Chapter-5 :

I/O ORGANIZATION AND SYSTEM S/W & PROGRAMMING TECHNIQUES

i) VARIOUS I/O DEVICES

The input output subsystem of a computer, referred to as I/O, provides an efficient mode of communication between the central system and the outside environment. Programs and data must be entered into computer memory for processing and results obtained from computations must be recorded or displayed for the user.

A computer serves no useful purpose without the ability to receive information from an outside source and to transmit results in a meaningful form. The most familiar like keyboard that allows a person to enter alphanumeric information directly.

The CPU is an extremely fast device capable of performing operations at very high speed. When input information is transferred to the processor via a slow keyboard, the processor will be idle most of the time while waiting for the information to arrive. To use a computer efficiently, a large amount of programs and data must be prepared in advance and transmitted into a storage medium such as magnetic tapes or disks.

Devices that are under the direct control of the computer are said to be connected on-line. these devices are designed to read information into or out of the memory unit upon command from the CPU and are considered to be part of the total computer system. Input or output devices attached to the computer are called peripherals. Among the most common peripherals are keyboards, display units, and printer.

Every time a key is depressed, the terminal sends a binary coded character to the computer. There are different types of video monitors, but the most popular use a cathode ray tube(CRT). The CRT contains an electronic gun that sends an electronic beam to a phosphorescent screen in front of the tube. The beam can be deflected horizontally and vertically. To produce a pattern on the screen, a grid inside the CRT receives a variable voltage that causes the beam to hit the screen and make it glow at selected spots. A characteristic feature of display devices is a cursor that marks the position in the screen where the next character will be inserted. The cursor can be moved to any position in the screen, to a single character, the beginning of a word,or to any line.

Printers provide a permanent record on paper of computer output data or text. There are three basic types of character printers. The daisywheel printer contains a wheel with the characters placed along the circumference. To print a character, the wheel rotates to the proper position and an energized magnet then presses the letter against the ribbon.

The dot matrix contains a set of dots along the printer mechanism. For example, a 5 X 7 dot matrix printer that prints 80 characters per line has seven horizontal lines, each consisting of 5 x 80 = 400 dots. Each dot can be printed or not, depending on the specific characters that are printed on the line. The laser printer uses a rotating photographic drum that is used to imprint the character images. The pattern is then transferred onto paper in same manner as a copying machine.

Disks are used mostly for bulk storage of programs and data. magnetic disks have high speed rotational surfaces coated with magnetic material. Other input and output devices encountered in computer systems are digital incremental plotter, optical and magnetic character readers, analog to digital converters, and various data acquisition equipment.

Input and output devices that communicate with people and the computer are usually involved in the transfer of alphanumeric information to and from the device and the computer.

See the UNIT II for ASCII alphanumeric characters Table

ii) I/O ADDRESSING TECHNIQUES

Input output interface provides a method for transferring information between internal storage and external I/O devices. Peripherals connected to a computer need special communication links for interfacing them with the CPU.

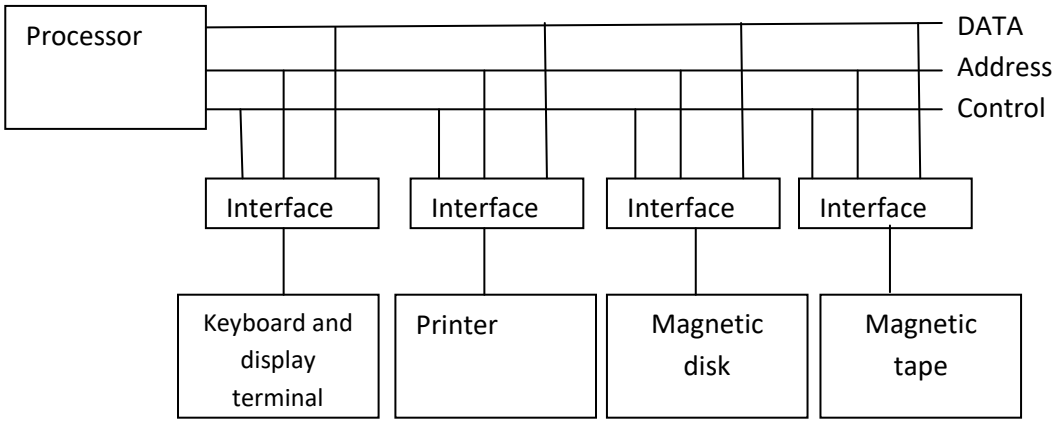
The purpose of the communication link is to resolve the differences that exist between the central computer and each peripheral.

The major differences are:

1. Peripherals are electromechanical and electromagnetic devices and their manner of operation is different from the operation of the CPU and memory, which are electronic devices. Therefore, a conversion of signal values may be required.
2. The data transfer rate of peripherals is usually slower than the transfer rate of the CPU, and consequently, a synchronization mechanism may be needed.
3. Data codes and formats in peripherals differ from the word format in the CPU and memory.
4. The operating modes of peripherals are different from each other and each must be controlled so as not to disturb the operation of other peripherals connected to the CPU.

To resolve these differences, computer systems include special hardware components between the CPU and peripherals to supervise and synchronize all input and output transfers. These components are called interface units because they interface between the processor bus and the peripheral device.

In addition, each device may have its own controller that supervises the operations of the particular mechanism in the peripheral.



--Connection of I/O bus to input output devices--

There are four types of commands that an interface may receive. They are classified as control, status, data output and data input.

A control command is issued to activate the peripheral and to inform it what to do. The particular control command issued depends on the peripheral, and each peripheral receives its own distinguished sequence of control commands, depending on its mode of operation.

A status command is used to test various status conditions in the interface and the peripheral. For example, the computer may wish to check the status of the peripheral before a transfer is initiated. During the

transfer, one or more errors may occur which are detected by the interface. These errors are designated by setting bits in a status register that the processor can read at certain intervals.

