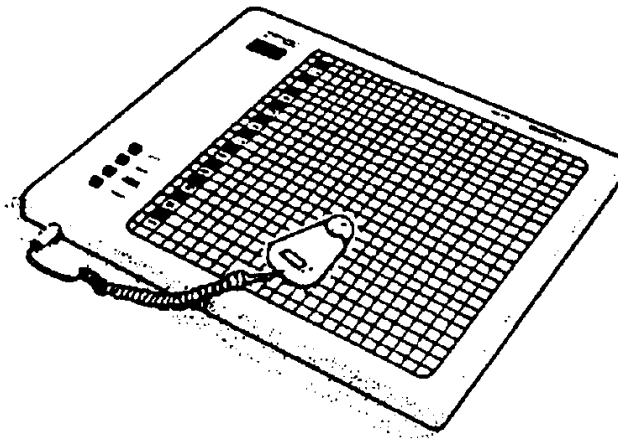
You now know that CAM provides the programs and data needed by the shop-floor machines to carry out the manufacturing operations. Those operations have to meet production schedules given by the manufacturing resources planning (MRP) system to CAM.

You have seen that simulation is a means of predicting the behaviour or performance of a product or system before the product or system has a physical existence. There are considerable savings to be made by using simulation, especially in the reduction of prototype construction requirements. Simulation software may be present as part of a special-purpose system like CAD or is available for use on any suitable computer.

Computer Applications for Engineers

—◆—
From Light Pen To Laser Printers

—◆—
I/O DEVICES



A digitizer

TOUCH SCREEN

A touch sensitive screen is a device that allows a screen to be activated by the user touching the screen with a finger. This is particularly useful where a menu of processing options is available on the screen for selection.

LIGHT PEN

A light pen is shaped like a pen and contains a photo-electric or light-sensitive cell in its tip. When the pen is pointed at the screen the light from the screen is detected by the cell and the computer can identify the position of the pen. By 'mapping' the screen to allocate particular functions to particular locations on the screen, the position of the pen indicates a particular function. The light pen enables specific parts of a picture on display to be selected or altered in some way, making it particularly useful for applications such as computer aided design (CAD).

JOYSTICK

A joystick is similar to a car's gear lever, except that fine variations in the angle of movement can be achieved. The cursor movement is a reflection of the movement of the joystick in terms of both direction and speed. It is commonly used for computer games and for CAD.

The reading of hand-printed characters presents particular problems because of the almost infinite variation of printing styles. Recognition is possible provided that the person preparing the data has a visual guide of the preferred style. The character set will usually be limited to numerals and a few alphabetic characters.

Artificial intelligence techniques are being applied to OCRs to allow the 'learning' of new character sets.

APPLICATIONS OF Optical Character Readers (OCR)

OCR is often used to copy printed material. The user would place the original material on the scanner and activate the OCR software. This captures the information and it is displayed as text on the computers screen where it can be modified and saved to disk or printed out.

It is necessary to proof read any text that has been scanned in as the OCR software may misinterpret some characters. For instance the letter h may be interpreted as an I followed by an n (In).

OPTICAL MARK READERS (OMR)

An OMR is designed to read marks placed in pre-set positions on a document. The document is pre-printed and the values which can be entered are limited as each value is represented by, for example, a box in a certain position. Thus, a suitable application for OMR is a multi-choice exam paper, where the answer to each question has to be indicated by a pencil mark in one of several boxes after the question. optical readers can read up to 10,000 A4 documents per hour.

Many applications involve the need to record numeric data or a combination of category data and numeric data Two examples of how this can be achieved are shown below.

Date of Birth			Employee Number					
Day	Month	Year						
3	July	46	2	7	3	7	0	1
(0)	(0) Jan <input type="checkbox"/>	(0) (0)	(0)	(0)	(0)	(0)	(0)	(0)
(1)	(1) Feb <input type="checkbox"/>	(1) (1)	(1)	(1)	(1)	(1)	(1)	(1)
(2)	(2) Mar <input type="checkbox"/>	(2) (2)	(2)	(2)	(2)	(2)	(2)	(2)
(3)	(3) Apr <input type="checkbox"/>	(3) (3)	(3)	(3)	(3)	(3)	(3)	(3)
(4)	(4) May <input type="checkbox"/>	(4) (4)	(4)	(4)	(4)	(4)	(4)	(4)
(5)	(5) Jun <input type="checkbox"/>	(5) (5)	(5)	(5)	(5)	(5)	(5)	(5)
(6)	(6) Jul <input checked="" type="checkbox"/>	(6) (6)	(6)	(6)	(6)	(6)	(6)	(6)
(7)	(7) Aug <input type="checkbox"/>	(7) (7)	(7)	(7)	(7)	(7)	(7)	(7)
(8)	(8) Sep <input type="checkbox"/>	(8) (8)	(8)	(8)	(8)	(8)	(8)	(8)
(9)	(9) Oct <input type="checkbox"/>	(9) (9)	(9)	(9)	(9)	(9)	(9)	(9)
	Nov <input type="checkbox"/>							
	Dec <input type="checkbox"/>							

