SECTION "A" – HARDWARE

Q. 2 (a) (i) Basic Components of the CPU:

Arithmetic Logic Unit:

Arithmetic logic unit (ALU) of a computer system is the place where actual execution of instructions takes place during processing operation. To be more precise, calculations are performed and all comparisons (decisions) are made in the ALU. Data and instructions stored in primary storage before processing are transferred as and when needed to the ALU where processing takes place. Intermediate results generated in the ALU are temporarily transferred back to primary storage until needed later. Hence, data may move from primary storage to ALU and back again to storage many times before processing is over.

Control Unit:

Control unit acts as a central nervous system for other components of a computer system. It manages and coordinates the entire computer system. It obtains instructions from the program stored in main memory, interprets the instructions, and issues signals causing other units of the system to execute them.

Central Processing Unit:

Control unit (CU) and arithmetic logic unit (ALU) of a computer system are together known as the Central Processing Unit (CPU) is the brain of a computer system. In a computer system, all major calculations and comparisons take place inside the CPU and the CPU is responsible for activating and controlling the operations of other units of the computer system.

(ii) Microprocessor:

A *microprocessor* contains all circuits needed to perform arithmetic logic and control functions, the core activities of all computers, on a single chip. Hence, it became possible to build a complete computer with a microprocessor, a few additional primary storage chips, and other support circuitry. It started a new social revolution – *personal computer (PC)* revolution. Overnight computers became incredibly compact. They became inexpensive to make, and suddenly it became possible for anyone to own a computer.

Q. 2 (b) (i) Sequential Access and Direct Access Devices:

A *sequential access storage device* is one in which the arrival at desired location may be preceded by sequenci8ng through other locations so that access time varies according to location. That is, information on a sequential access device can be retrieved in the same sequence only in which it is stored. Sequential processing is suitable for such applications like preparation of monthly pay slips, monthly electricity bills, etc., where most, if not all, of the data records need to be processed one after another. In these applications, data records for every employee or customer needs to be processed at scheduled intervals (in this case monthly). However, while working with a sequential access device if an address is required out of order, it can be reached only by searching through all the addresses stored before it. Magnetic tape is an example of a sequential access storage device.

In many applications, we need to access information in a more direct manner than sequential access devices allow. For example, in a computerized bank, at any instance, it may be required to determine the exact balance in the savings account of a customer. Similarly, in a computerized airline ticket booking system immediate access may be required to reservation system records to find out if seats are currently available on a particular flight. In such applications, if we use a sequential access device for data storage, the time taken to access the desired information may be long enough to cause frustration to the customer. Secondary storage devices exist that permit access to individual information in a more direct or immediate manner. These direct access devices are also called random access devices because out of all the information stored, any randomly chosen piece of information can be accessed immediately. Hence, a random access storage device is one in which any location in the device may be selected at random, access to the information stored is direct, and approximately equal access time is required for each location.

(ii) Advantages of IC Technology Over Transistor Technology:

In 1958, Jack St. Clair Kilby and Robert Noyce invented the first integrated circuit. *Integrated circuits* (called ICs) are circuits consisting of several electronic components like transistors, resistors, and capacitors grown on a single chip of silicon eliminating wired interconnection between components.

Characteristic Features of Third-Generation Computers:

Characteristic features of third-generation computers are as follows:

1. They were more powerful than second-generation computers. They were capable of performing about 1 million instructions per second.
2. They were smaller than second-generation computers requiring smaller space.
3. They consumed less power and dissipated less heat than second-generation computers. The rooms/areas in which third-generation computers were located still required to be properly air-conditioned.
4. They were more reliable and less prone to hardware failures than second-generation computers requiring lower maintenance cost.
5. They had faster and larger primary and secondary storage as compared to second-generation computers.
6. They were general-purpose machines suitable for both scientific and commercial applications.
7. Their manufacturing did not require manual assembly of individual components into electronic circuits resulting in reduced human labour and cost involved at assembly stage. Hence, commercial production of these systems was easier and cheaper. However, highly sophisticated technology and expensive setup was required for the manufacture of IC chips.
8. Standardization of high-level programming languages allowed programs written for one computer to be easily ported to and executed on another computer.
9. Timesharing operating system allowed interactive usage and simultaneous use of these systems by multiple users.
10. Timesharing operating system helped in drastically improving the productivity of programmers cutting down the time and cost of program development by several fold.
11. Timesharing operating system also made on-line systems feasible resulting in usage of these systems for new on-line applications.
12. Unbundling of software from hardware gave users of these systems an opportunity to invest only in software of their need and value.