A data output command causes the interface to respond by transferring data from the bus into one of its registers. The interface responds to the address and command and transfers the information from the data lines in the bus to its buffer register.

The data input command is the opposite of the data output. In this case the interface receives an item of data from the peripheral and places it in its buffer register.

The processor checks if data are available by means of a status command and then issues a data input command. The interface places the data in the data lines, where they are accepted by the processor.

iii) PROGRAMMED I/O

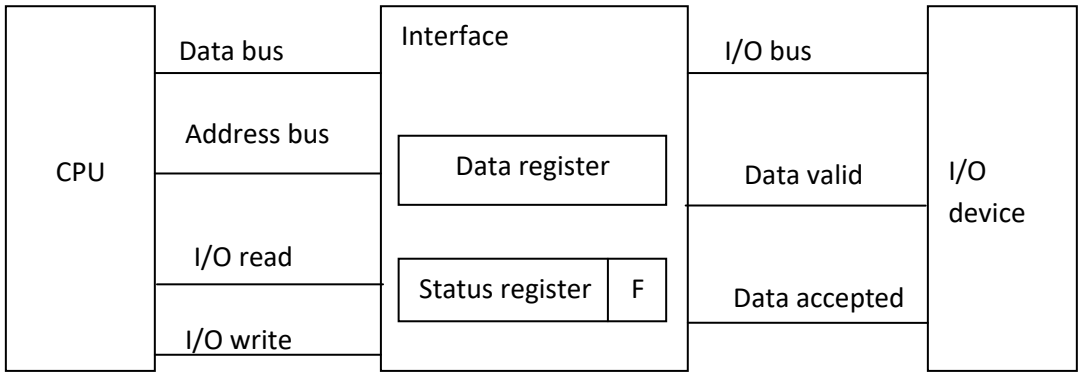
Programmed I/O operations are the result of I/O instructions written in the computer program. Each data item transfer is initiated by an instruction in the program. Usually, the transfer is to and from a CPU register and peripheral. Other instructions are needed to transfer the data to and from CPU and memory. Transferring data under program control requires constant monitoring of the peripheral by the CPU.

Once a data transfer is initiated, the CPU is required to monitor the interface to see when a transfer can again be made. It is up to the programmed instructions executed in the CPU to keep close tabs on everything that is taking place in the interface unit and the I/O device. In the programmed IO method, the CPU stays in a program loop until the IO unit indicates that it is ready for data transfer. This is a time consuming process since it keeps the processor busy needlessly.

Example of Programmed I/O:

In the programmed I/O method, the I/O device does not have direct access to memory. A transfer from an I/O device to memory requires the execution of several instructions by the CPU, including an input instruction to transfer the data from the device to the CPU and a store instruction to transfer the data from the CPU to memory.

Other instructions may be needed to verify that the data are available from the device and to count the numbers of words transferred. An example of data transfer from an I/O device through an interface into the CPU is shown in the figure.



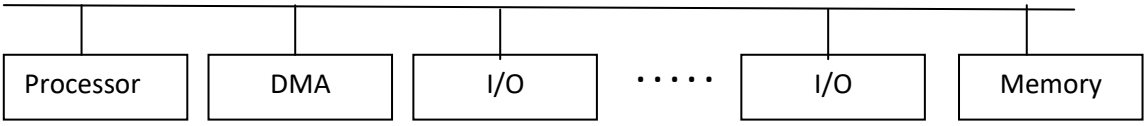
The device transfers bytes of data one at a time as they are available. When a byte of data is available, the device places it in the I/O bus and enables its data valid line. The interface accepts the byte into its data register and enables the data accepted line. the interface sets a bit in the status register that we will refer to as an F or **"flag"** bit. The device can now disable the data valid line, but it will not transfer another byte until data accepted line is disabled by the interface.

A program is written for the computer to check the flag in the status register to determine if a byte has been placed in the data register by the I/O device. This is done by reading the status register into a CPU register and checking the value of the flag bit. If the flag is equal to 1, the CPU reads the data from the data register. The flag bit is then cleared to 0 by either the CPU or the interface, depending on how the interface circuits are designed. One the flag is cleared, the interface disables the data accepted line and the device can then transfer the next data byte.

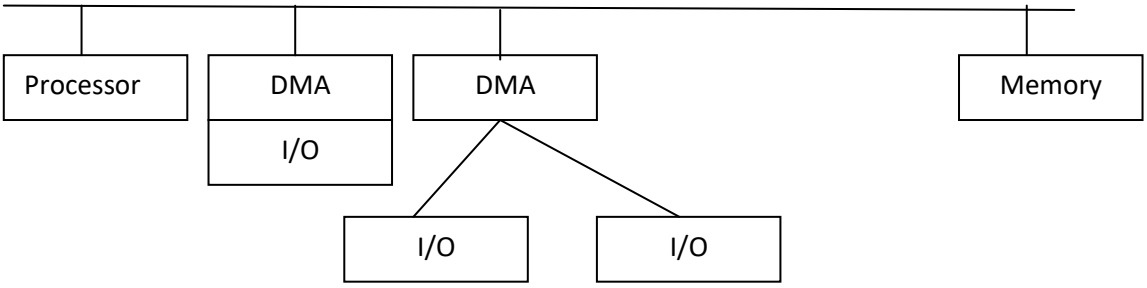
Each byte is read into a CPU register and then transferred to memory with a store instruction. A common IO programming task is to transfer a block of words from an I/O device and store them in a memory buffer. The programmed I/O method is particularly useful in small low speed computers.

iv) DIRECT MEMORY ACCESS (DMA)