BAR CODE READERS

The bar code is also an optical code which is normally read by a light pen. The code makes use of a series of black bars of varying thickness. The gaps between each bar also vary. These bars and gaps are used to represent numeric data. The values represented are often printed underneath in decimal form.

A typical bar code



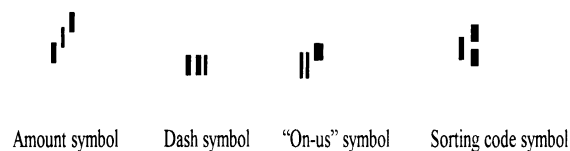
Bar codes are commonly used to store a variety of data such as price and stock code regarding products in shops and supermarkets. A sticker with the relevant bar code (itself produced by computer) is attached to each product. Sometimes, the checkout will have a built-in scanner station over which the goods pass. This is convenient if packages of a regular shape but soft packages with creases may cause problems for the scanner. In such cases, the light pen or wand provides a more practical solution. By using the data from the code, the cash register can identify the item, look up its latest price and print the information on the customer's receipt.

Another useful application is for the recording of library issues. A bar code sticker is placed inside the book cover and at the time of issue or return it can be scanned and the library stock record updated. By issuing each library user with a bar coded library card, the information regarding the individual borrowing a book can be linked with the details of books issued at the time of issue.

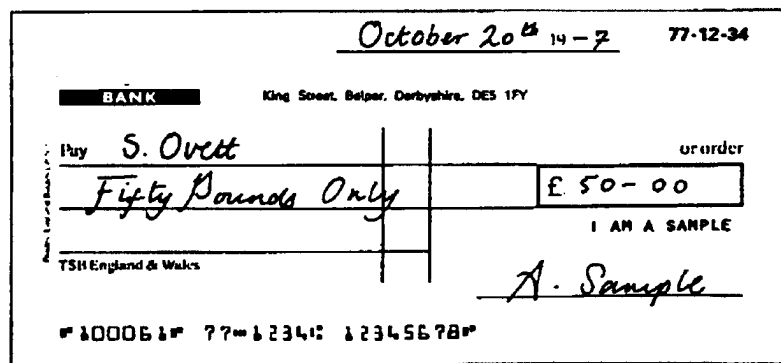
MAGNETIC INK CHARACTER READER (MICR)

This particular device is used almost exclusively by the banking industry, where it is used for sorting and processing cheques in large volumes. The millions of cheques which pass through the London Clearing System could not possibly be sorted and processed without the use of MICRS.

Highly specialised characters such as the ones shown below, are printed along the bottom of the cheques by a special printer, using ink containing iron oxide.



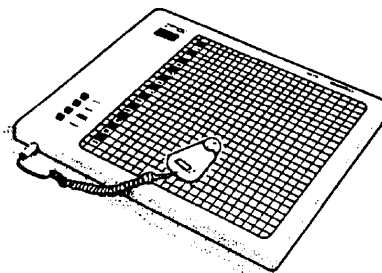
The MICR first magnetises the characters as the cheque passes through and then decodes them by induced voltage signals. A high degree of reliability and accuracy is possible, partly because of the stylised font used, but more importantly because the characters are not affected by dirty marks. This is obviously important when cheques may pass through several hands before reaching their destination. Such marks may cause problems for an optical character reader.



Bank cheque

DIGITISER

Writing pads and tablets are input devices where the movement of a pen or puck over a sensitive pad is translated into digital signals giving the pen's and puck's position. This device is now widely used in Computer Aided Design (CAD) work.



A digitizer

PRINTERS AND PLOTTERS

PRINTERS

Printers can be categorised according to SPEED of operation and the QUALITY of print.

Printers are also identifiable as either IMPACT or NON-IMPACT devices.