13. Minicomputers of third-generation made computers affordable even by smaller companies.

Q. 3 (a) Registers:

As instructions are interpreted and executed by a computer's CPU, there is movement of information between various units of the computers. In order to handle this process satisfactorily and to speed up the rate of information transfer, a number of special memory units called *registers* are used. These registers are used to hold information on a temporary basis and are part of the CPU (not main memory).

Commonly Used Registers:

1. Memory address register (MAR);
2. Memory buffer register (MBR);
3. Program control register (PC);
4. Accumulator register (A);
5. Instruction register (I);
6. Input/output register (I/O).

Q. 3 (b) Reasons Why Optical Disk is a Better Storage Medium for Data Archiving:

1. Cost-per-bit of storage for optical disks is very low because of their low cost and high storage density. Additional cost benefit comes from the fact that some optical disks can be erased and reused many times.
2. Use of a single spiral track makes optical disks an ideal medium for reading large blocks of sequential data such as audio or video.
3. Optical disk drives do not have any mechanical read/write heads to rub against or crash into the disk surface. This makes optical disks more reliable storage medium than magnetic tapes or magnetic disks.
4. Optical disks have data storage life in excess of 30 years. This makes them better storage medium for data archiving as compared to magnetic tapes or magnetic disks.
5. Since data once storage on CD-ROM/WROM disks becomes permanent, the danger of stored data getting inadvertently erased/overwritten is not there.
6. Due to their compact size and lightweight, optical disks are easy to handle, store, and port from one place to another.
7. Audio CDs can be played on a computer having a CD-ROM drive along with a sound card and speakers. This allows computer systems to be also used as music systems, whenever desired.
8. A computer having a DVD drive can be used to play DVDs allowing it to used for watching videos such as movies.

Q. 3 (c) Parameters of Disk Access Time:

Disk access time depends on the following three parameters:

1. **Seek Time:**

   As soon as a disk unit receives a read/write command, it positions the read/write heads on the specified track (cylinder) number first by moving the access arms assembly in the proper direction. The time required to position the read/write head over the desired track/cylinder is called seek time. Seek time varies depending on the position of the access arms assembly when a read/write command is received. If the access arms assembly is positioned on the outer most rack and the track to be reached is the inner most one, the seek time will be maximum. It will be maximum. It will be zero if the access arms assembly already happens to be on the desired track. The average seek time is thus specified for most systems. It is of the order of 10 to 100 milliseconds.

We also learnt that some disk systems have multiple read/write heads on each access arm. This is done to reduce seek time. For example, a disk system may have two sets of read/write heads for each surface, one for reading/writing on the inside tracks and another for the outside tracks. This will reduce average seek time by half because each read/write head needs to cover and move across only half of the total number of tracks.

2. **Latency:**

Once the heads are positioned on the desired track, the head on the specified surface is activated. Since the disk is continuously rotating, this head should wait for the desired data (specified sector) to come under it. This rotational waiting time, i.e., the time required to spin the desired sector under the head is called latency. Latency, also known as rotational delay time, is also a variable and depends on the distance of the desired sector from the initial position of the head on the specified track. It also depends on the rotational speed of the disk that may be anywhere from 300 rpm (rotations per minute) to 7200 rpm. Modern ultra fast disk can reach 10K rpm or more. An average latency time is thus normally specified and is of the order of 5 to 80 milliseconds. Note that average latency of a disk system is equal to half the time taken by the disk to rotate once. Hence, average latency of a disk system with rotational speed of 3600 rpm will be 0.5/3600 minutes = 8.3 milliseconds.