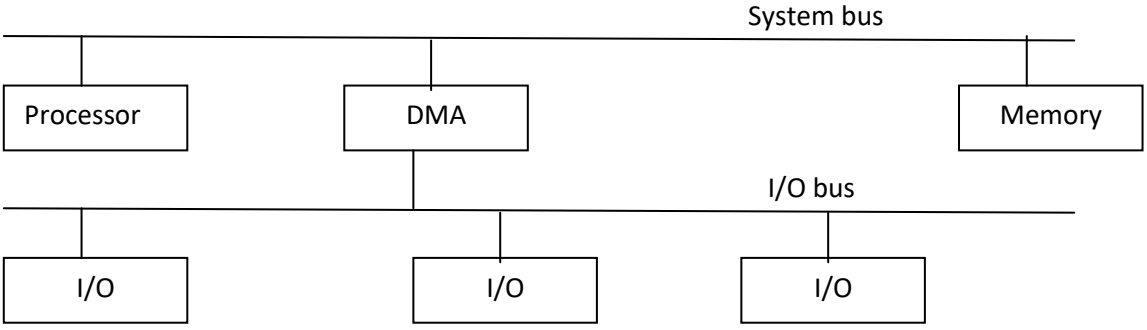
The transfer of data between a fast storage device such as magnetic disk and memory is often limited by the speed of the CPU. Removing the CPU from the path and letting the peripheral device manage the memory buses directly would improve the speed of transfer. This transfer technique is called direct memory access (DMA).



1)--Single bus, detached DMA--

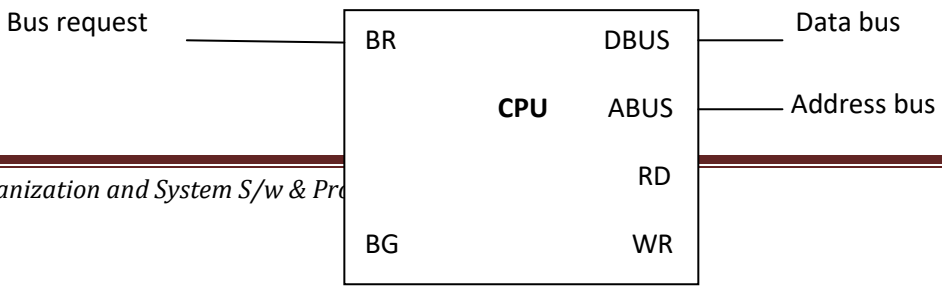


2)--Single bus, Integrated DMA-I/O--



3)--I/O Bus--

During DMA transfer, the CPU is idle and has no control of the memory buses. A DMA controller takes over the buses to manage the transfer directly between the I/O device and memory. The CPU may be placed in an idle state in a variety of ways. One common method extensively used in microprocessors is to disable the buses through special control signals.



Read

Write

Bus grant

[High-impedance (disable) when BG is enabled]

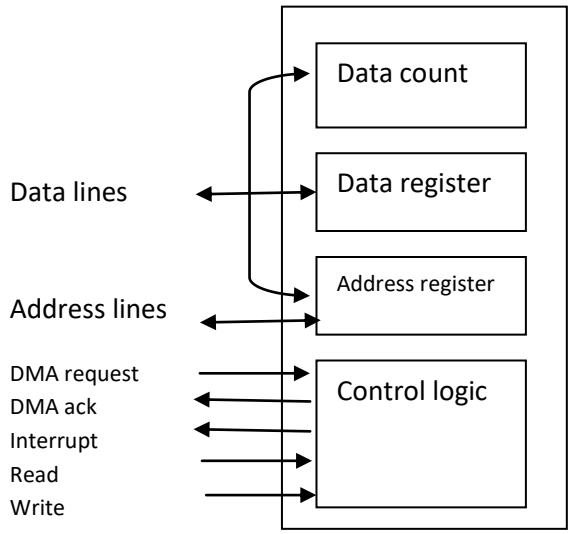
The figure shows two control signals in the CPU that facilitate the DMA transfer. The bus request (BR) input is used by the DMA controller to request the CPU to release control of the buses. When this input is active, the CPU terminates the execution of the current instruction and places the address bus, the data bus, and the read and write lines into a high-impedance state. The high-impedance state behaves like an open circuit, which means that the output is disconnected and does not have a logic significance.

The CPU activates the bus grant (BG) output to inform the external DMA that the buses are in the high impedance state. The DMA that originated the bus request can now take control of the buses to conduct memory transfers without processor intervention. When the DMA terminates the transfer it disables the bus request line.

The CPU disables the bus grant, takes control of the buses, and returns to its normal operation. When the DMA takes control of the bus system, a communicates directly with the memory. An alternative technique called cycle stealing allows the DMA controller to transfer one data word at a time, after which a must return control of the buses to the CPU. The CPU merely delays its operation for one memory cycle to allow the direct memory I/O transfer to "steal" one memory cycle.

When the transfer is complete, the DMA module sends an interrupt signal to the processor. Thus, the processor is involved only at the beginning and end of the transfer.

Typical DMA block diagram:



When the processor wishes to read or write a block of data, it issues a command to the DMA module, by sending to the DMA module the following information:

- Whether a read or write is requested, using the read or write control line between the processor and the DMA module.
- Address of the I/O device involved, communicated on the data lines
- The starting location in memory to read from or write to, communicated on the data lines
- The starting location in memory to read from or write to, communicated on the data lines and stored by the DMA module in its address register.

- The number of words to be read or written, again communicated via the data lines and stored in the stored count register.