IMPACT PRINTERS

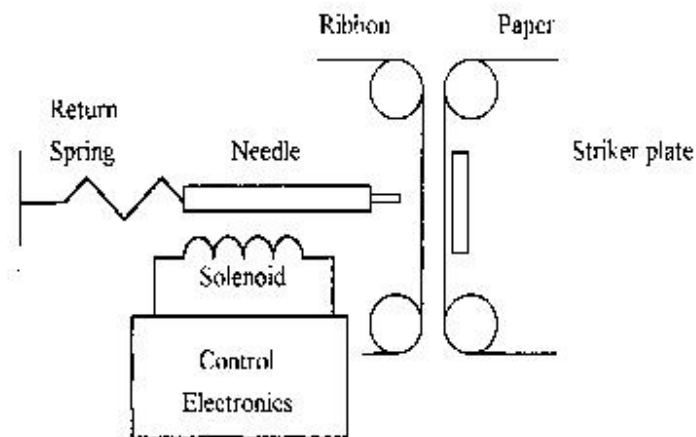
Impact printing uses a print head to strike an inked ribbon which is located between the print head and the paper. Individual characters can be printed by either a dot-matrix or by print heads which contain each character as a separate font (solid font type).

DOT-MATRIX PRINTING

Characters can be formed from a matrix of dots, typically a 7 x 7 or 9 x 7 matrix. The density of the matrix largely determines the quality of the print. The impact is carried out by a number of 'needles' which can be projected or withdrawn according to the pattern that is required. A ROM (Read Only Memory) 'chip' within the printer provides it with a character set. Certain built in functions such as automatic feed and alternative character sets can be altered by switching small 'dip' switches inside the printer. The intensity of print can be improved by passing the print head over a line twice (double-strike). In summary, a dot matrix printer can be programmed to effect a wide variation of printing results.

With a 3-colour ribbon, a variety of colours can be produced by a dot-matrix printer. Colours beyond those directly available can be mixed by using slight paper shifts, very small movements of the paper, and double striking to mix new colours. This method makes use of software control to achieve colour output comparable with more expensive multi-colour printers.

SCHEMATIC ARRANGEMENT OF A DOT MATRIX PRINTER MECHANISM



SOLID FONT PRINTING

A solid font head uses a separate font for each character and character sets have to be changed by changing the head. Because they form a solid image, they generally provide a better quality of print than the dot-matrix type. There are a number of types which are described below.

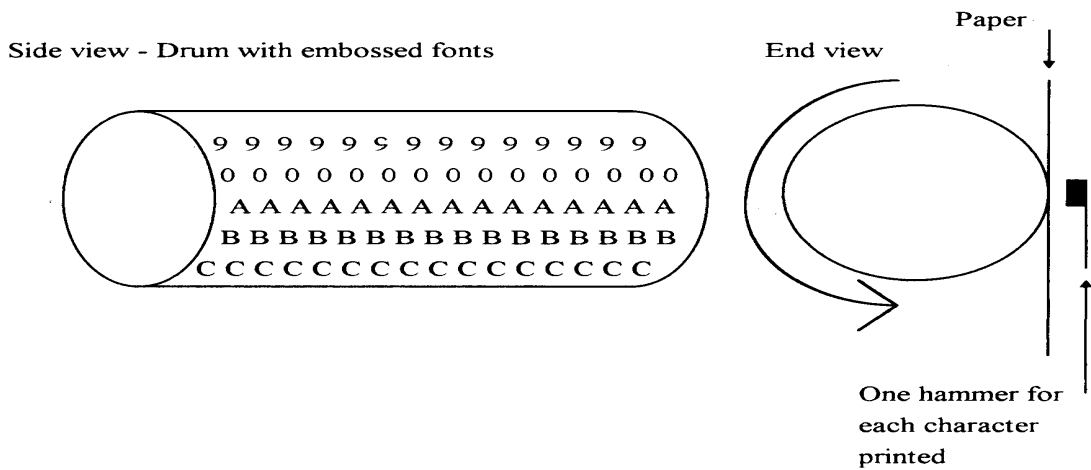
DAISY-WHEEL HEAD

As the name suggests, character fonts are attached to 'petals' on a central wheel which has to revolve to place a particular character in the print position. Inevitably, the considerable movement required between each character print means that daisy-wheel printers operate relatively slowly, about 30 to 60 characters per second. The quality of print is very high which makes it a popular device for the production of, for example, legal documents and other output where image is vital.

BARREL PRINTERS

The barrel printer has a band with a complete set of characters at each print position. Each print position has a hammer to impact the print ribbon against the paper. The mechanism is illustrated below.