3. **Transfer Rate:**

Transfer rate refers to the rate at which data is read from or written to a disk. Once the read/write head is positioned over the addressed sector, the desired data is read/written at a speed determined by the rotational speed of the disk. If rotational speed of a disk is 3600 rpm and the disk has 125 sector/track with 512 bytes/sector, the amount of data transferred in one full revolution of the disk will be 125 x 512 = 64,000 bytes = 64K bytes (approximately). Hence, transfer rate of the disk system will be 64,000 x 3600/60 bytes/second = 38,40,000 bytes/second = 3.8 Megabytes/second (approximately). Notice that the transfer rate of a disk system depends on the density of stored data and rotational speed of the disk.

SECTION "B" – SOFTWARE

Q. 4 (a) Advantages and Limitations of High-level Languages:

High-level languages enjoy following advantages over assembly and machine languages:

1. **Machine independence:**

A program written in a high-level language can be executed on many different types of computers with very little or practically no effort of porting it on different computers. This means that a company changing computers, even to one from a different manufacturer, will not be required to rewrite all the programs that it is currently using. This also means that commercial software will have a larger market because in need not be restricted to one type of computer.

2. **Easier to learn and use:**

High-level languages are easier to learn because they are very similar to the natural languages used by us in our day-to-day life.

3. **Fewer errors:**

   While programming in a high-level language, a programmer need not worry about how and where to store the instructions and data of the program and need not write machine-level instructions for the steps to be carried out by the computer.

4. **Lower program preparation cost:**

   Writing programs in high-level languages requires less time and effort, ultimately leading to lower program preparation cost.

5. **Better documentation:**

   Statements of a program written in a high-level language are very similar to national language statements used by us in our day-to-day life. Hence, a programmer familiar with the problem domain can easily understand them.

6. **Easier to maintain:**

   Programs written in high-level languages are easier to maintain than assembly/machine language programs. This is because they are easier to understand, and hence, it is easier to locate, correct, and modify instructions whenever desired.

7. **Lower efficiency:**

   Generally, a program written in a high-level language has lower efficiency than one written in a machine/assembly language to do the same job. That is, programs written in high-level languages result in multiple machine language instructions that may not be optimized, taking more time to execute and requiring more main memory space.

Q. 4 (b) ACQUIRING SOFTWARE:

Earlier, application and system software were included in the purchase price of a computer. Today, software is usually not included in the purchase price of a computer. For most computer manufacturers, the purchase price of a computer includes only its hardware and a minimum of system software. A customer normally has to pay extra charges for additional system software and application software that he/she may wish to purchase.

Following are the advantages and limitations of buying a pre-written software package:

1. Pre-written software packages usually cost less because many customers share their development and maintenance cost.
2. With a pre-written software package, a user can start the planned activity almost immediately. The user need not wait for the software to be developed and tested. This may be very important, if the development and testing efforts involve several months.
3. Pre-written software packages are usually general purpose, so that they can meet the requirements of as many potential users as possible. Due to this, many times, the operating efficiency and capability to meet the specific needs of a user more effectively is not as good for pre-written software packages as for in-house developed software packages.

Q. 4 (c) i. **Multiprogramming:**

In a multiprogramming system, often there are multiple jobs in ready state (waiting for CPU to be allocated). Hence, when CPU becomes free, operating system must decide which of these ready jobs should be allocated to CPU for execution. Part of the operating system that takes this decision is called CPU scheduler, and the algorithm it uses for this is called CPU scheduling algorithm.

ii. **Multitasking:**

Technically speaking, multitasking is same as multiprogramming. Many authors do not distinguish between multiprogramming and multitasking because both refer to the same concept. However, some authors prefer to use the term multiprogramming for multi-user systems (systems that are used simultaneously by many users such as mainframe and server class systems), and multitasking for single-user systems (systems that are used by only one user at a time such as a personal computer or a notebook computer.

iii. **Multithreading:**

Treads are a popular way to improve application performance. In traditional operating systems, the basic unit of CPU utilization is a process. Each process has its own program counter, its own register states, its own stack, and its own address space (memory area allocated to it). On the other hand, in operating systems, with threads facility, the basic unit of CPU utilization is a thread. In these operating systems, a process consists of an address space and one or more threads of control. Each thread of a process has its own program counter, its own register states, and its own stack

iv. **Multiprocessing:**

The use of I/O processors improves the efficiency of a computer system by making concurrent input, processing, and output operations possible. CPU performs arithmetic and logical operations, while I/O processors carry out I/O operations concurrently.