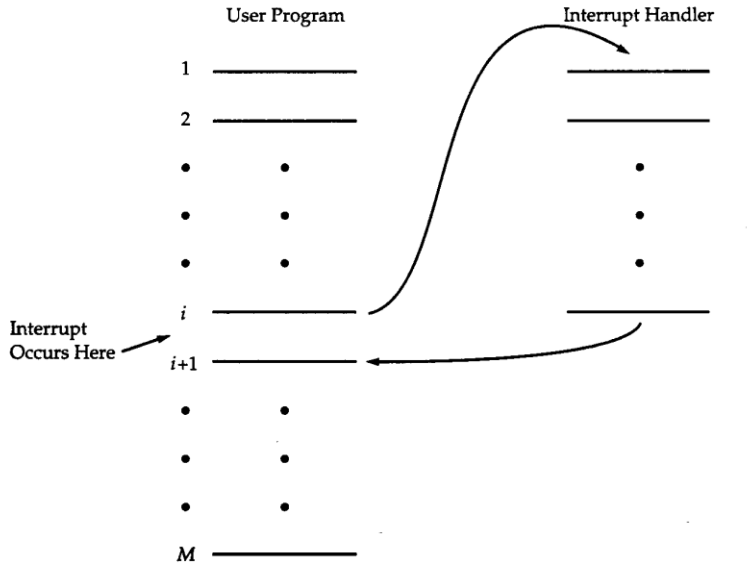
The processor then continuous with other work.

v) INTERRUPT CONTROLLED I/O

The mechanism by which other system modules may interrupt the normal processing of the CPU. These devices are 1-10 orders of magnitude slower than the CPU

– CPU can waste vast amounts of processing cycles waiting for these slow devices to perform their tasks

Interrupts let the CPU execute its normal instruction sequence and pause to service the external devices only when they signal (the interrupts) that they are ready for the CPU's attention. The processor and the O/S are responsible for recognizing an interrupt, suspending the user program, servicing the interrupt, and then resuming the user program

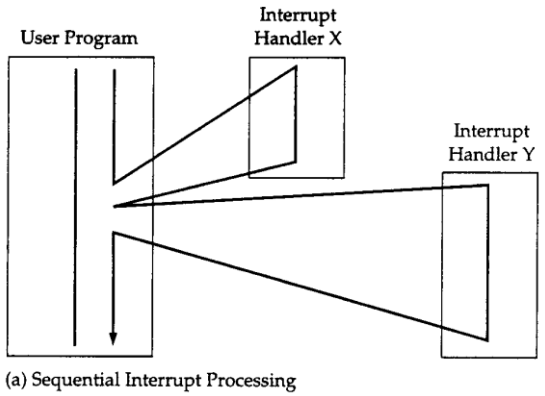


--Transfer of control via interrupts--

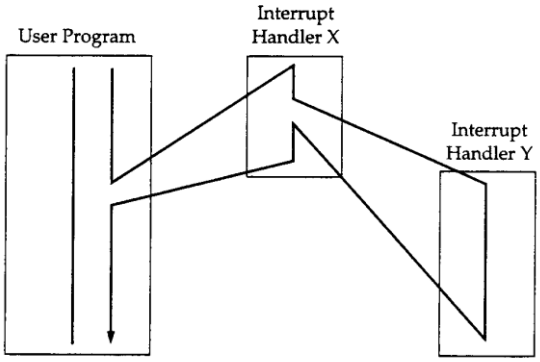
Interrupts are processed in an interrupt cycle within the overall instruction cycle
– At the end of an instruction cycle (operand storage step), check to see if any interrupts are pending
– If there aren't any, proceed with the next instruction
– If there are
» Suspend execution of the program and save its "state"
» Jump to the interrupt service routine(ISR) and resume the "normal" instruction cycle
» When the ISR is completed, restore the state of the program and resume its operation

Multiple Interrupts
– A typical system can support several to several dozen interrupts
– How should the system respond if more than 1 interrupt occurs at the same time?
» Systems prioritize the various interrupts
» At the start of the interrupt cycle, the highest priority pending interrupt will be serviced
» Remaining interrupt requests will be serviced in turn
– What if an interrupt occurs while an ISR is being executed (a result of a previous interrupt)
» Ignore the second interrupt (by disabling interrupts) until the ISR completes -- e.g., MC68HC11 microcontroller

» Recognize and service the interrupt only if it has a higher priority than the one currently being serviced -- e.g., 8085



(a) Sequential Interrupt Processing



(b) Nested Interrupt Processing

--Transfer of control with multiple interrupts--

vi) I/O PROCESSOR

Instead of having each interface communicate with the CPU, a computer may incorporate one or more external processors and assign them the task of communicating directly will all I/O devices. an input output processor(IOP) may be classified as a processor with direct memory access capability that communicates with I/O devices. in this configuration, the computer system can be divided into a memory unit, and a number of processors comprised of the CPU and one or more IOPs.

A processor that communicates with remote terminals over telephone and other communication media in a serial fashion is called a data communication processor (DCP). The IOP is similar to a CPU except that it is designed to handle the details of I/O processing. The IOP can fetch and execute its own instructions. IOP instructions are specifically designed to facilitate I/O transfers. In addition, the IOP can perform other processing tasks, such as arithmetic, logic, branching and code translation.

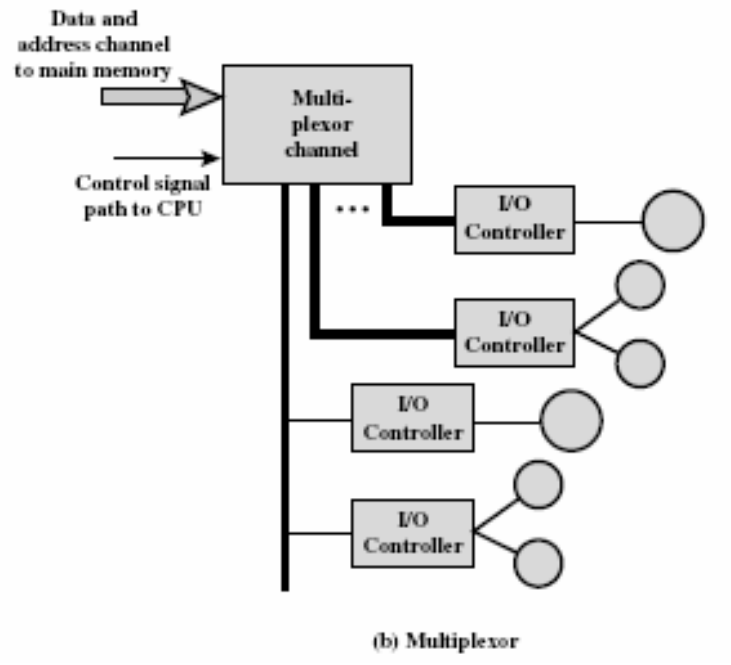
The data formats of peripherals devices differ from memory and CPU data formats. The IOP must structure data words from many different sources.