Layout of a Barrel printer



One complete revolution of the barrel exposes all the characters to each print position. Therefore a complete line can be printed in one revolution. The characters on the barrel are arranged so that all characters of the same type are in the same horizontal position. Thus, in a line of print any requires As can be printed, then Bs and so on, until the complete line is printed. The barrel revolves continuously during printing, the paper being fed through and the process repeated for each line of print.

GOLF-BALL HEAD

The golf-ball head also has characters embossed on its surface but it differs from the cylinder print head in three main ways. Firstly, the spherical shape accommodates a wider range of characters than the cylinder shape. Secondly, different character sets can be obtained by changing the print head. Thirdly, the head is made to strike the ribbon by a cam mechanism. No hammer strikes the head. Individual characters are obtained by the rotation and tilting of the head.



Dot-matrix printers are much faster than solid-font printers. Speeds of 100 to 300 characters per second are

common for low-speed, impact dot matrix printers.

All the printers described above are CHARACTER printers in that they print a single character at a time. Faster printing can be achieved by LINE printers. A line printer prints a complete line of characters, rather than in the serial fashion used by character printers.

Non-Impact Printers

Non-impact printers do not require mechanical hammers and print heads do not strike the paper. A variety of printers are available using a wide range of technologies. The most popular are as follows.

THERMAL PRINTERS

Characters are burned onto heat-sensitive thermographic paper. The paper is white and develops colour when heated above a particular temperature. The heat is generated by rods in the dot-matrix print head. By selective heating of rods, individual characters can be formed from the matrix. Printing can be carried out serially, one character at a time or, through the use of several heads, on a line-by-line basis. Serial thermal printing is slow but speeds in excess of 1 000 lines per minute are possible with line thermal printing.

ELECTROSENSITIVE PRINTERS

This type produces characters in a similar fashion to the thermal printer except that the paper used has a thin coating of aluminium which covers a layer of coloured black, blue or red dye. Low voltage electrical discharges in the matrix rods produce sparks which selectively remove the aluminium coating to reveal the layer of dye underneath. Operated as line printers with heads at each print position, printing speeds in excess of 3000 lines per minute are achieved.

INK-JET PRINTERS

Ink-jet printers spray high-speed streams of electrically charged ink drops from individual nozzles in the matrix head onto the paper to form characters. Many will hold colour cartridges to produce excellent colour output.

LASER PRINTERS

Laser printers use a combination of two technologies, electrophotographic printing used in photo-copying and high intensity lasers. A photoconductive drum is initially charged and then a high intensity laser beam selectively discharges areas on the drum. As with photocopiers, toner material is spread over the surface to form an ink image. This is then transferred to the paper and made permanent through heating.

Laser printers offer a very high quality print at exceptional speeds. However they are expensive.

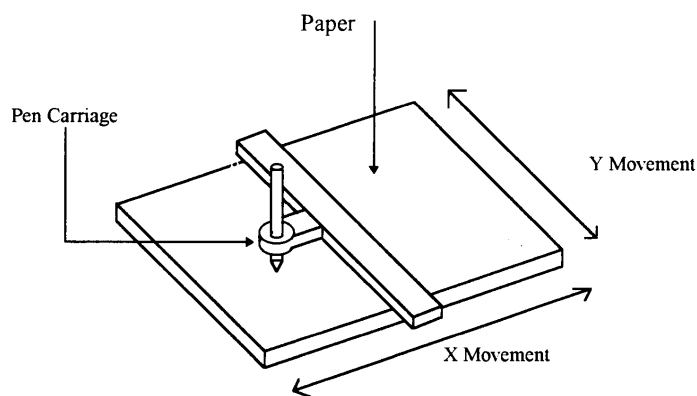
PLOTTERS

A plotter is an output device that produces an image on paper by controlling the motion of a pen carriage that draws lines. The plotter sees the output medium of paper as a series of X,Y co-ordinates. If you plotted similar co-ordinates on a piece of graph paper and drew lines connecting the points you would have a visual approximation for how the plotter draws lines.

The computer system sends the plotter a series of numbers representing X,Y co-ordinates in the desired pattern. The pen carriage then moves to a different position while holding the pen on the surface of the paper. Curves are drawn by linking a series of very short straight lines together.