Multiprocessing systems, however, require a very sophisticated operating system to schedule, balance, and coordinate the input, output, and processing activities of multiple processors. Designing such an operating system is complex and time taking. Moreover, multiprocessing systems have higher initial cost and their regular operation and maintenance is costlier than single-processor systems.

v. **Time-sharing:**

Time-sharing is a mechanism to provide simultaneous interactive use of a computer system by many users in such a way that each one feels that he/she is the sole user of the system. It uses multiprogramming with a special CPU scheduling algorithm to achieve this.

Q. 5 (a) Assembler:

A computer can directly execute only machine language programs that use numbers for representing instructions and storage locations. Hence, an assembly language program must be converted (translated) into its equivalent machine language program before it can be executed on the computer. This translation is done with the help of a translator program called assembler. Assembler is system software supplied by computer manufacturers. It translates an assembly language program into its equivalent machine language program. It is so called because in addition to translating, it also "assembles" the machine language program in main memory of the computer, and makes it ready for execution.

Compiler:

A computer can execute only machine language programs directly. Hence, a high-level language program must be converted (translated) into its equivalent machine language program before it can be executed on a computer. This translation is done with the help of a translator program called compiler. Hence, a compiler is a translator program (must more sophisticated than an assembler is) that translates a high-level language program into its equivalent machine language program. A compiler is so called because it compiles a set of machine language instructions for every program instruction of a high-level language.

Linker:

Small programs can be written by one programmer and can be stored in a single source code file. However, large size software often consists of several thousands, even several millions, of lines of program code. For such software, it is impractical to store its entire source code in a single file due to following reasons:

1. The large size of the file would make it very difficult, if not impossible, to work with. For example, it would not be possible to load the file for compilation on a computer with limited main memory capacity. In addition, while editing the file, it would be very tedious and time-consuming to locate a particular line of code.
2. It would make it difficult to deploy multiple programmers to work concurrently towards the development of the software for completing it within specified time.
3. Any change in the source program, no matter how small, would require the entire source program to be recompiled. Recompilation of large source programs is often a time-consuming process.

Interpreter:

Interpreter is another type of translator used to translate a high-level language program into its equivalent machine language program. It takes one statement of the high-level language program, translates it into machine language instructions, and then executes the resulting machine language instructions immediately. That is, in case of an interpreter, the translation, and execution, and execution processes alternate for each statement encountered in the high-level language program.

Q. 5 (b) Difference between Testing and Debugging:

Sr. No.	Testing	Debugging
1	Testing is the process of validating the correctness of a program. Its objective is to demonstrate that the program meets its design specifications.	Debugging is the process of eliminating errors in a program. Its objective is to detect the cause of error, and remove known errors in the program.
2	Testing is complete when all desired verifications against specifications are completed.	Debugging is complete when all known errors in the program are fixed. Note that, debugging process ends only temporarily because it restarts whenever new errors are detected in the program.
3	Testing is definable process that can and should be planned and scheduled properly.	Debugging, being a reactive process, cannot be planned ahead of time. It is carried out whenever errors are detected in a program.
4	Testing can begin in the early stages of software development. Although the test runs of a program are carried out only after the program is coded, but the decision of what to test, how to test, and with what	Debugging can begin only after the program is ready. The approach used for debugging largely depends on the personal choice of a programmer and the type of error in the program.

	kind of data to test can and should be done before the coding is started.	