The CPU informs the IOP where to find the commands in memory when it is time to execute the I/O program.

The processor in the IBM 370 computer is called a channel. A typical computer system configuration includes a number of channels with each channel attached to one or more I/O devices. there are three types of channels:

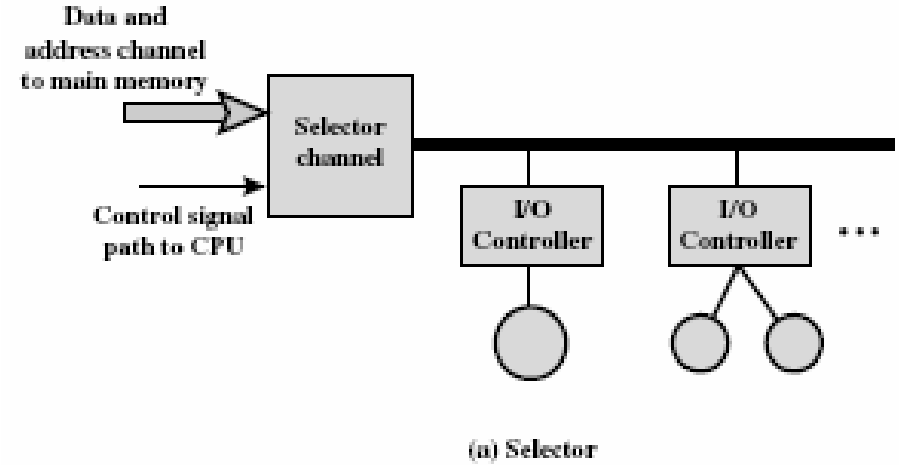
Multiplexer, selector, block multiplexer

Multiplexer channel: it can be connected to a number of slow and medium speed devices and is capable of operating with a number of I/O devices simultaneously.



(b) Multiplexor

Selector channel : it is designed to handle one I/O operation at a time and is normally used to control one high speed device.



(a) Selector

Block multiplexer channel: it combines the features of both the multiplexer and selector channels. It provides a connection to a number of high speed devices, but all I/O transfers are conducted with an entire block of data.

The CPU communicates directly with the channels through dedicated control lines and indirectly through reserved storage areas in memory.

vii) SERIAL COMMUNICATION

A data communication processor is an I/O processor that distributes and collects data from many remote terminals connected through telephone and other communication lines. it is a specialized I/O processor designed to communicate directly with data communication networks.

With the use of a data communication processor, the computer can service fragments of each network demand in an interspersed manner and thus have the apparent behaviour of serving many users at once. In this way the computer is able to operate efficiently in a time-sharing environment.

The task of the data communication processor is to transmit and collect digital information to and from each terminal. Also communicate with the CPU and memory in the same manner as any I/O processor.

Since telephone lines were originally designed for voice communication and computers communicate in terms of digital signals, some form of conversion must be used. The converters are called modems. A modem converts digital signals into audio tones to be transmitted over telephone lines and vice versa.

A communication line may be connected to a synchronous or asynchronous interface, depending on the transmission method of the remote terminal.

Synchronous transmission does not use start-stop bits to frame characters and, therefore, makes more efficient use of the communication link. High speed devices use synchronous transmission to realize this efficiency. Contrary to asynchronous transmission, where each character can be sent separately with its own start and stop bits.

In synchronous transmission, where an entire block of characters is transmitted, each character has a parity bit for receiver to check(ie., to check whether error occurred or not during the transmission). If the receiver finds an error in the transmitted block, it informs the sender to retransmit the same block once again.

Data can be transmitted between two points in three different modes: Simples, half-duplex, or full duplex.

A simplex line carries information in one direction only. The receiver cannot communicate with the transmitter to indicate the occurrence or errors. Examples of simplex transmission are radio and television broadcasting.

A half duplex transmission system is one that is capable of transmitting in both directions but data can be transmitted in only one direction at a time. The time required to switch a half-duplex line from one direction to the other is called the turnaround time.

A full duplex transmission can send and receive data in both directions simultaneously. This can be achieved by means of a four wire link, with a different pair of wires dedicated to each direction of transmission.

The communication lines, modems, and other equipment used in the transmission of information between two or more stations is called a data link.

A data link control protocol is a set of rules that are followed by interconnecting computers and terminals to ensure the orderly transfer of information. The purpose of a data link protocol is to establish and terminate a connection between two stations, to identify the sender and receiver, to ensure that all messages are passed correctly without errors, and to handle all control functions involved in a sequence of data transfers.

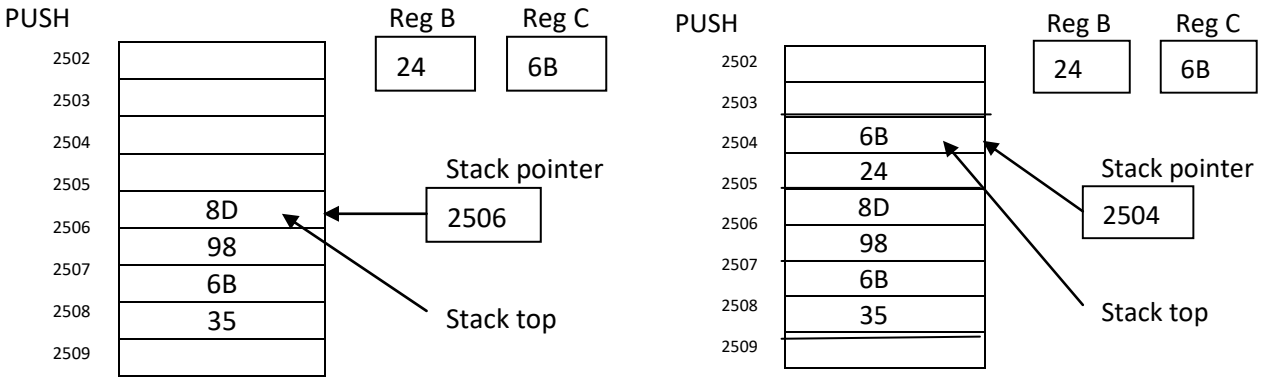
viii) STACK AND SUBROUTINES

During program execution the contents of certain registers are needed to be saved because the registers are required for some other operations in the subsequent steps and the saved contents will be needed at the later stage of the program execution. The contents of registers are saved in certain memory locations set aside by the programmer for this purpose in the very beginning while writing the program. The memory space set aside for this purpose is called stack. Thereafter the registers are used for other operations. After

completion of these operations the saved contents are brought back from the memory to the registers. The last occupied memory location of the stack is known as stacktop.