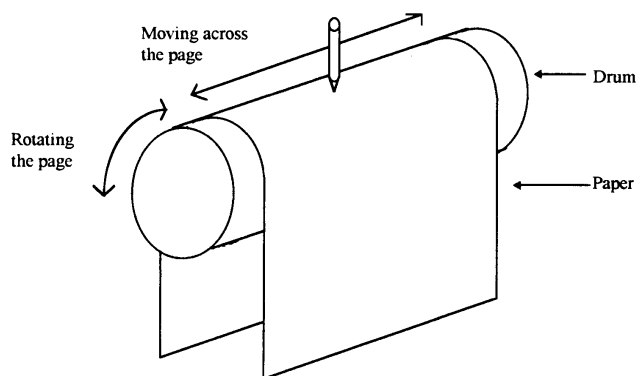
The two major types of plotters are **Flat Bed** and **Drum**. With flat bed plotters the paper remains stationary while the pen moves in both X and Y directions.

Schematic layout of a Flat Bed plotter



Some plotters employ a rotating drum that rotates, moving the paper forwards and backwards whilst moving the pen carriage across the paper.

Schematic layout of a Drum plotter



In either case it is possible to specify and draw quite complex shapes in multiple colours by changing pens or by having the pen carriage hold multiple pens. In addition to small images plotters are used to draw very large images at very high resolution, such as posters.

PRINTERS AND PLOTTERS SUMMARY

IMPACT PRINTERS				
TYPE	DESCRIPTION	PRINT SPEED	RIBBON TYPE	QUALITY
DOT MATRIX	Builds up characters by pattern of dots	MEDIUM 30 - 600 cps	Nylon or carbon	FAIR
DAISYWHEEL	Circular print head with characters mounted on stalks radiating from the centre	SLOW 15 - 100 cps	Carbon	VERY GOOD
THIMBLE	Similar to daisywheel print head except characters on outside of thimble-shaped head	SLOW 25 - 100 cps	Carbon	GOOD

NON IMPACT PRINTERS			
TYPE	DESCRIPTION	PRINT SPEED	QUALITY
THERMAL	Impresses a print head onto aluminium coated paper. A column of pins/wires forms the characters and an electrical charge "bums" character on the wires into the paper.	FAST 5 sheets/min	POOR
LASER JET	A laser beam prints characters using the Xerox reprographic method.	FAST 10 sheets/min	GOOD
INK JET/ BUBBLE JET	Sprays a finely controlled jet of ink onto the paper. Shape of character is controlled by electrical fields acting on the electrically charged ink droplets.	MEDIUM 25 - 250 cps	FAIR

PRINTING-PLOTTING DRAWINGS	
TYPE	DESCRIPTION
X,Y PLOTTER	Operates on co-ordinate geometry. Can be flat bed where pen moves in X and Y or cylinder where pen moves in X but cylinder rotates to give Y.
PRINTER PLOTTER	Printer prints a very fine dot matrix (bit map) to produce the drawing

Paper feed mechanisms

FRICTION FEED Paper gripped between two rollers, as in a typewriter, suitable for single sheets.

TRACTION FEED Specially designed paper with holes along the edges fits over wheels with corresponding spokes for the holes. The wheels revolve, drawing the paper through the printer. Suitable for listing paper.

CUT SHEET FEEDER (or FORM FEED) A device which automatically feeds sheet paper into friction feed printer.

Features which have improved printing speeds include **BI-DIRECTIONAL** printing (printing in two directions). This is where the printer prints from left to right, then right to left. This is made possible by a buffer (Small memory)

For use with a PC, impact printers of all types are becoming obsolete. This is mainly due to the high quality of print at competitive prices offered by ink jet & laser printers.

REQUIRED COMPUTER FACILITIES

Basic facilities which you would require of a computer to operate the applications reviewed.

Application	Facilities required						
	A	B	C	D	E	F	G
Process control	√	√		√		√	√?
Computer-aided design			√		√	√	√
Computer-aided manufacture			√		√	√	
Simulation & prediction			√		√	√	√
Quality control			√		√		
Stores control/Database systems			√		√		
Financial control/numerical analysis			√		√		
Information storage & retrieval	√?		√		√	√?	

Column headings

- A: multiple I/O ports
- B: small-medium backing store requirement
- C: large backing store requirement
- D: fast response to input signals
- E: medium response to input signals
- F: fast transfer rate from backing store
- G: fast number-crunching

The table must be regarded as being indicative only of the capabilities required. A particular implementation of an application may require more or less than the table suggests.

A system devoted to record storage and retrieval, serving many terminals, would require multiple input-output channels and probably a fast data transfer rate to support them. There are many computers, even in the microcomputer class, which can meet most, if not all, of the requirements listed in the table above