SECTION "C" – NETWORK & SECURITY

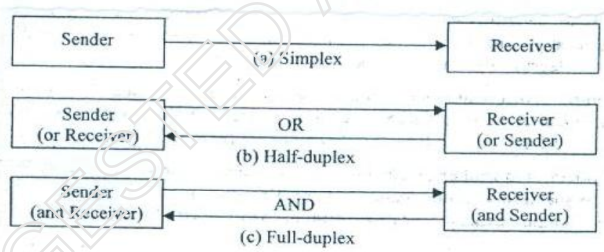
Q. 6 (a) i) **Narrowband.** Narrowband or sub-voice grade channels have speed in the range of 45 to 300 baud. They are used to handle low data volumes, and are adequate for low-speed devices. They are used mainly for telegraph lines and low speed terminals.

Voice band. Voice band channels have speed up to 9600 baud. They are so called because their major application is in ordinary telephone voice communication. They are used also for data transmission from slow I/O devices to CPU or vice versa.

Broadband. Broadband channels are used for transmission of large volumes of data at high speed. They have speed of 1 million baud or more. Broadband facility is used for high-speed computer-to-computer communication or for data transmission to several different devices simultaneously.

ii) The three modes of data transmission are simplex, half-duplex, and full-duplex described below:

!



!

Simplex:

A simplex communication system can transmit data in one direction only. Devices connected to such a circuit are either send-only (such as keyboard) or receive-only (such as printer).

Half-Duplex:

A half-duplex communication system can transmit data in both directions, but in only one direction at a time. Hence, it can alternately send and receive data. It requires two wires. It is most suitable for voice communication using telephones in which one person speaks at a time. It is suitable also for connecting a terminal to a computer in which the terminal transmits data and then the computer responds with an acknowledgement.

Full-Duplex:

A half-duplex system needs to switch direction each time the direction of data transfer reverses. This requires a special switching circuit and a delay of about 150 milliseconds. When compared with the high processing capabilities of modern computers, this delay is often unacceptable. Moreover, some applications require bi-directional data transfer simultaneously. In such cases, a full duplex system is used that allows data to flow in both directions simultaneously. It requires four wires. It improves efficiency because it eliminates the direction switching delay of a half-duplex sysem.

Q. 6 (b) i) **Switching Techniques:**

In its simplest form, data communication takes place between two devices that are connected directly by some transmission medium – twisted wires, coaxial cables, microwave, or satellite links. However, it is often impractical or uneconomical for two devices to be connected directly. Instead, communications between a source and destination nodes is achieved by transmitting data from source to destination through a network of intermediate nodes. These nodes provide switching facility that moves data from node to node until the destination is reached. Three different methods of establishing communication links between a sender and receiver in a communication network are circuit switching, message switching, and packet switching.

ii) Routing Techniques:

In store-and-forward, method, when multiple paths exist between a source and destination nodes, any one of the paths may be used to transfer a message/packet. Two popularly used routing techniques are:

1. **Source routing.** In this method, source node selects entire path before sending a message/packet. That is, all intermediate nodes, via which the message/packet will be transferred to its destination, are decided at source node, and this routing information is included along with the message/packet.
2. **Hip-by-hop routing.** In this method, each node along the path decides only the next node. That is, each node maintains information about status of all its outgoing channels and adjacent nodes, and the selects a suitable adjacent node for the current message/packet and transmits it to that node.

iii) Network Topologies:

Topology of a network refers to the way in which the network"s nodes (computers or other devices that need to communicate) are linked together. It determines the various data paths available between any pair of nodes in the network. Although number of possible network topologies is seemingly limitless, four major ones are star network, ring network, completely connected network, and multi-access bus network.

iv) Asynchronous Transmission:

In asynchronous transmission, data is transmitted character by character at irregular intervals. That is, sender sends a character at any convenient time and receiver accepts it. For instance, transmission of data from a terminal to a computer is asynchronous because during data entry by an operator, time between successive keystrokes varies.

Synchronous Transmission:

In synchronous transmission, a group of characters is blocked in same way as records are blocked on magnetic tape. A header and trailer are then added to each block to convert it into a frame. The header contains synchronization information used by the receiving device to set its clock in synchronism with sender end clock. It also contains information to identify the sender and receiver. The header is followed by a block of characters containing actual message to be transmitted.

THE END!