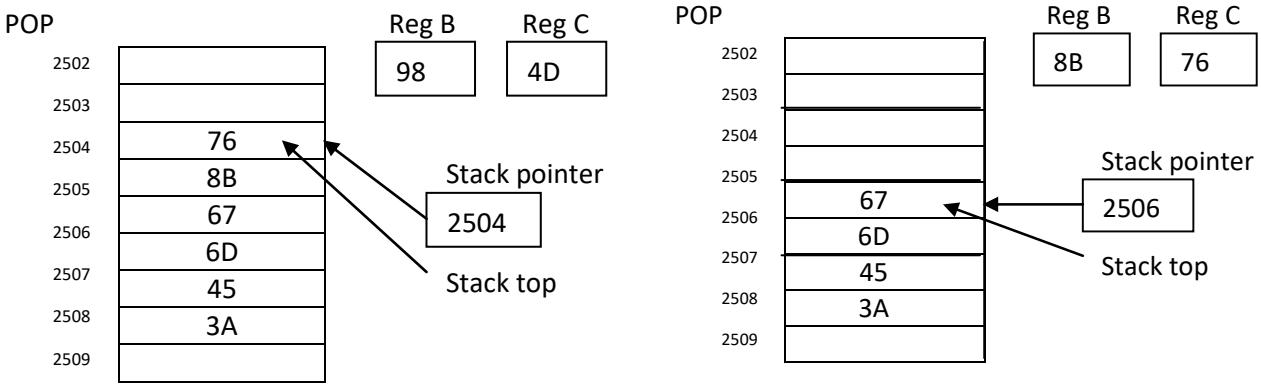
A special register known as stack pointer(SP) holds the address of the stacktop to keep track of the stack memory locations. Any area of the RAM can be used as stack, but usually the last user's area of the RAM is set aside for this purpose.

The PUSH instruction is used to transfer the contents of the registers to the stack, and the POP instruction to bring back the contents from the stack to the registers. The data are stored in the stack on last in first out (LIFO) principle.



--Before PUSH-- --After PUSH--

Figure shows the stack position before PUSH operation. Suppose the contents of B-C register pair are to be saved. Second one shows the stack position after PUSH operation.

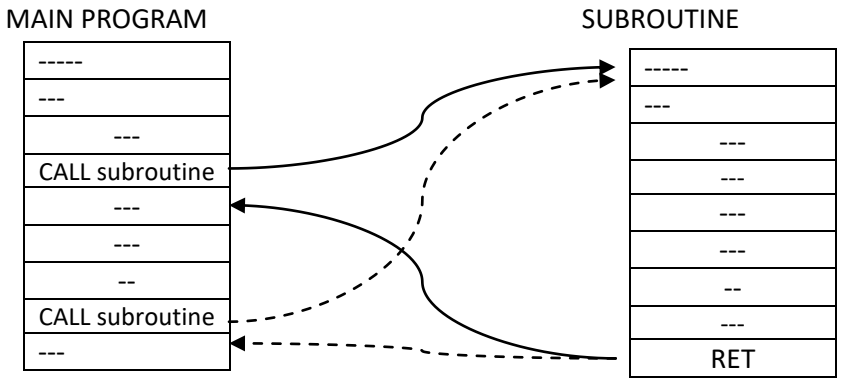


The stack positions before and after POP operation are shown in Figures. Stack access is faster than memory access.

SUBROUNTINES:

Often, we need to execute several times a sequence of instructions to perform a subtask within a program. In such cases, we write the sequence of instructions which perform subtask as a separate subprogram. This subprogram can be called at any point in the main program whenever required using a special instruction (CALL) for this purpose. Such subprograms are known as subroutines or procedures. On the completion of the subroutine, the execution of the main program from the next instruction after the CALL instruction in the main program. Before starting the execution of a subroutine the content of the program counter, PC is saved on the

stack. The contents of PC point to the instruction following the CALL instruction(ie., instruction next to the CALL instruction in the main program).



At the end of a subroutine RET(return) instruction is used. It restores the contents of PC so that the control is transferred back to the main program. Figure shows the main program, subroutine and CALL-RET structure.

ix) LANGUAGES & PROGRAM DESIGN

To perform a particular task the programmer writes a sequence of instructions, called program. An instruction is a command given to the computer to perform certain specified operation on given data. A set of programs written for a computer is called software. The software needed to execute the user's program is known as system software. The system software consists of operating system, assembler, compiler, interpreter, debugging programs, text editors etc.,

A program which is prepared by a programmer to solve certain problem or to perform certain specified task is known as user's program.

The operating system is a collection programs that controls the overall operation of the computer. The term firmware is used for the software stored in read only storage devices.

Machine Language:

A computer understands information composed of only zeros and ones and hence, it uses binary digits for its operation. The computers instruction are coded and stored in the memory in the form of 0s and 1s. A program written in the form of 0s and 1s called machine language program. There is a specific binary code for each instruction. For example, to add the contents of register A and register B, the binary code is 10000000 for Intel 8085. Each microprocessor has its own instruction set and corresponding machine codes.

Assembly Language:

The writing of programs in machine language is very difficult, tiresome and boring job for a programmer. To solve this problem and to facilitate programmer easily understandable languages have been developed. Assembly language is one of them. Programs can easily be written in alphanumeric symbols instead of 0s and 1s. For example, ADD for addition, SUB for subtraction, CMP for comparison,. Etc., such symbols are known as mnemonics. A program written in mnemonics is called assembly language program.

Assembler: a program which translates an assembly language program into a machine language program is called an assembler.

Advantages: the computation time for an assembly language program is less. An assembly language program runs faster to produce the desired result..

Disadvantage: programming is difficult and time consuming, machine oriented ie., programmer must have the detailed knowledge of registers and instruction set of the computer, connections of ports to the peripherals etc., program is not portable.

High level language(HLL):

HLL permit programmers to describe tasks in a form which is problem oriented or object oriented rather than computer oriented. The instructions written in a high level language are called statement. The statements more clearly resemble English and mathematics as compared to mnemonics in assembly languages. Example, BASIC, PASCAL, FORTRAN, COBOL, C, C++, LISP, Java etc., A HLL is independent of a computer.

Advantage : easier to learn, faster to write, provide better documentation, programs are portable.

PROGRAM DESIGN:

The important techniques which are useful in designing programs are as follows:

i) Modular Programming ii) Structured Programming iii) Top-down and bottom up design iv) Object oriented programming (OOP)

i) Modular Programming:

When a program becomes very long and complex, it becomes a very difficult task for the programmer to design, test and debug such a program. Therefore, a long program can be divided into smaller programs called modules. A division of a long program into smaller programs(or modules) is called modular programming.

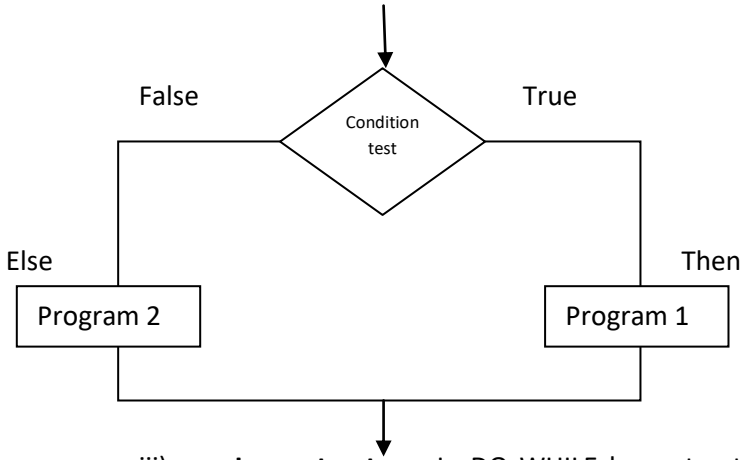
ii) Structured Programming:

With the increasing capacity of the memory, the programs also became longer and longer. The long and complex programs may be well understood by the programmers who developed but not by the persons who had to maintain them. To overcome this difficulty a technique known as structured programming was developed to write a program.

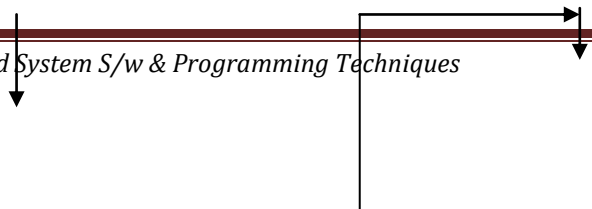
The three basic logic structures are:

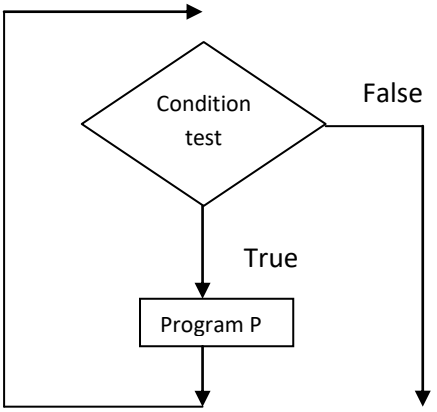
i) **Simple sequence structure**: it is a linear structure in which instructions or statements are executed consecutively in a sequence.

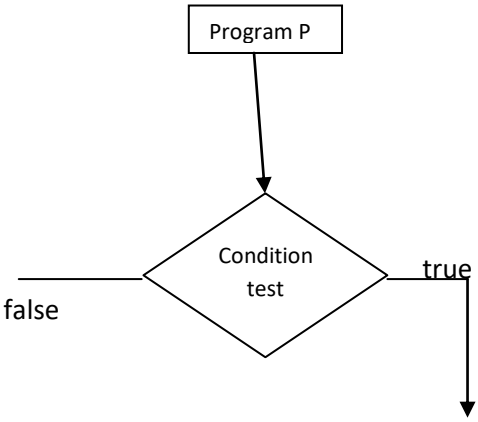
ii) **Conditional structure**: in this structure a condition is tested. The condition is followed by two alternative program control paths.



iii) **Loop structure**: In DO WHILE loop structure, the program is executed once or more while the condition is true. When the condition becomes false, the looping process ends. In DO UNTIL structure the looping process is repeated until a condition becomes true.







--do while loop structure-- -- do until loop structure--

iii) ***Top-down and Bottom Up design:***

In top down technique the design of the system program is started at the system level. The programmer first develops the overall supervisor program which is used to outline and control subprograms.

In bottom up design is a traditional design technique. In this inner subprograms are prepared first for specific tasks and then integrated into a complete system. It should be used for frequently used subprograms whose speed is critical to the speed of the whole problem and whose functions are clearly understood initially. This ensures that correct parameters are selected for the subprograms.

iv) Object oriented Programming (OOP)

The main problem with computer programs is complexity. Software errors in a large program may create several problems. OOP offers a new and powerful way to cope with complexity. It results in the availability of more easily maintained programs. A few terms in connection with OOP are explained below:

Object: An object consists of data and functions. An object oriented language relates data to its functions. An object may be an item, person or any other entity. The data and functions related to an object are given within the object.

Class: the two examples given above Lecturer and Professor are similar. Both objects have the same set of data and functions, and hence, they can be placed in the same class (or category). This class can be named "employee". The class employee includes clerks, librarians, peons, etc.,

x) S/W DEVELOPMENT

While preparing programs the following factors should get due considerations:

***Reliability*:** A program must work reliably. It should perform the task properly for which it has been developed.

Speed: A program must execute the specified task quickly. The time taken by a program to perform a given task should be as minimum as possible.

Programming Time and Cost: the cost of processors, memory and peripherals are decreasing but the cost of programming is rising. Due to this reason more attention is being given on programming techniques like structured programming and top down design which increase programmer's output.

Ease of Use: A program must be easily understood by others.

Error Tolerance: a program must react to errors. It should give some information regarding errors or malfunctions without shutting the entire system down.

Extendibility: A program that can be extended to tasks other than for which it has been designed and developed is definitely a better program. The modular programming is more useful in attributing the feature of extendibility to a program.

Stages of Software Development:

The software development may be divided into the following stages:

Problem Definition: At this stage the problem to be solved or the task to be performed is defined. Inputs, outputs, processing requirements, system constraints such as execution time, accuracy etc., and error handling methods are specified.

Program Design: At this stage the program is designed to meet the specified requirements according to its definition. The important design techniques are top-down, structured programming, modular programming and flowcharting. The flowcharts help in explaining programs and describing program structure.

Preparation of Actual Program : At this stage computer instructions are written according to its design.

Testing: At this stage program is tested to check whether it performs the required task or solves the given problem. This stage is also called validation.

Debugging: Errors in the program are detected and corrected. Important debugging tools are simulators, logic analyzers, breakpoints, trace rountines, memory dumps, etc.,

Documentation: It indicates what functions are performed by the program and how these functions are carried out. It helps users to understand and maintain the program.

Maintenance: At this stage programs are corrected and updated to meet the need of changing conditions.

Extension and Redesign: A program can be extended to other tasks. If necessary, it can be redesigned to get its improved version or to perform other tasks.

xi) OPERATING SYSTEM

An operating system is a collection of programs that controls the overall operation of a computer. It allows users to format disks; create, print, copy, delete and display files, read data from files, write data to files, control most input/output operations, execute programs, allocate memory locations, process interrupts etc.,
It provides users an interface to computing resources. It processes user's commands.

In short we can say that an operating system monitors the execution of user programs and the use of resources.

The physical resources available in a computer are CPU, memory and I/O devices. In a multiuser system a resource is used by a number of program and hence, the operating system has to maintain scheduling.

Usually, operating systems are large. Most of them are too large to be stored in the memory at a time. Therefore, they can be divided into a number of parts. Some portions of the operating system must always be present in the memory. These sections perform the basic operations such as starting and terminating user programs, allocations of memory and files and basic input/output operations. Interrupts are also handled by these resident portions of the operating system.

The portion of the operating system which is always present in the memory is called nucleous and kernel. The kernel is a master program of operating system. It co-ordinates all other parts of the operating system. It is also called supervisor. Other portions of the operating system which are brought into the memory when needed and removed when not needed, are called transient programs.

Booting(boot strapping)

The process of loading the operating system into memory is known as booting. When a computer is turned on the operating system must be brought into the computer's memory from the hard disk memory. The process is normally started by a small program called bootstrap loader. This program resides in a ROM as firmware. A computer is designed to fetch its very first instruction from the ROM when power is turned on. The first instruction is bootstrap loader. It is a very simple program sufficient only to direct the CPU to look for a specific

file ie., operating system file, on the disk memory, and execute the instructions stored in the file. The file contains machine codes of the operating system. The first part of instructions in the file contains codes to direct the CPU to continue loading the rest of the operating system into the memory. When the operating system is fully loaded into the memory, the computer is ready to accept user's commands.

Some important operating systems are : UNIX, XENIX(version of UNIX, developed by Microsoft) , Windows Family of Operating system(windows-95-98,ME,XP, 2000,2003...),Sun's Solaris, Apple's Macintosh (first operating system), Novell NetWare(network operating system)..

xii) UTILITY PROGRAMS

These are the set of programs, which help the users maintenance of the computer system. They are generally small programs, having specific tasks to perform.
There are two types of utility programs: files management utilities and program development utilities.
Most common functions of system utilities are

1. ***File management :*** these utilities make easier to manage data files. Many programs or commands are available to help users to find the files, create, directories, move, and delete files etc.,
2. ***Backup:*** it is the process of safeguarding the data. Backups become very useful in case data files corrupted or unfortunately deleted.
3. ***Data Recovery:*** it si the process of retrieving deleted or inaccessible data from failed storage media such as hard disk or optical devices using this tools, technicians can successfully recover the 100% of lost data.
4. ***Virus Protection :*** This utility provides the security to the system from virus attack. Antivirus programs are essential system utility for a computer system functioning in network. Eg., Norton antivirus, Avast, Mcafee antivirus...
5. ***Disk Management:*** Disk defragmenter, data compressor and disk formatting are the various system software included in disk management programs.
6. ***Firewall*** : Firewalls forms a carrier between networked computer within organization and outside the organization to protect data.
7. ***Disk cleanup***: Disk cleanup can be significantly improve the system performance by deleting the unwanted files which no longer used.

xiii) APPLICATION PACKAGES

An application package or application program is the software that has been written to process or perform specific job. Application package are available for all types of tasks. They may be for business applications, engineering designs, home applications, teaching aids etc., some examples are: word processing packages for processing and manipulating text, spreadsheet packages for calculating finances and data analysis, CAD for designing or drafting, dBASE to work with records (management of files, management of database, storage and retrieval of information), engineering design packages, MATLAB for scientific and engineering computation etc.,